A Semantic Search Algorithm for Peer-to-Peer Open Hypermedia Systems

Jing Zhou, Vijay Dialani, David De Roure, and Wendy Hall

University of Southampton, Southampton SO17 1BJ, United Kingdom,
jz00r,vkd00r,dder,wh@ecs.soton.ac.uk

Abstract. We consider a collaborative application scenario in Open Hypermedia Systems. We describe a semantic search algorithm to discover semantically equivalent or related resources across distributed link databases, otherwise known as linkbases. Our approach differs from traditional crawler based search mechanisms because it relies on clustering of semantically related entities. It creates clusters of related semantic entities to expedite the search for resources in a random network. It uses a distance-vector based heuristic to guide the search. Our results confirm that the algorithm yields high search efficiency in collaborative environments where the change in content published by each participant is rapid and random.

1 Introduction

Open Hypermedia [20] is a model that has been adopted by the hypertext community for many years. It is principally characterized by having hypermedia link information stored separately from the documents that it describes. The links are stored in linkbases. This approach allows links to be managed and maintained separately from the documents, and that different sets of links can be applied to a set of documents as appropriate.

The development of the first Open Hypermedia System ("Microcosm" [12]) predates the Web. The first implementation of the Microcosm philosophy on the Web was the Distributed Link Service (DLS) [5] [11]. This was extended so that link resolution was also distributed around the Web [10], and the service paradigm now extends to recent developments such as ontology services [7]. COHSE [7] provides tools for the Semantic Web that builds upon the concept of the DLS and ontologies.

The Semantic Web [2] augments current Web technologies by associating machine understandable annotations (a.k.a. metadata) with contents. Metadata provides an abstract representation of information and is primarily produced to facilitate inference techniques to co-relate information from different providers. Current search techniques used in Semantic Web technologies focus on annotating static information and fail to take into consideration dynamic and asynchronous variation in contents. It should be noted that, though some may consider services based architectures like DAML-S [1], which use Semantic Web technologies, to be a form of dynamic content system, we differ from [19] and consider it to be an application of the Semantic Web to active entities rather than dynamic entities. We consider the Semantic Web to be dynamic, if it is created spontaneously by a set of collaborating nodes, where each node can dynamically update its published contents. While Semantic Web technologies are generic in their application, in this paper, we restrict ourselves to their application in collaborative environments, which facilitates resource sharing between dynamic collections of participants. As a participant can act both as a resource provider and a resource consumer, a peer network is constituted by collaborating entities. Resources are owned by individual participants and are subject to asynchronous updates, with a requirement to propagate updates to the current resource consumers. Peers collaborate to locate semantically equivalent or related entities.

Efficiency of any search algorithm in peer networks critically depends on peer topology and query routing. Two approaches: centralised and Distributed Hash Techniques (DHTs) and their hybrids are widely employed to organise peer networks. A centralised search uses specialised nodes to maintain an index of resources available within a collaborative environment. The resource of interest is located by querying an index node to identify the resource providers. Napster [15] employs an efficient centralised search mechanism. However, a centralised system is vulnerable to attacks and poses difficulties in updating the indices. On the other hand, DHTs are widely adopted to improve the resilience of peer-to-peer systems. Examples include CAN [16], Chord [18] and Pastry [17]. Typical DHTs resolve a keyword to a location where the contents are located or from where the contents can be routed from. The inherent self-organisation is attributed to the distribution of keys in a uniform space where node and object identifiers share the same key space. Adopting DHTs requires unique hash techniques that could transform the search criterion into a unique key set. However, such a technique is not suitable for semantic search mechanisms, as a typical semantic search may consist of a random combination of entities and the relationships between them. Any search technique employed should be able to search on a set of related entities rather than a single hash expression.

Another important aspect that differentiates the semantic overlay from DHTs is the necessity to maintain relationships between resources (in our case "linkbases") of participants. In a DHT, immediate neighbours do not have to share any relationship and are primarily responsible for monitoring the connectivity with neighbouring peers. While in a collaborative environment, the arrival of a peer modifies the relationship with its neighbours as more potentially discoverable resources are added to the network. The departure of a peer invalidates its relationship with neighbouring peers. Hence the scope of an update is not limited to the peer storing the discovery information of the resources, but to all the peers that are semantically connected to the arriving or departing peer.

In such collaborative environments, most searches are not required to be exhaustive [13]. The maximum achievable level of recall (i.e. percentage of the matches that can be found) is traded off for better performance of the system.

The keyword-based search does not suffice for a collaborative environment primarily consisting of peers with resources that may or may not be semantically correlated with each other, since it is not inherently designed to utilise such relationships between keywords or terms. For example, we define that term tp is "parent-of" term tc if tp is semantically a super class of tc. Semantically equivalent terms convey similar meanings in terms of semantics though being syntactically unequal.

1.1 Summary

As discussed in [16], peer-to-peer search techniques are highly effective in systems with dynamic content. However, such search techniques rely on the uniqueness of the "search key" or "hashed key" in the entire peer network. In our case, the resources, though unique in their types, are varied in their instantiations. Our aim is to devise mechanisms that allow the inspection of the peers by the resource type and return the information about the occurrences of these types. A similar problem in peer-to-peer networks - that of service discovery, has been addressed in [8]. In [8], authors suggest the use of hash techniques to generate a unique identifier for the resource, in their case a service. The information about the occurrences of a services is then associated with hashed keys. We are critical of such an approach for the following reasons:

1. A typical topic distribution is more likely to follow a Zipf pattern rather than exhibiting a uniform distribution. Using the same hash key to represent each of the services is bound to the formation of hot spots.
2. In a DHT technique, there is an inherent assumption about the global trust between the participating peers. It assumes that privacy is retained even by the publication of objects hosted by individual peers.

In addition, we are equally concerned about time to stabilize in a peer-to-peer network. Our algorithm, as described in section 3, minimizes the stabilisation time by restricting the cached information to the neighbours of peers.

2 Application Scenario

We conceive an Open Hypermedia System called the Distributed Dynamic Link Service (DDLS) based on a peer-to-peer architecture. To the best of our knowledge, most implementations of the DLS maintain linkbases on the link server side; however, one of them proposes [6] the basic network model for a decentralised style of link service, where linkbases are located both at the client side and on the remote server.

The DDLS is a complementary hypermedia service that clients inquire about a set of linkbases. By decentralizing both linkbases and link service components amongst peers, the DDLS enables link resolution and linkbase communication for multiple online users who want to share their link resources. Linkbases are

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF  xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
    xmlns:rdfs=http://www.w3.org/2000/01/rdf-schema#
    xmlns:lb="http://www.semanticweb.com/rdf/linkbase-ns#">
    <rdf:Description about="http://www.semanticweb.com/linkbase
          /research/linkbase.xml">
        <rdf:type resource="http:// www.semanticweb.com/rdf
          /linkbase-ns#Linkbase" />
        <lb:topic>theory</lb:topic>
    </rdf:Description>
</rdf:RDF>
```

Fig. 1. A Linkbase Expressed in RDF Syntax

maintained locally and provide data mobility with minimal constraints - a feature unique to the DDLS amongst the Open Hypermedia System implementations.

Linkbases constitute a part of the resources of a DDLS, as its primary objective is to serve links. We employ RDF [14] to encode information about linkbases in sets of triples that associate metadata with linkbase. Properties associated with linkbases in the DDLS are described using RDF schema [4]. A linkbase of the DDLS is expressed in RDF syntax as in Fig. 1. We observe that each resource (a linkbase) can be represented by topics that convey its prominent content. Topics associated with a linkbase form a “topic vector”. Peers may instigate a query to retrieve matching linkbases. As linkbase providers, peers are allowed to add, delete and update their linkbases. A typical query expression on these linkbases is defined in Section 3.3. A query expression specifies a list of topics and a list of logical operators. A query result consists of a set of semantically related topics as specified in the query. However, it should be noted that, the semantic similarity can be described in different ways and we assume the existence of such a mechanism. It may exist in the form of a controlled vocabulary or may be based on inferencing techniques, or otherwise.

3 Our Contribution

We describe a search algorithm that allows semantic search over a set of semantically related entities. As described by DAML search mechanisms [9], a semantic search should facilitate the lookup for resources expressed as a combination of entities and relationships connecting them, i.e. subject and predicate based search. However, DAML imposes no restriction on the type of entities that can be used for describing resources or the type of relationships that connect them. A typical search may be based on the subject or predicate or a combination. The DAML’s crawler based approach creates a connected graph to facilitate the search for related entities. However, the DDLS does not permit the creation of any such centralised search mechanism. Following subsections describe an algorithm that creates a peer topology based on the semantic contents hosted by individual peers.

3.1 Notations

Table 1. Notations Employed in the Algorithm Description

G (Peer network topology)	G (V_t,E_t)	(Set of nodes, Set of edges){Note: Both are time dependent functions.}
$P_i \in V_t$ (Peer 'i' in graph G)	Id^{peer}, LP, LT	(Peer: identifier, list of neighbours, list of topics)
LT_i (List of topics published by the individual peer)	Id^{topic}	(Topic: identifier)
LP_i (List of semantically related neighbouring peers)	Id^{peer}[], LT^{common}[], α^{dist}, $\beta^{direction}$	(List of peers, with the common semantic information LT^{common} at distance α^{dist} and $\beta^{direction}$ of the edge)

Consider a graph G, which consists of a dynamic list of peers, each peer is uniquely identified by an identifier to route messages to individual peer. Each peer P_i publishes a list of topics LT_i, the list of topics can be asynchronously updated by individual peers. Each peer maintains a list of neighbouring nodes that hold semantically equivalent or related topics to facilitate entity-based clustering. Initially, when a new peer P_{new} joins the network G, it contacts a set of randomly selected peers represented by a set P_{random}. P_{new} exchanges the list of topics LT_{new} with each of the randomly selected peers (P_{random}). Execution of the algorithm leads to the formation of an overlap between P_{new} and the peers in set P_{random}.

Variables: $LT_i := 0$, $LP_i := 0$, when P_{new} joins graph G,

– Online := true
– Alow Queries := false
– Randomly select a set of peers P_{random} from the identifier space, or use the multicast to randomly select a set of peers

3.2 Process the Individual Responses ($P_{response}$) from Each Peer in P_{random}

– For each received $LT_{response}$ from the randomly chosen peers,
  - Calculate α^{dist} := number of topics in $LT_{response} \bigcap LT_{new}$
  - Add to LP_{new} the list of peers, distance, and the intersection set with $\beta^{direction}$:= true

- If received $\mathrm{P}_{response}$ already exists in LP_{new}, select another set of peers
- If $\mathrm{LT}_{response} \bigcap \mathrm{LT}_{new} = \{\}$, store the information as a uni-directional set where LP_{new} contains the list of peers at α^{dist} = null and the intersection set with $\beta^{direction}$ = false

The above algorithm leads to the creation of graphs representative of partial information available at individual peers. This procedure is carried out in parallel on each of the peers in set P_{random}. In combination, the algorithm leads to the creation of an overlay with clustered information. The randomly connected graph consists of peers that are able to determine the semantic distance to other peers via α^{dist}. However, not all the peers may have an overlap in the semantic description of the resources that they publish individually. Hence, in cases in which there exists no relationship between semantic information of the neighbouring peers, information is stored as unidirectional information with α^{dist} := null.

A semantic search expression is evaluated against the information held on individual peers. The query initiator formulates the query and calculates the distance between the query expression and cached information LT_i. In case the query evaluator finds a perfect match, it routes the query to the list of Id^{peer} [] for the particular entry. The query is then successively evaluated by each of the recipient peers.

3.3 Search Algorithm

The approach for organising peers as discussed in Section 3.2 leads to the creation of clusters of information, whereby each peer stores partial information about peers holding semantically related entities. The proximity of entities is measured by a relative distance represented by α^{dist}. The distance between peers is measured as the amount of overlap between their topics. We use this distance information to propagate the search queries amongst peers. Any of the participating peers can initiate a semantic search. The search is evaluated against the initiating peer's cached information to determine the distance between the query expression and the cached information about the neighbouring peers. In certain cases where there may be no overlap between the query and the cached information, the query is propagated to all the neighbours of the recipient peers. A typical query expression is defined in Fig. 2.

Query processing at Peer $\mathbf{P}_i$ Let LT_{query} represent the list of topics in the query. Let n represent the number of topics in LT_{query}.

– For each α^{dist} in $\mathrm{LP}_i \geq n$,
  - If $\mathrm{LT}_{query} \bigcap \mathrm{LP}_i \rightarrow \mathrm{LT} = \mathrm{LT}_{query}$, propagate the query
  - If $\mathrm{LT}_{query} \bigcap \mathrm{LP}_i \rightarrow \mathrm{LT} = \{\}$, forward the query to all the neighbours in LP_i

This heuristic propagates the query to peers with similar information. If there is no overlap between the query and the information available at a peer, it uses the neighbour broadcast.

```
<rdfq:rdfquery>
  <rdfq:From eachResource="http://www.semanticweb.com/
         collabrative_environment_x/peer_linkbase">
    <rdfq:Select>
      <rdfq:Condition>
        <rdfq:and>
          <rdfsq:sequal>
            <rdfq:Property name="lb:topic"/>
            <rdf:String>Theory</rdf:String>
          </rdfsq:sequal>
          <rdfsq:sequal>
            <rdfq:Property name="lb:topic"/>
            <rdf:String>Practice</rdf:String>
          </rdfsq:sequal>
        </rdfq:and>
        <rdfq:or>
          <rdfsq:sequal>
            <rdfq:Property name="lb:topic"/>
            <rdf:String>Thoughts</rdf:String>
          </rdfsq:sequal>
        </rdfq:or>
      </rdfq:Condition>
    </rdfq:Select>
  </rdfq:From>
</rdfq:rdfquery>
```

Fig. 2. A Typical Query Specification

4 Experiments

Our experimental evaluation is divided into three parts. The first experiment demonstrates the convergence of a query in a controlled environment, where the topic list is assumed to be static. Our test environment consists of a set of peers. At bootstrap, each peer is allowed to randomly select a random number of distinct topics. It then simultaneously selects a group of neighbours. As each peer builds its overlay, it maintains the information about the semantically related entities, as mentioned in Section 3.

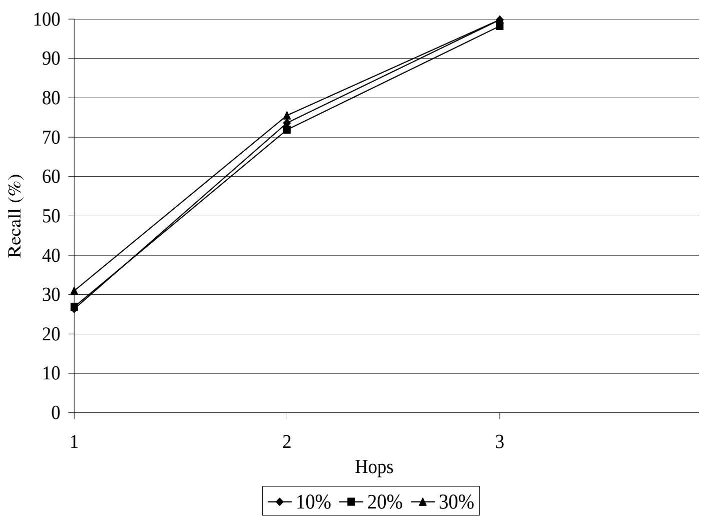

Fig. 3. Algorithm Performances with Static Peers

Fig. 3 represents the average performance of the algorithm in an environment consisting of 100 peers for 50 consecutive runs, with static content, i.e. their individual content topics are kept unchanged throughout the simulation. Each peer could cache the published topics of 30% of the total peers. It chooses a randomly selected list of topics from a global list of 300 entities. For ease of simulation, we impose an upper bound on the maximum topics that each peer could publish. To accurately measure the recall, we use a probabilistic distribution to ensure a specific percentage of peers hosting semantically related topics. Our aim is to determine the number of hops required to achieve the maximum recall. We measured the recall for topics with 10% to 30% of the distribution. The clustering ability of the algorithm should ideally increase the effectiveness of the search as the percentage of peers publishing semantically related entities increases. As the peers are randomly organized, query routing may depend on the cache rate (i.e. percentage of the cached peers to all peers in the system) of the instigating peer, we overcome this limitation by randomly choosing a peer within the network to instigate a random query and measure the average performance over a number of runs.

The search algorithm performs very well in the controlled environment. At least 98.24% of peers with query topics are located within 3 hops from the query peer under varying percentages of related entities. With 30% peers having related topics, the recall level reaches 99.86%. The probability of a peer to locate other peers with query topics tends to be higher when more peers have related topics, which potentially leads to a more densely clustered overlay. Clustering of related peers leads to the formation of information islands. We use neighbour broadcast to propagate the query between disjointed clusters.

Our next experiment evaluates the search performance while topics published by individual peers vary dynamically. Updates are propagated to immediate neighbours. We carried out the experiment with 100 peers by varying the percentage of peers that dynamically update their published topics. The experimental settings were retained. Fig. 4 demonstrates the performance of the algorithm, where a selected percentage of peers update their published topics.

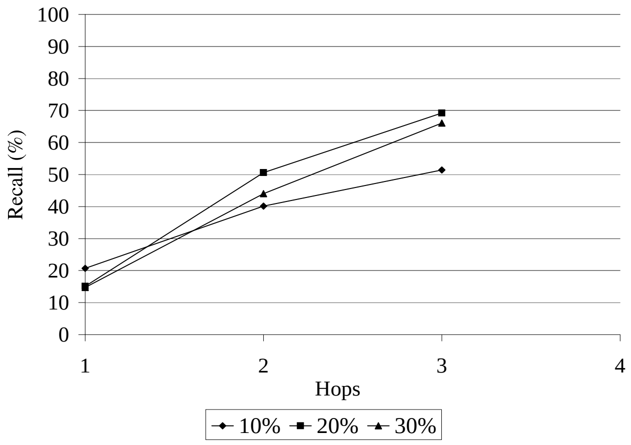

Fig. 4. Performances with Dynamically Evolving Peers

As expected, the search performance deteriorated in comparison to the static environment. With 20% peers updating their published topics dynamically, the recall level reaches 69.23% within 3 hops. The recall decreases to 51.45% with 10% peers updating topics. While individual peers are responsible for informing their neighbours of a change in published topics, notifications are propagated asynchronously and have precedence over the query propagation. If a query is instigated to locate the peers that happen to update their topics before the cached information has been refreshed, the search may result in missing peers due to stale information maintained about the published topics.

It was observed that individual simulations failed to discover the entities in certain cases in which the updated information was unavailable. One of the many reasons is that peers with query topics may have not been incorporated into the semantic overlay due to the reorganisation of the overlay. Without any guarantee of the synchronisation of all updates of dynamically evolving peers, the search algorithm performance heavily depends on timely notifications.

From the simulation results we found that, the search algorithm reaches the highest recall within 3 hops from the query peers in a network of 100 collaborating peers. In our third and final experiment, we evaluate the performance of the algorithm for a set of peers with varying degree of cache rate and examine its effect on the hops within which the potentially highest recall could be achieved.

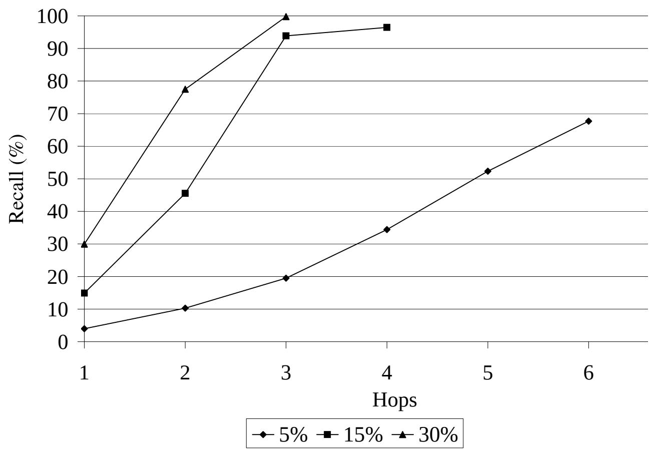

Fig. 5. Hop Counts with Varying Cache Rates

We performed the experiment with 100 peers in a controlled environment and measured the performance for topics with 30% distribution. We retained the condition on upper bounds for published topics. Simulations varied the cache rate from 5%, 15% to 30%. It is shown in Fig. 5 that with the cache rate of 5%, the search algorithm obtains its highest recall within 6 hops. When the cache rate rises up to 15%, 93.87% peers with required query topics can be located within 3 hops while the highest recall of 96.46% is achieved within 4 hops.

The experiment also disclosed it to us that the cache rate not only affects the hops needed but also restricts the highest achievable recall. When the cache rate was varied between 5% and 30%, the number of hops needed for the highest recall to be achieved falls from 6 down to 3. In the meanwhile, the potential highest recall rises from 67.69% up to 99.78% with the same range of cache rate variation.

5 Future Work

The above algorithm leads to the creation of a semantic overlay consisting of clusters of related information. Changes to cached entries of individual peers yield continuous reorganisation of the overlay, which affects the query performance. Probable heuristics that improve the query performance for varying query profiles are under investigation.

Our algorithm assumes that the relation between the semantic entities is statically defined and does not cater for the environments where semantic relationships are constantly redefined. In our case, semantic entities defined for the DDLS are immutable, while the same does not apply to Grid Services environment. However, in a Grid Services Environment the semantic relation between the semantic entities may not be immutable and may be subject to conditional evaluation.

In addition, our algorithm does not take into account the trust mechanisms between individual peers. As highlighted by [3], the peer-to-peer approach in OHS is subject to the existence of adequate trust mechanisms, as is the case with evolving peer-to-peer networks in general. The DDLS relies on the faithful delegation of trust between the participating peers, which requires further investigation.

6 Conclusions

This paper presented a semantic search algorithm for the collaborative Open Hypermedia System that creates a semantic overlay of related entities and uses clustering to optimise the search. Our algorithm performs very well in controlled environments with static content. The number of hops required to achieve the same percentage of recall varies in direct proportion to the cache rate between topologies. The search algorithm also performs satisfactorily when peers update their topics randomly and has been proven suitable for locating information in an environment where peers change their contents randomly. However, clustering of related entities at times leads to the formation of information islands and the ways to reorganise the topology in terms of published contents also form a part of the future study.

References

1. Ankolekar, A. et.al.: DAML-S: Semantic Markup for Web Services. Proceedings of the International Semantic Web Working Symposium (SWWS)(2001)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American (2001)

3. Bouvin, N. O.: Open hypermedia in a peer-to-peer context. Proceedings of the thirteenth conference on Hypertext and hypermedia, College Park, Maryland, USA (2002) 138–139
4. Brickley, D., Guha, R. V.(ed.): RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft. http://www.w3.org/TR/rdf-schema/ (2003)
5. Carr, L. A., De Roure, D. C., Hall, W., Hill, G. J.: The Distributed Link Service: A Tool for Publishers, Authors and Readers. Proceedings of the Fourth International World Wide Web Conference: The Web Revolution, Boston, Massachusetts, USA (1995) 647–656
6. Carr, L. A., De Roure, D. C., Hall, W., Hill, G. J.: Implementing an Open Link Service for the World Wide Web. World Wide Web Journal, Vol.1, No. 2 (1998)
7. Carr, L. A., Hall, W., Bechhofer, S., Goble, C. A.: Conceptual Linking: Ontology-based Open Hypermedia. Proceedings of the Tenth International World Wide Web Conference, Hong Kong (2001) 334–342
8. Castro, M., Druschel, P., Kermarrec, A., Rowstron, A.: One Ring to Rule them All: Service Discovery and Binding in Structured Peer-to-Peer Overlay Networks, url = "citeseer.nj.nec.com/530455.html"
9. Dean, M., Barber, K.: DAML Crawler. http://www.daml.org/crawler/ (2002)
10. De Roure, D. C., Carr, L. A., Hall, W., Hill, G. J.: A Distributed Hypermedia Link Service. Proceedings of the Third International Workshop on Services in Distributed and Networked Environments (SDNE96) (1996) 156–161
11. De Roure, D., Walker, N., Carr, L.: Investigating Link Service Infrastructures. Proceedings of ACM Hypertext 2000 (2000) 67–76
12. Fountain, A., Hall, W., Heath, I., Davis, H. C.: Microcosm: An Open Model for Hypermedia with Dynamic Linking. In Rizk, A and Streitz, N and Andre, J, (ed.) Proceedings Hypertext: Concepts, Systems and Applications, Proceedings of ECHT'90, Paris (1990) , 298–311
13. Ganesan, P., Sun, Q., Garcia-Molina, H.: YAPPERS: A Peer-to-Peer Lookup Service Over Arbitrary Topology. Proceedings of IEEE INFOCOM, San Francisco, California, USA (2003)
14. Lassila, O., Swick, R. R.: Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, http://www.w3.org/TR/REC-rdf-syntax (1999)
15. Napster. http://www.napster.com
16. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A scalable content-addressable network. Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications, San Diego, California, USA (2001) 161–172
17. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany (2001)
18. Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., Balakrishnan, H.: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. Proceedings of the 2001 ACM SIGCOMM Conference, San Diego, California, USA (2001) 149–160
19. Sycara, K., Lu, J., Klusch, M., Widoff, S.: Dynamic Service Matchmaking among Agents in Open Information Environments. Journal ACM SIGMOD Record, Special Issue on Semantic Interoperability in Global Information Systems, Ouksel, A., Sheth, A.(ed.) (1999)
20. Wiil, U. K.: Open Hypermedia: Systems, Interoperability and Standards. Journal of Digital information, Vol.1, No.2 (1997)