Support of Knowledge Management in Distributed Environment

Ján Paralič*, Marek Paralič**, Marián Mach*
*Dept. of Cybernetics and Artificial Intelligence, **Dept. of Computer Science,
University of Technology in Košice, Letná 9, 042 00 Košice, Slovakia
Phone: +421 95 602 4128, Fax: +421 95 62 535 74
[paralic,paralicm,machm]@tuke.sk

Keywords: knowledge management, knowledge modelling, mobile agents, agent collaboration, e-Democracy

An original approach to support of knowledge management within an organization is presented in this article. This approach has been designed, implemented in form of the KnowWeb system and tested on various pilot applications. Special attention is paid to organizations with distributed environment. For this purpose an experimental system for support of mobile agents that combines the power of high-level distributed programming with the mobile agent paradigm has been proposed and is presented here as well. Finally, experiences from two of the KnowWeb pilot applications as well as further application possibilities in the area of e-Democracy are sketched.

1 Introduction

Knowledge can be simply defined as actionable information (Tiwana 2000). That means (only) relevant information being available in the right place, at the right time, in the right context, and in the right way.

The knowledge life cycle defined in (Nonaka & Takeuchi 1995) hinges on the distinction between *tacit* and *explicit* knowledge. Explicit knowledge is a formal one and can be found in documents of an organization: reports, articles, manuals, patents, pictures, images, video, sound, software etc. Tacit knowledge is personal knowledge given by individual experience.

A considerable amount of explicit knowledge is scattered throughout various documents within organizations. It is quite often that this knowledge is stored somewhere without being retrieved and reused any more. As a result, most knowledge is not shared and is forgotten in relatively short time after it has been invented or discovered. Therefore, it has become very important for advanced organizations to make the best use of information gathered from various document sources inside companies and from external sources like the Internet. On the other hand, tacit knowledge of the documents' authors provides important context to them, which cannot be effectively intercepted.

Knowledge management (KM) generally deals with several activities that appear in knowledge life cycle (Abecker et al. 1998): identification, acquisition, development, dissemination (sharing), use and preservation of organization's knowledge. From these activities, dissemination (sharing) is crucial. Knowledge that does not flow does not grow and eventually ages and becomes obsolete and useless. By contrast, knowledge that flows, by being shared, acquired, and exchanged, generates new knowledge (Borghoff & Pareschi 1998).

Our approach to knowledge management supports most of the activities mentioned above. Based on this approach, KnowWeb[1] toolkit has been designed, implemented and tested on 5 pilot applications. Firstly, it provides tools for capturing and updating of tacit knowledge connected with particular explicit knowledge inside documents. This is possible due to ontology, which is used for representation of organization's domain knowledge. Section 2 describes in more detail domain knowledge modelling in general and our approach to it in particular.

Secondly, intelligent retrieval is enabled making use of both kinds of knowledge linked together within the organisational memory. How organisational memory in our approach is organised and what functionality it offers presents section 3.

As next, efficient communication and support for distributed groups to share knowledge and exchange information efficiently is provided (section 4). For these purposes an experimental framework for mobile agents has been designed and implemented and is introduced in section 5. Experiences from two of the KnowWeb pilot applications as well as further application possibilities in the e-Democracy context are sketched in section 6.

2 Domain knowledge modelling

2.1 General

[1] EC funded project ESPRIT 29065 "Web in Support of Knowledge Management in Company (KnowWeb)"

Theoretical foundations for the research of domain modelling can be found in the works (Chandrasekaran et al. 1999; Gruber 1993; Wielinga et al. 1997), and others on ontologies and knowledge modelling. Ontology is a term borrowed from philosophy where it stands for a systematic theory of entities what exist. In context of knowledge modeling, Gruber introduced the term ontology as a set of definitions of content-specific knowledge representation primitives consisting of domain-dependent classes, relations, functions, and object constants. The ontology represents formal terms with which knowledge can be represented and individual problems within a particular domain can be described. Ontology in this sense determines what can 'exist' in a knowledge base. Chandrasekaran understands ontology as a representation vocabulary typically specialised to some domain. He suggests basically two purposes why ontologies may be used:

- to define most commonly used terms in a specific domain, thus building a skeleton,
- to enable knowledge sharing and re-using both spatially and temporally - see also (Motta & Zdrahal 1998).

Ontology with syntax and semantic rules provides the 'language' by which KnowWeb(-like) systems can interact at the *knowledge level* (Newell 1982). Ontology allows a group of people to agree on the meaning of few basic terms, from which many different individual instantiations, assertions and queries may be constructed.

Once there is a consensus on understanding what particular 'words' mean, knowledge represented by these words can be adapted for particular purposes. Knowledge must be defined unambiguously because different people in the organisational structure of an organization need to use them with the same meaning. Thus, it is possible to re-use and share the knowledge thanks to understanding of its representation.

Common understanding of the meaning of notions used in a given domain (the understanding may be domain-specific) results in the definition of *concepts*. Concepts are more or less abstract constructs on which a relevant part of the world is built, or better, which can be used to describe this relevant part of the world. Since concepts can differ in their character, several types of concepts, namely *classes*, *relations*, *functions* or *procedures*, *objects*, *variables* or *constants*, can be distinguished. These primitive constructs can be represented differently in different applications but they have the same meaning in all applications – i.e. when someone wants to communicate with somebody else, he/she can do it using constructs from the ontology of shared concepts.

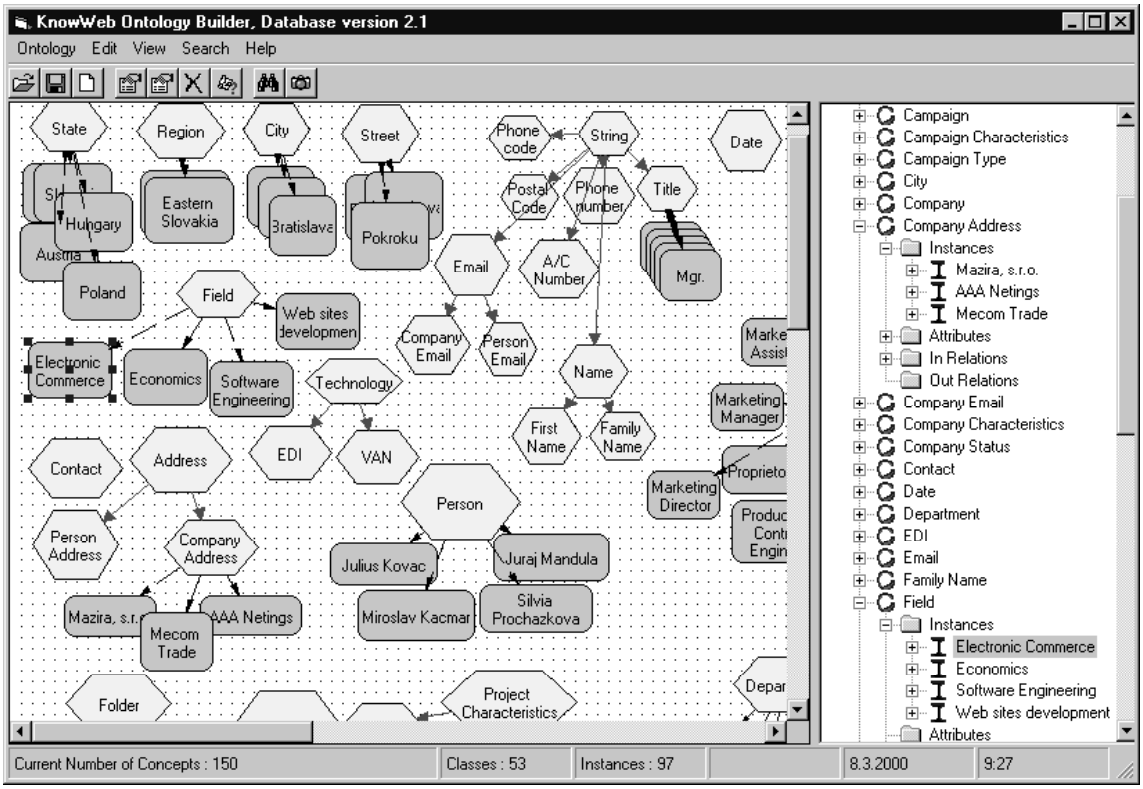

Figure 1. Sample of a domain knowledge model (ontology).

The concepts create usually a very complicated hierarchical, network (or tree)-like structure. However, even a complex structure covers only a specific part of the world, e.g. a narrow world of an organization and its activities. This structure models the world from a certain point of view. And here emerges the notion from the title

of this section – *Domain Knowledge Modelling*, as the concepts are usually highly domain-dependent or subject-dependent, and can be meaningfully used only in the frame of the particular domain. In other words, what is acceptable and important, for example, for a property management company may be not suitable for a company dealing with distance delivery of educational courses.

2.2 Our approach

Based on the needs analysis of several pilot applications, two types of concepts have been identified as necessary and satisfactory as well. They can be either generic (type *class*) or specific (type *instance*). Both of them have attributes. Concepts and relations among them are used to construct domain model. Formally, a relation in KnowWeb is an oriented link between two concepts. Two basic types of relations can be distinguished: *subclass_of* for relations between classes and *instance_of* for relations between classes and their instances. These two relation types enable inheritance of attributes and their values. The inheritance is an important mechanism for the development of a hierarchical ontology. Also multiple inheritance is supported i.e. a class concept can inherit its attributes from several parent class concepts. Figure 1 represents an example - a very small part of a domain knowledge model.

What shall be included in the domain model? The simple but vague answer is - everything what is relevant and important to describe a particular domain. In case of a company such model may conceptually describe the company specific concepts, such as its activities, projects, customers, employees etc., as well as relations among these concepts.

Each organization has some knowledge already gathered in the form of various databases and/or documents containing information about various technologies, products, customers, suppliers, projects or employees. Each company has usually some internal procedures how to perform specific tasks. Simply said – knowledge exists in an established environment. This knowledge is traditionally called organization's *goals* and *know-how*. From the knowledge modelling perspective a repository of know-how, goals etc. may be addressed as an *organisational memory* or a *corporate memory*.

3 Organisational memory (OM)

3.1 Conceptualisation and retrieval

The KnowWeb system enables author of any document to store his/her background knowledge together with the document attaching the relevant concepts from ontology – i.e. document is stored with its context. Context can be attached to the document as a whole or to a specific part(s) of a document called text fragment(s). *Text fragment* is a continual part of text within the document (e.g. a sentence or paragraph). In present version, the KnowWeb toolkit is able to process MS Word documents where no restrictions are given on text fragments and HTML documents where text fragment cannot cross any HTML tag. In order to cope with these documents the system provides a set of tools. They differ in their functionality but together they enable documents' authors and users to manage knowledge in a company in an easy and user-friendly way.

First, in order to place a document in the organisational memory, it is necessary to attach context knowledge (i.e. a piece of tacit knowledge) to it. This context can be in the form of a conceptual description (CD). By conceptual description is meant a set of links between a document (or its marked text fragments) and concepts in the domain knowledge model. Conceptual descriptions will enable to refer not only to explicit knowledge contained in the document but also to make use of tacit knowledge. In such a way easy sharing of knowledge in the future is enabled. The CD links can be created manually or semi-automatically User can select a text fragment and link this fragment to the domain knowledge model. The linking can be done directly to ontology concepts or to some template (for semi-automatic linking). Association links are of many-to-many type. It means that it is legal to link a document (or a text fragment) to several concepts and obviously, a concept can be linked to several documents (and or text fragments).

When a document with its description is available (after manual or semi-automatic linking within the KnowWeb system or after receiving the document with its description from outside and consecutive automatic linking), it can be incorporated into the organisational memory represented by a KnowWeb server. The description of this document is stored in the KnowWeb server as well. Another possibility is to store a conceptual description of a document without storing the document (the document can be located on other KnowWeb server in a distributed company or somewhere else, e.g. on Internet). If an author is not satisfied with the conceptual description of a document stored in the organisational memory, he/she has the possibility to modify it and subsequently upload the document with the modified description into the organisational memory.

The aim of storing document in the organisational memory is to access the right knowledge in the right time or situation. In order to express requirements on documents, which should be retrieved from the organisational memory, the user has to formulate a query. To formulate a query he/she can use concepts (or their attributes) from the domain model. They can be composed into a more complex structure using various operators (e.g. logical connections). In general, the concepts specified in the query will be used to search conceptual descriptions of documents.

3.2 Implementation

The *Conceptualisation tool* (*CT tool*) is built on the top of three modules, namely *DocView*, *OntoView*, and *TemplateEditor* (see Figure 2). The *CT tool* serves as an "envelope" or integrator for these modules. The *CT tool* is able to deal with two types of documents. First of all, it is possible to open a new document, which will be linked to the corresponding concepts in the domain knowledge model, and a copy of this document will be stored on the local KnowWeb server. Two kinds of association links are available for this kind of documents.

- Association links between the whole document and (a) specific concept(s) from the domain knowledge model
- Association links between a text fragment of the document and one or more concepts from the domain knowledge model

The same may be done for documents that are already stored on the local KnowWeb server and can be already (at least partly) linked to some concepts from the domain model. Existing association links can be edited/removed and/or new links can be added.

Another sort of documents are those, which will not be stored locally (e.g. documents accessed by remote retrieval function) and in principle can be located on any web server on Internet. We refer to the documents of this kind as *referenced documents*, because only their URLs are provided and serve as references to such documents. For this kind of documents, it is possible to define only the association links between a document as a whole and (a) concept(s) from domain knowledge model. The same applies for other then MS Word and HTML types of documents, where text fragments cannot be defined.

In order to create association link(s), the user uses the drag-and-drop functionality of the *CT tool*. It means that it is possible to "grab" a concept from the domain model, drag and drop it on a highlighted text fragment or header of a document currently opened in *DocView* window.

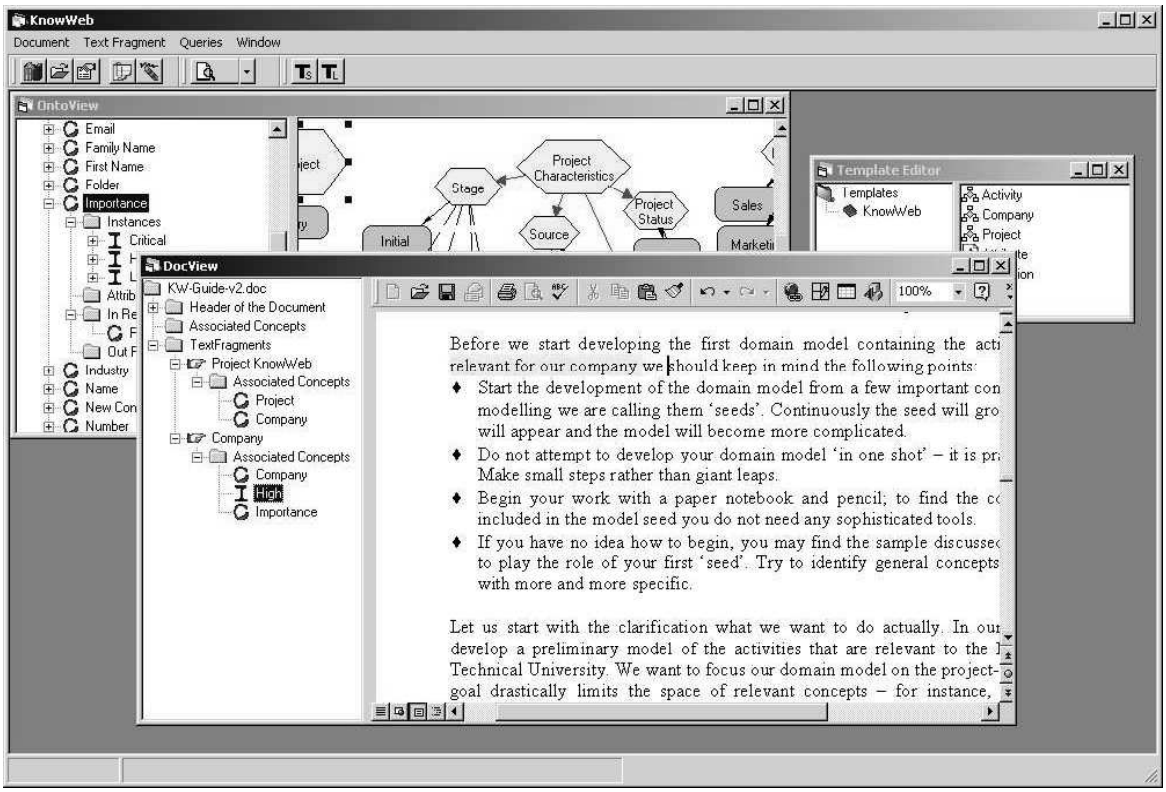

Figure 2. KnowWeb client interface to organizational memory.

The purpose of the *DocView module* is to provide users with possibility to preview documents with option of highlighting those text fragments, which are linked to the domain knowledge model. It also supports definitions and modifications to the annotation structure of the document, which results in the modified definitions of text fragments. Assignment of specific attributes to these text fragments is also possible.

The analysis of pilot applications by our industrial partners (e.g. application in the retail sector) identified a special requirement for "automatic" conceptualisation of documents with rigid structure (e.g. a structure of daily reports in a retail chain generated each day in each shop is fixed). This requirement is fulfilled by so-called "quiet mode" of the *CT tool*. In this case no visual component is

started and the documents are linked automatically to predefined concepts.

TemplateEditor module is a good supporting tool for both "quiet" as well as manual modes of *CT tool*. The purpose of this module is to give the users a tool for definition of special rules for selection of appropriate concepts. Linking a document (or its text fragment) to a template means that association links from this document (or text fragment) to all concepts resulted after application of particular template to present ontology will be created automatically.

In general, all documents have (possibly empty) headers, which represent various properties of the documents. For example, they can contain information regarding the document name, date and time of its creation, authors of documents, comments, etc. The set of attributes applicable for a header definition is application-dependent. Some properties may be compulsory while others are optional. Within a template three basic kinds of concept references are available.

- A concept can be *directly referenced* (only target concept must be given).
- A concept can be a result of an *"if – then" rule* application to values of document properties.
- A concept can be *referenced by a document property*. In this case, the name of a concept is determined by the value of a document property.

The first two kinds of references are especially useful for manual linking, but can be used for automatic linking as well. The last type is well suited for the purposes of the quiet mode of the *CT tool*, i.e. for automatic linking of whole documents. In this particular case, the target concepts are given indirectly and depend on the document property values as given in the document header.

Moreover, means for automatic creation of instances in the domain knowledge model (in quiet mode) are provided as well. This was another user-defined requirement.

4 Agent-based support for distributed organisational memory

As already mentioned in previous sections one of the most important requirements for success of OM is a support for the distributed environment of an organization. The OM should be flexible enough to fit different organization network settings and also it should remain open to the external sources such as Internet. The state-of-the-art solution to the first problem lies in the utilisation of the distributed objects. The most popular architectures for supporting distributed objects are *CORBA* (Common Object Request Broker Architecture) and *DCOM* (Distributed Component Object Model) (Pritchard 1999).

Objects in these architectures can capture the high-level logic (so-called business logic) of a distributed application and are accessible for processes outside the computer running them. Present implementations of these standards are based on commonly used communication protocols (e.g. TCP/IP). In the research

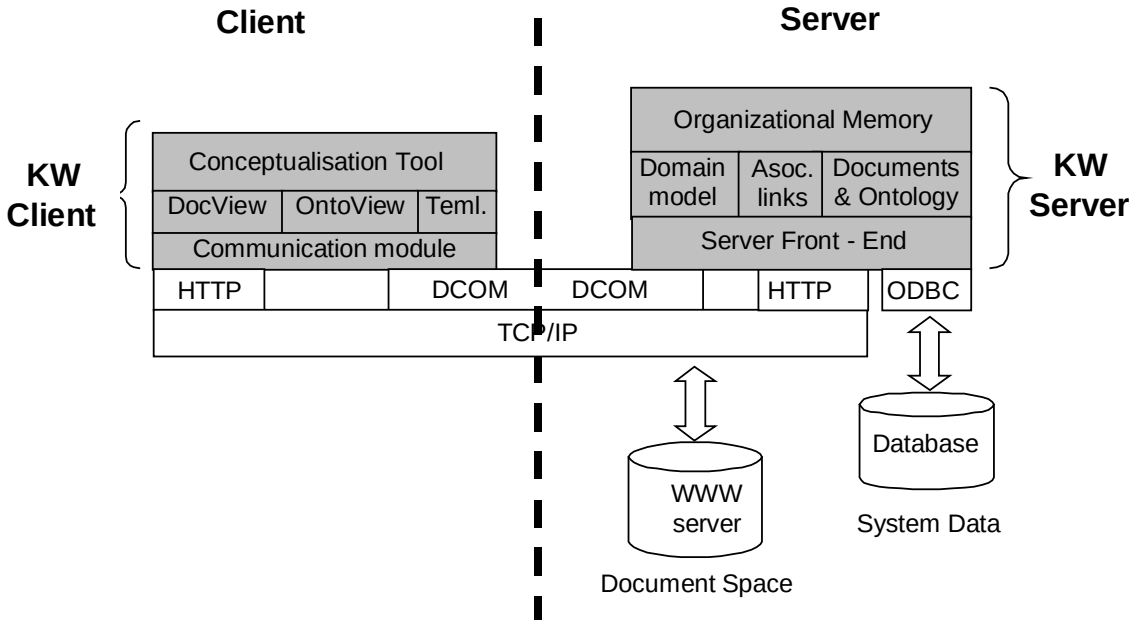

Figure 3. Distributed architecture of the KnowWeb environment – intranet solution.

prototype version of the KnowWeb toolkit a 3-tiered architecture was designed and implemented. We are distinguishing the following tiers:

- data source tier represented by the relational database,
- middle tier (called 'Server Front-End') that offers services to clients independently from the database implementation, and
- KnowWeb clients.

The relations among these three tiers are depicted in Figure 3. The document store and OM functionality is mapped to the KnowWeb server. Server offers key services – (i) acquisition and modification of the domain knowledge model state, (ii) association links of the documents, and (iii) retrieval in the document or ontology (domain model) space. All these operations are at the implementation level carried out as operations upon a relational database.

Solution as described above fulfils the requirement of flexibility in the corporate intranet settings and partially addresses the openness to the external sources. KnowWeb server is usually separated on an own local area network, contains (mostly one) domain knowledge model and typically serves many clients. The KnowWeb client can communicate with more than one KnowWeb server, potentially outside of the intranet scope.

However, the accessibility of KnowWeb servers outside the fast intranet networks assumes a reliable network connection with guaranteed throughput especially when users want to introduce new documents into OM and browse already introduced documents. Unfortunately, this is not the case with most portable computers (notebooks etc.) or organization branches connected through a low-speed dial-up network connection.

To solve the major problem with the throughput the mobile agent-based solution has been proposed and implemented. Mobile agent (MA) technology is perhaps the most promising paradigm that supports application design for dynamically changeable, networked environment with distributed information and computation resources. The most significant properties of mobile agents regarding their role are *autonomy* and *mobility* (Rothermel *et al*, 1997). MAs are autonomous because of their capability to decide which locations in a computer network they will visit and which actions they will take in these locations. This ability is embodied in the source code of every agent (implicit behaviour) or by the agent's itinerary, which can be dynamically modified (explicit 'orders'). Mobile agents can move between different locations in a network. A location is the basic environment for execution of mobile agents' code and therefore an abstraction of the underlying computer network and operating system.

Usual benefits of mobile agents are (i) reduced network load, (ii) overcome of the network latency, (iii) encapsulation of different protocols, (iv) asynchronous and autonomous execution, and (v) natural heterogeneity (Harrison *et al.* 1995; Lange & Oshima 1999). We claim that robust and scaleable OM systems can profit from these features; this will be especially visible in the companies with a world-wide and/or distributed structure. Modified distributed structure of the KnowWeb system is depicted on Figure 4.

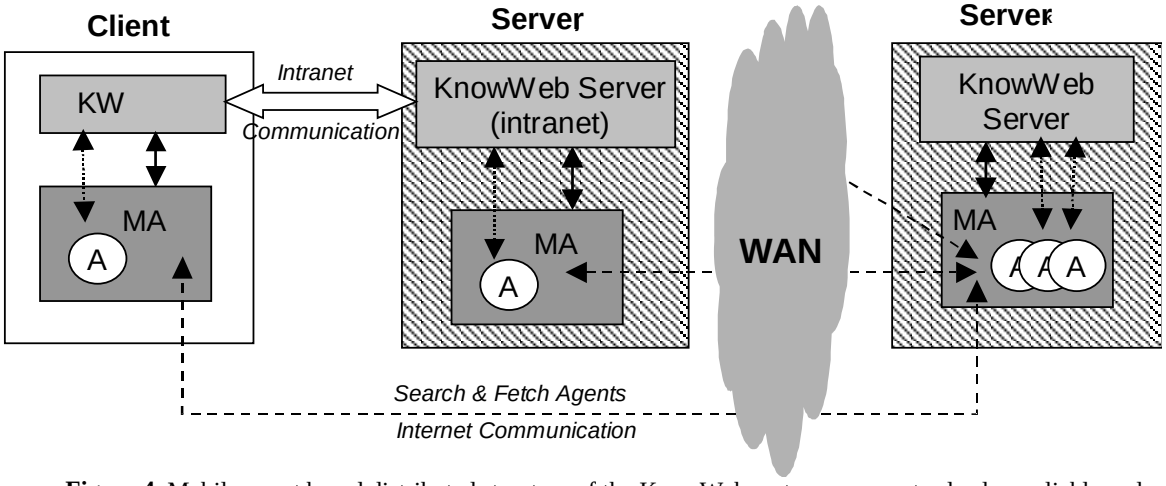

Figure 4. Mobile-agent based distributed structure of the KnowWeb system – supports also less reliable and low speed WANs

Dedicated mobile agents can do the most critical operations such as retrieval and gathering of documents at non-intranet servers. In retrieval operation client can formulate the query locally and either send an appropriate searching agent directly or demand the sending of an agent by a local KnowWeb server. The situation with gathering an already introduced document (accessible through other KnowWeb server) is similar. The main advantage of this functionality is the possibility for a client to get off-line for the period of operation

execution. By the next re-connect the results of ordered operations will be presented to the user. Such an approach brings significant savings to the communication between distributed KnowWeb servers containing a distributed OM.

To enable the work with mobile agents the *Mobile Agent Environment* (MAE) must be available on each concerned computer (client or server). MAE offers the following functionality: (i) creation of a mobile agent with a unique identity, (ii) transport of an agent, (iii) sending a message to an agent (possibly on another host), and (iv) getting the status information about any agent. MAE used in the research prototype of the KnowWeb toolkit is described in the following section.

5 Experimental framework for mobile agents

Basic functions of mobile agent environments (in today's mobile agent systems represented by agent servers) are identified by the Mobile Agent System Interoperability Facility (MASIF) (Milojcic et al. 1998) and include: (i) transferring an agent, which can include initiating an agent transfer, receiving an agent, and transferring classes, (ii) creating an agent, (iii) providing globally unique agent names, (iv) supporting the concept of a region, (v) finding a mobile agent, and (vi) ensuring a secure environment for agent operation.

A MA-based application usually requires to be programmed in two separate parts: mobile agent and location like *context* in Aglets (Oshima et al. 1998), *location services* in Ara (Peine & Stolpmann 1997), *place* in Gypsy (Jazayeri & Lugmayr 2000) or *service bridges* in Concordia (Wong et al. 1997). Mobile agent has a predetermined interface and restricted sources to communicate with on each visited location.

Mobile agent based distributed architecture of the KnowWeb system use the ESMA toolkit – developed and implemented at the Dept. of computers and informatics at the TU Košice (Paralič M. 2000). ESMA - the experimental system for support of mobile agents combines the power of high-level distributed programming with the mobile agent paradigm. As implementation platform the Mozart system (The Mozart Programming System) was chosen. It implements the Distributed Oz (DOz) programming language and offers simultaneously the advantage of a *True Distributed System* and the means for building a *Mobile Code System* (Picco 1998). In the following subsections mobile agent environment built from servers and its services and the methodology of how to build an agent-based application in our framework are shortly introduced.

5.1 Mobile agent environment

The basic functions offered by a mobile agent environment are transport and management of agents. In today's agent systems these services are offered by servers, which must be installed and running on every host computer that should be available for the mobile agents. Similarly our experimental framework in DOz offers basic functionality mentioned above.

MAE in the current implementation of the system can be started only once per host computer. Every agent created on a local MAE is a *home agent* for this MAE. On all other sites it will visit, it gains the status of a *foreign agent*. The MAE stores information about all its home agents and current available foreign agents in local database. The information about foreign agents is stored in the system only during time between the successful receiving from previous and successful sending an agent to the next host.

For the programmer of a mobile agent based application, the mobile agent environment is represented by the MAE module, which must be imported. Importing a MAE functor causes either the launching of a new environment with initialization values from persistent database for home and foreign agents and already connected other MAE servers or getting a reference to an already started MAE local server. This process is realized not only by launching a new local mobile-agent based application but also by resuming every incoming mobile agent. Thus, an agent gets access to the key services of the mobile agent environment. The possibility of dynamic loading and installation of first-class modules (Duchier et al. 1998) is thereby very important.

The communication between MAEs is realized in two layers: the first layer uses TCP sockets for exchanging of tickets for Oz ports. Oz ports then build a second, high-level communication layer, which can take advantage of Oz space data transparency. The Oz space offers the possibility to transparent copying of stateless entities and creating references for worldwide unique stateful entities. These possibilities can be fully utilized especially by the inter-agent communication.

5.2 Mobile agent based applications

Creating a MA-based application in the proposed framework is straightforward and requires only the following steps:

1. Identifying all fixed, not transferable resources needed by the application (i.e. their type) by means of abstract names and identifying parts of the transferable agent state.
2. Design and implementation of application-specific classes that are derived from the *MobileAgent* class and deal with agent state and other resources through their abstract names.
3. Designing and implementing an application that creates one or more instances of mobile agents, specifies their itinerary, sends them away, waits until they finish their jobs (or until the owner stops their work), and processes the results.

4. Design and installation of special environment modules (functors) in a compiled form. They map the abstract sources of MA to the real local resources of the host computer that should enable the execution of the MA based application.

5.3 KnowWeb and ESMA

To make the distributed structure of the KnowWeb system more flexible and suitable not only for fast intranet networks two special agent classes were created. The first one is *KW_SearchAgent* and the other one is *KW_FetchAgent*. Both mobile agent classes were proposed and implemented according to the methodology in section 5.2. The main task of the *KW_SearchAgent* is to get a query from the KnowWeb client and walk according to the travel plan of the KW servers in order to get the list of relevant documents – i.e. their exact addresses. Based on this list a KW client can send one ore more agents from the class *KW_FetchAgent*, which gather the whole document and return back with it.

At the client side are mobile agents created automatically after storing a query or fetching requirement in special form at local file system in predefined directory. At the server side *KW_FetchAgent* communicates directly with the WWW server that maintains the document space in the KnowWeb system. *KW_SearchAgents* store their queries at the local file system in predefined directory and are waiting for the answers, which can read also from the local file system.

6 Applications

6.1 KnowWeb pilot applications in BRD

Botnia Retail Data Inc. (BRD), located in Finland, has developed two of the five existing pilot applications based on the KnowWeb system. Main business activity of the BRD group is software development and consultancy for retail and industry. Their main product is WINPOS® - A Point-of-Sale (POS) software package.

In BRD felt that the best way to understand how to use the KnowWeb system is to first use the software within their organization with a domain knowledge model suitable for them. The second phase was the simulation of a retail environment with a domain knowledge model suitable for retail chains.

The database being used was an SQL Server database running under Windows NT to which documents are stored when they are finalised. Each user, who belongs to the personnel of BRD, uses Windows NT workstations on which KnowWeb clients are operating. The documents entered by a user shall typically be linked to a group of contexts. In order to quickly find the right contexts where to link the documents they have created a number of individual concept conditions. They are handling mainly documents of the MS Word or HTML types. They often scan in news articles and other printed material, which is first, pasted into a MS Word document and attached with explanations before the document is stored into the KnowWeb space.

The advantages identified in BRD after a couple of months of active KnowWeb using are the following.

1. **Internal efficiency improved**. They have been able to organise themselves much better than before. Now anyone in BRD can access relevant documents anytime without having to hunt for a document and wasting the time of his/her colleagues.
2. **Faster customer response**. Support personnel are now able to on-line check all information related to various WINPOS® versions and features.
3. **Exact and broader feedback to development**. They are tracing competitor information as well as feedback and feature requests from their end-users and dealers of their WINPOS® software product. Before they did not systematically trace this information. With the new system the development engineers are receiving much better information as a base for decisions about further enhancements to the product range.
4. **Improved marketing**. Through their systematic entering of competitors' information and their own marketing material as well, combined with the better feeling for what features customers are actually asking for, they have the feeling the system has helped them in improving their marketing activities.

On the other hand the following disadvantages have been reported:

1. **User interface** of the KnowWeb system prototype version seems to be too complicated to use for ordinary users.
2. There is a **time overhead in inserting the documents into the KnowWeb system**. During a busy day this is simply not being done.
3. The **success** of the introduction of the KnowWeb system to a quite large extent **depends on the discipline of the staff** using it.

Main motivation of the second BRD's pilot application was to structure all the events that within a retail chain cause disturbance to the present information system. These exceptions are in modern network environments most often reported in form of emails and documents sent between retail head office departments and the shops; today often without structure and forgotten after some time - thus leaving incorrect information in various databases as reports of present POS information systems. When planning a campaign or when making judgments e.g. regarding profitability on certain article groups, the marketing departments are often relying on historical information, which may go several years back. If the figures are unreliable, the decisions taken may be incorrect - the problem today is that many of the systems in the market today are not able to trace disturbances that have happened in the past.

Having this in mind, in BRD have linked their WINPOS® Point-of-Sales software to the KnowWeb system within the second pilot application. The end-of-day routines in the WINPOS® Point-of-Sales package (normally run at the end of the day when the shop has closed) are in this pilot application automatically generating shop-specific html documents, which are automatically introduced to the KnowWeb space. This means that the html-reports are automatically inserted in the Know-Web database and automatically linked to some predefined concepts via templates.

Moreover, certain date related information leads to dynamic generation of new instances in the domain model. As examples of documents being stored automatically we can mention for example shop-report, that is: sales, profit and payment media information summarised for a shop. Another example is the so-called department report, which contains sales amount, quantities and profit for the main article groups of the company. The advantages the customer will have from this system are:

1. **Internal efficiency improved**.
2. **Faster customer response**.
3. **Improved marketing**.
4. **Sales and profit analysing related to disturbances**. Automatically created and linked POS reports on one side are combined within the KnowWeb system with information about events causing various disturbances in retail chain (see above) providing realistic view on calculated numbers in context of occurred events.

6.2 Webocrat system

Another, very interesting application domain is electronic democracy (Paralič & Sabol 2001). An architecture of a Web based system Webocrat[2] is being designed with aim to empower citizens with innovative communication, access and voting system supporting increased participation of citizens in democratic processes and increase transparency and accessibility of public administration (PA).

Basic scheme of the proposed WEBOCRAT system is depicted in Figure 5. The system is based on knowledge modelling technology. Ontological knowledge models are employed in order to index all the information present in the system. Therefore first layer - *knowledge model* is depicted in the core of the scheme on Figure 5.

From the technical point of view, the system will be based on results achieved within the KnowWeb project focused on organising information using domain knowledge models. Their employment enables precise annotation of information based on its content, which results in efficient and powerful information retrieval capability.

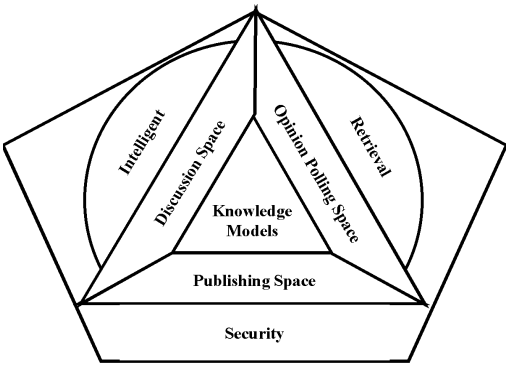

Figure 5. The principal scheme of the Webocrat system.

The second layer is represented by *publishing*, *discussion* and *opinion polling spaces* providing means for storing, updating and managing (in principle three slightly different types of) documents and their relations among themselves, as well as their relations to the knowledge model in the WEBOCRAT system.

The *Discussion Forum* (DF) module will support intelligent communication processes between public authorities, citizens and their elected representatives. DF will be responsible for documents in *discussion space*.

The *Web Content Management* (WCM) module will support publication of documents on the Internet, i.e. it will be responsible for documents in the *publishing space*.

The *Opinion Polling Room* (OPR) will enable electronic opinion polling featuring also with support for authentication and voter's privacy. OPR will be responsible for documents in the *opinion polling space*.

The third layer is composed from two retrieval-focused modules supporting retrieval capabilities that need only read access to the three above-mentioned spaces as well as to the knowledge model. The *Information Desk* (ID) will retrieve relevant documents of different types that are stored in the system.

The *Reporter/Summary* (REP) module will provide means for calculation of a variety of statistics as well as some more sophisticated approaches based on, e.g. data mining techniques. Moreover, this module will provide also alerting functionality, which has been required by user partners.

User registered in the system as an individual entity (i.e. not anonymous user) is provided with a personal access page ensuring him/her an individual access to the system. This page is built in an automatic way and can contain several parts. Some of them can be general and the other are person specific.

[2] EC funded project IST-1999-20364 Webocracy (Web Technologies Supporting Direct Participation in Democratic Processes)

The former can serve as a starting point for browsing all published documents accessible to the user, all conferences he/she is allowed to participate in, all running polls for which he/she is eligible, using search facilities of the system, read hot information, etc. The latter parts are devoted to user's personal newsletter, links to documents and conferences topics of which match the user's area of interest.

The personal access page hides division of the system into modules. Terms 'publishing space', 'discussion space', and 'opinion polling space' do not confuse users. The personal access page enables user to access all functionality of the system that he/she is allowed to access in a uniform and coherent way.

7 Acknowledgents

This work is done within the Webocracy project, which is supported by European Commission DG INFSO under the IST programme, contract No. IST-1999-20364 and project No. 1/8131/01 "Knowledge technologies for information acquisition and retrieval" supported by the Grant Agency of Ministry of Education and Academy of Science of Slovak Republic.

The content of this publication is the sole responsibility of the authors, and in no way represents the view of the European Commission or its services.

8 References

[1] Abecker A., Bernardi A. Hinkelmann K. Kühn, O. & Sintek M. (1998): Toward a Technology for Organizational Memories, *IEEE Intelligent Systems*, 13, May/June, p.40-48

[2] Borghoff U. M. & Pareschi R. Eds. (1998) Information Technology for Knowledge Management. Springer Verlag

[3] Chandrasekaran B., Josephson J.R. & Benjamins V.R. (1999) What Are Ontologies and Why Do We Need Them. *IEEE Intelligent Systems*, 14, p. 20-26

[4] Duchier D., Kornstaedt L., Schulte Ch. & Smolka G. (1998) A Higher-order Module discipline with separate Compilation, Dynamic Linking, and Pickling, *Technical Report*, Programming Systems Lab, DFKI and Universität des Saarlandes

[5] Gruber T.R. (1993) A Translation approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5, 2

[6] Harrison C.G., Chess D.M. & Kershenbaum, A. (1995) *Mobile Agents: Are they a good idea?* Research Report, IBM Research Division

[7] Jazayeri M. & Lugmayr W. (2000) Gypsy: A Component-based Mobile Agent System, Proceedings of the *8th Euromicro PDP'2000*, Rhodos, Greece

[8] Lange, D.B. & Oshima, M. (1999) Seven Good Reasons for Mobile Agents. *Communications of the ACM*, 42, 3, p. 88-89

[9] Mach M., Sabol T., Paralič J. & Kende R. (2000): Knowledge Modelling in Support of Knowledge Management. *Proceedings of the CRIS'2000 Conference*, Espoo-Helsinki, Finland, p. 84-88.

[10] Milojcic D., Breugst M., Busse I., Campbell J., Covaci S., Friedman B., Kosaka K., Lange D., Ono K., Oshima M., Tham C., Virdhagriswaran S. & White J. (1998) MASIF: The OMG Mobile Agent System Interoperability Facility. *Proc. of the Second International Workshop, Mobile Agents '98*, Springer-Verlag, p. 50-67

[11] Motta E. & Zdrahal Z. (1998) A principled approach to the construction of a task-specific library of problem solving components. *Proceedings of the 11th Banff Knowledge Acquisition for KBS Workshop*, Canada

[12] Newell A. (1982) The Knowledge Level. *Artificial Intelligence*, 18, p. 87-127

[13] Nonaka I. & Takeuchi H. (1995) *The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford Univ. Press

[14] Oshima M., Karjoth G. & Ono K. (1998) Aglets Specification (1.1), *IBM Corporation*

[15] Paralič M. (2000) Mobile Agents Based on Concurrent Constraint Programming. *Lecture Notes in Computer Science*, Vol. 1897, pp. 62-75, Springer Verlag.

[16] Paralič, J. and Sabol, T. (2001) Implementation of e-Government Using Knowledge-Based System. Paper accepted for the 2nd Int. DEXA Workshop "On the way to Electronic Government", Munich, September 2001

[17] Peine H. & Stolpmann T. (1997) The Architecture of Ara Platform for Mobile Agents. *Proc. of the First International Workshop on Mobile Agents, MA'97*, LNCS 1219, Springer Verlag

[18] Picco G.P. (1998) Understanding, Evaluating, Formalizing, and Exploiting Code Mobility, *PhD thesis*, Dipartimento di Automatica e Informatica, Politecnico di Torino, Italy

[19] Pritchard J. (1999) *COM and CORBA Side by Side: Architectures, Strategies, and Implementations*. Addison-Wesley

[20] Rothermel K., Hohl F. & Radouniklis N. (1997) Mobile Agent Systems: What is Missing? *Proc. of International Working Conference on Distributed Application and Interoperable Systems DAIS'97*

[21] The Mozart Programming System, Available at http://www.mozart-oz.org/

[22] Tiwana A. (2000) *The Knowledge Management Toolkit*. Prentice Hall

[23] van Heijst G., Schreiber A.T. & Wielinga, B.J. (1997) Using explicit ontologies in KBS development. *Int. Journal of Human-Computer Studies*, 46, p. 183-292

[24] Wong D., Paciorek N., Walsh T., DiCelie J., Young M. & Peet B. (1997) Concordia: An Infrastructure for Collaborating Mobile Agents, *Proc. of the First International Workshop on Mobile Agents, MA'97*, LNCS 1219, Springer-Verlag