INTEREST-BASED OVERLAY CONSTRUCTION AND MESSAGE ROUTING IN SERVICE-ORIENTED PEER-TO-PEER NETWORKS

Michele Amoretti, Matteo Agosti, Francesco Zanichelli
Departement di Ingegneria dell'Informazione
University of Parma
Parma, Italy
email: {amoretti, agosti, zanichelli}@ce.unipr.it

ABSTRACT

Ontology-based annotation of services is emerging, in the context of Semantic Web or Web 3.0, as an effective technique for automatic service discovery and composition, yet the highly invoked convergence with the peer-to-peer paradigm is still an open issue.

We propose a structured peer-to-peer architectural model in which connectivity rules are interest-based, and service location information is placed in a Distributed Hash Table (DHT), according to a deterministic strategy driven by semantic matching between service profiles and peer interests. The basic idea is to build a key space in which each ID provides complete information about the categories to which the resource (*i.e.* peer or service) is associated. In this context, the concept of interest corresponds to the categories to which a service belongs, or those for which the peer searches the network.

The paper shows the results of several simulation experiments we performed to evaluate the performance of the search algorithm in terms of network resource coverage and to assess system resilience in presence of churn.

KEY WORDS
Peer-to-Peer, Services, DHT, Semantic Routing.

1 Introduction

Service-Oriented Architectures (SOAs) are rapidly becoming the key approach for achieving new levels of interoperability and scalability in the development of highly distributed applications. While current SOA approaches rely on centralized registries, efficient virtual organizations in which all participants can be both resource providers and consumers in a totally decentralized peer-to-peer fashion, appear highly appealing for multiple application domains. Modern peer-to-peer middleware such as JXTA has the potential to enable flexible, scalable, yet controlled information and service sharing across enterprise boundaries. An effective technique for automatic service discovery and composition, the semantic annotation of services is a key incredient of the Semantic Web (*a.k.a.* Web 3.0), although the desirable convergence with the peer-to-peer paradigm is still an open issue.

In this work we propose a structured peer-to-peer architectural model in which connectivity rules are interest-based, and service location information is placed in a Distributed Hash Table (DHT), which is used as a substrate, according to a deterministic strategy driven by semantic matching between service profiles and peer interests. Lookup queries are forwarded across overlay paths to peers in a progressive manner following the same strategy used in the P2P network construction. The architectural model exhibits a remarkable performance of several emerging characteristics, such as the nontrivial node degree distribution, and the dependence of performance indicators on the relation between the average number of interests of each peer, the total number of service categories, and the network size.

The paper is organized as follows. Section 2 gives an overview of related works on interest-based peer-to-peer networks and semantic routing algorithms. Section 3 presents our architecture, with particular emphasis on service categories, construction of semantic keys (*i.e.* peer and service identifiers) and interest-based message routing. Section 4 illustrates several simulation results, while some conclusive remarks and future research directions complete the paper.

2 Related Work

Semantics-based architectural models represent a hot topic in the peer-to-peer research field. In this section we summarize the most interesting state-of-art solutions, trying to point out their major strengths and flaws.

Interest-based Clustering Network (ICN) [1] extends Freenet, by the introduction of caching mechanisms and a three-layered structure made of classes, clusters and nodes. Classes are content macro-categories, while clusters are sub-categories of classes. A node can participate in many clusters, also belonging to different classes. This approach is interesting but semantics is used only in the overlay construction and maintenance processes, while the research process is based on the routing algorithm of Freenet.

Foster and colleagues [2] propose a completely different solution, in which information about consumed data is used for real-time computing of common interests among peers, without aprioristic grouping. When a node searches for a file, it asks one of the storage nodes (*i.e.* peers which share their resources) in its neighbors list, to know which other nodes requested the same file. This process leads to a possible reconfiguration of the overlay network topology, with the emergence of new interest groups. One of the most interesting aspects of this approach is that it works also if peer interests dynamically change.

In Personalized Web (PW) [3], message routing is based on the Semantic-Driven Hashing (SDH) function, whose input is the ontologic description of the resource to be published/discovered. In details, this scheme extracts the synset ID of the concept associated to the resource description, and uses it as input for the SDH function. The resulting key is used to find, in the DHT space established by the overlay network, the node to which the responsibility for the maintenance of the resource information must be assigned. The DHT guarantees deterministic overhead and scalability of the routing process with respect to the size of the overlay network. The main drawback of this approach is that information related to most popular resources (we should say *most popular concepts*) are routed towards few unfortunate nodes, thus leading to an unbalanced workload distribution among peers.

Another schema-based solution is Edutella [4], in which resources are described by means of RDF metadata. User queries are translated in RDF_QEL messages, which based on the Datalog semantics. A similar approach is used in SWAP [5], which aims to support knowledge sharing in a broader sense (Edutella addresses educational contents only).

3 Architectural Model

From the state of the art on resource sharing in peer-to-peer networks, radically different approaches emerge, but the common denominator is the tendency to enrich, by means of semantics, the description of easily annotable resources, such as documents or multimedia contents. Our work focuses on services, which require a much more rich semantics in terms of descriptive power and inferential mechanisms.

In this section we illustrate our interests-based architectural model for service-oriented peer-to-peer networks. We start with the description of the service ontology. Then, we show how peer and service identifiers are constructed from the categories in which a peer is interested. In the last part, we explain the overlay construction algorithm, the service publishing and discovery strategies, and the churn management mechanisms.

3.1 Service Ontology

The best known service ontology is OWL-S [6], built on top of OWL, which is considered the language of the Semantic Web. OWL-S is made to support autonomous agents in Web Service discovery, invocation, composition and monitoring, enhancing traditional languages and protocols such

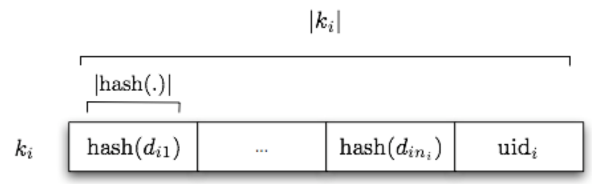

Figure 1. Structure of the key built upon resource interests.

as WSDL and UDDI. OWL-S defines a superior ontology for service profiles, models and implementations.

The framework we propose defines semantics-based peer-to-peer protocols for service publication and discovery, not considering service composition and invocation issues. For this reason, in the following we recall only the part of OWL-S ontology which refers to service profiles.

OWL-S provides a representation for the `ServiceProfile`, namely the `Profile` class, but the latter can be modified by means of OWL's *subclassing* mechanisms. The `Profile` describes the following aspects of a service:

- *Provider*
- *Operations*
- *Other properties*

The latter includes, among others, the `ServiceCategory` class, which has a fundamental role in our routing algorithm. The `ServiceCategory` class allows to classify services by means of an external taxonomy. Moreover, the `Profile` class is characterized by the `serviceClassification` and `serviceProduct` properties, which are used to specify the type of service and the products it manages. Both properties represent concepts which are similar to the `serviceCategory`, but differ in the content, since they refer to OWL instances rather than links to non-OWL business taxonomies.

3.2 Key Construction Process

The routing algorithm, which is illustrated in next section, require that peers and services (in general, resources) are unambiguously identified. Our approach is to build a key space in which each resource ID provides complete information about the categories to which the resource is associated. In this context, the concept of *interest* corresponds to the categories to which a service belongs, or those for which the peer searches the network.

Let $D_i = \{d_{i1}, d_{i2}, \ldots, d_{in_i}\}$ be the set of the n_i interests associated to the i-th resource, and K the set of all keys. Key $k_i \in K$ is built by linking together the categories and a unique identifier associated to the i-th resource. The names of the categories are hashed, resulting in fixed-size strings in the same space of the unique identifier. Thus, $k_i = \text{hash}(d_{i1})\text{hash}(d_{i2})\ldots\text{hash}(d_{in_i})\text{uid}_i$, where uid_i is the unique identifier. The size of the resulting key depends on the number of categories associated to the resource. In details, $|k_i| = |\text{hash}(.)|\,(n_i + 1)$. Figure 1 illustrates the structure of k_i.

As we previously anticipated, all peers refer to the same service ontology, *i.e.* OWL-S, which is used to associate services to interests, using the `ServiceCategory` class. The latter could refer to a specific business taxonomy, but in our opinion this approach is extremely limitative. For example, the NAICS taxonomy [7]uses the same code to identify a wide range of services which appear to be very different, depending on the point of view.

Our framework is based on a different approach, *i.e.* the mappings of service categories to concepts extracted from the WordNet ontology of english language [8]. In WordNet nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. The whole ontology is organized as a hierarchical tree whose nodes, the concepts, are connected by means of *is-a* relations (hyponymy/hyperonymy, meronymy, role, etc.).

3.3 Routing Algorithm

The core of our architecture is the routing algorithm, which is used to build an overlay network characterized by high clustering of nodes which share the same interests, and to efficiently forward service advertising/query messages towards interested and experienced nodes. In our model, the overlay network is a Distributed Hash Table (DHT) in which published service advertisements and peers responsible for storing them are identified by interest-related keys (whose construction process has been described above).

To support this kind of approach, each node of the network has three information repositories:

- a *routing table* containing neighbors' keys, each one with a rating parameter and a timestamp which represent, respectively, the number of forwarded messages, and the time of the last forwarding[1] performed by that neighbor;
- a *knowledge base* which contains OWL-S profiles of services that have been cached by the peer, along with their keys and the keys which identify their providers;
- a *partial table of services* containing only the keys of the services and related providers which were forwarded by the peer, without caching the OWL-S profile.

In the following we illustrate how the routing algorithm is involved in different aspects which characterize the system.

[1]Thanks to the timestamp it is possible to implement a *least recently used* policy to manage the entries of the routing table, which has a fixed size.

3.3.1 Overlay Network Join

The overlay network's growth is driven by a distributed algorithm which routes incoming nodes towards nodes which best match their interests. A number of *public* peers with no interests, forming a complete graph, is supposed to be always running and reachable, in order to provide known and stable access points for joining peers. Public peers are the starting point for interest-based node propagation, but they are neither central nodes in the growing overlay network, nor bridges between groups of peers with different interests. In fact, most connected nodes are not the public peers, but nodes with most widespread interests. Having no interests, public nodes are not involved in service publications and searches, and the (unlikely) failure of one of them has the only effect of reducing the number of possible access points to the network.

Interest-based node propagation is a distributed iterative algorithm, since every involved node performs the same procedure, defined in the following of this section.

Let A be a new node joining the network, and B an already connected node to which A has been redirected. Peer A sends its key k_A to B, which extracts the first n_A blocks, *i.e.* a k_A^- key equivalent to $k_A \setminus \text{uid}_A$. Let RT_B the routing table of peer B. Each k_P^-, with $P \in RT_B \cup B$, is compared by B, block by block, with k_A^-. At the end of the matching process, peer B has a ranked list of neighbors, ordered by the number of interests they share with A. If many peers have the same highest matching degree, the one whose uid is much closer to uid_A is chosen.

It may happens that no peer in RT_B, including B itself, matches A's interests. In that case, k_A^- is compared to the keys of the services whose advertisements are stored in B's partial table of services. The selection criterion is the same of the one which is used for the RT_B-based matching.

If also this process fails, peer B compares k_A^- with the categories of the services whose OWL-S profiles are stored in its knowledge base.

The last option, used if the knowledge base does not provide useful information for semantic routing, is to extract from RT_B a random peer among the first α peers with highest popularity.

Let C be the peer resulting from the matching phase. If $C \equiv B$, the forwarding process stops at peer B. Otherwise, the connection request ofr peer A is forwarded to peer C, whose popularity is increased by 1 in RT_B. Since the join process of a new peer could run into an infinite loop, temporary tables are used to keep track of already visited nodes.

The choice of privileging the partial table of services over the knowledge base, as alternatives to the routing table in the interests matching process, is not casual. If both the interests of peer B and its routing table RT_B do not match with peer A's interests, it is free from doubt that the information in the knowledge base of peer B is less relevant than the content of its partial table of services. The reason is that the knowledge base contains the advertisements of services which match with the interests of peer B, while the partial table contains information about services in which peer B is not directly interested. Thus, a check on the partial table of services helps the join process of those peers which enter the network from an access point which is far from their interests.

Let X be the last node in the peer forwarding process. When A connects to X, both peers exchange their keys and put them in their routing tables. Knowledge bases are not exchanged.

3.3.2 Service Publication

The service publication process uses the semantic routing algorithm in order to propagate advertisements towards nodes which are interested in the same categories of the services which have to be published. The routing table is the best source of information for supporting the publication process.

Supposing peer A wants to publish service s_{1A}, the first step consists in the creation of the $k_{s_{1A}}$ key, based on the OWL-S profile of the service. Next, peer A compares each k_P^-, being $P \in RT_A$, with $k_{s_{1A}}^-$, to obtain a ranking ordered by decreasing number of interesting categories. At the end of this phase, peer A sends $k_{s_{1A}}$ and the OWL-S$_{s_{1A}}$ description to the first β peers of the list, which cache received data in their knowledge base and eventually carry on the propagation process. If RT_A does not contain keys of peers which are interested in the service, the advertisement is sent to the γ most popular peers. This strategy implies that peers which are not interested in s_{1A} receive its key and OWL-S profile. These peers do not cache received data in their knowledge base, but store $k_{s_{1A}}$ and k_A in their partial tables of services, and propagate both $k_{s_{1A}}$ and OWL-S$_{s_{1A}}$ with the same strategy adopted by peer A.

As for the join process, temporary tables are used in order to avoid loops.

3.3.3 Service Discovery

To search for a service, the semantic routing algorithm is used in order to propagate queries towards nodes which are interested in the same categories of the services which have to be located.

If peer A needs a specific service s_q, first of all it builds a generic OWL-S$_{s_q}$ profile, including the required categories, and the related $k_{s_q}^-$ key. Then, peer A performs a semantic matching between OWL-S$_{s_q}$ and its knowledge base. If no result is found, peer A compares $k_{s_q}^-$ with the keys which are stored in its partial table of services. If the resulting ranked list of interested peers is not empty, peer A forwards the query (containing OWL-S$_{s_q}$ and $k_{s_q}^-$) to the first δ peers of the list. Otherwise, the matching is performed on the routing table.

The process is iterative, *i.e.* it is performed by each peer which receives the query. To avoid loops, temporary tables are used.

3.4 Churn Management

Node arrival and departure (*churn*) has many effects, in a P2P system: data unavailability, routing table inconsistency, overlay network fragmentation. Adequate strategies must be implemented in order to guarantee the system's adaptivity to any churn rate, resulting in a constant performance degree of all its functionalities.

In this sense, the proposed architecture supports both declared disconnections and node failures. A first recovery mechanism is executed when a node decides to leave the network. Parts of its routing table are sent to *interested* neighbors (the matching algorithm is the one we described above). Moreover, to avoid network fragmentations, the neighbors of the leaving node check their routing tables, and eventually reconnect to the network through public nodes (repeating the process we illustrated in section 3.3.1).

In case of node failure, the previous mechanism cannot be triggered. Nevertheless, there are several moments during the node lifecycle in which it is possible to detect a node failure, and to take the appropriate countermeasures. For example, during the connection phase each node is redirected towards peers which increasingly match its interests. If one of them cannot be contacted, it means that it undeclaredly leaved the network and some peers (which were its neighbors) have to update their routing tables, starting from the one which redirected the joining peer towards the departed node. Once the inconsistencies have been eliminated, an alternative route is proposed. A similar procedure is adopted when publication and discovery messages are routed among peers.

4 Simulation Results

We implemented the conceptual framework described above as a set of protocols for the PeerSim simulator [9].

One of the issues which arise when simulating a complex architectural model is the out-of-hand growth of memory requirements, in particular when the network size increases by factors of ten. In our case, a 10^6 nodes network implemented in PeerSim does not run even on a 2GHz AMD Opteron 246 (a 64bit machine) with 1024KB cache and 6GB RAM. The good new is that the number of interests and their distribution, rather than the number of nodes, affects the performance of the architectural model For this reason we can be satisfied for the simulation of a 10^5 nodes network, which seamlessly runs on a 1GHz Intel Pentium IV with 512KB cache and 1GB RAM.

We introduced a large set of parameters which affect the system behavior, but for this work we chose to operate only on the most significant subset, which includes: *network size*, *number of interests of each node*, and *search range* (*i.e.* considered number of hops for service discovery messages). Other variables have been fixed to reasonable values, minimally affecting the computational complexity. In particular, the total number of available categories has been set to 10^4, which is the order of magnitude of common taxonomies, such as NAICS [7] and UNSPSC [10].

Each node's interests are extracted from the whole set of categories, according to a uniform distribution. Future work will investigate cases of non-uniform (*e.g.* Zipf-like) distributions of interests.

Finally, the churn rate has been modelled to achieve a total 10% fraction of compensated departures, during the oscillatory phase.

4.1 Precision and Recall

Precision and Recall indicators allow to evaluate the efficacy of the peer selection mechanisms in the discovery process. In details, we measured the following indicators (for different values of the search range):

- Precision_peer = for a given query, how many of the peers that were selected had relevant information.
- Recall_peer = indicates, for a given query, how many of the peers that had relevant information were reached.
- Precision_service = indicates how many of the returned service advertisements are relevant.
- Recall_service = states how many of the total number of relevant service advertisements in the network are returned.

The Recall indicator is the most important, since it discloses the performance of the discovery algorithm in terms of information extraction capacity. The Precision indicator, on the other side, emphasizes the faculty of pointing searches towards useful paths.

Figure 2 illustrates how the indicators change depending on the maximum number of traversable nodes, in the case of a 10^5 nodes network. The analysis of these results shows that, for both peers and services, the Recall is greater than the Precision. The reason of this fact is the presence of connections among peers which do not share the same interests and which are not interested in the same service categories of the query generator, but through which messages must be propagated in order to reach more suitable peers. We can finally observe that the average number of interests of each peer does not affect these indicators.

4.2 Query Hit Ratio

The Query Hit Ratio (QHR) is the fraction of successful queries, *i.e.* queries with at least one useful response. Typically, QHR values over 90% denote high efficiency of the search algorithm, but also a good distribution of service advertisements in the network.

Figure 3 illustrates how the QHR changes depending on the maximum number of traversable nodes, for a network of 10^5 nodes. We can observe that, over a search range threshold the QHR is constant. The search range

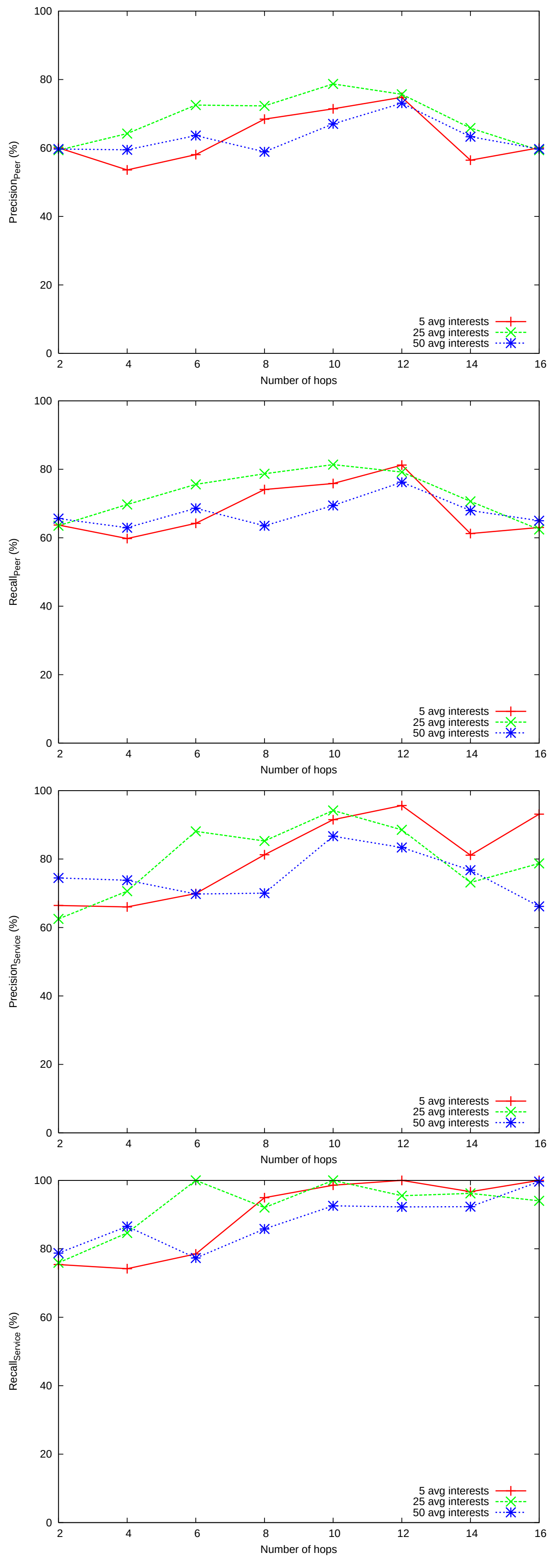

Figure 2. $Precision_{Peer}$, $Recall_{Peer}$, $Precision_{Service}$ and $Recall_{Service}$ of a 10^5 nodes network. In each window, three graphs are shown, each one referring to a different average number of interests of each peer.

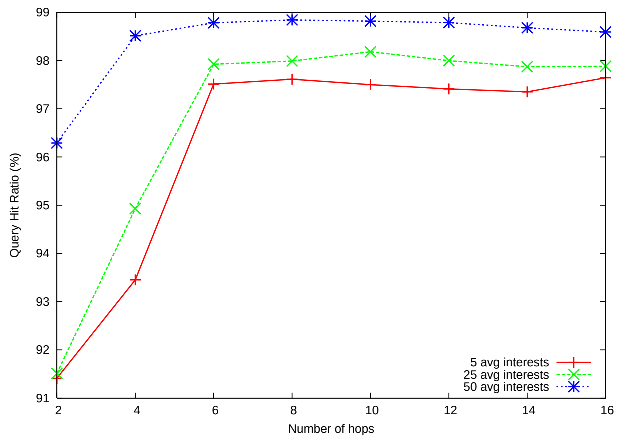

Figure 3. Query Hit Ratio versus search range (*i.e.* maximum number of nodes which can be traversed by a query message).

threshold is approximately equal to the average number of redirections a node is subject to, when joining the overlay network. That is, the search range threshold equals the average number of nodes that a peer traverses in order to reach an optimal position in the network, according to the matching criteria defined by the architectural model.

The efficiency of the search algorithm is proven by QHR values which are over 90%, when the search range is over the threshold. The ratio between the network size and the average number of interests has emerged as a fundamental parameter, since a larger number of service providers requires more specific queries from service consumers, in order to guarantee a high efficiency level.

5 Conclusions

In this paper we have introduced our interests-based architectural model for service-oriented peer-to-peer networks. We have clearly illustrated the overlay construction algorithm, the service publishing and discovery strategies, and the churn management mechanisms.

Sophisticated simulations of the overlay network dynamics have shown several emerging characteristics, such as the nontrivial node degree distribution, and the dependence of performance indicators on the relation between the average number of interests of each peer, the total number of service categories, and the network size. The Recall indicator has disclosed the efficacy of the search algorithm in terms of information extraction capacity, while the Query Hit Ratio (whose values are always over 90%) has proven its efficiency.

There are many interesting directions in which this work can be taken. The architectural model should be analitically formalized, in order to investigate the predictability of the features which have been experimentally observed. Moreover, the framework will be implemented in a real platform like JXTA middleware, whose code requires some refactoring to allow the substitution of default routing protocols.

6 Acknowledgements

This work has been partially supported by the Italian Ministry for University and Research (MIUR) within the project PROFILES under the PRIN 2006 research program.

7 References

[1] X. Bai, S. Liu, P. Zhang, and R. Kantola, ICN: Interest-based clustering network, *Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P 2004)*, Zurich, Switzerland, 2004, pages 219-226.
[2] A. Iamnitchi and I. Foster, Interest-aware information dissemination in small-world communities, *Proceedings of the 14th IEEE International Symposium on High Performance Distributed Computing*, Research Triangle Park, NC, USA, 2005, pages 167-175.
[3] C. Sangpachatanaruk and T. Znati, A community-based, agent-driven, P2P overlay architecture for personalized web, *Proceedings of the 8th Asia-Pacific Web Conference (APWeb)*, Harbin, China, 2006, Proceedings, pages 616-627.
[4] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch, Edutella: A P2P Networking Infrastructure based on RDF, *Proceedings of WWW2002*, Honolulu, Hawaii, USA, 2002.
[5] P. Haase, R. Siebes, and F. van Harmelen, Peer selection in peer-to-peer networks with semantic topologies, *Proceedings of the International Conference on Semantics of a Networked World: Semantics for Grid Databases*, Paris, France, 2004.
[6] OWL-S: Semantic Markup for Web Services, *http://www.w3.org/Submission/OWL-S/*, November, 2004.
[7] North American Industry Classification System (NAICS), *http://www.census.gov/naics/*.
[8] G. A. Miller, C. Fellbaum, R. Tengi, P. Wakefield, R. Poddar, H. Langone and B. Haskell, WordNet - a lexical database for the English language, *http://wordnet.princeton.edu*.
[9] M. Jelasity, A. Montresor, G. P. Jesi and S. Voulgaris, PeerSim: A Peer-to-Peer Simulator, *http://peersim.sourceforge.net*.
[10] United Nations Standard Products and Services Code (UNSPSC), *http://www.unspsc.org*.