Project Number 001907

DELIS
Dynamically Evolving, Large-scale Information Systems

Integrated Project
Member of the FET Proactive Initiative Complex Systems

Deliverable D6.5.2

Report on the Architectural Design of the P2P Prototype Testbed
Start date of the project: January 2004
Duration: 48 months
Project Coordinator: Prof. Dr. math. Friedhelm Meyer auf der Heide
Heinz Nixdorf Institute, University of Paderborn, Germany
Due date of deliverable: June 2005
Actual submission date: August 2005
Dissemination level: PU – public

Work Package 6.5: P2P System Architecture and Testbed
Participants: Max Planck Institute for Informatics (MPII), Saarbrücken, Germany
Computer Technology Institute (CTI), Patras, Greece
Authors of deliverable: Gerhard Weikum (weikum@mpi-sb.mpg.de)
Peter Triantafillou (peter@ceid.upatras.gr)
1 Overview

1.1 Design Rationale

One of the major goals of DELIS SP6 is to develop an architecture for Web information search in an Internet-scale peer-to-peer (P2P) system. In the DELIS approach every peer, e.g., the home PC of a scientist or student, has a full-fledged search engine that indexes a small portion of the Web, according to the interest profile of the user. Peers are autonomous, each peer may crawl some Web fragment at its own discretion. The per-peer search engines collaborate for answering queries posed by any of the peers. Such an architecture has four major advantages over a centralized server farm like Google:

1. As the data volume and the query load per peer are much lighter, the peer’s search engine can employ much more advanced techniques for concept-based rather than keyword-based search, leveraging background knowledge in the form of thesauri and ontologies [18, 36, 39] and powerful mathematical and linguistic techniques such as spectral analysis and named entity recognition [13, 15, 20, 23].

2. Peers can collaborate for finding better answers to difficult queries: if one peer does not have a good result locally it can contact a small number of judiciously chosen peers who are considered “knowledgeable” on the query topic [4, 24, 29]. This approach should often be able to exploit the small-world phenomenon on the Web: knowledgeable peers are only a short distance away.

3. A P2P system can gather and analyze bookmarks, query histories, user click streams, and other data about user and community behavior; the implicit and explicit assessments and recommendations derived from this input can be leveraged for better search results [16, 25]. In contrast to a central server, the P2P approach provides each user with direct, fine-grained control over which aspects of her behavior may be collected and forwarded to other peers.

4. A politically important issue is that a P2P search engine is less susceptible to manipulation, censorship, and the bias induced by purchased advertisements.

DELIS SP6 is implementing a prototype for this envisioned P2P Web search functionality. As a consequence of the above issues, its architectural design aims at the following properties:

- **High autonomy:** the regard for the user controlling her data, her computer and network resources, and what is being monitored. For example, users can crawl and collect data autonomously; this may enrich the local content of a peer. Communities of peers should be able to add metadata they view of value to a P2P engine, unlike privately-owned engines. Finally, a P2P search engine should enjoy advantages similar to those of “open software”, expecting to harvest the intellectual prowess of large numbers of individuals all over the world adding functionality and improvements.

- **Large scale at low cost:** the potential to build huge-scale search engines of great aggregate computing power with great amounts of aggregate relevant data, without great investment costs burdening a single organization but instead at negligible costs being distributed to a huge number of users of the engine.

The main purpose of the P2P search prototype within DELIS is to serve as an experimental testbed for studying the interplay of different models and algorithms and for evaluating new methods in a realistic setting. As the models and algorithms are the result of the research carried out by different partners in other WPs of SP6 and other SPs within the DELIS project, the following general design goals are crucial:
• The testbed should be highly modular: making it easy to replace components and to plug in new methods for specific functionalities. A long-term purpose of experimentation with the testbed is to identify the best methods that provide high search result quality, good efficiency, and robustness in the presence of P2P system dynamics.

• The testbed should be very light-weight: easy to install and operate in the system environments of the project partners, as experiments may require running a small P2P system that spans multiple partner sites.

1.2 System Architecture

In the P2P system, each peer has a full-fledged Web search engine, including a crawler and an index manager. The crawler may be thematically focused or crawl results may be postprocessed so that the local index contents reflect the corresponding user’s interest profile. With such a highly specialized and personalized “power search engine” most queries should be executed locally, but once in a while the user may not be satisfied with the local results and would then want to contact other peers. A “good” peer to which the user’s query should be forwarded (aka. query routing) would have thematically relevant index contents, which could be measured by statistical notions of similarity between peers. Both query routing and the formation of “statistically semantic” overlay networks could greatly benefit from collective human inputs in addition to standard statistics about terms, links, etc.: knowing the bookmarks and query logs of thousands of users would be a great resource to build on. Note that this notion of Web search includes ranked retrieval and thus is fundamentally much more difficult than Gnutella-style file sharing or simple key lookups via overlay networks such as distributed hash tables. Further note that, although query routing in P2P Web search resembles earlier work on metasearch engines and distributed information retrieval [28], it is much more challenging because of the large scale and the high dynamics of the envisioned P2P system with thousands or millions of computers and users.

This overall system architecture is depicted in Figure 1. The prototype implementation, serving as an experimental platform within the DELIS project, has been coined Minerva [4, 5, 6, 7].

In Minerva, peers are connected by an overlay network based on a distributed hash table (DHT). Peers analyze their local information content and prepare compact statistical synopses that capture the relevance for specific query terms (i.e., keywords, stemmed words, or concepts onto which words are mapped), the richness, authority, and freshness of the content, the behavioral characteristics of the peer including the corresponding user’s thematic interests, the peer’s quality-of-service properties, etc. These synopses are posted into the overlay network: disseminated to specifically chosen (e.g., by the DHT hash function) peers, often in a redundant manner with judiciously chosen replicas on different peers, such that the overall network forms a conceptually integrated but physically massively distributed directory for metadata and statistical summaries.

Each peer posts the terms that are statistically most characteristic for its local content and the URLs of its bookmarks that reflect the user’s interests. The peer or the peers that are responsible for maintaining the directory entry for a given term or URL maintain ranked lists of peers that have good information about the term or have bookmarked the URL, respectively. The decentralized directory can be efficiently queried by all peers.

A user query is normally executed on the local index first, thus avoiding network costs unless involving other peers is justified by unsatisfactory local results. In the latter case, a query routing decision is made about which other peers should be contacted in order to evaluate the query. Clearly this is the technically interesting case, which would typically arise with advanced information demands of “power users” like scientists, students, or journalists. For dynamically selecting target peers, the originating peer can consult the directory and base its decision on the statistical summaries

---

1 Minerva is the Roman goddess of science, wisdom, and learning, and is also the icon of the Max Planck Society.
Figure 1: System Architecture for P2P Web Search

that have previously been posted. The originator may additionally use further information about the candidate peers’ content, network bandwidth and latency, trustworthiness, and behavior that it has locally cached from previous interactions and observations. Once the target peers for executing the query are determined, the query is processed using a top-k algorithm, either by a) running the complete query on each selected peer and merging the search results, or by b) decomposing the query into individual subqueries, like one subquery per term, and using a network-conscious distributed top-k algorithm [27], where the extra difficulty is to reconcile the network costs (i.e., bandwidth consumption and latency) and the processing costs of the involved peers (i.e., CPU time, disk accesses, memory consumption).

The architecture of the Minerva experimental platform is depicted in Figure 2. Note that all the major components shown in the figure can be replaced by alternative implementations, for experimenting with new models and algorithms.

The local search engine comprises:

- functions for data import, e.g., Web crawling, along with the necessary analysis of the data, e.g., parsing and extraction of terms,
- functions for managing concept spaces such as thesauri and ontologies or latent-concept spaces based on spectral analysis techniques, along with support for mapping words onto concepts with word sense disambiguation (WSD),
- the index manager that maintains inverted index lists for efficient lookups, including precomputed scores that capture the weight of individual terms in a document (based on probabilistic IR models such as Okapi BM25, advanced matrix decomposition techniques such as probabilistic LSI, or authority scores derived from analyzing the links in the Web graph),
- the local query processor that provides efficient top-k ranked retrieval, supports query expansion (based on thesauri, ontologies, and other concept spaces, possibly combined with relevance feedback and other means), and techniques for customizing the search engine to the interest profile and bias of the individual user.
The *collaborative search engine* communicates with other peers; it comprises:

- one or more *overlay networks*, which can be distributed hash tables, random graphs with particular properties (e.g., expander graphs with small diameter), or semantic overlay networks where the network topology reflects thematic similarities among peers,

- the *query router* that decides to which other peers a query by the local user should be forwarded for collaborative query processing, based on a variety of statistics-driven strategies, and the result merger that consolidates the query results obtained from different peers into a global ranking.

![Collaborative Search Engine](image)

**Figure 2: Architecture of the Minerva Experimental Platform**

## 2 Local Search Engine

Each peer runs a local search engine whose indexing, scoring, and query processing techniques can be customized to the peer’s interest profile and behavior. The engine is driven by precomputed inverted index lists and employs a state-of-the-art algorithm for efficient top-k query processing [17], including approximation methods with probabilistic guarantees developed within DELIS [38]. Scoring and ranking are exchangeable modules; currently tf\*idf measures based on word frequency statistics or the Okapi BM25 scoring model is applied. It is planned to experiment also with some of the following alternative information retrieval models which are developed or extended within DELIS.

- Thesaurus- or ontology-based query expansion techniques have been integrated into the top-k search algorithms [39]. By carefully weighting expansion terms based on correlation statistics and by incrementally merging index lists for expansion terms at query run-time, this new algorithm reconciles good retrieval quality (in terms of precision, recall, and MAP values) with efficient query processing. In contrast to most of the prior work, it avoids the need for tuning expansion thresholds and thus fits well with a self-organizing environment.
• Retrieval methods that make use of the latent concepts underlying large collections of text have been developed [1, 2, 3]. The concepts are derived from the raw data by spectral analysis or other matrix decomposition techniques, without training or dictionaries. Novel insights have been found into what makes such approaches work in practice. Various new methods are derived, which improve search results, have a better theoretical foundation, and are well-suited for a P2P setting. Complementary work on concept spaces with explicitly identified, ontology-based concepts and mappings from words to concepts has been started, too [21].

• A new method, coined QRank, for incorporating query-log and click-stream information into Web page ranking has been developed. It is based on a Markov model that extends the traditional PageRank approach by query nodes and edges for query refinement, query-result clicking, and node-pair similarity [25].

• Advanced link analysis techniques, in the spirit of Google’s PageRank [10] but with additional salient properties, are being developed [8, 9, 11, 32]. These methods provide decentralized computations of authority rankings and other ways of improving the importance assessment of Web pages.

Further details on the DELIS research results on retrieval models and search algorithms have been reported in the DELIS Deliverable D6.1.1.

3 Overlay Network

Peers are connected by an overlay network, for example, a distributed hash table (DHT); in Minerva a Chord-based DHT is used [37]. Such overlay networks and the resulting neighborhood structures ensure:

1. short routing paths (by a very small graph diameter and appropriate routing tables), usually $O(\log n)$ where $n$ is the number of peers, and
2. low memory overhead per peer, usually by bounding the size of the routing tables,
3. unlimited scalability, so that the aggregated performance capacity increases linearly with the number of peers in the network,
4. self-organization in the presence of churn, i.e., the high dynamics of peers that join and leave the network without notice and at possibly very high rates, and
5. self-healing capabilities in the presence of peer failures.

Based on a DHT backbone, DELIS has been developing optimizations and new methods for overlay networks [14, 26, 30, 31, 34, 40, 41]. These have been presented in more detail in the DELIS Deliverables D6.1.2, D.6.1.3, and D.6.1.5. The Minerva testbed will allow us to experiment with these methods, aiming to identify the most robust and efficient solutions.

4 Query Routing

Query routing is a core piece of a P2P search engine. It aims to select the most promising peers among a large set of candidates, for executing a given query posed by one of the peers. This problem is also known as database selection or resource selection in the information retrieval (IR) literature. However, collaborative P2P search is substantially more challenging than the traditional setup for distributed IR over a small federation of sources such as digital libraries, as these prior approaches
mediate only a small and rather static set of underlying nodes, as opposed to the large scale and high
dynamics of a P2P system. Prior work on query routing has been discussed in the DELIS Deliverable
D6.1.1.

The rationale for the query routing strategy developed in DELIS is based on the following three
observations [4, 6]:

1. The query initiator should prefer peers that have similar interest profiles and are thus likely to
hold thematically relevant information in their indexes.

2. On the other hand, the query should be forwarded to peers that offer complementary results.
   If the remote peer returns more or less the same high-quality results that the query initiator
   already obtained from its own local index, then the whole approach of collaborative P2P search
   would be pointless.

3. Finally, all parties have to be cautious that the execution cost of communicating with other peers
   and involving them in query processing is tightly controlled and incurs acceptable overhead.

We address the first two points by defining the benefit that a remote peer offers for the given
query to be proportional to the thematic similarity of that peer and the query initiator and inversely
proportional to the overlap between the two peers in terms of their local index contents. To limit
the overhead of estimating these measures, we use the Kullback-Leibler divergence between the
bookmark documents of the two peers as a basis for estimating benefit. Here we view the index
contents of a peer as being generated by the peer’s bookmarks, which served as seeds for the peer’s
Web crawls and possibly also as training data for a thematically focused crawler [12, 35]. For overlap
we consider a large sample of a peer’s locally indexed documents, and we represent the set of these
documents using a compact statistical synopsis for information exchange among peers [6], e.g., a
Bloom filter, hash sketch, or min-wise permutation structure. Then benefit is defined as the quotient
of similarity and overlap. An additional aspect of benefit that we are currently exploring is to
incorporate also measures of reputation (authority) and trust, both of which could be implemented
using PageRank-style Eigenvector computations [10, 22, 23]. The difficulty is to compute such a
measures in a fully decentralized way, without having to build a global graph structure, capturing
links, recommendations, or opinions, on a central site. Promising techniques along these lines have
been developed in [32].

We reconcile the notion of benefit with the third of the above observations by considering the
benefit/cost ratio of peers, where cost is estimated based on tracking the utilization and resulting
response time of different peers.

The outlined strategy as well as various alternative strategies from the prior literature have been
implemented in the Minerva testbed. Experimental studies on benchmark datasets show that the
estimation of overlap makes query routing more cost-efficient, by reducing the number of peers that
need to be contacted while at the same time ensuring very high recall and precision.

5 Semantic Overlay Networks

The query routing decisions are made at the run-time of queries, so the overhead for statistical
similarity and overlap comparisons may adversely affect the user-perceived query response times. To
ensure that routing decisions have acceptably low overhead, various kinds of precomputations can
be employed and leveraged. In essence they all lead to some form of “semantic overlay network” in
addition or on top of the DHT-based overlay network that connects all peers. In the semantic overlay
network, SON for short, only peers are directly connected as neighbors that have high likelihood of
being beneficial to each other’s queries. In principle, one could think of the SON as the result
of running a content-similarity-based clustering algorithm on the peers, but this approach would
be static and would not self-adjust the resulting neighborhood structure to the dynamics of a P2P system. In DELIS we rather pursue four other approaches, which are better suited for a decentralized large-scale setting with high dynamics:

1. benefit-based ranking of postings in the DHT-based semantic directory,
2. opportunistic semantic caching at every peer,
3. proactive dissemination based on epidemic spreading methods, and
4. dynamically self-organizing SONs based on random graphs with benefit-driven bias.

The semantic directory approach is fully implemented in the Minerva testbed. In addition to posting terms, term-frequency and other statistical measures, and bookmarked or otherwise important URLs to the directory, the peer that is responsible for some of a term or URL simply computes a global ranking of the best peers for the term or URL. At query-routing time, when the directory peer is inquired by the query originator, only a small number of the best candidate peers is returned, thus ensuring low overhead. The directory should be continuously updated, at an acceptable background-activity rate, to avoid degradation by stale statistics.

In the opportunistic semantic caching approach, every peer maintains a history of its interactions with other peers. This includes statistical similarity, overlap, and execution-cost information required for query routing, actual query results for peers that were involved in query execution, additional content statistics that such peers may have piggybacked on their replies, and also information about queries posed by other peers when the given peer conversely processes remotely issued queries. All this information would simply be collected in a local cache, with appropriate invalidation, update, and replacement strategies when cached entries become stale or the cache becomes full. This enables a peer to build up its own, possibly personalized or otherwise biased, statistical view of interesting other peers. On this basis, a peer could effectively determine a short list of most interesting peers and consider them as virtual neighbors in an additional overlay structure. Note that piggybacking, adding a small amount of extra data to the payload of a network message that needs to be sent anyway, is a well established technique in distributed computing and particularly intriguing in the P2P setting. It may even include peers that are involved in lower-level network routing, e.g., the small number of peers that lie on the IP-layer path from a sending to a receiving peer in the DHT overlay. This way, peers may learn about frequent queries by other peers even if they never become involved in the actual query execution. Obviously, these techniques should be controlled by the degree to which peers are willing to share such information.

In addition to merely caching useful information when the opportunity arises, we can employ a proactive dissemination scheme. Typically, this will be based on the epidemic spreading paradigm. A peer chooses a small number of peers for “random rendezvous” and sends them a compact synopsis of statistical information about itself, and possibly also about its subjective view of the global-network statistics. Such a dissemination step is transitively repeated. The rate of these background messages can be controlled so as to limit its network bandwidth consumption.

Finally, the last approach that we investigate, and currently favor, for the Minerva system is to dynamically maintain a SON based on a random but biased graph [33] (cf. also [26] on random-graph-based overlay networks with particular salient properties). The maintenance of this graph can actually exploit any of the above approaches for propagating statistical summaries in the network. We assume that a peer maintains a list of other interesting peers, its current “friends” (in addition to the DHT-provided connectivity). For simplicity, we assume that this is a fixed-size list (alternatively, a variable but bounded number of friends may be maintained). The friends should always be those peers, known to the given peer, that are predicted to provide the highest benefit to the given peer’s (future) queries. As mentioned before, benefit is measured as a quotient of estimated thematic
similarity and estimated overlap. When the given peer interacts with other peers and learns about their potential benefit, it can consider replacing one of its current friends by the newly met peers. In addition, the peer may consider “blind dates” with randomly chosen peers, to obtain a broader perspective and avoid getting stuck in a local or thematically narrow neighborhood. When a friend is replaced, this amounts to re-wiring the graph on which the SON is based. The whole approach is fully self-organizing as it does not require any centralized coordination or explicit control by humans, and every peer can use its own strategy and bias for meeting other peers and considering them as new friends. Further work on how this approach relates to “social tagging” methods in the evolution of social networks [19] is underway.

6 Overview of Software Deliverable

![Diagram of Minerva Software Package]

The Minerva testbed is open-source software fully implemented in Java, using the J2SE 5.0 platform, for portability and easy installation on a wide variety of operating systems. It uses various components developed as open-source software at the Max-Planck Institute for Informatics:

- the BINGO! focused crawler for collecting, analyzing, and indexing Web pages,
- a restricted version of the TopX engine as a local search engine,
- a Java-based reimplementation of the Chord distributed hash table.

All these components are simply Java packages that are included in the Minerva jar file. They do not require any extra installation.

In addition, Minerva relies on a persistent data manager for its index lists, document contents, and metadata. To this end, Minerva can either use the open-source database library Cloudscape 10.0 (see http://www-306.ibm.com/software/data/cloudscape/) as part of the Minerva jar file, or it can communicate with an Oracle 10g or a MySQL 5.0 database server. In all three cases, JDBC is the protocol for SQL calls between Minerva and the data manager. Cloudscape does not require an explicit installation and is thus more easy to use in the P2P environment, whereas Oracle and MySQL are servers and require explicit installation and certain expertise in database administration.

It is straightforward to incorporate also further measures like estimated execution cost, reputation, trust, etc.
Yet another alternative that Minerva can use is to use files for persistent storage and load all index lists into memory; this approach is restricted in terms of the size of the managed corpus.

Figure 3 shows the structure of the entire Minerva software package. Minerva comes with a GUI for starting and controlling crawls, for creating and joining P2P networks, and for running keyword queries. The GUI is shown in Figure 4.

The local search engine comprises the crawler, the persistent data manager, and the query processing engine. The scoring for Web documents currently uses either a $tf^*idf$ score function based on word frequency statistics or the Okapi BM25 method, one of the best known scoring methods derived from a probabilistic information retrieval model. Scores are precomputed and stored in inverted index lists. It is very easy to add new scoring methods and store alternative scores in the index lists; this merely requires understanding the database schema of the index lists and some modest skill level in using SQL. For example, authority scores based on link analysis are straightforward to incorporate; such additional features will be released soon.

The collaborative search engine runs on each peer and communicates with other peers. It comprises the software for the overlay network, by default a Chord-based distributed hash table, the directory with summaries and statistics about the peers’ data contents, and the query routing strategies. The directory is a decentralized data structure whose entries are distributed across the nodes of the DHT, thus providing scalability. The query routing component supports a variety of strategies, driven by term frequency statistics or statistical language models and by estimating the overlap between peers. The number of peers to which a query is forwarded can be controlled in the GUI;
the query results from different peers are merged into a global result list with various choices for the score reconciliation technique.

The first Minerva release is packaged for the DELIS Software Deliverable D6.6.2. It will be made available to all DELIS partners for experimentation and extensions. The first release has limited functionality, extensions and improvements will be made in future releases within the DELIS project.

References


[16] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, Wei-Ying Ma: Query Expansion by Mining User Logs. IEEE Transactions on Knowledge and Data Engineering 15(4), 2003


[34] Christian Schindelhauer, Gunnar Schomaker: Weighted Consistent Hashing, ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), Las Vegas, USA, 2005


[38] Martin Theobald, Gerhard Weikum, Ralf Schenkel: Top-k Query Evaluation with Probabilistic Guarantees, International Conference on Very Large Data Bases (VLDB), Toronto, Canada, 2004

