

Super-Peer-Based Routing Strategies for RDF-Based Peer-to-Peer Networks

Wolfgang Nejdl^a Martin Wolpers^a Wolf Siberski^a
Christoph Schmitz^a Mario Schlosser^a Ingo Brunkhorst^a
Alexander Löser^b

^a*Learning Lab Lower Saxony, University of Hannover, 30167 Hannover, Germany*

^b*Computation and Information Structures Institute, Technical University Berlin, 10587
Berlin, Germany*

Abstract

RDF-based P2P networks have a number of advantages compared to simpler P2P networks such as Napster, Gnutella or to approaches based on distributed indices on binary keys such as CAN and CHORD. RDF-based P2P networks allow complex and extendable descriptions of resources instead of fixed and limited ones, and they provide complex query facilities against these metadata instead of simple keyword-based searches.

In this paper we will discuss RDF-based P2P networks like Edutella as a specific example of a new type of P2P networks - schema-based P2P networks - and describe the use of super-peer based topologies for these networks. Super-peer based networks can provide better scalability than broadcast based networks, and provide support for inhomogeneous schema-based networks, with different metadata schemas and ontologies (crucial for the Semantic Web). Based on (dynamic) metadata routing indices, stated in RDF, the super-peer network supports sophisticated routing and distribution strategies, as well as preparing the ground for advanced mediation and clustering functionalities.

Key words: Peer-to-Peer, Semantic Web, Schema-Based Routing, Distributed RDF Repositories, Routing Protocols, Semantic Networks

Email addresses: nejdl@learninglab.de (Wolfgang Nejdl),
wolpers@learninglab.de (Martin Wolpers), siberski@learninglab.de
(Wolf Siberski), schmitz@learninglab.de (Christoph Schmitz),
schlosser@learninglab.de (Mario Schlosser),
brunkhorst@learninglab.de (Ingo Brunkhorst),
aloeser@cs.tu-berlin.de (Alexander Löser).

1 Super-Peer Networks for Distributed RDF Repositories

P2P applications have been quite successful, e.g., for exchanging music files, where networks use simple attributes to describe these resources. A lot of effort has been put into refining topologies and query routing functionalities of these networks, and simple systems like Napster and Gnutella have inspired more efficient infrastructures such as the ones based on distributed hash tables (e.g. CAN and CHORD [1,2]). Less effort has been put into extending the representation and query functionalities offered by such networks, and projects exploring more expressive P2P infrastructures [3,4,5,6] have only slowly started the move toward schema-based P2P networks.

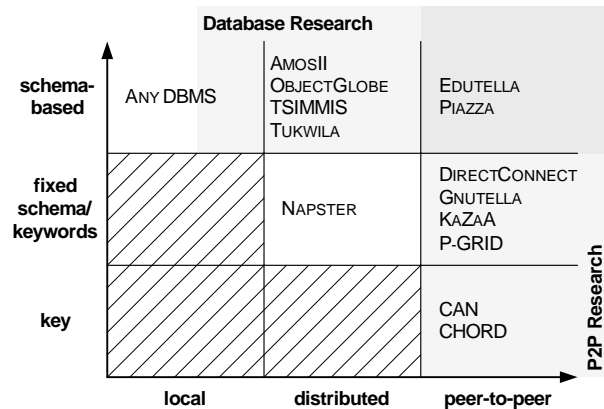


Fig. 1. Schema Capabilities and Distribution

Schema-based P2P networks finally allow us to solve the shortcomings of P2P networks with restricted and fixed metadata elements, by enabling truly expressive as well as distributed data and metadata repositories. They do this by building upon peers that use explicit and possibly heterogeneous schemas describing their content, and implementing sophisticated routing strategies based on this knowledge. This approach parallels recent developments for distributed data management, which also start to evolve towards a higher degree of distribution, but so far have usually assumed static distribution and query plans. As we show in figure 1 schema-based P2P systems are the next step in distributed P2P networks with advanced data management and storage features (see also [7]). Their architectures have to take into account the typical characteristics of P2P systems, i.e. *local control of data, dynamic addition and removal of peers, only local knowledge of available data and schemas and self-organization and -optimization*.

In the Edutella project [8,3,9] we have been exploring some issues arising in that context, with the goal of designing and implementing a schema-based P2P infrastructure for the Semantic Web. Edutella relies on the W3C metadata standards RDF and RDF Schema (RDFS) [10,11] to describe distributed resources, and uses basic P2P primitives provided as part of the JXTA framework [12].

Two other approaches are the ones investigated by Bernstein and Aberer. Bernstein et.al. [4] propose the Local Relational Model (LRM) enabling general queries to be translated into local queries with respect to the schema supported at the respective peer, using the concept of local translation/coordination formulas to translate between different schemas. Aberer et.al. [5] proposes schema-based peers and local translations to accommodate more sophisticated information providers connected by a Gnutella-like P2P topology.

The infrastructure we will discuss in detail in this paper is based on a so-called super-peer topology, where peers connect to super-peers that build up the routing backbone for the whole network (see [13] for the general characteristics of super-peer networks, and Kazaa, Grokster and Morpheus for existing super-peer systems). The responsibility of a super-peer is the efficient query and answer routing as well as the distribution and execution of query plans based on local routing indices at each super-peer. Further capabilities like mediation and transformation of queries and answers also can be implemented in a super-peer.

In the following we will start by describing the general topology for our schema-based super-peer network in section 2, and discuss how peers register to this network. In section 3 we describe the schema-aware routing indices, which are needed for query planning and routing within the super-peer network and from the super-peers to the associated peers. Section 3.5 discusses (distributed) routing based on these indices. Finally, section 4 briefly introduces the implementation of the super-peer network.

2 Super-Peer Networks for Schema-based P2P Systems

Peer usually are not created equal but have different characteristics with respect to their capabilities, e.g. available bandwidth, storage space or processing power. As discussed in [14], exploiting the different capabilities in a P2P network can lead to an efficient network architecture, where a small subset of peers, called super-peers, takes over specific responsibilities for peer aggregation, query routing and possibly mediation. The KaZaA network [15] uses a simple version of a super-peer based architecture, more elaborate versions are described in [16] and [17]. Super-peer-based P2P infrastructures usually exploit a two-phase routing architecture, which routes queries first in the super-peer backbone, and then distributes them to the peers connected to the super-peers. The last step can sometimes be avoided if the super-peers cache data from their connected peers. Super-peer routing is usually based on different kinds of indexing and routing tables, as discussed in [16] and [17].

2.1 The Edutella Super-Peer Topology

The Edutella super-peers [17] employ routing indices, which explicitly acknowledge the semantic heterogeneity of schema-based P2P networks, and therefore include schema information as well as other possible index information. This super-peer backbone is responsible for message routing and integration / mediation of metadata. Scaling a P2P network to a large number of super-peers while maintaining certain properties such as low network diameter requires guiding the evolution of the network topology upon peer joins and departures. In the Edutella network, super-peers are arranged in a hypercube topology, according to the HyperCuP protocol [18]. For a simple example see figure 2.

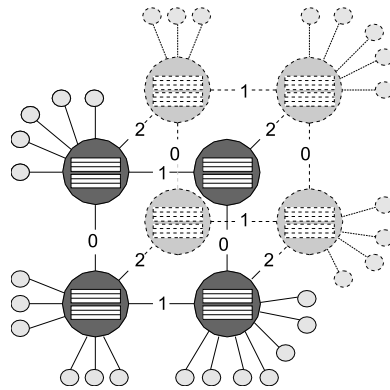


Fig. 2. HyperCuP Super-Peer Topology

A new super-peer is able to join the network by asking any other, already integrated super-peer which then carries out the peer integration protocol. $O(\log(N))$ messages are sent in order to integrate the new super-peer and maintain a hypercube-like topology. Any number of super-peers can be accommodated in the network: If some peers are “missing” in order to construct a complete hypercube topology which consists of 2^d nodes in a d -dimensional binary hypercube, some super-peers in the network occupy more than one position on the hypercube. When new super-peers join the network, they fill the gaps in the hypercube topology and possibly extend the dimensionality of the hypercube.

The advantages of HyperCuP are an efficient and guaranteed non-redundant query broadcast, which we will then further restrict by using the indices as described in section 3 and 3.5. For broadcasts, each node can be thought of as the root of a specific spanning tree through the P2P network. The topology allows for $\log_2 N$ path length and $\log_2 N$ number of neighbors, where N is the total number of nodes in the network (i.e. the number of super-peers in our case). Moreover, the topology is vertex-symmetric and thus features inherent load balancing among super-peers. Thus, we can use the topology to carry out efficient communication and message forwarding among super-peers: Certain updates (which we will communicate by broadcast to other super-peers) can be executed efficiently, without message overhead. Also, a path of $\log_2 N$ length exists between any two super-peers, thus any

two distinct schemas can be reached within a short number of hops from each other. Alternatives to this topology are possible, as long as they guarantee the spanning tree property for the super-peer backbone, which we exploit for maintaining our routing indices and for building our distributed query plans (section 3, section 3.5, and [19]).

Peers themselves do not connect to each other in super-peer networks. Instead they connect to a super-peer thus forming a star-like local network at each super-peer. In turn the super-peer takes over specific responsibilities for peer aggregation, query routing and possibly mediation. This topology allows the peers to use their resources more efficiently while the super-peer provides the necessary bandwidth and processing power to enable efficient and reliable query processing and answering.

2.2 Registering Peers at Super-Peers

In order to avoid broadcasting a query to all connected peers of a super-peer we introduce the super-peer/peer routing indices that are used to forward the query to the respective peers only. These indices store information about metadata usage at each peer. This includes schema information such as schemas or attributes used, as well as possibly conventional indices on attribute values.

On registration the peer provides the super-peer with its metadata information by publishing an XML registration message. This advertisement encapsulates metadata based description of the peers' properties. As the registration may involve quite a large amount of metadata, we build upon the schema-based approaches which have successfully been used in the context of mediator-based information systems (e.g. [20]). A peer must register at least one schema (e.g. the LOM or the DC element set) with a set of properties or with information about specific property values.

The behavior of (super-) peers is rather unpredictable in a P2P network. Thus, these registration messages are valid for a certain period only, and peers have to re-register periodically. By invalidating the peers' registrations periodically we chose a behavior similar to other protocols for dynamic settings (like DHCP) since peers may leave the network without any notice. If a super-peer fails, its formerly connected peers must re-register with another super-peer. We are currently investigating deterministic reconnection strategies using testaments which specify alternative super-peers, and clustering strategies, grouping similar peers (in terms of supported schema) together.

3 Index Structures and Index Updates

The introduction of super-peers in combination with routing indices at these peers reduces the workload of peers significantly by distributing queries only to the appropriate subset of all possible peers (see also [16] which discusses routing indices based on various aggregation strategies of content indices).

3.1 Super-Peer / Peer Routing Indices

The first level index, the SP/P index, is an index which describes the characteristics of all peers connected to a specific super-peer, and thus guides the forwarding of queries from a super-peer to a connected peer. Furthermore, it forms the basis of the second level of indices, the SP/SP indices, which are derived from the SP/P indices, and facilitate routing within the super-peer backbone. Both indices contain information at different granularities.

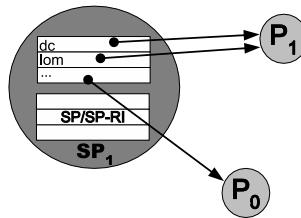


Fig. 3. Super-Peer/Peer Routing Index

Schema Index. At the schema level we assume that different peers will support different schemas. These schemas are uniquely identified by their respective namespace, therefore the SP/P routing index contains the schema identifier and the peers supporting the respective schema.

Property/Sets of Properties Index. Routing indices also contain properties or sets thereof thus enabling peers to support only parts of schemas. The properties are uniquely identified by namespace/ schema ID and property name and form the routing index entry together with those peer IDs where the properties are used.

Property Value Range Index. For properties which contain values from a predefined hierarchical vocabulary we can use an index which specifies taxonomies or part of a taxonomy for properties (the property value range). This is a common case in Edutella, because in the context of the Semantic Web quite a few applications use standard vocabularies or ontologies. In our example, peers could be characterized by their possible values in the dc:subject field, and the query would not be forwarded to peers managing “ccs:networks” or “ccs:artificial_intelligence” content (as these sub-hierarchies are disjoint from the ccs:software_engineering sub-hierarchy), and will not be forwarded to peers which use the MeSH vocabulary

(because these peers manage medical content). Note that the subsumption hierarchy in a taxonomy such as ACM CCS can be used to aggregate routing information in order to reduce index size.

Property Value Index. For some properties it may also be advantageous to create value indices to reduce network traffic. This case is identical to a classical database index with the exception that the index entries do not refer to the resource, but the peer providing it. The index contains only properties that are used very often compared to the rest of the data stored at the peers. This would be used e.g. for string valued properties such as `dc:language` or `lom:context`.

3.2 Super-Peer / Super-Peer Routing Indices

Super-peers connect to each other using the HyperCuP protocol. Even though this protocol enables an efficient message broadcast, further indices can be used to restrict message propagation even further. These super-peer/super-peer routing indices are essentially extracts and summaries from our local SP/P indices. They basically contain the same kind of metadata information as the SP/P indices, but refer to the direct neighbors of a super-peer (as shown in figure 4). Queries are then forwarded to super-peer neighbors based on the SP/SP indices, and sent to connected peers based on the SP/P indices.

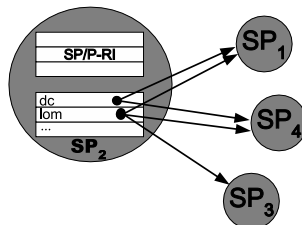


Fig. 4. Super-Peer/Super-Peer Routing Index

A sample SP/SP routing index is shown in figure 5 (taken from figure 7, super-peer SP_2). Again, we see that such an index contains metadata information on different granularities. For example, SP_2 knows at the schema level that all of its neighbors (SP_1 , SP_3 , SP_4) use the DC namespace, but only SP_1 and SP_4 contain information described in the LOM schema. The index contains also index entries based on the property value range, used in combination with a classification hierarchy: `ccs:networks` is a common super concept of `ccs:ethernet` and `ccs:clientserver` in the ACM CCS taxonomy.

Granularity	Index of SP_2		
<i>Schema</i>	dc	SP_1, SP_3, SP_4	
	lom	SP_1, SP_4	
<i>Property</i>	dc:subject	SP_1, SP_3, SP_4	
	dc:language	SP_1, SP_4	
	lom:context	SP_1, SP_4	
<i>Property</i>	dc:subject	ccs:networks	SP_3
<i>Value Range</i>	dc:subject	ccs:software-eng.	SP_1, SP_4
<i>Property</i>	lom:context	“undergrad”	SP_1, SP_4
<i>Value</i>	dc:language	“de”	SP_1, SP_4

Fig. 5. SP/SP Index of SP_2 at Different Granularities

3.3 Updating SP/P Routing Indices

An update of the SP/P index of a given super-peer occurs, when a peer leaves the super-peer, a new peer registers, or the metadata information of a registered peer changes (e.g., new attributes are added or deleted).

In case of a peer leaving the super-peer all references to this peer have to be removed from the SP/P index of the respective super-peer. The same applies if a peer fails to re-register periodically. In the case of a peer joining the network or re-registering, its respective metadata/schema information are matched against the SP/P entries of the respective super-peer. If the SP/P routing index already contains the peers' metadata only a reference to the peer is stored in the index otherwise the respective metadata with references to the peer are added to the index.

The following algorithm formalize this procedure: We define S as a set of schema elements¹: $S = \{s_i \mid i = 1 \dots n\}$. The super-peer SP_x already stores a set S_x of schema elements in its SP/P index. The SP/P index of a super-peer SP_x can be considered as a mapping $s_i \mapsto \{P_j \mid j = 1 \dots m\}$. A new peer P_y registers to the super-peer SP_x with a set S_y of schema elements.

- (1) If $S_y \subseteq S_x$, then add P_y to the list of peers at each $s_i \in S_y$
- (2) Else if $S_y \setminus S_x = \{s_n, \dots, s_m\} \neq \emptyset$, then update the SP/P index by adding new rows $s_n \mapsto P_y, \dots, s_m \mapsto P_y$.

¹ A complete schema, e.g., *dc* is also considered as schema element

3.4 Updating SP/SP Routing Indices

First, let us consider how to update the SP/SP indices in the backbone, when one of them has been modified as described in the last section. We assume here, that each SP/P modification triggers the update process for SP/SP indices, though we can also collect the modifications for a given period and only then trigger the SP/SP update process.

We further assume that the super-peers cluster peers according to their schema characteristics, so that peers connected to a super-peer usually have similar characteristics, and SP/P modifications trigger SP/SP index updates less frequently. If we take for example the network in figure 7 and the example SP/SP index of SP_2 shown in figure 5, a new peer P_x registering at super-peer SP_1 with the property *dc:language* does not trigger the update process since this metadata information already exists in the SP/P index. If a new peer P_y registers at SP_1 with the property *dc:created*, the SP/SP update process starts, as this property was not included in the index before.

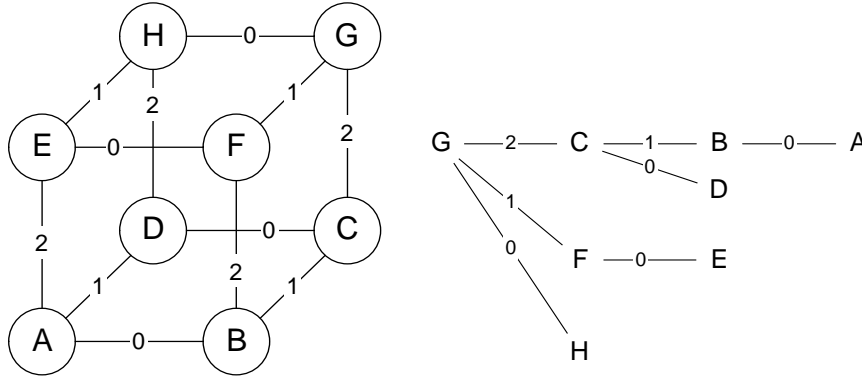


Fig. 6. HyperCuP Topology and Spanning Tree Example

SP/SP Update Process. Remember that the super-peers in the network are organized into a HyperCuP topology, which implicitly defines each super-peer as root of a spanning tree. Query routing takes place along the spanning trees (restricted by the SP/SP indices), so the update of SP/SP indices has to be done in the reverse direction. For these updates, again each super-peer acts as the root of a spanning tree (in the “backward direction”), as we show in figure 6 for the super-peer called G. In this example we have a simple (complete) cube, which has three dimensions (0,1,2), such that every node has 3 neighbors.

In order to update the SP/SP indices after an update of the SP/P index of the super-peer SP_G we build the spanning tree of the super-peer SP_G as follows: SP_G sends the update message to all its neighbors, tagging it with the edge label (dimension) on which the message was sent. Super-peers receiving the message update their SP/SP index accordingly and forward the update message, but only to those super-peers tagged with lower edge labels. Furthermore, whenever a message does not

change the SP/SP index at a receiving super-peer SP_y , forwarding stops. The spanning tree for SP_G is given in figure 6. The update is done as follows:

- For all $s_i \in S_x \cap S_y$ add dimension of SP_x to the list of dimensions at row s_i if this dimension do not exist.
- For all $s_i \in S_x \setminus S_y$ add a new row $s_i \mapsto dimension(SP_x)$

Adding new Super-Peers. Adding a new super-peer is a bit more complicated. For a new super-peer, the HyperCuP protocol takes care of identifying new neighbors as discussed in [18]. In this process one of the super-peers is “responsible” for integrating the new super-peer. In most cases the new super-peer will fill a “vacant” position in the hypercube, which has temporarily been administered by the responsible super-peer. In this process, this super-peer, who has been holding an additional SP/SP and SP/P index for the vacant position, transfers these indices to the new super-peer. If the new super-peer opens a new dimension, it has to take over some peers from the old super-peer, and the SP/SP index has to be split into two indices. The neighboring super-peers have to update their indices accordingly, by exchanging the responsible super-peer with the new super-peer on the appropriate dimension. Beyond the immediate neighbors, no further update is necessary.

Removing Super-Peers. The HyperCuP protocol also takes care of a super-peer leaving the backbone. We usually assume that the leaving super-peer coordinates this operation, and specifically asks appropriate super-peer(s) (more than one if the leaving super-peer temporarily fills several positions) that will administer its position afterwards. In this process the administering super-peers take over the SP/SP and SP/P indices of the leaving super-peer, and the neighbors of the leaving super-peer as well as of the administering ones have to update their SP/SP indices. Again, no update is required beyond the immediate neighbors. Peers of the leaving super-peer reconnect to the super-peer which administers the vacant position.

In the case of unexpected link failure its neighbors determine the “closest” (regarding smallest hop distance) super-peer. This super-peer then coordinates the administration of the open position with the same procedure as described above. Peers of the failing super-peer have to reconnect at some other super-peer, possibly triggering further SP/SP update messages.

3.5 Query Routing

The motivation for improved topologies for P2P networks is usually to make query forwarding more efficient. The goal is then to forward queries only to the appropriate peers, which can answer these queries. This can be easily achieved based on the indices described in the previous section, by matching the query elements against the super-peer/super-peer and super-peer/peer indices and forward the queries only to those super-peers and peers, which can support the elements contained in the

query. A match means that a peer understands and can answer a specific query, but does not usually guarantee a non-empty answer set (except in the case of a property value entry).

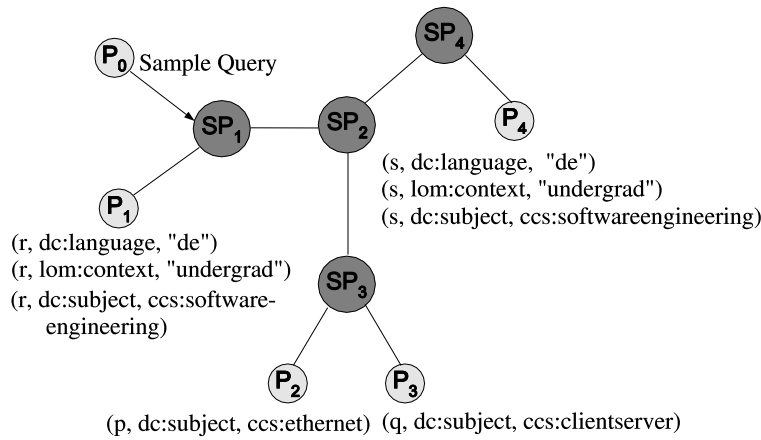


Fig. 7. Routing Example Network

The algorithm described here handles queries which can be evaluated without joining data from several peers. We are in the process of extending this algorithm to dynamically create query evaluation plans based on our index information which allow forwarding sub-queries to different peers and combining the results appropriately [19].

To illustrate this kind of routing mechanism, we use the following sample query: *find lectures in German language from the area of software engineering suitable for undergraduates*. In the Semantic Web context this query is formalized using the dc schema for document specific properties (e.g. title, creator, subject) and the lom schema which provides learning material specific properties, in combination with classification hierarchies (like the ACM Computing Classification System, ACM CCS) in the subject field. In line with RDF/XML conventions, we will identify properties by their name and their schema (expressed by a namespace): “dc:subject” therefore denotes the property “subject” of the DC schema. So, written in a more formal manner, the query becomes:

Find any resource where the property dc:subject is equal to dc:language is equal to “de”, ccs:softwareengineering and lom:context is equal to “undergrad”.

Figure 8 shows the values requested in the query at the different granularities; e.g. the query asks for DC and LOM at the schema level, while it requests a lom:context value of “undergrad” at the property value level, etc.

Figure 7 shows how peer P_0 sends the sample query mentioned above to its super-peer SP_1 . In our example, this query could be answered by the peers P_1 and P_4 , attached to SP_1 and SP_4 , respectively. These contain metadata about resources r and s which match the query. Based on a schema-level-index, super-peer SP_1 forwards

Granularity	Query	
<i>Schema</i>	dc, lom	
<i>Property</i>	dc:subject, dc:language, lom:context	
<i>Property Value Range</i>	dc:subject	ccs:sw'engineering
<i>Property Value</i>	lom:context	“undergrad”
	dc:language	“de”

Fig. 8. Query Elements of the Sample Query at Different Granularities

the sample query only to peer P_1 which supports the schemas *lom* and *dc*. Based on the property-level-index, the sample query in figure 7 will be forwarded by SP_1 to P_1 because it is the only peer at SP_1 that using at least dc:subject, dc:language and lom:context.

Obviously, indices only help if they can exploit and express regularities present in the peer and data distribution. Clustering peers based on similarity measures therefore is a necessary ingredient for improving index effectiveness and thus query efficiency.

4 Implementation

We are currently in the process of verifying the performance of our protocol and routing mechanisms. To simulate a system based on the protocol, we have implemented our algorithms within the current Edutella framework. The existing Edutella framework has been extended in two areas: The first area consists of support for construction of a network of super-peers based on such topologies as the HyperCup topology discussed in section 2. The second area introduces additional components for super-peer construction, including services for peer registration and query routing table management. The services a peer provides are specified by configuration. Services are composed of standard modules (like a JXTA Endpoint handler which manages service requests arriving via the JXTA infrastructure) and service-specific modules. Attached to each service is a service advertisement which is published in the network on peer startup. Discovery of published services is already provided by the JXTA framework.

Figure 9 shows a minimal service configuration for super-peer. The super-peer provides four services:

- *Bind Service*. The bind service handles peer registration. Provider peers call this service with their self-description to establish the connection to a super-peer. The bind service takes care of the hand-shaking process and updates the SP/P indices accordingly.

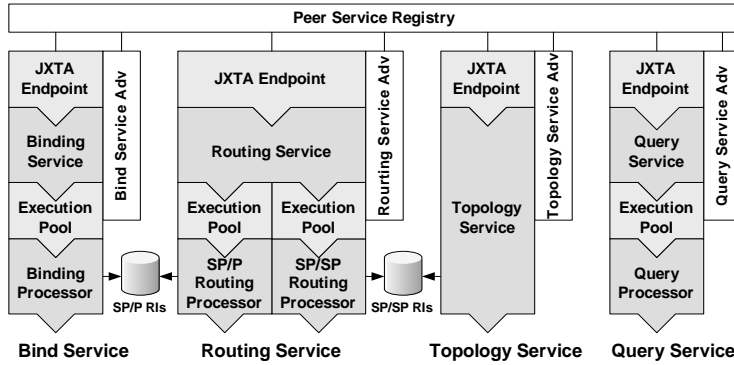


Fig. 9. Super Peer Service Configuration

- *Routing Service.* This service is doing the real work. It routes queries it receives to the appropriate peers and super-peers, based on the indices created by the binding and topology service.
- *Topology Service.* The topology service takes care of maintaining the super-peer network topology, and also keeps the SP/SP indices up-to-date. If a new super-peer starts, its topology service connects to the topology service of another super-peer, and the location of the new super-peer in the network is negotiated (as described in [18]). Afterwards the new neighbors exchange SP/SP routing information.
- *Query Service.* The query service provides a defined interface to issue new query requests within the network. These requests are then then distributed via the routing service.

Communication between service components within a peer is done by sending events to monitoring listeners, according to the Observer design pattern. On startup, the components for the configured services are instantiated and the necessary observer relations are created. For example, the query service doesn't have to know how the query is finally distributed. Instead, the routing service is tagged as *matching actor* for query events in the configuration, and thus the routing service is registered as listener at the query service. This way queries received by the query service are passed to the routing services as query events.

5 Conclusion

RDF-based P2P networks have a number of important advantages over previous, more simple P2P networks. In this paper we have discussed RDF-based P2P networks as prime examples of a new kind of P2P networks, schema-based P2P networks. Peers provide and use explicit (possibly heterogeneous) schema descriptions of their content, and are therefore an infrastructure ideally suited for P2P networks consisting of heterogeneous information providers.

We identified a super-peer topology as a suitable topology for these schema-based P2P networks and discussed different kinds of super-peer indices describing the data and schema characteristics of the peers connected to the network. Super-peer indices exploit the RDF ability to uniquely identify schemas, schema attributes and ontologies, and provide a necessary ingredient for a schema-aware peer-to-peer data management infrastructure. We discussed how these indices are updated and how they are used for routing and query distribution between super-peers and peers as well as within the super-peer backbone network. Finally, we discussed our current implementation in the context of the Edutella project.

References

- [1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, A scalable content addressable network, in: Proceedings of the 2001 Conference on applications, technologies, architectures, and protocols for computer communications, ACM Press New York, NY, USA, 2001.
- [2] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, in: Proceedings of the 2001 Conference on applications, technologies, architectures, and protocols for computer communications, ACM Press New York, NY, USA, 2001.
- [3] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, T. Risch, EDUTELLA: a P2P Networking Infrastructure based on RDF, in: Proceedings of the Eleventh International World Wide Web Conference.
- [4] P. A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, I. Zaihrayeu, Data management for peer-to-peer computing: A vision, in: Proceedings of the Fifth International Workshop on the Web and Databases, Madison, Wisconsin, 2002.
- [5] K. Aberer, M. Hauswirth, Semantic gossiping, in: Database and Information Systems Research for Semantic Web and Enterprises, Invitational Workshop, University of Georgia, Amicalola Falls and State Park, Georgia, 2002.
- [6] A. Y. Halevy, Z. G. Ives, P. Mork, I. Tatarinov, Piazza: Data management infrastructure for semantic web applications, in: Proceedings of the Twelfth International World Wide Web Conference, Budapest, Hungary, 2003.
- [7] S. Gribble, A. Y. Halevy, Z. G. Ives, M. Rodrig, D. Suciu, What can databases do for peer-to-peer, in: Proceedings of the Fourth International Workshop on the Web and Databases, Santa Barbara, CA, USA, 2001.
- [8] The Edutella Project, <http://edutella.jxta.org/> (2002).
- [9] W. Nejdl, B. Wolf, S. Staab, J. Tane, Edutella: Searching and annotating resources within an RDF-based P2P network, in: Proceedings of the Semantic Web Workshop, Eleventh International World Wide Web Conference, Honolulu, Hawaii, USA, 2002.

- [10] O. Lassila, R. Swick, W3C Resource Description Framework model and syntax specification, <http://www.w3.org/TR/REC-rdf-syntax/> (Feb. 1999).
- [11] D. Brickley, R. V. Guha, RDF vocabulary description language 1.0: RDF Schema, <http://www.w3.org/TR/rdf-schema/> (Jan. 2003).
- [12] L. Gong, Project JXTA: A technology overview, Tech. rep., SUN Microsystems, <http://www.jxta.org/project/www/docs/TechOverview.pdf> (Apr. 2001).
- [13] B. Yang, H. Garcia-Molina, Designing a super-peer network, in: IEEE International Conference on Data Engineering.
- [14] B. Yang, H. Garcia-Molina, Improving search in peer-to-peer systems, in: Proceedings of the 22nd International Conference on Distributed Computing Systems, Vienna, Austria, 2002.
- [15] N. Hemming, KaZaA, Web Site - www.kazaa.com.
- [16] A. Crespo, H. Garcia-Molina, Routing indices for peer-to-peer systems, in: Proceedings of the 22nd International Conference on Distributed Computing Systems, Vienna, Austria, 2002.
- [17] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst, A. Löser, Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks, in: Proceedings of the Twelfth International World Wide Web Conference, Budapest, Hungary, 2003.
- [18] M. Schlosser, M. Sintek, S. Decker, W. Nejdl, HyperCuP—Hypercubes, Ontologies and Efficient Search on P2P Networks, in: International Workshop on Agents and Peer-to-Peer Computing, Bologna, Italy, 2002.
- [19] I. Brunkhorst, H. Dhraief, A. Kemper, W. Nejdl, C. Wiesner, Distributed queries and query optimization in schema-based peer-to-peer systems, in: International Workshop on Databases, Information Systems, and P2P Computing, colocated with 29th International Conference on Very Large Databases, Berlin, Germany, 2003.
- [20] G. Wiederhold, Mediators in the architecture of future information systems, IEEE Computer 25 (3) (1992) 38 – 49.