A Semantic Web Approach to Experience Management in Public Organisations

Simon Lambert, Alvaro Arenas, and Alistair Miles

Business and Information Technology Department,
CCLRC Rutherford Appleton Laboratory,
Chilton, Didcot, Oxon, OX11 0QX, UK
{S.C.Lambert, A.E.Arenas, A.J.Miles}@rl.ac.uk
http://www.bitd.clrc.ac.uk/

Abstract. The Pellucid project is developing a customisable platform for experience management in public organisations. This paper introduces the Pellucid model of experience management, relates it to the design of the platform, and discusses how Semantic Web technologies have been applied in the project. The principal idea is the active hint as a transmitter of experience, generated spontaneously according to the employee's working context. The working context is modelled using ontologies for work processes and specific domains of work.

1 Introduction

Human problem solving in many fields is based on experience. *Experience management* is a special kind of knowledge management, focussing on the dissemination of specific knowledge situated in a particular problem-solving context [5]. Information and communication technologies can play a vital role in experience management systems, providing the connectivity that is required to share experience.

Pellucid is a project tackling the management of experience in public organisations, particularly those aspects related to organisational mobility, the movement or circulation of staff from one unit to another within an organisation [11]. Organisational mobility is increasingly commonplace in public organisations, and presents both opportunities and challenges: opportunities for improving working practices through the introduction of new perspectives, and challenges arising from the constant loss of experience and the learning curve of the newly arrived staff.

The Pellucid project is developing an adaptable, customisable software platform for rapid deployment of integrated experience management systems in public organisations. The basic metaphor for an experience management solution is of an intelligent assistant that looks over one's shoulder and answers questions one might have at a particular point of work [12]. The assistant detects that an employee is working in a particular context, offering knowledge resources that facilitate their work according to their need. To this end, the Pellucid platform must integrate technologies such as autonomous cooperating agents, organisational memory, workflow and process modelling, and metadata for accessing document repositories. The machinery of the Semantic Web, including the Resource Description Framework (RDF) and Web Ontology Language (OWL), provides a powerful framework for achieving this integration.

This paper introduces the experience management model followed in Pellucid and describes a Semantic Web approach to implement such a model. Section 2 presents Pellucid experience management model. Section 3 explains the system architecture. Two important components in experience management systems are the working context and the organisational memory. Section 4 shows the method followed for modelling working context. Section 5 presents a simple example, and treats some practical aspects of the project, while section 6 goes into more detail about how Semantic Web technologies are exploited in the project, in particular for the organisational memory. Section 7 compares our system with related work. Finally, section 8 presents the main conclusion and highlights future work.

2 Experience Management in Pellucid

The experience management model in Pellucid exploits the experience sharing concepts expressed in [5]. It is based on two main ideas: every employee in an organisation is both a provider and a user of experience; and employees communicate their experience to a common medium, then retrieving experience in a future from this common medium. Pellucid aims to be such a common medium. The Pellucid experience management model comprises three phases: *Capture and Store*, *Analysis and Presentation*, and *Experience Evolution*.

The *Capture and Store* phase is concerned with observing and storing experience in a particular context. There are three ways of capturing experience: analysing employees' actions and workflow events, analysing documents entered into the system, and by direct input from workers. Capturing experience from working actions and events is particularly beneficial in repetitive tasks; they are used to create common patterns that can be retrieved in the future in order to assist other employees. Documents constitute an important asset in an organisation. Metadata is added to documents, enabling the system to retrieve documents useful in a particular working context. The direct capture of experience from employees is carried out through free-text notes written by the employees themselves. This constitutes a good source of knowledge, particularly in the transmission of experience from experienced employees to novices.

The purpose of Pellucid is to support and enhance employees' performance by providing them with the knowledge required by the activity they are performing at the time they are actually performing the activity. The *Analysis and Presentation* phase is concerned with providing such knowledge. To do so, the concept of an *active hint* is introduced. An active hint is the vehicle by which experience is delivered to the employees of an organisation. An active hint is triggered for an employee when that employee enters a specific working situation (working context) and includes a suggested action, a knowledge resource and a justification for the hint. The working context of an employee is deemed to be those features of that employee's working situation and environment that are relevant to the decisions they are making and the tasks they are executing. An action corresponds to an act on a knowledge resource, for example use a document template, read a document or a note, or consider a contact list. The justification gives to the employee a reason for the hint.

The idea of active hints owes something to the DECOR project [2], with its workflow-triggered knowledge delivery in which queries to a knowledge archive are automatically launched from the running workflow. The main difference is that DECOR is focussed on delivery of documents from its archive, whereas Pellucid broadens the scope of hints to cover contacts, free-text notes, situation-specific advice, etc. Table 1 depicts a schematic example of an active hint in the context of a proposal evaluation when managing a project.

Table 1. An example of a simple active hint

Context:	Proposal Evaluation
Action:	Consider
Resource:	List of People and Documents
Justification:	People in the list have evaluated similar proposals in the past, and they have used the associated documents for such activity.

The aim of *Experience Evolution* is updating the available experience. Due to the rapidly changing environment, experience may have only a limited lifetime. Invalid experience must be identified and removed or updated. To this end, the final Pellucid platform will include a set of methods and tools to allow knowledge engineers and expert users to update the experience stored in the organisational memory.

The Pellucid project has three end-user pilot sites with different experience management needs and working environments.

– The management of publicly funded projects in the Mancomunidad de Municipios del Bajo Gualdaquivir (MMBG) in Spain.

– The installation and maintenance of traffic light plants in the Traffic and Mobility Management Department of the Comune di Genova in Italy.

– The call center for management and resolution of fixed telephony breakdowns of the Consejería de la Presidencia de la Junta de Andalucía in Spain.

3 Pellucid Architecture

The agent-oriented paradigm was employed in the design of the architecture of the Pellucid system. Agent-oriented design provides an intuitive way of modelling the division of responsibility between concurrently executing processes. This is especially useful where system components are required to be autonomous and proactive. Agent-oriented design also provides a convenient framework for describing how and when concurrent processes interact with each other. Essentially, each agent is like a worker, each has a set of responsibilities, and they work together to provide a coherent service.

A modular, agent-oriented design also confers the following benefits:

– The system behaviour can be easily customised or extended by adding new behaviours to agents.
– A clear concurrency model provides a strong basis for scalability.

Fig. 1 shows the block architecture of the Pellucid system in terms of agents. The responsibilities ('behaviours') of the Pellucid agents are described below.

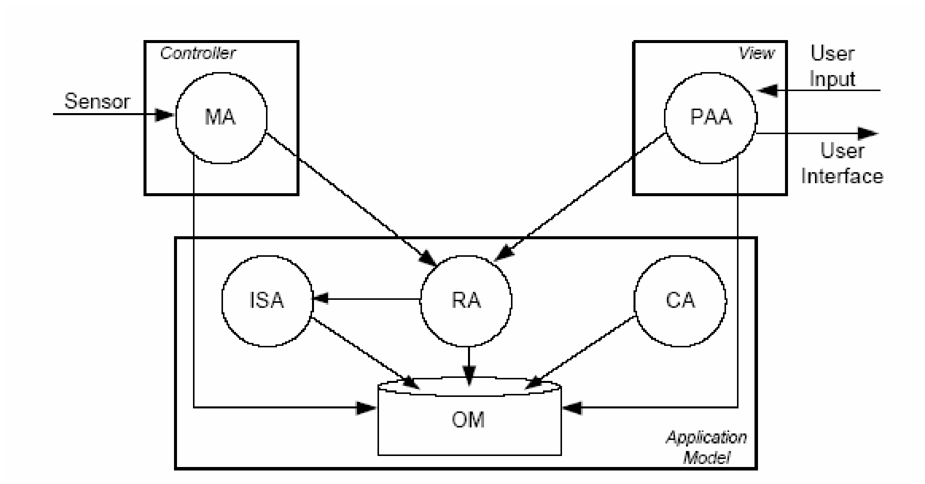

Fig. 1. The Pellucid block architecture

The Monitoring Agents (MA) are responsible for managing all interaction between the Pellucid system and other computational systems present within the organisation. An MA deploys a set of sensors, through which it gathers information about the state of the organisation, and captures events relating to the working context of the organisation's employees. Essentially the MAs are the eyes and ears of the Pellucid system.

Each Personal Assistant Agent (PAA) is responsible for managing all interaction between the Pellucid System and an individual employee. A PAA provides, via a graphical user interface, a personalised portal through which the employee can interact with the organisational memory. Information can be *pushed* to the employee, via *Active Hints*, and it can be *pulled* by the employee via search and browse functionality. A PAA can also capture information directly from an employee.

The core of the system comprises Role Agents (RA), Information and Search Agents (ISA) and Capitalisation Agents (CA). This module includes the functions and data that together deliver the experience management functionality of the Pellucid system. Role Agents (RA) manage the update of the Organisational Memory (OM). They also build and maintain the hint firing engines that deliver active hints to the users. Information Search Agents receive and execute queries on the OM and other organisational repositories. The Capitalisation Agents analyse the contents of the OM to discover trends in and infer new facts about employee behaviour.

Let us relate the Pellucid architecture to our model of experience management. The MA implements the Capture phase, by 'capturing' events performed by workers, events such as workflow activities and manipulation of document. The RA receives new events from the MA and stores them into the OM, after adding some features of the current working context. These actions of the RA correspond to the Store phase. The RA, with the aid of the ISA, is also in charge of firing active hints; such hints are sent to the PAA in order to be displayed to the worker. These actions correspond to the Analysis and Presentation phase. Finally, the CA asserts new facts based on review of historical data from OM, such facts are then validated by experts within the organisation. These actions correspond to the Experience Evolution phase.

4 Context Modelling in Pellucid

The platform must be able to represent and handle the working context in order to manage the active hints for fulfilling the purpose of Pellucid: supporting the employees in their working situation and make their task easier. The approach is that of context-based retrieval information [4]. Context is generally regarded as a major concept for the meaning of the knowledge.

The development of ontologies is one of the central threads of modelling work within the Pellucid project. We have developed a workflow ontology, including general concepts of the business process of the organisation, as well as a domain ontology, which encompasses the relevant conceptual description about the actions (what the employee is able to do), the resources (document, person) and the relevant descriptive concepts closely related to the domain application (e.g. type of document, topic of document, role of person, etc). A full description of the developing of ontologies in Pellucid is presented in [3] and [10].

The working context is described by the position in the work process (starting or completing a task, opening a document, …) and domain-specific characteristics. In the Pellucid project, three pilot applications have been studied, as listed above. In each case, a workflow is defined and implemented. This may be quite flexible—it does not have to be a rigid procedure. The domain is directly related to the domain application, i.e., call centre, traffic light installation and project management. Therefore, the instance in the workflow ontology is an incidence resolution, a dossier and a project. Table 2 illustrates part of the working context modelling in the framework of each pilot application. Note that a wide variety of working domains may be represented within the Pellucid ontology of working context. At the higher levels the ontology is domain-independent; all work domains with a reasonable degree of structure will have actors and activities. At lower levels domain-specific concepts appear. The filling out of the ontology is part of the task of customisation.

Table 2. Modelling working context for the Pellucid pilot applications

Working Context modeling				
Concepts	**Representation**	**Examples**		
		Call centre	**Traffic light control**	**Project management**
Business Entity: Actor	Employee which starts an wfActivity in order to manage the wfInstance	*Id_agent*	*Id_officer*	*Id_project_manager*
WfActivity	The current stage of the process	*Intervention_ request*	*Preliminary_analysis*	*Proposal_submission*
WfInstance	The studied concept from the application	*Id_incidence*	*Id_dossier*	*Id_project*
Concept1	Other domain or workflow concepts used as relevant descriptive information in order to compute the similarity measure.	*Id_site*	*Id_crossing_road*	*Id_topic*
Concept2		*Id_severity*	*Id_number_lanes*	*Id_type_document*
Concept3		*Id_type_site*	*Id_existing_traffic_ light*	*Id_role_actor*
Concept4, etc		*Id_operator*	*Id_applicant*	*Id_objective*

5 Pellucid in Practice

In order to make the ideas presented in the previous sections more concrete, a concise example is shown in Table 3 of the process of hint generation from working context. It is given as a sequence of transactions between the agents of the Pellucid architecture, expressed informally in order to make clear the basic principles. The sequence can be referred to the block diagram of Figure 1.

This example is of just one kind of active hint, albeit a very generic hint which is applicable across a range of applications. One of the strengths of the Pellucid design is the flexibility inherent in the hints. Very generic hints such as the one just shown may be applied in very different applications; the only requirement is that there are defined steps in the work process and that these steps have a history of other employees having performed them using certain documents that are still accessible. Of course, this does not mean that such hints will necessarily be valuable in all applications—the job of customising the Pellucid platform includes requirements acquisition to determine where the benefits from experience management will be felt.

Free-text notes, which at first sight appear to be a very different kind of hint, as their content is entered by users themselves, in fact fit within the same framework. The text of the note itself is regarded as a resource for retrieval and presentation to the employee. Thus the same mechanism is available for different applications. In addition, specially crafted types of hint may be constructed for particular applications. This can be seen as a knowledge engineering job, rather like building an expert system for experience management.

Table 3. Sequence of hint generation in Pellucid

1. Employee starts task of proposal evaluation
   Obviously the system must be able to detect some employee behaviour. This is achieved through the workflow system.
2. Workflow system reports event to MA
3. MA translates event into terms of ontology and passes to RA
4. RA writes event to OM
5. Event is stored with features extracted from current context
   Features include expertise of employee, specific information about proposal worked on.
6. Event listener for hint engine reacts to event and triggers hint builder
   The generic hint type triggers when a new activity is started
7. Hint builder constructs hint to advise employee that previous documents were consulted at the same stage in similar proposals
8. RA requests the ISA to search the OM and/or external repositories for documents with specified properties
   For example, certain keywords or ontological annotations, or used by other employees at the same stage in other process instances.
9. ISA returns set of documents
10. The completed hint is passed to the PAA
11. PAA presents the hint to the employee
    Taking into account their preferences (simple user modelling)

Customisation of the Pellucid platform consists in adapting and reusing elements, the type of reuse depending on the nature of the element itself: for example, the software representing an agent class might be stripped down into a template for future application, and/or provided as an example to be modified. The general reusable elements include the following:

– ontologies (general and domain-specific);
– general user interface (Web application);
– external system interfaces (to document repositories and workflow systems);
– the agents themselves;
– the organisational memory.

Table 4 indicates a preliminary view of how the elements can be provided in the platform, and the corresponding kinds of customisation that will be applicable.

Table 4. Customisation of the Pellucid platform

Element provided in platform	**How customized**
Core (e.g. of ontology)	Expand/Populate/Possibly adapt
Template (e.g. of agent class)	Fill in/Populate
Prior case (e.g. of agent class)	Adapt
Building blocks (e.g. of org. memory)	Assemble
Guidelines (e.g. for interfacing to document repositories)	Implement
Specifications (e.g. for interfacing to workflow management system)	Implement

The Pellucid system has been successfully integrated with the three pilot applications of the project, and work is in progress on populating them with a full range of active hints and on refining the user interfaces and interfaces to workflow systems. The emphases of each pilot application are somewhat different: at the Comune di Genova, free-text notes; at MMBG, hints about previously used documents and contacts; at SADESI, hints tailored to the specific needs of the call centre environment. Nonetheless they all fit comfortably within the general framework.

6 Semantic Web Technologies in Pellucid

This section describes some of the finer detail of the implementation of the Pellucid platform, and focuses on how semantic web technologies were used to achieve the desired experience management functionality.

6.1 The Organisational Memory (OM)

To be an effective resource for experience management an OM must maintain two bases of information:

– A description of the current state of the organisation.
– A history of all organisational events.

Conceptually the Pellucid OM was divided into two parts, corresponding to the two information bases described above.

The description of the current state of the organisation is implemented as an RDF model. This model is called the *current state model*. An RDF model is a set of RDF statements. An RDF statement is an assertion of the form *(subject, predicate, object)*. Each RDF statement in the current state model represents an atomic assertion about the state of the organisation. These assertions combine to constitute a set of facts about the organisation at the present moment in time.

Typically, the current state model will include a representation of the organisation's structure, its employees and their roles, a description of any working processes including their state, and descriptions (metadata) about knowledge assets (documents and contacts) in the possession of the organisation. The Pellucid generic ontology encoded in OWL provides the basic *vocabulary* for creating an RDF description of the organisation. Each domain specific ontology typically extends the Pellucid generic ontology to include vocabulary for describing domain specific aspects of the organisation.

The history of the organisation is also implemented as an RDF model. This model is called the *events model*. The events model stores a description of all the events that have occurred in relation to the business of the organisation, including for each event who or what was involved, and what the consequences where (i.e. what, if any, aspects of the organisation changed).

It is possible to represent an organisational event as three sets of RDF statements. The first set of statements is a description of the event itself, including the date and time at which it occurred, and who and/or what was involved in its cause and/or consequence. The second set of statements is the set of atomic assertions about the organisation that became true as a consequence of the event (*positive consequences*). The third set of statements is the set of atomic assertions about the organisation that were no longer true as a consequence of the event (*negative consequences*).

As each event is stored in the events model, RDF reification of statements is used to make assertions about which statements are the positive and negative consequences of the event being added.

To support historical analysis of events, it is necessary to be able to recall information about the state of the organisation immediately before each event occurred. However, storing a complete copy of the current state model with each event would be impractical. Therefore, each event is stored with a small set of *context features*, which are those features of the organisations state immediately before the event occurred that are deemed to be relevant to the occurrence of that event. The context features, as with the positive and negative consequences, are implemented as a set of reified RDF statements.

The Jena 2.0 Semantic Web Framework [6] was used to generate and manipulate the RDF models of the OM.

6.2 Organisational Dynamics

Generating Events. In the Pellucid agent architecture, the Monitoring Agents (MA) collect information about the state of the organisation by monitoring the organisation's applications, process and databases. The information received by an MA from its sensors is *interpreted*, i.e. is converted into an RDF description of an organisational event. This RDF description is then serialised, and passed as a message to a Role Agent (RA).

An example of a serialisation of an RDF description of a simple organisational event is presented below. This is an event in which an employee opens a document using MS Word2000. The single (positive) consequence of this event is that the employee ('Pete') has an open file on his desktop. Events relating to use of documents can be a valuable resource, because knowing what documents are useful in what context is an important element of experience.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
        <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
        <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
        <!ENTITY pel "http://www.clrc.ac.uk/pellucid/ontology/owl/generic/1.0">
        <!ENTITY clrc "urn:clrc/">
]>
<rdf:RDF xmlns="&pel;#" xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;" xml:base="&pel;">
        <EventFileOpened>
                <occurred>15-Dec-2003 12:50:46</occurred>
                <involvedActor>
                        <User rdf:about="&clrc;user/Pete"/>
                </involvedActor>
                <involvedFile>
```

```
                                <ComputerFile rdf:about="file:///folder/guide-v03.doc"/>
                        </involvedFile>
                        <involvedApplication>
                                <Application rdf:about="#MSWord2000"/>
                        </involvedApplication>
                        <add>
                                <rdf:Statement>
                                        <rdf:subject rdf:resource="&clrc;user/Pete"/>
                                        <rdf:predicate
                                                rdf:resource="#userOpenFile"/>
                                        <rdf:objectrdf:resource=
                                                "file:///folder/afile.doc"/>
                                </rdf:Statement>
                        </add>
                </EventFileOpened>
</rdf:RDF>
```

The RA receiving this message parses it into an RDF model. Role Agents have responsibility for event handling, which is the process by which the OM is updated.

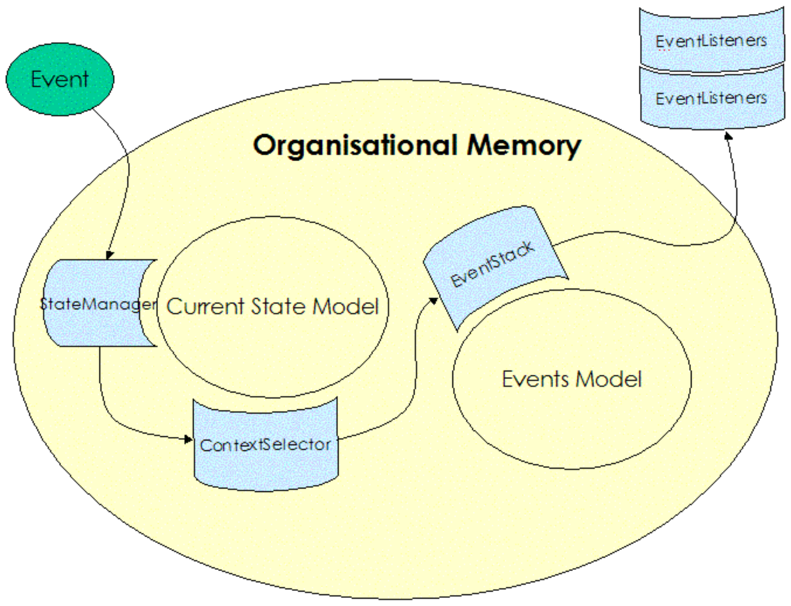

Fig. 2. Event handling

Event Handling. Fig. 2 illustrates the stages involved in event handling.When an RA receives an event, it invokes a fixed set of *event handlers*. The first event handler is the *state manager*, the second is the *context selector*, and the third is the *event stack*. After being processed by these handlers, the event is then passed on to any further *event listeners* that have been registered with the OM.

The *state manager* uses the positive and negative consequences of an event to update the current state model. Quite simply, the statements representing the positive consequences are added to the current state model, and the statements representing the negative consequences are removed from it.

As mentioned above, each event is stored along with a set of *context features*. Because the set of relevant context features will depend heavily on the *type* of the event, each event type is defined with its own *context selector* properties. Each of an event type's context selector properties defines an RDF query that, when applied to the current state model, will extract the relevant context features for that type of event.

The *event stack* receives and stores all events in the events model. It orders all events according to the date and time at which they occurred, to support historical analysis functions.

An *event listener* is any object interested in being notified on the reception of an event by the OM. Any number of event listeners may be registered with an OM.

Firing Active Hints. Role Agents also build and maintain *hint engines*, which trigger the creation and delivery of active hints to users when the user enters an appropriate working context. An OM may have any number of hint engines bound to it at any time. This allows for different underlying mechanisms of hint triggering to be used concurrently.

The *rule-based hint engine* uses forward rules to trigger hints when a particular set of statements occurs in the current state model. The forward engine used is the Jena2 ReteReasoner.

The *event-based hint engine* uses the reception of certain event types to trigger hints. It stores a set of mappings from event patterns (implemented as RDQL queries) to hint types. Thus, when an event matching a particular pattern is received, the corresponding hint is triggered.

After a hint is triggered, it is built and then sent to the Personal Assistant Agent of the appropriate user.

7 Related Work

Pellucid can be seen as an example of an Electronic Performance Support Systems (EPPS) [7], systems that aim to support and enhance users' performance by providing them with the knowledge required by the task they are performing at the time they are actually performing the task. Other examples of EPPS are the VirtualOffice and KnowMore systems [1] and integration of knowledge and business processes [12].

The VirtualOffice and KnowMore projects aim to support knowledge-intensive activities by providing automatic access to relevant information [1]. Each activity belongs to some comprehensive business process which is explicitly modelled and enacted by some WfMS. The activities are to be supported based on an available information space, which contains information sources of various types and characteristics together with suitable access structures. A central component is an intelligent assistant, which bridges between the information space and the knowledge-intensive activities and performs a process identification job (similar to the context identification job carried out by Pellucid) in which the system detects the particular circumstances of a process. The Virtual Office tool integrates paper-based information into workflow using a document management system for information extraction, following a request from the workflow. The Know-More project focuses on delivering goal-specific information in a proactive way by analysing the incoming stream of information that the organisation receives. These works were further extended in the DECOR project [2].

The work of Staab and Schnurr in [12] is close to our work in putting an intelligent assistant to work within a business process environment. It also exploits the inferential power of ontology-based retrieval on top of the Ontobroker software, using a notion of context-based views for coupling workflow and retrieval. In building the information system, they start with an analysis process that explores the interdependence among the documents employed in the business process. Then, domain-specific information is added by including domain ontologies describing the content of documents, and contextual information. During the execution phase, the system acts as facilitator for sharing, creating and retrieving knowledge, providing users for active help as a response to their request.

One of the influential works in the definition of the Pellucid experience management model is [5]. Bergmann’s experience model consists of a *Knowledge Kernel*, a *Problem Solving Cycle*, and a *Development and Maintenance Methodology*. The *Knowledge Kernel* contains the experience base and the reuse-related knowledge as well as the vocabulary on which both are based. The *Problem Solving Cycle* describes problem solving that is supported by experience reuse; it includes steps such as problem acquisition, experience evaluation and retrieval, experience adaptation, and experience presentation. The *Development and Maintenance Methodology* addresses the acquisition and maintenance of the knowledge in the knowledge kernel as well as the technical, organisational, and also managerial aspects of the problem solving cycle and its implementation. Bergmann’s Knowledge Kernel corresponds to Pellucid’s organisational memory. Some actions of the Capture and Store phase are also linked to the kernel. The Problem Solving cycle is associated with Pellucid’s Analysis and Presentation phase, although Bergmann’s cycle is more general but strongly linked to case-based reasoning.

8 Conclusions

This paper has presented the application of Semantic Web technologies in the development of the European research project Pellucid. In particular, we describe the system architecture and the way active hints are generated, based on the information stored in the organisational memory.

One of the strengths of the Pellucid platform is its adaptability: some parts are directly reusable from one application to another (for example, the basic hint firing mechanism), others are reusable with adaptation (the ontologies representing working context), while others may be developed specially for the needs of the application (the user interface). We believe that such a high grade of adaptability has been possible due to the adoption of Semantic Web technologies.

One of the final important tasks of the project will be to define more precisely the knowledge evolution phase of the experience management cycle and to define to what extent Semantic Web technologies can assist in this process.

Acknowledgements

This work was supported by the IST Programme of the European Commission, Project number IST-2001-34519. The authors are grateful to other project partners for helpful comments. Pellucid partners comprise ACC CYFRONET-AGH (Poland), Institute of Informatics of Slovak Academy of Sciences (Slovakia), Sadiel S.A. (Spain), Softeco Sismat SpA (Italy), Consejeria de la Presidencia de la Junta de Andalucia (Spain), Comune di Genova (Italy), MMBG (Spain), and SADESI (Spain).

References

1. Abecker, A., Bernardi, A., Maus, H., Sintek, M.,Wenzel, C.: Information Supply for Business Processes: Coupling Workflow with Document Analysis and Information Retrieval. *Knowledge-Based Systems* 13(5) (2000) 271–284
2. Abecker, A., Dioudis, S., van Elst, L., Houy, C., Legal, M., Mentzas, G., Muller, S. Papavassiliou, G.: Enabling Workflow-Embedded Organizational Memory Access with the DECOR Toolkit. In [8] 63–14
3. Arenas, A., Lambert, S., Miles, A.: Engineering Knowledge-Intensive Tasks in Public Organisations. EU-LAT Workshop on E-Government and E-Democracy, Chile (2004)
4. Bauer, T., Leake, D.: Exploiting Information Access Patterns for Context-Based Retrieval. Proceedings of the 2002 International Conference on Intelligent User Interfaces, IUI-01, ACM Press (2002)
5. Bergmann, R.: *Experience Management*. Lecture Notes in Artificial Intelligence, Vol. 2432. Springer (2002)
6. Carroll, J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: Implementing the semantic web recommendations. Technical Report HPL-2003-146, Hewlett Packard Laboratories (2003)
7. Cole, K., Fisher, O., Saltzman, P.: Just-in-Time Knowledge Delivery. *Communications of the ACM* (1997) 40(7)
8. Dieng-Kuntz, R., Matta, N. (eds.): *Knowledge Management and Organizational Memories*. Kluwer Academic Publishers (2002)
9. Goldberg, A.: Information Models, Views, and Controllers. *Dr. Dobbs Journal*, July 1990
10. Kitowski, J. *et al.*: Model of Experience for Public Organisations with Staff Mobility. Knowledge Management in Electronic Government, KMGov 2004. Lecture Notes in Artificial Intelligence, Vol. 3035. Springer (2004).

11. Lambert, S. et al.: Knowledge Management for Organisationally Mobile Public Employees. Knowledge Management in Electronic Government, KMGov 2003. Lecture Notes in Artificial Intelligence, Vol. 2645. Springer, (2003) 203–212
12. Staab, S., Schnurr, H.: Knowledge and Business Processes: Approaching an Integration. In [8] above