A Multi-Agent System to Support Exploiting an XML-based Corporate Memory

Fabien Gandon Rose Dieng Olivier Corby Alain Giboin

INRIA, ACACIA Project, 2004 Route des Lucioles, 06902 Sophia Antipolis, France
<fgandon|dieng|corby|giboin>@sophia.inria.fr

Abstract

A corporate memory and the World Wide Web have in common that they are both heterogeneous and distributed information landscapes. They also share the same problem of relevance of results when one wants to search them. However, compared to the Web, a corporate memory has a delimited and better defined context, infrastructure and scope : the corporation. Taking into account the characteristics of a corporate memory we show in this paper the assets of an approach combining XML technology designed for the Web and the distributed nature of multi-agent systems. In particular, we consider the heterogeneity and distribution of the multi-agent system as a solution to the heterogeneity and the distribution of the corporate memory.

1 Introduction

The information overload and the inefficiency of keyword-based search engines on the Web are problems widely acknowledged. The "Semantic Web" is a promising approach where the semantics of documents is made explicit through metadata and annotations to guide later exploitation. Ontobroker [Dec99], Shoe [Hef99] WebKB [Mar99] and OSIRIX [Rab00] are examples of this metadata technique, relying on annotation based on ontologies. In parallel there is an increasing industrial interest in the capitalization of corporate knowledge leading to the development and deployment of knowledge management techniques in more and more companies. The coherent integration of this dispersed knowledge in a corporation is called a corporate memory. It has the objective to "promote knowledge growth, promote knowledge communication and in general preserve knowledge within an organization" [Ste93]. Corporate memory projects are facing the same problem of relevance as Web search engines when retrieving documents because the information landscape of a company is also a distributed and heterogeneous set of resources. Therefore, it seems interesting to consider a distributed and heterogeneous system to explore and exploit this information landscape such as a Multi-Agent System (MAS). The purpose is to allow the information sources to remain localized and heterogeneous in terms of storage and maintenance, while enabling the company to capitalize an integrated and global view of its corporate memory. The MAS approach allows users to be assisted by software agents usually distributed over the network. These agents have different skills and roles trying to support or automate some tasks: they may be dedicated to interfacing the user with the system, managing communities, processing or archiving data, etc. Our objective is to build and organize a corporate memory to ease the search inside it and the use of its content by members of the organization. This memory contains unstructured, semi-structured or fully-structured data. The importance of relying on standards that are widely accepted led us to use XML technology for exchanges and storage [Rab00]. The XML technology enables us to build a structure around the data, and RDF (Resource Description Framework) allows us to improve search mechanisms using semantics of annotations. In this paper we show that an approach combining XML and MAS technologies, offers a lot of advantages for corporate memory management. In the first section we will introduce the specificity of a corporate memory project and present the CoMMA project we are involved in that led us to study agent systems. The second part will describe the aspect of XML we are interested in and the prototype CORESE [Cor00] we developed to search annotation bases. The third section will present in details the current results of our investigations on multi-agent systems applied to corporate memory with the architecture and the roles we identified so far.

Proc. of the Third Int. Conf. on Practical Aspects of Knowledge Management (PAKM2000)
Basel, Switzerland, 30-31 Oct. 2000, (U. Reimer, ed.)
http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-34/

2 Context of Intervention

2.1 Stakes and Specificity of a Corporate Memory Management System

We define a corporate memory (CM) as an explicit, disembodied and persistent representation of knowledge and information in an organization, in order to facilitate their access and reuse by members of the organization, for their tasks [Rab00]. Compared to the World Wide Web, a corporate memory has a delimited scope: the corporation. Therefore we can precisely identify the stakeholders (e.g.: information providers) and moreover this community shares some common global views of the world (e.g.: company policy, best practices) and thus an ontological commitment is conceivable. The corporation also has its own organization and infrastructure. From a knowledge engineering point of view this means that besides the user's model, an enterprise model can be obtained through a data-collection phase, both models being based on an ontology specific to the corporate memory management task. The user models characterize the different roles and profiles of the stakeholders and are used to customize the interactions and the behavior of the system. The enterprise model presents organizational aspects such as organization charts, processes, documents, and so on. The two models are obviously linked and tangled. They will be used to annotate and search the corporate memory in a user-friendly and efficient fashion. Some organizational aspects are hidden but important for the systems, for example the fact that the organization chart and the acquaintance network do not take into account transversal groups such as "communities of interest" may lead to a functionality that supports the emergence of such communities when they are known to exist but are not precisely identified. Another example is the fact that the intranet infrastructure and network resources policy results in an heterogeneous and distributed set of information sources that changes from one company to another and therefore the system has to be modular enough to cope with this constraint.

2.2 The CoMMA Project

The ACACIA research team, which we belong to, is part of the CoMMA consortium. CoMMA (Corporate Memory Management through Agents) is an IST project [CoM00] funded by the European Commission, which started in February 2000. The main objective of the project is to implement and test a Corporate Memory management framework integrating several emerging technologies: agent technology, knowledge modeling, XML technology, information retrieval and machine learning techniques. The project intends to implement the system in the context of two scenarios:

1. The insertion of new employees in the company.
2. The support of technology monitoring processes.

The solution proposed in CoMMA is based on a MAS architecture of cooperating agents, being able to adapt to the user, to the context, and supporting retrieval of relevant information in the CM. These agents will be able to communicate with the others to delegate tasks, and to make elementary reasoning and decisions, supporting choices between several documents. They will have inference mechanisms exploiting ontologies. They may help authors to annotate documents, to perform technological monitoring on the Internet and to circulate the acquired innovative ideas to the interested employees of the company. The project focuses on the case where the corporate memory is materialized by XML documents and annotated by meta-information in RDF in order to offer intelligent search functionalities and improve document retrieval. We also intend to exploit machine learning techniques in order to make agents adaptive to their users and context. In CoMMA, the realization of the MAS will be simplified by using a pre-existing software framework for the development of agent applications called JADE [Berg00] compliant with the FIPA specifications [FIP97]. Integration of these technologies in one system is already a challenge, yet another is the definition of the methodology supporting the whole design process. In the process of proposing an architecture for the MAS, we have been led to think about the characteristics of a multi-agent system applied to the exploitation of corporate memory from a general point of view; Section 4 presents our first results.

3 Principles and Motivations of this New Approach to Corporate Memory

3.1 XML and MAS: Metadata Approach

The eXtensible Markup Language (XML) is a description language recommended by the World Wide Web Consortium for creating and accessing structured data and documents in text format over internet-based networks.

```
<contact_details>
 <name>INRIA-Sophia</name>
 <address country="FR">
  <street>2004 Route des Lucioles</street>
  <city>Sophia Antipolis</city>
  <postal>06902</postal>
 </address>
 <phone>04 92 38 77 00</phone>
</contact_details>
```

Figure 1. XML example

The XML syntax uses start and end tags to mark up information elements (for example <name> and </name> in Figure 1). Elements may be further enriched by attaching name-value pairs called attributes (for example, country="FR" in Figure 1). Its simple syntax is easy to process by machine, and has the attraction of remaining understandable to humans. XML makes it possible to deliver information to agents in a form that allows

automatic processing after receipt and therefore distribute the processing load over the MAS. It is also a standard, and therefore a good candidate to exchange data and build a cooperation between heterogeneous and distributed sources which is exactly the type of problems tackled by multi-agent information systems adopting, for instance, the wrapper agents approach. XML is extensible: one can define new tags and attribute names to parameterize or semantically qualify data and documents. Structures can be nested to any level of complexity so database schemas or object-oriented hierarchies can be represented. Moreover, the set of elements, attributes, entities and notations that can be used within an XML document instance can optionally be formally defined in a document type definition (DTD) embedded, or referenced, within the document. The DTD gives the names of the elements and attributes, the allowed sequence and nesting of tags, the attribute values and their types and defaults, etc. The main reason to explicitly define the language is that documents can be checked to conform to it. Therefore once a template has been issued, one can establish a common format and check whether or not the documents placed in the corporate memory are valid. Figure 2 presents a DTD corresponding to the XML example of Figure 1. Unfortunately the semantics of the tags cannot be described in a DTD. However if an agent knows the semantics, it can use the metadata and infer from it to help the users of the corporate memory. The semantics must be shared to allow cooperation among the agents and unambiguous exchanges; ontologies are a keystone of multi-agent systems. By describing the meaning of the actual content, structure description will help an agent find relevant information and enable matchmaking between producer and consumer agents. Unlike HTML, XML tags describe the structure of the data, rather than the presentation. Content structure and display format are completely independent. The eXtensible Stylesheet Language (XSL) can be used for expressing style sheets, which have document manipulation capabilities beyond styling. Thus a document of the corporate memory can be viewed differently and transformed into other documents to adapt to the need and the profile of the agents and the users while being stored and transferred in a unique format. Figure 3 presents a style sheet extracting the name and the phone number from the document given in Figure 1. The output of this style sheet is an HTML file given in figure 4. The ability to dissociate structure content and presentation enables the corporate memory documents to be used and viewed in different ways. Therefore XML has a lot of assets to materialize company documents and further forthcoming features of XML will complement this aspect:

- The addressing and linking languages will provide facilities for asserting multidirectional typed link relationships between resources, for annotating links, for out-of-line links, and addressing parts.
- The XML Query language should enable data extraction, transformation, and integration, supporting data-intensive operations, such as joins and aggregates, and construction of new XML data.

```
<!DOCTYPE contact_details [
 <!ELEMENT contact_details (name, address,
                                    phone)>
 <!ELEMENT name (#PCDATA)>
 <!ELEMENT address (street, city, postal)>
 <!ELEMENT phone (#PCDATA)>
 <!ELEMENT street (#PCDATA)>
 <!ELEMENT city (#PCDATA)>
 <!ELEMENT postal (#PCDATA)>
 <!ATTLIST address country CDATA #REQUIRED >
]>
```

Figure 2. DTD example

```
<xsl:template match="/">
<HTML>
 <HEAD>
  <TITLE>Phones</TITLE>
 </HEAD>
 <BODY>
   <xsl:apply-templates />
 </BODY>
</HTML>
</xsl:template>

<xsl:template match="contact_details">
 <xsl:value-of select='name'>
 <xsl:text> : </xsl:text>
 <xsl:value-of select='phone'><BR/>
</xsl:template>
```

Figure 3. XSL example

```
<HTML>
 <HEAD>
  <TITLE>Phones</TITLE>
 </HEAD>
 <BODY>
INRIA-Sophia : 04 92 38 77 00<BR>
 </BODY>
</HTML>
```

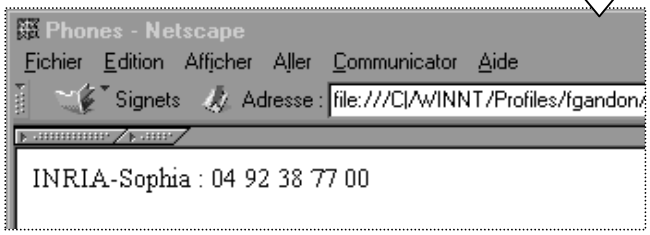

Figure 4. HTML output example

3.2 RDF and MAS: Annotation approach

In their article about "Agents in Annotated Worlds" [Doy98] Doyle and Hayes-Roth explain that software agents must have the ability to acquire useful semantic information from the context of the world they evolve in : "knowledge can literally be embedded in the world as annotations attached to objects, entities and locations". Doyle and Hayes-Roth introduce the notion of "annotated environments containing explanations of the purpose and

uses of spaces and activities that allow agents to quickly become intelligent actors in those spaces". Although the authors choose for their application domain the field of believable agents inhabiting and guiding children in virtual worlds, their remark is transposable to information agents in complex information worlds. This leads us to say that annotated information worlds are, in the actual state of the art, a quick way to make information agent smarter. If the corporate memory becomes an annotated world, agents can use the semantics of the annotation and through inferences help the users exploit the corporate memory. Tim Berners-Lee defines RDF as providing "the necessary foundation and infrastructure to support the description and management of (Web) data." [Bern99] The Resource Description Framework (RDF) uses a simple data model expressed in XML syntax as the basis for a language for representing properties of resources such as images, documents and the relationships between them. One can describe the content of documents through semantic annotations and then use and infer from these annotations to successfully search the mass of information of the corporate memory. RDF defines a mechanism for describing resources through annotations either internal or external to the document, and that makes no assumptions about a particular application domain, nor defines *a priori* the semantics of any application domain. A legacy application is a program or a group of programs in which an organization has invested time and money and usually it cannot be changed or removed without considerable impact on the activity or the workflow. Just as an important feature of new software systems is the ability to integrate legacy systems, an important feature of a corporate memory management framework would be the ability to integrate the legacy archives, especially the existing working documents. Since RDF allows for external annotations, existing documents of the corporate memory may be kept intact (word processor document, spreadsheet, image, etc.) and annotated externally. The annotations are based on an ontology and this ontology can be described and shared thanks to RDF Schema. The idea is (a) we specify the corporate memory concepts and their relationships in ontologies, (b) documents of the memory are annotated using these ontologies. (c) these annotations are used to search the memory and navigate into it. RDF Schema is related to object models (Classes, Properties, Specialization, etc.) using an XML syntax.

```
<Class ID='Document'/>
<Class ID='Article'>
  <subClassOf rdf:about='#Document'/>
</Class>
<Property ID='reviewer'>
  <domain resource='#Document'/>
  <range resource='Literal'/>
</Property>

<Article about='MyArticle.ps'>
  <reviewer>Fabien Gandon</reviewer>
</Article>
```

Figure 5. Simplified Schema & Annotation

However, property objects are defined independently from the classes; an example of a simplified schema and annotation are given in Figure 5, asserting that 'Fabien Gandon' is the reviewer of a given article. The whole model powerfully combines modularity through namespaces, multiple inheritance and multiple instantiation.

3.3 Inferences: Advantages of the association of Conceptual Graph and RDF formalisms

Traditional IR search engines are limited to the *extensional* aspect of concepts. The introduction of ontologies frees us from this restriction and enables us to reason at the *intensional* level. In order to infer over annotation bases, we developed CORESE [Cor00], a prototype of a search engine enabling inferences on RDF annotations by translating the RDF triples to Conceptual Graphs (CGs) and vice versa. As far as we know there are no RDF inference engine available yet. CORESE combines the advantages of using the standard RDF language for expressing and exchanging metadata, and the query and inference mechanisms available in CG formalism. Among Artificial Intelligence knowledge representation formalisms, CGs are widely appreciated for being based on a strong formal model and for providing a powerful means of expression and very good readability. Moreover, inference and query mechanisms have been developed and tested, and are available to manipulate CGs. There exists a real adequacy between the two models: RDFS classes and properties smoothly map onto CG concept types and relation types. More precisely, RDF statements are mapped to a base of CG facts, the class hierarchy defined in an RDF schema is mapped to a concept type hierarchy in the CG formalism and the hierarchy of properties described in the RDF schema is mapped to a relation type hierarchy in CG. The concept type hierarchy and the relation type hierarchy constitute what is called a support in the CG formalism: they define the conceptual vocabulary to be used in the CGs for the considered application. In CORESE Queries are RDF statements with wildcard characters to describe the pattern to be found and the values to be returned. The RDF query is translated into a CG which is projected onto the CG base to isolate any matching graphs and extract the requested values that are then translated back into RDF. The projection mechanism takes into account the hierarchies and specialization relations described by the CG support obtained from the RDF schemas. It also allows for tuning the matching processes, enabling approximate matching or generalization. We are currently investigating the development of a complete query language based on RDF and its mapping to CG projection. Other ongoing work is the extension of the functionalities previously developed for the engine in order to implement agent behaviors related to archiving and searching the documents in the corporate memory. Figure 6 presents examples of RDF and corresponding mapping to CGs. Figure 7 shows a screenshot of the query interface of CORESE and an example of result in raw RDF.

RDF Schema example :

```
<rdfs:Class rdf:ID='document'/>

<rdfs:Class rdf:ID='financial_report'>
 <rdfs:subClassOf rdf:resource='#document'/>
</rdfs:Class>

<rdf:Property ID='title'>
 <rdfs:domain rdf:resource='#document/>
 <rdfs:range rdf:resource='http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Literal'/>
</rdf:Property>

<rdf:Property ID='author'>
 <rdfs:domain rdf:resource='#document'/>
 <rdfs:range rdf:resource='http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Literal'/>
</rdf:Property>

<rdf:Property ID='finance_controller'>
 <rdfs:domain rdf:resource='#financial_report'/>
 <rdfs:range rdf:resource='http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Literal'/>
</rdf:Property>
```

Translated into a conceptual graph support :

```
concept type document < Resource
concept type financial_report < document
relation type title (Document, Literal)
relation type author (Document, Literal)
relation type financial_report (financial_report, Literal)
```

An RDF annotation using the schema :

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
         xmlns:ns='http://www.inria.fr/acacia/FinanceSchema#'>
 <ns:financial_report rdf:about='http://intranet.mycompany.net/~finance/reportB078.doc'>
  <ns:author>Jeremy Smith</ns:author>
  <ns:title>STI Project</ns:title>
  <ns:finance_controller>Steven Clarck</ns:finance_controller>
 </ns:financial_report>
</rdf:RDF>
```

Translated into a conceptual graph:

```
[financial_report: http://intranet.mycompany.net/~finance/reportB078.doc] - {
 -> (author) -> [Literal : Jeremy Smith]
 -> (title) -> [Literal : STI Project]
 -> (finance_controller) -> [Literal : Steven Clarck]}
```

Figure 6. An example of translation from RDF to conceptual graph

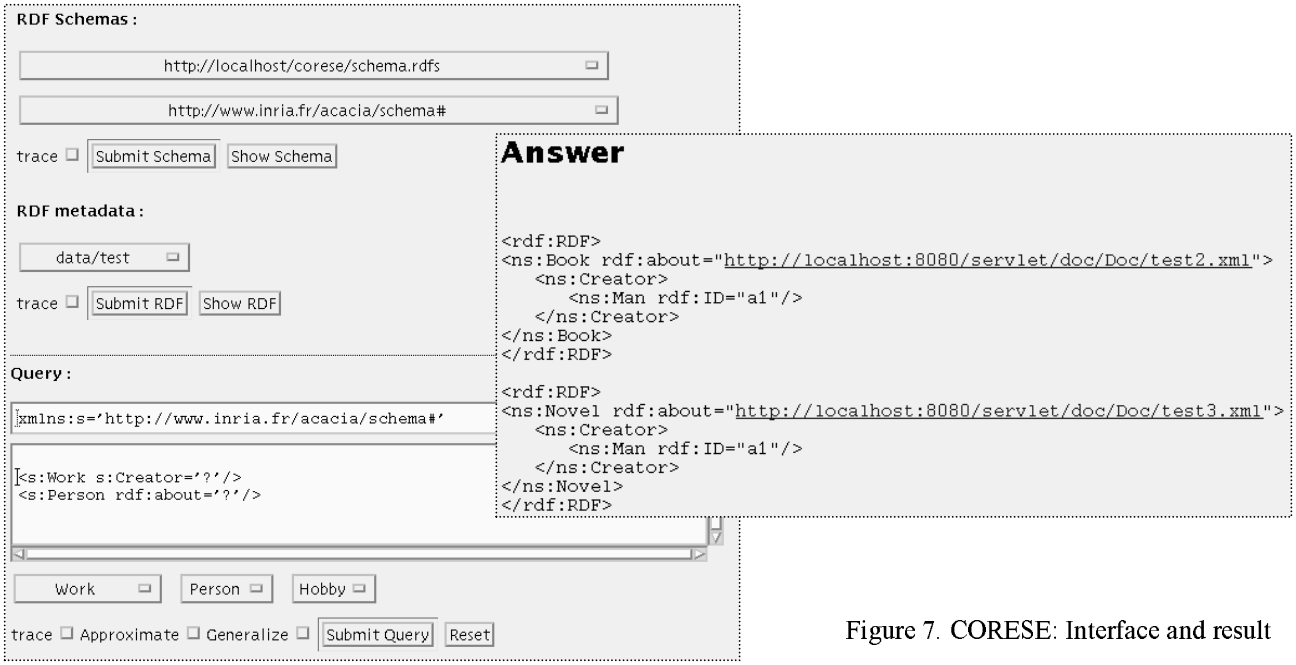

Figure 7. CORESE: Interface and result

4 Designing the Multi-Agent Information System Architecture

In a MAS "Details of the technology underlying the agent and the resources the agent accesses are 'abstracted' away - that is, they are user-transparent. The agent enables a person to state what information he or she requires; the agent determines where to find the information and how to retrieve it" [Etz95]. This is done using a population of provider and requester agents statically distant across the intranet exchanging and exploiting information thanks to a shared formal ontology. In this section we present the architecture and the agent roles we envisaged for a multi-agent corporate memory system assisting members of the organization in their day-to-day exploitation of the memory. The implementation of these agent roles and the associated agent skills, will integrate the modules developed for CORESE [Cor00] to enable agents to handle and infer on a XML-based corporate memory.

4.1 Overall Architecture

The architecture of a MAS is a structure that portrays the different kinds of agencies possible in the agent society and the relationships among them. A configuration is an instantiation of an architecture with a chosen arrangement and an appropriate number of agents of each type. One given architecture can lead to several configurations. In some MAS the configuration is dynamic (mobile agents, dynamic creation and destruction of agents, etc.) but in the case of a corporate memory, we expect it to be rather static except for the user interface. However, the configuration is tightly linked to the topography and context of the place where it is rolled out (organizational layout, network topography, stakeholders location), therefore it must adapt to this information landscape and change with it. The architecture must be designed so that the possible configurations cover the different corporate organizational layouts foreseeable. We consider the heterogeneity and distribution of the MAS as a solution to the heterogeneity and the distribution of the corporate memory. The modularity of MAS solutions is one of the reasons why a multi-agent system is well suited. A multi-agent corporate memory system is likely to be an heterogeneous MAS as it likely integrates several classes of agents. The three-layer model is now one of the classical approaches in multi-agent information systems. For instance in [Klu99A] the author differentiates among three types of agents:

- *Provider agents*: provide their services and capabilities (eg: resource agents, wrappers, etc.)
- *Requester agents*: consume information and services offered by provider agents in the system. (eg: user agents, front-ends, etc.)
- *Middle agents*: mediate among requesters and providers for some mutually beneficial collaboration. (eg: brokers, matchmakers, etc.)

The architecture poses, at the macroscopic level of the MAS, the issues of engineering the interaction and the organization within the MAS society to get, from the overall point of view the system, the functionalities matching the user's requirements. Considering the functionalities and the tasks to be performed by such a system (namely: document retrieval, ontology management, user assistance) we can identify four dedicated sub-societies of agents:

- The agents from the *ontology dedicated sub-society* are concerned with the management of the ontological aspects of the information retrieval activity especially queries about the hierarchy of concepts, associated terms and synonyms, and the different views.
- The agents from the *document dedicated sub-society* are concerned with the exploitation of the documents and annotations composing the corporate memory; they will search and retrieve the references matching the query of the user with the help of the ontological agents.
- The agents from the *user dedicated sub-society* are concerned with the interface, the monitoring, the assistance and the adaptation to the user.
- Finally, the agents from the *connection dedicated sub-society* are in charge of the matchmaking of the other agents based upon their respective needs.

Figure 8 shows the acquaintance graph at the sub-society level.

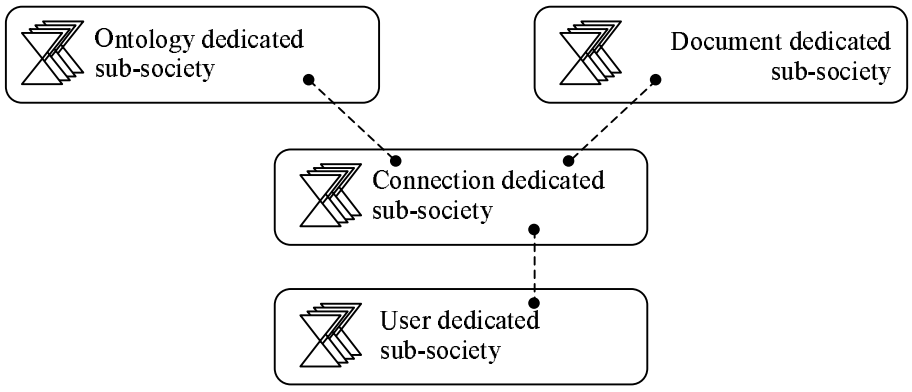

Figure 8. Identified Sub-Societies

4.2 Typology of Sub-Societies

Analyzing the resource dedicated sub-societies (Ontology, Document and Yellow Pages Agents) we found that there was a recurrent set of possible organizations. We now present and compare the three options identified.

4.2.1 Hierarchical sub-society

In this society we distinguish between two kinds of agents:

- *Representative agent*: they are mediators between their society and the rest of the MAS. They are in charge of dealing with the external queries, breaking them down into several sub-queries if needed, contact the resource agents and compile the relevant replies to answer the external requester.
- *Resource dedicated agent*: they are dedicated to a local information resource (data base, annotation repository, document archive, etc.) and contribute to solve the queries they receive as much as they can with their local resources.

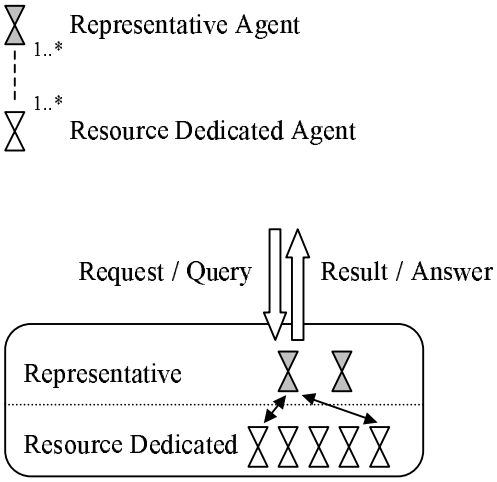

Figure 9. Hierarchy sub-society

In this sub-society the information is distributed over the resource dedicated agents, or more precisely at least one resource agent is present on a node where some information must be accessed. This enables the localization of the information repositories and their maintenance. For example: financial ontology and annotations could be located in the finance department server and therefore ease maintenance by local experts and maybe avoid network loads if a study of the organization reveals that most of the queries concerning finance are issued by people working in the finance department. Workload can also be distributed a great deal considering that resource agents only work with the resource they have locally and that they leave the fusion work to the mediator agents. Mediators can be placed on powerful machines that don't necessarily hold a repository of information and thus balance the workload over other network nodes. Specializing agents and distributing roles allow workload distribution but on the other hand this approach is much more network consuming than the others.

4.2.2 Peer-to-peer sub-society

In this society, peer-to-peer relations are established between the agents. Roles are no longer distributed, they are completely redundant inside the sub-society. Any agent can be contacted from outside the society to solve a query concerning the resource type its society is dedicated to. It will then have to cooperate with its peers to efficiently solve the query. Agents are only specialized through the content of the local information resource they are attached to. Therefore the workload is a bit less distributed than in the previous example but the 'network-load' may be decreased. It must be noticed that from an implementation point of view one difference with the hierarchical sub-society is that the behavior of an agent integrates both the representative and the resource dedicated roles. In other words, there is still a need for developing the same sets of skills but they will be merged in one role.

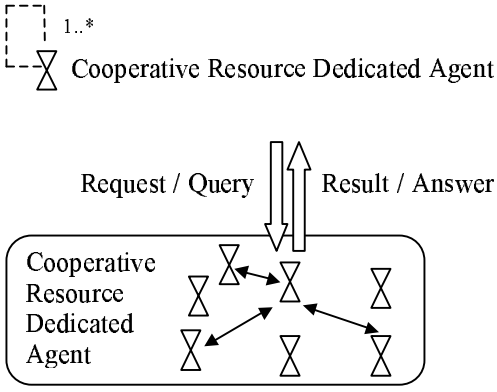

Figure 10. Peer-to-peer sub-society

4.2.3 Replication sub-society

This is a subtype of the previous case: neither the roles nor the content are distributed. Each agent keeps up to date a complete copy of all the information and is able to solve queries by itself. Therefore the only social interactions that exist are replication and content update. The workload is even less distributed than in the previous case and the content has to be replicated everywhere

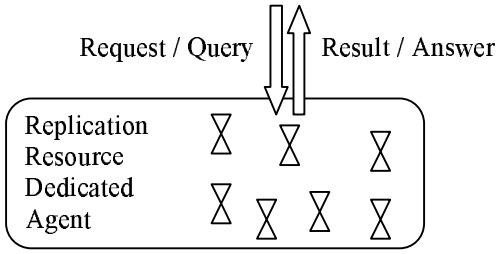

Figure 11. Replication sub-society

which can be highly restricting. On the other hand, there is no specialization; so the system is highly redundant, thus resistant to failure, and the network use is minimal when dealing with a query. From an implementation point of view only one role exists, with archivist skills.

4.3 Dedicated Sub-Societies

In this section, we study thoroughly the dedicated sub-societies using our previous typology. It is important to stress that we are concerned with agent roles more than real agents. This means that one agent we describe may be split into several agents or merged with other agents when implementing a system.

4.3.1 Ontology Dedicated Sub-Society

"Ontology agents are essential for interoperation. They provide a common context as a semantic grounding, which agents can then use to relate their individual terminologies; provide (remote) access to multiple ontologies; and manage the distributed evolution and growth of ontologies." [Sin99]. Ontology dedicated agents are provider agents. They provide downloads and updates of the ontologies for other agents, and may be additional services such as terms and synonyms for the concepts, resolutions of queries on the hierarchy of concepts and relations, etc. They provide the user agents with the ontologies needed for query elicitation and the mediators and archivists with the ontologies needed for query solving. When the system handles several ontologies, ontology agents may be in charge of mapping and translating between ontologies using, for example, mappings to a common ontology. We will not consider these activities in details here but we do acknowledge that ontology management (update, coherence, etc.) is a very hard problem. For the ontology sub-society, the three types of organizations are conceivable:

- In the first case (hierarchical) we would have an Ontology Master Agent type (in charge of resolving external ontological queries) and an Ontology View Agent type (in charge of a part or a view of the ontology, or of one or more ontologies if the system handles several of them).
- In the second case (peer-to-peer) we would have cooperative Ontology View Agent type.
- In the last case (replication) the complete ontology or set of ontologies of the system is replicated so that each agent has a complete copy of ontology information and can resolve queries by itself.

The last case is conceivable when the ontology is stable and a consensus is reached by the users so that the ontological commitment is centralized and the global ontology is updated and propagated over the agent society. The two other possibilities may be interesting when the ontology changes quite often and a mechanism must support the consensus process. Then the agent society can support the break-up of the ontology and maintain the coherence between the different views or the different ontologies and their local modification.

4.3.2 Document Dedicated Sub Society

Document dedicated agents are typical provider agents. They are not Web wrapper agents working on unstructured heterogeneous sources of data as one can find in other MAS projects such as [Mus99]. Document dedicated agents exploit the annotation and metadata to resolve queries. Concerning this sub-society, only the two first types are conceivable:

- In the first case (hierarchical) we would have a Query Mediator Agent type and an Annotation Archivist Agent type.
- In the second case (peer-to-peer) we would have a cooperative Annotation Archivist Agent type (combining both previous roles).
- The third case (replication) is not realistic because it would imply to replicate a full image of corporate memory over each resource agent. This condition is obviously not acceptable since corporate memory, is a huge amount of information broken up and distributed over an intranet.

The Query Mediator agents are both provider and requester agents: they typically provide their services to user-agent to solve their queries and request the services of the resource agents to effectively solve them:

- They decompose and execute queries on distributed relevant sources with the help of middle agents.
- They compose the partial responses obtained from the resource agents to build the final result.

The Annotation Archivists are attached to a local annotation or document repository. When they receive a query, they try to extract at least partial results from their repository to enable the mediator to handle results distributed over several information sources.

4.3.3 Connection Dedicated Sub-Society

As noted in [Hay99] "the adoption of an appropriate coordination mechanism is pivotal in the design of multi-agent system architectures". The use of middle agents such as brokers, facilitators, mediators or matchmakers appears to be the most frequently implemented coordination mechanism. According to [Klu99A] middle agents mediate among requester and provider agents for some mutually beneficial collaboration. Each provider must first register itself with one (or multiple) middle agents and advertise its capabilities by sending some appropriate messages describing the kind of services it offers. The requests are then matched to these descriptions to find which provider may be able to provide the required services. Middle agents allow to decouple the service providers and the service requesters; each agent is no longer obliged to maintain a complete acquaintance list of all other providers it may need to contact. Instead, it only has to know an agent providing Yellow Pages services and may be an agent providing White Pages services for locating appropriate agents with appropriate capabilities. In fact, there are several types of middle agents proposed and used in the literature, and sometimes the differences between them are not clear : the same designation may be used by different projects to

refer to different agent behaviors. However the two main classes are described in [Klu99A]:

- A broker agent in charge of identifying the relevant providers, of transmitting the request to them and returning the result to the requester.
- A match maker agent only in charge of identifying the relevant providers, and returning the selection of candidate(s) to the requester.

As an example, The Directory Facilitator societies provided in JADE [Berg00] are matchmakers in charge of managing Yellow Pages. Their society corresponds to a peer-to-peer sub-society.

4.3.4 User Dedicated Sub-Society

User agents are typical requester agents. Because they are not related to one resource like others (ontology, annotation or Yellow Pages), they cannot be studied using the typology we defined. Roles defined in this sub-society and their distribution depends on additional functional specifications of the system. In [Sin99] user agents are defined as agents "which contain a mechanism to select an ontology; support a variety of interchangeable user interfaces, such as query forms, graphical query tool, menu-driven query builders and query languages; support a variety of interchangeable result browsers and visualization tools; maintain models of other agents; and provide access to other information resources such as data analysis tools, workflows, and concept learning tools". In other applications, learning and adaptation to the user or agent mobility are stressed. However we can distinguish at least two recurrent roles in this sub-society. The first role is the *user interface management* itself: dialogue with the users to enable them to express their request, to refine them and to present results in a comprehensive format. If this role is implemented as an independent agent, this agent may or may not be online (no temporal continuity) depending on whether or not the user is logged onto the MAS system. The second recurrent role concerns the *management of user's profile*. If implemented as an independent agent, user profile agents are just like archivist agents except their annotations are about users. They are ever running agents (temporal continuity), and enable the profiles to be used for interface purposes but also learning techniques, pro-active searches, etc. These two recurrent roles in MAS systems may be merged into one agent or more roles may be added to implement specific functionalities of the system. In the following section we will present examples of roles we introduced for the purpose of CoMMA.

4.4 Customizing a multi-agent corporate memory system

As we said before, modularity gives us the ability to adapt to special needs. For example, by distributing roles or merging them in one agent you can respectively distribute workload or relieve network-load. You can also introduce agents with new roles to add functionality that will immediately benefit to the whole system. In this section we analyze the possible choices currently envisaged for the CoMMA project.

4.4.1 Distributing Roles

In the CoMMA project we envisage three main roles for handling the user's profile and interface:

- *User Interface Manager (UIM)* is in charge of monitoring the user interface and the interaction of the user with the system. This agent is not persistent, it is created on the machine where the user logs in, it only exists for the time of the session and it is destroyed at logout.
- *User Profile Manager (UPM)* is responsible for updating and exploiting the user's profile when the user is logged on to the system. UPMs are persistent, and at login time each freshly created UIM negociates with the nearest UPMs (network-load would be saved since the traffic between these two agents is likely to be important) to determine which one of them is going to manage the profile of its user during the session.
- *Profile Archivist (PA)* is in charge of the profile storage and access. At login, the UIM contracts one of the nearest UPM and the UPM retrieves information about the user from the corresponding PA. When the user logs off, the profile is released.

The idea is that, depending on the request, the UIM will send messages either directly to other agents or through the UPM, so that the user's profile can be enriched or the request may be enriched or refined using the user's profile. For example, if the user is expressing a query with characteristics of a document he is looking for, the UPM could add contextual and profile annotations to enrich the query. However if the interface needs a part of the ontology to be downloaded so that the user can browse it, maybe it is not interesting for this message and its reply to go through the UPM. The Profile Archivist has been introduced to enable the user to logon anywhere on the intranet. The UIM not being continuously running, the storage cannot be one of its roles and the UPM is not always the same since the UIM negotiates with the nearest UPM. However depending on the constraints and the functionalities one envisages for a given system, these roles could be merged in a single agent. On the other hand if a user-profile consists of a distributed set of annotations (annotations from the finance department, from the human resource department, from logistic department, etc.), then one could mention the possibility of profile dedicated sub-society since a profile can be viewed as a resource. This case would be much more complex to engineer (e.g.: distributed updates). In fact, our choice to distinguish the three roles is also motivated by the introduction of other roles corresponding to required functionalities. Indeed the roles we describe in the next section also need to access the profile, even when the user is not logged on and therefore they need the PA.

4.4.2 Introducing New Roles

In CoMMA we are interested in introducing 'proactiveness', therefore we identified several possible

roles. We said the UPM is in charge of using and updating the profile when the user is logged on. It should perform learning on the fly but also proactive actions (ex: register an aborted query to be reconsidered later). As soon as the user logs off another role called User Profile Processor is in charge of working on the user's profile to analyze it and try to proactively identify interesting and relevant information (ex: launch complex queries derived from the user's queries and profile). These first ideas are at user's level. Another level is collaborative information filtering defined in [Klu99B] as the automation of "the process of 'word of mouth' in a given user community. The main purpose and application is to enable agents to anticipate the individual needs of a use in the context of the users". The idea would be to introduce a little more proactiveness than that that could be provided by an individual user's profile, so we envisage the introduction of grouped interest agents that represent a special point of view or set of interests and monitor new documents in the corporate memory. They could also actively search for new documents and broadcast the results to interested user agents. We identified two possible roles and approaches:

- *Public Interest Group Monitor* in charge of analyzing event echoed by Annotation Archivists and Query Mediators to the Profile Archivists; the event would be filtered using the definition of the interest. Another source of 'proactiveness' would be from possible recurrent queries included in the definition of the interest. This approach could be compared to a newsgroup because the existence and the subject of the group is explicit and people register to it.
- *Emergent Interest Detector* looking for implicit communities of interests analyzing the users' profile. When it determines that a user belongs to a community of interest, it uses the profile of the other users to make suggestions. This approach could be compared with collaborative filtering described in [Gut99]

The technology monitoring scenario in CoMMA led us to consider another problem : the automated annotation of external data (from the Web, external databases, etc.) inside the intranet-based corporate memory. A special type of agent could be envisaged, deriving from the wrapper agent that extracts content from unstructured external sources and performs appropriate data conversion. They would be hybrid between wrapper agents and data analysis agents. We identified two options:

- *On-the-fly conversion* where the wrapper agent uses its skills to convert information whenever it is solicited. This approach has the advantage of always providing the up-to-date information but the conversion process may slow the agent's answer. These agents would be a special kind of provider just like archivist agents.
- An *independent daemon triggering checks* at chosen intervals and, if needed (eg: a new version of the monitored Web page is detected), applying conversion mechanisms to update the structured image of this data inside the corporate memory. This approach has the advantage of providing information very quickly since the agents work on the structured image of the information. However, depending on the settings, the data may not be up-to-date (ex: cyclic update every 30 minutes) and also the amount of information duplicated in the intranet may be important.

Our preference goes to the second option where the annotation generator updates an annotation base that is then used by archivist agent when queried. This approach has two advantages:

- It is fast when dealing with a query because the structured data are always directly available when needed
- It decouples and isolates the intranet from the Internet, which is an appreciable feature from a security point of view.

In [Berg99] a mechanism is described for generators of translator: "the agent is generated based on the description of the conversion that needs to take place for queries received and results returned." The idea of facilitating the annotation generator design is indeed interesting since agents have to be customized for each new sources. A library of useful functions or a toolkit for Web Wrappers could be a valuable asset for feasibility and acceleration of the process of developing such agents. For example [Mus99] describes an approach to wrapper induction "based on the idea of hierarchical information extraction" where extraction rules are described as finite automata learned by the agent. This would allow users to develop and launch a population of agents, each monitoring an assigned source. In the case of CoMMA the solution we are envisaging is not exactly a new agent: our annotation generators do not have social abilities and therefore are not compliant with the weak notion of agents [Woo95]. In fact, this new role corresponds to a daemon automatically generating annotations about external resources using annotation patterns and associated trigger events. To automatically generate annotations, we can imagine that the user will be able to fire an annotation generator daemon from his interface augmented by an adequate and customizable toolkit. This daemon would automatically transpose the information extracted from the external source into one or more annotations and add them to a repository managed by an Annotation Archivist agent.

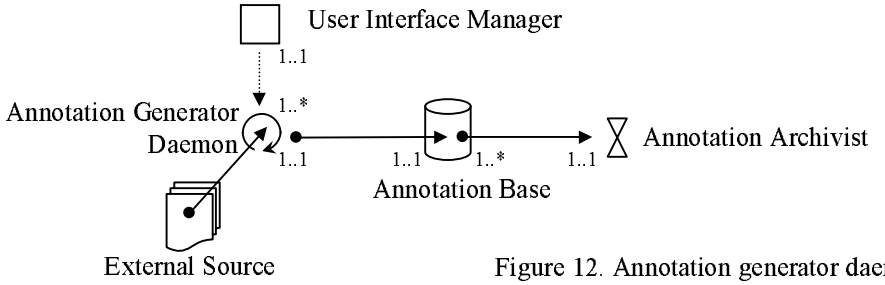

Figure 12. Annotation generator daemon

4.4.3 Scenario Examples

To illustrate the previous architecture choices, we present in this last subsection simple use cases as we imagined them in CoMMA : for pulling information from the system (the user submits a query to the system) and pushing information to the user (the systems detects a new annotation/document).

Running a query

(a) Interface loads / updates the ontology if needed.
(b) The user is presented with in interface to generate a query.
(c) The query is built using the ontology and a query language.
(d) The query could be enriched by the UIM with contextual information.
(e) The query is sent by the UIM to the UPM.
(f) The UPM enriches the query with user's profile information.
(g) The UPM contacts a Middle-Agent to get the name of a Query Mediator.
(h) The Middle-Agent sends a list of possible candidates.
(i) The UPM chooses one candidate and sends its query to it.
(j) The Query mediator contacts a Middle-Agent to get the list of Archivists.
(k) The Query Mediator sends the query to the Archivists.
(l) The Archivists do their best to resolve even partially the query.
(m) The Archivists send the results to the Query Mediator.
(n) The Query Mediator tries to combine partial results to find complete results to the query.
(o) The final result is sent back to the UPM.
(p) The UPM prepares the result for the user.
(q) The UPM sends the results to the UIM.
(r) The results are displayed by the UIM using an appropriate format.

Adding a document

(a) A new spreadsheet is added to the repository of the finance department.
(b) The author of the spreadsheet creates an annotation about this spreadsheet.
(c) The annotation is added to an annotation repository.
(d) The Archivist agent detects the new annotation.
(e) The Archivist agent broadcast the annotation to the agent that registered for such an event.
(f) The Grouped Interest agents (for instance) receive the new annotation.
(g) If the new annotation correspond to the interest they represent, they broadcast the new event to the agents managing the profile of the users that register to their group.

5 Conclusion

As far as we know, such a complete integration of XML technology and multi-agent systems in the framework of knowledge management does not exist yet. In this article we presented some specific points of the corporate memory and some similarities it shares with the Web. We asserted and justified that XML technology is an excellent candidate to handle the structural and distributed aspect of the knowledge contained in a corporate memory and that Multi-Agent technology is an excellent candidate to manage the distributed aspect of its exploitation. An approach combining these technologies seems extremely promising. XML galaxy brings structure and semantics, and thus encourages exploitation and inference on the corporate memory. MAS brings modularity and cooperation, enabling the adaptation to different information landscapes and their exploitation as integrated corporate memories. Our objectives now are the design of efficient cooperation protocols and a trial implementation for validation.

5.1 Acknowledgements

We would like to thank all the partners of CoMMA project for the fruitful discussions we have.
We would also like to thank Peter Eklund for commenting the first version of this paper.
Finally we thank the European Commission for funding this work, carried out in the IST project CoMMA (n. IST-1999-12217).

6 References

[Berg99] Bergamaschi S., Beneventano D., *Integration of Information from Multiple Sources of Textual Data*, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet p53-77, Matthias Klush, Springer 1999

[Berg00] Bergenti F., Poggi A., Rimassa G., *Agent Architectures and Interaction Protocols for Corporate Memory Management Systems*, University of Parma, ECAI'2000, Workshop on Knowledge Management and Organizational Memories, Berlin, August 2000

[Bern99] Berners-Lee T., W3C *Issues Recommendation for Resource Description Framework (RDF)*, Introduces model for defining and organizing information, http://www.w3.org/Press/1999/RDF-REC

[CoM00] CoMMA Consortium, *Corporate Memory Management through Agents*, Conference E-Work & E-Business in Madrid, October 2000

[Cor00] O. Corby, R. Dieng, C. Hébert. *A Conceptual Graph Model for W3C Resource Description Framework*. To appear in Proc. of the 8th International Conference on Conceptual Structures (ICCS'2000), August 2000; Darmstadt, Germany

[Dec99] Decker S., Erdmann M., Fensel D., Studer R., *Ontobroker: Ontology based Access to Distributed and Semi-Structured Information*. Univ. Karlsruhe, Germany, Kluwer Academic Publishers, Boston, 1999.

[Doy98] Doyle P., Hayes-Roth B., *Agents in Annotated Worlds*, In Proceedings of the Second Annual Conference on Autonomous Agents p173-180, Minneapolis, MN USA1998, ACM Press / ACM SIGART

[Etz95] Etzioni O., Weld D.S., *Intelligent Agents on the Internet: Fact, Fiction and Forecast*, University of Washington, IEEE Expert/Intelligent Systems & Their Applications Vol. 10, No. 4, August 1995

[FIP97] *FIPA Specifications 1997*. 1997. Available at http://www.fipa.org

[Gut99] Guttman R., Moukas A., Maes P., *Agent as Mediators in Electronic Commerce*, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet, p131-152, Matthias Klush, Springer 1999

[Hay99] Hayden S.C., Carrick C., Yang Q., *A Catalog of Agent Coordination Patterns*, In Proceedings of the Third Annual Conference on Autonomous Agents p412-413, Seattle, WA USA MAY 1-5, 1999 Edited By Oreb Etzioni ; Jörg P. Müller ; Jeffrey M. Bradshaw ACM Press / ACM SIGART

[Hef99] Heflin J., Hendler J., Luke S., *SHOE: A Knowledge Representation Language for Internet Applications*. Institute for Advanced Computer Studies, University of Maryland at College Park. 1999.

[Klu99A] Klush M., *Introduction of Part 1* of the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet, p3-9 Springer, 1999

[Klu99B] Klush M., *Introduction of Part 2* of the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet, p127-130, Springer, 1999

[Mar99] Martin P., Eklund P., *Embedding Knowledge in Web Documents*, from Griffith University, Australia, 8th International World Wide Web Conference in Toronto 1999

[Mus99] Muslea I., Minton S., Knoblock C., *A Hierarchical Approach to Wrapper Induction*, In Proceedings of the Third Annual Conference on Autonomous Agents, p190-197, Seattle, WA USA MAY 1-5, 1999 Edited By Oreb Etzioni ; Jörg P. Müller ; Jeffrey M. Bradshaw, ACM Press / ACM SIGART

[Rab00] Rabarijaona A., Dieng R., Corby O., Ouaddari R., *Building a XML-based Corporate Memory*, IEEE Intelligent Systems, Special Issue on Knowledge Management and Internet, May-June 2000, p56-64

[Sin99] Singh M.P., Huhns M. N., *Social Abstraction for Information Agents*, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet p37-52 Matthias Klush Springer 1999

[Ste93] Steels L., Corporate Knowledge Management. In J. P. Barthès ed., Proceedings of ISMICK'93, Compiègne, October 1993, p9-30

[Woo95] Wooldridge M., Jennings N. R., Intelligent Agents: Theory and Practice, Knowledge Engineering Review, October 1994, Revised January 1995