RAL-TR-2006-009

State of the art survey on trust, security and contract management in web services and Grid computing

Editors: E.Lupu, M.D. Wilson

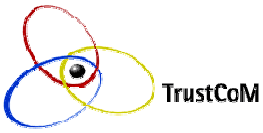
March 2006

Council for the Central Laboratory of the Research Councils

Library and Information Services
CCLRC Rutherford Appleton Laboratory
Chilton Didcot
Oxfordshire OX11 0QX
UK
Tel: +44 (0)1235 445384
Fax: +44 (0)1235 446403
Email: library@rl.ac.uk

CCLRC reports are available online at:
http://www.clrc.ac.uk/Activity/ACTIVITY=Publications;SECTION=225;

ISSN 1358-6254



Deliverable

28

Trustcom State of the Art Update

WP10 State of the Art

Michael Wilson
(Editor), CCLRC

December 2005

Version 2

TrustCoM

A trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations

SIXTH FRAMEWORK PROGRAMME

PRIORITY IST-2002-2.3.1.9

Prepared for the European Commission under FP6 Contract No. 01945 as a deliverable form

LEGAL NOTICE

The following organisations are members of the Trustcom Consortium:

Atos Origin,
Council of the Central Laboratory of the Research Councils,
BAE Systems,
British Telecommunications PLC,
Universitaet Stuttgart,
SAP AktienGesellschaft Systeme Anwendungen Produkte in der Datenverarbeitung,
Swedish Institute of Computer Scence AB,
Europaeisches Microsoft Innovations Center GMBH,
Eidgenoessische Technische Hoschschule Zuerich,
Imperial College of Science Technology and Medicine,
King's College London,
Universitetet I Oslo,
Stiftelsen for industriell og Teknisk Forskning ved Norges Tekniske Hoegskole,
Universita degli studi di Milano,
The University of Salford,
International Business Machines Belgium SA .

Deliverable datasheet

Project acronym: TrustCoM

Project full title: *A trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations*

Action Line: Action Line 7

Activity:

Work Package: WP10 State of the Art

Task:

Document title:

Version:

Document reference:

Official delivery date: 30 September 2005

Actual publication date:

File name:

Type of document: Report

Nature:

Authors: Alvaro Arenas (CCLRC),

Reviewers:

Approved by:

Version	Date	Sections Affected
V.1	30/08/05	All
V 2	23/12/05	All

Executive Summary

This deliverable is an update to the state of the art report that was produced at the start of the Trustcom project. The Trustcom project is half way through its three year research period, and has produced its initial conceptual models, architecture, designs and component implementations. At this point the state of the art in the technologies which contribute to Trustcom are being reviewed to ensure that decisions made so far in the project based on what was known 18 months ago are still sound. Issues reviewed include the innovative breakthroughs in technologies that could be used in the project, the adoption of technologies as standards that could be built on by the project, and the release of commercial products on the market which could lead the way to Trustcom exploitation, act as competition to Trustcom or show that topics being researched have already been developed to a commercial state so that alternative routes to procuring the technology other than development should be considered. Topics covered include VO management, contracts and service level agreements (SLA), collaborative business processes and enabling technologies. The conclusion of the review is that the main course of Trustcom has not been achieved elsewhere, and it is still consistent with the trend of development. However, both opportunities and threats to the Trustcom approach are identified.

Table of Content

1 Introduction

This deliverable is an update to the Deliverable 2 state of the art report that was produced in June 2004 after six months of the Trustcom project. An abridged summary of that deliverable was later produced as Deliverable 2A in December 2004. The Trustcom project is now half way through its three year research period, and has produced its initial conceptual models, architecture, designs and component implementations. At this point the state of the art in the technologies which contribute to Trustcom are being reviewed to ensure that decisions made so far in the project based on what was known 18 months ago are still sound.

The original state of the art review was a substantial document of 374 pages whereas this is a brief update focussed on technologies which the project has either adopted, or chosen not to incorporate for various reasons. This brief document should be viewed as an update on the existing document and not as a thorough state of the art review in its own right.

The topics addressed by this review cover those areas in which Trustcom is undertaking developments, as well as others where it has chosen not to, assuming that work elsewhere will reach an appropriate maturity for the technologies to converge.

The innovations in Trustcom are expected through the integration of reputation management, policy, role based security, and collaborative business process modelling technologies for the application of VO management by contracts and SLA. All of these areas were very thoroughly reviewed in the original state of the art 18 months ago. These are areas where the project includes researchers at the forefront of the field who were well aware of what was being done then. Not all of these areas are represented in this review since it was not judged that any significant work which had not been described in the previous review had taken place. This was because the ongoing basic research projects were reviewed, the integration projects such as Trustcom are still underway, and no standardisation or industrial application has occurred. Consequently, reputation, security and trust are not included in this review. There have been moves towards standardisation in the security area, not least in extensions to XACML, but these are reported elsewhere in the project standardisation deliverables, and often involve Trustcom project members. Trust has become a more discussed issue, but apart from publications supporting the Trustcom approach [45] of increasing trust by managing risk through trust substitute mechanisms there have been no major breakthroughs. Industrial adoption in these areas is addressed in the section on *Platforms and Tools*.

The areas which are included mainly focus on those areas where there has been industrial or standardisation activity. Developments in these areas 18 months ago may not have been made public by those undertaking them, and may only have surfaced subsequently, or in the case of VO management, both issues and research has come to the project's attention since the last review.

Each area is reviewed in terms of the impact of the external work on the course of the project.

2 Virtual Organisation Management

There has been a growing interest in the practical reality of Virtual Organisations (VO) during the last year since the original State of the Art review. There has also been developments in the corporate ERP sector to address the same issues through the procurement, or value chain as B2B relations rather than as VO. Each of these will be briefly reviewed.

2.1 VO Management Tools

The main existing, running VO management system is now VOMS[1] developed and implemented on the EGEE particle physics grid and European Data Grid for scientists to manage VO. The service is basically a simple account database, which serves the information in a special format (VOMS credential). The VO manager can administrate it remotely using command line tools or a web interface. It is very limited in its functionality since it does not address the trust, security and contract issues that are required for commercial developments.

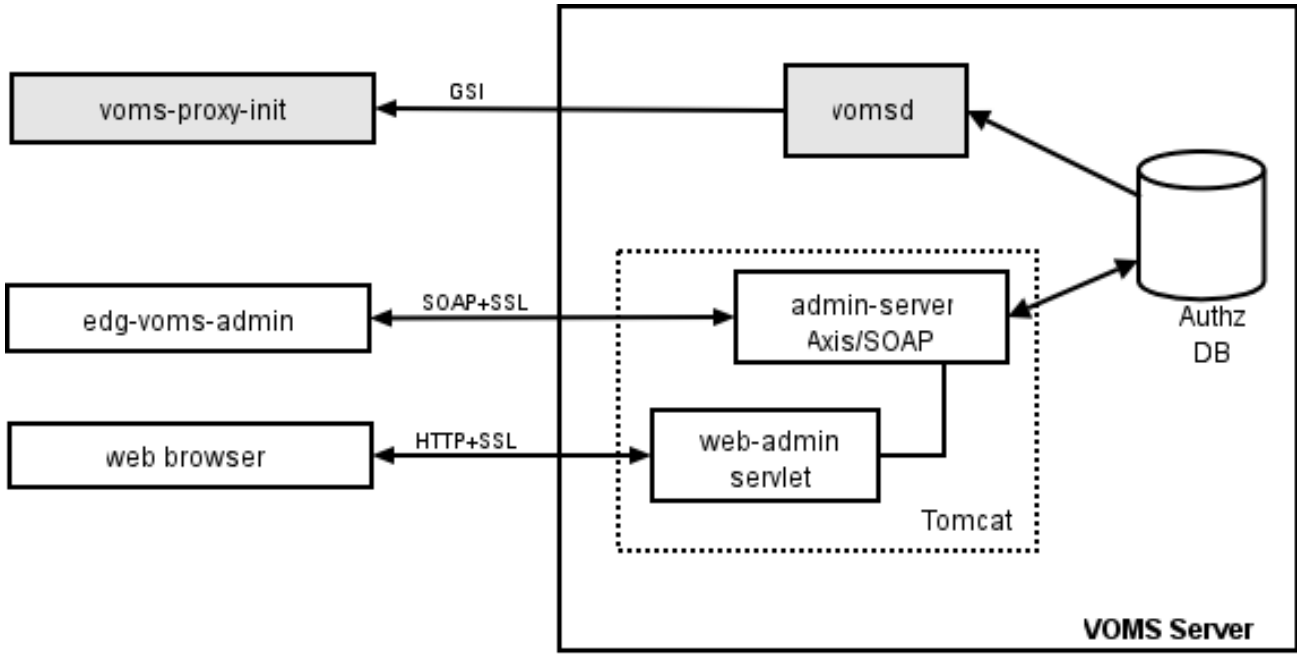


Figure 1: The VOMS architecture.

[1] http://edg-wp2.web.cern.ch/edg-wp2/security/voms/

The extended functionality is implemented as a web service, with command line and web interfaces:

- **Admin:** to provide the administrative functionality.
- **Compatibility:** to provide access to the user list for *gridmap-file* generation.
- **Request:** to handle user requests for administrative events and provide a simple framework for administrators to process them.
- **History:** to provide lookup functionality in a past timeframe, to answer questions like *"was this user member of my VO last summer?"*
- **Core:** to provide the basic functionality for users.

VOMS offers a model for Trustcom, and shows how VO management operates in an academic environment, but it does not include the functionality required for either secure science, nor for commercial use.

The scientific community that is currently using VOMS requires more secure VO management, and presents a possible exploitation opportunity for Trustcom, although there are no obvious lessons to be learned for design.

2.2 Social Networks and VO ecosystems

It has become apparent to many both inside and outside the project since Trustcom started that one of the biggest problems in the application of technology mediated VO's is that of attracting the right organisations to join the VO. A range of terms have entered the vocabulary to describe a simple agreement to potentially join virtual organisations, including online social network, Electronic Market Ecosystem**,** Digital Ecosystems, VO ecosystem, VE breeding ground, Enterprise Network or economic web.

Much of the research and the practical application of this idea has been focussed on networks of individual persons[2] who exhibit social behaviour in these environments and can use them as personal employment agencies, gathering reputations through the recommendation of previous employers who are also members of the same infrastructure.

Current IST projects such as ECOLead [39] are taking this idea and bridging the gap from individuals to organisations by providing sustainable collaborative networks that will act as breeding environments for the formation of dynamic virtual organizations. This aspect of the VO lifecycle was neglected in the first review, and the early phases of the Trustcom project.

Dafermos (2001)[3] argued that these can be catalysed by a common technology platform such as social software to build online social networks, or that which Trustcom offers:

An economic web is a dynamic network of companies whose businesses (are built around a single common platform to) deliver independent elements of an overall value proposition that strengthens

[2] http://www.firsttuesday.com/

[3] http://www.firstmonday.org/issues/issue6_11/dafermos/

as more companies join. Webs are not alliances. There is no formal relationship among the web's participants as the latter are independent to act in any way they choose to maximize their profits. These features drive them into weblike behaviour. Typical example is the Microsoft-Intel ("Wintel") web, composed of companies that produce Windows-Intel-based software applications and related services for PC users. Unlike alliance networks, in which companies are invited to join by the dominant company, economic webs are open to all and numbers equal power. The purpose of a network platform is to draw together participating companies by facilitating the exchange of knowledge among them.

The platform (a technical standard) of an economic web does not affect participating companies' relationship with the shaper (the company that owns the standard) and enables them to provide complementary products and services. Two conditions must be present for a web to form: a technological platform and increasing returns.

The technological standard reduces risk as companies need to make heavy investments in R&D in the face of technological turbulence, while the increasing returns create a dependency among participants by attracting in more producers and customers.

Trustcom has introduced an Enterprise Network into its VO formation process to serve this role, and is following the guidance of this line of research in presenting it so as to attract the maximum number of potential VO partners.

2.3 Automated VO partner selection

Minimising the risks in partner and supplier selection is key to automated VO and supply chain management. There has been recent research into technologies for this which could be adopted into the Trustcom VO component.

Petersen and Divitini [42] have proposed a simple utility model for selecting partners bidding to join a VO as:

$$\text{Utility Value} = \sum (\text{attribute value} * \text{weight})$$

Where attribute values in a bid to join a VO which do not meet the stated constraints are given a value of zero, and utility values are normalised before ranking, with the highest utility value at the top. Bittencourt and Rabelo [38] present an Analytic Hierarchy Process (AHP) for determining the weights to be used based on a candidate set of metrics derived from the Supply Chain Operations Reference Model (SCOR). Although this method has face validity, it has no rigourous justification beyond theuse of existing practice as a justification.

The VO initiator can choose to negotiate with the highest ranked potential partner for each role, or can introduce factors to select the best overall team. Team selection can be influenced by factors including the risks of resource failure by overloading a partner which trades off against the increased commitment of a partner with a more significant involvement, or the destabilising influence of a partner with too large a role in the VO, or the potential for a partner to pick up other roles if chosen partners fail and leave the VO. The set of factors affecting the team risks is large, but has been further investigated by several Chinese researchers.

Ip et al [41] report a genetic algorithm for partner selection based on risk evaluation by considering the capacity constraints on the provider to fulfil more than one role in a VO.

The optimal partner combination is calculated by minimising the risk of failure and lateness to deliver, although cost and quality of delivery are not considered. One clear result of their formulation is that the solution is nonlinear, and not convex, not continuous and differentiable, and that the model cannot be solved by general mathematical programming methods. Cao and Wang [40] have further refined the algorithm, while Zeng et al [44] further analyse the formulation of the problem and prove that it is NP complete, but that both the objective function and the constraint function have monotonicity properties so that a Branch and Bound algorithm can be effective.

However, these algorithms have not yet been proven to any significant extent on data, and only consider limited sets of the variables available within the Trustcom VO management system.

2.4 ERP Tools and supply chain management

In an October 7, 2005 AMR Research Alert, “Supply Chain Risk Management Strategies, Part I”, analysts Mark Hillman and Lora Cecere commented, “In third-generation sourcing and procurement applications, buy-side contract management plays an increasingly important role. This connector between sourcing and procurement links legal, procurement, and financial departments together to view and manage risks.”

The SAP ERP tools include a major utility addressing Supplier Relationship Management (SRM) whose function is to manage the supply of goods and services required for the business. Previously, and SRM utility would procure goods and services as required by the plan for the main enterprise, ensuring that they were available when required. However, the toll support was limited to procurement from established catalogues provided by suppliers with whom an business relationship already exists. Such SRM systems manage cost savings in many ways, but they are also marketed as developing relationships between organisations as a basis for future collaborations.

The current release of SAP ERP R/3 is 4.6, while plans for release 6 in 2007 include contract management to extend this functionality to include supplier qualification (akin to reputation management), contract construction (through editing clauses in templates of existing contracts), and contract monitoring (which does not address quality issues but should monitor time and cost, as well as feeding back to the supplier qualification system).

Also, in Oct 2005 Frictionless Commerce, who operate with SAP, has produced an SRM contract-management module which enables users to gain control over the contract generation and management process—providing a repository, automated management, contract and clause templates, and visibility into spend.

Frictionless Commerce’s Contract Management provides a single contract repository, automated management, and a contract and clause template library. Going beyond basic workflow approvals, the Contract Management module enables configurable workflow phases, approval “gates” for policy enforcement and approval hierarchies.

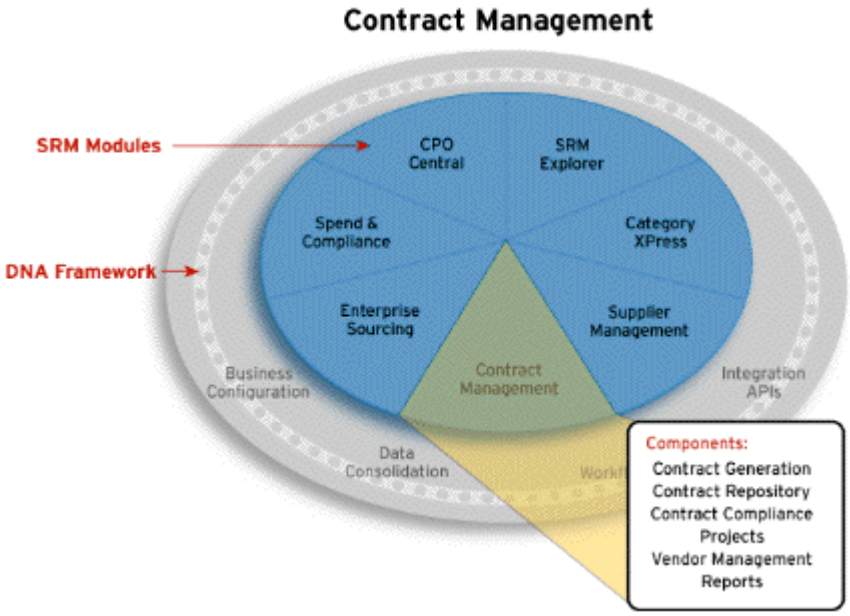


Figure 2: Frictionless Commerce's Contract Management Module

Contract Management enables you to:

- Realize negotiated cost savings through automated contract management and compliance reports.
- Develop fully configurable agreement types supporting all commodity categories
- Easily configure search queries to identify agreements by contract manager, commodity, expiration date, supplier, etc.
- Reduce sourcing cycle times by creating contracts in parallel with negotiation events. Frictionless automatically carries over key contract data from an RFx or Auction into the new contract record
- Track all enterprise spend against contract with the highly flexible spend transaction import capability supporting any downstream procurement/ERP system transaction data

Through a series of system-wide reports, you can analyze spend against the contract at both the summary and item level to achieve compliance. In addition, it matches contracts with spend transactions captured in e-Procurement and ERP systems to analyze off-contract spending.

As with the SAP application, this implementation does not include security, or SLA quality monitoring or enforcement, but it is coming closer to the Trustcom view than traditional workflow or ERP solutions.

Oracle's PeopleSoft Enterprise Supplier Contract Management in its SRM V8.9 released in the summer of 2005 claims to be the application that creates and enforces better supplier contracts. PeopleSoft Enterprise Supplier Contract Management manages the entire supplier contract lifecycle, from authoring, collaboration, and negotiation to execution, status tracking, and compliance. Effectively managing the entire supplier contract lifecycle is critical in reducing spend on goods and services by enforcing contracted pricing with embedded terms and conditions.

The figure below illustrates the clause editor of the PeopleSoft Contract Manager, showing how contracts can be build from a library of clauses held centrally. It is notable that one of the marketing advantages put forward for this technology is that a central contract repository provides full visibility and thereby reduces risk – a step towards increasing trust.

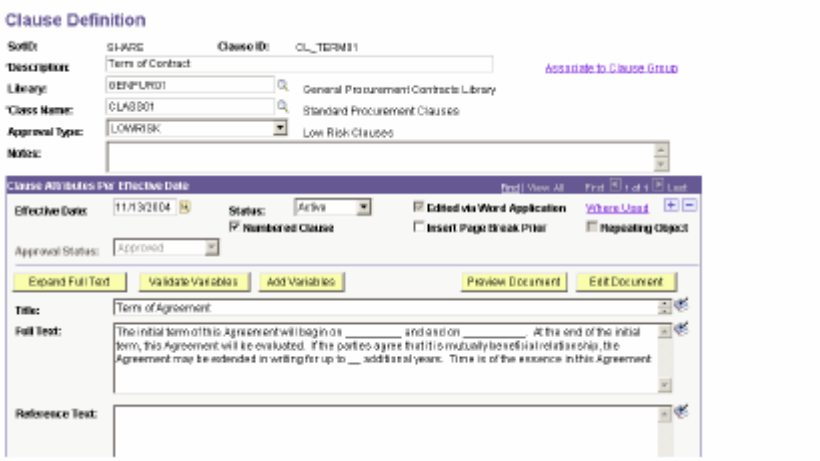


Figure 3: Oracle's SRM contract editor interface.

At present the details of these advances are not easily available, and may not be confirmed, but they show that:

- the issues that Trustcom is addressing are also being addressed by ERP suppliers;
- ERP suppliers address these issues as SRM solutions rather than through VO;
- if ERP suppliers establish the value of the technology for their large customers, smaller customers may want less monolithic solutions providing this functionality;
- the technology that Trustcom is developing could be included in ERP products to provide the planned functionality;
- The Trustcom contract and SLA monitoring and enforcement technology is still ahead of these commercial products.

3 Contracts and Service Level Agreements

3.1 Standards for SLA Languages

Since the review of the state-of-the-art performed by the TrustCoM consortium in June 2004, very little movement has been noticed regarding languages for the specification of Service Level Agreements. WS-Agreement continues to be the main channel for standardization regarding this topic.

3.1.1 WSLA

Version 1.0 of the WSLA specification is of April 2003. Since then, it has become apparent that its developing team has shifted attention to GGF's WS-Agreement specification. No further development of WSLA is expected.

3.1.2 WS-Agreement

The specification of WS-Agreement has been constantly updated since first examined by the TrustCoM project. A final version is expected by March 2006. The specification is currently (November 2005) going through a Public Comment Period.

Work on WS-Agreement has followed the evolution of other WS-* standards. In particular, it relies on WS-Addressing, WS-ResourceProperties, WS-Resource-Lifetime and WS-BaseFaults.

WS-Agreement itself constitutes a framework for concluding agreements on several different domains. The details of what is to be agreed are domain-specific and fall out of the scope of the specification of WS-Agreement. However, most attention has been given to agreements for job-submission only. This was considered an important obstacle for integration and for that matter TrustCoM has so far preferred using WSLA as its SLA specification language. Since the development of WSLA was discontinued long ago, TrustCoM should reconsider using WS-Agreement, even though this will require developing its own domain-specific sublanguages (in lack of appropriate standards).

During the GGF-15 meeting, in October 2005, the GRAAP-WG (responsible for WS-Agreement) started work on requirements for negotiation protocols.

Reference Implementations:

Cremona (Creation, Monitoring, and Management of WS-Agreement) is a WS-Agreement middleware function complementing the basic Web service stack. It helps providers manage agreement templates, implement the agreement protocol, check availability of service capacity, and expose agreement states at run time. The latest release of Cremona is part of IBM's ETTK v2.3 (May 2005) and includes new agreement template functions, improved interoperability, and tighter conformance to the WS-Agreement specification.

Other implementations, by NEC, Fraunhofer Gesellschaft, Forschungszentrum Jülich GmbH and EGEE, are not openly available.

3.2 Negotiation

Moving into the second phase of the project, TrustCoM will address the issue of contract negotiation. This section is meant as an initial exploration of the area of contract negotiation for web and electronic services.

3.2.1 WS-AgreementNegotiation

WS-AgreementNegotiation[4] describes the re/negotiation of agreements between two parties. The specification is only at the level of an initial draft.

One of the requirements for WS-AgreementNegotiation is to use WS-Agreement to express negotiation offer. Obviously, this ties it strongly to WS-Agreement. WS-AgreementNegotiation complements WS-Agreement in that it would make it possible to switch to different negotiation protocols.

The specification provides a web-service interface that allows renegotiation of existing agreements and can be initiated by both provider and requestor.

Finally, notice that negotiations are assumed to be only bilateral. The case of multiparty negotiation is not addressed.

3.2.2 WS-Negotiation

WS-Negotiation[5] is an independent declarative XML based language that can be used with different types of agreement templates.

The general form of a negotiation session in WS-Negotiation is shown in Figure 4.

[4] https://forge.gridforum.org/projects/graap-wg/document/WSAgreementNegotiationSpecificationDraft.doc

[5] PCK Hung, H Li, JJ Jeng. WS-Negotiation: An overview of research issues. In Proceedings of the 37th Hawaii International Conference on System Sciences - 2004.

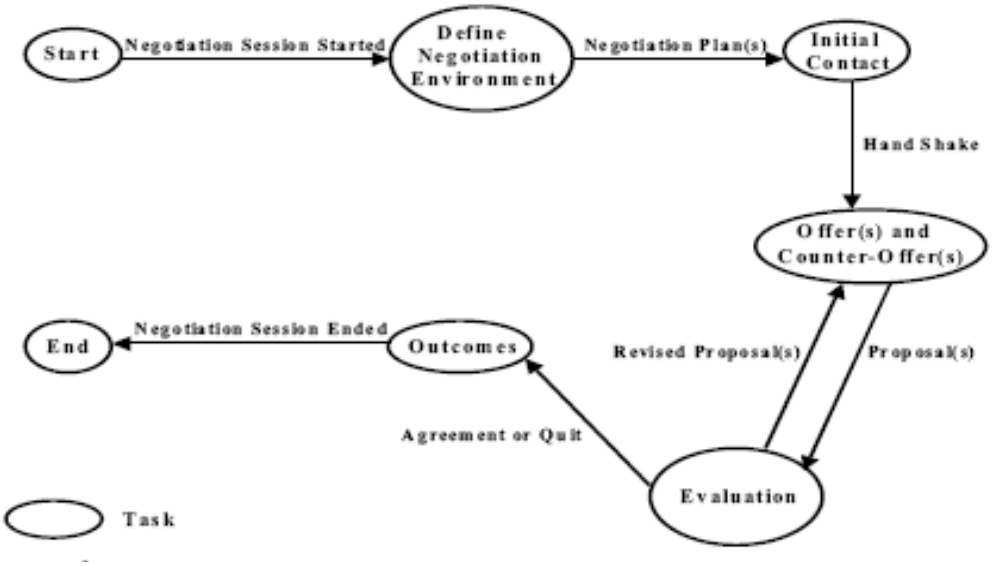


Figure 4: A Negotiation Session in WS-Negotiation

WS-Negotiation consists of three parts (Figure 5):

1. Negotiation Message: This part describes the format of the messages exchanged. Some suggested message types are: Offer, Counter-Offer, Rejected, Accepted, etc
2. Negotiation Protocol: describes the mechanism and the rules the negotiation parties should follow. Messages can be exchanged between requestor and provider as well as a third-party negotiation service (Negotiation Support System-NSS). Negotiation primitives are also defined. A negotiation primitive sets the pre and post conditions that should hold as well as rules and constraints that should be applied during the negotiation. They may have corresponding negotiation messages and they can be implemented as Web methods in both sides. Example of negotiation primitives are the “Propose” primitive for proposing an offer/ counter-offer to the other party, the “Modify” primitive to modify the sent offer/counter-offer before receiving the other party’s reply etc.
3. Negotiation Decision-making: This is the “internal and private decision process” that is based on the negotiation strategy each party has chosen (e.g. cost-benefit strategy) and the agreement template.

Figure 5: WS-Negotiation Structure

In short, the negotiation protocol orchestrates the negotiation messages which trigger and conduct the negotiation decision-making process (Figure 6).

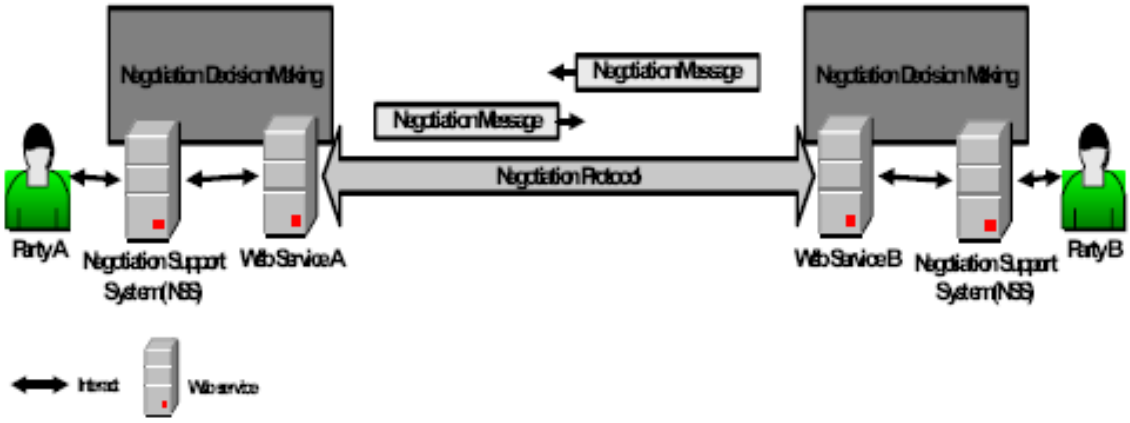


Figure 6: WS-Negotiation Framework

- Negotiation issues vary form one business domain to another but there are some issues that are common or fixed in a domain. So, domain specific vocabularies are introduced for different types of business negotiations.
- A template is composed by all the common and fixed issues and can be seen as an initial layout of an agreement.
- A Service Level Agreement is “a formal contract between a Web service requestor and a provider guaranteeing quantifiable issues at defined levels only through mutual concessions”. The negotiation issues are described as SLA parameters which are based on the domain specific vocabularies.
- A SLA template can be seen as a SLA document with blanks to fill in with information concerning the parties involved, obligations and guarantees.
- The SLA template model proposed suggests that SLA documents are generated by domain specific vocabularies which derive a negotiation process. WSLA can be used with this model.
- One suggestion on how to implement WS-Negotiation is to embed the negotiation messages into the SOAP messages.

So far, only bilateral scenarios (one-to-one negotiation) are considered. The multi-party case (one- to- many) is currently being investigated.

3.3 Conclusions

The slow evolution of contract and SLA standards for web services during the last 18 months reflects on the need to first consolidate the lower levels of the web-services standards stack. Nonetheless TrustCoM should consider moving soon from (the almost obsolete) WSLA towards WS-Agreement. Regarding SLA and contract negotiation, WS-AgreementNegotiation and WS-Negotiation are just two of many proposals, all at very initial stages of development.

4 Collaborative Business Processes

4.1 Introduction

In version 1 of this document the need for choreographies and orchestration specifications was established. Additionally it investigated the state of the art and the following standards for collaborative business processes and web service composition have been described:

- Web Services Transaction (WS-Transaction)
- Web Services Coordination (WS-Coordination)
- Web Service Choreography Interface (WSCI)
- Business Process Management Language (BPML)
- Business Process Execution Language for Web Services (BPEL4WS)

Recently, new developments and standards have emerged and some existing ones are less well supported. We are following these trends and have identified the following standards or standardization efforts that have or might have potential impact on the TrustCoM project. Particularly, the following will be described in detail:

- Web Services Choreography Description Language (WS-CDL)
- Business Process Modelling Notation (BPMN)

4.2 WS-CDL[6]

WS-CDL is an XML-based language that describes peer-to-peer collaborations of parties by defining, from a global viewpoint, their common and complementary observable behaviour. It is a choreography specification language opposed to orchestration languages, such as BPEL4WS. Its intended use to model complex business protocols, such as order management, enabling interoperability between any types of application components, regardless of the supporting platform or programming model used. In that sense complement orchestration languages that are concerned with executing the local view of the business process.

WS-CDL is based on a variant of the Pi calculus and uses linear typing to support safety and liveness properties.

WS-CDL specifies collaboration in terms of roles and work units. Work units consist of activities and ordering structures and especially important are interaction activities that result in exchange of information along a channel. It can reference its endpoints of the collaboration, the used services, using the WSDL specification. A

[6] http://www.w3.org/TR/ws-cdl-10/

role enumerates the observable behaviour a party exhibits in order to collaborate with other parties.

WS-CDL specification currently exists as a working draft of the W3 consortium. It looks set to become the adopted standard for defining choreographies of web services.

4.3 BPMN[7]

BPMN is a graphical notation that depicts the steps in a business process. BPMN depicts the end to end flow of a business process. The notation has been specifically designed to coordinate the sequence of processes and the messages that flow between different process participants in a related set of activities.

Since BPMN is a graphical notation to describe collaborations, it competes more with other graphical notations, such as UML, rather than orchestration languages, such as BPEL. It distinguishes itself by being process-oriented allowing simpler transformation in executable languages, such as BPEL.

A diagram in BPMN is made up of graphical elements and uses similar techniques to flowchart diagrams. Because of this familiarity, it is easy for e.g. a business analyst to pick up the notation. There are flow objects (events and activities), connecting objects (arrows), swim lanes and artefacts (data and documents).

BPMN has been released as a standard by the Business Process Management Initiative (BPMI).

4.4 Conclusions

The language of choice for the choreography has changed since the last version of the document. It is now WS-CDL instead of the combination of WSCI / BPML. The support for the latter started vanishing. So the switch has been made to leverage and participate in current and industry-wide efforts and be ready for future development. The choice of BPEL as the orchestration and execution language has been kept, since it starts to find wide acceptance, but, of course, no one can predict which standards will emerge and prosper in such a dynamic and competitive environment. These provide building blocks for the design of the Collaborative Business Process workpackage and several extensions and features have been developed based on them that are discussed in other deliverables.

The other choice for the graphical presentation of the business process was UML. It was chosen over BPMN, since it seemed to be more mature and provided better tool support at the start of the development phase.

[7] http://www.bpmn.org/

5 Distributed System Policy Management

Research in Policy Based systems has been continuing since the last version of the State of the Art. As usual the Policy Workshop [29] has been one of the main forums where studies on this topic have been presented. In 2005 the workshop has been hosted by one of the members of the TrustCoM project and received substantial contributions from others. It would be difficult to summarise in a short section progress across a large array of research studies. Instead, we briefly describe below two significant advances in this area: the release of the *policy middleware for autonomic computing (PMAC)* framework from IBM and the progress of on-going work on policy analysis and refinement. The first is important because it represents a strong initiative to support policy driven mechanisms at the core of a general purpose adaptive computational paradigm by one of the largest commercial organisations in this space. The second is important because it is one of the most challenging research problems to be addressed and which is not explicitly addressed within the project.

5.1 Policy Middleware for Autonomic Computing (PMAC)

The Policy Middleware for Autonomic Computing (PMAC) framework has been developed as part of IBM's autonomic computing initiative [30]. The framework consists of a policy editing tool, a federator, an autonomic manager and a managed resource (see Figure 7). Policies specified using the editor are published to the federator which acts as a publish/subscribe hub for distributing policies to the relevant autonomic managers. The principal component in this framework is the autonomic manager, which is responsible for providing a local policy-based control loop to manage the resource assigned to it. Managed resources provide two interfaces, referred to as *sensors* and *effectors*, which respectively represent the attributes that can be read from the resource and the management operations that can be performed to change the state of the resource. Since an autonomic manager can be treated as a managed component, it also provides an interface to its sensors and effectors for use by other autonomic managers.

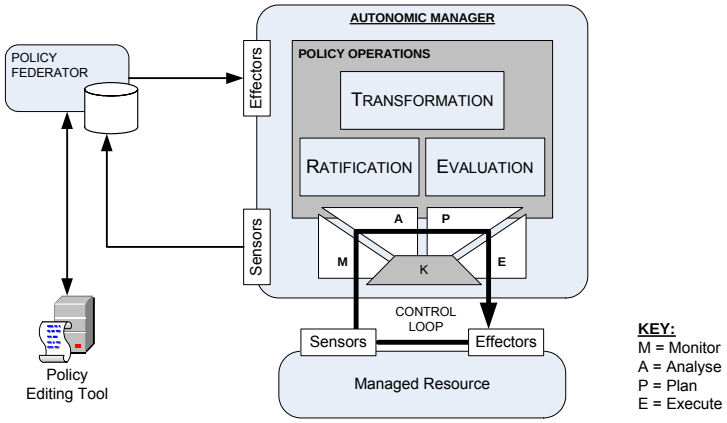


Figure 7 Policy Management for Autonomic Computing

Policies in the PMAC framework are specified as Event-Condition-Action (ECA) rules and their structure is defined using an XML schema. Additionally, the framework includes a general constraint specification language, also specified using XML, which can be used in the ECA policy rules to define the condition clause. The use of XML schema based definitions for the policy and constraint language allows the PMAC framework to extend the set of types supported by the language by simply including off-the-shelf XML definitions which come with pre-defined syntax and semantics. The problem with using an XML representation for policy is that the specifications tend to be quite verbose and not easily interpreted by the user. This problem is addressed in the PMAC framework by providing a *Simple Policy Language (SPL)* which supports the specification of ECA rules. At present, there is no documentation regarding the syntax of SPL, but it is expected to be similar to the obligation policy syntax of the Ponder framework.

PMAC provides support for policy analysis and conflict resolution using a process called *Policy Ratification* [31]. Essentially this process provides the administrator with information about the impact of a new policy on the existing set of policies. The ratification process consists for four domain-independent, generic operations: dominance checking, conflict checking, coverage checking and consistent priority assignment.

Dominance checking determines if there is a policy in the system that is subsumed (or dominated) by another. For example a policy that specifies that “password length > 5” is dominated by one that states “password length > 8” since the latter policy makes the former redundant. Generally it can be stated that a policy P1 is dominated by policy P2, if the constraint of policy P1 logically implies the constraint of P2.

One of the main challenges in policy analysis is in dealing with the constraints that control the applicability of a policy and identifying the situations in which two policies have constraints which are satisfied at the same time. The *Conflict checking* operation in PMAC is used to identify policies whose constraint statements are simultaneously true, and therefore may cause a problem if the actions defined in those policies are incompatible with each other. For example a policy that defines that “Joe has access to the database between 9am and 5pm” would be in conflict with a policy that states “Joe should not be granted access to the database on weekends” since, if both these policies are enforced, Joe is both granted and refused permission between 9am and 5pm at weekends. PMAC addresses the conflict detection requirement by recognising that the key operation in detecting a potential conflict is to determine whether the conjunction of two Boolean expressions can be satisfied. To this end, the framework implements a number of analytical algorithms for checking the satisfiability of different classes of Boolean expressions [31]. Whilst this approach is effective at detecting potential conflicts in a policy specification, since the analysis does not use a model of the system behaviour, it cannot take into account changes in system state caused by enforcing policies.

Coverage checking allows an administrator to ensure that for a given range of input parameters there is at least one policy that is applicable. This is achieved by checking whether a disjunction of Boolean expressions (e.g. the constraints of the policies) implies another (e.g. the input parameter range).

Finally, the PMAC framework assumes that problems identified through the above ratification operations will be resolved by the administrator marking policies as inactive or assigning some relative priority to them. In this latter case there is the possibility that introducing a new policy requires that existing priority values are reassigned so that the

overall ordering of the policy rules remains consistent. The fourth ratification operation, *consistent priority assignments* automates this priority assignment process using a modified version of an algorithm used to insert items into an ordered list.

Policy refinement is supported in PMAC through the definition of transformation rules. These take the form "if certain conditions on a policy are satisfied (e.g. if the Role is WindowsSecurity) then update a certain portion of the policy with a new entity (e.g. replace the Condition and the Action by a new Boolean expression)." Since these transformation rules are implemented as policy rules that operate on the policies themselves, they can be executed using the same enforcement system as used by any other PMAC policy. One drawback of this approach is that the transformation rules have to be defined manually by an expert in the application domain and there is no means of verifying that the transformed policy satisfies the original goal of the user.

To summarise, the PMAC framework provides a language for specifying policy and a policy deployment and enforcement architecture. The system has been designed to be extensible through the use of XML schemas and is capable of policy analysis and refinement operations. However, the analysis process does not take into account the behaviour of the managed system and therefore cannot detect inconsistencies that are caused because the enforcement of one policy causes a state change such that some other policies then conflict. Finally, the policy refinement process is defined in terms of explicit transformation rules which must be specified by a domain expert. Therefore it is not possible to verify the correctness of the policies generated by applying the transformation rule with respect to the original policy.

5.2 Policy Analysis and Refinement

Despite significant efforts in developing different policy specification techniques, there remain a number of issues to be addressed. In particular, since one of the objectives of using policy-based systems is to fulfil organisational goals, the ability to refine such goals into concrete policy specifications would be useful. As we have discussed in this paper, it is desirable to maintain the properties of correctness, consistency and minimality when performing any refinement transformation. Of course, this is only possible if the chosen policy specification technique provides support for checking whether these properties hold. To this end, we have developed a mapping of policy specifications into a more formal, logic representation which is based in on the Event Calculus [32]. In order to exploit the properties of decidability and lower computational complexity, it is intended that the formal representation would be based on first-order stratified logic. This notation is then used in conjunction with the goal elaboration approach, developed by [33], to develop a usable policy refinement technique.

As part of solving the policy refinement problem, it will be necessary to address some of the outstanding issues related to policy analysis and conflict detection. Unless we have a means of checking for conflicts in a policy specification, it will be impossible to maintain the required consistency property in a given refinement. In Ponder modality conflicts arise between positive and negative policies that apply to the same subjects, targets and actions [34]. These can be detected by syntactic analysis of the policies as the conflict can be determined by detecting overlap of subjects, targets and actions. However, the analysis detects only potential conflicts rather than actual conflicts since constraints may limit the

applicability of the policy to disjoint sets of circumstances e.g., different times of day. While modality conflicts can be detected syntactically, other conflicts can only be determined by understanding the actions being performed by the policies. For example, there will be a conflict between two policies that result in the same packet being placed on 2 different queues. Similarly, separation of duty conflicts arise from authorisation policies, which permit the same person to approve payments and sign cheques. Generally, these conflicts are application specific and to detect them it is necessary to specify the conditions that result in conflict. The approach is therefore to specify constraints on the set of policies (i.e. meta-policies) using a suitable notation and then analyse the policy set against these constraints to determine if there are any conflicts [34]. Whether conflicts occur or not may depend on run-time parameters specified in constraints such as time or the current state of the components to which the policy apply. It is thus rather difficult to determine all possible conflicting conditions in advance and so it is still necessary to detect conflicts at run-time. Furthermore, when conflicting policies are detected it is not obvious how to resolve the conflicts automatically. Explicit priority may work in some cases. In some situations, negative authorisation policies should override positive ones, but in other situations the positive authorisation is an exception to a more general negative authorisation. In some situations more specific policies that apply to a department may override general policies applying to the whole organisation. We have been experimenting with meta-policies that define application specific precedence relationships between conflicting policies.

Although some progress has been made in dealing with policy conflicts [34, 35], significant challenges remain to be addressed. In particular, how can one detect conflicts when arbitrary conditions restrict the applicability of the policies? Sometimes, it is possible to compare restrictions placed by the constraints. For example, it is possible to detect if two time intervals overlap or if the policies apply when subjects are in different states e.g., active or standby. Other challenges concern the different levels of abstraction at which policy is specified. Conflicts between organisational goals will inevitably lead to conflicts between the policies derived from these goals. Some policies will trigger complex management procedures, which require the execution of actions that may be specified as part of different policies. This renders the task of ensuring the consistency of a policy specification much more complex.

Recent efforts on formalising the specification of policies and the behaviour of managed systems have gone some way to addressing the challenges of policy analysis and refinement [36]. This work proposes a formalism that is based on the standard Event Calculus [32] that models both authorisation and management policy specifications together with the behaviour of the managed system. Event Calculus was chosen as both the policies and the management behaviour we are modelling are event driven. Additionally, since an Event Calculus specification of a system can be generated from a state transition model, users can specify the management behaviour using a familiar high-level notation. In a similar fashion, it is possible to translate policies specified in a high-level policy specification language, like Ponder [37], into an Event Calculus representation that describes the semantics of the policy language. This eliminates the need for the user to become conversant with the details of logic programming and the Event Calculus notation.

In order to analyse the policy specification, the Event Calculus representation supports the specification of rules for detecting a range of consistency properties. This includes modality and application specific conflicts such as conflicts of duty together with policy validation. Additionally, the formal representation supports a number of types of review

query, allowing the administrator to obtain different views of the information in the policy specification.

This analysis technique uses a combination of deductive and abductive reasoning. Deductive techniques are primarily used for policy validation and to perform review queries whereas abduction is used to detect conflicts between policies. By using abductive reasoning techniques, it is possible to analyse the policy specifications to identify existing conflicts and provide explanations on how they might arise. Because the abduction process is applied to a specification that models both the systems behaviour and the policy specification it is possible to detect conflicts when the applicability of the policies is constrained on the runtime state of the system. Furthermore, by using abduction, the analysis can be performed even with partial specifications of the system state.

This formalism can be combined with the KAOS goal elaboration technique [33], to provide a framework for policy refinement. However, the low-level goals derived using this technique cannot be directly used in policies without first identifying the management operations that will achieve them. This identification process is supported by the introducing the concept of a *strategy*. A strategy is the mechanism, by which a given system can achieve a particular goal, i.e., a strategy is the relationship between the system description and the goal. By having a formal specification of the latter two types of information abductive reasoning is used to infer the strategy. The KAOS goal elaboration technique is used because it provides the concept of domain-specific and domain-independent refinement patterns, logically proven goal refinement templates that can be easily reused. Such patterns capture the refinement of goals that are commonly encountered in policy-based management, thus simplifying the refinement process for the user. Additionally the refined policies derived using this process can be encoded in policy refinement patterns that can be later reused when the administrator wishes to satisfy a similar goal.

6 Enabling Technologies

6.1 Introduction

The innovations in Trustcom are expected through the integration of Reputation management, Policy, role based security, and collaborative business process modelling technologies for the application of VO management by contracts and SLA. However, the project has committed to develop its reference implementation and its demonstrators using publicly defined open specifications of technologies as implemented in tools and platforms for web services and the grid. Each of these two topics will be reviewed for updates to the state of the art in the eighteen months since the last review.

Trustcom has also committed not to develop advanced transparent semantic web applications although the project believes that they will be required in the future for applications such as VO management. However, since there is a large community undertaking work in this area, it would be a waste of resources to duplicate their work, and the project team will wait to adopt their results when they are stable. Consequently, this area of enabling technologies will also be assessed to determine if this decision was appropriate, and whether the output of the community's work will converge with that of Trustcom as expected.

6.2 Web Services and the Grid

From the start TrustCoM has made a commitment to web service standards (WS-*) and associated technologies. This constituted at the same time an opportunity and a risk for the project. It was an opportunity because web services seemed to be the technology of choice for interoperability and collaboration across heterogeneous implementation domains. It was a risk because beyond the basic SOAP messaging mechanisms many of the more advanced specifications were still in a draft phase, with some specifications overlapping or even incompatible due to no widespread agreement.

Grid systems are of major relevance to TrustCoM as they exhibit a degree of similarity with Virtual Organisations. They share particular aspects such as service discovery and composition, and SLAs. However, Grid systems focus on sharing of resources and distribution of computational tasks, and typically do not address some of the more complex problems such as the composition of VO structures, the use of business processes, or the support for different trust relationships between the participants in a VO. From an architectural and implementation perspective, Grid systems were largely monolithic, with numerous dependencies, making it difficult to reuse specific tools or components.

In the reporting period the standardisation process of the fundamental specifications for the Web Services and Grid domain has continued to get more mature and stable. In particular the integration from specifications well beyond the

very basic WS-I profile has started. Most of the specifications have been updated and have become much more mature (and there are also more of them). A number of the advanced WS-* specifications, initially driven by specific companies, have now also gone into standardisation initiatives. Part of this process is also the increased availability of interoperable implementations also covering the area of so called Grid toolkits delivering at least implementations of the WS-RF family of specifications and of WS-Notification.

In particular the gap between the Grid and Web Services domain has continued to narrow mainly driven by the decision within the community to discontinue the Open Grid Service Infrastructure (OGSI) based specifications. Furthermore the finalisation of the Open Grid Service Architecture (OGSA) in an initial version together with the approach of defining OGSA Profiles limiting the number of specifications will lead to further increased interoperability of these toolkits in the near future. One OGSA Profile is based on WSRF. Generally, web services are the common baseline for Grid infrastructures and while Grid systems were often monolithic in the past, the principles of composability and profiles have been adopted in the Grid world. This means for example that TrustCoM-specific work (such as in security and SLA) should be compatible with WSRF based Grids as well as any other web services based systems.

Even if the process of reaching convergence is evolving there are still a wide range of alternative or competing specifications that can be used for similar purposes. Most notably this is in the area of providing state information (WS-Transfer <-> WS-Resource), the delivery of events (WS-Notification <-> WS-Eventing) and the related areas of management of Web Services based applications. This issue has been addressed by restricting the usage of these specification e.g. of WS-Resource in order to allow a move to WS-Transfer or a potential combination of both in the future.

In general TrustCoM has to date made the correct decisions with respect to the selection of WS-* specifications to be used for the framework. It has carefully restricted the usage of specifications that are not at this time mature in order to make it clear which one will be taken up by the community. This approach will continue to be used for the remainder of the Project.

The most relevant updates since the initial state-of-the-art analysis are included in the table below:

Specification	Standardisation Group	Maturity	Implementations	Role in TrustCoM
WS-BaseNotification	WS-N (OASIS)	Public Review	Several (including WSRF.NET, Axis-Pubscribe, GT4)	Restricted to basic features in order to limit dependency
WS-BrokeredNotification	WS-N (OASIS)	Public Review	Limited support	Not yet used

Specification	Standardisation Group	Maturity	Implementations	Role in TrustCoM
WS-Topics	WS-N (OASIS)	Converging	Limited support	Restricted to basic features in order to limit dependency
WS-Eventing	None	Available	Experimental	Not yet used.
WS-Transfer	None	Available	Experimental	Not yet used.
WS-Enumeration	None	Available	None.	Not yet used.
WS-Addressing	W3C	Standard	Several (including Axis, WSE, GT4)	Essential
WS-ResourceProperties (WSRF-RP)	WSRF (OASIS)	Public Review	Several (including WSRF.NET, GT4, pyLite)	Used with basic features in order to limit dependency
WS-ResourceLifetime (WSRF-RL)	WSRF (OASIS)	Public Review	Several (including WSRF.NET, GT4, pyLite)	Not yet used.
WS-BaseFault (WSRF-BF)	WSRF (OASIS)	Public Review	Limited.	Not yet used.
WS-ServiceGroup (WSRF-SG)	WSRF (OASIS)	Public Review	Limited.	Not yet used.
WS-Naming	GGF	Discussion	Initial not interoperable implementations	Not yet used.
WS-ByteIO	GGF	Converging	No reference implementation so far	Not yet used.
WS-ReliableMessaging and WS-RM Policy	OASIS[8]	Converging	Several	Not yet used.

[8] WS-ReliableMessaging and WS-RM Policy were submitted to a newly formed OASIS Web Services Reliable Exchange (WS-RX) TC, including many companies driving the now retired WS-Reliability specification.

Specification	Standardisation Group	Maturity	Implementations	Role in TrustCoM
MTOM	W3C	Converging	WSE	Not yet used.
WS-DistributedManagement	OASIS	Converging	Apache MUSE	Used with basic features
WS-Management	DMTF[9]	Available	No reference implementation so far	Not yet used.
WS-Coordination, WS-AtomicTransaction, and WS-BusinessActivity	OASIS[10]	Available	Limited.	Used with basic features
SOAP Message Security + various security token profiles	OASIS and WS-I	Standard (version 1.1)[11]	Wide range	Essential
WS-SecureConversation, WS-Trust, WS-SecurityPolicy	OASIS[12]	Available	WSE (amongst others)	Essential

It is recommended to continue with the chosen approach, ie, to carefully consider which functionalities are essential for the framework and limit the usage of certain specifications in order to reduce the dependency of the TrustCoM on immature specifications. In particular, this affects the usage of the WSRF family of specifications. Despite the increased support for these specifications by toolkits and the increased interoperability at the toolkit level, it is still possible that concepts from WS-Transfer, WS-Enumeration will influence the final versions of WSRF. The same observation applies to the concept of asynchronous message delivery where the more complex usage models possible with the WS-N set of specifications

[9] WS-Management was submitted to DMTF.

[10] WS-Coordination, WS-AtomicTransaction, and WS-BusinessActivity were submitted to a newly formed OASIS Web Services Transactions (WS-TX) TC. It should also be noted that the OASIS Web Services Composite Application Framework (WS-CAF) is working on an overlapping Web Services Coordination Framework specification.

[11] OASIS SOAP Message Security ("WS-Security") along with various security token profiles are at version 1.1. The WS-I is completing interoperability profiles for these specifications. The promotion of additional WSS token profiles by individual companies (e.g., RSA's OTP-WSS-Token) illustrates the wide acceptance of SOAP Message Security.

[12] WS-SecureConversation, WS-Trust, and WS-SecurityPolicy were submitted to a newly formed OASIS Web Services Secure Exchange (WS-SX) TC.

should be avoided in order to allow a potential move toward WS-Eventing if this is required. The highest risk identified so far is the lack of convergence on how to address the problem involving the transfer of large data sets in binary format. The specifications within the GGF around ByteIO are substantially different from considerations within W3C around MTOM. The focus of monitoring and experimenting should therefore be focused on this particular area where a lack of convergence between the Grid and Web Service community continues to be visible. An overview of the implementation of the specifications discussed here in the available toolkits can be found in http://www.cs.virginia.edu/~humphrey/papers/WSRFComparison2005.pdf.

6.3 Tools and Platforms

6.3.1 Web and Grid services middleware

The majority of the TrustCoM developments so far have been on Microsoft .NET/WSE and Apache Axis. These were the platforms selected as a basis for development based on specification support as well as availability/maturity.

Overall, there appear to be no reasons to reconsider this selection.

1.6.3.1 Globus Toolkit 4 [13]

The Globus toolkit is an implementation of the WS-RF and WS-N specifications integrated with other relevant specification such as WS-Addressing, WS-Naming, WS-Security, … However the toolkit is not limited to this and provide implementation of higher level components such as WS-GRAM for managing the execution of services. The toolkit has reached in the reporting period release status and is available in Java and C versions. The toolkit is comparably well documented but requires still large effort for installation and is tightly integrated with components and services that are targeting to backwards compatibility with older version of the toolkit and cannot be easily removed.

As it constitutes the most commonly used platform in the Grid community the use of GT4 is recommended within TrustCoM.

1.6.3.2 pyGridWare[14]

The python implementation of the WS-ResourceFramework pyGridWare is an alpha level implementation. It includes support for WS-Addressing, WS-Notification, WS-Lifetime management and WS-Security. The maturity of the code is seen as not yet stable enough to be used within TrustCoM.

[13] http://www.globus.org/toolkit
[14] http://dsd.lbl.gov/gtg/projects/pyGridWare/

1.6.3.3 WSRF::Lite[15]

WSRF::Lite is based on the implementation of OGSI::Lite. It implements, in Perl, the Web Service Resource Framework (WSRF) and is supported by the Managed Programme of the Open Middleware Infrastructure Institute (OMII) through the project Robust Application Hosting under WSRF::Lite (RAHWL).

WSRF::Lite provides support for the following Web Service Specifications:

- WS-Addressing.
- WS-ResourceProperties.
- WS-ResourceLifetimes.
- WS-BaseFaults.
- WS-ServiceGroups.
- WS-Security.

The implementation is reasonably stable to be used within TrustCoM but is not recommended as the number of platforms should be limited.

1.6.3.4 Apache Web Services Toolkits (move this to 7.3?)

The Apache Axis Toolkit and the corresponding Apache projects Apollo (WSRF), Pubscribe (WSN) and Muse (WSDM) have reached release status and the incubation of the WSRF, WSN, WSDM parts is ongoing. The software is almost not documented and the different releases e.g. from Muse rely not on the most recent specification of Apollo and Pubscribe which make interoperability and integration with other toolkits a challenge.

The lack of appropriate documentation of this toolkit is seen as a risk which would lead to the recommendation to limit the use within TrustCoM. However as it realises the only WSDM implementation so far the use of this toolkit is recommended

1.6.3.5 WSRF.NET

The WSRF.NET toolkit provides a .NET based implementation of the WSRF specifications that can be used as a simple extension to the Web Services .NET framework. During the project's course, WSRF.NET has evolved to version 2.1 which, though still buggy, is currently widely used in particular in academic research. With the upcoming release of .NET framework 2.0, a new version of WSRF.NET has been announced that will resolve many issues that have been identified by the widely spreading WSRF.NET community.

The University of Virginia which develops the WSRF.NET toolkit is furthermore taking great care to address interoperability issues between platforms, as far as standard implementation(s) are concerned, i.e. in particular the Globus Toolkit 4 (see above) and (implicit) the Apache project. With the partially lacking detail of the

[15] http://www.sve.man.ac.uk/Research/AtoZ/ILCT

specifications, such cooperation between the development teams is of particular importance.

TrustCoM, too, makes use of the WSRF.NET toolkit and has provided feedback and reports to the University of Virginia which develops the WSRF.NET toolkit. It is recommended to continue the usage of the toolkit.

As mentioned above, WSRF has overlapping functionality with the WS-Transfer related set of specifications. While no toolkit for WS-Transfer has been released so far, research results by the University of Virginia indicate that WS-Transfer would prove more stable and reliable than WSRF. Since no serious advances have been made in this area since the last state of the art report, no further statements will be provided here.

1.6.3.6 UNICORE/GS[16]

The UniGrids (http://www.unigrids.org) is realizing a WSRF compliant framework based on the Unicore Grid Middleware widely used and deployed. As the maturity of the implementation is still very experimental the use of this toolkit within TrustCoM is not recommended.

1.6.3.7 .NET Framework 2.0 / WSE3.0

Microsoft has most recently released a new version of its .NET Framework (.NET 2.0), along with a new version of the Web Services Enhancements (WSE 3.0). Amongst others, WSE 3.0 supports the latest versions of the web services security specifications. Of further interest for TrustCoM will be the support of SOAP1.2 and MTOM which allows improved transport of large data sets across services – this is particularly relevant with respect to the Collaborative Engineering scenario.

Besides for the "programmatical" improvements as stated in http://msdn2.microsoft.com/en-us/library/t357fb32.aspx, the .NET Framework 2.0 also improves asynchronous web service invocation – in how far the new release is still fully interoperable with the Java platform and whether further impacts exist needs to be elaborated. The usage of this toolkit is recommended.

1.6.3.8 Other

Other web services middleware platforms include IBM ETTK, IBM Websphere, BEA WebLogic, and Systinet.

6.3.2 Business processing and transaction, including ESB

During the course of the project, the (mostly commercial) portfolio of business processing and service integration platforms, supporting or building upon web services, has expanded. Available products now include Microsoft BizTalk, IBM Websphere[25], Systinet, Blue Titan, Cape Clear[2], Iona Artix[17], Fiorano ESB, Amberpoint, Apache Synapse, and Arjuna Transaction Service Suite.

[16] http://unicore.sourceforge.net

There has been a clear evolution from standard enterprise application integration to message oriented middleware such as Corba (industry standard supported by Sun), Blue Titan, Cape Clear (see list above). However, although one should acknowledge the progress made in terms of ease of use, application interoperability, and the use of standards, there isn't a single solution relying wholly on predefined and established standards. It is by using open available standardized technologies that TrustCoM is aiming at filling in the gaps left by the above solutions. Only a solution akin to TrustCoM can easily enable on-the-fly thus dynamic service aggregation giving birth to the virtual organization concept.

Ideally, in the virtual organization realm, one would like services to be added automatically as demand for the given service evolves. All the different components should be tied together seamlessly providing users and other applications & / or services short or long-lived transaction tailored to their needs.

Service administration, service rights, service federation, service access should be administered automatically providing more dynamism.

6.3.3 Web services security products

It is particularly relevant to highlight the wide range of web services security based products that have been introduced or announced in the market during the course of the project.

Web services security appliances and web services security management products include: DataPower XS40, Amberpoint, Forum Systems XWall, Layer7, Reactivity Gateway, Vordel XML Security Appliance.

Federated web services security and identity products include: BMC Federated Identity Manager[1], Computer Associates SiteMinder[3], HP Select Federation[6], IBM Tivoli Federated Identity Manager[7], Microsoft Active Directory Federated Services (ADFS)[9], Novell Access Manager[10], Oracle COREid Federation[11], Ping Identity PingFederate[12], Sun Java System Access Manager[13], Symlabs Liberty Identity Manager (SLIM)[14], Trustgenix IdentityBridge[15].

This clearly shows the enormous momentum in this area around web services security and federated identity. Continuing to adopt and build upon the web services security and federation specifications definitely remains important for the TrustCoM security developments.

The distribution of applications over a closed or possibly open network has created the need for added, enforced security, sometimes introducing new security concepts.

The standards bodies (OASIS and W3C) have foreseen this need and a lot of work to formalize proposed security standards has been done, with much more still underway. Software and hardware companies have been implementing these standards using them at their own discretion. Where some companies see security as a network-level component, others implement security directly into software. The evolution of both visions will give us more insight on the security approach.

TrustCoM itself is basing itself on the latest available standards, making sure it abides by their definitions in order to enable interoperability. Furthermore, with

TrustCoM, security concepts have been refined (e.g. trust management, security policy management, secure federation). TrustCoM is the one opportunity to translate theory into viable software solutions and components.

Overall, the fact intensive research is going on proves distributed computing and the inherent security issues are the future of EAIs, ESBs and organization IT as a whole. The ability to create short or long-lived transactions in virtual organizations will be a key feature of the next-generation information systems. In this sense the research led within TrustCoM but also within IBM's ETTK or MS's WSE (and in a broader dimension, .NET) is highly important.

6.4 Semantic Web and Ontologies

Following the review of the state of the art of research in Semantic Web technologies and Ontologies in June 2004 the Trustcom project noted that the field was active in research from several well established research teams who were developing both semantic technologies as well as web technologies. The state of development was such that several teams were about to propose their individual approaches as the basis for standardisation to standards bodies (mostly W3C), and their was no consensus on a single approach to ontologies relating to Web Services and issues of quality, security, trust, reputation and business processes. As a result of this observation it was decided that Trustcom would neither adopt any existing Semantic Web technologies, nor would it develop its own. Rather Trustcom would integrate functional WS* specifications using semantically impoverished service descriptions, and the expectation that semantically rich representations would be agreed by the research community, and could then be incorporated into the Trustcom approach.

This review considers the advances in semantic web technologies in the last year, and whether these decisions are still appropriate.

Overall developments in Semantic Web technologies can be characterised by the figure below which compares those technologies that are rich or poor in semantic expressiveness, with those which are general or specifically designed for the Web architecture.

Figure 8 illustrates how recent developments have either been high in semantic expressiveness or in Web specificity, but not in both, while the arrows indicate the expected trend in research to move technologies to fill the top right hand quadrant. When this quadrant is populated by robust technologies that have realistic adoption hopes, then it will be appropriate to adopt them within the Trustcom framework.

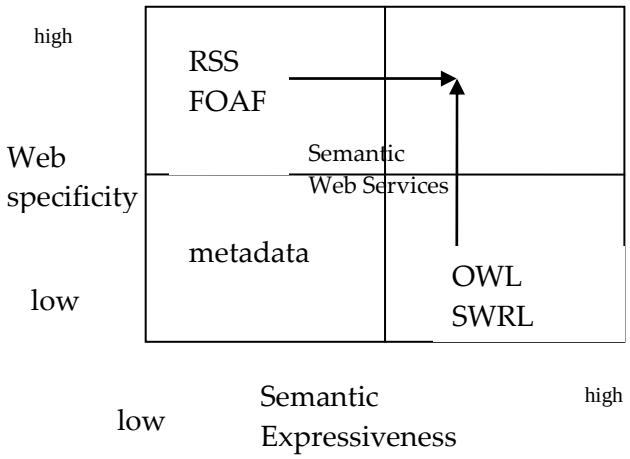


Figure 8: Semantic Expressiveness against Web Specificity of recent technologies

6.4.1 Developments in Web Reasoning

Several projects that have been developing rule languages for the web, either from existing AI languages, or from Web technologies have completed and have consequently made submissions to W3C for their specifications to become the basis of a recommendation.

These submissions include:

SWRL – Semantic Web Rule Language, May 2004: National Research Council of Canada, Network Inference, and Stanford University

SWRL-FOL - Semantic Web Rule Language First Order Logic, Jan 2005: National Research Council of Canada, Network Inference, and Stanford University

WRL - Web Rule Language – June 2005: DERI Galway at the National University of Ireland, DERI Innsbruck at the Leopold-Franzens-Universität Innsbruck, The Open University, Software AG, Forschungszentrum Informatik (FZI), BT, and National Research Council Canada

As well as the established cwm tool to reason over rules in N3 developed by Tim Berners-Lee which did not require formal submission.

As a result of this activity, W3C formed a working group[17] in November 2005 to develop a recommendation for a Rule Interchange Language for reasoning over the semantic web.

These developments are in line with the project's expectations, showing that those active in the area are continuing development, and it would be a duplication of effort for the project to put effort into this activity rather than wait to use their agreed output.

[17] http://www.w3.org/2005/rules/wg

6.4.2 Developments in Semantic Web Services

In the area of Semantic Web Services a similar position holds, with several submissions being received to address ontologies and frameworks for web services:

OWL-S, Web Ontology language for Services[18], Nov. 2004 - France Telecom, Maryland Information and Network Dynamics Lab at the University of Maryland, National Institute of Standards and Technology (NIST), Network Inference, Nokia, SRI International, Stanford University, Toshiba Corporation, and University of Southampton

WSMO – Web Service Modelling Ontology[19], April 2005: DERI Innsbruck at the Leopold-Franzens-Universität Innsbruck, Austria, DERI Galway at the National University of Ireland, Galway, Ireland, BT, The Open University

SWSF – Semantic Web Services Framework[20], April 2005: National Institute of Standards and Technology (NIST), National Research Council of Canada, SRI International, Stanford University, Toshiba Corporation, and University of Southampton

Web Services Semantics[21] – Oct 2005: IBM

As a result of these W3C hosted a workshop in June 2005 on Frameworks for Semantics in Web Services in order to establish the variation of approaches in the research community, and whether a consensus could be reached to work toward a recommendation. Trustcom submitted a position paper to this meeting in order to have a voice in any plans that were made, but without the expectation of committing substantial resources. Since no clear consensus could be reached on a single approach, and given that the area is clearly active with those wishing to reach a standardised approach, W3C established an interest group[22] on the topic in Nov. 2005 where issues can be raised and resolved to move towards the degree of commonality required to initiated a recommendation track working group.

This development is unfortunately not as advanced as the Trustcom project had hoped, but given the conflicts that exist within the technical community it would not be productive to join in at this stage, so the project confirms its decision 18 months ago not to develop technology in this area, but to wait for a consensus to arise.

[18] http://www.w3.org/Submission/2004/07/

[19] http://www.w3.org/Submission/2005/06/

[20] http://www.w3.org/Submission/2005/07/

[21] http://www.w3.org/Submission/2005/10/

[22] http://www.w3.org/2002/ws/swsig/

7 Conclusion

The purpose of this updated review is to summarise:

- ongoing basic research projects that might contribute new results that Trustcom should adopt;
- ongoing integration projects which may overlap with Trustcom with the result that common work could be rationalised;
- standardisation activity which Trustcom can build upon instead of developing things itself;
- industrial application of technologies that are ahead of Trustcom, and which should be adopted technically or seen as competition for exploitation.

Within Europe there has been funded work on virtual organisations which is building on the existing products for social networking to address the real problem of bringing potential Vo partners into a network. Trustcom has already introduced the notion of an Enterprise Network to address this issue, but will have to put more resources into this stage of VO development, given the risks identified.

It is clear that the agreement and publications of standards for WS-Agreement, WS-CDL and the Semantic Web is taking longer than was assumed, but that the first two will be agreed before the end of the project, while the Semantic Web technologies for Web Services may be yet further delayed while the community reaches consensus on the appropriate approach.

The industrial developments in the last 18 months have resulted in several major SRM products including aspects of contract management which they did not previously include. These currently address library and editor based human contract authoring, which is compatible with the Trustcom approach. However, the introduction of contract management into supply chain management for large corporations provides an alternative market for the Trustcom technology to the VO environments that have been its target.

Industrial developments in platforms and tools cast doubt on the stability of WSRF and the maturity of the current grid technology specifications which may change further as experience of their use continues. This is of course both going to influence the stability of the Trustcom reference implementation, and be an activity into which Trustcom will contribute through Specification Profiles in the Trustcom Framework.

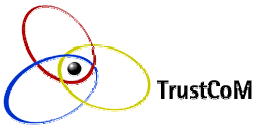
The main conclusion is that there are no major surprises that require the Trustcom project to change its course.

8 References

[1] BMC Software Federated Identity Manager 5.2
http://www.bmc.com/products/attachments/Federated_Flyer-4.pdf
[2] Cape Clear 6.5
http://www.capeclear.com/download/CC65_Datasheet.pdf
[3] Computer Associates SiteMinder
http://www3.ca.com/Files/DataSheets/etrust_siteminder_fss_data_sheet.pdf
[4] DataPower XS40 XML Security Gateway
http://www.datapower.com/docs/datasheet_xs40.pdf
[5] Grid Security: the Globus Perspective, GlobusWorlsd 2005,
http://www.globusworld.org/2005Slides/Session%203c.pdf.; http://www.globus.org
[6] HP Select Federation
http://www.managementsoftware.hp.com/products/slctfed/ds/slctfed_ds.pdf
[7] IBM Tivoli Federated Identity manager
ftp://ftp.software.ibm.com/software/tivoli/datasheets/ds-federated-identity-mgr.pdf
[8] Internet2:
http://www.incommonfederation.org/technical.cfm; http://www.incommonfederation.org/benefits.cfm
[9] MS Active Directory Federated Services
http://download.microsoft.com/download/3/a/f/3af89d13-4ef4-42bb-aaa3-95e06721b062/ADFS.doc
[10] Novell Access Manager 3.0
http://www.novell.com/collateral/4621399/4621399.pdf
http://www.novell.com/documentation/secure_access16/pdfdoc/secureaccess16/secureaccess16.pdf
[11] Oracle COREid Federation
http://www.oracle.com/products/middleware/identity-management/docs/wp_oracle_coreid-federation_25.pdf
[12] Ping Identity PingFederate 2.0 server: http://www.pingidentity.com/products/pingfederate
[13] Sun Java System Access Manager
http://wwws.sun.com/software/products/identity_srvr/wp-idsrvr-overview.pdf
[14] Symlabs Liberty Identity Manager
http://www.symlabs.com/Products/SLIM.html
[15] Trustgenix IdentityBridge Enterprise Edition
http://www.trustgenix.com/assets/IB-EE_Spec_Sheet_2005.pdf
[16] G. Gebel: Multiprotocol Federation Interoperability Demonstration. Burton Group Identity and Privacy Strategies, Identity Management technology thread, 1st November 2005,
www.burtongroup.com
[17] Iona Whitepaper – Overview of Artix Transaction support:
http://www.iona.com/devcenter/artix/articles/0304transactions.pdf
[18] Computer Associates whitepaper - eTrust TransactionMinder: Securing Web Services White Paper;
http://www3.ca.com/Files/WhitePapers/transactionminder_securing_web_services.pdf
[19] VS3000 - XML Security Appliance
http://www.vordel.com/downloads/vs3000_datasheet.pdf
[20] Forum Systems: Forum Sentry datasheet
http://forumsystems.com/papers/Sentry_Data_Sheet_Spring_2004.pdf
[21] Reactivity Gateway:
http://www.reactivity.com/products/gateway_d.html
[22] Microsoft WSE 3.0 overview
http://msdn.microsoft.com/webservices/webservices/building/wse/default.aspx?pull=/library/en-us/dnwse/html/newwse3.asp#new_topic2

[23] Microsoft WSE 3.0 in detail
http://msdn.microsoft.com/webservices/webservices/building/wse/wse30readme.aspx
[24] Arjuna Transaction Service Suite
http://www.arjuna.com/products/arjunats/
[25] IBM Websphere Application Server delivers Business Flexibility
[26] IBM ETTK Specifications
http://awwebx04.alphaworks.ibm.com/ettk/demos/wstkdoc/README.htm
http://www-306.ibm.com/software/tivoli/products/federated-identity-mgr/
http://www-306.ibm.com/software/webservers/appserv/was/WAS_V6_DH_Brown.pdf
[27] IBM Tivoli Standards
http://www-306.ibm.com/software/tivoli/features/soa/standards.html
[28] Securing Web Services for Enterprise-Class Systems: an Amberpoint whitepaper
[29] IEEE Workshop on Policies for distributed Systems and Networks, Stockholm, June 2005, IEEE CS press.
[30] D. Agrawal, S. Calo, J. Giles, K.-W. Lee, and D. Verma. *"Policy Management for Networked Systems and Applications."* In Proceedings of 9th IFIP/IEEE International Symposium on Integrated Network Management, Nice, France, IEEE, May 2005.
[31] D. Agrawal, J. Giles, K.-W. Lee, and J. Lobo. *"Policy Ratification."* In Proceedings of 6th IEEE International Workshop on Policies for Distributed Systems and Networks, Stockholm, Sweden, IEEE, June 2005.
[32] R. A. Kowalski and M. J. Sergot (1986). *"A logic-based calculus of events."* New Generation Computing **4**: 67-95, 1986.
[33] R. Darimont and A. van Lamsweerde (1996). *"Formal Refinement Patterns for Goal-Driven Requirements Elaboration."* 4th ACM Symposium on the Foundations of Software Engineering (FSE4): 179-190, 1996.
[34] E. C. Lupu and M. S. Sloman (1999). *"Conflicts in Policy-Based Distributed Systems Management."* In IEEE Transactions on Software Engineering - Special Issue on Inconsistency Management **25**(6): 852-869, 1999.
[35] D. C. Verma (2001). *"Policy-Based Networking: Architecture and Algorithms"*, New Riders Publishing, 2001.
[36] A. K. Bandara. *"A Formal Approach to Analysis and Refinement of Policies."* Doctoral Thesis, Imperial College London, London, 2005.
[37] N. Damianou, N. Dulay, E. C. Lupu, and M. S. Sloman. *"The Ponder Policy Specification Language."* In Proceedings of 2nd IEEE International Workshop on Policies for Distributed Systems and Networks, Bristol, UK, Springer-Verlag, January 2001.
[38] Bittencourt, F. and Rableo, R.J. (2005) A Systematic Approach for VE partners slection using the SCOR model and the AHP method. In L. M. Camarinha-Matos, H. Afsarmanesh and A. Ortiz (Eds.) Collaborative Networks and Their Breeding Environments, Springer, 99-108.
[39] Camarinha-Matos, L. M., Afsarmanesh, A., and A. Ollus., M., (2005) ECOLead: A holistic approach to creation and management of dynamic virtual organisations, in L. M. Camarinha-Matos, H. Afsarmanesh and A. Ortiz (Eds.) Collaborative Networks and Their Breeding Environments, Springer, 3-16.
[40] Cao, H and Wang, D (2003) A simulation based genetic algorithm for risk based partner selection in new product development. International Journal of Industrial engineering – theory, applications and practice, 10(1) 16-25.
[41] Ip, W.H., Huang, M, Yung, K.L and Wang, D. (2003) Genetic Algorithm solution for a risk-based partner selection problem in a virtual enterprise. Computers and Operations Research 30, 213-231.
[42] Petersen and Divitini, Using Agents to Support the selection of virtual enterprise teams. International Journal of Production Planning and Control, Vol. 12, No. 3, Issue Apr 2001, pp. 224-233
[43] S. A. Petersen, J. Rao and M. Matskin. AGORA Multi-agent Architecture for Implementing Virtual Enterprises. In Proceedings of the Norwegian Information Technology Conference (NIK'2003), Tapir, 2003

[44] Zeng, Z, Yan, L, Shujuan, L, Zhu, W. (2005) A new algorithm for partner selection in Virtual Enterprises. Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05), 884-886, IEEE.
[45] Karen S. Cook, Russell Hardin, Margaret Levi Cooperation Without Trust? (Russell Sage Foundation Series on Trust); Russell Sage Foundation Publications (July 31, 2005)



Deliverable

2a

State of the Art Evaluation

WP10 State of the Art – Conclusions and Recommendations

Editor - ICSTM

25/12/2004

Version 1.0

TrustCoM

A trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations

SIXTH FRAMEWORK PROGRAMME

PRIORITY IST-2002-2.3.1.9

Prepared for the European Commission under FP6 Contract No. 01945 as a deliverable form

LEGAL NOTICE

The following organisations are members of the Trustcom Consortium:

SchumbergerSema Sociedad Anonima Espanola,
Council of the Central Laboratory of the Research Councils,
BAE Systems,
British Telecommunications PLC,
Universitaet Stuttgart,
SAP AktienGesellschaft Systeme Anwendungen Produkte in der Datenverarbeitung,
Swedish Institute of Computer Scence AB,
Europaeisches Microsoft Innovations Center GMBH,
Eidgenoessische Technische Hoschschule Zuerich,
Imperial College of Science Technology and Medicine,
King's College London,
Universitetet I Oslo,
Stiftelsen for industriell og Teknisk Forskning ved Norges Tekniske Hoegskole,
Universita degli studi di Milano,
The University of Salford,
International Business Machines Belgium SA .

Deliverable datasheet

Project acronym: TrustCoM

Project full title: *A trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations*

Action Line:

Activity:

Work Package: 10

Task: State of the Art

Document title: State of the Art: Conclusions and Recommendations

Version: 1.0

Document reference:

Official delivery date: 30/1/2005

Actual publication date: 25/12/2004

File name:

Type of document: Public Report

Nature:

Editors: ICSTM

Reviewers:

Approved by: Michael Wilson

Version	Date		Sections Affected
V.1	25/12/04	All	
V1.1	28/2/05	NRCCL	9 – Legal Aspects

Table of Content

1 Introduction

This document is provided as an addendum to the State of the Art deliverable D2 of the TrustCoM project (IST 01945). It was requested following a formal presentation of the deliverable to the EC and project reviewers, and was given the specific remit of drawing *"clear conclusions from the state-of-the-art studies and recommendations related to the context of the envisioned TrustCoM project plan."* It should therefore be read in conjunction with the aforementioned deliverable as the technologies and conceptual frameworks covered in the State of the Art will be mentioned but not described in this document. Whilst the aim of the State of the Art was to be comprehensive and to identify applicable technologies and conceptual frameworks, this document aims to be analytical and provide a critical viewpoint on the related work. In addition to the remit specified above, we have also attempted to identify particular areas of risk for the project and potential mitigating factors. Most of the existing work related to TrustCoM is comparatively new and still evolving. Wherever possible we have attempted to extrapolate the trend of the current evolution rather than comment on detailed aspects of specific versions. As a consequence of the State of the Art review a number of more in-depth studies were conducted in the last few months as part of Work Package 4 ("Framework Specifications"). These studies aimed to evaluate the applicability of a number of web services (WS-*) specifications to the TrustCoM framework by investigating their use in a prototypical TrustCoM scenario. Although, it is not the aim of this document to give a detailed account of these studies, the conclusions and recommendations presented here take into account their results.

TrustCoM aims to provide an integrated framework enabling secure collaborative business processing in on-demand created, self-managed, scaleable and highly dynamic Virtual Organisations. This objective can be achieved only through the integration of a large spectrum of different tools and techniques that cater for the creation and management of Virtual Organisations, collaborative business processes, contract management and service level agreements, trust management and security. In addition, TrustCoM takes into account the legal aspects of virtual organisations and the socio-economic aspects of business collaboration and in particular fostering trust in business relations and understanding the impact of trust and reputation in the creation and management of virtual organisations. Consequently, this document is subdivided according to the thematic areas of the project in a similar manner to the state of the art deliverable. This also facilitates cross-referencing of information between the two documents.

2 Socio-Economic Aspects

In current business practices, organisations rely heavily on the use of Internet technologies. This has led over the last decade to transformations of the organisational structures and processes that leverage the use of highly dynamic and widespread business collaborations and relations. Two types of business relations are particularly affected by this change: dynamic and possibly transient collaborations for product development, purchasing, sales and services, and dependencies on a wider and more dynamic set of other businesses for the supply of products and services. In this emerging landscape, businesses need strong incentives and mechanisms to trust other businesses. Trust and reputation are concepts that play an increasingly significant role in the formation and maintenance of both business relationships and in the development of market places.

Analyses of the colloquial usages of trust show that it is often overloaded as a synonym for confidence in the ability of an agent (individual or organisation) to undertake a role in a business process (e.g. to have confidence in somebody's ability) within the terms and conditions agreed (e.g. time, quality, cost etc). Reputation is similarly often an evaluation of past performance to perform a role within the terms and conditions agreed in order to determine a measure of confidence in the competence to perform that role in the future. Beyond these usages, studies of trust identify its unique quality as being the intention of a supplier to share the intentions of a customer in circumstances when the agreed terms and conditions (e.g. by contract or SLA) may not apply, or be explicitly broken by the customer (e.g. to supply the goods that were intended, even if they were mis-named in the order; or to supply them to a deadline sooner than that contracted; or to supply the goods to a higher quality level than that contracted). The motivation for the supplier to establish the trust of customers in this specific sense, is to increase the probability of future business. Transfer of reputation and trust across business roles, or contexts have been shown to be important in brand management and the extension of brands to new products. Consequently, trust can transfer to new business opportunities in different ways to competence, and the relation of reputation to trust is more complex than that of reputation to competence. Current computational models of reputation and trust as predictions of risk in fulfilling roles in business processes, and simple legal analyses of contracts or SLAs do not address these unique elements of trust as identified by socio-economic analyses.

It is therefore essential to be able to understand the factors that influence trust related business decisions and the means to foster trustworthiness in a business environment. In particular, the TrustCoM project aims to develop a framework that provides support for trust related information such as risk, trust, experience, evidence and reputation. There are however various ways in which this information originating from the participant members of a VO can be aggregated and transferred across business contexts, in order to identify new partners for the VO or to participate in new VOs. The choice of how to aggregate the various trust information elements has to rely on their perceived value and use in business transactions. To achieve a better understanding of this perceived value and the

behaviour of individuals when faced with trust and reputation information, socio-economic studies have relied on experiments and trust games as well as on empirical observations from consumer systems such as eBay. Experimental studies in the laboratory and empirical or experimental studies in the field are to be considered as important complements.

An established experimental technique for evaluating trustworthiness is the *trust game* introduced by Berg, Dickhaut, and McCabe (1995)[1]. In this game, trust is measured by the amount that one of two players, the *investing player*, unilaterally invests by sending it to the other, the *trusted player*. The trusted player receives three times the amount invested and may then return some amount to the investing player. The amount he returns provides a measure of the trusted player's trustworthiness.

Such trust games and empirical observations have already led to a number of interesting preliminary observations. First, that reputation has monetary value and that trustworthiness can lead to users being prepared to accept higher costs in return for dealing with more trusted partners, second that behaviour differs depending on whether the transactions and participation in the market place are considered short term or long term and third that trust and reputation systems can be maliciously abused for profit. Other more theoretical studies attempt to achieve a better understanding and characterisation of the various components of trust and reputation such as confidence, assurance, good will, commitment and reciprocity.

Conclusions

- An understanding of the factors that underpin trustworthiness in business transactions will influence the TrustCoM framework from two different points of view: first, it will provide the basis for deciding how these factors need to be aggregated in the trust management part of the framework and second, it adds an objective to the overall architecture of the framework which needs to be designed in order to increase trust between the VO partners. Further studies in this direction would therefore be particularly beneficial.
- Equally important is to achieve an understanding of how the other aspects of the proposed TrustCoM framework and in particular the monitoring components, Service Level Agreements and the contract management part of the framework can be used to mitigate the risks, provide increased trustworthiness and compensate for deficiencies in the trust factors e.g., unknown reputation, unknown risk.

Recommendations

- Studies (particularly using the trust game) towards a better understanding of the trust and reputation component factors should be pursued. However, particular attention should be devoted to their combination.
- These studies should also be broadened to address the problem of trust in a wider context comprising assurance mechanisms such as contract management and monitoring.

[1] Berg J., J. Dickhaut, and K. McCabe, 1995, Trust, reciprocity, and social history, *Games and Economic Behavior* 10, 122-142.

3 Frameworks for Virtual Organisations

A large number of studies have already been conducted on the characteristics and behaviour of virtual organisations as well as on enterprise integration reference models. The related work is especially rich in a number of modelling frameworks and notations for expressing the different types of VO as well as their operation. Different types of VO include *targeted VOs* that are characterised by specific goals that the VO attempts to achieve through a deliberate cooperation between the participants as well as *dynamic VOs* that are characterised by the dynamic membership of their participants who may exhibit opportunistic behaviour. A substantial amount of effort has investigated the different operational models of these VO in terms of both their operational business processes and their evolution. A large number of taxonomies have been developed as well as a wide spectrum of techniques for enterprise modelling. Characteristic of this approaches are both the St. Gallen, the CIM/CIMOSA approach, the GIM/GRAI modelling framework as well as the PERA and GERAM reference architectures. However, each of these models, methodologies and reference architectures have distinct focuses e.g. manufacturing processes, human inter-relationships, planning and operations, and whilst all have some valuable input to offer at the conceptual level, none of them has gained widespread acceptance. Also, and more importantly, none provide a concrete implementation that automates the VO management functions and operations or takes into account technology aspects, with the exception perhaps of the RosettaNet implementation framework. The scope of the latter is however strongly focussed on document exchange. Nevertheless, a critical mass of research exists as well as a broad understanding of the issues surrounding enterprise modelling and business-to-business integration.

In contrast to the above approaches, Grid-based models of virtual organisations (e.g., OGSA) focus on practical implementations on lower technology and communication layers for resource sharing and task scheduling and distribution. Although a number of issues relating to service discovery and composition have been addressed as part of these efforts, their focus also constitutes their main shortcoming. Grid-based systems are mostly flat structures where a number of organisations share resources and task execution is distributed but a grid structure is not an autonomous organisation that can itself participate in other Virtual Organisations. Furthermore, although a number of security and access control issues have been addressed, Grid-based systems remain fundamentally open systems where issues of trust (and distrust) between partners focus on usage of resources rather than the confidentiality of information and integrity of systems. Finally, the implementations available form rather monolithic blocks that are not sufficiently modular to offer reuse of independent parts. Owing to the convergence between web service architectures and Grid based architectures, it is expected that the latter implementation issues will eventually be addressed. However, this may not occur in a sufficiently timely fashion to offer TrustCoM a basis for development.

In a similar fashion to Grid-based environments TrustCoM should have an implementation and technology driven approach. The aim is to provide a concrete implementation (or implementation components) that can be instantiated to support the operation of Virtual Organisations even if this implies loss of generality in the VO models that TrustCoM can support. The concrete benefit that TrustCoM can offer to the business landscape is thus not in richer abstractions but in greater support for automation both in the formation and evolution of Virtual Organisations but also in the implementation of their business model through service aggregation. Achieving this will require the development of novel concepts and techniques for:

- providing self-management and adaptability through an integrated policy-based approach
- integrating business processes with service level agreements and the underlying supporting infrastructure
- providing a rigorous yet flexible security model that can adapt to variations of trust between the different partners
- integrating the contract and service level agreement aspects with the trust management infrastructure in order to foster new business relationships
- providing a recursively composable model of virtual organisations

The latter aspect is essential and represents a considerable departure from all existing models. Together with the integration aspects and the policy-driven approach, they constitute the main elements of innovation but also the main challenges that the project faces both at the conceptual level as well as from an implementation viewpoint.

Conclusions

- A number of conceptual models and modelling frameworks have been developed in the course of several collaborative projects that provide valuable input for TrustCoM's conceptual model for VOs. However, particular attention has to be paid to the implementation and technology aspects that are all too often ignored in these studies.
- At the other end of the spectrum Grid-based systems provide some of the core elements necessary for implementing dynamic collaborations. However, they lack a number of conceptual elements required for adequate modelling and automation of VO structures.
- Conceptually, TrustCoM is midway between the former two models and strives to cater for business transactions and recursive compositions of VOs whilst retaining and increasing the level of automation in the VO support infrastructure.

Recommendations

- TrustCoM should develop its own conceptual model for VOs and this development needs to happen as early as possible in the development life-cycle. To this end it is necessary to establish small and highly cohesive teams working on each of the thematic areas (trust, security and contract management) and their earliest priority

should be to explicitly identify the concepts and information provided by the other thematic areas on which their model relies.

- Once completed the conceptual model should be evaluated against the results of past projects that have focussed on modelling frameworks in order to: i) identify additional functionalities that the model can cater for and ii) identify missing functionalities that are judged critical for the development of the TrustCoM framework.
- The success of the conceptual model and of the TrustCoM framework as a whole critically depends on two elements: providing self-management and adaptation, providing integration between the different thematic areas. Both of these aspects need to be addressed early in the architectural and implementation stages. Whilst the initial scoping of the integration effort will be addressed by characterising explicit dependencies between the thematic areas, providing self-management and adaptation is likely to rely on two core concepts that need to be developed and implemented early: business processes and policy (see also Enabling Technologies Section).

4 Contracts and Service Level Agreements

In essence, there are two types of studies belonging to this overall thematic area: work that aims to provide support for managing legal contracts between organisations and automate part of the process associated with their definition and enforcement (e.g., BCA architecture), and work that originates from the network and systems management community relating to customer-provider relationships and the Quality of Service promise associated with a (web-) service (e.g., WSLA, GRASP SLA framework, HP's Web Service Modelling Framework). In addition to the above, a number of parallel studies such as SLAng and the more recent work at SICS aim achieve a better formal treatment of the SLA concepts and notations in order to provide precise semantics and the means to formally reason about the specifications.

The BCA architecture defines a comprehensive infrastructure for dealing with legal contracts comprising sophisticated means of describing contracts as well as processes for contract arbitration and enforcement. However, the framework is rather complex and its implementation status is uncertain. It also does not appear to have been used outside a relatively restricted research environment around the DSTC. From a conceptual view point the framework does however propose a number of solutions that would be worth investigating in conjunction with a legal team.

Work on Service Level Agreements (SLAs) on the other hand is comparatively more mature and better understood. Originally developed as part of the network and systems management community in order to cater for the specification of the Quality of Service (QoS) parameters characterising the provision of network connectivity services, this work has evolved into general frameworks for the characterisation of application level services and more recently business services. Most of the solutions proposed in this area provide the means for: specifying SLAs and associating them with the WSDL services concerned, discovering and locating services based on profiles of QoS that can be delivered for those services, defining simple negotiation protocols for negotiating QoS parameters, and monitoring the compliance with the SLA objectives (including monitoring and metric definition). However, the extent to which these features are supported varies greatly amongst the different SLA solutions proposed. Probably the most concrete framework that is likely to provide a solid foundation for TrustCoM is WSLA, which in addition to specification and structuring of SLA agreements also provides detailed monitoring aspects including an extensible framework for metric definition. The other framework of particular interest is WS-Agreement. Originating initially from the OGSI framework, and a good example of how Grid platforms evolve towards a more open web service environment, WS-Agreement caters for the discovery of services including SLA retrieval and negotiation and is compliant with the other WSRF specifications. WS-Agreement is however a relatively new specification.

ebXML Trading Party Agreement and Collaboration protocol Agreement would also be an alternative. However, their specifications and implementations are tightly

coupled to the other ebXML specifications, which do not seem to integrate well with the other web service specifications.

Conclusions:

- The contract and service level agreement framework in TrustCoM can draw heavily on both the WSLA and WS-Agreement specifications and their implementation support. The aim is to design a framework that caters for both monitoring and enforcement aspects as well as service location and negotiation.
- Support for managing more formal business agreements of the VO needs to also be provided. Although, the BCA provides a significant conceptual model that can help in this development only a subset of the concepts described in the BCA are likely to be required or useful.
- None of the above studies (with the exception perhaps of some later studies on the BCA) examine in detail how trust and reputation information is to be used in conjunction with contracts, SLAs and their associated processes.
- Although, some of the frameworks mentioned above present some features for reacting to SLA violations, this is one of the areas in which TrustCoM could bring a significant contribution.

Recommendations:

- An explicit conceptual model for supporting agreements at both business and service level needs to be developed based on a conjunction of WSLA, WS-Agreement and relevant concepts from the BCA
- The development of this conceptual model needs to devolve significant efforts to two aspects: a) the impact and use of trust and reputation relationships in service discovery, SLA negotiation and enforcement phases and b) the handling of SLA violations in a more flexible form similar to business process descriptions.
- Concurrently, a group formed in conjunction with the legal team in TrustCoM should identify which elements within the BCA and contract management in general are likely to be the most useful within the framework as well as what security controls in terms of confidentiality, integrity and non-repudiation will be necessary.
- Finally, there is a need to identify which specific implementations or parts of implementations could be re-used within TrustCoM.

5 Collaborative Business Processes

By comparison with the other thematic areas considered in the State of the Art, Collaborative Business Processes are probably the best understood and defined technology. Indeed, the issues regarding executable collaborative business processes in the last few years have been more focussed towards standardisation aspects rather than basic research, as many software vendors and business integration consultants are using a wide spectrum of proprietary protocols. Standardisation allows addressing the problems of executable business process aggregation and collaboration across administrative domains that use proprietary solutions as well as outsource workflow control and implementation to third parties. A number of specifications have been investigated including: WS-Coordination that defines the means to coordinate distributed actions during process runtime including agreement on outcome through the propagation of activity contexts, WS-Transactions that extends context information to include transactional capabilities for both atomic transactions (WS-AtomicTransactions) and long running business transactions (WS-BusinessActivity), BPML/WS-CI that focuses on the choreography of message exchanges starting at design time across multiple parties and BPEL4WS that provides the means to describe abstract and executable business processes in terms of their structure, control as well as offered and invoked service interfaces. BPEL4WS and BPML/WS-CI have overlapping functionality, in particular for the business process specification although from different points of view. Whilst BPEL4WS relies on supporting Web Service standards such as the WS-Coordination model, which relies on the use of a single coordinator entity or a hierarchy of coordinators to control the execution of the workflow, WS-CI advocates a more loosely coupled choreography model with distributed control. Since many of the use-case scenarios established for TrustCoM do not explicitly require the use of a coordinator the latter mode may provide some flexibility. Regrettably, development of the BPML/WS-CI has been abandoned with most of the concepts being integrated in a new specification, WS-CDL. The latter however, is still evolving and is not sufficiently stable to base the TrustCoM development upon it, at least during the first stage of the project. WS-CDL is also not catering for a collaborative business process choreography description capturing complex message exchanges across administrative domains, for instance in tendering and quotation processes.

Another option that has been investigated is the use of the ebXML-*series of specifications. However, these do not seem to integrate well with the other WS-* specifications since they advocate their own way of implementing messaging, service repository access, security, etc. It was therefore felt that these should not be investigated further.

Conclusions

- The most promising and stable approach to be used within TrustCoM is based on BPEL4WS/WS-Coordination/WS-Atomic Transaction. WS-CDL should however be monitored for further developments.
- Few, if any of the existing studies address business processes in conjunction with SLAs and none in conjunction with trust and reputation information for service selection and composition.

Recommendations

- Business Process Modelling and implementation should proceed based on the above-mentioned specifications and any available packages providing adequate implementations.
- As the business process infrastructure needs to be deployed for application level services the consortium should investigate further how the same infrastructure can be used to support the administrative and possibly adaptation processes inside the VOs.
- After an initial phase of defining and implementing the core business process functionality, the efforts should focus on two aspects: integration with the SLA infrastructure and leveraging the availability of trust and reputation for providing enhanced flexibility in the enactment of the processes especially across administrative domains.

6 Enabling Technologies

The spectrum of enabling technologies available for TrustCoM broadly divides into the following categories: web service specifications and their infrastructure support, grid technologies, semantic web and ontology based techniques, and tools and platforms for implementation. Implementation tools and platforms further sub-divides in generic platforms and specific tools, implemented before the start of the project by the various partners in the project. Generic platforms have been the focus of more detailed studies as part of Work Package 4 (Framework Specifications) whilst specific implementation toolkits and tools from partners need to be further investigated in Work Package 5 ("Generic Methods and Tools") in terms of actual compatibility and interoperability between the specific implementations.

TrustCoM has made an early commitment to web service standards (WS-*) and associated technologies. This constitutes at the same time an opportunity and a major source of risk for the project. It is an opportunity because in the current state of development, web services seem to be the technology of choice for interoperability and collaboration across heterogeneous implementation domains. It is a major source of risk, because beyond the basic SOAP/XML-RPC communication mechanisms many of the more advanced specifications are still in the draft phase. The specifications are sometimes overlapping and occasionally incompatible as widespread agreement over the set of functionalities that each standard should cover has not been achieved. The implementation status of a number of the specifications is unknown. IBM and Microsoft are the most active players in this arena but their software packages that implement these emerging specifications are frequently updated, incomplete in some respects and the vendors classify the implementations as technology previews.

Of the various standards, SOAP, WSDL, XML Schema, WS-Addressing and WS-ReliableMessaging constitute the basis for communication across domain boundaries. Together they provide the means to describe the functionality of web services, address them in a transport layer agnostic fashion and exchange messages reliably between the various components. In addition WS-Addressing is relied upon by many other specifications. Services need to be discovered and bound to dynamically. Whilst UDDI provides adequate means for registering and discovering services and should be used in the project, it is likely that standard implementations will need to be extended to accommodate the additional information about a service used within TrustCoM such as SLAs associated with a particular service, trust level and reputation.

Event-based asynchronous notifications play a particularly important role within a VO as changes of state need to be communicated to a potentially large and dynamically changing set of components. Two specifications WS-Eventing and WS-Notification would form good candidates for this purpose. Both are robust specifications that define core features for event-based systems, although they lack some state-of-the-art properties. Additionally, the specifications overlap in scope and are currently incompatible because they target slightly different applications.

WS-Eventing is a basic and general-purpose publish-subscribe event-based system. WS-Notification is aimed at management of resources and considered to be used in conjunction with WS-RF. WS-Notification also provides more sophisticated means of managing subscriptions, brokering and topic based dissemination of events. Although, the two specifications are expected to merge at some point in the future, this is not likely to happen during the first phase of development in the project.

To support management of web services and resources using web service-based protocols both WS-Resource (and on tom WS DM MUWS/MOWS) as well as the WS-Management specification (including WS-Transfer and WS-Enumeration) are an emerging effort for getting and setting properties of services. Depending on the requirements of VO management and the maturity of available tools, TrustCoM may use these specifications to perform management operations.

WS-Resource, WS-Management (including WS-Transfer and WS-Enumeration), WSDM or a minimal self-defined service interface will be used for management of web services and associated resources as well as implementing adaptation.

Semantic Web technologies and in particular OWL-S can be used for describing a number of ontological structures related to a VO. In addition to representing structures such as role hierarchies and trust domain OWL-S also permits to publish semantic information related to a web service as well as process models explaining the dependencies between message exchanges. Such models are particularly useful in the cases where mutually intelligible vocabularies of terms for data and process descriptions need to be established between the participants in a VO. However, the expressiveness of OWL-S for web service sharing and integration in a VO context has not been proven yet. In consequence, TrustCoM should focus in a first stage on defining and implementing a working infrastructure for the establishment, evolution and enactment of VOs assuming a single notation for the specifications. Provided this first step is addressed successfully, a generalisation to the use of general ontologies for describing terms and establishing mutually agreed vocabularies can be undertaken in a second phase.

Grid based systems exhibit a degree of similarity with Virtual Organisations and sometimes adopt the same terminology. In particular aspects such as service composition service discovery and simple aspects of SLAs have been implemented and demonstrated. However, Grid-based systems remain open environments focussing on the sharing of resources and distribution of computational tasks. Therefore, they do not address some of the more complex problems related to the recursive composition of VO structures or the use of business processes. Although, a number of security issues have been addressed they also do not cater for recursively-composed structures and do not account for different trust relationships between the participants of a VO structure. From an implementation point of view Grid-environments have been largely monolithic up to now. There are numerous dependencies between the various elements of the frameworks making it difficult to reuse particular tools or components in isolation from the rest. There is substantial expertise within the TrustCoM consortium on building Grid environments and this expertise will be used in the first phase to identify particular tools and techniques that could be re-used within the project.

Conclusions

- SOAP, WSDL, XSD, WS-Addressing and WS-ReliableMessaging should be used as baseline. Extended UDDI should be used for service discovery. WS-Notification and possibly WS-Eventing should be used for Monitoring and Event dissemination.
- WS-Resource, WS-Management (including WS-Transfer and WS-Enumeration) or a minimal self-defined service interface should be used for management of web services and associated resources as well as implementing adaptation
- A number of Grid-services and algorithms could be used, however they need to be extracted from the entire grid-framework and used in isolation.
- Although a number of ontology techniques and in particular OWL-S could be used, these would add an additional level of complexity to the project.

Recommendations

- The project needs to investigate and choose a set of tools and platforms as basis for development. If a particular standard is not supported by existing implementations compliance with that particular standard needs to be abandoned. Attempting to provide interoperable implementations for standards where they do not exist constitutes a major development effort beyond the effort available in the project.
- No more than two platforms need to be selected for implementation. Each platform should have a well-defined set of implementations of the WS-* standards on which it will base any further development.
- Identify if any Grid-based implementation elements can be isolated and reused.
- Use of semantic web technologies and in particular ontologies should be delayed until the basic infrastructure is implemented deployed and tested on a simplified version of the scenarios.

7 Trust Management

Trust management remains a significant area of research despite numerous attempts to address this issue. The fundamental paradox of trust management as a research area is that although there is wide spread agreement on the importance of using trust in a variety of contexts including business transactions and although each one of us has an intuitive belief for what/who we trust, there is little agreement on what trust *is* or how to characterise it. Indeed, the various trust management frameworks proposed in the literature differ significantly both in their definition as well as is their computation of trust. The following aspects are by and large agreed in the various studies on trust:

- Trust is intimately linked (or derived from) different elements such as: recommendation, reputation, risk, and evidence of behaviour.
- Trust is linked to a well identified context that includes the activities being performed, the parties engaged in the interaction as well as other contextual elements of the transactions. However, none of the solutions in existence address this adequately.
- Trust may be expressed in relation to different characteristics of the parties involved in a transaction or the activities being performed such as competence, and honesty of the parties, correctness of the execution of the transaction or its result.
- Trust is quantifiable as otherwise little use could be made of it. However, no consensus has been reached on the desired metrics for its quantification.

The various studies can be broadly divided into two categories, those that focus on trust aspects of a security infrastructure in particular with regards to the authentication of users or disclosure of information and general frameworks for trust management that focus on trust analysis, quantification and trust services. The former are relatively well understood in particular when relating to PKI infrastructures. In addition, there are also a number of emerging studies on trust negotiation i.e., the incremental disclosure of security relevant information such as credentials and requirement for access although further studies are needed in this area. The latter have also been subject of a number of studies but there is little consensus on how to define, manage and compute trust based on an infrastructure of trust services.

Conclusions

- Trust management models need to be developed in TrustCoM together with a supporting infrastructure. Considering the lack of consensus on trust management at the research level it is unlikely that in a VO setting the participants will adopt similar trust metrics or computational models.
- There is some initial work on trust negotiation in terms of incremental disclosure of credentials and requirements. This work can be extended and included in the TrustCoM framework.

- PKI and other security aspects of trust are sufficiently well understood to be used as examples in the over all trust management framework and evaluate a subset of its expressiveness.
- Trust plays a very important role when establishing business partnerships and collaboration and maintaining them over time. However more evidence has to be acquired to demonstrate how trust is incorporated both in the contract content and, foremost, in the operational business processes.

Recommendations

- The trust management infrastructure needs to be agnostic to specific trust metrics or computational models. In addition protocols for negotiation and exchange of trust information need to be developed.
- Security aspects of trust are well understood and should be treated like a specific case in the overall trust management framework rather than as a distinct issue. This will also enable us to test (to a certain extent) the expressiveness and use of the framework.
- Given the current state of the art, it seems reasonable to begin with the development of a trust management model and infrastructure. This should include: what trust services are provided at the VO level and how the information is aggregated from the participants, what specifications of trust metrics and computational models are needed to provide coherent trust information to the decision functions within the VO, how this information will be used as part of SLA management and negotiation, business processes and autonomic security enforcement.

8 Policies and Security

Security aspects of a VO framework span a large number of concerns that broadly divide in the following categories: Access Control, Authentication, Secure Connections, Information Disclosure and Adaptive Security. These will each be addressed in turn in the paragraphs below. Overall security and policy are not only a substantial part of TrustCoM but one where the consortium has considerable expertise.

Access Control Models are well understood within a single administrative domain and new concepts such as Role Based Access Control are increasingly appearing in main stream products. Authorisation policies are used in a number of different frameworks (Ponder, Permis, SPKI, etc) and standards (XACML). Despite apparent differences between the specification languages their functionality is broadly similar. Their enforcement is sometimes different, in particular when applied in distributed environments but the advantages and disadvantages of the various solutions are again well understood. However, distributed access control within environments that cross-domain boundaries remains fundamentally an open research problem. Grid environments have attempted to address these issues in a number of platforms (Akenti, VOMS, CAS, etc.) however the assumptions on which these models are based are too restrictive for VO enforcement. In particular, most grid-platforms are concerned with access control to resources by distributed tasks and do not allow for recursively composable VOs in federated structures (i.e., a Grid is not itself a VO that can participate in higher-level VOs). One common characteristic across all platforms is however the increased usage of arbitrary security tokens to convey relevant security information. As domain boundaries are crossed, local identity loses any meaning and access control decisions are made based on properties that the requestor proves he possesses. These properties may include its role, qualifications and other attributes as well as privileges he/she holds or that have been delegated to him/her. This evolution is also evidenced in the more recent web-service standards such as WS-Trust, SAML and WS-Federation. The latter, in particular, focuses on the exchange and use of such tokens across domain boundaries. WS-Trust and SAML overlap in scope.

Authentication, and in particular authentication based on identity, becomes then a particular case of the more general token based framework described above. Recent studies and standards have particularly focussed on Single Sign-On systems such as Liberty Alliance and Shibboleth. Both of these overlap in scope with WS-Security, WS-Trust, WS-Federation based standards but tend to be less flexible (e.g., lack of support for “active” requestors), focus on identity management alone and rely on SAML for communication of information and SSL as the underlying secure transport protocol.

The WS-* series of specifications provides solutions for message integrity and confidentiality that are specifically tailored the SOAP messages exchanged with web-servicies. These are WS-Security and WS-SecureConversation and have been designed to work in conjunction with the other specifications from the series.

Invariably, access control, authentication and secure connections rely upon the security services being appropriately configured for communication and inter-operation. This is achieved through policies, and thus WS-Policy, WS-SecurityPolicy, WS-PolicyAssertions and WS-PolicyAttachement have been introduced. These specifications aim to provide an interoperable format for policies which can then be embedded and exchanged as secure tokens. Although originally conceived for the local configuration of services the common aspects have been generalised and some advocate the use of these standards for any kind of policies.

The ability to control information dissemination and disclosure including providing restricted access to sub-parts of a document is a long standing research problem. Although expertise exists within the consortium to address these issues, this would entail using a disproportionate amount of effort. In would therefore be appropriate to delay these issues until later on in the project when a security infrastructure is in place.

As often in the past most of the existing work focuses on security mechanisms rather than on the processes required for managing them or adapting their behaviour. Although there are several specifications on how to define policies there is little work on which policies to apply in which circumstances or how policies are dynamically changed in response to changes in context or trust. This issue is of particular relevance in a VO environment characterised by varying trust relationships and where the members and structure of the VO may change. Consequently this is arguably the most important challenge that the TrustCoM security framework will need to address. Even the ability to tackle this challenge is only possible because of the integration with Business Processes, Contracts and Trust.

Conclusions

- WS-Policy, WS-SecurityPolicy, WS-Trust, WS-SecureConversation and WS-Federation form a coherent group of standards that is explicitly focussed on open web-service environments.
- However, several issues need to be addressed in order to combine them in a security infrastructure that satisfies the needs of VO Environments
- Although aimed at being general, WS-Policy and WS-SecurityPolicy remain attached to low level service configuration. Therefore XACML or similar policies may prove more useful ad higher levels of abstraction for use within a VO.
- Information disclosure and dissemination remains a significant issue but it is unlikely that the project will be able to address this, at least in the first phases of development.
- Realising adaptive security represents the most important research challenge. However, little work exists in this area that can be directly leveraged.

Recommendations

- The efforts on security and policy should proceed along two parallel but side-stepped tracks
    1. The definition and implementation of a suitable model for authentication, access control and secure connections within a VO environment
    2. The design and development of the adaptive security model. The starting point for investigation should in particular concern adaptation as a function of trust.
- There is a need to identify early in the projects the implementation available for the WS-* security standards, their degree of maturity and interoperability. The lack of adequate implementation toolkits is a significant risk which would require significant re-development of the model to compensate.
- Attempts to address other issues such as dissemination control should be delayed to the later phases of the project.

9 Legal Aspects

Legal issues play an important part of the TrustCoM framework for three reasons: first because the handling of information such as quantifiable measures of trustworthiness and reputation is information that has legal implications when shared across administrative boundaries, second because the support infrastructure for contracts and service level agreements should provide a legal standing in case of disputes and third because the legal identity and standing of the VO need to be clearly identified when the VO partakes into external contractual relationships either with clients or through participation in other VOs. There is a relatively limited amount of work in the legal domain on these issues; most of it originates from the IST Alive project. This includes a taxonomy of legal issues in VO environments, a reference VO life-cycle from a legal standing, studies regarding legal identity of the VO and studies of the contractual issues as well as definitions for model contracts.

The other area of related work concerns methods for legal analysis. A wide spectrum of techniques have been investigated comprising: conventional legal analysis, legal risk analysis including both conventional risk analysis as well as contract analysis, semi-formal conceptual analysis based on UML representations of the conceptual models and formal analysis, which includes the formalisation of deontic concepts such as obligations, policies, right, power and trust. Conventional legal analysis has an informal form which may be described as a legal argument. It is primarily an activity of the interpretation of norms in relation to given circumstances, typically by a judge deciding a particular legal problem in relation to a case. However, it is questionable if conventional legal methods alone are sufficient for the legal analysis in TrustCoM. Future VOs supported by the TrustCoM framework will be interested in understanding and solving legal problems in advance, rather than deciding a particular legal problem when a dispute occurs. The legal analysis in TrustCoM should therefore seek to combine conventional legal methods with other methods that support a proactive legal risk management.

Conclusions

- The work in the Alive project provides a solid foundation from which to start the analysis. The Alive project has among other results produced model contracts that can assist in addressing some of the typical legal risks related to the different phases of a VO lifecycle. However, this template needs to be adapted to the specific risks that a particular VO may face.
- The VO model contract provided by Alive is envisaged as a high-level text-based contract drafted and monitored by humans. In addition to this contract level, VOs may enter into more specific and operational electronic contracts and SLAs, where contract drafting and/or monitoring may be automated. For the time being, a fully-automated drafting and monitoring of the rather complex high level VO contract as provided in the Alive template does not seem feasible. However, this does not mean

that the TrustCoM framework can ignore the high-level contracts. TrustCoM should provide tools and methods to support the drafting and monitoring of text-based VO contracts. These tools and methods should focus on the management of legal risks related to the participation in a VO.

- The advantage of the TrustCoM legal team is that it comprises substantial expertise for performing traditional legal analysis as well as both semi-formal and formal analysis of VO issues. This will allow a multi-disciplinary research, where legal concepts are not only described in a textual form, but also expressed applying semi-formal and formal methods. These different disciplines need to be integrated in a way that ensures that the analysis results are relevant for the overall project. The most promising integration of these different methods is a model-based legal risk analysis based on both semi-formal and formal methods.

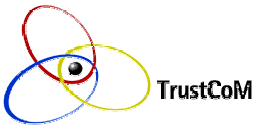
Recommendations

- The legal analysis should be based on the scenarios developed by the TrustCoM prototypes in order to ensure the relevance for the overall project. In particular, the TrustCoM scenarios should be used to uncover typical legal risks and to identify suitable treatments, with a particular focus on trust management, security management and contract management.
- The output that is expected from the legal risk analyses should consist of both textual and semi-formal representations of the identified legal risks with respect to the analysed scenarios as well as suitable treatments, with a particular focus on trust management, security management and contract management.
- The TrustCoM VO framework should support the drafting and monitoring of VO contracts at all levels, regardless whether or not they can be fully formalized. Hence, TrustCoM should also develop mechanisms (tools, methods and languages) for identification, assessment and treatment of legal risks to be dealt with in high-level VO contracts drafted and monitored by humans. These mechanisms should facilitate bridging of the communication barrier between experts within the legal domain and other relevant domains by applying graphical models of legal risks and treatments, which can be understood by professionals from different domains. These mechanisms should be seen as complementary to other TrustCoM methods and tools that primarily focus on the VO contract management level which can be fully formalized and automated.
- The graphical models for legal risk analysis should primarily be based on UML. Formal methods may be used to provide precise semantics for the graphical models.

10 Conclusions

Although a number of points have been presented in each of the thematic areas discussed above, together with conclusions and specific recommendations for each area, the most salient concerns deserve to be re-iterated here.

- **General project approach.** There is a large amount of literature across all thematic areas that has developed a substantial number of models, many of which aim to be exhaustive. Few of these models have been the subject of consensus in the research community, have been validated in any realistic fashion, or have been adopted by industry. Therefore, TrustCoM should strive to simplify its models wherever possible and address concerns from a practical and implementation-oriented point of view rather than purely from a modelling stance. It is worth re-iterating that TrustCoM is an industry-led integration project rather than a pure research endeavour.
- **Web-service standards specifications.** TrustCoM has made an early commitment to web-service standards. Although a reasonably comprehensive package of specifications exist, the availability of implementation toolkits for these standards as well as their maturity and interoperability is less clear. This is a significant risk because the TrustCoM consortium does not have sufficient resources to develop or debug implementations of the standards where they are missing. It is therefore recommended that based on the demonstrator environments a maximum of two implementation profiles be defined. These should comprise implementation toolkits for all of the WS-* standards that TrustCoM aims to use and interoperability between them should be demonstrated. Steps towards establishing this have already started within the project but this effort needs to be concluded when the first version of the architectural models is available.
- **Legal and socio-economic issues.** A major advantage of TrustCoM is the emphasis put on legal and socio-economic issues. As always however, there is a danger that these studies diverge in directions that are relevant but that cannot be supported by the TrustCoM framework. It is therefore important to periodically reassess the focus of these studies in order to maximise the input that they provide to both the conceptual modelling and the software implementation within the project. The studies of socio-economic and legal issues should have a clear focus on the scenarios developed and used in TrustCoM, in order to ensure that they address the business cases to be supported by the TrustCoM framework.



Deliverable

2

State of the Art Evaluation – phase1

WP10 State of the Art

Editor: Nilufer Tuptuk, Emil Lupu
Imperial College London

10th June 2004

Issue 1

TrustCoM

A trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations

SIXTH FRAMEWORK PROGRAMME

PRIORITY IST-2002-2.3.1.9

Prepared for the European Commission under FP6 Contract No. 01945 as a deliverable form

Deliverable datasheet

Project acronym: TrustCoM

Project full title: A *trust and Contract Management framework enabling secure collaborative business processing in on-demand created, self-managed, scalable, and highly dynamic Virtual Organisations*

Action Line:	**7**
Activity:	**7.1**
Work Package:	**WP10: State of the Art**
Task:	**7.1.1 State of the Art Evaluation**

Document title:	**State of the Art Evaluation – phase 1**
Version:	**Issue1**
Document reference:	
Official delivery date:	**10th June 2004**
Actual publication date:	**10th June 2004**
File name:	**SOA-D2-Issue1**
Type of document:	**Deliverable**
Nature:	**Public**

Authors:	Jon Bing, Gyrd Brændeland, David Chadwick, Joris Claessens, Theo Dimitrakos, Dave Golby, Jochen Haller, Andrew Jones, Claudia Keser, Mass Soldal Lund, Emil Lupu, Tobias Mahler, Lorenzo Martino, Brian Matthews, Xavier Parent, Christian Geuer-Pollmann, Babak Sadhighi, Jakka Sairamesh, Lutz Schubert, Ketil Stølen, Nilufer Tuptuk, Emily Weizenböck, Stefan Wesner, Konrad Wulf, Yücel Karabulut, Tomás Garcia Zaragoza
Reviewers:	Michael Wilson, CCLRC; Santi Ristol, AtosOrigin
Approved by:	

Version	Date Date	Sections Affected
Issue 1	10th June 04	First issue to be delivered to the commission.

LEGAL NOTICE

The following organisations are members of the Trustcom Consortium:

SchumbergerSema Sociedad Anonima Espanola,
Council of the Central Laboratory of the Research Councils,
BAE Systems,
British Telecommunications PLC,
Universitaet Stuttgart,
SAP AktienGesellschaft Systeme Anwendungen Produkte in der Datenverarbeitung,
Swedish Institute of Computer Scence AB,
Europaeisches Microsoft Innovations Center GMBH,
Eidgenoessische Technische Hoschschule Zuerich,
Imperial College of Science Technology and Medicine,
King's College London,
Universitetet I Oslo,
Stiftelsen for industriell og Teknisk Forskning ved Norges Tekniske Hoegskole,
Universita degli studi di Milano,
The University of Salford,
International Business Machines Belgium SA .

Neither the Trustcom Consortium, any member organisation nor any person acting on behalf of those organisations is responsible for the use that might be made of the following information.

The views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect the views of the European Commission or the member organisations of the Trustcom Consortium.

All information provided in this document is provided 'as-is' with all faults without warranty of any kind, either expressed or implied. This publication is for general guidance only. All reasonable care and skill has been used in the compilation of this document. Although the authors have attempted to provide accurate information in this document, the Trustcom Consortium assumes no responsibility for the accuracy of the information.

Information is subject to change without notice.

Mention of products or services from vendors is for information purposes only and constitutes neither an endorsement nor a recommendation.

Reproduction is authorised provided the source is acknowledged.

IBM, the IBM logo, ibm.com, Lotus and Lotus Notes are trademarks of International Business Machines Corporation in the United States, other countries or both.

Microsoft is a trademark of Microsoft Corporation in the United States, other countries or both.

SAP is a trademark of SAP AG in the United States, other countries or both.

'BT' and 'BTexact' are registered trademarks of British Telecommunications Plc. in the United Kingdom, other countries or both.

Other company, product and service names may be trademarks, or service marks of others. All third-party trademarks are hereby acknowledged.

Table of Content

Table of Figures

1 Introduction

TrustCoM aims to provide an integrated framework enabling secure collaborative business processing in on-demand created, self-managed, scaleable and highly dynamic Virtual Organisations. This objective can be achieved only through the integration of a large spectrum of different tools and techniques that cater for the creation and management of Virtual Organisations, collaborative business processes, contract management, service level agreements, trust management and security. Each one of these is a research area in its own right and has been the subject of numerous research efforts in both academia and industry. Furthermore, numerous and often competing standards aim to provide common ways of addressing some of the issues concerned. All these research areas are still active and the State of the Art is moving at rapid pace both in the research and in the standardisation arena. The TrustCoM objective is however based upon the realisation that these areas are both complementary and inter-dependent and it is only through their integration that the next generation environments for eBusiness can be built.

A State of the Art survey within this context is both a complex and delicate task. Complex, because it needs to address a large spectrum of different research areas within which different solutions have been proposed. Each one of these solutions addresses conceptual as well as implementation problems, and some aim to provide interoperability through standardisation. Delicate, because the evaluation of so many alternative solutions that are often interdependent and that have varying levels of tool support requires balanced judgement which often treads on a thin tradeoff line.

This State of the Art survey aims to be comprehensive yet due to the large amount of related work and the fast evolution of existing frameworks it cannot be complete. Nevertheless, it aims to achieve the following objectives:

- Provide an overview of the different conceptual frameworks and technologies, which could contribute towards the achievement of the TrustCoM objective. In particular, the survey aims to cover both existing standardisation efforts, existing tool support as well as exemplary approaches within each of the areas considered.
- Provide an evaluation of the existing work with regards to its applicability to the TrustCoM objective and identify work which TrustCoM can leverage to facilitate its development.
- Provide a common and comprehensive view of the State of the Art for the 16 partners within the TrustCoM project, which have substantially different backgrounds, in order to ensure a common base of knowledge and understanding.
- Provide a compendium of relevant work for external researchers who aim to work towards the development of services and applications for next generation eBusiness systems.

This survey is organised according to the thematic areas, which need to be developed within the project. These are:

Socio Economic issues – This section the business challenges that the TrustCoM framework aims to address as well as the role of *trust* in business relationships. In particular this section coves the socio-economic distinctions between the various trust concepts, the intimate link of trust to risk and reputation and the socio-economic techniques used in order to investigate the impact of reputation on business relationships and behaviour.

Frameworks for Virtual Organisations – This section covers the related work on the characterisation, taxonomy, deployment and infrastructure aspects of frameworks providing support for Virtual Organisations. In particular it presents a taxonomy of the different VO models based on their characteristics and examines in detail exemplary approaches of enterprise integration reference models providing support for virtual organisation concepts.

Contracts and Service Level Agreeements – This section presents the main conceptual models and frameworks for the specification of service level agreement, negotiation of SLA parameters and monitoring of compliance with the SLA. It then examines the different frameoworks providing support for the specification and negotiation of business contracts including DSTC's Business Contract Architecture, contractual resource sharing and ebXML's Trading Party Agreeements.

Collaborative Business Processes – The enactment of any Virtual Organisation aiming to fulfil business objectives relies on the composition and coordination of the services provided by the participants in the VO according to a well-defined process. This section provides a review of the business process definition languages, coordination protocols and infrastructure support for enacting these collaborations. As this area has been the subject of intensive standardisation efforts the section focuses on the various proposed standards in particular for Web Service platforms.

Enabling Technologies – Infrastructure support and in particular middleware plays a fundamental part in building the TrustCoM framework. It determines the implementation that can be leveraged as well as the interoperability level between the different aspects of the project. This section provides a review of the enabling technologies on which the TrustCoM framework can be built. Web Services technology as well as Grid environments are of particular importance both from a conceptual view point as well as for the implementation platform that they provide. This section also reviews other tools and platforms, which may be of use for business process integration and for security.

Trust Management – As the first section of this survey explains, trust is of vital importance in the development of highly dynamic and on-demand business collaborations. This section is devoted to the review of existing trust concepts, services, tools and techniques both for enabling business collaborations and as needed for the security infrastructure. In addition this section covers related work on trust negotiation and frameworks for trust management.

Policies and Security – The development of the next generation eBusiness environments that the TrustCoM framework aims to support, will become a reality only if the security concerns are adequately addressed and managed throughout the life-cycle of the highly-dynamic Virtual Organisations. This requires both a sound security infrastructure but also an adaptive security models and implementation that can react to changes in requirements or in the functioning of the Virtual Organisation. This latter requires a policy-driven approach to security management. This section reviews the state of the art in terms of policy and security frameworks for Virtual Organisations. The related work broadly falls within the following areas: new access control models and policy-driven approaches for access control, models combining access control and authentication for dealing with clients across administrative boundaries, web services security frameworks and standards, grid security frameworks and models and tools for adaptive security.

Legal Aspects – The creation of virtual organisations across administrative and country boundaries poses significant legal challenges. This section aims to review the main security concerns and the analysis frameworks for performing legal analysis, risk analysis and contract analysis for virtual organisations.

Some of the studies presented in this state of the art survey cross the boundaries defined above either within a particular thematic area or even across areas. At other times concepts have been repeated in order to provide a clearer understanding and context for the work discussed. This survey should provide not only a solid foundation upon which the TrustCoM project can arrive to fruition but also a tangible reminder of the diversity of aspects that need to be addressed during the project and the challenge that their integration represents.

2 Socio Economic Aspects

Edited by:Claudia Keser and Jakka Sairamesh
IBM

2.1 Introduction

Over the last decade we have witnessed a gradual transformation in the organizational structure and daily operations of Enterprises because of lower costs due to new innovations, better usage of the Internet, and the emergence pervasive computing technologies. Enterprises are leveraging the Internet's reach to establish better channels of communication and collaboration with partners, dealers, distributors, retailers and end consumers in the demand chain. However, there are still many process inefficiencies and not very secure channels of communication in the value chain. There are tremendous challenges for ensuring trust, security and privacy for transactions and operations over the Internet. Businesses need strong incentives and mechanisms to trust other businesses in open networks (see Chapter 4 and 5 on Business Collaboration). Monitoring business activities, contracts and ensuring trusted transactions and relationships are crucial for healthy business transactions over the Internet. In this section, we describe the State of the Art in relationship management, business models, economics of reputation and trust, and social network concepts for trust establishment amongst the diverse players in the value chain.

Businesses collaborate and communicate with other businesses for product development, purchasing, sales and services. Businesses are also depending on smaller businesses in the value chain to supply finished and completed product parts that can be easily assembled by the Enterprises at a low cost. With increasing complexity of products businesses have to collaborate with suppliers down stream and dealer upstream to manage and run their enterprises. In Figure 1, we illustrate a sample example of a value chain for the Automotive Industry. We chose this as a case study to illustrate that relationships have to be established between manufacturers and demand chain partners in order to enable better quality of service, better communication and better relationships with the end consumers.

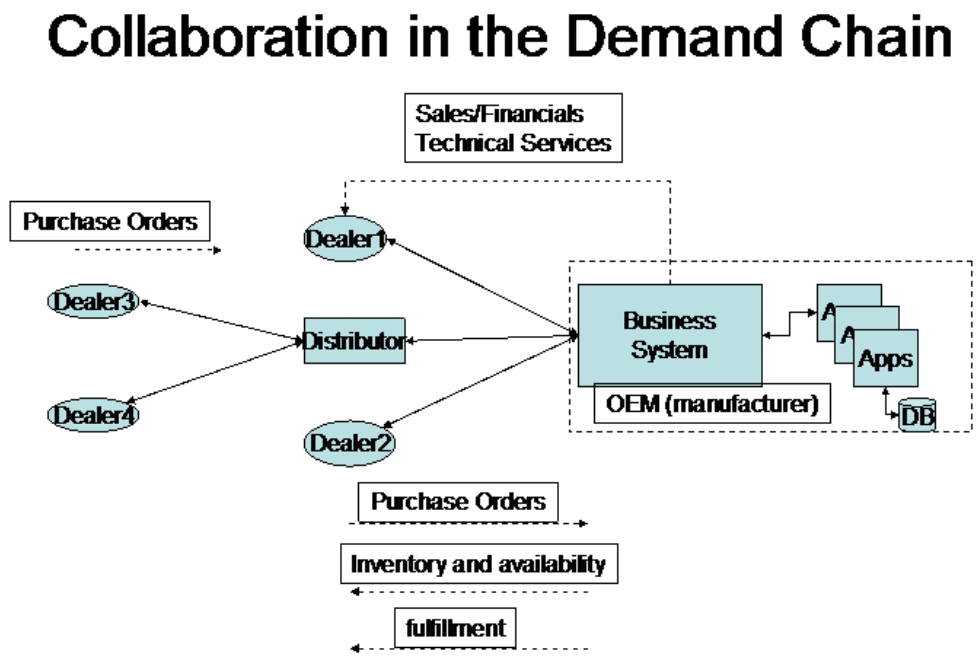


Figure 1 Trusted Collaboration and Relationships in the Demand Value Chain

In Figure 1, we illustrate a case study of the automotive aftermarket value chain with many diverse players involved in offering services once the vehicle is sold through the dealerships and distributors. Because of multiple tiers in the value chain, manufacturers do not have direct contact with consumers using the products (i.e., vehicles). Crucial information such as user preferences, product usage, failures, behavior, and symptoms that are normally shared between the consumers and dealerships is not shared with the actual manufacturer of the products. This lack of information results in poor feedback to the manufacturer.

For TrustCoM project to demonstrate value, business models for collaboration and economic models of trust and reputation are needed in order to provide a foundation for virtual organizations. Businesses participating in virtual organizations need to trust, rate and rank the partners in the business value chains. For collaborative business processes (chapters 4 and 5) to succeed in a virtual environment the appropriate business and technical challenges are needed.

Economic and business models also play a strong role in enabling standards such as WS-Agreements and WS-Policy (see section 8.16 Web Services Security and Policy) to evolve. For Virtual organizations (see Chapter 3 Frameworks for Virtual Organisations), which include manufacturers, partners and suppliers, sharing documents and crucial business information and processes in a virtual environment requires a clear understanding of the costs of sharing and models of enforcing contracts. Business and economic models provide the underlying foundations for enabling the quantification of the trust, enabling the key measures of trust.

2.1.1 Business Challenges

Networking the business processes: Enabling seamless business process management beyond the current Enterprise into the value chain is crucial in enabling the value chain to be efficient and competitive. The challenge is in formalizing and building flexible and semi-

automated business processes that extended into the value chain to provide the needed efficiencies.

Relationship and Trust Management*:* Establishing trusted relationships between value chain partners and offering service guarantees is becoming crucial for Enterprises to sustain and manage their value chains. Some of the core challenges includes in depth monitoring of partner transactions in the value-chain, audit trails, better accounting and contract enforcement for ensuring quality of service.

2.1.2 Technology Issues

Modeling and monitoring the business relationships: Enterprises need to understand the complex dynamics of the value chains that enable them to procure thousands of smaller products in order to build complex products. *Advanced monitoring technologies* are needed to capture information such as order transactions, inventory levels, real-time logistics, and so on. These monitoring capabilities will enable better escrow and dispute resolution mechanisms. The role of escrow companies to ensure that their system and the underlying process is robust and ensures that the transactions complete properly.

Complexity of collaborations: There is a lack of proper context-driven collaboration tools for addressing complex issues as they arise in the large value chains (e.g. automotive, industrials and others). These issues are often due to the complexity of the product, and the complexity of the symptoms that are exhibited by product failures. Close collaboration with domain subject experts is often required to achieve efficiency in problem resolution.

Audit Management: With the ever-growing Internet comes the issue of capturing the business activities and keeping an audit trail on each one of the transactions for future audits or future regulatory policies. Audit trails a very much needed for implementing new regulation in reporting by large Enterprises. In the next section we present economics of games for trusted computing and communication over the Internet.

2.2 Trust Economics

Trust may play a critical role in many business relations and, in particular, in the development of the Internet as a marketplace. Business-to-business collaborations, for example, require trust in our partners to behave ethically. A company that shares internal data such as sales reports, production schedules, product designs and logistical details with a supply-chain partner must trust the partner with that information.

Other examples where trust plays a role are informal online markets where individuals spread over the globe may buy and sell a wide variety of goods and services. Typically, single, isolated trades take place between anonymous counterparts, and there may be no opportunity for inspection of the item to be traded. Each of the trading parties might be tempted to cheat.

As a buyer of PEZ dispensers at eBay Inc., for example, I face some risk that the seller has not accurately described the condition of their PEZ dispensers, will not pack them properly for shipping, or will not deliver them in a timely fashion, if they will be delivered at all. If a seller chooses to deliver before receiving the payment, there are similar risks involved. To manage these risks several approaches have been proposed (see, for example, Kollock 1999[1], and Malaga 2001[2].) Third party escrow services could be used. They have the disadvantage, though, that they are time-consuming and costly. Frequent communication with the trading partner and insisting on the revelation of enough information to make the

[1] Kollock P., 1999, The production of trust in online markets, in: E. J. Lawler, M. Macy, S. Thyne, H. A. Walker (eds.), Advances in Group Processes (Vol. 16), Greenwich, CT: JAI Press.

[2] Malaga, R. A., 2001, Web-based reputation management systems: problems and suggested solutions, *Electronic Commerce Research* 1, 403-417.

trading partner identifiable could reduce some of the risks related with online trading. However, there seems to be little hope of actually tracking down a trading partner, given the opportunities to disguise identities using free e-mail services.

As a more powerful approach, many of the online market sites have developed reputation systems that allow each party in a transaction to leave feedback on the counter party's performance, which will be made available to all visitors of the site. Reputation systems play a double role. The first role is in enabling trade by making trade safer and increasing buyers' trust. Sending a check to a stranger requires a great deal of trust. The second role is in promoting satisfactory trade and increasing sellers' trustworthiness. Sellers should provide accurate descriptions of the items to be sold and ship in a timely fashion.

Online reputation systems are relatively recent and some of them have been subject to several modifications during their short existence. Up to now, there is nothing like a stand reputation system or a set of rules on how to design efficient reputation systems. This invites rigorous research on reputation management from various disciplines such as economics, marketing, sociology, psychology, computer sciences and law.

In Keser (2003)[3] we suggests the use of the experimental economics laboratory to examine certain design issues for reputation systems. We present a simple experimental design that allows us to measure the effect of reputation systems on trust and trustworthiness in a very controlled way. Based on this design, we compare the efficiency of two simple reputation systems that differ only in the amount of reputation history provided.

For TrustCoM, the models for reputation and models of trading can provide fundamental foundations for WS-Agreements, WS-Trust and WS-Policy. In addition, the experimental models from economics can play a strong role in designing mechanisms for business to collaborate and trust new partners in a virtual organization.

An Experimental Trust Measure

Our experiments are based on the *trust game* introduced by Berg, Dickhaut, and McCabe (1995)[4]. In this game, trust is measured by the amount that one of two players, the *investing player*, unilaterally invests by sending it to the other, the *trusted player*. The trusted player receives three times the amount invested and may then return some amount to the investing player. The amount he returns provides a measure of the trusted player's trustworthiness.

Our intuitive hypothesis, given the success of online markets such as eBay Inc., is that the introduction of a rating system into the trust game, in which the investing player rates the other player's trustworthiness, should increase both trust and trustworthiness. Note that for 2001 a Newsweek article reported almost 50 million of registered eBay users, 170 million of transactions, at a fraud rate of less than 1 percent of all transactions.

One of the two feedback systems that we examine manages short-run reputation while the other manages long-run reputation, and we are interested in whether the long-run reputation system is more effective than the short-run reputation system increasing trust and trustworthiness. Dingledine, Freedman and Molnar (2001)[5] for example observe a shortcoming of the full history, long-run reputation management mechanism used by eBay. They describe incidents in which sellers had built a reputation through a large number of low value transactions. They proceeded to offer a number of high value items, received payment for these items and then disappeared.

[3] Keser, C., 2000, Strategically planned behavior in public goods experiments, Working Paper, CIRANO, Scientific Series 2000s-35.

[4] Berg J., J. Dickhaut, and K. McCabe, 1995, Trust, reciprocity, and social history, *Games and Economic Behavior* 10, 122-142.

[5] Dingledine, R., M. J. Freedman, and D. Molnar, 2001, Accountability, in: A. Oram (ed.), *Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology*, O'Reilley & Associates.

We also investigate strategic aspects of rating and reputation building. Which fairness norms do people apply when they rate the cooperation of others? Do people care about their reputation? Does it financially pays to build up a positive reputation?

The results of empirical eBay studies suggest that buyers are willing to pay more for a good coming from a highly rated seller. Kalyanam and McIntyre (2001)[6], for example, find in their study on eBay auctions of Palm Pilot personal digital assistants that reputation has significantly positive returns. Houser and Wooders (2000)[7] also show that sellers in eBay auctions with a high reputation score receive higher bids than those with a lower score.

Note that experimental studies in the laboratory and empirical or experimental studies in the field are to be considered as important complements. In the experimental economics laboratory we have the advantage of being able to control the environment to a large extent at a relatively low cost. In our study we can, for example, directly compare the levels of trust and trustworthiness in an environment without any reputation management system to the levels of trust and trustworthiness in the same environment modified by the introduction of a specific rating system. Furthermore, we can compare the functionality of various rating systems.

Our experimental design, thus, involves three treatments, *a baseline and two rating treatments*. The baseline treatment is one in which participants repeatedly play the trust game, remaining in the role of either the investing player or the trusted player but interacting in each repetition with another unidentified participant. The latter implies that *strangers* interact with each other. La Porta et al.[8] (1997), for example, argue that trust is more essential to ensure cooperation between strangers than between partners who interact frequently and repeatedly. Among partners, reputation building and opportunities for future punishment could support cooperation even with low levels of trust. In our baseline treatment this kind of cooperation would be difficult to build up as reputation building and punishment could work only indirectly through an effect on the entire population. Our reputation treatments are similar to the baseline treatment: they also involve the interaction of strangers, but allow the investing player, at the end of each repetition, to rate the trustworthiness (cooperation) of the trusted player based on the amount returned. A player's trustworthiness may be rated as *positive*, *neutral* or *negative*. The trusted player is informed about his rating. In the *long-run reputation treatment*, the investing player is informed in the beginning of each repetition, before he makes his investment decision, about the most recent rating and the distribution of all previous ratings of his current trusted party. In the *short-run reputation treatment* he is informed of the most recent rating only.

2.3 The role of trust, reputation and related concepts

It is difficult to distinguish trust from related concepts, which on the surface resemble trust. Yamagishi and Yamagishi[9] (1994) argue that the most comprehensive definition of trust would be *taken-for-grantedness* of the reality, implying that trust is considered a psychological mechanism for reducing complexity in the environment (Luhmann 1988)[10]. However, trust is typically assigned another role: trust provides a solution to the problem caused by *social uncertainty*. Social uncertainty is defined to exist when I am incapable of

[6] Kalyanam, K., and S. McIntyre, 2001, Returns to reputation in online auction markets, mimeo, presented to IBM Consulting Group.

[7] Houser, D., and J. Wooders, 2000, Reputation in internet auctions: theory and evidence from eBay, University of Arizona.

[8] La Porta, R., F. Lopez-de-Silanes, A. Shleifer, and R. Vishny, 1997, Trust in large organizations, *American Economic Review* 87, 333-338.

[9] Yamagishi, T., and M. Yamagishi, 1994, Trust and Commitment in the United States and Japan, Motivation and Emotion 18, 129-166..

[10] Luhmann, N., 1988, Familiarity, confidence, and trust, in: D. Gambetta (ed.), Making and breaking cooperative relations, Oxford: Blackwell.

correctly determining the intentions of other persons who have an incentive to act against my own best interest. We will thus limit our attention to trust in other beings and organizations. Barber[11] (1983) distinguishes between two types of trust, *trust in another person's competence* and *trust in another person's goodwill*. The former is the expectation of technically competent role performance from those involved with us in social relationships and systems, while the latter is the expectation that partners in interaction will carry their duties in certain situations to place others' interests before their own. Yamagishi and Yamagishi suggest denoting the expectation of competency as *confidence*, and to define trust as the expectation of goodwill and benign intent. They further distinguish between trust and *assurance*, where they define assurance as the expectation of benign behavior for reasons other than goodwill of the other person. In other words, trust is based on the inference of the interaction of another person's traits and intentions, whereas assurance is based on the knowledge of the incentive structure surrounding the relationship. They give a nice example:

> *Suppose I have a special tie with the Mafia, and my trading partner knows this. I am certain that he will not cheat on me; he knows that if he does he will be quickly sent to a mortuary. My expectation of the partner's "honesty" is based on the fact that acting "honestly" is in his own interest, not on the belief that he is a benevolent person. Here, assurance exists but no trust.* (Yamagishi and Yamagishi 1994, p. 132)

Note that in the trust game by Berg, Dickhaut, and McCabe [4] the amount sent by the investing player yields a measure of trust in the goodwill of the other player. However, when we extend the trust game by the introduction of a reputation system, assurance will play some role. The investing player knows that, at least initially, the trusted player might want to build up a good reputation. Thus, we expect in the experiments with a reputation management system to observe higher investment levels than in the baseline experiments without such a mechanism. The difference in the trust levels of the experiments with a reputation management system and the baseline experiment may be considered a measure for assurance.

Yamagishi and Yamagishi [9] discuss commitment as another concept distinct from trust and assurance. To solve the problem of social uncertainty people form mutually committed relations. This reduces social uncertainty and thus the need for trust. In a repeated prisoners' dilemma situation, for example, it is possible to induce others to cooperate by the use of a tit-for-tat strategy (reciprocity) (see Axelrod 1984[12], Selten Mitzkewitz, and Uhlich 1997[13], Keser 2000[14]). Commitment plays an important role in repeated trust games with partners, such as in Cochard, Van Phu, and Willinger[15] (2000). Thus, the higher investment level in their experiment than in the previous one-shot experiments on the same game.

Reputation may play two different roles in social interactions involving trust. The first role is informational. It makes the recipient of positive reputation information trust more. Trust has been defined above as an expectation that (potential) partners have goodwill in their dealings with us. We do not have perfect information about their intentions, which we have to infer from available information, as for example their reputation. The second role of reputation is a kind of sanctioning. The attribution of a negative reputation may work as a sanctioning mechanism to punish dishonest behavior. This makes the owner of reputation act in a more trustworthy way. Thus, we expect in the experiments with a reputation

[11] Barber, B., 1983, The Logic and Limit of Trust, New Brunswick, NJ: Rutgers University Press.

[12] Axelrod, R., 1984, The Evolution of Cooperation, New York: Basic Books.

[13] Selten, R., M. Mitzkewitz, and G. Uhlich, 1997, Duopoly strategies programmed by experienced players, Econometrica 65, 517-555.

[14] Keser, C., 2000, Strategically planned behavior in public goods experiments, Working Paper, CIRANO, Scientific Series 2000s-35.

[15] Cochard, F., N. Van Phu, and M. Willinger, 2000, Trust and reciprocity in a repeated investment game, working paper.

management system to observe more trustworthiness than in the baseline experiments without such a mechanisms.

2.4 Conclusions

For virtual organizations to thrive, survive and succeed in open networks such as the Internet, a greater understanding of the business, social and economic principles of trust are needed. In the current economies worldwide, trust and reputation play a crucial role in enabling business to trade and handle movement of goods and services with minimal disruption or mistrust. For conducting trade over the Internet, similar trust environments and models are required for business to trust each other in virtual collaborative environments. For TrustCoM project, a very clear understanding of the business terms and conditions, the models of contracts, models of reputation, and incentives to conduct business with one or more partners is very much needed.

The action line (for the TrustCoM project) on social, economic and business are needed to build the foundations for trust and reputation mechanisms. In this chapter we presented the state-of-the-art in Business and Socio-economic aspects of Trust and reputation in business value chains. We described in detail some of the challenges for businesses to trust one another over the Internet, and the current research in understanding better the pay-offs in trusting one or more partners over the Internet. We expect that the socio-economic aspects will help influence the standards involved in defining trust mechanisms (e.g. WS-Trust and WS-Agreement).

3 Frameworks for Virtual Organisations

Edited by: Theo Dimitrakos
Council for the Central Laboratory of the Research (CCLRC)

3.1 Introduction

The purpose of this chapter is to analyse the main VO concepts and characteristic properties considered in TrustCoM and validate their relevance based on recent surveys of the main trends of VO.

During the conception and maturation of the TrustCoM integrated project initiative, delegates of the consortium have closely interacted with VO clusters such as VOSTER and European think-tanks and roadmap projects such as THINKcreative, VOmap and IDEAS, the project itself has chosen not to conduct at present a further survey in this area. Beyond and above resource limitation, it was judged that recent surveys such as [katzy1, katzy2, other] as well as the roadmap project deliverables have covered this area to a satisfactory level. Our focus has therefore shifted on combining and summarising these results, on the one hand, and elaborating the TrustCoM VO vision, on the other hand. In addition to providing a comprehensive reference for the Consortium and the community, our intentions is to use the results of the surveys and roadmap projects summarised in this chapter as one aspect of the base line against which the TrustCoM VO vision is to be compared and validated in the context of WP13: Scientific & Technological Roadmap.

This chapter is organised in four main sections:

1. An analysis of the basic VO and network concepts underpinning the TrustCoM VO vision;
2. An analysis of VO dynamics, including life-cycle models;
3. An analysis of VO structural properties;
4. A review of several VO modelling approaches in Europe.

A further collection of related VO concept definitions is provided in Annex II.

3.2 Concept Definitions

There is a plethora of partly converging definitions of virtual organisations and related concepts. Some of these definitions are summarised in ANNEX II: Summary of VO definitions and related concepts.

For the purposes of TrustCoM a Virtual Organisation is understood as a temporary or permanent coalition of geographically dispersed individuals, groups, organisational units or entire organisations that pool resources, capabilities and information to achieve common objectives. Virtual Organisations can provide services and thus participate as a single entity in the formation of further Virtual Organisations. This enables the creation of recursive structures with multiple layers of “virtual” value-added service providers. Frameworks that allow both structural and dynamic the modeling and analysis of VO's are important in TrustCoM. Roles, relationships, states, and process interactions are some of the key elements of the VO to be modelled with respect to Trust, Contract Management and Security.

The parties that form a virtual organization are typically part of a larger enterprise network of which a selection of partners is made. This phenomenon is known as “network activation” in

VO modelling theory (See for example 16,17 and 18). The entities in the Universe of such networks share some broad characteristics, e.g. belonging to the same economy or market sector, and their participation in the network indicates disposition to work together in a future market opportunity.

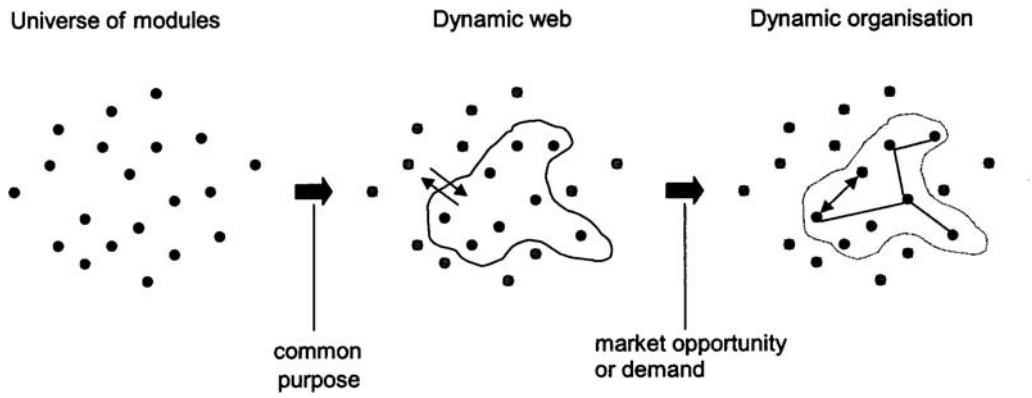


Figure 2 Network Activation [18]

Annex II of this document provides extracts from various publications regarding accepted VO definitions and concepts used for understanding both their structural properties and dynamic behaviour. Frameworks that allow both structural and dynamic modelling and analysis of VO's are important in TrustCoM. Roles, relationships, states, and process interactions are some of the key elements of the VO to be modelled with respect to Trust, Contract Management and Security.

3.3 Dynamic View

This analysis of VO dynamics focuses on the process and conditions of their formation, their dynamicity, and a description of their life-cycle model.

3.3.1.1 Targeted Virtual Organisations

Targeted VOs are formed and exist for a *purpose* and in response to a *market opportunity* or in order to fulfil a *market demand*. Consequently, VO focuses on a particular market segment or target group.

- The *purpose* of a VO is understood as the objective that provides the incentive for creating the VO and which holds the VO temporarily together[19].
- A *market demand* is understood as a want for specific products that are backed by an ability and willingness to some consumer to purchase them [20]. The specifications of the customer order including cost and quality requirements directly contribute to determining who is involved in the provision of the product or service.

[16] Wassenberg, A.F.P. (1995), Netwerken: organisatie en strategie, Amsterdam: Boom Meppel.

[17] Wildeman L, Alliances and networks: the next generation, International Journal of Technology Management, 15: 1/2, pp. 96-108 1998. [14]. Saabeel, W., Verduijn, T.M., Hagdorn, L., Kumar, K. (2002), A Model of Virtual Organisation: A Structure and Process Perspective, Electronic Journal of Organizational Virtualness, 4: 1. 2002

[18] A model of Virtual organisation: A structure and process perspective, W. Saabeel, T Verduijn, L Hagdorm and K Kumar, Electronic Journal of Organizational Virtualness, Vol4 No1 2002 ***24

[19] Shao, Y.P., Liao, S.Y. and Wang, H.Q. (1998), A model of virtual organisations, Journal of Information Science, 24: 5, pp. 305-312.

[20] Kotler, Philip. 1994. Marketing Management. Englewood Cliffs, NJ: Prentice Hall

– A *market opportunity* is understood as a (latent) area of need or requirement whose fulfilment generates profit. (A similar definition is described in [20]). The VO fulfil this need by bringing together the resources and competencies needed for designing, developing, producing and offering the appropriate products and/or services.

Targeted VOs are characterised by *goal-specificity* in the activities and interactions of their constituents and *deliberate cooperation* amongst them.

– *Goal-specificity* implies that activities and interactions within the VO are co-ordinated to achieve specified goals. Goals are specific to the extent that they are explicit, are clearly defined, and provide unambiguous criteria for selecting among alternative activities [21].

– *Deliberate co-operation* implies that the structure of relations is made explicit and can be "deliberately constructed and reconstructed". This requires that the rules governing behaviour are precisely and explicitly formulated and to the extent that roles and role relations are prescribed independently and commonly understood [21].

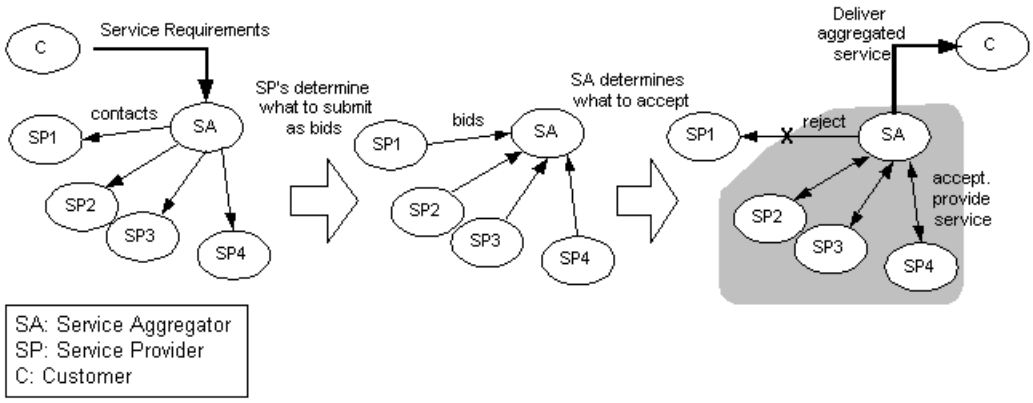


Figure 3 An example of Phases of federations of service providers

The Figure 3 describes the identification, formation and operation phases for the particular case of federations of service providers, which is related to one of the classes of scenarios considered in the context of TrustCoM.

3.3.1.2 Dynamic Virtual Organisations

Membership and structure of such a VO may evolve over time to accommodate changes in requirements or to adapt to new opportunities in the business environment. While the VO as a collective collaborates towards a common objective, parts of the VO capacity and capabilities are owned by different independent partners, which have their own (partly overlapping, partly conflicting) interests. When the goal of a partner is met or the partner feels its own objectives no longer align with the goal of the VO, it can step out of the VO. Partners collaborating in order to perform a task in a phase of a VO may leave and join a different, potentially competing VO within the life-time of the former. The ability to self-organise is a key attribute of cost-effectiveness for dynamic VOs. A specific kind of dynamic VO are those that have the capability to unite quickly in order to exploit an apparent opportunity or common goal. We refer to this as *"on-demand formation of a VO"*.

Dynamic VOs are particularly useful when faced with market incentives for responsiveness, dynamic service delivery, and charging based on usage. Beyond technological innovation, cultural adaptability in the organisation is essential for achieving an adequate degree of responsiveness.

[21] Scott, W.R. (1998), Organizations: Rational, Natural and Open Systems, New Jersey: Prentice-Hall.

3.3.1.3 VO life-cycle model

TrustCoM uses the following life-cycle model[22] as a reference, although we realise that not all of these phases would necessarily fit the specifics of all VO variants considered in the project.

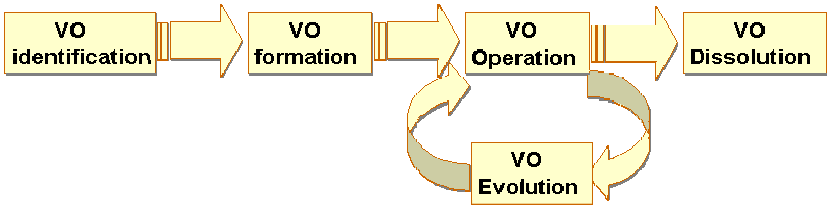


Figure 4 VO life-cycle model abstraction

- *VO Identification*. This phase involves opportunity identification, opportunity evaluation and selection [23].
- *VO Formation*. This phase involves partner identification, partner evaluation and selection, and partnership formation, including the binding of the selected candidate partners into the actual VO.
- *VO Operation & Evolution*. This phase is characterised by the controlled integration of the services and resources, offered by the VO partners in VO-wide collaborative processes leading to the achievement of shared business objectives. Membership and structure of VOs may evolve over time in response to changes of objectives or to adapt to new opportunities in the business environment. Changes of VO context may necessitate contract amendment or adaptation of policy and business process enactment.
- *VO Dissolution*. This phase is initiated when the market opportunity is fulfilled or has ceased to exist. The major decision processes in the termination phase include operation termination and asset dispersal.

3.4 Structural view

3.4.1.1 Self-management

Self-managed VOs are characterised by the ability to manage *at least* their operation and evolution without necessitating explicit intervention from the VO partners or other parties outside the VO. Self-management necessitates a form of integration that is enabled by the presence (or ability to form) an autonomic IOS that supports:

- The specification, negotiation and agreement of the conditions of involvement of VO participants by means of electronic contracts whose operation is supported and enforced by the IOS;
- Automation of membership management and trust establishment between the collaborating entities, be them the VO participants or the services and resources offered by the VO participants;
- Specification, negotiation and distribute organisation-wide policies which control the sharing and aggregation of services and resources in compliance to their agreement and

[22] Overall a similar life-cycle model has been used as a reference by the VOmap roadmap project "IST-2001-38379 Roadmap Design for Collaborative Virtual Organisations in Dynamic Business Ecosystems".

[23] T.J. Strader, F.-R. Lin, M.J. Shaw, *Information Infrastructure for electronic virtual organization management*, Decision Support Systems 23, 75-94 (1998).

are enforceable by the IOS, and allow adaptation (in real time) in response to changes of the context of interactions;

- Specification, distribution and autonomic enactment of business processes, which orchestrate the aggregation of information, services and resources in accordance to the consolidation of collaboration agreements and policies.
- The ability to resolve or adapt operation in response to conflicts between policies, agreements, and business processes both on the basis of their static description and their enactment.

The efficiency and effectiveness of self-management also depends on the extent that a VO achieves security and integration (elaborated in the sequel) and in particular organisational transparency, shared leadership, and separability in VO operation and management.

3.4.1.2 Scalability

Scalability refers to the ability of the VO framework to be realised in different scales depending on its objective and the kind of the parties involved (e.g. large corporations, SMEs, solitary entrepreneurs).

Scalability also allows the multiple interdependent layers of outsourcing as manifested by the constitution of recursive VO structures where similar VO formations may appear in micro- and macro- levels. This can be the case for example of a federation of providers offering high-performance end-to-end aggregate processes, where some activities are realised by aggregations of services provided by smaller-scale nested VOs and enacted over virtual execution environments which are also understood as VOs of execution hosts federating resources in order to enact a component.

Above all, the main criterion for VO scalability for TrustCoM is the ability of VO participants to gain value through collaboration within a VO by tackling collaborative projects that they *could not* undertake individually (either in absolute terms – i.e. feasibility – or in relative terms – in particular: performance and cost efficiency).

3.4.1.3 Security

This characterises the ability of the VO to comply with the confidentiality, integrity and availability requirements as expressed in mutually agreed security policies based on collaboration agreements (i.e. “contracts”). A balance must therefore be achieved between the necessity to share resources, capabilities, information and knowledge in order to improve VO performance, on the one hand, and to protect the assets and (potentially conflicting) interests of the participants, on the other hand.

3.4.1.4 Integration

This characterises VO structures which comply with the typology of *“dynamic networks”*[24] of Businesses and Governments, as explained above, while they also support at least the following *extreme aspects* of an organisational network, each which we examine in the sequel:

- *Organisational transparency*
- *Autonomic inter-organisational information system* (IOS)
- *Shared leadership*
- *Shared access to resources and shared capabilities*
- *Shared loyalty*

[24] Snow, C.; Miles, R.; Coleman, H. Managing 21st Century Network Organizations. Organizational Dynamics, Vol. 20, Winter 1992. Also Miles, R.E.; See also Snow, C.C. (1986): Organizations: New Concepts for New Forms. In: California Management Review 1986, Nr. 3, p. 62-73.

- *Mission overlap and co-destiny*
- *Separability*

To achieve the desired economical performance, such "integrated" VOs require a functional efficient corporative network. The dynamic VO network may be embedded in a larger network of corporations, from which certain members are recruited to deliver the required performances.

Organisational transparency of the VO: This means that although the VO supports frequent and fine-grained interactions with customers in order to facilitate mass-customisation, the cooperation of corporations participating in a VO may not be visible to customers.

Inter-organisational Information System of the VO: An effective inter-organisational information system requires the virtualisation of information and computation services and resources (at different levels of granularity) and their autonomic, "just-in-time" integration across the boundaries of the corporations participating in the VO in order to form an agile virtual information system and computational environment that becomes the basis for cross member co-operations; customised production and distribution of goods and services. The formation of such an IOS to accommodate the secure enactment of a collaborative business process place within the VO should ideally take place at the time of demand. Its operation should respect the agreements between VO partners, the policies of the services and resource providers (who may maintain overall governance of the assets the provide to the VO) and the service and resource distribution should be transparent to the consumer both within and outside the VO. (See also "organisational transparency" characteristic.)

Shared control of the VO: Shared control refers to the characteristic property that while every partner contributes to the operational management of the VO, it does not automatically controls the whole VO, although it effectively maintains high-level control within its own local administrative domain.

Shared leadership of the VO: Shared leadership refers to the characteristic property that while every VO partner maintains control of their own assets and serve their own interests, these *must* relate to and *may* be partly overlapping with, the interests of the collective.

Shared access to resources and capabilities within the VO: Shared access refers to the characteristic property of a VO that a member of a VO may access shared information or share capabilities located within the administrative domain of other members for the purposes of performing a collaborative task. Both service / resource owner and consumer need to be members of the VO and to perform in accordance with a mutual agreement, while the resource owners and hosts always maintain the high-level control of the resources offered to the VO.

Shared loyalty within a VO: This refers to the characteristic property that all network entities (employees, services and resources) within the local administrative domain of a VO partner identify themselves with the VO but *also* with their own organisation.

Partial mission overlap and co-destiny of VO partners: This refers to the characteristic property that there may be partners that are also doing business outside of the context of the VO (and have a partial mission overlap) in addition to those (having a complete mission overlap) that all business is conducted within the VO context. Notably, partial mission also allows for partners to do business within different and potentially competing VO, as long as the partial mission overlap of such partners is consistent and in accordance to the corresponding collaboration agreements. In either case, the semi-stable relations (less formal and less permanent than in a physical organisation), the sharing or resources, capabilities and knowledge and the shared risks make the VO partners also more dependent on each other therefore necessitating collaborations based on a sufficient level of trust.

Separability in VO operation and management: This refers to the characteristic property of achieving the following separation of concerns:

- Distinguishing the abstract requirements from the concrete implementation in order to reach the organisational goals.

- Distinguishing VO-wide *"global" strategic management level from "local"* operational management level.

For example, a VO wide collaborative process may include activities, which may be realised in different ways by different VO partners – such activities need not be concerned with the specifics of such realisation and the choice of realisation should not affect reaching the process objective. Analogously an Agreement or VO policy statement may be deployed and enforced by different means and in different administrative domains; policy deployment and enforcement should be separable from policy specification, and enforcement of the same policy statement should have the same observable effect to VO collaborators, irrespectively of how its has been realised.

3.5 Overview of research in VO modelling frameworks

According to [25] there are three environmental factors that have had the most decisive influence to encourage cooperation among organizations:

- Economic globalization: The world economy at the start of the twenty-first century is experiencing one of its moments of greatest dynamism and change. This dynamism is reflected in the growing interdependence of markets for goods, services and factors of production.
- Business uncertainty. The speed under which changes are occurring in the economic world is introducing large uncertainty. This is specially the case in business areas where constant transformations, resulting from reductions in technological and product life-cycles, improvement in productive processes, and so on, which are often difficult to predict, demanding greater follow-up capacity from enterprises in order to adapt to the new surrounding conditions.
- High level of competitive rivalry: The increased customer requirements and market saturation are forcing the enterprises to constantly dig deeper in their search for competitive advantages to improve their position in the market. As a result of this, there is a tendency for enterprises to concentrate on core know-how, or on those aspects of the added value chain they really master.

There are certain assumptions made in traditional organizational modelling, especially with regards to interoperability, that do not hold in the case of VO modelling e.g. same infrastructure, standards, environment, networking reliability, meaning of roles. Data and process protection policies may cause conflicts - a balance between sensitive organizational knowledge and views on shared processes need to be agreed upon and hence included in the models. During the last decade, there has been significant research activity in Europe in the area of Virtual Organizations[26]. From the EC funded activities more than 80 projects can be identified, which are complemented by a large number of national initiatives as well as 20 projects focusing on the development of Grid Computing infrastructures for VOs. However, as noted in [27] these initiatives correspond to fragmented research and in most cases due to the funding criteria, target very short-term objectives, focused on solving a specific problem, and too biased by "fashionable" short-life technologies. In summary, the situation regarding these projects is characterized as follows:

- Research on VO has created a critical mass and the European-wide intuitive understanding of the area.

[25] N.A. Penã, J.C.F. Arroyabe – Business Cooperation, Palgrave Macmillan, 2002

[26] Bernhard Katzy1, Gordon Sung, "State-of-the-Art of Virtual Organisation Modelling". In Proc. Of eChallenges Conference. e2003

[27] Luis M. Camarinha-Matos, Hamideh Afsarmanesh: A Roadmap For Strategic Research On Virtual Organizations. IFIP PRO-VE 2003: 33-46. Kluwer 2003.

- Required basic supporting infrastructures and relevant technologies are well identified, but the developments are often focused on particular needs and are based on ad-hoc experiments, hardly re-utilizable.
- Generic functions or harmonization of achievements are addressed only in few projects.
- Efforts on general plug-and-play architecture and interoperability are to a large extent missing. Consequently, no generally accepted reference model or interoperability base is available.
- Although several disciplines are concerned, the main focus has been on the ICT infrastructure. Research on social/organizational, including management, is mainly focused on best practice. Integration with technological development and impacts on organizational structures are not covered. In addition little research is focused on the social and organizational issues created by VOs.

Nevertheless there is a growing awareness that the VO developments should be based on contributions of a multidisciplinary nature, namely from the information and communication technologies, socio-economic, operations research, organizational, business management, legal, social security, and ethical areas, among others.

One of the main weaknesses in this area is the lack of widely acceptable rigorous modelling methods and theories to define the collaborative, networked organizations paradigm. Further to this there is no commonly accepted definition for basic concepts such as virtual organization or virtual enterprise (See Annex II). Camarinha-Matos and Abreu summarise in [28] the situation as follows:

- The lack of rigorous and well-founded methods for collaborative networks, collaborative decision-making and collaborative behaviour modelling. The focus on short term results and lack of recognition of VO modelling as a scientific discipline in its own right have had a negative effect in effectiveness of results.
- There are few VO reference models that have achieved wide acceptance.
- Most available modelling methods and tools were developed with single (potentially distributed enterprises) in mind and are not suitable for VO.
- Methods for integrating different models that offer partial solutions are missing.
- Existing approaches to model the social and human aspects in collaborative networks (soft modelling) are of poor quality.
- There is little support for dynamic Ontology creation and maintenance in a networked environment.

[28] Towards a foundation for virtual organizations, L.M. Camarinha-Matos, A. Abreu, in Proceedings of Business Excellence 2003 – 1st Int. Conference on Performance measures, Benchmarking, and Best Practices in New Economy, Guimarães, Portugal, 10-13 Jun 2003.

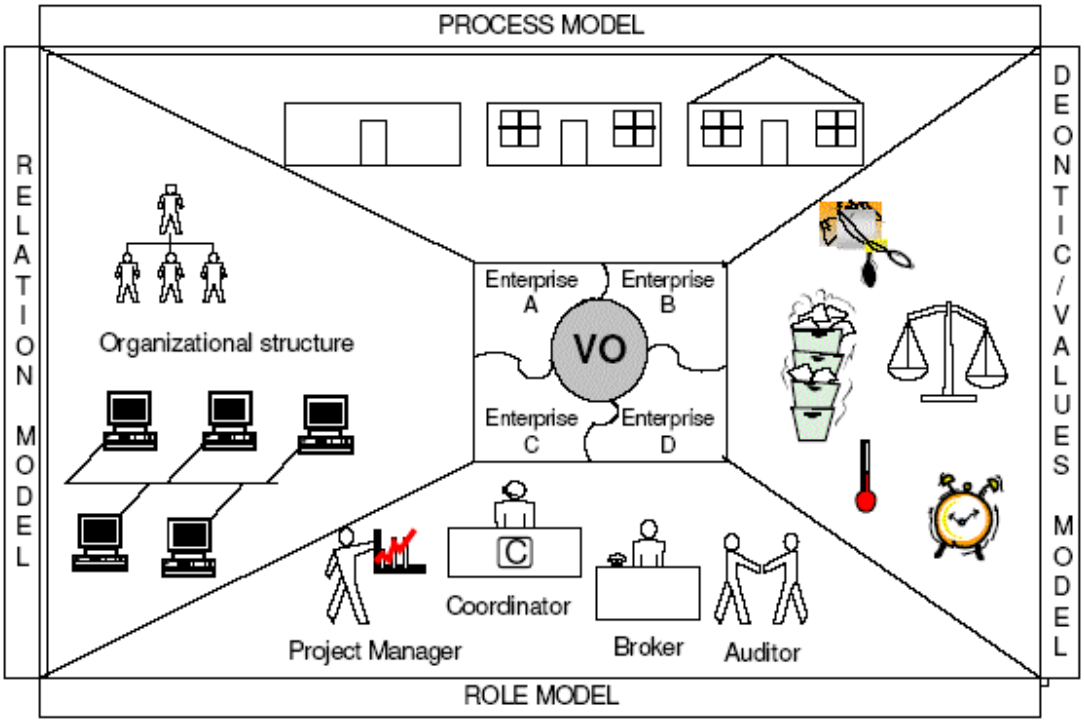


Figure 5 A visualisation of four complementary modelling viewpoints of a VO from [28]

The general approach considered in the context of the IST THINKcreative (www.thinkcreative.org) and VOmap (www.vomap.org) projects[29] has been to organise relevant modelling techniques in relation to how the could contribute to the following aspects of VO modelling. The figure above offers a graphical visualisation of such a separation of concerns in VO modelling.

Relationships models that describe the forms of interrelationship that can occur between components within a network. For instance, the following types of relationships can be identified: *control relationships* (which identify the authority structure within a network.), *dependence relationships* (which identify the topologic dependences between agents), *ownership relationships* (which define the boundaries of each agent), *peer relationships* (which identify agents at the same level).

Roles models that describe all roles and their positioning within the network structure. A role model implicitly defines a topology of interactions.

Process models that focus on dynamic courses of events. Some generic concepts such as activity and actor, time dependencies such as equal, during, starts, finishes, and resource-related perspectives such as necessary, sufficient, have to exist.

Deontic / Values Models that define constraints for all agents within a network at different levels, such as: *economic level* (where one may place constraints about cost, utility, etc), *organizational level* (where behaviour constraints relating to one's organisational role are placed), *operational level* (where one may place constraints about the order and interdependencies of actions that need to be performed in order to meet an objective. E.g.

[29] THINKcreative aimed at identifying and characterizing emerging organizational collaborative forms and their required infrastructures, modelling and application tools, as well as the corresponding socio-organizational needs for the next 5, 10 and 20 years. VOmap was a roadmapping initiative that aims at identifying and characterizing the key research challenges, required constituency, and implementation model for a comprehensive initiative to affirm the European leadership on dynamic collaborative virtual organizations.

relative ordering of tasks in a processes enactment.), *computational level*, where constraints about resources sharing, service and communication interoperability are placed.

Table 1 summarises and analysis of various modelling techniques that could be used for modelling aspects of VOs, and Table 2 relates them to the above viewpoint model classification.

Theories	Short Description	Applicability	Limitations/Challenges
Game Theory	A mathematical framework designed for analyzing the interaction between several agents whose decisions affect each other. An interactive situation is described as a *game* that has an abstract description of the players (agents), the courses of actions available to them, and their preferences over the possible outcomes. It is assumed that players employ rational decision-making, that is, each player's objective is to maximize the expected value of his own payoff, which measured in some utility scale.	Non cooperative game theory: good for selecting partners, sustaining cooperation and trust Cooperative game theory: distribution of responsibility and resources.	Need to identify all "players" Need to know all possible "moves" and associated results Assumes that player's behaviour does not change once the game starts It is difficult to capture subjective relationships.
Complex systems theories	A complex system is formed of critically interacting components (that has rich information neither static nor chaotic) that self-organize to form potentially evolving (environmental variation selects and mutates attractors) structures exhibiting a hierarchy (multiple levels of structure and responses appear -hyper-structure) of emergent system properties (new features).	Analysis of self-organizing behavior, Learn how to manage chaotic dynamics, The "small words" case can give some insights on understanding VOs	Models studied for other domains (e.g. biology, physics) cannot be easily applied in VOs, Further work is necessary for complex systems with social actors (human elements)
Graph theory	A branch of mathematics concerned about how networks can be encoded and their properties measured. The main goal is to represent a network in symbolic terms, abstracting reality as a set of linked nodes.	Represent networks of relationships -topology, routing, activity, flow Perform computations on flows Optimization	Basic theory is very rigid - needs extensions to represent non tangible, qualitative relationships (fuzzy dimensions), and multi-criteria
Social Actors Networks theory	Extension of graph theory to include relationships between social actors. Social Actors Networks are a way to highlight the structural relationships among social actors, enabling the conceptualization of their actions in a systematic way.	Analysis of social and organizational structure of VOs (connectiveness, trust, awareness, etc.) Creation / reconfiguration phases of collaborative networked organizations.	Offers a structural view on the VOs but does not provide any means to model an agency point of view i. e., what, how and why activities are performed in a VO. Needs to be used with a business process engineering method.

Multi-Agent Systems	A multi-agent system is a loosely coupled network of problem-solver entities that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity. Hence, it is concerned with coordinating intelligent behavior among a collection of autonomous intelligent agents, how they can jointly coordinate their knowledge, goals, skills and plans to take action or to solve problems.	Model societies of autonomous, heterogeneous, evolving entities Coalition formation and negotiation Simulate self-organizing behavior	Need further developments in social aspects of agency, dealing with uncertainty and interoperation with other models (e.g. process modeling languages)
Semiotics & Deontic logic	Semiotics is a science of signs and/or sign systems. Deontic logic is a logic of obligations. Together they provide a normative approach for systems modeling.	Model responsibility relationships and commitments Prescribe norms and roles, legal support Capture system requirements	Reasoning in deontic logic may lead to paradoxes; difficult to automate reasoning An promising approach: integration with agent logic
Formal Engineering Methods	A way to incorporate Formal Methods into the software development process to enhance the rigor (methodology), comprehensibility (human), and tool supportability (software tools) of software development process and consequently the quality of the final software product.	Describe the operational behavior of VO Formulating operational plans that bind partners in VO Verifying formally that the plans are indeed satisfied by the operational behavior	Difficult to develop and understand Economic and social aspects cannot be represented. Need intermediate-level formalisms and methods to bridge the gap between abstract formalism and practical ICT implementation
Formal Theory	Formal theories are based in mathematical tools (like: logic, set theory, algebra, etc) that is used for describing system properties, and for producing systems that satisfy those properties.	Solve design problems: architecture, protocols, network creation-Specify systems, verify specifications according to correctness and completeness Test and verification implementations versus specifications or standards	Lack of knowledge of formal methods in the current engineering community Need practical approaches for scalability Need to consider developments in communication networks
Metaphor Theory	Metaphors are an integral part of our society and language (informal or semi-formal language that can use graphic description like bubbles, arrows, charts, matrices) which makes it a form of communicating that is deeply ingrained and understood intuitively by Western cultures. That is one of the most important tools for trying to comprehend partially what cannot be comprehended totally.	Quick description for human communication (a possible help in expressing complex ill-defined concepts) Use in early stages (conceptual design)	Risk of taking metaphors too strictly Needs further evaluation and research in consistent understanding in the creation and interpretation of metaphors Needs to be combined with formal methods

Operations Research	A mathematical approach to decision-making, which seeks to determine how to best design and operate a system (optimization process), usually under conditions requiring the allocation of scarce resources.	Can be used mainly at operational level to support: Network creation, Network Optimization, Operations and production management, Logistic management	High level of abstraction and notation, Difficult communication, Heavy and computationally demanding tools.
Ontology	Ontology is a formal explicit description of concepts in a domain of discourse (classes, sometimes called concepts), properties of each concept describing various features and attributes of the concept (slots, sometimes called roles or properties), and restrictions on slots (facets, sometimes called role restrictions). Ontology together with a set of individual instances of classes constitutes a knowledge base.	Creation / Design Communication Understanding basic principles	Only captures static domain knowledge (not dynamic knowledge) Several conceptualization possibilities even for a given well-know domain, Building them is still an *art*.

Table 1 An assessment of some possible research directions in VO modelling based on VOmap roamdap project[30]

	Relation Model	Roles Model	Process Model	Deontic/values Model
Informal view	Metaphors theory	Metaphors theory	Metaphors theory	Metaphors theory
Semi-formal view	Graph Theory Social Actors Network Theory Mu lti-Agent Systems	Graph Theory Social Actors Network Theory Multi-Agent Systems	Graph Theory Multi-Agent Systems Ontologies	Graph Theory Social Actors Network Theory Multi-Agent Systems
Formal view	Game Theory Complex Systems Theory		Game Theory Formal Theory Operations Research Formal Engineering Methods	Game Theory

Table 2 A classification of the research directions assessed in Table 1 against the modelling perspectives summarised in Table 1.

[30] www.vomap.org

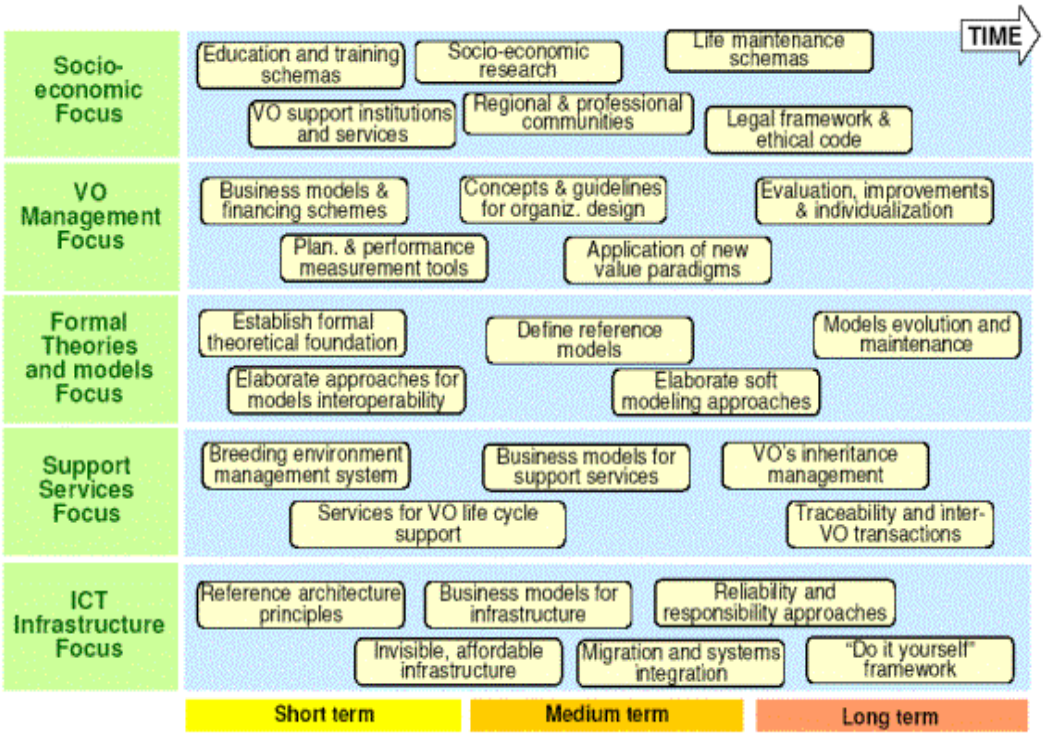


Table 3 A roadmap for the future of VO research based on the results of the VO map project[30]

There are certain goals within the VOmap that coincide with those of TrustCoM, even though with broader aims implied.

Selected VOMap Roadmap	TrustCoM Expected Contributions
VO Support institutions and services	Trust, Contract and Security management for and by VO support institutions
Establish formal theoretical foundations	Conceptual models for trust, contract management and security
Breeding Environment Management System	Enhance management of Collaboration Agreements
Reference Architecture Principles	Principles for trust, contract management and security
Socio Economic Research	Influences on trust, contract management and security
Concepts and guidelines for VO design	Actors, roles, services and patterns concerned with trust, contract management and security
Define reference models	(Similar to above)
Legal framework and ethical code	Regards VO agreements between partners as well as agreements with third parties and customers

Traceability and inter-VO transactions	Methods for managing business processes in recursive VOs
"Do it yourself" framework	Methods and tools for trust, contract and security management

Table 4 Congruence of VOMap Roadmap and TrustCoM contributions, going from short-term to long term

3.6 A classification of VO reference models

There are numerous existing enterprise modeling approaches and definitions for reference models that support the full range of needs from strategic business management to organizational design, Enterprise Software implementation and software development.

The following four are potentially useful reference examples, and good representatives of their kind, also visualised in Figure 6:

1. The St. Gallen Management Model[31] as a general management model
2. Value System Designer as an example of a model focusing on organizational process development
3. GERAM and related initiatives (e.g. VERAM, CIMOSA) focusing on enterprise integration and system design/enactment.
4. Rosettanet, as an example of information system integration [32] and [33].

Katzy and Sung in [34] investigate a framework of enterprise modelling approaches from literature review and present survey findings of the VOSTER Project on how current projects on VO use existing modelling approaches and in which directions those approaches are extended. The survey covers thirty (30) EU funded VO projects and ten (10) VO projects funded from other sources. Their framework is summarised in Figure 6, and results of their survey are presented in Table 5.

[31] Ulrich, H. and W. Krieg (1974). St. Gallen Management Modell, Bern, Haupt.

[32] Kosanke, K. and F. Vernadat (1999). CIMOSA: enterprise engineering and integration.

[33] Chen, D. and B. Vallespir (2002). Developing an Unified Enterprise Modeling Language (UEML) - Requirements and Roadmap, L. Camarinha-Matos, Kluwer Publisher.

[34] Bernhard Katzy1, Gordon Sung, “State-of-the-Art of Virtual Organisation Modelling”. In Proc. Of eChallenges Conference. e2003

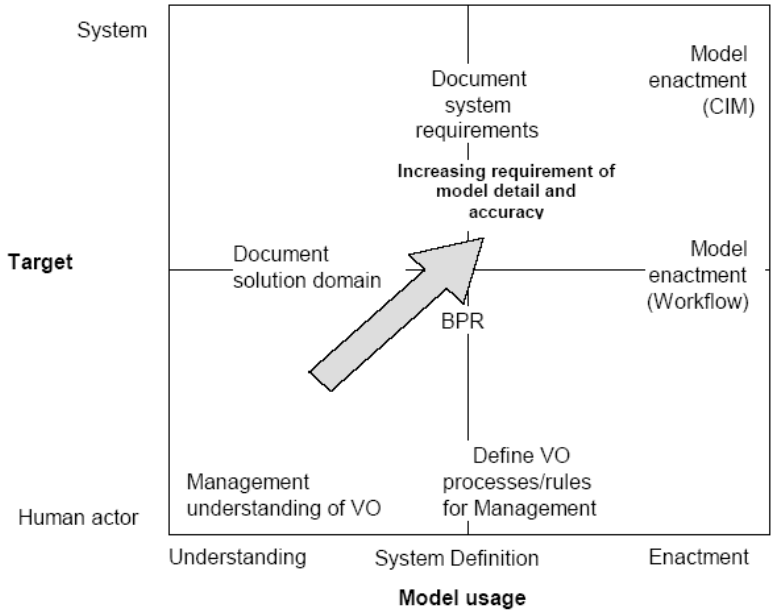


Figure 6 A methodology for categorising VO modelling approaches from [26]

The above, Figure 7 from [26] positions different purposes for modelling derived from literature findings in a matrix with one dimension being the target user (human actors use the model versus computer systems) and the other domain being the type of usage (understanding the enterprise versus enacting it). The lower and further left the objective for the modelling is positioned, the simpler and easier the models were found to be[35]. On the other hand, the further the objective is placed in the upper and right corner, the more detailed and accurate the models need to be. Management models for example typically are empty structures which need to be filled in by managers during their analysis [36, 37], while XML models of payment procedures are highly detailed and executable [38].

[35] Klaus-Dieter Thoben: System Design Principles in Customer-Driven Manufacturing. APMS 2002: 485-497. (Collaborative Systems for Production Management, IFIP TC5/WG5.7 Eighth International Conference on Advances in Production Management Systems, September 8-13, 2002, Eindhoven, The Netherlands)

[36] Van Schoubroeck, C., Cousy, H., Windey, B., Droshout, D. (2001), A Legal Taxonomy on Virtual Enterprises, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 357-364.

[37] Weitzenboeck, E. M. (2001), Building a Legal Framework for a Virtual Organisation in the Maritime Domain: the MARVIN Experience, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 337-346.

[38] Seilonen, I., Nurmilaakso, J.-M., Jakobsson, S., Kettunen, J., Kuhakoski, K. (2001), Experiences form the Development of an XML/XSLT-based Integration Server for a Virtual Enterprise Type Co-operation, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 321-328.

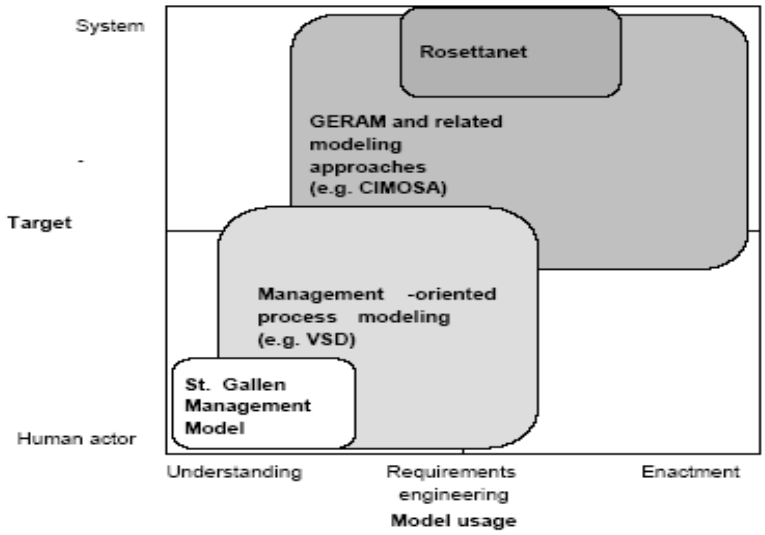


Figure 7 Summary of an analysis of some representative Enterprise Integration reference models from [34]

Enterprise engineering has developed approaches that combine these modelling extremes over the enterprise system life cycle[39]. However, surveys such as [34] show that the state of understanding and modelling virtual enterprises has not yet re ached that integration[40]. Only a small percentage of projects went all the way towards capturing the management level to detailed system design in their modelling effort [41,42,43]. This however hampers a review and good understanding of the resulting systems and tools by the users. The status of virtual enterprise research can be characterized as having achieved workable solutions on the level of well-defined domains or modules, e.g. modules for workflow[44] planning[45] and control in the IT domain[46], or management models of broker functions. This research, however, does

[39] Jochem, R. (2002), Enterprise Modelling and Enterprise Engineering - The Basis for Successful e-Business, in: Pawar, K., Weber, F., Thoben, K. (eds.), Proceedings of the The 8th International Conference on Concurrent Enterprising, Rome, Italy, University of Nottingham, 127-131.

[40] Schmidt, T., Rabe, L. (2001), A System to Realise Dynamic Networkd Organisations on Heterogenous Networks in the Consultancy/Agency Sector, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 163-170.

[41] Delebarre, C., Schönewolf, W., Yalniz, Z. (2002), Development of a Method for the Implementation of CE Tools in SME Manufacturing Networks, in: Pawar, K., Weber, F., Thoben, K. (eds.), Proceedings of the The 8th Inter-national Conference on Concurrent Enterprising, Rome, Italy, University of Nottingham, 87-90.

[42] Katzy, B. R., Sung, G. (2001), Information Infrastructure for Virtual Projects Requirements Specification from a Communication Perspective, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th Inter-national Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 187-191. Kosanke, K., F. Vernadat, et al. (1999). “CIMOSA: enterprise engineering and integration.” Computers in Industry(40): 83-97.

[43] Lillehagen, F., Korgstie, J., Jorgensen, H., Hildrum, J. (2002), Active Knowledge Models for Supporting eWork and eBusiness, in: Pawar, K., Weber, F., Thoben, K. (eds.), Proceedings of the The 8th International Conference on Concurrent Enterprising, Rome, Italy, University of Nottingham, 183-190.

[44] Ferreira, D., Ferreira, J. J. P. (2001), Designing Workflow-Enabled Business-to-Business Infrastructures, in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 81-90.

[45] Filies, O., Kujiken, A., Ridderand, L. d., Rodriguez, B. (2001), B2B and B2C in Mask Making (AutoMOPS), in: Thoben, K.-D., Weber, F., Pawar, K. S. (eds.), Proceedings of the 7th International Conference on Concurrent Enterprising, Bermen, Germany, University of Nottingham, 99-108.

[46] Chung, M. J., Kim, S., Han, H. S. (2002), A Process Management Framework for Collaborative Design and Manufacturing, in: Pawar, K., Weber, F., Thoben, K. (eds.), Proceedings of the The 8th International Confer-ence on Concurrent Enterprising, Rome, Italy, University of Nottingham, 211-214.

not yet provide consistent life cycle spanning support, which enterprise engineering has achieved for traditional hierarchies.

Chen et al.[47] also observe that modelling approaches can be distinguished by:

1. The distinct theoretical basis of computer science, systems theory, production management, organizational management, or knowledge management on which they are built.
2. The application areas that they address, be it information system design, business process reengineering, or computer systems design.

According to recent surveys such as [34] (which covered thirty EU funded VO projects and ten VO projects funded from other sources), about 70% of the projects assume the existence of a stable source network from which short-term cooperation in a VO emerges, but only few actually model it. Yet, there are clearly observable pattern of three network topologies, as a chain or supply-chain configuration, a hub & spoke or star topology, for example in construction industry, and third, a peer to peer network, for example in engineering networks.

By comparing supply chain topology with star consortia topology networks, we can realize that the former source network typically is more stable and has an inherent long-term orientation. Among peer-to-peer topology networks, there are some, but existing for only short periods, while others can exist long term, or be even permanent.

Most projects are focused on the operation phase of VO, sometimes also including the VO creation, but pay little attention to issues of the source network. Therefore, few projects make any proposition for the degree of change in the source network configuration over time. In terms VO composition during the operation phase, changes happened in 60% of the peer-to-peer and star/consortia types of reported VOs. Factors such as: size of the project, nature of VO activities, complexity of the process or product were indicated to influence the changes. While supply chain type of VOs, don't allow any changes during any single project or order. For the accountability of dynamic configuration and adoption, the majority of the reported projects used the IT platform and/or specially defined management roles like network -coordinator or project- manager.

All projects presume, that firms can be members of several VO projects at the same time. Thus multi-project involvement could be considered as one of the typical characteristics of VOs. About 75 % of the projects also suggest that firms could be members of more than one source network. It is however not clear, how this multi-membership is accounted for in the related IT systems and their usage, as systems from different networks might try to coordinate the same resources.Additionally Katzy and Sung note in [34] that research in VO engineering seems to face a barrier in adopting proven modeling methods from traditional enterprise engineering. For process modeling in networks, for example, only 40% of the projects relied on formal methods like IDEF and UML, while more than 60% were using Visio, PowerPoint and verbal descriptions in MS Word. Interesting enough is that a high

[47] Chen, D., B. Vallespir, et al. (2002). Developing an Unified Enterprise Modelling Language (UEML) - Requirements and Roadmap. Collaborative Business Ecosystems and Virtual Enterprises. L. Camarinha-Matos, Kluwer Publisher: 247-254.

number of projects that worked towards management models developed special modeling tools, e.g. VSD, Grade and Adonis. These kinds of tools allows presentation of process information in easy-to-understand way.

Modeling Aspects	Findings /Outcomes
Management Process in VO Source Network	The processes developed for managing the source network can be divided into three types: – Participation management (Integrate new partner, remove partner) – Management of the IT platform – Contract/rules definition management
Management Process in Operational VO	More than 60% of the reported projects (30% didn't cover this issue) focus on VO creation or configuration activities. The project (or contract) management during VO operation and the VO dissolution get less attention, and only one project has reported the modeling for all three processes.
Operational Process in VO	The spectrum of processes modeled for operational VO is approximately similar to the prcesses in a traditional enterprise.
VO Support Process	Comparable to the operational processes, the support process models reflect a wide scope of activities, which includes human resource management, financial controlling, performance measurement, configuration and management of IT platforms etc.
VO Source Network	Most of the reported projects have not provided information on the source network model even when the source network was considered an important factor for the VO. One reason might be that most projects have only focused on the operational phases of the VO, even though it seems difficult to develop suitable VO support systems without good understanding of the underlying source network.
VO Operation	Not all projects have reflected on the operational model underlying the VO, i.e. the type and structure of the links between the entities in the network. This could be a sign of the limited management orientation and understanding in some projects. However, peer-to-peer topologies and star-topologies seem to be most prevalent for VOs, while supply chain topologies might not require the special relationships between companies.
VO Governance	The General VO Governance model is following more or less the same trend as the VO operational model. Most of the star/consortia topology reported projects - 3 out of 5 - show co-existing of supply chain and star/consortia governance behaviors, while the remaining two projects exhibit purely star/consortia governance structure. A mixture of star/consortia and peer-to-peer topology co-exists within the peer-to-peer topology projects. No information on supply chain topology was given.
VO Management	Network coach / VE coordinator was indicated by majority of the reported supply chain topology projects, as the key management roles. While the Broker/Integrator was the dominating management role being reported by both peer-to-peer and star/consortia type of VO. This outcome, directly reflect the operation and governance models.

	Nevertheless, some projects also indicated that these management roles might changed according to the specific activities within the projects, i.e. product development, software application development, which might somehow misinterpreted the actual roles.

Table 5 Results form an extensive survey [34] of VO modelling approaches in Europe

3.7 Examples of Enterprise Integration reference models

We consider the following example of VO modelling frameworks:

1. The St. Gallen Management Model
2. The Generalized Enterprise Reference Architecture and Methodology (that supersedes CIM/CIMOSA, GRIM/GRAI and PERA).
3. Rosettanet
4. The Enterprise Viewpoint of the ODP Reference Model
5. The Open Grid Services Architecture VO model

3.7.1 St. Gallen Management Model

The **St. Gallen Management Model** has its origin in the pioneering work of Hans Ulrich on corporate modelling that started in 1954. In the mid-1960s, Hans Ulrich established a task force at the Insitute of Management of the University of St. Gallen in Switzerland, which studied the work of early writers on system theory and cybernetics. The first version of this model was developed by (Ulrich 1968; Ulrich and Krieg 1974) and further refined by (Rüegg-Stürm 1998; Rüegg-Stürm 2001).

The latest version of the St. Gallen Management Model (2002), depicted at the top row of Table 6, structures the enterprise into different important elements as. It distinguishes configuration from process and evolutionary views of VO modelling and it takes consideration of the context, main stakeholders and their interactions of relevance for the VO. It enhances the original Ulrich model (1972), depicted in the second row of Table 6, in various respects, including the following:

- There is more importance attached to the ethical, normative dimension of management.
- The new model reflects the enormously increased relevance of a process oriented view of the firm. This is especially due to innovations in information technology, intense time based competition, and the substantial role of social processes.
- There is much emphasis on the interpretative, meaning-based dimension of management.

With the integrating levels of strategy, structure, and culture, three main pillars of the second St.Gallen Management Model (third row of Table 6) developed in the Bleicher era (1991) also play an important role in the new model. In comparison, the latter again puts more emphasis on the process dimension. Also, the great contemporary issues of interactions (resources, norms and values, concerns and interests) reflected into the new model in relation to the stakeholders and VO context. Finally, Bleicher's concepts (fourth row of Table 6) expressing the challenges of normative and strategic management as a variety of specific tensions, e.g. the one between shareholder orientation versus stakeholder orientation in adjusting corporate policy has been incorporated in the new model through the interdependencies of what can be understood stakeholder roles that relate to each other through VO specific interactions (resources, norms and values, concerns and interests) or the environment (Society, Nature, Technology, Economy).

3.7.1.1 VO configuration

The configuration dimension of a VO focuses on "configuring forces" that cut across the process and evolutionary views, namely, strategy, structures and culture; the harmonization of strategic programs (or more general: activities), structure, and culture (or more general: attitudes) to a common chord, is seen as the essence of successful management.

3.7.1.2 VO processes

The process dimension of a VO emphasises the three categories of processes that cut across the configuration and evolutionary views. These include: management processes, business processes and support processes.

3.7.1.3 VO evolution

VO optimisation and renewal underpin the evolutionary dimension of a VO. Optimisation and renewal are complementary dimensions that cut across processes and configuring forces, therefore highlighting the dynamic nature of VOs.

3.7.1.4 Stakeholders

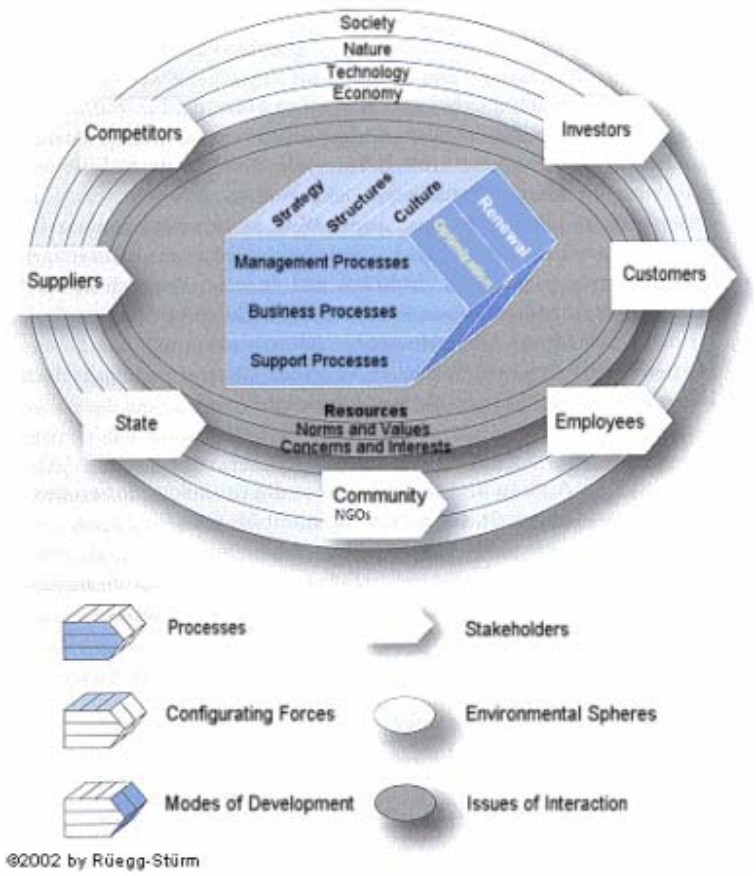
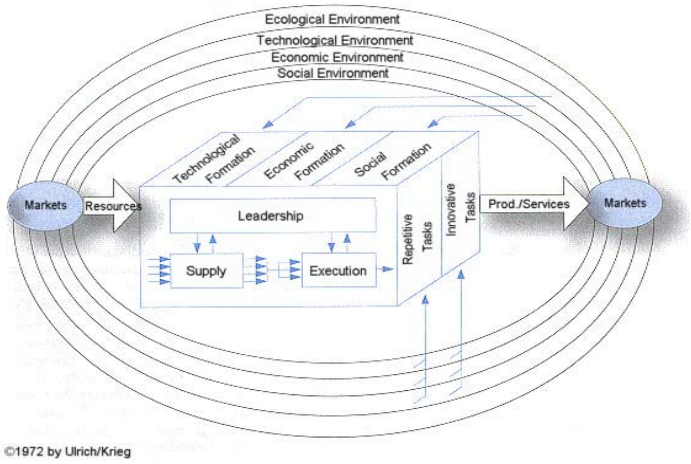
The stakeholders contextual dimension offers a classification of stakeholders' functional characteristics that can also be interpreted as roles of entities that interact with each other for the purpose of the VO.

3.7.1.5 Issues of Interaction

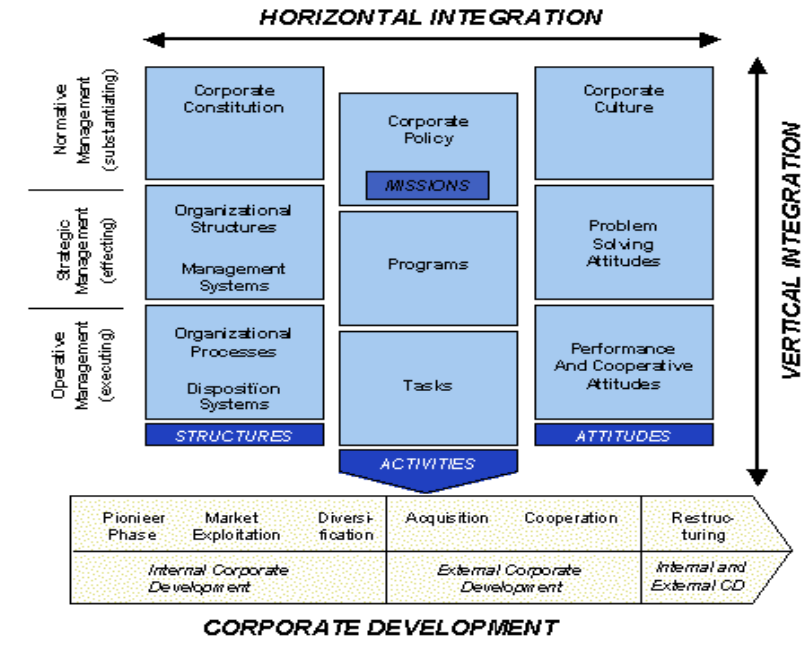
Issues of interaction characterise the main issues to be taken into account when modelling VO interactions and may indicate modes of interaction. These include, resources, norms and values, concerns and interests.

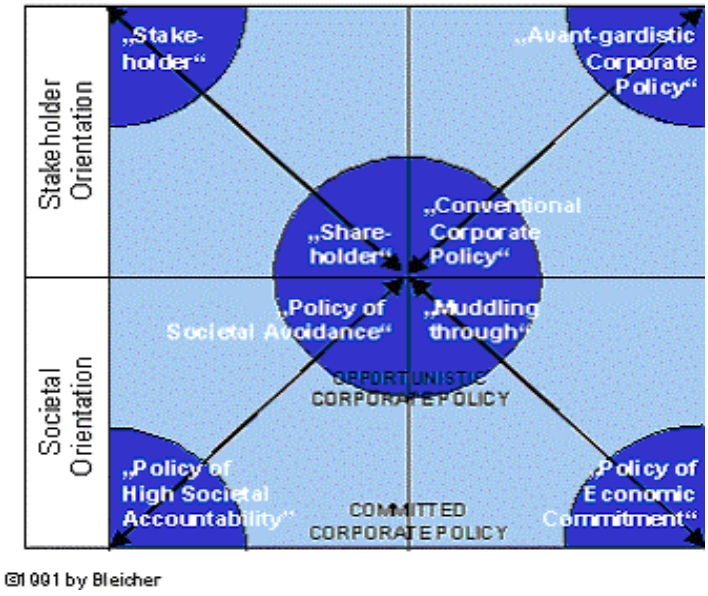
3.7.1.6 Environmental Spheres

St.Gallen Management Model also emphasises that both the stakeholders and the VO itself exist in a context and are influenced by their environment. A number of spheres of environmental influence are explicitly highlighted, including economy, technology, nature and society.

Rüegg-Stürm, J. (2002): Das neue St. Galler Management-Modell, Grundkategorien einer integrierten Managementlehre, der HSG-Ansatz, Berne, Switzerland etc. (Paul Haupt), 103 p., ISBN 3-258-06534-9	
Ulrich, H.; Krieg, W. (1972): Das St. Galler Management-Modell, Berne, Switzerland etc. (Paul Haupt), 54 p.; Latest edition: Ulrich, H. (2001): Gesammelte Schriften, Band 2, Berne, Switzerland etc. (Paul Haupt), 470 p., ISBN 3-258-06291-9	

Bleicher, K. (1991): Das Konzept integriertes Management, Frankfurt am Main, Germany etc. (Campus Verlag), XIX, 472 p., ISBN 3-593-34480-7





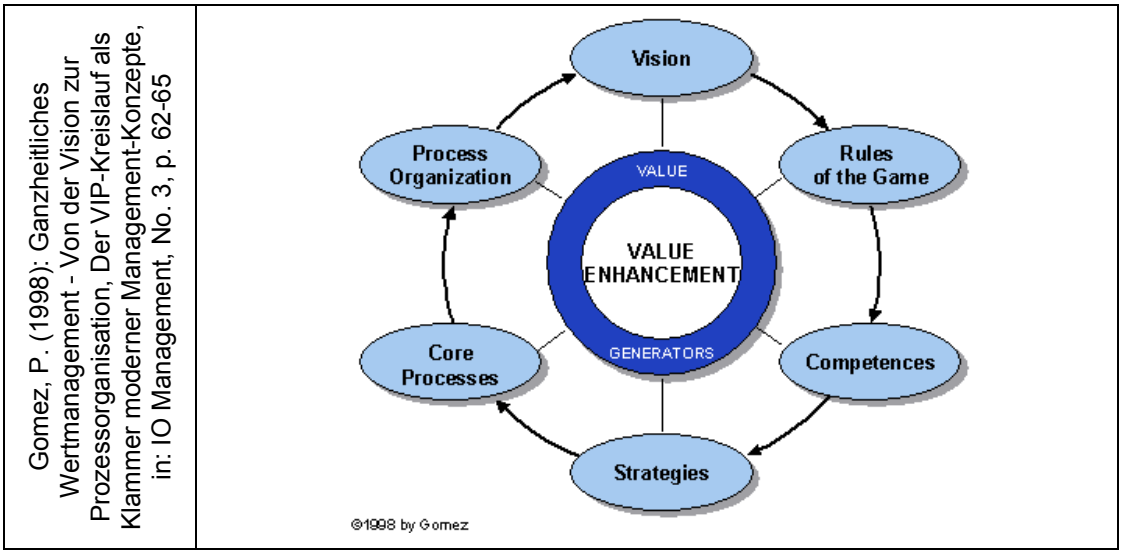


Figure 8 An overview of the development of the St. Gallen Management Model[48]

The purpose St.Gallen Management Model is to understand the enterprise and its behaviour better through observing, describing and analyzing it according to this structure. Every firm would be a specific combination of the different elements, which could be described verbally or through different diagrams. However the approach has taken the following assumptions [34]:

- Manifold and not immediately apparent relationships and interactions exist between the system's elements.
- Relationships and interactions are in a state of continuous and only marginally predictable development.
- Results are outcome of emerging behaviour from these relationships and interactions. Consequently they cannot be traced back to the behaviour of individual elements in any way, but originate in the interaction of the system elements and depend principally on particular patterns of ongoing interaction (Rüegg-Stürm 2001).
- The dynamic nature of such Organisational structures make it impossible to examine any given complex system from a single, central standpoint, to describe it fully and "objectively" and to depict it "accurately" within a model. Therefore a multimodal/multidimensional approach has been taken.

3.7.2 CIM/CIMOSA

In 1985, the ESPRIT Consortium AMICE started to work on the definition and specification of a CIM architecture for enterprise integration. Starting in 1990 the development work has been complemented by three independent ESPRIT projects (CIMPRES, CODE, VOICE) involved in CIMOSA validation. During a period of over ten years (CIMOSA completed in 1996) several companies and research organisations have been involved in its development and validation.

The CIMOSA Reference Architecture (Figure 9) aims to support the description of the enterprise, from the management level to the shop floor level. It consists of *(i)* an Enterprise modelling framework (reference architecture - RA), *(ii)* an Enterprise modelling language, and *(iii)* an Integrating infrastructure.

[48] System theory and cybernetics", in: Kybernetes, Vol. 30, No. 9/10, 2001

An overview of CIMOSA Reference Architecture is provided in Figure 9, its building blocks are summarised in Figure 10.

An overview of the process based enterprise modelling capabilities of CIMOSA is provided in Figure 9, where emphasis is put on the distinction between Domains, Domain Processes, Business Processes, Enterprise Activities, Functional Operations and Resources (Functional Entities), and their relationship. In particular, CIMOSA provides the means of describing enterprise domains and interacting domain processes Figure 11(A). The latter are decomposed in Business Processes and Enterprise Activities Figure 11(B). Business processes describe a process that allows a controlled aggregation of Enterprise Activities Figure 11 (B). Therefore Domain Processes eventually decompose to networks of Enterprise Activities, some of which are aggregated in accordance of business process rules Figure 11 (C). A functional model of an Enterprise Activity is provided in Figure 11(D). An Enterprise Activity can be further decomposed into an aggregation of functional operations that use resources (functional entities) in order to deliver their function Figure 11 (D).

Figure 12 summarises the CIMOSA Integrating infrastructure that provides a set of service entities for model engineering and enterprise operation control.

Figure 13 visualises a process guidelines that put the CIMOSA modelling process into the context of the life-cycle of an Enterprise system.

The main advantages of CIMOSA include the structuring of its Reference Architecture into generic and partial modelling levels, and also the substantial support that it provides for function, information, resource, and organisation modelling. It is also particularly strong for modelling and specifying process integration and exchange of information using domain processes, events, and “object” views. However, CIMOSA constructs definitions are not very clear, and largely depend on self-evident (as opposed of precisely defined) concepts and notations, its requirements definition modelling is weak and there are no reference models or meta-processes guiding the design of the system.

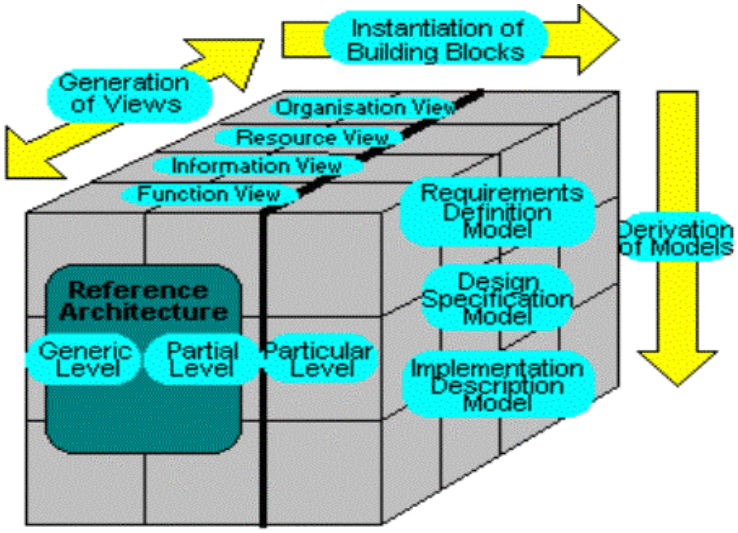


Figure 9 Overview of CIMOSA Reference Architecture (enterprise modelling framework)

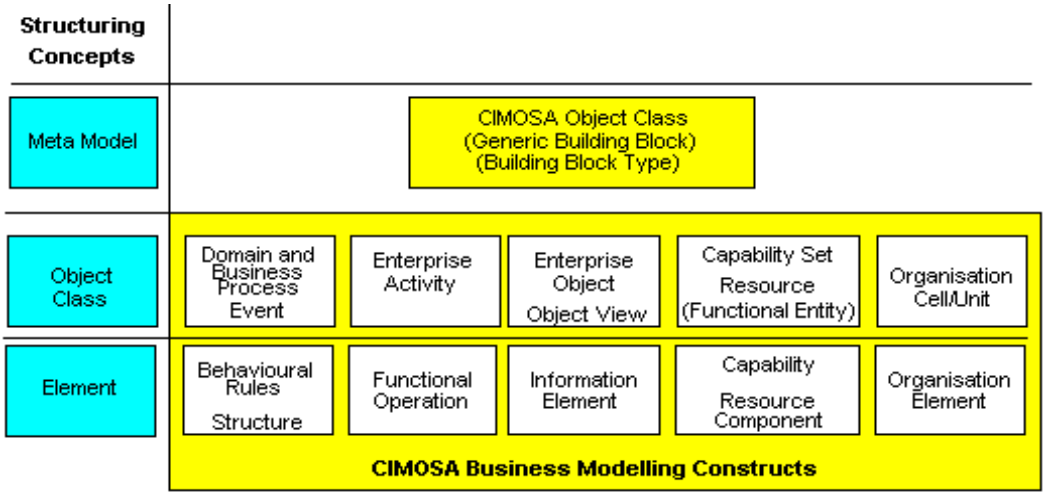
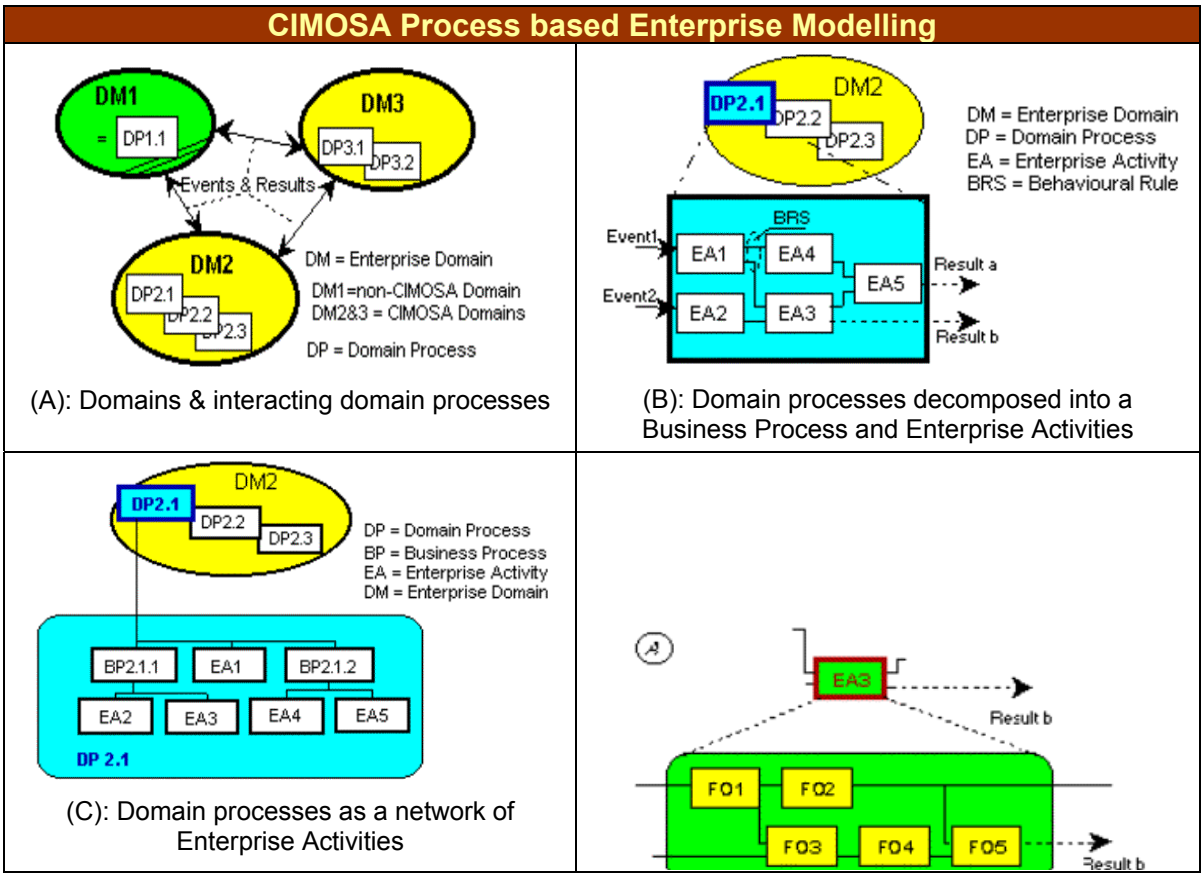
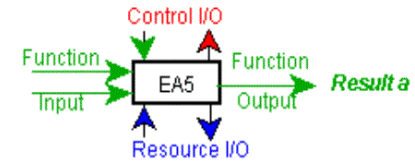


Figure 10 Overview of CIMOSA RA building blocks





EA = Enterprise Activity
I/O = Input/Output

(D): Enterprise Activity Specification

Figure 11 An overview of CIMOSA process based enterprise modelling

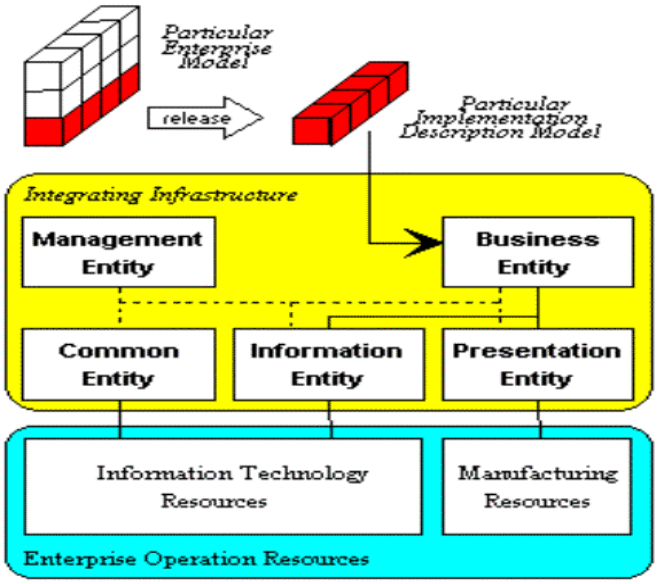


Figure 12 CIMOSA Integrating Infrastructure

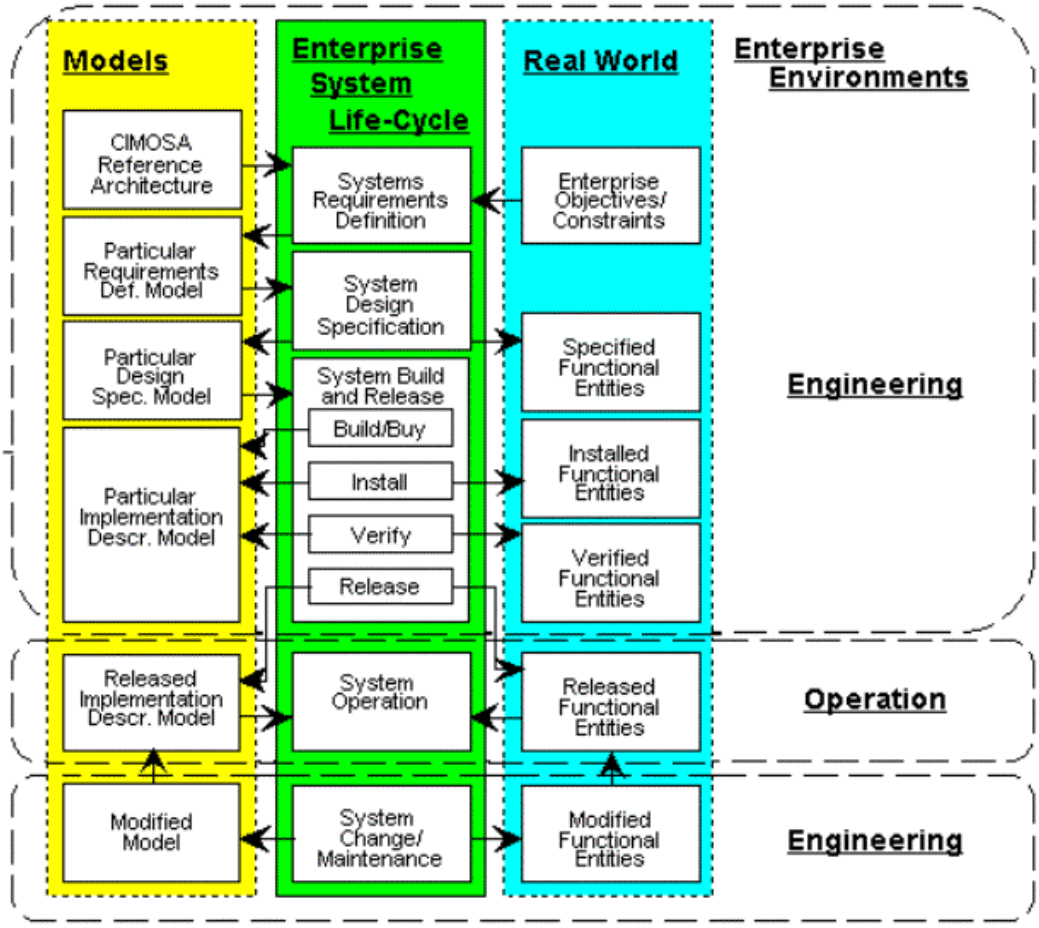


Figure 13 Relations between the Enterprise System Life Cycle and the progress of the CIMOSA modelling process

More information about CIMOSA is available at [49].

3.7.3 GIM/GRAI

Developed by the laboratory for automation and production at the university of Bordeaux-France since 1970's.

GIM is composed of the following elements:

- GRAI conceptual model: a representation of basic concepts of a manufacturing system decomposed into three sub-systems: physical system, decision and information system (See Figure 14).
- GIM modelling framework (RA) with three dimensions: views, life cycle, and abstraction level (See Figure 15).
- GIM structured approach: guide to show how to perform analysis and design of the manufacturing system in three main phases: analysis, user-oriented design, and technical-oriented design (See Figure 16).

[49] http://cimosa.cnt.pl/Docs/Primer/primer0.htm

– GIM modelling formalisms (languages): GRAI grid and GRAI nets for decision system modelling, IDEF0 and stock/resource for physical system modelling, ER for information system modelling, IDEF0 for functional system modelling (See Figure 17).

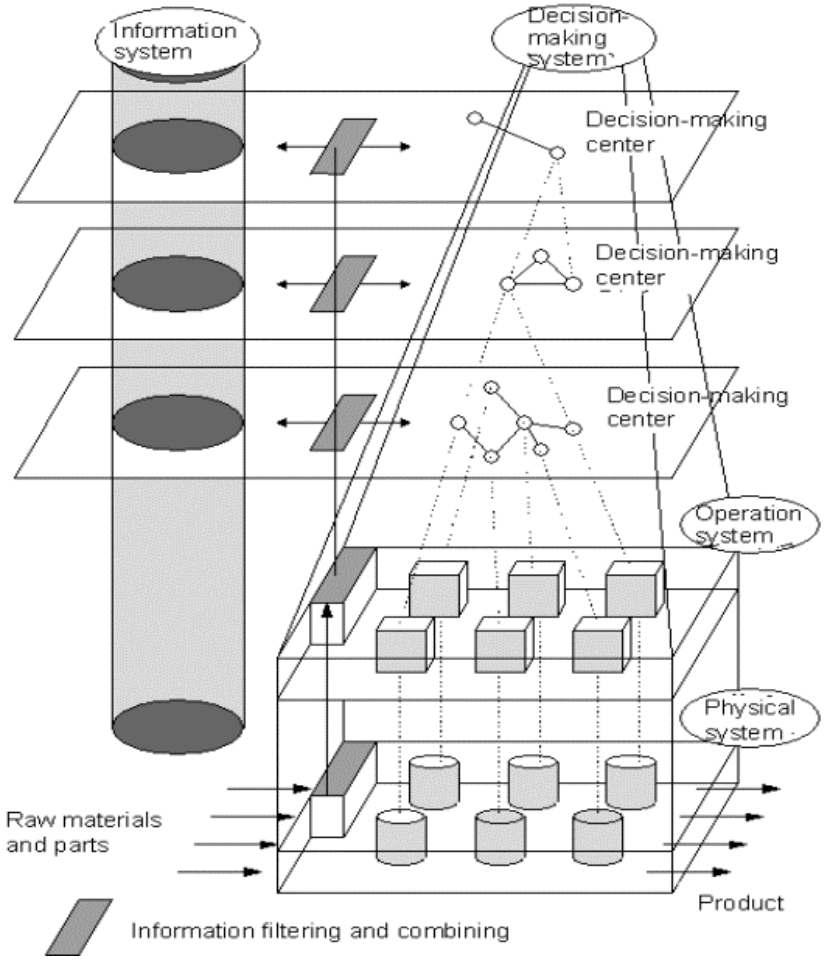


Figure 14 Overview of the basic concepts underpinning GRAI

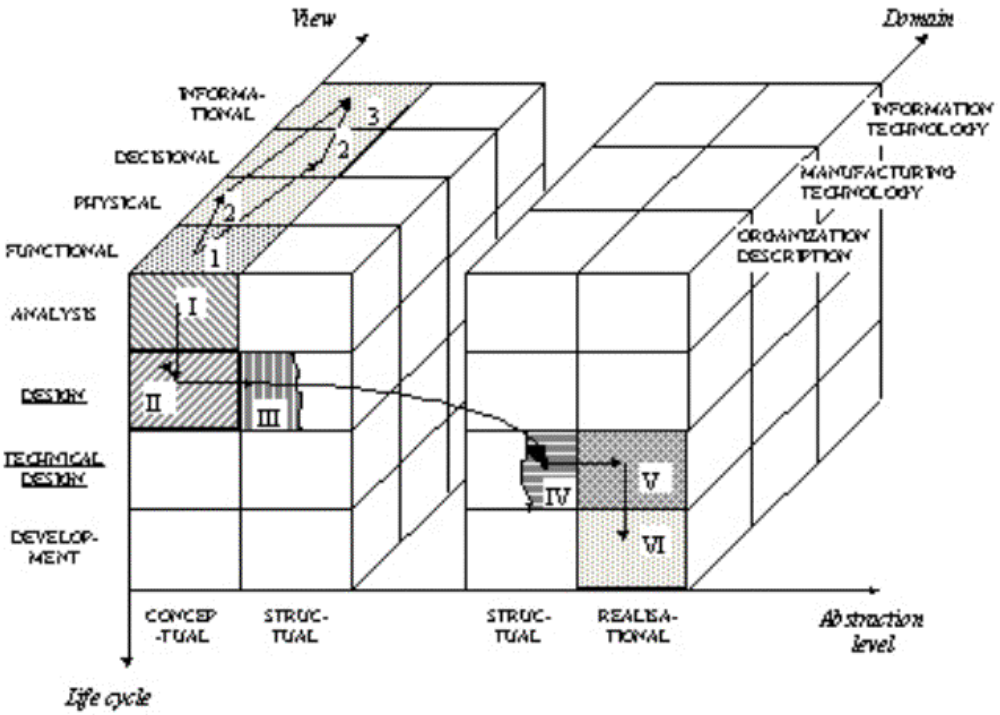


Figure 15 Overview of the GRIM reference architecture

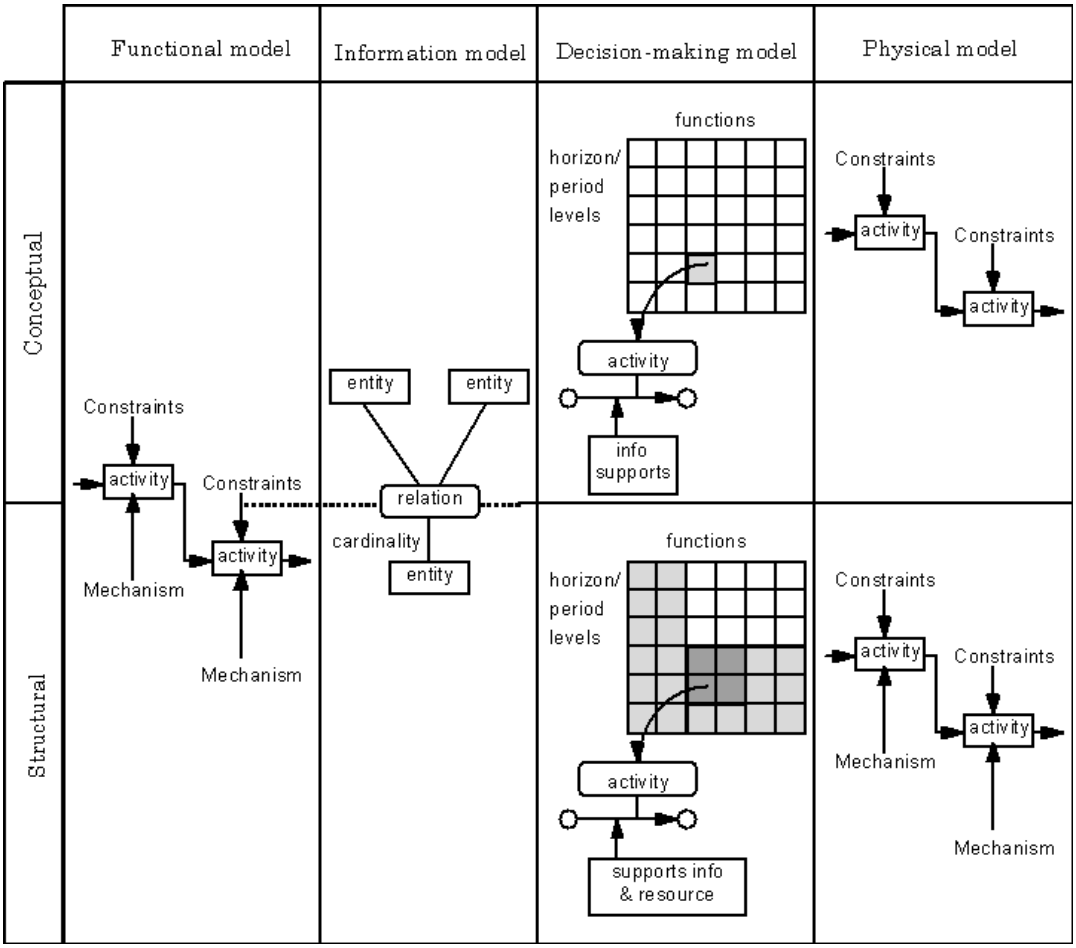


Figure 16 A summary of GIM modelling formalisms

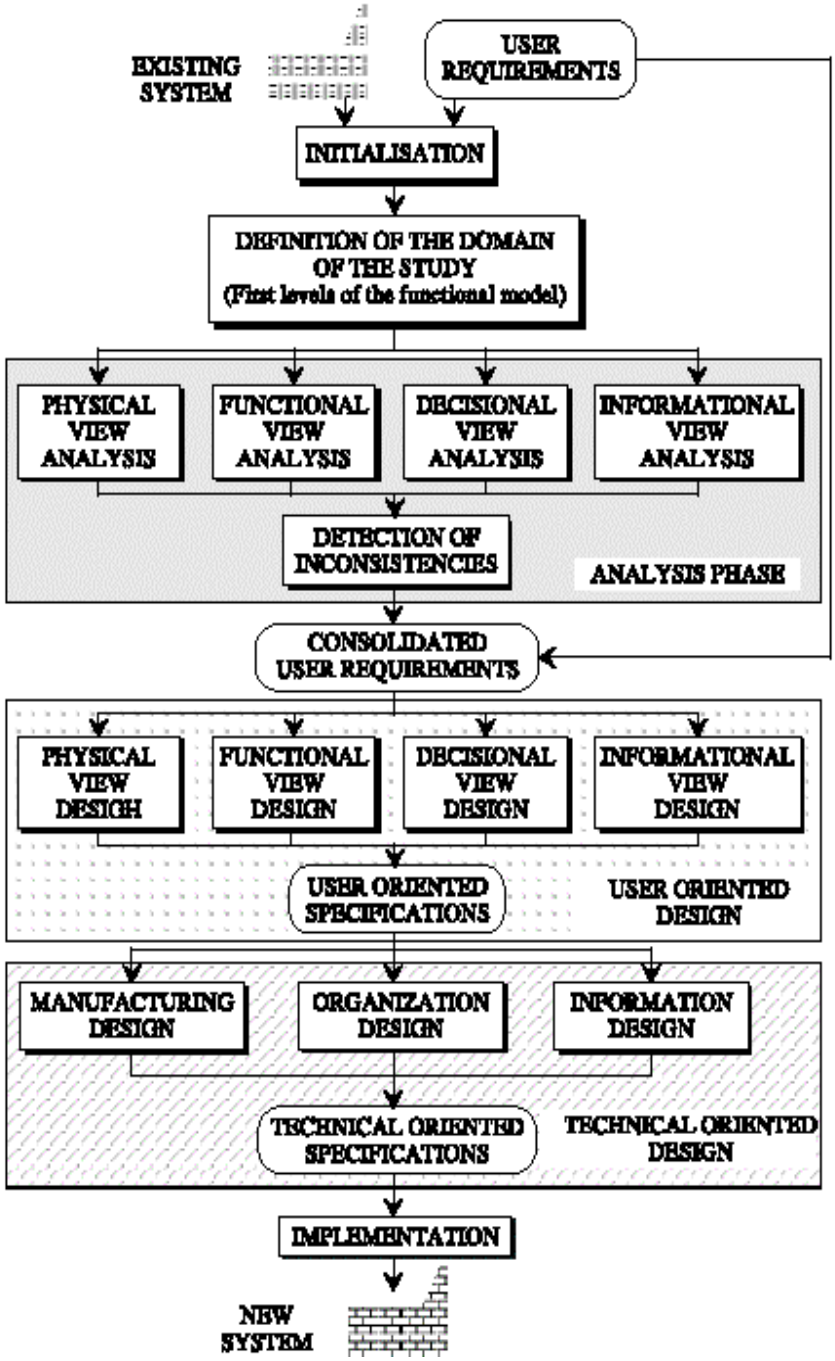


Figure 17 Overview of GIM structured approach (meta-process guidelines)

The main advantages of GIM/GRAI is the incorporation of an Enterprise system view that puts emphasis on control and decision making, and the presence of some reference models. On the negative side, although it provides a number of modelling notations for the different views it supports, some of these lack the richness of CIMOSA, and similarly to CIMOSA its definition level modelling is weak. For more information about CIM/GRAI see [50].

[50] D. Chen, B. Vallespir and G. Doumeingts. 1997. "GRAI integrated methodology and its mapping onto generic enterprise reference architecture and methodology" Computers in Industry. V33 p387-394.

3.7.4 PERA

PERA was developed at Purdue University during the period 1989-91. It focuses on elaborating some common concepts of systems engineering in enterprise integration.

It identifies the following concepts:

- The mission
- Separation of functions
- Networks of tasks
- The place of the human
- The life cycle
- Planning and organization of the integration effort (the master plan)

These are organised in the following Table 6. The reference architecture of PERA is summarised in Figure 18 (note that particular emphasis is placed on planning and on positioning the human in the enterprise modelling). Figure 19 provides a summary of the different the recommended models and tools for each phase of the life-cycle model of PERA.

PERA is distinguished by the focus on planning and operating the enterprise integration plan (PERA "master plan"), the emphasis placed in the role of the human, and a more complete coverage of the whole enterprise lifecycle, compared to CIMOSA and GRIM. However, PERA is impaired by the lack of a sufficiently rich collection of modelling formalisms and a clearly defined methodology.

Phase	Title	Description
1	Identification of the Enterprise Business	Identity and boundaries of the enterprise
2	Project Concept	Mission, vision and values Operational policies
3	Project Definition	Identify requirements, tasks and modules Develop flow diagram or other models of the Enterprise Entity
4	Project Specification or Preliminary Design	Identify human tasks, initial choice and specification of human organization Identify information and control equipment and mission fulfillment equipment
5	Completion of all detailed design needed for construction phase	Detailed design of human and organizational information, control, customer product and service components of the enterprise
6	Implementation, test and commissioning phase	Conversion of detailed design to actual plant elements, testing, operational trials and acceptance or commissioning
7	Operations phase	The enterprise is carrying out its mission
8	Decommissioning	Enterprise has come to the end of its economic life

Table 6 Overview of enterprise integration phases, according to PERA

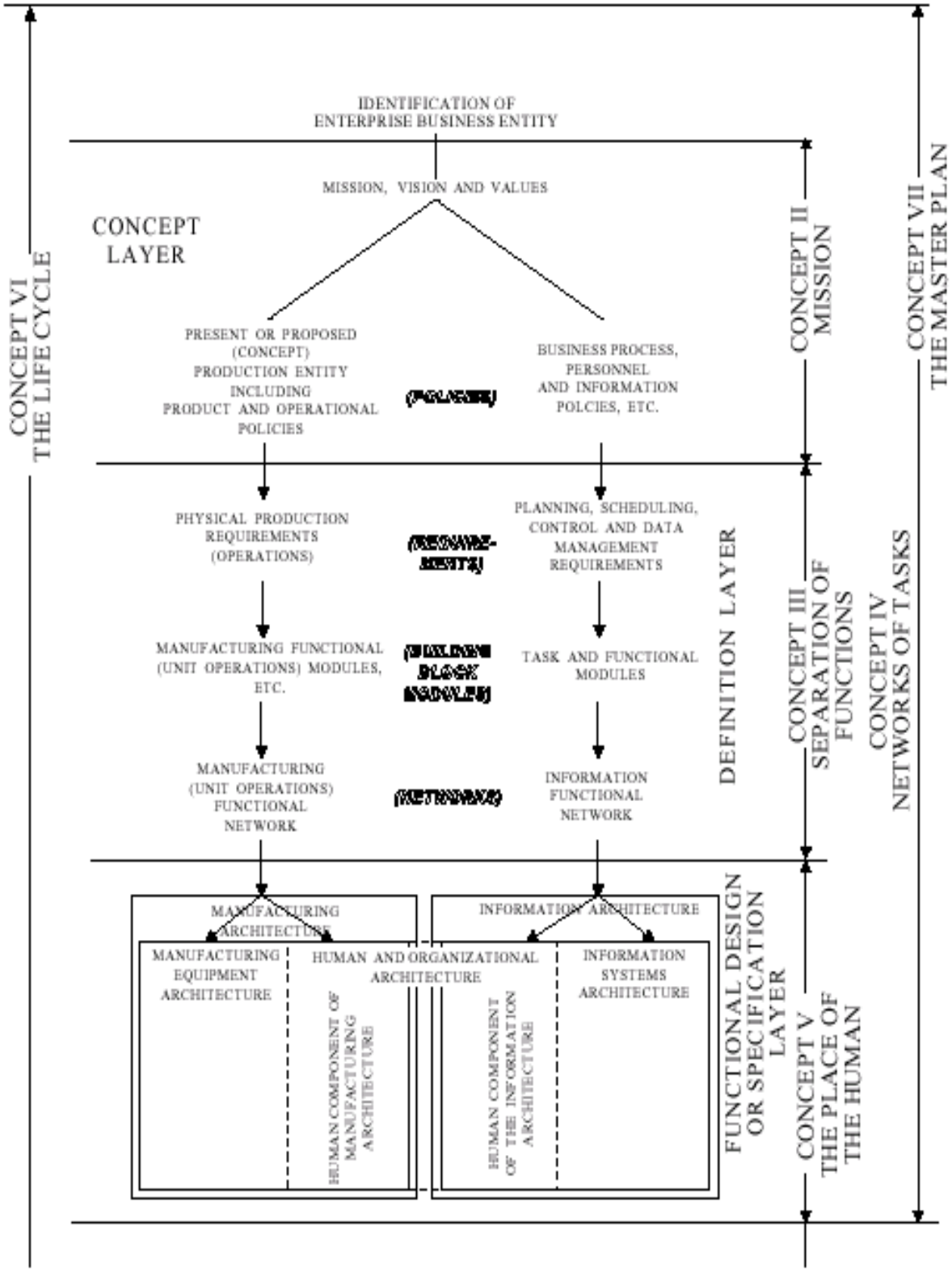


Figure 18 High-level overview of the PERA reference architecture

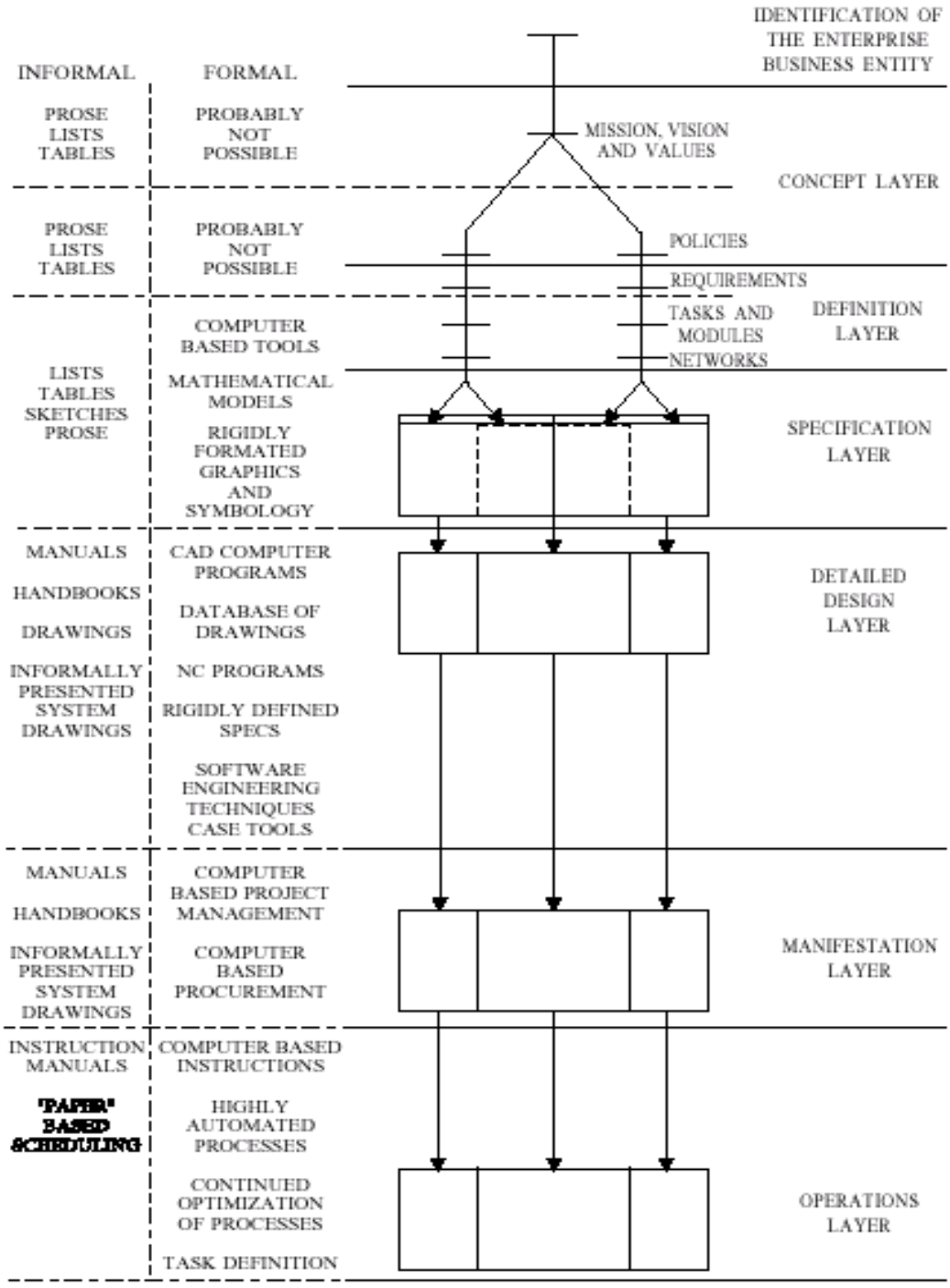


Figure 19 Summary of models & tools involved in each phase of the life-cycle, according to PERA.

For more information on PERA and its relationship to GERAM see [51].

[51] T. J. Williams, "PERA and GERAM – Enterprise Reference Architecture for Enterprise Integration".

3.7.5 Generalized Enterprise Reference Architecture & Methodology (GERAM)

The IFIP-IFAC task force on Enterprise Integration is currently developing GERAM. The most updated released is GERAM V1.6.3 from March 1999.

The following Reference Models were analysed by the IFIP-IFAC task force

- CIMOSA
- PERA, and
- GRAI-GIM,

and it was concluded that besides some overlap none of the models subsumed the others and each of them has something unique to offer (cf. report of IFIP-IFAC Task Force 1999). GERAM is a generalisation of existing architectures and other necessary elements. It also facilitates the unification of methods of several disciplines, such as methods of industrial engineering, management science, control engineering, communication and information technology and others.

An important characteristic of GERAM is its support for network of enterprises (virtual or extended enterprises); its scope can be part of an enterprise, a single enterprise or a network of enterprises.

As defined in [52] "GERAM defines a tool-kit of concepts for designing and maintaining enterprises for their entire life-history. GERAM is not yet-another proposal for an enterprise reference architecture, but is meant to organise existing enterprise knowledge. The framework has the potential for application to all types of enterprise. Previously published reference architectures can keep their own identity, while identifying through GERAM their overlaps and complementing benefits compared to others." The components that make part of the GERAM framework are defined in the following figure.

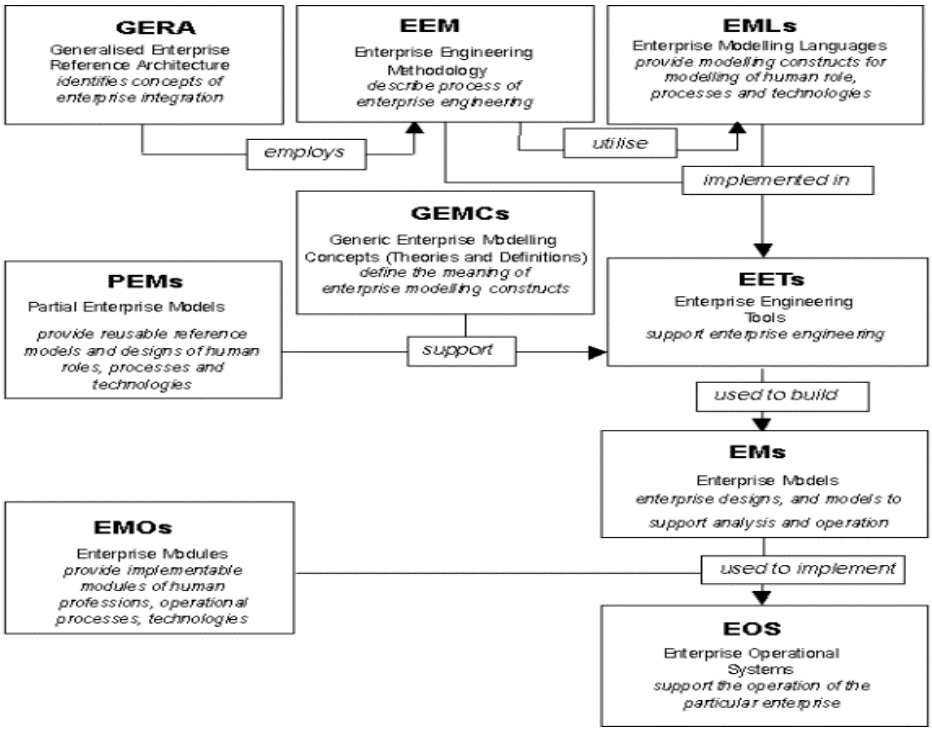


Figure 20 Overview of the GERAM framework components

[52] IFIP-IFAC Task Force on Enterprise Integration – Final report 2002. Joint effort between IFIP (WG5.12) and IFAC (TC MIA).

GERA: The Generalised Enterprise Reference Architecture- is the most important component in GERAM since it identifies the basic components to be used in enterprise engineering and integration.

EEMs: Enterprise Engineering Methodology- describes the process of enterprise engineering. For each type of change activity that describe ways of progression, identify tasks and tools.

EMLs: Enterprise Modelling Languages, provide modelling constructs for modelling the human role, processes, and technologies. Some of the languages should be formal, while other can be less formal.

GEMCs: Generic Enterprise Modelling Constructs define three forms of generic concepts:

- Glossaries - for end users - All the terminology used in formal and non formal models should be defined
- Meta-Models - for tool developers - Describe the concepts used, their properties and relationships
- Ontological Theories - for tool developers - They are formal models of concepts being used in enterprise representation. The paper from (Fox and Grunningen 1998) gives some pointers to research work being developed in this area.

PEMs: Partial Enterprise Models provide reusable reference models of human roles, processes and technologies. They are high quality tested models used for 'drag and drop' enterprise modelling - quickly and cost effectively.

EETs - Enterprise Engineering Tools, support enterprise engineering, and specifically must support the creation, use, and management of enterprise models.

EMs - Enterprise Models represent the particular enterprise and they include all those descriptions, designs, and formal models of the enterprise that are prepared in the course of the enterprise's life history.

EMOs - are implementable modules or products (software and hardware); human professions, operational processes, technologies, etc; the major functional components of the enterprise can only be planned for implementation if they are embodied in products available on the market. This includes available human resource on the job market, machinery, IT products, and services.

GERAM distinguishes between the methodologies for enterprise engineering (EEMs) and the modelling languages (EMLs). The modeling process produces enterprise models (EMs) using tools (EETs). The models will guide the implementation of the operational system of the enterprise (EOSs). The operation of these enterprise models is supported by specific modules (EMOs).

The GERA concepts can be divided in:

- Human oriented - deals with all the aspects that involve humans
- Process oriented - deals with the functionality and behaviour of enterprise operation; the activities that exist in each of the phases of the life-cycle are also dealt by these concepts
- Technology oriented - deals with the infrastructures that are needed to support processes. Resource models facility layout models, information system models, communication system models, and logistics models are examples of this type of concept.

From a process-oriented viewpoint, GERA emphasises the life-cycle of any entity as in Figure 21. The different life-cycle phases define types of activities that are pertinent during the life of the entity. Details about each of the phases can be found, for example, in the IFIP-IFAC Task Force 1999 report on GERAM.

The entities can be related in such a way that, for instance, an entity can support one phase of another entity. This aspect can be seen in Figure 21, where the operation phase of entity A supports the life-cycle activities for the design and implementation of activity B.

Examples of other relations between the life-cycle activities of enterprise entities may be defined. However, it is always the case that only the operational activities of entities influence the life-cycle activities of other activities. GERA introduces the concept of entity types and the relations between the different types. Two different ways of categorising enterprise types are particularly interesting: an operation oriented set and a generic and recursive set of enterprise entity types. The two sets have close relations to each other and both identify the product entity as the result of the operation of other entities. Figure 22 shows an example of relationship between GERA entity types and Figure 23 shows the relationship between their life cycles.

Another interesting concept is “life-history”; this represents the evolution of the system during its entire life span. The iteration shown in Figure 23 identifies the different change-flows required on the operational processes and, or the product, or customer services. “Life-history” is closely related to reengineering.

The Life cycle and Instantiation dimensions are quite similar to the Derivation and Instantiation dimensions of the CIMOSA cube representation (see CIMOSA summary subsection). But the view dimension is quiet different because in this dimension one can distinguish the co-existence of four model views:

- Model Content - function, information, resource, and organisation
- Purpose - customer service and product, management and control
- Implementation - human implemented tasks, management and control automated tasks, and customer service and product automated tasks (similar to PERA)
- Physical Manifestation - software, and hardware

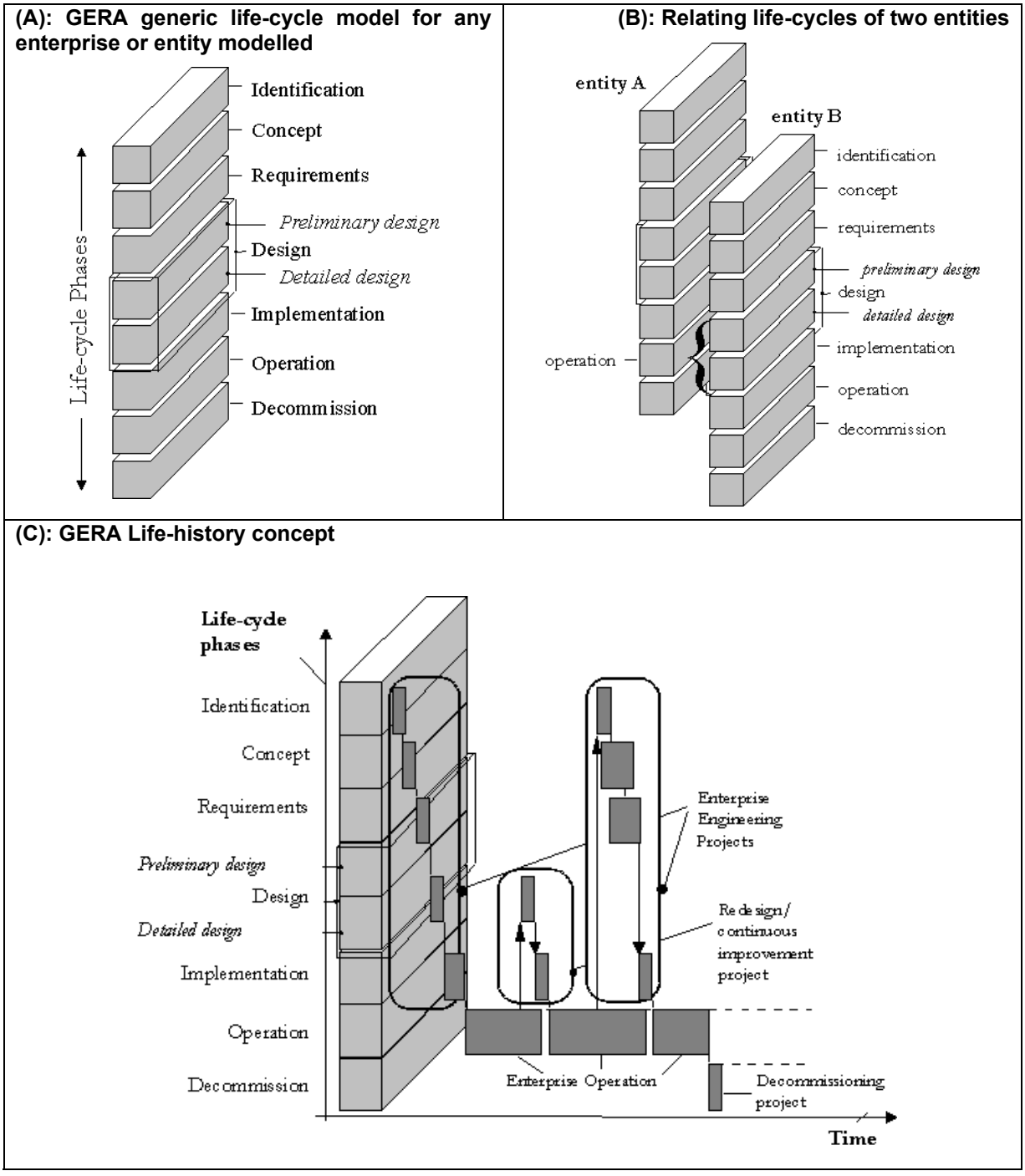


Figure 21 GERA dynamic (process) views

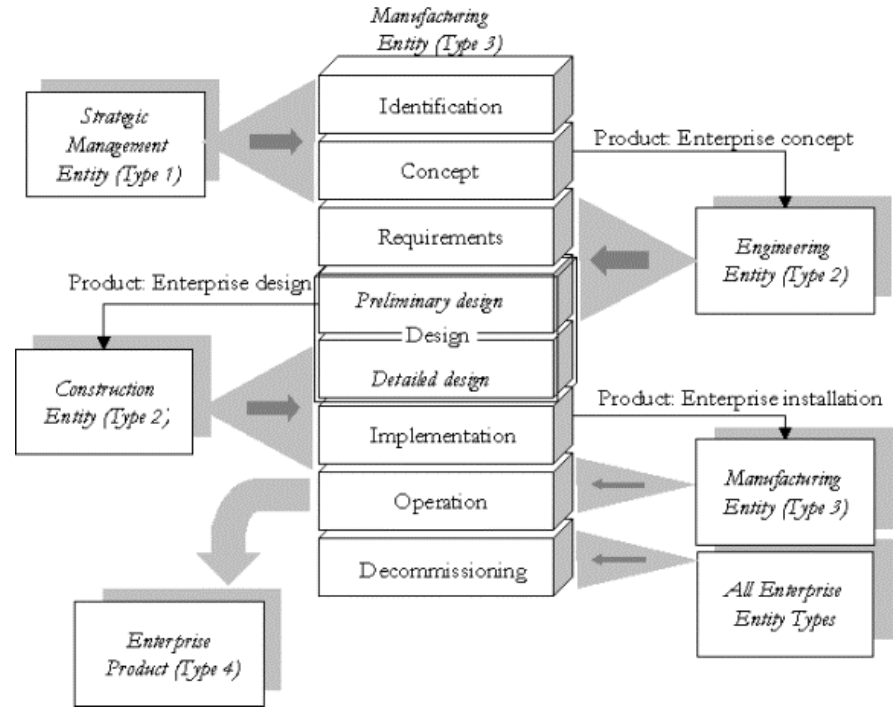


Figure 22 Relationships between GERA entity Types

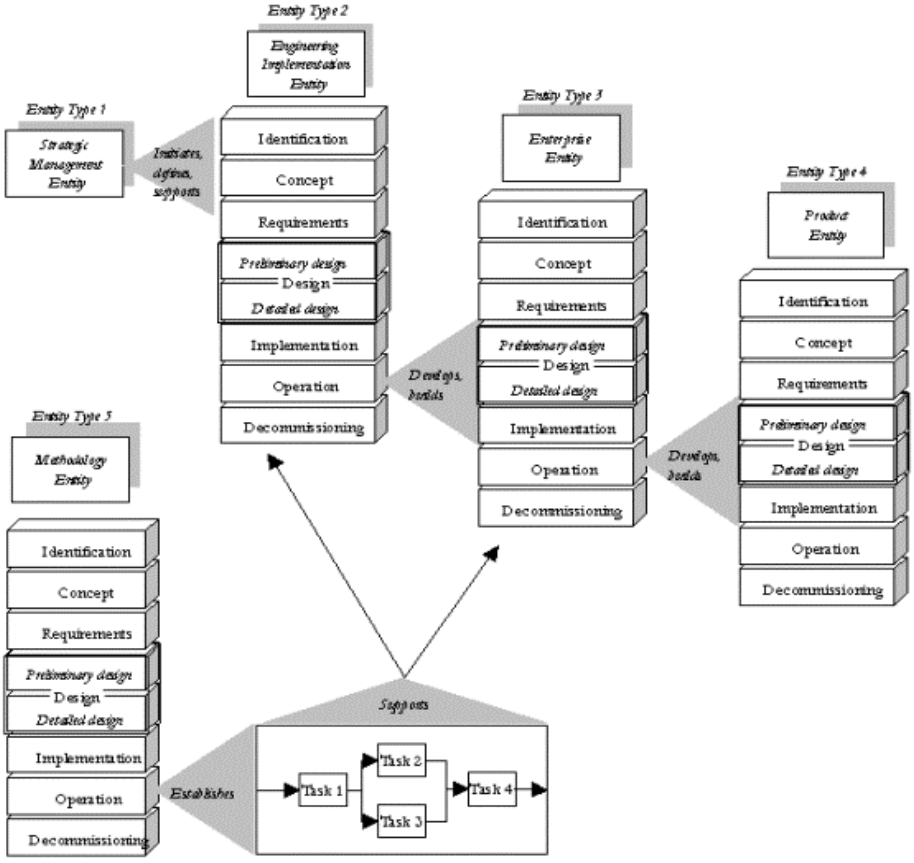


Figure 23 Relationships between life-cycles of GERA entity Types

The EEMs - Enterprise Engineering Methodology (), describe the process of enterprise engineering. For each type of change activity they describe ways of progression, identify tasks and tools.

The EMLs - Enterprise Modeling Languages (), provide modeling constructs for modeling the human role, processes, and technologies. Some of the languages should be formal, while other can be less formal.

The GEMCs - Generic Enterprise Modeling Constructs define three forms of generic concepts:

- Glossaries - for end users - All the terminology used in formal and non formal models should be defined
- Meta-Models - for tool developers - Describe the concepts used, their properties and relationships
- Ontological Theories - for tool developers - They are formal models of concepts being used in enterprise representation. The paper from Fox and Grunningen[53]) gives some pointers to research work being developed in this area.

The PEMs - Partial Enterprise Models provide reusable reference models of human roles, processes and technologies. They are high quality tested models used for 'drag and drop' enterprise modelling - quickly and cost effectively.

The EETs - Enterprise Engineering Tools, support enterprise engineering, and specifically must support the creation, use, and management of enterprise models.

The EMs - Enterprise Models represent the particular enterprise and they include all those descriptions, designs, and formal models of the enterprise that are prepared in the course of the enterprise's life-history.

The EMOs - Enterprise Modules are implementable modules or products (software and hardware); human professions, operational processes, technologies, etc; the major functional components of the enterprise can only be planned for implementation if they are embodied in products available on the market. This includes available human resource on the job market, machinery, IT products, services.

Concluding; GERAM unifies two distinct approaches of enterprise integration: one based on product models, and one based on business process design. Reengineering activities is supported by the architecture, which allows one to select from existing component archetypes. It combines the modelling capabilities of CIMOSA and associated tools with the PERA guide, and it is also extended with reused organizational blueprints from MIT and decision system analysis techniques from GRAI.

[53] Fox, M. S.; Gruninger, M. 1998. Enterprise Modeling. AI Magazine 19(3): 109-121

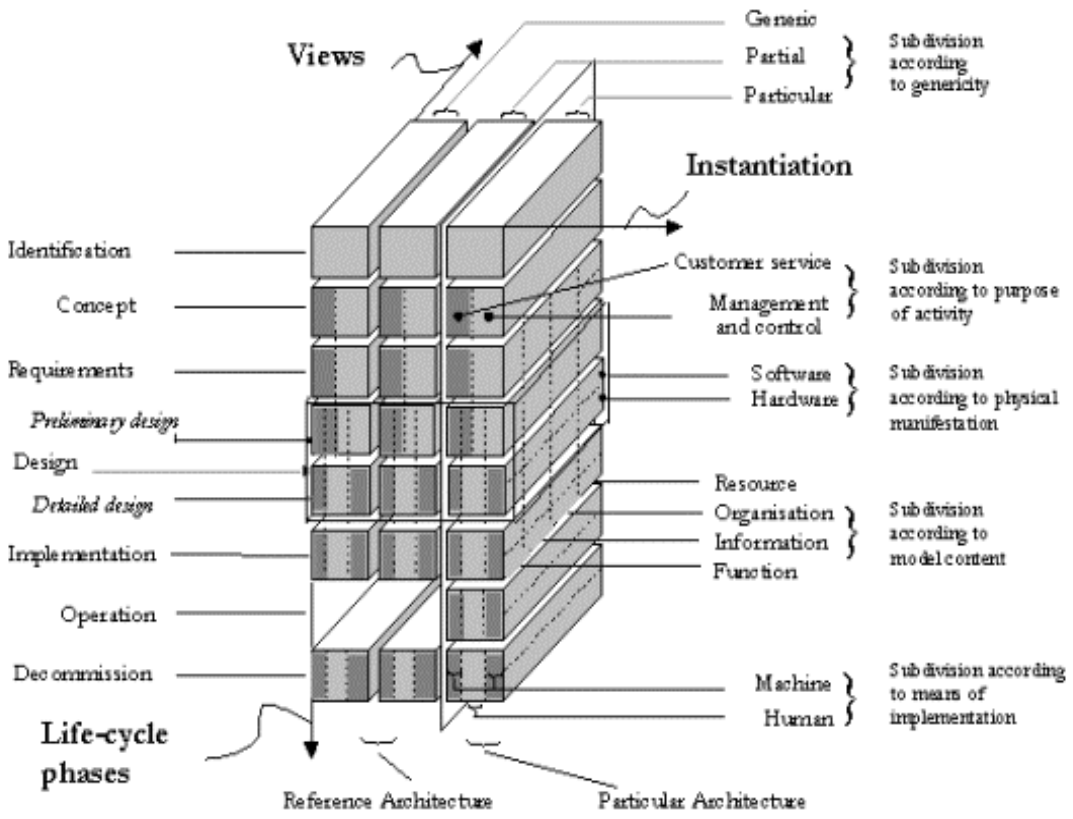


Figure 24 Overview of the GERAM modelling framework and associated views

Further information about GERAM is available at the task-force website at[54].

3.7.6 Rosettanet

RosettaNet[55] is a non-profit consortium of more than 500 organizations working to create, implement and promote open e-business standards and services. It comprises world-leading Electronic Components (EC), Information Technology (IT), Logistics (LG), Semiconductor Manufacturing (SM), Solution Provider (SP) and Telecommunications (TC) companies.

Its mission is to drive collaborative development and rapid deployment of internet-based business standards, creating a common language and open e-business processes that provide measurable benefits and are vital to the evolution of the global high-technology trading network.

Its main difference from other existing standards/protocols is that RosettaNet is process-centric, i.e. it deals with processes rather than messages. Therefore, it enables systems to interact in real-time rather than delayed actions (due to overnight batch treatment). Furthermore, RosettaNet deals with 100% of the B2B processes and since it's internet-

[54] http://www.cit.gu.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/v1.6.3.html

[55] RosettaNet main website: http://www.rosettanet.org

enabled and XML-based, it can be global. Those features are what make RosettaNet so powerful.

3.7.6.1 Key results

1. RosettaNet Implementation Framework (RNIF)[56]

- General Information: RNIF, a TRP (Transfer Routing and Packaging) specification aims at enhancing existing exchange protocols. RNIF enables companies to exchange data whether it is binary documents (PDF, GIF…) or text documents (XML-formatted). Main objective: drive down costs of system configuration that enable trading partners' systems to interact thus giving smaller companies the opportunity to join in.
- How does it work? RosettaNet's RNIF 2.0[57] deals with 3 majors aspects: packaging, protocol, and security. Packaging is based on the RosettaNet Business Message that packs together the business payload, associated header components and other entities. RosettaNet also specifies the use of MIME for the basic enveloping construct to package together the components of the RosettaNet message. These messages also include an instance of the preamble, delivery, and service header components (all XML documents). Lastly, the payload part of the message allows for optional attachments. The messages are not bound to any protocol. Therefore, it can be transferred between 2 RosettaNet endpoints. RNIF 2.0 can make use of the HTTPS protocol as well as the SMTP protocol. Other protocols may be supported in the future, such as FTP or BEEP (Block Extensible Exchange Protocol). The RNIF 2.0 provides the schemas for the Receipt Acknowledgement and Exception messages that govern the PIP message exchange. As for security, the core specifies authentication, authorization, encryption (confidentiality), and non-repudiation requirements essential for conducting secure electronic business over the Internet. Use of digital signatures can then help in non-repudiation as well as in detection of tampering and data integrity.

2. Partner Interchange Processes™ (PIP)[58]

The PIPs depict activities, decisions and interactions that fulfill a business transaction. They specify structure and format of business document payloads. PIPs can be broken down into clusters and segments.

As of today, there are 8 clusters that can be broken down into segments. The clusters are as follows:

(a) RosettaNet Support (cluster 0) This cluster provides administrative functionalities and includes 2 segments (administrative and testing). As of today the administrative part deals with failure only.

(b) Partner Product and Service Review (cluster 1). This cluster allows companies to share contact information and to send and receive acknowledgement of receipt. It also helps suppliers with the management of product information.

(c) Product Information (cluster 2). This cluster enables the distribution and periodic update of product and detailed design information (product change notices, product technical specifications). It also allows for collaborative design and engineering of a new product.

(d) Order Management (cluster 3). This cluster supports full order management business area from price and delivery quoting through purchase order initiation, status

[56] RosettaNet Implementation Framework (RNIF): http://www.service-architecture.com/web-services/articles/rosettanet_implementation_framework_rnif.html

[57] Realizing the Benefits of Implementing RNIF Version 2.0: http://www.rosettanet.org/RosettaNet/Doc/0/F9RGU5AMQBM4J66Q0BT84NQ247/RNIF2finalv3.pdf

[58] Partner Interface Process (PIP): http://www.service-architecture.com/web-services/articles/partner_interface_process_pip.html

reporting, and management. Order invoicing, payment and discrepancy notification are also managed using this Cluster of processes.

(e) Inventory Management (cluster 4). This cluster deals with collaboration, replenishment, price protection, reporting and allocation of constrained product. There are 6 segments dealing with forecasting, inventory management (allocation, reporting, replenishment), sales reporting, and price protection.

(f) Marketing Information Management (cluster 5). This cluster enables communication of marketing information, such as campaign plans, lead information and design registration. It's then broken down into 4 different segments (lead opportunity management, marketing campaign management, design win management, ship from stock and debit).

(g) Service and Support (cluster 6). This cluster provides post-sales technical support, service warranty and asset management capabilities. It includes 3 segments: segment 6A enables registration and product warranty support, segment 6B provides and administers asset management (it's combined with 6A), and segment 6C provides technical support and service management.

(h) Manufacturing (cluster 7). This cluster enables the exchange of design, configuration, process, quality and other manufacturing floor information to support the "Virtual Manufacturing" environment. 3 segments are included in this cluster: design transfer, management of manufacturing of WO (Work Order) & WIP (Work In Progress), and lastly distribution of manufacturing information.

RosettaNet is currently adding more PIPs and validating them one at a time. It's an ongoing process in which the companies behind RosettaNet take part actively.

3. **RosettaNet Dictionaries**

As RosettaNet dictionaries provide common properties for the PIPs, they ensure consistent information exchange during PIP™ execution. They are 2 different dictionaries that, together, provide a common vocabulary base for e-business, therefore getting rid of misunderstandings and confusion.

(a) A technical dictionary that specifies common product properties. This dictionary is made in such a way that partners don't need their own dictionary anymore when implementing PIPs. Furthermore, it isn't supply-chain specific anymore.

(b) A business dictionary that specifies common partner properties and enables partners to identify one another (very similar to a phonebook).

These dictionaries rely on common standards (partner and product identification, see below).

4. **Product & Partner Codes**

There are 3 codes in use as of today, all of which are specified by RosettaNet. These codes help identify products as well as partners uniquely. Therefore processes can be quickened and time spared.

(a) The GTIN (GTIN: Global Item Number) is used for the global product identifier in tks PIPs. It's an international multi-industry standard that makes it possible to identify products as well as services uniquely and globally.

(b) The UN/SPSC (United Nations/Standards Product and Services) is used in RosettaNet as a global class identifier in the PIPs. The UN/SPSC is a code standard for classifying products and services. Items are classified using numbers derived from the system's five-level hierarchy in which two digits are assigned at each level. The UN/SPSC allows trading partners worldwide to uniformly classify products and services, resulting in accuracy and efficiency throughout the trading network.

3.7.7 Global Company Identifier Company

RosettaNet specifies the Data Universal Numbering System (D-U-N-S®) for Global Company Identifier in its PIPs. The nine-digit D-U-N-S Number is a worldwide standard for company identification, distinguishing unique business locations around the globe. D-U-N-S Numbers are assigned and maintained by Dun and Bradstreet (D&B). To obtain a D-U-N-S Number, contact D&B. It usually takes 24 hours for organizations in the United States and longer for those in other countries. D-U-N-S Numbers enable organizations to clearly identify trading partners as well as accurately gauge risks and opportunities.

The Open Distributed Processing (ODP) Reference Model Enterprise Viewpoint[59] defines concepts for specifying an abstraction of a system within a defined environment in terms of the system's purpose, its scope and the policies that apply to the system from its environment as well as those defined within the system. The current specification defines terms such as policy, role, domain, and community but there is no associated language by which a system can be clearly specified, analysed or implemented.

A *community* is defined as a configuration of objects formed to meet an objective, which is expressed as a *contract* specifying how the objectives of the community must be met. The community is defined in terms of the following elements[60]:

- The *enterprise objects* comprising the community,
- The *roles* fulfilled by each of those objects,
- The *policies* governing the interactions between enterprise objects fulfilling roles,
- The policies governing the creation, usage and deletion of resources,
- The policies governing the configuration of enterprise objects and assignment of roles to enterprise objects,
- The policies relating to the environment contract governing the system.

A *contract* is a specification of the agreed behaviour necessary to meet the objectives of the community. It is specified in terms of: *roles* which define particular sets of behaviour associated with individual objects in the community, *relationships* which identify the interactions taking place between the objects fulfilling the roles and the *resources* associated with the actions and interactions identified by roles. It also specifies objectives of the community and the policies that apply to the objects and their activities. Policies can be obligations, permissions or prohibitions. Since roles relate to any enterprise objects, whether active actors or resources, two types of roles can be distinguished: *actor roles* which involve enterprise objects performing activities and *artefact roles* which are fulfilled by an enterprise object representing resources and are subject to the actions associated with the actor roles. The concept of contract in the enterprise viewpoint can then be mapped onto the notion of binding in the computational viewpoint. An example of enterprise viewpoint specification relating to the configuration of a leased line is given in [60].

Ponder, discussed in section 8.3 was not originally designed as an Enterprise Viewpoint Language, however it does provide a concrete representation to "realise" most of the tangible concepts being defined for the Enterprise Viewpoint.

3.7.8 The Service Oriented computing paradigm as an enabler of VO frameworks

Enterprise computing systems must increasingly operate within virtual organizations (VO) with similarities to the scientific collaborations that originally motivated Grid computing. Depending on the context, the dynamic ensembles of resources, services, and people that comprise a scientific or business VO can be small or large, short- or long-lived, single- or

[59] Enterprise-Viewpoint Reference Model, CD 15414, ISO/IEC JTC1/SC7 N2187, 1999.

[60] The Working document for RM-ODP Enterprise Viewpoint and Application Architecture (Canberra Output) ISO/IEC JTC 1/SC 21/WG7N1203, June 1997

multi-institutional, and homogeneous or heterogeneous. Individual ensembles can be structured hierarchically from smaller systems and may overlap in membership.

In all cases, however, the development of an effective IOIS for a VO has common requirements in terms of common security semantics, distributed workflow and resource management, coordinated fail-over, problem determination services, etc. across a collection of resources with heterogeneous and often dynamic characteristics.

Service Oriented computing frameworks allow for the creation, maintenance, and application of the service ensembles that VOs maintain. Key business functions are treated as services – that is globally identifiable and discoverable network-enabled entities that provide some capability through the exchange of messages over standardized extensible protocols that allow data-encapsulated cross-application invocations.

Adopting this uniform service-oriented model allows to architect systems that make all key components of the environment virtual therefore introducing transparencies that separate virtualized VO operations from the implementation of physical applications and resources on which the model is grounded.

Key aspects of interoperability via virtualization in Service Oriented computing

The service-oriented view partitions the interoperability problem into two subproblems:

- the definition of service interfaces and
- the identification of protocols that can invoke a particular interface.

A service-oriented view addresses the need for standard interface definition mechanisms, local and remote transparency, adaptation to local OS services, and uniform service semantics. A service-oriented view also simplifies virtualization through *encapsulation of diverse implementations behind interfaces* that are built on widely accepted open standards. Key advantages of virtualization include:

- Enabling consistent resource access across multiple heterogeneous platforms;
- Enabling mapping of multiple logical resource instances onto the same physical resource;
- Facilitating management of resources within a VO based on composition from lower-level resources;
- Enabling the composition of basic services to form more sophisticated services, regardless of how these services are implemented;
- Mapping common semantic service behaviour seamlessly onto native platform facilities.

Virtualization becomes easier when service functions are expressed in a standard form, so that any implementation of a service is invoked in the same manner. For example the Web Services Description Language (WSDL) distinguishes between the service interface definition and the protocol bindings used for service invocation, while ensuring that they are both distinct from the service implementation. That is, a single interface can have multiple bindings, including both distributed communication protocols and locally optimized bindings for interactions on the same host. Binding properties may also include reliability and other QoS guarantees, as well as mechanisms for authentication, authorization and delegation of credentials. When properly architected, the choice of binding is always transparent to the requestor with respect to service invocation semantics, but not to QoS, security, etc., guarantees: a requestor is able to choose a particular binding for performance reasons.

Multiple service implementation transparency

A service can support multiple implementations on different platforms, facilitating seamless overlay not only to native platform facilities but also, via the nesting of service implementations, to virtual resource ensembles.

For example, depending on the platform and context, one might:

- Construct a reference implementation for full portability across multiple platforms to support the execution environment for hosting a service;
- Use a platform possessing specialized native facilities for delivering service functionality to map from the service interface definition to the native platform facilities;
- Apply these mechanisms recursively, constructing a higher-level service from the composition of multiple lower-level services, which themselves can either map to native facilities or decompose further. This in turn dispatches operations to lower-level services.

The ability to adapt to operating system functions on specific hosts, whether this concerns performance monitoring, workload management, problem determination, or enforcement of native platform security policy, is central to virtualization of resource behaviors.

Service discovery and location mechanisms are also important in this regard, in that they should allow higher-level services to discover what capabilities a particular interface implementation supports.

For example, if a native platform supports reservation capabilities, a resource-management interface implementation can exploit those capabilities.

Conclusion

Service oriented computing has the potential to provide a ground for accommodating the ICT aspects of frameworks focusing on ICT aspects of Virtual Organisation (i.e. the Inter-Organisational Information System). In particular:

- Service oriented architectures support local and remote transparency with respect to service location and invocation.
- Service oriented architectures also provide multiple protocol bindings to facilitate localized optimization of services invocation when this can make a difference (e.g. when the service is hosted locally with the service requestor, or common application interfaces are in place).
- Service oriented architectures also enable protocol negotiation for network flows across organizational boundaries to allow choosing between several protocols, each optimized for a different purpose.
- The implementation of a particular service interface can map to native platform functions and capabilities.

However, the ability to virtualize and compose services depends on more than standard interface definitions. It also requires standard semantics for service interactions so that, for example, standard mechanisms for discovering service properties, communicating between services, invoking their operations and different services follow the same conventions for error notification. To this respect W3C has standardized an extensible core that builds on top of already established Web standards (such as XML and HTTP) in order to allow service interoperation over the Web. This Web Service core includes a Web Service Description Language (WSDL), a message communication protocol (SOAP) that also allows for encapsulating service operation invocations encapsulated in the messages exchanged between Web services, and a framework for Web Services registration, discovery and invocation (UDDI). Other standardization bodies such as OASIS, as well as, vendor coalitions (e.g. IBM, Microsoft, BEA, Verisign and others) are continuously develop new extensible standards proposals that build on top of the W3C core Web Services in order to support federating services as well as interoperability in relation to QoS, security, and manageability. Many of these established or emerging, specific open standards are examined in the remaining of this deliverable.

We can conclude that the Service Oriented Computing paradigm is clearly an enabler from a conceptual VO frameworks point of view. We will revisit Service Oriented Computing from a technical Web and Grid Services angle in chapter 6.

3.7.9 OGSA Virtual Organisations

OGSA[61, 62] is one specific Grid instance of the Service Oriented Computing paradigm. This section looks more closely into the various aspects of this specific instance, which are relevant from a Virtual Organisations point of view, including Grid Services (3.7.9.1) and (3.7.9.2), contract management (3.7.9.4), and trust and security management (3.7.9.5). We will come back to these different aspects in respectively Chapters 6, 4, and 8.

OGSA defines a *Grid service*—a Web service that provides a set of well-defined interfaces and that follows specific conventions.

The interfaces exposed by a Grid service contribute to

- Discovery,
- Dynamic service creation,
- Lifetime management,
- Notification, and
- Manageability;

The conventions address

- Naming and
- Upgradeability.

Every Grid service must support the GridService interface; in addition, OGSA defines a variety of other (optional) interfaces for notification and instance creation, and allows for extensible Grid service descriptions whereby architects and developers can also define arbitrary application-specific interfaces.

OGSA also offers some support for authorization and concurrency control, although these are expected to be addressed in conjunction with the Grid-aware environment where Grid services reside.

This core set of consistent interfaces, from the construction of hierarchal, higher-order services that can be treated uniformly across layers of abstraction.

Associated with each interface is a potentially dynamic set of *service data elements*. Service data elements provide a standard representation for information about Grid service instances. This important aspect of the OGSA model provides the basis for discovery (and selection based, e.g., on security and QoS criteria) and management of potentially dynamic Grid service properties.

Consistently to the service oriented paradigm, one can implement a particular Grid service—as defined by its interfaces and associated service data elements—in a variety of ways and host it in different environments.

Grid services can maintain internal state for their lifetime. The existence of state distinguishes one instance of a service from another instance that provides the same interface. The term *Grid service instance* refers to a particular instantiation of a Grid service.

The interfaces and conventions that define a Grid service are concerned, in particular, with behaviors related to the management of *transient service instances*.

VO participants often want to instantiate new transient service instances dynamically to handle the management and interactions associated with the state of particular requested activities. When the activity's state is no longer needed, the service can be destroyed.

---

[61] G rid Services for Distributed System Integration. I. Foster, C. Kesselman, J. Nick, S. Tuecke. Computer, 35(6), 2002.

[62] The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. I. Foster, C. Kesselman, J. Nick, S. Tuecke, Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.

Grid service interfaces also offer some support for authorization and concurrency control, although these are expected to be addressed in conjunction with the Grid-aware environment where Grid services reside. Because Grid services are dynamic and stateful, one needs a way to distinguish one dynamically created service instance from another. Thus, every Grid service instance receives a globally unique name, the *Grid service handle*. This handle distinguishes a specific Grid service instance from all other Grid service instances that have existed, exist now, or will exist in the future.

Furthermore, the existence of internal state can make it important that we guarantee a service has received a message once or not at all, even with failure-recovery mechanisms such as *retry* in use. In such situations, using a protocol that guarantees *exactly-once* delivery or similar semantics can be desirable, as can the protocol-binding behavior of mutual authentication during communication. OGSA is defining such bindings.

3.7.9.1 Standard Grid Service Interfaces:

OGSA defines standard behaviors and associated interfaces.

- *Discovery.* Applications require mechanisms for discovering available services, determining their characteristics, and configuring themselves and their requests to those services. In addition to the service data element, which defines a standard representation for information about Grid service instances, OGSA defines a standard operation, FindServiceData, which retrieves service information from individual Grid service instances, and a standard interface for registering information about Grid service instances with registry services.
- *Dynamic service creation.* The ability to dynamically create and manage new service instances, a basic tenet of the OGSA model, necessitates using service-creation services. The model defines a standard interface, Factory, and semantics that any service- creation service must provide.
- *Lifetime management.* Because OGSA services can be created and destroyed dynamically, they can be destroyed explicitly. They also can be destroyed or become inaccessible through a system failure such as an operating system crash or a network partition. Interfaces are defined for managing a service's life-time and, in particular, for reclaiming the services and state associated with failed operations. OGSA addresses this requirement by defining a standard *SetTerminationTime* operation within the required GridService interface for soft-state lifetime management of Grid service instances. Soft-state Protocols let OGSA eventually discard the state established at a remote location unless a stream of subsequent *keepalive* messages refreshes it. Such protocols have the advantages of being both resilient to failure—a single lost message need not cause irretrievable harm—and simple because they do not necessitate a reliable discard protocol message. Grid Service interface also defines an Explicit Destruction operation.
- *Notification.* A collection of dynamic, distributed services must be able to notify each other asynchronously of significant changes to their state. OGSA defines common abstractions and service interfaces for subscription to and delivery of such notifications, so that services constructed by the composition of simpler services can deal in standard ways with notifications of, for example, errors. Specialized protocol bindings can allow OGSA notifications to exploit various commonly and commercially available messaging systems for the delivery of notification messages with a particular QoS.
- *Manageability.* In operational settings, we may need to monitor and manage potentially large sets of Grid service instances. A manageability interface defines relevant operations.

3.7.9.2 Grid services realized as Web services associated with Resource descriptions:

Since 2001/2002 there have been a number of attempts to provide a standard based implementation framework that is able to accommodate the realization of Grid services leveraging on the success of Web services.

Until the end of 2003, the OGSI initiative within the Global Grid Forum (www.ggf.org) has been the main driver in this area. Following the release of the first stable version of OGSI specification in 2003[63], and albeit the almost spontaneous early adoption of OGSI in research and advance development projects mainly within scientific and technical computing communities, OGSI has received substantial criticism mainly on the grounds that in an attempt to provide a direct and somewhat monolithic encapsulation of all Grid service functionality, it has resulted in a monolithic and "all encompassing" framework that was adding too many proprietary extensions on the Web services standards that introduce implicit semantics, which are often confusing aspects of service orientation with the object based paradigm therefore indirectly violating assumptions about the loose-coupling of service integration that are intrinsic in the service oriented paradigm as it is realized by Web services technologies. See for example a paper named "A Grid Application Framework based on Web Services Specifications and Practices" (http://www.neresc.ac.uk/projects/gaf) by Savas Parastratidis et al. where the first version of OGSI is compared with WS-GAF that attempts to address some of the Grid computing concerns in a way that is less ambitious and closer to current Web service practices.

Criticism resulted in a fundamental revision of the concepts underpinning OGSI and a substantial decomposition of the OGSI specification in a number of autonomous building blocks which comply with current Web services practices and can be combined together in order to capture the OGSA Grid service concepts through their aggregation.

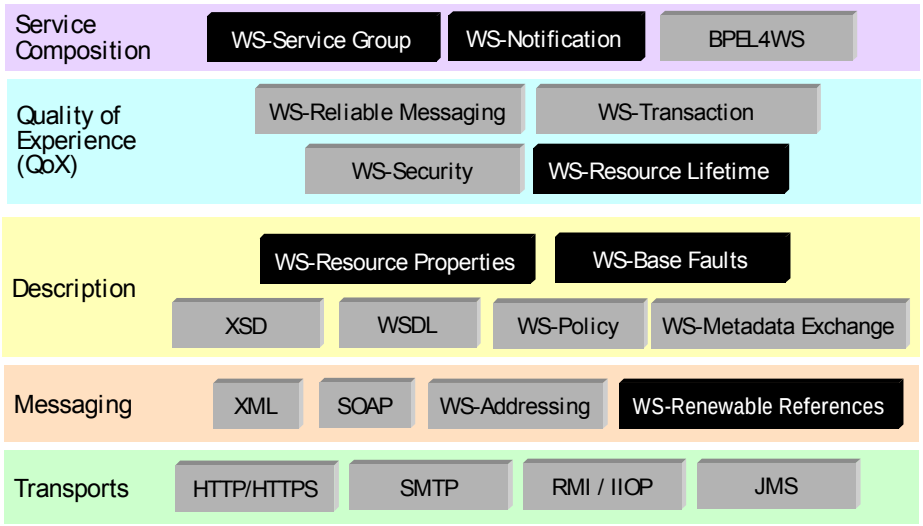
WSRF fundamentally revises OGSI in that it does not attempt to capture Grid service semantics by forcing an extension of the Web service concept. Instead, it introduces a new orthogonal concept – that of a WS-Resource[64] (not to be confused with the Semantic Web RDF resource concept) – and captures OGSA Grid service semantics by explicitly associating Web services with Resources.

The following figure summarizes the specifications constituting the WS-RF building blocks and their positioning in relation to other Web Services related standards. In the rest of this section we focus on explaining how WS-RF accommodates the realization of OGSA Grid service properties for the purpose of supporting the operation of Grid-enabled VOs. More technical details on OGSI and WS-RF are provided in subsequent chapters of this document.

[63] Open Grid Services Infrastructure (OGSI) Version 1.0. S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, D. Snelling; Global Grid Forum Draft Recommendation, 6/27/2003.

[64] Karl Czajkowski, Donald F Ferguson, Ian Foster, Jeffrey Frey, Steve Graham, Igor Sedukhin, David Snelling, Steve Tuecke, William Vambenepe. (Globus Alliance/ USC Information Sciences Institute, Argonne National Laboratory, IBM, Computer Associates International, Fujitsu Laboratories of Europe, Globus Alliance / Argonne National Laboratory, Hewlett-Packard) The WS-Resource Framework Version 1.0 03/05/2004

Positioning of WS-RF specifications against other WS standards



The WS-Resource concept has the following characteristics:

1. *It virtualizes various types of resources that may be associated with a Web service*: these may vary from physical entities (e.g.. processor, communication link, disk drive) to logical constructs (e.g. service level agreement, policy description, running task, subscription), they can be *static* (i.e. assumed pre-existing and long-lived) or dynamic (i.e. with managed life-time, including "as-needed" creation and destruction) and they can be *simple* (i.e. a singleton) or *compound* (i.e. a collection of WS-Resources).
2. *It is unique for a Web service*: it has a distinguishable local identity and lifetime.
3. *It is stateful*: it maintains a specific state that is materialized (together with other resource-specific metadata) by means of an XML resource description document.
4. *It may be accessed through one or more Web services*.

Both Web services and WS-Resources are referenced using an "Endpoint Reference". In particular, services that create or locate WS-Resources return "Endpoint References", exploiting the WS-Addressing standard. However, Web service and WS-Resource are conceptually separate and have distinct virtualizations (as distinct networked entities, with different lifetimes and different capabilities):

– A Web service is stateless
– A WS-Resource provides a context for stateful execution (that is then explicitly associated with a Web service)

The following figures (based on [65]) visualize:

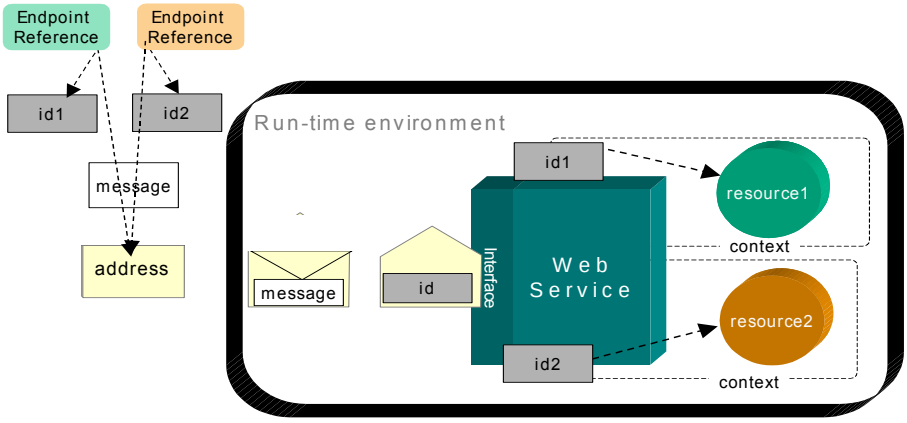
– How a requestor may access a WS-Resource via a web service by combining WS-Resource identification and the Web service address into an Endpoint Reference attached to a SOAP message,

[65] Danial Sabbah. Bringing Web Services and the Grid Together. Globus World. San Fransisco 20th of January 2004

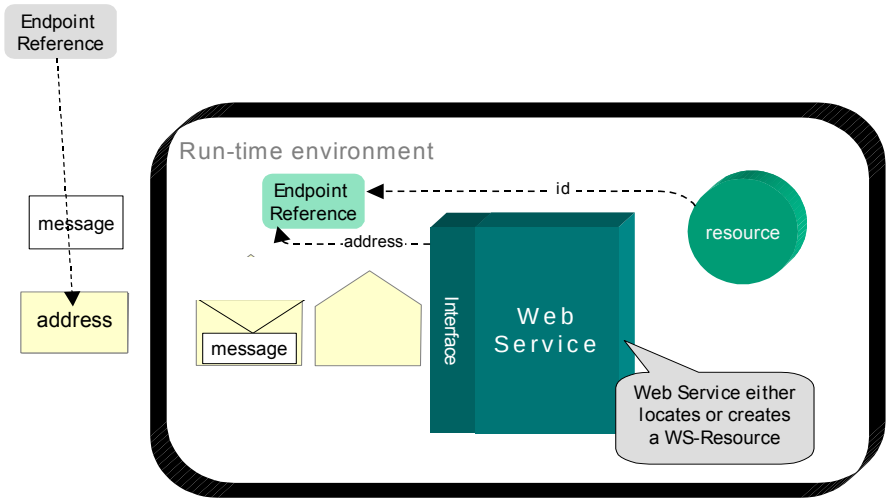
– How the Web Service accessing that resource is able to distinguish the resource and relate it to a context that is transparent to the requestor and

– How a Web service can create or locate a WS-Resource and combine its identification and the Web Service address in an Endpoint Reference, which can be then communicated to a requestor.

Using a Web service to access a WS-Resource



Creating or locating a WS-Resource



WS-Notification is used in combination with WS-Resource descriptions in order to inform interested parties about changes to WS-Resource meta-data descriptions including updates of state exposed by a WS-resource.

WS-Notification aims to enhance multi-enterprise VOs powered by Web services with a publish and subscribe messaging capability that matches the notification and event management functionality of a single enterprise. WS-Notification is based on a loosely-coupled, asynchronous messaging model that is native in a Web services context and allows one to exploit the WS Resource framework as well as other Web services technologies. The

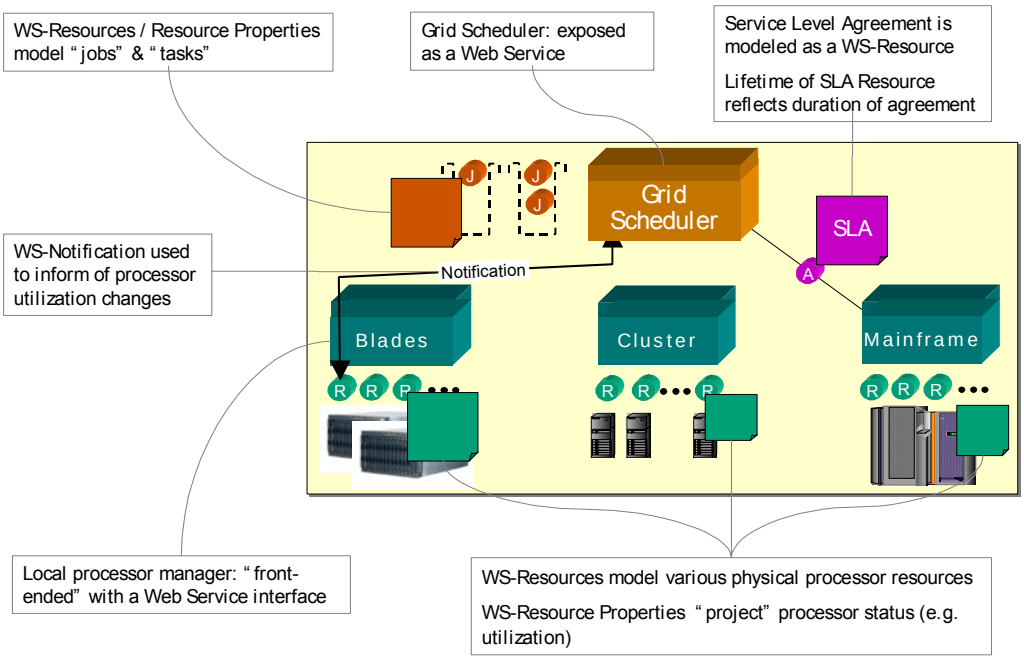
conceptual model behind WS-Notification is based on the *"subscriber"*, *"broker"* and *"publisher"* roles:

- A Subscriber indicates interest in a particular "topic" by issuing a "subscribe" request
- A Publisher produces notifications about a particular "topic"
- A Broker is an intermediary that permits decoupling Publisher and Subscriber, while ensuring that a Publisher's notifications in topics of a Subscriber's interest will reach the Subscriber. A Broker can examine current subscriptions and may also implement transformation among topics (including aggregation, decomposition and correlation of notifications) or federate in order to provide scalability.

Notably various subscriptions are possible, and although Web services can publish notifications, the publisher does not need to be necessarily a Web Service. In fact, a notification may be "triggered" by WS-Resource Property value changes. Furthermore subscriptions themselves are modeled as WS-Resources.

The following figure describes a simple Grid VO scenario where a Grid Scheduler decomposes a compute-consuming activity in a schedule consisting of a number of simpler task and jobs, which are partly outsourced to a cluster of servers and a mainframe in a different enterprises, which contribute resources and services to the VO based on already instantiated Service Level Agreements.

VO Scenario: Grid Resource Management & Scheduling



3.7.9.3 Grid hosting environments:

OGSA defines the interfaces and part of the behavioural semantics of a Grid service instance: how it is created and named, has its lifetime determined and communication protocols selected, and so on. However OGSA does not define the internal operational semantics or implementation of Grid services. For example, OGSA does not address issues such as the implementation programming model, programming language, implementation tools, or execution environment.

For OGSA Grid services to be placed in the context of a VO framework the concept of a Hosting Environment (HE) is introduced, as an abstraction of the a specific execution or hosting environment that instantiates Grid services.

HE defines not only the implementation model but it should also determine the operational semantics of a Grid service in a way that meets its obligations with respect to the OGSA prescribed external behaviour that constitutes the OGSA Grid service semantics.

By defining Grid service semantics, OGSA specifies interactions between services independent of any hosting environment. However, specifying baseline characteristics that all hosting environments must possess—defining the internal interface from the service implementation to the global Grid environment— can facilitate successful implementation of Grid services.

These characteristics would then be rendered into different implementation technologies such as J2EE, .NET, or shared libraries.

A hosting environment should address the following:

- Mapping of Grid-wide names, or Grid service handles, into implementation-specific entities such as C pointers and Java object references;
- Dispatch of Grid invocations and notification events into implementation-specific actions such as events and procedure calls;
- Protocol processing and data formatting for network transmission;
- Lifetime management of Grid service instances;
- Inter-service authentication and access control.

The following is a classification of HE that reflect different forms of ICT system integration:

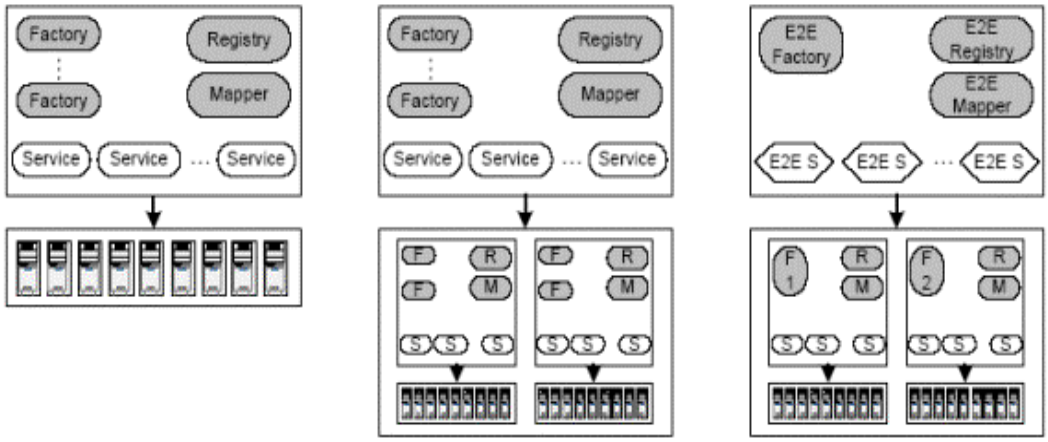
- Simple hosting environment: A simple execution environment provides a set of resources located within a single administrative domain that supports native facilities for service management. (This covers a spectrum of computing infrastructure topologies from an application server to a PC farm or a cluster). The user interface to such an environment will typically be structured as

    1. A (Grid service) *registry*,
    2. One or more *Grid service factories*, and
    3. A *Grid service handle Mapper* for mapping from a globally unique Grid service handle to binding information.

    Each factory is recorded in the registry so that clients can discover available factories.

- Virtual hosting environment: In more complex environments, the resources associated with a VO will span heterogeneous, geographically distributed *hosting environments*. Nevertheless, this virtual hosting environment, which could, for example, correspond to the set of resources associated with a B2B partnership, can be made accessible to a client via exactly the same interfaces used for the simple hosting environment.
- Collective services: One can also construct a virtual hosting environment that provides VO participants with more sophisticated virtual, collective, or end-to-end services. In this case, the registry tracks and advertises grid service factories that create higher-level service instances. Such instances are implemented by asking lower-level grid service factories to create multiple service instances and by composing the behaviors of those instances into a single, higher-level service instance.

An important consequence of OGSA's support for virtualization is that the user need not be aware of how a particular hosting environment implements OGSA interfaces and behaviors. The following figure from [62] illustrates this point, showing how a simple hosting environment, a virtual hosting environment, and collective services can implement the same interfaces.



3.7.9.4 Contract Management aspects of OGSA VOs:

In the area of contract management work within OGSA has been focusin in the area of Job execution and QoS assurance. Subsequently attention has focused on addressing aspects of Service Level Agreements in this particular context. Currently (2004) and is in general supporting the use of WS-Agreement for SLA management, albeit focusing more on real-time monitoring of SLA fulfillment and its integration with workflow execution and job scheduling mechanisms that other aspects of SLA management. Currently (2004) new set of requirements is being agreed, which is consistent to the vision of TrustCoM although over-restrictive in terms of the usage context, on which they focus. In the following paragraphs we summarise some of the key requirements, based on a current public document (draft-ggf-ogsa-spec-016, released at the end of May 2004) describing of the emerging OGSA approach in this area.

OGSA must enable submitted jobs to have coordinated access to VO resources, by automatically matching the requirements of the job with the available resources, while satisfying the resource allocation policies specified by the system administrator. Moreover, OGSA must provide functions such as monitoring the state of job execution, analysis and projection of resource usage, manageability of jobs and resources based, so that jobs can provide the desired QoS. The desired QoS is defined by an agreement between service requestor and provider. It is expected that the resources allocated to a job are dynamically adjusted based on the workload. The following functional requirements are related to this requirement.

- *Scheduling*. OGSA must enable scheduling and executing tasks based on such information as specified priority and workload of resources. It is also required to realize meta-scheduling of resources across administrative domains.
- *Lifecycle Management of Job*. OGSA must provide support for job execution throughout the lifetime of the job, including functions such as starting/stopping the job, monitoring the state of job execution, and error detection. It is required to be able to manage jobs in an integrated way without regard to the physical location or the number of the resources, even in the case that a job is using multiple distributed resources.
- *Workflow Management*. Many applications can be wrapped in scripts or processes that require licenses and other resources from multiple sources. Applications coordinate using the file system or based on events. A Fusion Grid network service is a workflow of multiple components.
- *Service Composition*. Both orchestration and choreography create higher level and possibly cross-organizational services that combine existing services. Orchestration refers to an executable process that can interact with both internal and external services. Choreography combines services at message level and includes logic and task execution ordering.

- *Service Level Agreements*. Service level assurance based on agreements is required in various domains, not limited to Grid systems. OGSA must provide standard interfaces to create and and manage agreements based on negotiation between service requester and provider.

WS-Agreement, WSLA and some advanced implementation of "proof-of-concept" SLA management systems in a Enterprise Grid Computing context (e.g. as a part of the GRASP project) are examined in the following chapter.

3.7.9.5 Trust and Security Management aspects of OGSA VOs:

There is limited work –if any– on Trust Management (in the sense that the term "Trust Management" is used in this document). On the other hand, there have been a number of loosely coordinated and often competing approaches to providing security support within OGSA. Currently (2004), a new set of requirements is being agreed, which is consistent to the vision of TrustCoM. In the following paragraphs we summarise some of the key requirements, based on a current public document (draft-ggf-ogsa-spec-016, released at the end of May 2004) describing of the emerging OGSA approach in this area.

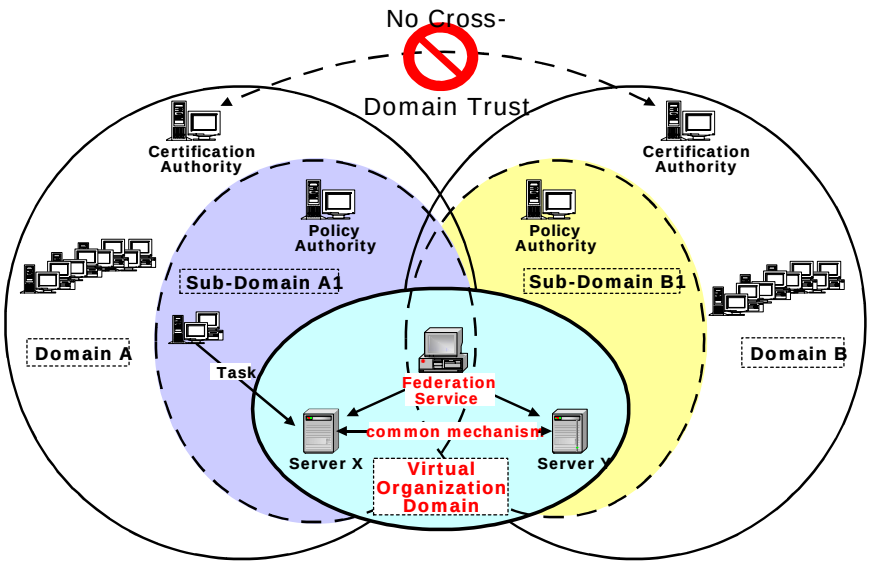
Safe administration requires controlling access to services through robust security protocols and according to some security policy. Thus, authentication mechanisms are required so that the identity of individuals and services can be established, and service providers must implement authorization mechanisms to enforce policy over how each service can be used. Mechanisms are also required for integrating and interoperating with existing security infrastructures. In addition, standard, secure mechanisms are required which can be deployed to protect Grid systems while supporting safe resource sharing across administrative domains. Sharing of resources by service users requires some kind of isolation mechanism.

The following functional requirements are related to this requirement.

- *Authentication, Authorization, and Accounting*. Obtaining application programs and deploying them into a Grid system may require authentication/authorization. The Grid system may have to identify users' security policies. Authorization should accommodate various access control models and implementation.
- *Multiple Security Infrastructures*. Distributed operation implies a need to integrate and interoperate with multiple security infrastructures. OGSA needs to integrate and interoperate with existing security architectures and models which are typically difficult to be replace.
- *Firewall Traversal*. OGSA needs standard and secure mechanisms that can be deployed to protect institutions while also enabling cross-firewall interaction without compromising local control of firewall policy.
- *Isolation*. Various kinds of isolation must be ensured, such as isolation of users, performance isolation, and isolation between content offerings within the same Grid system.
- *Delegation*. Mechanisms which allow for delegation of access rights from requestors to services are required. The authority transferred through delegation is scoped only to the task(s) intended and within a limited lifetime to minimize the risk of misuse of delegated authority.
- *Policy Exchange*. OGSA must allow service requestors and providers to exchange dynamically security (among other) policy information to establish a negotiated security context between them.
- *Manageability*. Manageability of security functionality is needed such as identity management, policy management, key management, and so forth. The need for security management also includes higher-level requirements such as anti-virus protection, intrusion detection and protection.

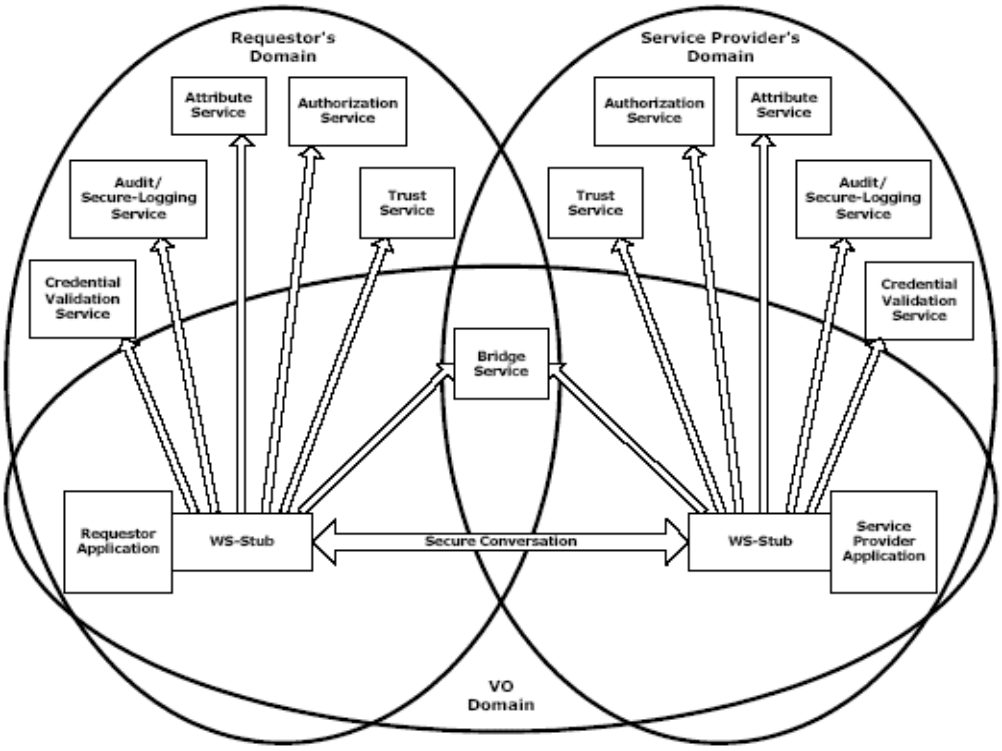
Web services and WS-Resources can be accessed over a variety of protocol bindings. Given that bindings deal with protocol and message formats, security functions such as confidentiality, integrity, and authentication fall within the scope of bindings and thus are outside the scope of OGSA proper.

Each participating end point can express the policy it wishes to see applied when engaging in a secure conversation with another end point. Policies can specify supported authentication mechanisms, required integrity and confidentiality protection, trust policies, privacy policies, and other security constraints. When invoking services dynamically, end points may need to discover the policies of a target service and establish trust relationships dynamically. Once a service requester and a service provider have determined each other's policies, they can establish a secure channel over which subsequent operations can be invoked. Such a channel should enforce the mutual agreed qualities of protection, including identification, confidentiality, and integrity. The security model must provide a mechanism by which authentication credentials from the service requester's domain can be translated into the service provider's domain and vice versa. This translation is required in order for both ends to evaluate their mutual access policies based on the established credentials and the quality of the established channel. OGSA's security model must address authentication, confidentiality, message integrity, policy expression and exchange, authorization, delegation, single logon, credential lifespan and renewal, privacy, secure logging, assurance, manageability, firewall traversal, and security at the WSRF layer. We can expect that existing and evolving standards will be adopted or recognized in the Grid security model.



Proposed OGSA approach:

Use Virtual Organization as Bridge



The above figure (from the current working draft of the OGSA specification, as of May 2004) shows relationships between a requester, a service provider and many of the security services. Note that both requester and service provider are always subject to the security policies dictated by their respective administrative domains. Furthermore, a VO can have its own security policy that can enable the sharing of the submitted resources, but the associated rights will always be capped by the overruling resource-local policy. For many Grid applications, the resource owners and the individual requesters will not “know” each other as they live in different administrative domains while their interactions are dynamically discovered and brokered by scheduler services and such. This implies that trust has to be dynamically established through introductions, and the concept of the VO as a bridge is seen as an important tool to build these dynamic trust relationships.

At present there are no mature Grid security solutions that support all of these requirements. The extend to which current Grid security solutions are interoperable and support a common of these requirements is also unclear. We examine some of the more prominent approaches in a subsequent chapter dedicated to already established and emerging Grid Security solutions.

3.8 Conclusions

From our investigation, there are no frameworks committed to the exclusive modelling of trust, contracts and security in VOs. The existing frameworks have been for the most part concerned with structural compatability, process interoperability, roles and relationships across multiple organization domains, or very broad notions of constraints on actors in the VO. Trust, Contracts and Security are mostly mentioned as orthogonal interests. In addition, these models and reference architectures have focussed on either reaching a conceptual understanding of VOs, enabling requirements derivation, or providing a framework for enacting processes of the VO. In TrustCoM we seek to cut across this spectrum by conceptually understanding trust, contract management and security in VOs, deriving their requirements and hence providing functionality to support our reference implementations. We will therefore not select one "champion framework" initially, rather consider the most appropriate as the project advances.

There is a wealth of reference models to which we can refer, in order to provide a basis for concepts throughout the project. They all present the VO as a multidimensional and multilayered phenomenon, with both structural and dynamic views. However, most models weakly depict the influence of socio-economic factors and legal aspects on the VO, and focus more on technology and operation-related views such as physical, functional, decisional and informational. Nevertheless, there is a common understanding throughout all the reference models pertaining to the lifecycle of the VO; all models in some way showed the VO going from a stage of Identification to Formation to Operation and finally to Decommission. The GERA model also broke down the formation phase into concept, requirements, designs and implementation, as it has a clear influence by collaborative engineering.

Aspects of Rosettanet appear to be potentially useful for TrustCoM. However the whole framework as such is rather heavy-weight and inflexible for the purposes of this project. Service Oriented approaches and in particular OGSA appear to be more relevant for the dynamic Virtual Organisations envisaged in TrustCoM. Also the VO security requirements that are now (2004) being put forward by OGSA in their lates communications are in line with the vision of the TrustCoM project. Unfortunately, however, OGSA is still under development and there is no concensus about already adopted standards and widely accepted reference implementation that meets these requirements. TurstCoM should monitor however the evolution in both OGSA and Web Services in order to influence their direction and avoid duplication of work.

Even if Serivce Oriented Compuitng paradigms and OGSA may provide the basis of an ICT framework, TrustCoM should also carefully select aspects of “softer” VO frameworks which adqueately address enterprise modelling and business management aspects of Virtual Organisation frameworks.

4 Contracts and Service Level Agreements

Edited by: Babak Sadighi[1] and Theo Dimitrakos[2]
[1]Swedish Institute of Computer Science (SICS)
[2]Council for the Central Laboratory of the Resarch Councils (CCLRC)

4.1 Introduction

This chapter focus on existing approaches to Business-to-Business (B2B) contract management and Service Level Agreement (SLA) Management systems supporting two types of contract that have been identified by the Community as key enablers for VOs[66]:

- Frame Contractual Agreements, including Collaboration Agreements
- Customer-Supplier Contracts, including Service Level Agreements

A contract management system provides mechanisms for specifying unambiguous agreements between the partners as well as mechanisms for secure creation of contracts including negotiation of contractual terms and secure signing of contracts. Furthermore it provides mechanisms for monitoring contractual performances of the parties and preventing repudiation of performances and non-performances and finally it provides mechanisms for activating obligations and enforcing remedial actions in case of violations.

To the extend that this is possible, we underline the relationship of the approaches evaluated with emerging industry standards such as ebXML Trade Partner / Collaboration Agreement, WSLA and WS-Agreement, both in terms of the unpinning architectural models and of the corresponding specification profiles.

4.2 Service Level Agreements

Service Level Agreement is a formal negotiated agreement between a service provider and his customer. When a customer orders a service from a provider, an SLA is negotiated and then a contract is drew up. The service provider must perform a SLA monitoring in order to verify whether the QoS parameters specified in the SLA contract are respected. The SLA monitoring involves monitoring the performance status of the offered service and provides relevant information to the service level management system. Then, the system management could assess the provider's commitments and applies penalties if those commitments weren't met.

More generally, another feature of WS related to business process is the service level management: it is the management of a single or a group of WS within the same domain or business field. It consists of the monitoring of operations of the WS and the control of the WS to meet the guaranteed service and QoS.

In order to define such provider's commitments, a lot of specification works has been recently carried out defining several XML based languages enabled to describe the contract between the service provider, his customer and a possible third party. These languages were defined closely to the common language allowing a common understanding of the service provider commitments to perform a service according to agreed guarantees. Several

[66] These categories of contracts have been identified as key constituents of a VO contractual Frameworks by the workings of the VO Industrial Interest Group (VOIG) of the Virtual Organisation Cluster VOSTER. (VOSTER is a European Project Cluster funded under IST-2001-32031 that brought together over 24 projects in the period 12/2001-05/2004 with an aim to "collect, analyse and synthesize the results of leading European research projects on Virtual Organisation". See http://cic.vtt.fi/projects/voster/)

languages have been defined (WSLA specification language [67,68], etc), all of them are a complement to the service description implemented by WSDL [69]. In general, such languages are used within a framework allowing the management of Web Services and their compositions.

In the following we present an overview of recent works most relevant to GRASP project dealing with Service Level Agreement in WS context, and we also present the OGSI-Agreement specifications (WS-Agreement).

4.3 Web Service Level Agreement (WSLA)

4.3.1.1 The language

The WSLA Language, developed by IBM[67,68], defines a type system for various SLA artifacts and it is based on XML schema. The Service Level Agreement must be an automated process, in order to facilitate such feature an SLA in the WSLA contains three parties:

- Parties' section: identifies the actors involved in the contract agreement. It distinguishes the signatory parties (signed the contract) and the supporting parties (sponsored parties), which provide services to the signatory parties as measuring the service's parameters. This section contains the contact information related to each party (representative, address, …).
- Service Description Section: define the SLA parameters of the service. These parameters are defined by Metrics, which define the way to measure and evaluate them (measurement directives). This description also specifies how to aggregate metrics to composite metric.
- Obligations Section: this last section of an SLA define the obligations and actions guaranteed in case of violation of Service Level Objectives (SLO) respective to the state of the related SLA parameter. The SLO contains the guaranteed condition of a service for example sending a notification on violation events.

4.3.1.2 Standard extension

WSLA language provides mechanism to extend its various types in the core WSLA language, defined as abstract. This abstraction allows defining new specific domain from existing language elements, using XML schema derivation such as Business topics. This extension provide to the authors of an agreement the capability to facilitate relating the WSLA to WSDL defined service and WSDL defined management actions, defining common metrics in the context of Web Services and a set of standard predicates to define the contractual guarantees.

4.3.1.3 Runtime Architecture

The following figure represents the WSLA monitoring framework building blocks Figure 25.

[67] Ludwig, H., Keller, A., Dan, A., King, R.P., Franck, R.: Web Service Level Agreement (WSLA) Language Specification, Version 1.0, Revision wsla-2003/01/28. International Business Machines Corporation (IBM). On-line at:http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf (2003)

[68] Keller, A., Ludwig, H.: The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management, Vol. 11, No 1 (Mar. 2003) Plenum Publishing (2003)

[69] http://www.w3.org/TR/wsdl

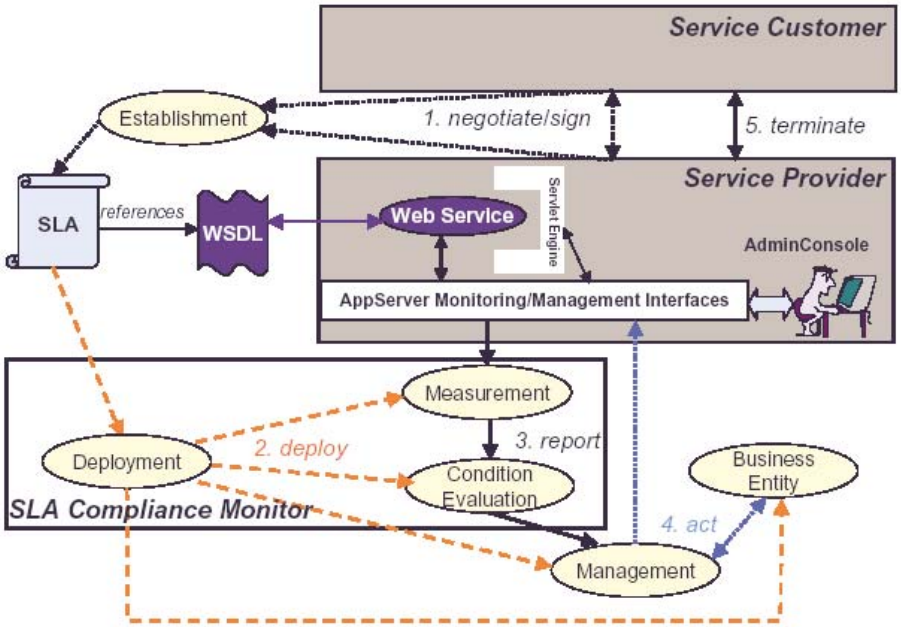


Figure 25 Runtime Architecture

The interface of a Web Service is defined by an XML document in the WSDL [69]. The SLA references to the WSDL document and extends the service definition with SLA management information. The SLA management lifecycle consists in five distinct phases :

Phase 1: the negotiation:

This phase consists of the drawing up of an agreement between the service provider and the customer for a service offered. In this phase the two parties establish different aspects of the SLA : QoS parameters, price, third parties and their roles, and so on … Actually, they write down on the SLA document all elements of the contract making it available for deployment.

Phase 2: the deployment:

This service allows the WSLA interpretation and the set up of the corresponding components managing the SLA. Each party (Service Consumer and Provider) is responsible for the deployment its functions and the set up of his supporting parties if necessary. This information has to be passed on a standard format to a party to another.

Phase 3: Measurement and Condition Evaluation Service:

This phase deals with controling during the runtime system wheather QoS promised by the provider are reached in monitoring the SLA parameters and evaluating the violation of the SLO.

The measurement Service collects runtime information on the metrics related to a SLA parameters. Multiple measurements services may simultaneously measure the same metrics. It notifies to the Condition Evaluation Service the results of the monitoring.

The goal of the Condition Evaluation Service is to determinate the probable violation of guaranteed QoS regarding the collected metrics. This can be done each time a new value is available, or periodically.

Phase 4: Corrective Management Action

Once a SLO parameter has been violated, corrective actions contained in the SLA need to be carried out. This functionality apply a two-pronged services:

Management service:

Following a recept of violation of SLO notification, this service must carry out the corrective actions described in SLA document. Before acting, it requests the business entity in order to verify if the proposed actions are allowable.

Business entity:

It contains the business policy of the service provider and verify if the corrective actions specified in a SLA, some time ago, are still in accordance with the business targets.

Phase 5: SLA Termination:

The conditions of termination of the service must specify in the SLA document. These conditions must describe the penalties in case of breaking one or more SLA clauses. This could be negotiated between service provider and consumer in the Negotiation phase.

The "SLA compliance Monitor", tool has been developed for the WSLA language by IBM and it is included in the IBM Web Services Toolkit.

The WSLA framework enables specifications of detailed SLA parameters, management information of the system, price/penalties for Web Services, nevertheless the monitoring resource is not already well-specified. Another WSLA design goal, is to address the "wide variety of SLAs" in providing SLA templates document which include several automatically processed fields limited to a small set of variant services using the same kind of SLA parameters.

The integration with existing resource management is work still in progress and special attention is paid on Common Information Model (CIM) .Actually, The WSLA infrastructure is powerful but very complex to perform.

4.4 Web Service Modelling Framework

This approach developed by HP[70,71] is closed related to the previous; the aim is to define a precise, unambiguous and flexible specification of service level agreements management system for WS. It focuses on the definition of SLA between two Web Services described as a contract that consists on automated process offering guarantees by one Web Service (the supplier) to another (consumer). This approach is fully compatible with WSDL and WSFL, and the grammar of the language has been defined in XML.

4.4.1.1 Language definition

An SLA specified with this approach contains the following information:

An SLA has a date constraint that consists on start and end time and the next evaluation time.

[70] Sahai A. et al, "Towards automated SLA Management for Web Services" http://www.hpl.hp.com/techreports/2001/HPL-2001-310R1.pdf

[71] Sahai A. et al, "Specifying and Monitoring Guarantees in Commercial Grids through SLA" http://www.hpl.hp.com/techreports/2002/HPL-2002-324.html, 2002

A set of SLO, that also has daytime constraint and a set of clauses. Each clause is based on measured item and functions needed to evaluate, at defined interval, the violation of the SLO on a set of samples of the measured item.

The measured item, specified within a clause, states the name of one or more monitored items across which the same measurement will be applied. Each monitored item contains information on the construct type and reference of the item and the location where the measure will be carried out.

The following Figure 26 gives an overview of the above description:

```
SLA  = Dateconstraint Parties SLO*

Dateconstraint = Startdate Enddate Nextevaldate

SLO = Daytimeconstraint Clause*

Dateconstraint = Day* Time*

Clause = MeasuredItem EvalWhen EvalOn EvalFunc EvalAction

MeasuredItem = Item*

Item  = MeasuredAt ConstructType ConstructRef
```

Figure 26 A SLA Example

4.4.1.2 Instrumentation of business process

Based on the previous specification, this approach defined a SLA Management Engine (SME) [70] that constitute a generic component that manages the overall process, collecting measured data from SLA Monitoring (Instrumentation Manager) component, storing through the Management Handler (service provider side) or Communicator (service consumer side) components these measurement data in a model repository that could be a database. The SME process controller (Business Process Management Agent) receives the SLA and carries out monitoring process flow in order to customize the SLA alarms (SLO section contains constraints). The “SLA Customizer” creates an SLO Object in the SLA repository. The SLO Object maintains the state of the SLO (valid, active, violated). The SLO evaluator component determines the compliance or violation of the SLA. The SLA violation engine maintains all the features of violation event in the database (clauses violated, level of violation, …). The figure below presents a detailed vision of the SME process controller:

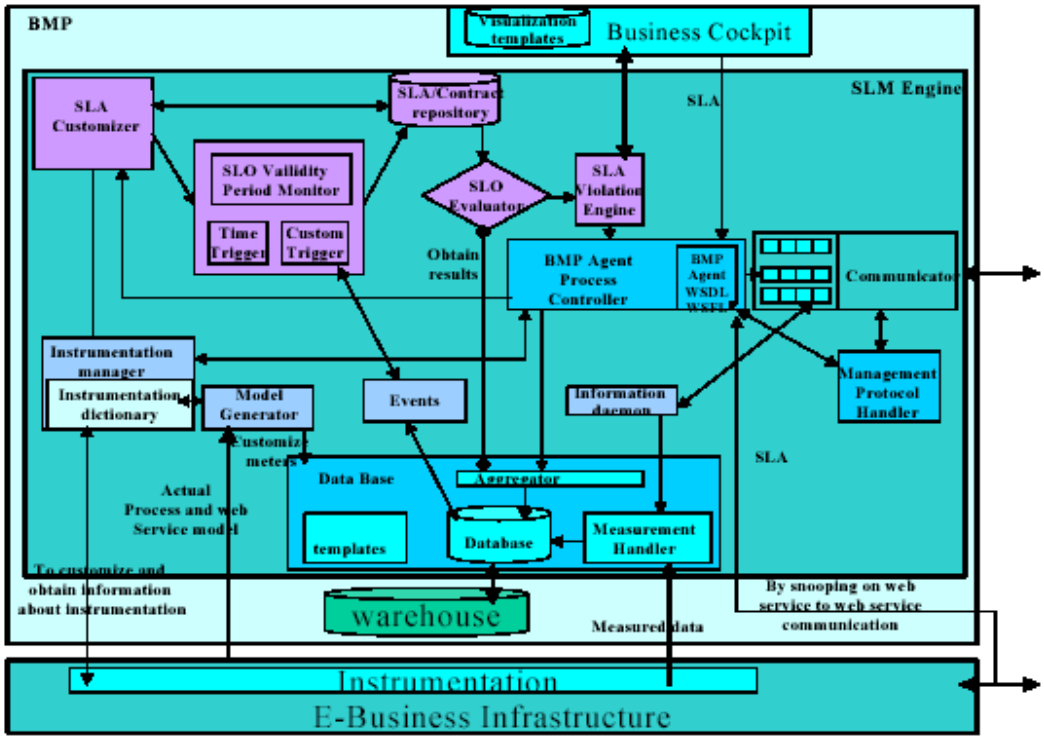


Figure 27 SLM Engine

Several tools have been developed for the HP SLA approach and it is used in the HP Openview Web Service Management Engine (WSME). A tentative to consider HP Utility data Center as a typical commercial Grid deployment environment had been experimented.

Like the preceding one, this approach, developed in an e-business context, enables specification of detailed parameters of SLA, and described all the necessary aspects of a suitable management infrastructure. However it presents an important centralized vision of the information system managing the whole of the data of the SLA.

4.5 SLAng

SLAng, SLA notation generator[72,73,74], is a XLM-based language designed in order to produce a formal language with a well defined syntax and semantics, for describing Service Level Specifications (QoS parameters) (SLS) in the domain of distributed systems and the context of e-Business. This work has been carried out by the TAPAS project[75]. The approach of SLAng is to define a such SLS at different levels: the application level as well as the application service (ASP), systems resources and so forth.

[72] Lamanna, D.D., Skene, J., Emmerich, W.: SLAng: A Language for Defining Service Level Agreements. In Proc. of the 9th IEEE Workshop on Future Trends in Distributed Computing Systems - FTDCS 2003 (Puerto Rico, May 2003). IEEE-CS Press (2003) 100-106

[73] Skene, J., Lamanna, D.D., Emmerich, W.: Precise Service Level Agreements. International Conference on Software Engineering (ICSE) 2004

[74] Lamanna, D.D., Skene, J., Emmerich, W.: Specification language for Service Level Agreements. Document submitted to EU IST Project 34069 TAPAS. March 2003.

[75] http://www.newcastle.research.ec.org/tapas/

4.5.1.1 SLAng model

SLAng introduces a reference model for inter-organisational service provision at storage, network, middleware and application level for a distributed component architecture.

The following figure shows the depicted model:

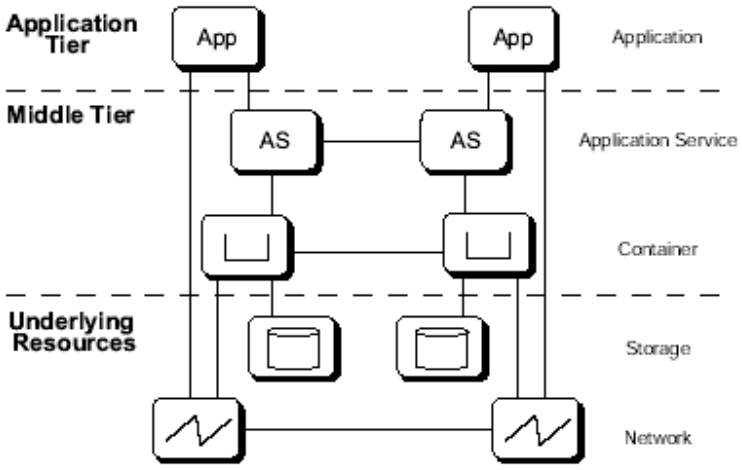


Figure 28 Service Provision Reference Model

This traditional layered architecture points out that service provisioning could occur at any level of the architecture, different parties could provider or consume services.

SLAng defines six different types of SLAs: three vertical and three horizontal. These types regulate the type of agreement occur between the different type of parties of this architecture.

The vertical ones are:

- Hosting: between service provider and host
- Persistance: between a host and storage service provider
- Communication: between container and network service provider

The horizontal are:

- Service: between an Application or service and ASP
- Container: between containers providers
- Networking: between network providers

This cross-layered architecture aims at taking into account the whole interactions cases between the actors taking share in an e-business model.

4.5.1.2 SLAng Language and semantic

As previous languages, SLAng syntax is based on XLM schema. This language has been modelled by the use of Unified Model Language (UML) and Object Constraint Language in order to precely defines the meaning of service level-agreement.

The description of an SLA in SLAng has the following structure according the different types of SLA below:

- End point description of the contractors containing information on provider-customer,

- Contractual statement define the agreement itself: kind of service, duration of the agreement, charging clause, violation clauses, and so on
- Service Level Specification describe the QoS parameters related to the service and the metrics associated.

The notion of a mutual responsibility between the customer and the provider of a service is defined in each SLA in order to take into account the bilateral aspect of the contract.

This approach enables an SLA management in e-Business context and distributed systems and focuses not only Web services issues but also on different types of SLAs. It has a broader scope compared to IBM and HP approach. But the detailed mechanism to create a Service Level Agreement is not described by SLang. Another aspect of SLang, the definition of the QoS metrics are defined into the SLang schema, thus only predefined SLA format could be used. SLang seems less flexible than the others approaches.

The TAPAS project supplements the Slang approach by implementing the QoS functionalities of the architecture using J2EE technologies, in particular with JBOSS and JONAS.

4.6 Web Service Offering Language (WSOL)

WSLO (Web Service Offering Language) is a language for specification of constraints and classes of service for Web Services[76,77,78]. The syntax of WSOL is defined by using XML schema. WSOL is fully compatible extension of WSDL.

This language allows to enable a Web Service to offer several different “Classes of service” to consumers that means a Web Service could provide different service levels defined by several classes of service. Classes of service can differ in usage privileges, service priorities, response time guaranteed to consumers, etc… The classes of service are defined at the level of Web Service and not QoS constraints.

WSOL defines the following constructs:

Constraint: three types of constraint are defined:

- Functional constraints are specific to the execution to a Web Service operation to be functionally correct. WSOL enables specification of several functional constraints such as pre-, post- and future conditions, as well as invariants[78]. A third party can evaluate functional constraint.
- QoS constraints (non functional) are specific to an operation invocation such as performance, reliability, and so on. Non-functional constraints check whether the monitored QoS metrics are within specified limits. These constraints are described by an external ontology of QoS metrics and measurement units. These external ontologies definitions can be reused for different Web Services.

[76] Tosic, V., Ma, W., Pagurek, B., Esfandiari, B.: Web Service Offerings Infrastructure(WSOI) – A Management Infrastructure for XML Web Services. Submitted for conferencepublication. Also published as: Research Report SCE-03-19, Department of Systems andComputer Engineering, Carleton University, Ottawa, Canada, Aug. 2003. On-line at:http://www.sce.carleton.ca/netmanage/papers/TosicEtAlResRepAug2003.pdf (2003)

[77] Tosic, V., Pagurek, B., Patel, B. Esfandiari, B., Ma, W.: Management Applications of the Web Service Offerings Language (WSOL). In Proc. of the 15th Conference On Advanced Information Systems Engineering - CAiSE’03 (Velden, Austria, June 2003). Lecture Notes in Computer Science (LNCS), No. 2681. Springer-Verlag (2003) 468-484.

[78] Tosic, V., Patel. K., Pagurek, B.: Reusability Constructs in the Web Service Offerings Language(WSOL). In Proc. of the Workshop on Web Services, e-Business, and the SemanticWeb (WES) at CAiSE’03 (Velden, Austria, June 2003). LNCS, Springer-Verlag. Extendedversion published as: Res. Rep. SCE-03-21, Department of Systems and Computer Engineering,Carleton University, Ottawa, Canada, Sep. 2003. On-line at:http://www.sce.carleton.ca/netmanage/papers/TosicEtAlRepSeptember2003.pdf (2003)

- Access rights constraint specifies conditions under which any consumer has the right to use any invoked operation in the service offering.

Management statement: the statement construct enables the specification of important information about the represented class of service offering. Three main XML schemas for types of management statement are defined:

- Price statement: cost of the use of a particular operation of the offering service.
- Monetary penalties statement: amount of money the service provider have to pay to the consumer whether it could not fulfil the constraints in the service offering.
- Responsibility statement specifies the management responsibility of a management entity. A management entity could be the service provider, the service consumer or a third party.

Service offering (SO): WSOL service offering contains the formal representation of various constraints and management statements that determine the corresponding class of offering. WSOL service offering can be viewed as one simple contract or one SLA between the service provider, consumer and eventually a management third party trusted by supplier and consumer.

Reusability elements: WSOL gives the possibility to reuse constraint and management statement constructs enabling easier specification of a new service offering from existing service offerings of the same Web Service or other Web Service by using inheritance, inclusion or template instantiation. WSOL defines several special reusability elements [78].

Service offering dynamic relationship (SODR): SODR states what service offering could be replaced by another whether constraints from the used service offering cannot be fulfilled. These relationships are specifies in a file outside of the WSOL file in order to avoid frequent modifications of the service offering definitions.

The WSOI infrastructure also integrated, on top of Apache Axis, solutions for monitoring WSOL service offerings. In WSOI, monitoring and accounting activities are developed through a specific handlers [77].

This infrastructure wants to be easily adaptable thanks to the reusability constructs, which can offer a comparison between different Web services, and especially a reusability for SLAs. This confer a means to compare the Web Services between them. But, this approach reduces any possibilities to enable instantiation of SLA on-demand, the concept of reusability involves a static definition of the SLA (predefined) for a service given. However, it provides solutions that are relatively simple to use and implement and lightweight in terms of run-time overhead.

4.7 SNAP

Grid computing is about exchange of computational power in multi-domain settings, where organisations create virtual organisations in which they provide various kinds of network and computational resources to each other's members. The Grid Computing paradigm was originally developed by academics for exchange of computational resources for academic projects. However, the idea has been generalized and it is nowadays a paradigm for exchange of computational resources in commercial context. The commercial applications of the grid computing paradigm require much more sophisticated mechanisms for creating, monitoring and enforcing commitments. This level of requirements is not needed by the academic applications of grid computing where "best effort" provisioning by service providers is an accepted level of quality of service by the consumers. SLAs in the Grid Computing paradigm are mainly used for resource management in multi-domain settings (virtual organisations) where the domains are independently managed.

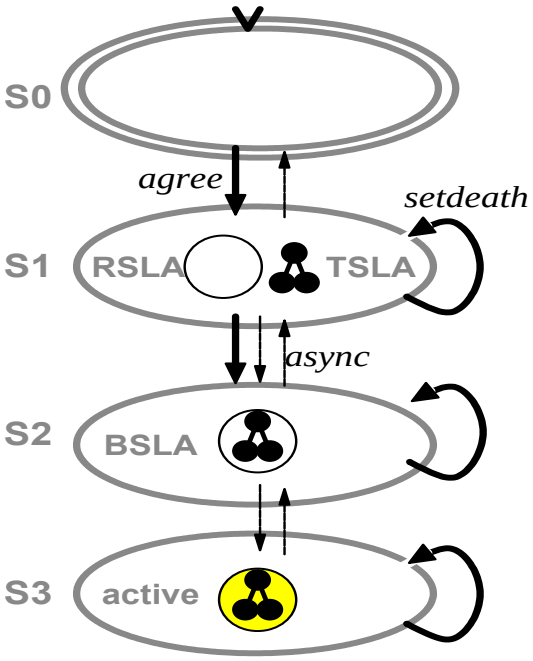
Here we describe briefly one of the approaches for SLA management in the field of grid computing. The first approach we describe here is called SNAP: A Protocol for Negotiation

of Service Level Agreements and Coordinated Resource[79]. SNAP is a resource management model based on the following three types of SLAs:

1. Resource acquisition agreements (RSLA),
2. Task submission agreements (TSLA) and
3. Task/resource binding agreement (BSLA).

The first type of agreement is about the right to use a resource or making resource reservation, the second type is about performance of an activity or task or a promise to execute a job, and the third type is about use of resources for a certain task. Based on these agreements one can submit a job, get promises for resources and bind these resources to the job.

SNAP is a simple 2-party protocol for negotiating SLAs. It has transitions between 4 states as shown below:



- S0: Start with no SLAs
- S1: Create SLAs
  - TSLA or RSLA
- S2: Bind task/resource
  - Explicit BSLA
  - Implicit provider schedule
- S3: Active task
  - Resource consumption
- Backtrack to S0
  - On task completion
  - On expiration
  - On failure

Figure 29 Agreement State Transitions

An SLA in SNAP is a tuple of the form $\langle l,c,t_{dead},d \rangle$ where ***l*** is an identifier for the SLA, ***c*** is its client, t_{dead} is the expiration time of the SLA and finally ***d*** is the description of the SLA in terms of a specific RSLA, TSLA or BSLA.

The description ***d*** is specified in terms of an extensible language **J** for describing tasks and a subset language $\mathbf{R} \subseteq \mathbf{J}$ for specifying resources.

[79] K. Czajkowski, I. Foster, C. Kesselman, V. Sander and S. Tuecke, SNAP: A protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems, 8th Workshop on Job Scheduling Strategies for Parallel Processing, Edinburgh, Scotland, July 2002.

Language **R** is constructed using resource metrics: time, scalar and the max and the min limits of the given scalar metric. Resources can of course be of composite type modelled in terms of simpler resources. The compositions are made using various typed constructs.

SNAP can be seen as a first approach towards an SLA framework for the grid. It does not consider many important features of an SLA and in particular it does not give any treatment of agreement violations. Therefore SNAP does not deal with monitoring performances to validate the satisfaction of SLA agreements.

4.8 WS-Agreement

In June 2003 the GGF released Version 0 of the OGSI-Agreement specification, which proposed a general agreement based management of Grid Service instances (http://www.globus.org/research/papers/OGSI_Agreement_2003_06_12.pdf). This initial version suggested the usage of Agreement Grid Services. Beside the fact that OGSI-Agreement, Version 0, naturally was quite general and lacked concrete implementation approaches, Agreement creation was done by invocation of Factory::createService with appropriate arguments, leading to a fault or the creation of a new service. Thus Negotiation and Instantiation were logically coupled.

Recently there has been a significant evolution of OGSI-Agreement that culminated in the release of Version 1.0, now called WS-Agreement t[80]. In the following we will give a short overview on the key concepts of WS-Agreement. In general, WS-Agreement aims to define a language for negotiation and monitoring of agreements between a service client and a service provider. The assumed key requirements are:

- Description of agreements about services independent of the domain
- Creation of agreements about single services as well as collaborating services
- Support for different condition languages used to define service level objectives or constraints
- Independence of the negotiation model
- Combination with any WS-* (WS-Policy, WS-Addressing, …) specification

As a consequence of the extensibility requirements above, specific condition expressions, service descriptions and metric definitions languages are outside the scope of WS-Agreement. Figure 39 shows the structure of an Agreement, which is built upon the following elements:

- Context – contains the participants and other information such as the termination time of the agreement
- Service Description – contains domain specific service descriptions and necessary information to interact with the service instance
- Guarantee Terms - includes the condition collection under which a service will be executed, often referred to as Service Level Objectives
- Negotiability Constraints – may describe rules for the negotiation phase

[80] https://forge.gridforum.org/projects/graap-wg/document/WS-AgreementSpecification/en/2

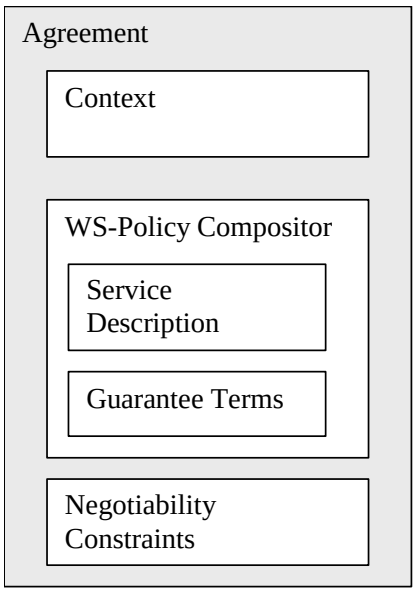


Figure 30 Agreement Structure

The *service description* is what a service provider agrees to provide, specified in a domain-specific way using a domain specific description language.

The *guarantee terms* specify the actual service levels that the parties have agreed on, in other words the values of the service attributes given in the service description. A guarantee term consists of three parts: qualifying condition, service level objective and business value. The qualifying condition may be based on service attributes or external factors such time. A service level objective is a set of the values of the service parameters that have to be met by the service provider. The business value is specifying a priority on the objective either by defining an *importance* value on the objective, or by specifying a penalty as the consequence of violation of the service level objective. The management system uses the guarantee terms to monitor the service and enforce the agreement.

The negotiability constraint can be given by a party to restrict the number of offers to be exchanged between him and other parties during the negotiation phase of the agreement. It is not by any means interpreted as a promise by the party but the values that it generally is willing to accept.

In the version 1.1 of the WS-agreement specification dated 2004-04-26, the conceptual model is based on two layers, the service and the agreement layer, as shown in the Figure 31, below.

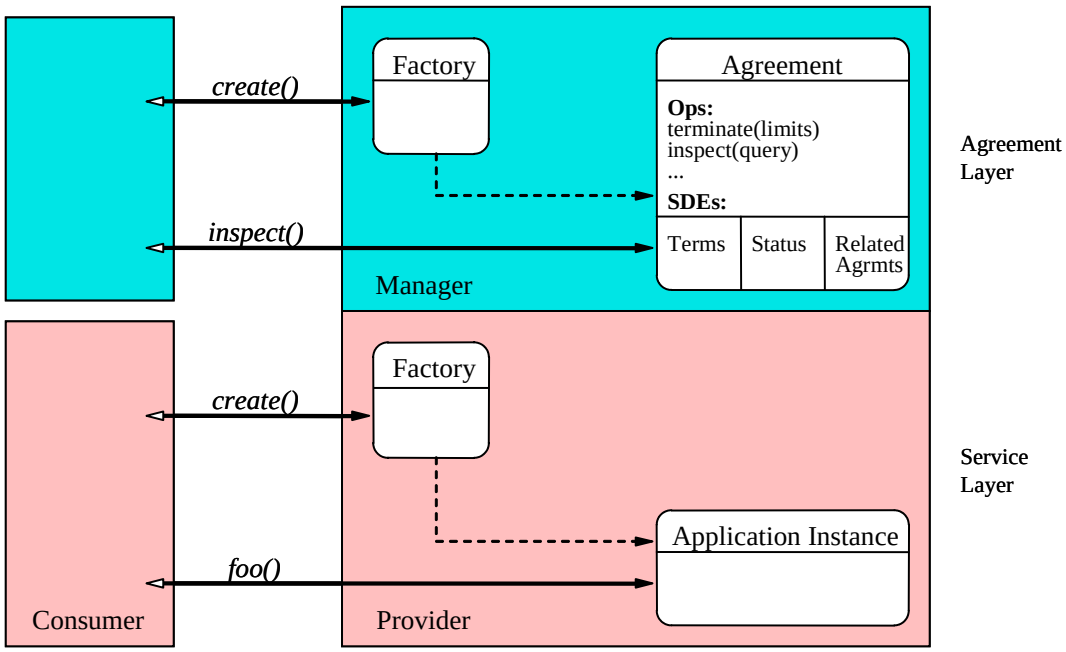


Figure 31 WS-agreement Conceptual Layered Service Model

The service layer is simply the application-specific layer of business services being provided. These services may or may not be specified by web service interfaces.

The agreement layer is a web service-based interface that can be used to represent and monitor agreements with respect to provisioning of services implemented in the service layer.

4.9 GRASP SLA Management Infrastructure

The European GRASP project[81] is prototyping a Grid-enabled platform for the next generation of Application Service Providers. Commercial usage of Grid platforms that have been developed following the Service-Oriented Architecture (SOA) paradigm is typically connected with a demand for assured Quality of Services (QoS). In this section we present the SLA Management Infrastructure that is being developed by the GRASP consortium where CCLRC and HLRS are respectively in charge or Architecture and Implementation work-packages. The GRASP architecture[82] is being implemented over Microsoft .NET, and leverages on WSLA, Microsoft WSE extensibility for .NET and the Open Grid Services Infrastructure implementation on .NET provided by the University of Virginia (USA). In the context of the European GRASP project the focus is on the provision of value-add administrative services for Grid service Orchestration, Contract Management, Enterprise Security, and Accounting and Billing.

This section is organised as follows: We first identify and discuss some general requirements and logical building blocks of a business related SLA Management Infrastructure. We then give a more detailed description of how these are realised by

[81] Dimitrakos T., D. Mac Randal, F. Yuan, M. Gaeta, G. Laria, P. Ritrovato, B. Serhan, S. Wesner, K. Wulf, *An Emerging Architecture Enabling Grid-based Application Service Provision.* Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference, IEEE Press 2003.

[82] GRASP architecture design – Deliverable 24 – GRASP consortium: www.eu-grasp.net

GRASP components. Other sources of information about GRASP security and contract management subsystems include[81,83].

4.9.1.1 Logical building blocks

Service Level Agreements in the context of GRASP are agreements between a service provider and a service consumer about the quality of the delivered services. In many domains QoS can be expressed in technical metrics like available bandwidth or response time. Such raw data can be usually measured through an operational system API. In many cases QoS also take specific calculations on this raw data into consideration. A typical function would be calculating an average value over a given time. This sort of time dependency already increases the complexity of a SLA monitoring system, but inside the business domain there are two further notable extensions of QoS. First it is fairly likely that a SLA between an ASP and a business customer will contain a composition of different QoSs, even in case of only one grid service. Second, as mentioned above, the kind of constraints will not only be on a technical level but will also include dynamic behaviour dependent on business rules. For example 'between 9.00 – 20.00, apart from weekends, task execution time must be less than 3 ms.' Thus we must be able to map abstract rules to measurable metrics.

In conclusion we see that a service level agreement in the ASP domain of GRASP must bring together raw data, calculated data and business rules to decide if a given SLA has been violated. We therefore state the demand for a sophisticated and extensible *Service Level Description Language SLDL*, in order to handle this infinity of possible ASP related Service Level Agreements. A more detailed discussion about the necessity for such a language can be found in [84]. It is clear that developers will need supporting tools to handle SLDL constructs. We summarize these tools under the term *SLA Parser.*

While the SLDL is the base for definition of SLAs, it does not say how this SLAs are used by service provider and service user. As the dynamic discovery and instantiation of Grid Services based on functional and SLA search criteria is a common scenario, we can generally assume that a Service Provider will offer a service in conjunction with a somehow limited set of possible service levels regarding this service. The totality of these service levels must be described with the SLDL and stored in a kind of repository. The latter can be considered as base offering from the Service Provider to the Service Consumer regarding a specific service. During a negotiation phase they agree on the utilization and the expected QoS and fix the contract inside the SLA of this service. We therefore identify two additional necessary logical building blocks. The *PRE-SLA Pool* that contains templates for all the SLAs that could be supported by a service provider and a *SLA Negotiator* that is able to take charge of a negotiation phase between the Service Consumer and the Service Provider. If service provider and service consumer agree on a specific SLA, the Grid Service will be instantiated and must be monitored regarding the agreed SLA. As mentioned above, the monitoring will be based on raw data, calculated data and rules.

The appropriate logical building blocks are some kind of *Data Providers* as source of raw and calculated data and a *SLA Monitor* that is able to apply functions and rules to the collected data in order to decide if a SLA has been or is being violated

Figure 32 depicts one possible classification for monitoring data that must be collected and interpreted by the SLA monitoring system. We emphasize the distinction between System Related and Service Related data. The latter provides technical or business information related to service instances, such as Memory Usage or Number of Requests. Another important aspect is the existence of user-defined data and thus the need for an extensible monitoring system.

---

[83] Dimitrakos T., D. Mac Randal, G. Laria, N. Romano, P. Ritrovato, B. Serhan, S. Wesner, Trust, Security and Contract Management Challenges for Grid-based Application Service Provision. 2nd International Conference on Trust Management. Springer-Verlag LNCS, March-April 2004.

[84] http://www.hpl.hp.com/techreports/2002/HPL-2002-324.html

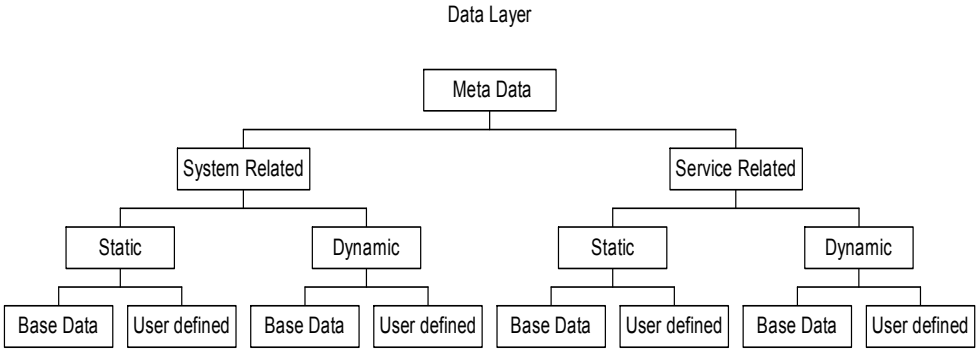


Figure 32 Classifications of Monitoring Data

If a SLA has been violated, there are in principal two possible reactions.

- The first is to start actions that change system conditions in order to meet the SLA. In that case a *SLA Load Balancer* is needed that will start corrective actions based on the values of specific system parameters. In a sophisticated environment such parameters will not only consider hardware oriented values like memory or processor load but also the kind and number of running processes. The assessment of the actual system situation might become a quite complicated task and some sort of *Host Benchmark Monitor* might become necessary.
- The second is the delegation of the necessary reaction to another module by sending a message through a *SLA Event Manager*. The Event Manager is also necessary to send messages to other subsystems if a SLA is violated or terminated.

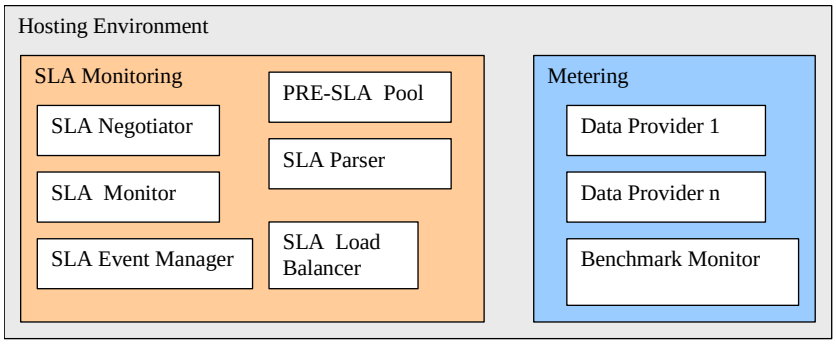


Figure 33 Logical SLA building blocks

Figure 33, above summarizes the identified building blocks, which will be the base for further discussions:

- The PRE-SLA Pool describes a set of possible SLAs that are offered by a Service Provider regarding a specific service. The SLAs are described in terms of a SLA language.
- The SLA Negotiator is responsible for the negotiation of a contract between a Service Provider and a Service Client. A succesful negotiation will result in a unique SLA.
- Data Providers are sources of system and service instance related data
- The SLA Monitor monitors the situation of a Grid service with respect to a specific SLA
- The Benchmark Monitor is responsible to calculate and provide composed system information as input for the Load Balancer

- The SLA Load Balancer is responsible to initiate corrective actions in case of upcoming or existing SLA violation
- The SLA Event Manager is responsible for handling all necessary messages that must be exchanged between the SLA monitor and other components.
- The SLA Parser provides tools to support the use of SLDL constructs.

4.9.1.2 SLA subsystem of the GRASP infrastructure

Due to resource limitations and its "proof-of-concept" nature, the current implementation of the above logical representation of the SLA monitoring system as a part of the GRASP Infrastructure, has limited some of the features describe above. To better present the resulting design and explain these limitations we depart from the following base scenario:

1. A Service Client (SC) runs a search for a specific type of service on the Service Locator based on functional criteria and provided QoS
2. The Service Locator returns a list of potential Service Providers (SP) that provide that kind of service
3. SC starts a negotiation with the different SP's and select the one with the best offer
4. SC and SP contract the SLA and the SP instantiates the requested service, which includes to give notice to Accounting
5. SP must monitor the running service regarding the agreed SLA and must perform appropriate actions in case of SLA violation
6. Finally the service will terminate regularly or due to a SLA violation. In both cases any interested party such as Accounting must be informed.

4.9.1.2.1 SLA Parser

Although the first step does not directly affect the SLA monitoring of a Hosting Environment, it is strongly related to a logical building block, namely the SLA Parser. The problem for the Locator and many other GRASP systems, is to understand a non trivial query for a specific kind of SLA, or in other words, to understand a Service Level Description Language. As described above it is assumed that a SLDL in the ASP domain has to support a high level of complexity. GRASP will rely on the Web Service Level Agreement (WSLA) specification, which is an XML based description language[85]. WSLA proposes for each SLA an XML schema composed of three parts:

- A first part describes the actors of the agreement, that is to say the Service Consumer and the Service Provider.
- Then, WSLA specifies a service definition to which is attached SLA Parameters.
- Finally, WSLA allows Metrics related to a SLA Parameter to be declared. Metrics represents how to measure an item. Those Metrics will be used by the SLA subsystem in order to know which measurements the SLA Monitoring will have to monitor. Metrics can be composed of a single Metric, but Metrics can be composite Metrics, so the WSLA language provides the ability to declare Metrics, which are an aggregation of Metrics. Those declarations will be used in order to do the mapping between the data collected and the SLA Parameters.

To facilitate the usage of WSLA we are going to provide a SLA Parser that exposes appropriate methods for the need of different subsystems.

[85] http://www.research.ibm.com/wsla/

4.9.1.3 SLA Pool and Negotiators

The next two steps pertain to negotiation of an SLA and include the usage of the Pre-SLA pool and the SLA Negotiator. First we must consider the internal structure of the Pre-SLA which has an impact on the negotiation phase. If we assume that a Pre-SLA contains some parameters that can be set by the client, we could imagine a scenario, where the SC fills in a parameter and sends this SLA suggestion to the SP. The SP checks if it is willing to accept this SLA and sends either back a confirmation, a new SLA suggestion or a request for another SLA. This kind of multi-phase negotiation obviously would lead to increased requirements for the SLA monitoring system. An HE (Hosting Environment) would need not only to be aware of the running services, but also of the services that are currently negotiated and thus must provide some kind of reservation system.

In the GRASP infrastructure implementation we will only provide a one-phase negotiation. This means the SC will send a Pre-SLA, maybe with some specified parameters, to the HE which will confirm or deny the service creation. A second call to the same HE with a changed SLA will be treated as completely independent request. Even in case of one-phase negotiation the problem remains, to check if a HE is able to serve a specified SLA. In a first step, this means checking if the HE is in general capable of supporting such a SLA, as we can not exclude invalid SLA requests. This is simply done with the *HE SLA Pool*, which will be implemented as a collection of static XML documents that describe the available SLAs of the hosting environment. Those XML documents will be the same kind of templates as published in the Locator. Keeping in mind the general architecture of a GRASP HE, we will map the SLA Negotiator into two GRASP components, the *HE Negotiator* and the *Host Negotiator*. Because a HE might comprise different Hosts, it is the role of the HE Negotiator, sitting on the Gateway, to do this preliminary check.

Obviously this is the prerequisite to the much more demanding task, to decide if the actual situation of the HE allows to instantiate this service with respect to the requested SLA. This will be explained in more detail below.

4.9.1.3.1 SLA Load Balancer and Benchmark Monitor

Negotiation of a SLA comprises two important subtasks.

1. The exchange of the considered SLA. Dependent on the number of negotiation phases this might occur several times between SC and SP and might comprise reservation of system resources. As said above, GRASP will support only one-phase negotiation, thus this part is reduced to the confirmation or rejection of a requested SLA.
2. The mapping of incoming requests onto measurable system metrics in order to decide if the HE is able to serve this request now. Obviously this decision could be made on the basis of some simple system data like processor load or very complex data such as type and behaviour of already running service instances. As a result of the GRASP architecture, there might be also two or more Hosts that provide factories for the requested service and could be in general able to serve the demanded SLA. GRASP implements this task in the following way:
    - The *HE Negotiator* will analyze the incoming SLA and will create a list of potential Hosts for this request
    - This list will be passed to the HE Monitor component running on the Gateway, which should sort the Host list considering actual system metrics of the pre-selected Hosts. Therefore it must collect appropriate data from each Host.
    - The returned Host priority list will be passed to the Instantiator, who asks the Host Negotiator of the first list entry, if he is able to serve this request. If the Host Negotiator returns true, the service will be instantiated on this Host, otherwise the Instantiator will contact the next Host in the list.

As described above, the *HE Monitor* needs to collect data from the different Hosts in order to build a priority list of the potential Hosts. Based on our metrics classification we could use system related or service related data. GRASP will use only system related data, because a

meaningful consideration of service related data would demand for a very complex analysis of service type, number and behaviour of all running services which is out of the scope of GRASP.

Because GRASP implementation is based on the Microsoft .NET framework, we provide a *Host Monitor* Grid Service, which leverages the Microsoft Performance Counters. On an average system there are some thousand base counters available out of the box. In addition it is possible to define and use custom counters. The *Host Monitor* exposes a PortType which is able to serve requests for an arbitrary combination of those counters.

When the *HE Monitor* tries to build the priority list, it has to know which kind of counters it should consider. This can be defined by the SP in the Pre-SLA. It will be for example possible to say 'Processor Load', priority 1, 'Free Memory', priority 2. The *HE Monitor* will extract this information from the incoming SLA, will ask the *Host Monitor* for the suitable counters and will calculate the list based on the measurement results and the given priority. Obviously it is possible to optimize this calculation mechanism without affecting the rest of the infrastructure. Note that we merged the Load Balancer and the Benchmark Monitor into the *HE Monitor* Grid Service.

Although the Instantiator receives a priority list about available Hosts, the decision about the instantiation of a service is made by the *Host Negotiator.* The latter one makes a more detailed assessment about the current situation of the Host, especially taking the *Host SLA Pool* into account. In contrast to the *HE SLA Pool,* the *Host SLA Pool* can be extended by the Host administrator with additional rules, for example a limitation of running services of this type.

4.9.1.3.2 SLA Monitor

The main task of the SLA monitoring system is the evaluation of running Grid Service instances. We have to keep in mind that from the SC viewpoint there is a one-to-one relation between a service and a SLA in the meaning, that a SLA always describes the contract between a SC and SP regarding the usage of a service under clearly defined QoS. From the technical viewpoint this does not necessarily means that there is a one-to-one relation between a SLA and a Grid Service instance. It might be the case that for performance reasons different SC share instances from an instance pool. The first GRASP prototype will not use instance pools, thus we focus on the scenario, that there is a one-to-one relation between SLA and Grid Service instances.

Independent from the usage of instance pools, the monitoring of Grid Services is challenging due to the dynamic nature. They have dynamic lifetime management, might be instantiated on different Hosts and even might move from one Host to another. GRASP will handle this through the introduction of Agreement Grid Services (AGS). Each time a Business Grid Service is instantiated on behalf of a SC, we instantiate an accompanying 'Watchdog' Grid Service that knows the relevant SLA and observes the running business service. We were pleased to see that some time after our basic design, the former OGSI-Agreement specification, now WS-Agreement, proposed a solution similar to ours. The instantiation of the AGS is done by the Instantiator component through an Agreement factory. During the monitoring phase, the AGS collects data from the Host Monitor as well as from Data sources that are able to expose service related data. In case of SLA violation, the AGS can decide if it is necessary to notify Accounting in order to apply penalties or even to shutdown the service instance. Figure 34 depicts the overall situation based on the considerations above.

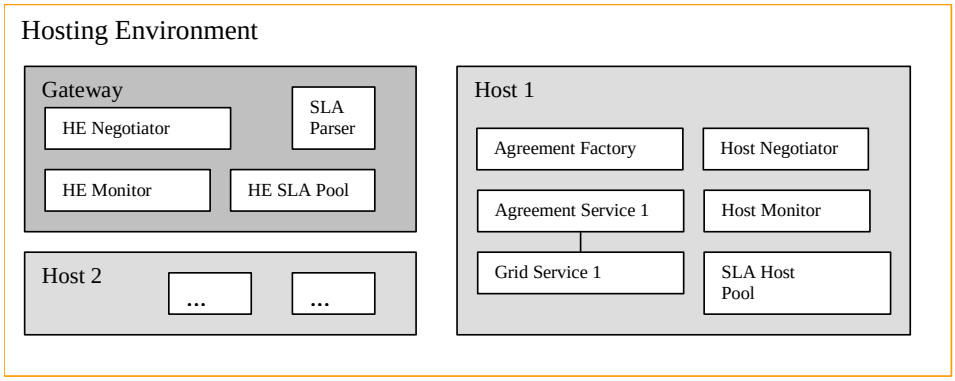


Figure 34GRASP SLA Monitoring Components

4.9.1.3.3 SLA Event Manager

GRASP exploits the notification model of OGSI.NET. For SLA monitoring this means that Accounting is informed by the AGS about SLA violation through notification. If the AGS decides to destroy a service it will also inform the Instantiator. Furthermore the AGSs themselves subscribe to the monitored business service in order to be informed if a service terminates, according to plan or unscheduled.

4.9.1.3.4 Orchestration

A special problem arises when it comes to the orchestration of Grid Services (one of the main GRASP goals), because the SLA of an orchestrated service is almost certainly not just the sum of the SLAs of all the orchestrated services. The purpose of an orchestration service is to provide a new business service. That means that from the viewpoint of the service consumer, it is the quality of the result and not the quality of the underlying services that is of interest, as long as the SLA regarding the composed service is not violated. Therefore each orchestration service must be monitored regarding the corresponding business process in a similar fashion to the monitoring of individual grid services. This implies that each orchestration workflow that models a business process must have its own SLA based on one of the template SLAs from the SLA pool.

4.9.1.4 Summary

Summarizing, GRASP implements the following SLA components:

- Locator – allows search for services based on functional and QoS criteria
- HE Negotiator - Analyzes the requested SLA and returns a priority list of potential Hosts to the Instantiator.
- Host Negotiator – Makes the final decision about the instantiation of a service applying Host specific rules.
- HE Monitor- Sorts a given Host list based on a given SLA by interfacing with the appropriate Host Monitors
- Host Monitor – Gives access to different kind of system related data
- HE and Host SLA pool – Maintain information about available Pre-SLA in the Host Environment.

Agreement Factory – Creates Agreement Services that are responsible to observe running business services regarding service level violation

4.9.2 Business Contracts Architecture

B2B relationships are usually about exchange of services or products on the one hand and funds on the other hand. These relationships may be for very long period of time or very short period time e.g. consisting of only one transaction. Contracts between the business parties are very useful as a means for coordinating their interactions and importantly creating a level of control to compensate the lack of trust between the parties. The second requires, of course, that the contract is legally binding.

E-contracting between business partners is about automated creation and executed contracts that are securely created and managed such that they can be considered as legally binding contracts.

The contracts can be of many different types. A typology of different contracts based on various business scenarios is given in [86]. In this comprehensive survey of the existing industrial and academic approaches as well as standards, the classification of contracts is made as follows:

- One-to-one contracting: simply a contract between a supplier and a consumer.
- Chain contracting/multiparty contracting: a contract that involves several parties in which some of the contracts are sub-contracts that are needed for one party to establish other contracts. For example, a product supplier may be dependent on other suppliers providing parts of its product. A contract between a product supplier and its costumer will be dependent on the auxiliary contracts between the product supplier and its subcontractors.
- One-to-many contracting: a contract between one party and several other parties. In fact this can be seen as several one-to-one contracts that all have the same entity as one their contractual parties.
- Complex chain contracting: a scenario based on composition of chain contracting and one-to-many contacting.
- Complex multipart contracting: a composition of multiparty contracting and one-to-many contracting.

One of the most comprehensive research pursued in this area is the Business Contracts Architecture (BCA) initiative driven by DSTC, Australia.

This section describes the basic components of architecture for contract establishment and execution based on [87]. The remainder of the section depicts this role-based architecture, and indicates key information flow between the roles, which in general are involved in more than one process. The architecture represents an extension of the Business Contract Architecture BCA described in [88].

[86] S.Angelov, P. Grefen, B2B eContract Handling – A Survey of Projects, Papers and Standards. Telematica Institute Report 2003)

[87] Zoran Milosevic, Audun Josang, Theo Dimitrakos, Mary A. Patton – Discretionary Enforcement of Electronic Contracts. Proc. EDOC '02. pp(s): 39 -50. IEEE CS 2002

[88] Zoran Milosevic - Enterprise Aspects of Open Distributed Systems. PhD thesis, Computer Science Dept. The University of Queensland, October 1995.

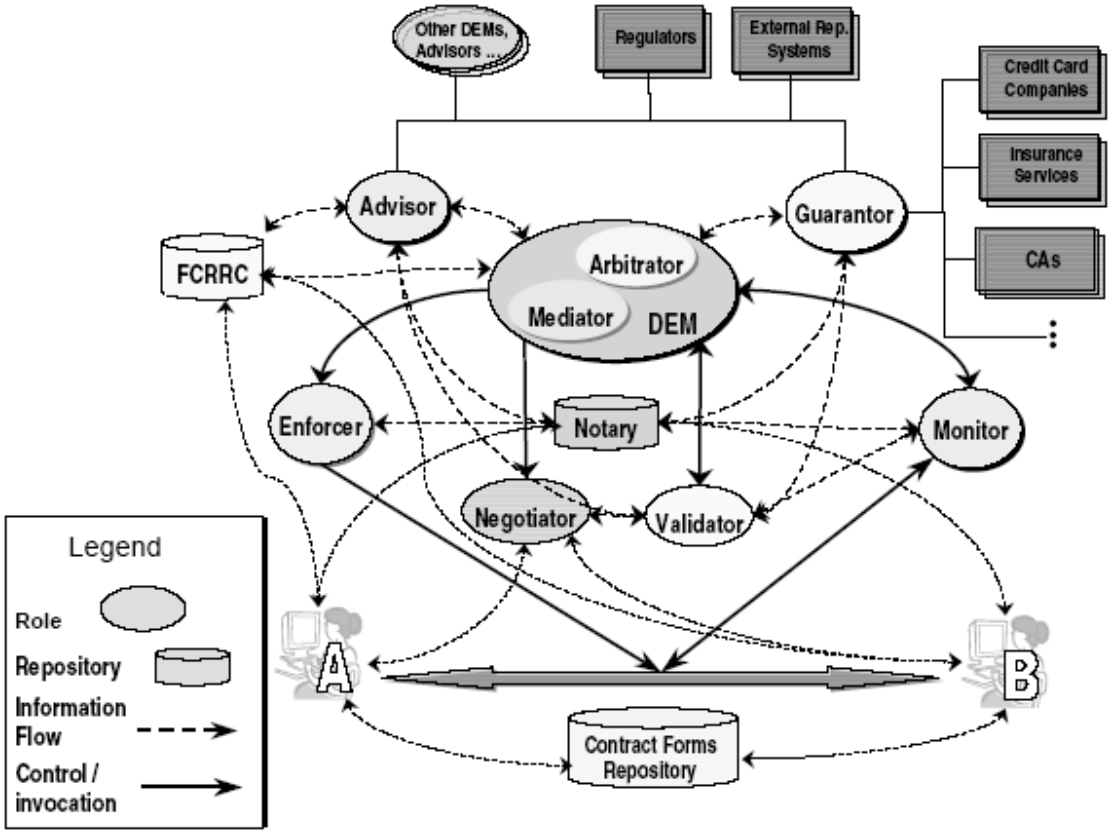


Figure 35 Overview of basic roles and interactions in Business Contract Architecture

4.9.2.1 Roles Supporting Contract Establishment

The following roles support the process of establishing a contract:

- **Negotiator** mediates the negotiation process. During the negotiation phase parties can exchange offers and counter-offers containing one or more of the following: contract templates, individual contract clauses and finally contract variables that are negotiable items. During contract negotiations it may be possible to do certain aspects contract validity checking as mentioned below.
- **Validator** ensures the creation of legally valid contract instances, assessing proposed contracts against various aspects of contract validity such as competence, clarity, legal purpose and consideration elements. See [89] for further details on contract validation.
- **Notary** is a trusted party that stores contract instances after the contract has been agreed upon, checked for validity and signed by both parties. Such contract instances can be later used as evidence of agreement in the contract monitoring and enforcement activities. Notary component can be also hosted by one or both parties involved in contract.
- **Contract Forms Repository** provides storage and access to standard contract forms or contract clauses, depending on contractual scenario. It can be used by parties to the contract who use pre-defined contract forms to produce individual contract instances or by contract drafters who are defining building blocks for contracts. There may be also a need for a specialised contract templates editor that can provide functionality of both text

[89] *Z. Milosevic, D. Arnold, L. O'Connor - Inter-enterprise contract architecture for open distributed systems: Security requirements. Proc. of WET ICE'96 Workshop on Enterprise Security, Standford, June 1996*

editing but also type definitions for the fields that represent negotiable items within the contract.

4.9.2.2 Roles supporting trust establishment.

We distinguish three special *roles* that entities mediating in a trust relationship can play in relation to contract establishment and execution. These roles are *guarantors, intermediaries*, and *advisors*. Note that an entity may play more than one mediating role in a business relationship.

- ***Guarantor*** is a party taking the responsibility that the obligations of the parties she acts as a guarantor for are fulfilled at an agreed standard. Guarantors assist the establishment or facilitate the increase of trust for a specific transaction by underwriting (a part of) the risk associated with the transaction. A typical example is a credit card company.
- ***Advisor*** is a party that offers recommendations about the dependability of another party. Advisors include the authorities maintaining blacklists for a community. Examples include, credit scoring authorities and reputation systems.
- ***Feedback Collection and Reputation Rating Centre (FCRRC)*** gathers feedback about a participant's behaviour over time, enabling a reputation rating to be derived for that participant. This has the potential to be utilised at a number of stages in the contracting process. In the first instance, an Advisor may utilise reputation ratings to assess a potential e-commerce partner prior to the establishment of a contract. The Advisor may have access to a range of external reputation systems, including reputation systems shared by a trusted network of business partners. The Advisor may also have access to an internal Feedback Collection and Reputation Rating Centre, which has gathered specific information about the business's prior interactions with the business partner being assessed.Information gathered by the FCRRC may also be taken into account by the Arbitrator in the arbitration decision-making process should a dispute arise, allowing for a more informed decision to be made. While certain external reputations systems may be subject to bias and remain open to manipulation by dishonest parties, tight controls are inherent in the internal reputation system. A reputation system utilised by a trusted network of businesses may also provide a more trusted source of reputation ratings than external systems. The Advisor and the Arbitrator may take these variations into account and derive an accurate overall reputation rating for partners in e-business contracts.
- ***Intermediary*** is a party that intervenes between other parties in a business transaction and mediates so that they establish a business relationship with or without their knowledge. Unlike advisors, an intermediary may also participate in a contract establishment or execution providing access to the services provided by another party, or representing as a third party representing a service provider. Examples of intermediaries include proxies, information portals, banks who offer to their customers non-financial services such as car rental or flight bookings through an allied service provider, bookshops who offer product delivery by a third party courier as a part of their service. Intermediaries are further classified in [90] with respect to whether they reveal the existence or identity of the party the mediate for.

These roles are not mutually exclusive; an entity may consistently perform more than one of them simultaneously. For example a Contract Negotiator may play the role of an intermediary (by providing a simple matching service) and also an advisor (if it provides recommendations about offers and counter-offers to the parties discussing a contract). In analogy, a certification authority may be an advisor (if possessing a certificate is interpreted as presenting a reference from a reputable third party) or a guarantor (if the certification authority is obliged to underwrite part of the damages caused by the non-performance of a party it certified).

[90] Dimitrakos T. - System Models, e-Risk and e-Trust. Proc. IFIP I3E-1, Kluwer Academic Publishers 2001.

4.9.2.3 Roles Supporting Contract Execution

The following roles support contract enforcement and performance monitoring during the performance of a contract.

- **Monitor** enables monitoring of the activities of parties, measuring their performance if needed and recording the relevant events. It can also signal a contract non-performance to the Discretionary Enforcement Moderator (DEM, see below) if it detects such an event.
- **Notifier** implements various notifications mechanisms needed to send warning messages to indicate a pending contract-significant event, including possible non-compliance event that may be detected. To simplify presentation, Notifier is not shown in the figure.
- **Enforcer** applies enforcing actions directly to the parties to ensure that some specific behaviour conforms to the contract. From a control theory point of view, this role is analogous to an actuator.

Discretionary Enforcement Moderator (DEM) forms an opinion about the extent of deviation by the non-performing parties. Once the arbitrator forms such an opinion, it chooses a route of action which may invoke settlement leading to the success of a suitably amended transaction. Alternatively, it may endorse the enforcement of corrective measures to be executed by a preventive security mechanism realised by the Contract Enforcer role. (An overview of the DEM's decision making procedure is modelled as a finite state machine in Figure 35 (See also [91],[92],[93]). The DEM forms its opinions on the basis of evidence about deviation of the non-performing parties, that is provided by the Contract Monitor, external advisors and possibly additional recommendations from agents representing the parties, in a spirit similar to a (human) judge's process for arriving at his ruling. During this process the DEM component may take the following specific roles; Mediator or Arbitrator.

- **Mediator** - who initiates a settlement leading to the success of an amended transaction or decides failure of mediation leading to the invocation of arbitration.
- **Arbitrator** - takes over when a settlement as per above cannot be reached, or when a party's deviation from the expected performance is high enough to justify the deployment of corrective measures. An arbitrator may initiate the enforcement of corrective measures through the Contract Enforcer, leading to the recoverable failure of the transaction and, potentially to penalising the non-performing party. In the absence of any suitable corrective measures, the Arbitrator may signal correction failure, in which case the Contract Validator is informed so as to prevent further access to the system by the non-performing parties, if necessary, and the case is carried on outside the Contract Architecture.

4.9.2.4 Contract Arbitration and Enforcement

In [94] we provided an abstract interpretation of the contract enforcement processes as a finite state machine (Figure 36). We used five different levels as a means of classifying the states and transactions between the sates of the enforcement process according to the

[91] Zoran Milosevic, Audun Josang, Theo Dimitrakos, Mary A. Patton – Discretionary Enforcement of Electronic Contracts. Proc. EDOC '02. pp(s): 39 -50. IEEE CS 2002.

[92] Dimitrakos T., Djordjevic I., Milosevic Z., Jøsang A., Phillips C. - Contract Performance Assessment for Secure and Dynamic Virtual Collaborations, to appear in Proc. of EDOC'03, 7th IEEE International Enterprise Distributed Object Computing Conference, September 16-19 2003, Brisbane, Australia.

[93] Daskalopulu A., Dimitrakos T., and Maibaum T. Evidence-Based Electronic Contract Performance Monitoring. INFORMS Journal of Group Decision and Negotiation, Special Issue: Formal Modeling of Electronic Commerce, Spring 2002.

[94] Zoran Milosevic, Audun Josang, Theo Dimitrakos, Mary A. Patton – Discretionary Enforcement of Electronic Contracts. Proc. EDOC '02. pp(s): 39 -50. IEEE CS 2002.

degree with which the transaction execution is on compliance with the prescribed agreement. Level 1 reflect full compliance whereas level 5 reflect total transaction failure and the inability to apply corrective measures to the non-compliant party.

- *Level 1* reflects the fact that the transaction is executed according to the prescribed contract. The virtual CCT monitor observes contract-significant events and evaluates whether the corresponding behaviour pattern was compliant with the contract. Deviations can occur while the execution remains on level 1 as long as notifications to the contractual parties, results in the execution getting back on track.
- *Level 2* reflects the fact that the transaction has deviated from the prescribed contract, and warnings to non-compliant parties have been ignored. The Monitor informs the DEM which in turn invokes the Mediator, who attempts to establish an amended contract between the two parties, supported by the Negotiator. In case of settlement the contract execution returns to Level 1 and resumes with the amended contract as a basis.
- *Level 3* reflects that the Mediator was not able to make the contractual parties agree on an amended contract. The Arbitrator collects all available evidence in order to reach the fairest decision possible. In case the decision by the Arbitrator is accepted by all parties (i.e. virtual CCT members and their corresponding local managers), the contract execution returns to Level 1 and resumes with the arbitrated contract as basis.
- *Level 4* reflects the fact that the arbitration decision is not accepted by some of the contractual parties. The Mediator attempts to apply penalties to the parties it sees as non-compliant, by invoking the corresponding Enforcer.
- *Level 5* reflects the fact that the DEM was not able apply penalties to the non-compliant parties. The suffering parties have the option of initiating legal procedures, outside of the realm of the electronic contract management system.

After transaction completion, at the first three levels, the contractual parties are invited to provide feedback about each others performance. The feedback is collected by the Feedback Collection Centre (FCC) and is used to derive a reputation rating about each party in the system. Once the fourth level is reached, feedback is not collected from the contractual parties because it is assumed that the hostility between them will make the feedback highly biased and unreliable.

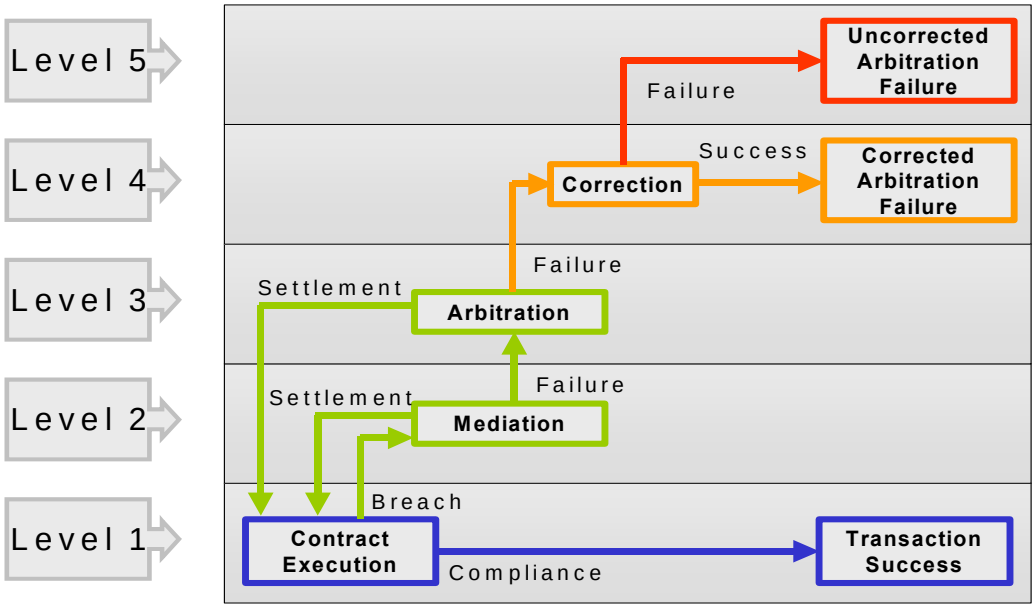


Figure 36 Contract enforcement process seen as a state diagram organised in layers of criticality

4.9.2.5 BCA implementation and interoperability

At present a team at Distributed Systems Technology Centre (DSTC) in Australia are implementing BCA. Current prototyping leverages on Web Services technologies and intends to be interoperable with web platforms such as Microsoft .NET™, IBM WebSphere™, and MySAP™.com For further information see [95].

4.10 Contractual Resource Sharing in VOs

In this section we summarise recent work at SICS and University of Milano towards a formal framework for regulated resource sharing across administrative domains.

4.10.1.1 Formal framework for resource sharing

In a paper by Firozabadi et al[96], the authors develop a formal framework for regulated resource sharing in coalitions or virtual organisations. The assumption is that a VO is defined in terms of a set of regulations prescribing how the VO members have to bring in resources to the VO and to share these with other VO members. These regulations are stipulated in a VO contract that each VO party needs to /comply with/ at any time of its VO membership. A party meets a contract if it publishes a local policy that specifies its plans for fulfilling its contractual obligations. The plan itself is dynamic and VO parties can change their plans over time in order to optimize usage of their resources. However, fulfilling a contractual obligation requires more than only publishing a plan for that. A party, upon a request from an entitled agent based on the local policy, shall release the requested resource. Otherwise, the obligation is violated and as the result of that another stricter obligation will be activated. In case there is no other stricter obligation specified in the contract then the entire contract is seen as violated.

The authors define a formal language for representing VO contracts and local policies. A VO contract is constructed by contract blocks. A contract block specifies a number of obligations that the obliged party has a free choice to choose which one to fulfill. In practice the structure of contract blocks defines a number of obligations such that violation of one of these results in another (stricter obligation) to becoming active.

4.10.1.2 Prototype Implementation

Currently there is a prototype of a reasoning engine and a demonstration package, developed at Swedish Institute of Computer Science (SICS) that implements the calculus given in the paper. The engine allows one to verify if a local policy is compliant with a VO contract or not. Further, it keeps track of the fulfilments and violations of the parties' obligations at each time-point of the VO life cycle. The next release of the engine will include functionalities for advanced scheduling of resources based on probabilities for future obligations to be fulfilled or violated.

4.11 ebXML Trading Party Agreement

ebXML is a set of specifications (supported by OASIS) proposed by a large groups of businesses, government standards commitees and academics to enable a well structured electronic business framework. The vision of ebXML is to enable a global electronic

[95] Elemental Hope Page: http://www.dstc.edu.au/Research/elemental-ov.html

[96] B. Sadighi Firozabadi, M. Sergot, A. Squicciarini, E. Bertino, A Framework for Contractual Resource Sharing in Coalitions, to be presented at Policy 2004.

marketplace where enterprises of any size and in any geographical location can meet and conduct business with each other through the exchange of XML-based messages. ebXML can also be seen as the next generarion of business protocols and computing and succeed Electronic Data Interchange (EDI).

eBXML, when compared to Web Services ,has many business specific data objects, protocols, networked models and business data interchange. With the emerging convergence of Web Services and ebXML, most implementations will hopefully leverage the vast Web Services deployment and realize ebXML based data exchange and protocols.

4.11.1 Who is endorsing ebXML?

Computer/technology companies are not the only entities that endorse ebXML; backers include a large number of industrial, shipping, banking, and other general-interest companies. The direct sponsors of ebXML are OASIS (Organization for the Advancement of Structured Information Standards) and UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business). Lots of standards bodies also have a finger in the pie, including NIST (National Institute of Standards and Technology) and W3C (World Wide Web Consortium).

4.11.2 What are the Specification and Component Details?

Registry: A central server that stores a variety of data necessary to make ebXML work. Amongst the information a Registry makes available in XML form are: Business Process & Information Meta Models, Core Library, Collaboration Protocol Profiles, and Business Library. Basically, when a business wants to start an ebXML relationship with another business, it queries a Registry in order to locate a suitable partner and to find information about requirements for dealing with that partner.

Business Processes: Activities that a business can engage in (and for which it would generally want one or more partners). A Business Process is formally described by the Business Process Specification Schema (a W3C XML Schema and also a DTD), but may also be modeled in UML.

Collaboration Protocol Profile (CPP): A profile filed with a Registry by a business wishing to engage in ebXML transactions. The CPP will specify some Business Processes of the business, as well as some Business Service Interfaces it supports.

Business Service Interface: The ways that a business is able to carry out the transactions necessary in its Business Processes. The Business Service Interface also includes the kinds of Business Messages the business supports and the protocols over which these messages might travel.

Business Messages: The actual information communicated as part of a business transaction. A message will contain multiple layers. At the outside layer, an actual communication protocol must be used (such as HTTP or SMTP). SOAP is an ebXML recommendation as an envelope for a message "payload." Other layers may deal with encryption or authentication.

Core Library: A set of standard "parts" that may be used in larger ebXML elements. For example, Core Processes may be referenced by Business Processes. The Core Library is contributed by the ebXML initiative itself, while larger elements may be contributed by specific industries or businesses.

Collaboration Protocol Agreement (CPA): In essence, a contract between two or more businesses that can be derived automatically from the CPPs of the respective companies. ebXML CPA is an XML format for describing agreement information for partners that agree to collaborate. This agreement is based on the ebXML architecture. When a company A finds another company B in the registry which it wants to do business, the it can create a

CPA as a reflection of the CPP of company B which was retrieved from the registry. To finalise a CPA, the companies have to negotiate on different business and transaction terms.

A CPA is intersection of two parties' CPPs plus some agreed parameters and variable values defining their IT capabilities and transactions in more details.

Simple Object Access Protocol (SOAP): A W3C protocol for exchange of information in a distributed environment endorsed by the ebXML initiative. SOAP provides and envelope that defines a framework for describing what is in a message and how to process it.

4.12 Conclusions

The goal of TrustCoM is to facilitate on-demand provision of resources and services among a number of independent entities that interact within the scope of a virtual organisation. Service providers and consumers in a virtual organisation will interact and exchange services and payments according to some contracts.

Of the approaches and technologies evaluated, the GRASP contract management subsystem, WSLA, WS-Agreement, SNAP, and the "Contractual resource sharing" approach are optimised for SLAs relating to functions exposed as web services and to the use of network and computational resources as utility. ebXML CPA focuses on a trade partners agreements (which can be understood as a special case of framework agreements) while BCA is a more general approach clarifying a number of common functionalities that contract management system should support for either case.

The main focus of ebXML is to provide basis for inter-operability between different business partners IT applications enabling advanced computerised business-to-business transactions. The business agreements between the parties can be specified in terms of CPAs. However, the ebXML standard can be seen as a complementary to the web-services technology and there is a need to investigate the possible relation between ebXML CPA and WSLA and WS-agreement. Although these support different levels of implementations, they probably have a lot of issues in common.

BCA is particularly interesting as perhaps one of the more established and, conceptually, the most comprehensive of the research approaches evaluated in this chapter. It does not

however places as emphasis on open dynamic systems and VO resource management as do solutions such as the GRASP Contract Management subsystem and WS-Agreement, and to a lesser extent, WSLA. Nevertheless, the philosophy and architectural concepts of BCA may provide useful guidance of the functionality that should be provided by the TrustCoM framework albeit in a more dynamic VO setting.

WSLA and WS-agreement are promising approaches for this purpose and they may constitute a base for the TrustCoM framework and the development. However, their usability shall be investigated for the TrustCoM scenarios and extensions and modifications shall be made if necessary.

The SNAP approach is also an interesting approach especially in the case of SLAs for computational resources as in grid computing. However, it seems that future work on this approach is planned to be within the frame of WS-agreement and SNAP will be soon outdated.

The GRASP Contract Management subsystem leverages on the WSLA language and preceded the development of WS-Agreement initiative. In fact, several of the WS-Agreement concepts had already been tested in GRASP before WS-Agreement emerged as the de-facto working pre-standard in the Global Grid Forum. At present provides a more comprehensive implementation of the resource allocation and SLA monitoring concepts that underpin WS-Agreement. The current implementation of the GRASP contract management subsystem, however, lacks adequate support for SLA negotiation and a reasoning scheme for resource planning and reservation across the multiple administrative domains and execution environments of a VO.

“Contractual resource sharing”, although still in its infancy, is a potentially interesting approach to investigate for improving the sharing and scheduling resources belonging to different and independent administrative domains. To that extent it complements the functionality of the GRASP contract management subsystem. However, the framework needs to be extended to be compliant with WSLA and/or WS-agreement for such an integration to be feasible.

None of the approaches evaluated places particular emphasis on the integration of the contract management mechanisms with mechanisms for trust establishment and security management. (Although some preliminary results in this direction have been published by Dimitrakos et al in [97] in relation to BCA and in [98] in relation to GRASP contract management subsystem.) TrustCoM innovation is expected to make a substantial impact in this area.

[97] Dimitrakos T., Djordjevic I., Milosevic Z., Jøsang A., Phillips C. - Contract Performance Assessment for Secure and Dynamic Virtual Collaborations, to appear in Proc. of EDOC'03, 7th IEEE International Enterprise Distributed Object Computing Conference, September 16-19 2003, Brisbane, Australia.

[98] Dimitrakos T., D. Mac Randal, G. Laria, N. Romano, P. Ritrovato, B. Serhan, S. Wesner, *Trust, Security and Contract Management Challenges for Grid-based Application Service Provision*. 2nd International Conference on Trust Management. Springer-Verlag LNCS, March-April 2004.

5 Collaborative Business Processes

Edited by: Yücel Karabulut
SAP

5.1 Introduction

Web services are rapidly emerging as the most practical approach for integrating a wide array of customer, vendor, and business-partner applications. To make the most of new Web services investments there must be a standard approach to Web services composition. Organizations need the agility to adapt to customer requirements and changing market conditions. But existing business process languages do not directly support Web services standards and, as a result, IT organizations may be tempted to take a short-term approach and create their own proprietary protocols for composing services together. Web services orchestration and choreography standards are efforts that can be long-term solutions for business connectivity. By connecting services through open, standards-based methods, organizations spare themselves the burden of maintaining these proprietary interfaces.

In this section we discuss five specifications: the Web Services Transaction, Web Services Coordination, the Web Service Choreography Interface (WSCI), Business Process Management Language (BPML) and Business Process Execution Language for Web Services (BPEL4WS). BPEL4WS (or alternatively WSCI/BPML combination), WS-Transaction, and WS-Coordination together form the bedrock for reliably choreographing Web services-based applications, providing business process management, transactional integrity, and generic coordination facilities respectively. These are designed to reduce the inherent complexity of connecting Web services together. Without them, an organization is left to build proprietary business protocols that shortchange true Web services collaboration. Recently, the terms orchestration and choreography have been employed to describe this collaboration.

5.2 WS-Coordination[99]

WS-Coordination describes an extensible framework for providing protocols that coordinate the actions of distributed applications. Such coordination protocols are used to support a number of applications, including those that need to reach consistent agreement on the outcome of distributed activities.

The framework defined in the specification enables an application service to create a context needed to propagate an activity to other services and to register for coordination protocols. The framework enables existing transaction processing, workflow, and other systems for coordination to hide their proprietary protocols and to operate in a heterogeneous environment.

Additionally the specification describes a definition of the structure of context and the requirements for propagating context between cooperating services.

The WS-Coordination specification talks in terms of activities, which are distributed units of work involving one or more parties (which may be services, components, or even objects). At this level, an activity is minimally specified and is simply created, made to run, and then completed.

[99] IBM, Microsoft, BEA. Web Services Coordination (WS-Coordination). September 2003.http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/wscoor.asp

WS-Coordination looks set to become the adopted standard for activity coordination on the Web. WS-Coordination provides only activity and registration services, and is extended through protocol plug-ins that provide domain-specific coordination facilities. In addition to its generic nature, the WS-Coordination model also scales efficiently via interposed coordination, which allows arbitrary collections of Web services to coordinate their operation in a straightforward and scalable manner.

5.3 WS-Transaction

While WS-Coordination provides only context management including the creation of contexts and registration of activities with those contexts, WS-Transaction leverages the context management framework provided by WS-Coordination in two ways. First, it extends the WS-Coordination context to create a transaction context. Second, it augments the activation and registration services with a number of additional services (e.g. Completion, Completion WithAck, PhaseZero, 2PC) and two protocol message sets (one for each of the transaction models supported in WS-Transaction) to build a full-fledged transaction coordinator on top the WS-Coordination protocol infrastructure.

WS-Transaction supports the notion of the service and participant as distinct roles, making the distinction between a transaction-aware service and the participants that act on behalf of the service during a transaction. A transaction-aware service encapsulates the business logic or work that is required to be conducted within the scope of a transaction. This work cannot be confirmed by the application unless the transaction also commits and so control is ultimately removed from the application and placed into the transaction's domain. The participant is the entity that, under the dictates of the transaction coordinator, controls the outcome of the work performed by the transaction-aware Web service.

The WS-Transaction specification proposes two distinct models: Atomic Transactions and Business Activities.

5.3.1 WS-AtomicTransaction[100]

WS-AtomicTransaction provides the definition of the atomic transaction coordination type that is to be used with the extensible coordination framework described in the WS-Coordination specification. The specification defines three specific agreement coordination protocols for the atomic transaction coordination type: completion, volatile two-phase commit, and durable two-phase commit. Developers can use any or all of these protocols when building applications that require consistent agreement on the outcome of short-lived distributed activities that have the all-or-nothing property.

5.3.2 WS-BusinessActivity[101]

WS-BusinessActivity provides the definition of the business activity coordination type that is to be used with the extensible coordination framework described in the WS-Coordination specification. The specification defines two specific agreement coordination protocols for the business activity coordination type: BusinessAgreementWithParticipantCompletion, and BusinessAgreementWithCoordinatorCompletion. Developers can use any or all of these protocols when building applications that require consistent agreement on the outcome of long-running distributed activities.

---

[100] IBM, Microsoft, BEA. Web Services Atomic Transaction (WS-AtomicTransaction). September 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/wsat.asp

[101] IBM, Microsoft, BEA. Web Services Business Activity Framework (WS-BusinessActivity). http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/wsba.asp

5.4 WS-Orchestration

Web services orchestration is about providing an open, standards-based approach for connecting web services together to create higher-level business processes. Standards such as BPEL4W, WSCI, and BPML are designed to reduce the complexity required to orchestrate web services, thereby reducing time-to-market and costs, and increasing the overall efficiency and accuracy of business processes. Without a common set of standards, each organization is left to build their own set of proprietary business protocols, leaving little flexibility for true web services collaboration.

5.4.1 BPEL4WS[102,103]

The Business Process Execution Language for Web Services is an initiative of the industry leaders BEA Systems, IBM, Microsoft, SAP AG, Siebel Systems to drive and ensure interoperability for the description and communication of business processes based on web services.

BPEL4WS defines a notation for specifying business process behaviour based on Web Services. Processes in BPEL4WS export and import functionality by using Web Service interfaces exclusively.

BPEL4WS provides a language for the formal specification of business processes and business interaction protocols. By doing so, it extends the Web Services interaction model and enables it to support business transactions. BPEL4WS defines an interoperable integration model that should facilitate the expansion of automated process integration in both the intra-corporate and the business-to-business spaces.

BPEL4WS represents a convergence of the ideas in the XLANG and WSFL specifications. Both XLANG and WSFL are superseded by the BPEL4WS specification

The specification provides an XML-based grammar for describing the control logic required to coordinate web services participating in a process flow. This grammar can then be interpreted and executed by an orchestration engine, which is controlled by one of the participating parties. The engine coordinates the various activities in the process, and compensates the system when errors occur. BPEL4WS is essentially a layer on top of WSDL, with WSDL defining the specific operations allowed and BPEL4WS defining how the operations can be sequenced. A BPEL document leverages WSDL in three ways:

- Every BPEL process is exposed as a web service using WSDL. The WSDL describes the public entry and exit points for the process.
- WSDL data types are used within a BPEL process to describe the information that passes between requests.
- WSDL might be used to reference external services required by the process.

BPEL4WS provides support for both *executable* and *abstract* business processes. An executable process models the behaviour of participants in a specific business interaction, essentially modelling a private workflow. Abstract processes, modelled as business protocols in BPEL4WS, specify the public message exchanges between parties. Business protocols are not executable and do not convey the internal details of a process flow. Essentially, executable processes provide the orchestration support described earlier while the business protocols focus more on the choreography of the services.

---

[102] IBM, Microsoft, BEA, SAP, Siebel. Business Process Execution Language for Web Services Version 1.1. 5 May 2003. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbizspec/html/bpel1-1.asp.

[103] OASIS. Web Services Business Process Execution Language. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

5.4.2 BPML[104]

The Business Process Management Language (BPML) is a meta-language for describing business processes. The BPML specification provides an abstract model for expressing business processes and supporting entities. BPML defines a formal model for expressing abstract and executable processes that address all aspects of enterprise business processes, including activities of varying complexity, transactions and their compensation, data management, concurrency, exception handling and operational semantics. BPML also provides a grammar in the form of an XML Schema for enabling the persistence and interchange of definitions across heterogeneous systems and modeling tools. BPML itself does not define any application semantics such as particular processes or application of processes in a specific domain; rather it defines an abstract model and grammar for expressing generic processes. This allows BPML to be used for a variety of purposes that include, but are not limited to, the definition of enterprise business processes, the definition of complex Web services, and the definition of multi-party collaborations.

By leveraging the WSCI specification, BPML enables the modeling of end-to-end processes that can be translated into collections of private implementations executed as BPML processes and public interfaces defined using WSCI. Together, they provide an end-to-end view that depicts the role of each individual business process in the overall choreography, and the business activities performed by each role. Both BPML and WSCI share the same underlying process execution model, as well as similar syntaxes. The BPML specification can also be loosely used compared to BPEL4WS, providing similar process flow constructs and activities. The features supported by BPML include persistence, instance correlation, and roles. The language was designed to manage long-lived processes, with persistence supported in a transparent manner. XML exchanges occur between the various participants, with roles and partner components similar to the BPEL constructs. Additionally, BPML supports recursive decomposition, the ability to compose sub-processes into a larger business process. Furthermore, BPML includes a robust exception handling mechanism. Finally, BPML provides the ability to nest processes and transactions, a feature that BPEL currently does not provide.

5.4.3 Evaluation

The ultimate goal of business process languages like BPEL4WS is to abstract underlying Web services so that the business process language effectively becomes the Web services API. BPEL4WS is at the top of the WS-Transaction stack and utilizes WS-Transaction to ensure reliable execution of business processes over multiple workflows, which BPEL4WS logically divides into two distinct aspects. The first is a process description language with support for performing computation, synchronous and asynchronous operation invocations, control-flow patterns, structured error handling, and long-running business transactions. The second is an infrastructure layer that builds on WSDL to capture the relationships between enterprises and processes within a Web services-based environment.

Taken together, these two aspects support the orchestration of Web services in a business process, where the infrastructure layer exposes Web services to the process layer, which then drives that Web services infrastructure as part of its workflow activities.

BPML has some complimentary components to BPEL4WS, both providing capabilities to define a business process. WSCI (see section 5.5.1) is considered a part of BPML: WSCI defines the interactions between the services and BPML defines the business processes behind each service.

[104] BPMI, BPML Specification, http://www.bpmi.org/.

5.5 WS-Choreography

Orchestration takes more of an "inside-out" perspective, describing an executable process from the perspective of one of the partners, while choreography takes more of a collaborative and choreographed approach.

5.5.1 WSCI

The Web Service Choreography Interface (WSCI) is an XML-based interface description language that describes the flow of messages exchanged by a Web Service participating in choreographed interactions with other services.

WSCI describes the dynamic interface of the Web Service participating in a given message exchange by means of reusing the operations defined for a static interface. WSCI works in conjunction with the Web Service Description Language (WSDL), the basis for the W3C Web Services Description Working Group; it can, also, work with another service definition language that exhibits the same characteristics as WSDL.

WSCI describes the observable behavior of a Web Service. This is expressed in terms of temporal and logical dependencies among the exchanged messages, featuring sequencing rules, correlation, exception handling, and transactions. WSCI also describes the collective message exchange among interacting Web Services, thus providing a global, message-oriented view of the interactions.

WSCI does not address the definition and the implementation of the internal processes that actually drive the message exchange. Rather, the goal of WSCI is to describe the observable behavior of a Web Service by means of a message-flow oriented interface. This description enables developers, architects and tools to describe and compose a global view of the dynamic of the message exchange by understanding the interactions with the web service.

5.5.2 Evaluation

BPEL4WS primarily focuses on the creation of executable business processes, while WSCI is concerned with the public message exchanges between web services. WSCI takes more of a collaborative and choreographed approach, requiring each participant in the message exchange to define a WSCI interface.

5.6 Conclusions

The two standards discussed here - the Web Service Choreography Interface (WSCI) and Business Process Execution Language for Web Services (BPEL4WS) - are designed to reduce the inherent complexity of connecting Web services together. Orchestration describes how Web services can interact at the message level, including the business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long-lived, transactional process. With orchestration, the process is always controlled from the perspective of one of the business parties. Choreography tracks the sequence of messages that may involve multiple parties and multiple sources. It is associated with the public message exchanges that occur between multiple Web services.

Each standard takes a somewhat different approach to orchestration and choreography. While BPEL4WS supports the notion of "abstract processes," most of its focus is aimed at BPEL4WS executable processes. BPEL4WS takes more of an *inside-out* perspective, describing an executable process from the perspective of one of the partners, while WSCI takes more of a collaborative and choreographed approach. This requires each participant in the message exchange to define a WSCI interface. Since TrustCoM employs Web services as the underlying technology for representing the services provided by the participants of a virtual organization, TrustCoM will utilize BPEL4WS or WSCI/BPML for defining and executing collaborative business processes within dynamic Virtual Organizations. As these standards are being defined, it is not clear which ones will emerge as industry standards for web services orchestration and choreography.

6 Enabling Technologies

Edited by: Joris Claessens and Christian Geuer-Pollmann
European Microsoft Innovation Center (EMIC)

6.1 Introduction

This chapter covers the generic, enabling technologies which are relevant to TrustCoM, as well as the TrustCoM related implementation-specific tools and platforms, of different TrustCoM partners and third parties. The focus of this chapter is on the underlying technologies beyond those presented for VO frameworks and VO collaboration. Section 6.2 covers the generic Web Services framework, while section 6.3 discusses the emerging Grid technologies. Both the Web Services framework and the Grid technologies include specific mechanisms and technologies related to policies and security and collaborative business processes. These specific mechanisms and technologies are mentioned here, but are discussed in more detail in the respective appropriate other chapters on Policies and Security (Chapter 8) and Collaborative Business Processes (Chapter 5) in this deliverable. Section 6.4 then presents the semantic web and ontology concepts which may impact the TrustCoM framework. Finally, section 6.5 gives an overview of the implementation-specific tools and platforms of the different TrustCoM partners and third parties.

6.2 Web Services

6.2.1 Introduction: conceptual model and architecture

Web Services bring the paradigm of service-oriented architecture in practice. They offer an interoperable framework for stateless, message-based and loosely coupled interaction between software components. These components can be spread across different companies and organisations, can be implemented on different platforms, and can reside in different computing infrastructures. As a common level of interoperability, Web Services are expected to play a central role within TrustCoM.

There are many and broad definitions of the Web Services concept. We here refer to Web Services as services that expose useful functionality on the Internet via XML messages, which are exchanged through a standard protocol, called SOAP. The interface of a Web Service is described in detail in an XML document using WSDL. Finally, a Web Service is registered at a UDDI server, and is as such discoverable. In addition to these basic features of Web Services, it is essential that Web Services are provided in a secure and reliable way, and that they support transactions. Specifications for secure, reliable, and transacted web services[105] have been and are being developed by IBM, Microsoft, and others.

Web Services: Model for Service Oriented Architecture

The Web Services model does not operate on the notion of shared types that require common implementation. Rather, services interact based solely on contracts (WSDL/BPEL4WS for message processing behaviour) and schemas (WSDL/XSD for message structure). This enables the service to describe the structure of messages it can send and/or receive and sequencing constraints on these messages. The separation between structure and behaviour and the explicit, machine verifiable description of these characteristics simplifies integration in heterogeneous environments. Furthermore, this

[105] IBM and Microsoft. Secure, Reliable, Transacted Web Services: Architecture and Composition. September 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnwebsrv/html/wsoverview.asp.

information sufficiently characterizes the service interface so that application integration does not require a shared execution environment to create the messages structure or behaviour.

The service-oriented model assumes a fully distributed environment where it is difficult, if not impossible, to propagate changes in schema and/or contract to all parties that have encountered a service. Service-orientation implies that contracts and schema should remain backward compatible and may contain information that is incompletely understood by particular processing systems. For that reason, the contract and schema technologies designed for use in service-oriented designs enable more flexibility than traditional object oriented interfaces. In particular, services use features such as XML element wildcards (e.g., xsd:any), schema extensions and optional SOAP header blocks to evolve services in ways that do not break deployed applications. These characteristics are the key to the composability of Web Services as discussed further. Procedural and Object-oriented designs typically equate type compatibility with semantic compatibility. Service-orientation provides a richer model for determining compatibility. Structural compatibility is based on contract (WSDL and optionally BPEL4WS) and schema (XSD) and can be validated.

One of the main concepts of a service-oriented architecture (SOA) is the flexible binding of services. Traditional procedural, component and object models bind components together through references (pointers) or names. A SOA supports more dynamic discovery of service instances that provides the interface, semantics and service assurances that the requestor expects. In procedural or object-oriented systems, a caller typically finds a server based on the types it exports or a shared name space. In an SOA system, callers can search registries such as UDDI for a service. The loose binding enables alternative implementations of web services to be used to address dynamic business requirements in VOs.

Web Services operate in an asynchronous environment, as opposed to traditional synchronous RPC-like middleware systems. Web Services do not require a common execution environment, and provide message-oriented, non-blocking, distributed computing. An SOA explicitly assumes that communication, availability, and type errors commonly happen. Web Services therefore explicitly rely on technologies dealing with asynchronous messaging, transactions, secure and reliable messaging, redundant deployment, protection against malicious messages, etc.

Composability of Web Services

A key element in the Web Services technology is the so called composability. Web Services specifications are being created in such a way that every specification is independent from the others, however they can be combined (composed) to achieve more powerful and complex solutions. This approach has a number of advantages which are important in the context of TrustCoM. Reliability, security, transaction capabilities and other features can be provided without adding unnecessary complexity to the specification. Moreover, the specifications are easily extended with new concepts, tools and services, by adding new layers and elements.

The following example, Figure 37 illustrates the composability feature, and shows how WS-Addressing, WS-Security, and WS-ReliableMessaging elements are added to a SOAP message. These elements are independent and can be used independently without altering the processing of other elements.

	<S:Envelope … > <S:Header>
WS-Addressing	<wsa:ReplyTo> <wsa:Address>http://business456.com/User12</wsa:Address> </wsa:ReplyTo> <wsa:To>http://fabrikam123.com/Traffic</wsa:To> <wsa:Action>http://fabrikam123.com/Traffic/Status</wsa:Action>
WS-Security	<wssec:Security> <wssec:BinarySecurityToken ValueType="wssec:X509v3" EncodingType="wssec:Base64Binary"> dWJzY3JpYmVyLVBlc…..eFw0wMTEwMTAwMD </wssec:BinarySecurityToken> </wssec:Security>
WS-Reliable Messaging	<wsrm:Sequence> <wsu:Identifier>http://fabrikam123.com/seq1234</wsu:Identifier> <wsrm:MessageNumber>10</wsrm:MessageNumber> </wsrm:Sequence>
	</S:Header> <S:Body> <app:TrafficStatus xmlns:app="http://highwaymon.org/payloads"> <road>520W</road> <speed>3MPH</speed> </app:TrafficStatus> </S:Body> </S:Envelope>

Figure 37 Composability of Web Services

Web Services specifications

The figure below presents the complete Web Services specifications "stack". Web Services provide XML-messages-based interaction between components. The messages are exchanged using a messaging framework that can run on top of different transport protocols (typically http, but also others are possible). Three different sets of specifications ensure that messages are exchanged in a secure, reliable, and transacted way. Messaging, security, reliability, and transactions specifications make extensive use of metadata. The metadata specifications allow to describe and communicate various and necessary properties of a web service, such as its messaging interface, its owner and the business description, its (security) policy, etc. Business processing can be performed on top of the framework for secure, reliable, and transacted messaging.

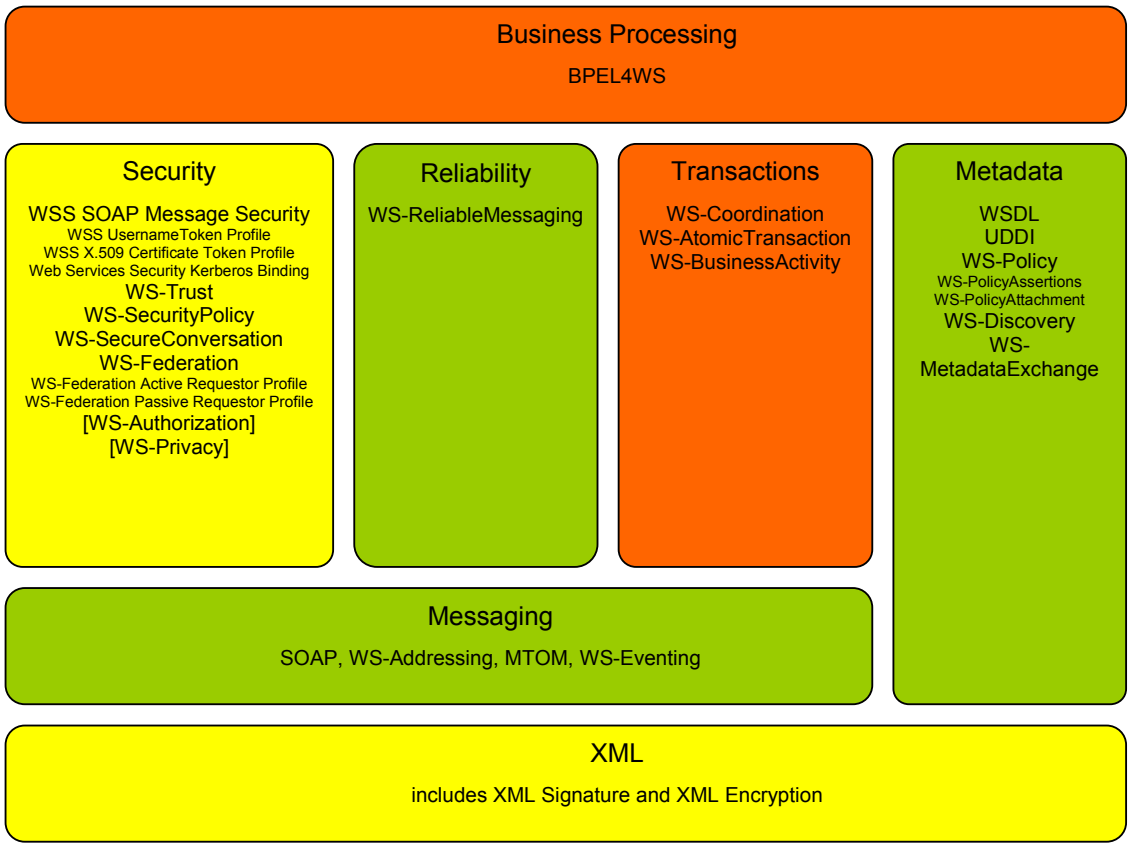


Figure 38 The Web Services framework specifications “stack”

Web Services are currently standardised in different bodies. The core underlying messaging related technologies (such as XML, SOAP, WSDL) are standardised within the W3C. Other mechanisms (such as UDDI and WS-Security) are standardised within OASIS. The more recent specifications are proposed as industry initiatives by IBM, Microsoft, and others.

This section presents web services from a general enabling point of view, and particularly discusses the aspects of (reliable) messaging and metadata. The specific Web services security and Web services transactioning and business processing specifications are respectively covered in Chapter Policies and Security and Chapter Collaborative Business Processes .

6.2.2 Web Services Specifications

6.2.2.1 (Reliable) Messaging

W3C SOAP[106]: SOAP Version 1.2 is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. "Part 1: Messaging Framework" defines, using XML technologies, an extensible messaging framework containing a message construct that can be exchanged over a variety of underlying protocols.

[106] W3C. SOAP Version 1.2. W3C Recommendation 24 June 2003. http://www.w3.org/TR/soap/.

WS-Addressing[107]: WS-Addressing provides transport-neutral mechanisms to address Web services and messages. Specifically, this specification defines XML elements to identify Web service endpoints and to secure end-to-end endpoint identification in messages. This specification enables messaging systems to support message transmission through networks that include processing nodes such as endpoint managers, firewalls, and gateways in a transport-neutral manner.

W3C MTOM[108]: MTOM describes an abstract feature and a concrete implementation of it for optimizing the transmission and/or wire format of SOAP messages. The concrete implementation relies on the XOP format for carrying SOAP messages.

WS-Eventing[109]: Web services often want to receive messages when events occur in other services and applications. A mechanism for registering interest is needed because the set of Web services interested in receiving such messages is often unknown in advance or will change over time. WS-Eventing defines a protocol for one Web service (called an "event sink") to register interest (called a "subscription") with another Web service (called an "event source") in receiving messages about events (called "notifications"). To improve robustness, the subscription is leased by an event source to an event sink, and the subscription expires over time. An event source may allow an event sink to renew the subscription. WS-Eventing defines means to create and delete event subscriptions, and to define expiration for subscriptions and allow them to be renewed. The specification also defines how an event sink can determine which subscriptions it is receiving notifications for, and how one event sink can subscribe on behalf of another. Through the composable nature of the Web Services specifications, WS-Eventing leverages the other Web service specifications for secure, reliable, transacted message delivery. WS-Eventing provides extensibility for more sophisticated subscription scenarios, and may as such support more distributed dissemination and correlation of complex events.

WS-ReliableMessaging[110]: WS-ReliableMessaging describes a protocol that allows messages to be delivered reliably between distributed applications in the presence of software component, system, or network failures. The protocol is described in this specification in an independent manner allowing it to be implemented using different network transport technologies. To support interoperable Web services, a SOAP binding is defined within this specification.

6.2.2.2 Meta data

W3C WSDL[111]: WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate, however, the only bindings described in this specification describe how to use WSDL in conjunction with SOAP 1.1, HTTP GET/POST, and MIME.

---

[107] IBM, Microsoft, BEA. Web Services Addressing (WS-Addressing). March 2004. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-addressing.asp.

[108] W3C. SOAP Message Transmission Optimization Mechanism. W3C Working Draft 09 February 2004. http://www.w3.org/TR/soap12-mtom/.

[109] Microsoft, BEA, TIBCO. Web Services Eventing (WS-Eventing). January 2004. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-eventing.asp.

[110] IBM, Microsoft, BEA, TIBCO. Web Services Reliable Messaging Protocol (WS-ReliableMessaging). March 13, 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-reliablemessaging.asp.

[111] W3C. Web Services Description Language (WSDL) 1.1. W3C Note 15 March 2001. http://www.w3.org/TR/wsdl.

OASIS UDDI[112]: The Universal Description and Discovery Interface (UDDI) specifies a "yellow pages" or repository for businesses and the Web Services they expose. The UDDI specifications describe the Web services, data structures and behaviours of all instances of a UDDI registry. Such a UDDI registry is in fact a 'metadata aggregation' service. UDDI allows organisations to publish business relevant information along with the specifics about the services they provide in a central repository. UDDI enables the discoverability of Web Services, and allow parties to search for Web Services that could meet the particular needs in their business processes.

WS-Discovery[113]: WS-Discovery defines a multicast discovery protocol to locate services. By default, probes are sent to a multicast group, and target services that match return a response directly to the requester. To scale to a large number of endpoints, the protocol defines the multicast suppression behavior if a discovery proxy is available on the network. To minimize the need for polling, target services that wish to be discovered send an announcement when they join and leave the network.

WS-MetadataExchange[114]: To bootstrap communication with a Web service, WS-MetadataExchange defines three request-response message pairs to retrieve three types of metadata: one retrieves the WS-Policy associated with the receiving endpoint or with a given target namespace, another retrieves either the WSDL associated with the receiving endpoint or with a given target namespace, and a third retrieves the XML Schema with a given target namespace. Together these messages allow efficient, incremental retrieval of a Web service's metadata.

6.2.2.3 WS-I

The complete Web Services framework consists of an extensive set of flexible individual specifications. Although all of these specifications are (being) standardized, it may be very difficult to effectively build interoperable applications, due to the many options that can be taken during implementation and that are supported through the flexibility of the specifications. Therefore, in addition to the various individual Web Services standards, WS-I is developing a core collection of profiles that support interoperability for general purpose Web services functionality. A Profile is a named group of Web services specifications at specific version levels, along with conventions about how they work together.

The WS-I Basic Profile 1.1[115] consists of a set of non-proprietary Web services specifications, along with clarifications and amendments to those specifications that promote interoperability. The WS-I Basic Profile covers aspects of SOAP, WSDL, UDDI, and HTTPS.

The WS-I Basic Security Profile[116] also consists of a set of non-proprietary Web services specifications, along with clarifications and amendments to those specifications, which promote interoperability. The WS-I Basic Security Profile covers transport layer security (HTTP over TLS) and SOAP message security. The latter defines proper, interoperable usage of security tokens (particularly username/password and X.509 certificates), timestamps, id references, security processing order, SOAP actors, XML signature and XML encryption, and security of SOAP attachments.

---

[112] OASIS. Universal Description, Discovery and Integration. UDDI Version 3.0.1. 14 October 2003. http://uddi.org/pubs/uddi_v3.htm.

[113] Microsoft, BEA, Canon, Intel. Web Services Dynamic Discovery (WS-Discovery). February 2004. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-discovery.asp.

[114] IBM, Microsoft, BEA, SAP. Web Services Metadata Exchange (WS-MetadataExchange). March 2004. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-metadataexchange.asp.

[115] Web Services Interoperability. WS-I Basic Profile Version 1.1. Working Group Draft. 6 December 2003. http://www.ws-i.org/.

[116] Web Services Interoperability. WS-I Basic Security Profile Version 1.0. Working Group Draft. 12 May 2004. http://www.ws-i.org/.

6.2.3 Application to TrustCoM

Web services provide a standardized framework for interoperable, secure, reliable and transacted messaging, and constitute the ideal underlying service-oriented messaging framework for collaboration within dynamic Virtual Organization across enterprises.

Metadata can be associated with Web services in different ways and for different purposes. Metadata allows describing the interfaces of web services, to register web services, to discover web services, and to formulate and attach policies to web services. The overall web services messaging framework and the generic metadata mechanisms should support and enable technologies for interoperable trust and contract management in VOs.

6.3 Grid Technologies

6.3.1 Grid Concepts and requirements

Grid computing enables the virtualisation of different kinds of resources such as computing, data and network resources. These resources are provided under certain conditions to members of a virtual organisation. These resources can be combined to build a dynamic application e.g. by the definition of cross organisational workflows consuming and using the resources offered. So the vision of having a virtual computing system, as defined at the beginning of Grid computing has been extended to a virtual IT infrastructure concept including resources on all levels from network up to the application layer. Additionally the increased number of participants in such settings imposed new requirements on management for such complex systems.

6.3.1.1 Virtual Organisations

As we already mentioned in the introduction, the Grid computing concept has been generalised to cover a *virtual organisation*, defined as any dynamic collection of individuals and enterprises, which are required to share resources to achieve a common goal. This generalisation effectively indicates sharing of information and knowledge resources in addition to computation and data.

To explain the idea of Grid computing the metaphor of a power distribution grid is commonly used. The goal of a power grid is the provision of electrical power wherever and whenever you need it. It does not matter if you would like to use a light bulb or a drill – just plug in. The idea of Grid computing where you just plug in and use the provided resources, is promising, but has three main differences to a power grid:

1. **Different kinds of resources:** As the Grid computing concept matures, there is a movement from sharing hardware resources to sharing data, and then to sharing applications with the view of sharing information and effectively knowledge. Indeed, for Grid computing to evolve to a comparable usability level to a power grid, the user will expect to consume at least three different resources: hardware, data and applications.
2. **Different kinds of interaction:** The heterogeneity of resource sharing inevitably results into heterogeneity of interaction. Moreover, businesses as well as private users will demand for different service levels, in accordance to criteria such as quality and price but also including security and trust.
3. **Dynamic resource allocation and integration:** A grid based on different kinds of shared computing resources requires a dynamic way to use these resources. This allows users to contribute resources whenever desired in addition to consuming resources on demand.

Currently the applications driving the development of this infrastructure are large-scale scientific collaborations, which have a clear need for the collaborative use of resources, both data and computational, and established communities which can pool their resources for

common goals. Tools supporting this kind of Grid computing concept emerged in the last years, most notably Globus[117], Unicore[118] and Avaki/Legion[119].

A branch of the Grid computing community is moving towards commercial applications of the Grid concepts, such as the EU projects GRASP[120], GRIA[121] and GEMSS[122]. Grid concepts are crucial for commercial computing not primarily as a means of enhancing capability, but mostly as a solution for new challenges relating to the on-demand creation of reliable, scalable, and secure distributed systems. As we have already explained, these challenges derive from the current need to decompose and distribute previously monolithic host-centric services through the network. This development is driven by well-established technology trends and long-lasting commercial pressures. In particular, the emergence of Service Providers (SPs) of various types, such as webhosting SPs, content distribution SPs, applications SPs, and storage SPs provides fertile grounds for the transfer of GRID technologies from eScience to eBusiness. By exploiting economies of scale, SPs aim to take standard e-business processes and provide them to multiple customers with a superior cost/performance ratio. Unlike the computing service companies of the past, which merely provided offline batch-oriented processes, resources provided by service providers offering continuous, on-demand access are often tightly integrated with enterprise computing infrastructures and used for business processes that span both in-house and outsourced resources.

Such SPs face their own technical challenges. To achieve economies of scale, they require server infrastructures that can be easily customised on demand to meet specific customer needs. Typical requirements for such infrastructures include:

- Supporting dynamic resource allocation in accordance with service-level agreement policies, efficient sharing and reuse of IT infrastructure at high utilization levels, and distributed security
- Delivering consistent response times and high levels of availability—which in turn drives a need for end-to-end performance monitoring and real-time reconfiguration.
- Support fast service creation and deployment to adapt to changing user requirements

Our experience indicates that the following problem areas need to be properly addressed for Grid computing to become successful in this domain:

- **Interoperability**: One of the main advantages of Grid computing is the on-demand usage of others resources. This increases performance and adaptability by supporting the use of the most suitable computing resources for solving the problem at hand, but demands for establishing means of communication between different systems that are independent of operating system or programming model. This is currently addressed through the Open Grid Services Architecture (OGSA) initiative of the Global Grid Forum (GGF).
- **Security Management**: Using Grid computing for different kinds of scenarios in a business world, demands for effective security policy management. There is still some work to be done.
- **Reliability**: Using Grid computing in a business context will also bring about various QoS requirements including expectations of high availability and reliable service. This

[117] The Globus Project, http://www.globus.org/

[118] Uniform Access to Computing Resources over the Internet, http://www.unicore.org/

[119] Avaki, http://www.avaki.com/

[120] Dimitrakos et. al, Overview on an architecture enabling Grid based Application Service Provision, Proceedings of the 2nd Across Grids Conference, Cyprus, http://grid.ucy.ac.cy/axgrids04/AxGrids/papers/E00-1591225682.pdf.

[121] Grid for Industrial Application, http://www.gria.org/

[122] Grid-Enabled Medical Simulation Services, http://www.ccrl-nece.de/gemss/

calls for designing advanced resource management systems that are able to guarantee reliability in service provision.

- **Scalability**: From a technical perspective, scalability together with dynamic resource allocation and integration are defining the Grid computing concept. From a business perspective, scalability must be extended to cover the negotiation and execution of potentially complex and interdependent service level agreements.

6.3.1.2 Resource Virtualisation

Another basic concept is the resource virtualization. The goal is to hide the complexity of the resource usage. Existing Grid toolkits solved the problems of providing this kind of virtualization for simple resources such as storage capacity and also computational systems as this kind of resources are rather easy to describe and similar to use. As outlined above the move of Grid computing away from the basic computational Grids and Data Grids towards the sharing of information and knowledge through shared applications impose more requirements. Beside the virtualization of IT resources such as computing cycles application and services offered by different entities must be sufficiently described in order to allow an on demand integration of theses parts e.g. using workflow descriptions to dynamic applications. This will require a much more comprehensive way of describing resources where the Grid computing frameworks will have to adopt technology from the Semantic Web area leveraging Grid computing to a Semantic Grid infrastructure.

The virtualization is enabling the usage of resources for the user in a single unified way, hiding the inherent complexity.

6.3.1.3 Conventional Grid Elements

Grid systems such as Globus Toolkit 2 started with a multilayer view representing the different level of virtualization from elements close to hardware up to the application layer. This layer scheme still provides a good starting point for the functional components needed for building a Grid system. The layers are defined as follows and are outlined further below in the discussion on Grid architectures:

- **The fabric layer** typically constitutes computational resources, storage resources, network resources, and code repositories.
- **The connectivity layer** deals with easy and secure communication by providing single sign on, delegation, integration with various local security solutions, and user-based trust relationships.
- **The single resource layer** is concerned with individual resources, and the two primary classes of resource layer protocols are information protocols and management protocols.
- **The collective multiple resources layer** provides directory services, co-allocation, scheduling, and brokering services, monitoring and diagnostics services, data replication services, grid-enabled programming systems, workload management systems and collaboration frameworks (problem solving environments), etc.

6.3.2 Grid Architectures

This section provides in section 6.3.2.1 an overview of the layered Grid architectures as it has been used for conventional Grid Systems such as the Globus Toolkit 2 that focuses on a classification of protocols that are useful for sharing resources within a Grid.

Section 6.3.2.2 provides an overview of Agent Grid architectures that are extending the software agent paradigm in order to cater for resource sharing and management within Grid infrastructures.

The Open Grid Service Architecture and the realization approaches OGSI and WSRF are discussed in greater detail in a separate section 6.3.3.

6.3.2.1 Conventional Grid Architecture

In a conventional Grid architecture, four levels of management can be distinguished: *fabric*, *connectivity*, *single resource*, and *collective* multiple resources. The fabric layer typically constitutes computational resources, storage resources, network resources, and code repositories. The connectivity layer deals with easy and secure communication by providing single sign on, delegation, integration with various local security solutions, and user-based trust relationships. The resource layer is concerned with individual resources, and the two primary classes of resource layer protocols are information protocols and management protocols. The collective multiple resources layer provides directory services, co-allocation, scheduling, and brokering services, monitoring and diagnostics services, data replication services, grid-enabled programming systems, workload management systems and collaboration frameworks (problem solving environments), etc. In the rest of this section we review in more detail this layered architecture structure.

This layered architecture does not aim to provide a complete enumeration of all required protocols, services, APIs, and SDKs, but rather to provide a general classification of the essential components. It can be therefore understood as an extensible, open architectural structure within which one can classify implementations offering concrete solutions to essential issues underpinning the operation of Virtual Organisations (VOs)[123]. This architectural structure follows [124] in organizing component classes into layers (See also Figure 39). Components within each layer share common characteristics but can build on capabilities and behaviours provided by any lower layer. Clearly, this architectural structure is high level and places few constraints on design and implementation. It therefore merely serves as an effective classification of the various types of protocol that may be applicable to the various Grid implementations.

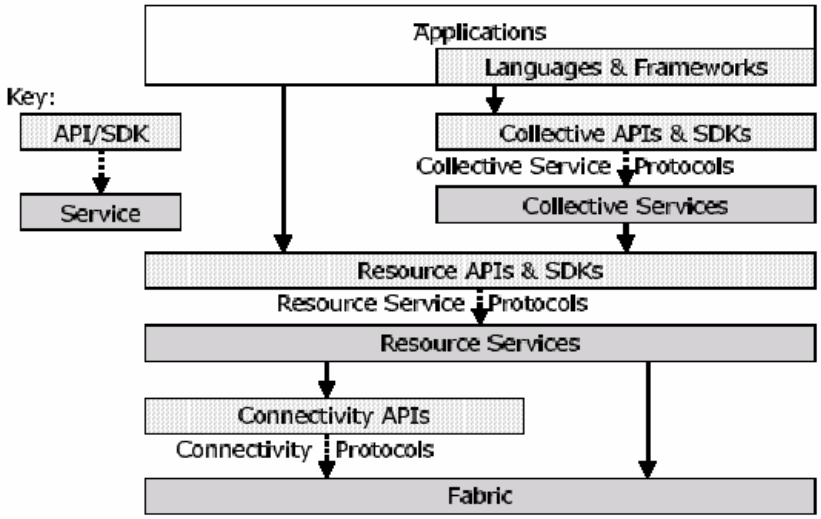


Figure 39 Layers of the Grid protocol stack superimposed over the analogous stack of Internet Protocols.

6.3.2.1.1 Fabric layer

The Grid *Fabric* layer provides the resources to which shared access is mediated by Grid protocols.

[123] Within this section the term Virtual Organisations

[124] Foster, I., Kesselman, C. and Tuecke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications, 15 (3). 200-222. 2001. www.globus.org/research/papers/anatomy.pdf.

A "resource" may be a logical entity, such as a distributed file system, computer cluster, or distributed computer pool; in such cases, a resource implementation may involve internal protocols, which may need to interface with the Grid architecture but are outside its scope. Consequently, from a Grid architecture perspective we focus on the interfaces of the Fabric to the local control.

The resources managed by a Grid infrastructure can be classified in the following categories identified in[125]. We provide a brief description of the indicative functionality that relates to each type.

- *Computational resources*: The management of shared computational resources requires mechanisms for starting programs and for monitoring and controlling the execution of the resulting processes. These often need to be supported by mechanisms that allow control over the resources allocated to processes and advance reservation mechanisms.
- *Storage resources*: One would typically expect mechanisms for putting and getting files, often supported by third-party file transfer mechanisms and functionality for reading and writing subsets of a file as well as executing remote data selection or reduction functions. In addition, advance reservation mechanisms and mechanisms that allow control over the resources allocated to data transfers (space, disk bandwidth, network bandwidth, CPU) are often useful.

For both types of resource, effective deployment requires the presence of appropriate enquiry functions. These support determining the hardware and software characteristics and as well as relevant state information, such as available space and bandwidth utilization.

Special cases of storage resources include:

- *Code repositories*, which require mechanisms for managing versioned source and object code.
- *Catalogue,* which require mechanisms for implementing catalogue query and update operations.
- *Network resources*: Mechanisms that provide control over the resources allocated to network transfers (e.g., prioritization, reservation) are often useful for managing such resources. Again, the deployment of the management mechanisms requires the presence of "enquiry function" implementations that support determining the network characteristics and load.

Fabric components are expected to implement the local operations that are specific to a particular resource (whether physical or logical) as a result of sharing operations at higher levels. This gives rise to a subtle interdependence between the functionality implemented at the Fabric level and the sharing operations supported by the Grid architecture. There is a trade-off: requiring richer Fabric functionality enables the Grid architecture to provide more sophisticated sharing operations, while simplifying the requirements on the functionality of the Fabric contributes to an easier and less expensive deployment of the Grid infrastructure.

6.3.2.1.2 Connectivity layer

The *Connectivity* layer defines core communication and authentication protocols required for Grid-specific network tasks. Communication protocols enable the exchange of data between Fabric layer resources. Authentication protocols build on communication services to provide cryptographically secure mechanisms for verifying the identity of users and resources.

Communication requirements include transport, routing, and naming. While alternatives certainly exist, it is usually expected that these protocols are drawn from the TCP/IP protocol stack: specifically, the Internet (IP and ICMP), transport (TCP, UDP), and application (DNS, OSPF, RSVP, etc.) layers of the Internet layered protocol architecture[126]. Of course, future

[125] Foster, I., Kesselman, C., Nick J M., and Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.

[126] Baker, F. Requirements for IP Version 4 Routers, IETF, RFC 1812, 1995. http://www.ietf.org/rfc/rfc1812.txt

Grid communications may demand new protocols that take into account particular types of network dynamics. As we explain in the next section there is already an emerging trend of using SOAP extensions for communication between services in the increasingly popular service oriented paradigm of Grid architectures.

Furthermore complexity of the security problem at the connectivity layer makes it important that any solutions be based on existing standards whenever possible. As with communication, many of the security standards developed within the context of the Internet protocol, as well as Web Service security are applicable.

6.3.2.1.3 Resource layer

The Resource layer builds on Connectivity layer communication and authentication protocols to define protocols (and APIs and SDKs) for the secure negotiation, initiation, monitoring, control, accounting, and payment of sharing operations on individual resources. Resource layer implementations of these protocols call Fabric layer functions to access and control local resources. Resource layer protocols are concerned entirely with individual resources and hence ignore issues of global state and atomic actions across distributed collections. The latter are addressed in the Collective layer.

Two primary classes of Resource layer protocols can be distinguished:

- *Information protocols* are used to obtain information about the structure and state of a resource, for example, its configuration, current load, and usage policy, etc.
- *Management protocols* are used to negotiate access to a shared resource, specifying, for example, resource requirements (including advanced reservation and QoS) and the operation(s) to be performed, such as process creation, or data access. Since management protocols are responsible for instantiating sharing relationships, they must ensure that the requested protocol operations are consistent with the policy under which the resource is to be shared. Application areas of management protocols include, accounting, payment, monitoring and controlling operations (eg. monitoring the status of an operation and terminating it when appropriate).

Examples of such protocols include the Globus GRIP (based on the LDAP standard) which is used to define a standard resource information protocol and the associated information model, GRRP which is used for registering resources, GRAM which is based on HTTP and is used for allocation of computational resources and for monitoring and control of computation on those resources, and GridFTP which extends FTP to manage data access by providing functionality for partial file access and management of parallelism for high-speed transfers.

6.3.2.1.4 Collective layer

The collective layer complements the resource layer by providing protocols and services (and APIs and SDKs) that are not associated with any one specific resource but rather are global in nature and capture interactions across collections of resources.

Collective components can usually implement a wide variety of sharing behaviours without placing new requirements on the resources being shared. Examples include:

- *Directory services* that allow VO participants to discover the existence and/or properties of VO resources. A directory service may allow its users to query for resources by name and/or by attributes such as type, availability, or load. Resource-level GRRP and GRIP protocols are used to construct directories.
- *Co-allocation, scheduling, and brokering* services allow VO participants to request the allocation of one or more resources for a specific purpose and the scheduling of tasks on the appropriate resources. Examples include AppLeS, Condor-G, Nimrod-G, and the DRM broker.
- . *Monitoring and diagnostics services* support the monitoring of VO resources for failure, malicious attack (“intrusion detection”), overload, and so forth.

- *Data replication services* support the management of VO storage (and perhaps also network and computing) resources to maximise data access performance with respect to metrics such as response time, reliability, and cost.
- *Grid-enabled programming systems* enable familiar programming models to be used in Grid environments, using various Grid services to address resource discovery, security, resource allocation, and other concerns. Examples include Grid-enabled implementations of the Message Passing Interface such as [127] [128] and manager-worker frameworks.
- *Workload management systems and collaboration frameworks* that provide for the description, use, and management of multi-step, asynchronous, multi-component workflows.
- *Software discovery services* discover and select the best software implementation and execution platform based on the parameters of the problem being solved. Examples include NetSolve[129] and Ninf[130].
- *Community authorisation servers* enforce community policies governing resource access, generating capabilities that community members can use to access community resources. These servers are expected provide a global policy enforcement service by building on resource information, and resource management protocols (in the Resource layer) and security protocols in the Connectivity layer
- *Community accounting and payment services* gather resource usage information for the purpose of accounting, payment, and/or limiting of resource usage by community members.
- *Collaboratory services* support the coordinated exchange of information within potentially large user communities, whether synchronously or asynchronously

6.3.2.1.5 Application layer

The Application layer comprises the user applications that operate within a VO environment. Applications are constructed in terms of, and by calling upon, services defined at any layer, with the exception that the Fabric layer – being an interface to a local resource - is generally called via one of the higher layers. At each layer, well-defined protocols provide access to some useful service: resource management, data access, resource discovery, and so forth. At each layer, APIs may also be defined whose implementation (ideally provided by third-party SDKs) exchange protocol messages with the appropriate service(s) to perform desired actions.

Of course, what is referred to as an “application” may in itself be a subsystem using other not Grid-specific sophisticated architectural framework, application specific protocols, services, APIs and libraries. A further extension of this layer is beyond the scope of this classification which focuses on the core Grid infrastructure.

6.3.2.2 Grid Agents architecture

A number of initiatives to apply agents in computational grids have been initiated in recent years.

---

[127] MPICH, http://www-unix.mcs.anl.gov/mpi/mpich/

[128] PACX-MPI, http://www.hlrs.de/organization/pds/projects/pacx-mpi/

[129] Casanova, H. and Dongarra, J. NetSolve: A Network Server for Solving Computational Science Problems. International Journal of Supercomputer Applications and High Performance Computing, 11(3):212-223. 1997.

[130] Nakada, H., Sato, M. and Sekiguchi, S. Design and Implementations of Ninf: towards a Global Computing Infrastructure. Future Generation Computing Systems, 1999.

Agent-based approaches may facilitate the management of large-scale Grids. However most of the current agent-based systems are not developed for large-scale environments. CoABS and AgentSpace are notable exceptions.

AgentScape[131], a large-scale distributed agent system, currently under development, designed to support heterogeneity and interoperability, facilitates extensibility: it is relatively easy to build agent environments "on top of" AgentScape. AgentScape is also relatively easily adapted to different (lower-level) operating systems and network infrastructures. As such, AgentScape aims to be relatively easily integrated with other environments and support agent-based approaches to grid resource management. However, AgentSpace is still at its infancy, an elaborate management system has to be incorporated to deal with performance, security, fault tolerance, and accounting. Furthermore scalable services for agents, such as name, location, and directory services. Agent-based scheduling and resource allocation algorithms have to be developed and evaluated.

Manola and Thompson present in [132] an overview of different perspectives to Grid environments and describe DARPA's Control of Agent-Based Systems (CoABS) agent grid. In the CoABS Grid, a number of application level and functional requirements hold. Specifically, applications are considered to have multi-year lifetimes, evolving and changing requirements, are adaptable and scalable, and allows for system management without explicitly monitoring all components all the time. Practically, agent technology is expected to help to provide more reliable, scalable, survivable, evolvable, adaptable systems, and help to solve data blizzard and information starvation problems. From a functional point of view, the CoABS Grid knows not only about agents, but also about their computational requirements, and about available computational (and other) resources. Hence, the CoABS Grid provides a unified, heterogeneous distributed computing environment in which computing resources are seamlessly linked.

Bradshaw et al.[133] remark that "cyberspace" is currently a lonely, dangerous, and relatively impoverished environment for software agents. Consequently, most of today's agents are designed for "solitary, poor, nasty, brutish, and short" lives of narrow purpose in a relatively bounded and static computational world. They argue that focusing greater attention to making the environment in which agents operate more capable of sustaining various types of agents and collaboration groups, would simplify some of these problems. The CoABS Grid provides the infrastructure for large-scale integration of heterogeneous agent frameworks. The CoABS Grid capabilities have been extended by integrating the NOMADS agent environment for strong mobility and safe execution and the KAoS framework for policy-based management of agent domains to support long-lived agents and their communities.

Other types of applications of agents in distributed parallel computing include:

- Rana and Walker identify in [134] the need to combine problem-specific problem solving environments (PSEs), facilitating interoperability between various tools and specialised algorithm each PSE supports. An agent based approach to integrate services and resources for establishing multi-disciplinary PSEs is described, in which specialised agents contain behavioural rules, and can modify these rules based on their interaction with other agents and with the environment in which they operate.

[131] Overeinder, B.J., Wijngaards, N.J.E., van Steen, M., and Brazier, F.M.T. Multi-Agent Support for Internet-Scale Grid Management. In proceedings of AISB02, London 2002.

[132] F. Manola and C. Thompson. Characterizing the Agent Grid. http://www.objs.com/agility/techreports/990623-characterizing-the-agentgrid.html, June 1999.

[133] M. Bradshaw, N. Suri, A. J. Ca˜nas, R. David, K. Ford, R. Hoffman, R. Jeffers, and T. Reichherzer. Terraforming cyberspace. Computer, 34(7):48–56, July 2001.

[134] O. F. Rana and D. W. Walker. 'The Agent Grid': Agent-basedresource integration in PSEs. In Proceedings of the 16th IMACS World Congress on Scientific Computing, Applied Mathematics and Simulation, Lausanne, Switzerland, August 2000.

- Master-slave computations in wide-area distributed environments[135] provide another type of application of agents in distributed or parallel computing. In these systems, large computations are initiated under control of a coordinating agent that distributes the computation over the available resources by sending mobile agents to these resources. In this perspective it is in some way similar to the Condor system or the SETI@home experiment, which also incorporate coordination and distributing the computation of the available resources. Essentially, the added value of the distributed computing agent systems is similar to the agent grid: coordination and seamless integration of the available distributed resources.

6.3.3 The open Grid Service Architecture (OGSA)

This section describes the Open Grid Service Architecture (OGSA), which extends the layered approach described in section 6.3.2.1 above by providing a service-oriented view of Grid architectures focusing on the operational functionality of a Grid and how Grid technologies can be implemented and applied, especially in relation to the emerging Web Service standards.

An obvious difference between the layered and service-oriented perspectives is that, while a layered view is structured in terms of the protocols required for interoperability among VO components, the service-oriented view focuses on the nature of the *services* that respond to protocol messages. From a service-oriented perspective, a Grid is viewed as an extensible set of *Grid services* that may be aggregated in various ways to meet the needs of VOs. In fact VO can be defined in part by the services that they operate and share.

The rest of this section is structured as follows: First, we provide an overview of service-oriented architectures and then we proceed by analysing the essential extensions of the Web Service technology and architectures that are introduced by OGSA.

6.3.3.1 Service Oriented Architectures for the Grid

A *service* is a network-enabled entity that provides some capability. Service oriented architectures (SOA) aim to provide the shared organising principles that underpin the collaborative operation of services in open dynamic distributed systems. Service-oriented architectures focus on how services are described and organised to support their dynamic, automated discovery and use at run-time. (In contrast, for example, to systems based on manually hardwired interactions, such as those used in EDI systems.) SOA provide the architectural concept behind Web Services, analysed in section 6.2.1 of this deliverable.

The fundamental characteristics of SOA include:

- Service providers publishing the availability of their services.
- Service brokers registering and categorising published services and providing search services.
- Service requesters using broker services to find a needed service and then employing that service.
- Computer-accessible, hierarchical categories, or taxonomies, based upon what the services in each category do and how they can be invoked. These taxonomies aim to assist the dynamic automated discovery of appropriate services.

The collaborations among these main roles (provider, broker, requester) are supported by a standardised network protocol. *Service descriptions* in a standard machine-readable format (such as XML) are associated with each service. These service descriptions are key to all

[135] R. Ghanea-Hercock, J. C. Collis, and D. T. Ndumu. Cooperating mobile agents for distributed parallel processings. In Proceedings of the Third Annual Conference on Autonomous Agents, pages 398–399, Seatle, WA, April 1999.

three roles in that they provide the information needed to categorise, choose, and invoke an e-business service.

It has to be emphasised that the focus of SOA is different to that of *service-based* architectures which focus instead on service-to-service message protocols, or on the details of how the various servers communicate rather than what they say to each other. Within a single corporate system, where the entire system is under the control of one group, a service-based approach can be used to break overly rigid legacy systems into collaborating services that provide dramatic improvements in flexibility and maintainability. However, service-based techniques alone do not scale beyond the span of control of an architecture group that defines and manages the semantic definitions of the services. This span is usually no larger than a single corporation.

In a service-oriented view, we can partition the interoperability problem into two subproblems, namely the definition of service interfaces and the identification of the protocol(s) that can be used to invoke a particular interface—and, ideally, agreement on a standard set of such protocols.

A service-oriented view addresses the need for standard interface definition mechanisms, local/remote transparency, adaptation to local OS services and uniform service semantics. A service-oriented view also simplifies the encapsulation mechanism that is necessary behind a common interface of diverse implementations. Such a mechanism allows for consistent resource access across multiple heterogeneous platforms with local or remote location transparency, and enables mapping of multiple logical resource instances onto the same physical resource and management of resources within a VO based on composition from lower-level resources. It also supports the composition of services to form more sophisticated services—without regard for how the services being composed are implemented, and underpins the ability to map common service semantic behaviour seamlessly onto native platform facilities.

Virtualisation (a term used in [136] for the encapsulation mechanism that is necessary behind a common interface of diverse implementations) is easier if service functions can be expressed in a standard form, so that any implementation of a service is invoked in the same manner. In the case of Web Services and OGSA, the XML based WSDL convention is adopted to support a service interface definition that is distinct from the protocol bindings used for service *invocation*. The service interface definition and access binding are also distinct from the *implementation* of the functionality of the service.

A service can support multiple implementations on different platforms, and also the nesting of service implementations, to virtual ensembles of resources. The following is an indicative list of implementation approaches:

1. A reference implementation constructed for full portability across multiple platforms may be used to support the execution environment (container) for hosting a service.
2. A mapping of the service interface definition to the native platform can be used over a platform possessing specialised native facilities for delivering service functionality.
3. A nested approach may be used so that a higher-level service is constructed by the composition of multiple lower-level services, which themselves may either map to native facilities or decompose further. The service implementation then dispatches operations to lower-level services.

Central to this virtualisation of resource behaviours is the ability to adapt to operating system functions on specific hosts. A significant challenge when developing these *mappings* is to enable exploitation of native capabilities so that the Grid environment does not become the least common denominator of its constituent pieces.

[136] Foster, I., Kesselman, C., Nick J M., and Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.

Service discovery mechanisms are important in this regard, allowing higher-level services to discover what capabilities are supported by a particular implementation of an interface. For example, if a native platform supports reservation capabilities, an implementation of a resource management interface can exploit those capabilities.

Finally, a service-oriented architecture for the Grid needs to support *local and remote transparency with respect to service location and invocation*. It also provides for *multiple protocol bindings* to facilitate localised optimisation of services invocation when the service is hosted locally with the service requestor, as well as to enable protocol negotiation for network flows across organisational boundaries where we may wish to choose between several interGrid protocols, each optimised for a different purpose. An implementation of a particular Grid service interface may map to native platform functions and capabilities.

6.3.3.2 An open Grid Service Architecture for the Next Generation GRID

The Global Grid Forum (GGF) has initiated the definition of the Next-Generation of Grid (NGG) middleware by extending the emerging web services technology that are currently being developed across the IT industry. One motivation is to bring Grid middleware development in-line with mainstream computing, another is to provide an open grid middleware platform with several, potentially competing, implementations.

This NGG is based on the concept of Open Grid Service Architecture (OGSA) that has been proposed as an enabling infrastructure for systems and applications that require the integration and management of services within distributed, heterogeneous, dynamic "virtual organizations" (see [137]). Either confined to a single enterprise or extending to encompass external resource sharing and service provider relationships, service integration and management in these contexts can be technically challenging because of the need to achieve various end-to-end qualities of service when running on top of different native platforms. Building on Web services and Grid technologies, OGSA proposes to define a core Grid service semantics and, on top of this, an integrated set of service definitions that address critical application and system management concerns. The purposes of this definition process are twofold: first to simplify the creation of secure, robust systems and second to enable the creation of interoperable, portable, and reusable components and systems via the standardization of key interfaces and behaviours.

In fact, OGSA is based on the industry standards for Web Services implemented using four main technologies: XML, WSDL, SOAP, UDDI (see section 6.2.2 for details).

Leveraging upon the Web Service technologies it allows to exploit a common base infrastructure that consists of tools for development, performance analysis, monitoring, and so forth. Furthermore Web Services are defined by open standards, so they provide high support for interoperability across multiple hosting environments based on different operating systems.

On the other side, thanks to the adoption of a service-oriented architecture that allows to build smaller and better-defined components, the reuse of single components and the creation of applications enhancing existing components, will be simpler.

OGSA inherits all these advantages and applies them to the Grid environment.

6.3.3.3 The Open Grid Service Infrastructure specifications

Despite the fact that with WSRF recently another alternative implementation close to Web Services has been published the discussion of OGSI in this document is still beneficial. First the WSRF specifications are still in a very draft form and neither implementations nor Grid systems based on WSRF are available so far. OGSI is implementing basic services of the Open Grid Services Architecture (OGSA). Although, OGSA is an overall view of the Grid Services architecture, including a vision of a complete platform and defining which Grid

[137] "Report on state of the art of ASP and Grid architectures" – Deliverable 4 – GRASP consortium

service are part of the architecture and how they relate to each other, the Open Grid Service Infrastructure (OGSI) is the foundation on which the Grid Services will be built.

It defines extensions to Web Services and mechanisms that support Grid Computing, such as service creation and lifetime management and a minimal set of interfaces and behaviours for information exchange, service discovery, management of the distributed and often long-lived state that is commonly required in advanced distributed applications.

The OGSI specification version 1.0 in its final version is available[138]. The specification v1.0 is now an official GGF "Proposed Recommendation," meaning it will not change from this point forward. It focuses on technical details, providing a full specification of the behaviours and Web Service Definition Language (WSDL) interfaces that define a Grid Service.

OGSI version 1.0 defines a component model that extends WSDL and XML Schema definition to incorporate the concepts of:

- Stateful Web services,
- Extension of Web services interfaces,
- Asynchronous notification of state change,
- References to instances of services,
- Collections of service instances, and
- Service state data that augments the constraint capabilities of XML Schema definition.

The OGSI specification describes the minimal set of extensions and interfaces necessary to support definition of the services that will build an OGSA compliant Grid system.

While we have a first “stable” OGSI specification, we don’t have a final specification of the OGSA platform that is currently in active development. It will consist of several documents, in particular, the OGSA platform document defines the high level architecture and it specifies the services that make up the OGSA platform and define the relationship between them. The OGSA architecture document is in draft.

A consequence of this current status of OGSA platform specification is that there is not an available implementation of OGSA. What we have instead, are implementations of OGSI, and several Grid Services that may or may not eventually become part of the OGSA platform.

In this frame projects such as GRASP developing, on top of existing implementation of OGSI specification, a set of services that can be put at the level of an OGSA platform provide an outlook on services needed for a full OGSA platform.

6.3.3.4 The purpose and scope of OGSI

The Open Grid Services Infrastructure (OGSI) is an interoperability framework realizing OGSI that has been recently proposed by the Global Grid Forum. OGSI integrates key Grid technologies with Web Services mechanisms to create a distributed system framework based around the notion of a *Grid service*. A *Grid service instance* is a (potentially transient) service that conforms to a set of conventions (expressed as WSDL interfaces, extensions, and behaviours) for supporting the basic functionalities that enable dynamically evolving VOs, such as lifetime management, discovery of characteristics, notification, etc. Grid services enable the controlled management of the distributed and often long-lived state that is commonly required in sophisticated distributed applications. OGSI also introduces standard interfaces (called “factory” and “registration”) for creating and discovering Grid services.

[138] “Open Grid Service Infrastructure” version 1.0, https://forge.gridforum.org/projects/ogsi-wg/document/Final_OGSI_Specification_V1.0/en/1

The explosion of Grid projects and applications worldwide has led to a diversity of approaches. Grid toolkits such as Globus, Avaki/Legion, and Unicore are widely used, but do not have universal acceptance and are also supplemented by an infilling of other specialised services. A Grid client is therefore constrained to use the conventions of the service it requires or furthermore when a client requires multiple services it may be faced with multiple conventions. The OGSI aims to provide a common framework where Grid services may be set up and published and where Grid clients may find, request and use them.

OGSI defines the interaction semantics of a Grid service instance: how it is created, how it is named, how its lifetime is determined, how to communicate with it, etc. However, while OGSI is prescriptive on matters of basic behaviour, it does not place requirements on what a service does or how it performs that service. In other words, OGSI does not address issues of implementation programming model, programming language, implementation tools, or execution environment.

In practice, the services defined following OGSI are instantiated within a specific execution environment or *hosting environment*. A particular hosting environment defines not only implementation programming models, programming languages, development tools, and debugging tools, but also how an implementation of a service meets its obligations with respect to the service semantics.

6.3.3.5 Grid Services requirements over and above Web Services

OGSI defines the concept of a Grid service: a Web service that provides a set of well-defined interfaces and that follows specific conventions. The interfaces address

- Discovery,
- Dynamic service creation,
- Lifetime management,
- Notification, and
- Manageability

Naming and upgradeability are addressed by the use of specific conventions for Grid services.

In the following paragraphs we highlight the essential extensions that Grid services bring about on traditional Web services. As OGSI evolves, authorisation and concurrency control for Grid services are also going to be addressed.

6.3.3.6 Standard Interfaces

The interfaces (in WSDL terms, portTypes) that define a Grid service are listed in the following table.

PortType	Operation	Description
Grid Service	FindServiceData	Query a variety of information about the Grid service instance, including basic introspection information (handle, reference, primary key, home handleMap: terms to be defined), richer per-interface information, and service-specific information (e.g., service instances known to a registry). Extensible support for various query languages.
	SetTerminationTime	Set (and get) termination time for Grid service instance
	Destroy	Terminate Grid service instance
Notification-Source	SubscribeTo-NotificationTopic	Subscribe to notifications of service-related events, based on message type and interest statement. Allows for delivery via third party messaging services.
Notification-Sink	DeliverNotification	Carry out asynchronous delivery of notification messages
Registry	RegisterService	Conduct soft-state registration of Grid service handles
	UnregisterService	Deregister a Grid service handle Factory CreateService Create new Grid service instance
Factory	CreateService	Create new Grid service instance
HandleMap	FindByHandle	Return Grid Service Reference currently associated with supplied Grid Service Handle

Table 7 PortTypes

Note: OGSI will define additional standard interfaces in the near future, to address issues such as authorisation, policy management, concurrency control, and the monitoring and management of potentially large sets of Grid service instances.

6.3.3.6.1 Transience

The interfaces and conventions that define a Grid service are concerned, in particular, with behaviours related to the management of *transient service instances*. The fundamental difference between Grid services and traditional persistent web services that handle complex activity requests from clients, is that Grid services focus on the on-demand instantiation of new transient service instances (which then handle the management and interactions associated with the state of particular requested activities). Transience has significant implications for how services are managed, named, discovered, and used.

Examples of situations where service instances may be extremely lightweight entities, created to manage short-lived activities include:

- The establishment of a videoconferencing session might involve the creation of service instances at intermediate points to manage end-to-end data flows according to QoS constraints.
- Service instances might be instantiated dynamically to provide for consistent user response time by managing application workload through dynamically added capacity.
- A query against a database.
- A data mining operation.
- A network bandwidth allocation.
- A data transfer.

- An advance reservation for processing capability.

6.3.3.6.2 Upgradeability

Versioning and compatibility between services must be managed and expressed so that clients can discover not only specific service versions but also compatible services.

Services (and the hosting environments in which they run) must be independently *upgradeable* without disrupting the operation of their clients. Upgrading a service must not impact other services

OGSI defines conventions that allow us to identify when a service changes and when those changes are backwardly compatible with respect to interface and semantics.

OGSI also defines mechanisms for refreshing a client's knowledge of a service, such as what operations it supports or what network protocols can be used to communicate with the service.

6.3.3.6.3 Discovery

Applications require mechanisms for discovering available services and for determining the characteristics of those services so that they can configure themselves and their requests to those services appropriately.

OGSI addresses this requirement by defining a standard representation for the following:

- *Service data*, which comprise information about Grid service instances, and are structured a set of named and typed XML elements called *service data elements*, encapsulated in a standard container format. The packaging of each element includes a name that is unique to the Grid service instance, a type, and time-to-live information that a recipient can use for lifetime management
- A standard operation, FindServiceData (within the obligatory *GridService* interface), for quering and retrieving service data from individual Grid service instances ("pull" mode access, compared with the "push" mode provided by the Notification service). The FindServiceData operation requires a simple "by name" query language, and is extensible to allow for the specification of the query language used.
- Standard interfaces for registering information about Grid service instances with registry services (*Registry*) and for mapping from "handles" to "references".

6.3.3.6.4 Dynamic service creation

The ability to dynamically create and manage new service instances is a basic tenet of the OGSI model and necessitates the existence of service creation services. The OGSI model defines a standard interface (*Factory*) and semantics that any service creation service must provide.

The CreateService operation of the *Factory* interface creates a requested Grid service and returns the GSH (Grid Service Handle) and initial GSR (Grid Service Reference) of the new service instance. The GSH is not a direct link to the service instance, but rather it is bound to a GSR. The GSR might be (the OGSI allows for other representations) the WSDL document for the service instance with the required "instanceOf" and other OGSI extensions. The idea is that the handle provides a constant way to locate the current GSR for the service instance, because the GSR may change if the service instance changes or is upgraded.

Note that the *Factory* interface does not specify how the service instance is created. One common scenario is for the factory interface to be implemented in some form of hosting environment (such as .NET or J2EE) that provides standard mechanisms for creating (and subsequently managing) new service instances. The hosting environment may define how services are implemented (e.g., language), but this is transparent to service requestors in OGSI, which see only the factory interface. Alternatively, one can construct "higher-level" factories that create services by delegating the request to other factory services.

6.3.3.6.5 Lifetime management

In a system that incorporates transient, stateful service instances, mechanisms must be provided for *reclaiming services and state associated with failed operations*. For example, termination of a videoconferencing session might also require the termination of services created at intermediate points to manage the flow.

OGSI addresses lifetime management by defining two standard operations: *Destroy* and *SetTerminationTime* (within the required *GridService* interface), for explicit destruction and soft state lifetime management of Grid service instances, respectively.

Grid service instances are created with a specified lifetime. The initial lifetime can be extended by a specified time period by explicit request of the client or another Grid service acting on the client's behalf (subject of course to the policy of the service). If that time period expires without having received a re-affirmation of interest from a client, either the hosting environment or the service instance itself is at liberty to terminate the service instance and release any associated resources.

6.3.3.6.6 Notification

A collection of dynamic, distributed services must be able to notify each other asynchronously of interesting changes to their state. OGSI defines common abstractions and service interfaces for subscription to (*NotificationSource*) and delivery of (*NotificationSink*) such *notifications*, so that services constructed by the composition of simpler services can deal with notifications (e.g., for errors) in standard ways.

The *NotificationSource* interface is integrated with service data, so that a notification request is expressed as a request for subsequent "push" mode delivery of service data. Note that this complements the FindServiceData operation of the *GridService* interface for retrieving service data from individual Grid service instances; this provides for "pull" mode access.

The OGSI notification framework allows clients to register interest in being notified of particular messages (the *NotificationSource* interface) and supports asynchronous, one-way delivery of such notifications (*NotificationSink*). If a particular service wishes to support subscription of notification messages, it must support the *NotificationSource* interface to manage the subscriptions. A service that wishes to receive notification messages must implement the *NotificationSink* interface, which is used to deliver notification messages. To start notification from a particular service, one invokes the subscribe operation on the notification source interface, giving it the service GSH of the notification sink. A stream of notification messages then flow from the source to the sink, while the sink sends periodic keepalive messages to notify the source that it is still interested in receiving notifications. If reliable delivery is desired, this behavior can be implemented by defining an appropriate protocol binding for this service.

6.3.3.7 The introduction of the WS-Resource Framework

Along with the OGSI work, the Web services architecture has evolved, with for example the definition of WSDL 2.0[139] progressing and the release of new draft specifications such as WS-Addressing[140]. OGSI 1.0 also combined into one-specification functions that are independently useful, for example event notification. This made the specification unnecessarily complex. Furthermore, OGSI compliant programming did not work well with existing Web services and XML tooling and OGSI was considered by a lot of people to be too object oriented instead of service oriented.

Therefore, in January 2004, the WS-Resource Framework was proposed as a refactoring and evolution of OGSI aimed at exploiting new Web services standards, specifically WS-

[139] W3C (5-May-2004). The Web Services Description Language, version 2.0 draft. See http://www.w3.org/2002/ws/desc/

[140] IBM (30-Mar-2004). Web Services Addressing - a new protocol for addressing messages in a distributed application environment. http://www-106.ibm.com/developerworks/library/specification/ws-add/

Addressing, and at evolving OGSI based on early implementation and application experiences[141]. The WS-Resource Framework retains essentially all of the functional capabilities present in OGSI, while changing some of the syntax (e.g., to exploit WS-Addressing) and also adopting a different terminology in its presentation. In addition, the WS-Resource Framework partitions OGSI functionality into five distinct, composable specifications, as depicted below.

Name	Description
WS-ResourceProperties	Describes associating stateful resources and Web services to produce WS-Resources, and how elements of publicly visible properties of a WSResource are, retrieved, changed, and deleted.
WS-ServiceGroup	Create and use heterogeneous by-reference collections of Web services.
WS-ResourceLifetime	Allows a requestor to create, manipulate and destroy a WS-Resource.
WS-BaseFault	Describes a base fault type used for reporting errors.
WS RenewableReferences	Annotate a WS-Addressing endpoint reference with information needed to retrieve a new endpoint reference when the current reference becomes invalid.
WS-Notification family of Specifications	Standard approaches to notification using a topic based publish and subscribe pattern.

Table 8 The Refactoring of OGSI yields five normative WS-Resource Framework specifications plus WS-Notification

At the core of the framework lies the **WS-Resource**, which is defined as a pairing of a Web Service and a resource properties document. The latter acts as a view on the actual *state* of a WS-Resource. The XML resource properties document serves to define the structure upon which service-requestor-initiated query and update SOAP messages can be directed. Thus, any operation that manipulates a resource property via the WS-Resource properties document must be reflected in the actual implementation of the WS-Resource's state to maintain consistency.

In other words, the **WS-ResourceProperties** specification defines the type and values of those components of a WS-Resource's state that can be viewed and modified by service requestors through a Web services interface. This specification knows the usual get and set operations. A *get* operation looks for example like this:

[141] Czajkowski, Karl et al. (5-Mar-2004). From Open Grid Services Infrastructure to WSResource Framework: Refactoring & Evolution. Version 1.1. http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf

```
<soap:Envelope>
        <soap:Header>
                <tns:resourceID> C </tns:resourceID>
        </soap:Header>
        <soap:Body>
                <wsrp:GetMultipleResourceProperty>
                        <wsrp:ResourceProperty>tns:p1</wsrp:ResourceProperty>
                </wsrp:GetMultipleResourceProperty>
        </soap:Body>
</soap:Envelope>
```

Figure 40 A sample SOAP message to retrieve a WS-ResourceProperty

So, as one would expect, the not modifiable resource identifier is carried in the SOAP header. This is the standard encoding for WS-Addressing endpoint reference ReferenceProperties. In this example above, the property p1 of the resource C is queried.

The *set* operations are namely, like in SQL, insert, update and delete. Here is the pseudo-syntax for it:

```
<wsrp:SetResourceProperties>
 {
        <wsrp:Insert >
        {any}*
        </wsrp:Insert> |
        <wsrp:Update >
        {any}*
        </wsrp:Update> |
        <wsrp:Delete ResourceProperty="QName" />
 }+
</wsrp:SetResourceProperties>
```

Moreover, the specification provides a fourth operation that allows the service requestor to execute an arbitrary query, e.g. an XPath expression.

The *identifier* of a WS-Resource is also a resource property, but one that shouldn't be interpreted or manipulated by the service requestor. Interestingly, although it is believed that portability of the identity is desired and that the format should look like a namespace-scoped value, no part of this framework or any other WS specification standardizes the identifier's format or the way how a requestor should obtain it (see [142], p.8).

The **WS-ServiceGroup** specification deals with bundling together multiple references to heterogeneous Web Services or WS-Resources and defines how these bundles shall be

[142] Czajkowski, Karl et al. (5-Mar-2004). The WS-Resource Framework. Whitepaper Version 1.0. http://www-106.ibm.com/developerworks/library/ws-resource/ws-wsrf.pdf

represented and managed[143]. This will allow e.g. to perform an operation on this grouped entity as a whole. The ServiceGroup specification is able to express ServiceGroup membership rules, membership constraints, and classifications using the resource property model from WS-ResourceProperties.

The relation between WS-Resource, WS-ResourceProperties and WS-ServiceGroup can be illustrated as depicted below:

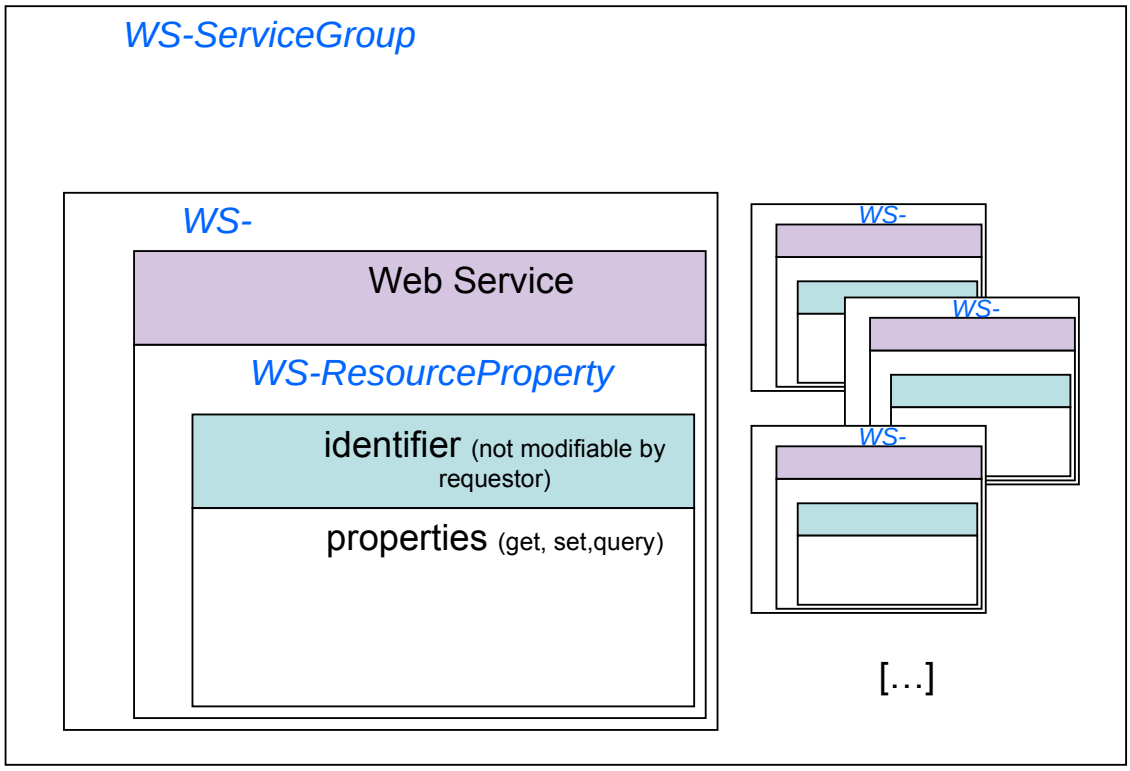


Figure 42 Overview of the correlation between WS-Resource, WS-ResourceProperties and WS-ServiceGroup

WS-ResourceLifetime addresses the management of a WS-Resource's lifetime, namely the creation, identity and the destruction. For the *creation* it uses the factory design pattern, like in OGSI. It introduces a *WS-Resource factory* which is defined by its capability to bring one or more WS-Resources to life. The idea behind that is to make the instantiation of a WS-Resource and its location more dynamic. The factory usually will return a *WS-Resource-qualified endpoint reference* directly to the requestor, although it is also possible to place it into a registry instead.

A ***WS-Resource-qualified endpoint reference*** is including a ReferenceProperties child element that identifies the stateful resource to be used in the execution of all message exchanges performed using this EndpointReference. This is a conventional use of WS-Addressing, referred to by the Globus/IBM people as an *implied resource pattern*. A request message directed to a Web service designated by a WS-Resource-qualified endpoint reference must include the ReferenceProperties information from the endpoint reference, as specified by WS-Addressing.

[143] Graham, Steve et al. (31-Mar-2004). Web Services Service Group – Specification. Version 1.0. http://www-106.ibm.com/developerworks/library/ws-resource/ws-servicegroup.pdf

For the *destruction* phase, the WS-ResourceLifetime specification offers two possibilities: either an immediate or a scheduled destruction.

The aim of **WS-BaseFault** is to bring consistency to the faults that are being returned by the operations within this framework. For this purpose a base fault type is being defined. As usual, an xml schema and rules round up the specification. A base fault has the following general syntax:

```
<BaseFault>
    <Timestamp>xsd:dateTime</Timestamp>
    <OriginatorReference>
        wsa:EndpointReferenceType
    </OriginatorReference> ?
    <ErrorCode dialect="anyURI">xsd:string</ErrorCode> ?
    <Description>xsd:string</Description> *
    <FaultCause>wsbf:BaseFault</FaultCause> *
</BaseFault>
```

Figure 43 The general format of a base fault

While the Timestamp element is mandatory, all the other elements are optional. OriginatorReference and ErrorCode can occur only once, whereas Description and FaultCause elements can be repeated as many times as needed within one base fault.

Each distinct type of fault associated with a WSDL operation must be listed as a separate fault response in the WSDL operation definition. Here is an example of how the specification defines the WSDL integration:

```
[...]
<!-- WSDL messages for each distinct fault -->
<wsdl:message name="hisFaultMessage">
        <wsdl:part name="fault" element="tns:hisFault"/>
</wsdl:message>
<wsdl:message name="herFaultMessage">
        <wsdl:part name="fault " element="tns:herFault"/>
</wsdl:message>
[...]
<wsdl:portType name="pt">
        <wsdl:operation name="op">
                <!-- WSDL operation fault elements for each distinct fault -->
                <wsdl:input … />
                <wsdl:output … />
                <wsdl:fault name="hisFault" message="tns:hisFaultMessage"/>
                <wsdl:fault name="herFault" message="tns:herFaultMessage"/>
                <wsdl:fault                                    name="BaseFault"
message="wsbf:BaseFaultMessage"/>
        </wsdl:operation>
</wsdl:portType>
[...]
```

Figure 44 Integration of WS-BaseFaults into WSDL 1.1

The **WS-RenewableReferences** specification has not been published, yet. The intention of this specification will be to describe how a WS-Addressing endpoint reference can be decorated with information on how a new version of an endpoint reference can be retrieved by a requestor when an endpoint reference has become invalid.

6.4 Semantic and Ontology Technology

6.4.1 Introduction

The Semantic Web[144] is an initiative of the World-Wide Web Consortium[145]. It has an ambitious long-term aim; nothing less than imbuing the Web itself with meaning. That is, providing meaning to the network of *resources* available on the web and, perhaps more importantly, meaning to the *links* that connect them[146]. Once the web has a mechanism for defining semantics for resources and links, then the possibility for *automatic processing* of the Web by software agents, rather than the constant mediation by people.

This ambitious aim has been there from the beginning[147] as Tim Berners-Lee's original description of the WWW included types for objects and links. However, it was not until the Semantic Web Roadmap[148] that the initiative became fully underway. Since 1998, and particularly more recently, there has been considerable activity in this area.

The Semantic Web has been developing a layered architecture:

- **Resource Description Framework**[149]: A basic knowledge representation language, describing a graph model and XML format for describing relationships between resources. The basic construct is a simple triple model, using URIs as identifiers for resources.
- **RDF Schema Language (RDF Schema)**[150]: A basic type modelling language for describing classes of resources and properties between them in the basic RDF model.
- **Ontologies**[151]: A richer type modelling language for providing more complex constraints on the types of resources and their properties.
- **Logic and Reasoning:** A (automatic) reasoning system provided on top of the ontology structure to *make new inferences.* Thus using such a system, a software agent can make deductions as to whether a particular resource satisfies it requirements (and vice versa).

Thus the Semantic Web initiative has an ambitious programme to bring existing work on knowledge representation and reasoning to bear on the Web and the work on all areas has been slow, with the process returning to its beginnings on several occasions. This has given the impression that the activity is of largely academic interest, whilst of course the application and potential of this work is enormous.

Nevertheless, recently there has been more rapid progress on the development of the Semantic web which has resulted in the following recommendations from the W3C released on 10th February 2004:

- RDF/XML Syntax Specification (Revised), Dave Beckett, ed.

---

[144] Berners-Lee, T., Hendler, J., and Lassila, 0: The Semantic Web. Scientific American, May 2001.

[145] The World Wide Web Consortium (W3C): http://www.w3.org

[146] M-R Koivunen, E. Miller (2001): W3C Semantic Web Activity, Proceedings of the Semantic Web Kick-off Seminar in Finland, http://www.w3.org/2001/12/semweb-fin/w3csw.

[147] T. Berners-Lee (1989): Information Management: A Proposal, *CERN* http://www.w3.org/History/1989/proposal.html.

[148] T. Berners-Lee (1998): Semantic Web Road Map http://www.w3.org/DesignIssues/Semantic.html.

[149] Resource Description Framework homepage http://www.w3.org/RDF/.

[150] O. Lassila R. Swick (1999): Resource Description Framework (RDF) Model and Syntax Specification *W3C Recommendation 22 February 1999* http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

[151] Web-Ontology (WebOnt) Working Group homepage http://www.w3.org/2001/sw/WebOnt/.

- RDF Vocabulary Description Language 1.0: RDF Schema, Dan Brickley, R.V. Guha, eds.
- RDF Primer, Frank Manola, Eric Miller, eds.
- Resource Description Framework (RDF): Concepts and Abstract Syntax, Graham Klyne, Jeremy Carroll, eds.
- RDF Semantics, Patrick Hayes, ed.
- RDF Test Cases, Jan Grant, Dave Beckett, eds.
- Web Ontology Language (OWL) Use Cases and Requirements, Jeff Heflin ed.
- OWL Web Ontology Language Reference, Mike Dean, Guus Schreiber eds., Frank van Harmelen Jim Hendler Ian Horrocks Deborah L. McGuinness Peter F. Patel-Schneider Lynn Andrea Stein
- OWL Web Ontology Language Semantics and Abstract Syntax, Peter F. Patel-Schneider, Patrick Hayes, Ian Horrocks eds.
- OWL Web Ontology Language Overview, Deborah L. McGuinness, Frank van Harmelen eds.
- OWL Web Ontology Language Test Cases, Jeremy Carroll, Jos De Roo eds.
- OWL Web Ontology Language Guide, Michael K. Smith, Deborah McGuinness, Raphael Volz, Chris Welty eds.

Progress on reasoning tools has been slower, with many proposals, varying from simple queries to modal logic theorem provers. This is still an active research and development area, from which we would anticipate that several approaches may emerge suitable for different purposes.

Applications of RDF have emerged, including, Dublin Core (Miller et al 1999[152]), RDF Site Summary[153], Composite Capability/Preferences Profiles[154], and proposals for the Protocol for Internet Content Selection[155,156] and Protocol for Privacy Preferences Project (McBride et. al. 2002[157]). These are applications that are ideal for the Semantic Web as they describe properties of web based resources. Nevertheless, each individually could be described using some domain specific method, and possibly in a more succinct manner. What has yet to be demonstrated is the benefits to be gained from expressing such applications within a single framework, one that allows semantics based interoperability; this is perhaps the key advantage of the approach. Web Services provides an ideal environment where the advantages of a joined up semantic approach could be demonstrated.

6.4.2 Semantic Web Advanced Development – Europe (SWAD-Europe)

EC Framework 5 project Semantic Web Advanced Development for Europe (SWAD-Europe[158]) aims to play a key role in the evolution of the Semantic Web, through education

[152] E. Miller, P Miller, D Brickley (1999): Guidance on expressing the Dublin Core within the Resource Description Framework (RDF) *Dublin Core Metadata Initiative Draft Proposal* http://www.ukoln.ac.uk/metadata/resources/dc/datamodel/WD-dc-rdf/.

[153] G. Beged-Dov et. al. (2000): RDF Site Summary (RSS) 1.0 http://purl.org/rss/1.0/spec.

[154] G. Klyne, F. Reynolds, C. Woodrow, H. Ohto,(2001): Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies *W3C Working Draft 15 March 2001* http://www.w3.org/TR/2001/WD-CCPP-struct-vocab-20010315/.

[155] D. Brickley and R.Swick.(2000): RDF Vocabulary Description Language 1.0: RDF Schema *W3C Working Draft* http://www.w3.org/TR/rdf-schema/.

[156] D. Brickley and R.V. Guha.(2002): PICS Ratings Vocabularies in XML/RDF *W3C Note* http://www.w3.org/TR/rdf-pics.

[157] B. McBride, R. Wenning, L. Cranor, (2002): An RDF Schema for P3P *W3C Note 25 January 2002* http://www.w3.org/TR/2002/NOTE-p3p-rdfschema-20020125.

[158] D Brickley, S Buswell, B Matthews, L Miller, D Reynolds, M Wilson (2002): SWAD-Europe: Semantic Web Advanced Development in Europe *Presented at the International Semantic Web Conference.*

and outreach to developers, organisations and content creators; through Open Source implementation and testing, and through pre-consensus technology development to drive and inform the creation of new Semantic Web standards.

The overarching aim of the project has been to provide, through all appropriate means, a body of answers to questions that have to date gone unanswered, and to foster grassroots communities within which such concerns are addressed. It is more important to offer clear answers to these questions than it is for us to write software or complex technical reports. The technical research and advanced development activities are a means to an end: facilitating wide-scale Semantic Web deployment. The project will therefore remain responsive to external developments (such as the appearance of unanticipated third-party work, software libraries etc.), refining the technical focus of the research to track the current state of the art, and to respond to the concerns of stakeholder communities.

6.4.2.1 Goals of SWAD-Europe

The SWAD-Europe project is designed to further the evolution of the Semantic Web through a combination of targeted research and community outreach, taking a set of use-case driven scenarios and illustrating how Semantic Web technologies relate to the practical needs of European consumers, business and content producers. Specific goals include:

- To implement scenario-led examples showing the integration of multiple Semantic Web technologies drawing practical use cases from industry, consumer, and developer perspectives.
- To develop a Semantic Web technology integration strategy that emphasises the utility of XML languages (such as SVG, HTML, MathML, XLink) as complementary rather than competing components of the Web.
- To ensure that the European developers, citizens and content creators are kept aware of Semantic Web technology for supporting universal accessibility, device independence and internationalisation.
- To ensure that European Community is kept aware of international best practice, and that best practice within Europe is recognised internationally.
- To undertake targeted research and development in support of these objectives, and in collaboration with the wider European developer community, W3C Member organisations, and related Open Source initiatives.

The project has several activities of immediate interest to TrustCoM including:

Semantic Web Services: Several aspects of the project address the need for convergence between the Semantic Web and Web Service perspectives. This is addressed at three levels: through demonstrating specific worked examples of Semantic Web Services such as those relating to Annotations and to Trust; through exploring options for technical convergence between RDF and Web Service specifications (e.g. W3C RDF and SOAP serialisation syntaxes); and through showing new functionality gained by the application of a Semantic Web approach to the creation, discovery and exploitation of Web Services. .

Trust: SWAD-Europe includes research work focussed on issues of trust management for the Semantic Web. Building on earlier RDF-based work at W3C/MIT and elsewhere, this will include an analysis of the *Capabilities* and *Proof Carrying Authentication* approaches to trust, to digitally signed Semantic Web content, and to the creation of '*Web of Trust*' applications. The related work on annotations provides a practical, consumer-oriented test-bed for exploring the deployment of this technology. This work has lead for example to a survey of security formats relevant to the (Semantic) Web http://www.ninebynine.org/SWAD-E/Security-formats.html, and work on case studies and models is continuing.

6.4.3 Semantic Web in Virtual Organisations

Security management must become autonomic and adaptation must occur automatically in real-time, rather than through human intervention. Furthermore, autonomic security management will have to be complemented by extensible and machine processable

standards for negotiating, validating and amending collaboration agreements, encoded by means of electronic contracts, which can be autonomically enacted by the platform.

Such extensible and machine processable standards require the development of common vocabularies and negotiation protocols, and the Semantic Web provides common machine-readable data onto the web, by allowing common vocabularies and conventions to be defined to describe web accessible resources precisely as required. That the Semantic Web of the W3C might provide an underlying framework to allow the deployment of a service architecture, as first suggested in [159]. The Semantic Web can support the deployment of a service based architecture which is augmented with trust policies and trust management.

6.4.3.1 Policy publication and enforcement

Service providers will publish policies for their use, detailing the obligations, privileges and expected levels of service, which a user should accept before using the service.

Some initial efforts in the use of Semantic Web representations for basic security applications (authentication, access control, data integrity, encryption) have begun to bear fruit. For example, Denker et al.[160] have integrated a set of ontologies (credentials, security mechanisms) and security extensions for Web Service profiles with the CMU Semantic Matchmaker. Kagal et al.[161] are also developing Rei, a Semantic Web based policy language. Furthermore, KAoS services and tools allow for the specification, management, conflict resolution, and enforcement of policies within the specific contexts established by complex organizational structures represented as domains[162] [163] [164] [165]. A comparison of KAoS, Rei, and more traditional policy approaches such as Ponder can be found in [166].

The KAoS policy ontology distinguishes between authorizations and (state or event triggered, conditional) obligations. Other policy constructs, including delegation and role-based authorisations, are built out of the basic primitives of domains and the basic policy types. “Action” is defined as the ontological class used to classify instances of intended or performed actions. Applicability of action instances relates to a policy (instance) through the association of the corresponding classes. The use of OWL enables reasoning about the controlled environment, policy relations and disclosure, policy conflict detection, and

---

[159] Dimitrakos, T., Matthews, B. and Bicarregui, J. Towards supporting security and trust management policies on the Web. ERCIM Workshop 'The Role of Trust in e-Business' in conjunction with IFIP I3E conference on October 3, 2001.

[160] Denker, G., Kagal, L., Finin, T., Paolucci, M. and Sycara, K. Security for DAML Web Services: Annotation and Matchmaking. In D. Fensel, K. Sycara, & J.Mylopoulos (Ed.), The Semantic Web—ISWC 2003. Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA, October 2003, LNCS 2870.

[161] Kagal, K., Finin, T. and Anupam, J. A Logical Policy Language for a Pervasive Computing Environment., 4th IEEE Int. Workshop on Policies for Distributed Systems and Networks, Lake Como, 4-6 June, 2003.

[162] Bradshaw, J., Uszok, A., Jeffers, R., Suri, N., Hayes, P., Burstein, M., Acquisti, A., Benyo, B., Breedy, M., Carvalho, M., Diller, D., Johnson, M., Kulkarni, S., Lott, J., Sierhuis, M. and Van Hoof, R. Representation and reasoning about DAML-based policy and domain services in KAoS. In Proc. of The 2nd Int. Joint Conf. on Autonomous Agents and Multi Agent Systems (AAMAS2003).

[163] Johnson, M., Chang, P., Jeffers, R., Bradshaw, J. M., Soo, V.-W., Breedy, M. R., Bunch, L., Kulkarni, S., Lott, J., Suri, N., & Uszok, A. KAoS semantic policy and domain services: An application of DAML to Web services-based grid architectures. Proceedings of the AAMAS 03 Workshop on Web Services and Agent-Based Engineering. Melbourne, Australia, 2003.

[164] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S. and Lott, J. (2003). KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Deconfliction and Enforcement. In Proc. of the IEEE Workshop on Policy 2003.

[165] Uszok, A., Bradshaw, J., Jeffers, R., Johnson, M., Tate, A., Dalton, J. and Aitken, S. Policy and Contract Management for Semantic Web Services. to appear AAAI Spring Symposium, Stanford University, California, USA, March 2004.

[166] Tonti, G., Bradshaw, J. M., Jeffers, R., Montanari, R., Suri, N., & Uszok, A. (2003). Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In D. Fensel, K. Sycara, & J.Mylopoulos (Eds.), The Semantic Web—ISWC 2003. Proc. of the 2nd Int. Semantic Web Conf., Sanibel Island, Florida, USA, October 2003, LNCS 2870.

harmonization, as well as about domain structure and concepts exploiting the description logic subsumption and instance classification algorithms. Taking advantage of OWL, platform/application-specific ontologies are easily loaded on top of the core policy classes.

KAoS provides a powerful tool-set that appears to be capable to address publication and deployment of complex policies for Semantic Web Services. However the incorporation of trust metrics and a distributed enforcement and performance assessment schemes remain the main challenges, in addition to the production of a critical mass of domain/application-specific ontologies to allow its uptake and validation in large scale systems. With respect to the latter there is an ongoing effort to adapt KAoS for use in Grid Computing environments in conjunction to OGSA[167].

6.4.3.2 Service Discovery

In order for a new service to be used it needs to be discovered and a mapping needs to be established between the requirements of the client and the capabilities of the service. On the service side, discovery is facilitate in the presence of a set of semantic descriptions. WSDL descriptions can be used to support this, but they fall short in providing any unambiguous semantic content for the service interface description they provide. Thus there has been approaches to describing the functionality of web services using OWL (***Semantic Web Services***) where in addition to publishing their interfaces, Web Services publish statements describing their intended or normative behaviour. These statements should be given common, machine processable, extensible semantics that support judgment of:

- Whether a service can perform a given task;
- The relative ranking of a set of services with respect to basic QoS criteria.
- And to then using reasoning to match service descriptions against requirements

See for example the DAML-S initiative[168]; http://www.swsi.org/ provides a common point of information on this growing body of work. This approach has yet to fully describe quality of service criteria.

On the client side, the client objectives must also be given semantics in order to enable achieving a "sufficiently good" similarity between objectives of requestor and the capabilities of the service, advertised by its provider. Generally, a match can be determined by heuristic algorithms, aided by domain-specific ontologies that define the terms used for service description as well as the objectives of the requestor. Again, there is a need to extend this work to non-functional requirements. P3P[169] [170] adds policies and requirement of the client with respect to Privacy; this would need to be extended to express the wider quality-of-service expectations of the client.

6.4.3.3 Service negotiation

Once a service has been selected, there needs to be a negotiation between service and user to establish a relationship. As part of this process, the policies of both parties have to be interrogated and a contract of use established, and a conversation needs to take place between the parties, establishing a mutually intelligible vocabulary of terms for data and process descriptions. This negotiation may involve third parties (brokers, guarantors, service framework providers etc), which may facilitate the relationship and foster trust between the parties.

---

[167] Johnson, M., Chang, P., Jeffers, R., Bradshaw, J. M., Soo, V.-W., Breedy, M. R., Bunch, L., Kulkarni, S., Lott, J., Suri, N., & Uszok, A. KAoS semantic policy and domain services: An application of DAML to Web services-based grid architectures. Proceedings of the AAMAS 03 Workshop on Web Services and Agent-Based Engineering. Melbourne, Australia, 2003.

[168] DAML Services homepage: http://www.daml.org/services/.

[169] McBride, B., Wenning, R. and Cranor, L.: An RDF Schema for P3P. W3C Note 25 January 2002: http://www.w3.org/TR/p3p-rdfschema.

[170] Platform for Privacy Preferences (P3P) Project: http://www.w3.org/P3P/.

In this process, there is a step of trust evaluation, either from previous experience of one another, as recorded in a "trustbase" of trust valuations, or an evaluation of the trust value from recommendations from third parties, or a calculation of trust across the network via intermediate trust valuations. Preliminary work in calculating trust values across trust networks in the semantic web have been studied by Goldbeck, Hendler and Parsia[171,172] and Richardson, Agrawal, and Domingos[173] which use a relatively straightforward model of trust which does not take into account context or uncertainty. They also consider reputation management, although this could be handled via an existing W3C recommendation, the Platform for Internet Content Selection (PICS[174]). This standard, designed originally for content filtering of web pages, can be used to express a general rating scheme, including for standards of reliability of web entities.

Once a trust valuation of the parties involved has been established an agreement needs to negotiate between the parties. This requires the interchange of vocabulary, and again the Ontology support provided via OWL in the Semantic Web is able to provide this mechanism; indeed as already noted, DAML-S has already started this for service descriptions, and KAoS for policies. We need to embed the trust valuations into the process, to use the expression of user requirements and preferences.

6.4.3.4 Experience monitoring and policy enforcement

During the execution of the service, which may be over a long period, its progress is monitored. The experience of the quality of the service may modify the relationship between the parties. For example, if the experience so far is good, then the parties may relax restrictions for the remainder of the service.

Policy statements need to be interpreted into lower-level rules which are then enforced at each network end-point. Web Services standards for SOAP-based message security and XML-based languages for access control (e.g. XACML; see section 8.4) are emerging. The use of XML as a basis for expression specification has the advantage of extensibility. Its semantics however are mostly implicit as meaning depends on a shared understanding derived from human consensus, and allow incompatible representation variations. Semantic Web-based policy representations could be mapped to lower level XML representations if required by an implementation.

Once an agreement has been established, then the client can start using the service. This usage may be long-lived, and the experience of the parties during the interaction may modify their behaviour for its remainder. For example, good experience may result in the loosening of restrictions and a higher-level of trust, changing the valuations in internal "trustbases", and reducing the policy enforcement overhead.

End-point enforcement of a web service in particular requires an agent:

- To interpret the data elements and procedure calls of messages, compare them with the rules in a policy statement and block unauthorized requests;
- To interpret outgoing messages in order to ensure that the service does not initiate communication with malicious agents or send unauthorized requests;
- To initiate an action or a specific request to another service in order to meet an obligation associated with the enforcement of a policy statement.

---

[171] Golbeck J., Parsia B., and Hendler J.: Trust networks on the semantic web. In Proceedings of Cooperative Intelligent Agents 2003, Helsinki, Finland, August 2003.

[172] Golbeck, J. and Hendler, J. Inferring Reputation on the Semantic Web. http://www.mindswap.org/papers/GolbeckWWW04.pdf, submitted to WWW'04.

[173] Richardson, M., Agrawal, R. and Domingos, P. Trust Management and the Semantic Web In D. Fensel, K. Sycara, & J.Mylopoulos (Eds.), The Semantic Web—ISWC 2003. Proc. of the 2nd Int. Semantic Web Conf., Sanibel Island, Florida, USA, October 2003, LNCS 2870.

[174] Platform for Internet Content Selection (PICS). http://www.w3.org/PICS/.

These processes are likely to take place largely internally to the services taking part, but nevertheless a common vocabulary for interpreting the action of the other and third parties is needed.

6.4.3.5 Service review

After the interaction has been completed, there should be a stage of review when trust valuations are reassessed, policies modified in the light of experience, and any recommendations to third parties propagated, within the common vocabularies provided within the Semantic Web.

6.4.4 Application to TrustCoM

XML will be the underlying language used in various parts of the TrustCoM framework: contracts, (security) policies, VOs, etc, will be described in standardized XML schemas. However, while XML provides a common syntactic framework it doesn't guarantee that the information conforming to different schemas is easily combined. In some cases, the schemas use common elements which improve interworking between them. Semantic technologies and ontologies may allow the integration of different XML-based descriptions. They may also enable software components to effectively reason about the static XML descriptions, and make autonomous (security) decisions, as needed in dynamic Virtual Organisations.

Clearly, trust management, contract management and autonomic security mechanisms are important aspects in the practical deployment of the service architecture across different organisations, especially involving governmental and commercial organisations. Failure to provide adequate technical, legal and economic mechanisms to allow participants to act with confidence, will slow the acceptance the service architecture as an enabler of collaborations, and may prevent its uptake altogether, users instead using closed proprietary solutions which lack the benefits of an open system.

Semantic Web Services (e.g. complement the rapidly advancing Web Services technology by defining and implementing new capabilities which more fully harness the power of Web Services through explicit representations of the semantics underlying Web resources. They provide an infrastructure capable of fully exploiting these semantics. Semantic Web Languages such as OWL extend RDF to allow users to specify ontologies composed of taxonomies of classes and inference rules.

This approach is expected to allow software agents to understand and autonomously manipulate other agents or services, therefore enabling discovery, meaningful communication and collaboration among software agents and services, relying on control mechanisms that implement policy statements capturing human imposed constraints.

Thus the Semantic Web offers the infrastructure to share the vocabulary and semantic of policies and trust valuations. This has the advantage of using an established body of languages and tools designed to function over open distributed systems to leverage this sharing in an effective and economic manner. There has been some preliminary work in providing piece of this architecture carried out in different places. However, there has been no coherent scheme to bring these together in one policy-based service architecture.

6.5 Tools and platforms

6.5.1 Web and Grid Services middleware

6.5.1.1 Microsoft .NET

Microsoft® .NET is a set of Microsoft software technologies for connecting information, people, systems, and devices. It enables a high level of software integration through the use of Web services—small, discrete, building-block applications that connect to each other as well as to other, larger applications over the Internet. The Microsoft .NET Framework is a platform for building, deploying, and running Web Services and applications. It provides a highly productive, standards-based, multi-language environment for integrating existing investments with next-generation applications and services as well as the agility to solve the challenges of deployment and operation of Internet-scale applications. The .NET Framework consists of three main parts: the common language runtime, a hierarchical set of unified class libraries, and a componentized version of Active Server Pages called ASP.NET.

Microsoft .NET is based on the 'Common Language Runtime' (CLR) and a Base Framework. The CLR and the framework form the 'Common Language Infrastructure' (CLI), in which applications written in multiple high-level languages may be executed in different system environments without the need to rewrite the applications to take into consideration the unique characteristics of those environments.

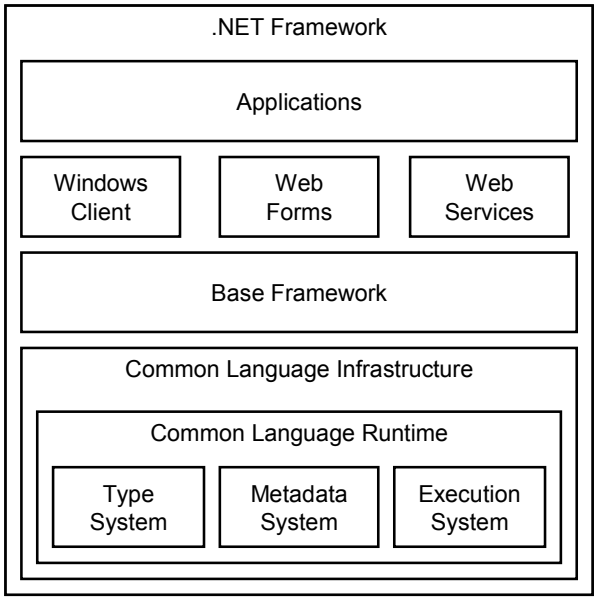


Figure 45 .NET Framework

The CLR provides a runtime environment for the execution, security, and management of .NET applications and components. The CLR contains a common type system (CTS) that defines common data types that can be used by .NET programming languages, garbage collection for automated memory allocation and de-allocation of memory (thereby reducing the number of developer-induced memory leaks), rich metadata that supports deployment without the need for Windows Registry accesses, and so called strong named assemblies and a Global Assembly Cache (GAC) which eases the versioning of system libraries ('DLL Hell'). .NET applications and components are now compiled into an intermediate 'MSIL' code (MSIL – Microsoft Intermediate Language) that is then deployed into users' workstations or servers. On execution, the MSIL code is converted into native platform executables by the JIT compiler.

The CLR provides a number of services, including the following:

- Code management (loading and execution)
- Application memory isolation
- Verification of type safety
- Conversion of IL to native code
- Access to metadata (enhanced type information)
- Managing memory for managed objects
- Enforcement of code access security
- Exception handling, including cross-language exceptions
- Interoperation between managed code, COM objects, and pre-existing DLLs (unmanaged code and data)
- Automation of object layout
- Support for developer services (profiling, debugging, and so on)

The .NET Framework supports multiple programming languages, including

- C# (C Sharp), a modern and innovative programming language. Introduced in 2001, C# offers a familiar syntax, which is attractive to C++ and Java developers, along with unique language constructs that offer *code-focused* developers a more elegant experience when developing applications for the .NET Framework.
- Visual Basic .NET (VB.NET) which is a major evolution from the Visual Basic language and provides full–blown, object-oriented features to the platform;
- Visual J#, the Java-language tool for Microsoft .NET. Visual J# .NET gives *Java-language and existing Visual J++* developers complete access to the .NET Framework, while maintaining a familiar language and syntax.
- Managed C++, which includes extensions to develop 'managed code' in C++ and allows C++ developers to utilize the new CLR-based programming model.

Besides the above-mentioned languages, which are directly supported by Microsoft, 3rd party vendors have created .NET implementations of Objective-C, FORTRAN, COBOL or Perl.

Besides the above-mentioned languages, which are directly supported by Microsoft, 3rd party vendors have created .NET implementations of Objective-C, FORTRAN, COBOL or Perl.

The compilers for these languages compile the code into MSIL code. MSIL is the CPU-independent instruction set into which .NET Framework programs are compiled. It contains instructions for loading, storing, initializing, and calling methods on objects. Combined with metadata and the common type system, MSIL allows for true cross-language integration. Prior to execution, MSIL is converted to machine code. It is not interpreted.

Both C# and the CLI have been standardized by ECMA (http://www.ecma-international.org/) and ISO/IEC (http://www.iso.org/). In April 2003, ISO ratified the standards as ISO/IEC 23270 (C#), ISO/IEC 23271 (CLI) and ISO/IEC 23272 (CLI TR). For more details, look at the Microsoft specification page[175]. More information on the Microsoft .NET Platform can be found at the Microsoft web site[176].

6.5.1.1.1 Application to TrustCoM

[175] ECMA and ISO/IEC C# and Common Language Infrastructure Standards, http://msdn.microsoft.com/net/ecma/

[176] Microsoft .NET Information http://www.microsoft.com/net/

The Microsoft .NET defines an execution environment with excellent support for the required web service and security specifications. Microsoft .NET is the environment in which future developments for the Microsoft Windows platform will be supported, so in order to have TrustCoM components being natively supported under Windows, usage of .NET is an important path to reach broad support for the TrustCoM framework.

6.5.1.2 Microsoft .NET Web Services Enhancements (WSE)

Web Services Enhancements for Microsoft® .NET (WSE)[177] is an add-on to Microsoft Visual Studio .NET and the Microsoft .NET Framework providing developers the latest advanced Web services capabilities to keep pace with the evolving Web services protocol specifications. WSE is available in two versions:

- Web Services Enhancements (WSE) 1.0 SP1 from March 2003 and
- Web Services Enhancements (WSE) 2.0 Technology Preview[178] from July 2003. *In this document, the term "WSE" refers to the WSE 2.0 Technology Preview.*

WSE's message-oriented programming model enables asynchronous communication for Web services implementations that require support for long lived operations, batch processing, peer to peer programs, or event driven application models. Web services that leverage WSE can now be hosted in multiple environments including ASP.NET, standalone executables, NT Services, etc. and can communicate over alternative transports including HTTP (http://.../) and TCP (soap.tcp://…/). So WSE 2.0 may be hosted without the ASP.NET runtime and IIS.

WSE 2.0 supports the following WS-* specifications:

- WS-Security,
- WS-Policy,
- WS-SecurityPolicy,
- WS-Trust,
- WS-SecureConversation,
- WS-Addressing.
- WS-Referral
- WS-Routing
- WS-Attachments with Direct Internet Message Encapsulation (DIME)

WSE 2.0 supports the following new capabilities:

- Token-issuing framework (WS-Trust, WS-SecureConversation) provides capabilities that build on WS-Security and define extensions to request and issue security tokens and to manage trust relationships and secure conversations.
- Roles-based authorization with integration into Windows security enables corporations to leverage their existing Windows domain credentials when accessing Web services or to integrate their own access control engine.
- Declarative programming model (WS-Policy, WS-SecurityPolicy) enables developers to author policies that operate a runtime component, responsible for processing the SOAP headers in Web services that contain security and routing information and play a role in the validation of incoming and outgoing messages. For example, the runtime can

[177] Microsoft Web Service Enhancements for .NET http://msdn.microsoft.com/webservices/building/wse/default.aspx

[178] Web Services Enhancements (WSE) 2.0 Technology Preview: http://www.microsoft.com/downloads/details.aspx?FamilyId=21FB9B9A-C5F6-4C95-87B7-FC7AB49B3EDD&displaylang=en

automatically sign and encrypt a message based on the authored policy without the developer having to write code.

- Message-based object model (WS-Addressing) provides customers with a message-based programming model over TCP and HTTP, allowing them to explore alternative types of SOAP-based applications such as ad hoc peer-to-peer applications.

6.5.1.2.1 Application to TrustCoM

WSE is the .NET implementation of many of the security-related WS-* specifications. Therefore, WSE will be a cornerstone of TrustCoM support on the Microsoft platform.

6.5.1.3 SUN J2EE

J2EE and Web Services

Web Services are application components that are designed to support interoperable machine-to-machine interaction over a network. This interoperability is gained through a set of XML-based open standards, such as the Web Services Description Language (WSDL), the Simple Object Access Protocol (SOAP), and Universal Description, Discovery, and Integration (UDDI). These standards provide a common and interoperable approach for defining, publishing, and using web services.

The Java 2 Platform, Enterprise Edition (J2EE) version 1.4[179] has evolved to integrate web services. Web services are now one of the many service delivery channels of the J2EE platform; existing J2EE components can be easily exposed as web services. Many benefits of the J2EE platform are available for web services, including portability, scalability, reliability, and no single-vendor lock-in. For example, J2EE containers provide transaction support, database connections, life cycle management, and other services that are scalable and require no code from application developers.

From a software architect's point of view, a web service can be considered as a service-oriented architecture, which consists of a collection of services that communicate with each other (and end-user clients) through well-defined interfaces. One advantage of service-oriented architecture is that it allows the development of loosely coupled applications that can be distributed and accessed, from any client, across the network.

The J2EE 1.4 SDK provides the tools needed to quickly build, test, and deploy web services and clients that interoperate with other web services and clients running on Java technology-based or non-Java technology-based platforms. In addition, it enables businesses to expose their existing J2EE applications as web services. Servlets and Enterprise JavaBeans (EJBs) components can be exposed as web services that can be accessed by Java technology-based or non-Java technology-based web service clients. J2EE applications can act as web service clients themselves, and they can communicate with other web services, regardless of how they are implemented.

The process of developing and deploying web services is coupled with the runtime system. For example, deploying a web service on Apache Axis is different from deploying the same web service on Apache SOAP or any other platform. The Java Community Process (JCP) specification JSR 109 (Implementing Enterprise Web Services) promotes building portable and interoperable web services in the J2EE 1.4 environment. JSR 109 leverages J2EE technologies to provide an industry standard for developing and deploying web services on the J2EE platform, and it provides a service architecture that is familiar to J2EE developers. This specification outlines the lifecycle of web services to include:

- **Development**: Standardizes the web services programming model as well as the deployment descriptors
- **Deployment**: Describes the deployment actions expected of a J2EE 1.4 container

[179] J2EE 1.4 SDK: http://java.sun.com/j2ee/1.4/download-sdk.html

- **Service publication**: Specifies how the WSDL is made available to clients
- **Service consumption**: Standardizes the client deployment descriptors and a JNDI lookup model

JAX-RPC is a Java API for XML-based Remote Procedure Calls (RPC)[180]. You can use it to build web services and clients that use RPC and XML. An RPC is represented using an XML-based protocol such as SOAP, which defines an envelope structure, encoding rules, and convention for representing RPC calls and responses, which are transmitted as SOAP messages over HTTP. The advantage of JAX-RPC is that it hides the complexity of SOAP messages from the developer.

The J2EE 1.4 platform provides comprehensive support for web services through the JAX-RPC 1.1 API, which can be used to develop service endpoints based on SOAP. JAX-RPC 1.1 provides interoperability with web services based on the Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP). The J2EE 1.4 platform also supports JSR 109[181], that builds upon JAX-RPC and focuses on the programming model for implementing web services, as well as deploying web services in the J2EE 1.4 platform. In addition, J2EE 1.4 supports the **WS-I Basic Profile** to ensure that web services developed using the J2EE platform are portable not only across J2EE implementations, but are also interoperable with any web service developed, using any platform that conforms to the WS-I standards.

Figure 46 shows how the Java APIs for XML Registries (JAXR) and Java APIs for XML Remote Procedure Calls (JAX-RPC) play a role in publishing, discovering, and using web services.

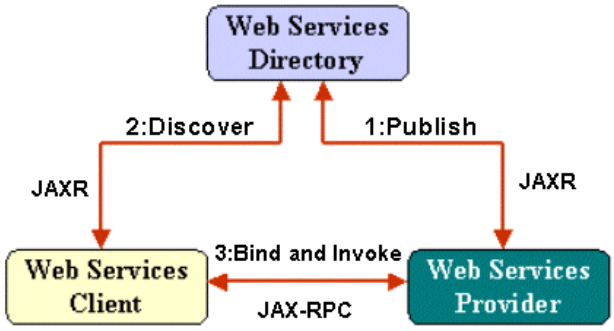


Figure 46 J2EE 1.4 Publish-Discover-Invoke model

The J2EE 1.4 SDK includes the following:

- J2EE 1.4 Application Server
- Java 2 Platform, Standard Edition (J2SE) 1.4.2_01
- J2EE Samples (Java Pet Store, Java Adventure Builder, Smart Ticket, and others)
- Sun ONE Message Queue
- PointBase Database Server

Note that J2EE web services can be invoked by any web service client, and any J2EE web service client can invoke any web service. Figure 47 shows how a Java client communicates with a Java web service in the J2EE 1.4 platform. Note that J2EE applications can use web services published by other providers, regardless of how they are implemented. In the case of non-Java technology-based clients and services, the figure would change slightly.

[180] XML-RPC Apache Implementation: http://ws.apache.org/xmlrpc/

[181] JSR 109: Implementing Enterprise Web Services: http://jcp.org/en/jsr/detail?id=109

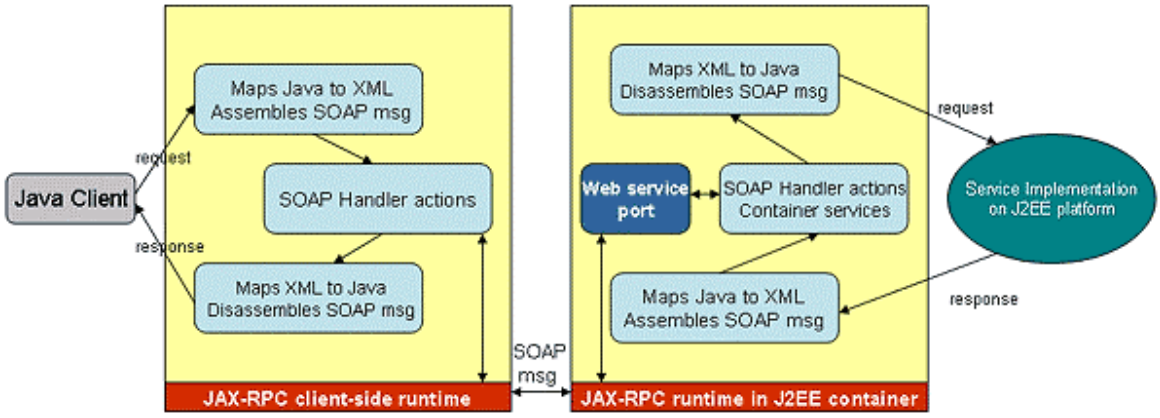


Figure 47 A Java client calling a J2EE web service

Information about developing Web Services with J2EE 1.4 Platform is detailed in [182].

More information on the J2EE Platform can be found at the Java Sun web site[183].

6.5.1.3.1 J2EE and Grid

Grid offers the capability to assign processing power, storage, and server capacity on an as-needed basis independent of the underlying technical infrastructure. A 'grid' of loosely coupled individual hardware/server units connected via the "grid fabric". The fabric is essentially a minimal infrastructure software layer that relies on the intranet/internet for connecting the various units of the grid.

Unlike SETI or Globus type grids, where the key challenge is to define generic work-units and distribute them across the nodes of the grid, J2EE (and Application Servers in general) have the advantage of well-defined work-units and requests. The 2.1 version of Jxta, included in Java 2 Platform, Standard Edition (J2SE), adds metering and monitoring, enabling the inspection of traffic and memory usage on a network peer, and querying of remote peers for this information.

Web and Grid services could be defined with Web Services Description Language (WSDL), and J2EE provides an appropriate containers environment to implement these services.

6.5.1.3.2 Application to TrustCoM

Java provides a runtime for applications on both Microsoft Windows and UNIX platforms. Many of the already existing tools in the TrustCoM consortium are based on the Java2 platform.

6.5.1.4 IBM WebSphere

Business Integration

Business Integration lets you realize the benefits of end-to-end integration through five core capabilities: model, integrate, connect, monitor and manage. WebSphere integration provides the best of breed connectors, adapters and interchange engines for businesses to connect their crucial value-chains and internal applications.

Foundation & Tools

[182] Developing Web Services with Java 2 Platform, Enterprise Edition (J2EE) 1.4 Platform: http://java.sun.com/developer/technicalArticles/J2EE/j2ee_ws.

[183] Java 2 Platform, Enterprise Edition (J2EE). http://java.sun.com/j2ee/

Open Services Infrastructure: WebSphere Application Server lets you deploy a core operating environment for a reliable foundation capable of high volume, secure transactions and Web services. WebSphere Studio lets you deliver a rapid and efficient response to business needs through new e-business applications. Enterprise Transformation, which includes WebSphere Host Integration and WebSphere Studio, lets you leverage existing business assets and skills to satisfy new e-business requirements.

Business Portals

Interactive User Experience: WebSphere Portal helps people interact in a personalized way with diverse business resources.

Access On Demand: WebSphere Everyplace and WebSphere Voice let you easily access information and take action anywhere, anytime, using any choice of devices. Selling and Channel Management through WebSphere Commerce helps you optimize marketing, business relationships and channel management to maximize e-commerce revenue.

Rational Tools

Rational software helps organizations create business value by improving their software development capability. Rational software powers the IBM Software Development Platform, a complete and modular solution that encourages teams to: Adopt iterative development practices that reduce project risk. Focus on architecture to develop more resilient systems. Continuously ensure quality across the software development lifecycle. Effectively manage change and protect critical strategic assets. The result: a more proficient software development team, and a more responsive, resilient, and focused business. Regardless of platform or application type — Eclipse, Java, .NET or embedded and pervasive applications — Rational takes you from start to finish, ensuring the complete success of your project.

6.5.1.5 The Apache Software Foundation Web Services Project

This is a brief note on the ASF project on web-service related technologies as of April 2004. After a brief overview of the web-service related technologies particular focus is given to the Apache Axis toolkit.

Glossary

Term	Description
SAAJ	SOAP with Attachments API for Java
JAX-RPC	Java API for XML-based RPC
WSDD	Web Service Deployment Descriptor. Axis web-service configuration file.
EJB	Enterprise Java Bean.

The technologies within ASF for web services include the following:

Technology	Description
Axis	A Java-based SOAP engine
WS-FX[184]	A family of sub-projects dedicated to implementing WS- security standards. Currently includes the following sub-projects: WS-Addressing WS-ReliableMessaging WS-Security

[184] WS-FX Home Page, http://ws.apache.org/ws-fx/

JaxMe[185]	An implementation of the JAXB Java-XML binding.
jUDDI[186]	A Java implementation of UDDI 2.0. Currently in incubation.
SOAP	Pre-cursor SOAP toolkit to Apache Axis.
WSIF	Web Service Invocation Framework. A means of accessing web services based on the WSDL description. Enables a client to access web services using arbitrary protocols, eg, SMTP, Java-based protocols or a local Java class.
WSIL4J[187]	Web Services Inspection Language. Implements WS-Inspection, which specifies how to inspect a web-site for installed web services. Provides an API for locating and inspecting WS-Inspection documents.
WSRP4J[188]	A Java implementation of the Web Services for Remote Portlets OASIS specification. WSRP intended to standardize interactive presentation-oriented web services. Currently in incubation.
XML-RPC[189]	Implementation of XML-RPC specification.
WSS4J	Web Services Security for Java. Implementation of WS-Security for Axis.

Apache Axis

Apache Axis is a Java-based web service platform[190]. Current stable version of Apache Axis is 1.1. Provides support for W3C SOAP 1.1 and SOAP 1.2 Candidate specifications. Axis 1.1 supports WSDL v1.1. Based on Java technologies SAAJv1.1 and JAX-RPCv1.0. WS-I Basic Profile supported in the next release, Axis 1.2. Beta version of 1.2 has just been released.

Axis includes a basic web-service engine with some basic development tools. The engine is a J2EE compliant servlet application. The engine can therefore run within any J2EE compliant servlet container- most popular choice at present is Tomcat. A set of tools can be used for generating server skeletal and client-stub code fragments from the WSDL file. Conversely, a WSDL file can be generated from an initial interface description written in Java.

Stateless-session EJB's can also be exposed as web-services using this toolkit using the WSDD file, described below.

Web-service handlers can be developed and deployed to process SOAP messages. Basic handlers provided with Axis include user-authentication and message logging handlers. The former of these is not WS-Security compliant. Other individuals and projects are implementing WS-Security for Axis[191]. The status of these projects is not known at this current time.

Customized Serializer and De-serializer classes can be developed by the user for converting Java types to XML and for de-coding XML fragments into the user's own Java classes. Support is given for serializing Java collections (e.g. hash tables), but this is not guaranteed

[185] JaxMe Home Page, http://ws.apache.org/jaxme/

[186] jUDDI Home Page, http://ws.apache.org/juddi/

[187] Web Service Inspection for Java, http://cvs.apache.org/viewcvs/*checkout*/ws-wsil/java/README.htm

[188] Web Services for Remote Portals Specification, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp

[189] XML-RPC Apache Implementation, http://ws.apache.org/xmlrpc/

[190] Apache Axis Home Page, http://ws.apache.org/axis/

[191] WSS4J Home Page, http://ws.apache.org/ws-fx/wss4j/

to be interoperable with other SOAP implementations. The recommended way to send collections is as basic array types.

Supported messaging modes include:

- RPC- Remote Procedure Call: basic RPC messaging using SOAP encoding of basic data-types- the default
- Document/Wrapped: where the message can contain non-SOAP data types defined by the user. Axis provides two modes for accessing the content: either directly via 'document' mode where data types are mapped to JavaBeans or via 'wrapped' modes where the contents are arguments to a web service method based on the name of the data type.
- Message: where the user has direct access to the XML elements within the message.

6.5.1.6 Existing OGSI based middleware

OGSI specifies a minimal, integrated set of extensions and interfaces necessary to support definition of the services that will compose OGSA. Actually, it doesn't define precise services but establishes a nucleus of behaviour common to all Grid services and that has been observed to be compliant with OGSA.

In particular, the OGSI specifications propose detailed specifications for the conventions that govern how clients create, discover, and interact with a Grid service instance. That is, they specify:

- How Grid service instances are named and referenced;
- The base, common interfaces (and associated behaviours) that all Grid services implement;
- The additional (optional) interfaces and behaviours associated with factories and service groups.

The Specifications do *not* address how Grid services are created, managed, and destroyed within any particular hosting environment. Thus, services that conform to this specification are not necessarily portable to various hosting environments, but any client program that follows the conventions can invoke any Grid service instance conforming to this specification (of course, subject to policy and compatible protocol bindings).

Then, an advantage of OGSI (and OGSA more generally) is that it is not tied to a particular implementation technology. In fact, at the moment, there are already multiple implementations of OGSI:

- **GT3**[192]: It is the best known and most widely deployed implementation of OGSI. It represents the third version of the Globus toolkit, and, in addition to the core interfaces foreseen by the OGSI specifications, GT3 uses this specification to provide powerful services for resource monitoring, discovery, management, security and file transfer. These additional services are the OGSA compliant version of the tools available with the previous version of Globus toolkit (GT2). These services can be put at the level of OGSA platform.
- **MS.NET Grid**[193]: EPCC (Edinburgh Parallel Computing Centre) has been developing an implementation of the Open Grid Services Infrastructure (OGSI) for hosting environment, based upon Microsoft technologies. In particular, their implementation uses :NET technologies wherever possible.

[192] Globus GT3: http://www.globus.org/.

[193] MS.NET Grid: http://www.epcc.ed.ac.uk/~ogsanet/.

- **OGSI.NET**[194]: This is another implementation of OGSI for Microsoft-based hosting environments. Their work is leaded by the aim of demonstrating the effectiveness of Microsoft :NET platform in implementing OGSA compliant services.
- **OGSI::Lite**[195]: OGSI::Lite is an experiment in creating a Grid Services Container using Perl. It is based on the SOAP::Lite package and related modules. It partially supports lifetimes, stateful services, factories, notification and ServiceGroups. At the moment version 0.3 is available for download and they aim to make it fully OGSI compliant with the final OGSI specifications.
- **PyOGSI**[196]: The pyGridWare project's objectives are to make Globus GT3 accessible through a Python interface, develop a standalone Python OGSI server, and develop a full Python implementation of an OGSI-compliant server.

6.5.1.7 WSRF based middleware

The WS-ResourceFramework tries to integrate the functionalities of "the web" with the ones of "the grid". WSRF.NET[197] provides web services running under ASP.NET with the ability to access WS-Resources that are independent of the service.

By design, the WSRF specifications envision a rather flexible structure of the WS-Resources, which includes (1) modeling "state", i.e. providing an object, respectively a set of objects to the executing web service that can be treated as data members. Various storage-types are possible as WSRF.NET uses Microsoft's support for ODBC database connections; and (2), as a second type of WS-Resource, an "active programmatic entity" is envisaged, i.e. lines of code that may be executed separately from the executing web service, as an independent thread.

In order to achieve these goals, WSRF.NET will offer a programming model, which features a similar approach as OGSI.NET's attribute-based model. This means that the service logic (and data) are annotated with meta-data that can be processed with static tools or dynamically by the wrapper service. The attributes supported so far:

(a) Determine which and how data members presented to the web service's methods will be loaded and stored (like loading when accessed, automatic write-back etc.).

(b) Another set of attributes controls the generation of WSRF.NET web services and WS-ResourceProperty documents by static tooling, like specifying functionality that is to be imported into the wrapper service.

(c) The last set so far defines the structure upon which query and update messages can be directed, i.e. annotates the data members which will be integrated into the WS-ResourceProperty document (and hence into WSDL).

These specifications underlie an information flow structure as drawn out in, Figure 48; an ISAPI-filter intercepts the request message of a client at an early point in IIS-procession and dispatches it to ASP.NET. A "wrapper" web service, which overlays the original service in order to encapsulate the WSRF functionality receives the message and contacts the WS-Resource in order to retrieve the data for the web service (depending on the policies defined by the respective attributes).

Regarding security, WSRF.NET will provide all the WSE features from Microsoft, which are available to normal web services running under ASP.NET, too. Furthermore, the GSI protocol from the Globus project will be supported.

---

[194] OGSI.net: http://www.cs.virginia.edu/~humphrey/GCG/ogsi.net.html.

[195] OGSI::Lite Perl and Grid Services: http://www.sve.man.ac.uk/Research/AtoZ/ILCT.

[196] PyOGSI: http://www-itg.lbl.gov/gtg/projects/pyOGSI/.

[197] WSRF.net. http://www.cs.virginia.edu/~gsw2c/wsrf.net.html.

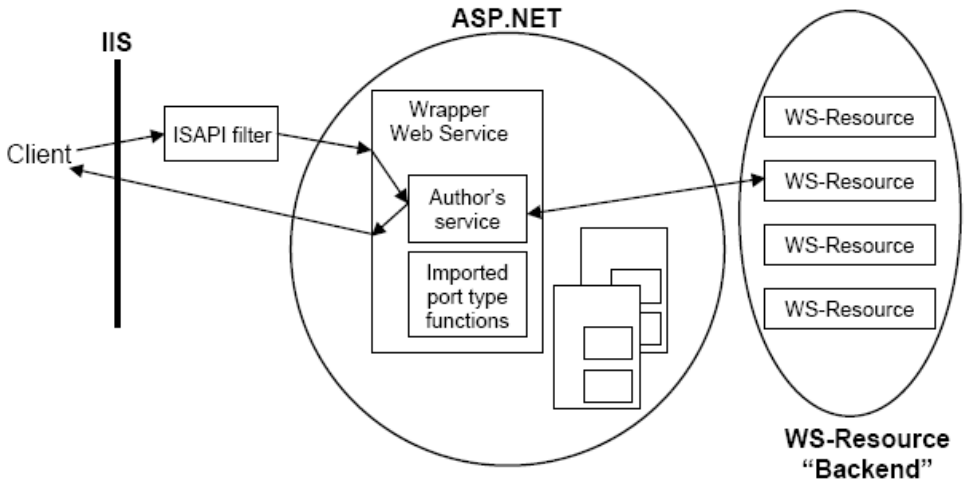


Figure 48 Information flow in WSRF.NET

6.5.1.7.1 Outlook

As yet, the WSRF.NET specifications don't include a specific programming model, besides for the above-mentioned attribute-based model. The development of the architecture is planned to be carried out in three phases, that is

(1) Creating the functionality defined by the various WSRF port types within ASP.NET.

(2) Developing static tooling to assist WSRF.NET service authors.

(3) Finally expanding functionality of WS-Resources in order to allow them to be active entities.

6.5.2 Business Process integration & contracting

6.5.2.1 Microsoft BizTalk

Microsoft BizTalk[198] is a product, which allows the orchestration of business processes inside an enterprise. Microsoft BizTalk Server supports the goal of creating business processes that unite separate applications into a coherent whole. It enables enterprises to connect diverse applications, and then to graphically construct and modify business processes that use the services those applications provide. Additionally, it brings a mechanism for specifying business rules, better ways to manage and monitor applications, and support for single sign-on for those applications.

BizTalk Server also offers services for information workers. These services include a group of Business Activity Services, such as Business Activity Monitoring (BAM) for analyzing running business processes; support for business process provisioning and configuration; and services that enable information workers to set up and manage interactions with trading partners. BizTalk Server contains technology for creating Human Workflow Services (HWS), enabling people to interact with business processes using familiar tools, such as Microsoft Outlook.

[198] Microsoft BizTalk: http://www.microsoft.com/biztalk/

BizTalk offers the ability to define Web services-based business processes by using the Business Process Execution Language for Web Services (BPEL4WS, commonly called just BPEL). BizTalk Server 2004 is built completely around the .NET Framework and Visual Studio .NET. It also has native support for communicating through Web services, along with the ability to import and export business processes described in BPEL. BizTalk is designed to work well both with the emerging world of standard Web services and with the large number of applications already in place.

The following two scenarios are the main scenarios for using BizTalk:

- Connecting applications within a single organization commonly referred to as enterprise application integration (EAI).
- Connecting applications in different organizations, often called business-to-business (B2B) integration.

For TrustCoM, especially the B2B scenario is important as a first step to combine together communications processes of multiple companies.

6.5.2.2 IBM Tivoli

Tivoli Software enables the following:

a) Ensuring the health and appropriate functioning of IT environments. The power of automation in the IBM on demand operating environment.

b) Business Service Management. Aligning IT resources with business priorities. Orchestration. Sensing, triggering, and responding according to business goals. Simplification through coordinated orchestration.

c) Provisioning. Making the right resources available to the right processes and people.

d) Security. Ensuring information assets, confidentiality, and data integrity. Improve security and lower costs -- Identity management solutions.

e) Storage and Optimization. Protecting and maximizing the integrity and availability of your e-business data.

Tivoli software helps your organization become integrated for efficiency -- leveraging open systems technologies across your entire IT operation and increasing productivity by using business processes to automate workflow. With Tivoli software, your business can become automated for productivity -- reducing the amount of time and resources spent diagnosing infrastructure problems and employing self-configuring, self-healing, self-optimizing and self-protecting technology, helping you meet customer needs quickly and build and maintain high levels of customer service. And Tivoli software helps you optimize for business value -- aligning your IT infrastructure with your business processes and priorities to enable your business to do business on demand.

Tivoli intelligent management software combines open, integrated, rapidly deployable computing technology, autonomic functionality and best practices and methodologies to provide more efficient and effective management of your entire IT infrastructure with fewer resources. IBM has expanded its portfolio of intelligent management software to better help you achieve the critical goals of integrating, automating and optimizing your business for the on demand world.

6.5.2.3 SAP Exchange Infrastructure

Enabling Process-Centric Collaboration

SAP Exchange Infrastructure (SAP XI) is SAP's platform for process integration based on the exchange of XML messages.

SAP Exchange Infrastructure enables process-centric collaboration by

- Providing a technical infrastructure for XML-based message exchange in order to connect SAP components with each other, as well as with non-SAP components
- Delivering business-process and integration knowledge to communicating entities, in the form of predefined business scenarios
- Providing an integrated toolset for building new business scenarios by defining and maintaining all integration-relevant information ("shared collaboration knowledge")
- XML Data Exchange — Flexible and Open Communication

One of the key principles of SAP Exchange Infrastructure is that data is exchanged between application systems in the form of XML messages. XML provides a whole new set of standards and technical building blocks that provide a huge step forward in terms of interoperability.

XML is not only used for meta data, it is also the syntax for everything that is exchanged between heterogeneous systems. With XML, you have a flexible basic format for transporting and transforming information. Message formats can be based on open XML standards for business documents, or can be custom designed. Using XML-based mapping, it is possible to transform messages to and from a common XML format or directly from inbound into outbound format.

6.5.2.4 SAP Web Application Server

The SAP Web Application Server (SAP Web AS) is the application platform for SAP applications and solution, i.e. it provides the complete infrastructure to develop, deploy and run all SAP applications. The major key capability of SAP Web AS is the full support for both the proven ABAP[199] technology and the innovative open source internet-driven technologies Java, Java 2 Enterprise Edition (J2EE) and Web Services.

6.5.2.5 The SAP Web AS

- Integrates the ABAP and Java personalities in one application server – a homogeneous infrastructure for J2EE-based and ABAP-based applications – supporting the existing ABAP applications - ABAP-based, Java-based and Web-based application development – Operating System (OS) and Database (DB) portability for both ABAP-based and Java-based applications - high performing internal ABAP-Java communication.
- Brings together the benefits of a proven, scalable and reliable infrastructure with the interoperability and flexibility of Web Services technology may act as both the Web Services client and server for easily integrating existing and new enterprise applications driving collaborative scenarios that cross administrative boundaries.
- Is the powerful Java platform
    - J2EE certified application server
        - SAP enhancements enable the enterprise quality application development in Java

Open SQL for Java framework provides with the DB independency and high performance database programming in Java.

[199] An SAP business application oriented interpreter language

6.5.2.6 X.509 Parsing Server

The X.509 and IETF PKIX standards assume that Public Key Certificates (PKCs), Attribute Certificates (ACs) and Certificate Revocation Lists (CRLs) will be held in LDAP directories, and will be fetched from their by PKI clients (e.g. when construction validation paths). Unfortunately, LDAP servers do not provide the ability to search for PKCs, ACs and CRLs, so path construction can be extremely arduous. LDAP clients can only retrieve PKCs, ACs and CRLs if they know the distinguished names of the entries they are held in.

XPS Server

A recent project at the University of Salford has produced an X.509 Parsing Server (XPS), integrated into the OpenLDAP software. When presented with an X.509 PKC, AC or CRL, the XPS front end parses the certificate, extracts all the fields, and then stores each field as a separate attribute in newly created X.509 entry. LDAP clients are now able to search for PKCs, ACs and CRLs that match specific criteria e.g. a PKC with a particular keyUsage field, a CRL with a particular thisUpdate field, and AC with a particular value of a role attribute etc.

6.5.2.6.1 Application to TrustCoM

This feature can be used in policy management. Since certificates can (should) hold the IDs of the policy/ies under which they are issued, it is now possible to search for all certificates containing certain policy IDs.

This feature can help to provide privacy for users. With classical X.509, the distinguished name of the user must be the same in their PKCs and ACs, so that they can be matched together. With XPS servers in place, other fields, rather than distinguished names, could be used to link PKCs to ACs together. Distinguished names could then be pseudonymously assigned to users.

This feature could help in enforcing separation of duties, for example if the same real world person had multiple certificates under different DNs, but containing the same email address or other identifier, then it would be possible to search for this and detect the other certificates.

This feature can be used to help to locate the certificates of services. If a property of a service is known, and this is held in its certificate, then it is possible to search the LDAP directory to find the certificate without needing to know the distinguished name of the service or server.

In short, if any component of TrustCoM needs to find an X.509 certificate containing a certain field, then using the XPS server to front-end LDAP servers will enable this to be done.

6.5.2.7 Trust and security management tools

Specific trust and security management tools are discussed in the respective chapters on trust management, and policies and security. We especially refer to Ponder (section 8.3), Sultan (section 7.8), Permis (section 8.12), Delegent (section 8.13), and Trust-X (section 7.6).

6.6 Conclusions

Web Services are expected to play a central role within TrustCoM. Web Services provide a standardized framework for interoperable, secure, reliable and transacted messaging, and constitute the ideal underlying service-oriented messaging framework for enabling collaboration within dynamic VOs across enterprises. The core Web Services specifications constitute a common level of interoperability. The core specifications are accompanied by an extensive, composable set of Web Services based specifications dealing with security, reliability, transactability, and business processing, which are heavily metadata supported and driven. As described in more detail in other chapters, these specifications can – in addition to providing a common level of interoperability – form a flexible and extensible basis for TrustCoM. The specifications should be able to support and complement tools and technologies for trust, security and contract management. Support and complementarity can particularly be realized through the composability of the Web Services specifications, and the extensibility of individual specifications (e.g., WS-Security supports any type of XML security tokens, and allows a flexibility usage of these tokens). Metadata can be associated with Web services in different ways and for different purposes. Metadata allows to describe the interfaces of web services, to register web services, to discover web services, and to formulate and attach policies to web services. The overall web services messaging framework and the generic metadata mechanisms should therefore support and enable technologies for interoperable trust and contract management in VOs.

The Grid computing concept has been generalised to cover the notion of a Virtual Organisation, defined as any dynamic collection of individuals and enterprises which are required to share resources to achieve a common goal. An important trend is that Grid technology is converging with Web services. Particularly, while traditional Grid technologies were sometimes quite monolithic and domain-specific, new Grid technologies are leveraging Web services and their composable nature, in order to provide specific Grid features. The TrustCoM framework should be built upon this emerging convergence, and also leverage and extend Web services standards. As a consequence, the TrustCoM mechanisms should then be able to support VO scenarios using different domain-specific extensions of the core Web Services standards such as defined in the Grid domain or the business domain.

The Semantic Web offers the infrastructure to share the vocabulary and semantic of policies and trust valuations. This has the advantage of using an established body of languages and tools designed to function over open distributed systems to leverage this sharing in an effective and economic manner. There has been some preliminary work in providing piece of this architecture carried out in different places. However, there has been no coherent scheme yet to bring these together in one policy-based service architecture.

The tools and platforms which have been identified so far should form a solid basis for the implementation of TrustCoM. With Microsoft .NET and the Java platform (including Apache), two enterprise programming models exist which are a good foundation for TrustCoM. Both Microsoft .NET WSE and the IBM WebSphere platform implement many of the security-related web service specifications envisioned for usage inside TrustCoM. Both platforms already have proven that interoperability in the security space exists. With SAP's Exchange Infrastructure, IBM's Tivoli and Microsoft's BizTalk, a wide range of commercially available platforms exist which are used in today's business processes and which form a promising technology target for the adoption of TrustCoM-enabled technologies. Finally, a number of specific tools related to trust and security management exist which may be relevant in the TrustCoM project.

7 Trust Management

Edited by: Lorenzo Martino
University of Milan

7.1 Introduction

Blaze and Feigenbaum and Lacy[200] defined trust management as " ..a component of security in network services. Trust management problem include formulating security policies and security credentials, determining whether particular sets of credentials satisfy the relevant policies and deferring trust to third parties".

The topic of trust incorporates issues such as trust establishment and trust management. The latter one can be achieved by exploiting the properties of trust, such as its relativity to a given context (not absolute), its directionality (from a relying party to a trusted party), its quantifiability, its existence and evolution in time and its transferability (potentially in absence of relational transitivity). Trust is modelled differently based on the reference application and nature of the established relationships between interacting entities within an (virtual) organisation[201].

According to the above distinction between trust establishment and trust management, this chapter will firstly review the proposed trust models and trust metrics. Then trust services emerging mainly in the e-commerce area will be analysed. Trust models, metrics, and trust services will set the framework and the requirements where security-oriented aspects of the Security infrastructure will be analysed, such as policy-related proposed extensions of X.509, attack resistance of certification paths, CA interconnections and cross-certification

Then the emerging Trust Negotiation approach will be addressed, and finally more comprehensive solution to trust management, encompassing concept such as experience, risks, reputation and trusting propensity in order to specify and evaluate trust, will be described.

7.2 Trust models and Trust metrics

7.2.1 A survey of Trust Definitions

This section is based on surveys by Grandison et al.[202] and Dimitrakos et al.[203].

The general notion of trust is overly complex and appears to be attributed many different meanings depending on how it is used. There is also no consensus in the computer and information sciences literature on what trust is, although its importance has been widely recognised. On the other hand, as it is highlighted in [202] and [203], many researchers assume an (unprovided) definition of trust and use the term in a very specific way related to

[200] M. Blaze, J. Feigenbaum, J. Lacy. Decentralized Trust Management. IEEE Symposium on Security and Privacy. Oakland CA, 1996

[201] IST 6th Framework Program Integrated Project Proposal. TrustCoM A Trust and Contract Management Framework enabling secure collaborative business processing in on-demand created, sel-managed, scalable and highly-dynamic Virtual Organization.

[202] Grandison T., Sloman M. A Survey of Trust in Internet Applications. In *IEEE Communications Surveys and Tutorials*, Fourth Quarter 2000.

[203] Dimitrakos T., Bicarregui J.C. Towards A Framework for Managing Trust in e-Services. In Proceedings of the *4th International Conference on Electronic Commerce Research*, ATSMA, IFIP, November 2001. ISBN 0-9716253-0-1.

authentication and authorisation or to paying for purchases. The following are among the few attempts to provide definitions of trust that are useful for information technology.

Kini and Choobineh examine trust in [204] from the perspectives of personality theorists, sociologists, economists and social psychologists. They highlight the implications of these definitions and combine their results to create their definition of trust in a system. They define trust as: *"a belief that is influenced by the individual's opinion about certain critical system features".* Their analysis covers various aspects of human trust in computer dependent systems but they do not address the issue of trust between parties (humans or processes) involved in e-commerce transactions.

Gambetta examines trust in [205] and propose the following definition: *" …trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [the trustor] can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects [the trustor's] own action."* Gambetta's definition stresses that it is fundamentally a belief or an estimation.

Castelfranchi and Falcone[206] extend this definition to include the notion of competence along with predictability.

The Trust-EC project (http://dsa-isis.jrc.it/TrustEC/) of the European Commission Joint Research Centre (ECJRC) defines [207] trust as: *"the property of a business relationship, such that reliance can be placed on the business partners and the business transactions developed with them".*

They state that the issues of the identification and reliability of business partners, the confidentiality of sensitive information, the integrity of valuable information, the prevention of unauthorised copying and use of information, the guaranteed quality of digital goods, the availability of critical information, the management of risks to critical information, and the dependability of computer services and systems. In particular, they emphasise the following aspects of dependability:

- Availability, reliability and integrity of infrastructure;
- Prevention of unauthorised use of infrastructure;
- Guaranteed level of services;
- Management of risks to critical infrastructure

are key to the emergence of e-commerce as a viable commercial activity.

Grandison and Sloman survey in [202] various definitions of trust. Following a brief analysis of these definitions, they build their own definition of trust as:*"the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context".* They argue that trust is a composition of many different attributes - reliability, dependability, honesty, truthfulness, security, competence, and timeliness - which may have to be considered depending on the environment in which trust is being specified.

---

[204] Kini A., Choobineh J., "Trust in Electronic Commerce: Definition and Theoretical Consideration". Proc. *31st International Conference on System Sciences*, IEEE, 1998.

[205] Gambetta D., "Can We Trust Trust?" In *Trust: Making and Breaking of Cooperative Relations*. Basil Blackwell, Oxford, 1990, pp 213-137.

[206] Castelfranchi, C., Falcone, R., "Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification". In Proc. of the Third International Conference on Multi-Agent Systems, ed. Demazeau, Y. IEEE C.S., Los Alamitos, 1998, pp. 72-79.

[207] Jones S., "TRUST-EC: requirements for Trust and Confidence in E-Commerce", European Commission, Joint Research Centre, 1999.

Dimitrakos[208] has defined trust of a service requestor A to a service provider B for a service X as follows:

> *"Trust of a party A in a party B for a service X is the measurable belief of A in B behaving dependably for a specified period within a specified context in relation to X. Where:*
>
> - *A party can be an individual entity, a collective of humans or processes, or a system; (obviously, the trustor must be an entity that can form a belief).*
> - *The term service is used in a deliberately broad sense to include transactions, recommendations, issuing certificates, underwriting, etc.*
> - *The above mentioned period may be in the past, the duration of the service, future (a scheduled or forecasted critical time slot), or always.*
> - *Dependability is used broadly to include security, safety, reliability, timeliness, and maintainability (following [209]).*
> - *The term context refers to the relevant service agreements, service history, technology infrastructure, legislative and regulatory frameworks that may apply.*
> - *Trust may combine objective information with subjective opinion formed on the basis of factual evidence and recommendation by a mediating authority.*
> - *Trust allows one agent to reasonably rely for a critical period on behaviour or on information communicated by another agent. Its value relates to the subjective probability that an agent will perform a particular action (which the trustor may not be able to monitor) within a context that affects the trustor's own actions."*

In analogy to the above, Dimitrakos differentiates distrust from "lack of trust" as follows:

> *"Distrust of a party A to a party B for a service X is A's measurable belief in that B behaves non-dependably for a specified period within a specified context in relation to service X."*

Distrust is useful in order to revoke previously agreed trust, obstruct the propagation of trust, ignore recommendations, and communicate that a party is "blacklisted" for a class of potential business transactions.

Some aspects of these definitions are common, other are complementary. For example, [202] emphasises that trust is a belief in the competence of an entity within a specified context, while [204] lay stress on that the entity that manifests trust (the "trustor") is the human - not the system. A somewhat similar view is expressed in [210] where entities are distinguished into *passionate*, who have free will, and *rational,* who don't. According to [204] and [210] trustors are *passionate* entities. [204,205] emphasise that trust is in part subjective. The definition in [207] focuses on another aspect of trust: in commerce, *trust is relative to a business relationship*. One entity may trust another entity for one specific business and not in general. This diversity of the purpose of trust is also mentioned in [210] but not incorporated into a definition. Finally, none of the above emphasises that trust is not only inherently measurable (viz. quantifiable) but also it exists and evolves in time.

Notably, the definition in [208] differs from [204,205] with respect to the trusting subjects. Intelligent agents who negotiate can be either humans or programs and in both cases they need to manifest trust intentions and establish trusting relationships. Intelligent software

---

[208] Dimitrakos T. System Models, e-Risk and e-Trust. Towards bridging the gap? in *Towards the E-Society: E-Business, E-Commerce, and E-Government*, eds. Schmid B., Stanoevska-Slabeva K., Tschammer V., Kluwer Academic Publishers, 2001. (Proceedings of the 1st IFIP conference on e-commerce, e-business, e-government.)

[209] Laprie J.C., *Dependability: Basic Concepts and Terminology,* Springer-Verlag, 1992.

[210] Jøsang A. "The right type of trust for distributed systems". In *Proc. of the New Security Paradigms Workshop*, ACM, 1996.

agents are adaptive autonomous programs featuring the ability to acquire knowledge and to alter their behaviour through learning and exercise. Their decision making can be enhanced so that they form trust intentions and make decisions relying on trust.

The definition in [208] also differs from [202,207] and [211] with respect to the inherent measurability and the subjective nature of trust. As Yahalom *et al* also emphasise in [173], regarding an entity as being *generally "trusted"* or *"untrusted"* may result in an oversimplified view. Its entity is expected to perform (or not perform) various tasks in the same or different business contexts. Each of these tasks has its own characteristics, significance and verifiability. In reality, it is often more reasonable to trust an entity with respect to some tasks and not necessarily with respect to some others. They also note that it may be reasonable to assign different *degrees of trust* to each class but throughout [173] they assume only the two extremes for each class. In our working definition we view trust as quantifiably belief. Its metric is based on evidence, experience and perception. The measurement can be quantitative (e.g. as a probability) or relative (e.g. by means of a partial order). There are some interesting arguments, mainly of a philosophical nature, for and against each of these alternative metrics. In practice, either type of metric may be preferable depending on the deployed trust management scheme.

The definition in [208] also differs from [207,211] as it allows different parties with different roles in a transaction to have different views on trust in each other or in third parties. To a certain extent, trust is subjective. Furthermore, it differs from [202,205,206,210] in that trust differentiates between services and it is active for critical periods of time. According to our definition trust refers to a particular business transaction and momentum. Different laws may govern different trusting relationships for different business transactions at different times. Since trust may be relativised to a service that is relevant for a critical interval, it is reasonable to expect that trust statements are time-stamped and may become irrelevant outside this interval. Finally, our definition allows for trust in oneself to be defined and be quantifiable. This supports the ability of an agent to delegate or offer a task to another agent in order to improve efficiency or reduce risk.

7.2.1.1 Some basic properties of Trust relations

The particular characteristics of trust may differ from business to business. Nevertheless, there are some common delimiters that indicate the existence of general principles governing trust relaitons. Dimitrakos conducted in [208] and [212] an analysis identifying some fundamental properties of trust relations in B2B and B2C contexts and then studied properties related to the propagation and transferability of trust through business relationships. In this subsection we provide a summary of these results.

The following are general properties of trust and distrust.

1. *Trust is relativised to some business transaction.* A may trust B to drive her car but not to baby-sit.
2. *Trust is a measurable belief.* A may trust B more than A trusts C for the same business.
3. *Trust is directed.* A may trust B to be a profitable customer but B may distrust A to be a retailer worth buying from.
4. *Trust exists in time.* The fact that A trusted B in the past does not in itself guarantee that A will trust B in the future. B's performance and other relevant information may lead A to re-evaluate her trust in B.

[211] Yahalom R., B. Klein and T. Beth. "Trust relationships in secure systems -A distributed authentication perspective". In Proc. of the *1993 IEEE Symposium on Research in Security and Privacy*, pages 150--164, May 1993.

[212] Theo Dimitrakos A Service-Oriented Trust Management Framework Trust, Reputation and Security: Theories and Practice LCNS/LNAI special volume, Springer Verlag, (2003)

5. *Trust evolves in time, even within the same transaction.* During a business transaction, the more *A* realises she can depend on *B* for a service *X* the more *A* trusts *B*. On the other hand, *A*'s trust in *B* may decrease if *B* proves to be less dependable than *A* anticipated.

6. *Trust between collectives does not necessarily distribute to trust between their members.* On the assumption that A trusts a group of contractors to deliver (as a group) in a collaborative project, one cannot conclude that A trusts each member of the team to deliver independently.

7. *Trust is reflexive, yet trust in oneself is measurable.* A may trust her lawyer to win a case in court more than she trusts herself to do it. Self-assessment underlies the ability of an agent to delegate or offer a task to another agent in order to improve efficiency or reduce risk.

Remarks.

- Property 1 states that trust depends on the *tasks* that each entity is expected to perform in the context of some particular business. As Yahalom *et al* also emphasise in [211], regarding an entity as being *generally "trusted"* or *"untrusted"* may result in an oversimplified view. Its entity is expected to perform (or not perform) various tasks in the same or different business contexts. Each of these tasks has its own characteristics, significance and verifiability. In reality, it is often more reasonable to trust an entity with respect to some tasks and not necessarily with respect to some others. They also note that it may be reasonable to assign different *degrees of trust* to each class but throughout [211] they assume only the two extremes for each class.

- Property 2 states that there are different *degrees of trust*: an agent *A* may trust agent *B* more than *A* trusts agent *C* for the same task and in the same business context. The metric is based on evidence, experience and perception. The measurement can be quantitative (e.g. as a probability) or relative (e.g. by means of a partial order). There are some interesting arguments, mainly of a philosophical nature, for and against each of these alternative metrics. In practice, either type of metric may be preferable depending on the deployed trust management scheme.

- Property 3 states that different parties with different roles in a transaction may have different views on trust in each other or in third parties. To a certain extent, trust is subjective.

- Property 4 states that trust refers to a particular business transaction and momentum. Different laws may govern different trusting relationships for different business transactions at different times. Since trust may be relativised to a service that is relevant for a critical interval, it is reasonable to expect that trust statements are time-stamped and may become irrelevant outside this interval.

- Property 5 emphasises dependence of trust on a sequence of events. Assume *A* trusts *B* for a service *X* during a business transaction that lasts for a limited period. During this transaction, *A* keeps information about *B*'s performance and may use this and any other relevant information (such as recommendations about *B*) to re-evaluate her trust in *B* throughout the service. The more *A* realises she can depend on *B* for *X* the more *A* trusts *B* for this service. Whereas *A*'s trust in *B* may decrease if *B* proves to be less dependable for *X* than *A* expected. (Being *"less dependent"* involves, for example, *A* observing that *B* is less competent than expected, some reliable source discrediting *B* to *A* or *B* trusting one of *A*'s competitors for a related service, etc.) At the end the end of the service *A* may store the overall performance of *B* and consider this information before she enters into a future business relationship with *B*. However, the fact that *A* trusted *B* in the past does not in itself guarantee that *A* will trust *B* in the future. Changes in *B*'s reputation for services of this type and the establishment of new trust relationships between entities may make result in *A* distrusting *B* for the provision of the same type of service in a different business context or time.

- Property 6 distinguishes trust in a collective from trust in its members. On the assumption that *A* trusts a group of contractors to deliver (as a group) in a collaborative project, one cannot conclude that *A* trusts each member of the team to deliver in the project. A potentially bad performance of a member of the group can be overshadowed by potentially excelling performance of another.
- Property 7 supports the ability of an agent to delegate or offer a task to another agent in order to improve efficiency or reduce risk.

Dimitrakos in [208] notes that trust is not necessarily transferable. That is, on the assumption that A trusts B for a service X and B trusts C for X (or any part of it) one cannot necessarily infer that A trusts C for X. However, at least unintentional transferability of trust within a locus may be acceptable in specific contexts. *Note that "transferability" in our case corresponds to influencing the level of trust rather than relational transitivity.* Then in [208] and [212] he analyses some fundamental properties related to the transferability of trust. These properties provide the foundation of a role-based model (which was first sketched in [208] and then elaborated in [203] and [212]) that is able to support an elaborate analysis of some basic trust relationships and the structural properties underpinning the propagation of trust in open dynamic systems. We revisit these properties and explain them in detail. We distinguish three special *roles* that entities mediating in a trust relationship can play. These roles are *guarantors, intermediaries*, and *advisors*. Of course the same system entity may play more than one mediating role in a business relationship.

As we elaborate in the sequel, at least unintentional transferability of trust within a locus may be acceptable in specific contexts. *Note that "transferability" in our case corresponds to influencing the level of trust rather than relational transitivity.* We distinguish three special *roles* that entities mediating in a trust relationship can play. These roles are *guarantors, intermediaries*, and *advisors*. Note that an entity may play more than one mediating role in a business relationship.

- ***Guarantor*** is a party taking the responsibility that the obligations of the parties she acts as a guarantor for are fulfilled at an agreed standard. Guarantors assist the establishment or facilitate the increase of trust for a specific transaction by underwriting (a part of) the risk associated with the transaction. A typical example is a credit card company.
- ***Intermediary*** is a party that intervenes between other parties in a business transaction and mediates so that they establish a business relationship with or without their knowledge. We distinguish the following types of intermediary:
  - *Transparent:* an intermediary that identifies the parties she is mediating between to each other. An example is `Lloydstsb.com`, a bank, who offer to their on-line customers a comprehensive car rental and flight booking service powered by `Expedia.co.uk`, an on-line travel agency. A trivial example is an entity that simply redirects to another entity.
  - *Translucent*: an intermediary that identifies the existence of the parties she mediates between but not their identity. An example is a retailer advertising product delivery by courier without identifying which delivery company is responsible for this.
  - *Overcast:* an intermediary that hides the existence of the parties she is mediating between from each other. Examples include virtual enterprises, and ventures selectively outsourcing tasks to unidentified strategic allies.
  - *Proxy:* an intermediary who is authorised to act as a substitute of another entity.
- ***Advisor*** is a party that offers recommendations about the dependability of another party. Advisors include the authorities maintaining blacklists for a community. Examples include credit scoring authorities and reputation systems.

Trust and distrust propagate according to the following rules:

8. *(Dis)trust is not transferred along an overcast intermediary*. Assume that A (dis)trusts an overcast intermediary T for a service X provided by B. Since A is not aware that B provides the service, her (dis)trust is placed in T.
9. *Trust is transferred along transparent intermediaries – distrust is not*. Assume that, for a service X, A trusts a transparent intermediary T mediating for B. By agreeing to the service, A expresses trust in B for X instigated by T's mediation.
10. *(Dis)trust in a subcontractor of a transparent intermediary is transferred to (dis)trust in the intermediary.* If a party A (dis)trusts a subcontractor of a transparent intermediary T for a service X, then A is inclined to (dis)trust T for this particular service.
11. *Trust is transferred anonymously along translucent intermediaries – distrust is not.* Assume that A trusts a translucent intermediary T for X and T trusts B to subserve for X. By agreeing to the service, A effectively expresses trust in a third party to subserve for X without necessarily knowing the identity of that party.
12. *Trust in an advisor is transferred to the recommended party - distrust is not*. The more A trusts T the more she relies on her recommendation.
13. *Distrust in a recommended party is transferred to the advisor – trust is not*. A's distrust in a party B recommended by T for a service X prompts A to question T's competence as an advisor for X.
14. *Advisors distinguish between recommendations based on "first hand" and "second hand" evidence. In the latter case they ought to identify their sources.* If T_1 and T_2 both pass to A advise by T as their own observations then T gains an unfair advantage in influencing A.
15. *Distrust propagates through trust and it obstructs the propagation of trust.* If A distrusts an intermediary T for a service X then A will ignore T's mediation to the extent of the distrust.

Note that properties 9, 10 and 12, 13 above, allow for trust and distrust to be transferred in opposite directions. This does not necessarily result in a conflict. The opposite initial values will affect each other and the final decision will depend on the resulting balance between trust and distrust in each party, and the tendencies of the trustor. This would not have been possible, had trust been viewed as a binary operator, because transitivity of trust would have lead to inconsistency. Furthermore *distrust propagates through trust and it obstructs the propagation of trust.* If A distrusts an intermediary T for a service X then A will ignore T's mediation to the extent of the distrust.

7.2.2 Trust Definition and Properties of Trust Relationships underpinning SULTAN

This section is based on the work carried by Grandison and Sloman[213].

Trust can be defined as a quantified belief by a trustor with respect to competence, honesty, security and dependability of a trustee within a specified context [213] where the trustor is the entity that trusts and the trustee is the entity that is trusted.

In general, a trust relationship is not absolute — A will never trust B to do any possible action it may choose. A trustor trusts a trustee with respect to its ability to perform a specific action or provide a specific service within a *context*. For example, a person is only trusted to deal with financial transactions less than $2000 in value. Even trust in oneself is not usually absolute, and there is a need to protect resources you own from mistakes or accidents you may cause. Examples include protecting files from accidental deletion or mechanisms to prevent a person driving a car when under the influence of alcohol.

213 T. Grandison and M. Sloman. Specifying and Analysing Trust for Internet Applications, 2nd IFIP Conference on E-Commerce, E-Business and E-Government. 2002. Lisbon, Portugal.

A trust relationship can be one-to-one between two entities, however it may not be symmetric. A's trust in B is not usually the same as B's trust in A. It may be a one-to-many relationship in that it can apply to a group of entities such as the set of students in a particular year. It can also be many-to many such as the mutual trust between members of a group or a committee, or many-to-one such as several departments trusting a corporate head branch. In general, the entities involved in a trust relationship will be distributed and may have no direct knowledge of each other, so there is a need for mechanisms to support the establishment of trust relationships between distributed entities.

There have been suggestions that trust relationships should not be transitive[214]. However, some trust scenarios do exhibit transitivity. The concept of trust delegation is a prime example of the application of trust transitivity. When I delegate my trust decisions to another, for example John, I authorize John to make trust decisions on my behalf. Thus, when I delegate to John and John trusts an unknown entity (say Tim), John is essentially stating that I trust Tim. According to Christianson and Harbison[215] the concept of transitivity should be avoided, as it can result in entity B adding trust assertions to an entity A's trust base without A's explicit consent, leading to *unintentional transitivity*. We agree that transitivity of trust may have unexpected and adverse results if it implies updating the trust base of a trustor to include derived assertions, but it may be necessary in some situations. We consider transitivity to be inherent in some relationships and so should be considered in the analysis of trust systems in order to determine which undesired side effects should be prevented.

There is often a *level* of trust associated with a relationship[216]. Some entities may be trusted more than others with respect to performing an action. It is not clear whether this level should be discrete or continuous. If discrete values are used, then a qualitative label such as high, medium, or low may be sufficient. Some systems support arithmetic operations on trust recommendations, so numeric quantification is more appropriate. It is also possible to provide a mapping from qualitative to numeric labels. However, there is still a problem relating to representation of ignorance (or the unknown) with respect to trust. Jøsang's Opinion Model, based on subjective logic, may be suitable technique for assigning trust values in the face of uncertainty[217,218,219,220]. An opinion is a representation of a belief and is modeled as a triplet, consisting of: *b* (a measure of one's belief), *d* (a measure of one's disbelief), and *i* (a measure of ignorance), such that $b + d + i = 1$. It is assumed that *b*, *d* , and *i* are continuous and between 0 and 1 (inclusive). This model's strength lies in the ability to reason about the opinions (on a mathematically sound basis) and its consensus, recommendation, and ordering operators [217]. However, its major weakness is that it cannot be guaranteed that users will accurately assign values appropriately.

From the literature it is clear that there are many different types of trust, which relate to the specific purposes or nature of a trust relationship.

214 D. Povey, "Trust Management," 1999, http://security.dstc.edu.au/presentations/trust/

215 B. Christianson and W. S. Harbison, "Why Isn't Trust Transitive?" Security Protocols Int'l. Wksp., 1996, University of Cambridge.

216 F. L. Mayer, "A Brief Comparison of Two Different Environmental Guidelines for Determining 'Levels Of Trust' (Computer Security)," 6th Annual Computer Security Applications Conf., 1990, http://ieeexplore.ieee.org/iel2/319/3856/00143781.pdf

217 A. Jøsang, "Artificial Reasoning with Subjective Logic," 2nd Australian Wksp. Commonsense Reasoning, 1997, http://www.idt.ntnu.no/~ajos/papers.html

218 A. Jøsang, "Prospectives for Modeling Trust in Information Security," Australasian Conf. Information Security and Privacy,1997: Springer-Verlag, http://www.idt.ntnu.no/~ajos/papers.html

219 A. Jøsang, "A Subjective Metric of Authentication," 5th European Symp. Research in Computer Security (ESORICS'98), 1998: Springer-Verlag, http://www.idt.ntnu.no/~ajos/papers.html

220 A. Jøsang, "The Right Type of Trust for Distributed Systems," ACM New Security Paradigms Wksp., 1996, http://www.idt.ntnu.no/~ajos/papers.html

7.2.3 Trust metrics

7.2.3.1 Definition of Trust Metrics

The meaning of "trust metric" is to measure the level of trust. One of the early attempts at defining a trust metric is from the Public Key Infrastructure world[221]. In this model, a trust metric is represented in a directed graph, where each node corresponds to a public key, and each edge between a pair of nodes corresponds to a digitally signed certificate. Given an edge from *s* to *t,* the certificate itself is a string identifying *t*, along with a digital signature of this string generated by *s*. This type of trust metric can be used to evaluate trust assertions in a distributed information system. Apart from PKI, new areas and research fields have come to make trust metrics gain momentum. Most generally, trust metrics can be used to represent and answer the following questions:

***A** trusts **B**, **A** trusts **C**, **B** trusts **D**, **C** trusts **D**.*

*Shall **A** trust **D** and if so by how much (*maybe a numerical value)?

This trust model can be simply represented in Figure 49.

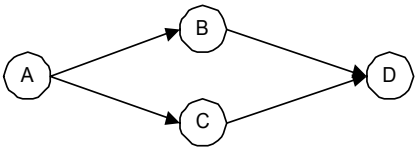


Figure 49 A Simple Trust Model

Moreover, trust models can help in detecting paradoxes, for example "A trusts B, A trusts C, but C totally distrusts B". This paradox should be highlighted rather than simply missing as in Figure 49. Trust models can also help in detecting malicious users which is in fact one of the main advantages of trust models.

In order to measure the trust as a numerical value, weighted edges are introduced in the trust model. These weights can provide primary data for acquiring a trust value. As long as trust values are just “total or missing” (e.g. ***A*** trusts ***B*** and ***C***, no trust statement is expressed to all the other entities), it is pretty easy to represent a trust metric in a weighted directed graph. Using a weighted directed graph raises some problems. If a real value is used to measure a trust value (e.g. ***A*** trust ***B*** as 9.95/10), then what does a negative trust value represent? Are users willing to give negative ratings to other users? Should negative trust values just stop the trust chain or can they be used to infer something such as "***A*** should not trust ***B***"? Using trust metrics also involves issues of:

- How can users easily provide trust statements?
- What is the best way to explain it to users?
- What representation of trust statements can be used to help users?
- How are the results to be explained? (e.g. "***A*** should trust ***B*** as 9.4/10 because ...")
- Can a better representation be used to visualize a social network?

More generally, a trust metric can be defined as “[it] specifies the trustee properties that are relevant to the trustors’ decision of whether or not to place trust in a trustee. Furthermore, a trust metric defines either qualitatively or quantitatively how to assign a rating to a trustee based on observed values of these properties”[222]. The main goal of a trust metric is to rank entities in an online community based on how much the community as a whole "trusts" each entity. In a simple case, given a source entity and a target entity, the goal of a trust metric is

[221] Levien, R. L. (2002) Attack resistant trust metrics, *Phd thesis of the University of California at Berkeley,* USA.

[222] Toone, B., Gertz, M. and Devanbu, P. (2003) Trust Mediation for Distributed Information System, *In: Proceedings of SEC 2003: Security and Privacy in the Age of Uncertainty,* Athens, Greece, 26-28 May 2003.

to determine whether the target is trustworthy. A trust metric assigns an exact amount of trust to each entity. Thus it usually measures how much the community trusts that entity. Because trust metrics are widely used in P2P networks, ubiquitous, mobile computing, and rating systems for online communities, the goals for using them are different. For example, in Levien's project[223], the goal of the trust metric is to accept as many valid accounts as possible, while also reducing the impact of attackers.

The purpose of a trust metric is to create a secure system based on trust between entities. Trust metrics based on a graph-flow model allow the work of certifying Internet users to be distributed throughout the community. Such systems' notions of "trust" include confidence in users' ability to rate others and to distribute the trust accordingly.

7.2.3.2 Classification of Trust Metrics

Trust metrics can be classified simply into two categories: quantitative and qualitative. This classification is determined according to what type of measurement is applied to the trust metric: quantitative (a numerical scale) or qualitative (value judgement words such as minimal, average, good). Formally, a metric can be represented as a measurement of a distance between two points. A short distance might correspond to a high trust between two entities, and a long distance to a low trust, i.e. trust is inversely proportional to distance. Distance quantification, which reduces a measurement to a number, is the easiest approach to implement, but a quantitative measurement is not always available in some metric spaces. Also, trust isn't one dimensional. I trust my mechanic to fix my car, but not to fix my teeth. Thus trust must be measured in context. The more parameters/dimensions that are fixed to the context, the less that need to be taken into account when determining the trust metric. Due to the different natures of trust relationships, some of the contexts may be suitable for quantitative measurement (e.g. selling on eBay), but others may not (e.g. trusting a colleague at work to do a particular task). If in a distributed trust model, both qualitative and quantitative metrics are used, then these can only be combined if a conversion factor for the metrics is available.

Ziegler et al.[224] addressed that "…available metrics can hereby be defined and characterized along various classification axes". They suggested identifying three principal dimensions (classification axes) with distinctive features. Figure 50 illustrates these three classification axes, which are network perspective, computation locus and link evaluation. We have to keep in mind that not all combinations in this space are valid. Figure 51 presents the relations among these three classification axes and the classification scheme.

According to Ziegler and Lausen [224], trust metrics may basically be divided into ones with global, and ones with local scope (the network perspective). Global trust metrics take into account all peers and trust links connecting them. However local trust metrics only take into account personal bias. Many global trust metrics, such as EigenTrust, trust metric in Guha's open rating system and Richardson *et al*'s trust metric, borrow their ideas from the renowned PageRank algorithm[225] to compute web page reputation. These three trust metrics all are quantitative, since nodes in the metrics are ranked using numerical values. Local trust metrics comprise Levien's Advogato trust metric, metrics for modelling the PKI, Golbeck's metrics for Semantic Web trust, and Sun Microsystem's Poblano.

---

[223] Levien, R. L. (2002) Attack resistant trust metrics, *Phd thesis of the University of California at Berkeley,* USA.

[224] Ziegler, C. and Lausen, G. (2004) Spreading Activation Models for Trust Propagation,

IEEE International Conference on e-Technology, e-Commerce, and e-Service (EEE '04), March 28-31, 2004, Taipei, Taiwan.

[225] Page, L., Brin, S., Motwani, R. and Winograd, T. (1998) The pagerank citation ranking: Bring order to the web. *Technical report,* Staford Digital Library Technologies Project.

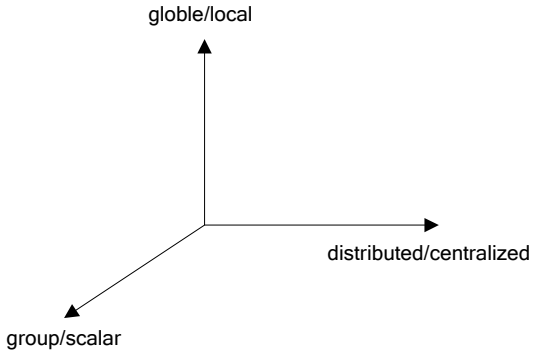


Figure 50 Three classification axes

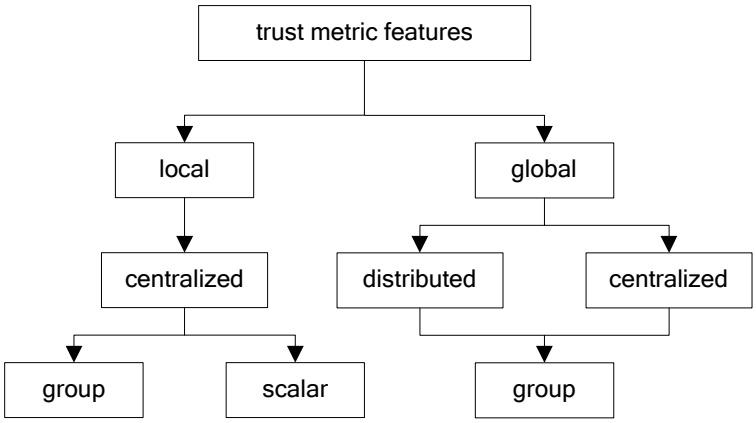


Figure 51 Trust Metrics Classification

Trust metrics can be classified as distributed or centralized according to the second classification axis, computational locus. Local or centralized approaches perform all computations in one single machine and hence need to be granted full access to relevant trust information. The trust data itself may hereby be distributed over the network. Most of the trust metrics mentioned before count among the class of centralized approaches. Distributed metrics for computation of trust, such as EigenTrust and Richardson *et al*'s trust metric, equally deploy the load of computation on every trust node in the network. The entire process of trust computation is necessarily asynchronous and its and its convergence depends on the eagerness or laziness of nodes to propagate information. According to the Figure 51, distributed trust metrics are inherently global. Though the individual computation load is decreased with respect to centralized computation approaches, nodes need to store trust information about any other node in the system.

Finally the third classification axis, termed link evaluation, distinguishes scalar and group trust metrics. Scalar metrics analyze trust assertions independently, while group trust metrics evaluate groups of assertions "in tandem". PageRank [225] and related approaches count among the global group trust metrics, for the reputation of one page depends on the ranks of referring pages, thus entailing parallel evaluation of relevant nodes by means of mutual dependencies. Advogato[226], represents an example for local group trust metrics. Most other trust metrics belong to the category of scalar ones, which searches for paths from sources to targets and does not perform parallel evaluations of groups of trust assertions. The eBay trust metric is a local scalar metric, which is inherent to centralized category. This trust

[226] Levien, R. L. (2002) Attack resistant trust metrics, *Phd thesis of the University of California at Berkeley,* USA.

metric is used in the eBay feedback system, which enables buyers and sellers to rate other buyers and sellers, and presents that reputation information to everyone.

7.2.3.3 Evaluation Of Trust Metrics

Given a trust metric, its adequacy and suitability could be evaluated in terms of ccompleteness, coverage, accuracy, timeliness, consistency, complexity and effectiveness.

The completeness concerns whether a trust metric can provide a complete trust management service, which can be measured qualitatively and/or quantitatively. For example, a trust metric meets user requirements “completely”, “incompletely”, “excessively”, “overlapped” or “wrongly”. Obviously, quantitative measurements can be described more precisely than qualitative measurements since they use numerical values. The completeness measurement can be modeled as comparing two sets. A set is used to denote the actual information that a trust metric provides, and another one represents the trust system’s expected information.

The coverage criterion requires that all paths of a trust metric are accessible and all control transfers in the metric are determinant. Evaluating the adequacy of a trust metric along this criterion helps to reduce redundancies in the metric, so that the efficiency of the system is increased.

The accuracy criterion is used to measure how precise and reliable the trust values are.

Trust is not static but changes with time as a result of experience. The timeliness criterion is designed for assessing this type of time dependencies among the nodes for trust computation.

Consistency considers whether a trust metric can provide consistent results.

Complexity concerns how easily a trust metric can be managed in the trust computations. A complex trust metric used in a trust management system normally results overloaded requirements to users.

The effectiveness criterion measures the performance of a trust metric in terms of computational time and space consuming.

7.2.4 A conceptual framework relating Trust and Risk

Trust management aims to provide a coherent framework for determining the conditions under which a party A takes the risk to depend on a party B with respect to a service X for a specific period within a specific context, and even though negative consequences are possible. On the one hand, increasing the levels of trust facilitates processes to become more efficient but also increases the risk of allowing for the exploitation of vulnerabilities. On the other hand, reducing risks by introducing more security controls increases the overhead and may make electronic services less cost effective. Indeed in several cases it may make sense to live with known vulnerabilities because the overhead of more secure service provision will in the long term cause higher loses than a potential exploitation of these vulnerabilities. One would consequently aim, in principle, to solve an optimisation problem by weighing trust against risk in order to maximise cost efficiency. Hence, trust management subsumes and relies on risk management:

One may employ tailored risk analysis in order to analyse environmental risks, relate them to service goals and assess the most tangible aspects of trust (e.g. the dependability of the information technology infrastructure, the compliance to legislative frameworks, etc.).

Also, risk management allows us to weigh e-service transaction risks against trust, evaluate the impact of a failure in trust and help device countermeasures.

Finally, risk management allows us to analyse the business risks caused by reduced efficiency due to lack of trust.

Note that the above three analyse potentially different types of risk. The first of the above refers to risks that reside in the environment. The second relates to risks caused by vulnerabilities in information technology enabled interactions. The third relates to risks caused by the inability to meet service or business goals.

Trust management becomes more tractable in the presence of a conceptual classification of the different aspects of trust and the corresponding ways they influence behaviour. For this purpose, based in the results of the surveys in [202] and [203], and the analysis in [208] and [203], Dimitrakos extended in [203] and [212] the conceptual framework proposed by McKnight et al. [227]. In the following paragraphs we analyse the derived conceptual model and emphasise the role of risk management as a means of controlling the transition from one layer of this classification to another.

7.2.4.1 Trust Inclinations

Trust inclinations is an intentionally broad term referring to the tendencies of an agent. These are typically influenced by the agent's own view of the environment it inhabits, by the extent it is willing to depend on another potentially unknown agent in a given circumstance, and by the extent it perceives the known institutions and infrastructure to be dependable. The following classification focuses on trust inclinations inherent in an agent or acquired through the agent's exposure to an environment. These constructs do not exist in isolation; they are interdependent. An overview of some basic relationships between them is given in Figure 54 to Figure 57.

- **Situational trust** measures the extent to which a party is willing to depend on an unspecified party in a specific role and a given circumstance.
- **Beliefs** describe an agent's schema about the environment it inhabits. Four categories of primitives contribute to belief formation [227]:
    - Benevolence, i.e. the belief that one cares about the others welfare;
    - Honesty, i.e. that one makes an agreement in good faith;
    - Competence, i.e. that one is able to perform a specific task;
    - Predictability, i.e. that one's behaviour is predictable in a given situation.
    - Dispositional trust is a fifth primitive referring to an agent's persistent tendency to trust oneself and others across a wide spectrum of situations.
- **System trust** measures the extent to which an agent believes that it can depend on the known institutional structures such as legislative, regulatory, reputation systems and the underlying technology infrastructure.

Trusting beliefs weigh the information by which we make decisions in trust and guide the formation of intentions to trust. They are based on ("first-hand") evidence, recommendations (discounted by the trust to the recommender), previous experience, or mere intuition. Trusting beliefs correspond to the measures by which one determines whether a given entity should be trusted given a specific risk profile[228].

[227] McKnight D.H., Chervany N.L. "What is Trust? A Conceptual Analysis and an Interdisciplinary Model". Proc. the 2000 Americas Conference on Information Systems (AMCIS2000). AIS, Lohng Beach, CA, August 2000.

[228] Povey D. "Developing Electronic Trust Policies Using a Risk Management Model". In *LNCS, Vol. 1740*, Springer-Verlag, 1999.

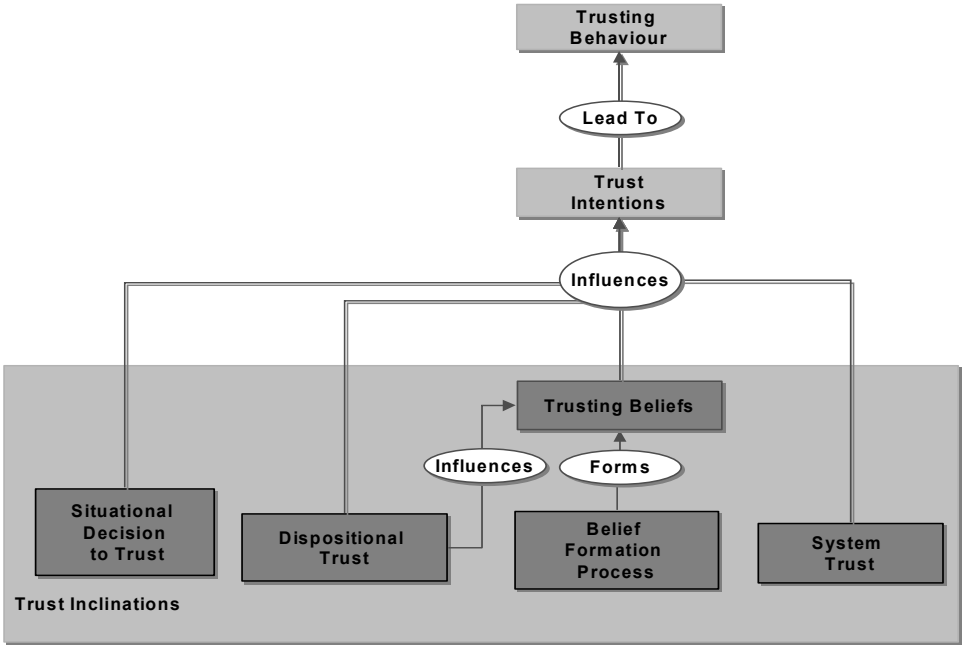


Figure 52 Relationships between trust inclinations and other trust concepts

Trusting beliefs must be relativised to a service and their importance in influencing the confidence by which an agent establishes an intention to trust another entity may vary from service to service. In commercial transactions for example, the issue of benevolence is of a minor importance compared to competence (which is often the prime concern), predictability and honesty. In transactions within virtual communities centred around social interests or charities, on the other hand, benevolence may have significant influence in formation of trusting beliefs.

Dispositional trust contributes to the belief formation process (Figure 52 and Figure 55) but may override this process and directly influence the confidence by which an agent establishes intentions to trust. Dispositional trust may be attributed to an agent's persistent tendency to exhibit trust across a group of contexts either because the agent anticipates a better outcome by exhibiting trust or because the agent has been trained in a controlled environment where she was primarily interacting with trusted parties.

System trust is important as a means for providing stability between system entities, human agents and organisations. Legal and regulatory systems punitive mechanisms to discourage malicious behaviour, while accreditation and certification schemes provide systems that allow us to evaluate the competence of an organisation for a specific task.

Situational trusts is somewhat similar to dispositional trust in that either of them is a general inclination to trust which may override the belief formation process. However, dispositional trust refers to a broad spectrum of contexts whereas situational trust relates only to specific circumstances which may influence confidence in an intention to trust for a particular service and within a given contexts. Situational trust is particularly useful for capturing exceptions to general rules of the belief formation process.

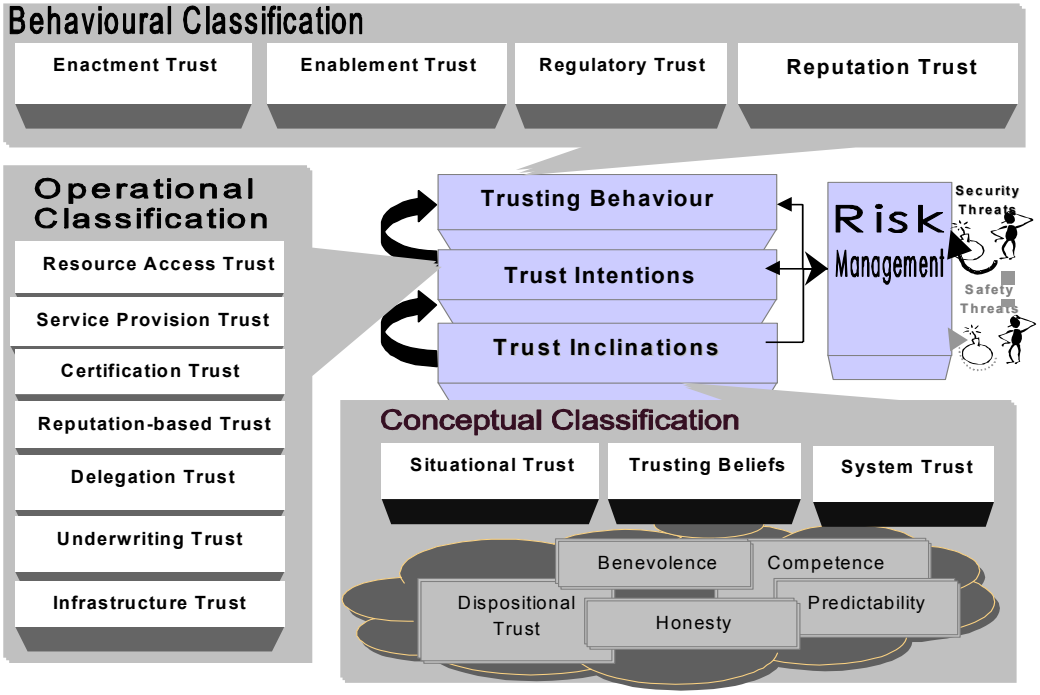


Figure 53 A pictorial overview of the proposed trust-management scheme

7.2.4.2 Trust Intentions

Intentions describe the extent to which a party is willing to depend on other parties (including oneself) for a specified period, within a specified context and in relation to a specific service. Dependable intentions can be modelled within policies, where a policy is viewed as "a rule that can be used to change the behaviour of a system" [Policies allow the management of e-services to be tied into a trust relationship. Indeed, policies can be used as a mechanism for enhancing trust within an e-service environment: an e-service takes on policies and must act in accordance with these policies. These policies are either associated with the result of a trust relationship between two parties or they are specified requirements given by a third party enabling trust relationships to be established.

The following operational classification of trust relates to this viewpoint (Figure 53), focusing on how the intention to trust is controlled and exercised.

- **Resource Access Trust:** for the purposes of a service X, A trusts B to access resources that A controls. This type of trust forms the basis for authorisation policies that specify actions the trusted party can perform on the resources, and constraints that apply such as time periods for when the access is permitted.
- **Provision of Service Trust:** A trusts B to for a service X that does not involve access to A's resources. Application service providers (ASPs) are typical examples of entities that would require service provision trust to be established.
- **Certification-based Trust:** A trusts B for a service X on the basis of criteria relating to the set of certificates presented to A by B and provided by a third party C. Certificates are often used to authenticate identity or membership to a group.
- **Reputation-based Trust:** A trusts B for a service X on the basis of criteria relating to the opinions of other parties who have considered interacting with B in the past for similar services. Examples include reputation systems in e-auctions such as `eBay.com`. This type of trust is often complementary to certification-based trust.

- **Delegation Trust:** For a service X, A trusts B to make decisions on A's behalf about resources that A owns or controls. Examples include the delegation of decisions regarding investment to one's financial advisor.
- **Underwriting Trust:** A trusts B for a service X based on criteria related to the reduction of risk caused by the intervention of a third party C underwriting X. Examples include insurance companies underwriting loss or damage, and credit-card companies guaranteeing payment for a purchase.
- **Infrastructure Trust:** For the purposes of a service X, party A trusts the base infrastructure (subsystem B) upon which the provision of a service will take place.

Typically, a policy is defined at a high level and refined so that it is meaningful in terms of the real system entities and the various locations and contexts in which they exist. A policy need not be directly enforceable but should be a meaningful system constraint that is directly or indirectly measurable. In decentralised open distributed systems, policies apply within a locus, i.e., a subsystem. The latter may be a resource, a single agent, a community of agents or a whole distributed system.

Policies are governing the way a system works and as such they are relatively static with well-controlled procedures for change. An important element of each policy is the set of conditions under which the policy is valid; they must be made explicit in the policy specification. The validity of a policy however, may depend on other policies existing or running in the system within the same scope or context. These conditions are usually impossible or impractical to specify as part of each policy, and therefore need to be specified as part of a group of policies.

As perception and knowledge evolve, an agent may find herself in a position where, according to one policy, pursuing a business relationship with another agent is to her interest, but according to another policy, the same business relationship with the same agent has to be avoided. Meta-policies (i.e., policies “about which policies can coexist in the system or what are permitted attribute values for a valid policy”[229]) are particularly useful for resolving such conflicts[230], often by superimposing an order (viz. priority) on potentially conflicting policies.

In order to build and manage trust and security effectively in globally interconnected electronic communities, a universally acceptable machine-readable policy specification framework, over which different policy descriptions can be interpreted while their semantics are preserved, has to be developed.

7.2.4.3 Trusting Behaviour

Trusting behaviour describes the extent to which a party exhibits trust. It implies acceptance of risks (potential of negative consequences) and their effect. At this level, the agent’s inclinations and intentions have been analysed and endorsed resulting in patterns of behaviour.

The following classes of trust relate to this viewpoint (Figure 53), focusing on the roles of the stakeholders as they engage in a business relationship.

- **Enactment trust** is the trust between parties that engage in a business relationship through e-services, including customers and retailers.
- **Enablement trust** is the trust in those who enable or mediate in the provision of e-services including the technology and platform providers.

---

[229] Damianou N., Dulay N., Lupu E., Sloman M. "The Ponder Policy Specification Language" Proc. Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, 29-31 Jan. 2001, Springer-Verlag LNCS 1995, pp. 18-39

[230] Lupu E. C., Sloman M., "Conflicts in Policy-Based Distributed Systems Management". *IEEE Trans. on Software Engineering*, 25(6): 852-869 Nov.1999.

- **Regulatory trust** is the trust in the legislative, regulatory, standardisation and advisory bodies for e-business at a local or a global level.
- **Reputation trust** is the trust in reputation systems or the recommendation of arbitrary agents.

7.2.4.4 Risk Management

According to the ISO / IEC TR13335 standard [231] for the management of IT security, risk management is the "total process of identifying, controlling and minimising the impact of uncertain events". According to the AS/NZS 4360 standard [232] it consists of "the culture, processes and structures that are directed towards the effective management of potential opportunities and adverse effects". Risk management often involves a form of risk analysis. The latter is *"the process of identifying risks, determining their magnitude, and identifying areas needing safeguards"* [231]. Risk analysis is critical for achieving the right means of abstracting information from reality into a formal model. Its importance has been recognised in the process industry and finance – business areas where elegant methods for risk management have been developed.

The following sub-processes are distinguished, based on the AS/NZ 4360 [232] and HB 4360:2000 [233] standards:

- Establish the context**:** the aim of this sub-process is to establish the strategic, enterprise and system context within which risk management takes place. Establishing the context involves
- Identify risks**:** the initial step in this sub-process is to provide a description of the relevant aspects of the system to be analysed (called "the target system") which must be as complete as possible. Based on such a model of the target system, the assets and the parameters underpinning risk are identified. Analyse risks: this sub-process typically starts with a frequency analysis, which aims at assessing the likelihood of threats or risks, and continues with the consequence analysis, which tries to describe the possible consequences of threats.
- Evaluate risks**:** this involves comparing the level of risks found during the risk analysis with the risk criteria, established during the determination of the context.
- Accept risks**:** the results of the analysis and evaluation of risks are combined into a risk description, which is used as supporting evidence for deciding whether the risks is acceptable or not. If not, corrective measures must be recommended.
- Treat risks**:** this focuses on treating (otherwise unacceptable) risks. The different options for risk treatment can in principle be subsumed by avoiding the risks by not performing an activity; reducing the likelihood of the occurrence of a risk; reducing the possible consequences of a risk; transfer the risks to someone else; or retaining the (residual) risks. Of course, implementing counter-measures to treat risks has a cost associated with it, and they may expose additional risks or retain residual risk. This should be balanced against the expected utility of implementing this counter-measure.
- Monitor and Review: most recent risk management standards identify additional concurrent sub-processes for monitoring and review and communication and consultation running in parallel with the above.

Dimitrakos in [208] and [212] sees risk management supporting both the analysis of trust inclinations leading to the formation of trust intentions, and the analysis of trust intentions leading to the endorsement of dependable behaviour. (See also Figure 53). We anticipate

[231] Information technology-Security techniques-Guidelines for the management of IT Security (GMITS) Part1: Concepts and models for IT Security. ISO/IEC TR13335-1:1996.

[232] AS/NZS 4360:1999. Risk Management. Australian/New Zealand Standard (1999).

[233] HB 4360:2000. Australian Standard (2000): Information security risk management guidelines. Strathfield: Standards Australia.

different kinds of risks to be analysed in these two phases. The focus in the former case is on analysing the effect that an agent's persistent tendencies and risks from the environment have on the formation of this agent's trust for a specific service. The focus in the latter case is on balancing intentions to trust against interaction risks in order to endorse an informed and dependable behaviour.

Overall, the concepts analysed in this section provide a vocabulary for describing how trust is contrived and how it affects exhibiting (or is affected by observing) dependable behaviour. Building on top of this analysis, Figure 54 to provide an illustration of the relationship between basic risk management concepts and trust primitives, which is consistent with the trust management scheme proposed in this section. Notably in our analysis, the level of trust depends on cost and contributes through its relation to utility to the endorsement of trusting intentions to bring about trusting behaviour. There is also a feedback loop between beliefs, intentions, behaviour and risk: trusting behaviour exposes to risk while changes in risk influence trusting intentions directly and indirectly through situational decisions to trust.

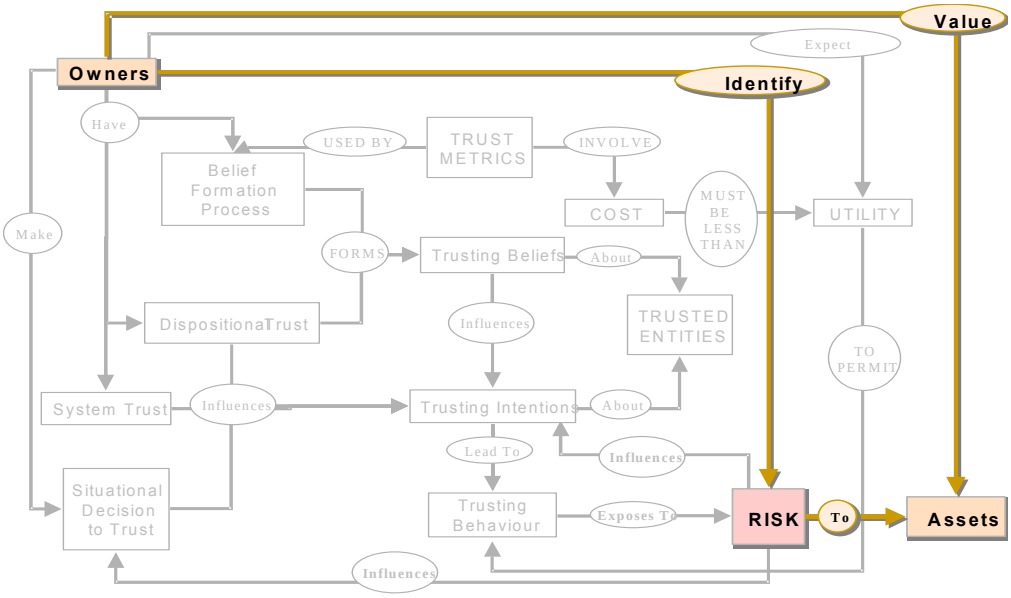


Figure 54 Dependencies between basic risk management concepts and trust primitives emphasising the role of asset.

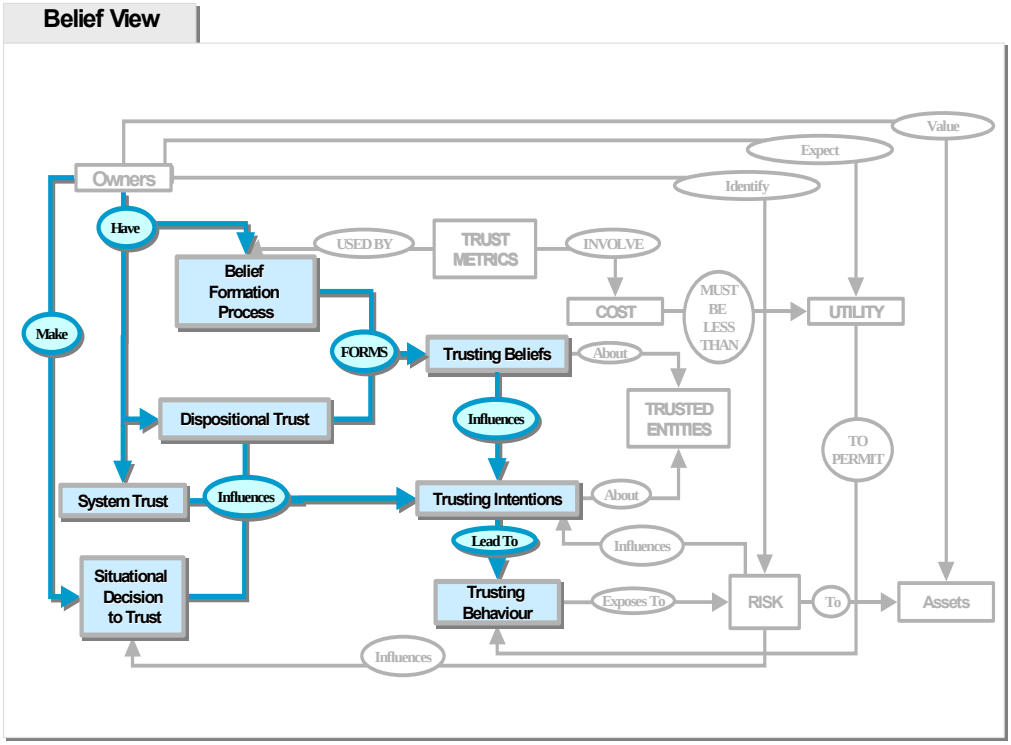


Figure 55 Dependencies between basic risk management concepts and trust primitives emphasising the role of belief formation.

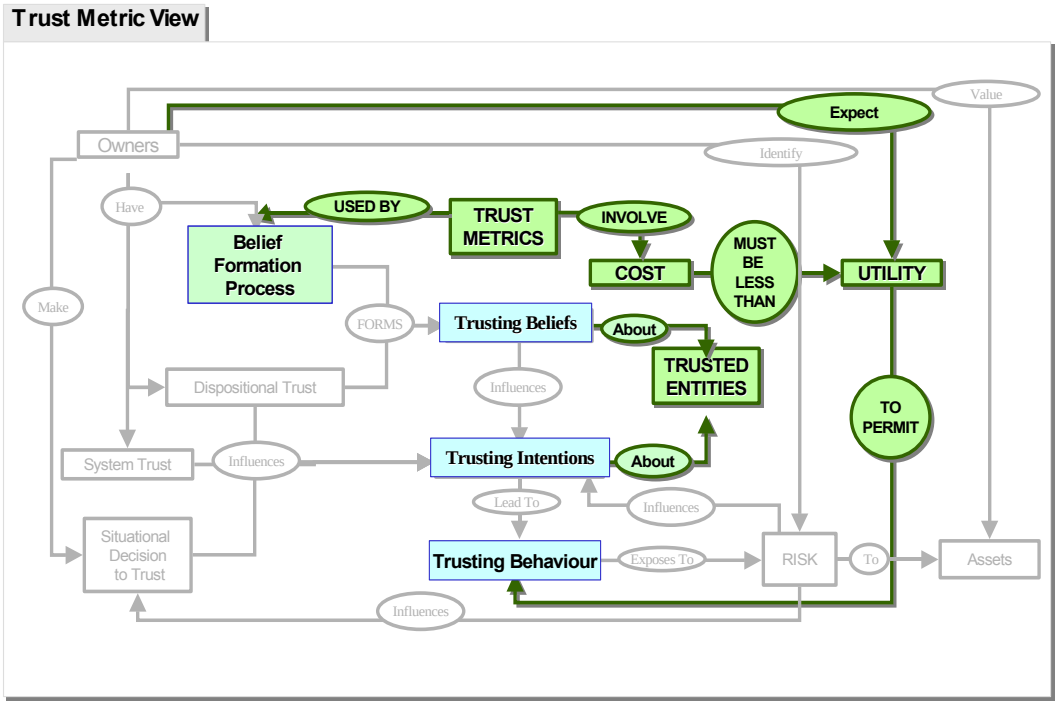


Figure 56 Dependencies between basic risk management concepts and trust primitives emphasising the role of trust metrics.

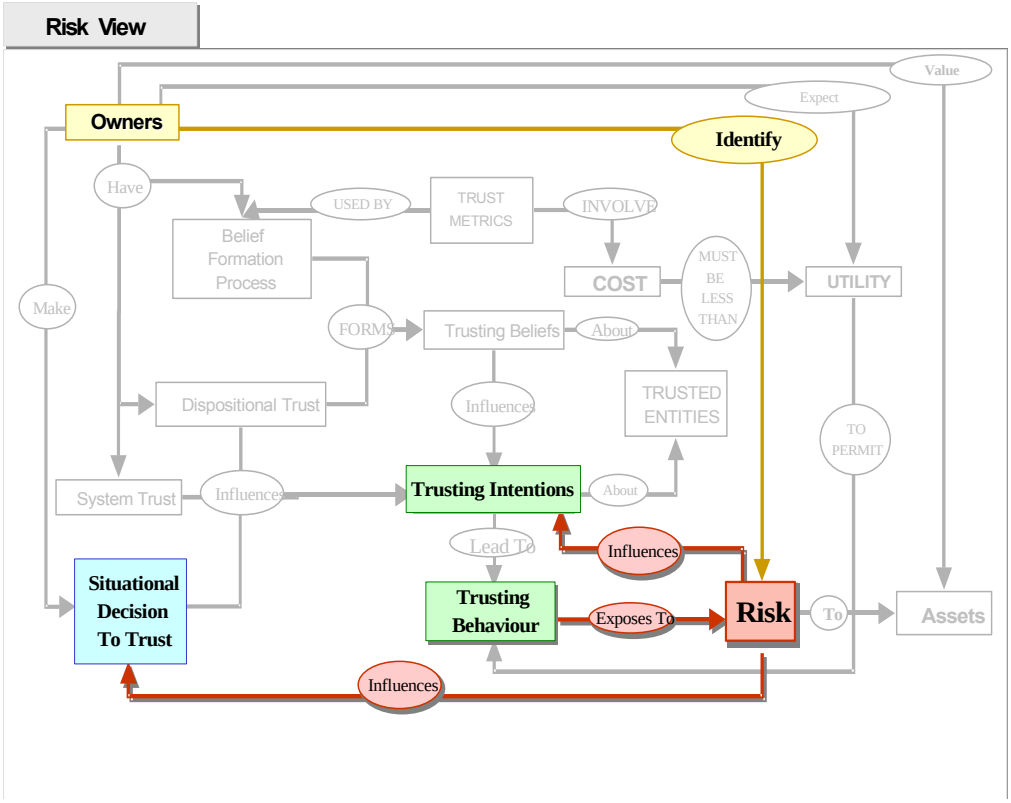


Figure 57 Dependencies between basic risk management concepts and trust primitives emphasising the feedback loop between risk, trusting beliefs, trusting intentions and trusting behaviour.

7.3 Trust Services

The notion of trust services is not new; there are various financial, insurance, and legal services available that make commercial activity simpler and less risky. In the e-commerce context, Trust Services are emerging as a business enabler, aiming to deliver trust and confidence at various stages of business interaction, including: establishing and maintaining trust, negotiations, contract formation, fulfilment, collaboration, through to dispute resolution[234].

Trust services attempt to solve problems such as: establishing the authenticity of electronic communications; ensuring that electronic signatures are fair and legally binding, creating an electronic audit trail that can be used for dispute resolution. It is not yet clear what the range of trust services will be. Reasonably, it can be expected that they include mechanisms to support trust establishment, negotiation, agreement and fulfilment, such as:

a) Identity services

b) Authorization services with support for the delegation and control of fine-grained access control at the data, resource and service levels.

c) Anonimity services

d) Trust rating and recommendation services

[234] A. Baldwin, Y. Beres, M. Casassa Mont, S. Shiu, - Trusted E-Services Laboratory - HP Laboratories Bristol. Trust Services: A Trust Infrastructure for E-Commerce. HPL-2001-198, Available at http://www.hpl.hp.com/techreports/2001/HPL-2001-198.pdf

e) Notarisation

f) Guaranteed message delivery

g) Auditable logs

h) Secure storage

Trust services should be built (and trust services providers could be evaluated) along the following guidelines:

- Accountability: service providers should assure that their processes will stand up to scrutiny in disputes; ideally they should assume liability for the service they offer.
- Survivability/Longevity: Each service and the industry as a whole must produce technology and businesses that will be available to resolve disputes decades after events.
- Confidentiality: trust services and the trust services providers must ensure confidentiality of customers' highly sensitive data even within their own organisation.
- Integrity: Because digital data is easily created and forged, trust service and service providers must be able to demonstrate the integrity of their information or the information they keep. This aspect is linked with accountability and longevity,
- Simplicity: trust services must be easy to use, but at the same time they must take account of existing infrastructure.

7.3.1 Reputation Systems and Services

Trust changes with time as a result of the subjects' experience and/or of reputation. Reputation can be broadly defined as the evaluation of experience. Ideally, subjects would evaluate an information source or a service and establish a rating themselves. But in reality, subjects may not have the expertise to perform such evaluation. The evaluation can be performed by an external trust authority (application domain expert), that can provide it as a service to clients.

Reputation system collects, distributes, and aggregates feedback about participants' past behavior.

In the Internet world, Reputation systems, known also as *Online Feedback Mechanisms*, use the Internet's bi-directional communication capabilities in order to artificially engineer large-scale word-of-mouth networks in which individuals share opinions and experiences on a wide range of topics, including companies, products, services, and even world events235. Reputation systems and mechanisms have been used as a technology for building trust in electronic markets.

Trust changes with time as a result of experience/reputation. Reputation can be defined as the evaluation of experience[236]. Reputation systems collect, distribute, and aggregate feedback about participants' past behaviour. To operate, reputation systems require three properties [236]:

1) Entities are long-lived**,** so that there is an expectation of future interaction.

2) Feedback about current interactions is captured and distributed. Such information must be visible in the future.

3) **Past feedback guides buyer decisions**: People must pay attention to reputations.

---

[235] C. Dellarocas. The Digitization of Word-of-Mouth: Promise and Challenges of Online Feedback Mechanisms. MIT Sloan School of Management Working Paper 4296-03 March 2003.

[236] Resnick P., Zeckhauser R., Friedman E., and Kuwabara K.. (2000) 'Reputation systems.' *Communications of the ACM*, 43(12).

There are a few existing online reputation systems such as the feedback system of eBay, Yahoo! Auction, and Auction Universe. Most of these systems use the single factor of feedbacks as the reputation measure, which often fails to capture the trustworthiness of users effectively. Yahoo! Auction, Amazon and other auction sites feature reputation systems like eBay's, with variations such as a rating scale from 1-5, or using several measures (friendliness, prompt response, quality product, etc), or averaging rather than totaling feedback scores.

A. Jøsang proposed the Beta Reputation System[237], a reputation engine based on the beta probability density function. In contrast to most other reputation systems which are intuitive and ad hoc, the beta reputation system has a firm basis in the theory of statistics. The beta reputation system can also be used in a distributed setting.

The construction of trustworthy Reputation systems requires to address several issues, such as:

- To what extent can their operators and participants manipulate them? This, in turn, raises the point of the assumptions underlying the "computation" of the reputation rate (such as the assumption that the feedback is given honestly and with no bias).
- How such systems can be compromised (for example by colluding participants), and the counter-measures to prevent such a situation.
- The context factors taken into account. This requires that the adopted trust metric be general enough to adapt to different communities under different transactional or community specific contexts[238].

7.3.1.1 Operational Models for Reputation Servers

There are two main axes for categorising reputation servers. The primary axis distinguishes between who performs the evaluation of an entity's reputation. The choice is between the actors who participate in transactions with the given entity or the reputation service itself. The second axis distinguishes between how the data is collected, summarized, evaluated and published. The reputation service can either gather and evaluate the data itself (the data pull mode) or the actors can spontaneously send data to the reputation service (the data push mode). When these two axes are combined together we get the 2x2 matrix shown in Table 9 below.

	Data Push	Data Pull
Actor evaluation	Voting model	Opinion Poll model
Reputation Server evaluation	MP model	Research model

Table 9 Operational Models

The Opinion Poll model

[237] A. Jøsang. The Beta Reputation System. 15th Bled Electronic Commerce Conference. Bled, Slovenia, June 17 - 19, 2002

[238] Li Xiong, Ling Liu. A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities. IEEE International Conference on E-Commerce, June 2003, Newport Beach, California

In this operational model, the reputation server actively collects reputation data from the actors. Each actor performs its own evaluation about the reputation of an entity, based on its experience of performing transactions with it. The role of the reputation server is merely to gather and summarise the data. In real life humans do this sort of evaluation all the time about the shops they frequent, the people they meet, the political parties they vote for etc. And opinion poll companies regularly collect this information and publish the results. The most difficult aspect of this model to implement for reputation servers, is to discover where the actors are and how to contact them. In the physical world, opinion poll companies use electoral roles to determine who the actors are, and telephone directories to find out how to contact them, or they simply stop people in the street as they are passing by. Engineering this in the virtual world is more difficult. The nearest thing we currently have to the electoral role is the DNS. This lists all services on the Internet, along with their IP addresses. The DNS is core to the functioning of the Internet. However, the DNS does not as yet hold data about actors who can provide reputation information to opinion poll servers. Two possibilities exist for this. One would be to define a new type of resource record (RR), say the reputation (RP) record, that holds the name and reputation of the entity being reputed. Actors would then write this DNS record into their DNS entry for each entity they were assigning a reputation to. Opinion poll servers could then scan the DNS to sample these records are derive collective reputations for entities. However this scheme is not likely to be implemented for a number of reasons. Firstly the DNS is already heavily over-utilised and performs poorly. The IETF tries to keep tight control over it, and therefore is highly unlikely to sanction the definition of this new RR type. A more fruitful approach would be to define a new protocol for the gathering of reputation information from actors, say the Reputation Gathering Protocol (RGP), and then to register this protocol with IANA and get a well known port allocated to it. Once this is achieved, actors can simply register their RPG servers in the DNS, using the existing WKX RR, and this will allow opinion poll servers to contact them. However, this does not quite solve the problem, since the DNS does not provide a search capability. DNS clients have to already know the DNS name of the entity they want to look up, before contacting the DNS to get its IP address. In the general case opinion poll servers wont know the DNS names of the actors they want to poll. Thus we need a search and discovery service that will allow opinion poll servers to search for all actors, or a subset of actors that meet pre-defined search criteria such as: size of business, no of transactions undertaken, currency of the data etc. This implies that we need to define schema for either UDDI or LDAP servers (or both) to enable this searching to take place.

The Voting Model

In this model, the actors evaluate the reputation of entities, and then forward their decisions to a central Voting server. The role of the voting server is simply to collect messages that arrive, collate summarise them, and publish the results. E-bay is one example of this type of reputation server in use today. Various shopping mall web sites also allow customers to register their votes about how well the stores in the mall are performing their various aspects of service provision, for example, timeliness of goods delivery, and quality of after sales service etc. This operational model is much easier to implement, since the Voting server does not need to have access to a discovery service, unlike the opinion poll model. Voting servers keep their own lists of actors, and only allow these actors to register votes with them. Very often these lists will be commercially sensitive, and, if they contain personal data, will be protected by a data protection act. Thus the voting servers are highly unlikely to make these lists public, or available to opinion poll servers.

The Research Model

In this model, the reputation server actively searches for information, then evaluates it and publishes the results. The role of the reputation server is a complex one and difficult to engineer. Not only does it have the problem of finding the raw information, as in the Opinion Poll model, but also it has to determine how to process and evaluate the raw data in order to compute the reputations. Several example of this model exist in real life, for example Standard and Poor provide financial ratings, and Dunn and Bradstreet provide credit ratings. Clearly this is a successful business model – if the results are valuable they can be sold. But

for this reason the algorithms that are used to provide the computed data may be commercially sensitive and not open to public scrutiny.

The MP model

MP stands for Member of Parliament, a person elected to the UK government to represent a constituency. MPs should represent their constituency, but often they do not. When it comes to voting on issues in the House of Commons, they either usually follow the party line, or if a free vote is allowed, on such issues as capital punishment or hunting with dogs, they follow their own conscience. So even if constituents have sent them lots of letters imploring them to vote one way, they may quite freely decide to vote the opposite way.

A reputation server following the MP model, will be sent (pushed) lots of data by the actors. Some of it may be raw data about transactions the actor has undertaken with various entities, other might be entity reputations evaluated by the actors themselves. Regardless of this, the MP server determines which information to use, which information to discard, and which other private information to use as well. Then using its own, usually unpublished, algorithms, it computes the reputation of entities and either sells of publishes the results.

7.3.1.2 Evaluating Reputation Systems

One approach taken to evaluating, in general, the effectiveness of using a reputation system as a feedback mechanism between strangers in an online environment, is outlined in[239]. Here online markets utilising reputation mechanisms to establish trust between actors are compared to similar markets in which no reputation mechanism exists, and to markets which rely on the continual interaction of the same actors to establish trust. The conclusions here establish that the use of reputation mechanisms improve, in general, the efficiency of the markets which utilise them. This approach can be used to establish the validity of having a reputation system in place rather than having no such system at all, but in order to evaluate the pros and cons of specific systems a more detailed and specific approach is required.

7.3.1.2.1 Underlying Assumptions

Most reputation services in existence at the moment rely on the assumption that the feedback given to the system in order to determine reputation is given honestly[240], in other words that the users of these systems, responsible for determining the reputation of others, use the system in the manner it was intended and do not abuse the rights they have to bias the system in any way. This assumption, while generally holding true in most reputation systems, when it does not hold it can seriously undermine the usefulness of such systems [240].

Despite this presumption of altruism being an underlying assumption behind the development of reputation systems, it has still been argued that this assumption is valid. Howison[241] argues that work done in the field of biology regarding demonstrating that indirect reciprocity can act, as a valid evolutionary strategy is a close analogy to the assumption of altruistic behaviour in reputation systems. The assumption that users will act altruistically rather than selfishly the majority of the time also has backing from studies carried out to examine the reasons behind users feedback in reputation systems such as eBay[242]. Even so, attempts have been made to ensure that honest feedback is given. For

---

[239] Bolton, G. E., Katok, E. anf Ockenfels, A. (2003) 'How Effective are Electronic Reputation Mechanisms? An Experimental Investigation' Working Paper Series in Economics, 3, University of Cologne, Department of Economics (available at http://ideas.repec.org/p/kls/series/0003.html)

[240] Dellarocas, Chrysanthos N., (2001) 'Analyzing the Economic Efficiency of eBay-like Online Reputation Reporting Mechanisms' . MIT Sloan Working Paper No. 4181-01. http://ssrn.com/abstract=289968

[241] Howison, J. (2003) 'An Introduction to the Literature on Online Reputation Systems for the MMAPPS Project' at http://wirelessgrids.net/RepLitIntro/

[242] Dellarocas, C., Fan, M. and Wood, C. A. (2003) 'Self-interest, reciprocity and participation in online reputation systems' WISE 2003: Workshop on Information Systems and Economics December 13 - 14, 2003 in Seattle, Washington, USA

example Miller et al.[243] present a system whereby scoring rules are applied to a given reputation score, adapting the score to reflect how well it predicts future reputation scores assigned to the actor being rated.

Online reputation systems, such as eBay, generally take a 'passive' approach to the use of reputation for determining the effect it has on users of the system[244], in that the system itself does not act on the reputation of any given actor in the system, rather the reputation 'score' of any actor is acted on by other actors who are to consider dealing with them. This 'passive' approach also introduces another assumption, that, even with an accurate measure of reputation, the users of a system will act on this information in the desired manner. Dellarocas [244] has shown this to be the case that sellers with high reputations usually obtain more for the goods they sell.

Mui et al.[245] has taken a different view in determining the reputation of actors within a system. The model of trust and reputation put forward by them is based on a statistical function for computing reputation based on the history of a user's actions between all actors rather than the standard feedback mechanism of recommendations and complaints. This overcomes the problem of assuming that users will provideg feedback information honestly, however it only does this by completely disregarding the input of other users, who, in a lot of cases, will be the most valuable and knowledgeable source of reputation information.

7.3.1.2.2 Evaluating the Quality of Feedback

As has been noted, most reputation systems use a single factor feedback for determining reputation, with no other mechanism for determining the accuracy or value of these recommendations. One interesting example of a system in which reputation is used to determine a user's ability to perform actions within the system, but which also takes into account the quality of the feedback from other users, is the Slashdot community website[246]. Slashdot is popular website dealing with issues surrounding open source software and other, more general, technical news items, with a large and vocal membership. The website allows users to post comments to the site in relation to the news stories it links to. The volume of posts relating to each story tended to be so large that a moderation system was put in place which is used to give a higher score to relevant or interesting posts, and a lower score to 'off topic' or redundant posts. The score of the messages can then be used by users of the website to filter the number of posts that they see down to a reasonable and manageable number. The moderation of posts within the system is carried out by normal members of the community, rather than a select group of elite moderators. The ability to moderate posts, rather than being a continual right of a user, is given out to users for a limited amount of time at intervals, and is determined by an algorithm which takes into account the users use of the system (with 'average' users, that is, those users whose use of the system in terms of time spent logged into the system and the number of posts they read, *et cetera*) and the user's 'karma' (essentially a measure of reputation). This karma score is determined by the ratings given to a user's posts by other moderators and by the meta-moderation carried out on their own moderations (described below).

Where the Slashdot system differs from the majority of other reputation style systems, is in its system of 'meta-moderation'. Here all users have the ability to view a series of posts which have been moderated by other users, and allows them to rate the moderation given to those posts as either 'fair' or 'unfair'. Users whose own moderations are continually rated as fair are more likely to be granted the right to moderate, and inversely users whose

---

[243] Miller, N., Resnick, P. and Zeckhauser, R (2002) 'Eliciting Honest Feedback in Electronic Markets' . KSG Working Paper Series RWP02-039. http://ssrn.com/abstract=348940

[244] Dellarocas, C. (2003) 'Efficiency through Feedback-contingent Fees and Rewards in Auction Marketplaces with Adverse Selection and Moral Hazard' in Proceedings of the 4th ACM Conference on Electronic Commerce, San Diego CA

[245] Mui, L., Halberstadt, A. and Mohtashemi, M. (2001) 'A Computational Model of Trust and Reputation' in Proceedings of the 35th Hawaii International Conference on System Sciences.

[246] Http://slashdot.org

moderations are continuously rated as 'unfair' are less likely to be granted the opportunity to moderate. This is an attempt to overcome the problem, prevalent in the majority of reputation systems, in which the quality of reputation feedback is presumed (not necessarily correctly) to always be fair. One criticism that could be levelled at Slashdot is that the more highly rated postings tend towards the views of the majority, and therefore don't allow dissenting views to be adequately heard (a point often raised, and subsequently moderated down, on the site itself!). An overview of the Slashdot moderation system can be found at (http://slashdot.org/faq/com-mod.shtml#cm520).

7.3.1.2.3 Generalizing the Context of Reputation Systems and Taking into Account Temporal Measures

While numerous reputation-based systems are in existence, most are still designed to cater to a select field of applications or are designed specifically to perform a single task. In order for a reputation system to generalize to different communities or areas of application the trust metric used to measure trust and reputation must be able to adapt to different communities using differing transactional or community specific contexts[247]. The model of trust and reputation evaluation presented in [247] attempts to do this by building a model which, as well as taking into account the feedback of peers and the quantity of this feedback also provides a mechanism for stipulating adaptive methods of dealing with transaction context and community context. The context specific factors allow the system to be tailored to the needs of the specific context in which it is being used. An example of this would be the ability of the system to factor in a weighting to an assigned reputation to take into account the size of a business transaction on which the reputation was being based, which would serve to prevent users building up reputation with small transactions only to betray that reputation for a larger transaction. This system also provides the ability to specify a temporal context to reputation measurements. So that that reputation information in the past can be degraded in weight over time in order to adapt to changing circumstances.

7.3.1.2.4 Portability of feedback data between different feedback systems

Resnick et al. [236] highlights the fact that whilst many different reputation feedback style systems are in place, the systems they use for determining reputation are often proprietary, with no mechanism in place for sharing reputation information with other systems and even in cases where services have allowed users to import reputation information from other services (such as was the case with Amazon allowing users to import eBay reputation information) this ability to import has been stopped by companies claims that their reputation systems and information are owned by them and cannot be used by other systems. It is also noted that attempts to create generalised online systems which would potentially act as a reputation service to be used across domains have failed to be taken up, which appears to still be true four years after the publication of this paper.

As for reputation systems for domains and user groups more focused than publicly accessible websites, the ability to produce generalised forms of reputation system with adequate adaptability to generalise across contexts has been demonstrated[247,245,248], although no one system has emerged as being widely used enough to be able to share reputation information on a practical rather than a theoretical basis.

7.3.1.3 E-cognos

The following section outlines the e-cognos knowledge management system and, specifically, its system of representing knowledge in terms of knowledge representations and

[247] Li Xiong, Ling Liu. (2003) 'A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities'. IEEE International Conference on E-Commerce, June 2003, Newport Beach, California

[248] Schlosser, M., Sintek, M. and Garcia-Molina, H. (2003)'The EigenTrust algorithm for Reputation Management in P2P networks, in WWW 2003.

knowledge representation links. A description of the system is given. Further, the application of this work in relation to the TrustCoM project is discussed.

7.3.1.3.1 Overview

The e-cognos project produced a Knowledge Management infrastructure tailored to the needs of the construction industry[249]. This infrastructure was based on the following design.

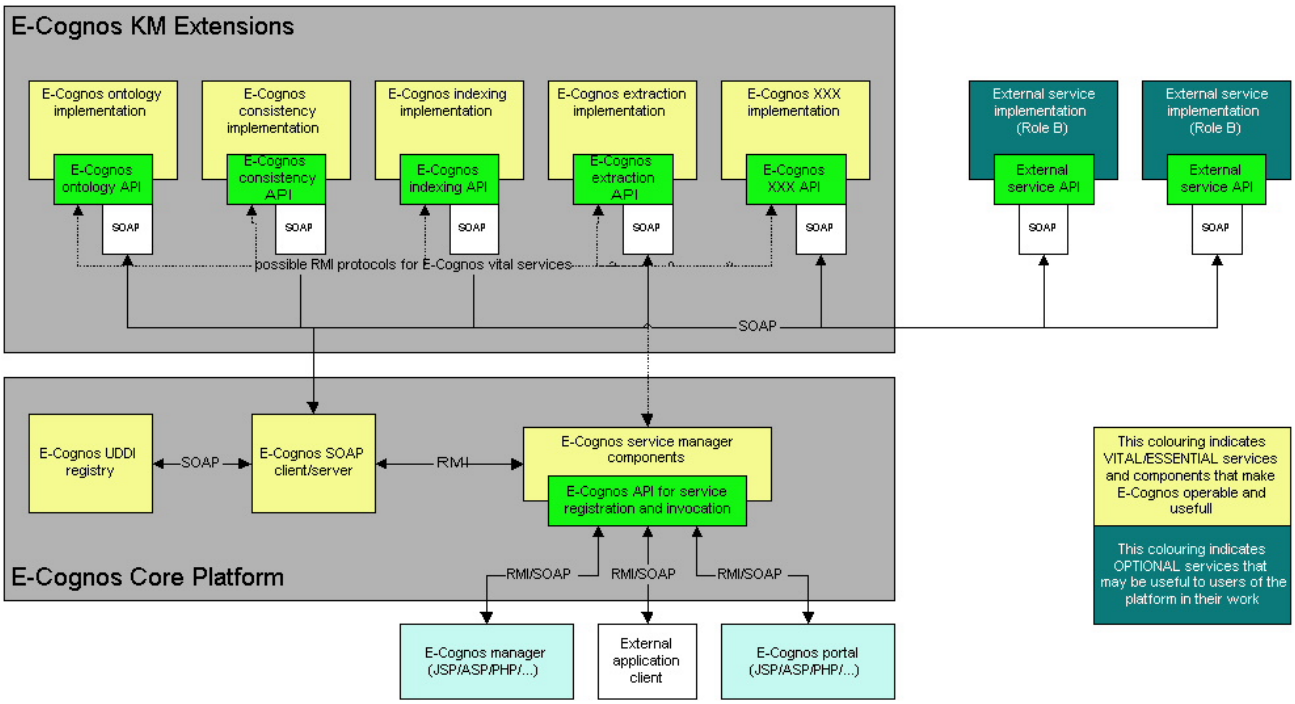


Figure 58 E-Cognos Infrastructure

The infrastructure was designed around the concept of a core platform, which handled users access to a set of services both built into the system (the E-Cognos KM Extensions) and external to it. These services dealt with the manipulation of 'knowledge' within the system. This knowledge was held in the form of 'Knowledge Representations', structures within the core of the platform which hold information about entities dealt with by the system, such as Actors, Organizations, Documents etc. Although the system was designed to be specific to the needs of the Construction industry this specialization came in the form of an Ontology service which was used to index and reference industry specific information. In effect this meant that the core of the system acted as a generalized knowledge management kernel, the structure of which is described below.

7.3.1.3.2 The e-cognos Knowledge Management Kernel

Knowledge Representations

The concept of a Knowledge Representation is key to the core of the e-cognos kernel. All information held within the system is held in the form of a Knowledge Representation. A Knowledge Representation is in effect a meta-level description of an item of Knowledge that exists outside the system, which the system in some sense has access to. For example, in the case of a document, the Knowledge Representation would store information about the

[249] Wetherill, M., Rezgui, Y., Lima, C. and Zarli, A. (2002) 'Knowledge Management for the Construction Industry: the e-cognos Project' ITcon (7:183)

location of the document (e.g. a URI) the documents format, information such as the Author, creation date, modification date etc and a set of keywords and a summary representing a concise description of the contents of the document. In the case of a Knowledge Representation representing an Actor, the information would include name, contact details, company information etc. Knowledge Representations are generalized in such a way that any potential piece of Knowledge outside the system can be encapsulated in the form of a Knowledge Representation. The definition of a Knowledge Representation is given in the form of an XML schema describing the information held in a given type of Knowledge Representation.

One piece of information common to all Knowledge Representations is set of keywords and weights. This tuple is used by the system to index Knowledge Representations and serves to capture the essence of what each piece of Knowledge actually represents.

Knowledge Representations on their own serve to capture information about individual 'pieces' of Knowledge, however, the key to a Knowledge Management system is it's ability to represent and uncover relationships between such 'pieces' of Knowledge. The e-cognos Knowledge Management kernel performed this function using the concept of Knowledge Representation links, described below.

Knowledge Representation Links

As well as capturing the essence of individual items of knowledge, as was the purpose of the Knowledge Representation object, the system also aimed to capture the essence of the relationships between items of knowledge, in a similarly flexible way. In order to do this, the system also defines a Knowledge Representation Link object. A Knowledge Representation link defines a directed relationship from one Knowledge Representation to another. The semantics of the link are defines by the links attributes, for example a link could be defined which represents a simple is-a relationship between two items, as in an ontology, or a more complex relationship, for example the degree of relevance (measured numerically) that one item has to another. These attributes, as with those of Knowledge Representations, are specified in terms of an XML schema. Built into the system are algorithmic hooks in the form of an API for defining operations over links and Knowledge Representations.

The way in which this model was tailored towards the specifics of the e-cognos project included links being defined which represented the right of a Knowledge Representation (in practical terms, one representing an actor within the system) to perform an action on another Knowledge Representation (for example the right of a user to delete a document from the system). Another type of link in the e-cognos system represented an actors interest in other Knowledge Representations. This allowed the system to uncover new items of knowledge, of potential interest to a user, by examining the network of interests that existed between Knowledge Representations and inferring potential interests from this.

7.3.1.3.3 Application to TrustCoM

In terms of applying this model to the kind of trust infrastructure potentially required by the TrustCoM project the following points can be noted. Trust can modelled as weighted relationships (i.e. the amount of trust each individual actor in a system holds towards the others, as a numerical value) between objects in the system (i.e. Knowledge Representations) representing actors. Actions can then be performed on these objects to determine features of trust relationships and to manipulate trust between actors in various ways. It is also important to note that in this model Knowledge Representations can represent actors of any type, for example the trust that a service holds in a user, which would enable the system to decide whether a user can access a given service.

As an example of the implementation of such a system take the following. A system is created whereby users of this system have access to various services. Whether or not the user can use each of these services is dependent of the reputation that the user has within the system. Each service, depending on its nature has a trust threshold i.e. the potential risk that misuse of the service poses, or the time it takes to use the service. If the service 'trusts' a user, in terms of the user's reputation, more than the threshold, then the user has access

to that service. Using the e-cognos model, this could be implemented in the following way. Users are modelled as Knowledge Representations, with the user KR holding an ID and security information etc for each user. The services in the system are also represented as Knowledge Representations, with the KR for these holding, amongst other things, the trust threshold for the use of the service. Now, in terms of the links within the system, a link is defined which holds a trust value between KR's. This trust value reflects trust between users (when the link is between two user KR's) and the degree to which a service trusts a user (in the case of a link from a service KR to a user KR). The system then works as follows: actors using the system can rate each other in terms of trust, this changes the values held in the trust links between them. The services calculate the trust they have in a user by aggregating the trust other users have in a given user and using this value as the trust that the service holds in a user. This value would then determine the ability of a user to operate a service. Furthermore transitive trust can be represented by building a new link between a user, and the link between two other users. This new link would then represent the trust that a user has in the trust between two other users.

Using the existing code base, which has been published under an open source license, to build a reputation service relevant to TrustCoM presents the opportunity of short cutting some of the development work required in implementing the underlying foundations of a trust or reputation srvice.

7.3.1.4 Trustworthiness of the Reputation Servers

Clearly if one is relying on the data provided by a reputation server, one needs to ask how reliable or trustworthy is the information that the reputation server is providing. The Opinion Poll model is inherently the most trustworthy and reliable, since the raw reputation data has been calculated by many actors, thus it is very difficult to skew the results. Also the raw reputation data is, in principle, available to be collected by any opinion poll server. Thus the resulting computed reputations are more easily validated and the calculations more easily repeatable by any opinion poll server (or any actor for that matter). Thus the results do not rely on the reputation of the reputation servers themselves.

The Voting model should provide the next most trustworthy set of results. The raw data has similarly been determined by many actors, and therefore it should be more difficult to skew the results. However, because the list of actors is not public, it is not possible to validate the composite result. The reputation server could skew the results by discarding votes it did not like, or indeed by inserting false votes to increase or decrease an entity’s reputation. We are all familiar with this type of activity in real life!

The Research model provides the next lower level of trustworthy results. Because the results are difficult to arrive at, it is very difficult for actors to reproduce the results without a large investment of capital and time. Therefore one has to trust the reputation of the service that is providing the results. Thus it will typically take time for this reputation to be established, but once established, it will become a great asset of the reputation server itself. We might therefore expect it will be many years before these types of reputation servers will become a common feature of the Internet.

The MP model is inherently the least trustworthy and reliable of them all. The more trustworthy MP servers will be open to public scrutiny and will publish their algorithms and raw data (within the limits of the data protection act). But in general there is no requirement to do this, and therefore the trustworthiness of MP servers will at best be variable.

7.3.2 Notarisation Service

Notarisation is a trust service that provides evidence of the existence of documents and messages at particular points in time. Notarisation services emerging on the Internet provide notarisation records of digital documents including undeniable timestamps of the content of these documents. A proof of the notarisation act is usually returned to the owner of the digital document by means of a receipt.

Examples of such services are:

- Surety (www.surety.com), which is a provider of digital record notarisation service. Surety's Digital Notary Service enables enterprises and people to notarise electronic files and records before they are distributed or publicised, guaranteeing file content and enabling the owner to verify their content for years to come.
- Timestamp.com (www.timestamp.com), which provides a time-stamping service to digitally timestamp digital documents. The hash value of a document is digitally signed and time stamped and the result returned to the requester.
- Financed by the European community, OpenEvidence - part of European Project Group FP5 - is an open source framework for data certification, time stamping and data archival that aims to provide technology for evidence creation, validation and long term protection of documents, building an architecture that can be applied to different business models like notarisation. Based on standards ISO 17799, British Standard 7799, IETF PKI RFC 3161 and IETF PKI RFC 3029[250].

7.4 Trust Related Aspects of the Security Infrastructure

Existing security infrastructure such as Public Key Infrastructure will certainly play a role in supporting the VO life cycle. Trust management as defined at the beginning of this chapter, that is the management of belief taking into account the subjective dimension of trust, neither was not nor is in the scope of PKI. However, several trust-related aspects of the PKI infrastructure may influence the possibility of using it to support the VO lyfe-cicle, exclusively or in combination with a trust management system and/or the variety of other emerging trust services. This section discusses the following topics:

- Trust –related aspects of X.509 and PKI infrastructure
- Proposed extensions to PKI, particularly those that are intended to increase the trust of a relying party on PKI certificate
- Hybrid PKI models
- SPKI certificates

7.4.1 X509 and PKI aspects of Trust

The Public Key Infrastructure (PKI) can be defined as "…the set of hardware, software, people, policies and procedures needed to create, manage, store, distribute and revoke public-key certificates based on public-key cryptography"[251].

[250] *http://www.com-and.com/openevidence.html*

[251] Arsenault, A, & Turner, S. Internet X.509 Public Key Infrastructure: Roadmap. IETF, internet Draft. July 2002.

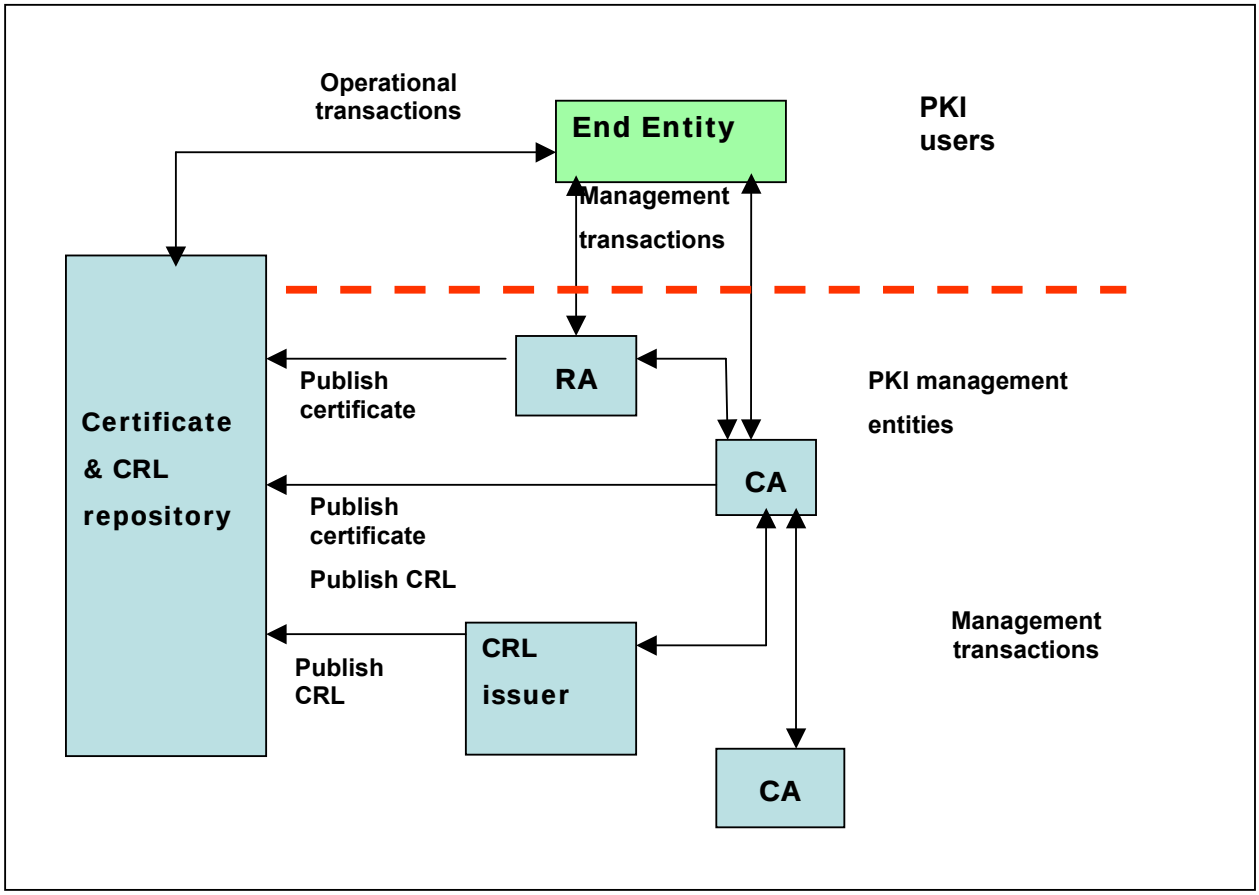


Figure 59 PKI components[252]

The components of a PKI infrastructure, depicted in above, are the following:

- Certification authority (CA): it is the authority trusted by users. Its main duties are:
  - Issue certificates;
  - Manage certificates (suspend, renew, publish, revoke);
  - Generate and publish certificates status information;
  - Keep safe its own private key.
- Registration authority (RA): it is the entity responsible for the registration of users requiring a certificate. It verifies the identity of the certificate applicants and the validity of their certificate requests. It is an optional component of PKI, since its duties can be carried out by the CA.
- Certificate (and Certificate Revocation List) Repository: it is the entity managing logical storage for certificates and other information such as CRL made available by the CA
- Certificate Revocation List (CRL) Issuer: an optional system to which a CA delegates the publication of certificate revocation lists
- Relying party (not represented in Figure 1 above): it is an entity that has to make some decision based on a certificate content.

[252] Figure taken from R. Housley, W. Polk, W. Ford, D. Solo. RFC 3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. April 2002

- End Entity (EE): he/she/it is the holder of the private key corresponding to the public certified one, and hence the subject of the certificate. It can be a certificate user or a Certification Authority.

CCITT Recommendation X.509[253] defined a public key-based "Authentication Framework" in which the Directory Service provided key management facilities for open applications. Certificates play a central role in this authentication framework. A public key certificate is a data structure used to securely bind a public key to others attributes. These attributes can consist of identification information, e.g., a name, or an authorization information, e.g., permission to use a resource. X.509 established a standard syntax for identification certificates which bind a public key to a directory name and identify the entity who vouches for the binding. X.509 certificates are signed with the private key of a trusted entity called a Certification Authority (CA). X.509 also defines a format for revocation lists, which are stored under the CA entry in the Directory.

A certificate is validated by verifying the signature applied by the issuer of the certificate. "The use of certificates transforms the problem of acquiring the public key associated with a user into the problem of acquiring the public key of the issuer of the user's certificate. The issuer will also be represented by a certificate, and thus this process is recursive . It implicitly defines a certification graph. A certification path logically forms an unbroken chain of trusted points between two users wishing to authenticate each other. The objective of the certification path verification process is to enable one entity (the verifier) to determine the public key and security attributes of another entity (the claimant) for later use by various security mechanisms"[254].

The original X.509 framework did not specify a trust model but only suggested a common and hierarchical CA distribution, isomorphic to the Directory Information Tree hierarchy. In this model, every CA is trusted to sign any certificate. Such a model does not accurately represent users' beliefs, as some CAs will be regarded as more trustworthy than others. Moreover, it is not realistic to envisage that all CAs in operation will have the same assurance level and degree of responsibility.

Moreover, the original X.509 framework did not support a clear definition of the role, distribution and jurisdiction of certificates and issuers in the name and key spaces. The specification of a CA jurisdiction could allow to determine if a given entity is trusted to issue certificates, to whom and under which conditions.

The original X.509 certificate format did not allow:

- The identification (and description) of CA certification policy: a CA should adopt a security policy to ensure that a certificate will only be issued if the corresponding private key is known only to the identified user, and to ensure that the identity/properties asserted by this user is one that the user has a right to assert.
- To specify the scope of certificates: it should be indicated to users of a certificate what its subject is authorized to sign with the corresponding private key as well as what level of trust in the subject's entity is implied by a certificate.

7.4.1.1 Attribute certificates (AC)

X.509 certificates were conceived as Identity certificates, even though they could contain attribute information that can be used for authorization purposes. Attribute certificates255

[253] "Information Technology - Open Systems Interconnection - The Directory -Authentication Framework" ISO-IEC STANDARD 9594:1990-8 | CCITT X.509 (Blue Book Series), 1989

[254][S. Mendes and C. Huitema. A new approach to the X.509 framework: Allowing a global authentication infrastructure without a global trust model. In proceedings of 1995 Internet Society Symposium on Network and Distributed Systems Security, February 1995.

[255] ITU-T RECOMMENDATION X.509 | ISO/IEC 9594-8: - INFORMATION TECHNOLOGY - OPEN SYSTEMS INTERCONNECTION - THE DIRECTORY PUBLIC-KEY AND ATTRIBUTE CERTIFICATE FRAMEWORKS. May 2001.

are an extension of Identity certificates, specifically introduced for the management of authorizations. An attribute certificate (AC) is a structure similar to an Identity certificate; the main difference is that the AC contains no public key. An AC may contain attributes that specify group membership, role, security clearance, or other authorization information associated with the AC owner. A profile256 for the use of X.509 Attribute Certificates in Internet Protocols was also defined.

7.4.1.2 X509 V3 and IETF PKIX extensions

The X.509 Version 3 certificate defines a general certificate extension mechanism and a core set of extensions that allow authorizations and security policies to be bound into a certificate as well as authentication. The X.509 V3 certificate format also allows communities to define private extensions to carry information unique to those communities. The extensions defined for X.509 V3 certificates provide methods for associating additional attributes with users or public keys and for managing the certification hierarchy.

X.509 certificate extensions can be grouped in four main categories:

1) Certificate subject and certificate issuer attributes;
2) Key and policy information;
3) Certificate path constraints;
4) CRL distribution points.

The *certificate subject and certificate issuer attributes* extensions support alternative names in order to identify the subject or the issuer in a way consistent with the application that requires the certificate. These extensions can convey also additional information about the certificate subject; this may increase the relying partys' confidence about the fact that the certificate subject is a specific person or entity.

The *subject and issuer alternative name* extensions allow to bind one or more unique names to the subject of the certificate.

The *key information* extensions convey additional information to identify a specific key or to restrict key usage to specific operations.

The *authority key identifier* extension allows an end entity to identify a particular key used to sign a certificate. This extension could be used for example by a CA that uses two key pairs (one for low and one for high assurance operations).

The *key usage* extension identifies the range of applications for which a certain public key can be used.

The *policy information* extensions provide additional information about the policy used in certificate creation and management. A certificate policy is a (named) set of rules that indicates the applicability of a certificate to a particular class of communities/applications with common security requirements. This extension can be used by a relying party with specific certificate policy requirements to verify if the policies specified in the certificate is one of the relying party accepted policies.

The *certification path* constraints extensions allow to include constraints in a CA certificate to limit its certification power. The possible types of constraints are:

- Basic constraints, that tell whether an entity is a CA or not
- Name constraints that restrict the domain of trustworthy names that can be placed by the CA inside the certificates.

[256] S. Farrell, R. Housley. RFC 3281 An Internet Attribute Certificate Profile for Authorization. April 2002

- Policy constraints that restrict the set of acceptable policies that can be adopted by a CA in its operations

The *CRL distribution point* extension identifies the point of distribution for the CRL that can be used in determining the validity of a given certificate.

Since extension mechanisms provided by X.509 V3 certificates may cause interoperability problems, the IETF Internet Public Key Infrastructure PKIX working group has produced several informational and standards track documents, mainly targeted to the generic Internet usage. The PKIX working group have started to define *certificate profiles*[257] of the proposed extensions, to be used by specific applications or environments.

The PKIX group proposed also extensions intended to ease the evaluation by a RP of the policy adopted by the CA. In more details, the Attribute Certificates Policies[258] extension allows to explicitly state the Attribute Certificate (AC) policies that apply to a given Attribute Certificate. The goal is to allow relying parties to perform an additional test when validating an AC, i.e. to assess whether a given AC carrying some attributes can be accepted on the basis of references to one or more specific AC policies.

The Internet X.509 Public Key Infrastructure: Certificate Policy and Certification Practices Framework (RFC3647)[259] proposes "…a framework to assist the writers of certificate policies or certification practice statements for participants within public key infrastructures, such as certification authorities, policy authorities, and communities of interest that wish to rely on certificates. In particular, the framework provides a comprehensive list of topics that potentially (at the writer's discretion) need to be covered in a certificate policy or a certification practice statement."[259]

7.4.1.3 Evaluation of policy extensions

Although Certificate Policies extensions and Certification Practice Statements can contribute to enhance the RP trust on a CA, the main problem, from a business/legal point of view, is that CAs provide only a limited assurance, that is, they do not assume financial liability in the event that the assurance that the sender was not who the sender claimed to be, and that a party's reasonable dependence on that assurance resulted in economic cost.

7.4.1.4 CA interconnection and CA cross-certification

Cross-Certification extends third-party trust relationships between Certification Authority domains. For example, two trading partners, each with their own CA, may want to validate certificates issued by the other partner's CA. Cross-certification allows different CA domains to establish and maintain trustworthy electronic relationships. Different PKI interconnection topologies, also known as PKI trust models, were proposed[260].

Hierarchical PKI

In this model, represented in Figure 60, below, the root CA (also known as Top Level CA - TLCA) issues certificates to its immediate descendants, which in turn certify their descendants.

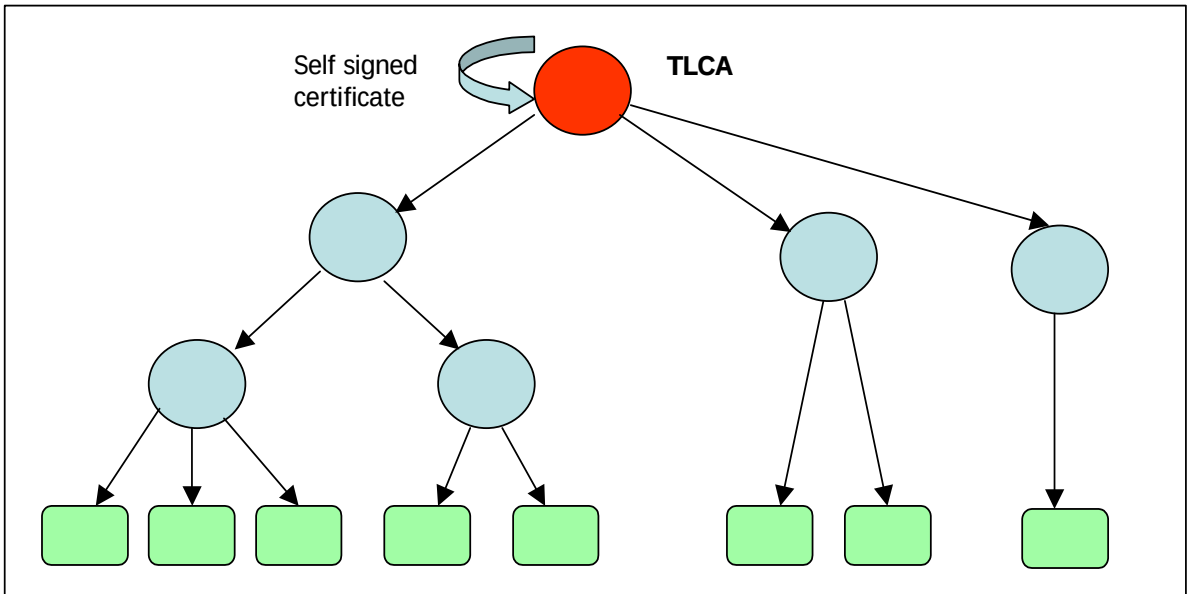


Figure 60 Hierarchical PKI

Each CA between the TLCA and the EE is known as intermediate CA. Certificates are issued in only one direction, from parent to child. Only the TLCA has a self-signed certificate. Thus, all EE certificates can be verified, except that of TLCA. Therefore, the TLCA certificate must be distributed to EE in a secure way; if someone succeeds in substituting the TLCA certificate maintained by an EE, he can get the EE to trust a completely different infrastructure.

Most PKI products and commercial CA service providers adopt the hierarchical model.

Integrating an existing foreign CA into a hierarchical PKI requires the EE of the foreign CA to trust the Root TLCA of the hierarchical PKI: this could be difficult to achieve in a peer-to-peer environment (like in business relationship).

This model has the following advantages:

- Scalability (for a specific community in the hierarchy)
- Ease of administration (for a specific community in the hierarchy)
- Simple construction of the certification paths between any two entities

Its main disadvantage is due to the fact that there is a single point of failure: if the private key of TLCA is compromised, then so is all the PKI.

This model is applicable in isolated hierarchical organizations, whereas it is difficult to apply to across organizational boundaries.

Trust List

In this model, represented in Figure 3 below, each Relying Party (RP) must maintain a list of trusted TLCA. Thus, interoperability is achieved at the application level of the RP. This model is the most successful one on the commercial ground.

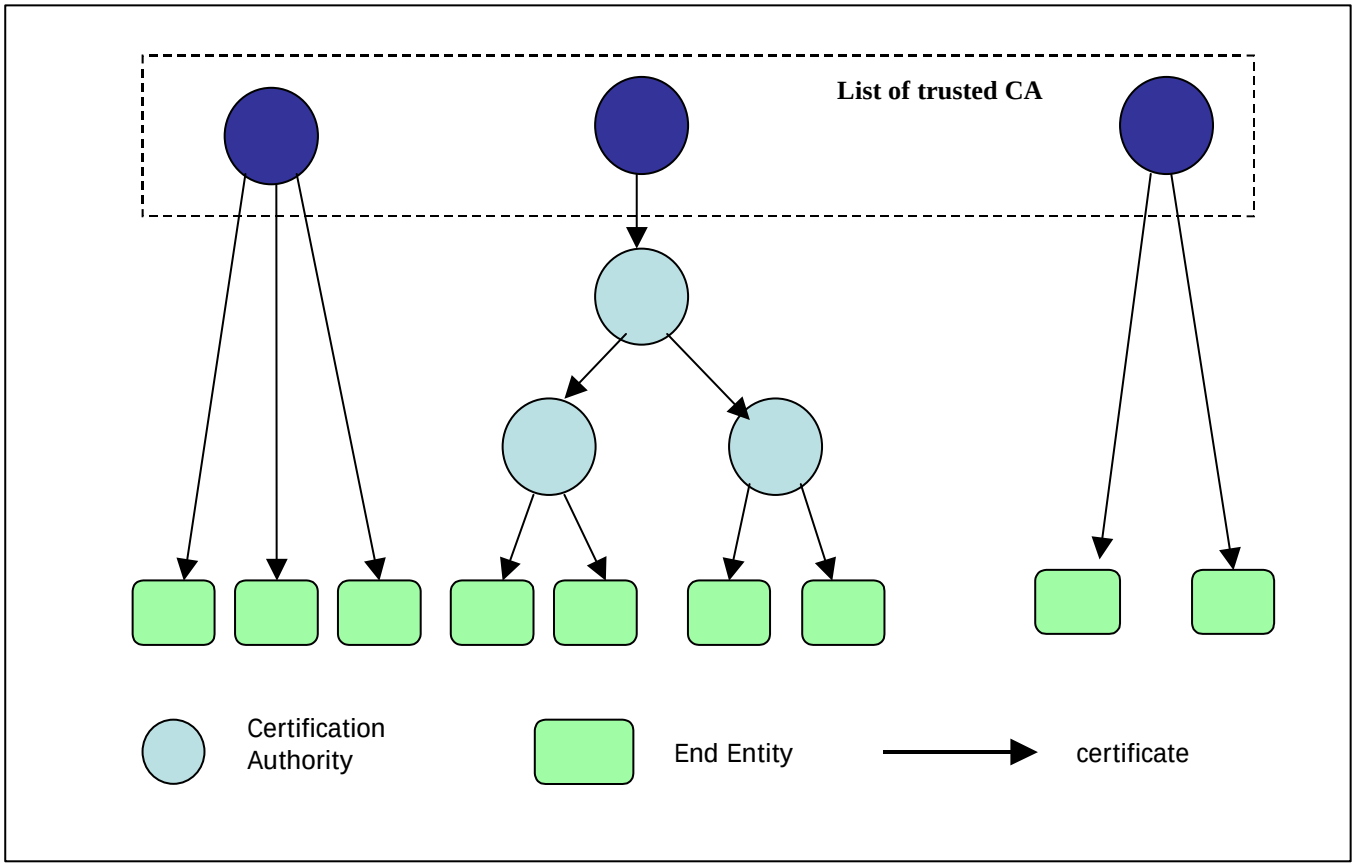


Figure 61 Trusted CA list

The main disadvantage of this model is the fact that trusts management is moved toward EE; actually, the organization must rely on the EE to take the actions to handle the CA (add/remove a CA to/from the Trust list), to configure policies and to maintain certificate status info up to date.

Moreover:

- The revocation of a TLCA is almost impossible (the certificate of that TLCA must be removed from the trust list of each EE)
- There is no way to limit the span of a specific hierarchy (TLCA in the trust list have all the same “value”; user-defined name constraints would be needed to restrict a specific hierarchy to a subset of the global name space.)

Trust lists are useful for small numbers of globally well-known CA, for direct use within enterprises and for interactions across a predefined set of enterprise boundaries.

7.4.1.5 CA cross-certification models

Various model of CA cross-certification have been proposed to overcome the problem of hierarchical PKI when interconnecting different realms.

Mesh Model

In this model (also known as network model), represented in Figure 62 below, *all* CA are self-signed. Trust flows via cross-certificates. An EE directly trusts only the CA that issued its certificate; it can trust another CA only if its direct CA cross-certified the foreign CA.

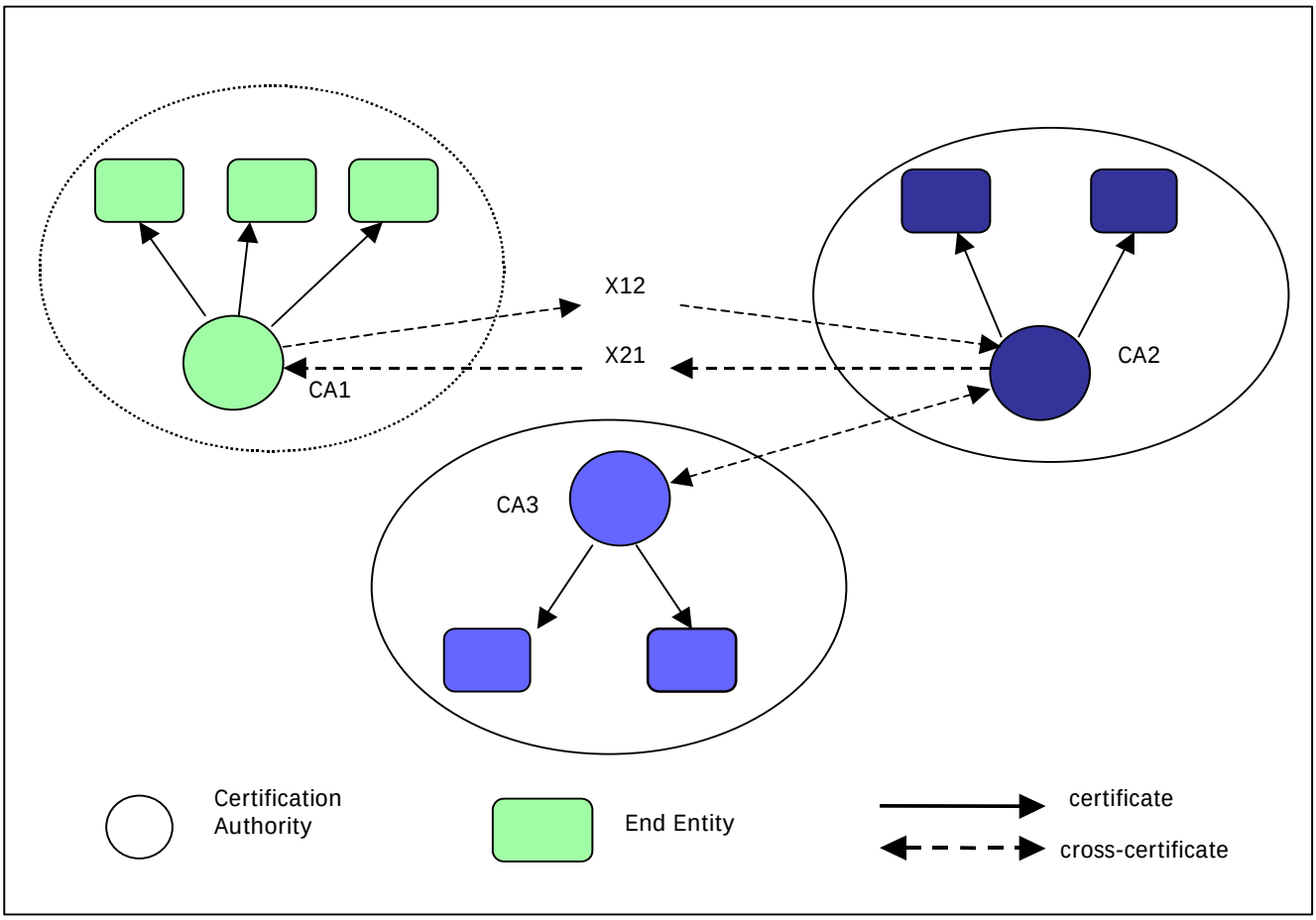


Figure 62 Cross certification: the Mesh Model

The mesh in Figure 62 above shows three PKI "domains" that trust each other via cross-certificates. Certificates can be uni-directional (as in the case of X12, meaning that CA1 cross-certifies CA2) or bi-directional (as in the case of the cross-certificate between CA2 and CA3).

The main advantages of these models are the following:

- The cross-certification is transparent and does not impact the users of the network (but the users must be able to retrieve the cross-certificates from a global repository).
- A foreign CA can be easily integrated into a mesh PKI, without changing the relative point of trust for its and the other pre-existing PKI "domains".

This model presents however the following drawbacks:

- It requires a global accessible repository to distribute cross-certificates to PKI clients. Without it, a RP cannot find the cross-certificates necessary to build the certification path to the CA that it directly trusts.
- The construction of a certification chain becomes complex, because multiple paths can exist between a certificate and the RPs' trust anchor.

The hybrid model

The hybrid trust model, shown in Figure 63, is similar to the mesh model, but only the TLCA of the hierarchy needs to cross-certify with the other CA of the mesh. This allows to reduce the number of cross-certificates.

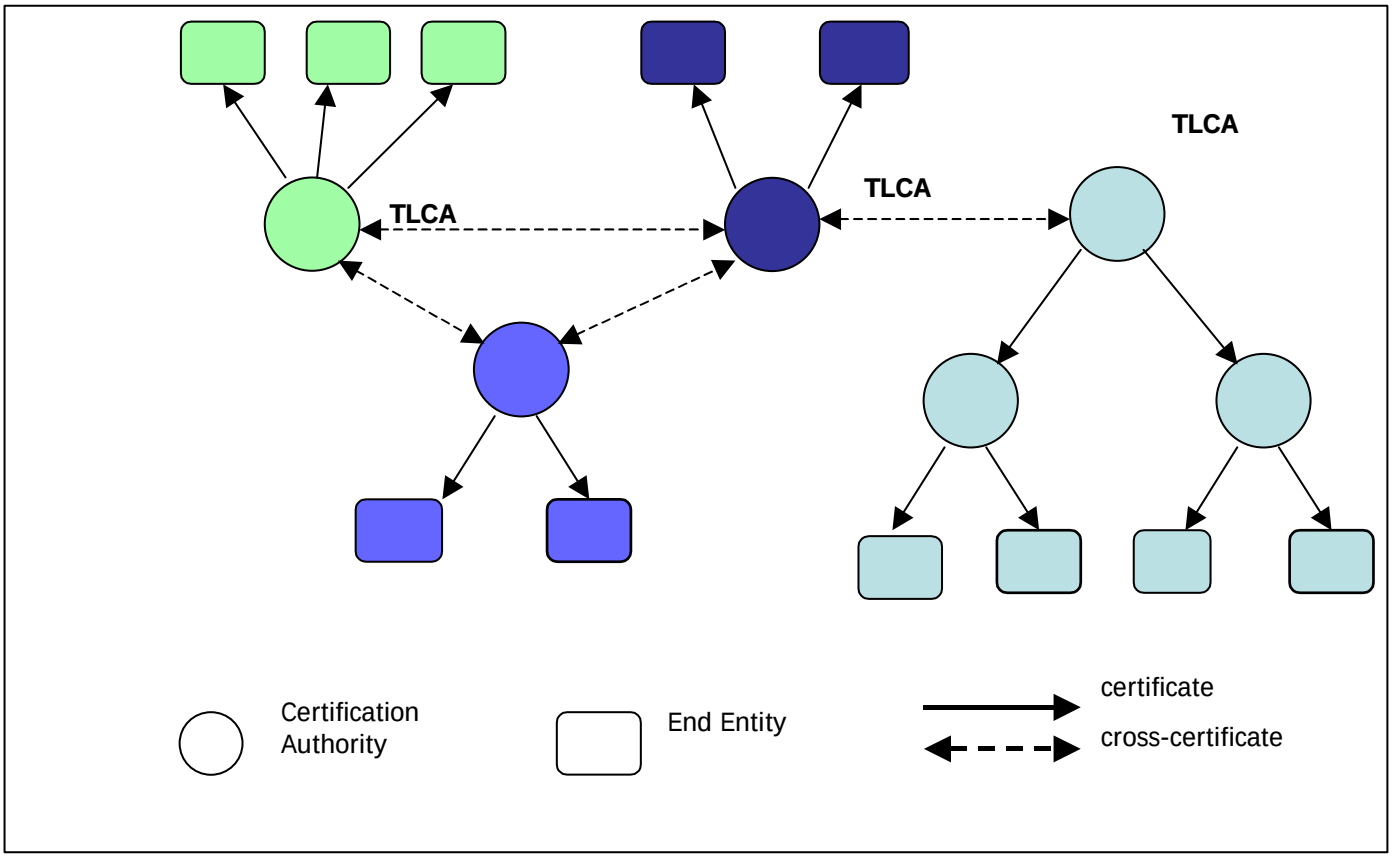


Figure 63 Hybrid Trust Model

In general, PKI cross-certification is subject to the following problems:

- The algorithm for certification path construction is complex
- Unwanted “trust paths” could be created
- The security level may decrease when a PKI with restrictive operating policies is cross-certified with a PKI adopting less restrictive policies

Bridge CA

The Bridge CA (also known as hub and spoke), shown in Figure 64, is another approach to the interconnection of PKI through cross-certification. In this model the Bridge certification authority (BCA) acts as a central point to establish trust paths among different PKI. In this configuration there is a principal CA (PCA) in each PKI “domain”, which cross-certifies with the BCA. While in the hybrid PKI model the number of cross-certificates grows quadratically, in this model the number of cross-certificates grows linearly with the number of participating PKI.

This model reduces the number of certificates needed in the mesh model too, but it requires participating organizations to agree with the BCA either the certificate format and the security policies, to reduce the risk of unwanted trust.

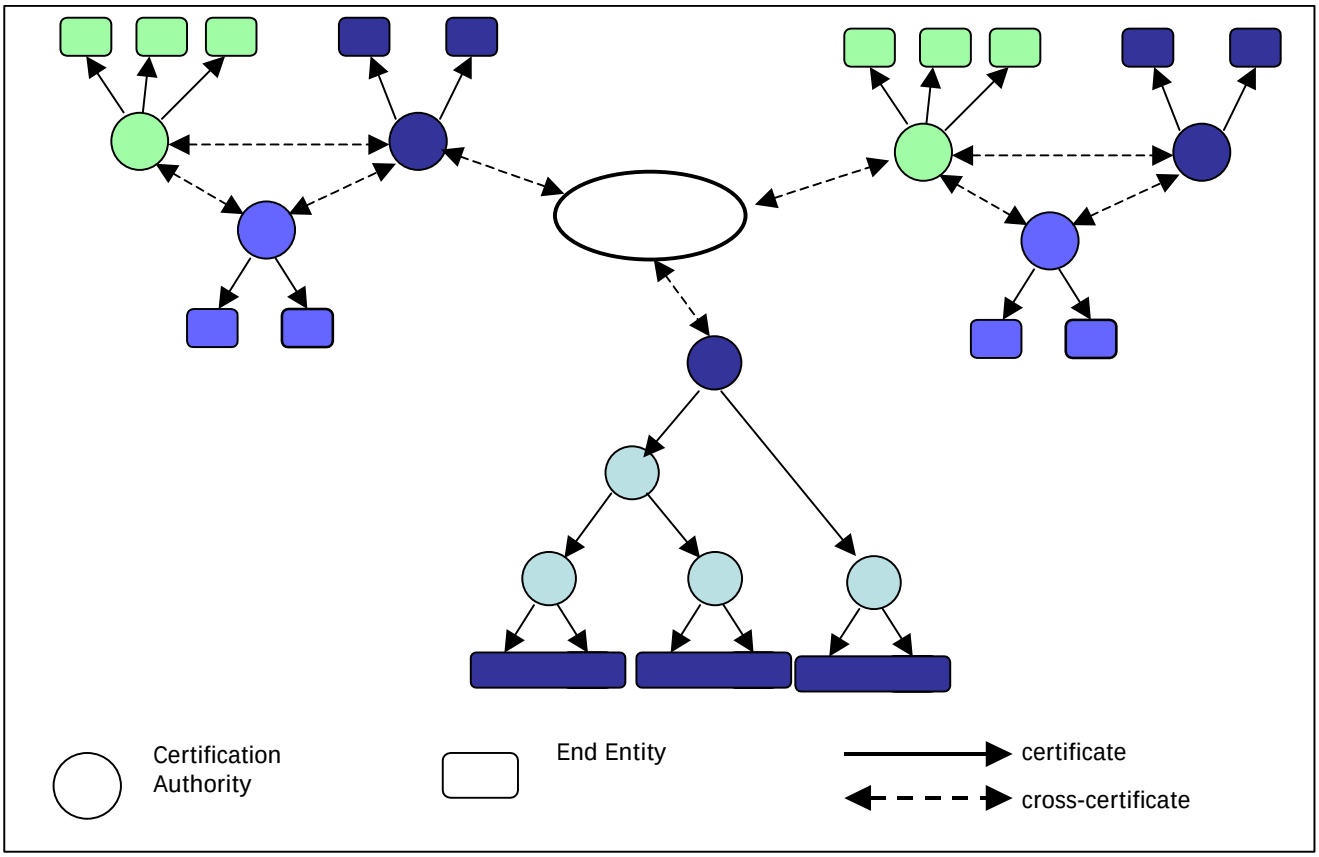


Figure 64 Bridge CA model

7.4.1.6 PKI-related Operational Trust Metrics

Although trust metrics are strictly related to the underlying trust model, in the context of authentication through a certification path of trusted intermediaries, trust metric can be defined as a function that computes a trust value from a set of digitally signed certificates. In this context, trust metrics has been studied in relation with the following problems:

- Determining the owner of a public key or, viceversa, determining the public key for a user (and then authenticating the binding between a public key to its owner), and
- Performing the authentication using a path of trusted intermediaries in large scale systems composed of different administrative domain where there is not a single authority for providing this information.
- The resistance of the certification infrastructure to attacks, namely the forgery of more than one key. Attacks can be distinguished in:
    - Node attack: the attacker is capable to generate arbitrary certificates (most general case); it corresponds to stealing the secret key of the victim.
    - Edge attack: when tricking owners of secret keys into certifying that untrustworthy keys are trustworthy

This problem has been studied by Reiter and Stubblebine[261], who proposed guidelines for the design of metrics supporting the evaluation of the "confidence" afforded by a set of certification path.

7.4.1.7 Weaknesses of X.509 and PKI

X.509 certificates (either Identity certificates - PKC - and Attribute certificates - AC -) and the PKI infrastructure itself have been subject to several criticisms as to the extent they can be really trusted.

As far PKC are concerned, X.509 V3[262] explicitly states "…A certificate user should review the certificate policy generated by the certification authority (CA) before relying on the authentication or non-repudiation services associated with the public key in a particular certificate. To this end, this standard does not prescribe legally binding rules or duties." Thus, PKC on themselves do not guarantee the CAs' trustworthiness. t X.509 V3 provided extensions such as Certificate Policy and Certification Practice Statements[263] and policy mappings[264], intended to allow the application to determine if the certification path is acceptable based on the contents of the certificates instead of a priori knowledge of PCA (Policy Certification Authorities). However, the acceptance and the adoption of such mechanisms by PKI infrastructure implementations and (especially) Internet applications is yet to be seen.

As far as ACs (Attribute Certificates[265]) are concerned, the Attribute Certificate Policies Extension[266] proposes that the AA (Attribute Authority) can explicitly state the Attribute Certificate (AC) policies that apply to a given Attribute Certificate. The goal is to allow relying parties to perform an additional test when validating an AC, i.e. to assess whether a given AC carrying some attributes can be accepted on the basis of references to one or more specific AC policies.

As far as ACs (Attribute Certificates[267]) are concerned, the Attribute Certificate Policies Extension[268] proposes that the AA (Attribute Authority) can explicitly state the Attribute Certificate (AC) policies that apply to a given Attribute Certificate. The goal is to allow relying parties to perform an additional test when validating an AC, i.e. to assess whether a given AC carrying some attributes can be accepted on the basis of references to one or more specific AC policies.

Registration processes

A CA needs to undertake some form of authentication process in order to verify itself that the claimed association actually exists. A conventional approach is to depend on the services of a Registration Authority (RA). A comprehensive registration process that an EE should undertake can be too expensive for individuals. As a result, PKI implementation schemes

---

[261] K. Reiter and S.G. Stubblebine. Toward Acceptable Metrics of Authentication. Proceedings of IEEE Symposium on Security and Privacy, pp 10-20, Oakland, May 1997

262 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (RFC3280).

263 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (RFC3280). Paragraph 4.2.1.5 Certificate Policies.

264 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (RFC3280). Paragraph 4.2.1.6 Policy Mappings.

265 An Internet Attribute Certificate Profile for Authorization (RFC 3281). Available at http://www.ietf.org/rfc/rfc3281.txt

[266] Attribute Certificate Policies extension. Available at http://www.ietf.org/internet-drafts/draft-ietf-pkix-acpolicies-extn-05.txt. Status: Internet Draft; expires on June 2004

[267] An Internet Attribute Certificate Profile for Authorization (RFC 3281). Available at http://www.ietf.org/rfc/rfc3281.txt

[268] Attribute Certificate Policies extension. Available at http://www.ietf.org/internet-drafts/draft-ietf-pkix-acpolicies-extn-05.txt. Status: Internet Draft; expires on June 2004

generally compromise on registration requirements, relying, for example, on some prior relationship between the person and the RA or CA

Hierarchical model of trust

X.509v3-based PKI is inherently hierarchical. Each layer of CAs may need to be attested to by some superior layer. This may result in scaling up to some "root" authority in which everyone is assumed to have ultimate trust.

Private Key management (generation, storage & backup, escrow)

PKI assumes that consumers and citizens will have, and must use, private keys. It is assumed that the holder of a private key will be able to ensure its security. The keys should be generated in a secure and certifiable manner. However, the keys are stored in personal workstations, such as Windows, MacOS and Linux machines directly connected to the Internet via commercial Internet access service providers. Although the keys are protected by cryptographic measures, such personal workstations do not yet provide a "trusted computing environment". Thus, private keys are potentially subject to several risks, such as of capture, and invocation without the authority ofthe consumer/citizen.

Name space and X.501 structure of the subject name

Multiple name spaces are supported, but within each name space each CA imposes a single, unique identifier on every subject under their jurisdiction. A 'distinguished name' has to be unique within each name-space. This prevents individuals to use alternative identifiers, and implicitly denies individuals the capability to have and use multiple key-pairs, and multiple certificates.

Revocation process

Time-stamping is a critical aspect of revocation processes; but this is not an assured, secure service. There are further difficulties in implementing an effective revocation process. This is especially serious if retrospective revocation is permitted (i.e. notification to a set of recipients that a private key had been compromised since some past time, and that the sender reserves the right to repudiate transactions signed after that time). A further concern is that many implementations fail to implement effective revocation procedures, using either the CRL or OCSP specifications.

Complexity

The X.509 standards are rich and complex and imprecise. Thus, interpretations of the standard are often required, and many variants, commonly termed 'profiles', exist. This may cause interoperability problems between different implementations.

7.4.2 Simple Public Key infrastructure (SPKI)

According to the charter[269], the Simple Public Key Infrastructure, or SPKI, intended to provide Internet standards for an IETF sponsored public key certificate format, associated signature and other formats, and key acquisition protocols.

SPKI was conceived to provide mechanisms to support security in a wide range of Internet applications, including IPSEC protocols, encrypted electronic mail and WWW documents, payment protocols.

As to the trust model underlying SPKI, while it was explicitly mentioned in the charter that "It is intended that the Simple Public Key Infrastructure will support a range of trust models"[269], SPKI was mainly inspired by the Web "trust model" where people must be able to choose whom they trust or consider reliable trust roots (even though with varying reliabilities).

SPKI addressed the issue of <name,key> bindings arguing that PKI identity certificates were of limited use for trust management because "…..A keyholder's name is one attribute of the

[269] C. Ellison. RFC 2692 SPKI Requirements, September 1999

keyholder, but a person's name is rarely of security interest. Rather, a user of a certificate needs to know whether a given keyholder has been granted some specific authorization"[269].

According to that view, SPKI was designed in accordance to the following requirements[269]:

- Authorization: the main purpose of an SPKI certificate is to authorize some action, give permission, grant a capability, etc. to or for a keyholder, although SPKI can be used to issue also name certificates (key-name mapping).
- Autonomy: the definition of attributes or authorizations in a certificate is up to the author of code, which uses the certificate. The creation of new authorizations should not require interaction with any other person or organization but rather be under the total control of the author of the code using the certificate.
- Local scope of identifiers: A keyholder is identified by an arbitrary local name, meaningful only to the issuer of the certificate. An SPKI certificate's ID binds a key to a keyholder or group of keyholders. It is explicitly stated in the SPKI/SDSI literature that others must not assume any binding between the local identifier and a particular person, based on the spelling of that name.
- Privacy preservation:
    - Each SPKI certificate should carry the minimum information necessary to prove the keyholder permission to act.
    - Certificates are distributed directly by the keyholder to the verifier, because SPKI certificates might carry information that the keyholder might not want to publish. As a consequence, PKI does not mandate for the use a global repository, such as LDAP, the global PGP key server or the DNS database.
- Anonymity support: SPKI certificate must be able to assign an attribute to a blinded signature key.
- Certificate validation and revocation:
    - An SPKI certificate should be able to carry a validity period, that is the dates within which it is valid. To verify the validity of an SPKI certificate, CRL can be used, but the requirement is that the certificate that uses it must explicitly tell the verifier where to find the CRL, the CRL must carry explicit validity dates and the dates of a sequence of CRLs must not overlap. Under this set of requirements, behaviour of certificate validation is deterministic.
    - SPKI should support both positive and negative on-line validations.

7.5 Trust Negotiation

7.5.1 Overview

Trust Negotiation (TN)[270] is an emerging approach exploiting the concept of properties of the entities as a means for establishing trust, particularly in open environments such as the Web, were interacting entities are mostly unknown to each other. Trust negotiation is a peer-to-peer interaction, and consists of the iterative disclosure of digital credentials, representing statements certified by given entities, for verifying properties of their holders in order to

[270] K. Seamons, M. Winslett et al. Requirements for Policy Languages for Trust Negotiation. Third International Workshop on Policies for Distributed Systems and Networks, Monterey, California, June 2002

establish mutual trust. In this approach, access to resources (data and/or services) is allowed only after a successful trust negotiation that is only after mutual trust is established.

7.5.2 Basic Concepts

TN deals with concepts like formulation of security policies and credentials, determining whether particular sets of credentials satisfy the relevant policies, and deferring trust to third parties.

In this section we briefly introduce how negotiations are generally intended, and identify the main phases and functional components of negotiations.

TN building blocks

A TN involves two entities, namely a *client*, that is, the entity asking for a certain resource, and a *server*, that is, the entity owning (or more generally, managing access to) the requested resource. The model is basically peer-to-peer: both entities may possess sensitive resources to be protected and thus must be equipped with a compliant negotiation system. The notion of *resource* comprises both sensitive information and services, whereas the notion of entity includes users, processes, roles, and servers.

The term resource is intentionally left generic to emphasize the fact that the negotiations we refer to are general-purpose, that is, a *resource* is any sensitive object (e.g., financial information, health records, credit card numbers) whose disclosure is protected by a set of *policies*.

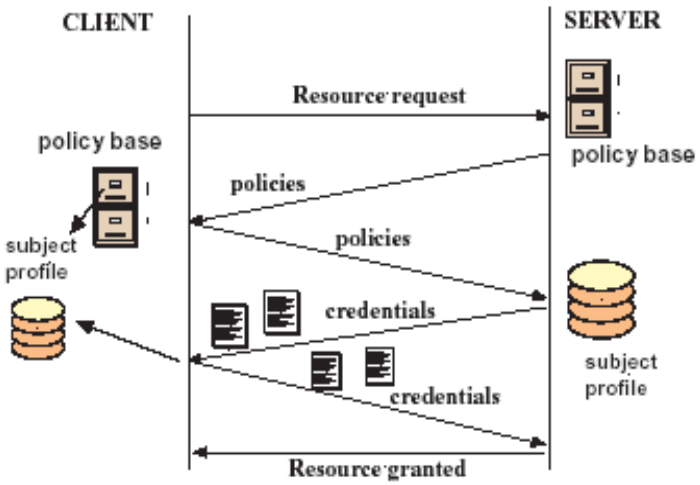


Figure 65 Sketch of a Trust Negotiation Process

In the example, negotiating participants perform a TN in which the client obtains a service after exchanging policies and credentials with the server.

Figure 65 sketches a typical negotiation process. During a negotiation, trust is incrementally built by iteratively disclosing *digital credentials* in order to verify properties of the negotiating parties. Credentials are typically collected by each party in appropriate repositories, also named profiles.

Another key component of any TN is a set of access control policies, known as **disclosure policies**, that govern access to protected resources by specifying credential combinations that must be submitted to obtain authorizations. The fundamental elements of trust negotiations are thus digital credentials and policies, discussed in the following sections.

To carry out a trust negotiation, parties usually adopt a strategy, which is implemented by an algorithm defining which credentials to disclose, when to disclose them and whether to succeed or fail the negotiation. There can be a number of strategies for negotiating trust, each with different properties with respect to speed of negotiations and caution in releasing credentials and policies.

The efficiency of a strategy depends on two factors: the communication cost and the computational cost. The communication cost includes the sizes of the messages exchanged and their number. Communication and computational costs of a negotiation strictly depend on the adopted strategy and vary from exponential, if a brute force strategy is adopted, to more efficient strategies.

Digital Credentials

Digital credentials are usually defined as assertions describing one of more properties about a given subject, referred to as the "owner", certified by trusted third parties. Entities are thus identified and described through set of *digital credentials*, whereas trusted third parties correspond to Certification Authorities (CAs).

Digital credentials contain properties about the owner and both must be unforgeable and verifiable. To ensure such properties credentials are digitally signed using PKI[271]. Typically, a digital credential contains a set of properties specified using name/value pairs that are signed by the issuer's private key and can be verified using the issuer's public key. Although some proposals exist[272] for encoding digital credentials, up to now there does not yet exist a widely accepted standard for their representation. The X.509 V3 standard[271] for public key certificates makes a step in this direction, due to its extensibility mechanisms.

The extensions concern the provision for fields such as additional subject identification information, key attribute information, policy information, and certification path constraints. However, because the X.509 certificate was not conceived for on line negotiations, it does not properly support neither the notion of attributes nor protects privacy. As a result, other formats have been recently proposed that can better support entities property description or that can achieve privacy and non-forgeability[273].

Disclosure policies

Disclosure policies state the conditions under which a resource can be released during a negotiation. Conditions are usually expressed as constraints against the credentials possessed by the interacting parties and their properties. Further, depending on their contents, credentials may be sensitive. For example, a credential may contain non-public attributes about an individual such as a credit card number. Because of the sensitive nature of digital credentials, their disclosure must be carefully managed according to policies that specify which credentials must be received before other credentials can be disclosed.

Disclosure policies also may be regarded as sensitive information because they are often related to the business and governance processes of organizations. Therefore, recent researches[274],[275] consider disclosure policies as sensitive as other resources. The presence of sensitive disclosure policies adds new requirements to trust negotiation processes.

Trust has to be gradually established and policies for the involved resources have to be sent to the other party according with the level of trust established.

---

[271] W. Stallings. Cryptography and Network Security: Principles and Practice, Second Edition. Prentice Hall, 1999

[272] A. Herzberg, Y. Mass, Relying Party Credentials Framework. RSA Conference, San Francisco, CA, April 2001.

[273] S. Brands, Rethinking Public Key Infrastructure and Digital Credentials MIT Press, 2000.

[274] K. Seamons, M. Winslett, T. Yu Limiting the disclosure of Access Control Policies during Automated Trust Negotiation. Network and Distributed System Security Symposium, San Diego, California, April 2001.

[275] T. Yu, M. Winslett. A Unified Scheme for Resource Protection in Automated Trust Negotiation. IEEE Symposium on Security and Privacy, Oakland, CA, 2003.

7.5.2.1 TN building blocks

7.5.2.2 TN Requirements

In this section we outline the dimensions we consider more relevant for evaluating policy languages. We have classified the dimensions in two main groups, i.e., those related to the adopted language and those related to the system and its components.

It is important to note that the requirements we have devised are a partial list and other requirements will likely be identified as research and deployment of negotiation systems progress, given also the increasing number of researchers actively contributing to the trust management area[276].

Language Requirements

Trust negotiation policy languages[277],[278] are a set of syntactic constructs (e.g., credentials, policies) and their associated semantics, encoding security information to be exchanged during negotiations. Good TN languages should thus be able to simplify credential specification and also to express a wide range of protection requirements through specification of flexible disclosure policies.

The dimensions we have identified to reach these goals deal with language expressiveness and semantics, and are described in what follows.

- **Well-defined semantics:** A well-defined policy language should have a simple, compact, formally defined semantics, resulting thus independent from the particular implementation of the language. The semantics may be effectively expressed using a variety of formalisms such as logic programs or relational algebra.
- **Monotonicity:** The monotonicity requirement deals with the fact that once a set of credentials allowing the disclosure of a certain resource is found; the disclosure of additional credentials and policies should only result in the grant of additional resources, if possible. This aspect implies that negation must be carefully handled: if a policy requires that a subject must not have a given property to obtain a resource, then it is not enough that the subject simply fails to disclose the corresponding credential. Rather, the verification of such negative conditions should be carried out by directly checking with the credential issuer authority. The check of absence of a credential can be managed at the policy level as long as the policy owner has the capability to perform such a check.
- **Credential combination:** The set of properties characterizing a given subject may be likely described by a set of different credentials. Thus, a policy language should be expressive enough to require submission of combination of credentials, using conjunction and disjunction operators.
- **Authentication:** Each party can have multiple identities stated by different credentials issued and signed with different public keys to prevent collusion. At runtime, the credential submitter, that is, the Certificate Authority, or one delegated entity by means of credential chains, will thus have to demonstrate the knowledge of the private key associated with the public key within which the credential is signed.
- **Constraints on property values:** Each credential is usually a structured object conveying information about subject properties. Each property is usually represented as a name/value pair. Credentials can be associated with a given credential type, thus simplifying credential specification and management. A policy language should include

[276] P. Nixon and S. Terzis Proceedings of Trust Management, First International Conference, iTrust 2003, Heraklion, Crete, Greece, May 28-30 2003, Springer Lecture Notes in Computer Science.

[277] E. Bertino, E. Ferrari, A. Squicciarini. Trust-*X*: A Peer-to-Peer Framework for Trust Negotiation. To appear in IEEE Transactions on Knowledge and Data Engineering

[278] A. Herzberg, Mihaeli, Y. Mass, D. Naor, and Y. Ravid, Access Control System Meets Public Infrastructure, Or: Assigning roles to Strangers IEEE Symposium on Security and Privacy, IEEE Computer Socety Press, Oakland, CA, May 2000.

constructs to constrain the requested credentials to have a certain type and restrict the valid property values. For example, a rental car agency may ask to submit a driving licence, and, further, that the driving licence has been issued after a given date.

- **Inter-credential constraints:** To better evaluate remote party properties policies may express contraints to compare values of different credentials belonging to the same subject, even if they use different keys.
- **Protection of sensitive policies:** As introduced in previous section, a disclosure policy itself may be sensitive. By analyzing policies content, outsiders may infer sensitive information about the parties. Thus, disclosure policies have to be protected in the same way as other resources, providing a fine-grained control over their disclosure. Policy protection can be realized at language level, or at system level. In the first case, the policy language must have constructs to express constraints on policy disclosure, whereas in the second case, the runtime system must check disclosure policies and dynamically add constraints instead of disclose them unconditionally.
- **Unified formalism and use of interoperable languages:** These requirements mainly focus on the applicability of negotiation approaches, since we believe that in designing negotiation languages it is essential to focus on solutions that can be effectively adopted in real environments and that can be easily integrated in existing contexts. The first requirement deals with the possibility of protecting uniformly credentials and policies, thus simplifying protection mechanisms. The latter requirement facilitates the transmission and interoperability among negotiation participants. In this respect, the use of metalanguages such as XML[279] may facilitate submission and exchange of credentials and policies.

System Requirements

A negotiation system is the system supporting a trust negotiation. The challenging aspect in the development of such kind of systems is to devise solutions trading off among several requirements, often in conflict with each other. On the one hand, such systems should be flexible, scalable and portable. On the other, they should support advanced functions, such as support for credential chains, authentication of multiple identities, and complex compliance checker modes whose efficient implementation is often difficult.

In particular, the compliance checker should be able to interpret a remote policy and check whether there exists a set of local credentials that satisfy the received policy.

In the following, we detail the requirements we have identified at the system level:

- **Credential ownership.** During a negotiation, when a remote credential is received, the runtime system must challenge the sender to prove the ownership of the private key associated with the public key used to identify the subject in the credential itself. Various security protocols can be used for such task. A key issue is to integrate the negotiation framework with the existing tools and systems maximizing security controls over the exchanged data.
- **Credential validity**. The validity of the exchanged credentials is fundamental to ensure a correct functioning of the whole negotiation. Thus, each time a credential is received the integrity of the credential content must be verified by using digital signature to guard against forgery. Further, the runtime system must always check that credentials are not expired and that have not been revoked.
- **Credential chain discovery.** The credentials needed during a negotiation may not be readily available locally. A runtime system should include extra machinery and tools for credential chain discovery to retrieve in real time credentials that are not locally cached.

[279] World Wide Web Consortium Extensible Markup Language (XML) 1.0, 1998. Available at: http://www.w3.org/TR/REC-xml

- **Privacy protection mechanisms.** Disclosing policies and resources should ensure a good protection of privacy of the parties, which are typically interested in disclosing the minimal set of information necessary to succeed in the process. Intuitively, the unconditioned disclosure of policies and credentials may leak sensitive information. Complementary mechanisms should be integrated with the system to address privacy requirements of the parties.
- **Support for alternative negotiation strategies.** The negotiation system should support a variety of negotiation strategies, for instance by maximizing information protection or considering first the computational effort required. A well-designed system should provide a number of different strategies, leaving the parties free to choose the preferred one. The strategies may be chosen independently by the parties, or negotiated as the other resources. In the first case, it is essential to ensure a correct functioning of negotiations even if parties do not adopt the same strategy.
- **Fast negotiation strategies.** It is expected that in many scenarios there will be standard, off-the-shelf policies available for widely used resources (e.g., VISA cards, passports). It is thus likely that in case of negotiations involving such common resources the same sequences of credentials will be used several times to perform similar negotiations. In such cases it might be useful to keep track of the sequences of credentials more often exchanged, instead of recalculating them for each negotiation. The strategies should thus include approaches to let the parties establishing trust by using pre-computed sequences, if desired. Additionally, once two parties have successfully negotiated, the system might give the possibility of exploiting such negotiation result to speed up subsequent negotiations. Finally, when commonly used resources are involved an ideal system should automatically select and suggest the policies to be exchanged, even when parties are total strangers.

7.5.3 Evaluation

The main characteristics of a TN system can be summarized as follows:

- A negotiation is a peer-to-peer interaction, where the peers are mostly unknown each other. Thus, the TN approach is suitable for an open and dynamic environment such as B2C e-markets or Virtual Communities VO.
- Negotiations are based on the properties of the participating entities: the knowledge of and the authentication of the identity of the participating entities is not mandatory.
- Negotiationare policy-driven; each participating parties may adopt its own negotiation policy. Moreover, policies are explicitly stated. Thus the TN approach preserves the autonomy of participating entities and contributes to possibility of the involved parties of taking informed decisions. This could be useful, for example, in the context of GRID data services, where the explicit policy could be a SLA.
- Negotiation policies are first-order objects that is their disclosure can be protected by the same mechanisms controlling the disclosure of a resource.
- Up to now, in the proposed TN systems the output of a negotiation is the disclosure of a resource; the disclosure is more precisely the grant to the requesting party of permission on the requested resource(s). In this way, TN can be seen as a layer on top of a discretionary access control system. It is conceivable to extend the approach to cope with RBAC-based access control systems.
- On the other side, proposed TN systems do not address the management over time of the trust relationships established between the parties. Indeed the relationships between the parties are not “persistent” objects neither explicitly modelled nor managed.
- However, a TN system basically relies on a trustworthy security infrastructure: PKI (or XML-based) infrastructure should guarantee the unforgeability of certificates and the trustworthiness of the Certificate and Attribute Authorities, as well as standard and

secure protocols. In perspective, a TN system could also be extended to use upcoming trust reputation services.

The challenge is to make emerging TN negotiation system modular, pluggable components of a more comprehensive, distributed Trust Management Infrastructure, supporting the different phases of a VO life-cycle, from the VO formation to the VO operation.

7.6 Trust-X

7.6.1 Overview

Trust-*X* is an XML-based framework developed at University of Milan. It is aimed to take into account all the aspects of the trust negotiation process: from the specification of the profiles and policies of the involved parties to the determination of the strategy to succeed in the negotiation. Trust negotiation is performed through a mutual exchange of digital credentials. Disclosure of credentials, in turn, is protected through the use of policies that specify which credentials must be received before the requested credential can be disclosed.

7.6.2 Description

Trust-*X* provides X-TNL[280], an XML-based language, for specifying certificates and policies. **Trust-*X*** certificates convey information about the profile of the parties involved in the negotiation and can be either credentials or declarations.

A credential is a set of properties of a party certified by a Certification Authority, whereas declarations contain information that may help the negotiation process (such as for instance specific preferences of one of the party) but do not need to be certified. Additionally, to facilitate certificate exchange, credentials and declarations can be further structured into data sets and X Profiles. Each data set collects a class of credentials and declarations referring to a particular aspect of the life of their owner, and can be used to facilitate certificate exchange.

Disclosure policies regulate the disclosure of a resource by imposing conditions on the certificates the requesting party should possess; they are encoded using XML. A resource can be either a service, a credential, or any kind of data that need to be protected. Moreover, during negotiation disclosure policies for a resource can be gradually disclosed, according with the grade of trust established with the counterpart, ensuring better protection of sensitive information exchanged.

The language we have developed has been especially conceived for handling multiple and heterogeneous credentials and it is flexible enough to express a wide range of protection requirements.

Additionally **Trust-*X*** comprises architecture for negotiation management, which is symmetric and peer-to-peer.

7.6.3 Methodology and Approach

A **Trust-*X*** negotiation consists of a set of phases to be sequentially executed. The key phase of a **Trust-*X*** negotiation is the policy evaluation phase which consists in a bilateral and ordered policy exchange. The goal is to determine a sequence of certificates of the

[280] E. Bertino, E. Ferrari, A. Squicciarini. X-TNL – An XML based Language for trust Negotiation. Fourth International Workshop on Policies for Distributed Systems and Networks, Como, Italy, June 2003

parties that when disclosed allow the release of the requested resource, in accordance to the disclosure policies of both parties.

Once a trust sequence has been determined, the certificate exchange effectively takes place until the disclosure of the requested resource. In this way we maximize the protection of the involved resources: indeed, we apply the principle of separation between policy exchange and resource disclosure. This distinction realizes an effective protection of all the resources involved during negotiations.

Disclosing of certificates and services is executed only after a complete counterpart policies evaluation, that is, only when the parties have found a sequence of certificate disclosure that makes it possible the release of the requested resource. Another distinctive feature of **Trust-*X*** negotiation is the support of alternative ways to carry out a negotiation, according with trust requirements of the negotiating parties.

The first is based on the use of trust tickets, and can be adopted when the parties have already successfully ended a negotiation for the same resource. The last mode exploits a notion of similarity between negotiations and it is based on the observation that usually a service provider handles a large amount of quite similar negotiations.

In particular, when a negotiation successfully ends, the corresponding trust sequence is processed by a module that decides whether to cache the trust sequence for further use in similar negotiations. Finally, another key feature of **Trust-*X*** is the support of privacy techniques and mechanisms.

7.6.4 Advantages and Disadvantages

The language we have developed has been especially conceived for handling multiple and heterogeneous credentials and it is flexible enough to express a wide range of protection requirements.

Moreover, the model allows to treat disclosure policies as first order objects, that is, during negotiation disclosure policies for a resource can be gradually disclosed, according with the grade of trust established with the counterpart, ensuring better protection of sensitive information exchanged.

However, a disadvantage of the model is that Trust-X implementation is a prototype and it has not been used in any real world application so far.

7.6.5 Application to TrustCoM

Peer-to-peer trust negotiation between parties is one of the mean to establish initial trust and thus is relevant to the VO formation phase in the VO life cycle.

7.7 Intelligent Computation of Trust

The 3-year Intelligent Computation of Trust (ICT) project (98-01), built a system that allows the trust index of a Certification Authority (CA) to be computed both statically and dynamically. Static calculation is based on a CA's published Certificate Policy (CP) and Certification Practice Statement (CPS), whilst dynamic calculation is based on the actual current practices of the CA. At the heart of the system is an expert system that has knowledge about the factors that are important in computing the trust in a CA. This knowledge was gained by interviewing know PKI experts, and the results of these interviews are published in281.

[281] D. W. Chadwick, A. Basden. "Evaluating Trust in a Public Key Certification Authority", Computers and Security, Vol 20, No 7, (Nov 2001) pp 592-611.

Static calculation of trust may be performed in one of two ways. In Method 1, the expert system asks the user (the CA's relying party) a series of questions, which he can answer by consulting the published CP/CPS of the CA. A web interface is provided for this, and the ICT server is a permanent running service on the Internet (see http://huan.isi.salford.ac.uk:7007/). At the end of the session, the expert system provides a trust index for the CA in question. Values range from 0, meaning absolutely no trust can be placed to the CA, to 1, meaning that the CA is absolutely trustworthy. In practice is it impossible for any CA to score 1.0. However, answering all the questions asked by the expert system is not an easy process, and may take the user up to an hour to answer all the questions. Consequently a more automated method is required.

In Method 2, the expert system asks the same set of questions to a CPS Server, which derives its answers from an XML formatted CPS. This requires the CA administrator to first produce an XML formatted CPS (the DTD of this we have defined) and then to publish this in its LDAP directory along with its public key certificates and revocation lists. The relying party can now point the expert system to this CPS server, in order for the trust index to be quickly computed. The CPS server retrieves the XML CPS as a signed X.509 attribute certificate, to provide tamper resistance, and feeds answers to the questions posed by the expert system using a Simple SOAP protocol that we designed. (Note that we designed this protocol before SOAP was a published standard.)

Dynamic calculation of the trust index may be based on evidence gathered from up to five sources: an Audit Certificate created by the external auditors of the CA, dynamic performance monitoring of the CA's rate of publication of Certificate Revocation Lists, information gathered by the relying party, information gathered by the subscriber, and information gathered about the vendor of the CA's PKI software. We have currently only implemented the first two of these. The software has been written in Java and also provides tools that enable Audit Certificates and CPSs to be prepared and published.

7.7.1 Application to TrustCoM

Future authorisation infrastructures will no doubt link what a user is authorised to do into the level of authentication that the user underwent when logging into the VO. For example, a user who presented only a username and password may have less access rights granted to him that one who presented a smart card, PIN and biometric. Indeed, the Universities of Salford and Manchester currently have a project (FAME-PERMIS) to do just this, so that the PERMIS authorisation infrastructure will grant or deny access based partly upon the level of authentication undergone by the user.

Consequently the trustworthiness of a Certification Authority in authenticating a user during the process of certifying their public key and providing them with an X.509 public key certificate, would be a factor to take into account when determining the level of authentication of a user, and subsequently, when providing that user with access to resources in a VO. Thus if the providers of Public Key Infrastructures used by VO members, were to provide XML versions of their CPSs, and publish these in the same place as their CRLs, (either on the Web or in an LDAP directory), then the CPS server provided by the ICT project would be of value to the TrustCoM project in determining the reputation or trustworthiness of the Certification Authorities, and thereby the level of authentication that their users have undergone.

The mechanism used by the ICT project could also be extended and applied to all Asserting Authorities (AAs). If each AA was to publish its Assertion Policy in XML, then it would be possible to provide an expert system capable of reading in this policy and determining the inherent trustworthiness of each assertion published by the AA. This trustworthiness would then be basic information that could be fed into the reputation system used by TrustCoM.

7.8 SULTAN

Trust management frameworks discussed in the following chapter, namely PolicyMaker, KeyNote and IBM's Trust Establishment Framework) do not provide mechanisms to deal with the dynamic nature of trust, which changes with risks, time and experience. These frameworks assume trust relationships are static and they do not adjust trust levels using evidence or experience.

SULTAN (Simple Universal Logic-oriented Trust Analysis Notation)[282] provides a more comprehensive solution to trust management problem than current models. SULTAN is an abstract, logic-oriented framework designed to facilitate the specification and analysis of trust relationship. It attempts to incorporate concept such as experience, risks, reputation and trusting propensity in order to specify and evaluate trust. System administrators with global view of the system resources and needs will use SULTAN to define their trust relationships. These relationships may be used as the basis for generating access control and authentication policies.

SULTAN's trust management consists of the following components:

- Trust Establishment defines the protocols by which the parties negotiate and exchange the evidence and credentials, which are needed by for evaluation. Evidence may include credentials (e.g., identity certificates, qualifications, and etc.), risk assessments, usage experience or recommendations.
- Trust analysis is the process of checking the semantic properties of the trust and recommendation specifications to determine conflicts and implicit relationships.
- Trust Evaluation service collects and evaluates the evidence for defining the trust relationship.
- Trust Specification defines trust relationship in terms of the entities involved and the context of the interaction.
- Trust Monitoring is needed to update experience and risk information. This allows for the re-evaluation of the trust specifications that based on the evidence (which is, experience from interactions, new risk evaluation methods or changes in an entity's credit ratings).

SULTAN uses two constructs for specifying trust relationships: the *trust* construct and the *recommend* construct.

1) *Trust construct:* Used to specify both a trust and distrust relationship.
   `PolicyName : trust (T1, T2, ActionSet, Level ) ? ConstraintSet;`

   T1 (Trustor) trusts/distrusts T2 (Trustee) for a context specified by ActionSet at trust/distrust Level if ConstraintSet is true; where PolicyName is a unique identifier for the assertion.

   Example:

   `S1: trust (Bob, Bank, NewAccount(Bank), 50) ? online(Bank);`

   Bob trusts Bank with respect to opening an account at Bank at a medium l evel (50) if the bank provides online access.

2) *Recommend construct:* Recommendations does not imply that some entity must be trusted; it may be used as the basis for defining new relationships. They can be positive or negative.

   `PolicyName: recommend ( R1, R2, ActionSet, Level )? ConstraintSet;`

[282] T. Grandison and M. Sloman. Specifying and Analysing Trust for Internet Applications, 2nd IFIP Conference on E-Commerce, E-Business and E-Government. 2002. Lisbon, Portugal.

R1 recommends/does not recommend *R2* with a confidence of Level to perform *ActionSet* if *ConstraintSet* is true; where *PolicyName* is the unique name of the rule being defined.

Example:

```
R1: recommend (Bob, Sainsbury, buy_products(Sainsbury), HIGHREC) ?
BasketCost(Sainsbury) < £40;
```

Bob recommends Sainsbury to provide a buy_products service with high confidence if the cost of a standard basket of products at Sainsbury is less than £40.

Analysis of the trust and recommendation rules is based on checking whether specified properties hold. These properties can be based on specification source (essentially program reasoning) and trust relationships to identify scenarios of interest. SULTAN analysis includes support for checking conflicts, ambiguity detection and cycles to resolve any ambiguities.

The prototype for analysis model is implemented in prolog and in order to perform an analysis query, the rules in the analysis model are used. SULTAN uses the following prolog construct to perform an analysis query:

```
query(Vars, Conds, ResultSet)
```

This finds all the *Vars* (variables) that satisfies the conditions *Conds* and stores the result in *ResultSet*. SULTAN uses two other functions to measure risk and update experience:

1) The *risk* function*:* ***risk*** *(B, C, A)* returns the risk entity B undertakes when entity C performs A. The risk value is the probability for the failure of an activity (A), and the risk level is defined using the interval 0-100; 0 indicating no risk, and 100 indicating highest risk possible.
2) The *experience* function: ***experience*** *(B, C, A)* returns entity B's estimate of the experience it had with entity *C* with respect to action set A. The experience value is defined using the interval –100 to 100. Negative integers (< 0) represent a negative experience and positive integers (> 0) a positive experience.

Sultan provides several components to support trust management. A *Specification Editor* is provided for administrators to specify their initial trust requirements. Constraint, risk and experience information are held in a *State Information Server*, and the trust and recommendations specifications are stored in a *Specification Server*. The *Analysis Tool* is used to evaluate trust and recommendations specifications to detect conflicts. The *Risk Service*, which performs risk calculations and retrieves risk information from the State Information Servers. The *Monitoring Service* updates the information relating to experience or reputation of the entities held within the State Information Server.

7.8.1 Advantages/Disadvantages

Previous trust management solutions focus on authentication and access control decisions whereas SULTAN proposes a wider scope for managing, reasoning trust relationships by considering trust and risk analysis, and experience issues. Sultan provides mechanisms for detecting consistencies among trust relationships. SULTAN's prolog query syntax can be rather complex for the average users because it requires considerable programming expertise.

7.8.2 Application to TrustCoM

SULTAN can be used in TrustCoM as a decision support tool to aid entities to reason and assess trust relationships, and propagate trust through recommendation. SULTAN makes use of evidence, and considers risks to dynamically assess trust relationships. Ponder

(Chapter 8) and SULTAN can be connected either by using SULTAN in Ponder policies or by using Ponder as the target for the refinement of SULTAN rules. Ponder policies can use SULTAN either to check that someone be trusted/recommended or to update experience information. This could be used to define authorisation policies.

7.9 Conclusions

In this section we analysed several aspects of trust management that, each one to a different extent, can be useful to support the life cycle of the different types of VO foreseen in the TrustCoM project.

In the VO formation phase, trust must be established among VO participating entities. To this end, the parties should negotiate and exchange the evidence and credentials, which are needed by for evaluation. Evidence may include credentials (e.g., identity certificates, qualifications, and etc.), risk assessments, usage experience or recommendations.

The Trust Negotiation approach can be used in the VO formation phase; this approach preserves the autonomy of participating entities and contributes to possibility of the involved parties of taking informed decisions. This could be useful, for example, in the context of GRID data services, where the explicit policy could be a SLA. Up to now, in the proposed TN systems the output of a negotiation is the disclosure of a resource; the disclosure is more precisely the grant to the requesting party of a permission on the requested resource(s). In this way, TN can be seen as a layer on top of a discretionary access control system. It is conceivable to extend the TN approach to cope with RBAC-based access control systems.

Decisions could be based not only on the evidence of authenticity of entity identity and/or entity properties (identity and/or attribute certificates), but also on evidence of entitys' reputation: to this end, four different operational models for reputation servers were presented, and the inherent trustworthiness of each model evaluated. Servers that use publicly accessible data, and that process this in an open and transparent way are inherently the most trustworthy, whilst those that do not disclose either the source of their data, or the algorithms they use to compute reputations are inherently the least trustworthy.

The policy extension of PKI certificates can be used too, reasoning upon them to determine the reputation or trustworthiness of the Certification Authorities themselves, and thereby the level of authentication that their users have undergone.

A more comprehensive solution to trust management, such as that provided by SULTAN, attempts to incorporate concept such as experience, risks, reputation and trusting propensity in order to specify and evaluate trust. In this systems trust relationships may be used as the basis for generating access control and authentication policies.

During the VO operation phase, that is once that the initial trust among the VO participants is established, the mechanisms provided by the PKI infrastructure and Attribute Certificate could be used by a VO member to authorise the access to its own resources to another requesting VO member.

8 Policies and Security

Edited by: Nilufer Tuptuk
Imperial College London

8.1 Introduction

This section presents various security frameworks with particular focus on policy-driven approaches. It gives a critical evaluation of each framework and their application to TrustCoM. Security management involves specification and deployment of access control policies, as well as taking measures for detecting and responding to security events, such as intrusion detection and response activities. When an event occurs, the security management action to be carried out depends on the organisation policy.

The frameworks introduced in this section cover *Access Control*, *Certificates and Delegated Models*, *Web Services Security and Grid Security*, *Information Flow* and *Adaptive Security.*

Access Control is concerned with granting access rights to subjects, to resources. The traditional access control decisions are based on individual names, however in VOs, this may not work because it may not be possible to know all the users and their credentials in advance to grant them access to resources. Furthermore, these traditional models do not address administration of jointly owned resources, which will be present in VOs. We present Role Based Access Control (RBAC), which is attracting increasing attention as it reduces the cost of administering security policies. Well-known policy languages with examples are presented to illustrate how they could be used to specify both access control activities, and other security management activities.

We present *Certificates and Delegated Models* with their management infrastructure, and illustrate how they deal with authenticating entities, authorisation and delegating rights to other entities, using public key certificates and assertions. Most of these frameworks are concerned with access control for unknown entities. Some of them provide a distributing infrastructure, and others are more localised.

Further in the section, we focus on *Web Services Security and Grid Security* frameworks. Most of these frameworks are designed to support interoperability on the web. TrustCoM framework will incorporate and enhance existing mature frameworks and therefore it is important to be aware of them.

We introduce *Information Flow*, and discuss how RBAC and other access control are limited in protecting the flow of information.

One of objectives of TrustCoM is to provide self- managed Adaptive Security, where security is not one-time task but an ongoing adaptable process. We briefly discuss what Adaptive Security entails, and its main components, namely monitoring, analysing and responding. We further discuss how policies can be used to specify Adaptive Security functionalities.

8.2 Role-Based Access Control (RBAC)

Although Role-Based Access Control (RBAC) is not directly concerned with policy specification, it has been accepted as a security model which permits the specification and enforcement of organisational access control policies. The fundamental concept on which RBAC is that permissions are associated with roles rather than users, thus separating the assignment of users to roles from the assignment of permissions to roles. Users acquire access rights by virtue of their role memberships, and they can be dynamically assigned or removed from roles without changing the permissions associated with their role. Multiple

users can be assigned to the same role and multiple roles can be assigned to the same users.

Although the concept of role has existed in a fairly similar form for a long time both in systems security and in Role Theory, the work presented by Sandhu[283] and at the first RBAC workshop[284] have prompted a renewed interest in this approach, which is now adopted to different degrees in many commercial products.

The main goal of RBAC goes beyond the concept of role and aims to simplify permission management in large organisations. To achieve this, roles must be combined in a structured way and permissions must be "reused". The most popular approach relies on role inheritance where senior roles such as team leader, or project supervisor inherit the permissions of junior roles such as employee, team member, etc. However other approaches such as assigning roles to other roles can also be found in the literature.

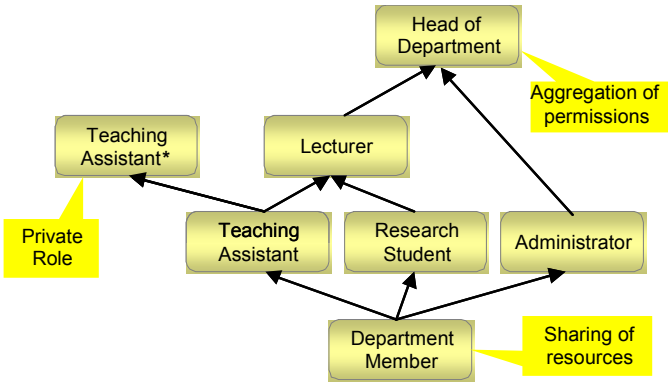


Figure 66 Role Hierarchies

A possible role hierarchy for an academic institution is described in Figure 66, which illustrates how role inheritance provides the sharing of resources through common lower level roles and the aggregation of permissions through inheritance of permissions to the higher level. However, such role inheritance hierarchies are not without shortcomings as there are numerous exceptions to the rule that senior roles inherit all the permissions of junior roles. Most notably, access to private files and permissions granted by virtue of a competency are not inherited. For example, the head of department does not usually inherit the access right of a system administrator. To accommodate such situations it is necessary to create private roles as shown in Figure 67, which group the permissions that are not inherited upwards in the hierarchy. Implementing an RBAC system with inheritance of permissions between roles considerably reduces the number of permissions in the system. However, in a distributed system it may also render access control checks, performed on each invocation, more complex since the inherited roles may be stored remotely and checking the inherited permissions may require several remote invocations. To avoid this increased complexity, a capability-based system may be more appropriate for RBAC since it shifts the responsibility for collecting the inherited permissions to the user (subject) system and this is done prior to the access control check.

Several constraints may apply to an RBAC model across its associations, between users and roles, roles and permissions or between roles themselves (inheritance)[285]. Amongst these, separation of duty constraints, which identify mutually exclusive sets of permissions that a user is not allowed to hold, have been the subject of the most intensive work.

[283] R.S. Sandhu, E.J. Coyne, H.L. Feinstein and C.E. Youman. "Role-Based Access Control Models", IEEE Computer, vol. 29, no.2, 1996, pp. 38-47.

[284] Proceedings of the First ACM/NIST Workshop on Role Based Access Control, Gaithersburg, ACM Press, 1995.

[285] Chen, F. and R.S. Sandhu, "Constraints for Role-Based Access Control", *Proc. 1st ACM/NIST Role Based Access Control Workshop*, Gaithersburg, USA, 1995.

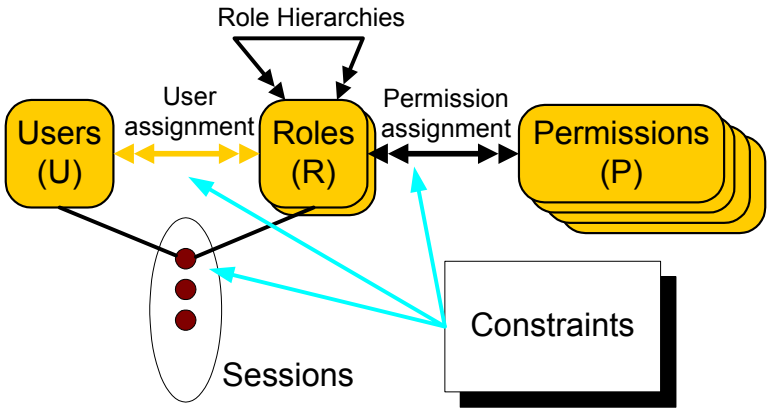


Figure 67 RBAC Model[283]

Figure 67 shows an overview of the RBAC model with the various relationships between users roles and permissions, and the constraints. The model presented in [283] also introduces the concept of sessions. A session groups the permissions from a selected number of roles, which the user may want active at a given moment in, time i.e., in a work session. The user can then use all access rights from the active roles in order to carry out his tasks.

The RBAC model presented above is the foundation from which many variations have developed. In particular, the team-based access control has developed as a means of simultaneously activating a set of related roles e.g., the surgeon, nurses and other personnel in an operating theatre.

Although, the concept of role is not new, the introduction of RBAC models has fostered a change from the traditional mandatory and discretionary access control models to a new framework where the access control policy is neither rigidly embedded in the implementation nor left to the owner of each resource but can be implemented on the basis of clearly specified organisational policies.

8.2.1 Application to TrustCoM

Roles could be used in TrustCoM to specify more general access control requirements in VO. Making use of roles simplifies the access control management, and speeds the access control decision. The use of inheritance relationships requires further work to handle the shortcomings discussed earlier such as handling access to private files and not inheriting the permissions of the system administrator.

8.3 Ponder

Ponder[286] is a declarative, object oriented language for specifying management and security policies. Ponder authorisation policies can be map onto various access control mechanisms for firewalls, operating systems, databases and programming languages such as Java. Ponder can be used to specify security management activities such as registration of users or logging and auditing events for dealing with access to critical resources. It supports obligation policies that are event-condition-action rules for policy based management of networks and distributed systems. One of the key concepts of the language include roles to group policies relating to a position in an organisation, relationships to define interactions between roles and management structures to define a configuration of roles and relationships pertaining to an organisational unit such as a department. Ponder supports delegation policies to cater for temporary transfer of access rights to agents acting on behalf of a client, and specifies both positive and negative authorisations.

8.3.1 The Ponder Language

Ponder supports access control by providing *authorisation*, *delegation*, *refrain* and *information filtering* policies that are described below.

Ponder uses the term *subject* to refer to users, principals or automated manager components, which have management responsibility. A subject accesses *target* objects (resources or service providers), by invoking methods visible on the target's interface. The granularity of protection for access control in Ponder is thus an interface method. Authorisations refer to methods in contrast to elementary access. References to both subject and target objects are stored within domains maintained by a domain service. Policies are specified in terms of domains. *Domains* provide a means of grouping objects to which policies apply and can be used to partition the objects in a large system according to geographical boundaries, object type, responsibility and authority or for the convenience of human managers. A domain does not encapsulate the objects but merely holds reference to objects. Domains are similar to directories and have been implemented using an LDAP service. An advantage of specifying policy scope in terms of domains is that objects can be added and removed from the domains to which policies apply without having to change the policies.

Authorisation Policies define what activities a member of the subject domain can perform on the set of objects in the target domain. These are essentially access control policies, to protect resources and services from unauthorized access. A positive authorisation policy defines the actions that subjects are permitted to perform on target objects. A negative authorisation policy specifies the actions that subjects are forbidden to perform on target objects. Authorisation policies are implemented on the target host by an access control component. The language provides reuse by supporting the definition of policy types to which any policy element can be passed as formal parameter. Multiple instances can then be created and tailored for the specific environment by passing actual parameters as shown in the authorisation policy below:

```
type auth+ PolicyOpsT (subject s, target <PolicyT> t) {
action load(), remove(), enable(), disable() ;
}
inst auth+
switchPolicyOps=PolicyOpsT(/NetworkAdmins,/Nregion/switches);
inst auth+ routersPolicyOps=PolicyOpsT(/QoSAdmins, /Nregion/routers);
```

[286] N. Damianou, N.Dulay, E. Lupu and M. Sloman,. *The Ponder Policy Specification Language*. in *Workshop on Policies for Distributed Systems and Networks*. 2001. Bristol, UK.

The two instances of PolicyOpsT allow members of /NetworkAdmins and /QoSAdmins to execute the actions on policies within the /Nregion/switches and /Nregion/routers domains respectively.

Figure 68 Example of Ponder Authorisation Policy

Policies can also be declared directly without instantiating a pre-existent policy type:

```
inst auth– /negativeAuth/testRouters {
        subject /testEngineers/trainee ;
        action performance_test() ;
        target /routers ;
}
```

Trainee test engineers are forbidden to perform performance tests on routers. The policy is stored within the/negativeAuth domain.

Figure 69 Example of Ponder Direct Policy Declaration

Information Filtering policies can be used to transform the values of the input parameters in an action and the information returned from the action. For example, a location service might only permit access to detailed location information, such as a person is in a specific room, to users within the department. External users can only determine whether a person is at work or not.

Delegation policies permit subjects to grant privileges, which they possess (due to an existing authorisation policy), to other subjects called grantees to perform an action on their behalf e.g., passing read rights to a printer spooler in order to print a file. A delegation policy specifies the authority to delegate; it does not control the actual delegation and revocation of access rights. Note that, a delegation policy is always associated with an authorisation policy, which specifies the access rights that can be delegated. Negative delegation policies forbid delegation of certain actions. See [286] for further details and examples of these policies.

Refrain policies define the actions that subjects must refrain from performing (must not perform) on target objects even though they may actually be permitted to perform the action. Refrain policies act as restraints on the actions that subjects perform and are implemented by subjects. Refrain policies have a similar syntax to negative authorisation policies, but are enforced by subjects rather than target access controllers. They are used for situations where negative authorisation policies are inappropriate because the targets are not trusted to enforce the policies (e.g., they may not wish to be protected from the subject).

```
inst refrain testingRes {
        subject s=/test-engineers ;
        target /analysts + /developers ;
        action discloseTestResults() ;
        when s.testing_sequence = "in-progress" ;
}
```

This refrain policy specifies that test engineers must not disclose test results to analysts or developers when the testing sequence being performed by that subject is still in progress, i.e., a constraint based on the state of subjects. Analysts and developers would probably not object to receiving the results and so this policy is not a good candidate for a negative authorisation.

Figure 70 Example of Ponder Refrain Policy

The Ponder **composite policies** (groups, roles, relationships and management structures) allow structured, reusable specifications, which cater for complex, large-scale enterprises. They provide the ability to group policies and structure them to reflect organisational structure, preserve the natural way system administrators operate or simply provide reusability of common definitions. This simplifies the task of policy administrators.

A ***group*** is a packaging construct to group related policies together for the purposes of policy organisation and reusability and is a common concept in most programming languages. There are many different potential criteria for grouping policies together – policies may reference the same targets, relate to the same department or apply to the same application. It can contain zero or more basic policies, nested groups and/or meta-policies in any order. Meta-policies specify constraints on the policies within the scope of the group. Reusability can be achieved by specifying groups as types, parameterised with any policy element or system attribute, and then instantiating them multiple times. As an example, policies related to the login process can be grouped together since they would always be instantiated together.

Roles provide a semantic grouping of policies with a common subject, generally pertaining to a position within an organisation such as department manager, project manager, analyst or ward-nurse. Specifying organisational policies for human managers in terms of manager positions rather than persons permits the assignment of a new person to the manager position without re-specifying the policies referring to the duties and access rights of that position. A role can also specify the policies that apply to an automated component acting as a subject in the system e.g. security manager agent. Organisational positions can be represented as domains, which are called subject domains, and are associated with roles. A role is thus the set of authorisation, obligation, refrain and delegation policies with the subject domain of the role as their subject. Roles can include a group of policies in which all the policies have the same subject, which is defined implicitly as illustrated below.

```
type role /mgmtInfo/roles/ServiceEngineer (CallsDB callsDb) {
      inst oblig serviceComplaint {
            on     customerComplaint(mobileNo) ;
            do     t.checkSubscriberInfo(mobileNo, userid) ->
                   t.checkPhoneCallList(mobileNo) ->
            investigate_complaint(userId);
            target t = callsDb ; // calls register
      }
      inst oblig deactivateAccount { . . . }
      inst auth+ serviceActionsAuth { . . . }
      // other policies
}

inst role /mgmtInfo/roles/Area1ServiceEng =
     mgmtInfo/roles/ServiceEngineer(Area1CallsDB)@/SD/area1ServiceEng;
```

The role type ServiceEngineer models a role in a mobile telecommunications service, which is responsible for responding to customer complaints and service requests. The role type is parameterised with the calls database, a database of subscribers in the system and their calls. The obligation policy serviceComplaint is triggered by a customerComplaint event with the mobile number of the customer given as an event attribute. On this event, the subject of the role must execute a sequence of actions on the calls-database in order check the information of the subscriber whose mobile number was passed in through the complaint event, check the phone list and then investigate the complaint. Note that the obligation policy does not specify a subject as all policies within the role have the same implicit subject.

Figure 71 Example of Ponder Role Policy

Ponder allows specialisation of policy types through the mechanism of inheritance. When a type extends another, it inherits all of its policies, may add new policies and overrides policies with the same name. Inheritance is only defined for composite policy types.

Managers acting in organisational positions (roles) interact with each other. A ***relationship*** groups the policies defining the rights and duties of roles towards each other. It can also include policies related to resources that are shared by the roles. It thus provides an abstraction for defining policies that are not part of the role specifications, but are part of the interaction between the roles. The syntax of a relationship is very similar to that of a role but a relationship can include definitions of the roles participating in the relationship. Participating roles can also be defined as parameters within a relationship type definition as shown below.

```
type rel ReportingT ( ProjectManagerT pm, SecretaryT secr) {
      inst oblig reportWeekly {
            on timer.day (“monday”) ;
            subject secr ;
            target pm ;
            do mailReport() ;
      }
// . . . other policies
}
```

The ReportingT relationship type is specified between a ProjectManager role type and a Secretary role type. The obligation policy reportWeekly specifies that the subject of the SecretaryT role must mail a report to the subject of the ProjectManagerT role every Monday. The use of roles in place of subjects and targets implicitly refers to the subject of the corresponding role.

Figure 72 Example of Ponder Relationship Type

Many large organisations are structured into units such as branch offices, departments, and hospital wards, which have a similar configuration of roles and policies. Ponder supports the notion of management structures to define a configuration in terms of instances of roles, relationships and nested management structures relating to organisational units. For example a management structure type would be used to define a branch in a bank or a department in a university and then instantiated for particular branches or departments. A management structure is thus a composite policy containing the definition of roles; relationships and other nested management structures as well as instances of these composite policies. In this respect Management Structures provide a model very close to a VO.

8.3.2 Deployment Model

The deployment architecture supports the instantiation, distribution and life cycle management of policies, as well as their enforcement by automated enforcement components. A detailed description of the deployment architecture can be found in [287]. An overview of the policy deployment architecture is illustrated in Figure 73. It includes three supporting services: a domain service, a policy service, and an event service. The *Policy Service* acts as the interface to policy management, it stores compiled policy classes, creates and distributes new policy objects and otherwise supports policy management actions not provided elsewhere in the model. The *Domain Service* manages a distributed hierarchy of domain objects and supports the efficient evaluation of subject and target sets at run-time. Each domain object holds references to its managed objects but also references to the policy objects that currently apply to the domain. In concept, the domain service is similar to a directory service such as LDAP, with extensions to allow changes to the membership of a directory to be distributed to interested parties, e.g. via events. However, the domain service can also be implemented by database systems. An *Event Service can be used to* collect and compose events from the underlying systems and from the managed

[287] N. Dulay, E.Lupu, M. Sloman and N. Damianou. *A Policy Deployment Model for the Ponder Language*. in *Proc. IEEE/IFIP International Symposium on Integrated Network Management*. 2001.

objects in the system, and forwards them to registered policy management agents triggering obligation policies.

After a policy object is instantiated, it can be loaded into its enforcement agents, and once loaded, it can be enabled causing its enforcement agents to actively implement it. An enabled policy can be disabled and later re-enabled, or disabled and then unloaded, removing it from its enforcement agents. Unloaded (i.e. dormant) policies can either be reloaded or deleted.

For obligation policies, Policy Management Agents (PMAs) register with an event service to receive relevant events generated from the managed objects of the system. On receiving an event, the PMA queries the domain service to determine the target objects used in the obligation method and performs the policy actions, provided no constraint or refrain policy prevents the action. Composite policies map to objects that elaborate the instances within them while delegation policies map to authorisation policy objects that allow grantors to invoke the delegate operation on the policy service, with respect to a specific authorisation policy.

The Ponder compiler also generates for each policy, an enforcement class that the policy object distributes to its enforcements agents. The enforcement class provides the specific implementation behaviour needed to enforce the policy at the enforcement agent. For authorisation policies, each target object has a single access controller (AC), which enforces all the authorisation policies for the target object. Each AC (e.g. firewall, operating system) normally enforces many authorisation policies and protects many different target objects. Obligation and refrain policies are handled in a similar way to authorisation policies but the distribution is based on the subject set. The policy object evaluates the subject set and passes to each subject, a copy of the enforcement object for the policy. For obligation and refrain policies the members of the subject set are policy management agents that enforce the policies.

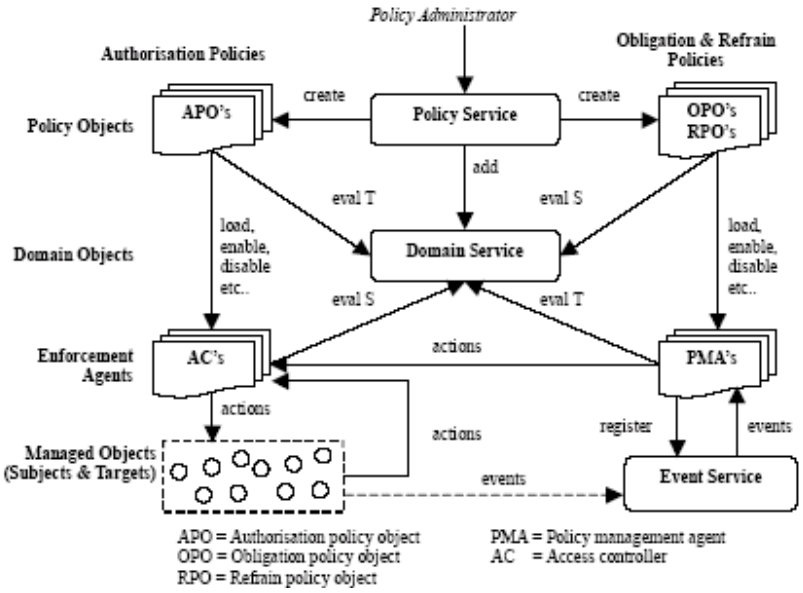


Figure 73 Ponder Deployment Model

8.3.3 Ponder Toolkit

As part of the Ponder framework, a complete toolkit has been developed to support the users of the language. Available components of the toolkit are further discussed in the

following section. The Ponder toolkit can be downloaded under a *GNU Lesser General Public License* from: http://www-dse.doc.ic.ac.uk/Research/policies/ponder.shtml.

8.3.3.1 Domain Browser

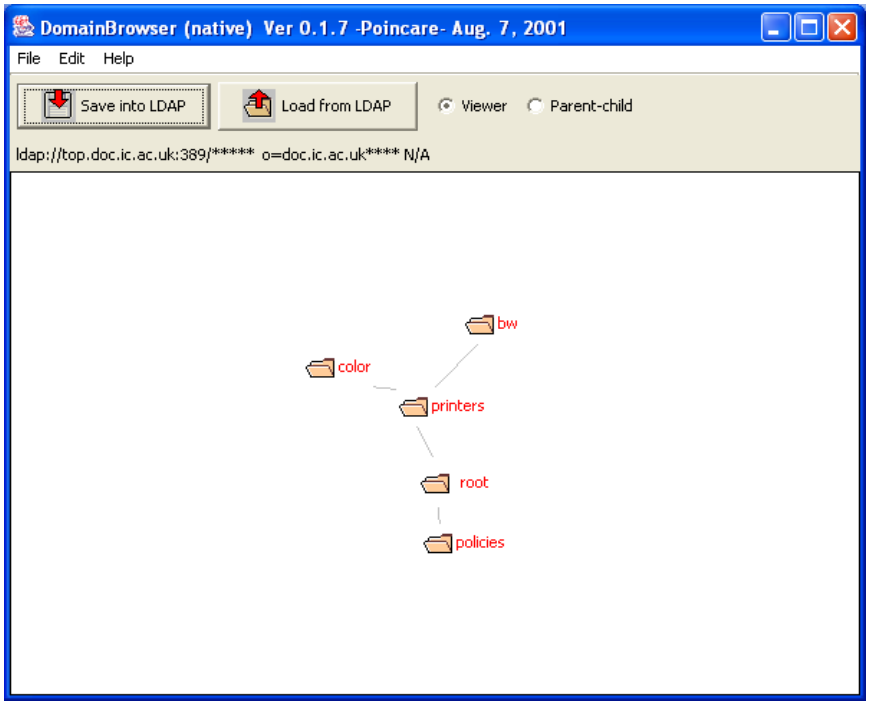


Figure 74 Ponder Domain Browser

The PONDER domain browser provides a common user interface for all aspects of an integrated management environment. It can be used to group or select objects for applying policy, to monitor them or to perform management operations, although the current implementation only supports policy management.

The domain browser reads data from the domain service and provides a graphical trees structured view of the directory structure. A domain structure is created using the domain browser as shown in Figure 74. Administrators can use the domain browser to manage the domain structure, group objects into domains to apply a common policy, modify or create new objects. Objects can represent users, roles, network components or manager agents.

8.3.3.2 Compiler Framework

The PONDER compiler maps policies to low-level representations suitable for the underlying system or into XML for transfer around the network.

Authorization policies can be mapped onto a variety of heterogeneous security platforms and mechanisms, such as firewalls, operating systems security, database security and Java authorisations. For example, if servers used to store data in the AI research group are Linux based while servers in other departments are Windows 2000 based, then appropriate code will be generated based on the type of server. Dedicated code generators (compiler back-ends) must be implemented to translate the PONDER specification into the desired format. The compiler framework is designed for extensibility with custom code generators without recompiling the system.

8.3.3.3 Policy Editor

The policy editor, Figure 75, is integrated with both the domain browser and the PONDER compiler and provides an easy to use development environment for specifying, reviewing and modifying policies.

Templates can be used to create policies easily. The domain browser can be invoked to select the subject and target domains for policies. Existing policies and policy types can be selected from the directory with the aid of the domain browser, loaded into the editor, modified, recompiled and stored back to the directory. Code generators added to the compiler framework, are accessible and can be enabled from within the editor to select the type of code to be generated.

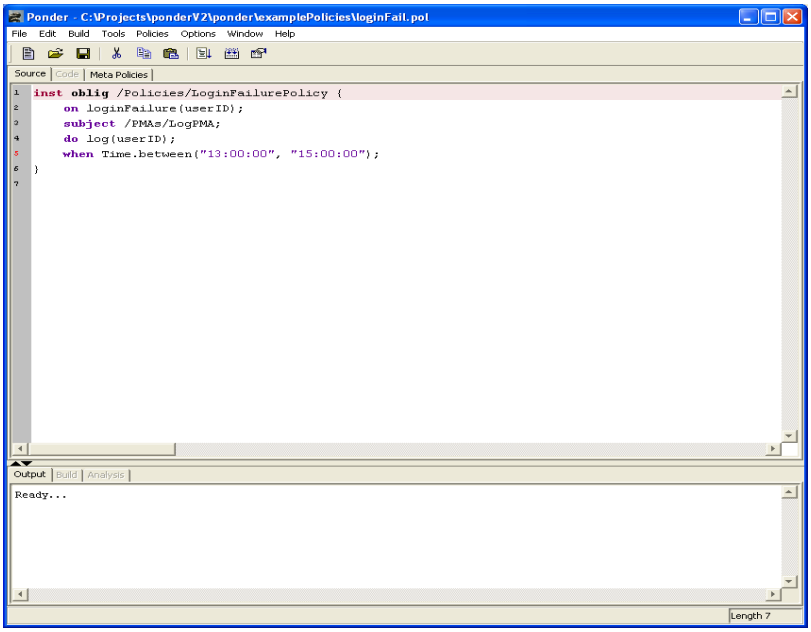


Figure 75 Policy Editor

8.3.3.4 Management Console Tool

Ponder toolkit comes with a management console tool for dynamically managing policies. The tool has two main views:

In the **Policy Objects View**, a policy instance can be selected from the directory (using the domain browser) and loaded into the management console. Similarly, if a domain is selected all policy instances under that domain will be loaded into the management console in an expandable tree-navigator. Policies can then be selected and *loaded*, *unloaded*, *enabled* or *disabled* as needed. Details about the selected policy are displayed including the policy-status. When a new backup policy for a specific user is specified, a policy administrator uses the management console to select the policy from the directory, load it and enable it. Multiple management consoles could manage the same domain of policy objects, but LDAP does not support concurrency control.

In the **Enforcement Components View,** enforcement components can be selected and information about the policies loaded into them is displayed in a tabular format. A **Command-line Window** can be used to type single-line commands to the PONDER compiler. This allows interactive instantiation of policy types.

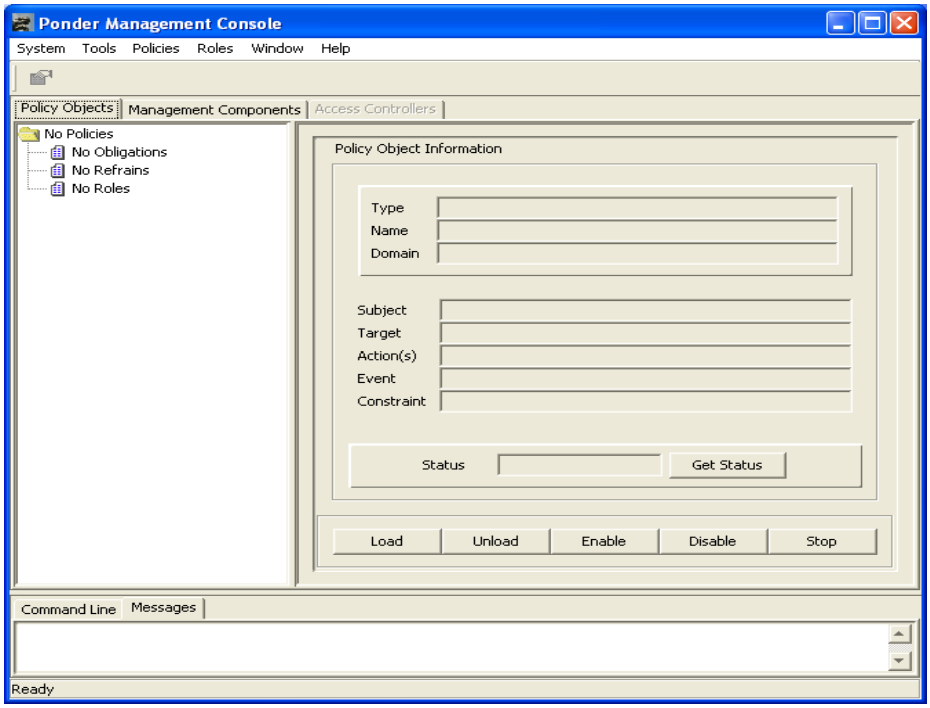


Figure 76 The Management Console Tool

8.3.4 Advantages and Disadvantages

Ponder is more flexible and extensible than most of the existing policy languages as it can be used to specify wide range of policies including authorisation, information filtering, refrain and delegation policies for specifying access control decisions, and obligation policies to specify management actions. Ponder provides a uniform means of specifying policy relating to a wide range of management applications network, storage, systems, application and service management. Ponder is not just a policy language, it provides a framework supporting all the phases of policy life cycle.

Ponder policies can be shared among heterogeneous security systems and platforms but this requires agreement on interfaces. The Ponder composite policies (groups, roles, relationships and management structures) allow structured, reusable specifications, which cater for complex and large-scale enterprises. A complete open source toolkit has been developed for the specification and management of policies, including a compiler and editor and a policy management tool.

Current work involves conflict analysis and refinement of Ponder policies, which require transforming policies into an event calculus representation. Most policy languages require considerable computing expertise to specify security policies. Ponder is designed for simpler policy specification and direct readability.

8.3.5 Application to TrustCoM

In VOs, various institutions will control resources, each institution as well as the VOs will need to define their own access control requirements to protect their resources. Ponder language can be used to specify these access control decisions. Ponder can be used to support the dynamic nature of the VOs. It handles dynamic changes to domain structures and dynamic adaptation of the system to such changes. Domain membership is dynamic and objects can be added to, or removed from a domain as needed.

Ponder can also be used to support trust management life cycle in dynamic environments. It can act as a lower level of both specification and implementation for SLA or trust specifications. It is possible to refine trust specifications to Ponder security policies. For example, Ponder authorisation policies can be used to specify the conditions under which trust relationships can be established.

8.4 **XACML**

The eXtensible Access Control Markup Language (XACML)[288] is an XML specification for defining access control policies for information access over the Internet, developed by the Organisation for the advancement of Structured Information Standards (OASIS), and became a standard in 2003. Traditional application specific access control policy languages do not support-distributed policies, thus are not suitable for defining distributed security policies, which can be shared across different applications. XACML is designed to overcome this problem by separating policies from application and providing a standard for expressing authorisation and entitlement policies, which can be shared by heterogeneous security systems.

The XACML policy language is used to express access control policies written in form of XML for expressing requirements to access particular resources. In addition to policy language, XACML specifies a request and response protocol for describing queries about a particular request and decisions made regarding the request (responses).

A typical XACML usage scenario is: a subject (e.g. person, workstation) wants to access a protected resource. Access to this resource is controlled by a Policy Enforcement Point (PEP). When the subject makes a resource request, the PEP re-formalise the request using the XACML request language describing the requester, action, resource and other relevant information, and sends it to a Policy Decision Point (PDP) for making the authorisation decision. PDP will fetch the applicable policies, which are written in the XACML policy language from a policy store and determine the authorisation decision according to the XACML rules for evaluating policies. The authorisation decision, response will be expressed using the XACML response language and deliver the decision to the PEP, which can then permit or deny access to the requester.

The XACML policy language has two top-level elements, Policy and Policy-Sets. The Policy element is the smallest element PDP can evaluate, defining a single access control policy. The PolicySet aggregates other PolicySet elements or Policy elements or policy references to external policies. A PolicySet may have multiple policies and a single Policy may have multiple rules, it is possible for each of them to evaluate to different access control decisions. XACML uses a collection of Combining Algorithms[289] to reconcile the multiple decisions. The Combining Algorithms represent various ways to define a single outcome; Deny-Overrides, Permit-Overrides, First Applicable and Only-One.

A Policy is composed of a *Target*, a set of *Rules* and an optional set of Obligation elements that apply to a request. The T*arget* element is used to find the applicable policy or rules to a given request. A policy *Target* specifies the conditions that the requesting Subject, Resource and Action must meet for a PolicySet, Policy or Rule to be applicable to the requested resource. The Target provides a built in method to indexing and looking up policies. Once the Policy is found, the next step is to evaluate its rules. The main components of a XACML Rules are a *Target*, an *Effect* and *Condition*s. *arget* discussed above selects the applicable policies. A C*ondition* is a Boolean decision function to decide if *Effect* is applicable. The *Conditions* tests the relevant attributes within a Policy, and each condition evaluates to a True or False outcome. All the outcomes are combined together,

[288] *OASIS eXtensible Access Control Markup Language TC:* http://www.oasis-open.org/committees/xacml/repository/.

[289] *eXtensible Access Control Markup Language (XACML) Version 1.0*, OASIS Standard.

and yield an *Effect* of Permit, Deny or Indeterminate (indicating an error, the information was not enough to derive an exact answer).

8.4.1 Advantages/ Disadvantages

The XACML standard can replace application specific languages as it provides a single language to define and enforce access control policies in a variety of environments, and locate policies in distributed environment. This means security administrators no longer need to rewrite their policies in other languages. XACML supports Role Based Access Control, and handles conflicting policies. XACML supports integration with SAML and XML Digital Signature.

The policy language is very wordy with long URLs, and it is likely that tools will be used to generate policies, as it can be too complex for human users to define and understand them. Its complexity enables doing most things in more than one way. XACML is a fairly new standard, and the tool support for building and administering XACML applications remains restricted although growing.

8.4.2 Application to TrustCoM

The XACML can be used within TrustCoM project to share authorisation policies and security information across virtual organisations. As discussed earlier, SAML assertions hold security information and XACML defines how to use this security information. The XACML standard can be integrated with the SAML to access resources. For example XACML receives a SAML request (containing authentication information about the individual making the request), to determine if access should be granted to a resource based on rule sets or policies that are defined by the provider. The resource providers can use XACML to specify the rules on issuing SAML assertions.

8.5 **SAML**

The Security Assertions Mark-up Language (SAML)[290] is an XML based framework to support the exchange of security information between business partners over the Internet, developed by the Organisation for the advancement of Structured Information Standards (OASIS). The purpose of SAML is to enable interoperability between different applications, which require security services. SAML supports Single-Sign-On (SSO) to multiple domains, the users are authenticated in one domain and their credentials are passed to other partner domains without having to re-authenticate. This way, SAML provides the technology to allow a business to securely communicate with users from other partners such as vendors, suppliers, customers, and etc.

An SAML document is composed of one or more assertions made by the SAML authorities stating certain facts about a subject, e.g. a user. The current SAML framework supports three kinds of security assertions: A*uthentication*, *Attribute* and *Authorisation decisions*, however it is possible to extend SAML to make other kinds of assertions. An *Authentication* assertion states subject S was authenticated by means M at this time T. An *Attribute* assertion states subject S is associated with the set of attributes A with values B (for example, that Alice is associated with attribute “TrustCoM” with value “Research Scientist”). An *Authorisation decision* assertion states what the subject S is entitled to do on resource R (for example, whether to permit the user to access a web service). The SAML authorities namely authentication authorities, attribute authorities or policy decision points issue the assertions. SAML can be used to make assertions about credentials however it does not check or revoke the credentials.

[290] *OASIS Security Services TC*: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

An example of SAML authentication assertion, stating Alice was originally authenticated using a password mechanism at 2004-04-02T17:05:17 is shown below:

```
<saml:Assertion
      xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
            MajorVersion="1" MinorVersion="1"
            AssertionID="biuEZCGxcGiF4gIkL5PNltwU7duY1az"
            Issuer="www.eu-trustcom.com"
            IssueInstant="2004-04-02T17:05:37">
            <saml:Conditions
                  NotBefore="2004-04-02T17:00:37"
NotOnOrAfter="2004-04-02T17:10:37"/>
      <saml:AuthenticationStatement

      AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
            AuthenticationInstant="2004-04-02T17:05:17">
            <saml:Subject>
                  <saml:NameIdentifier
                        NameQualifier=www.eu-trustcom.com
                        Format="http://www.customformat.com/">
                        uid=alice
                  </saml:NameIdentifier>
                  <saml:SubjectConfirmation>
                        <saml:ConfirmationMethod>
                        urn:oasis:names:tc:SAML:1.0:cm:artifact-01
                        </saml:ConfirmationMethod>
                  </saml:SubjectConfirmation>
            </saml:Subject>
      </saml:AuthenticationStatement>
</saml:Assertion>
```

SAML defines a request and response protocol that consists of XML-based messages where a client can request a specific assertion or make authentication, attribute, and authorisation decisions queries from a SAML authority and get a response from them. These messages can be bound to many different underlying communication and transport protocols. The current SAML framework defines a binding, a way to transport SAML request and responses by sending them within SOAP exchange messages over HTTP. Another important concept of SAML is *Profiles*, defining the flow of assertion and protocol messages that describe how SAML can be used for a particular purpose.

8.5.1 Advantages and Disadvantages

Previous Single Sign-on Services solutions were designed for closed systems, and they did not offer interoperability. SAML provides an interoperable open standard for exchanging security information. The SAML assertions provide statements regarding the security events such as authentication that have already occurred, as well as attribute certificates which mean the issuing parties, can choose the methods to authenticate users (e.g. X.509 certificate or password). SAML has been designed to integrate with XACML, but the integration is not complete yet. A use of XACML with SAML in grid environment is described in section 8.18.6 Cardea. It is important to bear in mind that SAML is still under development and continuously developing, and it may affect the applications using older versions of the specification.

8.5.2 Application to TrustCoM

One of the pre-requisites of collaborative environment such as virtual organisations is to build trust relationships. In such dynamic environments, trust is established by exchanging security information about the entities such as X.509, attribute certificates, reputation and etc. SAML provides a common syntax for the exchange of attribute and authorisation

information. The SAML assertions can be used to provide single sign on services within partner virtual organisation.

8.6 PolicyMaker

A fundamental problem remains, how to provide access control for unknown entities. In this section, and following sections, we introduce some early work done to authorise strangers, i.e. entities that are not in the resource administrator domain.

Public key certificate frameworks such as PGP and X509 are used for authentication, certifying the identity associated with a public key. These frameworks do not bind access rights to the certified identity (subject of the certificate), and leave the authorisation decisions outside the certificate framework. After certificate is successfully associated with a user, applications need to handle the access rights of the user using another framework. PolicyMaker[291] is a trust management system, providing a unified solution for authorisation problem rather than handlingthe problem indirectly via authentication and access control.

PolicyMaker appears as a query engine, which can either be built into the applications (through a linked library) or run as a separate "daemon" service. The engine accepts as input the local policy, received credentials and an action string (query), and returns a simple yes/no answer or the details (required credentials) that would make the proposed action acceptable. The form of a query is:

key_1, key_2…key_n **REQUESTS** ActionString

Queries are requests to find out whatever a public key (or a sequence of public keys) is permitted to carry out a particular action. *ActionStrings* are application specific, which describe a trusted action requested by a public key(s). The action strings are evaluated only by the calling application, and it is the responsibility of the application to determine the semantics of action strings. Applications call PolicyMaker after they verify their own credentials and translate them into PolicyMaker credentials. It is responsibility of the application to verify the public keys.

PolicyMaker interprets queries based on local policies and credentials, called 'assertions'. A policy tells who is trusted to do what (actions), and credentials pass on trust to another entity (e.g. a public key). The difference between policies and credentials is that policies are unconditionally trusted whereas credentials are signed by other entities and the signatures must be verified before the credentials can be used. PolicyMaker defines the assertions in terms of *filters*, predicates associated with public keys. Filters accept or reject action strings based on the trust relationships. The format of an assertion is:

Source **ASSERTS** AuthorityStruct **WHERE** Filter

Source refers to the source of the assertion; it can be the local policy or the public key of the trusted third party in the case of signed assertions. *AuthorityStruct* refers to the public key or keys to whom the assertion applies. Often the authority structure is just a single public key, but other complex authority structures are possible (e.g. at least three keys are required for an action to take place). *Filter* is the predicate the action strings must satisfy for the assertion to hold. Filters associate public keys with the assertions. PolicyMaker filters are interpreted programs that are run within a wrapper. PolicyMaker supports three filter languages: a regular expression system, AWK and a macro language, but other languages can be added. Filter programs take as inputs the current action strings and the environment containing information about the current context (e.g., date, time, application name, etc). Filters can use the environment to enforce contextual constraints such as expiration dates. The interpreter for a filter language is external to PolicyMaker itself however the name of the language is

[291] M. Blaze, J.F.a.J.L. *Decentralized Trust Management*. in *IEEE Symposium on Security and Privacy*. May 1996. Oakland, CA.

given in the assertion and must be known by anyone who needs to use the assertion. PolicyMaker ignores the assertions written in unknown filter language.

```
Alice_ Policy
        ASSERTS gp_key WHERE  patient = Alice
```

Applications will call PolicyMaker for advice on authorisation. For example, the calling application could send the following statement to PolicyMaker to decide whether bob_key (Bob) is allowed to use Alice's medical record for research purposes:

```
gp_key
        ASSERTS bob_key WHERE
        patient = Alice && use = research
```

To check if Bob can use Alice's record, the external application will ask the PolicyMaker:

```
bob_key REQUESTS  "use = research"
```

8.6.1 Advantages and Disadvantages

PolicyMaker provides a general trust management solution by separating the trust management mechanism from application specific policy (which is specified by each application). PolicyMaker uses the credentials to prove that a requested action complies with the local policy (known as compliance-checking mechanism[292]). Generally the following steps are taken for each request:

1. Generate an action string (query) for the action to be considered.
2. Get the application specific "credentials" (e.g. X509 certificates) that are needed to support the actions string, and verify the signatures.
3. Translate the application specific credentials into PolicyMaker credentials.
4. Input the action string, the policy and credentials to the PolicyMaker.
5. Use PolicyMaker evaluation to perform or not perform the action.

PolicyMaker provides some level of privacy as public keys are bound to predicates that describe the actions they are trusted, rather than names of public key holders. The major weakness of PolicyMaker is trust establishment and credential verification is left up to the calling application.

8.6.2 Application to TrustCoM

PolicyMaker manages access control by binding public keys directly to access rights through the use of certificates signed by trusted third parties. PolicyMaker was the first proposed trust management solution, and although it introduces the basic principals, its tools and ability to work in a broader framework is left entirely to the calling application.

8.7 Keynote

KeyNote[293, 294], a successor of PolicyMaker retains the same design principles, it directly authorise actions instead of dividing the authorisation task into authentication and access

[292] M. Blaze, J.F.a.M.S. *Compliance Checking in the PolicyMaker Trust Management System*. in *Financial Cryptography*. 1998. Anguila

[293] M. Blaze, *Using the KeyNote Trust Management System*. 1999, AT&T Research Lab, Updated March 2001.

control. In contrast with PolicyMaker, KeyNote has two additional design goals; standardization and ease of integration[294]. KeyNote shifts some of the responsibility of calling-application to the trust management engine. Public key verification is carried by the trust management engine in KeyNote (this is done by the application in PolicyMaker). PolicyMaker allowed any choice of language for writing the policies and the credentials, which did not support interoperability. KeyNote requires that credentials and policies be written in a specific assertion language, which allows simpler integration with the compliance checker. The caller application passes to the KeyNote engine a list of credentials, policies, requester public keys and an "Action Environment". The "Action Environment" consists of a list of attribute/value pairs, containing all information considered relevant to the request and crucial for the trust decision. These attribute/value pairs must reflect the security requirements of the application accurately. After the evaluation process the KeyNote engine returns an application-defined string back to the application, a simple reply may say something like "authorised".

The following examples of KeyNote policy and assertion are taken from[294]:

```
A.    Authorizer: "POLICY"
      Licensees: "RSA:abc123"

B.    KeyNote-Version: 2
      Local-Constants: Alice="DSA:4401ff92"
                       Bob="RSA:d1234f"
      Authorizer: "RSA:abc123"
      Licensees: Alice || Bob
      Conditions: (app_domain == "RFC822-EMAIL") &&
                  (address ~= "^.*@keynote\\.research\\.att\\.com$");
      Signature: "RSA-SHA1:213354f9"
```

As in PolicyMaker, KeyNote assertions can be local policies or credentials that describe the trusted actions permitted by the owners of specific public keys. The difference between policies and credentials is policies (with keyword POLICY in the *Authorizer* field) is unconditionally trusted and requires no signature.

Policy A says it unconditionally authorises RSA key *abc123* for all actions. Credential B says RSA Key *abc123* trusts Alice (RSA key *4401ff92*) and Bob (DSA key *d1234f*) as certification authorities for the *keynote.research.att.com* domain. The *Authorizer* field refers to the principal issuing the assertion. The *Licensees* field refers to the principal(s) authorised by the assertion. The *Conditions* field gives the 'conditions' under which the *Authoriser* trusts the *Licensees* to perform the action. The programs in KeyNote are encoded in the *Conditions* field, and are used to test the action environment variables. The tests used by KeyNote are string comparisons, numerical operations and comparisons, and pattern matching operations[295]. The *Signature* field identifies a signed assertion and gives the encoded digital signature of the principal issuing the assertion (i.e. *Authorizer*).

KeyNote does not support credential revocation services, but credentials can be written to have an expiry data by including a date test in the predicate.

---

[294] M. Blaze, J.F., J. Ioannidis and A. Keromytis, *The KeyNote Trust Management System,Version 2*. September 1999.

[295] M.Blaze, J.F., J. Ioannidis and A. D. Keromytis. *The Role of Trust Management in Distributed Systems Security*. in *Secure Internet programming: security issues for mobile and distributed objects*: Springer-Verlag

8.7.1 Advantage/Disadvantages

KeyNote is a general trust management system, which provides authorisation based on public key. KeyNote provides advice to the calling application and it does not provide any mechanisms to enforce policies. The caller application is responsible for selecting the appropriate credentials and policy assertions with which to run a particular query. Like PolicyMaker, one of the limitations of KeyNote is that it cannot directly write an assertion such as if someone has these attributes, then she has this permission.

One of the lacking properties of these framework is that there is no trust metrics to assess trustworthiness of the various entities in dynamic environments using evidence and risk factors.

8.7.2 Application to TrustCoM

Current KeyNote model may not be sufficient for TrustCoM framework, and requires further work to establish dynamic trust relations and to reason about them. However, the underlying concepts of KeyNote can be extended to provide a well-defined trust management system for virtual organisations.

8.8 REFEREE

Rule-Controlled Environment For Evaluation of Rules and Everything Else (REFEREE)[296] is a trust management system for web applications. REFEREE provides a language for stating trust policies, and supports a general policy evaluation mechanism for web clients and servers. Like KeyNote and PolicyMaker, REFEREE is a query engine, giving advice to web applications.

REFEFEE has three primitive data types; programs, state lists and tri-values. REFEREE interprets policies and credentials as programs. Both credentials and policies can return both tri-values and statement lists.

Tri-value can be true, false or unknown. “True” means an action may be taken because there are sufficient credentials for the action to be approved. “False” means an action may not be taken because there are sufficient credentials to deny the action. “Unknown” means the system was unable to find sufficient credentials either to approve or to deny the requested action.

A statement list is a collection of assertions. Each statement in REFEREE is a two-element structure consisting of some content and a context for that context. Both context and the content of a statement are each arbitrary s-expressions; the context determines how the content is to be interpreted, and the interpretation of the context is depends on the agreement between the calling application and REFEREE. A statement list is an unordered list of REFEREE statements.

Policies defining actions are programs which return true or false depending on whether the available statements are enough to prove an action can be taken or not, or unknown if evaluation can not be made. A policy could reject downloading of code and give statements explaining whether the code is known to be malicious or local machine is too loaded to allow downloads. Sometimes it can be desirable for policies to call other policies for judgement on some point, for example Bob may let his children to view any web page that Alice permit her children to view. Second, evaluation of a particular request may involve dangerous activities, e.g. network access, and invocation may require such dangerous actions to be executed from within REFEREE policies. This way the dangerous activities can be controlled by the policy.

[296] Y. Chu, J.F., B. LaMacchia, P. Resnick and M. Strauss. *REFEREE: Trust Management for Web Applications*, AT&T Laboratories, MIT and W3C.

A credential is a program that examines the initial statement it receives and derives additional statements[296]. The new statement supplied by credential may be conditional on the initial statements or it could refer to environment factors (e.g. space on the local hard disk). Credentials programs can indicate the success of the execution (a tri-value) and also return a list of statement.

REFEREE uses profiles-0.92 language[296] for writing policies and uses PICS labels[297] for stating credentials. PICS labels are used to state properties of Internet resources (e.g., application code has been virus-checked). Profiles-0.92 has a construct, *invoke*, for invoking (calling) another REFEREE program, and invoking "load-label" programs to look for PICS labels, either embedded in documents or retrieved over the network from a label bureau. When the PICS labels are found, they are parsed and written as REFEREE statements. Profiles-0.92 provides tri-value generalisation of the Boolean operators AND, OR and NOT, in addition to operators *true-if-unknown* and *false-if-unknown*. The language also provides a mechanism for patter matching that examines the statement list for statements of a particular form.

The following example taken from[296] says view any URL with a PICS label in the "musac" system (a PICS rating system) with "s<2". The policy is about required use of saxophones in music, and abbreviated "s" stands for saxophones.

```
invoke "load-label" STATEMENT-LIST URL "http://www.musac.org/"
(EMBEDDED))
 (false-if-unknown
        (match
        (("load-label" *)
        (* ((version "PICS-1.1") *
         (service "http://www.musac.org/") *
               (ratings (RESTRICT < s 2)))))
       STATEMENT-LIST))
```

REFEREE assumes that the publishers are honestly declaring the number of saxophones.

The above policy has two steps. The first step invokes the `load-label` to find and download labels for the given URL; any labels found will be put on the statement list. Then it runs a pattern-matcher over the modified statement list, looking for any label using the rating service from *"http://www.musac.org/"* and with a saxophone (s) rating less than 2. If no musac label with an s dimension is found, then it returns *unknown*, and, if it finds a matched label, it returns *true* or *false* depending on whether or not the associated value is less than 2.

8.8.1 Advantages/Disadvantages

REFEREE can be used to write policies about trust delegations, cryptographic keys, signature checking and anything else. It does not make arbitrary decisions but controls operations under policy control. REFEREE provides a mechanism for making access decisions relating to web applications, however it assumes web publishers are honest about declaring the content of their web sites.

8.8.2 Application to TrustCoM

The REFEREE trust management framework puts all security access decisions under direct policy control. Its underlying conceptual framework is useful, although the implementation is not widely used.

[297] *Platform for Internet Content Selection (PICS)*. 10/22/2003, W3C: http://www.w3.org/PICS/

8.9 TPL

Trust Establishment Framework[298] developed by IBM provides access to strangers by assigning roles to them based on their certificates issued by trusted third parties. The current architecture uses X.509 v3 certificates but other certificates types are supported.

Trust Establishment extends the existing Role Based Access Control mechanism as illustrated in the diagram below. The system consists of a Trust Establishment module and a Trust Policy Language (TPL). On receiving a X.509 certificate, the Trust Establishment verifies it, and assigns the subject of the certificate to a role based on a given role-assignment policy set by the owner of the resource and on the roles of the issuer of the certificate. The certificate may not bind to user's identity but could state attributes of the user (e.g. the user is an employee of IBM), which can be used to map the user to a role.

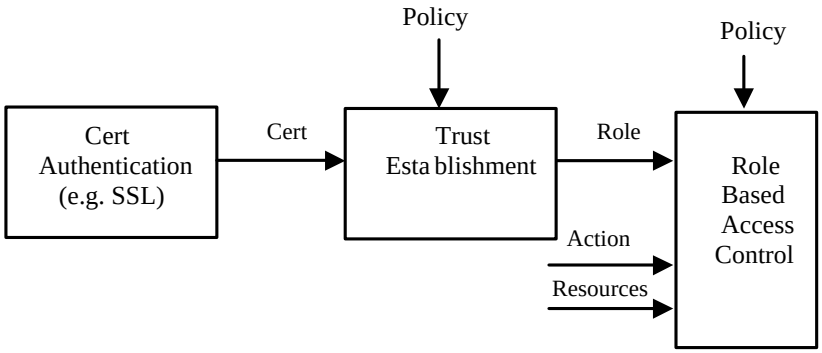


Figure 77 Trust Establishment and RBAC

Certificates contain two components for handling missing certificates. The certificate presented by the client contains an address, a pointer to look for additional certificates related to the issuer (this is to map the issuer to some group), and to check the certificate revocation lists to verify certificate validity. The certificate contains another address, a pointer to look for more certificates related to the subject. This is required in case the subject is unable to provide all the certificates (for example it may have limited capacity on its accessing device).

TPL is used to define policies to map entities to roles, using logical rules and defines the set of actions the role is permitted to do. A role in TPL is similar to RBAC, a group of entities that can represent a specific organizational unit (e.g. consultants, system analysts and etc.). XML is used to define TLP policies to ease interoperability.

A policy in TPL consists of the following tags[298]:

- The <*GROUP*> tag is used to define the groups (roles). The only attribute of <*GROUP*> is the name of the group. The scope of the <*GROUP*> tag contains one or more <*RULE*> tags, each defining single rule for membership in the group.
- Within the <*RULE*> tags, there could exist multiple <*INCLUSION*> tags and at most one <*FUNCTION*> tag.
- The <*INCLUSION*> tag specifies the certificate that must exist for a role to hold.
- The <*FUNCTION*> tag is used to define necessary condition on the certificate.

The following example, taken from [298] illustrates a rule for adding a new hospital to the Hospitals group:

[298] A. Herzberg, Y.M., J. Michaeli D. Naor and Y. Ravid. *Access Control Meets Public Key Infrastructure Or: Assigning Roles to Strangers*. in IEEE Symposium on Security and Privacy. 2000: IEEE Computer Society Press.

```
<GROUP NAME="Hospitals">
<!---- hospital recommended by at least 2 hospitals --->
<RULE>
<INCLUSION ID="reco" TYPE="Recommendation"
  FROM="hospitals" REPEAT=”2”></INCLUSION>
<FUNCTION>
    <GT>
<FIELD ID="reco" NAME="Level"></FIELD>
<CONST>1</CONST>
    </GT>
</FUNCTION>
</RULE>
</GROUP>
```

TPL allows the system administrator to define flexible rule based on attributes of digital certificates. Each role is associated with one or more rules defining how a certificate holder can become a member. Requesting entities only need to satisfy one of the rules in order to join the role. The Trust Establishment Framework does not require user's identity in order to map it to a role; other attributes can be used for mapping. The framework collects missing certificates to reach a decision.

8.9.1 Advantages/Disadvantages

TPL provides access to entities from other domains by assigning roles to them based on their certificates issued by trusted third parties. It assumes trust relationships are monotonic, and the emphasis is on access control decisions.

8.9.2 Application to TrustCoM

Often members of a VO may need to access resources owned by other members for the purpose of performing a collaborative task. Resource owners need to control access to their resources, and combining trust management with RBAC which is supported in TPL, offers more flexibility as access control policies can be written in terms of roles rather than individuals. RBAC can also be used to define the organisational structure of the virtual organisation.

8.10 SPKI/SDSI

The SPKI/SDSI approach[299,300,301] shares many views with the trust management approaches of PolicyMaker and KeyNote. SPKI and SDSI were started independently. Later, they merged into a collaborative effort SPKI/SDSI 2.0. Both approaches are motivated by the inadequacy of the public-key infrastructures based on global name hierarchies, such as X.509 and Privacy Enhanced Mail[302].

[299] C. Ellison. SPKI/SDSI certificates. http://world.std.com/~cme/html/spki.html, Aug. 2001

[300] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen. Simple public key certification. Internet draft, work in progress. http://www.ietf.org/ids.by.wg/spki.html, June 1999.

[301] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, and R. L. Rivest. Certificate chain discovery in SPKI/SDSI. Journal of Computer Security, 9(4):285* 322, 2001.

[302] S. T. Kent. Internet privacy enhanced mail. Communications of the ACM, 36(8):48*60, Aug. 1993.

Contrary to KeyNote and PolicyMaker, SPKI/SDSI does not does not define an application-independent inference engine, since the authors in[303] state the processing of certificates and related objects to yield an authorization result as the province of the application developer.

SPKI/SDSI adopts SDSI's localized naming scheme[304,301]. In SDSI, there are principals and local identifiers. Principals are public keys, and therefore unique. Each principal has its own local name space. Names are formed by linking principals and local identifier. A local name is a principal followed by a local identifier, while an extended name is defined as a principal followed by at least two identifier. A principal can use arbitrary local names and two principals might use the same name differently. SDSI allows the name spaces to be linked. Linking of name spaces allows principals to use definitions another principal has made. SPKI/SDSI has two kinds of certificates: *name certificates* and *authorisation certificates*.

Principals can issue *Name certificates* to bind local names to public keys. The binding is one to many, i.e. a single name can be bind to one or more keys. Each principal have their own local name space containing the names it defines, and therefore these names are not unique in global name space.

Authorisation certificates came originally from SPKI. An authorisation certificate delegates a resource specific permission from a principal representing a resource owner as issuer or one of its delegatees to another principal (i.e. the entity being authorised in this certificate) stated in the certificate's subject field. The trust evaluation in SPKI consists of finding a delegation chain that delegates the authority from the original issuer to the principal encoded in the subject field of the final certificate of the chain. The capabilities being delegated are determined by the intersection of all the capabilities along the chain. The main content of a SPKI certificate is illustrated in Figure 78:

- ***Issuer:*** The public key that signs the certificate.
- ***Subject:*** The public key that receives the rights given with a certificate.
- ***Delegation bit:*** Delegation field, a Boolean value indicating if the issuer allows the subject to re-delegate the rights to other subjects.
- ***Authorisation:*** Stating the set of access rights the subject will have.
- ***Validity dates:*** Defines the validity period of the certificate by an issuer.

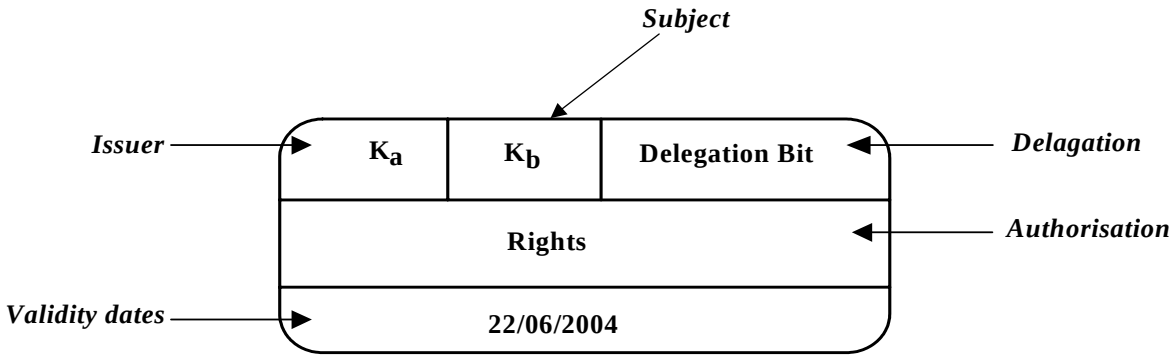


Figure 78 An Authorisation Certificate structure

[303] C. Ellison, B. Frantz, B. Lampson, R. Rivest, and T. Ylonen. SPKI certificate theory IETF RFC 2693. http://www.ietf.org/rfc/rfc2693.txt RFC 2704, Internet Engineering Task Force, Sept. 1999.

[304] N. Li. Local names in SPKI/SDSI. In Proceedings of the 13th IEEE Computer Security Foundations Workshop (CSFW-13), pages 2*15. IEEE Computer Society Press, 2000.

8.10.1 Advantages and Disadvantages

SPKI/SDSI provides a simple mechanism to authorise entities to perform particular operations on protected resources. Access Control Lists (ACLs) are created, and require modification in rare events. In SPKI/SDSI, each public key is a certificate authority. There is no trusted root authority, and it is not a hierarchical global infrastructure like PKI. SPKI/SDSI assumes that certificates have relatively short lifetime (expiration time), and the revocation is a very rare event. Therefore, it does not support certificate and key revocation lists like PKI, which are maintained by a central unit. It suggests mechanism like a Key Compromise Agent (KCA) or a suicide bureau (SB)[305] to support revocation.

8.10.2 Application to TrustCoM

SPKI/SDSI can be used within VO to authorise entities for a period of time to access protected resources. Authorisation certificates can be used by entities to delegate permissions to other entities to carry out certain operations on their behalf.

8.11 Hybrid PKI Model

The Hybrid PKI Model[306,307,308,309,310,311,312,313,314] has been developed to be used for specifying and enforcing credential-based access control policies in distributed computing systems. The hybrid model integrates, unifies and extends previous approaches, in particular X.509[315] dealing with free properties (characterizing attributes of entities, including group membership) and SPKI/SDSI (see section 8.10) dealing with bound properties (promises to get a specific service, including capabilities). The Hybrid PKI Model allows protocols that enable servers to permit access for strangers as clients only based on their approved characterizing properties. Speaking more technically: a client can characterize himself by showing properties that are assigned to him by potentially other entities and that are encoded in certificates and credentials; a server can either base access decisions directly on such properties, or first

[305] M. Burnside, D. Clarke, S. Devadas, and R. Rivest. Distributed SPKI/SDSI-Based Security for Networks of Devices. To appear, December 2002.

[306] J. Biskup and Y. Karabulut: A Hybrid PKI Model for Access Control in Distributed Systems with Mediation. March 2004, submitted to a journal for publication.

[307] J. Biskup and Y. Karabulut: A Hybrid PKI Model with an Application for Secure Mediation. In 16th Annual IFIP WG 11.3 Working Conference on Data and Application Security, pages 271-282, Cambridge, England, July 2002. Kluwer Academic Press.

[308] J. Haller, Y. Karabulut and P. Robinson: Integrating the Hybrid PKI Model into Trust Relationship Lifecycle for Dynamic Virtual Organizations. Workshop on Towards a Trust & Contract Management Framework Enabling Trustworthy Business Processing in Dynamic Virtual Organisations within eChallenges e2004, Vienna, October 2004, to appear.

[309] J. Haller, Y. Karabulut and P. Robinson: Investigating Trust Management Approaches for Trustworthy Business Processing for Dynamic Virtual Organizations. Special Session on Security and Privacy in E-Commerce within the 7th International Conference on Electronic Commerce Research, INFOMART, Dallas, June 2004, to appear.

[310] Y. Karabulut: Towards a Next-Generation Trust Management Infrastructure for Open Computing Systems. Security and Privacy Workshop within the 2nd Pervasive Computing Conference, Vienna, April 2004, to appear.

[311] Y. Karabulut: Developing a Trust Management Based Secure Interoperable Information System. Special Session on Security and Privacy in E-Commerce within the 6th International Conference on Electronic Commerce Research, INFOMART, Dallas, October 2003.

[312] Y. Karabulut: Implementation of an Agent-Oriented Trust Management Infrastructure Based on a Hybrid PKI Model. In 1st International Conference on Trust Management, LNCS 2692, pages 318-331, Crete, Greece, May 2003.

[313] J. Biskup and Y. Karabulut: Mediating Between Strangers: A Trust Management Based Approach. In 2nd Annual PKI Research Workshop, pages 80-95, Gaithersburg, Maryland, USA, April 2003.

[314] Y. Karabulut: Secure Mediation Between Strangers in Cyberspace, Ph.D. Thesis, University of Dortmund, 2002.

[315] X.509: http://www.ietf.org/html.charters/pkix-charter.html.

convert such properties into potentially conditional promises to be returned to the client for later use and then examine these promises on request; property conversion can also be delegated to mediating agents.

In all cases, at some step, whether explicit or implicit, a property conversion of free properties into bound properties takes place. Here the term "bound" is intended to express some kind of promise for a service and thus a relationship of the property holder to a server, whereas the term "free" is intended to indicate a feature of the property holder alone. The interpretation of properties as free or bound depends on the context, or more technically on the security domains where the properties are assigned. In order to cross domain borders, again property conversions might be necessary, and manageable indeed. As testbed for the design the authors analysed protocols dealing with some form of property conversion for *secure mediated information systems*[316,317,318]. Thereby they identified the need of different kinds of security policies, including those for confidentiality, (pure) property conversion, delegation, and reconfirmation, respectively.

8.11.1 Advantages/Disadvantages

The authors argue that many applications require to use both X.509 and SPKI/SDSI in a unifying manner, and to link them by so-called property conversion. Though the proposed Hybrid PKI Model must not be understood as the ultimate answer to all PKI- related challenges, the business advantage of a comprehensive model should be clear: It provides a framework for seamless interoperation between heterogeneous and autonomous security domains, such that organizations can broaden their customer and collaborator base. The current model does not treat certificate revocation. Instead the authors assume appropriate mechanisms to handle this issue. Furthermore, the model needs to be extended for supporting negotiations[319].

8.11.2 Application to TrustCoM

An organization acting as a VO planner need to quickly and effectively identify, select, and negotiate with the best partners to achieve the best portfolio of suppliers to optimize price, risk, and performance. In particular, organizations may not know all marketplaces (or catalogs) where potential qualified suppliers are registered, and even if they know them, the VO planner may not be in a position to understand and verify the certified properties of the suppliers. In order to achieve the lowest total cost for an upcoming product development effort, the VO planner needs to find the optimal portfolio of suppliers that satisfy its supply requirements. By expanding qualified supplier base, a VO planner can also maximize the competition across suppliers and thus achieve the lowest total cost.

More concretely: *What is the problem?* Whatever composed services a VO planner has to offer, it may aim at contacting a wide range of potential qualified service providers, which are in general unknown in advance and may belong to heterogeneous and autonomous security domains. *Why is the problem a problem?* The conceptual challenge arising in this situation concerns how a VO planner and supplier, potentially acting in different security domains, reach a common understanding of the meaning of properties, and how they establish trust that are claimed properties are valid indeed. On the one hand, characterizing properties (e.g. certified and recognized supplier) are assigned to suppliers in their autonomously operating

---

[316] J. Biskup and Y. Karabulut: Mediating Between Strangers: A Trust Management Based Approach. In 2nd Annual PKI Research Workshop, pages 80-95, Gaithersburg, Maryland, USA, April 2003.

[317] C. Altenschmidt, J. Biskup, U. Flegel and Y. Karabulut: Secure Mediation: Requirements, Design and Architecture. Journal of Computer Security, 11(3):365-398, 2003.

[318] Y. Karabulut: Secure Mediation Between Strangers in Cyberspace, Ph.D. Thesis, University of Dortmund, 2002.

[319] Yu, T., Winslett, M., and Seamons, K. 2003. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. ACM Transactions on Information and System Security 6, 1 (Feb.), 1-42.

security domains, in principle without knowing their later usage. On the other hand, the VO planner independently defines its security policies in terms of these properties. *What is the solution?* In such situations the VO planner wish to be assisted to determine potentially qualified service providers. To reach potentially eligible providers, a VO planner could use a specific software agent, which could map between the VO planner's property-based security policy and suppliers' characterizing properties which have been asserted by some trusted parties. *Why is the solution a solution?* Electronic business transactions will involve asserted commitments, properties, etc. from many parties and the participants of such transactions will, in general, not be in a position to understand or manage everything that is involved. To identify and select qualified suppliers, which might belong to remote security domains, organizations acting as VO planners will need to trust mediating software agents as trusted partners having the required domain expertise as well as the relationships with the potential clients.

8.12 **Permis**

The EC Privilege and Role Management Infrastructure Standards Validation (PERMIS)[320] project provided a framework for the creation and management of a distributed authorisation infrastructure using X.509 standard attribute certificates (ACs) to hold users roles. This Privilege Management Infrastructure (PMI) used by PERMIS defines a complete set of processes required to provide an authorisation service. It defines an authorisation API in Java, a SAML protocol interface, and a Policy DTD for the construction of authorisation policies.

The PERMIS software is freely available for research and education purposes as part of the US National Science Foundation's Middleware Initiative (NMI) software release. (OpenSAML and Globus Toolkit are also part of the NMI release.)

8.12.1 Description

PERMIS implements a distributed role based access control (RBAC) infrastructure using X.509 attribute certificates to store users roles (see Figure 79). PERMIS implements hierarchical RBAC as defined in the NIST RBAC model [284], whereby superior roles inherit the privileges of subordinate roles.

[320] Privilege and Role Management Infrastructure Standards Validation: http://sec.isi.salford.ac.uk/permis/

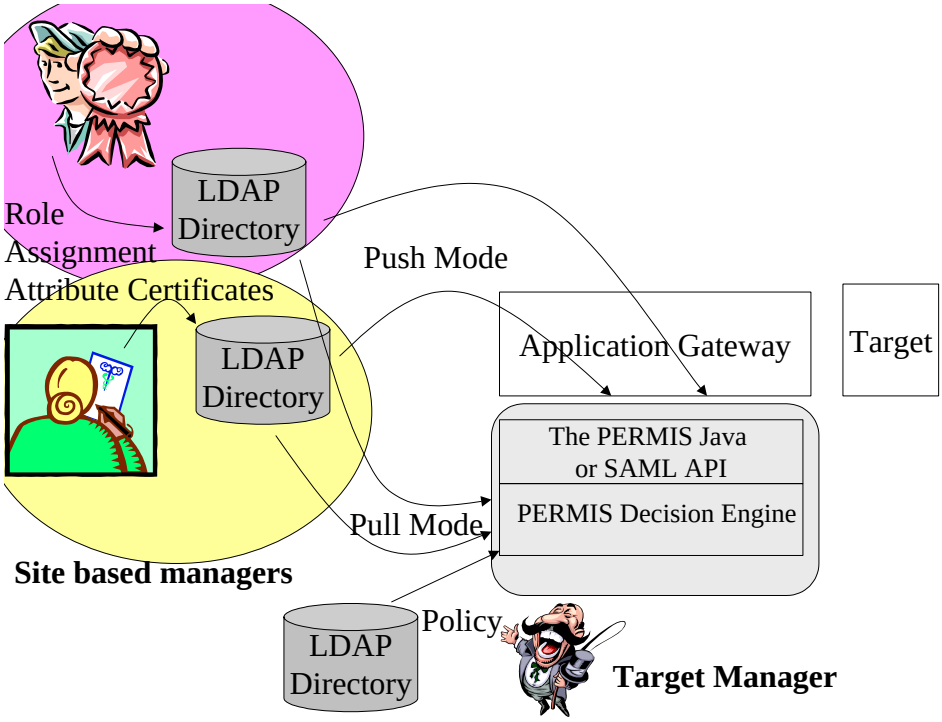


Figure 79 The PERMIS Distributed Authorisation Infrastructure

Administrators at various sites in a VO may allocate roles (or attributes) to their users, in the form of X.509 attribute certificates, and store these in a local (or remote) LDAP directory. The manager of a VO resource (the Target Manager in Figure 79) determines the authorisation policy for the target resources under his control. This authorization policy is written in XML, according to a defined DTD, and stored in an X.509 attribute certificate in the target manager's entry in the local LDAP directory. Part of this policy – the role assignment policy – says which remote site based managers are trusted to allocate which roles to which groups of users. For example, manager Jane from organisation A is trusted to allocate the role of project manager to users within the computing dept of organisation A. Thus the target manager has complete overall control over who is allowed to access the resources under his control, but is able to delegate the allocation of roles to trusted site based managers within the VO. Another part of the policy – the target access policy – says which permissions are given to which roles, and under which conditions. For example, a project manager is allowed to update project plans on the local server between the hours of 9am and 5pm. A user friendly GUI is available to target managers to help them create their PERMIS authorisation policies.

When a user attempts to access a target resource, the application gateway (typically a web server) will trap the user's request; will authenticate the user in an application dependent way (e.g. X.509 client certificates, username/password, or SSO using Shibboleth or Liberty etc.); and will then call the PERMIS decision engine to find out if the user is authorised to perform the requested action on the target or not. The PERMIS decision engine may be called via either its Java API, or a SAML protocol interface. In the former case, the PERMIS engine is built as part of the application gateway, in the latter case the PERMIS decision engine runs as a stand-alone server, and may be called by many different application gateways.

PERMIS can operate in either push or pull mode. In push mode, the user pushes his X.509 ACs to the application gateway along with his request. In the pull mode, PERMIS pulls the user's X.509 ACs from the various LDAP directories that it is told about.

Access control decisions are made according to the authorisation policy currently in force, the roles currently valid for the user, the target being accessed, the action being requested, and environmental variables such as the time of day.

PERMIS supports two flavours of the SAML protocol. It can be called via standard SAML as implemented by the OpenSAML software, which is part of the US NMI software release. Alternatively, PERMIS can be called via the Grid authorisation profile of SAML[321], as implemented in Globus Toolkit release 3.3.

The PERMIS Java API consists of two methods, GetCreds and Decision, and a constructor. The constructor, used to build the PERMIS decision engine, takes arguments representing the Target Manager (called the Source of Authority (SOA) in X.509), which is the root of trust for authorization, a policy identifier and a list of LDAP locations from which attribute certificates can be retrieved.

The GetCreds method is called when the user initiates the first call to the target and is used to fetch and validate all the roles (i.e. attribute certificates) of the user. At each action request, the parameters of that action and the identity of the target are passed to the Decision method. Based on the user's valid roles the Decision method then determines whether or not the given action is allowed. The Decision function indicates its outcome by returning either a Granted or Denied message. The Source of Authority may dynamically impose a new authorization policy on the domain at any time, by causing the application gateway to construct a new PERMIS decision engine.

For a full description of the PERMIS framework and a complete overview of the API see http://sec.isi.salford.ac.uk/permis/.

8.12.2 Ongoing additions to PERMIS

Current projects at Salford are in the process of adding the following features to PERMIS. All of the following features are planned for future US NMI releases and will be completed within the life of the TrustCoM project.

a) Linking authorization to the level of authentication undergone by the user. This is a joint project with the University of Manchester, and will make use of additions to SAMLv2.0, which will allow the authentication service to pass in the Authentication Statement details about the level of authentication undergone by the user. The PERMIS authorization policy will be enhanced so that the target manager can dictate which levels of authentication are required in order to access which resources.

b) Integrating PERMIS with Shibboleth. The Shibboleth infrastructure (see section 8.14) provides a mechanism for single sign on and the retrieval of user attributes, based on the SAML protocol. The Shibboleth PERMIS integration project will allow Shibboleth to pass the attributes to PERMIS, and for PERMIS to make an authorization decision based on the policy and the user's attributes.

c) Dynamic delegation of authority. Currently the target manager is responsible for statically delegating authority to remote site based managers, and must enter their names into the authorization policy. When dynamic delegation is implemented, a site based manager will be able to dynamically delegate to one of his subordinates, without the need to enter his name into the authorization policy. Of course, the target manager will still be able to control the amount of dynamic delegation that is allowed.

d) Separation of duties (static and dynamic). Static separation of duties will be enforced by enhancing the PERMIS policy to state which roles are mutually exclusive. If a user possesses more of the mutually exclusive roles than is allowed by the policy, the user

[321] Von Welch, Frank Siebenlist, David Chadwick, Sam Meder, Laura Pearlman. "Use of SAML for OGSA Authorization", Jan 2004, Available from https://forge.gridforum.org/projects/ogsa-authz

will be denied access. Dynamic separation of duties will be enforced by keeping a historical log of events, and checking that the same user has not invoked different roles for different session, thereby trying to circumvent the static separation of duties.

e) Distributed dynamic authorization. Currently PERMIS can operate with a single standalone decision engine (called via SAML), which can co-ordinate all accesses to all targets throughout a VO. However this poses a potential performance bottleneck. Alternatively, PERMIS can operate with the Java API being built into each application gateway, and each using the same policy (if desired). This has optimum performance and throughput, but lacks the ability to co-ordinate decision making between multiple application gateways. What is needed is the ability to co-ordinate decision making between application gateways, as well as to adapt policies to the constituency of the VO, without effecting performance. An additional requirement is to dynamically evolve the policy as the VO changes. This challenge is currently being addressed by building a higher level expert system that will generate PERMIS policies on demand, and distribute them to all the application gateways in a VO, as and when required.

8.12.3 Comparison of Permis and Akenti

Akenti (see section 8.18.1) is an authorisation infrastructure from the Lawrence Berkeley National Laboratory in the USA. PERMIS is an authorization infrastructure from the University of Salford, UK, and is a product of the EC funded PrivilEge and Role Management Infrastructure Standards validation (PERMIS) project [320]. Both the Akenti and PERMIS Authorisation Infrastructures are trust management infrastructures according to the definition of Blaze[322], and have the 5 components necessary for this, which are:

i) A language for describing `actions', which are operations with security consequences that are to be controlled by the system.

ii) A mechanism for identifying `principals', which are entities that can be authorized to perform actions.

iii) A language for specifying application `policies', which govern the actions that principals are authorized to perform.

iv) A language for specifying `credentials', which allow principals to Adelegate authorization to other principals.

v) A ‘compliance checker', which provides a service to applications for determining how an action requested by principals should be handled, given a policy and a set of credentials.

Both infrastructures have similar architectures[323,324]. This comprises the compliance checker, called the Akenti server by Akenti[325], and the Access Control Decision Function (ADF) by PERMIS (after the ISO Access Control Framework[326]). Both have a gateway controlling user access to resources, called the Resource Gateway by Akenti and the Application Gateway by PERMIS. Both of them write their policies in XML, and store their policies in certificates. Both of them can store their user credentials as certificates in LDAP directories. Hence on

[322] Blaze, M., Feigenbaum, J., Ioannidis, J. “The KeyNote Trust-Management System Version 2”, RFC 2704, Sep 1999

[323] Johnston, W., Mudumbai, S., Thompson, M. “Authorization and Attribute Certificates for Widely Distributed Access Control,” IEEE 7th Int Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE), Stanford, CA. June, 1998. Page(s): 340 -345 (see also http://wwwitg.lbl.gov/security/Akenti/)

[324] D.W.Chadwick, A. Otenko. “The PERMIS X.509 Role Based Privilege, Management Infrastructure”, Proc 7th ACM Symposium On Access Control Models And Technologies (SACMAT 2002), Monterey, USA, June 2002. pp135-140.

[325] Akenti Homepage: http://www-itg.lbl.gov/security/Akenti/

[326] ITU-T Rec X.812 (1995) | ISO/IEC 10181-3:1996 “Security Frameworks for open systems: Access control framework

the face of it, the Akenti and PERMIS authorisation infrastructures seem to be almost identical.

However at the implementation level Akenti and PERMIS are very different. Akenti is written in C++, Permis in Java. The Akenti compliance checker can be called either via a function call in the gateway or as a standalone server via TCP/IP, whereas the PERMIS compliance checker is either invoked as a Java object in the gateway or called via a SAML protocol[327]. PERMIS credentials are built according to the latest X.509 standard[328], whereas Akenti credentials are built in a proprietary format [325]. Akenti requires the user to be PKI enabled and to present an X.509 public key certificate (PKC) at authentication time, whereas PERMIS is authentication agnostic and leaves it up to the application to determine what type of authentication to use. Whilst both PERMIS and Akenti policies are written in XML, their DTDs are very different[329,330]. Furthermore, Akenti ignores the XML in its policy certificates and only uses the Base64 encoded section at the end (which means that the XML and Base64 sections could be different). PERMIS policies are held in one policy X.509 Attribute Certificate (AC), whereas Akenti policies are hierarchical and distributed between proprietary Policy Certificates and Use-Condition Certificates. Akenti has concentrated on classical access control lists (discretionary access controls) whereas PERMIS has implemented role based access controls. Therefore at a practical level there are a significant number of differences between the two infrastructures.

8.12.4 Advantages/Disadvantages

The PERMIS API is Java based allowing its deployment across a wide range of platforms and business environments. It uses the established X.509 attribute certificate structure to guarantee the security of its transactions and role based access decisions. The framework has been deployed and tested across a range of environments demonstrating its applicability to a range of solutions and its flexibility in adapting to differing business environments.

8.12.5 Application to TrustCoM

The PERMIS project has defined a complete and flexible framework for the construction of role based access control infrastructures, the policy DTD for which is hosted on the xml.org homepage. TrustCoMs aim of providing a ``framework enabling the definition and secure enactment of collaborative business processes within Virtual Organisations'' fits with the underlying philosophy and design of the PERMIS project. By defining a framework within which disparate organisations can securely share access to functionality and applications, allowing partners to collaborate on projects outside the scope of their individual enterprises, the PERMIS framework can go some way to fulfilling the aims of the TrustCoM project.

8.13 Delegent

Delegent is an authorisation server based on research done at SICS. The primary focus of Delegent is to decentralise management of authorisations in order to improve efficiency and

---

[327] Von Welch, Frank Siebenlist, David Chadwick, Sam Meder, Laura Pearlman. "Use of SAML for OGSA Authorization", Jan 2004, Available from https://forge.gridforum.org/projects/ogsa-authz

[328] ISO/ITU-T Rec. X.509(2000) The Directory: Authentication Framework

[329] Mary R. Thompson, S. Mudumbai, A. Essiari, W. Chin. “Authorization Policy in a PKI Environment”, Proceedings of the First Annual PKI Workshop, Dartmouth College, April 2002, pages 137-149. see www.cs.dartmouth.edu/~pki02

[330] D.W.Chadwick, A. Otenko. “RBAC Policies in XML for X.509 Based Privilege Management” in Security in the Information Society: Visions and Perspectives: IFIP TC11 17th Int. Conf. On Information Security (SEC2002), May 7-9, 2002, Cairo, Egypt. Ed. by M. A. Ghonaimy, M. T. El-Hadidi, H.K.Aslan, Kluwer Academic Publishers, pp 39-53.

flexibility. The decentralisation is based on a delegation model. Another goal of the model is to guarantee some degree of control in the spread of authorisations by delegation. This is done by means of constraints on how delegations may be performed.

8.13.1 Description

Delegent is an authorisation server whose main functionality is to provide answers to access queries and to manage who is authorised to administer the policy. Delegent offers an XML-based API to perform access queries or to update the policy. Delegent controls policy updates, so only authorised updates can be performed. Delegent is not a policy enforcement point and Delegent is not for authenticating users.

The Delegent model is based a recursive definition of an authorisation. An authorisation can be an access level permission, which will grant the right to perform an access on some object. The access level permissions are based on a user, object, method model, similar to XACML. Only positive permissions are available, so there is no need for conflict resolution.

An authorisation can also be an administrative level authorisation, which is the right to grant another authorisations. The possible authorisations that can be created with the administrative authorisation are described in the form of an authorisation with constraints, hence the recursion. These administrative authorisations make it possible for Delegent to control updates to the policy.

To decentralise administration of authorisations, high-level managers/administrators are granted administrative authorisations. The high-level managers can issue either access level permissions, or delegate the administrative rights to lower level managers.

Delegent makes a difference between an access level permission and an administrative authorisation, and neither implies the other. This is useful for instance in outsourcing scenarios, where another organisation can be granted the right to manage some resources, without the possibility to use the resources themselves.

Authorisations are removed either by letting them expire or revoking them. Delegent supports several different kinds of revocation depending on the needs of the situation.

The authorisations for a given resource form delegation chains, which must be rooted at the source of authority of the resource to be valid. The source of authority is defined by means of an authorisation with a special form. The delegation chains are shaped by the structure of the organisation, which uses the resource. One use of this structure is in what we call authority resolution, which is implemented in Delegent. Authority resolution has to do with overrides of denied access. Delegent supports emergency override of a denied access. Such overrides should be audited, but in a large organisation there will be no single entity with authority over all resources, so it is not evident who should perform this audit. Authority resolution is a mechanism that finds from the structure of delegation, who the authorities of a given resource are.

8.13.2 Comparison of Delegent

Delegent is a reference implementation of Privilege Calculus[331]. The idea behind Privilege Calculus is to support decentralized management of authorizations in dynamic environments. Although Privilege Calculus is influenced by ideas in Trust Management, originally defined by Blaze et al.[332], its goal is somehow different from the goal of Trust Management approach. The main difference between the Privilege Calculus and the Trust Management models is the interpretation of the notion of “delegation”. In trust management

[331] B. Sadighi Firozabadi, M. Sergot and O. Bandmann, Using Authority Certificates to Create Management Structures, in proceedings of *Security Protocols, 9th International Workshop,* Cambridge, UK, April 2001.

[332] M. Blaze, J. Feigenbaum and J. Lacy. "Decentralized Trust Management." *IEEE Symposium on Security and Privacy*, Oakland, CA. May 1996.

and most of the other approaches for authentication and authorization management, delegation is used as the means for impersonation or sometimes described in terms of "speaks-for" relation, whereas in Privilege Calculus delegation is the means of creating access rights and delegation rights.

The main characteristic feature of the Privilege Calculus is the possibility of decoupling an access right from the right to delegate it. Issuing an access right to an agent does not imply that the agent necessarily obtains the delegation right for that, and at the same time issuing a delegation right to an agent does not imply that the agent also gets the access right, or even the delegation right to create that access right.

Beside the differences in the underlying model, Delegent is different from Permis, Akenti and the existing implementations of the Trust Management approach, as it is not an authorisation infrastructure but only an authorization server. Currently, Delegent does not support any standard formats for representing authorisations and delegations. It has its own XML API for dealing with authorization and delegation information. Applications use Delegent as a decision support system for their access right decisions as well as secure updates of access rights and delegation rights.

8.13.3 Advantages/Disavantages

The biggest advantage of Delegent is that it offers a unique model for decentralised administration of access control policies. The authority resolution mechanism is also unique to Delegent, and offers an interesting potential for more flexibility.

A disadvantage is that Delegent has not been used in any real world application so far. The forms of constrains in the access control policy that can be expressed are currently limited to group membership and time intervals. (A more advanced constraint model is in the works and is planned for 2004.)

8.13.4 Application to TrustCoM

One of TrustCoMs objectives is to provide a methodology for secure management of resources in virtual organisations. As a virtual organisation consists of a number of independent organisations controlling access to their own resources, the management of resources has to be fully decentralised. Delegent supports decentralised management of access permissions where there are several independent administrative domains as this is the case for virtual organisations.

8.14 Shibboleth

Shibboleth is a project run by the Internet2 consortium in the USA (it comprises academic partners and IBM). Shibboleth defines a protocol for providing users with access to remote resources via authentication at their home site and authorisation via a set of user attributes provided by the home site. Shibboleth therefore provides a Single Sign On solution to accessing web services.

Shibboleth access takes place in two stages

- Obtaining a handle for an authenticated user
- Using the handle to get a set of attributes for the user

A user makes a request to access a remote web site. The user can be stationed at his home site, or anywhere else on the Internet. The web site knows nothing about the user, so needs to find out attributes of the user in order to grant him/her access. The Shibboleth Indexical Reference Establisher (SHIRE) is the service that will try to get a handle on the user. It does this by sending the user back to his home site via a Where Are You From (WAYF) service. The SHIRE uses the Http Redirect reply to re-direct the user to its Where Are You From

service. This prompts the user to choose his home site from a picking list. The WAYF knows the name and location of the Handle Service for each origin site that is participating in Shibboleth. The user picks his home site, and then the User is re-directed to the Handle Server at his home site by the WAYF service.

The Handle Service (HS) is responsible for making sure the user is authenticated locally at the home site, and for creating a handle that can be used to retrieve attributes about the user. The Handle Service prompts the user to login and provide his authentication tokens. The home site can use whatever type of authentication it likes e.g. username/password, Kerberos, digital signatures etc. This is of no concern to the remote Web Service. Once the user has authenticated him/her self, the HS, in cooperation with the local Attribute server, produces a local handle for the user. The content of the handle is left entirely up to the home site. A local handle ensures that the user's name remains private to the local site, and the Web Service will never know the identity of the user that is accessing it. Thus Shibboleth automatically provides Privacy Protection of user identities. Note however, that if a site wishes to make the user's identity known, then the handle can contain information about the user's identity.

The HS passes the handle (in the form of a SAML Authentication statement) along with additional info (called a "handle package") back to the user's browser inside an HTML form that POSTS the data back to the destination SHIRE. This information includes the location of the Attribute Authority server at which the handle will be usable. This message is digitally signed by the HS to prove its authenticity. The SHIRE must check the signature and the message contents to ensure its validity.

The SHIRE passes the handle, AA contact info, and the origin site name to the SHAR (Shibboleth Attribute Requestor). The SHAR sends an Attribute Query Message to the Attribute Authority (AA) server at the user's home site. This is a standard SAML Attribute request message. This request ideally needs to be protected and mutually authenticated and SSL with client side authentication will satisfy this.

The AA server returns an Attribute Response Message (ARM) to the SHAR. (This is a SAML Attribute Statement). Once the ARM is received and validated, the embedded attributes are passed to the web service by the SHAR. The Web service can now grant or deny access to the user based on these attributes. The web service could make use of software such as PERMIS to determine which privileges to grant to the user, based on his attributes (and in fact there is a current project at the University of Salford to seamlessly integrate Shibboleth and PERMIS).

8.14.1 Application to TrustCoM

Shibboleth provides a framework that will allow the members of a virtual organisation to authenticate once to the VO (via their home site), and then to move between the different computer systems and services of the VO, that may be physically located at different members sites, without needing to authenticate again. In addition, Shibboleth allows a user's attributes to be retrieved from the home site, and for these to be used in authorisation decisions at remote sites. Such a SSO will be of great value to TrustCoM, in that it will provide a standard way for user authentication and pre-authorisation throughout all the services of a VO.

Of the two, Shibboleth or Liberty, the author recommends Shibboleth for the following reasons:

i) Each VO member will have a home site, so it makes sense to authenticate via the home organisation rather than via a third party identity provider

ii) Shibboleth software is OpenSource, is based on SAML v.1.0, and is in wide use today in the academic community. There is thus a large body of experience in using it.

iii) Liberty is more a framework specification than a prototype system, and as such, no OpenSource software currently exists for Liberty.

Shibboleth and PERMIS will be seamlessly integrated together before the end of 2004 and the software will be part of the US NMI release.

8.15 Liberty Alliance

Liberty Alliance is a Single Sign On scheme, similar to Shibboleth. But whereas, with Shibboleth, a user always authenticates via his home institution, in Liberty, a user authenticates via an identity provider, which may be any Internet service provider. As in Shibboleth, the actual method of authentication is not specified, and identity providers may use whatever authentication method they deem to be appropriate e.g username/password, X.509 public key certificates, Kerberos, one-time password schemes etc.

Liberty Alliance is based on circles of trust. A Liberty Alliance circle of trust comprises a number of service providers and an identity provider. Service providers are organizations offering Web-based services to users. This includes most organizations on the Web today: Internet portals, retailers, transportation providers, financial institutions, entertainment companies, not-for-profit organizations, governmental agencies, etc. Identity providers on the other hand are service providers offering business incentives so that other service providers will affiliate with them. Establishing such relationships between service providers creates a circle of trust.

Identity federation is based upon linking users' service provider and identity provider accounts. This account linkage, or *identity federatio*n, underlies the other Liberty services. Single sign-on enables users to sign on once with a member of a federated group of identity and service providers (i.e. with a member of a circle of trust) and subsequently use the various Websites in the group without signing on again.

Each user is typically known by a different login identity at each service provider site (e.g. A/c 12345 at BankX and Policy 2456 at Insurance Co Y). Identity federation causes these identities to be linked together, but the user still continues to use each site-specific login identity at each site. No site knows the login identity used by the user at any other site i.e. login identities are not exchanged. Rather the login identities are referenced by Liberty user handles. A Liberty handle is an identifier known by both sites and unique within the circle of trust. A Liberty handle is created by performing a hash of the user's login identity and other information known only to the provider. Because the handle is at least 128 bits, it is virtually guaranteed to be unique. Note that each handle is only known by the two sites that are federated together, and a service provider will use different handles for the same user with different service providers. The user is in charge of federating sites together, and new handles are created each time, so that multiple sites cannot exchange information about the same user by using one handle.

Liberty works via HTTP Redirects or POST messages sent from a service provider to the user, redirecting their browser to the identity provider, whereupon authentication takes place. The redirection specifies that an SSL connection (https) must be set up between the user's browser and the identity provider's web server, so that the user's credentials are not visible on the Internet. The redirection contains a digitally signed SAML Authentication Request message. The identity provider will prompt the user for their credentials across the SSL link before returning the SAML Authentication Response to the service provider. The SAML Authentication Response is then sent back to the service provider via a HTTP redirect or form-POST message body.

There are currently discussions taking place between the Liberty Alliance consortium and the Internet2 consortium to see if Shibboleth and Liberty can be merged together so that the same SAML messages are used by both. In this way, the home site in Shibboleth would become an identity provider in Liberty.

8.15.1 Application to TrustCoM

Liberty Alliance provides a framework that will allow the members of a virtual organisation to authenticate once to the VO, and then to move between the different computer systems and services of the VO, that may be physically located at different members sites, without needing to log on again. Such a SSO will be of great value to TrustCoM. Of the two, Shibboleth or Liberty, the author recommends Shibboleth (see section 8.14.1 for reasons).

8.16 Web Services Security and Policy

As concluded in Chapter 6 Enabling Technologies , Web Services provide a standardized framework for interoperable, secure, reliable and transacted messaging, and constitute the ideal underlying service-oriented messaging framework for collaboration within dynamic VOs across enterprises. This section covers in detail the Web Services Security (in the remainder of this section referred to as WSS) specifications, following the architecture and roadmap proposed by IBM and Microsoft in April 2002[333].

The WSS specifications provide a comprehensive and composable security framework for SOAP-based web services, supporting and integrating various security models, mechanisms, and technologies in a way that enables a variety of systems to securely interoperate in a platform- and language-neutral manner. As indicated in the figure below, the WSS roadmap consists of different, flexible and extensible specifications, each addressing specific parts of the security framework, including WS-Security, WS-SecureConversation, WS-Trust, WS-SecurityPolicy, WS-Federation, WS-Authorization, and WS-Privacy.

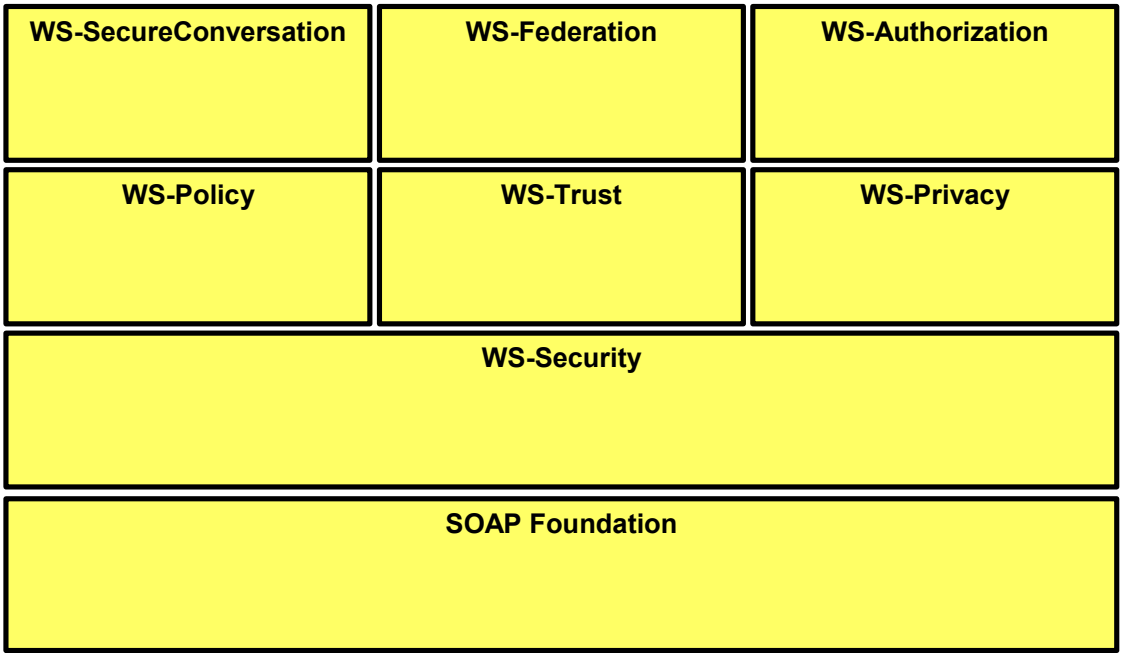


Figure 80 Web Services Security Architecture and Roadmap

8.16.1 Web Services Security and Policy in summary

The architecture essentially supports the secure exchange of SOAP messages as follows.

[333] IBM and Microsoft. Security in a Web Services World: A Proposed Architecture and Roadmap. April 2002. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnwssecur/html/securitywhitepaper.asp.

WS-Security specifies how to apply XML Signature and XML Encryption within a SOAP message in order to provide single-message authentication (origin authentication and integrity) and single-message confidentiality. It also specifies how to attach, or refer to, the associated security tokens in a SOAP message.

WS-SecureConversation specifies how to establish, and how to reference to, a security context, and supports a secure conversation with multiple messages.

WS-Trust specifies how to request, issue, validate, and exchange security tokens. Security tokens are cryptographically protected claims (e.g., identity or authorization assertion) and/or cryptographic keys. Security tokens may be forwardable, delegatable, or proxiable. Security tokens are issued by a Security Token Service (STS). Security token requests may be secured using WS-Security (when exchanging one security token for another), or are secured with an explicit challenge/response or other negotiation protocol (when the requestor does not have a WS security token yet). The WSS framework can in fact support any type of security token. However, security tokens that are to be used across different domains should be interoperable. The following tokens are currently standardized and explicitly supported: username/password combinations and X.509 certificates; binary security tokens, such as Kerberos tickets; and XML security tokens, such as SAML assertions and XrML tokens.

WS-Policy provides a framework that allows Web Services to describe and communicate (publish) their policies to Web Service requestors. WS-SecurityPolicy specifies the security policy assertions that can be used in this framework. The security policy assertions state requirements on the kind of security tokens used, whether or not a message has to be signed or encrypted, etc.

WS-Federation describes how to manage and broker trust relationships in a heterogeneous federated environment including support for federated identities, attributes, and pseudonyms. A federation consists of multiple Web Services domains, each with their own STS, and with their own security policy. WS-Federation specifies scenarios using WS-Trust for example to allow requesters from the one domain to obtain security tokens in the other domain and subsequently to get access to the services in the other domain. Additionally, mechanisms are defined for single sign-in and sign-out, sharing of attributes based on authorization and privacy policies, and integrated processing of pseudonyms (aliases used at different sites/federations).

WS-Authorization and WS-Privacy are to be defined. WS-Privacy will describe a model for how Web services and requesters state privacy preferences and organizational privacy practice statements, while WS-Authorization will describe how to manage authorization data and authorization policies.

8.16.2 Web Services Security and Policy in detail

8.16.2.1 OASIS WSS SOAP Message Security ("WS-Security")

WSS SOAP Message Security[334] describes enhancements to SOAP messaging to provide message integrity and confidentiality. The specified mechanisms can be used to accommodate a wide variety of security models and encryption technologies. The specification also provides a general-purpose mechanism for associating security tokens with message content. No specific type of security token is required; the specification is designed to be extensible (i.e., support multiple security token formats). For example, a client might provide one format for proof of identity and provide another format for proof that they have a particular business certification. Additionally, the specification describes how to encode binary security tokens, a framework for XML-based tokens (including support for SAML and XrML tokens), and how to include opaque encrypted keys. It also includes

[334] OASIS. Web Services Security: SOAP Message Security 1.0. March 2004. http://www.oasis-open.org/committees/download.php/5531/oasis-200401-wss-soap-message-security-1.0.pdf.

extensibility mechanisms that can be used to further describe the characteristics of the tokens that are included with a message.

A complete example of WS-Security secured SOAP message is given below:

```
<?xml version="1.0" encoding="utf-8"?>
<S11:Envelope xmlns:S11=".." xmlns:wsse=".." xmlns:wsu=".." xmlns:xenc=".."
xmlns:ds="..">
   <S11:Header>
      <wsse:Security>
         <wsu:Timestamp wsu:Id="T0">
            <wsu:Created>2001-09-13T08:42:00Z</wsu:Created>
         </wsu:Timestamp>
         <wsse:BinarySecurityToken
               ValueType="...#X509v3"
               wsu:Id="X509Token"
               EncodingType="...#Base64Binary">
            MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
         </wsse:BinarySecurityToken>
         <xenc:EncryptedKey>
            <xenc:EncryptionMethod Algorithm=
                     "http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
            <ds:KeyInfo>
               <wsse:KeyIdentifier
                     EncodingType="...#Base64Binary"
                     ValueType="...#X509v3">
                  MIGfMa0GCSq...
               </wsse:KeyIdentifier>
            </ds:KeyInfo>
            <xenc:CipherData>
               <xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...</xenc:CipherValue>
            </xenc:CipherData>
            <xenc:ReferenceList>
               <xenc:DataReference URI="#enc1"/>
            </xenc:ReferenceList>
         </xenc:EncryptedKey>
         <ds:Signature>
            <ds:SignedInfo>
               <ds:CanonicalizationMethod
                     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
               <ds:SignatureMethod
                     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <ds:Reference URI="#T0">
                  <ds:Transforms>
                     <ds:TransformAlgorithm=
                                 "http://www.w3.org/2001/10/xml-exc c14n#"/>
                  </ds:Transforms>
                  <ds:DigestMethodAlgorithm=
                                 "http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
               </ds:Reference>
               <ds:Reference URI="#body">
                  <ds:Transforms>
                     <ds:Transform Algorithm=
                                 "http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </ds:Transforms>
                     <ds:DigestMethodAlgorithm="http://www.w3.org/2000/09/xmldsig#sha
                     1"/>
                  <ds:DigestValue>LyLsF094hPi4wPU...</ds:DigestValue>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>Hp1ZkmFZ/2kQLXDJbchm5gK...</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#X509Token"/>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S11:Header>
   <S11:Body wsu:Id="body">
      <xenc:EncryptedData
            Type="http://www.w3.org/2001/04/xmlenc#Element"
            wsu:Id="enc1">
```

```
         <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
         <xenc:CipherData>
            <xenc:CipherValue>d2FpbmdvbGRfE0lm4byV0...</xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </S11:Body>
</S11:Envelope>
```

8.16.2.2 OASIS WSS UsernameToken Profile

WSS UsernameToken Profile[335] describes how to use username/password credentials with the Web Services Security specification, and particularly how to add them as XML security token. The following example illustrates using a digest of the password along with a nonce and a creation timestamp:

```
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu= "...">
   <S11:Header>
      <wsse:Security>
         <wsse:UsernameToken>
            <wsse:Username>NNK</wsse:Username>
            <wsse:Password Type="...#PasswordDigest">
               weYI3nXd8LjMNVksCKFV8t3rgHh3Rw==
            </wsse:Password>
            <wsse:Nonce>WScqanjCEAC4mQoBE07sAQ==</wsse:Nonce>
            <wsu:Created>2003-07-16T01:24:32Z</wsu:Created>
         </wsse:UsernameToken>
      </wsse:Security>
   </S11:Header>
</S11:Envelope>
```

8.16.2.3 OASIS WSS X.509 Certificate Token Profile

WSS X.509 Certificate Token Profile[336] describes how to use X.509 certificates with the Web Services Security specification. We refer to the example of the secured SOAP message, given above.

8.16.2.4 Web Services Security Kerberos Binding

WS Security Kerberos Binding[337] describes how to use the Web Services Security specification with Kerberos.

8.16.2.5 WS-Trust

WS-Trust[338] defines extensions that build on WS-Security to request and issue security tokens and to manage trust relationships. More specifically, WS-Trust defines a request/response protocol for security token issuance, validation, and exchange. A requester sends a <RequestSecurityToken> with a request to issue a new security token (wsse:ReqIssue), validate a given security token (wsse:ReqValidate), or exchange a given security token for a new one (wsse:ReqExchange). In case of issuance or exchange, the <RequestSecurityTokenResponse> message will subsequently contain a

---

[335] OASIS (IBM, Microsoft, Sun, VeriSign). Web Services Security: UsernameToken Profile 1.0. March 2004. http://www.oasis-open.org/committees/download.php/5532/oasis-200401-wss-username-token-profile-1.0.pdf.

[336] OASIS (IBM, Microsoft, Sun, VeriSign). Web Services Security: X.509 Certificate Token Profile 1.0. March 2004. http://www.oasis-open.org/committees/download.php/5533/oasis-200401-wss-x509-token-profile-1.0.pdf.

[337] IBM, Microsoft. Web Services Security Kerberos Binding. December 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-security-kerberos.asp.

[338] IBM, Microsoft, RSA, VeriSign. Web Services Trust Language (WS-Trust). Version 1.0. December 18, 2002. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-trust.asp.

<RequestedSecurityToken> (e.g., public key certificate) and/or a <RequestedProofToken> (e.g., corresponding private key).

WS-Trust allows to formulate certain security token requirements, and provides a number of extensions:

- Scope requirements: <wsp:AppliesTo>, <Claims>
- Key and encryption requirements
- Delegation, forwarding, and proxy requirements: <OnBehalfOf>, <DelegateTo>, <Forwardable/>, <Delegatable/>, <Proxiable/> (e.g., used for Kerberos)
- Lifetime and renewal requirements
- Policies references: <wsp:Policy>, <wsp:PolicyReference>
- Challenge/response: <SignChallenge> and <SignChallengeResponse>,
- Other negotiation protocols to obtain security token <BinaryNegotiation>

8.16.2.6 WS-Policy

WS-Policy[339] provides a general-purpose model and corresponding syntax to describe and communicate the policies of a Web Service. WS-Policy defines a base set of constructs that can be used and extended by other Web Services specifications to describe a broad range of service requirements, preferences, and capabilities. WS-Policy by itself does not provide a negotiation solution for Web services.

8.16.2.7 WS-PolicyAssertions

WS-PolicyAssertions[340] specifies a set of common message policy assertions that can be specified within a policy.

8.16.2.8 WS-PolicyAttachment

WS-PolicyAttachment[341] specifies three specific attachment mechanisms for using policy expressions with existing XML Web service technologies. Specifically, it defines how to associate policy expressions with WSDL type definitions and UDDI entities. It also defines how to associate implementation-specific policy with all or part of a WSDL portType when exposed from a specific implementation.

8.16.2.9 WS-SecurityPolicy

WS-SecurityPolicy[342] is an addendum to WS-Security and indicates the policy assertions for WS-Policy, which apply to WS-Security. WS-SecurityPolicy currently explicitly supports the following security policy assertions:

- What kind of <SecurityToken> is required, from whom, and which claims it should provide; supported tokens are: wsse:X509v3, wsse:Kerberosv5TGT, wsse:Kerberosv5ST, wsse:UsernameToken, wsse:SAMLAssertion, wsse:XrMLLicense

---

[339] IBM, Microsoft, BEA, SAP. Web Services Policy Framework (WS-Policy). Version 1.1. 28 May 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-policy.asp.

[340] IBM, Microsoft, BEA, SAP. Web Services Policy Assertions Language (WS-PolicyAssertions). Version 1.1. 28 May 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-policyassertions.asp.

[341] IBM, Microsoft, BEA, SAP. Web Services Policy Attachment (WS-PolicyAttachment). Version 1.1. 28 May 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-policyattachment.asp.

[342] IBM, Microsoft, RSA, VeriSign. Web Services Security Policy Language (WS-SecurityPolicy). Version 1.0. December 18, 2002. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-securitypolicy.asp.

- <Integrity> protection: which message parts and how
- <Confidentiality> protection: which message parts and how
- Required <Visibility> of message parts for intermediaries
- Specific behaviours of <SecurityHeader>
- Maximum allowed <MessageAge>

8.16.2.10 WS-SecureConversation

WS-SecureConversation[343] defines extensions that build on WS-Security to provide secure communication. Specifically, it defines mechanisms for establishing and sharing security contexts, and deriving session keys from security contexts.

A <wsse:SecurityContextToken> is created by a security token service, by one of communicating parties and propagated with a message, or through negotiation. A <wsse:SecurityContextToken> can contain the following elements: <wsu:Identifier>, <wsu:Created>, <wsu:Expires>, <wsse:Keys>, <xenc:EncryptedKey>, and <wsse:SecurityTokenReference>.

A security context token contains a shared secret. It is recommended that derived keys are used for signing and encrypting messages associated only with the security context. Different key derivations may be defined for different purposes, and for refreshing keys. The <DerivedKeyToken> token is used as a mechanism for indicating which derivation is being used within a given message.

8.16.2.11 WS-Federation

WS-Federation[344] defines mechanisms that are used to enable identity, account, attribute, authentication, and authorization federation across different trust realms.

The WS-Federation specification particularly describes models and a framework for federation of identity, attribute, authentication, and authorization information. The key driving requirements are to enable appropriate sharing of identity, authentication, and authorization data, brokering of trust and security token exchange, not requiring local identities at target services, and optionally hiding of identity information and other attributes. The models build upon the foundations specified in WS-Security, WS-Policy, and WS-Trust. WS-Trust is extended to allow attributes and pseudonyms to be integrated into the token issuance mechanism to provide federated identity mapping mechanisms.

WS-Federation provides federation of meta-data. Web services may want to indicate where requestors can obtain security tokens in order to satisfy the services claims requirements. The mechanisms defined in WS-PolicyAssertions are therefore extended with a <wsse:RelatedService> assertion, allowing a trust domain to indicate where to find its identity provider (wsse:ServiceIP), its security token service (wsse:ServiceSTS), its attribute service (wsse:ServiceAS), and its pseudonym service (wsse:ServicePS).

WS-Federation provides a mechanism for cleaning up any cached state and security tokens that may exist within the federation. A requester may send a <wsse:SignOut> message to its security token service. This message is then federated/forwarded to security token services in other domains where the requester is “logged in”. The precise implication of a sign-out on currently active transactions is undefined and is resource-specific.

WS-Federation foresees an Attribute Service, with the possibility to expose attribute stores as UDDI endpoints.

---

[343] IBM, Microsoft, RSA, VeriSign. Web Services Secure Conversation Language (WS-SecureConversation). Version 1.0. December 18, 2002. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-secureconversation.asp.

[344] IBM, Microsoft, RSA, VeriSign. Web Services Federation Language (WS-Federation). Version 1.0. July 8 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/ws-federation.asp.

WS-Federation specifies a Pseudonym Service through which parties (via a request/response protocol) can get, set, or delete pseudonyms. This is essentially a way a mapping an identity in one domain to another identity in another domain.

8.16.2.12 WS-Federation Active Requestor Profile

WS-Federation Active Requestor Profile[345] defines how the cross trust realm identity, authentication and authorization federation mechanisms defined in WS-Federation are used by active requestors such as SOAP-enabled applications.

The Active Requestor Profile is the WS-Federation instantiation, which is relevant for TrustCoM.

8.16.2.13 WS-Federation Passive Requestor Profile

WS-Federation Passive Requestor Profile[346] describes how the cross trust realm identity, authentication and authorization federation mechanisms defined in WS-Federation can be utilized used by passive requestors such as Web browsers to provide Identity Services. Passive requesters of this profile are limited to the HTTP protocol.

8.16.2.14 WS-Authorization

WS-Authorization will describe how to manage authorization data and authorization policies. No specification has been published yet.

8.16.2.15 WS-Privacy

WS-Privacy will describe a model for how Web services and requesters state privacy preferences and organizational privacy practice statements. No specification has been published yet.

8.16.3 Application to TrustCoM

An initial investigation[347] shows to what extent the WSS specifications can address some of the trust and security issues in VOs, how the specifications can support trust and security in dynamic VOs as a flexible and extensible underlying technology, and how they can complement specific tools and technologies for trust, security and contract management.

8.16.3.1 Trust establishment (VO Formation)

How trust is established or determined, is out of the scope of the current WSS specifications. Federations across different organisations are setup assuming mutual trust. From a WSS technical point of view, organisations trust that their respective Security Token Services do the correct mapping between an identity, a role, or an access right, and the associated public key. These statements need to be securely communicated, so the organisations need to establish trust in the cryptographic root keys with which the authenticity of the tokens and assertions can be verified. Trust management and secure exchange of root keys happens entirely out-of-band.

---

[345] IBM, Microsoft, RSA, VeriSign. WS-Federation: Active Requestor Profile. Version 1.0. July 8, 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/grfWS-FederationActiveRequestorProfile.asp.

[346] IBM, Microsoft, RSA, VeriSign. WS-Federation: Passive Requestor Profile. Version 1.0. July 8, 2003. http://msdn.microsoft.com/webservices/?pull=/library/en-us/dnglobspec/html/passive-client-profile.asp.

[347] J. Claessens and C. Geuer-Pollmann. Investigating the Web Services Security Specifications as an Enabling and Extensible Framework for Trustworthy Business Processing in Dynamic Virtual Organisations. Submitted to eChallenges 2004.

Dynamic VOs would benefit from trust establishment through mediation with trusted services. These trusted services act as roots of trust, reputation services, or recommendation services. Trust and reputation statements could also be communicated in between peers directly. The WSS specifications can complement trust management frameworks (e.g., SULTAN see section 7.8) in terms of the secure communication of trust statements, and the support of specific trust-related security tokens. Particularly, trust and reputation statements and recommendations could be published in UDDI; WS-Trust can be leveraged to request, issue, and validate trust and reputation statements/assertions and recommendations; and trust/reputation can also be seen as a WS-Federation attribute service. Trust in the cryptographic key themselves, with which trust statements can be verified, can be seen as a recommendation service, similar as in the "web of trust".

Similarly, the WSS specifications can support further maintenance of trust knowledge (VO Dissolution), whereby organisations can securely push trust knowledge, and provide recommendations.

8.16.3.2 Security policy consolidation based on a contract (VO Formation)

The WSS specifications allow to describe and to publish the security policy associated with a web service. The security policy can be retrieved in various ways. The formation and negotiation of the security policy is however out of the scope of the WSS specifications. Without automated support for security policy consolidation, WSS policies will be fairly static, and will not easily allow for very dynamic VOs.

The WSS specifications should be complemented with a contract negotiation framework. The contract defines the rights and obligations of each organisation in the dynamic VO, related to the overall business process. The WSS policies should then be derived from the negotiated contracts. Possibly, a higher-level policy language, in between WS-Policy and the contract could be introduced (e.g., Ponder, see section 8.3). WS-(Security) Policy may need to be extended to be able to capture all high-level dynamic VO policy semantics into low-level WSS terms.

When forming the VO, the contract should be negotiated. The security policies and trust knowledge of the participating organisations give input to this process. The WSS specifications can support secure contract negotiation and secure communication of the policy and trust inputs. A negotiated contract can be stored in UDDI.

Trust and contract negotiation may require interactive and selective disclosure of policies and credentials, such as offered by Trust-X (see section 7.6). This is not supported within the WSS specifications. As WSS policy is derived from trust and a contract, direct negotiation of web service policy and security token usage – requiring an interactive protocol between requester and web service – does not seem to be needed. Negotiation may be an interesting feature though when requesting security tokens (e.g., trust recommendations) through WS-Trust.

8.16.3.3 Security policy deployment and enforcement (VO Operation)

The WSS specs only deal with security on the level of web services, and for example assume that SOAP nodes are accessible on the network layer; this assumption cannot be made in a dynamic context. Security policy enforcement must be done in a consistent way across different layers (business process, web service, and network). This requires the WSS specifications to be complemented with a higher-level policy (see above) and authorisation framework (e.g., PERMIS, see section 8.12), which drives policy and authorisation at different, levels, including WSS. At the WSS level, this can be realized if the authorization framework foresees the necessary WS-Trust interface.

In a typical WSS scenario, security policy enforcement is based on looking at a single web service request only. Security policy enforcement should however also take into account the business process and the context related to that specific web service request. In a dynamic VO, the context is typically determined by the contract associated with the VO and the business process.

Federations of organisations do not need a single PKI anymore, but traditionally still have a limited number of central security token services, across which trust is brokered. Dynamic VOs require further security decentralization (cfr. GRASP (see section 8.18.5) and Hybrid PKI (see section 8.11) and explicit authorisation delegation (e.g., Delegent, see section 8.13). Dynamic VOs also need to be able to cope with peer-to-peer and "off-line" (i.e., no central security token service available) scenarios. The WSS specifications in principle support any type of (XML) security tokens. Aspects such as delegation can be properly addressed by adding explicit support for the appropriate, delegatable security tokens. Peer-to-peer and off-line scenarios in WSS means that end nodes should be able to play the role of security token service.

8.16.3.4 Security policy adaptation (VO Evolution)

During the lifetime of the VO, security policies may need to be adapted (e.g., to reflect leaving, joining, or re-assessing trust of members). The WSS specifications do not include a standardized way of pushing policy updates (new policies can be published, but these need to be polled by the requesters). As dynamic policy updates should reflect contract amendments, WSS policy updates should be pushed from the contract management, rather than directly across the web services. Interaction should happen at the trust and contract management.

Trust management frameworks deal with quantified trust. This allows to make more fine-granular statements (as opposed to boolean) about trust in partners, and re-assess these during the lifetime of the VO. An interesting question is then how this influences security policy adaptation? Quantified trust has definitely an impact during contract negotiation, and consequently indirectly an impact on the VO security policies (e.g., more restrictive policy with more extensive auditing is applied with partners who are less trusted). A good approach seems to be that in the scope of a contract (which can be updated), the security policy is fixed and partners are "fully trusted" (under the rights and obligations specified). Trust re-assessments lead to contract changes, which subsequently are then pushed down into updated WSS policies.

It is not clear whether it is a good idea to allow trust to be a direct parameter in the WSS policy itself. Autonomic security decisions should for example not become unpredictable. Direct trust-based authorization would require trust assertions to be attached to web service requests, and be validated by the service. WS-Policy would need to be extended with rules that can interpret and make access control decisions based on the trust claims in the tokens. This is less straightforward to realize, and may be better left to a higher level policy language.

8.17 WS-Aba: Web Service Attribute Based Access Control

WS-ABA is fine-grained negotiation-based access control model for Web Services. The goal of the model is to express, validate and enforce access control policies without assuming pre-established trust in the users of Web services, while at the same time being in line with recent developments on identity management.

8.17.1 Description

WS_ABA is an access control model for Web services characterized by capabilities for negotiating service parameters. It is intended to be used within SOAP standard.

The model allows express, validating and enforcing access control policies without assuming pre-established trust in the users of Web services.

Access conditions are expressed in terms of identity attributes of the requesters. Moreover, in order to support a fine-tuning of access control, access conditions also take into account the parameters characterizing web services.

Once trusted, users can change dynamically their access requests in order to obtain authorizations. Other key features of WS-ABA are the efficient support of digital signatures and a mechanism supporting composed delegation. The model, the underlying architecture and the implementation are under development.

8.17.2 Advantages and Disadvantages

The model, the underlying architecture and the implementation are under development.

8.17.3 Application to TrustCoM

The model allows express, validating and enforcing access control policies without assuming pre-established trust in the users of Web services.

8.18 Grid Security Frameworks

8.18.1 Akenti

In this subsection we present an overview of Akenti based on [348] and [349].

The Akenti [349] authorization system aims to meet these two needs: to use a virtual organization-wide user identity (e.g. an X.509 public key certificate); and to facilitate setting access policy by multiple independent stakeholders remote from the actual resource gateway. The goal of the Akenti project is to provide a practical, easy to use, authorization service that meets the needs of collaboratories and computational Grids.

Akenti is an authorization service (PDP) that uses authenticated X.509 certificates to establish identity and distributed digitally signed authorization policy certificates to make access decisions about distributed resources. It supports authorization decisions based on policy that it gathers from many sites. It returns authorization decisions as a signed capability certificate, which can be used directly by a PEP to grant access or could be used by the subject of the certificate as a rights-granting authorization assertion. It supports Globus proxy identity certificates, and could be extended to handle restricted delegation credentials. We have implemented an Apache Web server module which allows the same authorization policy to be used to control access to Web accessed resources as well as resources accessed by other remote methods.

Akenti assumes that X.509 certificates[350] and the SSL/TLS[351] connection protocols have been used to securely authenticate a user that is requesting access to a resource. It represents the authorization policy for a resource as a set of (possibly) distributed certificates digitally signed by unrelated stakeholders from different domains. These policy certificates are independently created by authorized stakeholders. When an authorization decision needs to be made, the Akenti policy engine gathers up all the relevant certificates

---

[348] M. THOMPSON, W. JOHNSTON, S. MUDUMBAI, G. HOO, K. JACKSON, A. ESSIARI 1999 Certificate-based Access Control for Widely Distributed Resources, *Proceedings of the Eighth Usenix Security Symposium,* Aug. 1999

[349] Thompson M., Essiari A., Mudumbai S: Certificate-based Authorization Policy in a PKI Environment. ACM Transactions on Information and System Security (TISSEC), Volume 6, Issue 4 (November 2003) pp: 566-588, LBNL-49512; http://www-itg.lbl.gov/security/Akenti/Papers/ACMTISSEC.pdf

[350] R. HOUSLEY, W. POLK, W. FORD, D. SOLO 2001 Internet X.509 Public Key Infrastructure Certificate and CRL Profile *draft-ietf-pkix-new-part1-12.txt,* http://www.ietf.org/internet-drafts/draft-ietf-pkix-new-part1-12.txt

[351] T. DIERKS, C. ALLEN 1999 The TLS Protocol, Version 1 *IETF RFC 224*6; http://www.ietf.org/rfc/rfc2246.txt

for the user and the resource, verifies them, and determines the users rights with respect to the resource.

8.18.1.1 Authorization model

The Akenti model consists of *resources* that are being accessed via a *resource gateway* (the Policy Enforcement Point - *PE*P) by *user*s. These users connect to the resource gateway using the SSL handshake protocol to present authenticated X.509 certificates. The *stakeholders* for the resources express *access constraints* on the resources as a set of *signed certificate*s, a few of which are self-signed and must be stored on a known secure host (probably the resource gateway machine), but most of which can be stored remotely. These certificates express the attributes a user must have in order to get specific rights to a resource, who is trusted to create use-condition statements and who can attest to a user's attributes. At the time of the resource access, the resource gatekeeper (PEP) asks a trusted Akenti server (the Policy Decision Point - *PD*P), what access the user has to the resource. The Akenti server finds all the relevant certificates, verifies that each one is signed by an acceptable issuer, evaluates them, and returns the allowed access. See Figure 81 [349].

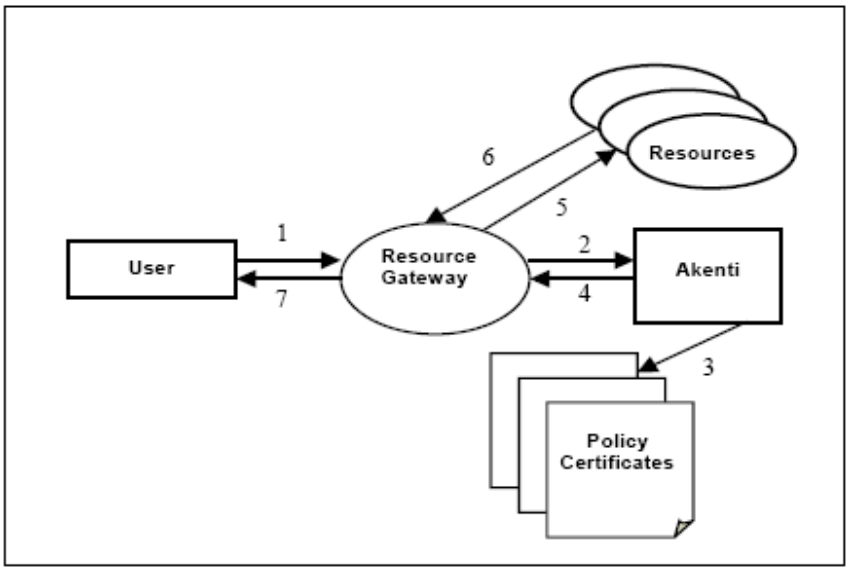


Figure 81 Akenti Authorisation Model[349]

Akenti system mostly concentrates on the pull (vs. push) model in order to allow applications to use Akenti authorization over standard TLS connections that transport and verify X.509 certificates. It has been also experimented with a push model where Akenti is contacted by the user and returns a signed capability certificate containing an authorization assertion consisting of a subject's Distinguished Name (DN), public key, the CA that signed for this name, the name of the resource and the subject's rights. If this is presented by the user to a resource gatekeeper, along with an authenticated identity certificate, the gatekeeper need only verify Akenti's signature of the certificate, and verify that the subject named in the capability is the same as that in the identity certificate. These capability certificates are short-lived in order to avoid the problems of revocation.

8.18.1.2 Akenti policy language

Akenti policy is expressed in XML and stored in three types of signed certificates: *policy certificates, use-condition certificates* and *attribute certificate*s. Policy certificates specify the sources of authority for the resource. Use-condition certificates contain the constraints that

control access to a resource. Attribute certificates assign attributes to users that are needed to satisfy the use constraints.

Policy certificates are self-signed, co-located with the resources to which they apply and contain only minimal information since they are centrally located and may be administratively difficult to update. They are used to bootstrap and to provide closure for the trust chain by specifying the sources of authority for a resource. The sources of authority are the CAs who may sign X.509 certificates for all the principals involved in an authorization decision and the stakeholders who may issue use-condition certificates for the resource. Whenever a certificate is used, the Akenti policy engine will check that it has been signed by an acceptable issuer, and that the signature verifies. The CAs are represented by their X.509 certificates which provide a trusted copy of their public keys and information about where they publish certificates and certificate revocation lists. The stakeholders are represented by their Distinguished Names (DN) and the DN of the CA that issued a certificate for that name and a list of places, specified by URLs, where the stakeholders put the use-condition certificates that they issue. A policy certificate may optionally contain a list of URLs in which to search for attribute certificates.

Resources controlled by Akenti authorization may be grouped into a *resource realm.* A resource realm can be organized as a flat structure of resources such as instruments or compute platforms, or a hierarchical structure such as a file system or set of Web documents. Each resource realm has at least one policy certificate which must be stored in a known and secure place. Normally it is on the same machine that controls access to the resource, but it could also be on the platform where the Akenti server is running, if they are different. In the case of hierarchical resources, there must be at least one policy certificate at the top of the tree (referred to as the root policy). In addition, there may be a policy certificate at any level where there are new stakeholders, or restrictions on the allowed CAs. Levels without their own policy certificates inherit policy from higher levels.

Each stakeholder group for a resource must create at least one and possibly more use-condition certificates for the resource. A use-condition certificate consists of a constraint, which is a relational expression of the attributes a user must have to get a certain set of rights and a list of the principals who can attest to the required attributes. Components of the X.509 distinguished name can be used as attributes, or attributes can be defined in the context of the resource. For example, *role = researcher* or *group = accountin*g. These attribute requirements can be combined with the Boolean operators *&&* or ||. Negative permissions such a *group != accounting* are not supported because of the difficulty in requiring all the relevant attribute certificates to be found. It is also possible to specify real-time or system attributes such as *time<=5PM && time>=9A*M, or *system_load* < 2. If Akenti is unable to evaluate such system attributes it may return them to the resource gateway for evaluation. An attribute authority (consisting of an issuer and its CA) is specified as the signing authority for each attribute-value pair.

A stakeholder may put use-condition certificates in more than one place for reliability, but each directory must contain the complete set. Since use-conditions restrict access to a resource, it is essential that either all or none of them be found. If no use-conditions are found for a stakeholder group, all access to the resource is denied. This is not the case with attribute certificates since they only serve to increase access. Thus a missing attribute certificate may limit or deny a user's access, but will never allow an access that should be denied.

Attribute certificates contain an attribute-value pair and the principal to whom it applies. They are signed by attribute authorities that have been specified in a use-condition certificate. Attributes can apply to more than one resource, although they are likely to be applicable in only a single resource realm. Akenti attribute certificates are XML documents rather than the proposed IETF ASN.1 attribute certificates. The complete XML schema for Akenti certificates can be found on the Akenti Web site[352].

[352] Akenti Page (publications & source code): http://dsd.lbl.gov/security/Akenti/

A language to describe access policy typically involves making statements about some or all of the following elements: requestor identity, grantor identity, a set of access rights, a set of constraints[353]. The target resource to which the rights apply may be explicit in the policy statement or may be implicit in the context. The identities may be expressed as names or as public keys. The access rights are usually arbitrary strings whose meaning is agreed on between the policy creator and the PEP. Constraints can be expressed as a set of tokens, as a Boolean expression or in a special purpose language. The names of the constraints e.g., group, role, time only need to be self-consistent within the policy, the attribute assertions and the PEP.

Two basic types of authorization policy statements are: authorization assertions, including rights delegation or capabilities, used in push model authorization systems and resource-centric access policy statements used in pull models [354]. The Akenti static certificates contain resource-centric access policy including trust relationships and attribute assertions. Akenti returns its access decision in a dynamic short-lived capability certificate, signed by the Akenti server. This certificate can be used as a delegation of rights proxy, where the bearer is the user who made the authorization request and the grantor of the rights is the Akenti authorization server. XML was used to make certificates more human readable at the expense of compactness.

8.18.1.3 Creating policy

Since policy is contained in signed XML certificates, which are interdependent, a stakeholder needs some tools to assist in the creation of certificates. A stakeholder starts by creating the root policy certificate for the resource realm. The X.509 certificates of all the trusted CAs must be available from a trusted source and are placed in the root policy certificate. This certificate also contains the URLs of the locations where these CAs publish certificates and certificate revocation lists. The first stakeholder must decide if there are other stakeholders for the resource and, if so, include their DNs and CAs in the root policy certificate. In a hierarchical set of resources, only the top-level stakeholders need to be known initially. They in turn, can delegate control to other stakeholders for resources lower in the hierarchy.

Akenti certificates can either be created by a command line tool that signs an XML input certificate, or by a GUI program that steps a stakeholder through a menu of choices for each field in the certificate. The GUI program is supported by a resource definition server running on the resource host, which in turn reads a resource definition file and any existing policy certificates to find stakeholder names, acceptable attributes and actions for a resource realm. The resource definitions file is only used to provide suggestions to the policy creation GUIs. It includes the names of the CAs, and their publishing directories, attribute names and values, the principals that are acceptable for issuing specific attribute values, and a list of actions that are relevant to the resource realm. In summary the two methods of getting started are:

- Create an XML version of a root policy certificate, following one of the templates provided by the Akenti distribution, and use a command line program to sign it with the stakeholder's private key contained in a pkcs12 format file, and store it in the resource tree.
- Create a resource definition file, start the resource definition server, and then use the GUI program to create, sign and store a policy certificate.

---

[353] T. RYUTOV, B.C. NEUMAN 2000, Access Control Framework for Distributed Applications, IETF draft work-in-progress draft-ietf-cat-acc-cntrl-frmw-05.txt, Nov 2000

[354] The paper of Lampson, et al. which describes a formal theory for authentication in distributed systems defines a "speaks for" relationship that describes the delegation of rights [Lampson, et al. 1992]. Neuman's paper on proxy-based authorization [Neuman 1993] gives a clear description of various kinds of proxy certificates: full proxies, restricted proxies, bearer and delegated proxies. The CRISIS security architecture [Belani 1998] implemented these ideas in *transfer certificates* in which one principal can delegate a subset of its rights to another principal. The transfers are expressed as a list of capabilities. There can be a chain of transfer certificates delegating a more limited set of capabilities to additional principals.

The stakeholder must now create at least one use-condition certificate for the resource. Anyone can create a use-condition certificate, but it will only be used during authorization if it is issued and signed by one of the stakeholders currently listed in the resource's policy certificate.

In creating a use-condition certificate the stakeholder will be presented with a menu of possible stakeholders for the resource (of which he must be one), previously defined attribute/value pairs and their allowed attribute authorities and the defined actions for the resource. The stakeholder is also asked about such details as the length of time for which this certificate should be valid; the scope of the use-condition (does it just apply to the one resource or to a hierarchy of resources); and whether it is a critical use-condition (it must be satisfied or the user gets no access to the resource even if she satisfies other use-conditions). The use-condition certificates must be stored in a directory that is specified in the policy certificate. When creating an attribute certificate the stakeholder will be presented with a list of defined attributes and values for the resource realm.

Once a set of policy, use-condition and attribute certificates has been stored, the stakeholder can use a Web-based interface to see what access is allowed to the resource.

8.18.1.4 Checking access

The Akenti authorization service can be called in several ways: It can be invoked as a function call by the PEP and thus run as part of the gatekeeper. It can be contacted as a server through TCP or TLS and it will return a signed capability certificate. If an insecure protocol is used, the gatekeeper must have a copy of the Akenti server's public key and verify the certificate, before it can trust the information. The Akenti server always returns a signed capability certificate that may include both conditional and unconditional rights. Conditional rights are rights that may have some conditions attached that only the PEP can evaluate, such as current machine load, disk availability or the state of some related system variable. An API wrapper will extract the unconditional actions and return them as strings, and will parse and evaluate the runtime conditions calling an evaluator function provided by the PEP.

The Akenti policy engine finds all the use-conditions by searching in the URLs specified in the policy certificates and verifying the issuer and signature on each certificate. If a use-condition certificate cannot be found for each stakeholder group, access to the resource is denied. Attribute certificates are searched by following URLs in either the policy certificates and/or use-conditions. Again, the issuer and signature of each certificate is verified. This signature verification requires that the Akenti policy engine be able to find the X.509 certificates for each issuer. If the CAs who issue certificates publish them in an LDAP server, Akenti will look there. Otherwise, there must be some setup actions taken to put all the expected certificate issuers' X.509 certificates in a file system or at a location specified by a URL. Akenti caches all the certificates that it finds in order to reduce subsequent search time. It also caches the authorization decision as a capability certificate that contains the access rights of a user for a resource, so that subsequent requests for the same resource by the same user require no repeated decisions. The lifetime of the cached certificates is set in the policy certificate for the resource.

Akenti was developed for use in distributed environments that rely on X.509 certificates and TLS to establish authenticated secure connections between the users and the resources. Hence, it was natural to rely on X.509 certificates for identity and to implement a pull authorization model. With the advent of the Web services model for distributed computing environments, the underlying security mechanisms are changing. TLS will be replaced with secure connections being made at the SOAP message level. The suggested protocols for secure connections support a number of different authentication methods. With the communication protocol consisting of XML messages, it is much easier to extend the security protocols to push attribute or authorization information to a PEP. It is anticipated that there will be a standard interface for a Grid authorization service that will standardize the authorization request and response messages. The request message will contain additional assertions which can include roles as well as a generalized authentication token. As Akenti

evolves to fit into the Grid Services environment, it will need to address the issue of multiple authentication tokens and additional attribute assertions. Akenti should be able to conform to new authorization interfaces while keeping intact its fundamental goal of accommodating access policy statements that are independently created by stakeholders from unrelated domains.

The Akenti authorization has been used as part of the Diesel Combustion Collaboratory[355] to control access to Web-based documents and remote execution and is now being integrated with the Globus job manager to control access to legacy applications in the National Fusion Grid[356].

8.18.2 EDG security and VOMS

The European Data Grid (http://www.eu-datagrid.org) and the Data TAG (http://www.datatag.org) projects have been focusing on the development of Data Grid Research Infrastructures for Europe. Both projects have contributed to the development of the EDG Security Infrastructure, which is planned to be further developed in the context of the Research Infrastructures I3 project EGEE. http://public.eu-egee.org/

The EDG Security Infrastructure extends the Globus Toolkit Implementation of the Grid Security Infrastructure (GSI), as depicted in the following figure. In the context of this state-of-the-art review we focus on VOMS, which is its main innovative figure.

[355] C. PANCERELLA, L. RAHN, C. YANG 1999 The Diesel Combustion Collaboratory: Combustion Researchers Collaborating over the Internet*, Proceedings of ACM/IEEE SC99 Conference,* November 13-19, 1999. Portland, Oregon, USA, http://www-collab.ca.sandia.gov/dcc/

[356] K. KEAHEY, T. FREDIAN, Q. PENG, D.P. SCHISSEL, M. THOMPSON, I. FOSTER, M. GREENWALD, D. MCCUNE 2001 Computational Grids in Action: The National Fusion Collaboratory, *Future Generation Computer System, 2001.* http://www.fusiongrid.org

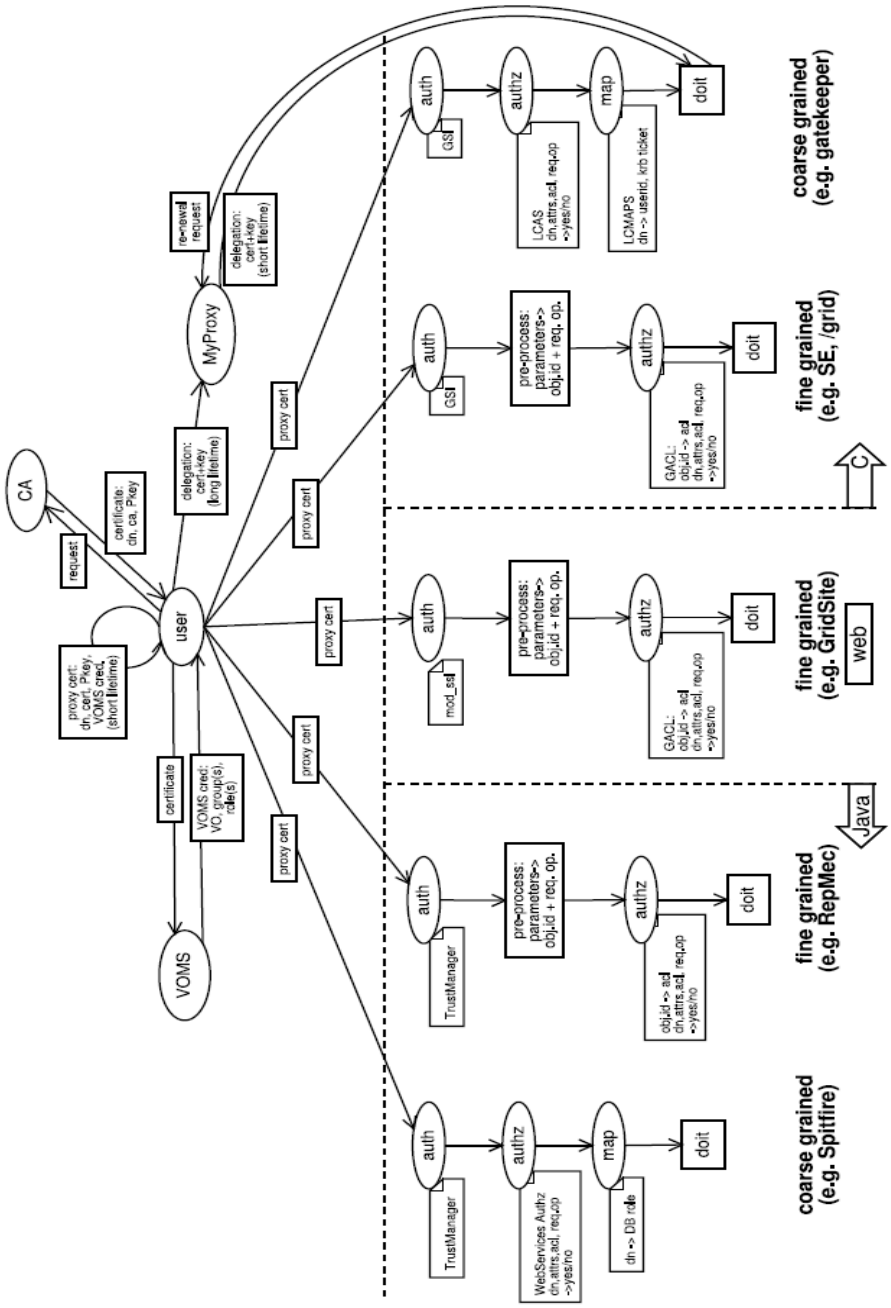


Figure 82 Overview of EDG Security Infrastructure

The VOMS architecture uses the authentication and delegation mechanisms provided by the Globus Toolkit Grid Security Infrastructure (GSI).

Users of a VO are organized in groups, which in general forms a hierarchical structure with the VO itself as the root; the management of a group can be delegated. Excluding the root, each group may have, in general, several ancestors; note that it cannot have cycles in this structure (i.e. a group which is subgroup of itself). Thus one can represent the VO structure with a Direct Acyclic Graph (DAG)

Users are normally contained in subgroups: for a user being member of a particular group G implies that he is also contained in all ancestor groups, even if it not explicitly stated, up to the root (i.e. in all groups contained in the paths from G to the root).

Users are also characterized by roles they can cover in a group or at VO level (but in the VOMS/EDG model the VO is functionally equivalent to a group) and capabilities (properties to be interpreted by the local sites, e.g. ACL's). Roles are inherited by group members from ancestor groups (i.e. if a user as a role in a group and if he is member of one of its subgroups, he covers the same role in the subgroup), while the opposite is not generally true. The same inheritance rule applies for capabilities.

In conclusion, within this model, if a user U is member of the groups **_G_**,. . . ,**G__**, noting with the triplet (**G_**,**R_**,**C_**) the membership, roles and capabilities of U relatively to the group **G_**, the complete authorization information about U is formed from the set (**G_**,**R_**,**C_**),. . . , (**G_**,**R_**,**C_**).

VOMS was developed in order to address shortcomings observed with the security solution adopted at the first testbed implementation of EDG. In particular concerns about flexibility and scalability: No roles, subgroups memberships and any other user peculiarity are supported. Moreover, the use of a RP-based database (i.e. the grid-mapfile), periodically updated, hardly scales in a production environment with a large number of users, each, potentially, with his groups, roles and capabilities, whereas in the test-bed the users situation is almost static, and user policy is very simple.

VOMS attempts to let users present the authorization data as they try to access the local resources (i.e. shifting from pull to push model); it also differentiates from the use of LDAP protocol that was judged not the best choice to sustain the burden of a potentially high number of complex queries.

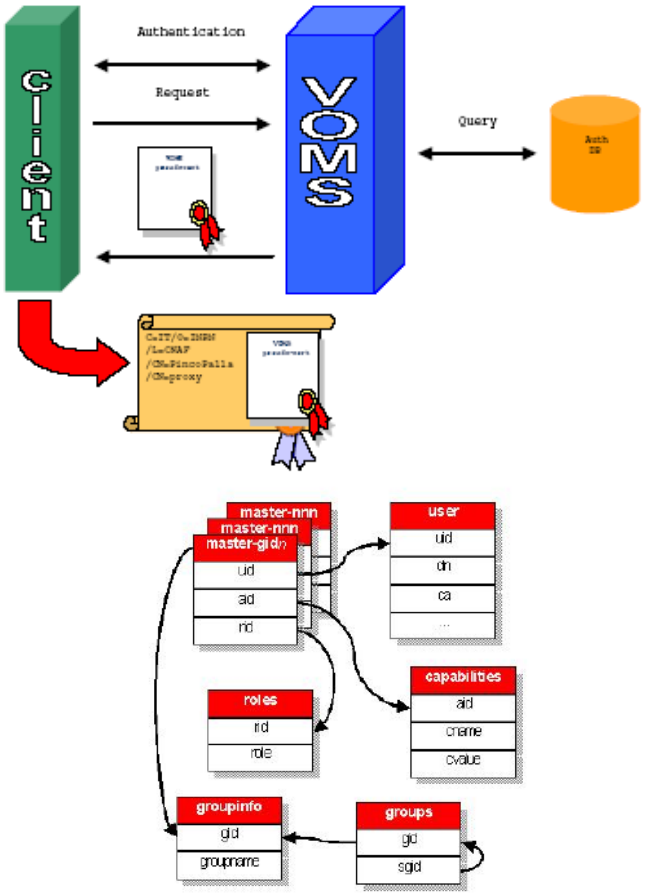


Figure 83 Overview of VOMs architecture and VO structure captured in Attribute Certificates

The VOMS System is composed by the following parts:

- **User Server:** Receives requests from a client and returns information about the user.
- **User Client:** Contacts the server presenting a user's certificate and obtains a list of groups, roles and capabilities of the user.
- **Administration Client:** Used by the VO administrators (adding users, creating new groups, changing roles, etc...)
- **Administration Server:** Accepts the requests from the clients and updates the Database.

8.18.2.1 User side operations:

VOMS uses user's proxy certificates that contain user authorization info from the VOMS server(s). This info is returned in a structure emulating an Attribute Certificate containing also the credentials both of the user and of the VOMS server and the time validity. All these data are signed by the VOMS server itself. In more detail:

1. The user and the VOMS server mutually authenticate using their certificates;
2. The user sends signed request to VOMS Server;
3. The VOMS Server checks correctness of user's request;
4. The VOMS Server sends back to the user the required info (signed by itself) in a structured form (Attribute Certificate);
5. The user checks the validity of the info received;
6. The user eventually repeats process for other VOMSs;
7. The user creates the proxy certificates containing all the info received from the VOMS Server into a (non critical) extension;
8. The user may add user-supplied authentication info (kerberos tickets, etc).

8.18.2.2 Administrator side operations:

The VOMS server is essentially a front-end to an RDBMS, where all the information about users is kept. The system is composed of the following parts (See Figure 84):

- **Voms-proxy-init:** The user client program, which contacts the server presenting a user's certificate and obtains a list of groups, roles and capabilities of the user.
- **Vomsd:** The VOMS server, which receives requests from *voms-proxy-init* and returns information about the requester user.
- **Edg-voms-admin:** Command line interface for VO administrators (adding users, creating new groups, changing roles, etc...)
- **Admin GUI:** Graphical user interface for VO administrators
- **Web browser – web-admin servlet:** Web based administrative interface for VO administrators
- **Admin-server:** The administrative server, which enforces the database consistency and the fine-grained access control. This service is used by all of the administrative interfaces.

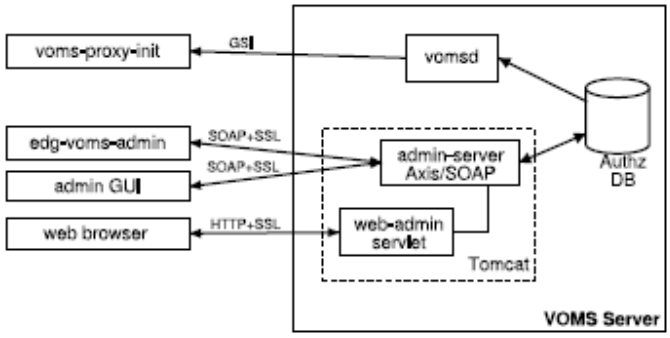


Figure 84 The VOMS System

In order to process this authorization information, the Gatekeeper, in addition to normal certificate checking, has to extract the additional information embedded in the proxy (AC). Access to VOMS server is given only to authenticated users via https. For the transition phase RP's can query the VOMS server. The VOMS Authorization Manager is capable of parsing and checking the VOMS AC and utilize its attributes in the authorization process.

The Administration server supports the SOAP protocol for connections, so that it can be easily converted into an OGSA service. It consists of the following services:

1. The Core, which provides the basic functionality for the clients;
2. The Admin, which provides the methods to administrate the VOMS database;
3. The History, which provides the logging and auditing functionality (the database
4. Scheme provides full audit records for every changes);
5. The Request, which provides an integrated request handling mechanism for new users and for other changes;
6. Web and command line administrative interfaces are available.

8.18.2.3 General security considerations

Compromising the VOMS server itself would be not enough to grant illegal access to resources since the authorization data must be inserted in a user proxy certificate (i.e. countersigned by the user himself). Hence the only possible large-scale vulnerabilities are denial of service attacks (e.g. to prevent VO users to get their authorization credentials).

The main security issue about proxy certificates is the potential of impersonations and the lack of a revocation mechanism; on the other hand these certificates have short lifetimes (in EDG implementation this is 12 hours, typically).

8.18.3 Globus Toolkit CAS

The Globus Toolkit normally uses existing local resource mechanisms for authorization. A user is authenticated and then mapped to a local identity (e.g., a Unix account) by a local configuration file (the “gridmapfile”). This mapping also serves as an access control check: if the user is not listed in the local mapping configuration, access to the resource is denied.

Once the user is mapped to a local identity, the Globus Toolkit then relies solely on local policy management and enforcement mechanisms to constrain the user’s actions to those allowed by local policy. This approach removes the fine-grained policy configuration and decision making from the GT services (e.g., GridFTP, GRAM) and allows the local operating system to act as a sandbox. Thus, administrators can use normal policy administration tools to configure policy. For example, a Globus Toolkit user is normally mapped to a local Unix

account. Standard Unix filesystem permissions, quotes, group memberships, and so forth are then used to configure and enforce policy.

Similar techniques could be used in conjunction with dynamically allocated accounts or virtual machine technology[357].

The classic Globus Toolkit authorization system described has the advantage of being easy for site administrators to understand and configure because it uses existing local policy management and enforcement mechanisms with which the administrator is presumably already familiar. In terms of supporting a large VO, however, the GT has several shortcomings:

- **Scalability:** Each personnel or policy change requires changing policy at each participating site;
- **Lack of expressiveness:** Native OS methods may not be expressive enough to support VO policies;
- **Consistency:** Different native OS methods may not support the same kinds of policies;
- **Distribution:** In order to maintain a consistent policy across the VO, each policy change must be propagated to each site involved. Any failure in propagation will cause an inconsistency in the policy.

To solve these problems, part of the Globus Toolkit team undertook the development of the CAS system.

8.18.3.1 CAS policy management

CAS allows a VO to maintain its own set of policies explicitly and communicate those policies to sites. The sites then combine their local policies (about what the VO is allowed to do) with the VO's policies (about what the individual user is allowed to do as a VO member) and enforce this combined policy.

The VO, through administration of the CAS server, maintains the VO's portion of this combined policy. This portion of the policy includes the following:

- The VO's access control policies regarding its resources: which rights are granted to which users (e.g., which users can read which files);
- The CAS server's own access control policies, such as who can delegate rights or maintain groups with the VO. These policies can be expressed at a fine-grained level (e.g., a user may be allowed to grant rights on only certain resources, or add users to only certain groups);
- The list of VO members.

The other part of this combined policy, the resource provider's (RP) policy regarding the VO, is maintained by the resource provider using the same native mechanisms used for non-VO users. For example, a site may create a local identity representing a VO and add local configuration mapping users presenting credentials from that VO's CAS server to that identity. The site would then use local mechanisms to set policy on the VO as a whole, for example, change file ownerships to allow the VO identity read and write access to a particular subset of the file system, or set file system quotas limiting the amount of space that the VO can use. A resource provider may use this mechanism to maintain policies for several VOs, each running its own CAS server.

[357] R. Figueiredo, P. Dinda, and J. A. Fortes. Case for Grid Computing on Virtual Machines. 23rd International Conference on Distributed Computing Systems, 2003.

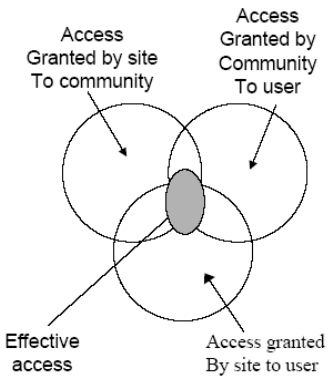
8.18.3.2 CAS policy enforcement

Resource providers participating in a VO with CAS will deploy CAS-enabled services (i.e., services modified to enforce the policy in the CAS credentials) onto resources they assign to the VO. A user wishing to access those resources first contacts the VO's CAS server and requests a CAS credential. The CAS server replies with a CAS credential that contains a policy statement of that user's rights, cryptographically signed by the CAS server.

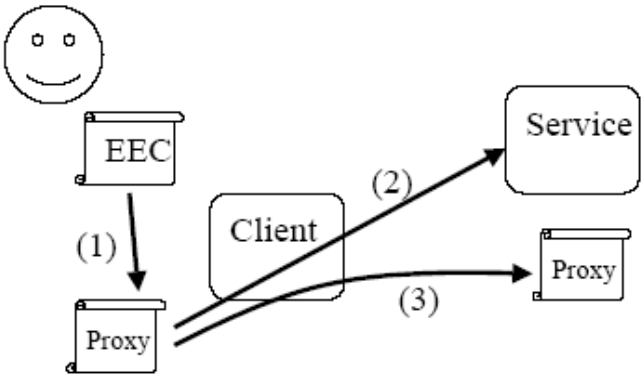
When making a request to the resource, the user presents the CAS credential. Upon receiving the CAS credential, a CAS-enabled service takes several steps to enforce both VO and local policy:

- Verify the validity of the CAS credentials (e.g., signature, time period).
- Enforce the site's policies regarding the VO, using essentially the same method as an unmodified server. However, the identity used when enforcing the site's policies is the identity of the signer of the policy assertion (i.e., the VO's CAS server), not the identity of the individual user authenticating.
- Enforce the VO's policies regarding the user, as expressed in the signed policy statement in the CAS credential.
- Optionally, enforce any additional site policies in regard to the user (for example, a site may keep a blacklist of end users who are not allowed to perform any action, regardless of any VO policy).

Thus, as shown in the Figure 85, the set of rights the user is effectively granted by these steps is the intersection of the set of rights granted by the resource server to the VO and the set of rights granted by the VO to the user



Effective Access Control Policy in CAS



Proxy certificate creation and delegation

Figure 85 Overview of CAS effective policy access and proxy certificate creation and delegation process

CAS is designed and implemented to work with the Grid Security Infrastructure (GSI)[358,359], which provides the security functionality of the Globus Toolkit. For authentication, GSI uses X.509 proxy certificates[360] to provide credentials for users and to allow for delegation and single sign-on.

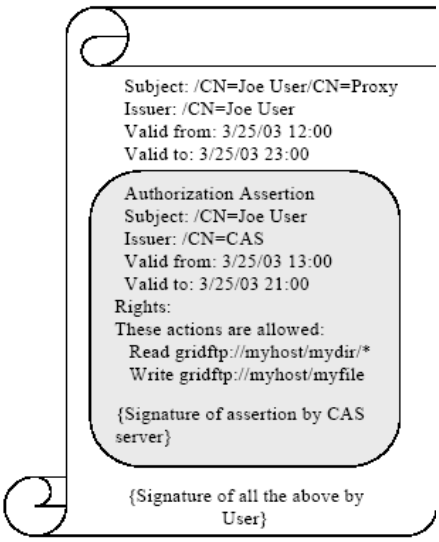
[358] Butler, R., Engert, D., Foster, I., Kesselman, C., Tuecke, S., Volmer, J. and Welch, V. A National-Scale Authentication Infrastructure. IEEE Computer, 33 (12), 2000, pp. 60-66.

[359] I. Foster, C. Kesselman, G. Tsudik,. and S.Tuecke. A Security Architecture for Computational Grids. ACM Conference on Computers and Security, 1998, pp. 83-91.

[360] S. Tuecke, D. Engert., I. Foster, V. Welch, M.Thompson, L. Pearlman, and C. Kesselman. Internet X.509 Public Key Infrastructure Proxy Certificate Profile, IETF, 2003.

Proxy certificates are similar to the standard X.509 end entity certificates (EECs)[361] from which they are derived. The primary difference is that users issue proxy certificates to create a short-term (e.g., one-day) delegation of their rights to another entity (e.g., a process running on their behalf) as opposed to EECs, which are issued by certificate authorities to assign a long-term identity. The short-term nature of proxy certificate credentials allows them to be more lightly protected than the credentials associated with the long-term EEC credentials that are used to create them. For example, proxy certificate credentials are usually protected with local file-system permissions as opposed to being encrypted. This approach allows their use for single sign-on and by unattended processes.

As shown in Figure 85, the creation of a proxy certification can be local to a single resource (usually for use by subsequent clients, to enable single sign-on) [step 1]. The proxy certificate can then be used with GSI to authenticate to a remote service and establish a secure connection [step 2]. Proxy certificates can also be used to issue other proxy certificates, allowing for a chain of delegations, as in [step 3] where a proxy certificate is created across the GSI-secure network channel (to delegate rights to the service so it can act on the original user's behalf). Normally a proxy certificate allows its bearer to assert the full rights of the bear of the EEC that issued it. This delegation can be restricted through the use of a policy embedded in the proxy certification. An example of CAS (prototype alphaR2) credential, simplified for clarity, is provided in the following figure. Note that a user proxy is combined with a policy assertion issued by the CAS server.



Notes:

- The policy assertions issued by the CAS server includes a set of access rights along with the user's identity.
- This assertion allows a resource to verify the user's VO membership, by the fact the assertion has the user's identity in it, and the user's rights within the VO as listed in the assertion.
- The assertion also includes a validity period and signature allowing resources to verify the validity of the assertion.
- This separation of CAS policy assertion from user proxy credentials allows resources accepting these credentials to identify the user involved in a request as if they were using normal GSI credentials.
- This strategy in turn allows a site to use normal mechanisms for auditing and to apply an additional level of policy enforcement based on the user's identity. The applied policy becomes the intersection of the rights granted to the community by the site, the rights granted to the user by the community and any restrictions placed on the user by the site.

Figure 86 Example of CAS credential; the model combines a proxy certificate issued by the user with a signed policy assertion issued by the CAS server.

[361] CCITT Recommendation X.509: The Directory-Authentication Framework. 1988.

8.18.4 PRIMA (Virginia Polytechnic Institute)

In this section we provide an overview of PRIMA based on [362] and [363].

PRIMA [363] describes a Privilege Management model, which empowers users, establishes trust on a user-to-user basis, and is fully decentralized.

PRIMA empowers users. This means that subjects (users) can possess and delegate fine-grained privileges to resources for which they are authoritative to other subjects (i.e. subjects can act as subject authorities). Resource authorities can use the same mechanisms to grant privileges to users and to issue policy statements for their resources.

PRIMA allows for the establishment of individual trust-relationships between two entities from different administrative domains without the need for formal collective trust between the domains themselves.

PRIMA is designed for fully distributed operation. This allows PRIMA to support the creation of small, transient and ad-hoc communities without imposing the requirement to deploy group infrastructure components like community servers. However; if such services are available (for example a VOMS attribute server) the PRIMA model can incorporate them and provide additional services and improved manageability for larger groups.

PRIMA was developed following the least privilege access and separation of concern principles. In PRIMA, a service request by a subject may not be awarded any access permissions based on the subject identity, rather; access permissions are granted based on subject attributes (privileges) that the subject holds and chooses to group with this request. The combination of privileges from multiple sources enables collaborative scenarios where participants can contribute their resources to the community. The ability to specify exactly what privileges are to be used with a specific access permits a subject to define access requests with the minimum privileges required. This minimizes the risk a subject has when utilizing a less trusted resource or service and protects services from accidentally over- or misusing resources, e.g. in the case of software failures.

While PRIMA binds access rights directly to subjects, role based access control (RBAC) is also possible. In PRIMA, privileges can be grouped (and thus constitute a role), and the group of privileges can then be applied to a set of users. Furthermore the issuer of privileges (subject authority) can apply rules on how subjects may combine these privileges with others they hold.

One of the fundamental differences of PRIMA to other existing system is that PRIMA relies on the creation, configuration and management of user accounts on-demand, based on subject privileges issued by authoritative entities such as resource administrators, resource owners, or group and project leaders. In the environments where dynamic user accounts may not be permitted, the creation and use of dynamic accounts is effectively disabled on this resource and static accounts are required.

PRIMA's enforcement is based on controlling the environment within which the application will execute.

8.18.4.1 PRIMA Architecture

The PRIMA system architecture is shown in Figure 87. Policy authorities use the *Policy Creator* to create self-contained policy statements, which are provided to the PRIMA policy

[362] Markus Lorch, David Adams, Dennis Kafura, Madhu Koneni, Anand Rathi, Sumit Shah, "The PRIMA System for Privilege Management, Authorization and Enforcement in Grid Environments", Proc. 4th Int. Workshop on Grid Computing - Grid 2003, 17 November 2003 in Phoenix, AR, USA.

[363] Markus Lorch and Dennis Kafura, "Supporting Secure Ad-hoc User Collaboration in Grid Environments", Proc. 3rd Int. Workshop on Grid Computing - Grid 2002, Pages 181 - 193, Baltimore, USA, November 18th, 2002

decision points on the resources affected by the statements. Attribute authorities leverage the *Privilege Creator* to create self-contained privilege attributes bound to specific subjects (users). These privileges are then provided to the users which in turn select a subset of their privilege attributes to be used with a specific (set of) grid request(s) and group these privileges with their authentication credential using the *Privilege Combinator* tool. The resulting grid credential is then used by the subjects to make least privilege grid requests.

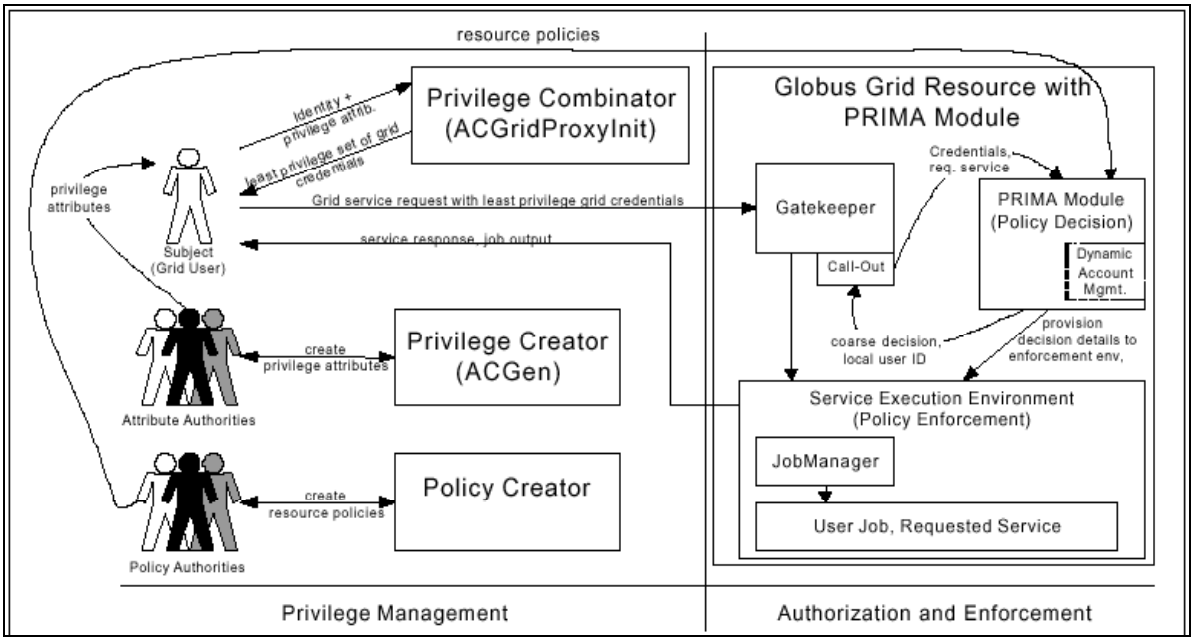


Figure 87 PRIMA Architecture

For authorization and enforcement, the PRIMA implementation is integrated with the Globus Toolkit as a modular authorization component that is called by the resource gatekeeper once a subject has requested a service. The PRIMA module then acts as a PDP and makes a coarse-grained authorization decision based on the supplied user attributes and the local resource policy. If this decision is positive, the mapping to a local user account, which may be dynamically allocated, is performed. The local user account is then configured with the fine-grained privileges, such as file access permissions, user quotas or network access and the gatekeeper is provided with the local user identifier. In the next step the Globus gatekeeper instantiates the requested service through the JobManager, another Globus component, which will eventually execute and monitor the user's job. The fine-grained privileges applied to the execution environment will remain active for their full lifetime as specified in the privilege attribute that was originally provided. Another PRIMA component, the Privilege Revocator (not shown in Figure 87) is present on each PRIMA-enabled resource and watches over the validity period of dynamically allocated user accounts and all fine-grained privileges revoking the privileges when they expire.

As the PRIMA model only deals with subject attributes that can convey additional privileges, trusted repositories which guarantee that all restrictive attributes are available to the PDP are not needed. Currently, the resource policies are stored on the resource directly; in future, an extension should leverage a site-wide centralized policy decision point to evaluate resource access decisions.

8.18.4.2 Privilege Attributes

The PRIMA system uses X.509 attribute certificates (AC)[364] to securely bind privilege attributes to subjects and policies to resources.

The privilege statements are contained as the payload in attribute certificates. A single attribute certificate may contain a set of privileges and can also be bound to a set of entities. This allows creation of attribute certificates that define simple role credentials, containing a set of privileges, that are allocated to a specific role and can then be bound to a group of user subjects (the role holders). The holder field in the ACs corresponds to the entity that is holding the privileges, the issuer field reflects the authoritative entity that created and signed the AC. A validating party has to verify that the private key used to create the signature is associated with the issuer (via the issuers X.509 public-key certificate) and that the issuer was authoritative for the attribute. The validity field contains a time frame (not before – not after) that denotes the validity period.

8.18.4.3 The Privilege Creator

The Privilege Creator, ACGen, is a graphical user tool implemented in Java using the IAIK Java Cryptography Provider. The "Issuer" and "Holder" entities of the AC are defined through the X.500 distinguished names (DN). The issuer uses his end-entity credentials to sign a newly created AC. ACGen allows issuers with multiple credentials (e.g. identity certificates from different CAs) to choose the appropriate credential from a given list. The holder DN can either be acquired by searching an LDAP server or through direct interaction between the users (e.g. through e-mail). The current implementation is limited to specifying a single holder DN. Future implementations will allow for the specification of a set of holder DNs to support the notion of roles.

The privileges are specified in the AC as a text blob using a simple, platform independent format that consists of the privilege type, the hostname the privilege applies to, the path to the resource and a comma separated list of the actual rights. Wildcards in host and pathnames are permitted by the format and allow for a single privilege statement to apply to a set of resources. Currently supported privilege types are FilePrivilege, AccessPrivilege, or NetworkPrivilege. In the case of file privileges the set of supported rights are read, write, and execute (applies also to directories). In the case of a NetworkPrivilege, the network port and read or write privileges are possible. In future implementations, a more powerful language will be used. An extension of XACML is being considered to allow for the specification of privileges very similar to the way policies are currently specified in XACML.

Created attribute certificates are saved in a PEM formatted file together with the certificate path of the issuer. An encapsulation boundary of "----- BEGIN ATTRIBUTE CERTIFICATE -----" is used to distinguish ACs from identity certificates, which are denoted by "----- BEGIN CERTIFICATE -----".

8.18.4.4 The Policy Creator

The Policy Creator is a variation of the Privilege Creator except that the created ACs contain policy statements in XACML issued by a policy authority. The AC holder is the resource to which the policy applies. The initial prototype provides a basic graphical interface to aid the user in the creation of the policy.

8.18.4.5 The Privilege Combinator

The Privilege Combinator (ACProxyInit) allows the subject to group selected privileges (attribute certificates) with his identity by embedding them during the proxy certificate's creation. Issuer information on the ACs (the AC issuer certificate path) is also embedded in the proxy certificates to aid the resources in the process of AC validation. Embedding ACs into Proxy certificates as X.509 certificate extensions achieves a secure endorsement by the

[364] S. Farrell, R. Housley, "An Internet Attribute Certificate Profile for Authorization", IETF RFC, April 2002

user about which of his privileges (ACs) should be used for the specific access (i.e. the user bundles the delegated identity used for a specific access with the privileges to be used by this delegated identity and signs both of these components).

8.18.4.6 The Authorization Module

The PRIMA Authorization Module handles the task of making an authorization decision based on resource policy and provided subject attributes. Once a decision has been reached the mapping of the subject to a local user account is performed which may include the allocation of a dynamic user account. This mapping constitutes a coarse access decision. Before the allocated local user identity is returned to the Globus gatekeeper component the fine-grained components of the access decision (individual privileges) are provisioned to the enforcement mechanisms that later control resource access via the execution environment.

As an extension of the Globus gatekeeper, an authorization callout interface is developed (and is available). It allows a dynamically loaded authorization module (DAM) to authorize access and perform the mapping of an authenticated subject DN to a local user account.

The GSI leverages the Generic Security Services API (GSS-API)[365] for authentication. The interface thus passes a reference to the established GSS-API security context, together with the requested service name to the DAM. PRIMA DAM utilizes standard GSS-API calls to extract the authenticated subject ID which will later be used for mapping to a local user account.

In order for the DAM to access subject attributes an extension to the GSS-API is implemented. The extension not only returns the attributes in a mechanism independent format but also validates the attributes with respect to their validity period and the attribute issuer. Therefore, the PRIMA DAM code is completely independent of the underlying format of the attribute

8.18.4.7 User Account Management

The dynamic creation, configuration, and management of user accounts at grid resources through privilege delegation is briefly described here and outlined in Figure 88. Dynamic accounts are assigned from a pool created by the system administrator. These accounts are similar to a standard “nobody” accounts that disallows direct login and has minimal rights. When a dynamic account privilege is presented, the system will check for an existing account, matching the distinguished user name and project identifier, and map that user to the existing dynamic account. During this mapping an extension can be made to the lifetime of the account. If a user holding a dynamic account privilege is not found then a new account is assigned from the pool of available accounts.

[365] T. Ryutov, G. Gheorghiu, C. Neuman. “An Authorization Framework for Metacomputing Applications “. Cluster Computing, Vo l. 2, pages 165-175, 1999.

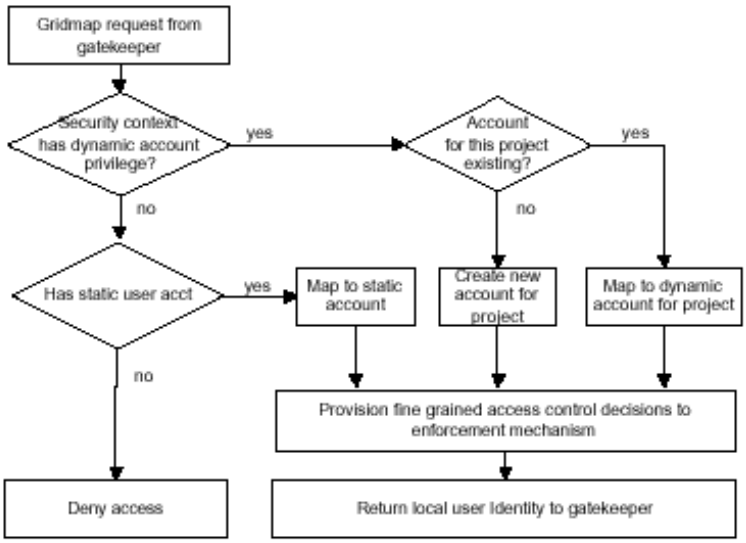


Figure 88 Account Mapping and Allocation Logic

Management of delegated dynamic accounts is controlled through a single file with the following space delimited format, “Issuer Distinguished Name” “Holder Distinguished Name” “User-ID” “Project ID” and “Expiration Date” where each User -ID maps a holder’s distinguished name to a particular account from the pool. Accounts remain valid for the time period until the expiration date is reached at which point they can be automatically cleansed and returned to the dynamic user account pool. Standard UNIX advisory record locking is used to ensure that the one to one mapping integrity of dynamic account delegation is maintained. Using this method the administrator retains control over the number of accounts available at any given time to dynamic grid users and has the tools required to satisfy accounting needs based on the issuer, project, or holder distinguished name.

8.18.4.8 Policy Enforcement Mechanisms

Enforcement of fine-grained access rights is defined as the limitation of operations performed on resources by a user to those permitted by an authoritative entity. In PRIMA these actions are defined in privilege attributes that the subject conveyed to the grid resource with his service request. The PRIMA authorization module verifies the validity of the privileges and provides all fine grain privileges, together with their verified, local issuer names (the local user ID of the issuer) to the enforcement component. It is now the task of the enforcement component to apply these privileges to the local user environment and thus configure a least privilege execution environment for the requested service. If the environment already exists from a previous access, only the changes (e.g. additional privileges) have to be applied.

The current implementation supports two enforcement mechanisms: POSIX.1E file system access control lists[366] which are available on prevalent Unix operating systems and XML-based Grid Access Control Lists (GACLs) which were developed as part of the Slashgrid project[367] and are available for Linux. The selection of the enforcement mechanism is completely transparent to the PDP and depends solely on the local configuration at the

[366] “IEEE standard portable operating system interface for computer environments”, Withdrawn IEEE Draft Standard 17, Posix 1003.1, 30 Sept. 1988, available from http://wt.xpilot.org/publications/posix.1e.

[367] http://www.gridpp.ac.uk/authz/slashgrid/, visited 2003-05-02

resource. The reason for use of Slashgrid is twofold, first Slashgrid provides an alternative enforcement mechanism in case POSIX ACLs are not available, and second Slashgrid can be used for long-term storage of user data, where it would be difficult to maintain consistent DN-UID association and security.

POSIX.1E file system ACLs extend the standard file permissions to allow control on a user-by-user basis. POSIX ACLs do not support privileges for multiple groups. POSIX ACLs associate privileges to UNIX UIDs. Figure 89shows a typical ACL listing specifying additional access permissions for users abazaz, kafura and akarnik for a file owned by user mlorch. The enforcement mechanism operates as follows. First, it verifies that the issuer is authoritative for the file or directory tree to which a privilege applies. This means checking that the issuer is the file owner. Second, it applies the privilege by modifying the corresponding ACL and storing the change in a system wide privilege database with the expiration time for later revocation.

```
# file: /projects/sim/sim.sh
# owner: mlorch
# group: users
user: : rwx
user: sshah: r-x
user: kafura: rwx
group: : r—
mask: : rwx
```

Figure 89 POSIX File System ACL

```
<?xml version="1.0" ?>
<gacl version="0.0.1">
<entry>
  <person>
   <dn>/CN=Markus
Lorch/O=vt/C=US</dn>
  </person>
  <allow>
    <write/><admin/><read/><list/>
  </allow>
</entry>
<entry>
  <person>
   <dn>/CN=Sumit Shah/O=vt/C=US</dn>
  </person>
  <allow>
```

Figure 90 A Grid Access Control List

Grid Access Control Lists (GACLs) are an XML representation of access control lists with additional features provided to accommodate users belonging to different groups. Figure 90shows a typical GACL listing specifying the issuer, the user, and the associated permissions. Slashgrid uses GACLs in a grid-aware file system. Subject privileges with respect to files and directories are associated directly with the subject-distinguished name (DN) provided in the certificate rather than a local user ID (UID). The enforcement mechanism operates as follows. First, it verifies that the issuer is authoritative for the file or directory tree that a privilege applies to. In the case of GACLs, this is a check to verify that the issuer has the “admin” right on the object. Second, it applies the privilege by modifying the corresponding ACL and storing the change in a system wide privilege database with the expiration time for later revocation.

8.18.4.9 The Privilege Revocator

The Privilege Revocator, present on each PRIMA enabled resource, automatically revokes privileges and dynamic user accounts when they expire. The privileges are stored in a shared data file that is periodically checked by the Privilege Revocator. Each line in the data file consists of the object the privilege is intended for, the holder of the privilege, the permissions for the object conveyed in this privilege, and the expiry date and time. When an individual fine-grained privilege, such as write access to a specific file, expires, the Privilege Revocator simply removes this right from the respective ACL. In the case of privileges for a dynamic user account, the Privilege Revocator will notify the account holder of an upcoming revocation one week before the account privilege expires. This gives the account holder enough time to present a new privilege for his account (with a longer lifetime) to extend his account lease and keep the data and other privileges associated with this account. If an account privilege expires the account is cleansed, returned to its initial state (all associated fine-grained privileges are revoked, despite their validity) and the account is made available again.

8.18.5 GRASP Security Infrastructure

GRASP security infrastructure has been developed in parallel (and independently) to EDG security and VOMS, in the context of the European Project GRASP (www.eu-grasp.net) that has been exploring a OGSA[368] based infrastructure for Enterprise Grids and is being implemented (2002-2005) over Microsoft .Net leveraging on OGSI.NET implementation of the University of Virginia and Microsoft WSE extensibility on .NET.

The GRASP environment can be understood as a Virtual Organisation ecosystem, where services and resources can be created on demand and shared among different *Hosting Environments* (HE) for the purpose of securely enacting a task associated with the execution of an application instance. Each HE can be understood as a single administrative domain, whose security is managed by a dedicated local security manager (LSM). A HE consists of a number of hosts each of which provides one or more *Factory* services that are able to realise potentially transient service instances consuming resources in that particular Host and contributing to the enactment of a common task in collaboration with other service instances in the same or different HE. Each HE also provides a dedicated service, called a *Service Instantiator* (INS), which receives all requests for the creation of new services within the HE and manages their realisation by selecting a suitable Host within the HE and invoking the *Factory* of that Host. Selection is based on criteria related to local security policy, SLA obligations and resource availability information[369].

The purpose of the GRASP Security Infrastructure is on the one hand to segregate the administrative services managing each HE and the rest of its resources from the service instances it contributes to the enactment of a specific collaborative task, and on the other hand to allow the dynamic establishment of a virtual perimeter across the virtual community of services and resources enacting a collaborative task across multiple HE.

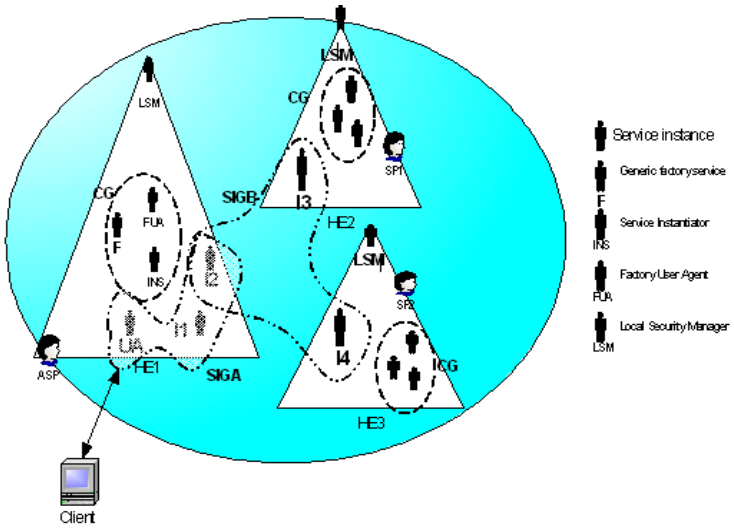


Figure 91 Overview of a Grid Application Service Provision environment with CG& SIGs

Notably, one can distinguish two types of communities that differ in purpose and kind of entities they bring together:

[368] I. Foster, C.Kesselman, J. Nick, S. Tuecke. Grid Services for Distributed System Integration. Computer, 35(6), 2002.

[369] Wesner S., Sehran B., Dimitrakos T, Mac Randal D., Laria G., Ritrovato, P., Towards a platform enabling Grid based Application Service Provision. eChallenges Conference e-2004 (submitted)

- A **Core Group (CG)** includes all management services within a single HE. These are always distinguishable from those offered by the HE to virtual collaborations.
- A **Service Instance Group (SIG)** includes all instances contributed by several HE for the purpose of enacting a collaborative task.

Figure 91 shows three HE each of which has a CG and contributes to some SIG (i.e. [UA,I1,I2], [I2,I3,I4]), to enacts distinct collaborative tasks.

8.18.5.1 Overview of the dynamic security perimeter architecture underpinning GRASP-SI

This security architecture underpinning the GRAPS Security Infrastructure is being developed with two main goals in mind:

- Enabling communication within dynamically created communities, that is secure, scalable, accountable, robust and independent of network topology.
- Enforcing dynamic security perimeters, which adapt to the evolution of a virtual community (in terms of membership, performance and security policy).

These goals are addressed through the following means:

- Attribute Certificates to manage group membership and privileges.
- Role based security policies describing permissions, prohibitions and obligations within the collaboration teams, set by, and negotiated between, the community mangers.
- Mechanism for distributed enforcement of the security policies at each service instance that protects individual members within a community and the community as a whole.

The logical structure of the security architecture distinguishes several roles and interaction modes that capture the required functionalities (Figure 92). These include:

Group-peers (or peers) are the service/resource instances that are created within a HE for the purpose of contributing to the enactment of a collaborative task. Each peer possesses a security token (resembling a Public Key Certificate - PKC) as a proof of registration with the overall environment. This is presented whenever basic authentication of the peer is required, and the key-pair itself may be used for encrypting or signing parts of the messages it may exchange.

At a Group-peer's initial setup, LSM assigns each peer to one or more *organisational roles* and provides it with attribute certificates (AC) as a proof that it can assume these roles, subject to the security policy of the HE. Such *organisational roles* define the default security settings (e.g. default authorisations, secure communication settings) of a peer and are bound to the administrative domain of its HE; they are not necessarily recognised outside that HE. A Group-peer can interact with other peers by creating a new community or joining an already existing one (therefore becoming a Group-member). At any time a Group-peer can be a member of one or more communities. Introduction of a Group-peer to a community has to be agreed by both its LSM and the corresponding Group manager.

Local Security Managers (LSM) are responsible for the population of "Group-peers" within a HE. A LSM sets the default security configuration of, and controls the enforcement of any security policy on any member of the peer population that it is responsible for.

Group Managers are responsible for managing Group membership, and coordinating the level of security within a community. Group managers define security policies for the whole Group, and coordinate the enactment of such policies. However the actual enforcement of Group policies is delegated to the corresponding LSM of the contributing HE, who have the control over the security enforcement agents the corresponding peer. Group managers are realised as dedicated functionalities exposed by the security administration of selected HE. In particular, selected LSM may be authorised to assume such a role or selected HE may be able to provide Group Manager factory services.

Group Members are Group-peers that have joined a specific Group (i.e. the CG of a HE or a SIG). In addition to possessing AC to prove the *organisational role* that they may assume within their HE, for each SIG, the Group Managers assign each SIG member to one or more *collaboration roles*, and provides it with attribute certificates (AC) as a proof that it can assume these roles in interactions with other members of that Group. Such *collaboration roles* are bound to a Group: they are recognised only by the HE that contribute Group peers to that Group and only within the scope of interactions in the context of that Group; they are not necessarily recognised in any other HE or in relation to different transaction contexts.

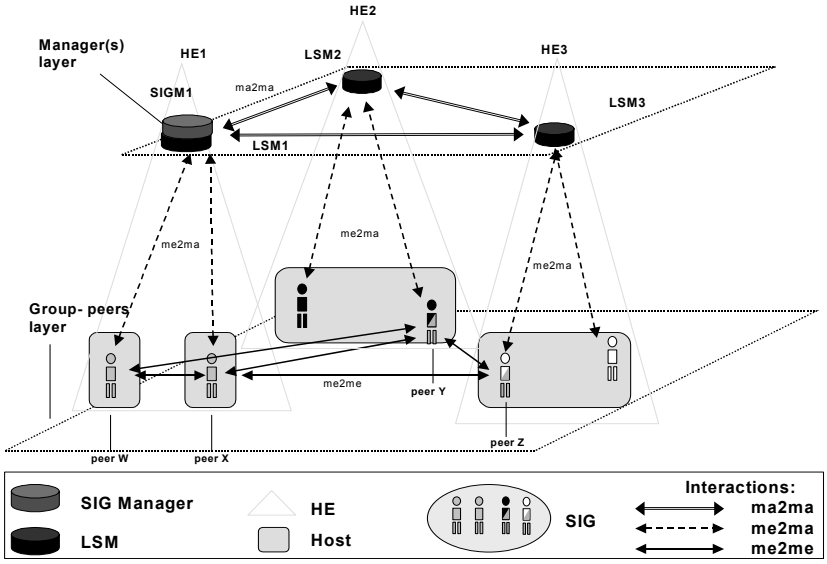


Figure 92 Main roles and interactions of the dynamic security perimeters architecture

Basic Interactions: Once a Group-peer joins a SIG, it interacts directly with other SIG members, without manager's involvement. SIG members are informed of each others identity and location and interact on a peer-to-peer basis through message exchange (e.g. SOAP). Depending on the purpose of each interaction they are required to embed different kind of certificates in the messages they interchange. Occasionally, interactions between managers may take place as appropriate, in order to support creation of SIGs across multiple HE. Figure 3 illustrates three main types of interactions in the SIG environment:

- ***me2me*** (*member-to-member*): direct p2p communication between Group Members. This requires a (unique) group certificate, which is issued by the Group Manager.
- ***me2ma*** (*member-to-manager*): interactions between a member and its LSM about Group management (e.g. local distribution of group policy updates, request to create a new group or join/leave an existing group, etc.). They require member's **PKC**.
- ***ma2ma*** (*manager-to-manager*): direct p2p communication between managers (including Group Managers and LSM) about security management (e.g. membership management, policy negotiation, policy update, group membership management etc.). They require a ***ma2ma*** certificate.

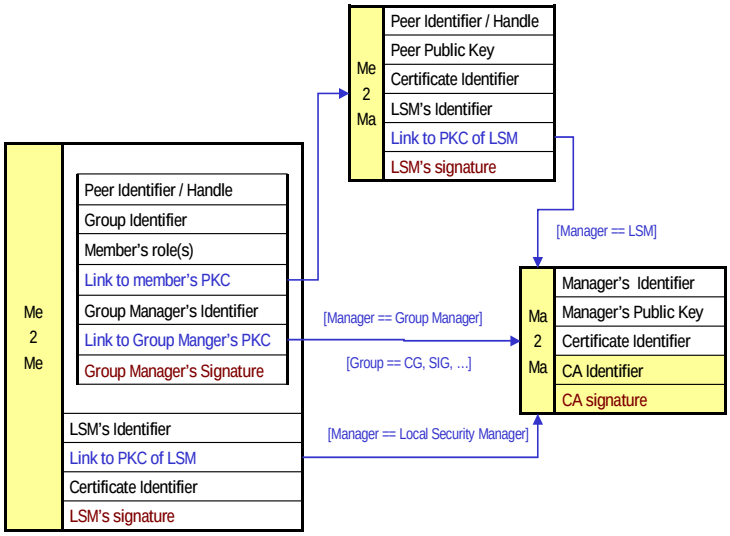


Figure 93 Basic Certificate Type Structure

Figure 93 highlights different roles and interaction modes. Figure 94 provides an overview of the corresponding certificate structures.

Security Enforcement Agents (SEA) Security policy enforcement takes place at each end-point by means of a multilayer shell that effectively protects each Group-peer (Figure 94). SEA virtualises the security enforcement mechanisms that are attached to each Host (Network layer) or Group-peer (Application layer) within a HE. Conceptually, this extends the notion of a distributed firewall[370] by introducing further layers of control for member authentication and fine-grained role-based access control to service instance data and operations that are enforced by dedicated SEA at each peer.

A Host-SEA virtualises the Host-instance of a distributed firewall that protects the whole HE and is administered by LSM. A peer-SEA, virtualises the security enforcement mechanisms of that Group peer. For every interaction between peers, the following per-message security checks are performed at a distributed security perimeter instance[371]:

- For an outgoing message, the corresponding context is identified and the recipient's identifier is validated against community membership records. If validation fails, the message is destroyed and a failure notification is sent to LSM. If validation succeeds, then the intended action (e.g. encapsulated by a SOAP-RPC) is checked against the security policies associated with the sender's roles in that context. If compliance fails, the intended action is replaced by a failure notice and the LSM is notified. If compliance is confirmed, the appropriate certificate proving the peer's role is attached to the message. Finally certain parts of the message are encrypted or signed, as required by the corresponding security policy. Finally, the message is sent via the distributed firewall instance protecting the Host accommodating the sender.
- For an incoming message, at first the packet stream is examined by the distributed firewall instance protecting the Host that accommodates the recipient. Then, the context of the corresponding interaction is identified, and the sender's certificate is validated. If validation fails, then the message is destroyed and the LSM is notified. If validation is successful, then the message content is analysed against the security policies

[370] Bellovin S.M. – Distributed Firewalls. login, pp.37-39, Nov 1999., www.research.att.com/~smb/papers/distfw.pdf

[371] The use of Web/Grid services technology entails that all interactions reduce to message exchanges between peers and that all actions including method invocations are encapsulated as data in the messages exchanged.

associated with the role and context claimed. Each intended action (e.g. encapsulated by a SOAP-RPC) is checked prior to its execution. Invocations corresponding to authorised actions are realised, whereas if an unauthorised action is requested then a fault is logged and the LSM is notified.

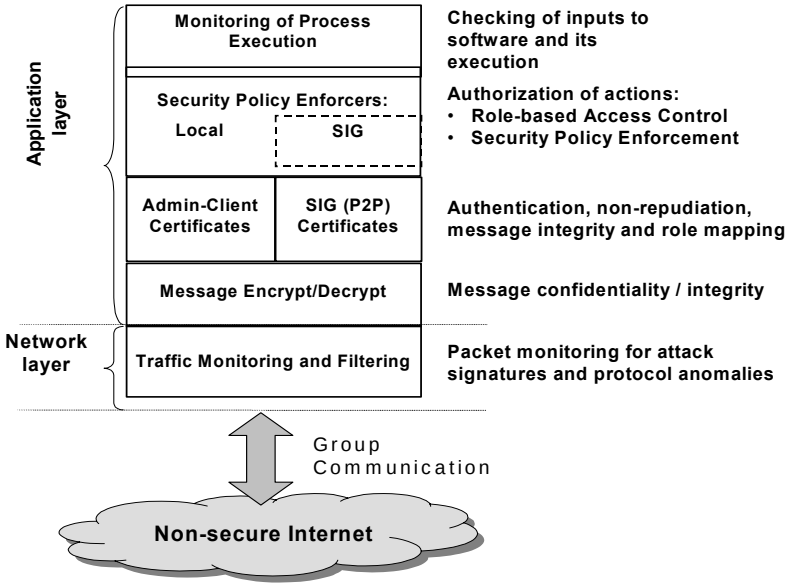


Figure 94 Overview of the security enforcement model

Figure 94 provides an overview of the security enforcement scheme. Notice the separation between the network and service/application layers of the security perimeter that reflects the functional separation between SEA explained in this section. A third dimension concerns monitoring of the process execution (via "monitors" distributed to each service, which communicate events of interest to their group manager, or a trusted third party, where analysis can be performed based on evidence/data collected by all Group members).

For more details on the enforcement architecture see [372]. Notably the clear distinction between policy specification, policy decision and policy enforcement is a key feature that brings flexibility and scalability.

8.18.5.2 Overview of the associated security policy management scheme

The policy management model associated with this security solution can be viewed from two (intersecting) viewpoints: organisational and collaborative:

From an *organisational viewpoint*, the environment consists of a number of administrative domains that accommodate resources, which may engage in the collaboration(s). The administrative domain of each HE is organised in a hierarchical manner, where one Local Security Manager (LSM) defines the basic *organisational policy* (i.e. compulsory policy statements, that underpin any collaboration), and registers the services or resources it administers by defining their identifiers (e.g. service handles / local addresses) and issuing public key certificates bound to that identifier. At this level, policy contains basic constraints that have to be met by every entity under LSM administration, irrespective of the subsequently defined collaboration policy. At a Group peer's initial setup,

[372] Djordjevic I., Dimitrakos T., Phillips C. "An Architecture for Dynamic Security Perimeters of Virtual Collaborative Networks" IFIP/IEEE NOMS2004.

LSM assigns each peer to one or more *organisational roles* and provides it with Attribute Certificates (AC) as a proof that it can assume these roles. Such *organisational roles* define the default security settings (e.g. default authorisations, secure communication settings) of a peer and are bound to the administrative domain of its HE; they are not necessarily recognised outside that HE. This part of the architecture does not in itself provide the infrastructure for the formation of communities that enable peer-to-peer collaboration for the enactment of a common task. It provides, however, a basis for negotiation on creation or joining such a community, as well as the base line security settings that precede any collaboration policy.

From a *collaboration viewpoint*, each community of group-peers (be it a CG or a SIG) contains a number of group-peers, which are brought together in order to enact a collaborative task. A Group Manager is assigned to each such community, and security administration becomes a collaborative task that is coordinated by the group Manager and enacted by a community of administrative security services that includes the LSM of each contributing HE, in addition to the Group Manager. *None* of these administrative security services is a member of the group-peer community that they manage.

Prior to SIG creation, the LSM of the contributing HE negotiates the *collaboration policy*. The group Manager is responsible for coordinating this negotiation. In addition, the group Manager is responsible for managing community membership and the defining the role of each Group member: The Group Managers assign each SIG member to one or more *collaboration roles*, and provides it with attribute certificates (AC) as a proof that it can assume these roles in the context of p2p interactions within that Group. Such *collaboration roles* are bound to a Group: they are recognised only by the HE that contribute Group peers to that Group and only within the scope of interactions in the context of that Group; they are not necessarily recognised in any other HE or in relation to other transaction contexts.

All the communication between Group-members and their Group Manager (e.g. regarding requests/responses for join/leave a SIG, or any policy updates) is always directed via their LSM, which is responsible to endorse the community management action that applies to the peers of its HE, and may choose to refuse forwarding of any message encapsulating interactions that violate the organisational policy of its HE[373].

8.18.5.3 Security perimeter dynamics: Overview of the community life-cycle model

In this section we further explain the security perimeter solution presented in this paper by focusing on the basic interactions and security functionalities that underpin its dynamic behaviour. To simplify presentation we describe the main interactions following the life cycle model of a community:

Creation and expansion can be initiated by any Group-peer (wishing to join a community) or by the Group Manager. P2P interactions at the Group-peer level are permitted only within already established communities. All legitimate communication between Group-peer and Group Manager is channelled via the Group-peer's LSM. Interactions between Group-peer and Group Manager can be understood as happening within a secure channel with two end-points (Group-peer, Group-Manager) and a single intermediary (Group-peer's LSM). The LSM is able to check, endorse or object and forward or reject a request from the Group-peer to the Group Manager. However the LSM cannot change the original content of the Group-peer's message. To ensure this, these messages are digitally signed by Group-peer. Interactions leading to a SIG creation and/or expansion are presented in the following Figure 95:

[373] Communication with Group Manager via a Member's LSM can be realized by taking advantage of SOAP header blocks that allow restricting the message path by addressing a message to an ultimate recipient (viz. Group manager or Group member) while explicitly specifying an intermediary (viz. LSM) who must understand and process designated header blocks in accordance to an explicitly specified role.

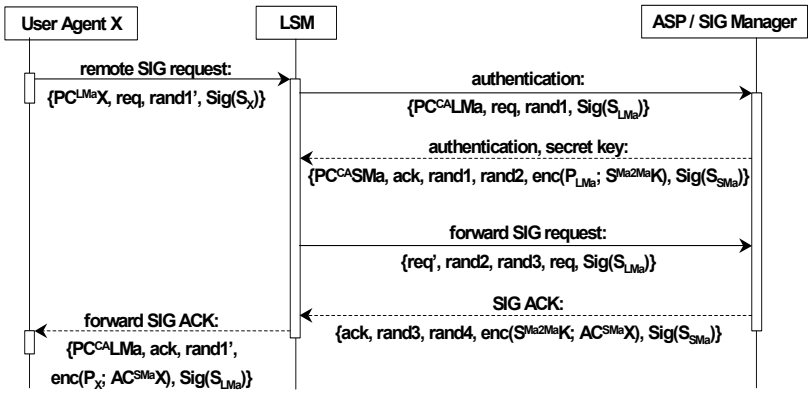


Figure 95 Interactions to lead a SIG creation and/or Epansion

Operation: Once a community is formed, Group members engage in p2p interactions. These interactions are restricted to Group members only and neither require nor entail any involvement of the Group Manager or LSM. Communication is performed by means of (typically SOAP) messages, which may be encrypted with session keys exchanged between SIG members at the beginning of a session. Legitimate messages must have embedded a (*me2me*) Group certificate. Patterns of interactions between SIG members are summarised in the following Figure 96:

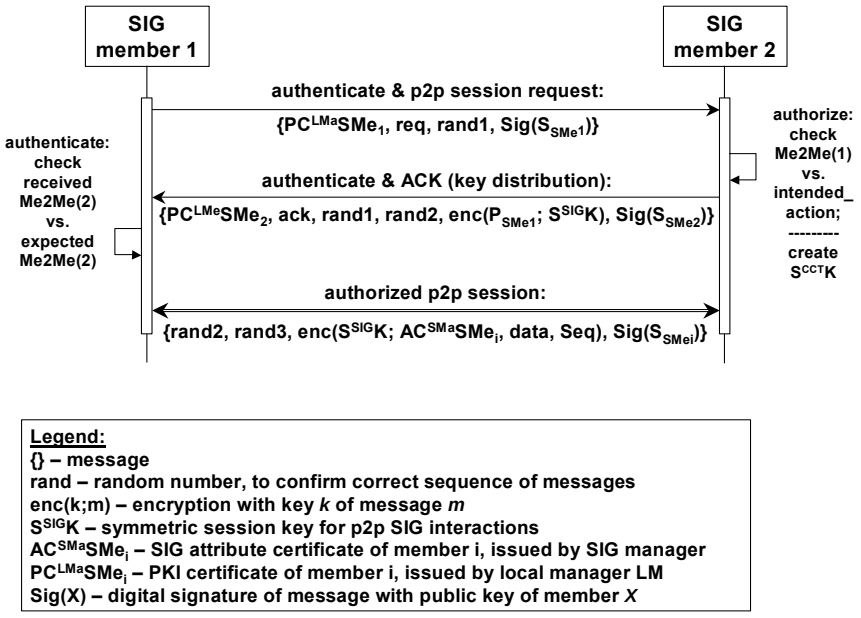


Figure 96 Patterns of Interactions between SIG Members

Changes to security policies, as well as the introduction of new Group members and the departure or expulsion of existing Group members, necessitate interactions between the Group Manager and some LSM. These are p2p interactions between peers at management level. In analogy to the interactions between SIG members, interactions among the Group Manger and the corresponding LSM require using a *ma2ma* certificate and may be encrypted.

Shrinkage occurs when a member leaves the community. This can be caused by *(i)* a peer that has reached the end of its operational life-time and is about to be destroyed, releasing the resources it occupies; *(b)* a peer that is forcefully destroyed (e.g. a user logs off); *(c)* a peer is expelled from the Group[374]. In any of the above cases an update of the Group membership is issued by the Group Manager and the and the corresponding LSM are instructed to update the SEA configuration so as to ensure that any interaction, in the context of that Group, between peers that have left the group and remaining Group members is blocked.

Dissolution: This is effectively the departure of the last remaining member of a Group or the expulsion or destruction request of all remaining Group members. It can be requested by the Group manager or any LSM (potentially acting upon the request of some Group member in their HE). As with all Group management actions, enforcement of Group dissolution can only be initiated and coordinated by the Group Manager; it is enacted by the corresponding LSMs that have ultimate control over the SEA of each Group member.

8.18.5.4 Summary and current status of implementation

GRASP Security Infrastructure enables the dynamic formation and self-management of security perimeters protecting communities of users, services and resources. The interaction model of the proposed architecture integrates a layered peer-to-peer model (between the managers administering network entities and between the administered entities), with a centralised community management model (between community members and their local security managers) and a master/slave model (between security managers and enforcement agents). It supports on-demand creation and management of dynamic virtual collaborations in the form of secure communities of peers (user agents, services, resources, etc.) that are independent of network topology and span across multiple administrative domains, execution environments and enterprise boundaries.

Additional flexibility and protection of the SIG perimeters could be achieved by introducing elements of trust, as means of assessing confidence in a network entity on the basis of evidence accumulated by observing behaviour in different SIGs and by integrating and co-using the process execution monitoring and enforcement mechanism with other subsystems or specialised applications focusing on SLA administration and accounting. Research in this direction has been initiated in [375] and brought further in [376].

Implementation of the GRASP Security Infrastructure is currently ongoing within the European project GRASP[377,378,379]. The required security management functionalities are

[374] Possible reasons for expulsion include: underperformance, unavailability, hostile behaviour, completion of required tasks before the end of the Service instances lifetime.

[375] Dimitrakos T., Djordjevic I., Matthews B., Bicarregui J., Phillips C., Policy-Driven Access Control over a Distributed Firewall Architecture Policy 2002: IEEE 3rd International Workshop on Policies for Distributed Systems and Networks IEEE-CS, Monterey, California, U.S.A.,June 2002

[376] Dimitrakos T., Djordjevic I., "Towards Dynamic Security Perimeters for Virtual Collaborative Networks" in Proceedings of iTrust2004: the 2nd International Conference on Trust Management. Springer-Verlag LNCS 2004

[377] Dimitrakos T., D. Mac Randal, F. Yuan, M. Gaeta, G. Laria, P. Ritrovato, B. Serhan, S. Wesner, K. Wulf, An Emerging Architecture Enabling Grid-based Application Service Provision. Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference, IEEE Press 2003.

[378] Dimitrakos T., D. Mac Randal, G. Laria, N. Romano, P. Ritrovato, B. Serhan, S. Wesner, Trust, Security and Contract Management Challenges for Grid-based Application Service Provision. 2nd International Conference on Trust Management. Springer-Verlag LNCS, March-April 2004

realised as grid services; community lifetime management and membership management are facilitated by the presence of standard web/grid services interfaces. The design of the required certificates and security tokens takes advantages of WSE extensibility which allows the use of custom security tokens as security headers in the SOAP messages. For each certificate class there is an explicit schematic definition of its format, the corresponding authentication and validation rules. WSE SOAP elaboration pipeline is then reconfigured with the information required for a token validation. Consequently, in this implementation, the enforcement agent task is based on a rule verification process applied to the SOAP message. WSE SOAP elaboration pipeline parses the incoming and the outgoing SOAP message and it is formed by a set of filter. Each filter analyses specific section of the SOAP message. Community-specific security policies of relevance to a service or resource are interpreted in custom filters that implement the enforcement agent tasks (e.g. accepting or denying an access request encapsulated in a SOAP RPC).

8.18.6 Cardea (NASA IPG)

In this section we provide an overview of NASA IPG Cardea system based on [380].

Cardea is a distributed authorization system, developed as part of the NASA Information Power Grid[381,382,383], which dynamically evaluates authorization requests according to a set of relevant characteristics of the resource and requester rather than considering specific local identities. Potentially accessed resources within an administrative domain are protected by local access control policies, specified with the XACML syntax, in terms of requester and resource characteristics. Further, potential users are identified by X.509 proxy certificates[384,385] but only modeled according to the characteristics they can reliably demonstrate. The exact information needed to complete an authorization decision is assessed and collected during the decision process itself. This information is assembled appropriately and presented to the PDP that returns the final authorization decision for the actual access request together with any relevant details. See Figure 97 for a system architecture illustration.

Cardea is currently implemented in the Java[386] language as a set of independent components. Conceptually, the system contains a SAML Policy Decision Point (SAML PDP), one or more Attribute Authorities (AA), one or more Policy Enforcement Points (PEP), one or more references to an Information Service (IAS), an XACML context handler, one or more XACML Policy Administration Points (PAP) and an XACML Policy Decision Point (XACML PDP). Although all these components may be co-located on the same machine to use local communication paradigms, they may also be distributed across several machines and their functionality exposed as web service portTypes[387].

Communication between components is specified directly by the XACML and SAML standards, such as the request and response formats for obtaining information. Although

---

[379] Wesner S., Sehran B., Dimitrakos T, Mac Randal D., Laria G., Ritrovato, P., Towards a platform enabling Grid based Application Service Provision. eChallenges Conference e-2004 (submitted)

[380] Cardea: Dynamic Access Control in Distributed Systems Rebekah Lepro. NASA Advanced Supercomputing (NAS) Division NASA Ames Research Center, Moffett Field, CA 94035 NASA Technical Report NAS-03-020. November 2003

[381] Foster, I., Kesselman, C., Tsudik, G., and S. Tuecke: *A Security Architecture for Computational Grids.* ACM Conference Proceedings, Computers and Security, ACM Press, p.83-91, 1998

[382] Foster, I and C. Kesselman: *Globus: A Toolkit Based Grid Architecture.* The Grid, Blueprint for a Future Computing Infrastructure, Morgan Kaufmann, San Fracisco, p. 259-278, 1999

[383] The Information Power Grid - http://www.ipg.nasa.gov, visited 2003-07-14

[384] R. Housley et al: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC, January 1999

[385] S. Tuecke et al: Internet X.509 Public Key Infrastructure Proxy Certificate Profile. IETF draft, 2001

[386] The Java Language, http://java.sun.com, visited 2003-08-01

[387] E. Christensen et al: *Web Service Description Language,* W3C Note, Mar. 15, 2001

XACML and SAML are transport independent, the initial implementation binds these protocols to the Simple Object Access Protocol (SOAP) v. 1.1[388]. Support for SOAP-based communication comes from the Java reference implementations of the API for XML messaging[389] and utilizes the Apache Axis[390] architecture as an engine to transmit SOAP messages atop the http and https communication protocols. The Axis engine extracts the raw SAML or XACML construct from a message payload and forwards to the appropriate endpoint, as configured. From this point, message content is treated as native SAML and XACML and is thus shielded from its method of delivery.

To preserve message integrity, the body of each SOAP message is signed using XMLSig before transmission to the intended recipient. Custom handlers specified for the request and response flows within Axis provide common mechanisms to sign and verify this content of independently of content generation logic. As each message is signed only after processing is complete, the native format of the signed content is opaque to the signing process. Therefore, no dependencies between signature and content must be supported.

8.18.6.1 Cardea Architecture

The following figure provides an overview of the Cardea architecture. The subsequent paragraphs elaborate its main functionalities.

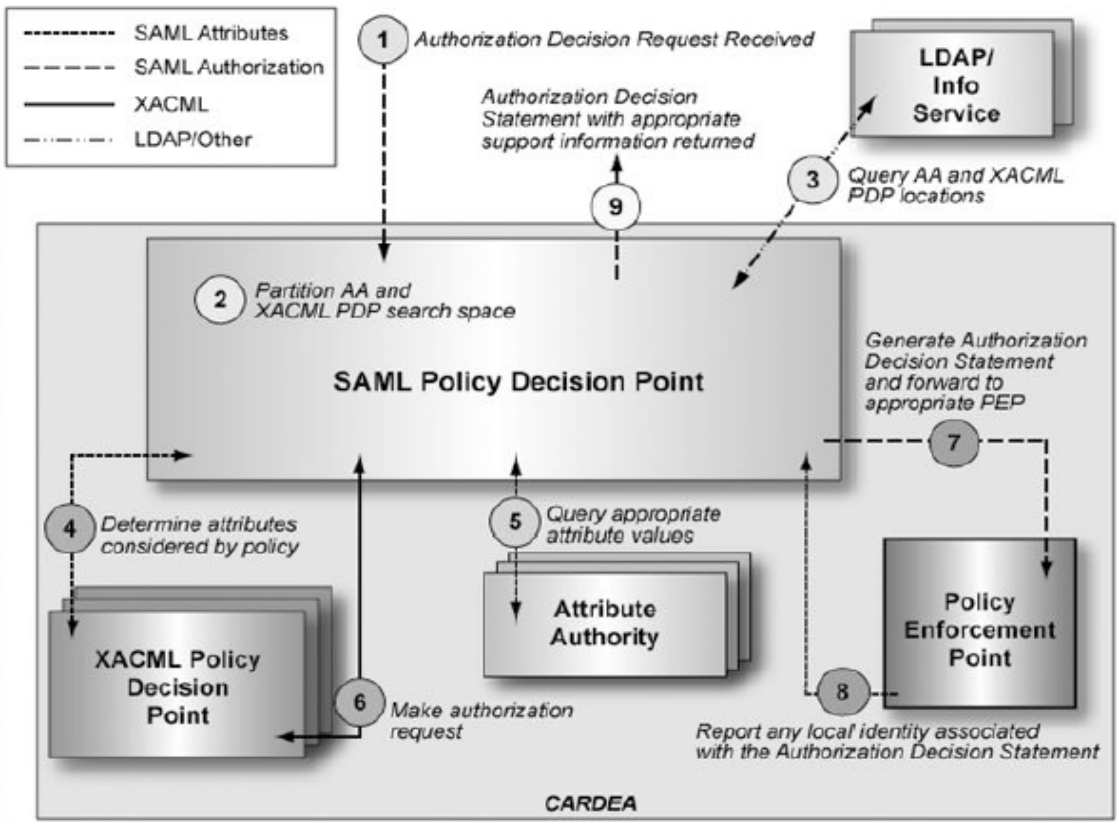


Figure 97 Cardea Architecture Overview

[388] Don Box et al: *Simple Object Access Protocol (SOAP) 1.1*. World Wide Web Consortium Note, May 2000

[389] The Java API for XML Messaging, http://java.sun.com/xml/jaxm/, visited 2003-07-31

[390] http://xml.apache.org/axis, visited 2003-07-14

8.18.6.2 System Prerequisites and required configuration

Although the system minimizes the amount of negotiation and configuration that must occur before the authorization framework may be implemented, there are several local items that must be defined according to the standard semantics of XACML and SAML. First, local access control policies must be defined in general terms according to the combined characteristics of salient user-resource combination before they may be enforced by PEP. Additionally, authorities must be populated with appropriate attribute values. Although there is no inherent restriction on how attributes may be maintained or represented internally to its attribute authority, each attribute value must be available to a qualified requester as a SAML Assertion.

8.18.6.3 System input

Initial system input must contain the requestor identity, the requested resource and the requested actions. Cardea presumes a SAML AuthorizationDecisionQuery structure packages this information together. Additionally, each AuthorizationDecisionStatement contains a unique request identifier as per the SAML specification. All system components use only the data contained in this SAML structure or information obtainable using only this data contained within this structure. In the initial system prototype, custom logic generated this initial system input.

However, a connector for existing grid toolkits a web service handler or any other custom logic can generate this original SAML AuthorizationDecisionQuery. Within the SAML AuthorizationDecisionQuery, the SubjectConfirmation element contains information on the issuer of the user credential. The SAMLSubject contains the distinguished name from the credential presented with the request. An attribute of the AuthorizationDecisionQuery element holds the Uniform Resource Name (URN) that identifies the requested resource.

As a particular requestor may wish to use permissions assigned with one or more groups on a given resource, these named groups in which the requestor wishes to participate must be identified within the request. Participation in each named group is then treated as an action to be authorized or denied and is thus represented as a specific SAMLAction. Further, to support charge accounting, the name of the charge group must be provided to the system. Charging to a particular account is also treated as an action to be authorized or denied. Finally, each request may be an initial access request, a follow-up request to query the status of an ongoing process or to terminate a process. Therefore, the specifically requested action requested must be provided as input to the system. Groups, projects and actions are all represented as Uniform Resource Names with a defined namespace.

8.18.6.4 System output

The final system output is a SAML AuthorizationDecisionStatement containing the identity of the authorized requestor and the authorized actions including groups, charge account, and authorized resource identity. The identity of the authorized user is represented as a SAMLSubject with the data extracted from the initial AuthorizationDecisionQuery. Each SAML AuthorizationDecisionStatement contains a unique response identifier and a reference to the request identifier for which this response is generated. If any errors were encountered during processing, only a SAMLError structure is returned. Otherwise, the response contains SAMLAssertion structures containing the attribute representations of all authorized actions, including resource access, group participation and charge accounting. The specifically authorized group, charge and resource data are represented as URNs in the same format as they are initially presented in the AuthorizationDecisionQuery. Finally, the XACML PDP decision is also represented within the SAML AuthorizationDecisionStatement.

8.18.6.5 SAML architecture overlay XACML architecture

The main component of Cardea is a SAML PDP that accepts an AuthorizationDecisionQuery and responds with AuthorizationDecisionStatement. The SAML standard places no constraints on how a SAML PDP generates query responses. Therefore, Cardea uses XACML functionality to support this function. However, an XACML PDP uses only the information contained directly within a request during evaluation. Within the XACML model, a ContextHandler manages this collection process. Thus, the SAML PDP must act as the ContextHandler to collect and format the needed information into an XACML request to forward to the XACML PDP. Thus, the SAML PDP also generates any necessary SAMLAttributeQueries and collects the responding SAMLAttributeStatements. Once all attribute statements are received, the SAML PDP transforms the SAMLAttributeStatement data into native XACML format and forwards an appropriate XACML request. The XACML conceptual model assumes that the correct PEP presents the request to the XACML PDP, via the ContextHandler. Therefore, an XACML response contains only the authorization decision and no details from the initial request within response. To ensure that all the information required by the actual PEP is available, Cardea's SAML PDP emulates the ContextHandler to the XACML PDP and preserves all details of the initial request. Then, the XACML PDP decision is incorporated with the preserved context information into the final SAMLAuthorizationDecisionStatement forwarded to the appropriate PEP.

8.18.6.6 Decoupling decision and enforcement

A key feature of the Cardea architecture is the division of the conceptual functions of the PDP, AA and PEP into separate components. This allows each site to determine the level of functionality they require. For example, a resource domain need not carry the weight associated with providing user attribute authority functions. By communicating only via the SAML standardized interfaces and according to the selected trust model, the PEP is free to enforce policy decisions from any trusted PDP using its own internal mechanisms. Although each PEP must establish mechanisms to trust an approved set of PDPs, a PEP is not tightly coupled to a specific PDP.

8.18.6.7 Reaching an authorization decision

Each administrative decision within Cardea is processed according to a general algorithm that requires minimal a priori knowledge of participants. The next paragraphs illustrate several critical steps within that authorization process, as handled by Cardea. It specifically highlights communication between distinct system components and how the input and outputs for each component are related.

Authorization Decision Request Received: In this step, an initial authorization decision request, formatted according to SAML request protocol specifications is accepted by the system. It is important to note that there are no restrictions on the origin of any accepted requested other than is required to enforce local access control policy. For example, an authorization domain may require that any request it processes be authenticated by a trusted source. Any request presenting from an untrusted source would be discarded, regardless of the fact that it could actually be completely processed by the system. All requests in Cardea are processed if they are digitally signed by an identity guaranteed by a trusted authority.

Partition search space for locating attribute authorities: When an access control decision is required, the authenticated identities of the parties involved are first resolved to their specific authorities that maintain the relevant attribute information. All requesters present a credential to Cardea when requesting an authorization decision. The system works with existing authentication mechanisms to verify the provided credential. An authority issues each credential. Therefore, the identity of the authority that issued the credential is used to build a query to an information server.

Query an information service to locate the authoritative AA and PDP locations: After determining the appropriate search parameters by examining the authorization decision request received, Cardea generates queries to an information service to determine

necessary location and binding data. These queries use the Java Naming and Directory Interface (JNDI) [JNDI] API to interact with an LDAP server that maintains the appropriate location and binding information for each trusted credential issuer. Cardea places no requirements on the security of interaction with the LDAP server. Each implementation must directly define and support the appropriate means to identity and interaction with trusted information stores. Currently, Cardea assumes location data will be in URL format and needs no authority-specific binding data. As the system matures, sufficient data to locate a portType specification for interaction with an individual authority should be determinable via the information service query.

Determine attributes considered by controlling policy: Location information for an attribute authority is used to construct a SOAP endpoint that represents an interface to that authority. To minimize the set of attribute assertions presented to the PDP for evaluation, a custom interface was built into the PDP to report the attribute identifiers expected within each request. This interface assumes that the identification of attributes within the XACML policy corresponds with their identification within the attribute servers queried with the SAML protocol. The initial functionality maps resource identifiers to the set of subject attributes required by the policy governing that resource. There is no precedent for this functionality directly within XACML. However, if reporting these attribute identifiers significantly reduces the number of potential attributes that must be collected, it results in a significant efficiency boost over blindly presenting all available attributes to the PDP within each request. Obviously, XACML cannot then specify a format for reporting the set of attributes required by a PDP. Therefore, this information is formatted as SAML attribute statements, permitting a standard interpretation of each result set.

Query appropriate attribute values: Once locations to obtained attribute information are identified, the relevant attribute values must be queried. Again, XACML specifies only the framework to present a complete set of attribute values to within an authorization request. The standard does not address how to collect the values contained within that set. Thus, SAML Attribute Queries are constructed for the original requester for each attribute required by the controlling PDP. Depending on the initial authorization request, this may require interaction with several distinct attribute authorities, particularly if credentials from several distinct authorities are relevant to the request. Regardless of the actual attribute authority contacted, the SAML protocol specifies the semantics of extracting the appropriate attribute values. Although not currently enforced, each attribute authority may also choose to accept or reject requests from an untrusted requester.

Make authorization request: All attribute information collected within the scope of a single authorization request is preserved to include in the XACML authorization request**.** Once the complete set of requester attributes has been queried, all returned values are formatted as XACML subject attributes. Resource and action attributes are handled in a similar fashion. Cardea employs custom functionality to achieve the necessary integration between XACML and SAML to transform collected attribute assertions to a format recognizable to the XACML PDP. This functionality presumes a correspondence between the attribute identities used in both the XACML and SAML representations of logically equivalent attributes. After populating the request, it is enclosed in a SOAP message destined for the PDP that controls the desired resource. The payload of the response received contains the evaluation decision made by that PDP.

Generate authorization decision statement to enforce and forward to appropriate PEP: XACML does not define mechanisms to provide detailed information about the access granted in a particular decision. Further, enforcing an authorization decision often requires knowledge of some attributes demonstrated by the requester. For example, if logical group memberships are represented as attributes, the PEP must know in which groups the requester is a valid member. A SAML attribute assertion contains the identity of a particular group where XACML authorization decision specifies membership validity. Therefore, the system bundles the actual authorization decision together with all the attribute values presented to the PDP within the authorization request when interacting with the appropriate PEP. Although not currently incorporated into the final SAML authorization decision

statement, evidence used to evaluate the request and conditions attached to the decision may also be presented to the PEP.

Report any local identity associate with the authorization decision statement: Once the PEP receives a SAML authorization decision statement, its first task is to verify the identity of the SAML PDP that generated the statement. Cardea uses the same generic signature verification handlers added to the Axis [AXIS] request chain as other subsystems. The PEP must define rules that govern how authorization decision statements will be enforced. Initial system design scope concentrated solely on generating the statements to be enforced by a PEP. Several alternative technologies may be used to enforce the design. The only constraint placed on enforcement functionality by Cardea design requires a PEP to report any local identity bound to the authorization decision statement be returned to the initial SAML PDP in the form of a SAML attribute assertion. This constraint facilitates further distribution of the authorization process between distinct yet cooperating PDPs.

8.18.6.8 XACML's role in the authorization process

Initial results demonstrate that XACML fills a critical role within the distributed authorization framework. Although it does not address all gaps identified in the selected authorization model, XACML's interoperability with other standard protocols provides the mechanism to bridge those gaps. Several of these subjects fall completely outside the scope of the XACML, such as management and retrieval of authorization attributes, or the location of applicable policy

decision points. Complimentary technologies, such as SAML or XMLDSig are required to provide this functionality. However, the system must still mediate between the interoperating protocols and develop mechanisms to locate appropriate authorities. Further, although functionality exists to collect attribute information and present it to the appropriate PDP, semantically definitions for common attribute names and legal values must still be negotiated. The remainder of this section examines several such issues that must be addressed and the way that they are addressed in the Cardea system.

Creation and management of access control policies: XACML provides a mechanism independent representation of access rules that vary in granularity via a standard yet flexible language. This flexibility permits the combination of multiple policies (e.g. from different authoritative parties) into a single applicable policy set to use when making access control decisions for resources in a widely distributed system with overlapping competencies. Further, this mechanism-independent representation of access rules allows a single policy to be applied to heterogeneous resources throughout and across administrative domains. This common representation greatly reduces errors, discrepancies, and auditing complexity. However, creation of actual XACML policies is not a simple task. Further, supporting XACML in heterogeneous environment calls for fully specified data type and function definitions that produce a highly verbose document even if the actual policy rules are trivial. Manual creation of such policies by ordinary users, as required in the PRIMA distributed authority model (see corresponding section in this deliverable) by resource administrators, as required in the Cardea system, is not reasonable. Therefore, additional management tools, such as the introduced PRIMA policy creator, to support policy file management and administration are required.

Locating the correct PDP: Before an authorization decision can be obtained, an authoritative PDP must be located. This bootstrapping problem is common to any distributed system and not specific to authorization systems based on XACML. Thus, XACML does not provide a standard mechanism to resolve this issue but relies on individual implementations to handle it appropriately to their environment. Initial system implementations either assume that PDP locations are fixed and policy file discovery depends on the requested resource or that each PDP may be located via an information service query to a trusted source. For example, Cardea assumes that a directory service contains the necessary location and binding data for the appropriate PDP. Once a PDP is identified, XACML functionality provides for the location of applicable policy files, including policies to be retrieved from a remote location.

XACML request preparation and request context management: XACML considers the collection and encoding of attributes used in an authorization system to lie outside its core focus. Further, XACML views attributes as an external form of access control information that must be converted from their native form to be included in an XACML authorization decision request in the form of a request context by a context manager component. XACML does not standardize interactions to retrieve this data for an authorization request. Two

distinct approaches have been implemented within the introduced systems to share subject data used for authorization. The first provides a framework by which this information is shared via SAML. The second uses privilege attributes managed by subjects to directly influence the context creation.

The XACML model is based on the authorization pull sequence [RFC2904] and requires the context manager to maintain state information to associate requests that it created with received responses. If another authorization sequence such as the push or the agent sequence [RFC2904] are desired, the contextual information necessary for a PEP to enforce an access decision response from a PDP has to be supplied to the PEP through a supplementary mechanism.

Current work on SAML 2.0 proposes to include the original authorization decision request context with an authorization decision response, which would address this issue.

Encoding of descriptive attributes: Cardea employs SAMLAttributeAssertions to collect and encode attribute data for an authorization decision request. Custom functionality transforms the collected SAMLAttributeAssertions into a valid XACML attribute format. Although specific mappings need not be predefined, the functionality presumes a correspondence between the attribute identities used in the XACML and SAML representations of each logically equivalent attribute.

By supporting such transformations, these attributes are available both within the decision and enforcement phases of authorization. Therefore, Cardea augments XACML functionality with SAML functionality to provide this data to all participants in an authorization decision.

8.18.6.9 SAML's role in the authorization process

Like XACML, SAML can also play a central role in the authorization process. SAML clearly defines how to represent attribute information used for access control decisions, a protocol to share attribute information between entities when authorization is distributed, and a protocol to share assertions about authorization decisions. However, SAML does not currently address related issues that impact the architecture of the authorization process. The remainder of this section examines several of these issues and how they fit within the SAML framework.

Representing attribute information. SAML provides a framework for specifying the format of and parameter naming within an authorization decision request and response. This offers a common basis for heterogeneous entities to make and interpret decision without knowledge of the underlying systems that implement the decision process. SAML purposely places no restrictions on how a PDP interprets and responds to a request that it receives. Implementers must agree upon expected attribute names, data types and acceptable subject representations for exchanged assertion requests and responses. However, the SAML standard defines constraints on legal representations and how to package this information into assertions. Therefore, using SAML to support communication significantly reduces the amount of agreement required.

Requesting an authorization decision. SAML provides a framework that answers many of the design questions that any system architect needs to answer when building an interoperable authorization system [COHEN]. Specifically, it provides a blueprint for the communication necessary to transform an assertion request into an assertion statement. However, it places no requirements on the methods and functionality by which a particular authority implements these transformations. With Cardea, this process requires contact with suitable authorities, collection of support information and evaluation of relevant access

control policies. Although this bootstrap problem of locating the proper authorities is not peculiar to SAML based systems, it must still be addressed. Cardea uses well-known grid information services to locate the appropriate authorities whereas alternative solutions may rely on UDDI, LDAP, RDBMS or OGSA discover technologies. Fortunately, SAML provides a way to model data items needed during the authorization process: as attributes that may be queried from an attribute authority and specified in the form of assertions.

Therefore, each PDP needs to develop only the logic for location mechanisms and the SAML functionality to collect the information it requires. Further, XACML functionality specifies a standard mechanism to format access control policies and evaluate requests according to a specified policy. Therefore, overlaying SAML functionality on XACML functionality provides a clear roadmap to deterministically transforming an initial SAML AuthorizationDecisionQuery into a SAML AuthorizationDecisionStatement.

8.18.7 Comparison of evaluated Grid Security Systems

The Community Authorization Service (CAS) is enhancement of the Globus GSI (Grid Security Infrastructure). It separates the administration of resource specific issues from those that are community specific. Resource administrators can grant course-grained access control policies to a community as a whole (e.g. a Virtual Organisation) and community administrators then decide what subset of a community's rights an individual member will have. Group members authenticate to a resources with a group credential that has restrictions applied to it (limited proxy credential) narrowing the individual's rights to a subset of the rights the community has at the resource. This is accomplished by defining extensions to X.509 Certificates to carry restricted policies. Although administration is partitioned between community and resource administrators, knowledge of resources, user identities, community access rights and group memberships must be established before authorisation may occur.

Similarly to the CAS, the Virtual Organization Membership Service (VOMS) also performs authorisations according to community membership privileges. A (community-centric) VOMS server encodes this information into non-critical certificate extensions that may be presented to the resource. Thus, the relevant authorisation information is packaged directly with the identity information used for authentication and pushed to the requesting resource. In VOMS however, the subjects authenticate with their own credentials (in contrast to a limited group credential in CAS) and subject attributes allow for the use of community privileges. The system assumes that group and role information is reported in a format natively understandable to the authorised resource. Therefore, although the system distributes management responsibilities between resource and VO administration, it still assumes this information is known before the authorisation decision may occur.

The PRIMA system focuses on the management and the enforcement of fine-grained access rights. It employs standard attribute certificates to bind rights to users and enables the high level management of such fine grained privileges which may be freely delegated, traded, and combined. It is similar to VOMS in sense that the subjects authenticate with their own credentials (in contrast to a limited group credential in CAS) and subject attributes allow for the use of community privileges, with the distinction that in PRIMA the attributes are not issued by a community server but rather come directly from the individual attribute authorities. Enforcement in PRIMA is provided by POSIX operating systems extensions (available for common platforms) that extend standard file permissions and regulate resource usage through access control lists.

Akenti authorisation system supports multiple stakeholders to impose use conditions on a particular access control request. It provides a way to express and to enforce an access control policy without requiring a central enforcer and administrative authority. Akenti uses standard X.509 public key certificates for authentication, and proprietary XML certificate structures for the binding of attributes to users (e.g. roles, group membership) and to resources (e.g. use condition statements). Attribute and use case condition certificates are collected during the authorisation decision process (in a pull fashion) and evaluated according to the Akenti policy language, requiring use of specialised data formats and APIs.

GRASP Security Infrastructure enables the dynamic formation and self-management of security perimeters protecting communities of users, services and resources, across the administrative domains. Security policy of communities is managed in a federated fashion (between security administrators and community managers), and enforced by each community member through master/slave model (between security managers and enforcement agents). Direct P2P interactions between the members are supported with authorisation tokens which are issued by the community administrators and endorsed by the security administrators of corresponding members. Current GRASP implementation provides limited functionality for security policy management and enforcement. It builds on top of the OGSA realised on Microsoft .NET. The design of the required certificates and security tokens takes advantages of WSE extensibility which allows the use of custom security tokens as security headers in the SOAP messages. The enforcement agent task is based on a rule verification process applied to the SOAP message, where WSE SOAP elaboration pipeline parses the incoming and the outgoing SOAP message. Community-specific security policies

of relevance to a service or resource are interpreted in custom filters that implement the enforcement agent tasks.

Focus of Cardea system is support for authorisation scenarios that span multiple administrative domains. Authorisations are independent of local identities at the resources. Authorisation requests are evaluated by rendering attributes of the resource and of the requester against applicable resource policies specified in XACML. Required attribute values are assessed, collected and presented to a PDP by the PEP with the request. Cardea exposes web service portTypes as external interfaces and is implemented in Java. For communication between system components, SAML and XACML are used for message formats and SOAP is used as a transport mechanism. Messages are secured using the XML Digital Signature specification.

8.18.7.1 Application to TrustCoM

As already mentioned in Section 3, the current OGSA Grid security requirements are aligned with the objectives of the TrustCoM project. One would expect that as Grid Security tools mature they will seek to meet these requirements and therefore will be of use for TrustCoM. All approaches reviewed in this section (Grid Security) are continuously improving prototypes. VOMS and CAS are both based on a community centric paradigm but suffer from the absence of an expressive policy description language (e.g. one that exploits the concept of role) and in their current versions they tightly couple authentication information with access rights. PRIMA appears to be making a better use of the concept of attribute authorities, but still lacks an expressive policy language and a flexible distributed access control enforcement mechanism. Akenti achieved decentralised access control enforcement and supports abstractions such as groups and roles. However it is dependent on a proprietary access control language. Cardea does not directly address community membership management but offers support for authorisation across multiple administrative domains through an interesting combination of SAML and XACML-based solutions.

GRASP-SI came later than VOMS and CAS and has the capability to combine loosely-coupled community membership management mechanisms with role based policies for controlling access. GRASP-SI separates authentication tokens from authorisation tokens (which can be explicitly associated to the former) and its conceptual model relativises roles within communities following in that matter Enterprise Viewpoint of ODP concepts. Its current implementation (to be completed in January 2005) on .NET takes advantage of SOAP messaging and the WS-Security and WS-Secure-Conversation protocols. One further advantage of GRASP-SI is that some of the partners involved in its design and implementation are also involved in TrustCoM. Similarly to the other tools evaluated, a disadvantage of GRASP-SI is the abasence of a powerful policy description language. (However, a new UK advanced development project, funded by JISC for two years starting from September 2004, will attempt to integrate GRASP-SI with PERMIS.) In relation to TrustCoM and based on the evaluation of tools in this section we suggest that combination of the GRASP-SI conceptual models for community management and distributed enforcement with NASA IPG Cardea for access control management could come some way addressing many of the project objectives in the VO security area. This may need to be further supported by flexible policy lanaguages such as those described in earlier sections of this chapter.Other Policy Based Approaches

There has been considerable interest in logic-based approaches to specifying authorisation policy as exemplified by [391] and [392]. They assume a strong mathematical background, which can make them difficult to use and understand, and they do not easily map onto implementation mechanisms. The ASL language[391] includes a form of meta-policies called integrity rules to specify application-dependent rules that limit the range of acceptable access control policies. Although it provides support for role-based access control, the language does not scale well to large systems because there is no way of grouping rules into structures for reusability. A separate rule must be specified for each action. There is no explicit specification of delegation and no means of specifying authorisation rules for groups of objects that are not related by type.

---

[391] S. Jajodia, P. Samarati, and V.S. Subrahmanian, “A Logical Language for Expressing Authorisations”, IEEE Symposium on Security and Privacy, Oakland, USA, 1997, pp.31-42

[392] Ortalo, R. “A Flexible Method for Information System Security Policy Specification”, Proc. 5th European Symposium on Research in Computer Security (ESORICS 98). 1998, Louvain-la-Neuve, Belgium, Springer-Verlag.

Ortalo[392] describes a language to express security policies in information systems based on the logic of permissions and obligations, a type of modal logic called *deontic logic*. Standard deontic logic centres on impersonal statements of the form “it is obliged that p” which do not necessarily identify to whom the obligation applies. Ortalo accepts the axiom Pp = Op (“permitted p is equivalent to not p being not obliged”) as a suitable definition of permission. In our view, an obligation policy requires a relevant authorisation policy to permit the actions defined in the obligation but an obligation policy does not imply an authorisation policy.

Others, focus on the specification and implementation of access control policies for mobile agent systems. In most cases the studies focus on reconfigurable access control policies in the Java environment. They have included both the translation of higher-level policies into Java security policies as well as different access control mechanisms within Java[393, 394].

8.19 Information Flow

While role-based access control policies, and access control policies in general, define security policies that are easy to enforce, they are limited in what one can express. What one really wants to protect is not access to information, but the flow of information. For example, with access control policies one can protect the reading of secret information from a file. However, one cannot prohibit that someone (a system user or a program) with read access to this file makes this information publicly available by writing the information from this file into a publicly available file. Therefore, one has to trust the user, and all programs acting on behalf of the user, that they adhere to this information flow policy, i.e., that secret information may not flow to public files.

There are different models of information flow control policies now in the literature, the classic one being Dennings lattice model[395], where the flow of information is controlled by security classes which are arranged in a lattice: information is allowed to flow from one security class to another, if and only if the second dominates the first. This means, information may flow from low security classes to high security classes, but not from high ones to low ones.

Precisely defining what is meant by the intuitive notion of "flow of information" is quite delicate, and research in this area lead to a wide range of so called "non-interference" properties. These properties in general express that the observable behaviour (e.g., the contents of public files) of a system may not depend on secret actions (e.g., the reading of confidential information). Such properties are usually expressed in terms of closure properties on the set of possible traces of the system's behaviour. For example, if a certain trace is possible, then the trace where no secret actions happened must also be possible.

Mechanisms to enforce information flow control policies can be classified into run-time and compilation-time mechanisms. Run-time mechanisms have the advantage that they can also control the actions of human system users, but they are only applicable to a restricted class of information flow control policies, because in general one needs to keep history information about previous accesses to decide if some action is allowed, and in some cases one would even need information about possible future actions, if one does not want the mechanism to be too restrictive.

Compile-time mechanisms apply a static analysis of the program's source code, in order to validate that the programs complies to the information flow policy. Most techniques here are language-based[396], meaning that the type system of the programming language is extended with security labels and additional type inference rules. The type inference rules then guarantee that a type-correct program will adhere to the information flow policy.

---

[393] Corradi, A., R. Montanari, C. Stefanelli, E. Lupu and M. Sloman. “Flexible Access Control for Java Mobile Code”, 16th Annual Computer Security Applications Conference (ACSAC2000), Dec 2000, New Orleans USA.

[394] Hashii,B., S. Malabarba, R. Pandey M. Bishop. “Supporting reconfigurable security policies for mobile programs” Computer Networks, Elsevier Publishing, vol. 33, June 2000, pp. 77-93.

[395] D. Denning *Lattice Model of Secure Information Flow*, Communications of the ACM, vol. 19, no. 5, May 1976, pp. 236-243.

[396] Andrei Sabelfeld, Andrew C. Myers. *Language-Based Information-Flow Security*, IEEE Journal on Selected Areas in Communications, vol. 21, no. 1, January 2003.

8.20 Author-X (Policy-based access control for XML documents)

Author-X[397] is a Java-based system, developed at the University of Milan's Department of information science, to address the security issues of access control and policy design for XML documents.

Author-*X* supports the specification of policies at varying granularity levels and the specification of user credentials as a way to enforce access control. Access control is available according to both push and pull document distribution policies, and document updates are distributed through a combination of hash functions and digital signature techniques.

The **Author-X** approach to distributed updates allows a user to verify a document's integrity without contacting the document server.

Other important features of the access control model supported by **Author-*X*** are*:*

- Its temporal dimension, in that it is possible to express policies that hold only for a specific period of time (such as for instance a particular day of a week);
- A set o *propagation options* that allow to reduce the number of policies that need to be specified.

The main advantage of Author-X is that it enables the specification of access control policies for the dissemination of XML documents at different levels of granularity for both push and pull document distribution policies. The access control model also supports temporal constraints.

However, the current implementation of Author-X is a prototype, which has not been used in any real world application so far.

Within the context of TrustCoM, collaboration in a VO may require the dissemination of XML documents containing information at different sensitivity levels and suitable control policies to access them. Autor-X provides a comprehensive approach for data protection that includes mechanisms for enforcing access control policies based on data contents, on subject qualifications and characteristics.

8.21 Adaptive and Agile Security

Virtual Organisations are highly dynamic as both members of the VO and infrastructure services may change dynamically throughout the lifetime of the VO. Furthermore, the security of the VO depends on the security enforced in each of its constituent members and the VO must change dynamically in order to react to security relevant events. Thus, there is a need to adapt the security infrastructure in response to events and to provide automatic methods for reliably establishing trust, detecting intrusions, adapting security policies to new situations and recovering from critical security conditions.

Adaptive Security, also known as *Agile Security* aims to provide these needs by establishing intelligent techniques to adapt the security needs of the dynamic environment. The main purpose of the Adaptive Security is to identify the problems and automatically correct them. Traditional access control mechanisms are often based on static access control configurations and do not provide adaptive security. In Adaptive Security, security does not rely solely on a set of static system configurations defined by a human administrator, but an ongoing adaptive process in which policy based techniques are used to provide automated configurations to dynamically handle security events. The security management follows the same feedback loop encountered in network and systems management, which includes monitoring, diagnostic/analysis and applying corrective actions (response)[398]. Monitoring involves instrument and detecting change in the environment where the VO is deployed or in the VO itself. The analysis phase includes both diagnostic and identification of the

[397] Securing XML Documents with Author-X, IEEE Internet Computing, vol. 5., no. 3, May/June 2001.

[398] Workshop on Logical Foundations of an Adaptive Security Infrastructure: http://www.aero.org/wolfasi/

changes which have led to the events as well as deciding upon the corrective actions to be performed according to pre-defined policies. Response phase includes enforcing the configuration changes dictated by policy or decided during the analysis phase actions directed by the analyser.

Although advocated by many, there is relatively little work in the area of *adaptive* or *agile security*. The following sections present a few of the frameworks proposed. Most of them focus on the integration of existing security mechanisms or products e.g., intrusion detection in order to provide adaptive behaviour. A more reliable and flexible model would involve mechanisms for establishing and reasoning about trust, and dynamic negotiation of policies and security parameters.

8.21.1 Tivoli Risk Manager

In large organisation, it is hard to predict the normal traffic of the network, and often intrusion detection tools can generate thousands of events a day with many often being false alarms. Often administrators do not know if the alarm is a malicious event. Firewall, intrusion detection tools, access control and web services all have different security functions, and often do not interoperate with each other. These independent vendor products have no interaction between them, and each have separate consoles are managed and administrated individually. A more effective solution would integrate these solutions to minimise security threats.

Tivoli Risk Manager[399] developed by IBM aims to manage security threats, malicious users and other vulnerabilities across an enterprise security checkpoints by correlating security information and alerts from a set of devices including intrusion detection systems, firewalls, routers and networks. Tivoli Risk Manager comes with an intelligent engine, which correlates all the information from multiple sensors and presents a single alarm for each attack.

Tivoli Risk Manager tries to achieve the following main objectives[400]:

- Provides a single centralised control point (a single web-based security console) to monitor and manage security alerts across the enterprise.
- Integrates products such as security applications (firewalls, anti-virus tools), networks and operating systems to provide a comprehensive security management environment.
- Helps system administrators to identify types of threats, pattern of intrusions and attacks accurately using advanced correlation techniques, aggregation and summarisation to speed the respond time and to reduce the false alerts.
- Provides a variety of pre-defined respond tasks (automatic tasks) to resolve urgent or severe security attacks such as denial of service attacks, policy violations or unauthorised access. These responds include disabling user accounts and reconfiguring firewall policies.
- Provides analytical historical reports to assess business risks and to support decision-making.

The Risk Manager architecture contains the following set of components[401]:

- ***Event Generating Components:*** A range of Risk Manager adopters and sensors collecting information about possible intrusions and passing this information as sensor events to the Risk Manager Correlation Engine.
- ***Real-Time Alerting Components:*** These components (including the Risk Manager Correlation Engine) are responsible for performing correlation and writing events to the event repository. The correlation is based on normalising the information, aggregation rules and situation analysis

[399] IBM Tivoli Risk Manager: http://www-306.ibm.com/software/tivoli/products/risk-mgr/

[400] IBM Tivoli Risk Manager, Problem Determination Guide, 2002: http://publib.boulder.ibm.com/tividd/td/RiskManager4.1.html

[401] Recommended Practices for Risk Management with Tivoli Risk Manager. RedBook, published by IBM, 2002: http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/redp0202.html?Open

to compute the severity and of the situation. If there is an attack, the Correlation Engine sends situation alarms to administrators at a centralised console. The components include servers, which can respond to or modify the events automatically.

- ***Historical Reporting Components:*** These components consist of data mining and analysis mechanisms for decision support, and other components for risk management.

8.21.1.1 Advantages/Disadvantages

Tivoli Risk Manager is one of the first examples of "automatic software management software". It provides a composite solution by collecting data from various security tools.

8.21.1.2 Application to TrustCoM

Tivoli Risk Manager, and other similar solutions like ISMS (see next section) are aimed to provide a local autonomic security management. However, in Virtual Organisations, security events may be seen at geographically disseminated sites, and require a distributed solution. For example, event detectors at each site may need to cooperate with other detectors in other sites to detect event patterns and to raise alarms. Tivoli Risk Manager could be extended to provide an automated security management system for virtual organisations.

8.21.2 Intelligent Security Infrastructure Management Systems (ISMS)

The Intelligent Security Infrastructure Management Systems (iSIMS) technology platform developed by Symbiot, has been recently released[402]. Similar to Tivoli Risk Manager, iSIMS interoperates with other security infrastructures such as firewalls, intrusion detection/presentation systems and virtual private networks to accumulate security events in real-time, and uses the security event data to build a risk model. The risk model provides information about the *significance* (how threatening is an attacker) and the *impact* (the cost of the attack if it succeeds). The system measures cost in dollar terms.

The measures of threats are defined using a risk score. The risk metric used is similar to a credit score provided by the credit reporting bureaus. Risk scores are used throughout the iSIMS platform and the Symbiot.NET repository to provide accountability, consistency and standardisation. The Symbiot.NET is the central repository containing attacker attribution profiles based on cooperative surveillance and reconnaissance collected by network participants. The knowledge base of iSIMS is updated using the data from the Symbiot.NET. This data is used to identify attackers, what they can do, evaluates their methods and intentions and recommend appropriate countermeasures.

The *Expected Value of Risk,* EV(R) is a function of the following terms integrated over time[403]:

$$\mathrm{EV(R)} = \int \left[\mathrm{P}_{\mathrm{THREAT}}(t) \cdot \mathrm{P}_{\mathrm{VULNERABILITY}}(t) \cdot \mathrm{Value}(t) \right] dt$$

The probability of threat $P_{TREAT}\ (t)$, is an estimator derived from the cumulative alerts generated by existing security components and can be self-generated by the ISIMS platform based on anomaly, behaviour and signature analysis of the live network traffic. The probability of vulnerability, $P_{VULNERABILITY}\ (t)$, is estimated based on an evaluation of how the network will respond to threats. Value is determined using both the asset cost and the Net Present Value (NPV) of the revenues associated with the exposed infrastructure.

Symbiot.NET provides a model of *graduated response* against intrusion events, from simple techniques such as blocking traffics to more aggressive operations. The iSIMS platform and Symbiot.NET attacker knowledge base use a range of rules to establish recommendations for countermeasures to be enacted including blocking traffic, rate-limiting (adjusting the bandwidth

[402] Symbiot Security Web Site: http://www.symbiot.com/isimstechnology.html

[403] Enterprise Specific Event Significance: http://www.symbiot.com/es2.html

available), diverting traffic and quarantine (redirecting into a special area for analysing the characteristics).

8.21.2.1 Advantages/Disadvantages

The iSIMS platform comes with a console to aggregate, correlate and visualise security event data in real time.

Compared with other Enterprise Network Management (ENM) such as Tivoli Risk Manager it seems that iSIMS provides an enhanced framework; an extensive data store, an analysis process after the correlation, a sophisticated console and feedback loops to regulate multiple security point solutions[404].

8.21.2.2 Application to TrustCoM

Like Tivoli Risk Manager, ISMS requires further work to provide an automated security management framework for virtual organisations.

8.21.3 Adaptive Security Policies

The work presented in paper[405] was one of the first to advocate the use of adaptive security policies in highly secure environments. It argues computer security policies must be adaptive to react to changes in the security environment. The paper compares various methods for implementing security policies by separating the definition of the policy in a security server from the enforcement, which is done by the microkernel. A prototype operating system, the Distributed Trusted Operating System (DTOS) is used to evaluate policies. The DTOS design consists of a microkernel and a collection of security servers. Security servers define the policies enforced by the microkernel. When a request for a service is made to the kernel, the kernel submits the request with various information such as security context of the subject and object, to the security server to determine if the access is permitted. The security policies are similar to firewall rules.

This paper presents and compares a number of approaches for changing the security policies to adapt to dynamic security environmentthat are further developed in[405]. However, these approaches require either reloading of the policy or changing the algorithm, which the security server uses to make its security computation. None of the proposed methods provide a concrete solution and they have weakpoints.

More recent work[406] argues that the policy implementation defined in the DTOS is not effective and scalable, and proposes an authorisation framework to specify and enforce security policies, which assist in detecting and responding to security attacks and misuse. In[406], security policies accommodate changes in the security requirements and assist in detecting and responding to intrusion and of abuse of user privileges. Authentication policies can require more information from a user when suspicious activity has been detected. For example, a policy can be specified to contain the followings[407]:

```
Alice can run a process on host doc.ic.ac.uk. If the request fails, a
notification must be sent to a system administrator. The process must not
consume more than 10% of the CPU. An audit record about the completed
process must be generated.
```

---

[404] P. X. Nathan. A Trajectory for the Evolution of SIMS Architecture, Revised December 2003: http://www.symbiot.com/media/SIMS_Evolution.pdf

[405] M. Carney and B. Loe, A Comparison of Methods for Implementing Adaptive Security Policies.

7th USENIX Security Symposium, 1998, San Antonio, Texas, USA.

[406] T. Ryutov and C. Neuman, The Specification and Enforcement of Advanced Security Policies. 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02), Monterey, California.

[407] T. Ryutov, The Specification and Enforcement of Advanced Security Policies. Presentation slides: www.policy-workshop.org/2002/Files/ Slides/AdvancedPolicies.pdf

Here **`Alice`**, **`10%`**, **`notification`** and **`audit`** are conditions, **`run`** is the access right and **`doc.ic.ac.uk`** is the target object. The policies are defined using a range of conditions that provide run-time adaptation in the event of possible attacks or misuse. Conditions include access identity, authentication mechanisms, time (periods for which access is permitted), and location of the user, payment, system threat level and notification. Failure of conditions may indicate distrustful behaviour. The conditions are classified as *pre-conditions* (conditions which must be satisfied before the execution), *request-result conditions* (conditions which must be activated when the authorisation is granted or denied), *mid-conditions* (conditions which must be true during the execution) and *post-conditions* (conditions which must be satisfied after execution). Some of these conditions may trigger simple responses such as limiting the resource computation.

To enforce the policies, they adopt a three-phase policy enforcement scheme; access control phase, execution control phase and post-execution actions phase. They make use of Generic Authorization and Access-control API (GAA-API)[406], which provides a general-purpose execution environment in which policies are evaluated. The GAA-API returns the status values for each phase to describe policy enforcement process.

8.21.3.1 Advantages/Disadvantages

Ponder (see section 8.3) can be used to support similar policies. In Ponder, obligation policies are used to specify what changes (management actions) need to be performed in response to events generated by the system. Ponder Authorisation and Obligation policies are expressed independently, whereas in this framework, both the authorisation and obligation requirements are expressed in a single policy structure. Authors argue combining these requirements lead to fewer conflicts between authorisation and obligation policies. Their three-phase policy enforcement model allows for parts of a policy to be enforced at various times. In Ponder, obligation policies are triggered by system events, whereas in this framework the actions are triggered by other conditions in the policy, such as system threat level.

One of the assumptions this framework makes is that conditions are evaluated consecutively and the authorisation requests do not overlap. So they do a single condition evaluation at a time, and avoid the problem of coordination of multiple condition evaluation processes. The authors argue these assumptions make the system not scalable.

8.21.3.2 Application to TrustCoM

Further work is required to overcome the weaknesses of the framework, however the underlying concepts can be used to design adaptive security policies.

8.21.4 Security Agility for Dynamic Execution Environments

Security agility[408] is a technique, which extends the functionality of software components to make them aware of their dynamic security environment and adapt to policy changes, and respond to intrusion detection. These components are able to enforce their part of the global policy, and contain internal mechanisms to automatically adapt when security policies change.Figure 98 shows an abstract view of the strategy. Each component has a rule set which provide components with built-in knowledge of security policies, models, and mechanisms to adapt to changes. These components also interact with the Agility Authority to receive policy updates. The Agility Authority transmits authorised security policy change requests to agile security components, which then dynamically behave according to the new security policy requirements. In response to a policy change, a component may take a number of actions, for example, terminating connections, tighten access to resources, changing cryptographic algorithms or accepting new security enforcement responsibilities.

[408] M. Petkac, L. Badger and W. Morrison, Security Agility for Dynamic Execution Environments. Proceedings of the DARPA Information Survivability Conference & Exposition Volume I of II, Hilton Head, South Carolina.

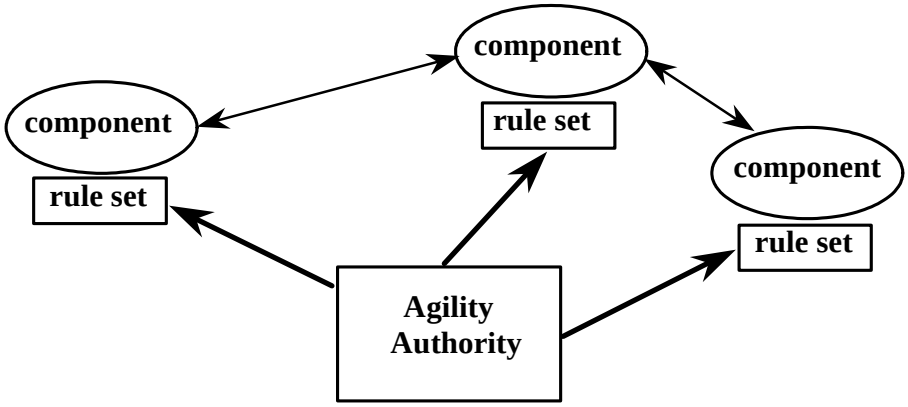


Figure 98 Security Agility Solution Strategy[409]

The Figure 99 shows the general architecture of a security *agile* component and indicates the interaction of the key elements of the security agility toolkit. The component-specific code implements the component's non-security responsibilities. The security-specific functionality is carried out by the agility subsystem. The component-specific code includes a number of Control Transition Points (CTPs), which are interfaces that conditionally transfer control to the security agility subsystem. The CTPs provide the agility subsystem with an outline of the component's behaviour. The subsystem uses this information to provide the appropriate security services (e.g., cryptography) on behalf of the component.

The security-agility subsystem is able to carry some reconfiguration internally (e.g., closing files) but it may require some help from the component-specific code. The security-agility subsystem will invoke callback functions to achieve such tasks. The agile system component receives notification of dynamic changes to the system security policy and information to modify the application-level security policy from the Agility Authority.

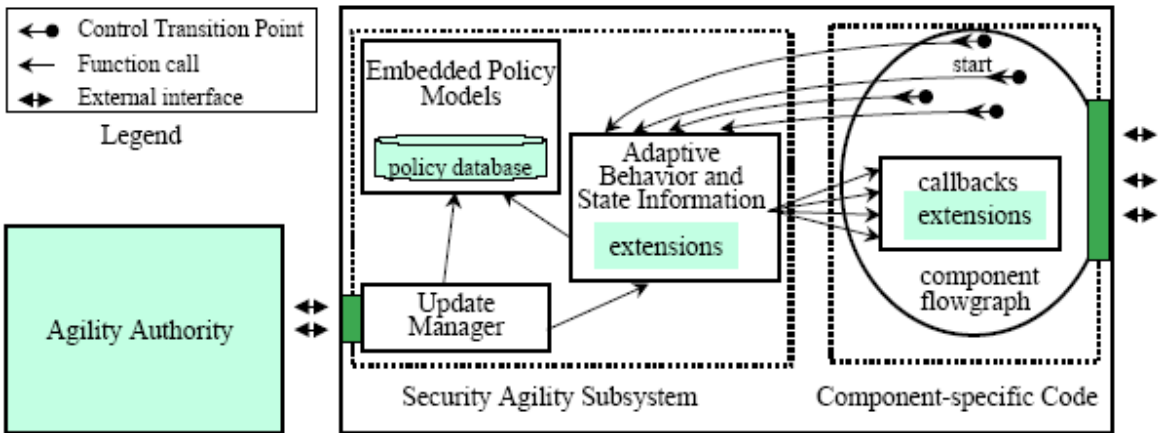


Figure 99 Security Agile Component Arthictecture[86]

8.21.4.1 Advantages/Disadvantages

A prototype toolkit is implemented and integrated with various UNIX-based system components to demonstrate the framework. The current toolkit aims to provide the basis for defining automated respond to intrusion detection events. The toolkit can be provided in a plug-in tool for software developers or added via wrappers without compiling the software.

[409] Project Profile, *Security Agility for Dynamic Execution Environments*, NAI Labs security Agility Research, 2001

8.21.4.2 Application to TrustCoM

Further work is required to evaluate the framework on other operating systems, and to increase robustness and provide simpler management. However, the underlying concept, making the components aware of their security environment is an interesting topic, and could be used within TrustCoM framework to support Adaptive Security.

8.22 Model Driven Security

Model building is standard practice both in system engineering and in security policy specification. However, the integration of system design models with security models is poorly understood and inadequately supported by modern software development processes and tools. Although security requirements and threats are often considered during the early development phases (requirements analysis), and security mechanisms are later employed in the final development phases (system integration and test), there is a gap in the middle. As a result, security is typically integrated into systems in a post-hoc manner, which degrades the security and maintainability of the resulting systems.

Several approaches to overcome these problem based on UML have appeared in the last years. For example, a UML profile called UMLsec was proposed[410,411] to annotate UML diagrams with security requirements.

In another approach, called Model Driven Security[412,413,414] the Model Driven Architecture (MDA) paradigm is applied to the security area. In MDA, models are not only used for specifying system properties, but also for generating system artifacts that implement these specified properties. The goal of Model Driven Security is now to provide methods and tools to tightly integrate security into an MDA-based development process. The key idea is to use high-level, visual models that integrate system design and security policies, and to use generative techniques to automate the construction of systems from these design.

Models that integrate system design and security policies demand modeling languages that are capable of expressing both these aspects. However, there are many different security requirements and also many different system design-modeling languages to consider. Because of this, Model Driven Security pursuits a generic or parametric approach to constructing such integrated languages. In this approach, domain-specific modeling languages for both system design and security policy specification are developed independently, the abstract syntax (the so called metamodel) of these languages being described with MOF, the Meta-Object Facility. Then, a selected pair of such languages can be combined by an additional MOF-model (a so-called dialect) that describes the "glue" between these languages. For example, one can use MOF-generalizations (i.e., inheritance) to define selected model elements of the system design modeling language to be protected resources in the sense of the security policy specification language.

This combination of languages has actually to be done also on the level of concrete syntax or notation (how should one draw models in this combined language?), on the level of semantics (what shall models in this combined language mean?), and on the level of transformation rules (how should the security policy specification be enforced in the system?).

---

[410] Jan Jürjens, *Towards Development of Secure Systems using UMLsec* in Fundamental Approaches to Software Engineering (FASE/ETAPS 2001), Springer LNCS 2029, pp. 187-200.

[411] Jan Jürjens, *UMLsec: Extending UML for Secure Systems Development* in UML 2002 - The Unified Modeling Language, Springer LNCS 2460, pp. 412-425.

[412] Torsten Lodderstedt, David Basin and Jürgen Doser, *SecureUML: A UML-Based Modeling Language for Model-Driven Security* in UML 2002. The Unified Modeling Language, Springer LNCS 2460, pp. 426-441.

[413] David Basin, Jürgen Doser and Torsten Lodderstedt, *Model Driven Security for Process-Oriented Systems* in Proceedings of the 8th ACM Symposium on Access Control Models and Technologies, ACM Press,2003, pp. 100-109

[414] David Basin, Jürgen Doser and Torsten Lodderstedt, *Model Driven Security: from UML Models to Access Control Infrastructures* to appear in ACM Transactions on Software Engineering and Methodology. See also Technical Report No. 414, Department of Computer Science, ETH Zürichi, for a preliminary version.

8.22.1 Evaluation and Application to TrustCoM

The generative approach of Model Driven Security makes development of secure application both easier (quicker) and more robust, especially when considering frequent changes of requirements. For example, a full round-trip engineering would be possible.

Also, the idea of combining modeling languages by relatively small dialects scales better when one considers the multitude of security requirements, trust management policies, etc. that have to be considered in TrustCoM.

Finally, having policy specifications fully integrated into the system design models promises richer possibilities for analyzing these policies, e.g., inconsistencies, hidden implications, etc.

8.23 Conclusions

This section covered an overview of various frameworks, which support security management and their application to TrustCoM. It is important to realise that security management is not one-time task, which is configured at the installation phase but an ongoing active process, which should be managed throughout the life cycle of the system. The dynamic structure of VO needs support for changes in requirements to provide a secure environment and protect resources.

There are multiple and often compatible Authorisation and Access Control models. Some of these frameworks assign authorisations to keys, which may result in authorisations that are difficult to manage. Both Access Control for unknown entities and delegation were investigated at conceptual level. TrustCoM needs to combine them. This also applies for integration of Access Control and Public Key Infrastructure (PKI).

Policies play an important role in security management. TrustCoM need a flexible, expressive and extensible policy language to support the security requirements in VOs. Policies can be attacked as well, and therefore security management should be able to detect such attacks and modify the current policies within the system or load new policies. Current policy frameworks presented provide access control but lack support to control the flow of information to protect data confidentiality, which requires further work.

The emerging Web services security specifications (WS-Security, WS-Policy, SAML, XACML, etc.) already form the basis in a number of security frameworks in different domains, such as Grid and business processing, and are therefore expected to be of value in the TrustCoM framework. Further investigation is however needed in how exactly and to what extent these specifications will be able to address the security issues in VOs.

Adaptive Security models are starting to take some shape, but it is still early and requires substantial work, for example they do not cater for trust relationships. In VO trust relations may not be long term and stable, and the collection and evaluation of trust evidence will become a relatively frequent and crucial process. TrustCoM will need to design an Adaptive Security Management framework, which can monitor, integrate information from vendor independent devices and compute the representation of security events that may pose a threat and support response mechanisms in VOs.

Most of the introduced security frameworks are on-going projects and largely untested in systems of reasonable complexity. Further work is necessary to evaluate these frameworks on the basis of VO scenarios to identify security requirements for VOs, and examine the required technologies that make up the VO environment.

9 Legal Aspects

Edited by: Tobias Mahler
Norwegian Research Center for Computers and Law (NRCCL)

9.1 Introduction

Many attempts have been made, primarily in business and economics literature, to provide a definition of the virtual organization (VO) or to identify its main characteristics.[415] For the purposes of the legal part of the State of the Art report we mean by virtual organization a collaboration between several different and independent legal or natural persons that:

- Come together with their core competencies to provide a good or service on the basis of a common business understanding;
- Has not been set up by the partners as a separate legal person as such but appears to third parties as a homogenous enterprise;
- Cooperates and communicates internally principally through the use of ICT.
- A virtual organization is connected to a mission and ends with that mission[416].

This chapter describes the state of the art in those parts of legal analysis that are most important to the TrustCoM endeavour and to the understanding of virtual organizations. The chapter consists of two main sections. Section 9.2 provides an overview of studies on legal issues in virtual organizations, and assesses their relevance for legal aspects of the TrustCoM project. Section 9.3 shifts the emphasis to methodological issues, and surveys various description techniques of particular interest to the legal analysis of virtual organizations. The following four methods are discussed in turn: conventional analysis; legal risk analysis; semi-formal conceptual analysis; formal conceptual analysis. The goal here is to organize, integrate, and evaluate previous research from which TrustCoM might benefit. Suggestions for future research are then presented in conclusion.

9.2 Legal Issues in Virtual Organizations

The objective of this section is to provide an overview of studies on legal issues in virtual organizations and to assess their relevance for the legal study in the TrustCoM project.

9.2.1 Studies on Legal Issues in Virtual Organizations

The amount of legal literature on legal issues in virtual organizations is still limited[417]. However, legal issues in VOs were among the subjects studied in some EU research projects, most prominently the ALIVE project.

---

[415] For example, the ALIVE taxonomy tries to identify the characteristic features of a virtual organization from two points of view: economic and legal. See further C. Van Schoubroeck, H. Cousy, D. Droshout, B. Windey, *Virtual Enterprise Legal Taxonomy*, ALIVE Project Deliverable, 2001, available at http://www.vive-ig.net/projects/alive/Documents/Virtual_Enterprise_Legal_Issue_Taxonomy.zip (last visited 30.3.2004) at pp. 14-18. See also the ALIVE project's final report, pp. 12-15.

[416] See the definition by P. Mertens & W. Faisst (1997) translated into English by C. Odendahl & A.-W. Scheer, "The Concept of Virtual Enteprises and its Relevance for the Maritime Domain" in C. Guedes Soares, J. Brodda (Eds.), *Application of Information Technologies to the Maritime Industries*, Edições Salamandra, Lisbon, 1999. p. 13.

[417] Relevant legal literature includes the following: M. Mazzeschi, The Virtual Organisation; C. Van Schoubroeck, H. Cousy, B. Windey, D. Droshout, A Legal Taxonomy on Virtual Enterprises; T. M. Hassan, C. Carter, M. Hannus, J. Hyvärinen, eLEGAL: Defining a Framework for Legally Admissible Use of ICT in Virtual Enterprises; Emily M. Weitzenboeck, Building a legal framework for a virtual organisation in the maritime domain: the MARVIN experience, all aforementioned papers appear in

9.2.2 ALIVE IST Project

The ALIVE (Advanced Legal Issues in Virtual Enterprises) IST project[418] has conducted a study to identify and address legal issues in virtual enterprises (VE). Among the results, the following documents are publicly available[419]:

- The reference Virtual Enterprise life-cycle[420]
  - This document includes a survey of VO models (Virtual Enterprises, Extended Enterprises, concurrent Companies Network etc.).
- A reference Taxonomy for Virtual Enterprise Legal Issues[421]
  - This document identifies the key legal issues to be addressed during the rest of the project. The taxonomy is a reference framework, whose primary goal is to classify the main legal issues concerning the Virtual Enterprise in a certain hierarchic structure.
- The VE Legal Identity[422]
  - This document is aimed at analyzing the nature and the characteristics of the Virtual Enterprise, the relevant VE's legal structure and the legal identity of the Virtual Enterprise. It also contains a legislative proposal suitable for establishing a new corporate form, suitable for addressing the Virtual Enterprise's needs.
- The Role Of Actors in VE[423]
  - The report identifies the main legal issues concerning the key actors in the Virtual Enterprise.
- Specific ICT for VE[424]
  - Some of the most relevant topics in the field of information technology are reported in this document from a legal perspective. New and emerging technologies (among them, the software agent technology), suitable for supporting VE operations, are analysed also with respect to the context of electronic commerce as defined by Directive 2000/31/EC, where the VE is to be regarded as an Information Society Service (ISS), and the context of the Application Service Provider (ASP) portals.
- Liability & Insurance for VE[425]
  - The aim of this document is to outline the risk positions concerning the VE business and analyse the different possible liabilities that the partners and other parties of the VE

K.-D. Thoben, F. Weber & K.S. Pawar (eds), Proceedings of the 7th International Conference on Concurrent Enterprising: Engineering the Knowledge Economy through Co-operation, Bremen, Germany, 27-29 June, 2001, 2001, U.K., ISBN 0 85358 098 7, pp. 331-355; H. Cousy, C. Van Schoubroeck, B. Windey, The Virtual Enterprise Report on Techno-Legal Issues, 1999; E. Berwanger, The Legal Classification of Virtual Corporation According to German Law, in Organizational Virtualness and Electronic Commerce: Proceedings of the 2nd International VoNet Workshop, September 23-24, 1999, Simowa Verlag Bern; E. M. Weitzenboeck, Legal issues of Maritime Virtual Organisations, Complex 4/01, Unipub Forlag, Oslo, 2001, ISBN 82-7226-043-3, ISSN 0806-1912; Determining Applicable Law and Jurisdiction in Contractual Disputes Regarding Virtual Enterprises, in K.S. Pawar, K.-D. Thoben, F. Weber & (eds), Proceedings of the 8th International Conference on Concurrent Enterprising: Ubiquitous Engineering in the Collaborative Economy, Rome, Italy, 17-19 June, 2002, 2002, U.K, ISBN 0 85358 113 4, pp. 27-34.

[418] IST-2000-25459, www.vive-ig.net/projects/alive.

[419] The descriptions of the reports are adapted versions of the texts available on the ALIVE web site, http://www.vive-ig.net/projects/alive/docs.html, last visited 25 Mar. 04. All the reports are available there.

[420] CE Consulting, Sergio Sorrentino, Roberto Santoro.

[421] Institute of Trade Law and Insurance Law of the K.U.Leuven-University, Caroline Van Schoubroeck.

[422] Mazzeschi & Partners, Marco Mazzeschi.

[423] KSW, Thomas Wallentin / Ernst Muehlfellner.

[424] CIRSFID, Claudia Cevenini.

[425] University of Lapland, Juha Poyhonen.

could face and how the liability relations could be arranged, not only within the VE, but also with respect to third parties.

- Intellectual and Industrial Property Rights[426]
    - In a complex structure like the VE in which the members combine to develop a common project without organizing an independent legal person, protection of intellectual and industrial property rights play a role of prime importance for its success. This document analyzes the legal and technical aspects of the protection of intellectual creations, inventions, trademarks and know-how in the context of the VE.
- Consumer Protection and contracting with 3rd parties[427]
    - The subject matter of this document is the analysis of issues relating to contracts concluded by the VE and third parties other than the VE members. Contracts with third parties include both B2C and B2B transactions. Where necessary specific mention is thus made to the consumer protection law.
- Competition Law and Control Legislation[428]
    - This report analyses the degree to which competition law regulates the activity of a VE. This may depend, among other factors, on whether the VE can be classified as a SME.
- Tax Matters in Virtual Enterprises[429]
    - This report deals with the main consequences of the taxation in Virtual Enterprise (VE) operations scenarios, giving particular attention to the questions of tax presence and transfer pricing.
- VE Interchange Agreement[430]
    - This report contains the research work of the Legal Issue sub-group on the VE Interchange Agreement, which is the agreement setting up a legal framework for the virtual enterprise (also referred to as the "VE Agreement").
- VE Model contracts[431]
    - This report contains some suggested legal templates, which, with adequate legal advice, can be used in setting up a legal framework for a virtual enterprise.

9.2.3 Application to TrustCoM

The existing research on legal issues in VOs is of high relevance to the TrustCoM project. The abovezmentioned ALIVE reports “VE Interchange Agreement” and “VE Model Contracts” can function as the legal reference framework for setting up a virtual organization.

The legal studies to be conducted in TrustCoM will built upon the ALIVE results as a point of departure. All the three strands of legal research, i.e. Data Protection Law, Intellectual Property Law and International Issues in relation to VOs were to a certain degree covered in the ALIVE project. The objective of the following section is therefore to give a brief overview of the legal problems identified regarding the three strands of research described in the TrustCoM Description of Work.

---

[426] Garrigues, Ana Gil-Robles.

[427] CIEEL, Zoi Kardasiadou.

[428] FPS, Alexander Schmitz-Elsen/ Hendrik Härterich / Birgit Bert.

[429] QWM, Laura Edgar.

[430] NRCCL, Emily M. Weitzenboeck,

http://www.vive-ig.net/projects/alive/Documents/VE_Interchange_Agreement.zip

[431] NRCCL, Emily M. Weitzenboeck

http://www.vive-ig.net/projects/alive/Documents/VE_Model_Contracts.zip

9.2.3.1 Strand 1: Intellectual Property Law

Strand 1 of the TrustCoM activity "Legal Context" encompasses an investigation of legal issues in the following area of law: "Examination of selected measures in European Community law on copyright and sui generis database rights in intangible property".

The ALIVE research on intellectual property law in virtual enterprises[432] is of high relevance for this research. The ALIVE report covers issues like the protection of IPR depending on the legal identity of the VO and the protection of results through copyright, patents, and trademarks. The protection of know-how during the different stages of the life-cycle of a VO are discussed, and technical instruments for the protection of IPR are surveyed (Smart cards, biometric systems, certification authorities, digital notaries, digital signatures and time stamping services).

9.2.3.2 Strand 2: Privacy and Data Protection Law

The second strand of the legal research in TrustCoM will be the examination of selected data protection constraints on the collection, use, processing, storage, transfer, etc. of personal data, and requirements on data quality and data security as per the EU's Data Protection Directive (Directive 2002/58/EC) and the EU's Directive on privacy and electronic communications (Directive 2002/58/EC).

In the ALIVE project, privacy and data protection law was analyzed together with consumer protection law issues[433]. In the section dealing with data protection law, the report discusses whether the VO is a single data controller. The author points out that each VO member has to comply with the national data protection rules. This leads to a number of specific obligations are surveyed in the report. They include, inter alia, the duty to specify the purpose of the processing of personal data and the obligation to assure that the data is necessary for the aforementioned purpose. No situation is explicitly identified where the obligations of VO members differ from the data protection obligations of any other type of organization. Consequently, VO members have to cope with the same obligations under data protection law as any other organizations. However, this does not exclude the possibility that the certain aspects of VO structures lead to special data protection problems. The TrustCoM legal study will have to analyze this structure in more detail.

9.2.3.3 Strand 3: International Issues in Relation to Virtual Organizations

The third strand of legal research in TrustCoM will be the analysis of problems arising from the international nature of the virtual organisation, where its members are established in more than one country: The national law of 1 to 2 jurisdictions (e.g. Norway, England) will be selected to illustrate how a virtual organisation may be legally classified and the effect this may have on the liability of its members. The legal classification of virtual organizations is analyzed in a number of ALIVE reports[434]. However, an analysis of the international nature of the VO, combined with a study of selected jurisdictions has not been carried out.

9.3 Methods for Legal Analysis

In the following we survey methods and techniques of interest to the legal analysis of virtual organizations. The presented approaches are classified into conventional analysis (section 9.3.1), legal risk analysis (section 9.3.2), semi-formal conceptual analysis (section 9.3.3), and formal conceptual analysis (section 9.3.4).

---

[432] http://www.vive-ig.net/projects/alive/Documents/Intellectual_and_Industrial_Property_Rights.zip

[433] ALIVE report Legal Issue Subgroup 6, Consumer Protection & Contracting with Third Parties, p. 16, available at http://www.vive-ig.net/projects/alive/Documents/Consumer_Protection.zip, last visited 25 March 04.

[434] Particularly the report on legal identity an classification of a VO, available at http://www.vive-ig.net/projects/alive/Documents/VE_Legal_Identity.zip.

9.3.1 Conventional Legal Analysis

A conventional legal analysis is in its form informal, and may be described as a legal argument. It may be seen as basically consisting of

- Exploring the legal issue to be solved
- Retrieving possible relevant legal sources
- Identifying which of the retrieved sources are relevant
- Interpretation of the relevant sources with respect to applicable legal norms
- Possible harmonisation of conflicting applicable norms
- Representing the resulting understanding of the applicable norms (or law)

A legal issue – a problem – is always the basis of a legal argument. The problem may be specific; as would be a case before a court, or more general; as would be the task of an academic lawyer writing a textbook. The applicable legal norms or the applicable law must always be based on legal sources, a term used to denote the sources determined by the basic meta-norms of the jurisdiction on which an argument of the existence or content of a legal norm must be based. The distinction between a legal norm and another type of norm (social, ethical) is determined by this basis on legal sources.

What are to be qualified as legal sources, will partly be determined by sources of high rank, like the constitution of a jurisdiction. But there is no known instance of an exhaustive list being formalised, therefore it will in the end be based on a consensus in the lawyer community, and the status of some types of sources may be contested (as are decisions by first or appellate instance court decisions in Norway). Typical sources are statutes, regulations (or secondary legislation), cases by supreme courts, and legal literature. The types vary between jurisdiction, for instance, legislative history is a source used intensively in Norway, but generally not recognised in United Kingdom.

Lawyers will rely on different strategies to retrieve sources that may be relevant to an issue. The two major strategies will be legal background knowledge and retrieval systems. A lawyer obviously may have extensive prior experience from similar issues, and will therefore know where to look for possible relevant sources, for instance which sections of the statutes that may apply. In addition, there are numerous retrieval systems available, from back-in-the-book indexes of key words or systematic terms, to sophisticated computerised retrieval systems (lawyers being the first profession to have all their primary material in full text available on-line). Hyperlinks will be part of the retrieval tools, traditionally in the form of citations.

The sources will typically (the exceptions are of little interest in this context) be texts. These will be made subject to interpretation. The process of interpretation may be trivial, reduced to a question of “reading” the texts. But it may also be more sophisticated, in which the doctrine on interpretation govern the process. This is qualitatively different from “reading” or “understanding” a non-legal natural language text, there being for instance norms governing the use of legislative definition, inter- or intra-consistence between regulations, analogue reasoning etc.

The interpretation process is also a learning process, through interpretation the lawyer understands more of the legal issues, and may have to redefine the problem, retrieve supplemental sources etc. This implies that stages 1-4 are iterative, and may be repeated until the lawyer has what is deemed to be an appropriate understanding of the law governing the issue (or more trivial, has run out of time or other scarce resources, and has to proceed anyhow).

The texts are of syntactic nature, the understanding of the law of semantic nature, and in the head of the lawyer arguing the issues. It may be described as arriving at an understanding of the norms governing the issues, a “norm” is somewhat further explained below. In some cases, the sources may contain sufficient leeway for there being available more than one set of norms with outcomes that not simultaneously can apply – in this case, there is a conflict of norms, and this must be harmonised. There are several principles for harmonisation, one being lex superior (a norm based on a source of higher rank is given predominance over a norm based on a source of lower rank) or client loyalty (the norm most favourable to the client is chosen). Also, the process of interpretation may have as an objective to remove possible conflicts of norms, to some extent the lawyer may

have the choice to harmonise the arguments in such a way that there is no conflict, or construe the arguments in order to identify conflicting norms).

In principle, the process of interpretation and harmonisation takes place in the thoughts of the lawyer. Obviously, they have to be represented – and the lawyer will ideally not only like an oracle come up with an applicable norm solving the issue, but explain which sources are identified as relevant, what problems of interpretation and harmonisation have been encountered, including how they have been resolved and why the lawyer has chosen to resolve them in that way, which then will lead up to the reasons (or justification) for the decision.

If the legal analysis concerns a contract, the legal method must reflect the nature of contracts. A contract is a document explicating the rights and duties between two or more parties. In this context, a "contract" is qualified as a text document, while a binding agreement does not have to be in writing (subject to the norms of formality in the law governing the formation of an agreement). Otherwise, the words "agreement" and "contract" often are used as synonyms.

A contract has to be contained within the regulatory norms applicable. Typically, these norms are very wide, and the situation is often described as giving the parties "freedom" to draw up contracts regulating in practice anything. The regulations will censor some contractual clauses (the traditional Norwegian statutory provision being that contracts have to be within the limits of "decency and good faith").

In principle, the contract is a legal source, but of a different kind than the legal sources mentioned above. Regulatory instruments are based on the authority of the legal system, which in the last instance is derived from the constitution (or, in the rare jurisdictions lacking a constitution, some basic norms typically of customary nature). The contract is based on the authority of the parties as physical or legal persons, which have the freedom to bind themselves legally by accepting duties. The legal system will back this up by resources for enforcing the contracts, typically through the court system and executive authorities, in the case of violation of the contractual duties.

9.3.2 Legal Risk Analysis

Legal risk analysis is founded on classical methodology for risk analysis like:

- FMEA – failure modes, effects and criticality analysis[435];
- FTA – fault tree analysis[436];
- HAZOP – hazard and operability analysis[437];
- Markov analysis.[438]

Over the years these methodologies originating from process industry and the safety sector have been reinvented and adjusted to new domains including the legal domain. Richard Susskind, who predicts a shift from legal problem solving to legal risk management, argues[439]:

"While legal problem solving will not be eliminated in tomorrow's legal paradigm, it will nonetheless diminish markedly in significance. The emphasis will shift towards legal risk management supported by proactive facilities, which will be available in the form of legal information services and products. As citizens learn to seek legal guidance more regularly and far earlier than in the past, many potential legal difficulties will dissolve before needing to be resolved. Where legal problems of today are often symptomatic of delayed legal input, earlier consultation should result in users understanding and identifying their risk and controlling them before any questions of escalation."

[435] Bouti, A., Ait Kadi, D. (1994) A state-of-the-art review of FMEA/FMECA. International Journal of Reliability, Quality and Safety Engineering 1:515-543.

[436] IEC 1025: (1990) Fault tree analysis (FTA), 1990.

[437] Redmill, F., Chudleigh, M., Catmur, J. (1999) Hazop and Software Hazop. Wiley.

[438] Littlewood, B. (1975). A reliability model for systems with Markov structure. Appl. Stat. 24:172-177.

[439] Susskind, R., The future of law, Clarendon Press, Oxford 1996.

Professor at Stockholm University, Peter Wahlgren has written a book[440] surveying how different methods from conventional risk analysis can be used to support and systemise legal work processes. He establishes that methods from risk analysis are well suited as a means to support and systemise the working process of several legal work tasks in particular contract analysis. The rest of this section is based on Wahlgren's book. For further references on risk analysis and law, see Baldwin[441] and Hart[442].

9.3.2.1 Legal Work Tasks from a Risk Perspective

Wahlgren claims that the risk perspective is relevant for legal questions for two reasons:

- Legal work tasks are tightly connected with risk analysis and
- Some legal work tasks explicitly require a risk analysis.

He gives an overview of legal work tasks from a risk perspective. Walhgren divides the work tasks into two categories: general work tasks, that takes place in most kind of legal processes, and situation dependent work tasks. Wahlgren points out that lack of knowledge about relevant documentation constitutes a risk, both for the legal subject and for the lawyer conducting the analysis in several of the work tasks he describes. Thus, the risk perspective is present in some way, either directly or indirectly, in most legal work tasks.

9.3.2.1.1 General work tasks

- *Legal analysis* includes the analysis of relevant laws and regulations that affects the actual business under consideration. Legal standard agreements often include an overview of different areas that can lead to uncertainty. Rule synthesis that are done for example by insurance companies often indicates risks to be clarified.
- *Situation analysis* includes investigating the factual situation of the bussiness undergoing legal analysis.
- *Managing communication* concerns the fact that lawyers must tackle various types of communication channels. The aim of the lawyer is to eliminate any room for misunderstandings. There is a risk connected to insufficient communication that may lead to misunderstandings.

9.3.2.1.2 Situation dependent work tasks

- *Risk analysis*; in some situations the law actually requires that a risk analysis is conducted. For example the Norwegian provisions that regulates the compilation and storage of personal data requires that the person responsible for handling the data conducts a risk analysis in order to clarify the likelihood and consequences of a security breach. For further references to regulations in other European countries that requires risk analysis, see Suokas et al.[443].
- *Contract analysis* includes identifying involved parties, identifying and distributing work tasks, controlling access to legal protection conflict solution and reparative functions. When lawyers set up a business agreement, it is common to identify risks actualised by the agreement. Several agreement points address risks. A typical business agreement also distributes risks among the partners. This aspect of legal agreements is also described by Skidmore[444].

---

[440] Wahlgren, P., Juridisk riskanalys – Mot en säkrare juridisk metod. Jure AB, 2003.

[441] Baldwin, R., (ed) Law and uncertainty: Risks and legal processes, Kluwer law international, London, The Hague, Boston 1997.

[442] Hart, D., Towards risk management in contract law. Perspectives of critical contract law, Dartmouth publishing company limited, Aldershot 1993.

[443] Suokas, J., Kakko R., Safety analysis, risk analysis, risk management. Quality management of of safety and risk analysis, Elsevier Science Publishers B.V., Amsterdam, 1993.

[444] Skidmore, P., Whose risk is it anyway? Allocation of entrepreneurial risk in employment contracts. Law and uncertainty: Risks and legal processes, Kluwer Law International, London, The Hague, Boston 1997. pages 221-239.

Based on this survey Walhgren ascertains that several legal work tasks can be seen as pure risk management. He also concludes that several legal work tasks require that a risk analysis is conducted from a non-legal point of view.

9.3.2.2 Legal Risk Management

Walhgren describes methods used in conventional risk analysis and sketch how these methods can be used in order to support and systemise the general task of legal analysis and the more specialised task of contract analysis.

Statistic and quantitative methods can be used in legal analysis in order to document which questions are normally the most difficult to review over a longer period of time, which legal sources one depends on, the cost of information search etc. Within contract analysis statistic methods can be used to document the types of questions that need to be regulated through comparing data from similar industries or earlier business deals for similar industries.

Within contract analysis a *fault tree analysis* is well suited to investigate what can go wrong in particular situations by decomposing an event into smaller components. It can also be used within legal analysis in order to get an overview of the different constituents of a certain rule.

The scenario method is well suited within contract analysis to guard against unforeseen events, such as force majeure. In the scenario method one takes one event as a starting point and brainstorms in order to identify all kinds of negative events that the initial event can lead to.

The use of *detailed checklists* is well suited within contract analysis to ensure that nothing essential is left out. Such lists identifies information that needs to be controlled with regard to the target of the agreement,

Matrixes serves the same purpose as check lists but can be used to bring an extra dimension to identified risks, for example though relating risks to certain parts of a contract.

Exponentiation studies can be used within legal analysis to study effects of lacking knowledge with regard to certain communication tools. An exponentiation study is an analysis of how exposed an industry is with regard to already identified risks.

Walghren concludes that methods from risk analysis are well suited as a means to support and systemise the working process of several legal work tasks in particular contract analysis. Furthermore he argues that the risk managing features of legal tasks implies that legal issues should be taken into consideration at a much earlier stage in planning processes than are the case to day. Walhgren argues that integrating legal work tasks with other security work, can lead to a shift in the focus of legal problem solving.

9.3.3 Semiformal Conceptual Analysis

By semiformal conceptual analysis we mean conceptual analysis using semiformal description techniques. A semiformal description technique looks formal, but is called semiformal because the grammar and/or meaning of descriptions expressed with the help of this technique are not fully defined[445].

In the following we briefly survey semiformal description techniques of relevance for legal analysis of virtual organisations.

[445] In contrast, a formal description technique has a well-defined grammar and a meaning captured in some well-understood mathematical structure.

9.3.3.1 Arrowdiagrams

Arrow diagrams are a “formalism” proposed by Layman Allen, using a graphical approach the representation of legal norms[446]. Any legal norm will have the general structure of an antecedent and a consequent: if [A] then [B]

The antecedent prescribes which factual circumstances have to be present for the norm to fire. The consequent indicates the effect of the norm being applied, often the effect is only a link to further norms, which chained together will represent the norms governing the case at hand. See section 13.1, of the appendix for more details.

9.3.3.2 Flowcharts

A flow chart is a graphical illustration of which criteria that must occur in order for certain consequences to happen. It is often well suited to get an overview of a contract. In fact, Walghren argues[447] that the transparency of a flow chart representation helps to avoid risks.

9.3.3.3 Unified Modeling Language

A recent standard incorporating and building on the same ideas as flow charts and sequence diagrams is the Unified Modeling Language (UML)[448] that has been standardized by the Object Management Group (OMG). OMG is the leading standardization body for the software industry. Almost every commercial software developer uses UML or related notations in one form or another.

9.3.3.4 CORAS UML Profile for Security Analysis

The CORAS methodology[449] for model-based security risk analysis[450] and its UML profile for security risk analysis that has recently become a recommended OMG standard[451], facilitate risk analysis based on graphical descriptions in the style of UML. The profile provides a meta-model defining an abstract language supporting model-based security risk analysis. The classes in the meta-model are mapped to modelling elements by definition of so-called stereotypes. A stereotype is a specialisation of a predefined modelling element in UML. The profile introduces special symbols (icons) for representing these stereotypes in UML diagrams. Since the profile is defined by means of extension mechanisms defined in the UML standard, it is compatible with UML. The profile supports the practical use of UML in security risk management in general, and security risk analysis in particular.

In the model-based security risk analysis methodology of CORAS, UML models are used for three purposes:

- To describe the target of evaluation at the right level of abstraction.
- To facilitate communication and interaction between different groups of stakeholders involved in a security assessment.
- To document security assessment results and the assumptions on which these results depend to support reuse and maintenance.

The UML profile supports all these objectives, but has a special emphasis on communication and documentation. Documentation is supported because the meta-model of the profile is consistent with

---

[446] ”Norm” is preferred to ”rule”, the reason for this is of little consequence to this note, but is derived from a theory of norms in which ”rule” is one of several categories of norms.

[447] Wahlgren, P., Juridisk riskanalys – Mot en säkrare juridisk metod. Jure AB, 2003.

[448] OMG, Unified Modeling Language: Superstructure. OMG ad/03-04-01, (2003).

[449] The CORAS methodology, profile and open source tool were developed in the 5th FP EU project CORAS (IST-2000-25031). The CORAS-tool and full documentation is freely available at coras.sourceforge.net.[449]

[450] Mass Soldal Lund, Ida Hogganvik, Fredrik Seehusen, Ketil Stølen. UML profile for security assessment. Technical report STF40 A03066, SINTEF, December 2003.

[451] OMG: UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms, Revised submission to OMG RFP ad/2002-01-07. OMG document: realtime/2003-08-06. Object Management Group (2003)

a data structure of security risk analysis documentation developed as part of the CORAS project. Communication is supported by the definition of easy-to-understand icons associated with the modelling elements of the profile, and because these specialised modelling elements are consistent with the ontology of security risk analysis.

See Page 368 of the appendix for further details.

9.3.3.5 Extensible Rights Markup Language

XrML[452] (eXtensible rights Markup Language) is a language to specify "rights". XrML is an XML-based syntax for specifying rights and conditions to control the access to digital content and services. XrML had its roots in Xerox Palo Alto Research Center. Digital Property Rights Language (DPRL) was first introduced in 1996. DPRL became XrML when the meta-language (used to construct the language) was changed from a lisp-style meta-language to XML in 1999. See Page 370 of the appendix for further details.

9.3.3.6 LegalXML

LegalXML is an attempt to create standards for the electronic exchange of legal data, in particular laws, statutes etc. This relates to theimprovement of legal information systems and falls outside the scope of TrustCoM.

9.3.3.7 Enterprise Privacy Authorization Language

The Enterprise Privacy Authorization Language (EPAL 1.1)[453] is a formal language for writing enterprise privacy policies. It was developed by the IBM Zurich Research Laboratory in 2003. EPAL is based on XML and is designed to administrate authorizations to conduct privacy-relevant actions defined by any enterprise. See Page 371 of the appendix for further details.

9.3.4 Formal Conceptual Analysis

This section is structured into four subsections:

Sections 9.3.4.1, 9.3.4.2 provide general comments about the multi-modal language described in [454], including remarks on methodology.

Section 9.3.4.3 pertains to the application of the multi-modal language to the characterisation of communicative interaction, on the assumption that this is of key importance to the formal analysis of aspects of contractual situations. It also touches on the issue of contract violation, and the formal representation of 'reparational' obligations.

Section 9.3.4.4 pertains to the application of the multi-modal language to the analysis of *obligation, permission, right, power, role* and *trust*. These concepts will play a central part in the analysis of legal aspects outside of strictly contractual situations – for instance in the regulations themselves, and in the policies governing organisational procedures (This is not to deny, however, that most of these concepts will *also* figure in many contracts).

The reader is asked to keep in mind that this section only summarises some main features of particular interest to the TrustCoM project in general, and to the 'legal aspects' work in particular, and is referred to [454,455,456] for development of details.

---

[452] XrML 2.0 Technical Overview, Version 1.0, March 8, 2002.

[453] This text is based on the EPAL Specification (IBM Research Report), available at http://www.zurich.ibm.com/security/enterprise-privacy/epal/, last visited 21 April 2004.

[454] Andrew J I Jones, 'A Logical Framework', to appear in J. Pitt, ed., *The Open Agent Society*, John Wiley & Sons, Chichester, UK, forthcoming. This work is part of the output of the European 5th Framework Project ALFEBIITE (IST-1999-10298).

9.3.4.1 Introduction

The modal-logical language presented in [454] contains a number of modalities (the 'basic building blocks'), together with illustrations of how these modalities may be combined to yield formal analyses of more complex notions; the illustrative examples chosen are communicative interaction, role and trust. Each of the component building block modalities is characterised axiomatically and model-theoretically, the latter in terms of 'minimal model semantics' in the sense of [457]. Soundness and completeness results for the multi-modal logic have been proved by Xavier Parent[458], and will be published in due course.

The principal modalities are:

- Praxiological modalities to represent the actions, attempted actions and abilities of individual agents;
- A modality to represent agent beliefs;
- Directive normative modalities, to represent an agent's permissions and obligations;
- Evaluative normative modalities, to represent that which is optimal, or ideal;
- A dyadic modality to represent so-called 'counts as' conditionals (More on this below, in connection with the concept of institutionalised power).

In addition to their application to the analysis of communicative interaction, role and trust, these modalities have also been employed in the formal characterisation of normative positions, which themselves provide a platform for characterising rights. A very recent example of this work is presented in [455], the bibliography to which supplies references to earlier work of this kind. See section 9.3.4.4, below, for more on normative-informational positions.

The analysis of rights would be seriously incomplete if it failed to cover those rights which agents acquire when they are empowered by some institution to establish particular types of states of affairs by means of their performance of designated acts: for instance, when a Head of Department is empowered to create valid claims for expenses by appending his signature to a claims form. This notion of institutionalised power is of fundamental importance in understanding how institutions/organisations – including of course legal institutions – conduct their business. Its formal analysis is described in some detail in [456], principally in terms of a combination of the action and 'counts as' modalities. Again, see section 9.3.4.4, below, for more on this.

9.3.4.2 Aim of Formal Conceptual Analysis

The aim of formal conceptual analysis in relation to a given domain is to give precise characterisations of the meanings of the concepts central to that domain and, importantly, to show how those concepts are related to each other. Using a logic as the tool for expressing a formal conceptual analysis has the advantage of facilitating the systematic investigation of implication relations between sentences containing the concepts themselves, and – the other side of the same coin – testing for the consistency of sets of sentences in which the concepts figure. Furthermore, the logical language can be used to pin down, in a very precise fashion, ambiguities that may arise in natural-language expressions in which the concepts occur. Consider, to take just one example, the concept authorisation, in relation to the following two sentences:

- The HoD is authorised to park his car in the car park.
- The HoD is authorised to assign teaching duties.

---

[455] Andrew J I Jones, 'On Normative-Informational Positions', to appear in A. Lomuscio & D. Nute, eds., Proceedings of the Seventh International Workshop on Deontic Logic in Computer Science (□EON04), LNCS/LNAI, Springer-Verlag, Berlin, Germany, May 2004.

[456] Andrew J I Jones & Marek J. Sergot, 'A Formal Characterisation of Institutionalised Power', Journal of the IGPL volume 4 (3) 1996, pp. 427-443.

[457] Brian F. Chellas, Modal Logic: an introduction, Cambridge University Press, Cambridge, UK, 1980, chapters 7 and 10.

[458] Xavier Parent has been appointed Research Assistant on the TrustCoM project at King's College London.

In the first case, 'authorised' means 'permitted' (and if the permission is a right, it will also carry an obligationn on relevant others, e.g., the car park gatekeeper, to grant entry). In the second case, 'authorised' means 'empowered'. Recognition of the conceptual distinction between permission and empowerment goes back at least 100 years in legal theory (see [456] for discussion). In representing the norms, rules and procedures that govern the behaviour of agents in organisations, this distinction is important; within the multi-modal language, it is readily articulated.

What then is the aim of this kind of formal conceptual analysis in relation to the enterprise of software engineering? Essentially, the answer is that the formal-logical conceptual characterisations form a 'middle-layer' between, on the one hand, the initial natural-language descriptions of the rules, policies and procedures that govern and constitute some organisation or business process, and, on the other hand, the specification and design of a computer system whose function is to support, or to take over, some aspects of the organisational or business processes. Natural-language descriptions of the obligations, rights powers, etc. that lie at the core of a norm-governed system are conceptually complex, and are permeated by ambiguity and nuance. The 'middle-layer' provides a means of sorting out these complexities prior to system design, so that the systems engineer is supplied with a precise semantical interpretation of those rules, policies and procedures to which his system, when built, will be required to conform. There are obvious dangers involved if the software engineer moves directly, from poorly articulated intuitions about the interpretation of the initial, natural-language description of the rules, policies and processes, to the level of system implementation. The aim of the 'middle-layer' is to help guard against these dangers by providing a rigorously formulated semantics for the key concepts.

Now it may well be that the software engineer, in moving from 'middle-layer' to system construction, finds that he does not need to capture, or take into account, the full range of semantical detail that the 'middle-layer' provides; there may be particular aspects and nuances reflected in the 'middle-layer' that – for the purposes of the application he is designing – he can safely ignore: simplification may be perfectly acceptable. But note then that, in that situation, the 'middle-layer' provides a semantical reference point in relation to which the precise nature and degree of the simplification can be made fully explicit. In short, the three-layer methodology here advocated provides a systematic means for determining the extent to which the software system, when designed, captures the content of the initial natural-language description of the rules, policies and procedures that govern the organisational and business procedures themselves.

The criteria that guide the development of the 'middle-layer' itself are not primarily concerned with issues of computational tractability, but with semantical adequacy; that is, the key question is whether the formal conceptual framework is able to express, in a precise fashion, central intuitions about the interpretation of the basic concepts. Each of the three sources [454], [455], [456] provide examples of how the task of formally capturing such intuitions proceeds.

The comment in the previous paragraph should not be understood as as an underestimation of the importance of the transition from the 'middle-layer' to the computational level. In fact work on the logical framework described in [454] began prior to ALFEBIITE, during the EC ESPRIT Basic Research Projects 3125 MEDLAR (Mechanising Deduction in the Logics of Practical Reasoning, 1989-92) and 6471 MEDLAR II, (1992-95), and – to a lesser extent – in the EC ESPRIT Working Group 8319 MODELAGE (Modelling Cooperative Intelligent Agents, 1995-98). The main focus of the MEDLAR and MEDLAR II research was on a range of different approaches to mechanised deduction for modal logics, both normal systems of modal logic and non-normal systems[459] (The ALFEBIITE logical framework contains instances of each of these two types.) Furthermore, within the project ALFEBIITE (2000-2003) a considerable amount of research effort was devoted to making computational sense of some of the modal systems employed in the logical framework[460]. Further instances of this kind of work will be mentioned in the next section.

[459] See Journal of the IGPL volume 4 (1) and volume 4 (3), 1996, for papers representing some of the output of the MEDLAR endeavour.

[460] See, in particular, the contributions of Marek Sergot and Alessio Lomuscio to J. Pitt, ed., The Open Agent Society, John Wiley & Sons, Chichester, UK, forthcoming.

9.3.4.3 Methods for Contract Analysis

While it is clear that contracts will employ many of the concepts that are definable in terms of the modalities described in [454], it is perhaps particularly the analysis of communicative acts provided therein that is of most immediate relevance to the semantical characterisation of key aspects of contractual processes. Furthermore, since legal risk analysis will also often be concerned with scenarios in which inter-agent communication is involved, the modal representation of communicative acts will be of importance there too.

The theory of communication developed in [454] was in part formulated in response to a 'request for information' issued by the standards body FIPA (Foundation for Intelligent Physical Agents) in 2001, and was presented at three FIPA meetings in the course of 2001 and 2002. At that time, FIPA expressed an interest in approaches to ACLs (Agent Communication Languages) that departed from their intention-based paradigm and focused instead on publically accessible aspects of communication. The Jones account of ACLs puts the notion of convention, rather than intention, at the heart of the analysis, and employs the 'counts as' connective of the logical framework, together with action operators and a specific type of optimality operator, to specify the logical forms of those conventions. Several fundamental types of communicative act are defined; a strong case can be made for maintaining that other types are just variants of these, or variants that arise in the context of particular conversations (inter-connected sequences of communicative acts). In order to develop the essentially static account of communicative act types presented in [454] into a dynamic account of conversation, the logical framework has been enriched with a version of 'arrow logic'; a preliminary account of this development is contained in [461], and includes an illustrative application to the representation of an auction protocol. Significantly from the point of view of the monitoring of contract execution and the analysis of risk, the method facilitates the representation of the changing informational states of the communicating parties during an ongoing conversationally mediated transaction; this is currently the subject of further development.

Returning to the comparison with the FIPA approach to ACLs, it is noteworthy that the new convention-based approach drops FIPA's totally unrealistic assumptions concerning communicator sincerity, and leaves wide open the possiblity of insincere, unreliable and deceptive communication. The development of a formal theory of deception will be a natural next step. It is also worth noting that the 'intention v. convention' issue that has figured in discussion of so-called 'speech act theory' since the 1960's, and which is now a key factor in the development of ACLs for multi-agent systems, has a strikingly significant parallel in legal theories concerning the way in which contractual obligations arise. Recent work has shown how this issue is re-emerging in the discussion of the potential role of electronic agents in contract formation[462].

Finally, mention should be made of ongoing collaboration between Jones and Steven Kimbrough (The Wharton School, University of Pennsylvania[463]), aimed at a synthesis between the convention-based account of communicative acts and Kimbrough's FLBC (Formal Language for Business Communication)[464]. Since Kimbrough's formal theory is expressed in terms of first-order logic, the expected outcome of this collaboration is a clearer picture of the extent to which it is possible to approximate the detail of the formal language employed by Jones in purely first-order terms. This point has clear connections with issues raised in the 9.3.4.2, above, not least with respect to the development of computational models of the multi-modal logical framework.

With regard to the topic of contract analysis, another challenge to the development of adequate reasoning mechanisms pertains to the common phenomenon of reparational or correctional obligations – usually called 'contrary-to-duty' obligations by deontic logicians. These are obligations that come into force when some other obligation has been violated, as, for instance, when an

[461] Andrew J I Jones & Xavier Parent, 'Conventional Signalling Acts and Conversation', in F. Dignum, ed., Advances in Agent Communication, LNAI 2922, Springer Verlag, Berlin, Germany, 2004, pp.1-17. Xavier Parent has produced soundness and completeness results for the logical framework extended with arrow logic, to be published in due course.

[462] Emily Weitzenboeck, 'Electronic Agents and the Formation of Contracts', Journal of Information Law and Technology volume 9, 2001, pp.204-234.

[463] The Wharton School is one of the leading Business Schools in the USA.

[464] See, e.g., Steven Kimbrough, 'Formal Language for Business Communication: Sketch of a Basic Theory', International Journal of Electronic Commerce, volume 3 (2), Winter 1998-99, pp. 23-44.

obligation to notify the buyer comes into force when the vendor's obligation to deliver the ordered goods by date-due is not going to be met. Secondary, reparational obligations of this sort lie at the very heart of contractual arrangements, but their logical analysis has posed a number of tricky problems. A detailed account of the state of the art for treatments of these issues, together with an extensive new theory for their solution, is to be found in the monograph[465]. An automated reasoning system capable of handling the key features of the logical theory of [465] is currently being developed by Audun Stolpe[466].

9.3.4.4 Methods for Analysis of Extra-contractual Situations

[455] reports preliminary work on an application of the formal theory of normative positions to the analysis of norms governing the supply of information. The theory is intended to provide a platform for characterising such legal rights as the right to know and the right to silence, which are important elements in Data Protection Law. So the links with projected TrustCoM work on legal aspects are obvious. As the account in [455] indicates, the theory also provides a contribution to the systematic analysis of deceptive and misleading information transfer. In addition, future work will discuss P3P (Platform for Privacy Preferences) in relation to the theory of normative-informational positions.

As indicated in the introductory remarks, [456] develops the analysis of institutionalised power, and provides some illustrations of how that analysis can be applied to the formalisation of such notions as delegation and authorisation, which play a central role in the policies that govern organisational activities. Coupled with the account given in [454] of role and trust, there is good reason to believe that a firm basis can here be provided for a formal semantics for, for instance, the PONDER policy language.

There is also very good reason to believe that the account of institutionalised power, combined with the analysis of communicative acts of the declarative type provided in the convention-based approach to ACLs, can provide a rich semantical framework for interpreting the legal notions of representation, delegation and mandate[467]. Furthermore, as is indicated in [468], a multi-modal framework might also be applied to the formal analysis of aspects of XrML (eXtensible rights Markup Language). For reasons given in the paper referred to in footnote 467, however, there are grounds for supposing that the formalism adopted by Gelati et al. can be significantly improved.

9.4 Concluding Remarks

This chapter has provided a detailed and comprehensive review of the state of the art for a number of issues that are highly relevant to the analysis of legal aspects of the TrustCoM project. Potential directions for future research have also been identified. It is clear that these are suggestions only, at this stage, but they can nevertheless serve to focus the next steps in our research.

Legal analysis is time consuming and costly. A major role of legal risk analysis is to reduce costs increase effectiveness by pinpointing the most important legal vulnerabilities and risks requiring in-depth legal analysis. The legal risk analysis should be asset-driven taking the business assets of the virtual organisation to be designed as input. Legal risk analysis is a fairly recent, but active field of research. Some like Richard Susskind predicts a shift from legal problem solving to legal risk management:[469] "While legal problem solving will not be eliminated in tomorrow's legal paradigm, it will nonetheless diminish markedly in significance. The emphasis will shift towards legal risk management supported by proactive facilities, which will be available in the form of legal information

[465] José Carmo & Andrew J I Jones, 'Deontic Logic and Contrary-to-Duties', in D. Gabbay & F. Guenthner, eds, Handbook of Philosophical Logic, Revised Edition, volume 8, pp. 265-343, Kluwer Academic Publishers, Dordrecht, Holland, 2002.

[466] Formerly an RA with the King's College London ALFEBIITE group; currently at the University of Bergen, Norway.

[467] The account is supplied in the internal ALFEBIITE report 'A response to Gelati et al.', by Andrew J I Jones, Audun Stolpe and Xavier Parent, available on request from ajijones@dcs.kcl.ac.uk

[468] [GSR03], Jonathan Gelati, Antonino Rotolo & Giovanni Sartor, 'A Logic-Based Analysis of XrML', in LEA 2003: The Law and Electronic Agents, Complex 5/03, Norwegian Research Centre for Computers and Law, Oslo, Norway, 2003, pp.57-68.

[469] Susskind, R., The future of law, Clarendon Press, Oxford 1996.

services and products. As citizens learn to seek legal guidance more regularly and far earlier than in the past, many potential legal difficulties will dissolve before needing to be resolved. Where legal problems of today are often symptomatic of delayed legal input, earlier consultation should result in users understanding and identifying their risk and controlling them before any questions of escalation." Understanding how to exploit legal risk analysis in the legal analysis of virtual organisations is seen as an important issue of research for the legal studies to be conducted in TrustCoM.

It is also suggested that the legal studies to be conducted in TrustCoM should build upon the ALIVE results, as a point of departure. All the three strands of research described in the TrustCoM Description of Work (International Property Law, Privacy and Data Protection Law, and International Issues in relation to VOs) were to a certain extent covered in the ALIVE project. But, as we have seen, some important questions remain, and need to be further addressed. In particular, it cannot be assumed that VO structures do not create special problems of their own. For instance, there is very good reason to believe that VO structures give rise to special data protection problems. A central conjecture that will direct our enquiries is that the methods for legal analysis that have been described in this chapter can help us appreciate better some of the complexities involved in the notion of virtual organization. Although a thorough investigation of how these methods interact remains to be carried out in the coming months of our collaboration, we are confident that the cross-fertilising of these paradigms will provide new insights, which we shall endeavour to feed into the construction of the TrustCoM conceptual framework.

10 Conclusions

This State of the Art document surveys a vast and comprehensive collection of related work covering both implementation frameworks and conceptual models across all the thematic areas relevant to the TrustCoM project. Their potential contribution to the project has been highlighted in each case and areas where a more detailed study is required have been identified. In most cases this detailed study will need to be undertaken against the scenarios from which the detailed requirements for the TrustCoM Framework will be drawn and which are described in Deliverable D3 of the project. To our knowledge, it is the first time that such a comprehensive collection of related work has been reviewed and the knowledge that it embodies will be invaluable beyond the realm of the TrustCoM project to other attempts at building integrated frameworks for the next generation eBusiness environments.

Perhaps the most important conclusion that can be drawn from this State of the Art survey is that there is little, if any, work that attempts to combine support for virtual organisations with collaborative business processes, contract management, trust and security in an integrated framework. Any integration studies address at most two of these areas and the "integration" remains rather loose. Therefore, there are no examples upon which the TrustCoM framework can build and the integration of these areas remains a fundamental research problem as well as an engineering one.

Evaluations of the most important tools, techniques and conceptual frameworks have been given throughout the document and the reader is referred to them for more information. However, it is worth summarising here the most salient points on which there is overall agreement between all parties involved.

- **Socio-Economic Aspects**. The impact of trust and reputation in business transactions needs careful evaluation. Although preliminary studies exist, this work must also continue within the TrustCoM project. Most existing studies refer to business transactions in general and do not take into account support frameworks that provide the means of maintaining and periodically re-evaluating quantified measures of trust, risk and reputation. Of particular interest are therefore the assurances that the TrustCoM framework can provide to the user by adapting the underlying mechanisms to changes in trust, risk and reputation. It is expected that by adapting the security infrastructure the overall trust in the functioning of the Virtual Organisation can be maintained or increased while reducing the amount of risk and preserving the reputation of each of the participants in the VO and of the VO as a whole.
- **Frameworks for Virtual Organisations.** From our investigation, there are no frameworks committed to the exclusive modelling of trust, contracts and security in VOs. The existing frameworks have been for the most part concerned with structural compatability, process interoperability, roles and relationships across multiple organization domains, or very broad notions of constraints on actors in the VO. Trust, Contracts and Security are mostly mentioned as orthogonal interests. In addition, these models and reference architectures have focussed on either reaching a conceptual understanding of VOs, enabling requirements derivation, or providing a framework for resource sharing and enacting processes of the VO. In TrustCoM we seek to cut across this spectrum by conceptually understanding trust, contract management and security in VOs, deriving their requirements and hence providing functionality to support our reference implementations. We will therefore not select one "champion framework" initially, rather consider the most appropriate combination as the project advances. A better understanding and representation of social actors and business relationships is also important for properly representing trust, contracts and security management of VOs, as they will inevitably conduct transactions with physical enterprises and customers.
- **Contract and Service Level Agreements**. Despite common features required in the management of both collaboration agreements and service level agreements the areas remain quite distinct. Work on service level agreements derives mostly from network and systems management and has an operational focus tied into concrete implementation. While substantial work has been done towards the specification of service level agreements that refer to specific service instances with well-defined interfaces and Quality of Service metrics, further work

remains to be done towards automating the instrumentation of services and applications in order to verify compliance with the agreement. Work towards automating negotiation of collaboration agreements and service level agreement remains in its early stages and is mostly limited to negotiation of SLA parameters in pre-defined templates. Amongst the various emerging standards WSLA and WS-Agreement are particularly promising: the first because it adopts a very pragmatic viewpoint which is close to the implementation, the second because of the promised integration with the other WS-standards. Support for contract management beyond service level agreements remains weak and this is perhaps the area where innovation in TrustCoM would lead to significant results. Amongst the work reviewed, the Business Contract Architecture (BCA) stands out because it attempts to cater for most stages in the contract life-cycle. However, it is less clear to what extent the functionality described in the BCA framework has been implemented.

- **Collaborative Business Processes.** There is substantial work within this area and standards are relatively mature. Amongst the different standards reviewed the Web Service Choreography Interface (WSCI) and Business Process Execution Language for Web Services (BPEL4WS) are designed to reduce the inherent complexity of connecting Web services together and stand out as the most promising technology to be used within the TrustCoM framework. Their focus is different and complementary: BPEL4WS focuses on the executable processes and control from the view point of one of the parties whilst WSCI focuses on the exchanges of messages across the web-services involved.

- **Enabling Technologies.** In essence, there are two types of enabling technologies that have been considered in this survey: middleware technologies that form the interoperability layer between the components of the TrustCoM framework, facilitate their operation and enable the functioning of the virtual VO, and specific tools that have been developed by TrustCoM partners in the past and could be used towards achieving the project objective. TrustCoM has already committed to a web-service oriented architecture, and several standards and implementation platforms described in this document could be used to support the development effort. The choice between competing standards and competing implementation platforms needs to be made based on a detailed study of the requirements emerging from the scenarios and the expertise of the partners. Grid platforms offer a particularly relevant collection of enabling technologies on which the TrustCoM platform could be built. However, they introduce strong dependencies on other services within the same platform, have a steep learning curve, and the dependablity of the various implementations varies substantially. Therefore, adoption of any Grid environments for use within the TrustCoM project needs to be decided carefully based on actual need in specific scenario cases. There are a number of tools developed by the various partners over the years and a preliminary evaluation shows that they would prove valuable within the context of the project. However, the implementation maturity of these tools varies substantially and almost all of them would require significant adaptation to be used within the TrustCoM framework.

- **Trust Management.** Several metrics have been defined for trust and reputation though none of them has proved to be successful or to gain universal acceptance. It is therefore important that the TrustCoM framework is designed in such a way to allow different metrics to be used or combined. The area of trust management is still new and substantial work remains to be done towards the deployment of trust management frameworks, especially within an eBusiness setting. The overall building blocks of risk assessment services, trust services and reputation services are often used. Their integration in a coherent framework is much less understood. The need, role and operational model of mediation and arbitration services needs to be closely investigated with respect to the scenarios. On the other hand the public-key infrastructure aspects of trust are relatively well understood at least at the conceptual level. An emerging area is that of trust negotiation in terms of incremental disclosure of information between partners. This is particularly important in an eBusiness setting as policies, certificates or other attributes are often considered business sensitive and should be disclosed only in specific circumstances.

- **Policies and Security.** As the contents of the section indicate there is a substantial amount of work in this area and many partners within the TrustCoM consortium have substantial expertise. Numerous standards are emerging including XACML, SAML and the WS-Policy series which includes WS-Trust, WS-SecurityPolicy etc. Some of them have already reasonably well-known

implementations, for others, implementations are in progress or restricted to specific platforms. However, to a certain extent, the expertise and prototype tools developed by partners within the TrustCoM consortium are more advanced than the solutions proposed by the emerging standards. The challenge is therefore to develop an integrated solution that remains interoperable with the major standards being developed whilst providing advanced solutions to security management within virtual organisations. This also implies that the standards currently under development need to be monitored closely throughout the project. One aspect which stands out across all of the related work is that most frameworks consider security as being either static or managed by a human administrator. Substantial work is therefore needed within the TrustCoM project in order to develop an adaptive security framework that can be reconfigured dynamically according to changes in requirements, risk, or trust in the other entities without human intervention.

- **Legal aspects.** Substantial work has been done in the Alive project which will need to be extended, improved and adapted for use within TrustCoM. There is also a need to conduct legal analysis of specific areas within the TrustCoM project which will need to be identified based on both the scenarios and the architecture of the TrustCoM Framework.

11 ANNEX I – Basic Security Technologies

11.1 Introduction

This section gives details of the network protocols, authentication mechanisms, firewalls, and intrusion detection and intrusion response systems to help those readers, which may not be familiar with basic security technologies.

11.2 HTTPR

The delivery of messages using a reliable transport mechanism is a fundamental component for middleware in e-business systems, and a needed technology in enterprise computing. However, in the wider context of the Internet, synchronous transport protocols such as HTTP do not currently provide those facilities. Reliable HTTP (HTTPR), a standard specification proposed by IBM, addresses these deficiencies by proposing rules that make it possible to ensure that all messages are delivered to their destination in their exact form and only once. In cases where the message delivery fails, the protocol will reliably report the message as undeliverable[470].

HTTP version 1.1 serves as the base upon which HTTPR builds its reliability. As such, all of the facilities of HTTP/1.1 (SSL, keep-alive, communication through proxies and firewalls, and so on) are available. HTTPR provides reliability through assigning a unique identifier, delivering feedback for each step of the request-response-communication, expecting intermediate storage of messages and separating the request from the response (thus allowing for asynchronous communication). Hence, HTTPR would normally be implemented via the proven concept of messaging agents. In contrast to HTTPR, existing messaging middleware products (MS Message Queuing, Oracle Message Broker etc.) unfortunately use non-standard, product specific protocols. The advantage of this proposal is that transactional reliance is being put in the transport level, allowing the application level to concentrate more on the business logic itself and also probably improving performance. With SOAP extensions outlined before (ebXML, GXA) this type of transactional reliance could also be achieved, but on application level[471].

11.3 Encryption

The contemporary encryption algorithms may be subdivided into three groups:

1. ***Symmetric ciphers*** encryption (involving secret keys);
2. ***Public key*** encryption (involving a pair of public and secret keys);
3. ***Hash functions*** and MAC (Message Authentication Code) algorithms used for ***Digital Signatures***.

Symmetric key encryption is much faster than public-key encryption, but public-key encryption provides better authentication techniques.

[470] The HTTPR specification draft can be found at: ftp://www6.software.ibm.com/software/developer/library/ws-httprspecv1-1.pdf

[471] A primer for HTTPR: http://www-106.ibm.com/developerworks/library/ws-phtt/

11.3.1 Secure Socket Layer (SSL) and Transport Layer Security (TLS) Protocol

The primary goal of the Secure Socket Layer (SSL) Protocol is to provide privacy and reliability between two communicating applications. Netscape Communications proposed the final version v3 in 1995. This version had been developed as a result of public discussion, became very popular and was implemented in various products.

Nevertheless, SSL has never been submitted to a standardizing body. Only Netscape's "internal" document is available[472]. Instead its descendant – Transport Layer Security (TLS) protocol - was standardized in the RFC 2246 of IETF[473]. This document and the TLS protocol itself are based on the SSL 3.0 Protocol Specification as published by Netscape. The differences between this protocol and SSL 3.0 are not dramatic, but they are significant enough that TLS 1.0 and SSL 3.0 do not interoperate (although TLS 1.0 does incorporate a mechanism by which a TLS implementation can back down to SSL 3.0).

The SSL protocol uses a combination of public-key and symmetric key encryption. SSL runs above TCP/IP and below higher-level protocols such as HTTP or IMAP as shown in Figure 100. It uses TCP/IP on behalf of the higher-level protocols, and within the process, allows an SSL-enabled server and client to authenticate one another and to establish an encrypted connection afterwards.

Optionally, to achieve even more trustworthiness an SSL can be certified by a third party certificate authority (CA).

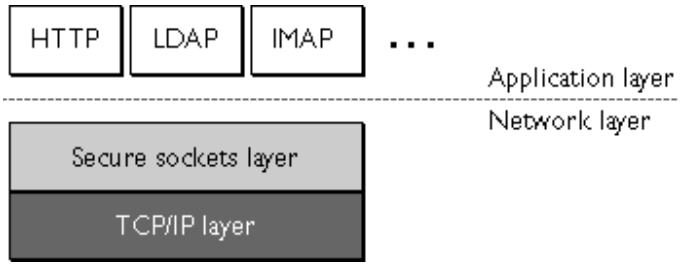


Figure 100 SSL runs above TCP/IP and below high-level application protocols

The SSL protocol includes two sub-protocols: the **SSL record protocol** and the **SSL handshake protocol**. The SSL record protocol defines the format used to transmit data. The SSL handshake protocol is used to establish an SSL connection.

A SSL session always begins with an exchange of messages called the SSL handshake. The handshake allows the server to authenticate itself to the client using public-key techniques and then symmetric session keys are being created. Optionally, the handshake can also require authentication in the other direction, from client to server.

[472] Available at: http://wp.netscape.com/eng/ssl3/draft302.txt

[473] http://www.ietf.org/rfc/rfc2246.txt

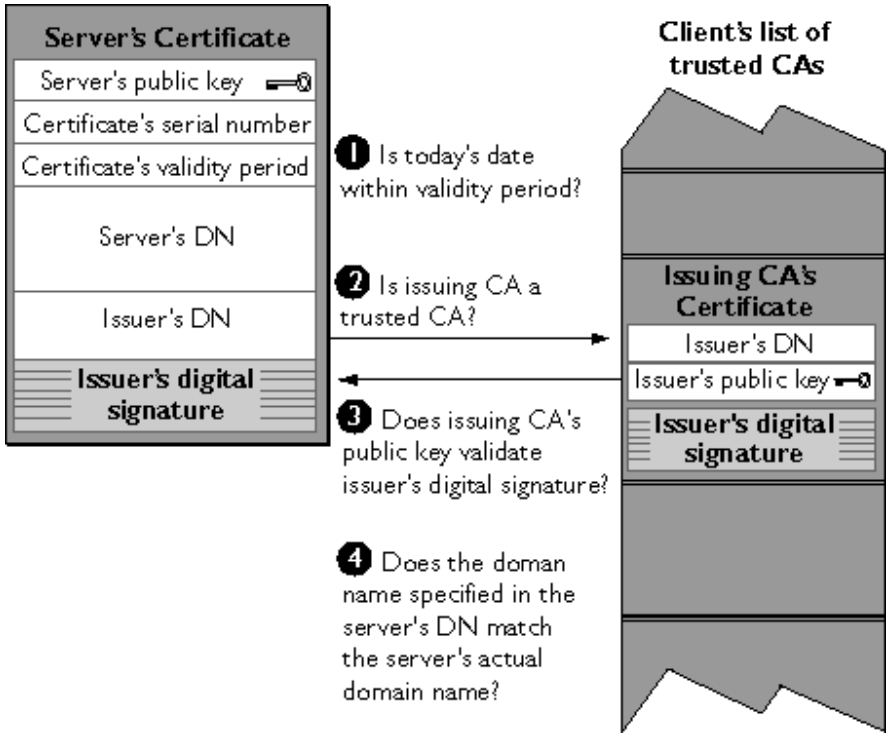


Figure 101 Client side authentication of a SSL server certificate

The SSL protocol supports the use of a variety of different cryptographic algorithms, or ciphers, for use in operations such as authenticating the server and client to each other, transmitting certificates, and establishing session keys. Clients and servers may support different cipher suites, or sets of ciphers, depending on factors such as the version of SSL they support, company policies regarding acceptable encryption strength, and government restrictions on export of SSL-enabled software. Among its other functions, the SSL handshake protocol determines how server and client negotiate with each other about which cipher suites to use in the further process.

In the following some of the ciphers that can be used, within SSL but also by themselves, will be outlined.

11.3.2 Symmetric Key Algorithms

11.3.2.1 Data Encryption Standard (DES) Algorithm

The DES is the most well-known symmetric-key block cipher. Recognized worldwide, it set a precedent in the mid 1970s as the first commercial-grade modern algorithm with openly and fully specified implementation details. In 1993, DES has been standardized in the U.S. by the FIPS 46–2 specification[474].

DES relates to the, so-called, block cipher algorithm, which parses the incoming plain text into the number of fixed length blocks, and converts each block to the appropriate block of encrypted text. The DES algorithm may be considered as a predecessor of further symmetric block ciphering such as Blowfish and IDEA.

[474] The original text of he specification: http://www.itl.nist.gov/fipspubs/fip46-2.htm

11.3.2.2 International Data Encryption Algorithm (IDEA)

IDEA is symmetric block cipher developed by the Swiss Federal Institute of Technology. This algorithm is considered as one of the emerging techniques that is going to replace DES. The widely known Pretty Good Privacy (PGP) toolkit, for example, uses this cipher.

IDEA uses 128-bit key to encrypt 64-bit blocks of initial plain text (instead of 56-bit key in the case of DES). The development goals for the algorithm were on the one hand to increase cryptographic strength and, on the other hand, to simplify program implementation.

11.3.2.3 Blowfish

Is a freely available symmetric block cipher, developed by Bruce Schneier, with many open source implementations in various programming languages[475]. The author pursued the following goals:

- **Speed** - provide speed encryption on 32-byte processors
- **Compactness** – may consume not more than 5 Kbytes of memory
- **Simplicity** – the Blowfish structure is rather simple not only for implementation, but also for cryptography strength evaluation
- **Variable key length** – the lentgth of secret key may vary from 32-bit to 448-bit, thus giving the developer the choice between cryptography strength and performance.

The algorithm converts 64-bit block of incoming plain text to 64-bit block of output ciphertext.

Mathematical details of the algorithm may be found e.g. in Chap. 4 of William Stallings book on cryptography[476].

Blowfish is only suitable for applications where the key does not change often, like a communications link or an automatic file encryptor. It is significantly faster than DES and IDEA when implemented on 32-bit microprocessors with large data caches, such as the Pentium and the PowerPC[477].

11.3.2.4 RC5 Algorithm

This algorithm was proposed by RSA Data Security, Inc. in 1996[478] RC5 is a parameterized algorithm of the pattern ***RC5(w,r,b)***. ***w*** stands for the word size in bits. Allowable values are 16, 32, and 64. ***r*** represents the number of rounds ranging between 0 and 255. Finally, ***b*** is the number of bytes in the secret key K, which again takes values between 0 and 255.

There are three routines in RC5: key expansion, encryption, and decryption. In the **key expansion** routine, the user-provided secret key is expanded to fill a key table whose size depends on the number of rounds. The key table is then used in both encryption and decryption. The **encryption** routine consists of three primitive operations: integer addition, bitwise XOR, and variable rotation. The heavy use of data-dependent rotations and the mixture of different operations provide the security of RC5. In particular, the use of data dependent rotations helps defeat differential and linear cryptanalysis.

---

[475] Published in *"Description of a New Varible-Length Key, 64-bit Block Cipher (Blowfish)", Bruce Schneier, "Fast Software Encryption", Lecture Notes in Computer Science No. 809, Springer-Verlag 1994*

[476] Cryptography and Network Security, Second Edition, by William Stallings, Prentice-Hall, 2001.

[477] The main page by the author: http://www.counterpane.com/blowfish.html , performance comparison between IDEA and Blowfish: http://ece.gmu.edu/courses/ECE543/reportsF01/jeyasankar.pdf

[478] "The RC5 Encrypion Algorithm", Ronald Rivest, "Fast Software Encryption II", Lecture Notes in Computer Science No.1008, Springer-Verlag 1995.

The technical description of RC5 is standardized and published in the RFC2040.[479]

11.3.3 Public Key Encryption

11.3.3.1 The basic concept (RSA)

The most commonly used implementations of public-key encryption are based on algorithms patented by RSA Data Security. Therefore, this section describes the RSA approach to public-key encryption.

Public-key encryption (also called **asymmetric encryption**) involves a pair of keys--a **public key** and a **private key**--associated with an entity that needs to authenticate its identity electronically or to sign or encrypt data. Each public key is published, and the corresponding private key is kept secret. Data encrypted with your public key can be decrypted only with your private key. Figure 102 shows a simplified view of the way public-key encryption works.

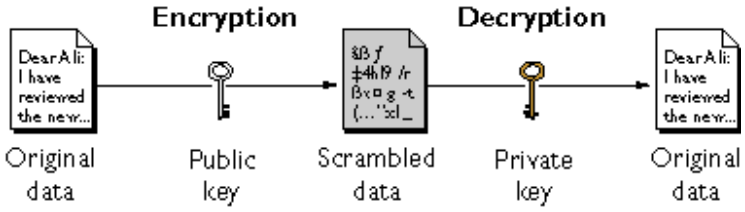


Figure 102 Public Key Encryption[480]

The scheme shown in Figure 102 lets you freely distribute a public key, and only you will be able to read data encrypted using this key. In general, to send encrypted data to someone, you encrypt the data with that person's public key, and the person receiving the encrypted data decrypts it with the corresponding private key.

Compared with symmetric key encryption, public key encryption requires more computation and is therefore not always appropriate for large amounts of data. However, it is possible to use public key encryption to send a symmetric key, which can then be used to encrypt additional data. This is the approach used by the SSL protocol discussed above.

As it happens, the reverse of the scheme shown in Figure 102 also works: data encrypted with your private key can be decrypted only with your public key. This would not be a desirable way to encrypt sensitive data, however, because it means that anyone with your public key, which is by definition published, could decrypt the data. Nevertheless, private-key encryption is useful, because it means you can use your private key to sign data with your digital signature--an important requirement for electronic commerce and other commercial applications of cryptography.

11.3.3.2 Key Length and Encryption Strength

In general, the strength of encryption is related to the difficulty of discovering the key, which in turn depends on both the cipher used and the length of the key. For example, the difficulty of discovering the key for the RSA cipher most commonly used for public-key encryption depends on the difficulty of factoring large numbers, a well-known mathematical problem.

Encryption strength is often described in terms of the size of the keys used to perform the encryption: in general, longer keys provide stronger encryption. Key length is measured in bits. For example, 128-bit keys for use with the RC5 symmetric-key cipher supported by SSL provide

[479] http://www.ietf.org/rfc/rfc2040.txt

[480] Illustration taken from: http://developer.netscape.com/docs/manuals/security/pkin/contents.htm

significantly better cryptographic protection than 40-bit keys for use with the same cipher. Different ciphers may require different key lengths to achieve the same level of encryption strength. The RSA cipher used for public-key encryption, for example, can use only a subset of all possible values for a key of a given length, due to the nature of the mathematical problem on which it is based. Other ciphers, such as those used for symmetric key encryption, can use all possible values for a key of a given length, rather than a subset of those values. Thus a 128-bit key for use with a symmetric-key encryption cipher would provide stronger encryption than a 128-bit key for use with the RSA public-key encryption cipher.

This difference explains why the RSA public-key encryption cipher must use a 512-bit key (or longer) to be considered cryptographically strong, whereas symmetric key ciphers can achieve approximately the same level of strength with a 64-bit key. Even this level of strength may be vulnerable to attacks in the near future.

Because the ability to surreptitiously intercept and decrypt encrypted information has historically been a significant military asset, the U.S. Government restricts export of cryptographic software, including most software that permits use of symmetric encryption keys longer than 40 bits.

11.3.4 Hash Functions and Digital Signatures

Cryptographic hash functions play a fundamental role in modern cryptography. While related to conventional hash functions commonly used in non-cryptographic computer applications – in both cases, larger domains are mapped to smaller ranges – they differ in several important aspects. Our focus is restricted to cryptographic hash functions (hereafter, simply hash functions), and in particular to their use for data integrity and message authentication.

Hash functions take a message as input and produce an output referred to as a hash code, hash-result, hash-value, or simply hash. More precisely, a hash function ***h*** maps bit strings of arbitrary finite length to strings of fixed length, say ***n*** bits.

The basic idea of cryptographic hash functions is that a hash-value serves as a compact representative image (sometimes called an imprint, digital fingerprint, or message digest) of an input message, and can be used as if it were uniquely identifiable with that message.

Hash functions are used for data integrity in conjunction with digital signature schemes, where for several reasons a message is typically hashed first, and then the hash-value, as a representative of the message, is signed in place of the original message.

A distinct class of hash functions, called message authentication codes (MACs), allows message authentication by symmetric techniques. MAC algorithms may be viewed as hash functions, which take two functionally distinct inputs, a message and a secret key, and produce a fixed-size (say n-bit) output, with the design intent that it be infeasible in practice to produce the same output without knowledge of the key. MACs can also be used for identification in symmetric-key schemes (see section 11.3.2).

A typical usage of (unkeyed) hash functions for data integrity is as follows. The hash value corresponding to a particular message $\mathbf{m_1}$ is computed at time $\mathbf{T_1}$. The integrity of this hash-value (but not the message itself) is protected in some manner. At a subsequent time $\mathbf{T_2}$, the following test is carried out to determine whether the message has been altered, i.e., whether a message $\mathbf{m_2}$ is the same as the original message. The hash-value of $\mathbf{m_2}$ is computed and compared to the protected hash-value; if they are equal, one accepts that the inputs are also equal, and thus that the message has not been altered. The problem of preserving the integrity of a potentially large message is thus reduced to that of a small fixed-size hash value.

Since the existence of collisions is guaranteed in many-to-one mappings, the unique association between inputs and hash-values can, at best, be in the computational sense. A hash-value should be uniquely identifiable with a single input in practice, and collisions should be computationally difficult to find (essentially never occurring in practice).

A hash function (in the unrestricted sense) is a function **h**, which has, as a minimum, the following two properties:

1. Compression - **h** maps an input **x** of arbitrary finite bit length, to an output **h(x)** of fixed bit length **n**.
2. Ease of computation - given **h** and an input **x**, **h(x)** is easy to compute.

To ensure security, the hash functions should satisfy to the following requirements:

1. **One-way property** - for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage **x** such that **h(x) = y** when given any **y** for which a corresponding input is not known.
2. **Weak collision resistance** - it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given **x**, to find another preimage **x' ≠ x** such that **h(x) = h(x')**.
3. **Strong collision resistance** - it is computationally infeasible to find any pair of distinct inputs **x**, **x'** which hash to the same output, i.e., such that **h(x) = h(x')**. (Note that here there is free choice of both inputs.)

11.3.4.1 MD4 and MD5 Algorithms

MD4 was developed by RSA in 1990. The message is padded to ensure that its length in bits plus 448 is divisible by 512 (according the specification, only first 2^{64} bits of the message are processed). A 64-bit binary representation of the original length of the message is then concatenated to the message. The message is processed in 512-bit blocks in the iterative structure, and each block is processed in three distinct rounds. Soon it was shown how collisions for the full version of MD4 can be found in under a minute on a typical PC. Clearly, MD4 should now be considered broken.

MD5 was developed by RSA in 1991. It is basically MD4 with “safety-belts” and while it is slightly slower than MD4, it is more secure. The algorithm consists of four distinct rounds, which have a slightly different design from that of MD4. Message-digest size, as well as padding requirements, remains the same. Crypto experts have found pseudo-collisions for MD5, but there are no other known cryptanalytic results

MD4 and MD5 (Message Digest) algorithms are described in RFC1320 and RFC1321 respectively and are thus open standards[481].

Both generate a unique, 128-bit cryptographic message digest value derived from the contents of a file. This value is considered to be a highly reliable fingerprint that can be used to verify the integrity of the file's contents. If as little as a single bit value in the file is modified, the MD4 and MD5 checksums for the file change. Forgery of a file in a way that causes MDx to generate the same result as that for the original file is considered to be extremely difficult.

11.3.4.2 SHA-1 Algorithm

The Secure Hash Algorithm (SHA), the algorithm specified in the Secure Hash Standard (SHS), was developed by the National Institute of Standards and Technology (NIST) and published as a Federal Information Processing Standard (FIPS).[482] SHA-1 was a revision to SHA that was published in 1994. The revision corrected an unpublished flaw in SHA. Its design is very similar to the MD4 family of hash functions developed by RSA.

The algorithm takes a message of less than 2^{64} bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5, but the larger message digest makes it more

[481] To be viewed at: http://www.ietf.org/rfc/rfc1321.txt

[482] The specification: http://www.itl.nist.gov/fipspubs/fip180-1.htm

secure against brute-force collision and inversion attacks. SHA is part of the U.S. governmental Capstone project[483].

11.3.4.3 HMAC: Keyed-Hashing for Message Authentication

This algorithm has been specified in RFC2104[484]. It is used in other Internet security protocols, e.g. SSL. Before HMAC, Message Authentification Codes (MACs) have been mostly constructed using block ciphers like DES. As hash functions like SHA-1 and MD5 are significantly faster than e.g. DES, it made sense to use these instead for message authentification. But as the hash functions mentioned above were not especially designed for message authentification, HMAC was developed to overcome gaps due to these differing design goals[485].

HMAC can be used in combination with any iterated cryptographic hash function ("black box principle"). HMAC also uses a secret key for calculation and verification of the message authentication values. The main goals behind this construction are:

1. To use, without modifications, available hash functions. In particular, hash functions that perform well in software, and for which code is freely and widely available.
2. To preserve the original performance of the hash function without incurring a significant degradation.
3. To use and handle keys in a simple way.
4. To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions on the underlying hash function.
5. To allow for easy replaceability of the underlying hash function in case that faster or more secure hash functions are found or required.

11.3.4.4 Digital Signature Algorithm (DSA) and Digital Signature Standard (DSS)

The Digital Signature Algorithm (DSA) was published by the National Institute of Standards and Technology (NIST) (see Question 146) in the Digital Signature Standard (DSS), which is also a part of the U.S. government's Capstone project. DSS was selected by NIST, in cooperation with the NSA, to be the digital authentication standard of the U.S. government. The standard was issued in 1994[486].

DSA is based on the discrete logarithm problem and derives from cryptosystems proposed by Schnorr and ElGamal. It is for authentication only. DSA requires the use of the SHA-1 algorithm, whereas RSA can optionally use MD2, MD5 or SHA.

In DSA, signature generation is faster than signature verification, whereas in RSA, signature verification is faster than signature generation (if the public and private exponents, respectively, are chosen for this property, which is the usual case). NIST claims that it is an advantage of DSA that signing is faster, but many people in cryptography think that it is better for verification to be the faster operation.

DSA has been criticized by the computer industry since its announcement. Criticism has focused on a few main issues: it lacks key exchange capability; the underlying cryptosystem is too recent and has been subject to too little scrutiny for users to be confident of its strength; verification of signatures with DSA is too slow; the existence of a second authentication standard will cause hardship to computer hardware and software vendors, who have already standardized on RSA; and

---

[483] http://www.w3.org/PICS/DSig/SHA1_1_0.html#fn1

[484] http://www.ietf.org/rfc/rfc2104.txt

[485] M. Bellare, R. Canetti, and H. Krawczyk. *RSA Laboratories' CryptoBytes* Vol. 2, No. 1, Spring 1996. online at: http://www.cs.ucsd.edu/users/mihir/papers/hmac-cb.pdf

[486] DSS specification: http://www.itl.nist.gov/fipspubs/fip186.htm

the process by which NIST chose DSA was too secretive and arbitrary, with too much influence wielded by NSA. Other criticisms were addressed by NIST by modifying the original proposal.

11.3.4.5 XML Encryption and XML Signature

An XML document, like any other, can be encrypted in its entirety and sent securely to one or more recipients. This is a common function of SSL or TLS, for example, but what is much more interesting is how to handle situations where different parts of the same document need different treatment. A valuable benefit of XML is that a complete document can be sent as one operation and then held locally, thus reducing network traffic. But this then raises the question of how to control authorized viewing of different groups of elements. A merchant may need to know a customer's name and address but doesn't need to know the various details of any credit card being used any more than the bank needs to know the details of the goods bought. A researcher may need to be prevented from seeing personal details on medical records while an administrator may need exactly those details but should be prevented from viewing medical history; a doctor or nurse, in turn, may need medical details and some, but not all, personal material[487].

As with general encryption, there's no problem in digitally signing an XML document as a whole. However, difficulty arises when parts of a document need to be signed, perhaps by different people, and when this needs to be done in conjunction with selective encryption. It may not be possible or desirable to mandate a particular sequence of sectional encryption by specified people acting in order, yet successful processing of the different parts of the document will depend on knowing this. Further, as a digital signature asserts that a certain private key has been used to authenticate something, it's prudent that a signer view the item to be signed in plain text, and this may mean decrypting part of something already encrypted for other reasons. In other cases, data that is already encrypted may be encrypted further as part of a larger set. The more the different possibilities are considered in sets of transactions involving a single XML document -- perhaps a Web form or a series of data records used in a workflow sequence, processed by a number of different applications and different users -- the more the huge potential complexity can be seen.

There are additional problems, as well. One of the strengths of XML languages is that searching is clear and unambiguous: The DTD or schema provides information as to the relevant syntax. If a document subsection, including tags, is encrypted as a whole, then the ability to search for data relevant to those tags is lost. Further, if the tags are themselves encrypted, then, being known, they may be useful as material for mounting plain text attacks against the cryptography employed.

The core element in the **XML encryption**[488] syntax is the `EncryptedData` element, which, with the `EncryptedKey` element, is used to transport encryption keys from the originator to a known recipient, and derives from the `EncryptedType` abstract type. Data to be encrypted can be arbitrary data, an XML document, an XML element, or XML element content; the result of encrypting data is an XML encryption element that contains or references the cipher data. When an element or element content is encrypted, the `EncryptedData` element replaces the element or content in the encrypted version of the XML document. When it's arbitrary data that is being encrypted, the `EncryptedData` element may become the root of a new XML document or it may become a child element. When an entire XML document is encrypted, then the `EncryptedData` element may become the root of a new document. Further, `EncryptedData` cannot be the parent or child of another `EncryptedData` element, but the actual data encrypted can be anything including existing `EncryptedData` or `EncryptedKey` elements. Figure 103 beneath shows an example.

[487] A short article on this topic: http://www-106.ibm.com/developerworks/xml/library/s-xmlsec.html/index.html

[488] XML Encryption specification: http://www.w3.org/TR/xmlenc-core

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
 <Name>John Smith<Name/>
 <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'
  xmlns='http://www.w3.org/2001/04/xmlenc#'>

<CipherData><CipherValue>A23B45C56</CipherValue></CipherData>
 </EncryptedData>
</PaymentInfo>
```

Figure 103 Credit Card data with encrypted elements using the XML encryption standard

XML signatures[489] can be applied to any arbitrary data content. Those that are applied to data within the same XML document as the signature are termed *enveloping* or *enveloped* signatures while those in which the data is external to the signature element are termed *detached* signatures. Listing 5, taken from the signature candidate recommendation document, is an instance of a simple detached signature.

[489] XML Signature specification: http://www.w3.org/TR/xmldsig-core/

```
[s01]                    <Signature                    Id="MyFirstSignature"
xmlns="http://www.w3.org/2000/09/xmldsig#">
[s02]   <SignedInfo>
[s03]   <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-
c14n-20010315"/>
[s04]     <SignatureMethod  Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-
sha1"/>
[s05]   <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
[s06]     <Transforms>
[s07]          <Transform  Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
20010315"/>
[s08]     </Transforms>
[s09]     <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
[s10]     <DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
[s11]   </Reference>
[s12] </SignedInfo>
[s13]   <SignatureValue>MC0CFFrVLtRlk=...</SignatureValue>
[s14]   <KeyInfo>
[s15a]    <KeyValue>
[s15b]      <DSAKeyValue>
[s15c]        <p>...</p><Q>...</Q><G>...</G><Y>...</Y>
[s15d]      </DSAKeyValue>
[s15e]    </KeyValue>
[s16]   </KeyInfo>
[s17] </Signature>
```

Figure 104 Example of a simple detached signature

The information that is actually signed is that between lines s02 and s12 in Figure 104, the SignedInfo element. Reference to the algorithms used in calculating the SignatureValue element is included within the signed section while that element itself is outside the signed section, on line s13. The SignatureMethod reference on line s04 is to the algorithm used to convert the canonicalized SignedInfo into the SignatureValue. It's a combination of a key-dependent algorithm and a digest algorithm, here DSA and SHA-1, possibly with other manipulation such as padding. The KeyInfo element (here lines s14 to s16 -- this element is optional) indicates the key that's used to validate the signature.

11.4 Firewall

The term firewall is not definitely clear. Some people call a piece of software, that keeps a watch on the system, a firewall. Others are talking about security infrastructure when they are talking about firewalls. From our point of view, a firewall in the context of a business scenario, and that’s what we are talking about, is a integrated concept consisting of security policy, a filter environment and security management.

11.4.1 Security policy

An existing security policy is the prerequisite for an effective use of a firewall. There is no sense in using an expensive firewall, while the employees are able to connect the Internet with other service providers. Therefore its necessary to define general security rules which every user must obey. For bigger systems it will be a really good idea to use for example one or both of the following two systematic approaches:

- Security checklists contain questions, which lead the responsible persons through the process of creating security policy. After regarding all questions, like the security level or the responsibilities for firewall revision, the result should be a consistent security policy
- Security pattern follows the well-known principle to describe standard solutions to specific common scenarios; to prevent the user from working on already solved problems. They usually contain a problem description, a possible solution and the advantages and disadvantages of that solution. The patterns are focusing on ‘what to do’ and not ‘how to do’, for example ‘use a single access point’ and not ‘use a single access point solution according technology X’. A security policy is defined by an amount of patterns.

11.4.2 Filter environment

The goal of filters is to scan received data for specific content. The treatment of the data, for example if they are dropped or accepted, depends on specific rules, which are designed by the system administrator. Obviously there are many basic decisions to make. One question is if we allow everything that is not forbidden, or if we say everything is forbidden, that is not explicitly allowed. Despite of this problem when designing the rules, there are mainly three type of filters

- **Packet filter** - A packet filter rests between the internal network and the rest of the world. Clients and servers connect directly, but the packets pass through the packet, which compares the packets to a set of filter rules. If the packet meets is allowed according to the rules, it is passed to the next destination. Otherwise the packet is discarded. Typically the comparison involves the source port, source address, destination port and destination address. So packet filter usually work on the transport layer and beneath
- **State dependent packet filter** - Those filters are very similar to packet filters, but offering a little expanded functionality, because they keep some information about the connections passing through them. The benefit for security is a wider range of attacks that can be detected. Because they know the actual state of a connection, they can apply rules, if a received request fits in the communication logic of a specific protocol, or if it tries to trigger damaging behavior
- **Application Gateway** - In contrast to packet filters, application gateways are working on the application layer. To filter the data, the application gateway is running one or more proxy-processes, which usually are able to handle a specific protocol such as HTTP. Differently from packet filters, there is no direct connection between the client and the server. The client connects to the application gateway, which connects the server after examining the data, and therefore is called a ‘proxy’. The great advantages of a proxy are the possibility to examine the payload of a request and to hide the network infrastructure

Security management

It seems to be obviously, that a security system must be managed. Unfortunately people often believe that the installation of a security system is a single action. A short look on current security pages shows the growing vulnerability problems. Therefore security systems must be actively managed. That embraces two main points:

- **Management tools.** The best security policy is worthless, if it can’t be realized because of complicate, inefficient or unsafe management tools. Therefore the decision for a specific tool must embrace points like vendor reliability and the problem of teaching the employees.
- **Management security.** Although its really clear, people often forget one point: The man who owns your security management owns your system. Therefore security management must be the safest part of your firewall, which must be defended against external but also internal abuse.

Summarizing, maintain a firewall means to combine and optimize the three mentioned points above, although even a firewall never will make a system completely secure.

11.5 Virtual Private Network (VPN)

A VPN is a type of Wide Area Network (WAN), where remote connections are set up via the Internet instead of hiring expensive and more rigid dedicated lines. To maintain security in terms of privacy, the messages between the participating computers are sent through a tunnel in an encrypted format as shown in Figure 105.

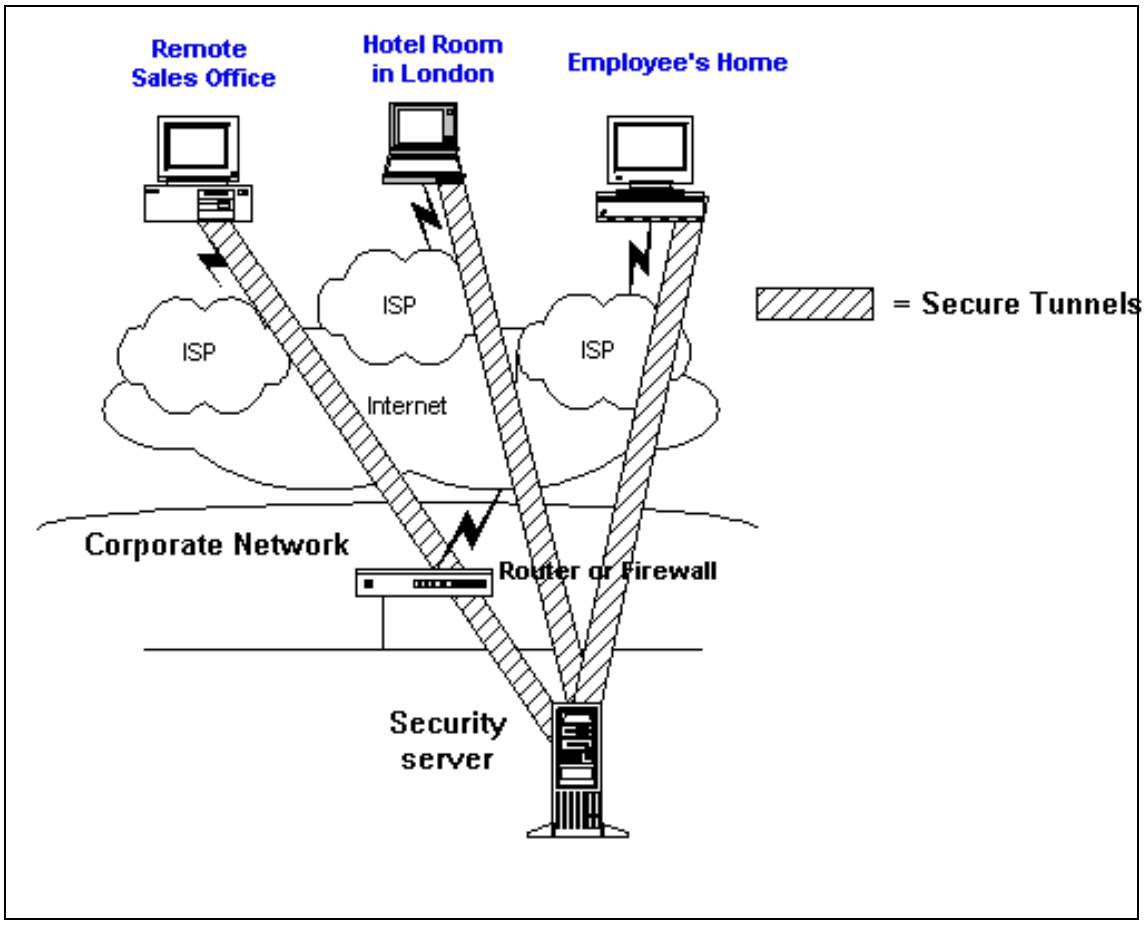


Figure 105 A VPN is established through tunnelling protocols

Also known as the Layer-2 Tunneling Protocol, L2TP is the combination of Cisco Systems' Layer-2 Forwarding (L2F) and Microsoft's Point-to-Point Tunneling Protocol (PPTP). L2TP supports any routed protocol, including IP, IPX, and AppleTalk. It also supports any WAN backbone technology, including frame relay, ATM, X.25, and SONET.

One key to L2TP is its use of PPTP. This Microsoft protocol is an extension of PPP and is included as part of the remote access features of Windows 95, Windows 98, and Windows NT. So, in the big picture, most PC clients come equipped with tunneling functionality. PPTP provides a consistent way to encapsulate Network-layer traffic for remote access transmission between Windows clients and servers. The protocol doesn't specify a particular encryption scheme, but the remote access

functions included in the Microsoft portfolio of operating systems are supplied with Microsoft Point-to-Point Encryption (MPPE).

The L2F portion of L2TP lets remote clients connect and authenticate to networks over ISP links. Besides the basic VPN capability, L2TP can create multiple tunnels from a single client. In practice, a remote client can create tunneled connections to various systems simultaneously - for instance, to a corporate database application and to the company's intranet.

The Internet Protocol Security, IPsec in short, is a suite of protocols that provide security features for IP VPNs. As a layer-3 function, IPsec can't perform services for other layer-3 protocols, such as IPX or SNA. IPsec provides a means of ensuring the confidentiality and authenticity of IP packets. The protocol works with a variety of standard encryption schemes and encryption negotiation processes, as well as with various security systems, including digital signatures, digital certificates, public key infrastructures, and certificate authorities.

IPsec works by encapsulating the original IP data packet into a new IP packet that's fitted with authentication and security headers. The headers contain the information needed by the remote end, which took part in the security negotiation process to authenticate and decrypt the data contained in the package.

What makes IPsec attractive is its adaptability to different underlying protocols. It doesn't specify a proprietary way to perform authentication and encryption. Instead, it works with many systems and standards. IPsec can complement other VPN protocols. For instance, IPsec can perform the encryption negotiation and authentication, while an L2TP VPN receives the internal data packet, initiates the tunnel, and passes the encapsulated packet to the other VPN end point.

11.6 Authentication Systems & PKI

11.6.1 Kerberos

Kerberos is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol is available from the Massachusetts Institute of Technology (MIT). Kerberos is available in many commercial products as well (e.g. Windows 2000).

It allows a process (a client) running on behalf of a principal (a user) to prove its identity to a verifier (an application server, or just server) without sending data across the network that might allow an attacker or the verifier to subsequently impersonate the principal. Kerberos optionally provides integrity and confidentiality for data sent between the client and server.

To be useful, Kerberos must be integrated with other parts of the system. It does not protect all messages sent between two computers; it only protects the messages from software that has been written or modified to use it. Kerberos does not itself provide authorization, but V5 Kerberos passes authorization information generated by other services. In this manner, Kerberos can be used as a base for building separate distributed authorization services.

Encryption in the present implementation of Kerberos uses the data encryption standard (DES). It is a property of DES that if ciphertext (encrypted data) is decrypted with the same key used to encrypt it; the plaintext (original data) appears. If different encryption keys are used for encryption and decryption, or if the ciphertext is modified, the result will be unintelligible, and the checksum in the Kerberos message will not match the data. This combination of encryption and the checksum provides integrity and confidentiality for encrypted Kerberos messages.

11.6.1.1 The Kerberos Ticket

The client and server do not initially share an encryption key. Whenever a client authenticates itself to a new verifier it relies on the authentication server to generate a new encryption key and distribute

it securely to both parties. This new encryption key is called a **session key** and the Kerberos ticket is used to distribute it to the verifier.

The Kerberos ticket is a certificate issued by an authentication server, encrypted using the server key. Among other information, the ticket contains the random session key that will be used for authentication of the principal to the verifier, the name of the principal to whom the session key was issued, and an expiration time after which the session key is no longer valid. The ticket is not sent directly to the verifier, but is instead sent to the client who forwards it to the verifier as part of the application request. Because the ticket is encrypted in the server key, known only by the authentication server and intended verifier, it is not possible for the client to modify the ticket without detection.

The Figure 106 shows the simplified messages required for a client to prove its identity to a server **KDC** (Key Distribution Center) also known as **AS** (Authentication Server). A typical client application (**C**) uses this exchange when it first establishes a connection to a server. Subsequent connections to the same server require only the final message in the exchange (client caching eliminates the need for the first two messages until the ticket expires).

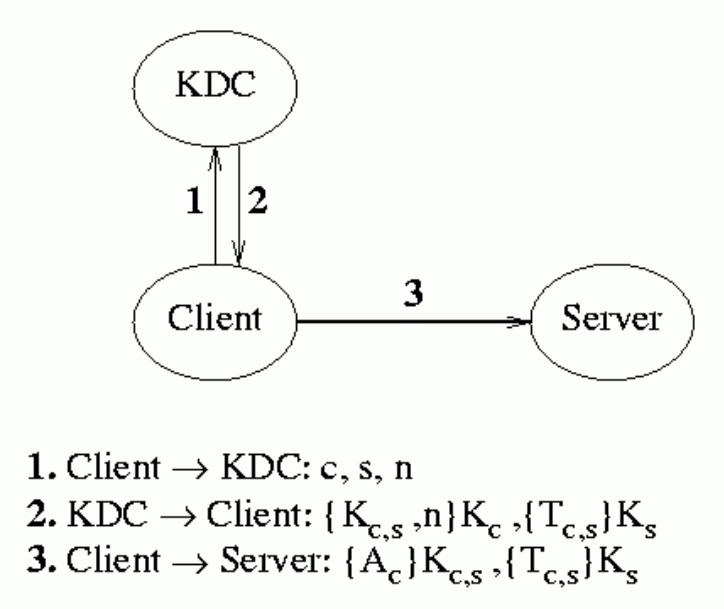


Figure 106 Basic Kerberos authentication protocol (simplified)

In the first message the client contacts the KDC, identifies itself (***c***), presents a nonce ***n*** (a timestamp or other unique identifier for the request), and requests credentials for use with a particular server.

Upon receipt of the message the KDC selects a random encryption key $K_{c,s}$, called the session key, and generates the requested ticket $T_{c,s}$. The ticket identifies the client, specifies the session key $K_{c,s}$, lists the start and expiration times, and is encrypted in the key K_s shared by the KDC and the server. Because the ticket is encrypted in a key known only by the KDC and the server, nobody else can read it or change the identity of the client specified within it. The KDC next assembles a response, the second message, which it sends to the client. The response includes the session key, the nonce, and the ticket. The session key and nonce are encrypted with the client's secret key K_c (in Version 4 all fields are encrypted in K_c).

Upon receiving the response the client decrypts it using its secret key (usually derived from a password). After checking the nonce, the client caches the ticket and associated session key for future use.

In the third message the client presents the ticket and a freshly generated authenticator A_C to the server. The authenticator contains a timestamp and is encrypted in the session key $K_{C,S}$. Upon receipt the server decrypts the ticket using the key K_S it shares with the KDC (this key is kept in secure storage on the server's host) and extracts the identity of the client and the session key $K_{C,S}$ To verify the identity of the client, the sever decrypts the authenticator (using the session key $K_{C,S}$ from the ticket) and verifies that the timestamp is current.

Successful verification of the authenticator proves that the client possesses the session key $K_{C,S}$, which it only could have obtained if it were able to decrypt the response from the KDC. Since the response from the KDC was encrypted in K_C, the key of the user named in the ticket, the server may reasonably be assured that identity of the client is in fact the principal named in the ticket.

If the client requests mutual authentication from the server, the server responds with a fresh message encrypted using the session key. This proves to the client that the server possesses the session key, which it could only have obtained if it was able to decrypt the ticket. Since the ticket is encrypted in a key known only by the KDC and the server, the response proves the identity of the server.

11.6.1.2 Kerberos Infrastructure and Cross-Realm Authentication

In a distributed system that crosses organizational boundaries, it is not appropriate for all users to be registered with a single authentication server. Instead, multiple authentication servers will exist, each responsible for a subset of the users or servers in the system. The subset of the users and servers registered with a particular authentication server is called a realm. **Cross-realm authentication** allows a principal to prove its identity to a server registered in a different realm.

To prove its identity to a server in a remote realm, a Kerberos principal obtains a ticket granting ticket for the remote realm from its local authentication server. This requires the principals's local authentication server to share a cross-realm key with the verifier's authentication server. The principal next uses the ticket granting exchange to request a ticket for the verifier from the verifier's authentication server, which detects that the ticket granting ticket was issued in a foreign realm, looks up the cross-realm key, verifies the validity of ticket granting ticket, and issues a ticket and session key to the client. The name of the client, embedded in the ticket, includes the name of the realm in which the client was registered. Since version 5, its also possible to contact a realm without registering the user, provided that the target realm trusts a realm who trusts the source realm, even if there some other realms between them.

The list of realms that are transited during multi-hop cross-realm authentication is recorded in the ticket and the verifier accepting the authentication makes the final determination about whether the path that was followed should be trusted.

Microsoft's implementation of the Kerberos protocol supports the interoperability characteristics sufficient for identity-based authentication. In addition, Microsoft integrates authorization data in the form of Windows NT group memberships in Kerberos tickets to convey access control information to Windows NT services. The native representation of the authorization data is the Windows NT Security IDs.

11.6.1.3 Public Key Infrastructure and X.509 Authentication Service

X.509 is the authentication framework designed to support X.500 directory services. Both X.509 and X.500 are part of the X series of international standards proposed by the ISO and ITU. The X.500 standard is designed to provide directory services on large computer networks. X.509 provides a PKI framework for authenticating X.500 services (RFC 2510).

The X.509 standard may be considered as an implementation of a Public Key Infrastructure (PKI).

11.6.2 Public Key Infrastructure

In its most simple form, a PKI is a system for publishing the public-key values used in public-key cryptography. There are two basic operations common to all PKIs:

- **Certification** is the process of binding a public-key value to an individual, organization or other entity, or even to some other piece of information, such as a permission or credential;
- **Validation** is the process of verifying that a certification is still valid.

At its most basic, a certificate is merely a public key value. In more traditional terms, a certificate is a collection of information that has been digitally signed by its issuer, like the example in Figure 107. Such certificates are distinguished by the kind of information they contain.

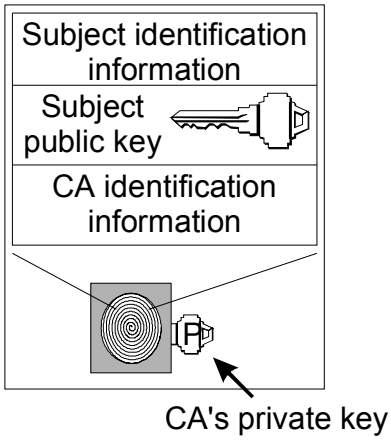


Figure 107 A basic certificate.

An **identity certificate** simply identifies an entity, called the **certificate subject**, and lists the public-key value(s) for that entity. A **credential certificate** describes non-entities, such as a permission or credential.

A **certificate user** is an entity that relies upon the information contained in a certificate. The certificate user trusts the issuing authority to issue “true” certificates. That is, certificates which truly identify the subject and its public key (in the case of identity certificates), or which truly describe a subject’s credentials (in the case of credential certificates). The certificate issuer is commonly called a **certification authority** (**CA**).

The information contained in a certificate is a basic characteristic of different PKIs. As well, the relationship between the CA, the certificate user and the certificate subject forms another basic PKI characteristic. All three may be distinct entities, or any two (or all three) can be the same entity. The trust relationships between the three also form a third basic PKI characteristic.

11.6.2.1 CA organization

It is obviously impractical to have a single CA act as the authority for the entire world. Therefore, most PKIs permit CAs to certify other CAs. In effect one CA is telling its users that they can trust what a second CA says in its certificates

How the CAs of a PKI are arranged is a basic PKI characteristic. Some PKIs use a general hierarchy, illustrated in Figure 108. In a general hierarchy, each CA certifies its parent and its children. Also shown in Figure 108 are some cross-certificates, indicated by the dashed arrows, which are certificates that do not follow the basic hierarchy.

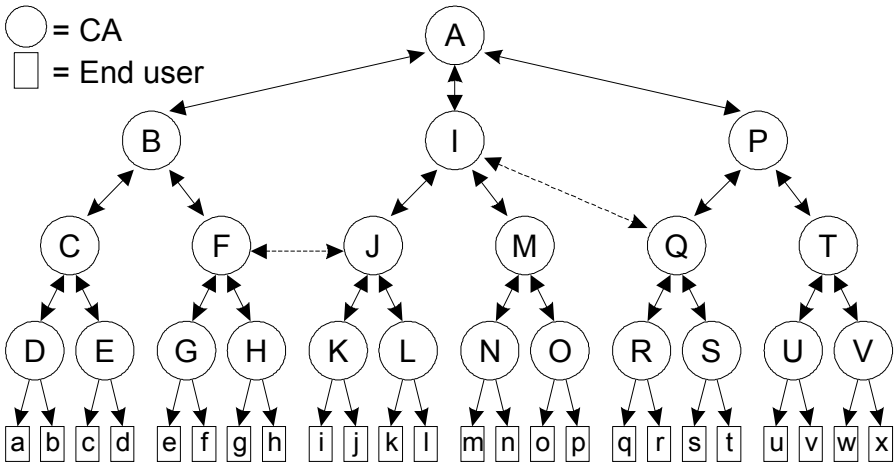


Figure 108 A general CAs hierarchy with cross-certificates

11.6.2.2 Validation

The second basic PKI operation is certificate validation. The information in a certificate can change over time. A certificate user needs to be sure that the certificate's data is true – we say that the user needs to validate the certificate. There are two basic methods of certificate validation:

- The user can ask the CA directly about a certificate's validity every time it is used. This is known as **online validation**.
- The CA can include a validity period in the certificate – a pair of dates that define a range during which the information in the certificate can be considered as valid. This is known as **offline validation**.

A PKI can use either or both methods. How a certificate user validates certificates is a basic PKI characteristic.

Closely related to the validation method is certificate revocation. Certificate revocation is the process of letting users know when the information in a certificate becomes unexpectedly invalid. This can occur when a subject's private key becomes compromised, or, more benignly, when a certificate's identifying information changes.

11.6.2.3 X.509

A full understanding of X.509 PKIs requires some basic knowledge of the X.500 directory that X.509 was originally designed for. An entry in an X.500 directory can contain a host of attributes, such as the name of the organization the person works for, her job title and her email address as shown in Figure 109. An X.500 directory entry can rep-resent any real-world entity, not just people but also computers, printers, companies, governments, and nations. The entry can also contain the certificate specifying the entity's public key.

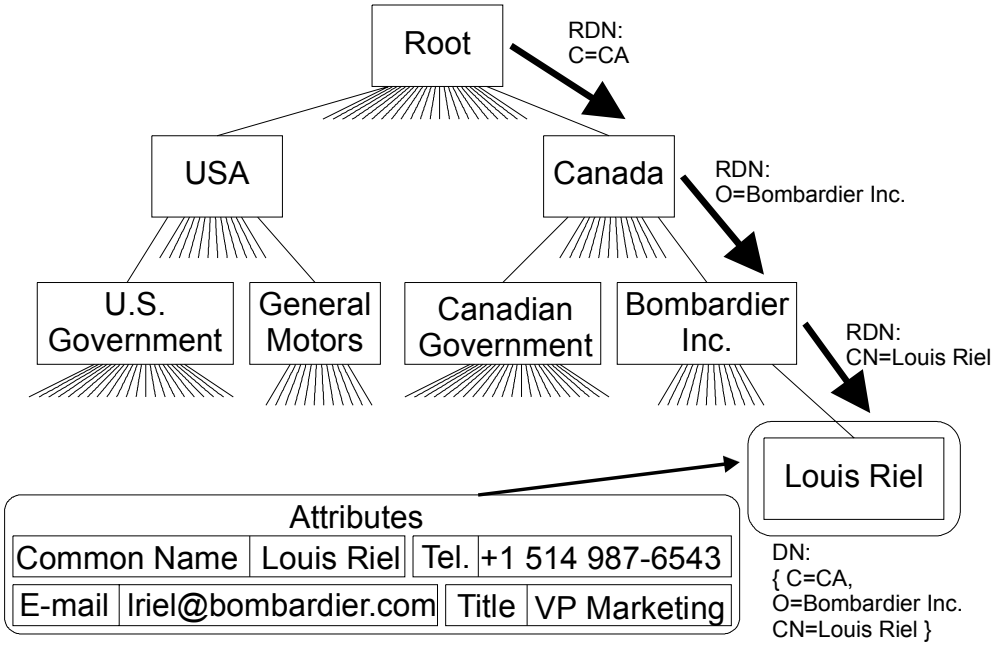


Figure 109 The X.500 Directory Information Tree[490]

To support looking up entries in the directory, each entry is assigned a globally unique name, called a distinguished name (DN). This is done with a hierarchic tree. Each node, or vertex, in the tree has one parent (except the root vertex) and any number of children. Each vertex, except the root, is assigned a relative distinguished name (RDN) that is unique amongst all the vertex's siblings. The RDNs of each of the vertex's ancestors are concatenated with the vertex's own RDN to form the unique DN.

An X.509 v3 certificate binds a distinguished name (DN) to a public key. DNs may include a variety of other name-value pairs. They are used to identify both certificate subjects and entries in directories that support the Lightweight Directory Access Protocol (LDAP).

The rules governing the construction of DNs can be quite complex and are beyond the scope of this document. For comprehensive information about DNs, see RFC1485 ("A String Representation of Distinguished Names").

Every X.509 certificate consists of two sections:

- **Data section**. The data section includes informations like version number of the X.509 standard supported by the certificate, the certificate's serial number which is unique among the certificates issued by that CA, information about the user's public key, including the algorithm used and several others --that is, whether it is a client SSL certificate, a server SSL certificate, a certificate for signing email, and so on. Certificate extensions can also be used for a variety of other purposes.
- **Signature section.** The signature section contains the cryptographic algorithm used by the issuing CA and the CA's digital signature, obtained by hashing all of the data in the certificate together and encrypting it with the CA's private key

Despite the possibility of creating a certificate by your own, which mainly is useful for test issues, its necessary to contact a Certification Authority[491].

[490] Figure taken from www.xcert.com

[491] A list of such companies can be found on http://www.pki-page.org.

11.7 Intrusion detection and Intrusion response systems

Firewalls and encryption can't guarantee entire security of the system. Therefore its helpful to scan for attempts to hack into a system and obviously for already running attacks. Intrusion detection systems IDS and Intrusion response system IRS are both used to identify such events. The difference is only, that IRS try to react automatically according to rules made by a system administrator. Such rules may vary in the range of writing log files and closing ports.

To identify illegal system access, there are two main approaches

- **Anomalous detection**. This approach tries to find significant deviation from normal user behavior. Therefore it needs a description of normal user behavior, something that might be difficult to find. In a simple scenario we may use a rule 'works from 9.00 a.m. to 5.00 p.m., not Saturday, not Sunday'. That means working after 5.00 p.m. will give alarm, while hacking is allowed in the given time frame. Obviously existing systems are a little bit smarter, but the main problem remains
- **Signature analysis**. Signature analysis tries to finding significant signatures of requests or system activities, which are related to known intrusions. For example, a common first step for an intruder is port scanning, to search for running services. In this case there might be a signature: Source IP scanning port 1, Source IP scanning port 2, and so on. Certainly the problems of this approach are clear. First we need to have as much comparing signatures of known intrusions as possible, and second the intruder must be friendly enough to use a standard attack. In our example we should also consider more clever attacks, which scan the ports by accident. So it reminds of the problems of heuristic virus scan methods.

Another way to classify intrusion systems is to make a distinction between host-based and network-based IDS. Host-based ID are loading a piece of software on the system to be monitored. The loaded software is able to create log-files or to provide snapshots from current system status. In contrast, a network-based ID system monitors the traffic on its network. Both have pros and cons. A host-based system can't detect an attack if its dangerous for the network, but not for an individual host. On the other hand a network-based IP is dependent on specific network traffic and therefore can't identify who received illegal access to a host system.

Regarding the growing amount of known security holes and attack tools, IDS and IDR are becoming more and more necessary to manage system security. Nevertheless it will be always difficult to define things like usual behavior or suitable system reactions. Therefore IDS and IDR are parts, but not the finally solution. As often, probably a combination of all approaches above are the best thing we can do.

12 ANNEX II: Summary of VO definitions and related concepts

12.1 Virtual Organization

12.1.1.1 Definitions:

The more a company or a group of companies is able to provide its products and services independent of its location and independent of any time restrictions, the more successful it might be. Information technology is one of the most important enablers to support location and time independency. Therefore I define a Virtual Organization as any institutionalized form of the ability to provide its products and services more time and location independent than its competitors.

Published in: Sieber, P.: Virtuelle Unternehmen in der IT-Branche, die Wechselwirkung zwischen Internet-Nutzung, Organisation und Strategie, Berner betriebswirtschafliche Schriften, Band 19, Bern, Stuttgart, Wien, Verlag Paul Haupt, 1998, p. 258. (ISBN: 3-258-05872-5)

A Virtual Organisation is primarily characterised as being a network of independent, geographically dispersed organisations with a partial mission overlap. Within the network, all partners provide their own core competencies and the co-operation is based on semi-stable relations. The products and services provided by a Virtual Organisation are dependent on innovation and are strongly customer-based.

Further, a Virtual Organisation is secondarily characterised by a single identity with loyalty being shared among the partners and the co-operation based on trust and information technology. In addition, there is also a clear distinction between a strategic and an operational level.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Newsletter, 2 (1998) 3, p. 16.

Virtual Organizations are defined and identified by its characteristics. The concept of 'Virtual Organization' includes all organizations that strongly exhibit any of the following characteristics:

- **Dispersion:** multiple locations, which may give rise to multiple local cultures, multiple languages, and problems of communication.
- **Empowerment:** the devolution of powers, responsibilities and accountabilities to the component parts and individuals of the organization.
- **Restlessness:** acceptance, even enthusiasm, for change to every aspect of the organization's operations: its scope, its organizational and geographical configuration, its products and services and its way of doing business.
- **Interdependence:** cooperation and synergy between autonomous organizations, or between empowered departments or individuals within a single organization. Interdependence may be temporary or permanent. It includes all types of associations: alliances, partnerships, value chains and outsourcing.

For French-speaking readers, these key words may be rendered as dispersion, responsabilisation, proactivité et interdépendence. For German-speaking readers, they are Zerstreuung, Ermächtigung, Rastlosigkeit und gegenseitige Abhängigkeit.

Published in: The IMPACT Programme (12 Sept. 1998), Exploiting the Wired-Up World – Best Practice in Managing Virtual Organizations, The Report of Working Group 4 of Project ACHIEVE, http://www.achieve.ch/

The virtual organisation is a system whereby organisations end up with more capabilities and power than they inherently possess.

Published in: Christie P.M.J., Levary R.R., Virtual Corporations: Recipe for Success, Industrial Management, Vol. 40, No. 4, 1998, p.7-11.

A framework of virtual organisation variables to classify and analyse virtual organisations:

- Connectivity - the creation of unity or linkage through structural change, breaking of constraints or overcoming of previously existing barriers.
- Purpose - the objective that provide the incentive for creating the new organisation and which serves as the cohesive force to hold the virtual organisation components at least temporarily together.
- Technology - the enabling factor that allow the break-through and makes the virtual form possible.
- Boundary - the separation of those who are part of the virtual organisation and those who are not, in the absence of any clear visible physical border lines. It defines who can share its activities and who receives benefits.

Published in: Shao Y.P., Liao S.Y., Wang H.Q., A model of virtual organisations, Journal of Information Science, Vol. 24, No. 5, 1998, p.305-312

The essence of Virtual Organisations is the metamanagement of goal-oriented activity in a way that is independent of the means for its realisation. Metamanagement is the management of a virtually organised activity. Such a virtually organised activity contains a set of requirements, a set of satisfiers and a procedure to map the satisfiers on the requirements.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter @ http://www.virtual-organization.net/, 2 (1998) 3, p. 9.

A Virtual Organisation is a combination of various parties (persons and/or organisations) located over a wide geographical area which are committed to achieving a collective goal by pooling their core competencies and resources. The partners in a Virtual Organisation enjoy equal status and are dependent upon electronic connections (ICT infrastructure) for the co-ordination of their activities.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter @ http://www.virtual-organization.net/, 2 (1998) 3, p. 9.

A Virtual Organisation is an organisation network, which is structured and managed in such a way that it operates vis à vis customers and other external stakeholders as an identifiable and complete organisation.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter @ http://www.virtual-organization.net/, 2 (1998) 3, p. 8.

Virtual organisations in short.

- Co-operation of several legally independent company units.
- Reciprocal supplement regarding the common performance in the sense of vertical integration.
- Quick formation of the co-operation through avoidance of a normally necessary overhead, thus renunciation of new hierarchies.
- Comprehensive utilisation of communication and information technology for the informational integration of units.
- (translated from German by Ulrich Franke)

Published in: Mews M.: Virtuelle Unternehmen zwischen Anspruch und Wirklichkeit, in: IT Management, 3 (1997), p.12-17.

A virtual organisation is based on a network of companies, which unite quickly (configuration and alternation of it) in order to exploit an apparent chance to compete. In a virtual organisation partners share costs, risks and knowledge. They act together in national and global markets whereby each 'player' contributes its 'comparative advantages'. A critical success factor is a sophisticated information infrastructure which connects the dispersed member companies over large distances. If a market task is accomplished, whether it be after a year or a century, the organisational structure disbands respectively make space for new alliances. (translated from German by Ulrich Franke)

Published in: Mertens P., Faisst W.: Virtuelle Unternehmen - Eine Organisationsform für die Zukunft?, in: Das Wirtschaftsstudium, 6 (1996), p.280-285.

A virtual organisation is a co-operative form of legal independent companies, institutions and / or individuals, which deliver a performance, based on a common business understanding. The co-operating units contribute foremost their core competencies to the horizontal and / or vertical co-operation and during its performance it appears externally as a single unit. By doing this it almost neglects the institutionalisation of central functions. The necessary co-ordination is realised by suitable information and communication systems. The virtual organisation exists until the business purpose has been achieved or has become obsolete. (translated from German by Ulrich Franke)

Published in: Arnold O. et al: Virtuelle Unternehmen als Unternehmenstyp der Zukunft?, in: HMD 32, 185 (1995), p.8-23.

Definition and features: Virtual organisations are a specific form of co-operation of two or several companies. Each company concentrates on those segments of the value chain with which it achieves the maximum value added contribution. A virtual organisation has

- No common legal framework and do not share a joint administration.

Pre-conditions: the conditions for the operation of a virtual organisation are:

- Sophisticated information technology which connects the individual units,
- Total mutual trust of the participants,
- Existence of individual core competencies,
- Ability to combine synergetic core competencies, having no apparent competition.

The ability to self-organise will be one of the central pre-conditions of virtual organisations: that means, that virtual organisations and its members organise themselves without any instructions (directions), whereby they can take also over to role of others, to live with paradoxes and uncertainty, to question themselves and foremost to try to optimise the interplay between value creation and self-organisation (Meta-Organisation).

Benefits: The advantages of the virtual organisation are, according to the protagonists:

- Flexibility and adaptivity as well as
- Utilisation of common synergetic potentials.

Published in: Scholz Ch.: Das virtuelle Unternehmen - Schlagwort oder echte Vision?, in: Manager Bilanz, 1 (1997), p.12-19.

VO refers to a temporary or permanent collection of geographically dispersed individuals, groups, organizational units -- which do or do not belong to the same organization -- or entire organizations that depend on electronic linking in order to complete the production process.

Published in: Travica, B.: The Design of the Virtual Organization: A Research Model, in: Proceedings of the Association for Information Systems 1997 Americas conference, Indianapolis, 15.-17. August 1997,

12.1.1.2 Attributes:

Based on Core competencies Partners will only contribute to the VO with their core competencies. The partners in the VO or the initiating company determine hereby the necessary business processes. The combination of all core competencies leads to synergy and enables a flexible way of meeting the customer demands. Excellence is important, because every partner brings in its core competence, it's possible to create a 'best-of-everything' organisation. Every function and process should be of world class.

Network of independent organisations Van Aken assigns the VO to the organisation networks. He defines a organisation network as a set of independent organisations connected by semi-stable relations. Byrne also states that a VO is a temporary network of independent companies.

One identity Another base characteristic of a VO is, that it must have its own identity. Besides the identity of the VO, the identity of the partners can also remain visible. These VOs are called 'Soft VOs'. A 'Hard VO' looks from the outside like one common organisation.

Based on Information technology Information technology is a key factor in the spread of VOs. Important for a VO are the advances in transportation, communication and computing. An information network will help widespread companies to link up and work together. The vision on VOs of Davidow is strongly based on computer-based information technology. Jägers states that IT is essential for a VO, though other authors call IT an enabler.

No hierarchy There exists no hierarchy in a VO, because of the equality of the partners. Byrne also states that there is no hierarchy within a VO. Sieber calls this the egalitarian structure of a VO. This enhances the efficiency and the responsiveness and decreases the overhead.

Distinction between a strategical and operational level (~ separability) On a managerial level exists a clear distinction between the abstract requirements and the concrete implementation to reach the organisational goals, which is called switching. Van Aken also makes the difference between a global strategic management level and a local operational management level. This is for coping with the difficult control problem.

Small sized partners: Small companies and/or parts of large companies Like mentioned before, the partners will only bring in their core competencies and this is often not the whole company. Furthermore, flexibility and fast moving is necessary for going after opportunities. Only small companies or parts of large companies can achieve this. Large companies are often slow in, for example, decision making and innovation, what is essential for responding to the opportunities.

Vague/fluid boundaries The VO redefines the traditional boundaries of organisations. More co-operation among competitors, customers, suppliers, designers, etc. makes it difficult to determine where one organisations begins and another end. Mowshowitz also distinguishes internal (between units e.g. partners) and external (between the VO and the outside world) boundaries.

Semi-stable relations The relations between the partners in a VO are less formal and less permanent. These relations create dependencies among the partners, but the partners can also survive without them.

Dependent on opportunism Companies will band together to meet a specific market opportunity and are most likely to fall apart once the need evaporates. The only reason that preserves the co-operation is a specific opportunity.

Shared risks VOs respond to opportunities in the market. As market-based incentives become greater, the risk-taking will increase. The risks have to be shared by every partner in the VO. An example of a risk is losing control when functions are contracted out to other partners.

Based on trust Byrne speaks of co-destiny, which means that the fate of each partner is dependent on the fate of other partners. The semi-stable relations (less formal and less permanent) and the shared risks make the partners also more dependent. And because of the sharing of information and knowledge, there must be a high amount of trust among the partners.

Shared ownership A VO lacks the 'classical trinity', which is important for an effective and efficient co-operation. Shared ownership means in this case that every independent partner has its own interests in the VO and parts of the VO can have different owners. So not only the VO has its interests but also the participating partners. When the goal of a partner is met or can't be met, it will/can step out of the VO.

Shared leadership Shared leadership means that every partner controls its own resources but not automatically the resources of the whole VO.

Shared loyalty The employees of every partner in the VO must identify themselves with the VO but also with their own company. Culture is strongly related to loyalty among the employees. People determine the success of a VO.

Dynamic network A VO is a dynamic network of co-operating organisations. The organisations can enter and leave the network at any time.

Dependent on innovation Essential for a VO are the marked-based incentives and the corresponding responsiveness. To react in an adequate way, innovative products or services are necessary. This not only includes technical but also cultural innovation. The innovation is dependent on the extent of mass-customisation and organisational learning.

Geographical dispersed Jägers states that a VO is characterised by a geographically dispersed structure.

No organisation chart & meta-organisation The VO is a network of all sorts of organisational structures. That's why it is difficult to draw an organisation chart. Mowshowitz states that a VO does not presuppose any particular form of organisation. He compares a VO with a shell or meta-organisation or a kind of umbrella organisation.

Customer based & mass-customisation Customers have particular needs and wishes, and ask for individual products, which is specified as mass-customisation. Organisations are collaborating in a VO to produce this mass-customisation. Davidow makes this concrete with the term Virtual Products. Strong customer interaction is essential for the development of a virtual product.

Lifespan of co-operation: temporary vs. permanent Byrne states that a VO is a temporary network of independent companies. They quickly unite, exploit an opportunity and disband afterwards. Ten Have also says that a VO exists temporary. According to Van Aken and Jägers VOs can exist temporally but also on a long-lasting base. (The subsistence is dependent on the customer demand and the need to collaborate.) Because of that, the concepts Project (temporary) vs. Program (long-lasting) are introduced.

Balance of power: equality of partners vs. core-partners The high dependency between partners within a VO leads to equal relations between these partners. The culture of control is replaced by a culture of information and knowledge sharing. Van Aken, on the contrary, makes a distinction between a VO with and without a core-partner. A core-partner is some sort of 'leader' of a VO to which the other partners have to comply. Ten Have agrees with this core-partner-principle.

Mission-overlap: Partial vs. Complete There are VOs with a partial mission-overlap and VOs with a complete mission-overlap. Partners that are also doing business outside the context of the VO have partial mission-overlap. With a complete mission-overlap all business is done within the organisational context.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter @ http://www.virtual-organization.net/, 2 (1998) 3, p. 9-12.

12.1.1.3 Dynamic networks –vs- organisation

The virtual organisation appears through its achieved performance to be a corporation or an organisation. Actually it is an organisational collective. More precisely it is a network of legally independent, and to varying degrees economically independent corporations. They unite in order to generate a certain economical performance by using information technology. The individual (network) corporations contribute certain resources to the virtual organisation which are mainly derived from their core competencies. If a virtual organisation, as it is often claimed, is a temporary co-operation, this corporation network complies with the typology of 'dynamic networks' (Miles & Snow, 1986). However, the virtual organisation has at least three extreme aspects of an organisational network:

- Information technological networks, more precisely the use of an inter-organisational information system (IOS), is the basis for cross company co-operations.
- Customised production and distribution of goods and services takes place ideally at the time of demand.
- The co-operation of corporations is not visible for customers.

For the economical performance the virtual organisation requires a functional efficient corporation network. The dynamic network is embedded in a larger network of corporations, from which certain members are recruited to deliver demanded performances.

Published in: Sydow J.: Erfolg als Vertrauensorganisation?, in: Office Management, 7-8 (1996), p10-13.

12.1.1.4 Further Classifications:

- **Internal VO** This sort of VO applies to one organisation which aims at operating with internal teams. Such a VO consists of several business units, which exist again of autonomous groups and teams. The management tasks are often carried out decentralised by the autonomous teams. The employees are available on many different places, which is the reason for the flexible structure of the organisation.
- **Stable VO** This sort of VO is based on the co-operation between different organisations and aims at contracting non core-competencies out by a 'main'-organisation (often a core partner). These non core-competencies are contracted out to the several committed suppliers, which are closely related to the 'main'-organisation.
- **Dynamic VO** A dynamic VO co-operates on a large scale basis with other organisations. The relations with these other organisations are based on opportunism and are always temporary. So the co-operation takes place when certain market incentives occur. This way of organising offers a great deal of flexibility.
- **Web-company** The web-company, often called 'agile organisation', is a temporary network of specialised organisations based on the use of the Internet. It aims at globally offering all sorts of products and services by using the Internet. Knowledge management and knowledge sharing between the co-operating partners is essential for a good functioning of the VO.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter @ http://www.virtual-organization.net/, 2 (1998) 3, p. 17.

Respondents identified the scope of the work, the projected length of time spent in virtual work, types of projects, the range of involvement and the number of personnel involved. These criteria suggested four distinct virtual organizational types: permanent virtual organizations, virtual teams, virtual projects, and temporary virtual organizations.

Permanent Virtual Organizations: This virtual organization was designed, from its inception, as a virtual organization to bring together market players and respond to opportunities for both improved revenue-generating activities as well as cost savings. This is a model which involves the virtual concept in all operations, including virtual tasks, teams, and management of the organization's activities.

Virtual Teams: Internal organizational use of the virtual concept has generated virtual teams in a variety of organizations. In most cases these teams come from a specific functional, process or strategic business unit within a larger organization. The organizational use of the virtual concept in this instance is in virtual tasks and virtual teams (Greiner and Metes, 1996).

Virtual Projects: A third incarnation of the virtual organization is the virtual project. In this design, organizations form alliances or consortia to bring complementary organizations together in meeting market opportunities. Alliances formed call on manufacturers, developers, and markets from a variety of organizations to respond more effectively to market opportunities. In many cases these are organizations based around similar industries or company types. Examples include new business alliances, industry trade associations, or cooperative activity and buying consortia joined together for the purpose of mutual benefit.

Temporary Virtual Organizations: An extension of the virtual project design is to establish a temporary virtual organization to take on multiple projects and develop responses to a specific market opportunity. When the market opportunity has ended, so has the organization. This is the initial virtual organizational model (Byrne, 1993, Davidow and Malone, 1993, Goldman, Nagel and Preiss, 1995), involving virtual tasks, teams, operation, and virtual management of the organization's activities.

	Virtual Teams	Virtual Projects	Temporary Virtual Organizations	Permanent Virtual Organizations
Topology	Internal to an organizational function or departmental unit	Across functions and organizations	Across organizations	Across organizations
Size	Small, local	Indeterminate	Typically larger	Typically smaller, but scaleable
Operational Mode	Teams on specific, ongoing tasks	Multiple organizational representatives working on specific projects	Multiple functions responding to a market opportunity	All functions and full functionality as a working organization
Membership	Membership varies, but form is permanent	Temporary	Temporary	Permanent
Infrastructure	connectivity, sharing embedded knowledge (e-mail, groupware)	Repository of shared data (databases, groupware)	shared infrastructure (groupware, WANs, remote computing)	Channel for marketing and distribution, replacing physical infrastructure (Web, Intranet)

Published in: Palmer J.W. and Speier Ch.: A Typology of Virtual Organizations: An Empirical Study, in: Proceedings of the Association for Information Systems 1997 Americas conference, Indianapolis, 15.-17. August 1997, http://hsb.baylor.edu/ramsower/ais.ac.97/papers/palm_spe.htm.

Three different kind of virtual organisation according to the degree of virtuality:

Virtual Vicinity (low virtuality)→Virtual Umbrella→Virtual Fishnet (high virtuality)

– Virtual Vicinity: The virtual vicinity is the most physical form of VO. The core business is centralised on a building, office or other physical location, where employees are physically present. Business is done under a single name, which make branding possible. the business

contains the core competence, everything else is outsourced. The company acts like the leader enterprise in the network. The advantage with this kind of organisation is that it allows people to interact, it concentrates the competence under one roof and it akes the company more tangible.

- Virtual Fishnet: The virtual fishnet is the most imaginary and the most imaginary and the most ennoble form of VO. The workplace is not necessary limited to the same area, it is rather a workspace. Instead of revolving around a common name, operations are formed as projects and are continuously appearing and disappearing. Accordingly new companies emerge when new projects are formed and vanish when they are completed. The advantage with this model is that it allows greater freedom for the involvees, they don't have to associate with a single name or business. The freedom is at the expense of a common business name. The virtual fishnet also allows geographical spreading and furthermore, there are no office expenses.
- Virtual Umbrella: In between these virtual extremes you find an organisation that has some parts of the business co-ordinated in a specific place, where the participants of the organisation can meet physically and that operates under a single name shared by the whole network. At the same time this form has some characteristics of the virtual fishnet. For example, involvees work independently geographically scattered, communication is done mainly through electronic surrogates, the co-ordinating centre acts like a connecting link while the actual business revolves around the involvees. The virtual umbrella is a sort of mixtures between two extremities.

Published in: Forslund Jonas, Hoegberg Mathias, Stahl Christiana: Importance of Organisational Structure and Branding in VO, in: Case Study, School of Business, Stockholm University, 1998, http://www.busnav.se/sagt2.htm.

Related Concepts

12.1.1.5 Virtual Corporation

The virtual corporation employs a mass customisation strategy in which both quality products are tailored to specific needs and low cost is achieved. In addition, the virtual corporation is exceedingly agile and flexible, linking a variety of organisations in an ever-changing network in which partner firms contribute to the overall enterprise based upon their core competencies. Work is performed by teams composed of members from across the functions and across the organisations in the network. Members of these teams collaborate whether they are and whenever they are able to do so. The authority to make decisions does not reside only on the top, but is distributed throughout the organisation. Finally, and very important, is the fact that the venture is based on openness, co-operativeness, and trust.

Published in: Powell S., Gallegos F., Securing Virtual Corporation, Information Strategy, Vol. 14, No. 4, 1998, p34-38.

Virtual companies are the routine formation by groups of agile manufacturing enterprises. Speed to market with complex new products is a major competitive advantage. Often, the quickest route to the introduction of a new product is by selecting resources from different companies and synthesising them into a singly, electronic, business entity: a virtual company. If the various distributed resources, human and physical, are ‘plug compatible’ with one another, that is, if they can perform their respective functions jointly, then the virtual company can behave as if it were a single company dedicated to one particular project. For as long as the market opportunity lasts, the virtual company continues in existence; when the opportunity passes, the company dissolves and its personnel turn to other projects.

Published in: Goldman S.L., Nagel R.N., Management, technology and agility: the emergence of a new era in manufacturing, International Journal of technology Management, Vol. 8, Nos 1/2, 1993, p.18-38.

A virtual corporation is a temporary network or loose coalition of manufacturing and administrative services that come together for a specific business purpose and then disassembles when the purpose has been met. Firms team up in a virtual corporation to exploit an opportunity in the market

before it evaporates. Once an intended objective is met, the alliance is disbanded. These ad hoc alliances are short lived, extremely focused, goal driven, and powered by time-based competition. They are both created and dissolved quickly. The live cycle of a virtual corporation depends upon factors such as the intended objective (s) of the alliance, the type of products manufactured, or the services rendered. Organisations that are partners in one instance can be rivals and competitors in the next. Virtual corporations are continuously evolving networks of independent companies linked together to share skills, costs, and access to one another's markets and data.

Published in: Christie P.M.J., Levary R.R., Virtual Corporations: Recipe for Success, Industrial Management, Vol. 40, No. 4, 1998, p.7-11.

The Virtual Corporation is a temporary network of independent companies linked by information technology to share skill, costs and access to one another's markets. The companies quickly unite to exploit a specific opportunity and will disperse afterwards.

Published in: Bultje, René, van Wijk, Jacoliene: Taxonomy of Virtual Organisations, based on definitions, characteristics and typology., in: VoNet: The Netwsletter, 2 (1998) 3, p. 9.

A virtual corporation is a temporary network of independent companies - suppliers, manufacturer, developer and customers - linked by information technology to share skills, costs and market success. Each company contributes only what it regards as its core competencies. The network has no or a very flat temporary hierarchy focusing on functionality along the value chain. The virtual corporation is represented externally by a partner or information/network broker compromising the competencies accordingly. It might be managed and organised internally by any means of management principles incorporating either a leading partner, information broker, steering committee, information technology (eg. workflow systems, groupware, executive information system), etc.

Published in: Erben, Kathrin/Gersten, Klaus: Cooperation Networks towards Virtual Enterprises; in: VONet: The Newsletter, Vol. 1, Nr. 5, 1. Dezember 1997, abrufbar über http://www.virtual-organization.net.

A virtual corporation is a temporary network of independent companies - suppliers, customers, and even rivals - linked by information technology to share skills, costs, and access to one another's markets. This corporate model is fluid and flexible - a group of collaborators that quickly unite to exploit a specific opportunity. Once the opportunity is met, the venture will, more often than not, disband. In the concepts purest form, each company that links up with others to create a virtual corporation contributes only what it regards as its core competencies. Technology plays a central role in the development of the virtual corporation. Teams of people in different companies work together, concurrently rather than sequentially, via computer networks in real time, Byrne (1993).

Published in: Byrne John A (1993), The virtual corporation, Business Week (BWE), Issue: 3304, date: Feb 8, 1993, p98-102.

12.1.1.6 Attributes of virtual corporations

The virtual corporation is different from the traditional 20th century company in several ways. The virtual corporation employs a mass customisation strategy to produce high-quality, feature- and information-rich products tailored to specific customer needs. It maintains low cost with flexible manufacturing and improvement programs such as just-in –time, lean manufacturing and TQM. A networked, team oriented, distributed-decision-making organisational approach provides the virtual corporation with the agility, flexibility and value-chain co-ordination necessary to bring a product to market quickly. The network is dynamic. At any time it synthesises the best available capabilities and resources by linking designers, producers, suppliers, distributors and customers. The virtual corporation features teams that:

- Collaborate, regardless of their individual location and time zones,
- Are granted significant decision-making authority

– Foster openness, co-operation and trust among network organisations and the people who comprise them.

Published in: Gallegos F., Powell S.R., Telecommunications Networks in Virtual Corporations, IS Audit & Control Journal, Volume III, 1997, p.26-28.

12.1.1.7 Virtual Enterprise

The design and manufacture of new products frequently requires the talents of many specialists. An industrial virtual enterprise is a temporary consortium of independent member companies coming together to quickly exploit fast-changing world-wide product manufacturing opportunities. Industrial virtual enterprises assemble themselves based on cost-efficiency and product uniqueness without regards for organisation size, geographic locations, computing environment, technologies deployed or processes implemented. Virtual enterprise companies share costs, skills, and core competencies that collectively enable them to access global markets with world-class solutions their members could not deliver individually.

Published in: Hardwick Martin, Bolton Richard: The Industrial Virtual Enterprise, in: Communications of the ACM, 40 (1997) 9, p59-60.

A virtual enterprise must be able to form quickly in response to new opportunities and dissolve just as quickly when the need ceases. From the information management point of view, communicating industrial information within a virtual enterprise offer many challenges.

Published in: Hardwick Martin, Spooner David L., Rando Tom, Morris K.C.: Sharing Manufacturing Information in Virtual Enterprises, in: Communications of the ACM, 39 (1996) 2, p46-54.

12.1.1.8 Examples of Virtual Corporations:

Virtual Corporation, Inc. operates in a similar way, providing technology consulting services primarily in the areas of project management and technology staffing. See http://www.virtual-corp.net/

Published in: Gebauer, Judith, Segev, Arie: Assessing Internet-based Procurement to Support the Virtual Enterprise, in: VoNet: The Netwsletter (1998) 3, p. 32.

Entovation International is a research and consulting network that specializes in guiding senior managers and policy makers towards successful strategies in the context of networked organizations. Consultants and administrative members of the network are dispersed globally, each contributing their unique skills to the organization's outcome as needed. See http://www.entovation.com/

Published in: Gebauer, Judith, Segev, Arie: Assessing Internet-based Procurement to Support the Virtual Enterprise, in: VoNet: The Netwsletter, 2 (1998) 3, p. 32.

The Internet-based bookstore **Amazon.com, Inc.** can be considered a virtual enterprise, since it gives Web users the opportunity to advertise and actually sell books from their individual Web sites, and collaborates with logistics companies for shipping and handling. Amazon compiles information about products, and provides facilities for searching and online communication among authors and readers. See http://www.amazon.com

Published in: Gebauer, Judith, Segev, Arie: Assessing Internet-based Procurement to Support the Virtual Enterprise, in: VoNet: The Netwsletter, 2 (1998) 3, p. 32.

VeriFone, Inc. is known for its role in payment processing by offering low-cost terminals for electronic credit card authorization. Consisting of widely dispersed geographical units worldwide partnering is playing a major role in application development and other business functions.

See http://www.verifone.com/

Published in: Gebauer, Judith, Segev, Arie: Assessing Internet-based Procurement to Support the Virtual Enterprise, in: VoNet: The Netwsletter 3, p. 31.

First Virtual Corp. realizes the idea of virtual enterprises in a two-fold way. On the one hand, it specializes in multimedia network products, such as video collaboration tools, that help enable desktop PCs to become high-quality communications platforms, thus providing the infrastructure of virtual enterprises. On the other hand, it operates as a virtual enterprise itself, focusing on its self-acclaimed core competencies of being able to innovate fast and continuously and of building powerful partnerships.

See http://www.fvc.com/

Published in: Gebauer, Judith, Segev, Arie: Assessing Internet-based Procurement to Support the Virtual Enterprise, in: VoNet: The Netwsletter, 2 (1998) 3, p. 31.

12.1.1.9 Virtual Factory

The virtual factory is a community of dozens, if not hundreds, of factories, each focus on what does it best, all linked by an electronic network that enables them to operate as one-flexibly and inexpensively - regardless of their location. This network makes it easy for companies with dissimilar computer systems to exchange information about inventory levels and delivery schedules. It allows companies with different CAD systems to collaborate electronically on design. It permits potential suppliers to gain entry to the system in order to bid on jobs with minimal hassle and little or no investment. And finally, it allows a small manufacturer to have to same access to information as a large partner.

Published in: Upton David M., McAfee Andrew: The Real Virtual Factory, in: Harvard Business Review, 74 (1996) 4, p123-133.

12.1.1.10 Virtual Office

The term ‘virtual office’ covers a variety of mobile and remote work environments. We have located the different virtual office environments on a continuum:

- **Telecommuting**, the most stationary arrangement, usually refers to situations in which workers with fixed offices occasionally work at home.
- **Hoteling**, is another type of virtual work. Hotel-based workers come into the office frequently, but because they are not always physically present they are not given a fixed office space. Instead, they can reserve a ‘hotel room’ (more likely a cubicle), where they can receive and make telephone calls and link their laptop computers to the network. The hotel space may be a regular downtown office previously for traditional offices, or it may be a suburban location specially selected for mobile work. Hoteling is popular with professional service firms because personnel frequently work at client sites.
- **Tethered in office**, further along the continuum is the ‘tethered’ worker, who has some mobility but is expected to report to an office on a regular basis. For example, workers of an advertising agency check in in the morning and receive a cellular phone and a portable computer. They are then free to wander around the office or nearby. One supposed benefit of the arrangement is creativity; it is presumably stimulating to move among different work areas in the same day.
- **Home** workers have no office at all other than a room, or possibly the kitchen table, in their homes. They may visit customer sites on some days. AT&T has equipped these workers with furniture, computer equipment, and high-speed phone lines. The work (e.g., customer service or telemarketing) is largely performed on the computer or the telephone.
- Finally, **fully mobile** workers do not even have home offices. They are expected to be on the road or at customer sites at all times during the workday. Typical fully mobile workers include field sales and customer service employees. The virtues of this approach include the ability of mobile workers to spend more time with customers and the flexibility in dispatching workers to customers locations. Inexpensive portable technologies have solved many problems of communicating with these workers.

Published in: Davenport T.H., Pearlson K., Two Cheers for the Virtual Office, Sloan Management Review, Vol. 39, No. 4, 1998, p.51-65.

12.1.1.11 Virtual Team

A virtual team, like every team, is a group of people who interact through interdependent tasks guided by common purpose. Unlike conventional teams, a virtual team works across space, time, and organizational boundaries with links strengthened by webs of communication technologies.

Published in: Lipnack, J., Stamps, J.: Virtual Teams- Reaching across space, time and organizations with technology, New York 1997, p. 7f.

12.1.1.12 Virtualness

Virtualness is the ability of the organization to consistently obtain and co-ordinate critical competencies through its design of value-adding business processes and governance mechanisms involving external and internal constituencies to deliver differential, superior value in the market place.

Published in: Venkatraman N., Henderson C., The architecture of virtual organizing: leveraging three independent vectors, Discussion Paper, Systems Research Center, Boston University, School of Management, 196.

13 ANNEX III: Legal Examples

13.1 Arrow Diagrams

Arrow diagrams are a “formalism” proposed by Layman Allen, using a graphical approach the representation of legal norms[492].

Any legal norm will have the general structure of an antecedent and a consequent:

if [A] then [B]

The antecedent prescribes which factual circumstances have to be present for the norm to fire. The consequent indicates the effect of the norm being applied, often the effect is only a link to further norms, which chained together will represent the norms governing the case at hand.

A “factual circumstance” is termed a “criterion”. They are derived form legal reasoning based on the relevant source material, which may contain statutes, regulations, court or administrative decisions etc, the identification of such sources itself is an essential part of a legal decision process. In most examples of arrow diagrams, as in the one in this note, one take a provision in statutory text, which though written in natural language, to some extent is formalised in the sense that some care has been taken in drafting the text. One should be clear that the “law” is not to be mistaken for such a provision, it is but one of several sources which a lawyer in his or her reasoning has to bring together in order to create an understanding of the law. In this sense, “law” and “legal norms” are of semantic nature, akin to concepts like “information” or “thoughts”, what is rendered is but a representation of the norms.

The criteria may be represented with ambiguities, different degrees of specificity etc, which may be reflected in arrow diagrams. It is proposed that any antecedent can be described as a set of criteria with an internal micro-structure, making the relations between the criteria explicit. These relations are generally rather simple, and can be represented by the Boolean operators “AND”, “OR” or “NOT”.

As an example, we shall look at the Norwegian data protection act sect 8, which sets out the conditions to be met for processing personal data:

§8. Vilkår for å behandle personopplysninger

Personopplysninger (jf. § 2 nr. 1) kan bare behandles dersom den registrerte har samtykket, eller det er fastsatt i lov at det er adgang til slik behandling, eller behandlingen er nødvendig for

a) å oppfylle en avtale med den registerte, eller for å utføre gjøremål etter den registrertes ønske før en slik avtale inngås,

b) at den behandlingsansvarlige skal kunne oppfylle en rettslig forpliktelse,

c) å vareta den registrertes vitale interesser,

d) å utføre en oppgave av allmenn interesse,

e) å utøve offentlig myndighet, eller

f) at den behandlingsansvarlige eller tredjepersoner som opplysningene utleveres til kan vareta en berettiget interesse, og hensynet til den registrertes3 personvern ikke overstiger denne interessen.

The translation for above act, Section 8 Conditions for the processing of personal data, says:

[492] ”Norm” is preferred to ”rule”, the reason for this is of little consequence to this note, but is derived from a theory of norms in which ”rule” is one of several categories of norms.

Personal data (cf. section 2, no. 1) may only be processed if the data subject has consented thereto, or there is statutory authority for such processing, or the processing is necessary in order

a) To fulfill a contract to which the data subject is party, or to take steps at the request of the data subject prior to entering into such a contract,

b) To enable the controller to fulfill a legal obligation,

c) To protect the vital interests of the data subject,

d) To perform a task in the public interest,

e) To exercise official authority, or

f) To enable the controller or third parties to whom the data are disclosed to protect a legitimate interest, except where such interest is overridden by the interests of the data subject.

The provision closely corresponds to the Data protection directive[493] art 7, but has a somewhat different structure. For reference, article 7 is cited below.

Article 7

Member States shall provide that personal data may be processed only if:

b) The data subject has unambiguously given his consent; or

c) Processing is necessary for the performance of a contract to which the data subject is party or in order to take steps at the request of the data subject prior to entering into a contract; or

d) Processing is necessary for compliance with a legal obligation to which the controller is subject; or

e) Processing is necessary in order to protect the vital interests of the data subject; or

f) Processing is necessary for the performance of a task carried out in the public interest or in the exercise of official authority vested in the controller or in a third party to whom the data are disclosed; or

g) Processing is necessary for the purposes of the legitimate interests pursued by the controller or by the third party or parties to whom the data are disclosed, except where such interests are overridden by the interests for fundamental rights and freedoms of the data subject which require protection under Article 1 (1)

[493] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.

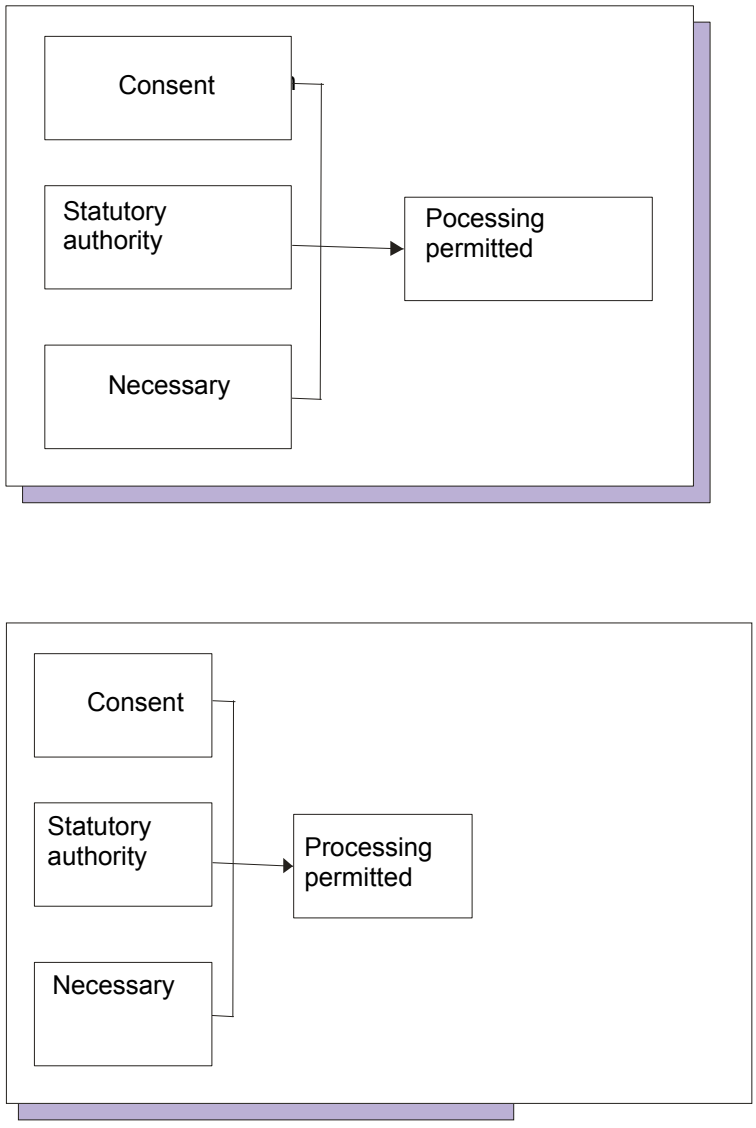


Figure 110 Simple Arrow Diagrams

This gives a simple arrow diagram showing the general structure of Sect 8 – there are three disjunctive criteria which may be the reason for permitting processing of personal data, one being the consent of the data subject (corresponding to Art 7(a)), one being that the processing is "necessary" (corresponding to Art 7(b)-(f)). In addition is the criterion that a statute authorizes the processing, which is not specified in Art 7, but is permitted by other provisions in the directive (it is a rather obvious criterion). The arrow diagram translates to the three criteria being disjunctive in establishing that processing is permitted.

"Necessary" is a generalisation covering six instances of a more specific nature, these can be indicated by creating a structure for this:

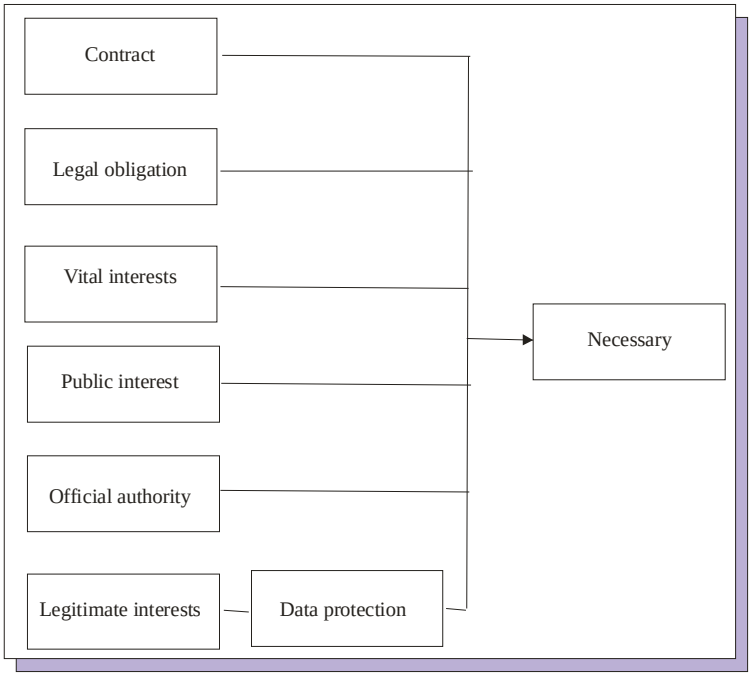


Figure 111 A Generalisation Example (“Necessary”)

In this structure is indicated the six disjunctive instances which may establish the criterion “necessary”, and it is indicated that the last of these instances consists of two conjunctive criteria, the processing must serve the legitimate interest of the controller AND the interest of the data protection of the data subject must not exceed these legitimate interest.

Of course, the structures can be joined, as indicated in the composite diagram below.

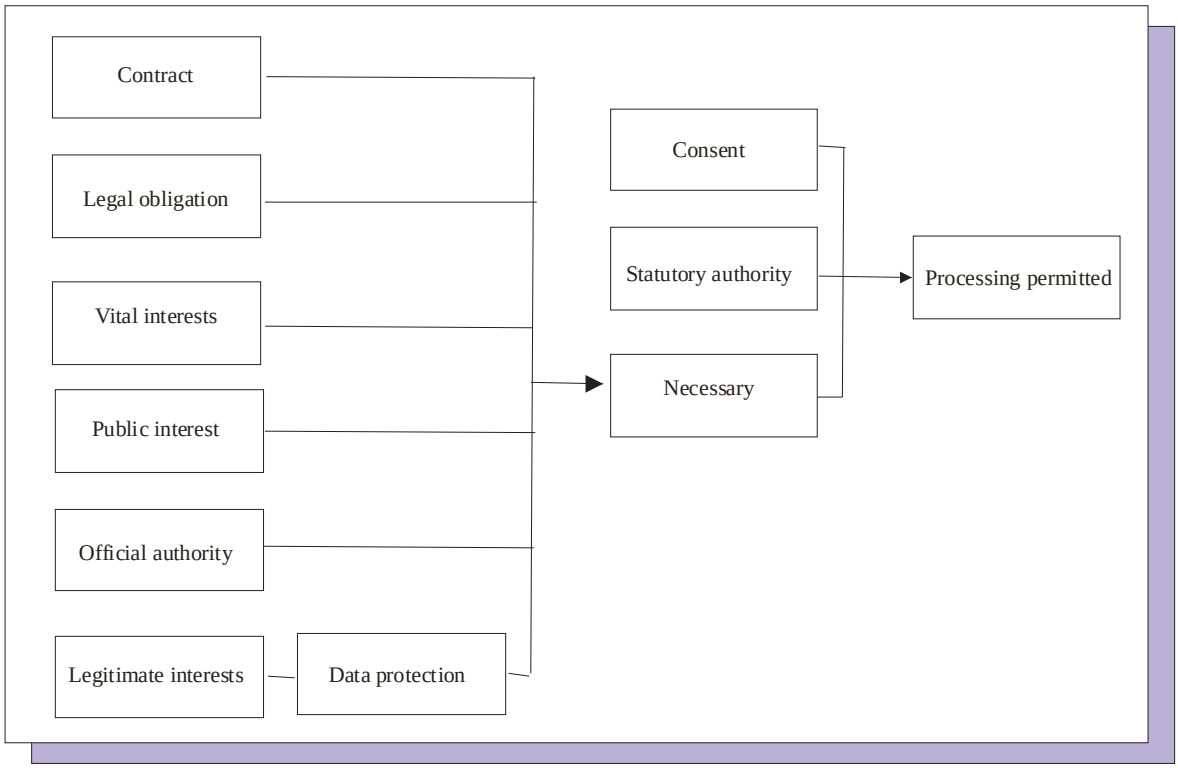


Figure 112 A Complex Arrow Diagram

Arrow diagrams are the result of a manual or intellectual process, and the structures have to be argued by traditional legal methods. Legal prose, even the terse prose of statutory provisions, cannot automatically be transformed into arrow diagrams. It has been claimed, however, that if the principle of legality applies (which briefly explained requires that any decision has authority in law), the structure of the statute may be seen as a high level structure of the legal domain in question, see on hierarchical diagrams below.

Arrow diagrams are very good for representing the micro-structures of criteria in the antecedent of a norm. What is “one norm” is a question of pragmatic, determined by the circumstances rather than the syntax or semantics of the norms. Often a problem is analysed by applying “several” norms, often these may be joined like above to create a continuous “normative space”. But for large structures, the arrow diagrams may lose their power of representing the relations between the different criteria, though they (1) may be compacted compared to what is illustrated above, (2) they may be organised hierarchically in layers (the diagram above may illustrate this, rather than feeding the criterion “necessary”, one may see the left part as a more detailed subordinate level, to be opened by “clicking” on the element – though I am not aware of software supporting such a solution), and using “connectors” similarly as in conventional flow diagrams.

One will note that the criteria which are not fed by other parts of the diagram, will have to be assigned values, one value being “true” or “present in the case”, the other “false”, “not present” or “presence unknown”. These values have to set intellectually. One will notice that criteria may be specified further by developing the “left hand side” of the diagram, the values of subsequent criteria may then be set by feed from these, as the example above feebly indicates.

This may make the distinction between vague and strict criteria interesting.

A strict criterion is interpreted in the same way by different persons and in different contexts. They include only a few categories:

- Quantifiable criteria (using measures for time, weight, volume, speed, price etc)

- Natural status (typically male or female, but also types of material: stone, wood, tin, water etc)
- Legal status (which is the result of a legal decision by some having authority to assign a certain status to a person, situation etc – like licensing a medical doctor, determining a speed limit by putting up a traffic sign etc, cf Jones and Sergot "institutionalised power")
- Relations (at least some of them, like which of two entities is heavier, or relations between humans like parent-child)

It is suggested that this list is exhaustive.

Vague criteria have at least two categories.

First, syntactical vagueness: Criteria may be vague because they are represented by natural language words, such words are by their very nature vague unless indicating a strict criterion. They are often described as "open textured", there being a core of understanding which is inter-personal and inter-contextual stable, and a surrounding zone of increasing uncertainly. This may be questioned as an appropriate metaphor, as words do not occur by themselves, but in a textual context, and the vagueness associated by the word alone is modified or possibly determined by the context. However, in legal argument there will be numerous examples of criteria, which are "vague" in the syntactical vagueness: When do several trees become a forest? Is a houseboat berthed and used for living "a building"?

Second, legal expert judgements: In this case, the uncertainty is not caused by the syntax of natural language, but the semantic of the decision being required to determine whether the criterion is satisfied.

One may compare this to uncertainty caused by statistical or causal probability. One may, for instance, in a will prescribe that the estate shall be given to one of two persons, this to be determined by the flip of a coin. In this case, the criterion will be uncertain until the coin is flipped, but the uncertainty is not caused by the natural language used to represent the criterion, but the underlying decision process. Such uncertainly does constitute its own category of vague criteria not further pursued here.

A legal expert judgement (Norwegian: "skjønn") is suggested – according to a theory of norms to which the author subscribes[494] – be seen as a special form of norms, with a antecedent composed in the usual way by factual circumstances, but where the consequent is the result of adding up several sub-norms.

For a legal expert judgement there are norms determining (1) which factual circumstances may be relevant for the judgement, (2) norms which assign value to the factual circumstances occurring in a specific case, ie which of two or more possible results is favoured by this occurrence, and (3) norms which assign a relative weight to the combination of a factual circumstance of a certain value.

The general from of a sub-norm would then be:

Factual circumstance	Value	Weight

There are examples of the application of this theory. The NRCCL developed in the beginning of the 1980s a system for analysing binary legal expert judgements called SARA, and this was used on several occasions to analyse a large number of decisions according to the model indicated. One of the criteria analysed was "domiciled". In this note, there is no need to go into the substance of this analysis; sufficient is to state that it is traditionally broken down to two criteria, called "factum" and "animus". The criterion "animus" is, in spite of its obvious literal meaning, based on an expert judgement. This can be (in an abbreviated from) represented as below:

[494] Nils Kristian Sundby.Om normer Scandinavian University Press 1975.

Factual circumstance	Value	Weight
Actual residence	Yes (+) or No (-)	(+ 1.00) (- 0.36)
Duration of residence	Yes (+) or No (-)	(+ 0.43) (- 0.29)
Place of work	Yes (+) or No (-)	(+ 0.43) (- 0.14)
Home and family	Yes (+) or No (-)	(+ 0.50) (- 0.79)
Estate concentration	Yes (+) or No (-)	(+ 0.07) (- 0.57)
Foreign decisions	Yes (+) or No (-)	(+ 0.20) (- 0.07)
Citizenship	Yes (+) or No (-)	(+ 0.21) (- 0.29)
Public law relations	Yes (+) or No (-)	(+ ---) (- 0.14)
Self declaration	Yes (+) or No (-)	(+ 0.79) (- 0.21)

This is calculated on the basis of decision on whether a person is obliged to pay taxes to Norway. The issue is whether the person is domiciled in Norway or not, “yes” indicating “domiciled”, “No” indicating an argument for not being domiciled. The factual circumstances are indicated by key-words which do not need further explanation in this note. He weights are calculated by SARA according to an algorithm developed to adjust weights to explain as may decisions as possible – one will note that the weights for the different values vary, and that for the positive value of the factual circumstance “public law relations” there is not available a calculated weight for the positive value, indicating that there probably are no decision in the material in which this circumstance occur with a positive value.

The theory basic to this formalisation (and the formalisation is not part of the theory, but the result of the research at the NRCCL), claim that legal expert judgement is non-deterministic due to two conditions:

One cannot in advice determine the types of factual circumstances which may be relevant, in a new case, there may be circumstances of a type not previous encountered

One cannot in advance determine the relative weight assigned to a circumstance with a certain value in a case, this will depend upon what other circumstances are present in the case.

One will note that the expert judgement may make a vague criterion less vague, as the criteria (factual circumstances) in the expert judgment on a “scale of vagueness” move towards the stricter end. The models of expert judgements may be integrated with arrow diagrams, feeding into a criterion.

Resolving a legal decisions into s set of factual circumstances is often called “factorising”, and in the field of research known as “artificial intelligence and law” there are now many examples of this, many of them much more sophisticated than the one indicated above. The NRCCL research is based on earlier studies, which generally tried to create a causal relation between the factual circumstances and the result, using for instance probabilities. However, it is maintained that the relative weights in the model above do not represent probabilities, but a normative weight. This is discussed extensively in the literature on the SARA model.

In the example, the weights are calculated on the basis of a large number of actual decisions, which are described according to the model, and then made subject to mathematical analysis. However, the weights may be assigned intellectually by a researcher on the basis of his or her understanding of the legal domain – much in the same way that a lawyer in general argues the law.

The model may be seen as a bridge from vague to strict criteria. It has been suggested that the model also could be used to represent the understanding of syntactic vagueness, but there are no examples of this having been undertaken. It is also doubtful if one by reduction is able to arrive at the “far left edge” by criteria, which are strict in the sense indicated above.

In principle, also contractual rules may be represented in the same way as discussed above. But in practice, there are differences of some importance.

First, the domain is circumscribed. The contract has an objective, which is concrete – typically goods or services are to be supplied by one party on the payment of a specified amount of money from the other party. It will be positioned in time and space, and the subjects to the rights and duties are specified.

Second, the contracts have a resolution in detail on a much more specified level than regulatory norms.

Third, the contracts have less “open texture” than regulatory norms. Regulations leave freedom for the parties to make decisions. But when these decisions are made, specific rules are created within the contract.

In principle, therefore, it should be possible to represent a contract as a set of arrow diagrams, where the “end node” or “right node” of each diagram would be the objectives of the contract. Much of the contract, however, will be concerned with qualifying elements – identifying the parties, identifying the object of the transaction, *etc*. To what extent it would be possible to represent a contract, as an arrow diagram would need testing. Flow charts and time lines are often used in conjunction with contracts, but they do not specify the *normative* structure as much as processes and sequences. Arrow diagrams do not necessarily capture such properties in an appropriate way.

As a contract is concrete, one would assume that the risks involved in the transaction could be disclosed by identifying critical elements. The arrow diagrams are themselves not sufficient to identify such elements, but they may be a tool for doing so, as they would make explicit dependence between elements.

13.2 CORAS UML Profile for Security Risk Analysis

The CORAS UML profile for security risk analysis[495][496] (security analysis, for short) introduces a metamodel that defines an abstract language supporting model-based security analysis. Furthermore, the profile provides a mapping of classes in the metamodel to modeling elements by defining so-called stereotypes, and introduces special symbols (icons) for representing the stereotypes in UML diagrams. A stereotype is a specialization of a predefined modeling element in UML, and since the profile is defined by the means of extension mechanisms defined in the UML standard, it is compatible with UML. The motivation for the profile is to facilitate the practical use of UML in security management in general, and security analysis in particular.

In the CORAS methodology, UML models are used for three different purposes:

(1) To describe the target of evaluation at the right level of abstraction.

(2) To facilitate communication and interaction between different groups of stakeholders involved in a security analysis.

(3) To document security analysis results and the assumptions on which these results depend to support reuse and maintenance. The UML profile supports all these objectives, but has a special emphasis on communication and documentation. Documentation is supported since the metamodel of the profile is consistent with the data structure of security analysis documentation developed as part of the CORAS project. Communication is supported by the definition of easy-

[495] Mass Soldal Lund, Ida Hogganvik, Fredrik Seehusen, Ketil STølen. UML profile for security assessment. Technical report STF40 A03066, SINTEF, December 2003.

[496] Lund, M. S., den Braber, F., Stølen, K., Vraalsen, F. A UML profile for the identification and analysis of security risks during structured brainstorming. Technical report in preparation, SINTEF ICT, 2004.

to-understand icons associated with the modeling elements of the profile, and because these specialized modeling elements are consistent with the ontology of security analysis.

Methods for identification and analysis of security risks make use of structured brainstorming sessions. The effectiveness of such sessions depends on the extent to which the involved stakeholders and analysts understand and are understood by each other. Since such sessions involve people with different backgrounds and competencies, like users, system-developers, decision makers and system managers, common understanding among the stakeholders is often not the case. The typical use of the UML profile in such a setting will be that the analyst leading the session presents diagrams for the rest of the participants. The participants need not be familiar with UML, to them the UML diagrams may as well be presented as merely illustrations of what they are discussing. For the analyst, however, the well definedness of UML is of high importance, since it supports structured and uniform documentation of the security analysis results, as well as support for automated consistency checking and analysis.

The profile provides support for the risk management process by providing modeling support for:

- Security analysis context
- Assets
- Strengths, weaknesses, opportunities and threats (SWOT) analyses
- Threats and unwanted incidents
- Risk estimates
- Risk themes
- Treatments
- Treatment evaluations

Figure 113, below illustrates the use of five stereotypes with icons from the security analysis profile; <<Treatment>>, <<UnwantedIncident>>, <<Asset>>, <<Attacker>>, and <<ThreatScenario>>. The asset in the diagram, being "Availability of Service" here, represents one part of the system that needs to be protected. An identified threat to the service availability is flooding by a malicious person. The flooding may lead to a denial-of-service situation. The suggested treatment is a form for service authentication.

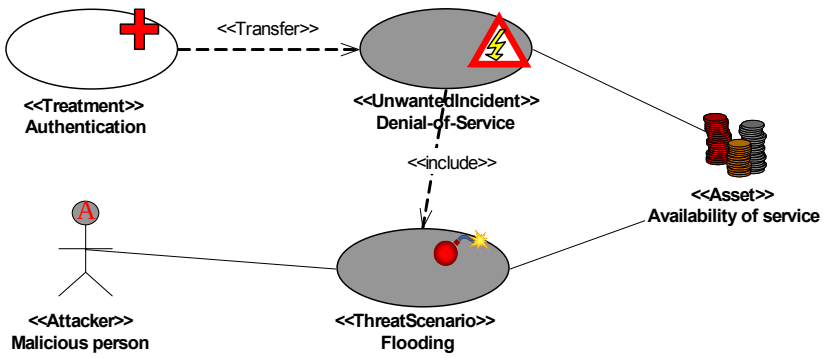


Figure 113 Stereotypes

The profile was developed in the CORAS project and is influenced by continual feedback from a number of large-scale trials where the methodology and the profile were applied in the analysis of real systems. The profile is also a part of the proposal for "UML Profile for Modeling Quality of

Service and Fault Tolerance Characteristics and Mechanisms"[497] that was adopted as a recommended OMG standard in November 2003. The standardization process itself provided much useful input to the development of the profile. The standardization process itself provided much useful input to the development of the profile.

13.3 Extensible Rights Markup Language

XrML[498] (eXtensible rights Markup Language) is a language to specify "rights". XrML is an XML-based syntax for specifying rights and conditions to control the access to digital content and services. XrML had its roots in Xerox Palo Alto Research Center. Digital Property Rights Language (DPRL) was first introduced in 1996. DPRL became XrML when the meta-language (used to construct the language) was changed from a lisp-style meta-language to XML in 1999.

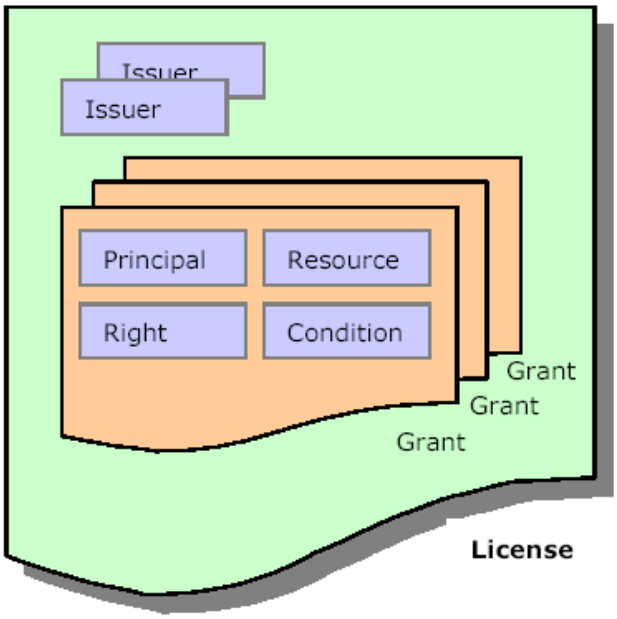


Figure 114 XrML Licence

The structure illustrated in Figure 114[499] presumes an "issuer", which issue a grant, this grant identifying the principal to which the grant is extended, the resource for which the grant is extended, the right that is extended, and conditions on which the grant has been extended. Structurally, a grant consists of the following:

- The principal to whom the grant is issued
- The right that the grant conveys to the specified principal
- The resource against which the specified principal can exercise or carry out this right
- The condition that must be met before the right can be exercised

[497] OMG: UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms, Revised submission to OMG RFP ad/2002-01-07. OMG document: realtime/2003-08-06. Object Management Group (2003)

[498] XrML 2.0 Technical Overview, Version 1.0, March 8, 2002.

[499] The figure has been copied from XrML 2.0 Technical Overview, Version 1.0, March 8, 2002 sect 3.0.

- The principal may be given the right to issue further grants (sub-licensing, in copyright terms), and will then be an issuer. Certificates may be used to ensure the identity of the principals.

In this introduction, the formalism cannot be presented in a way making justice to the approach. We are mainly interested in elements that may be of use for the copymarks introduced above.

The "condition" may include specification for the interval in which the license is valid. It presumes some sort of enforcement mechanism that checks that the principal still holds a grant at the time the interval expires. The "grant" element may specify territory for which the license is valid, payment against the material being made available, etc.

The "content extension" replaces the abstract element "right", and includes a number of items relevant in with respect to, for instance, copyright law:

Type of right	Definition
File management rights	
read	Represents the right to read the work from the repository.
Transport rights	
Loan	Represents the right to lend a work to another principal for a specific period of time. While the work is on loan, the original copy cannot be used.
Derivative work rights	
Edit	Represents the right to make changes to a work to create a new work based on the original work. Edit is like extract in that it creates a new work. It differs from extract in that it confers the right to make changes to the work.
Embed	Represents the right to include the work as part of a composite work. An embed operation places a copy of the work inside the composite work.

This tiny extract gives, perhaps, an idea of the richness of the XrML formalism. It enables full control of the material being made available to a "principal". But it would seem to presume the establishment of a repository that is controlled according to the terms in the grants by some sort of enforcing mechanism. There has been developed a trust model as part of the approach, in which several alternatives are open.

XrML is therefore a language for Digital Rights Management. It would in itself be an interesting task to evaluate the formalism and the control made possible through this formalism with the legal acts relevant, for instance, to copyright law. No doubt this has been part of the development of the XrML, but a mapping of the legal concepts to the definitions has not been available.

13.4 Enterprise Privacy Authorization Language

The Enterprise Privacy Authorization Language (EPAL 1.1)[500] is a formal language for writing enterprise privacy policies. It was developed by the IBM Zurich Research Laboratory in 2003. EPAL is based on XML and is designed to administrate authorizations to conduct privacy-relevant actions defined by any enterprise.

[500] This text is based on the EPAL Specification (IBM Research Report), available at http://www.zurich.ibm.com/security/enterprise-privacy/epal/, last visited 21 April 2004.

13.4.1 Elements

EPAL is based on the following elements:

- Data-categories,
- User-categories,
- Purposes,
- (Privacy-relevant) actions,
- Obligations and
- Conditions.

These elements are not defined in EPAL, but they have to be agreed to by the enterprise that wants to define a privacy policy. The elements can be organized as lists or hierarchically. Interestingly, these elements are somewhat parallel to elements in existing data protection rules. Thus, we will refer to relevant data protection legislation when describing these elements.

Data categories define the different types of data that have to be handled differently from a privacy perspective. Data categories could e.g. be medical data and contact data. The European Data Protection Directive mentions for example the categories sensitive[501] personal data and other personal data (Article 8.1.). EPAL also opens for defining sub-categories, and in the Directive the category of "sensitive data" consists, among others, of the following sub-categories: "data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs etc." In an enterprise context, the relevant categories will have to be quite specific, (e.g. "customer address", "purchase information", etc.) in order to mach the actual data that has to be handled differently.

User categories refer to the types of entities that may request to conduct privacy-relevant actions, e.g. employment department, tax auditor etc. Enterprises can thus define relevant roles, and EPAL also allows multiple roles. The Data Protection Directive defines some roles, like "processor" or "third party" (Article 2 (e) and (f). However, EPAL user categories should be much more specific in order to make sense in a particular business environment.

Purposes describe the intention for which the data is used. Again, according to EPAL, the categories of purposes have to be defined by the enterprise. The term purpose is also strongly related to many data protection rules. An example is Article 6.1. (b) of the above mentioned directive: [Personal data must be] "collected for specified, explicit and legitimate purposes and not further processed in a way incompatible with these purposes. Further processing of data for historical, statistical or scientific purposes shall not be considered as incompatible …" Thus, the Directive expressly mentions some categories of purposes and allows the data controller or the data subject to define their own (categories of) purposes.

Actions referred to in EPAL model how the personal data is used (e.g. disclose, read, analyzed etc.). This can be compared to the term "processing of personal data" used in the European Data Protection Directive. In Article 2 (a), processing of personal data is defined as any operation or set of operations which is performed upon personal data, […] such as collection, recording, organization, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, blocking, erasure or destruction." All the different ways of processing of personal data could possibly be defined as categories of action in EPAL. This requires a further definition of every chosen category in the specific context.

Obligations define actions that must be taken by the enterprise (e.g. delete after 30 days or get consent). Data protection laws contain a number of specific duties that are related to certain ways of processing personal data (i.e. actions in the EPAL terminology). For example, Article 18 of the Data Protection Directive refers to an obligation to notify the supervisory authority in certain cases.

[501] In the Directive, the term sensitive data is not used. Instead this group of data is referred to as "special categories of data". However, in some national implementations e.g. section 2 nr. 8 of the Norwegian Personal Data Act, the term sensitive data is used specifically.

Conditions in the EPAL terminology are Boolean expressions that evaluate the context (e.g. "the user category must be an adult" or "the user category must be the primary physician of the data-subject"). When applying any law, we always have to compare the conditions expressed in the law with our findings in reality. If the reality matches the law, then the legal rule is applicable.

13.4.2 Vocabulary

Prior to defining a privacy policy, the enterprise must define the necessary vocabulary. The above-mentioned elements have to be defined to meet the requirements of the specific business environment. Presumably, an analysis and categorization of all kinds of relevant personal data in an enterprise will be a relatively time-consuming activity which causes major expenses and may be of low priority to decision makers. The definition of vocabulary is necessary to be able to define a policy that deals with data protection issues at stake. In consequence, every enterprise will have its own vocabulary, which only reflects their specific needs. Thus, interoperability with other enterprises using EPAL requires that the involved parties agree on a common vocabulary, which also is a difficult and time-consuming task.

13.4.3 Policies

Once the vocabulary is defined, the enterprise can use EPAL to define a privacy policy. The policy itself is essentially a list of privacy rules that are ordered with descending precedence. IBM Zurich mentions the following example: "[…] an enterprise may have the following rule in its privacy policy:

Privacy Policy (informal):	*Allow a sales agent or a sales supervisor to collect a customer's data for order entry if the customer is older than 13 years of age and the customer has been notified of the privacy policy. Delete the data 3 years from now.*
EPAL Privacy Rule:	
ruling	allow
user category	sales department
action	store
data category	customer-record
purpose	order-processing
condition	the customer is older than 13 years of age
obligation	delete the data 3 years from now

Rules are used to determine if a request is allowed or denied. A request contains a user category, an action, a data category, and a purpose. Continuing with the same enterprise as above, consider the following request.

request (informal)	*A person acting as a sales agent and an employee requests to collect a customer's email for order entry.*

user category	sales department
action	store
data category	customer-record
purpose	order-processing

The above rule allows the request, so the sales agent would be permitted to store the customer's contact information. Additional rules can then govern how this stored data may be used."

13.4.4 Limitations

According to IBM Zurich, EPAL was not designed for directly encoding or enforcing specific privacy legislation in a generic and completely application and enterprise independent way. Additionally, as mentioned above, the interoperability between enterprises (which is crucial for TrustCoM) requires that all involved parties agree on a common vocabulary, which will be difficult and time-consuming. In consequence, EPAL is well suited for formalizing a specific privacy policy, but it does not suit the formalization of more abstract data protection concerns. Whether EPAL could be used to formalize other legal policies is an open question. Indeed, also parts of intellectual property (IP) law could be said to deal with categories of data (songs, films, construction data), categories of users (staff members of the IP holder, staff members of partner firms), actions (copy, publish), conditions (only three copies), maybe also obligations. In certain cases, even the purpose may be of relevance for IP law, e.g. when parties agree that a certain IP only can be used for the purpose of a defined project. However, EPAL was not developed to address these questions.