# The role of agents in knowledge management

E. Di Nitto, M. Pianciamore, P. Selvini
CEFRIEL – Politecnico di Milano, Italy
Email: dinitto@elet.polimi.it, {piancia, pselvini}@cefriel.it

***Abstract*—[1] Knowledge management is widely recognized as a critical issue in any kind of organization. It has to do with structuring information, ensuring that it is available to all potential users, easily accessible, and presented in such a way that all data relevant to the requesting users are effectively returned in a reasonable amount of time. When dealing with such issues one technology that comes in handy consists of software agents. Agents are software components featuring some nice properties that prove quite helpful to perform routine tasks, which are normally carried out by human users.**

**In this paper we identify some open issues in knowledge management and we discuss how agents can be exploited in this context. We also present the work currently underway in the context of the WIDE research project, whose purpose is to realize a knowledge management infrastructure to support innovative product design in an industrial environment in the automotive field.**

***Index Terms*— agents, ontologies, information retrieval, heterogeneous and distributed sources.**

## I. Introduction

Knowledge management (KM) is widely recognized as a critical issue in any kind of organization. It has to do with structuring information, ensuring that it is available to all potential users, easily accessible, and presented in such a way that all data relevant to the requesting users are effectively returned in a reasonable amount of time. When dealing with such issues one technology that comes in handy consists of software agents. Agents are software components featuring some nice properties that prove quite helpful to perform routine tasks, which are normally carried out by human users. These include processing of large quantities of information, searching over multiple sources spread all over the world, extracting selected portions of documents and so on. Agents can even move on the network carrying along the tasks they were assigned; they can even reduce processing times by self-organizing into societies by spawning children agents acting in parallel.

In this paper we identify some of the open issues in knowledge management and discuss how agents can be exploited in this context. We also present the work currently underway in the context of the WIDE research project, whose purpose is to realize a knowledge management infrastructure to support innovative product design in an industrial environment in the automotive field.

## II. Issues in knowledge management

In whatever organization the management of information and knowledge is a major issue to be dealt with. In most cases, single and separated pieces of information, contributing to form a specific knowledge in any field, are available on a multitude of sources, ranging from traditional, old-style relational database management systems up to repositories structured with XML and derived technologies. If left apart, such pieces of information would be difficult to use. Once put together, they may respond to complex requests, targeting goals that could otherwise be reached only through a long and boring analysis of all information pieces.

The focus of our current research has to do with the design and development of a system for information and knowledge sharing to support innovative product design, by accessing distributed and heterogeneous sources. The actors taking part into the overall scenario are two very distinct kinds of users from the automotive industry domain: stylists and engineers. Though being responsible of different activities, they share a common goal: the design and realization of car parts.

It appears evident how these two user categories present different requirements with respect to the data being handled: stylists or designers are mainly interested in drawings, layouts, colors etc., whereas engineers are used to deal with technical details, spec sheets, and the like. The system to be realized should allow both kinds of users to access the proper information, independent of where it is located and in what format it is represented. In order to address the above issues, we need a full-fledged search infrastructure, able to understand which category the requesting user belongs to, in order to rightly decide what kind of concepts do match best with his/her request. The users should be able to express a query using concepts and terms belonging to their own application domain and to get results back according to their preferences. In turn, the knowledge management infrastructure should be able to locate the information sources that could possibly offer some of the required data, properly reformulate the user query so that it is adjusted to the structure of each specific information source, and then arrange the results according to the user's preferences.

In this scenario some of the problems to be faced concern the organization of distributed information, the definition of

[1] This work is partly funded by the European Commission in the context of the IST-2001-34417 project (WIDE).

the capabilities of each source producing a piece of information, the definition of proper queries involving multiple sources, and the presentation of information to the final user.

### A. *Organization of information*

As we have mentioned before, information pieces are likely to be distributed and partially replicated on different repositories, often built using different technologies. This poses a number of problems concerning the way information is organized. The first problem is purely syntactical: the various repositories may adopt different formats for numbers, characters, and symbols. This problem, if not carefully addressed, could prevent a correct integration of information pieces.

Even if the information to be shared, managed, searched etc. is expressed, let us say, in the Italian language, or in the Metric system, it could be arranged according to completely different structures. In a relational database context this would mean that tables representing the same concept in different repositories could have a different structure (attributes with different names, missing attributes …). This would lead to mutual incomprehension and misunderstanding.

Both the syntactical and structural problems mentioned above could be solved by encapsulating information sources in proper wrappers that offer to the external users a uniform view over the data. Various works in the literature have addressed these issues. Some of them are cited in Section 4.

Indeed, there is still a third and most critical problem to be faced. It concerns the semantic interpretation that is given to data.

Up to now, information available on the web and inside databases is accessed mainly by humans. Each human interprets information according to his/her own expertise and skills about a given field (cars or company's products, administrative processes etc.). As such they are indeed able to understand what they find, what a system offers to them, and can judge if a piece of information is relevant or not to their needs.

As the web is progressively moving towards a fully automated interaction, where machines share data to fulfill the goal of some human users, the model above will soon become too restrictive and new technologies and paradigms will be needed. For two machines to be able to communicate, to understand what a single piece of information is about, and to discovery the parties that can possibly be contacted in order to get some pieces of information, a model describing the semantics of the information pieces has to be provided.

The Semantic Web [2] is being developed to cope with these issues. In particular, new languages are being created, mainly based on XML, which can describe the actual contents of a message or document, in terms of concepts, terms, and relationships between them. For instance, RDF [3] can express sentences like "entity A has a property, whose name is B and whose value is C", independently of what A, B and C actually are. RDF-Schema [4] and DAML+OIL [5] are further able to better characterize such entities, and say that, for example, A is a class (or concept) in a hierarchy, that such class is a descendant of another class-concept by means of a "IS-A" relationship, and, again, that it has something to do with another concept, through a previously defined relationship. Semantic Nets [6] are a formalism to represent the same concepts in a graphical and easy to visualize way.

Meta-structures binding concepts through relationships are usually called ontologies. They allow machines to make some inferences, processing and deductions over data that would otherwise be just a mere bunch of bytes. Several communities, companies, and institutions are striving to endow themselves with ontologies that will eventually form the Semantic Web, as it was originally thought.

A lot of work has, however, still to be done, since too many are the knowledge domains and fields that could, in principle, get in touch with one another. People in those fields are used to think in different ways, to describe things using their own concepts, with custom relationships between them. Thus, we definitely need a large base of common ontologies and ways to go from one to another through suitable mappings that should not loose any expressive power or cover just a few of the concepts in either of them. Moreover, we need tools for the machines to easily deal with ontological concepts and relationships, browse through them, and map them to the information pieces available at the single sources.

### B. *Description of source capability*

In order for a system to know which source to contact when looking for specific information, each source must be adequately characterized in terms of the capabilities it offers. The meaning of the word capability changes depending on the specific information sources being considered. For instance, the capabilities of relational databases could be the entity-relationship model its information items refer to, and the kind of interface it offers to users for executing queries.

The need for explicitly describing capabilities is especially addressed in the context of Web Services-like information sources. These usually export a number of operations that third parties may invoke and provide explicit descriptions of them by exploiting a language that is going to be standardized (WSDL [13]). By browsing through published capabilities, a system could immediately know what services and data could be found. Without such an enabling factor, sources would be hardly known and so they would remain unused. Repositories and directories like UDDI [7] for Web-Services are already of great utility in addressing this issue, but more sophisticated ways of stating the capabilities of a generic provider of information should be set up. In particular, besides the address of a service and the list of the signatures of the methods exported, it would be useful to know other non-functional properties, like the cost and the QoS (response time, frequency etc.). Through ontologies and techniques to represent knowledge, it would thus be possible to have a clear idea about what a generic service is, what it can offer, what kind of data it will manage and process, and according to which procedure.

### C. *Mapping users requests into database queries*

Following the search process going from a user request down to the repository of a specific information source, we realize that ontologies help in precisely matching at the conceptual level the requests with the offered information. However, eventually, such requests need to be mapped into low-level queries, in the language of the respective sources. Thus, we need a two-level mapping. The first mapping allows switching from the query as formulated by the user (and hence using concepts in his own domain) to a more general one, which makes use of concepts belonging to a common ground ontology known to the system.

The second mapping translates this intermediate query to a format suitable to be applied at the various repositories. This second mapping is not trivial at all and should be carefully designed, as it could compromise the efficiency of the overall query process. In particular, it should ensure that the pieces of information that are retrieved are all those needed to answer the user's request, and just those ones. Queries retrieving a quantity of information too large for both the system to process it and the user to browse would not address the above requirement.

### D. *Presentation of retrieved information*

By providing information with some structure, classification, and description, we can take a final aspect into account concerning the presentation of the retrieved information to the users. Presentation is more critical if requesters are humans, since a suitable style, layout, and visual features have to be added to the raw information. Moreover, different users usually have different preferences with regard to how information should be presented to them. Those preferences are usually stored in user profiles, which, however, up to now are not very informative about what exactly the user does want.

By exploiting ontologies and, in general, the semantics associated to bare information, profiles could be enhanced, in order to identify more precisely who a user actually is in terms of his/her interests, of which kinds of items should an answer be formed for him/her to receive, and much more.

Apart from the specific user's preferences, an explicit semantic model of users profiles allows the knowledge management system to post-process information, for example after a search activity has finished, the handful of data that have been found needs to be filtered, selected, with the intent to discard every useless element that would still be present. In addition, such model could also be used to perform some kind of fine tuning on the query before its execution, by adding additional constraints that reduce the set of results returned from the information sources.

## III. Agents and Knowledge Management

In the knowledge management domain, agents have been largely used in a multiplicity of projects and applications, to address a number of functions, roles and activities, as explained below.

By using searching agents one can conduct search operations over large repositories in a repetitive way, still maintaining a good level of efficiency. Agents, in fact, allow for decoupled interaction with information sources. They can simply be "fired and forgotten", and the results can come back when they will finish their job. This feature makes them more attractive than the traditional client/server approach where the communication between the two parts of the system is usually kept alive during all the time taken by the information search activity. Agents are already used to search for something in the Internet. Search engines themselves send their robots (or agents) to collect information and keep their databases up-to-date.

Other kinds of agents are those in charge of representing a user in all his/her activities. User agents, as they are called, are created after a user logs into a system and are responsible of interacting with him/her to receive any request and to act on behalf of the user they represent. They usually interface with another kind of agents that may be called Profiling Agents. These latter manage any information associated to the profile of a generic user, such as the credentials, address, preferences and so on. User agents access the profile information in order to decide how to behave, with respect to the user they represent. This means that, user agents will handle the requests coming from the user on the basis of the corresponding profile. For example, a generic request for a "plane" could be interpreted by a user agent as the need to arrange a trip by plan if the user is profiled as a "traveller" or as the need for aircraft technical specs if the user is a aero-spatial engineer.

User agents may also act as a guide for the user in his/her interaction with the system, preventing him/her from doing mistakes and suggesting the best choices: in this respect agents play a role of tutors.

Moving towards knowledge representation issues, as we have discussed above, a fundamental keystone for a suitable semantic layer to be realized is the development of proper ontologies, covering all the concepts and relationships in a given context. In order to access ontologies and to manipulate them, however, some skills are required. Using ontology agents to access and manage ontologies helps in isolating this functionality and to embody the required know-how in lightweight components (agents themselves) that can behave according to what stated above.

Agents can also be helpful in the last phases of a knowledge management and information retrieval process, when data are returned to the requesting users: by cooperating with the user and profiling agents a filtering agent can decide which data is relevant for the user and which is not, and take decisions about how to rank results according to the user profile-based policies. It can also process the raw results, assembling single pieces of information to form a more complex and organic answer, resolving dependencies and constraints in different parts of a query, to form the overall final result.

Additionally, ad-doc advertising agents could monitor an information source for updates to become available and notify a destination about this. They could either be reactive or proactive in this functionality: if reactive they are more properly called notification agents, where the notifications come as a result of an explicit subscription by someone; if proactive, on the other hand, they assume an advertising role, since it is up to them to decide what users to contact and what content to transfer.

## IV. A KM SYSTEM FOR PRODUCT DESIGN: HIGH LEVEL STRUCTURE

This section presents the work currently underway in the context of the WIDE European research project, whose purpose is to realize an information management and knowledge sharing system that allows users with different perspectives on a common set of concepts to access heterogeneous information spread over a number of distributed sources on the Web. More specifically, the purpose of our work is to support the design process in an industrial environment in the automotive field, in which stylists, engineers and other design team members want to collaborate and share knowledge without requiring the definition of a common language. Often these users present very different needs and profiles; as an example, stylists and engineers may both look for car design details, but from completely different perspectives: they may need data about the same subject (a car), but while engineers are interested in the technical drawings, stylists focus on the colors, shape, layout and so on.
The project will provide methods and tools to support concrete inter-working of different categories of users in the design team. To reach this goal it is necessary to categorize and classify all the information related to the design process, and this will be accomplished by using ontologies, dictionaries and thesauri. The ontologies are used to describe concepts belonging to a specific industrial domain and the relationships between them, while dictionaries relate a user's specific terminology to the system internal one and thesauri provide a set of synonymous for each domain ontology concept.

The second major goal of the project is to provide a robust system able to support the search and the information retrieval process with a high level of precision.

We consider different kinds of information sources, such as proprietary company applications (e.g., product data management systems, advanced CAD/CAE tools) that form the company information systems, but we also will deal with Web Services, search engines and semantic web sites, which form the, so called, web based resources. Hence potential source types span from relational DBs to XML documents and Web Services, as well as search engines and semantic web sites. Being distributed and heterogeneous, such sources may, in principle, store information in whatever format and language (WIDE will be required to manage CAD information as well as text and images). It will be up to the system to transparently manage all the heterogeneity issues.

Our KM system exposes at least two kinds of ontologies. The former is used to model the system domain concepts (for both designers and engineers). It includes also a description of the system actors themselves in order to characterize their different preferences and roles in the product design process. The latter is used to describe the information contained in the various sources.

Halfway between users and information providers, the system under study will use agents to face the various kinds of problems about management of information that we have described in the beginning of the paper.

In detail, the agent infrastructure will focus on two main aspects:

• Given the representation of the knowledge in this industrial domain provided by domain experts, we will need to properly characterize the information sources in terms of their respective capabilities, of the offered services and of how they structure information.

• Once the query as formulated by the user is parsed and translated into a form suitable for the system to "understand" thanks to its internal knowledge, we will need to process this second query in order to map and apply it at the various information sources. A twofold task is required here:

1. Sources need to be found by looking at their capabilities, and queried. The query process could either be driven by stationary agents or by agents moving to all destinations.
2. Low-level queries need to be expressed in a format understandable by sources (e.g. SQL, XQuery etc.).

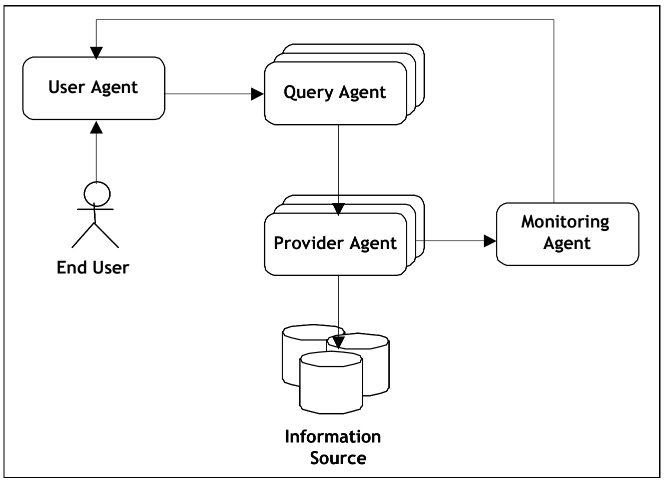

**Figure IV-1 - Internal Architecture**

The diagram in **Figure IV-1** introduces the various kinds of agents used within our infrastructure.

The *User Agent* acts on behalf of a single user, as described in section III. It receives the results of the submitted queries and the notifications about changes occurred in the Information Sources (see below). Such agent is activated when the user logs into the system. After receiving the results of a query previously submitted by the user, the User Agent could also perform some filtering operations, i.e., to customize the results on the basis of some user profile-related policies. Hence, if we refer to the classification presented in section III, this agent also behaves as a profiling and filtering agent .

The *Query Agent* is a classical searching agent, and it is responsible of the execution of a specific query on a set of information sources. Each query submitted by the end user is initially processed and expanded by other components, located outside the agent subsystem, which access the dictionary, the thesauri and the internal ontology in order to obtain a set of so called 'system queries', which are related to the initial one, and expressed in terms of the internal ontology. The User Agent assigns each of the created queries to a specific Query agent for the execution. In general each Query Agent could then submit its associated query to a number of different Provider Agents (see below). All collected results are then returned to the User Agent, after resolving dependencies and constraints in different parts of a query, to form the overall final result. All collected results are then returned to the user agent, after resolving dependencies and constraints in different parts of a query, to form the overall final result.

The *Provider Agent* plays the role of the Ontology Agent described in section III: it knows the ontology used to describe a specific information source and it is the responsible for performing the mapping from the internal ontology to the one describing the source. In addition to its ability in mapping concepts belonging different ontologies the Provider Agent is also able to translate the system query into the low-level queries expressed in a format directly understandable by the information source. The Query Agent interacts with the Provider Agent in order to create such low-level queries, to execute them and to finally collect the results.

The *Monitoring Agent* is an Advertising reactive agent (see section III) residing at each information source and aware of any updates in the data stored in the local information repository. Once a modification occurs in the repository, the information source, by means of the Provider Agent, notifies its monitoring agent about it. The monitoring agent checks its own list of subscriptions, previously received by users, and sees whether or not the updates occurred are relevant with respect to the concepts and items expressed by the various subscriptions. When a match is found, the Monitoring Agent becomes the initiator of the notification process, which eventually makes the user aware of the changes happened in all his data of interest.

## V. RELATED WORK

In the literature we have found quite some works addressing aspects and themes similar to the ones we are focusing on. However, none of them seems to fully cover all of the critical aspects at the same time.

We cite the case of ACQUIRE [8], in which mobile agents are used to reach remotely distributed sources. In that project the original query is not interpreted and mapped by means of ontologies, but is simply decomposed and optimized in terms of best routes to be taken and best query strategies to follow when it is executed at the source. The entire semantic layer seems to be missing.

In CoMMA [9], instead, ontologies play a fundamental role for identifying and dealing with so-called annotations in a corporate memory. The CoMMA system aims at managing information in a corporate semantic web. To this purpose, it uses agents organized into societies, each assigned to a specific role (ontology agents, user agents, interconnection agents etc.) and collaborating to allow users find what they are looking for. A great importance is given to the matchmaking of annotations with queries: a formal and mathematical model is defined to measure the distance between them. CoMMA does not deal explicitly with heterogeneity of data and it is mainly concerned with web resources and references, rather than databases and other kinds of sources.

MOMIS [10] is an information integration system, structured with a mediator and many wrappers, abstracting the differences of the heterogeneous sources being addressed. The mediator stores a global view of the various sources, which looks like a global ontology. Query agents are generated and sent to the wrapped sources, where they ask for information in the source's specific language. The inclusion of new information sources into MOMIS requires a new wrapper to be built (if the kind of source is a new one) and then a human designer must decide how to refine and extend the global view (or metadata) in order to reflect this addition. An automatic integration between the concepts of the new source and the ones defined in the global view would be more appropriate.

LARKS [11] and, more recently, DAML-S [12] address the problem of describing the capabilities of a generic source, be it traditional or more innovative like a Web Service. LARKS appears as a good tool when it comes to efficiently matchmaking between requests on one side, to offers (called advertisements) on the other. With LARKS it is possible to express inputs, outputs, and constraints of services. Queries on service capabilities can be defined with various degrees of precision.

Capability description can also be performed by exploiting the newly built DAML-S ontology, which appears to be more efficient than the existing WSDL language in characterizing what a Web Service is capable of offering. Both LARKS and DAML-S could be helpful for the matching phase and in the description of sources.

Finally, the SIMS project [14], recently evolved in Ariadne [15], addresses the problem of intelligent access to heterogeneous distributed information sources of different kinds (e.g.: DBs, flat files, and so on). Similarly to MOMIS, SIMS builds a domain model, in the form of a view or ontology. Local and individual sources are then mapped to this model by attaching the lower level concepts they represent to the global model through relationships. SIMS looks like a valid starting point for our system, since it fully addresses a lot of the initially mentioned issues about KM. However it is limited to relational database sources (this limitation appears to be removed in Ariadne, which is further able to parse web pages and to classify them on the basis of their contents) and does not exploit all architectural advantages offered by the adoption of an agent-based paradigm.

## VI. Conclusions

Knowledge management is one of the fields that is likely to become more and more critical as new and more powerful tools and languages for automating information handling are developed and used. Software agents present a wealth of features, from autonomy, to mobility, from their loosely coupled nature to their very high interoperability with a lot of systems. As such they may be useful in addressing some of the problems we have to face when striving to realize an efficient and reliable system to manage information in every environment, from a small company to a world-wide setting.

In this paper we have tried to point out what are the most crucial issues about knowledge management we have to address in designing and building a system intended to support knowledge sharing for innovative product design in an industrial domain. Though rich and full of good projects, we believe that the literature has covered parts of the mentioned problems without addressing them all in a single solution. Based on the experiences gained in such works, we will use the agent paradigm coupled with ontologies and knowledge representation techniques to develop a new KM system that is able to manage information retrieval by taking into account the profile of users and the structure of heterogeneous information sources.

## References

[1] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie and J. Siméon, “XQuery 1.0: An XML Query Language.” W3C Working Draft, 16 August 2002, http://www.w3.org/TR/xquery.

[2] T. B. Lee, J. Hendler and O. Lassila, “The Semantic Web.” Scientific American, May 2001 issue.

[3] O. Lassila and R. R. Swick, “Resource Description Framework (RDF) Model and Syntax Specification.” W3C Recommendation 22 February 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.

[4] D. Brickley and R.V. Guha, “RDF Vocabulary Description Language 1.0: RDF Schema.” W3C Working Draft 30 April 2002. http://www.w3.org/TR/rdf-schema.

[5] I. Horrocks, F. v. Harmelen and P. Patel-Schneider, “DAML+OIL (March 2001).” DAML Web site.
http://www.daml.org/2001/03/daml+oil-index.html.

[6] M. R. Quillian, “Semantic memory.” In M. Minsky, ed., Semantic information processing, MIT Press, Cambridge, MA, 227–270.

[7] D. Ehnebuske, D. Rogers, C. v. Riegen, “UDDI Version 2.03 Data Structure Reference”, UDDI Published Specification, 19 July 2002, http://www.uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf.

[8] S. Das, K. Schuster and C. Wu, “ACQUIRE: Agent-based Complex QUery and Information Retrieval Engine.” In Proceedings of the 2002 Conference on Autonomous Agents and Multiagent Systems, pages 631-638, 2002.

[9] F. Gandon, L. Berthelot, R. Dieng-Kuntz, “A Multi-Agent Platform for a Corporate Semantic Web.” In Proceedings of the 2002 Conference on Autonomous Agents and Multiagent Systems, pages 1025-1032, 2002.

[10] S. Bergamaschi, G. Cabri, F. Guerra, “Supporting Information Integration with Autonomous Agents.” Lecture Notes in Artificial Intelligence, 2182:88-99, 2001.

[11] K. Sycara, S. Widoff, M. Klusch and J. Lu, "LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace." Autonomous Agents and Multi-Agent Systems, 5, 173–203, 2002.

[12] The DAML-S Coalition. DAML-S: Web Service Description for the Semantic Web. In: Proceedings of the first International Semantic Web Conference 2002 (ISWC 2002), June 9-12 2002, Sardinia, Italia.

[13] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana, “Web Services Description Language (WSDL) Version 1.1” W3C Note, March 15, 2001 http://www.w3.org/TR/wsdl.

[14] Y. Arens, C. A. Knoblock and Wei-Min Shen. Query Reformulation for Dynamic Information Integration. Journal of Intelligent Information Systems - Special Issue on Intelligent Information Integration, 6(2/3):99-130, 1996.

[15] C. A. Knoblock and S. Minton, “The ariadne approach to web-based information integration.”
IEEE Intelligent Systems, 13(5), September/October 1998.

[16] D. B. Lange. “Mobile Objects and Mobile Agents: The Future of Distributed Computing?” In: Proceedings of European Conference on Object-Oriented Programming (ECOOP 98), Brussels, Belgium, July 1998.