From: AAAI Technical Report SS-03-01. Compilation copyright © 2003, AAAI (www.aaai.org). All rights reserved.

Peer–Mediated Distributed Knowledge Management

Matteo Bonifacio and **Paolo Bouquet**
Dept. of Information and Communication Tech. – University of Trento (Italy)
Istituto per la Ricerca Scientifica e Tecnologica, Trento (Italy)

Gianluca Mameli and **Michele Nori**
Istituto per la Ricerca Scientifica e Tecnologica, Trento (Italy)

Abstract

Distributed Knowledge Management is an approach to knowledge management based on the principle that the multiplicity (and heterogeneity) of perspectives within complex organizations is not be viewed as an obstacle to knowledge exploitation, but rather as an opportunity that can foster innovation and creativity. Despite a wide agreement on this principle, most current KM systems are based on the idea that all perspectival aspects of knowledge should be eliminated in favor of an objective and general representation of knowledge. In this paper we propose a peer-to-peer architecture (called KEx), which embodies the principle above in a quite straightforward way: (i) each peer (called a K-peer) provides all the services needed to create and organize "local" knowledge from an individual's or a group's perspective, and (ii) social structures and protocols of meaning negotiation are introduced to achieve semantic coordination among autonomous peers (e.g., when searching documents from other K-peers). A first version of the system, called KEx, is implemented as a knowledge exchange level on top of JXTA.

Introduction

Distributed Knowledge Management (DKM), as described in (Bonifacio, Bouquet, & Traverso 2002), is an approach to knowledge management (KM) based on the principle that the multiplicity (and heterogeneity) of perspectives within complex organizations should not be viewed as an obstacle to knowledge exploitation, but rather as an opportunity that can foster innovation and creativity.

The fact that different individuals and communities may have very different perspectives, and that these perspectives affect their representation of the world (and therefore of their work) is widely discussed – and generally accepted – in theoretical research on the nature of knowledge. Knowledge representation in artificial intelligence and cognitive science have produced many theoretical and experimental evidences of the fact that what people know is not a mere collection of facts; indeed, knowledge always presupposes some (typically implicit) interpretation schema, which provide an essential component in sense-making (see, for example, the notions of context (McCarthy 1993; Benerecetti, Bouquet, & Ghidini 2000; Ghidini & Giunchiglia 2001), mental space (Fauconnier 1985), partitioned representation (Dinsmore 1991)); studies on the social nature of knowledge stress the social nature of interpretation schemas, viewed as the outcome of a special kind of "agreement" within a community of knowing (see, for example, the notions of scientific paradigm (Kuhn 1979), frame (Goffaman 1974)), thought world (Dougherty 1992), perspective (Boland & R.V.Tenkasi 1995)).

Despite this large convergence, it can be observed that the high level architecture of most current KM systems in fact does not reflect this vision of knowledge (see (Bonifacio, Bouquet, & Manzardo 2000; Bonifacio, Bouquet, & Traverso 2002; Bonifacio, Bouquet, & Cuel 2002) for a detailed discussion of this claim). The fact is that most KM systems embody the assumption that, to share and exploit knowledge, it is necessary to implement a process of "knowledge extraction and refinement", whose aim is to eliminate all subjective and contextual aspects of knowledge, and create an objective and general representation that can then be reused by other people in a variety of situations. Very often, this process is finalized to build a central knowledge base, where knowledge can be accessed via a knowledge portal. This centralized approach – and its underlying objectivist epistemology – is one of the reasons why so many KM systems are deserted by users.

In this paper we describe a peer-to-peer (P2P) architecture, called KEx, which is coherent with the vision of DKM. Indeed, P2P systems seem particularly suitable to implement a DKM system. In KEx, each community is represented by a knowledge peer (K–peer), and a DKM system is implemented in a quite straightforward way: (i) each K–peer provides all the services needed by a knowledge node to create and organize its own local knowledge, and (ii) social structures and protocols of meaning negotiation are introduced to achieve semantic coordination (e.g., when searching documents from other peers). A first version of KEx has been implemented on top of JXTA, a P2P open source project started in 2001 and supported by Sun (see `http://www.jxta.org/`).

The paper goes as follows: first, we briefly discuss the centralized vs. distributed paradigm in KM; second, we describe the main features of KEx, a peer-to-peer system for knowledge discovery and exchange, and argue why it provides a suitable system for distributed KM; then we describe

Copyright © 2003, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

the implementation of KEx; finally, we draw some conclusions and future work.

Technological and social architectures

The main assumption underlying our work, which we share with the structurationist approach (Orlikowski & Gash 1994) in organization sciences, is that technology and organization are tightly interrelated dimensions, which need to be reciprocally consistent. The more an organizational process involves high level human activities, the stronger the interdependence between technological and organizational dimensions is. In particular, since each approach to cognition makes specific assumption on the role of communication, a technology that strongly structures social communication implies a particular model on how cognition occurs (Boland, R.V.Tenkasi, & Te'eni 1994).

In (Bonifacio, Bouquet, & Manzardo 2000; Bonifacio, Bouquet, & Traverso 2002), it is argued that technological architectures for KM can be designed according to two different approaches, each of which presupposes a different social architectures of organizational cognition. The first, called the *centralized* paradigm, views organizational cognition as a convergent process that collects peripheral "raw" knowledge, from various sources, and codifies it into a central repository; as a consequence, technology is viewed as a tool for enabling central control, standardization, high capacity, and robustness. The second, called the *distributed* paradigm, represents organizational cognition as a distributed process that balances the autonomous knowledge management of individual and groups, and the coordination needed in order to exchange knowledge across different autonomous entities; from this perspective, technology is viewed as a way enabling distributed control, differentiation, customization, and redundancy.

Through the analysis of a paradigmatic case study (a worldwide consulting firm), (Bonifacio, Bouquet, & Manzardo 2000) shows that centralized KM systems are often deserted by end-users; indeed, as Bowker and Star argue in (Bowker & Star 1999), any approach which disregards the plurality of interpretative schemas is perceived either as irrelevant (there is no deep understanding of the adopted and centralized schema), or as oppressive (there is no agreement on the unique schema, which is therefore rejected).

To overcome these problems, we proposed a distributed approach, in which technology plays the following roles:

- giving to each community the possibility of representing and organizing knowledge according to its goals and interpretative perspective;
- providing tools to support the exchange of knowledge across different communities without assuming shared meanings, but rather enabling the dynamic translation of different meanings;
- setting mechanisms and protocols to enable, through cooperation, the emergent and bottom-up formation of informal communities and communication practices (such as finding or addressing people to trusted individuals/communities).

In the following section we show how this architecture has been implemented in a P2P system, which provides a peer-mediated support to a distributed approach to designing KM systems. In the conclusions, we will make some remarks on the relation between P2P and agent–mediated knowledge management, and why we went for the first.

KEx: a P2P architecture for DKM

KEx is a P2P system which allows each individual or community to build their own knowledge space within a network of autonomous K–peers, to make knowledge in this space available to other K–peers, and to search relevant knowledge from the knowledge space of other K–peers. We stress the fact that a K–peer may contain not only a structured collection of documents, but also relational knowledge, such as references to experts in some domain, links to other K–peers, to external resources, and so on.

In the following sections, we describe the high-level architecture of KEx, and explain the role that each element plays in a DKM perspective.

K–peers

K–peers are the building blocks of KEx. From an organizational perspective, each K–peer represents a *Knowledge Node* (Bonifacio, Bouquet, & Cuel 2002), namely the reification of an organizational unit – either formal (e.g. divisions, market sectors) or informal (e.g. interest groups, communities of practices, communities of knowing) – which exhibits some degree of semantic autonomy, namely the ability to develop autonomous interpretative schemas (perspectives on the world) to interpret, organize, and store useful information.

In KEx, each K–peer can play two main roles: *provider* and *seeker*. A K–peer acts as a provider when it "publishes" in the system a body of knowledge, together with an explicit semantic view on it (called a *context*, in the sense defined in (Bouquet *et al.* 2002)); a K–peer acts as a seeker when it searches for information that matches some part of its context.

Each K–peer has the structure shown in Figure 1. Below we illustrate the main modules.

Document Repository The *Document Repository* is the place where the data of a knowledge node are stored. In general, it can be viewed as a private space in which document and other data are organized according to some local semantic schema (e.g., a directory structure, or a database schema) and managed by some local application (e.g., a DBMS, a HTTP server, a file system, a document management system).

Context Repository A context is an explicit semantic schema over a body of local knowledge. Of course, more that one context can be used to classify local knowledge. The contexts in use in a K–peer are stored in a *context repository*.

To make contexts usable in KEx, we use a web-oriented syntax for them, called CTXML. CTXML provides an XML–Schema specification of a context; currently, con-

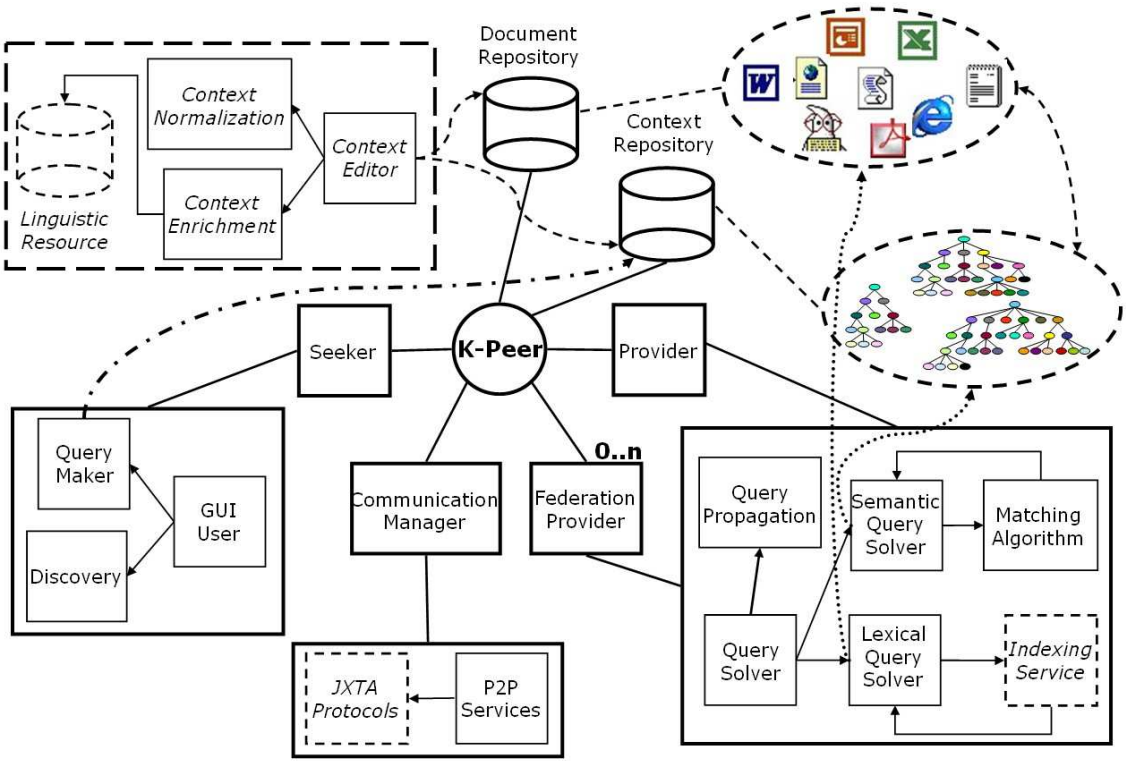

Figure 1: The KEx's main components

texts are concept hierarchies, whose nodes are labelled with words and phrases from natural language, arcs are Is-A, Part-Of or generic relations between nodes (more details can be found in (Bouquet *et al.* 2002)).

From an organizational point of view, a context is the manifestation of a KN's semantic autonomy. Even though a context can be a newly defined schema, in typical situations a context is a "translation" in CTXML of the local application schemas. For example, a context can be the representation of a user's file system (where directory names are used as concept names and the sub–directory structure is used as the structure of the concept hierarchy); or a context can be the representation in CTXML of the taxonomy of a document management system (where the taxonomy is used as a structure, and relations are Is-A relations).

From the standpoint of DKM, contexts are relevant in two distinct senses:

- on the one hand, they have an important role within each KN, as they provide a dynamic and incremental explicitation of its semantic perspective. Once contexts are reified, they become cognitive artifacts that contribute to the process of perspective making (Boland & R.V.Tenkasi 1995), namely the consolidation of a shared view in a KN, continuously subject to revision and internal negotiation among its members;
- on the other hand, contexts offer a simple and direct way for a KN to make public the perspective on the information that it can provide. Therefore, as we will see, contexts are an essential tool for semantic coordination/negotiation among different KNs.

Context management module. The context management module allows a users to create, manipulate, and use contexts. This module has two main components:

- **Context editor**: the context editor provides users with a simple interface to create and edit contexts, and to associate documents and other information with respect to a context. This happens by allowing users to create links from a resource (identified by a URI) to a node in a context. Examples of resources are: documents in local directories, the address of a database access services, addresses of other K–peers that provide information that a KN wants to explicitly classify in its own context.
- **Context Normalization and Enrichment**: this module provides two important services for achieving semantic coordination among K–peers. The first, called normalization, uses NL techniques (e.g., deleting stop words, tokenizing, tagging part-of-speech, etc.) on user defined contexts; the second, called enrichment, provides an interface with an external linguistic resource (in the current version of the system we use WordNet) or with an ontology to add semantic information to concept labels (for example, that in a given context "apple" means a fruit and not a computer brand). Both steps are described in detail in (Magnini, Serafini, & Speranza 2002).

This notion of enrichment is not equivalent to introduce a shared (universal) semantics in KEx. Indeed, the intuition is that the meaning of a concept label in a context has two components:

- the first is the linguistic component, which means that the words or phrases used as concept labels have a standard meaning (or, better, a set of meanings) in a "dictionary". This helps, for example, to distinguish between "apple" as a fruit and "apple" as a tree;
- the second is a sort of pragmatic component, which is given by its position in a context (e.g., in a concept hierarchy in CTXML). This helps in understanding what the user means on a particular occasion with a word (e.g., "apple" in a path like "computer/software/apple" is different from "apple" in a path like "computer/hardware/printers/apple", even though "apple" has the same dictionary meaning').

The first component is public, namely is shared among those who speak a given language. The second is highly contextual, and cannot be computed a priori, namely independently from the context in which it appears. In this sense, contexts are not to be thought of as an alternative to the use of ontologies in a KM system, but as a necessary integration. Indeed, a typical context does not provide semantic information on the terms used in a concept hierarchy nor on their relations, but only on how these terms are combined to create more complex concepts in a classification schema. Consider again the path "computer/hardware/printers/apple". Even if we can decide that "apple" is used in the sense of a computer brand, this path does not provide a definition of what a computer brand is, whereas this definition could be found in a general ontology. However, it is clear that understanding what is the concept associated to such a path requires to use a lot of ontological knowledge about computers, hardware, printers, and their relation in the domain of computers (e.g., that printers are a kind of hardware, that there are different printer makers, that Apple is one of them, and so on). Indeed, in the context matching algorithm we developed for coordinating K–peers (see below), both ontological and contextual information are used to find relations over concepts in different contexts.

It is also important to notice that different linguistic resources or ontologies can be used to enrich a context. So far, we've been using WordNet, but there's no reason why other resources can't be used to replace WordNet. From the standpoint of KM, this is an interesting feature, as we can imagine that different communities may decide to adopt a different linguistic resource or ontologies to enrich their contexts, namely those which better suit their needs. This fact has a significant impact on the mechanisms for sharing knowledge across K–peers, as we will discuss later in this paper.

Roles of K–peers in KEx

Each K–peer can act as a provider, as a seeker, or can play simultaneously both roles. The main components of the two roles are described in Figure 1; the interaction between seekers and providers is described in Figure 2. The following sections explain in details the components needed for implementing the two roles, and their interaction.

Seeker

The seeker module of KEx allows users to search and retrieve useful knowledge from other K–peers. The main module of the seeker component is the *query maker*. A query is built as follows:

- the user opens a context from the context repository;
- the user browses the context until a concept (i.e., a node in the context) is found which describes the topic of interest for that query;
- the user clicks on that concept, this way telling the system that she is interested in finding documents about that concept;
- the system extracts a focus, namely the portion of the selected context which contains relevant information about the selected concept (in the current version of the query maker, the focus is the entire path from the concept to the root of the concept hierarchy, (see (Magnini, Serafini, & Speranza 2002) for a formal definition of focus);
- if needed, the user can add one or more keywords to the query.

When the user submits the query, the seeker activates a session that is associated to that query and is in charge of resolving it (step 1 in Figure 2). The query is propagated to the P2P system (see below for details). The active session can receive asynchronously several incoming replies from those providers that have been selected/suggested by other peers, and collects results that are composed by the aggregation of all those that have been received; each result is made up of a list of document descriptors (name of the document, short description, and so on) and the indication of the part of the providers context that has been used in order to interpret the meaning of the query and provide a resolution. Finally the seeker allows the user to access the K–Peer downloading service; if the user finds in the result set one or more interesting documents, she can contact the providing K-Peer to download it.

Provider.

The provider contains the functionalities needed to accept and resolve a query, and to identify the results that must be returned to the seeker. When a K–peer receives a query (keywords and focus), it instantiates a provider, configured to use a set of contexts and some documents (a particular directory), and to resolve the query.

The main modules needed for the provider role are the following:

- **Query Solver**: the query solver takes a contextual query as an input and returns a list of results to be sent back to the seeker. A contextual query is resolved:
  - by the **Semantic Query Solver**, if a focus is associated to the query itself;
  - by the **Lexical Query Solver**, if a list of keywords is associated to the query.

If the query contains both a focus and a list of keywords, then both solvers are invoked and the result set is the intersection of the two result sets.

- **Query propagation**: this module is in charge of propagating a query to other K–peers (see K–services below for the two modes of propagation allowed in KEx).

The Semantic Query Solver invokes a context matching algorithm, namely an algorithm that finds relations between concepts belonging to different contexts. The details of the algorithm are described in (Serafini *et al.* 2003). Here we only say that the algorithm is based on two main ideas:

- that the information provided in a context presupposes a lot of implicit knowledge, which can be extracted automatically from an external resource (in our case, WordNet). In the current version of the algorithm, the result of this extraction, combined with the explicit information of the context, is represented as a logical formula associated to each node in the concept hierarchy of a context;
- that the problem of discovering what the relationship is between concepts of different contexts can be codified as a problem of propositional satisfiability of a set of formulae, namely the formulae associated to the concepts to be matched plus a set of "axioms" obtained by extracting relevant facts from the external resource.

Given two concepts in two different contexts, the current version of the algorithm we implemented returns one of the following relations as its output: (i) the two concepts are equivalent, (ii) the first is more general than the second, (iii) the first is less general than the second, (iv) the two concepts are disjoint. In KEx, if one of the first three relations holds, then the URIs of the resources associated to the concept on the provider side are returned to the seeker, together with the focus of the concept in the provider's context. This is important, as users on the seeker side may have the opportunity to learn how users on the provider side classify a document in their context.

It is important to observe that the semantic match is performed on the provider side. This means that the result reflects the provider's interpretation of the seeker's query. This explains why we can match contexts normalized and enriched with different linguistic resources or ontologies. The intuition is that, in this case, the provider will normalize and enrich the query's focus using its own resource, this way assigning to it a meaning from the perspective of its users. Of course, the seeker's users can disagree with the resulting match (if any). However, this is not dissimilar from what happens among human agents, as it may happen that an hearer's answer is completely incompatible with the speaker's intended meaning for the question.

The interaction between seeker and provider is depicted in Figure 2. The seeker sends a query to a provider (step 1). When a provider receives a query, it starts a query resolution session and selects relevant documents (step 2). The provider sends back to the seeker the result set (step 3). A provider can propagate a query to other providers that, from its perspective, are "experts" about the query's topic (step 4).

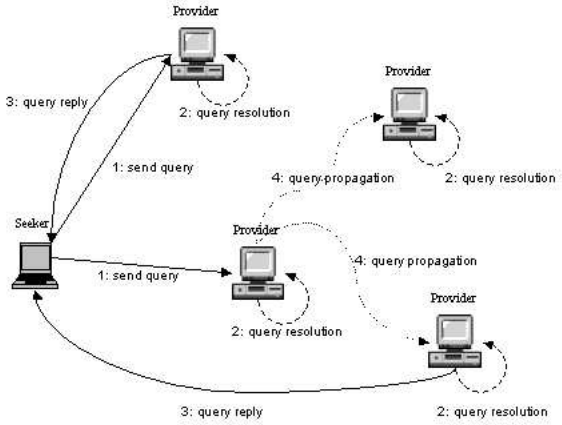

Figure 2: The KEx system: interaction between Seeker and Provider roles

Each provider to which the query is propagated activates a query resolution session, and sends the results to the seeker.

K–Services

KEx provides a collection of knowledge related services that have an important role in supporting knowledge exchange in a network of autonomous K–peers. The more important among them are described in the following sections.

K–federations. KEx provides a federation management service. A *K–federation* is a group of K-Peers that agree to behave like a unique entity when other K–peers perform a search. In other words, each K–federation is a "social" aggregation of K–peers that display some synergy in terms of content (e.g., as they provide topic-related content or decided to use the same linguistic resource to create a common "vocabulary", thus providing more homogeneous and specific answers), quality (certify content) or access policies (certify members). In this sense, the addition of federations to KEx is not trivial, as they embody a first (simple) form of semantic-driven aggregation.

To become a member of a K–federation, a K-Peer must provide a K–federation Service (quite similar to that required by the Provider role) that implements the required federation protocol (reply to queries sent to the K–federation) and observes the federation membership policy. Each K–peer can be member of more than one K–federation.

Currently, we do not make any assumption on how K–federations are formed. However, we anticipate two principal methods of constitution:

Top–down: a group of K–peers decide to create a federation for organizational, commercial, or strategic reasons. Depending on the type of agreement, a new K–federation can be created and membership policies can be defined;

Bottom-up: there are tools that observe the interactions among K–peers and detects the emergence of synergies among K-peers. In a corporation, this may suggest the

existence of a new (informal) community, and lead to the creation of a K–federation to support it.

From a technological point of view, a K–federation becomes a new possible target for a seeker's queries. And indeed, as we can see from Figure 1, the federation module uses the provider module, the main addition being that queries are forwarded to all members of the federation. Currently, the result of sending a query to a K–federation is the very similar to the result of sending the query directly to each member of the federation. Two are the main differences: on the one hand, each K–peer can select which of its contexts and documents can be used to answer a query that comes from each federation to which it belongs; on the other hand, K-peers that reply to a query notify seekers that they replied as members of that K–federation.

In the future, we plan to implement smarter policies for handling queries within K–federations, like a query preprocessing or a selection of the members to which the query should be forwarded.

Discovery. Discovery is a mechanism that allows users to discover resources in the P2P network. Users need to discover K–peers or K–federations available in the network as potential targets of a query. Each K–peer may advertise the existence of a resource by publishing an XML document (advertisement). In KEx, two type resources are currently advertised:

- **K–peers** that have a provider service to solve queries. The main elements of the advertisement are: a description of the peers contexts, and an address to contact the K–peer in order to send it a query or retrieve documents;
- **K–federations**, namely groups of peers that have a federation service to solve queries. The main elements of the advertisement are: the federation domain description, contact information, membership policy.

To discover resources in a P2P network, K–peers can send a discovery request to an already known K–peer, or send a multi-cast request on the network, and receive responses (list of advertisements) that describe the available services and resources. It is possible to specify search criteria (such as a keyword or textual expression) that are then matched against the contents provided by the advertisement related to each K–peer or K–federation description.

Query Propagation. When a provider receives a query, it can decide to forward it to another Provider that is considered "expert" about the query's topic. To decide to which peers the query is to be forwarded, a peer has two possibilities:

- physical "neighborhood": the query will be sent to peers known through the discovery functionality. This way, providers that are not directly reachable by the Seeker, or have just joined the system, can advertise their presence and contribute to the resolution of queries;
- semantic "neighborhood": if the provider computes some matching between a query and a concept in its own context, the system will look for addresses of other K–peers that are linked to that concept. If some are found, then the query is propagated to them, based on the assumption that K–peers classified under a concept may possess relevant information about it.

Obviously, there are several parameters and mechanism controlling the scope of the search and prevent a message "flooding": setting a time to live (TTL), limiting the number of hops, storing in the query the name of peers that already received the query, and so on.

Learning. When the matching algorithm finds a semantic correspondence between concepts of different contexts, the Provider can store this information for future reuse. This information is represented as a semantic "mapping" between concepts (see (Bouquet *et al.* 2002)), and can be used in three ways:

- when the K–peer receives a query from a seeker, it can reuse the corresponding stored mapping to avoid running the matching algorithm;
- a provider can use the existing mapping to forward the query to other peers that present a semantic relation with the topic of the query (see semantic propagation above);
- the seeker can search into the mapping relations in order to suggest the user a set of providers with which it had past interactions and are classified as qualified with respect to the meaning of the concept selected in a query.

Using this mechanisms, the K–Peer network will define and increase the number and quality of the semantic relations among its members, so that it becomes a dynamic web of knowledge links.

Development Framework

In this section we briefly show how the non-functional requirements of a DKM system drive the choice of a particular architectural pattern design (a *peer-to-peer* system) and an underlying technology framework (the *Jxta* Project).

In particular this knowledge exchange system is under development within the business setting of an Italian National Bank and of an international insurance company. For more details see (Bonifacio, Bouquet, & Cuel 2002).

In the DKM approach, a great emphasis is given to autonomy and coordination aspects, so that every KN can access external information and can allow other KNs to access its knowledge. With respect to these requirements, a peer-to-peer system can be depicted with the following capabilities (see (Bertolini *et al.* 2002)):

- being autonomous: each member of the system is seen as a peer that manages and has control over a set of local technologies, applications and services, and can decide how to provide these service;
- being dynamic: peers and resources can be added or removed at any time;
- being decentralized: the community of peers is able to achieve its goal independently from any specific member or component;

- being cooperative: every member must provide resources or services, as well as has a right to access the others'resources and services.

These features of a peer-to-peer system seem to match the spirit and the main non-functional aspects of a KN in a DKM application, and suggest this architectural solution as a logical choice; in particular:

- autonomy is guaranteed by the fact that each KN can be seen as a peer which owns local knowledge, stored and organized through local technologies and applications;
- coordination is guaranteed by enabling peers to collaborate with each other, using a set of dynamic and heterogeneous services that peers provides to each other, in order to support both communication features (as the discovery functionality) and semantic services (e.g. exchange information without imposing a common interpretation schema, but through a meaning negotiation service that automatically maps concepts among different systems of meanings).

From an implementation point of view, we focus on *JXTA*, a set of open, generalized peer-to-peer protocols that allow devices to communicate and collaborate through a connecting network. This P2P framework provides also a set of protocols and functionality as a decentralized discovery system, an asynchronous point-to-point messaging system, and a group membership protocol. A *peer* is a software component that runs some or all the JXTA protocols; every peer has to agree upon a common set of rules to publish, share and access "resources" (like services, data or applications), and communicate among each others. Thus, a JXTA peer is used to support higher level processes (based, for example, on organizational considerations) that are built on top of the basic peer-to-peer network infrastructure; they may include the enhancement of basic JXTA protocols (e.g. discovery) as well as user-written applications. JXTA tackles these requirements with a number of mechanisms and protocols: for instance the publishing and discovery mechanisms, together with a message-based communication infrastructure (called "pipe") and peer monitoring services, supports decentralization and dynamism. Security is supported by a membership service (which authenticates any peer applying to a peer group) and an access protocol (for authorization control). The flexibility of this framework allows to design distributed systems that cover all the requirements of a DKM application, using the JXTA P2P capabilities, completed and enhanced through the implementation of user-defined services. As shows in the previous sections, in the Kex system we combine the P2P paradigm (characterizing a KN network as a network of distributed peers) and JXTA as an implementation infrastructure in a coherent vision with the DKM paradigm.

Conclusions and Research Issues

In this paper, we argued that technological architectures, when dealing with processes in which human communication is strongly involved, must be consistent with the social architecture of the process itself. In particular, in the domain of KM, technology must embody a principle of distribution that is intrinsic to the nature of organizational cognition. This distributed approach is becoming more generally accepted, and other groups are working in the direction of building distributed organizational memories (see e.g. (van Elst & Abecker 2001)).

Here we also suggest that P2P infrastructures are especially suitable for distributed KM applications, as they naturally implement the principles of autonomy and distribution. it is interesting to observe that also other research areas are moving toward P2P architectures. In particular, we can mention the work on P2P approaches to the semantic web (Arumugam, Sheth, & Arpinar 2002; SWAP 2002), to databases (Giunchiglia & Zaihrayeu 2002), to web services (Papazoglou, Jang, & B.J.Kraemer 2002). We believe this is a general trend, and that in the near future P2P infrastructure will become more and more interesting for all areas where we can't assume a centralized control.

A number of research issues need to be addressed to map aspects of distributed cognition into technological requirements. Here we propose two of them:

- **social discovery and propagation**: in order to find knowledge, people need to discover who is reachable and available to answer a request. On the one hand, broadcasting messages generates communication overflow, on the other hand talking just to physically available neighbors reduces the potential of a distributed network. A third option could be for a seeker to ask his neighbors who they trust on a topic and, among them, who is currently available. Here the question is about social mechanisms through which people find – based on trust and recommendation – other people to involve in a conversation. A similar approach could be used in order to support the propagation of information requests;
- **building communities**: if we consider communities as networks of people that, to some extent, tend to share a common perspective (Boland & R.V.Tenkasi 1995), mechanisms are needed to support the bottom-up emergence of semantic similarities across interacting KNs. Through this process, which are based on meaning negotiation protocols, people can discover and form virtual communities, and within organizations, managers might monitor the evolving trajectories of informal cognitive networks. Then, such networks, can be viewed as potential neighborhoods to support social discovery and propagation.

A final remark concerns the relation between agent–based and P2P platforms for distributed KM. We believe that P2P infrastructures are a very straightforward way of mapping social architectures onto technological architectures, this way guaranteeing the coherence between the two structures. From a conceptual point of view, peers are much simpler than agents, and do not allow to exploit the potential of reasoning tools, coordination and collaboration, planning, that agents can provide. However, we need to be very careful in introducing software which can have a significant impact on the way people work and manage their own and corporate

knowledge. In other words, we need conceptual tools for designing agent-based applications which are coherent with the social architecture. An interesting attempt to provide such a methodology can be found in (Molani *et al.* 2003), where intentional modeling techniques are used to analyze organizations as sets of actors that cooperate (or compete) to achieve their goals. This analysis is applied in particular to applications of distributed KM.

Acknowledgments

This work is part of EDAMOK (Enabling Distributed and Autonomous Management of Knowledge), a joint project of the Institute for Scientific and Technological Research (IRST, Trento) and of the University of Trento funded by Provincia Autonoma di Trento.

References

Arumugam, M.; Sheth, A.; and Arpinar, I. B. 2002. The peer-to-peer semantic web: A distributed environment for sharing semantic knowledge on the web. In *WWW2002 Workshop on Real World RDF and Semantic Web Applications. Honolulu, Hawaii (USA)*.

Benerecetti, M.; Bouquet, P.; and Ghidini, C. 2000. Contextual Reasoning Distilled. *Journal of Theoretical and Experimental Artificial Intelligence* 12(3):279–305.

Bertolini, D.; Busetta, P.; Molani, A.; Nori, M.; and Perini, A. 2002. Peer-to-peer, agents, and knowledge management: a design framework. Technical Report tr0207, ITC-irst.

Boland, J., and R.V.Tenkasi. 1995. Perspective making and perspective taking in communities of knowing. *Organization Science* 6(4):350–372.

Boland, J.; R.V.Tenkasi; and Te'eni, D. 1994. Designing information technology to support distributed cognition. *Organizational Science* 5(3):456–475.

Bonifacio, M.; Bouquet, P.; and Cuel, R. 2002. Knowledge Nodes: the Building Blocks of a Distributed Approach to Knowledge Management. *Journal of Universal Computer Science* 8(6):652–661. Springer Pub. & Co.

Bonifacio, M.; Bouquet, P.; and Manzardo, A. 2000. A distributed intelligence paradigm for knowledge management. In *AAAI Spring Symposium Series 2000 on Bringing Knowledge to Business Processes, Stanford University (CA)*. AAAI.

Bonifacio, M.; Bouquet, P.; and Traverso, P. 2002. Enabling distributed knowledge management. managerial and technological implications. *Novatica and Informatik/Informatique* III(1).

Bouquet, P.; Donà, A.; Serafini, L.; and Zanobini, S. 2002. Contextualized local ontologies specification via ctxml. In Bouquet, P., ed., *Working Notes of the AAAI-02 workshop on Meaning Negotiation. Edmonton (Canada). July 28, 2002*. AAAI.

Bowker, G. C., and Star, S. L. 1999. *Sorting things out: classification and its consequences*. MIT Press.

Dinsmore, J. 1991. *Partitioned Representations*. Kluwer Academic Publishers.

Dougherty, D. 1992. Interpretative barriers to successful product innovation in large firms. *Organization Science* 3(2).

Fauconnier, G. 1985. *Mental Spaces: aspects of meaning construction in natural language*. MIT Press.

Ghidini, C., and Giunchiglia, F. 2001. Local Models Semantics, or Contextual Reasoning = Locality + Compatibility. *Artificial Intelligence* 127(2):221–259.

Giunchiglia, F., and Zaihrayeu, I. 2002. Making peer databases interact – a vision for an architecture supporting data coordination. In *6th International Workshop on Cooperative Information Agents (CIA-2002), Universidad Rey Juan Carlos, Madrid, Spain, September 18 - 20, 2002*. Invited talk.

Goffaman, I. 1974. *Frame Analysis*. New York: Harper & Row.

Kuhn, T. 1979. *The structure of Scientific Revolutions*. University of Chicago Press.

Magnini, B.; Serafini, L.; and Speranza, M. 2002. Linguistic based matching of local ontologies. In Bouquet, P., ed., *Working Notes of the AAAI-02 workshop on Meaning Negotiation. Edmonton (Canada)*. Edmonton, Alberta, Canada: AAAI.

McCarthy, J. 1993. Notes on Formalizing Context. In *Proc. of the 13th International Joint Conference on Artificial Intelligence*, 555–560.

Molani, A.; Perini, A.; Yu, E.; and Bresciani, P. 2003. Analyzing the requirements for knowledge management using intentional analysis. In Abecker, A.; van Elst, L.; and Dignum, V., eds., *Agent–Mediated Knowledge Management (AMKM-03)*. AAAI. AAAI 2003 Spring Symposium Series, Stanford University (CA).

Orlikowski, W. J., and Gash, D. C. 1994. Technological Frames: Making Sense of Information Technology in Organization. *ACM Transactions on Information Systems* 12(2):174–207.

Papazoglou, M.; Jang, J.; and B.J.Kraemer. 2002. Leveraging web-services and peer-to-peer networks.

Serafini, L.; Bouquet, P.; Magnini, B.; and Zanobini, S. 2003. An algorithm for matching contextualized schemas via SAT. Technical report, ITC-IRST.

SWAP. 2002. Semantic web and peer-to-peer. http://swap.semanticweb.org/. Project funded by the EC under the IST Programme.

van Elst, L., and Abecker, A. 2001. Domain ontology agents in distributed organizational memories. In *Proceedings of the Internationa Joint Conference on Artificial Intelligence (IJCAI'01)*. Morgan Kaufmann.