

Knowledge Management for Administration Processes *

Michal Laclavik¹, Zoltan Balogh¹, Ladislav Hluchy¹, Krzysztof Krawczyk²,
Mariusz Dziewierz², Jacek Kitowski², Marta Majewska²

¹ Institute of Informatics, SAS, Dubravska cesta 9, Bratislava 84237, Slovakia
{laclavik.ui, balogh.ui, hluchy.ui}@savba.sk

²ACC CYFRONET AGH, Nawojki 11, 30-950 Cracow, Poland
{krafcoo, aristot}@icsr.agh.edu.pl

Abstract. In this paper we present the use of ontology for knowledge representation and handling for Administration Processes. Motivation has come from Pellucid (Platform for Organizationally Mobile Public Employees) is European Project IST-2001-34519, where we need to capture and capitalize employee's knowledge in an organization. This knowledge is then presented to other employees as they work on particular tasks. The Pellucid System is built on Multi-agent system, thus ontology is used also for communication between agents and for knowledge representation as well.

1 Introduction

Motivation for this article has come from the Pellucid project. Pellucid (Platform for Organizationally Mobile Public Employees) is European Project IST-2001-34519. The Pellucid System is particularly aimed to capture, record and capitalize the knowledge of current employees about their work in an organization [1]. This knowledge is then presented to other employees as they work on particular tasks.

Human knowledge is based not only on facts which are true or false but also on uncertain knowledge which is true or false partially. Several methods can be used to represent such knowledge, e.g. probability measures, fuzzy logic or computing with words [4]. Some methods are known to represent uncertain knowledge even in agent systems by e.g. extended FIPA-SL language; however, uncertain knowledge is still quite complicated and not understandable especially for the agents themselves. When using uncertain knowledge or knowledge where true and false facts are not strongly defined, computer systems cannot discover new facts in existing knowledge base using logical operators. This is known as a fundamental problem of contradictory knowledge in computer systems[5].

This is why knowledge base in the Pellucid consists only of strongly true facts. Such facts are structured and defined by ontologies. Using this solution, it is easier and more straightforward for agents to understand knowledge and to discover new

* This work was supported by EC Project Pellucid 5FP RTD IST-2001-34519 and Slovak Scientific Grant VEGA 2/3132/23

knowledge from existing one. One could say that using such knowledge is not sufficient for such applications. By evaluation of administration application and pilot sites of the Pellucid project we conclude that for application where administration business processes are well defined it is reasonable and useful to use knowledge based on facts rather than on uncertain knowledge [6].

1.1 Pellucid Modules

In this section we briefly describe generic modules of the Pellucid System. Such modules are generic for any knowledge system dealing with administration processes. This description of functionality is needed for better understanding of described modules. Generic version of the Pellucid system has two modules:

- Intelligent Contact Management Module
- Intelligent Document Management Module

List of contacts is presented in each organization in a form of contact database, searchable by keywords with sorting ability, etc. Intelligent Contact Management Module provides a user with an intelligent contact list related to the activity, which is performed by a user. Organizational repositories are available in each organization. The Pellucid tries to relate documents to an activity, which is performed by a user and this is a duty of the Intelligent Document Management Module. Each module uses its ontology to define the relations among knowledge entities such as activity, document, or contact, and uses the common techniques to capture, capitalize and return knowledge to a user. [6].

2 Ontology

In this chapter we present knowledge model based on ontology, which covers a generic organizational, task and agent model influenced by CommonKADS methodology.

Knowledge management involves different resources (fig. 1), which are present in an organization. In figure 1 you can see a resource structure defined by ontology, which includes workflow related resources, documents, contacts (BusinessEntity) or human resources. Resources are then connected with other ontology elements such as an organization, experience or skill. Activity represents task performed by an actor in an organization, we will focus mainly on workflow activities (WfActivity) because such activities and its relation can be handled and monitored.

Figure 2 presents the main part of the Pellucid ontology model “Active Hint” and its relations. In ontology an active hint is defined. In order to query and create active hints we need to capture some “Events” about “actions” in an organization. By action we mean something preformed by an actor (an employee, a computer system or other entity) which can be captured. The Event is something what we can capture by computer system. Event can be “the document used”, “the person contacted”, “workflow

activity performed” or “the email received”. Thus the Event is modeled by “the employee” performing an “the action” on particular “the resource” in particular time/context (“wfActivity” and “wfInstance”). We are capturing events in the workflow context only because this is only a context we can relate to such activities. Note that the Event and Active Hint structure is very similar. Events help employee to create Active Hints easily during their work.

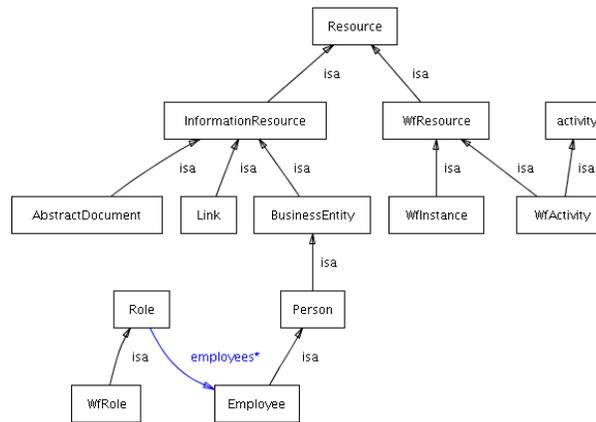


Fig. 1. Ontology Model - Resources

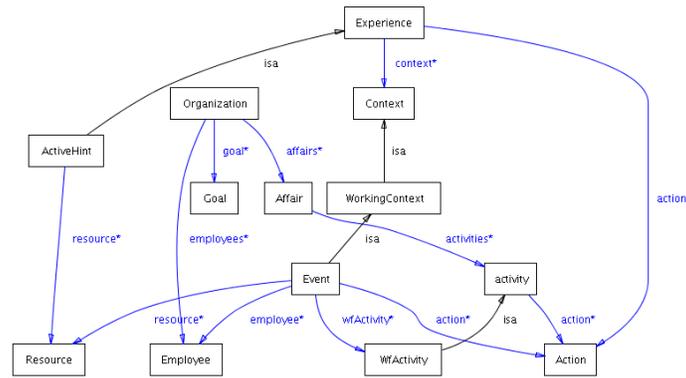


Fig. 2. Ontology Model – Active Hint, Event

In order to present an active hint to an employee, we need to identify a work context of the user. Administration Based Processes (workflow) is presented in public organizations and it can be well represented by the Workflow Management System (WfMS). The WfMS can provide the Pellucid with the important part of user work

context – workflow activities, roles, workflow process instances. The Pellucid provides a user with related Active Hints as a user works with the WfMS on particular activities.

2.1 Domain Specific Ontology

The Developed ontology needs to be customized/extended for particular application domain to fulfill knowledge management for certain application area. Such extension is described in this chapter. We verified this model on administration process of installation of traffic light application (Cdg) and ontology will be further verified on other 2 application areas in pellucid project.

The Pellucid can provide an employee with useful Active Hints only if some domain (application) specific data/information/knowledge will be available. Below there is a list of some domain specific entities which are modeled in domain ontology. It is always a problem to monitor domain specific work context. In Cdg application, most of domain specific content can be taken from an initial request for installation of traffic light. List of Domain Specific Entities is presented below [11]:

- Traffic Light = Workflow Process Instance
 - Installation request (as a document) - Sender (contact), Location, Problem description (finding similarity with same problems)
 - Crossing Location - Street names, X, Y co-ordinates
 - Crossing typology (shape) - Predefined shapes assigned to each dossier
- BusinessEntity
 - Address(District should be included)

There are more domain specific entities related to each Workflow Activity. For example “Economical Evaluation” involves different important domain specific terms/entities than “Technical Evaluation”, but this is not yet implemented in our model and will be part of future work.

Ontology is developed the way that we need to extend mainly workflow process instance with several domain specific properties and also extend some resource domain specific properties.

3 Implementation

The Pellucid ontology is developed by Protégé Ontology editor [12] with DAML+OIL plug-in [13]. Ontology is implemented using DAML+OIL [7]. HP Jena library [14] is used for basic manipulation of DAML+OIL ontology model and for storing and accessing knowledge. Agents are implemented using JADE Agent platform [8] based on Java. Architecture can be connected to Workflow Management System by XML-RPC interface. Architecture can plug-in any type of user interface.

Data for GUI are in XML format. We have developed web based user interface where XML data are parsed XSLT and it outputs as HTML.

3.1 Query Engine

Ontology can be partially understood as a structure of an object oriented database. Data stored in this database are then ontology concepts instances. The main idea of the Pellucid is to return appropriate Active Hint instances (representation of experience) in an appropriate work context.

$$\begin{aligned} \text{Experience} &= \text{Action} + \text{Context} \\ \text{Active Hint} &= \text{Action} + \text{Resource} + \text{Explanation} + \text{Context} \end{aligned}$$

The Pellucid query engine is very general. It queries all Experience Instances where a work context matches the current work context. Since Active Hint is representation of Experience in our model, only Active Hints instances are presented in model and thus Active Hint instances are returned. Work context is represented by workflow activity, workflow instance or events. Thus query engine matches such context instances, which are presented in the model.

Context instances and their properties are compared to a current work context which is presented by current workflow activity, current workflow process instance and its domain specific properties. For example, installation of traffic lights (Cgd) on Red Rose Street represents Process instance with domain properties such location, an initial request for installation or a shape of the crossing. Thus query engine can return active hints for appropriate general (workflow related) work context and also domain (application) specific work context.

Active Hints involves resources. Relevant resource can always be the same in the active hint, e.g. if we offer document, which is a template, but a resource can be defined also by a context related query for example if we offer appropriate contact to the Police from relevant district where the traffic light is located. Resources are queried according to the current work context by query engine, they are attached to the active hint and return to the employee.

3.2 Active Hint Creation

By modeling and capturing Events in the Pellucid we are able to help user in creating a useful active hint. Otherwise every hint would have to be fully manually entered by an expert. Our approach is manual with assistance of the system.

The user clicks that s/he wants to add a hint in a particular activity. The user can choose an action from a select box or type a new action if necessary. The user chooses a resource type (document, contact, link or workflow process instance) and searches for a proper resource instance(s) by a keyword e.g. The user chooses the concrete resource to attach with the action. The Pellucid will check if the chosen resource is somehow related to the activity or the workflow process instance. The Pellucid can check all events related to the resource and wfActivity and wfInstance. Based on an

analysis, the Pellucid will ask the user how a resource should be described and it will create sentences based on discovered resource relations, see an example:

- Do you want to attach this resource to active hint?
- Do you want to attach all resources which were created in same activity?
- Do you want to attach all contacts contacted in the previous activity in the current process instance?
- Do you want to attach resources containing the same keyword as you searched?

The user checks one or more options and based on this the Pellucid will create query for a resource related to the active hint. The user then types explanation of the active hint and submits it.

Example:

The user gains new experience that in the final activity in our application s/he needs to create report based on all documents created during the process. Thus s/he wants to add an active hint offering all such documents. The user clicks that s/he wants to add an active hint. The user chooses the action from the select box. The user chooses a type of a resource document and finds all documents created in this process instance. The Pellucid will find possible relations common for all the documents related to wfActivity and wfInstance. The Pellucid will offer user the following questions:

- Do you want to attach these documents to active hint?
- Do you want to attach all documents created during current XY workflow process instance?
- Do you want to attach all documents created during active workflow process instance?

If the user chooses the first question the active hint will always return the same documents as found by the user, which is not what was mentioned in this active hint. If the user selects the second question the active hint will always return the same documents as found by the user and maybe some more created in that particular wfInstance. If the user selects the third question, the Pellucid will store query, which will always return the document created in the current employee's process, such active hint will be useful for both, the experienced and inexperienced employees. After defining the resource, the user types explanation "this are documents created in this process and they are useful for creating the final report" and submits the hint.

3.3 Active Hint Examples

This chapter gives two examples of active hints. The active hint is structured as follows:

- An action in examples is written in *italic* font
- A resource in **bold**
- An explanation in regular font.

Active hints can be composed by operators as AND, OR, XOR or THEN.

Contact **Police** in same district because you need to send design for evaluation AND *Send* **Design document** to the police because you need to send design for evaluation

- **Police** – can represent more than one contact but it should be defined by query because we need to return contact from certain district: “TRAFIC_LIGHT.DISTRICT = CONTACT.DISTRICT”
- **Design Document** is a document from the previous activity: “Select document where eventCreated.wfactivity = design and evenCreated.processinstance.id = processinstance.id”
- Context: Workflow activity – evaluation

See **Map**, if a poster is near traffic lights THEN

Contact **Municipality of Genoa**, if a poster is near traffic light THEN

Send **Request** to move it.

- **Map** is a link to the Genoa map but it can be also a link with extended variables with street values of current traffic lights so it is useful for the experienced employee as well because it will show him/her the map of location and s/he will be able to see a poster, if any.
- **Municipality of Genoa** is the contact of the Genoa Municipality Advertisement Department
- **Request** is the document template for such request

4 Conclusion

In this paper we present design and implementation of architecture for experience management in a public organization. Influenced by the CommonKADS methodology, we have created the model of knowledge in a public organization. Based on this model we have developed the general ontology, which is further extended with the domain/application specific elements. We give examples of the domain specific extension for the Installation of Traffic Light. The architecture uses an ontology model to query knowledge. Knowledge is presented to the users as spontaneous suggestions (Active Hints) based on the work context of the user. Most of public organizations deal with well defined administration processes - workflow, so the architecture uses a workflow as a basic work context of the user. Developing of knowledge management systems involves lot of effort on modeling of a problem domain. We have modeled an application domain, where administration processes are presented and we have developed the architecture to manage such a model. When customizing the Pellucid for a particular administration application, customization effort is still required but one can build on the core of the Pellucid architecture. We believe a similar approach with some modification can be and will be used in next generation of knowledge management systems in the commercial area as well.

References

1. Pellucid Consortium: Pellucid Project Technical Annex (2001)
2. Guus Schreiber at all.: Knowledge Engineering and Management – The Common-KADS Methodology, MIT Press (2002)
3. FIPA: FIPA-SL Content Language Specification, <http://www.fipa.org/> (2000)
4. Paul P. Wang (Editor): Computing with Words, ISBN: 0-471-35374-4, (2001)
5. Michael Wooldridge: Introduction to MultiAgent Systems, ISBN: 047149691X, (2002)
6. M. Laclavik, Z. Balogh, L. Hluchy, R. Słota, M. Dziewierz, K Krawczyk: Distributed Knowledge Management based on Software Agents and Ontology. PPAM Conference (2003)
7. DARPA, DAML Website, <http://www.daml.org/> (2002)
8. Telecom Italia Lab: JADE website, <http://sharon.csel.it/projects/jade/> (2002)
9. M. Laclavik, Z. Balogh, L. Hluchy, G. T. Nguyen, I. Budinska, T. T. Dang: Pellucid Agent Architecture for Administration Based Processes, IAWTIC 2003, Vienna (2003)
10. K Krawczyk: Pellucid Organizational Model, (2003)
11. Pellucid Consortium, Lead partner Sadiel: Deliverable D4 - Environment and Requirements Analysis
12. Stanford University: Protégé Ontology Editor, <http://protege.stanford.edu/> (2003)
13. Grit Denker: Protégé DAML+OIL plug-in, SRI International (2003)
14. HP Labs: Jena Library, <http://www.hpl.hp.com/semweb/> (2003)