# Towards a Knowledge Grid: requirements for a GridOS to support the Next Generation Grid

# A Position Paper

Brian Matthews, Alvaro Arenas, Michael Wilson, Damian Mac Randal, Adomas Svirskas, Julian Gallop, Juan Bicarregui, Simon Lambert

CCLRC Rutherford Appleton Laboratory

## 1. BACKGROUND: SUPPORTING THE SCIENTIFIC PROCESS

CCLRC is a UK civil research laboratory which provides large scale facilities to the international scientific community – mainly university based, although also used by industry. Current and planned CCLRC facilities include particle accelerators/synchrotrons that produce neutrons, muons, x-rays and visible light for non-destructive materials testing using a large range of sensors; the world's most powerful pulsed laser; satellite build and test facilities; one of the world's 20 most powerful supercomputers; the UK node to collect, store and nationally distribute data from CERN; a nanotechnology fabrication unit ; an atmosphere radar; the World data centre for atmospheric science, and a satellite tracking groundstation. Users range from individuals and companies, to consortia of 1500 staff in 900 universities, who produce about 1200 academic journal publications annually from their use of CCLRC's facilities.

Our users are distributed globally both as individuals and organisations. With rising energy prices and increasing environmental concerns we need to reduce travel, and increase ICT mediated work. With the globalisation of science we need to distinguish brilliant, reliable and trustworthy users, from the mundane, disorganised, reckless or time wasting. The drivers which motivate the interest of CCLRC to be involved in the development of the NGG are to support the users of CCLRC facilities through:

- remote access to large-scale facilities, including supercomputing, data centres and computational steering of experiments and simulations;
- supporting the business processes of science;
- supporting collaborative work within distributed teams of scientists;
- supporting the collaborative process of science, including virtual organisations and business partnerships with differing requirements for security;
- across all modes of operation (including mobile);
- e-Science, Grid Computing.

Interoperable computing systems do not meet our expected needs for distributed security, scheduling, and maintenance that a single distributed autonomic operating system will.

## 2. REQUIREMENTS FOR A GRID OPERATING SYSTEM WITHIN THE NGG

### 2.1 A GridOS is an OS

If the GridOS is going to be a true operating system then it should support the "traditional" functions of an operating system. These include:

- Provide an abstraction layer for the hardware
- Communication management (networks, terminals, etc)
- resource management (filestores, directory structures) [dmr: are files the correct metaphor – we need to attach metadata/provenance data to all data, so maybe a DB analogy may be better.
- job submission
- resource scheduling, including multi-step schedules (ie workflows)
- security
- user registration
- Fault tolerance / recovery
- Efficiency – the management overhead must be small
- It must coexist and interoperate with other Operating Systems – both peers and those managing the platforms on which it runs.

One clear difference between a GridOS and conventional OS's is that it cannot assume it has complete control over the lower infrastructure, or that its interests coincide with or override those of its constituent parts.

Many of these features can be handled by the current generation of Grid systems (e.g. Globus, OGSA). However, these currently need considerable tailoring and human intervention to establish a Grid in relatively closed communities for restricted purposes. They are also rather sensitive to the platforms on which they execute, and interoperability leaves something to be desired. Consequently the operating system is not sufficiently in control of the whole distributed system at the lowest level – for a distributed computing system of 10,000 processors, each with a failure rate of one in three years, that is more than 8 failures per day; if it takes 3 hours to checkpoint programmes on CPU failure then no computational work can ever be done ! The need for an ability to undertake controlled repair on failure necessitates an OS rather than a distributed computing system communicating through best effort, unreliable protocols.

Proposed distributed operating systems such as IRIS and Chord from MIT and 2K from University of Illinois at Urbana-Champaign, use *distributed hash tables* (DHTs) to support multicast and anycast event notification. They aim to be robust in the face of failures, attacks and unexpectedly high loads using resource management within the operating systems to accommodate frequent change, while being scalable, thereby achieving large system sizes without incurring undue overhead but are also self-configuring, automatically incorporating new nodes without manual intervention or oversight. Such projects provide a baseline for what can be achieve to meet our requirements.

## 2.2 Security

As a distributed system, a GridOS should provide a uniform security infrastructure that can be applied over any security mechanisms provided by the underlying platforms. Since this precludes a single central scheme, reliable delegation mechanisms are needed (for Single Sign On at the minimum). Recognising the distributed nature of Grid users, (in the sense that several/many users may need to work together to achieve their common goal), the Security system should be able to (including but not limited to):.

- Support and secure the lifecycle of Dynamic Virtual Organizations comprising of resources and individuals authorized to use them
- Apply multiple authorization policies to control access to resources
- Enforce fine-grained access control at the resource
- Handle and facilitate interoperation of the multiple authorities necessitated by distributed VOs and resources
- Handle policy conflicts where individuals may play different roles, at the same time or at different times, and enforce separation of duties so that one individual acting in different roles may not leak information from one project to another
- Adapt policies to changes in the contract and the business processes
- Adapt policies in response to certain events in the overall execution environment: failures, external attacks, unauthorized access attempts, changes in other security policies, etc.
- Adapt policies in response to variations of trust vested into both external clients and internal partners, as well as changes in the risk associated with the activities
- Store different types of information about parties and to compute their reputations/trustworthiness based on this information and on trust/reputation algorithms. The actual algorithms and types of information that are used by the trust and reputation management system should be configurable, so that the system can be tailored to the domain of use
- Deal with intentional misrepresentations of trust in a party to (explicitly) damage reputation
- Store securely contract elements (including contract templates)

Of course, the basic security requirements such as identity, authentication, authorisation, message integrity, confidentiality, non-repudiation of origin, message traceability, preservation of privacy etc. must be satisfied be default.

## 2.3 Managing Virtual Organisations

The vision is to have a Grid system which spans different organisations and activities, which can establish access to resources quickly for new purposes

This has to be done in the context of a distributed system which cuts across other domains, which in their local areas of jurisdiction, must take precedence. Need to therefore in general be able to handle the establishment of VOs quickly and with little user intervention. This requires the GridOS to be able to handle SLAs, contracts, and establish virtual firewalls around the trust domains for particular collaborations.

Many researchers have operated for many years in both networks and project consortia or collaborations, which are forms of VO. Those who have participated in such research consortia will have experienced many of both the strengths and weaknesses of current VO's [10]. Such consortia can be established quickly and operated by stable member organizations easily as long as nothing goes wrong, and when the research products are public domain, so there is no required division of benefits. Once such VO's encounter problems with consequent liabilities, or result in benefits where financial reward needs to be allocated between partners then the VO can be bogged down in legal arguments between the members

To make the creation, operation and dissolution processes rapidly responsive, requires both the appropriate legal mechanisms, and dynamic management of the VO. There appears to be an obvious match between the business driven desire to create and manage dynamic VO's, and the technological solution available in composable Web services. Although proprietary IT implementations of composable Web service tools exist (e.g. [11]), secure tools for VO management (VOM) based on interoperating open standards are not yet available, let alone those that address the legal issues too. Consequently, IT based dynamic VOM is confined to the closed community that have adopted a single proprietary solution.

To open up the market for dynamic VO membership to the wider community who can present their businesses via Grid and/or Web Services (the two worlds now quite rapidly converging [14]), requires the development of VOM tools that apply non-proprietary interoperable technology standards that can address VOM including the legal issues.

Although there are no identified VO management systems that address the range of requirements of security, interoperability and legal issues, there are many Web Services composition approaches ranging from robust orchestrators, less rigid choreographers, to those applying the flexibility of the Semantic Web [13] to address service discovery. However, the closest any get to addressing the legal requirements of the VO management process is in stating Service Level Agreements (SLA) for specific services to specify accounting and billing procedures and quality of service.

One issue that falls outside the terms and conditions of an SLA, and therefore solely within the scope of the overall collaboration agreement, is that of trust, because when one can resort to the legal enforcement of binding terms and conditions, trust is not required. The terms and conditions of an SLA can be managed in terms of security and quality of service policies applied to each service. Trust is required when the terms and conditions that users actually "live by" do not apply, or are broken because of unforeseen circumstances. In those cases different parties rely on the good will, and commonality of goals of those involved to resolve the issues that arise in unforeseen circumstances.

Therefore, GridOS would greatly benefit from a built-in generic trust and reputation management architecture for modelling trust metrics and reputation algorithms across domains, allowing for the integration of trust/reputation information from across disparate domains into a single representation, the inference of reputation in new domains from existing knowledge of an agent's reputation and the creation of an infrastructure for modelling original metrics and algorithms.

The TrustCoM project [15] (http://www.eu-trustcom.com/) addresses these and other aspects of modern VOs by bringing together experienced partners from academia and IT industry across the Europe.

### 2.4 Using Knowledgeable Brokers

Users will not be using our resources alone, therefore resource discovery is needed as well as resource management – this requires a higher level of "intelligence" in the system – that is only provided by knowledgeable broker agents.

Proxy and Broker agents within the GridOS. Acting as the role of individuals and resources. Proxy agents have knowledge of the goals, obligations and preferences of the resource (user, computational resource, data-store) – couched in a machine readable knowledge representation (Semantic Web). Broker agents can then coordinate resource discovery – including

The GridOS to be able to handle workflows and business processes – manage and coordinate these, including negotiation of meaning, and audit trails – provenance data in resource metadata. Probably use a notion of "task-agent" as the proxy for a process across the GRID. Requires a more "knowledge-driven" and "goal-directed" approach than is currently supported by e.g. BPEL4WS. For example, the task agent directing the process needs to be able to negotiate with and take into account the requirements of user proxies.

Also, task agents need be able to reorganise their processes in the face of a changing environment – services may drop out, refuse access or not deliver results to the quality required (or even deliver more so subsequent stages are not needed). Then task agents need to discover appropriate resources, negotiate access and contract, to satisfy their process goals on-the-fly.

Differences in approaches: central orchestrating agent vs. distributed choreography. Need to be sorted out, but in general the choreography approach will be suitable to work across separately managed domains. However, very very few of the existing workflow systems (400+ and counting) are distributed. Most processes assume centralized authorization if not direct control. Fitting this "natural" centralization into a distributed workflow engine will require secure and efficient VO mechanisms (as above)

### 2.5 Accessible by everyone, everywhere

- Ambient devices
- transfer of context across devices
- Location specificity
- More flexible (but tighter!) security
- Accommodation to the user needs and preferences.

### 2.6 The Verified GridOS

A GridOS requires high levels of assurance as to the dependability level of the components. This includes the use of verification techniques in the design, development and on the fly deployment of the GridOS. A critical area is the verification of non-functional properties such as adaptability, security, trust, responsiveness, fault tolerance and quality of service (QoS), among others [Aig03].

The realisation of the Verified GridOS implies the convergence of diverse communities such as Grid Computing, Formal Methods, Dependability, Emergent Systems and Programming

Tools. It would include the development of formal descriptions that allow one on one hand to specify the main components of a GridOS and on other hand their non-functional properties. It would also include the development of tools to automatically verify the non-functional properties; such verification tools should be incorporated within programming environments, making their use transparent for developers.

The Verified GridOS will increase the quality of a GridOS by assisting developers in several areas. When a component is specified as a complex service in an abstract language, the goal of the designer might be to refine this component into an architecture built from basic services, following provably correct steps that guarantee the correctness of the transformation process. This might include new challenges in contrast to the traditional model of refinement [Pol05]. Given an architecture, the designer may want to verify that the protocols used are valid with respect to some non-functional properties. Adaptation is also a key aspect; a designer may want to change some parameters, for example QoS parameters such as bandwidth of connections, in order to meet non-functional requirements.

Researchers at CCLRC, in the context of the TrustCom project [Dim04], are working towards the development of a framework to secure collaborations in Virtual Organisations, emphasizing on non-functional properties such as trust and security.

### *2.6.1 Dependability and Component based software evolution*

A grid based OS cannot be static. Throughout its lifecycle, it will be subject to modifications, upgrades, reconfigurations, and extensions. In short, it will evolve. It will evolve from the moment it is conceived,
through all the stages of its development and throughout
its service lifespan. Its decommissioning, if this should ever happen, can be seen as the final stage of evolution: replacement by a fitter variety.

Interoperability and dynamic reconfiguration will be important at every stage of this lifecycle. From commissioning, through upgrades and reconfigurations, to decommissioning, a grid based OS cannot operate in isolation but must remain compatible with its environment as it changes and must adapt to environmental changes it without compromising its reliability.

Compatibility of system components in a compound system is today a major hurdle to the dependability of such systems. Even for shrink-wrapped off-the-shelf packages, compatibility with versions of operating systems and other applications can be an issue. Compatibility of modules in an even slightly more complex system configuration, such as a server running an operating system, database and variety applications on the database, is even more difficult and the dependability of each particular configuration is often based on testing rather than any designed-in features. On the other hand, incompatibility of components is the accepted norm and is tolerated as an inevitable consequence of complexity.

Today's version control and configuration management techniques can be successfully employed within a controlled environment of a particular software suite, but do not easily extend to heterogeneous systems or systems of systems, where demonstration of compositionality of system components essentially relies on testing in situ.

Software engineers, have for many years advocated compositionality as a way to manage complexity in software and systems design and development through decomposition of the

design and verification of it. This approach to structuring, if extended to more aspects of the lifecycle, for example, to requirements and testing, can go a long way to achieving interoperability between systems components. However, because of the wide variety of possible ways to configure software, and in particular the large number of potential interfaces to essentially the same piece of functionality, such structuring needs further features if it is to help with maintenance and reconfiguration. If software is to have the same success in reuse and reconfiguration as other engineering disciplines, we must also achieve flexibility through standardisation and reuse of commonly adopted components. What is required is the ability to put components together with confidence that the dependability of the whole will be predictable from the dependability of the parts.

A grid-based operating system built from dynamically reconfigurable components will necessitate demonstration, at compile time, that the each component performs and integrates as required. To achieve this will require significant advances in the field of component based systems and the mechanical verification of software.

### *2.6.2 Dynamic reconfiguration*

But standardisation of software functions and interfaces is not enough. To enable dynamically reconfigurable systems requires dynamic assessment of new components when they are brought into the system such as is done, for example, in the USB standard for peripherals which enables different peripheral devices to be added whilst dynamic interaction between platform and peripheral is undertaken to establish the correct interface for communication and to reconfigure the platform to exploit the new hardware. To establish comparable standards enabling dynamic reconfiguration of software would require dynamic assessment of the interface and function of software and is certainly way beyond the current state of the art of software engineering. A Verifying Compiler, [cite Tony Hoare] able to assess dynamically the properties of software in order to determine automatically whether a component is applicable to a given purpose would provide a very significant step towards realising this vision.

### *2.6.3 Mobility*

Diminishing cost of hardware and increasing numbers of installations open the potential for ubiquitous, highly-functioned assisting devices for very many aspects of everyday life. However major benefit arises only if these components can communicate in a dependable manner ultimately enabling trustworthiness to be placed on them. The emergence of grid infrastructures and in particular a grid based operating system brings with it the potential for increased mobility and "thin" client devices in the next generation grids environment which will further open the possibilities for such applications to facilitate geographically independent access to resources on a global scale. Provided the significant integration and interoperability aspects, described above, can be addressed, mobility will provide a further essential constituent technology to enable global ubiquitous computing.

## 3. THE CONTRIBUTION OF CCLRC

CCLRC has 40 years of experience managing very large (now petabyte) digital libraries and data curation services. CCLRC first hosted the world's most powerful supercomputer in 1964, and has continued since then to provide a world top 20 supercomputer to the UK research

community. We have staff expert in all aspects of service provision including management, legal issues and technical provision.

Arising from this service provision, CCLRC has existing requirements as a user for a robust, autonomic, distributed operating system that can securely schedule operations across thousands of CPU's and petabytes of data to be used by users around the world.

CCLRC has established skills in current Web Service and Grid technologies, being a partner in the European Datagrid, the home of the UK node of the CERN distribution Grid, the host of the UK National GRID service, and a technical developer in a range of security and trust orientated IST projects considering VO's: Grasp, TrustCom, Akogrimo. CCLRC's roles in these projects include providing requirements, architecture design, programme co-ordination and software development.

The CCLRC RAL supercomputer in 1973 was the first machine outside the USA on the ARPAnet, and CCLRC have continued at the forefront of high performance networking and distributed computing since then.

CCLRC hosts the UK and Ireland Office of W3C, and were one of the first 50 sites on the Web. We have been involved in ontology and rule based modelling for twenty years. Bringing these together, CCLRC has been at the core of the development of the Semantic Web since it was first envisaged. We expect these distributed knowledge management skills to be at the core of a GRID OS.

**References**


[Aig03] R Aigner, M Pohlack, S Roettger, S Zschaler. *Towards Pervasive Treatment of Non-Functional Properties at Design and Run-Time*. In proceedings of ICSSEA 2003, International Conference on Software & Systems Engineering and their applications, 2003.

[Dim04]. T. Dimitrakos, D. Golby, P. Kearney. Towards a Trust and Contract Management Framework for Dynamic Virtual Organisations. In Proceedings of eChallenges 2004.

[Pol05] F. Polack,S. Stepney. *Emergent Properties Do Not Refine*. To appear in Refinement Workshop 2005. Available at http://www.cs.york.ac.uk/nature/tuna/outputs/emergence.pdf .

10. Wildeman L, Alliances and networks: the next generation, International Journal of Technology Management, 15: 1/2, pp. 96-108 1998.

11. Ferguson, D.F., Storey, T., Lovering, B. and Shewchuk, J. (2003). Secure, Reliable, Transacted Web Services http://www-106.ibm.com/developerworks/webservices/library/ws-securtrans/

13. Biplav Srivastava and Jana Koehler2003, Web Service Composition – current solutions and open problems. ICAPS 2003 workshop on planning for Web services, Trento, Italy.

14. Shread, P., (2004) Grid Computing Planet "Grid, Web Services Get Closer" http://www.gridcomputingplanet.com/news/article.php/3304571

15. Dimitrakos, T. et al, 2004. TrustCoM - A Trust and Contract Management Framework ena-bling Secure Collaborations in Dynamic Virtual Organisations. ERCIM News No. 59, pp 59-60, Sophia Antipolis, France. http://www.ercim.org/publication/Ercim_News/enw59/dimitrakos2.html