Ontology-Addressable Contents in P2P Networks*

S. Castano, A. Ferrara, S. Montanelli, E. Pagani, G.P. Rossi

Università degli Studi di Milano
DICO - Via Comelico, 39, 20135 Milano - Italy
{castano,ferrara,montanelli,pagani,rossi}@dico.unimi.it

Abstract. A critical issue for P2P systems is to perform an effective content retrieval taking into account semantic properties of what is searched. Important requirements to be considered for knowledge sharing in P2P systems are related to the inherent dynamism of the P2P context demanding for a decentralized sharing and administration of knowledge and to the role of Semantic Web techniques to support data semantics representation and rich query languages for content retrieval. In this paper, we describe a comprehensive framework, called H^3, for ontology-addressable contents in P2P systems, which is composed by a knowledge infrastructure layer (Helios) and a communication infrastructure layer (Hermes). The H^3 framework proposes to build an *overlay* network among peers in which each peer maintains a peer ontology describing its knowledge of the network. For query routing, the topology of the overlay network mirrors the semantic neighborhood of the peers given by the semantic relationships among the ontologies they own.

1 Introduction

Peer-to-peer (P2P) networks are distributed systems whose nodes (peers) have equal roles and equal capabilities in exchanging information and services directly with each other. A critical issue for such systems is to perform an effective content retrieval, by taking into account semantic properties of what is searched. Solutions proposed in the literature for content retrieval in P2P systems, often exploit either flooding or broadcasting mechanisms to disseminate the queries when the precise location of searched contents is unknown (e.g., Gnutella [13], Freenet [12]). This approach is not scalable, and its behavior would be even worse if used for content retrieval, where the goal of search is specified in terms of concept descriptions. In this case, we expect to have a higher number of matches and thus replies than in the case of search for a specific file. On the other hand, if the impact of flooding is bounded, for instance by constraining the TTL of the queries, the hit ratio rapidly drops. To overcome some of these problems, Super-Peer Networks [15] (e.g. Morpheus [14], Edutella [9]) have been

* This paper has been deployed in the framework of the WEB-MINDS FIRB Project partially funded by the Italian Ministry of Education, University, and Research.

introduced. In Super-Peer Networks (SPN), peers are clustered with respect of their interests, and for each cluster a Super-Peer (SP) node is designated, acting as a centralized server for queries in a cluster. Moreover, SPs are also connected to each other to create an overlay network. Peer-based data management systems have recently appeared, which rely on SPs for sharing huge amounts of data [6, 7]. Important requirements to be considered for knowledge sharing in P2P systems are related to the inherent dynamism of the P2P context demanding for a decentralized sharing and administration of knowledge and to the role of Semantic Web techniques to support data semantics representation and rich query languages for contents retrieval. In this paper, we describe a comprehensive framework, called H^3, for peer-based knowledge sharing and evolution, which is composed by a knowledge infrastructure layer (Helios) and a communication infrastructure layer (Hermes). The H^3 framework proposes to build an *overlay* network among peers in which each peer maintains a peer ontology describing its knowledge of the network. For query routing, the topology of the overlay network mirrors the semantic neighborhood of the peers, based on the semantic relationships among the ontologies they own.

The paper is organized as follows. In Section 2, we provide a problem statement discussion. In Section 3, we describe the reference architecture of a H^3 peer. The knowledge infrastructure layer and the communication infrastructure layer are presented in Section 4 and in Section 5, respectively. In Section 6, we describe a operating scenario. In Section 7, we discuss related work. Finally, in Section 8 we give our concluding remarks with future research issues.

2 Problem statement

In Figure 1, we provide an example of operating scenario in order to discuss the problems addressed by the H^3 approach. We suppose that the `peer C` joins the network bringing information on the `Detective novel` concept. The neighbors of the `peer C` have no information on the `Detective novel` and are not interested in it. On the opposite, the `peer Y` provides information describing the `Mystery novel` concept and is interested in gaining knowledge on other related concepts such as `Detective novel` of `peer C`. The distance in terms of hops between the `peer C` and the `peer Y` is high. In a typical P2P scenario, the `peer C` would be basically isolated because the other peers would not send it queries on the `Detective novel` concept. Furthermore, the `peer C` and the `peer Y` would have few possibilities to share their knowledge because of the high distance and the TTL costraints on the broadcast diffusion of the queries.

H^3 provides a rich semantic infrastructure, where peer data are represented through ontological descriptions and information is shared through semantically rich queries and retrieval strategies (see Section 4). This approach allows peers to join communities of interest, and to share their knowledge in spite of their network neighborhood.

The H^3 framework proposes to build an *overlay* network among peers in which each peer maintains information about the ontologies located in other peers, in

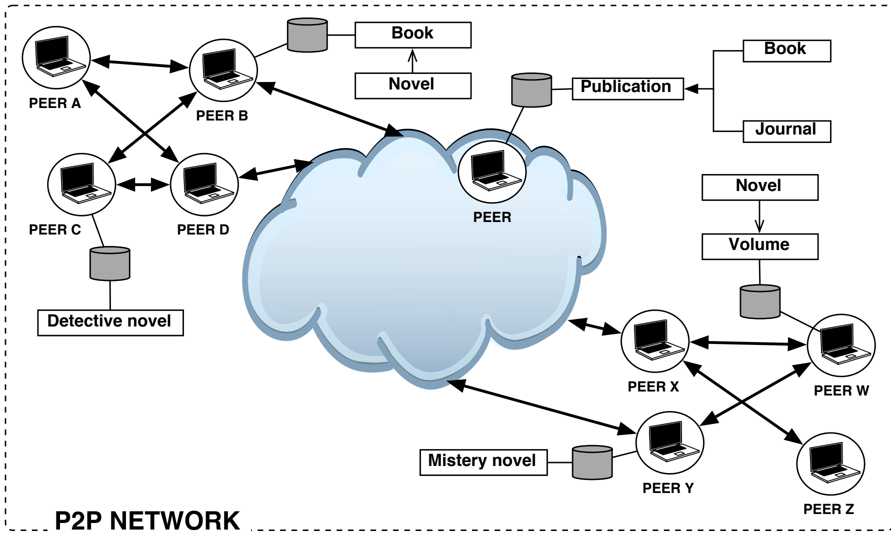

Fig. 1. Example of a peer-to-peer network scenario

order to perform a more precise and focused routing of queries. The topology of the overlay network may mirror the semantic neighborhood of the peers, that is, the semantic relationships among the ontologies they own. A peer joins community(ies) of interest according to its owned data; we outline a simple mechanism to support join requests to a community. The overlay network will guarantee that no isolated peer exists; moreover, different communities should be connected to support the routing of "anomalous" queries issued by the member of a community for concepts outside that community.

The H^3 framework can achieve several goals. The precise routing of the queries towards peers known to have related concepts guarantees a higher hit ratio with respect to other approaches oblivious of the (even approximate) content location. Moreover, it reduces both the network and the peers load, thus providing a greater scalability. The overlay network may dynamically and transparently adapt to peers joining and leaving the communities, as well as to changes of the data and ontologies located at a peer. The adaptation is achieved in the H^3 framework by a learning procedure, that is performed taking into account the frequency with which ontologies and data are requested. The learning can be performed according to a trade-off between the amount of memory needed to record information concerning other peers and the precision of the query routing.

3 Reference architecture of a H^3 peer

The reference architecture of a peer in H^3 (see Figure 2) is composed by two layers, the *knowledge infrastructure* layer and the *communication infrastructure* layer.

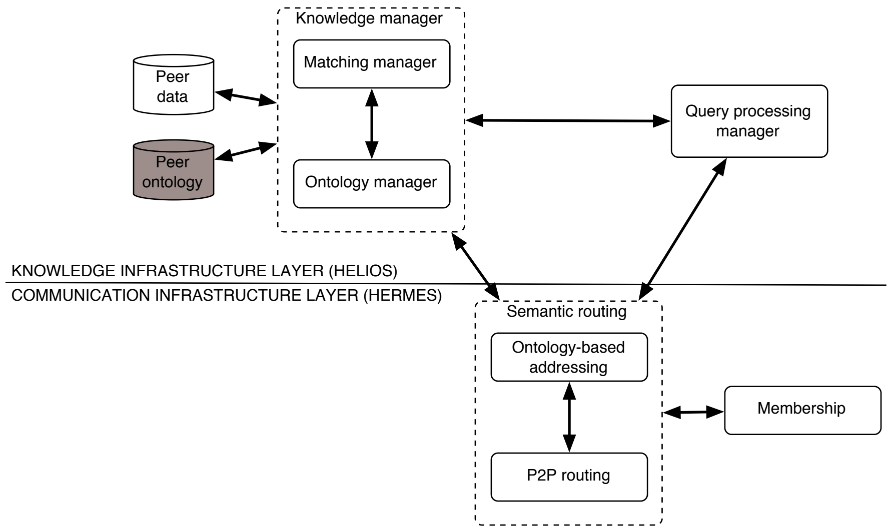

Fig. 2. Architecture of a H^3 peer

Knowledge infrastructure layer. This layer provides the software infrastructure to support the knowledge sharing and evolution processes. Such processes are based on *peer ontologies*, describing the knowledge of each peer (i.e., the knowledge a peer brings to the network and the knowledge the peer has of network), and on *interactions among peers*, allowing content retrieval and knowledge acquisition/extension, according to pre-defined *query models* and *ontology matching techniques*. Each peer can store data/contents (e.g., relational data, XML documents, files, legacy datasets), whose ontological description is provided by the peer ontology. The knowledge infrastructure layer is composed by the following components:

Query processing manager. The query processing manager receives incoming queries, performs the query processing and the answer composition.

Matching manager. The matching manager performs the comparison of a target concept (e.g., a concept in a query) against the peer ontology, in order to find the concepts matching the target concept (i.e., semantically related concepts).

Ontology manager. The ontology manager supports the ontology creation and evolution. It performs the extraction of the ontological description of peer data and performs the integration of new concepts in the peer ontology.

Communication infrastructure layer. This layer provides the software infrastructure to support the communication process and the query routing. With respect to a typical P2P routing, H^3 supports a semantically enriched routing process, by adding to the *P2P routing* component the *ontology-based addressing* component. The communication infrastructure layer is composed by the following components:
Membership. The membership service is invoked by a peer at the bootstrap, to discover other peers in the same community. It is also used to notify the neighbors about the Leave request of a peer, so that they can appropriately update the local overlay network topology by pruning the links with the leaving peer.
Ontology-based addressing. The ontology-based addressing is involved in the query forwarding: it exploits the services of the knowledge manager to discover whether peers exist owning the same or semantically similar concepts addressed in the query or not. In the former case, the query is addressed to those peers; otherwise, it is broadcast.
P2P routing. The P2P routing characterizes the subset of peers which are *neighbors* in terms of both physical network metrics or semantic similarity, and establishes *overlay links* with those peers. This component also performs the query forwarding according to the addresses decided by the *ontology-based* addressing component. Together with the *ontology-based addressing*, it forms the *Semantic routing* component.

4 The knowledge infrastructure layer

In this section, we provide a more detailed description of the Helios layer components.

4.1 Ontology manager

The *ontology manager* organizes and maintains the peer ontology. The peer ontology is the representation of the knowledge owned by a peer at a given time, namely the knowledge describing data/contents stored at the peer and the knowledge the peer acquires from others peers. We conceptualize a peer ontology as a network of *concepts*, where each concept is characterized by a set of *attributes* and a set of *relationships* with other concepts such as in [2]. Moreover, a concept in the peer ontology has associated *location attributes* specifying the network locations of other peers storing concepts and/or contents semantically related to the considered concept. A peer can augment its knowledge in the peer ontology by adding new concepts and/or by enriching existing concept descriptions in terms of new attributes and of new relationships acquired by other peers. The *ontology manager* is responsible for assimilating new concepts in the existing peer ontology, by properly matching them against the ontology for correct integration.

4.2 Matching manager

The *matching manager* has the task of finding ontology concepts that have a semantic relationship with a target concept. In our framework, we are interested in matching a target concept of a query against a peer ontology (knowledge sharing), or in assimilating new concepts returned by queries into a peer ontology (knowledge evolution). Ontology matching techniques of Helios are based on the schema matching techniques developed in the Artemis data integration system [3, 4], properly extended to the problem of concept matching in distributed environments with ontological requirements from autonomous peers. In particular, to take into account different levels of detail in concept descriptions, concept matching in Helios is performed considering the following matching features:

1. *Name.* Concept are matched with respect to semantic contents of their names. In fact, names are generally considered a heuristic indicator of the semantic similarity of concepts in different ontologies.
2. *Attributes.* Concepts are matched with respect to their attributes, to conclude about their matching on the basis also of their structure. In fact, concept names alone are in general a partial indicator of semantic similarity, which can be complemented by the analysis of the concept structure. In the literature, attributes have been considered an important comparison factor for schema matching [11] and ontology merging [8].
3. *Relationships.* The context of a concept, i.e., the set of concepts having a relationship with the considered concept (in the following called adjacents), also provides additional information that can be exploited for determining the level of matching of concepts. In fact, concepts having the same real world semantics are generally characterized by the presence of common/similar concepts in their contexts.

Affinity coefficients are calculated to assess the level of matching of two concepts with respect to each matching feature above. To assess the level of matching of two concepts in a comprehensive way, a Global Affinity measure is defined as the linear combination of the previous affinity measures. In the Global Affinity, weights are introduced to assess the relevance of each kind of affinity in the computation of the Global Affinity measure, allowing flexible matching strategies depending on the information available in the ontological description of the concepts to be matched. Exploiting the global affinity values, the number of concepts matching a target concept depends on the level of closeness we want to impose on them. Given a target concept C and a peer ontology PO, we denote by M_C the set of concepts of PO matching C, i.e., the concepts having a global affinity value $GA \neq 0$ with C. We define two *matching strategies* for the selection of matching concepts:

- **Similarity.** In the similarity-based strategy, the aim is to find the concepts which have a semantic correspondence with C. All concepts whose global affinity value is greater than or equal to a *similarity threshold* $S_T > 0$ are retrieved, that is,
$$M_C^S = \{C' \in PO \mid GA(C, C') \geq S_T\}.$$

– **Equivalence.** In the equivalence strategy, the aim is to find only the concept(s) considered equivalent. All concepts whose global affinity value is greater than or equal to an *equivalence threshold* E_T, with $E_T \gg S_T$, are retrieved, that is,
$$M_C^E = \{C' \in PO \mid GA(C, C') \geq E_T\}$$
with $M_C^E \subset M_C^S$.
This is a more restrictive strategy than the similarity strategy, because it returns a subset of the concepts retrieved by the previous strategy, that is, those having the highest affinity with C.

The interested reader can refer to [5] for a more detailed description of the matching procedure.

4.3 Query processing manager

The *query processing manager* performs the query processing and the answer composition. Three different query models are supported in Helios, namely, the *search model*, the *probe model*, and the *probe/search model*.
Search model. The search query model is used by a peer in order to find contents related to one or more concepts of interest (in the following called target concept). Each peer storing data matching the target concept(s) of a search query can answer to the requesting peer.
Probe model. The probe query model is used by a peer interested in extending its knowledge of the network. Each peer having concepts matching the target concept(s) of a probe query can answer to the requesting peer. The answer to a probe query is constituted by a set of metadata, with which the requesting peer can extend its knowledge on target concepts, in terms of concept description, concept attributes, concept relationships, and concept location.
Probe/search model. The probe/search model allows a peer to find both data and metadata related to target concept(s). With this type of query, a peer can perform a search activity and contemporary increase its knowledge of the network. The answer to a probe/search query is constituted by data and metadata.

When a peer receives a query from another peer, the *query processing manager* processes the incoming query in order to extract the target concept(s) and the query model used. In particular, the query is transformed into an ontological description of the target concept(s) for matching against the peer ontology. In the probe query model, the comparison has the goal of retrieving concepts matching the target concept(s). In the search query model, the comparison exploits the peer ontology to find also the peer data related to concepts matching the target concept(s). The comparison between a target concept and a peer ontology is performed through appropriate matching techniques. The matching process is performed according to the matching strategies previously described, and the number of matching concepts returned for each target concept depends on the matching threshold selected for the matching strategy. The matching strategy and the threshold value can be set either by the requesting peer (request-driven approach) or by the answering peer (answer-driven approach). In the first approach, the matching strategy and the matching threshold are specified directly in the query before submission. In the second approach, they are specified in the answer. Once concepts matching a target concept have been selected, they are returned in the query answer, which is the list of all concepts matching the target concept(s) according to the selected matching strategy.

5 The communication infrastructure layer

In this section, we provide a more detailed description of the HERMES layer components.

5.1 Membership

The *membership* module manages the *Join* and *Leave* requests of the peers. When a peer p boots, the H^3 toolkit it receives also provides the address of an initial peer. Then, p contacts that initial node, and receives from it the addresses of a subset S of peers belonging to the same community $\mathcal{C}$ that p wants to join. S is forwarded to the local *semantic routing* module, in order to establish overlay links.

By contrast, when a *Leave* request is issued, the *membership* takes in charge the notification of such event to the membership of all the peers connected to it in the overlay network, so that the overlay topology is adapted in accordance.

5.2 Ontology-based addressing

This module actively participates to the query forwarding procedure. When a query is received, either from a local user or from another peer, the ontology-based addressing requires to the local *knowledge manager* the addresses of the peers owning concepts semantically related with the concept addressed in the query. If peer addresses are received, the ontology-based addressing requires to the *P2P routing* to forward the query only to those peers. Otherwise, the ontology-based addressing labels the query with a broadcast address, so that it is forwarded to all the peer neighbors in the overlay network.

This policy aims at reducing the network traffic with respect to other approaches that adopt flooding as the forwarding mechanism (e.g., Gnutella [13]), while at the same time providing a high probability of success in the data retrieval.

5.3 Peer-to-Peer routing

The *P2P routing* component in the initialization phase receives from the *membership* the addresses of a set of peers belonging to the same community. *P2P routing* involves mechanisms that estimate the neighborhood between peers. The neighborhood can be defined by combining both the network metrics, such as

the round-trip time or the number of hops, and semantic affinity, that is, two nodes are neighbors if they own similar or equivalent concepts. *P2P routing* establishes overlay links with peers near to the local node. Those links contribute to characterize the topology of the overlay network.
P2P routing routes the queries according to the indications provided by the ontology-based addressing. For the actual message transmission, *P2P routing* exploits the services of the underlying TCP/IP protocol stack.
When a query is received by *P2P routing*, either carrying a broadcast address or addressed to the local peer, *P2P routing* delivers the query to the local *query processing manager* component for the appropriate processing and the creation of the corresponding answer. The role assumed by *P2P routing* in the learning phase consists in *snooping* all the replies that it routes toward the corresponding querier, to deliver them to the local *knowledge manager* to enrich the local peer ontology. As a matter of fact, the learning phase involves the dynamic replication of location information, at the peers where it is more likely needed for the appropriate query forwarding [10].
Some properties must be fulfilled by *P2P routing* in order to perform the above tasks. *P2P routing* must guarantee that peers owning related concepts, or belonging to the same community, are not partitioned; otherwise, data owned by a isolated peer could be not visible by other peers. When flooding is used for the query forwarding as a consequence of the unavailability of information concerning other peers, *P2P routing* must guarantee that loops do not form and duplicate queries are not forwarded, to prevent the collapse of the underlying network.
The overlay topology dynamically changes as a consequence of both peers that join or leave the community, and the creation or deletion of data and concepts at the peers. The link removals must guarantee the network connectivity. New links are created to involve in the overlay topology the newly joined nodes, but also to provide lower latency in the query resolving. As the peers learn about the data and concepts belonging to the other nodes, they may decide to establish links with peers owning concepts that are frequently requested by the local users, to expedite the data retrieval. By contrast, they may destroy links to nodes that do not own interesting data with respect to the queries issued by the local users. The dynamic overlay topology may as well be a consequence of the policy adopted in the learning phase. If the peers do *not* perform a learning stage after the initialization, but rather they acquire new concepts only via the processing of the replies, then initially the overlay links have to be formed by considering only the network infrastructure. When peer affinities are discovered as a consequence of the learning procedure, the topology can be adapted to move from a "network-aware" structure to a "semantic-aware" structure.

6 Operating scenario

In order to show the interactions among the internal components of a peer, we analyze the information flow in two main situations: when a peer sends a query and when a peer receives a query. To have a visual reference of the concepts exposed in this section, see Figure 3.

Sending a query. When a peer sends a query Q over the H^3 network, the request is passed to the *query processing manager* component for rewriting in terms of the ontological description of target concept(s). Rewritten query $\bar{Q}$ is then forwarded to the *semantic routing* component. The *semantic routing* component will send the query only to the peers that promise to supply information semantically related to the kind of concepts requested in the query (semantic neighbors). In order to choose the semantic neighbors, the *semantic routing* component exploits the services of the *knowledge manager* to retrieve ontology location links to the peers whose contents are semantically related to the target concept(s) in the query. The *knowledge manager* determines location links by matching the target concept(s) in $\bar{Q}$ against the peer ontology according to the strategies described in Section 4.2. Location links are returned to the *semantic routing* component, which uses them for query routing. If the query-driven approach is adopted, the requesting peer has to specify in Q the selected matching strategy and the associated threshold to be used for concept matching.

Receiving a query. When a peer receives a query, the *P2P routing* component forwards it to the *query processing manager*. The *query processing manager* analyzes the received query and performs the query processing according to the procedure described in Section 4.3. If no matching concepts are found, the query is discarded and no reply is returned. On the opposite, if matching concepts are found in the peer ontology, the query answer is composed and forwarded to the *semantic routing* component which sends back the reply to the requesting peer. If the answer-driven approach is adopted, the answering peer has to specify the matching strategy and the associated threshold that have been used for concept matching.

7 Related work

In this section, we overview the main peer-based systems for data management. **Edutella** [9] is an open source project that creates an infrastructure for sharing metadata in RDF format. It applies the P2P model using the JXTA protocol. The network is segmented into thematic clusters. In each cluster, a mediator semantically integrates source metadata. Edutella is an example of hybrid P2P architecture, in that each source sends queries to the mediator of its own cluster, and the mediator returns a list of nodes eligible to offer semantically related information. The effective data access holds in direct network connections among peers. The mediator handles a request either directly or indirectly: directly, by answering queries using its own integrated schema; indirectly, by querying other cluster mediators. The **PDMS** system [6] proposes a solution for the semantic integration of heterogeneous information sources in a distributed framework. Network nodes develop different functionalities according to their capabilities.

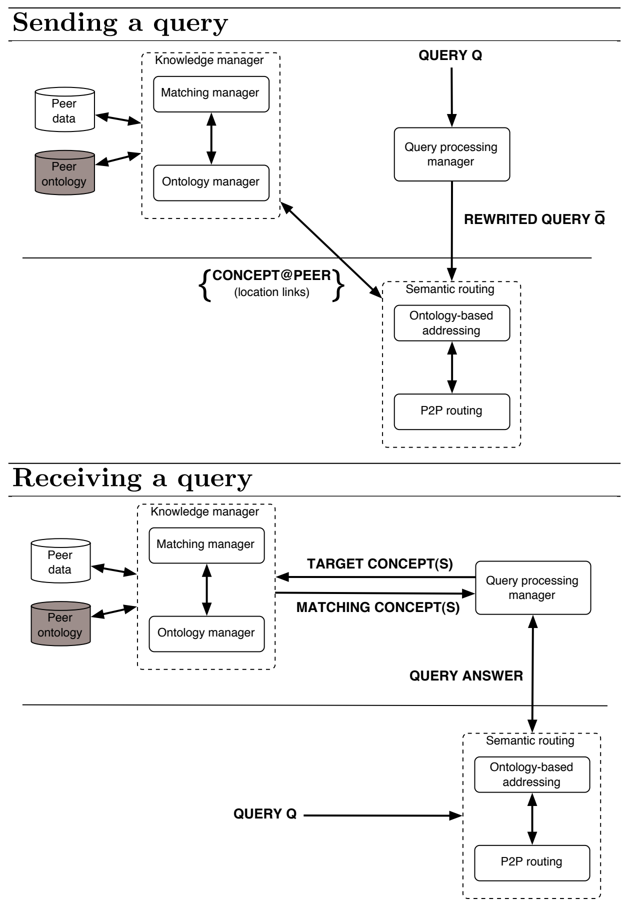

Fig. 3. Example of sending and receiving a query in H^3

In particular, nodes with high resource capabilities play the role of mediators in the network. The system implements a hybrid P2P solution: a mediator node receives a set of information sources schemas and executes the semantic integration step, to derive an ontology view of the acquired information. A set of mediators can be organized in a hierarchy, unifying their ontologies in a global view. When a mediator receives a query from any host, it consults its own ontology and returns a list of sources eligible to offer an answer to the query. A query can be received and analyzed by more mediators, without source clusterization. In **Data Mapping** [7], an approach to determine and handle mappings among heterogeneous data sources in a P2P framework is described. It is an example of pure P2P architecture: network nodes are really equipotential for functionalities and for capabilities and interact each others using a Gnutella-like protocol.

Each node determines semantic mappings among instances of its entities, and takes care of mapping consistency interacting with domain experts. Relations are shared with other peers, that run a comparison and search algorithm to create new relations between received mappings and their own data schemas. Results will be distributed again to progressively increase the knowledge of each community member. The **InfoQuilt**[1] system developed at LSDIS Lab supports heterogeneous information sharing in a distributed framework. Each node describes its own information using an ontological language (DAML+OIL). This description is semantically mapped onto an inter-domain ontology (composed of DAML+OIL classes) which resides in a central directory node. InfoQuilt has a P2P hybrid architecture: each peer queries the directory node and the inter-domain ontology residing at the node to derive the location of the peers storing semantically related information in the system. Data are addressed with a direct connection to the identified peers.

8 Concluding remarks and future research issues

In this paper, we have presented the H^3 framework architecture for ontology-addressable contents and for dynamic ontology knowledge sharing and evolution in P2P systems. The intended goal of H^3 (which constitutes also the original contribution with respect to the state of the art of knowledge sharing in P2P networks) is the use of semantic-based matching techniques for allowing dynamic knowledge sharing/evolution, by providing content location-aware routing facilities. The framework architecture of H^3 has been the first goal of our research activity within the WEB-MINDS project. There are a number of research issues concerned with our system which will be the goal of our future activity in WEB-MINDS, listed in the following.

Knowledge representation. For knowledge representation, we are working in the direction of choosing a Semantic Web compatible formalism for ontology knowledge representation and for supporting semantically rich queries. The representation of the ontology knowledge in H^3 is an important aspect, since it will have an impact on the subsequent development of query processing and ontology consistency techniques. For this purpose, we will rely on ontology knowledge representation facilities of the Artemis data integration system [2, 4].
Matching. The choice of a pure system such as H^3, imposes us to carefully estimate the performance and scalability issues. The core of the H^3 framework is the matching process, which is critical for the efficiency of the whole system. To this end, we are working on dynamic matching techniques scalable as much as possible, for efficient ontology matching, query processing, and ontology evolution.
Knowledge distribution. A further issue to be studied in deep detail in H^3 is related to strategies for the storage of new ontology concepts acquired from the network. A possible strategy is a "storage" approach: a peer stores all concept information it receives from the other peers, without filtering. On the opposite,

in a "link" approach, a peer maintains a location link to the peer(s) storing ontological descriptions of a target concept. In H^3, we will work in the direction of adopting a "mixed" approach, and we are studying criteria for deciding which concepts to store and which concepts to link.

Semantic routing. We are designing mechanisms to combine the several metrics that can be considered to evaluate the neighborhood between two peers, in order to build an overlay network that ensures an effective and focused query routing. The mechanism efficiency will be evaluated with respect to both the probability of success in data retrieving and the amount of generated traffic.

References

1. M. Arumugam, A. Sheth, and I. B. Arpinar. Towards peer-to-peer semantic web: A distribuited environment for sharing semantic knowledge on the web. In *Proc. of the International World Wide Web Conference 2002 (WWW2002)*, Honolulu, Hawaii, USA, 2002.
2. S. Castano and V. De Antonellis. A Discovery-based Approach to Database Ontology Design. *Distributed and Parallel Databases, Special Issue on Ontologies and Databases*, 7(1):67–98, January 1999.
3. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati. Global viewing of heterogeneous data sources. *IEEE Transactions on Data and Knowledge Engineering*, 13(2):277–297, 2001.
4. S. Castano and A. Ferrara. Knowledge representation and transformation in ontology-based dataintegration. In *Proc. of ECAI Workshop on Knowledge Transformation for the Semantic Web*, pages 51–59, Lyon, France, July 2002.
5. S. Castano, A. Ferrara, S. Montanelli, and D. Zucchelli. Helios: a general framework for ontology-based knowledge sharing and evolution in P2P systems. In *IEEE Proc. of DEXA WEBS 2003 Workshop*, Prague, Czech Republic, September 2003. To appear.
6. A. Halevy, Z. Ives, D. Suciu, and I. Tatarinov. Schema mediation in peer data management systems. In *Proc. of International Conference on Data Engineering (ICDE'03)*, Bangalore, India, March 2003.
7. A. Kementsietsidis, M. Arenas, and R. J Miller. Mapping data in peer-to-peer systems: Semantics and algorithmic issues. In *Proc. of ACM SIGMOD'03*, San Diego, California, USA, June 10-12 2003. To Appear.
8. P. Mitra, G. Wiederhold, and J. Jannink. Semi-automatic integration of knowledge sources. In *Proc. of Fusion'99*, Sunnyvale, California, USA, 1999.
9. Nejdl et al. EDUTELLA: a P2P networking infrastructure based on RDF. In *Proc. of the International World Wide Web Conference 2002 (WWW2002)*, pages 604–615, Honolulu, Hawaii, USA, May 2002.
10. E. Pagani and G.P. Rossi. Implementing on-line techniques to allocate file resources in large distributed systems. In *IEEE Proc. of the Euromicro Workshop on Parallel and Distributed Processing*, pages 377–384, Mantova, Italy, February 2001.
11. E. Rahm and P.A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4), December 2001.
12. The Freenet website. `http://freenet.sourceforge.net`.
13. The Gnutella website. `http://gnutella.wego.com`.
14. The Morpheus website. `http://www.musiccity.com`.

15. B. Yang and H. Garcia-Molina. Designing a Super-Peer network. In *Proc. of the International Conference on Data Engineering (ICDE03)*, Bangalore, India, March 2003.