OmniPaper

Smart Access to European Newspapers

Blueprint: a universal standard model for efficient information retrieval

deliverable D4.1

28/02/2005

Content

PART A Introduction and Problem Area 6

A.1 Problem area description 8

A.1.1 Distributed information 8

A.1.2 Diversity in information storage and access 8

A.1.3 Lack of good quality metadata 8

A.2 Approaches for distributed information retrieval and publishing 9

A.2.1 Web robots (pull method) 9

A.2.2 Centralised databases (push method) 9

A.2.3 OmniPaper approach 10

A.3 Existing solutions at different levels 11

A.3.1 Distributed information retrieval 11

A.3.2 Online news publishing 11

A.3.3 Search engines 13

A.3.4 Ontology-driven systems 14

A.3.5 Geo-reference systems (gazetteer) 15

A.3.6 User interface 16

PART B Research Work, Results and Guidelines 20

B.1 OmniPaper Architecture: a Framework for Distributed Information Retrieval 21

B.1.1 OmniPaper – Architectural Requirements 21

B.1.2 Architectural Design 22

B.1.3 Architecture Layering in OmniPaper 28

B.2 Prototype development: access gate to distributed information sources 30

B.2.1 Metadata definition 30

B.2.2 Distributed Information Retrieval Prototype (WP2) 32

B.2.3 Overall Knowledge Layer Prototype (WP3) 58

B.2.4 Final prototype for smart access to European newspapers (WP5) 73
B.2.5 Prototype cross-testing 74
B.2.6 User evaluation 96
B.2.7 Summary overview of search methods 96
B.2.8 Summary overview of developed prototypes 97
B.2.9 Open technical issues and possibilities for future work 100
B.3 User Interface Guidelines 107
B.3.1 Characteristics of Multilingual Search 107
B.3.2 User Interface Modules 112
B.3.3 References 155
PART C Applicability in other domains 156
C.1 OmniPaper at the Cross-Language Evaluation Forum 157
C.1.1 Keyword Recognition for document characterization 158
C.1.2 AKE Description 160
C.1.3 AKE evaluation 160
C.1.4 Natural language Processing 166
C.1.5 References 171
C.2 Architectural Reuse for distributed information retrieval 173
C.2.1 ERDDS 173
C.2.2 Project MIND 175

Document information

Document ID	OmniPaper_4_Blueprint
Title	Blueprint: a universal standard model for efficient information retrieval
Short title	Blueprint
Workpackage	4

Deliverable	D4.1
Subject	Blueprint, handbook, guidelines, information retrieval, distributed information
Description	This document contains an overview of the entire OmniPaper problem domain, including solutions studied by the OmniPaper project and a comparison with other existing solutions in the same area.
Security level	Confidential (also a Public version is available)

Version history

Version	Date	Author(s)	Description
0.1	05/11/2003	Bert Paepen	First Draft
0.2	08/04/2003	Markus Schranz	Interim Version, Architecture Description
0.3	23/09/2004	Markus Schranz	Final Version, Architecture Details, Component Description,
0.4	18/11/2004	Markus Schranz, Bert Paepen	Further Research Approaches, Commercial Products
1.0	23/2/2005	Bert Paepen, Ana Alice Baptista	Restructured, removed excessive details, removed exploitation part, included RDF prototype description and testing, added conclusions and guidelines.
2.0	28/02/2005	Bert Paepen	Revised for public version

Abbreviations

Term	Description
XTM	XML Topic Maps
MDDB	Metadata database
EWN	EuroWordNet
TMDB	Topic Map database
AKE	Automatic Keyword Extraction
RDF	Resource Description Framework
SOAP	Simple Object Access Protocol

Definitions

Term	Description
topic map	A graph structure representing topics and associations between them. In the OmniPaper system there is one such structure, denoted the Topic Map.
topic	An item in a topic map. A topic in the Topic Map represents either a concept, a subject, or a keyword.
keyword	A word or phrase used as a descriptor for any item in an information retrieval system. Keywords are used to describe articles and to describe concepts. Keywords that describe concepts are stored in the Topic Map as topics.
relevant keyword	A keyword in an article which is deemed as "relevant" under some criteria. These keywords are the only ones used as descriptors for articles, and are stored as meta-data of the article.
synset/concept	A set of keywords that represent a concept. Synsets are stored in the Topic Map as topics. Each topic representing a synset is associated via containment relations with the topics that represent the keywords that describe the concept of the synset. Synset is typical WordNet terminology, it will not be used in this document.
subject	A special type of concept that is used in a hierarchy of concepts. A subject is a synset that corresponds to a category in a given hierarchy. Subjects are stored as topics in the Topic Map, associated via equivalence relations with the topic that represents the corresponding synset. Topics representing subjects are associated via hierarchical relations that represent the hierarchy being used for categories.

PART A INTRODUCTION AND PROBLEM AREA

Throughout an entire decade, the Internet has brought unmanageable amounts of information to the average user's fingertips. Since this growth will only continue, it is vital that users are supported in converting this universe of information into improved productivity and opportunity instead of being swamped and paralyzed. Failing to address information overload will cost enterprises and individuals money, often in ways that are not easily measured: Costs that result from lowered productivity and from mislead business decisions. To really satisfy user needs and restricted budgets, the myriads of information need to be structured and organized in an intelligent and user-oriented way. Technically, appropriate architectures to integrate existing archives with an intelligent news retrieval engine are to be developed. The research approach in the discussed OmniPaper project has investigated ways for drastically enhancing access to many different types of distributed information resources.

The key objective is the creation of a **multilingual navigation and linking layer on top of distributed information resources**, thus providing a sophisticated approach to manage multinational news archives with strong semantic coupling, delivering to the user more than the sum of the individual service features.

As a proof of concept, the project consortium has constructed a prototype system that enables users to have simultaneous and structured access to news articles originating from a large number of digital European newspapers.

The major **technological** objective of the OmniPaper project lies in creating an intelligent uniform entrance gate to a large number of European digital newspapers, allowing readers a more objective view on subjects. This rather general technological objective will be split up in three parts which are easier to verify and measure:

1. Find and test mechanisms for retrieving information from distributed sources in an efficient way. This means that multi-archive retrieval should not be significantly slower than single-archive retrieval.

2. Find and test ways for creating a uniform access point to several distributed information sources.

3. Make this access point as usable and user-friendly as possible.

From the **scientific** point of view, the project approach can lift widely distributed digital collections to a higher level, by:

- Applying a common multilingual thesaurus superstructure to them;

- Linking them to each other;

- Enriching their quality and the navigational features through learning from user behaviour.

The OmniPaper project created this reference guide (Blueprint) and a prototype system for improving access to distributed information resources. As a prototype environment, it will build a uniform, multilingual access system to articles from various European newspapers. This system will enable users to search a newspaper article in one language, returning multilingual results originating from different important newspapers.

State-of-the art technologies (such as SOAP, RDF and Topic Maps) have been examined, compared and prototyped in order to find the best ways for creating flexible navigation, filtering of information, cross-lingual and cross-archive information retrieval. Artificial intelligence concepts have been incorporated to automate the creation and maintenance of this powerful knowledge layer in a self-learning way. Automatic keyword extraction provides a uniform relevance ranking mechanism across the different searched archives.

This project also serves an **economic** objective. Finding accurate but widely dispersed information is highly important for newspapers, whose success strongly depends on their speed of providing news.

By building a multilingual interface to distributed archives, the project's approach allows to take into account the local aspects of cultural and scientific information provision. Queries are automatically translated in the different languages that exist in the various archives. That way, readers can look up news information without having to know anything about the language of each of the archives. Newspaper articles themselves are available in the original newspaper's language, but can be automatically translated at the user's request[1].

OmniPaper is not a project about digitisation of news, but about bringing digitised news originating from various sources together through a single access gate. Therefore, the project assumes that the source material is already available in a digital form, containing sophisticated meta-data and navigational information. The added value brought by the OmniPaper system resides in the intelligent, multilingual and navigable knowledge superstructure built on top of this already enriched material.

This Blueprint documents the research work and the project results the consortium has performed and achieved during the three years of project duration. The purpose of the document is to provide guidelines for efficient information retrieval in a distributed and heterogeneous environment. Using this document, future research and practical efforts by subsequent organisations and projects shall be able to start from a well researched and clearly described position that reflects the consortiums experiences, major findings and developments and thoroughly researched component evaluations towards a standard architecture and process management for similar projects.

Throughout this document a number of research conclusions and derived guidelines are included. For ease of reading the following layout is used throughout the document:

- Results and conclusions are indicated in green boxes. On black and white prints these boxes are the darker ones.

- Guidelines are indicated in yellow boxes. On black and white prints these boxes are the lighter ones.

[1] Note that automatic translation of entire newspaper articles does not lay within the scope of the project. External modules obtained from experts in the field have been used for the automatic translation of articles at the user's request.

A.1 Problem area description

This section describes the problem areas in which OmniPaper has performed research and has developed prototypes.

A.1.1 Distributed information

As an inherent part of its conception is the networking of distributed computers, a logical consequence is that a lot of information on the Internet is physically distributed. By itself this does not have to be a problem, because it is the Internet itself that makes information available everywhere even it is stored on computers far away. The problem is that becomes hard to find out what information is out there and where to get it (on what addresses). Search engines solve a great deal of this problem and they are as such becoming more and more popular, not only for finding information for which users don't know where to look, but also for getting to information that users have already consulted before – so for *navigation* in stead of pure *search*.

Even with the help of search engines the main problem still exists: information is spread across the Internet and this makes it very hard to get a complete view on all available information about one topic or to make connections between parts of distributed information.

A.1.2 Diversity in information storage and access

Information is not only scattered physically, but also in terms of storage formats, environments, hardware, database formats, information structures, etc. This problem adds up to the problem described before: even if the data *source* is known, this doesn't mean that real *information* can be extracted from it easily. Heterogeneous environments and information formats make efficient information capturing more difficult.

A.1.3 Lack of good quality metadata

A lot of effort in the Semantic Web area has been spent in defining metadata standards: standard ways to describe metadata. The idea is that information will get much better accessible if it is well-described. For general metadata Dublin Core is one of the most important standards; for news articles NewsML is widely adopted.

However metadata initiatives in the Semantic Web are becoming more and more important, there still is a lack of good quality metadata. So now the phenomenon is taking place that everybody seams convinced of the advantages of metadata, but nobody really bothers for creating and maintaining metadata. Authors often see the creation of metadata as a burden. Even in the area of news publishing this still is a problem. Journalists and editors are under time pressure all the time for publishing news as soon as possible. As a result creating good quality metadata for their news articles often is the last thing on their mind.

These observations mean that, if an important tool for improving information retrieval is metadata, this metadata has to be gathered using other means than mere manual creation.

A.2 Approaches for distributed information retrieval and publishing

Several approaches exist for retrieving and publishing distributed information.

A.2.1 Web robots (pull method)

This method consists of the regular “crawling” of the web by so-called “web robots”: systems that scan an entire data collection and that make giant indexes of all found information. Most popular search engines use this method because it allows users to search very fast in huge amounts of information – if the indexes are efficiently constructed. A key feature of this method is that it “pulls” information from all directions into a centralised index. The original information sources do not initiate the search, they ‘undergo’ it.

How popular its applications are, this is basically a “brute force” mechanism. It needs huge databases for indexing entire collections (or even the “entire” web). Further it hardly uses metadata for making searching more efficient. “Grabbing” or “scraping” information from web pages also is an error-prone method. Finally another problem becomes more and more important: the inability of this kind of search engines to grab all information. A lot of information is stored in databases whose contents are only shown on the web using specific queries and scripts. This so-called *deep web* is a known issue that demonstrates the limitations of current search engines.

A.2.2 Centralised databases (push method)

This approach is completely opposite from the previous one as it requires complete co-operation of the data sources: they regularly send updates to the central database or index on their own initiative (“push”). An advantage is that the quality of information in the central index can be much better because it comes directly from the original source.

Disadvantages of this method are the difficult maintainability of such a central data store, possibility of outdated information and possible problems in information exchange if different data structures are being used.

A.2.3 OmniPaper approach

The OmniPaper approach combines several existing solutions in order to propose solutions from some of the described problems. Since its theoretical model - as it is described in section B.1 - slightly differs from the prototype developed in practice, both approaches are briefly explained here.

Table 1: OmniPaper approach for architecture vs. prototype

Theoretical model (architecture)	Developed prototype
The archives co-operate to the centralised search system.	The archives co-operate to the centralised search system.
All information exchange between the central system and the distributed archives happens through SOAP. This means that it is well-structured and uses XML syntax.	All information exchange between the central system and the distributed archives happens through SOAP. This means that it is well-structured and uses XML syntax.
Manually created metadata is enriched with automatically extracted keywords.	Manually created metadata is enriched with automatically extracted keywords.
Central database with only metadata (including extracted keywords). News article contents are never stored centrally but reside at their original location at the information provider.	No central database with metadata because development and maintenance of it are very expensive – they require huge efforts with a lot of manual intervention. News article contents are never stored centrally but reside at their original location at the information provider.
Push combined with pull: normally the distributed archives notify the central system if information updates are available, but the central system can also proactively ask the local archives to give a status update. Either ways should make sure that the central system has the metadata of all articles, including the most recent ones.	Pure pull approach: all search requests are forwarded to the local archives and are solved by them. Central system combines the answers, adds language and search refinement functionalities, and performs automatic keyword extraction for unified relevance ranking.
Query and navigation are both methods for *searching* and should be combined as much as possible. Navigation can provide support to querying and vice versa.	Query and navigation are both methods for *searching* and should be combined as much as possible. Navigation can provide support to querying and vice versa.

A.3 Existing solutions at different levels

This section gives an overview of existing solutions in the different parts of the OmniPaper area of work. Its intention is *not* to provide an exhaustive list of related systems and technologies, but to give an overview of different areas that are relevant for OmniPaper. For each area, a number of existing solutions are given. For each existing solution area its relevance for OmniPaper is assessed.

A.3.1 Distributed information retrieval

Problem description: information is distributed across different databases in possibly heterogeneous environments. The aim is to integrate this information in a way that is transparent to the user.

Solutions: existing solutions aim at integrating the heterogeneous sources in a central system. Many variations exist in the way this is done: from no co-operation whatsoever by the distributed archives to a situation where the archives deliver all their information to a central database. A similar solution to OmniPaper is in the MIND project.

Possible paradigms:

- Pull vs. push: In pull, the central system is responsible for gathering all information from the distributed archives; in push, the distributed archives deliver their content to the central system at their own initiative.
- Central database vs. distributed databases: in the first case the entire contents of the distributed archives is stored in a central database system; in the second case no central database exists, but all queries to the central system are in some way forwarded to the distributed archives.
- Adaptation of distributed archives: in a minimalist approach the information is “grabbed” from the distributed archives’ website without any adaptation or even co-operation from the archives[2]; at the opposite of this, the distributed archives adapt their databases in such a way that they become interoperable.

Relevance for OmniPaper:

OmniPaper uses an approach where the distributed archives are co-operating but with a minimal “change”. Using a SOAP extension to their database server, the archives can be plugged into the system.

A.3.2 Online news publishing

A.3.2.1 Online newspapers

A lot of newspapers are being published electronically on the web. Traditional newspaper publishers see online news often as a “teaser” for selling their paper version. Online news is derived from the same data source as is used for paper publishing, but a lot of differences can exist. Most online newspapers offer similar functions as OmniPaper: browsing trough news categories, searching for news using full-text and metadata and viewing news articles. See also [WP5REQ].

This category also contains news websites of television channels and news agencies.

[2] An example of this method can be found in: Ruben Tous & Jaime Delgado: “Interoperability Adaptors For Distributed Information Search On The Web”. In: Sely Maria de Souza Costa et. al. (eds): Proceedings of the 7th ICCC/IFIP International Conference on Electronic Publishing, Universidade do Minho, Portugal, 2003.

Examples:

– CNN: http://www.cnn.com/

– Reuters: http://www.reuters.com/

– Flemish television: http://www.vrtnieuws.net/

Relevance for OmniPaper:

The OmniPaper final prototype is also a kind of online newspaper. The difference is that it contains news from many news sources in many languages and that it uses ontology-based search and navigation. Also, the OmniPaper final prototype will not have as much functionality as most online newspapers, like the display of PDF versions of printed newspapers, multimedia material, etc.

A.3.2.2 News portals

News websites like the ones mentioned above mostly host only one or a few news sources. News portals are collecting news originating from many sources. They only provide an access gate to news: when a reader wants to retrieve a full article, he gets redirected to the website that hosts the particular article. The disadvantage here is that the user interface is always changing, making the news experience less user-friendly. The portals offer only the latest news items and do not offer searching in older news articles.

Examples:

– Google News: http://news.google.com/ (continuously searches on 4500 online news sources – fully automatic categorisation – goes back to first day of previous month)

– Kranten.com: http://www.kranten.com/ (24 Dutch and Belgian newspapers)

– NewsNow: http://www.newsnow.co.uk/ (automatically searches on 10880 online news sources every 5 minutes)

– Yahoo! News: http://dailynews.yahoo.com/ (goes back to 30 days maximum)

– NewsIsFree: http://www.newsisfree.com/ (collects headlines from over 6000 news sources)

– World News Network: http://www.worldnews.com/

– Columbia NewsBlaster: http://www1.cs.columbia.edu/nlp/newsblaster/ (generates automatic summaries of a collection of news articles about a certain topic)

Relevance for OmniPaper:

This is a very relevant area because the OmniPaper final prototype is also a news portal service. Competition is high because most news portals are quite good, have a lot of sources and are for free. Technically there are some differences though. While in the OmniPaper system the news archives co-operate using SOAP adapters, most news portals “grab” news from other online sources. Thanks to this co-operation the OmniPaper system has access to valuable metadata. A final difference is the ontology-based search and navigation of OmniPaper (using the widened and narrowed search), which is a quite new concept.

A.3.2.3 News clipping services

As opposed to news portals, these services are much more aimed at servicing specific customers that often automatically want to receive news about certain very specific topics. This way a personalised electronic newspaper is offered to the customer, for example containing all published news about his company or field of expertise. Because customers pay for this service, news clipping services have to make reselling agreements with their news sources.

Examples:

- My News: http://www.mynewsonline.com
- Mediargus: http://www.mediargus.com
- CustomScoop: http://www.customscoop.com/
- ClipGenius: http://www.clipgenius.com/

Relevance for OmniPaper:

Similar as news portals, but news clipping services have extra functionality, like profiled news publishing for specific customers and delivery of news by e-mail. Also, their content is gathered with the co-operation of the publishers, making it more valuable trough metadata. The ontology-based search and navigation could be a valuable addition to existing news clipping services.

A.3.3 Search engines

Problem description: find a small part of information in a "big pile". This information can be available on the Internet or on any other source.

Solutions: most search engines use full-text search for mapping the user query to the criteria used for searching in the big information pile. In order to make searching fast, smart indexes are maintained of the information pile. A lot of search engines make a distinction between simple and advanced search. In advanced search a number of search options can be defined, such as Boolean operators for combining different search words, language limitations, date, author, etc. Some search engines use stop word elimination for making a better mapping between the user query and the search operation. More sophisticated search engines use query expansion for increasing search performance.

Examples:

- Google: http://www.google.com (stop word elimination)
- Excite: http://www.excite.com
- Yahoo: http://www.yahoo.com
- Altavista: http://www.altavista.com

Relevance for OmniPaper:

A lot of techniques used by search engines are also useful for OmniPaper, for example the use of boolean operators, query expansion and stop word elimination.

A.3.3.1 Question-answer systems

Queries to search engines are mostly keyword-based. One step further is natural language queries, which allow "normal" sentences in such a way that a normal-sounding question is submitted to the search engine. The engine translates the question in the background to "search engine understandable" information.

Examples:

- Ask Jeeves: http://www.ask.com/ (uses databases of pre-compiled information, meta-searching, and other proprietary methods)

Relevance for OmniPaper:

This is a really different area of work, but some principles can be helpful, like the conversion of natural language queries into machine-processable queries.

A.3.3.2 Search result clustering

Traditional search engines return results in the form of a list, ranked by relevance, date, or other ranking criteria. Systems that perform result clustering apply a more semantically oriented approach, where results are not merely listed but grouped into meaningful result categories. This way, users have the opportunity to assess relevance in a more intuitive way.

Examples:

- Vivisimo: http://vivisimo.com/ (automatically organizes search or database query results into meaningful hierarchical folders)
- Northern Light: http://www.northernlight.com
- Carrot and Carrot^2 project: open source component-based framework for search results clustering: http://www.cs.put.poznan.pl/dweiss/carrot/index.php/index.xml

Relevance for OmniPaper:

This is a very interesting way of presenting search results in a more intuitive way than using a traditional result list with relevance ranking applied to it. A possible scenario for OmniPaper could be the integration of the “web of concepts” way of navigating with a more semantic way of displaying search results. Resulting news articles could then be visualised around semantically related concepts.

A.3.3.3 Semantically enhanced searching

Some search engines are experimenting with semantic enhancements to their traditional full text search robots. Google for example recently introduced a “synonym search”: using the tilde sign (~) before a query term, this term will be expanded into synonyms. For example the query “~violence ~classroom” results in a result list that also contains the words “education”, “learning”, “abuse”, “teaching” and “crime”.

A.3.4 Ontology-driven systems

These systems provide indexing, searching, navigation or filtering through the use of a smart, semantic-based, index or ontology. The ontology can be created manually or – to some extent – automatically. Popular areas of application are knowledge management and organisation, document management, etc.

Examples:

- Autonomy: http://www.autonomy.com/ (automatic content aggregation and organisation)
- Alliance project (automated creation of Topic Maps containing semantically related words – create clusters of words that have to be tagged manually)
- Metamorphosis: Automatic creation of Topic Maps for navigation based on metadata of documents. See the paper “Metamorphosis: A Framework to Specify and Manage Ontology Driven Websites” by Ramalho, J.C., Librelotto, G.R. and Henriques, P.R. in Sely Maria de Souza Costa, João Álovaro Carvalho, Ana Alice Baptista, Ana Cristina Santos Moreira: ELPUB 2003 Proceedings, Universidade do Minho, 2003.
- KAON: open source ontology management infrastructure. Uses RDF as ontology format. Contains an API for accessing an ontology in a programmatic way, for querying the ontology and for creating

a web portal based on the ontology (amongst others). Ontologies and RDF models are stored in a relational database. See http://kaon.semanticweb.org/. They have converted WordNet into their ontology format, the result of which can be downloaded at their website.

Relevance for OmniPaper:

Of course very relevant because OmniPaper also uses an ontology (WordNet) in the form of a Topic Map. The novel approach of OmniPaper is in providing search with this ontology. Most existing systems only use this for navigation. KAON is particularly interesting because it could provide us with a platform for storing our ontology and accessing it through Java.

A.3.4.1 Automatic classification of documents

Statistical methods allow the automatic classification of documents to a predefined classification tree (thesaurus). If a sufficient number of training documents are available, some sources claim that the precision of such classifiers might reach values of up to 95%.

Examples:

See the papers “Cross-Lingual Text Categorization” and “Automatic Multi-label Subject Indexing in a Multilingual Environment” in in T. Koch and I.T. Sølvberg (Eds.): ECDL 2003, LNCS 2769, pp. 126-139 and 140-151, Springer-Verlag Berlin Heidelberg 2003.

Relevance for OmniPaper:

A possible way for attaching the news articles to the common IPTC subject reference could be the use of an automatic document classifier. The first paper cited above claims to have quite promising results in classifying documents in different languages to the same classification tree.

A.3.4.2 Semantic browsing

“Semantic Browsing” is about adding semantics to web sites using RDF and visualising these semantics in the browser. If an RDF file about the website structure and semantics is added to the web server directory, it can be accessed by something like a “Web Task Pane” in the browser, which then allows a way of navigating through the site that is more structured than the traditional hyperlink way. Using a “Web Annotate Pane” the RDF statements can be created in a user-friendly way. With this pane users can indicate semantic associations between objects such as pictures, text documents, etc.

Examples:

– Semantic Browsing project at Cornell University, U.S.A. See the paper “Semantic Browsing” by Alexander Faaborg and Carl Lagoze in T. Koch and I.T. Sølvberg (Eds.): ECDL 2003, LNCS 2769, pp. 70-81, Springer-Verlag Berlin Heidelberg 2003.
– Add-on for Mozilla browser: Annozilla sidebar (http://annozilla.mozdev.org)
– Other semantic annotation tools: http://annotation.semanticweb.org/tools

Relevance for OmniPaper:

A possible UI scenario for OmniPaper could be to use this “Web Task Pane” as a navigable addition to searching. The sidebar could be an alternative to the current web of concepts; it would then also continuously adapt to the query and to other user behaviour. However it can be estimated that this lies beyond the scope of the project.

A.3.5 Geo-reference systems (gazetteer)

Geo-reference means that information is referenced using its geographic position rather than a keyword, title or name. This is very useful with geographically related information like tourist information, historical events or cities. Geo-referencing helps avoiding problems like different names for cities in different time periods.

Example:

– Alexandria Digital Library Project of the University of California, Santa Barbara, U.S.A.: http://www.alexandria.ucsb.edu/

Relevance for OmniPaper:

For OmniPaper geo-referencing might be useful because a news event is always linked to a geographic place (news happens somewhere in the world).

A.3.6 User interface

Related to user interface issues, all systems using meta-information are related systems, as they have some important comparable aspects which are meeting the OmniPaper objectives as well. In this subsection related system solutions are examined and typical user interface elements and solutions are captured to study them with regard to a potential use in the OmniPaper user interface. The most relevant related systems can be clustered into the following groups: digital libraries, online newspapers, search engines, multilingual portals, display solutions for large networked information spaces, recommender systems and automatic translation services.

A.3.6.1 digital libraries and electronic archives

As OmniPaper is a news archive, knowledge and best practice from digital library system should be taken into account. But there are major differences between library systems and OmniPaper: libraries offer more static content, while news are highly dynamic generated. Additionally the categorisation and processing of the content within OmniPaper will be automated, even the categorisation will change dynamically, new categories will be added.

Examples:

Association for Computing Machinery - Digital library (acm.dl),
Austrian National Library (aleph.onb.ac.at)
IEEE Computer Society Digital Library (www.computer.org/publications/dlib/)

Typical user interface solutions and elements:

- simple and advanced search
- login, user identification
- different sorting possibilities
- preview and detailed view of results
- download possibility
- bookshelf
- help
- different viewing options for results
- possibility to discuss/review article
- classification scheme
- make use of relations (e.g. citations)

A.3.6.2 online newspapers

As OmniPaper not only aims at serving as archive but also to provide news “as they happen”, concepts of online newspapers need to be studied.

Examples:

El Pais (www.elpais.es), Spain

Der Standard (derstandard.at), Austria
International Herald Tribune (www.iht.com), International
De Standaard (www.standaard.be), Belgium
The Times (www.timesonline.co.uk), United Kingdom

Typical user interface solutions and elements:

- overview start page with headlines
- different channels
- different possibilities to sort (e.g. by chronology or relevance)
- external links to source
- conceptual separation of actual news and special columns (weather, …)
- some articles are especially emphased (teaser)
- difference to print edition
- download whole newspaper as pdf
- news ticker

A.3.6.3 Search engines

To provide the users with most proper search interfaces and functionality the offer of professional search engines shall be analysed.

Examples:
Google (www,google.com)
Alta Vista (www.altavista.com)
Teoma (www.teoma.com)

Typical user interface solutions and elements:

- simple and advanced search
- result list
- offering of general search restrictions (e.g. only results in German language…)
- general preferences/options
- refinement suggestions
- display of estimation of quality of search
- spell check feedback

A.3.6.4 Multilingual portals

An important aspect of OmniPaper is that it is an multilingual system. Therefore experiences from other sides shall be used to avoid making errors that aren't necessary.

Examples:
Home page of the European Union (europa.eu.int)
European Patent Office (www.european-patent-office.org)
United Nations Organisation (www.un.org)
Yahoo! (www.yahoo.com)

Typical user interface solutions and elements:

- permanent possibility to select language vs. single decision
- offering documents to download via different links
- parallel/similar display of same content in multiple languages
- mixed display of content (as it is), but the language is explicitly displayed
- automatic translation

A.3.6.5 Recommender systems

As artificial intelligence is used to recommend related articles to the user existing solutions and their interfaces are analysed to be able to develop a highly usable solution.

Examples:

Movielens (www.movielens.org)
Amazon (www.amazon.com)
The Internet Movie Database (www.imdb.com)

Typical user interface solutions and elements:

- provide estimation of how much user will “like” recommended thing
- show highest rated recommendation (by categorie, date, …)
- feedback possibility if recommendation was correct/liked (implicit or explicit)
- explanation on why article was recommended
- possibility to explicitly tell the system preferences

A.3.6.6 Display solutions for large networked information spaces

OmniPaper will provide the possibility to navigate through the archive in a semantic structure. As the displaying of networked information is highly complex due to restrictions in screen space different approaches that have been undertaken shall be analysed whether they can be applied to the OmniPaper system or not. Due to the special complexity of the area the solutions are described in more detail than the other related systems.

Examples:

Omnigator (www.ontopia.net)

The Omnigator is a technology showcase and teaching aid designed to help understand the power of topic maps. It is a tool designed to display the information that is stored in topic maps for developers. As the designers specify themselves: “The user interface of the Omnigator is emphatically not what we would recommend in an end-user application!”

NicheWorks (www.amstat.org/publications/jcgs/pdf99/wills.pdf)

NicheWorks is a visualization tool for the investigation of very large graphs. NicheWorks allows the user to examine a variety of node and edge attributes in conjunction with their connectivity information. Categorical, textual and continuous attributes can be explored with a variety of one-way, two-way, and multidimensional views.

Matrix Browser (www.swt.iao.fhg.de)

The approach is based on the use of an interactive adjacency matrix for the representation of relations between concepts forming the nodes of the net. These concepts are either displayed as lists or hierarchies on both axes of the matrix. Different interactive markings in the matrix cells represent the relations between the concepts.

Kartoo (www.kartoo.com)

Kartoo is a meta search engine that displays the results in a cartographic representation of information. The particularity of the system is, that in the map relationships between the results are displayed as well.

Vivisimo (http://vivisimo.com)

The Vivísimo Clustering Engine automatically organizes search or database query results into meaningful hierarchical folders on-the-fly, out-of-the-box. It interfaces with any search engine or document database, transforming long lists of search results into categorized information without any clumsy pre-processing of the source documents.

Hyperbolic browser (http://www.ulib.org/webRoot/_hTree)

The hyperbolic browser is a focus+context (fisheye) scheme for visualizing and manipulating large hierarchies. The approach is to lay out the hierarchy uniformly on a hyperbolic plane and map this plane onto a circular display region. The projection onto the disk provides a natural mechanism for assigning more space to a portion of the hierarchy while still embedding it in a much larger context. Change of focus is accomplished by translating the structure on the hyperbolic plane, which allows a smooth transition without compromising the presentation of the context.

Typical user interface solutions and elements:

In general the different approaches to deal with big networked information can be clustered into the following two groups:

a) approaches that use a graphical representation for displaying relationships between concepts and

b) approaches, that rely on text to communicate the different relationships.

The big disadvantage of the graphical approach is, that only a limited and small set of items can be displayed at one time, so in most cases only a small clipping of the data can be presented. Also typically a lot of screen space is not used. Further disadvantages are difficulties to display and properly mark the different kind of relationships and that re-zooming needs a lot of calculation and therefore can significantly slow down the process of displaying the information.

Disadvantages of the “textual” approach are, that it is difficult to get an overview on the overall structure of the concepts and relationships between them.

PART B RESEARCH WORK, RESULTS AND GUIDELINES

This part of the Blueprint describes the performed research and development work conducted during the OmniPaper project. In a first subsection the OmniPaper architecture for distributed information retrieval is described. The second subsection details the development work done during the project. This part concludes with a section describing the user interface guidelines used for the OmniPaper prototype development work; these guidelines are also useful in other information retrieval prototypes.

B.1 OmniPaper Architecture: a Framework for Distributed Information Retrieval

This section gives an overview of the architectural reasoning conducted during the initial project phase of design and requirements analysis work within the OmniPaper project. The overall system architecture provides an integrative description of the components designed and developed for the creation and practical use of the OmniPaper system prototype.

The overall architecture describes requirement analysis results, involved logical components, usage views and information management processes developed for the OmniPaper prototype. The document aims to act as a general building plan for the prototype and provides support for component integration and overall management presentations of final functionalities and technical capabilities of the built service.

B.1.1 OmniPaper – Architectural Requirements

Since the emerging boom of the Internet a lot of newspapers are being published electronically. This increasing amount of news items remains scattered throughout various archives, countries and languages. Furthermore, the distributed electronic information has different data structures, storage formats and access methods. Searching for news is still mostly done the “brute force”-way using full-text search robots which lead to a search result quality that highly depends on the sophistication of the user’s search input. In fact - finding news from various international newspapers is still easier in an airport news kiosk than on the Internet. The current project is investigating techniques to obtain a novel online news experience, using up to date XML- and AI-related technologies.

The OmniPaper architecture starts from distributed news archives, all within different operating environments, database formats and indexing mechanisms. Heterogeneity, performance and usability are challenges to the responsible system architects. In a standardization effort, SOAP (Simple Object Access Protocol) has been selected to create a uniform access method to the existing archives. In addition to the simple access requirements, the intelligent news archive is required to extract specific contents and create relations between information units. Rich indexing and meta-data structures, such as Topic Maps and RDF are utilized to make intelligent search possible. A cross-archive intelligent index (or ‘knowledge layer’) contains concepts, relationships between them and occurrences in different languages.

Figure 1: Abstract architecture of OmniPaper

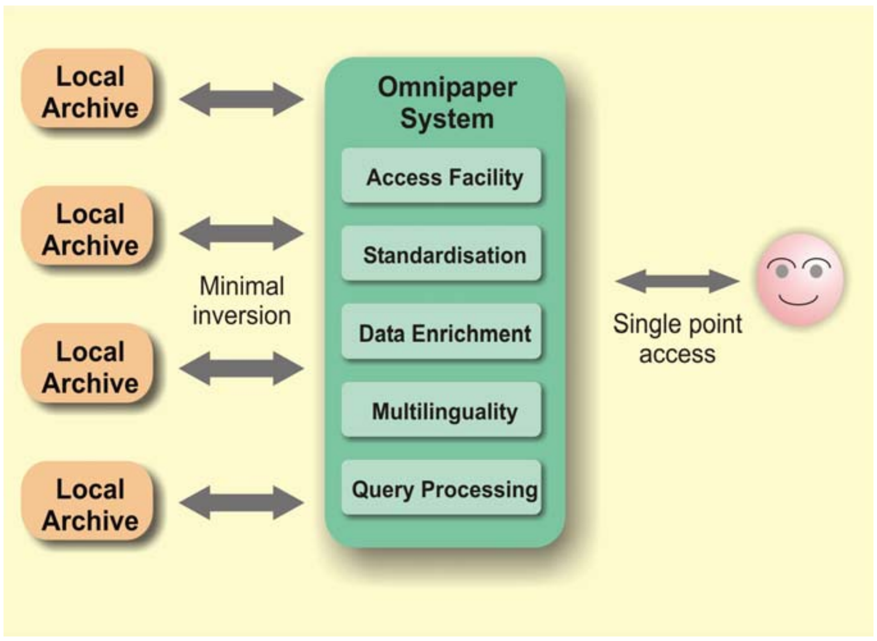


Figure 1 shows an abstract architecture of the OmniPaper system. Distributed source data exists in various standard formats, different languages and with varying depth of available information. The existing *local archive*s are connected to the OmniPaper system using minimal inversion i.e. existing access methods are reused and wrapped to standardized SOAP requests.

From the user interface, the architecture provides access via NL Queries or browsing the content by categories in the user's own language. The provided results are retrieved from all local archives.

B.1.2 Architectural Design

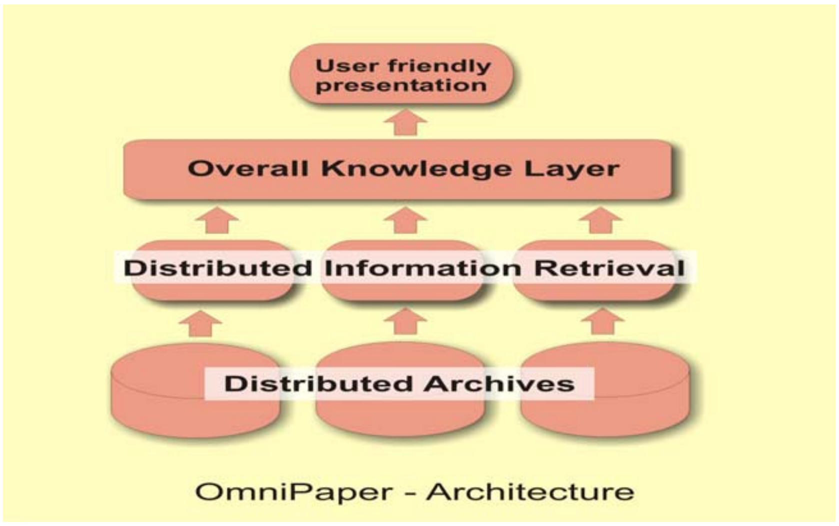
In order to meet the goals and fulfil the requirements of the multilingual news archive, research on and evaluation of related international projects[6], retrieval methodologies[11] and semantic relation approaches[13] has been applied in the field of digital libraries and news archives. Integrating experiences in complex technical environments in the area of Internet Services [10], we created a system architecture for the distributed heterogeneous news archive like the OmniPaper system.

The architecture is discussed from three different perspectives: we defined views on system components and system processes as well as on system interfaces. On the first hand a top level architecture shall describe an appropriate grouping of system components and related processes and give an overview on two major different usage interfaces.

B.1.2.1 Top Level System Architecture

The top level system architecture as shown in Figure 2 contains a multi-layer view on the technical architecture of a distributed news archive. The architecture is based on existing digital news archives as the bottom layer. The distributed information retrieval layer contains components and control processes that access directly or indirectly the existing archives for news retrieval and metadata management. The overall knowledge layer combines the features of integrating distributed information with the capability of creating semantic coupling of the corresponding content. The multilingual aspect is supported by extracting existing keywords and metadata from the heterogeneous archive information and associating it with existing domain specific thesauri for the relevant language. The overall knowledge layer contains a network of thesauri, thus coupling corresponding standardized terms and enabling the intelligent news archive to find corresponding articles in news archives over different countries and languages. Based on these layers, the topmost user interface layer allows journalists and researchers to investigate material on specific topics in a multilingual environment, relying on high result quality and content relevance.

Figure 2: Top level system architecture



Provider View

The top level system architecture from Figure 1 is constructed mainly from the point of view of a news provider for the end-user. Existing archives and news repositories act as data sources for the distributed news archive and provide their contents via standardized SOAP interfaces. The system itself offers simply a *data feed interface*, acting as black box for the existing archives. To provide openness and facilitate easy archive extension the used set of SOAP queries is limited to seven requests:

FullTextSearch
: Users may query existing archives for fulltext search. The system creates the corresponding SOAP request and relevant parameters and accesses the existing archives.

ParameterSearch
: Users may query for news articles using meta-data fields. A full-text search is performed in the given metadata fields. Since ParameterSearch is matching exact values appearing in metadata fields, no multilingual queries are supported as opposed to fulltext search.

IdentifiedSearch
: This request uses a system-wide unique ID to address an individual news article. The existing archive responds with all information on the requested article.

NewsUpdate
: The existing archive informs the distributed news archive about new articles. The requests transfers necessary information and initiates metadata retrieval.

MetaDataReady
: The SOAP request MetDataReady informs existing archives that MetaData on presented articles are available.

MetaDataFeedback

The relations and attributes for each article created within the intelligent multilingual distributed news archive are managed and updated regularly. The providing archive may request the latest version of meta data on the article at any time.

NewsSolicit

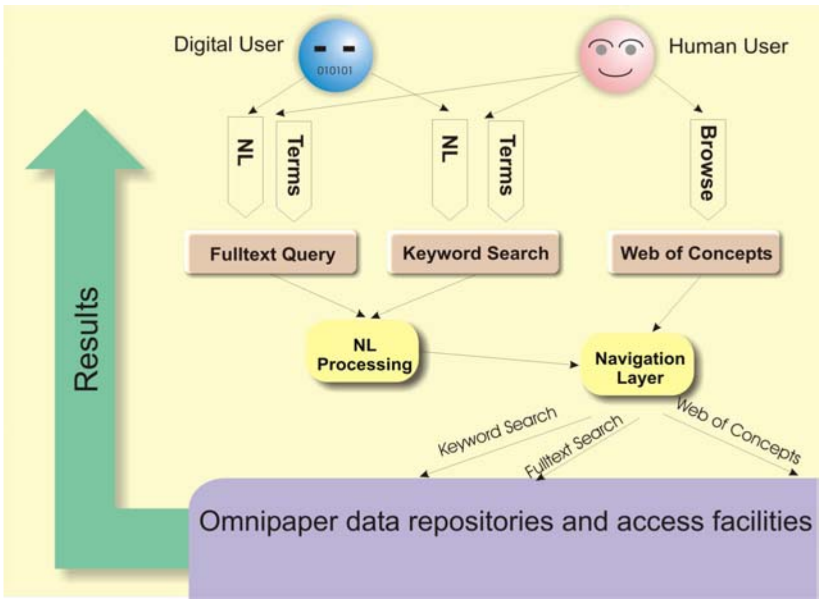
With the NewsSolicit Request the local archive is asked if new articles are available. The local archive decides on the presented parameters which articles have to be sent to OmniPaper.

Detailed definitions of the SOAP requests and relevant parameters are defined with the OmniPaper project documentation and are outside the scope of this blueprint. The usage of the SOAP queries within the providers interface is discussed in more detail in Section 3.3, News Archive Processes.

User View

From the user's perspective the distributed news archive offers a completely different interface. Either digital users (applications) or human users may access the archive using natural language queries, set of terms or browse through a web of related terms (concepts). The user interface layer translates the user queries into a manageable format for the distributed news archive as shown in Figure 3.

Figure 3: User interface to OmniPaper



The user interface is designed in the system architecture as a user-centered Web application. For digital users both simple access to Web forms and XML Web Services are planned.

B.1.2.2 Distributed News Service Components

Several information repositories with different types of content handling and interrelation mechanisms are utilized to compose the distributed news archive. A general overview of the OmniPaper system architecture is presented in the following figure. Components are designed basically as data repositories (database symbols) and coloured in yellow and green. Yellow components are manipulated directly by user accesses or manipulation processes from external interfaces. Green components are internal auxiliary components used to assist internal information processing and management within the OmniPaper system.

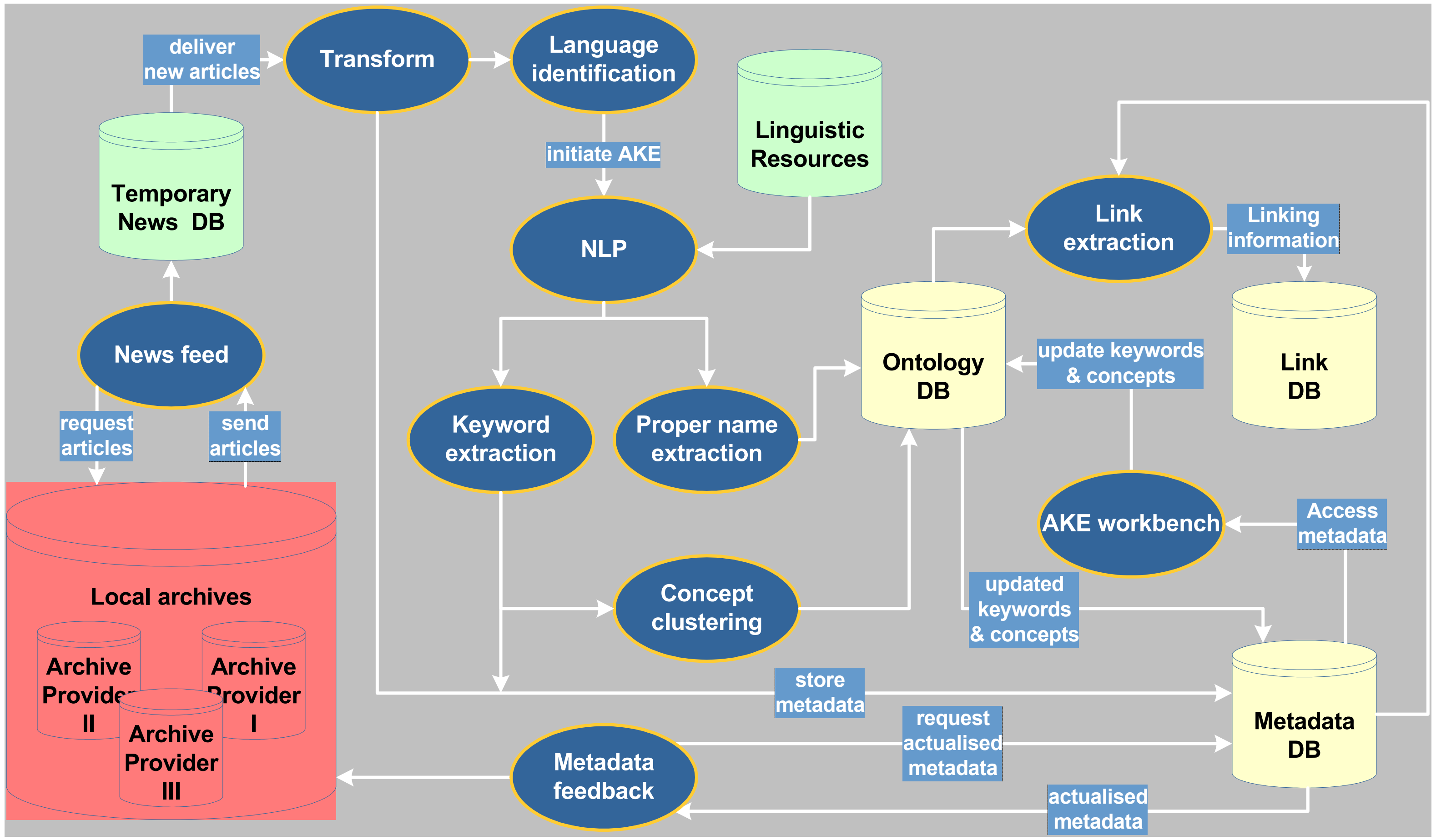
Processes are designed in the graphical architecture description as circles containing the corresponding process description. Interfaces between the individual processes and data exchange between the components are abstractly defined as links between the processes and components. Detailed specifics are delegated to the process implementations and described within the individual prototype descriptions. Generally, the OmniPaper consortium decided to use the XML-based standard SOAP as the communication protocol between external components (here coloured in red) and the OmniPaper service, but also plans to define internal interfaces between processes by SOAP Web Services. This way of information exchange is technically an overhead but provides strong modularity within the defined and developed subsystems.

Table 2 focuses on data repositories as part of the architecture. Several processes are managed to maintain the news information and create and store knowledge within this components.

Table 2 OmniPaper Components

Component	Intention
Local Archives	News archive(s) that manage significant amounts of information and provide proprietary interfaces for search and retrieval. SOAP query handling for three mandatory requests types (query 4 is optional) is required
Temporary news DB	Temporary news database for unprocessed news articles. This repository within the distributed news archive system is accessed within the feed news process. The content is processed within regular intervals. In the current prototype this temporary Database is optional, direct access to existing archives is provided through the above mentioned SOAP interfaces.
Ontology DB	Ontology Database, listing keys & concepts. Attributes are stored for each news article based on AI technology. Vectors of keywords and weights are managed to be recognized as related to abstract or concrete concepts (see 3.3.2.3)
Linguistic Resources	The Linguistic Resources Database holds a set of language specific thesauri and predefined relations between terms and keywords across languages.
Link DB	Database that holds automatically extracted links from a specific article to other information within the distributed news archive
Metadata DB	The Metadata-Database manages a set of data for each article within the distributed news archive. User queries are targeted to this database and forwarded to specific repositories if necessary.

Figure 4: OmniPaper knowledge layer architecture



B.1.2.3 News Archive Processes

Components and processes are either close to the news providers, thus providing small and manageable functionality to access and retrieve news information or they are used to build and maintain knowledge and create an intelligent distributed news archive.

In order to distinguish the utilization of the processes and describe a structured picture of the process architecture, the sub processes are organized in groups of “acquisition processes”, “knowledge management processes” and “user access processes”. In the following subsections a brief overview on the acquisition processes and the content access processes is given. See further details on the process management in the detailed prototype descriptions in sections B.2.1 and B.2.

Acquisition Processes

These processes cover all activities that involve the news providers in sending or retrieving news articles to the distributed news archive.

The acquisition processes include those for integrating new articles in the Overall Knowledge Layer of the OmniPaper service, to process Metadata and to give feedback on the current metadata status.

Access Processes

Access activities in the distributed news archive are triggered by human or digital user interaction. This type of processes take requests from the user interface and make use of the results of the previously described processes.

Process Fulltext Search

The fulltext search process accepts a set of terms from the user interface. A preceding natural language processing may transform a provided NL-query into the appropriate set of terms.

The set of terms is passed to the SOAP FulltextSearch query that is targeted to all involved existing archives.

Results are collected and fused by the distributed news archive. Specific cost measures, e.g. timeout, system availability, preferences determine the quantity of the result.

As a result, the user interface is provided with an XML listing of matched news articles.

Process Keyword Search

The *keyword search process* accepts a similar input as the *fulltext search process*: a set of terms is passed to it and handed over to a query on the MD database.

The caching nature of the MD database within the distributed news archive system provides fast access and immediate response for the user interface compared to the fulltext search process.

The information stored in the MD database is extracted according to the keyword search and the user interface is provided with an XML listing and weight information of matched articles within the distributed news archive.

Browse Web of Concepts

The process *browse web of concepts* provides an hypermedia approach to information retrieval within the distributed news archive.

At the user interface, named concepts are visualized with their relations between them and an appropriate browser provides the navigation through this web of concepts according to the user's input.

After selecting a specific concept, the process browse web of concepts is provided with the set of terms corresponding to the chosen concept. The set of terms is forwarded to the MD database for information retrieval.

As a result, the user interface is provided with an XML listing of news articles corresponding to the currently selected concept. The user interface is responsible for visualizing the returned results.

The prototype Web interface is capable of browsing the Web of Concept textually and graphically. The textual version allows widening and narrowing of related terms based on international thesauri (EuroWordnet) using Web Form entry fields. The graphical version uses SVG to display terms and their linguistic and semantic relations as nodes and links in a coloured graphic (see more details in section B.2.4).

Process Show Article Details

All previously described access processes produce as results XML listings of matching news articles. Every time a user interface requests a specific article, the process *show article details* is triggered.

Depending on the requested details, the process requests the information solely from the MD database or initiates the SOAP IdentifiedSearch request to the source archive, for which the id is also retrieved from the MD database.

The article details are provided together with metadata information on related keywords, concepts and links to further related articles within the distributed news archives.

As a result, the article details are provided to the user interface, which is responsible for visualization and enhancement of content and meta-information presentation.

B.1.3 Architecture Layering in OmniPaper

As a mature result of the project consortium, the discussed system architecture is currently implemented and verified with different system prototypes. The OmniPaper project follows the abstract concept defined in section 7.1 and abstractly structures the processes and components in layers.

B.1.3.1 Local Layer–Interface to existing archives

The local layer provides standardized interfaces to three news (re)distributors and enables to access about 8.7 million documents. The SOAP based interface provides unified access to the local databases t(existing as oracle, mysql and DBMS systems), hosted on Windows and Linux systems The local layer contains data structures to retrieve newspaper information from distributed archives. The queries focus on article selection by search criteria and keyword extraction.

B.1.3.2 Overall Knowledge Layer – AI, Multilingualism, Knowledge Management

The results from the local layer queries constitute the input for data management on the overall layer which is developed in distributed units. According to the architecture the required components and repositories are used and new techniques are analyzed and compared. XTM and RDF based auxiliary databases are core units of this layer, which enhances the initial available data by the use of AI, Multilingualism and Knowledge Management.

B.1.3.3 Usage Layer – Presentation Interface

A user-friendly presentation of the system, based on current HCI understandings, will be set up above the overall knowledge layer. Efforts in the corresponding project work packages concentrate on the news search engine, the display of the results from processes in the overall knowledge layer, and the visualization of newspaper articles and cross-links.

Obeying current technological developments, the user interface is planned to be developed on modern web service and web browser technology. Nevertheless, intelligent features from the overall knowledge layer need to be presented to the user in order to excel current information retrieval approaches.

- Special retrieval interfaces for multilingual search results
- Capturing of user behaviour for profiling and relevant search results

- ➢ The use of a central metadata database in a distributed information retrieval system creates the need for developing a sound update mechanism in order to keep the central DB up to date.
- ➢ In the area of news – which is a very volatile kind of information - the use of a central metadata database in a distributed information retrieval system will create a huge overhead on the day-to-day management of this system.

B.2 Prototype development: access gate to distributed information sources

B.2.1 Metadata definition

Within OmniPaper, a standard format for the news contents has been defined. These formats are used by all prototypes. The content of the system is actually the metadata of the articles; it is written in XML, which helps its interaction with standards like RDF and/or XTM. The connection with the content providers is via SOAP. Requests are used to retrieve documents for processing on uploading and to show the contents to the user on downloading, once the system has determined, based on the abovementioned metadata, that a news article fulfils a user's request.

The standard format for the metadata has been defined following widely accepted standards, in particular the Dublin Core Metadata Element Set (DCMES), and the News Industry Text Format (NITF), but also NewsML [23]. The format describes twenty three basic elements, grouped under these categories: Identification, Ownership, Location, Relevance, Classification, and LinkInfo.

→ The Identification category includes an identifier for the news metadata refers to (so that it can be requested to the provider), and sub-elements like Creator, Title, Subtitle, Publisher, Language, etc.

→ The Relevance category contains information that can be used to compare with the user profile to decide on how the news fits the intention of particular users based on their behaviour.

→ The LinkInfo category contains links and references to related documents that can be used to improve the results on particular information requests.

→ The Classification category contains sub-elements that allow classifying the document on certain criteria; in particular, and most relevant to the present paper, the Key_list and the Subject.

The Key_list has sub-elements which are the keywords identified in the keyword extraction process described in the rest of this paper. Based on these keywords, the document can be classified obtaining a Subject, which is also a sub-element of Classification. This subject is usually the main one, and can be used for relating the document to other documents based on the navigation of a Topic Map of categories and related subjects.

The tables below summarise the twenty-four defined metadata elements divided by six categories. Detailed explanation for each element's definition and use as well as the encoding schemes follows this section.

B.2.1.1 Category: Article Identification

The elements in this category contain basic information about the article's identification.

Metadata	Definition	Encoding Scheme(s)
Identifier	An unambiguous reference to a resource (an article).	URI
Creator	Author of an article.	vCard
Issued	Date of publication of an article.	W3C-DTF
Title	Title of an article.	
Subtitle	Subtitle of an article (if any).	
Publisher	The entity which an article belongs to.	

Language	The language in which an article is written.	ISO 1766 & 639
KindOfText	Nature or genre of an article.	
Section	Named section of a publication where an article appears.	
Edition	The name(s) of edition(s) in which an article is distributed.	

B.2.1.2 Category: Article Ownership

Information on copyright and ownership is described in this category.

Metadata	Description	Encoding Scheme(s)
Copyright	Container for copyright information.	VCard & W3C-DTF
Owner	The local archive that owns the article.	

B.2.1.3 Category: Article Location (Storage)

This category's elements are used to describe an article as data and also to specify the location of the article.

Metadata	Description	Encoding Scheme(s)
Medium	The physical or digital manifestation of an article.	IMT
Source	A reference to an article from which the present article is derived.	URI

B.2.1.4 Category: Article Relevance/Audience

The elements in this category try to define the relationship between the article and the user.

Metadata	Description	Encoding Scheme(s)
OfInterestTo	The target audience for an article, based on demographic, geographic or other groups.	DC Audience level?
Valid	Date (often a range) of validity of an article.	DCMI Period, W3C-DTF
Spatial	Geographical location that an article treats or is related to.	DCMI Point, ISO 3166, DCMI Box, TGN

B.2.1.5 Category: Article Classification

This category contains the elements used to classify the articles.

Metadata	Description	Encoding Scheme(s)
Abstract	A summary of the content of an article.	
Key_list	A list of keywords extracted from an article document.	
Subject	Topic of the content of an article, specified according to the common thesaurus.	IPTC Subject Code System, etc.

B.2.1.6 Category: Link Information

The elements in this category describes links between articles.

Metadata	Description	Encoding Scheme(s)
HasPart	The described article includes the referenced resource such as photo, table, diagram, etc	URI
IsVersionOf	The described article is a version of the referenced version	URI
Series	The name of series that an article belongs to	
References	The described article references, cites or points to the referenced article	URI

B.2.2 Distributed Information Retrieval Prototype (WP2)

In this project a **bottom-up approach** was used for prototype development. Several smaller prototypes have been developed in order to cross-test different technologies and to limit the project development risks. These smaller prototypes have been combined into the Distributed Information Retrieval Prototype.

All prototypes in this category use the same data set: a set of 1881 English articles from the Daily Telegraph (all published in September 2002). These articles have been provided by My News.

B.2.2.1 SOAP

The purpose of this prototype is to analyse and test how SOAP can be used for direct retrieval of news articles from a remote news archive. Using SOAP, it tries to solve the problems deriving from incompatibilities of remote computer systems and information retrieval. The prototype serves as a basis for further prototype development. It will develop and refine the SOAP functionality used by other prototypes

The prototype has three main functions for the end user:

1. Simple search: this allows user to search for keywords (or phrases) using full-text search in the entire news article and all its available metadata.

2. Advanced search: this allows users to search for keywords (or phrases) using a search in one or more specific metadata items of the news articles.

3. Hierarchical subject view: this allows users to browse through news categories and view all news articles in a specific category.

In all three cases, the user gets a list of all news article titles that correspond to the search action. When the user clicks on a title, the corresponding news article is shown.

The prototype has a simple web-based user interface allowing users to browse through news categories and to perform simple and advanced searches. Through the HTTP/SOAP communication protocol, the system works together with the SOAP server at the (remote) news archive site.

All queries or navigation actions by users are translated in the background to SOAP request messages, which are forwarded to the remove news archive. There a SOAP server responds to the request by sending a SOAP response message. So all of the search actions are done by the remote news server.

B.2.2.2 RDF

The main purpose of the Local Knowledge Layer is to provide a standard semantic description of all the existent articles in order to enable a structured and uniform access to the available distributed archives. The RDF prototype investigates efficient ways to describe and store the metadata information of provided news articles using RDF and related technologies. The results of this approach have been compared to the XTM approach and the SOAP approach.

The WP2 RDF prototype has the following functionalities:

Store information: The prototype is able to store received metadata described in predefined RDF/XML template.

Simple search: This search enables users to search for the query term(s) in all metadata fields.

Advanced search: This search enables users to search in one or more metadata fields. Those fields include title, creator, date, publisher and keyword.

The search result(s) are shown ranked by relevancy in the interface with title, date and abstract of the articles in a descending relevant order. Each title is linked to the entire content of the article. In the initial stage, the entire articles are stored locally. In later versions, SOAP request and response will be used to retrieve the article content from the local archive.

Besides the functionalities above, in WP3, the following functions have been added to the prototype.

Subject view: Users can navigate through this three-level hierarchical subject view based on IPTC subject code and also can see the titles of articles belong to each subject in the main frame.

Receive information: The prototype will have to request and receive metadata from external resources using SOAP.

The prototype has a web-based interface allowing users to do queries and navigate throw the metadata layer. All the metadata is maintained in a local (native RDF) database.

All queries or navigation actions by users are handled by the RDF database, so locally in the central system. “Simple search” actions are translated to a search in the top 20 keywords of all news articles. These keywords are stored in the metadatabase. They are obtained using the AKE system (see below).

Dublin Core

The Dublin Core Glossary defines metadata as “information that expresses the intellectual content, intellectual property and/or instantiation characteristics of an information resource” (Woodley, 2001). In the context of this document, a metadata application (MA) is considered to be an application that handles metadata, regardless of its goal.

Often metadata applications concentrate on either searching or browsing. When the application is designed for searching, catalogues of information resources are built and are then searched or indexed for searching. Browsing across these catalogues is often achieved through the explicit and implicit relationships between them (for example, relationships such as references, multiple-versioning, and so on). One of the basic motivations for building these catalogues is to facilitate resource discovery, making this process more efficient and effective across the Web. Here we call it the “Low-Level Approach” (LLA).

On the other hand, when the application is built primarily for browsing, a network (or web) of concepts is built, based on some kind of knowledge organization (ontology). In this approach, the main motivation is the desire to be able to browse through a network of concepts that link to resources, with these links having specific meanings. Here we call it the “High-Level Approach” (HLA).

Each of these approaches typically uses a different set of technologies that best reflects its philosophy. The Low-Level Approach many times uses plain Extensible Markup Language (XML) (http://www.w3.org/XML/) or the Resource Description Framework (RDF) (W3C, 1999). The High-Level Approach typically uses Topic Maps (TM) and its XML application, XML Topic Maps, or XTM (TopicMaps.Org Authoring Group, 2001), or RDF Schema (RDF-S) (W3C, 2000), or other ontology languages based on RDF-S, such as DAML+OIL (DARPA, ; Horrocks et al., 2001) and OWL (W3C, 2003).

Metadata applications usually handle one of these approaches, but it is not very common that they handle both:

– In most cases where an RDF application handles the LLA, some of the metadata elements refer to words or codes from existing controlled vocabularies, but they are only used for searching the catalogues. Usually they do not provide the functionality for users to navigate through those vocabularies and find information the other way around.

– Similarly, in most cases where an RDF application handles the HLA, while its records may point to real resources, it does not have searchable catalogue information about those resources. These applications are most often targeted to high conceptual navigation or browsing, not to providing search functionality.

There are some applications that handle both LLA and HLA approaches (for instance, MPress: http://mathnet.preprints.org). The OmniPaper RDF prototype goes one step further by implementing not only a subject thesaurus but also a lexical thesaurus on the HLA; these are directly connected with the articles' descriptions in the RDF metadatabase. These new developments allow the addition of important functionality, including manual and automatic query expansion.

The development of the OmniPaper RDF prototype comprised the following general steps:

1. Definition and development of the metadatabase: the LLA

2. Definition and development of the conceptual layer (subject + lexical thesaurus): the HLA

3. Integration of previously developed prototypes into one full prototype

Why using RDF

Metadata may be encoded using several different technologies such as relational databases, HTML, plain XML or RDF. Relational databases may be used together with WSDL to make information available to the outside world. HTML and plain XML may also be used and there are recommended guidelines from the Dublin Core Metadata Initiative (DCMI) to encode metadata using these technologies. However, the Resource Description Framework (RDF) has been chosen due, mainly, to the following reasons:

a) It is a standard specifically for encoding metadata – in fact, a World Wide Web Consortium (W3C) recommendation since 1999;

b) It is rich in expressing semantics

Being a standard provides the basis for interoperability with other applications that handle metadata, either suppliers or users (customers).

Richness in expressing semantics adds value to metadata applications once it adds meaning not only to resources' descriptions but, more important, to resources' relationships.

Additionally other reasons for having chosen RDF may be pointed out:

- The RDF is an infrastructure that enables the encoding, exchange and reuse of structured metadata;
- The RDF infrastructure enables metadata interoperability;
- RDF uses XML (eXtensible Markup Language) as a common syntax for the exchange and processing of metadata;
- RDF/XML is an XML application that contains methods of expressing semantics, enables consistent encoding, exchange, and machine-processing of standardized metadata;
- RDF supports the use of conventions that facilitate modular interoperability among separate metadata element sets;
- RDF provides a means for publishing both human-readable and machine-processable vocabularies;

The W3C Semantic Web Activity (SWA) was born inside the RDF community developments and is well rooted in it. This means all RDF developments are well contextualized in the SWA.

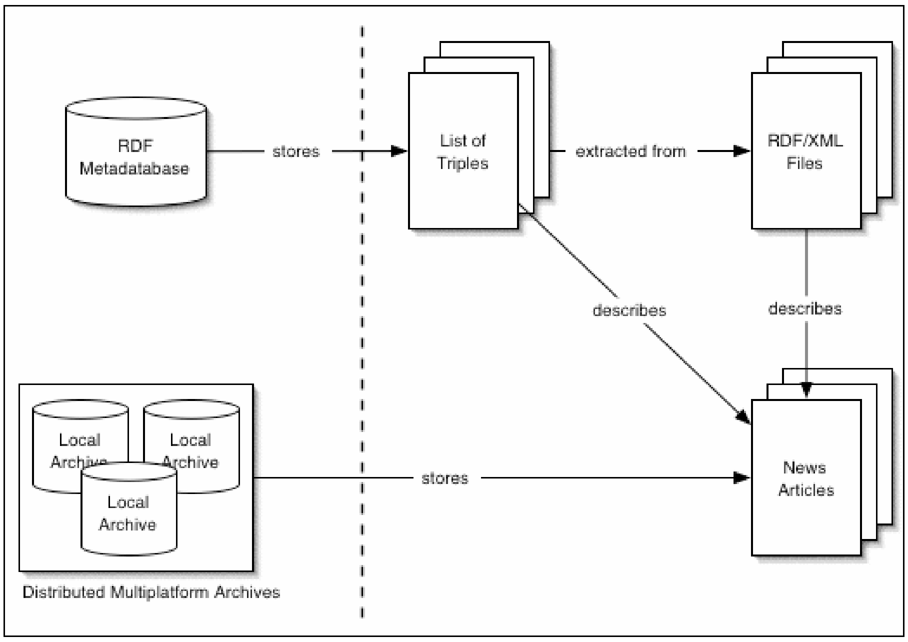
“The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF), which integrates a variety of applications using XML for syntax and URIs for naming”. [SWACT]

The Low Level Approach on the RDF prototype

After having defined the metadata application profile (see section B.2.1), we established rules for metadata encoding and built an RDF/XML template. We took into account all the recommendations made by Kokkelink and Schwänzl (Kokkelink & Schwänzl, 2001), although this document was still a proposed recommendation (DCMI) from the DCMI.

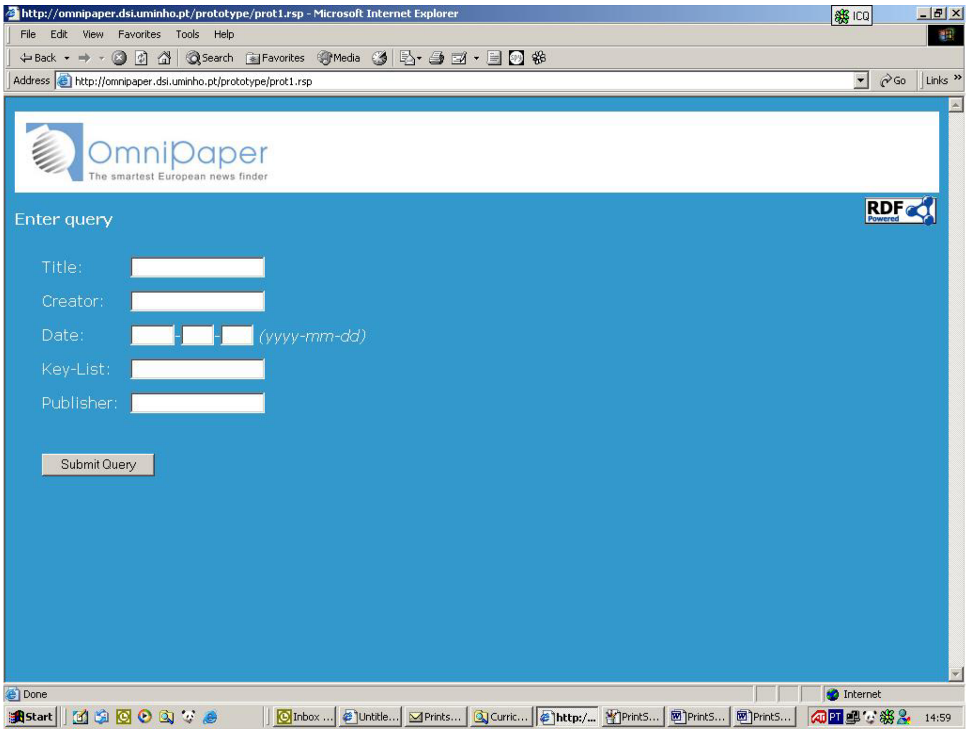
After proper validation, these RDF/XML files are uploaded to the metadatabase and converted to RDF triples. As the metadatabase platform, we chose to use RDF Gateway®, a Microsoft® Windows™–based native RDF database management system combined with a HTTP server. Some RDF Server Pages (RSPs) were created in order to provide some functionality for the end user. The overall architecture for this layer can be seen in Figure 1.

Figure 5 Overall architecture for the Local Knowledge Layer



As this prototype was meant to implement the LLA, only searching functionality was implemented. Functionality for navigating documents' metadata could have been implemented (by making use of relational metadata elements that hold universal resource identifiers to other resources), but it was not, merely because this was not a requirement for this specific prototype. Figure 6 shows a print screen of the user interface built for the first OmniPaper RDF prototype.

Figure 6 Print screen of the first RDF prototype for the Local Knowledge Layer


http://omnipaper.dsi.uminho.pt/prototype/prot1.rsp - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://omnipaper.dsi.uminho.pt/prototype/prot1.rsp
OmniPaper
The smartest European news finder
Enter query
RDF Powered
Title:
Creator:
Date: (yyyy-mm-dd)
Key-List:
Publisher:
Submit Query
Done
Internet
Start
14:59

The High Level Approach on the RDF prototype

Once that the news providers' partners used different vocabularies for indexing and categorization of news, the solution agreed upon was to use the International Press Telecommunications Council Subject Reference System (IPTC SRS) (IPTC, 2003).

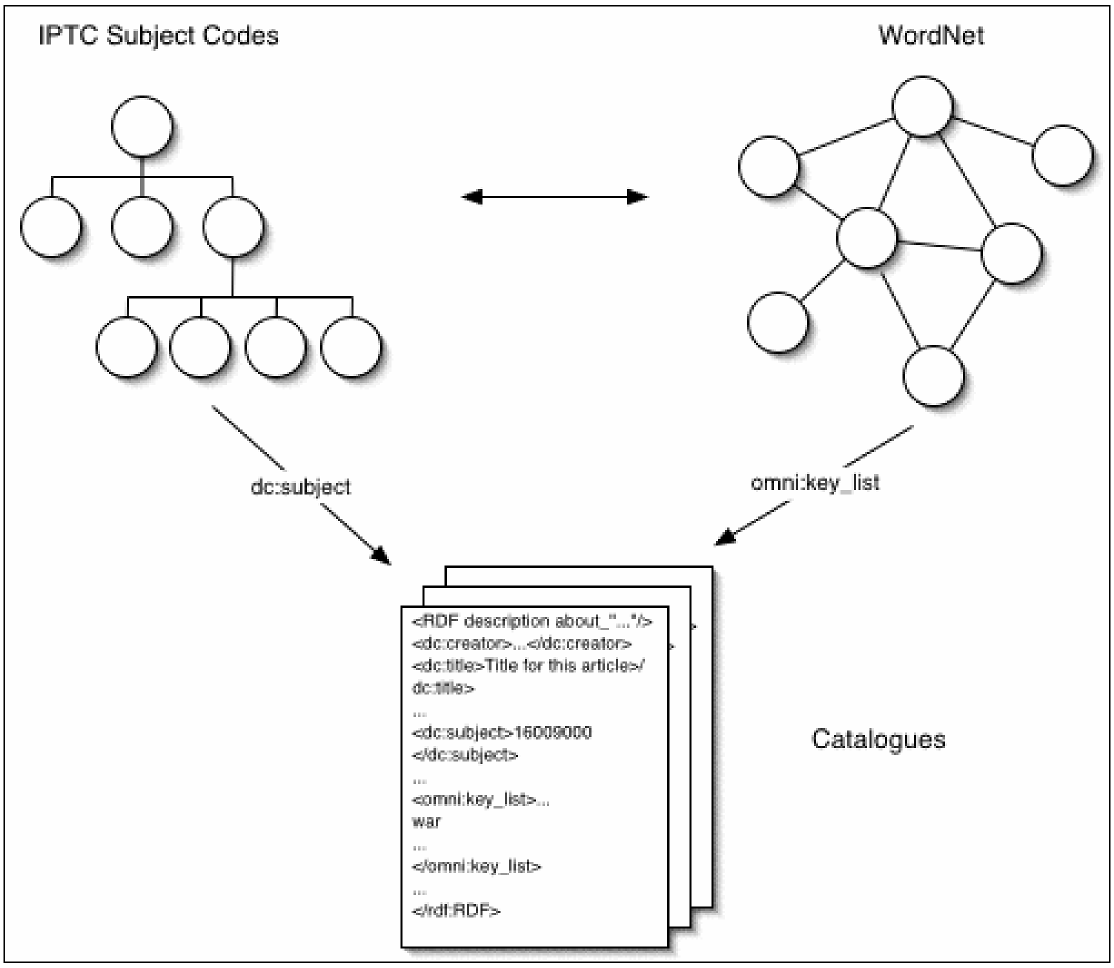
Several RDF-based vocabulary and ontology description languages were studied in order to choose one to codify the IPTC SRS (Pereira & Baptista, 2004). In the end, RDF Schema (W3C, 2000), an RDF vocabulary-description language, was chosen, mainly due to the fact that the IPTC SRS vocabulary is so simple that a more powerful (and complicated) language would be useless and inappropriate.

The connection between the Overall Knowledge Layer and the Local Knowledge Layer is made through the "dc:subject" metadata element as shown in Figure 7; that is, only values from the IPTC SRS can be used in the dc:subject metadata element for each description stored in the metadatabase. Conversely, dc:subject-based indexes may be stored together with each concept (value) of the IPTC SRS. For performance reasons, this may turn out to be an important feature in later phases that use a much larger metadatabase.

To add value to the Overall Knowledge Layer navigation and searching functionality, another empowering information-organization tool was included and linked to the metadatabase: WordNet®. WordNet is "an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets" (Princeton University Cognitive Science Laboratory).

WordNet version 1.6 was downloaded and included in a local metadatabase, and its connection to the articles metadatabase was made though the "omni:key_list" metadata element as shown in Figure 7. This is a very natural feature that can be used to perform query expansion, whether it is done automatically or manually.

Figure 7 OmniPaper metadata semantic layer

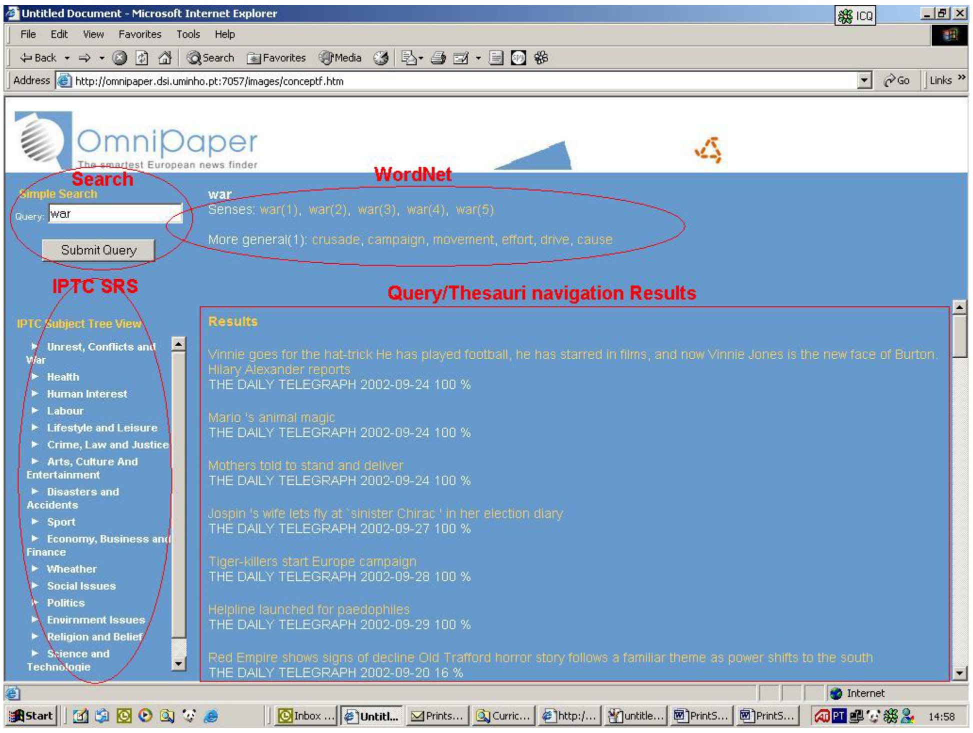

IPTC Subject Codes
WordNet
dc:subject
omni:key_list
<RDF description about_"..."/>
<dc:creator>...</dc:creator>
<dc:title>Title for this article</dc:title>
...
<dc:subject>16009000</dc:subject>
...
<omni:key_list>...
war
...
</omni:key_list>
...
</rdf:RDF>
Catalogues

No direct relationship between WordNet and the IPTC SRS has been implemented. The only relationship is that when a search for a concept is performed in the IPTC SRS, the same search is performed in WordNet for that particular word. It would be of major interest to implement word clustering into IPTC SRS concepts, but it was beyond the scope of this prototype.

➢ It would be of major interest to implement word clustering into IPTC SRS concepts, but it was beyond the scope of this prototype.

More functionality, particularly navigation functionality, was added to this prototype. This functionality implements the HLA by defining IPTC SRS concepts as a type of dc:subject metadata-element content. This allows users to navigate through the IPTC SRS tree (the left side of the screen on Figure 7) to find articles that have been indexed according to the subjects listed in the tree. Furthermore, because WordNet is also part of the system, when a user navigates the IPTC SRS, related words (associated with the branches of the IPTC SRS) appear on the screen (see the top-middle section of Figure 6). The user can click on these words (synonyms, antonyms, et cetera) to access results not previously retrieved by the system. When the user clicks one of these words, the system simply expands the previous query to search for that word in the omni:key_list metadata element’s contents.

Figure 8 Print screen of the RDF prototype for the Local Knowledge Layer and the Overall Knowledge Layer


Untitled Document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://omnipaper.dsi.uminho.pt:7057/images/conceptf.htm
OmniPaper
The smartest European news finder
Search
Simple Search
Query: war
Submit Query
WordNet
IPTC SRS
IPTC Subject Tree View
Unrest, Conflicts and War
Health
Human Interest
Labour
Lifestyle and Leisure
Crime, Law and Justice
Arts, Culture And Entertainment
Disasters and Accidents
Sport
Economy, Business and Finance
Wheather
Social Issues
Politics
Envirnment Issues
Religion and Belief
Science and Technologie
Query/Thesauri navigation Results
Internet
Start
14:58

war
Senses: war(1), war(2), war(3), war(4), war(5)

More general(1): crusade, campaign, movement, effort, drive, cause

Results

Vinnie goes for the hat-trick He has played football, he has starred in films, and now Vinnie Jones is the new face of Burton. Hilary Alexander reports
THE DAILY TELEGRAPH 2002-09-24 100 %

Mario 's animal magic
THE DAILY TELEGRAPH 2002-09-24 100 %

Mothers told to stand and deliver
THE DAILY TELEGRAPH 2002-09-24 100 %

Jospin 's wife lets fly at `sinister Chirac ' in her election diary
THE DAILY TELEGRAPH 2002-09-27 100 %

Tiger-killers start Europe campaign
THE DAILY TELEGRAPH 2002-09-28 100 %

Helpline launched for paedophiles
THE DAILY TELEGRAPH 2002-09-29 100 %

Red Empire shows signs of decline Old Trafford horror story follows a familiar theme as power shifts to the south
THE DAILY TELEGRAPH 2002-09-20 16 %

RDF Metadatabase Design

Although the RDF metadatabase behaved well in what concerns timing, some improvements have still been made regarding the metadatabase design.

In the previous prototype, in the key-list metadata element many anonymous nodes are kept for the sake of adding meaning. In practice we make several statements for saying that a key-list is a sequence of keywords, each one having a specific value and a specific weight. By only taking out the sequence feature, 1 anonymous node per keyword is saved and another one per key-list. If we talk about 10.000 articles with 20 keywords each, this means 210.000 anonymous nodes are saved!

The following kind of code (see Figure 9):

```
<omni:key-list>

      <rdf:Seq>

            <rdf:li rdf:parseType="Resource">

                  <rdf:value> pounds </rdf:value>

                  <omni:key-weight>24.000000</omni:key-weight>

            </rdf:li>

            <rdf:li rdf:parseType="Resource">
```

```
                   <rdf:value> art </rdf:value>

                   <omni:key-weight>5.087460</omni:key-weight>

             </rdf:li>

       </rdf:Seq>

</omni:key-list>
```

has been be replaced by (see Figure 10):

```
<omni:key-list rdf:parseType="Resource">

      <rdf:value> pounds </rdf:value>

      <omni:key-weight>24.000000</omn:key-weight>

</omni:key-list>

<omni:key-list rdf:parseType="Resource">

      <rdf:value> art </rdf:value>

      <omni:key-weight>5.087460</omni:key-weight>

</omni:key-list>
```

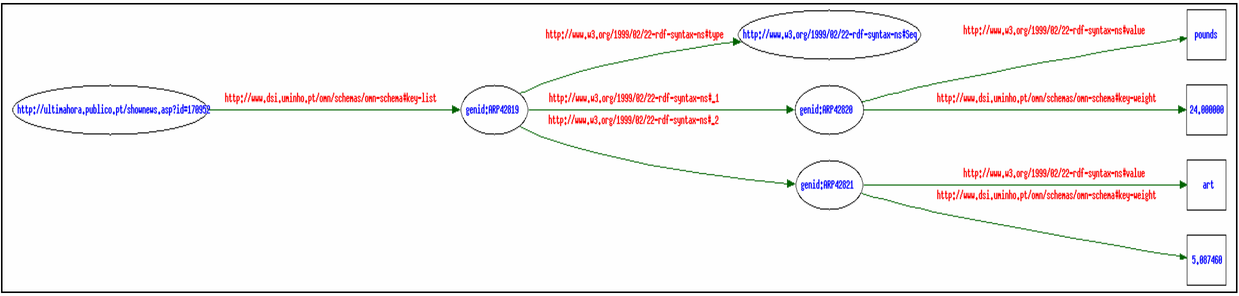

http://ultimahora.publico.pt/shownews.asp?id=170952
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-list
genid:ARP42819
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq
http://www.w3.org/1999/02/22-rdf-syntax-ns#_1
http://www.w3.org/1999/02/22-rdf-syntax-ns#_2
genid:ARP42820
genid:ARP42821
http://www.w3.org/1999/02/22-rdf-syntax-ns#value
pounds
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-weight
24,000000
http://www.w3.org/1999/02/22-rdf-syntax-ns#value
art
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-weight
5,087460

Figure 9 - Graph that corresponds to the first block of code

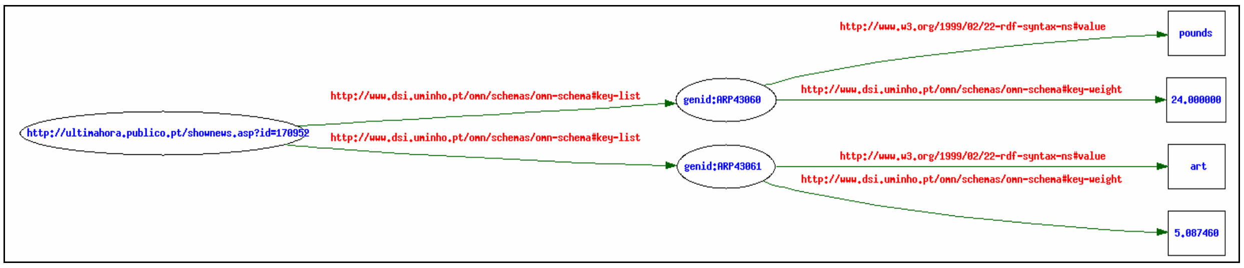

http://ultimahora.publico.pt/shownews.asp?id=170952
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-list
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-list
genid:ARP43060
genid:ARP43061
http://www.w3.org/1999/02/22-rdf-syntax-ns#value
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-weight
http://www.w3.org/1999/02/22-rdf-syntax-ns#value
http://www.dsi.uminho.pt/omn/schemas/omn-schema#key-weight
pounds
24,000000
art
5,087460

Figure 10 - Graph that corresponds to the second block of code

These kinds of savings have been done also for other metadata elements. In fact, it is a compromise between meaning and efficiency.

B.2.2.3 XTM

The purpose of this prototype is to analyze and test how Topic Maps can be used in the local knowledge layer approach. The Topic Map will be used to enhance intelligence search for news articles by creating a semantic web. The web contains links between the keywords that are extracted from the news articles. The results of this approach have been compared to the RDF approach and conclusions have been drawn for the development of the local knowledge layer prototype.

The prototype has four main functions to the end user:

1. Simple search: this allows the users to search for within keywords for identifying concepts. The articles related to the concepts are then shown. The user has the possibility to refine his search and the system should guide the user in doing this.
2. Advanced search: this allows users to search for keywords, but with the possibility to apply extra constraints on the metadata.
3. Relational concept view: this allows users to browse freely through the topic map.
4. Hierarchical subject view: this allows users to browse through news categories, by only showing the subject reference topics of the Topic Map.

In all these cases, the result is that the user will receive an overview of articles that are covered by the selected topics (by showing a part of the article's metadata, e.g. title, date,…). The full text article can then be retrieved via a SOAP request to a news archive server. This functionality is already designed in the SOAP Prototype and has been taken over for this prototype.

The prototype has a web-based user interface allowing users to do queries, to interact with the system to refine queries and to browse through the Topic Map and the hierarchical subject view. All necessary information (topic map and metadata) is maintained in a database at the prototype's location. The full text articles are provided by a remote news archive server.

All queries or navigation actions by users are answered by the central system by searching for matching keywords and concepts in the Topic Map. This map contains a network of keywords, concepts and semantic relations between concepts, all derived from the WordNet linguistic database. This means that the WordNet database has been fully converted to XTM format.

The system is able to make links between the Topic Map keywords and news articles thanks to the Automatic Keyword Extraction system (see B.2.2.4 AKE). This system extracts keywords from news articles and appoints a weight for each keyword. The higher the weight the more important that keyword is for this article.

The Topic Map (XTM) prototype uses these weights for calculating how relevant a certain news article is for a specific search or navigation action.

The XTM prototype only uses SOAP for retrieving the full text of news articles.

B.2.2.4 AKE (Automatic Keyword Extraction)

This prototype has two main goals:

→ Perform automatic keyword extraction on news articles so that these keywords can be used by other prototypes (like XTM and RDF)

→ Perform AKE search, a search prototype that uses the Vector Space Model for retrieving relevant news for a given query.

Since the XTM and RDF prototypes use the WordNet ontology (stored as a Topic Map) for smart search and navigation, the news articles need to have some kind of link to this ontology. The Automatic Keyword Extraction Process (AKE process) provides this link. It obtains the relevant keywords of multilingual news documents and indexes them in a database. This process requires enhancing the traditional indexing of documents in commercial database manager systems through the use of modern information retrieval techniques.

In information retrieval task, what the user really wants is to retrieve documents that are about certain topics and these topics are described by a set of keywords. The main objective of AKE is improving the quality of retrieved information by obtaining all (and only) adequately ranked documents concerning the user query. The problem to solve is to extract from documents those terms (words or phrases) that are significant and to eliminated those terms that are not

Currently, one of the main research topics focused on improving information retrieval technology is related to the characterization of documents and how it affects the information retrieval process; concerning this topic there are three main trends:

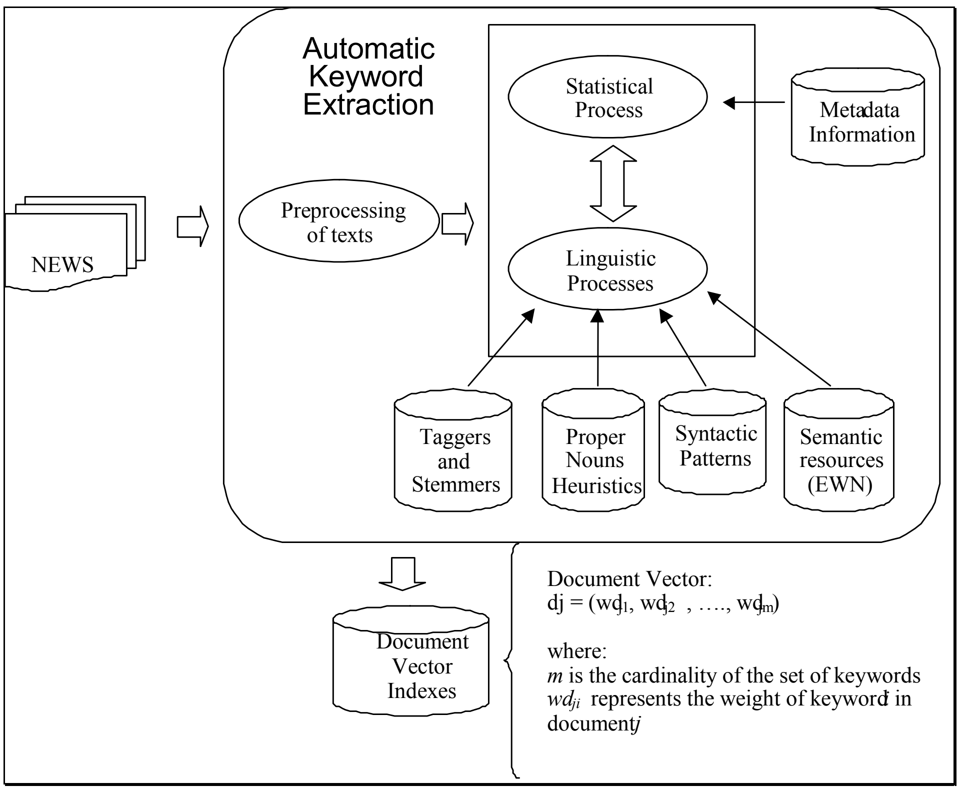
a) Semantic approaches that try to implement some degree of syntactic and semantic analysis of queries and documents; this involves reproducing in a certain way the understanding of the natural language text.

b) Statistical approaches that retrieve and rank documents according to the match of documents-query in terms of some statistical measure.

c) Mixed approaches that combine both of them trying to complement the statistical approach with semantic approaches by integrating natural language processing (NLP) techniques, in order to enhance the representation of queries and documents and, consequently, to produce adequate levels of recall and precision.

In the OmniPaper project the third approach is adopted: first, the statistical approach is considered, and then linguistic techniques will complement statistical framework through some kind of syntactic and semantic processing performed on the news and user queries, but in a shallow way (not for understanding the text). This approach requires having available linguistic resources for every language involved in the project, that is, linguistic approaches are language and domain dependent.

Figure 11: Overview of AKE process



Statistical frameworks break documents and queries into terms; these terms represent the population that is counted and measured statistically. Generally speaking, the set of terms that describe a document is composed of all the words (or phrases) of the document except stop words; optionally, these significant words could be stemmed. Moreover, not every word is used for indexing a document: usually, a filtering method is performed in order to select the most adequate, that configure the keywords of a document.

B.2.2.5 AKE workbench

The workbench is a tool within the OmniPaper system, that allows to manually verify automatically extracted information - where required - to improve the quality of the automatic extraction processes as well as the overall information retrieval quality.

The workbench is designed to be used in the development phase of the system. It is a tool that allows the users to compare articles or other “source material” and the corresponding information that was automatically extracted by the system. The user could and should specify the correctness of the extracted information as well as change the extracted values (respectively information) if required. Additional information that should have been extracted can be added or wrong extracted information deleted.

The taken modifications contain implicitly the knowledge of the user about the semantic structure of the „field“. This implicit information (i.e. the modifications) is captured by the system and then could and should be used to improve the automatic extraction processes.

To provide a better understanding of the required functionality and the integration of the workbench in the whole system the relevant automated processes are characterised shortly.

Keyword extraction

Basis for this process is a new news article. In the first step of the extraction process the stop words are eliminated and the remaining words are normalised. After this step is finished a list of (normalised) keywords exists. This list also includes the frequency of occurrence of the word within the article. Additionally it is thought about not only extracting keywords but also „key phrases“. For example, September 11th is only in the combination of number and month of a special meaning.

Vector extraction

In the next step an article vector is automatically extracted/calculated. By now it isn’t definitely determined how this calculation is processed. Anyhow, the process should result in a definite vector that represent the article as proper as possible. This vector will consist of keywords (and possibly key phrases) and corresponding (automatically calculated) relevance values.

Vector clustering

The article vectors can be mathematically clustered. It is expected, that the mathematically clustered articles correspond to „real-live“ topics/subjects/concepts.

Cluster – concept matching

The concept vector then can be matched to a topic map concept. It isn’t clear yet how this process will take place, especially if automation is possible and how this can be implemented.

Automatic link extraction

In WP3 automatic link extraction is researched. The goal is to automatically identify links between the articles to improve cross archive navigation. For example, an article from Source A is closely related to an article from source B. So it would provide additional value to link the to articles. This linking could be done by adding links at the end of the article (something like „read also about...“) or proper words within the article could be implemented as links.

Automatic classification of articles

The articles in the local archives could be assigned to OmniPaper system categories by an automated classification process. In most cases this process would be easy to implement and probably will work with high accuracy. If the same category exists in the local archives as well as in the OmniPaper system the classification can be taken over without modification. Anyhow, this can’t be expected to be always the case. The more difficult situation is when the local and the OmniPaper archives use different category sets. In this cases the classification needs to be adapted, which should take place automatically. The need for manual verification might emerge as well.

Extraction of keywords and vector extraction are the two processes that define the core functionality of the workbench.

B.2.2.6 Automatic Test Engine

The purpose of this system is to automate the cross-testing of prototypes.

This has several advantages:

- Testing is faster. Since the prototypes were tested by a predefined set of 20-25 topics, each topic having 3-5 queries, trying out all these queries by hand and writing down the results is too much work.

- Therefore, it is easier to adapt the prototypes according to the test results and re-testing the new prototype versions.
- Reporting is easier, faster and more uniform.

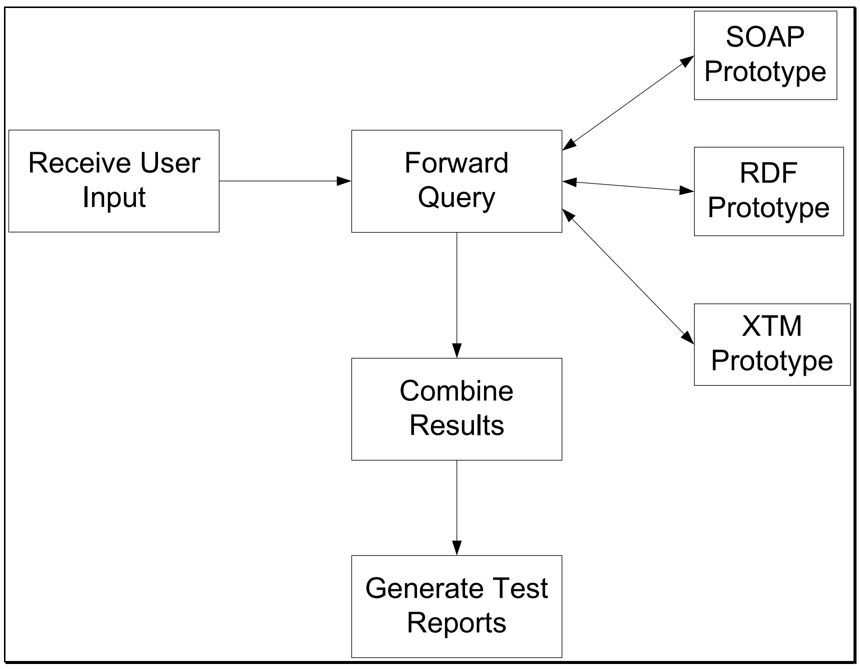
The testing engine has the following functions:

- Receive a number of test topics and queries from the user
- Submit these queries to the different WP2 prototypes
- Gather the results from these prototypes
- Generate reports from these results, both in textual and in graphical form
- Return these reports to the user

System diagram

After taking input from the user, the Auto Testing Engine forwards this input in a certain order to the three prototypes listed above. These prototypes return their results back to the Auto Testing Engine, which puts the results together and outputs it to the user in the form of test reports.

Figure 12: Automatic Testing Engine System Diagram



B.2.2.7 WP2 combined prototype: Distributed Information Retrieval Prototype

System Requirements Analysis and Specifications

The goal of this prototype is to provide a smart search and navigation layer on top of one news archive. This "Local Knowledge Layer" allows semantic-based search, navigation and filtering of English newspaper articles (from The Daily Telegraph).

This prototype is the end result of the WP2 work in the prototypes using SOAP, XTM, RDF and automatic keyword extraction. The goal of the prototype is to enhance the user experience in finding online news of interest. The prototype obtains its data from online news sources managed by a number of news providers and tries to build an intelligent top layer on the data that consists of metadata and an intelligent search interface. In order to achieve this goal the available metadata must be stored in a structured way and a number of dedicated query and navigation mechanisms to access this data must be designed.

In this prototype, querying and navigation are considered as alternative methods to find relevant information. Both interact with each other and together they produce a combined user experience that can be expressed as "find what you were looking for and then browse away from it". In fact, the prototype considers both querying and navigation as a kind of search action. The only difference is that in navigation the user follows predefined paths, whereas in querying the user is totally free in what he or she submits as a query. Querying is a way of searching that provides the user with a starting point in the vast amount of available information.

This prototype implements four kinds of query and navigation. A first method allows users to navigate through news subjects (categories) in a traditional, hierarchical way. A more sophisticated tool is the relational navigation, where users can browse through a "web of concepts". The starting point for relational navigation (the "focus concept" in OmniPaper terminology) is the result of the last navigation or query action and the predefined paths that can be followed are paths to concepts that are related to the focus concept in the knowledge map. Finally smart querying is enabled using a "knowledge map" of semantically related keywords and concepts.

The idea is that the exact words of a user query are just a starting point for the search engine. Once the query is analyzed and basic search terms are extracted, they can be applied to the knowledge map, where they can be expanded to other related keywords and corresponding news articles (either semantically widened or narrowed).

Two kinds of smart queries exist: simple query and advanced query, where advanced query is to be understood as a kind of filtering to restrict the number of results. When a query term is entered in combination with advanced query options, only the results that satisfy the advanced query constraints will be shown.

This prototype will be used by the OmniPaper consortium and developers for the cross-testing of all the prototypes in WP2. It will also be used by a test group that will evaluate the different prototypes in the later stage. Further it can be used by the wider public as a demonstration of the OmniPaper work.

Functional specification

The prototype performs the following functions for its users:

→ Article retrieval: show the full text of the news article

→ **Smart search** in news articles:
  - Simple search: only one search box
  - Advanced search: search box and metadata fields for limiting the search
  - Related functions:

        - ranking of search results
        - stemming of keywords

→ Automatic keyword extraction from articles

→ Navigation:

  - Hierarchical subject view: show predefined news subjects in a hierarchical way (like a tree of news subjects)
  - Relational concept view (web of concepts): show concepts relevant to the current query

→ Multilingual user interface

The primary functions requested by users are smart search and navigation.

The following functions are NOT handled by this prototype but will be part of the WP3 prototype:

- Automatic keyword extraction from queries (allowing natural language queries)
- Multilingual search and navigation
- Multi-archive search and navigation
- Dynamic update of knowledge layer with new articles
- Cross-archive linking

External interfaces

(See Figure 13 for an illustration)

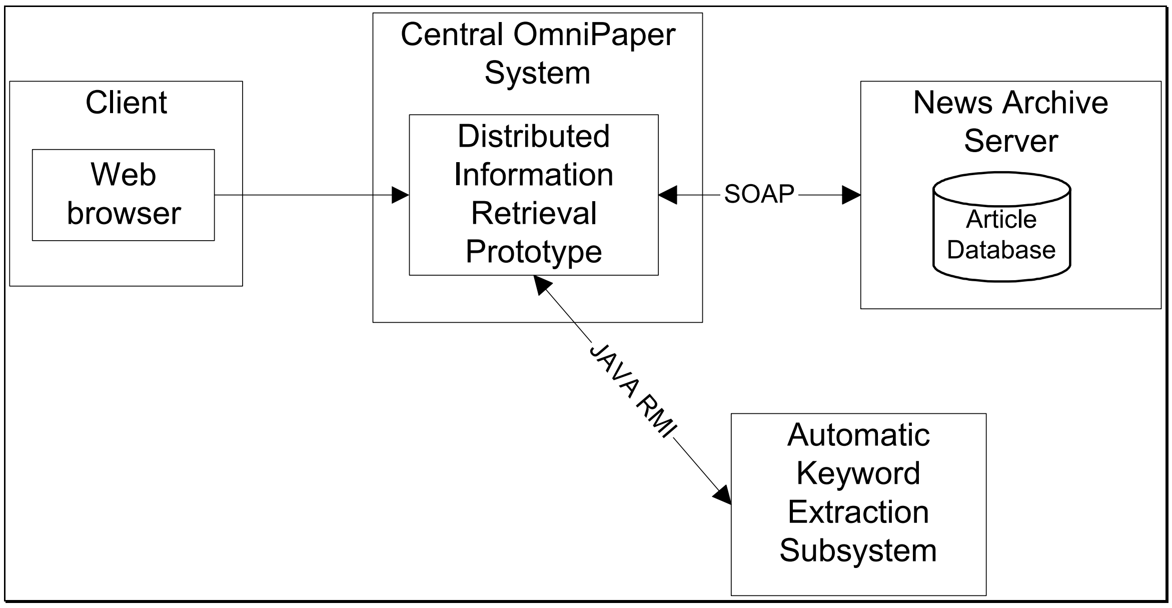
The prototype has a web-based user interface allowing users to do queries, to interact with the system to refine queries and to browse through the Topic Map and the hierarchical subject view. All necessary information (topic map and metadata) is maintained in databases at the prototype's location. The full text articles are provided by a news archive server. Through the HTTP/SOAP communication protocol, the system works together with the SOAP server at the (remote) news archive site.

For performing Automatic Keyword Extraction the central OmniPaper system communicates with the AKE server (subsystem). The AKE server provides a Java interface used for integration purposes with the central OmniPaper system. This server provides an external interface called *AKEServer*, where necessary methods to index an article and obtain related keywords are supplied. The interface implementation is included in a class called *AKEServerImpl*, which is also in charge of extracting article metadata to be stored in the AKE database and applied in the keyword extraction process.

Context diagram

The prototype operates within a web server. When a user sends a request using his web browser, the web server activates the prototype, which processes the query and returns the result to the client. If desired by the user full text articles are retrieved from the My News server by means of a SOAP request (see Figure 13: context diagram).

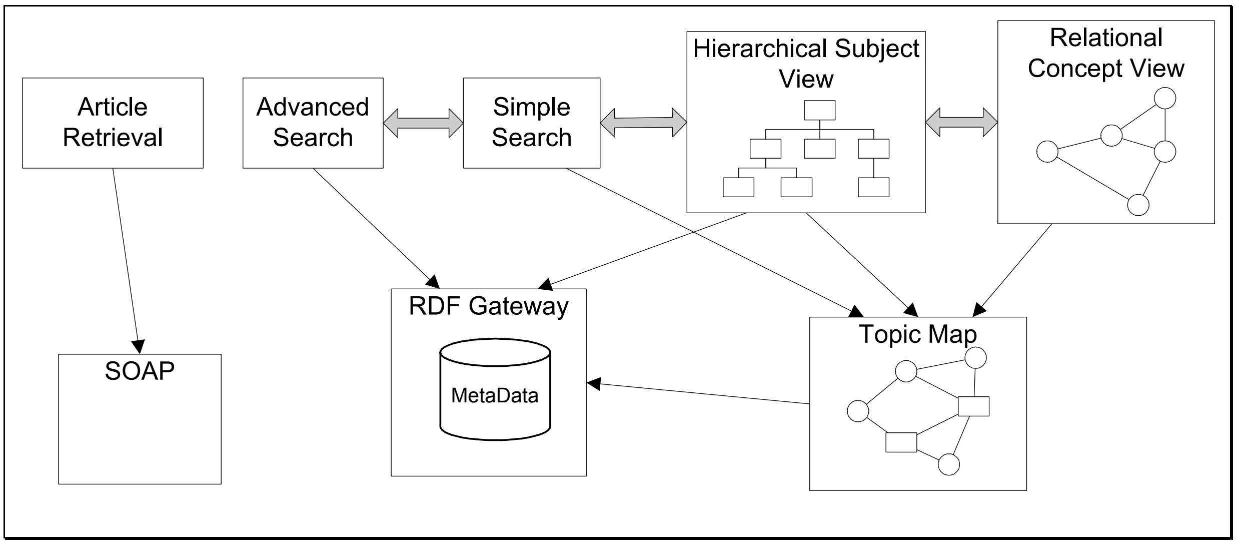
Figure 13: context diagram



System diagram

The functional specifications can be summarised into five main system functions: article retrieval, advanced search, simple search, hierarchical subject view and relational concept view. Figure 24 depicts how these functions relate to the different parts of the prototype (SOAP, RDF and Topic Maps).

Figure 14: system diagram



Functional Analysis

In this section all system functions are described in detail.

Article retrieval

This functions retrieves a news article from the My News database (in Barcelona) using SOAP and displays this article in HTML format.

Search

This function allows users to search for news articles in the predefined data set. The idea behind the prototypes is:

- Querying is used to narrow the search, in such a way that not too much results are shown and that the relevance ranking makes sure that the most relevant results are really shown on top.
- Navigation is used for widening the search into more general concepts, for re-focusing the query into other areas or for filtering/narrowing the search results to more specific areas.

Simple search

In the simple search users can type a query in a single input field and execute the query using the enter key or the "GO" button.

The following fallback scenario is used when a user types in a query:

1. "Smart search":
   Keywords are looked up in the WordNet. From a keyword, related keywords are searched using their common concept. From the common concept on, all concepts that are more specific up to a certain depth are looked. All articles that are attached to the keywords of the common concept and those of the specific concepts are returned.
2. (If no results)
   "Metadata search":
   keywords are looked up in Key_list metadata field using metadata search. All articles that have these keywords in their Key_list metadata field are returned.
3. (If no results)
   "Full-text search": keywords are looked up in full article text using full text search. All articles that have these keywords in their full text are returned.

This fallback scenario is performed in the background.

Advanced search (metadata query input fields):

With this function users can perform more sophisticated queries using one or more metadata fields to limit their query. Possible metadata fields: Title, Subject, Key-List, Abstract, Publisher, Creator, Issued-From, Issued-To. Two last fields are searching on the metadata field Issued. Fields are combined by default using the AND operator.

For the advanced search, three possibilities exist:

1. The user only types a query in one or more of the metadata fields: a metadata search is performed.
2. The user types a query in the "Search text" field and also in one or more of the metadata fields: a simple search is performed and the search results are filtered using metadata search.
3. The user types a query in the "Search text" field and nothing in the metadata fields: a simple search is performed.

Search results

This function displays a list of search results. For each article in the result list the following metadata fields are displayed: Title, Issued and Publisher. Results are ranked by relevance with the most relevant result on top of the result list. On top of the result list, the following information is displayed: the number of search results, the search time. Search results can be ordered by Title, Issued, Publisher or Relevance.

Query processing

This function processes the query that the user has typed into a query that can be understood by the search engine. A “correct” query consists of a number of keywords that can be combined using the following Boolean operators: AND, OR. Keywords and operators are separated using a space. Keywords separated with only a space are combined using the OR operator. The “ sign can be used to indicate exact query strings or keywords that consist of multiple words. Brackets can be used to logically separate parts of a query. The inclusion (+) or exclusion (-) signs can be used to indicate if a certain keyword must or may not appear in the resulting articles.

When a query is processed a query syntax tree is built. Results are combined using the extended Boolean model.

Display of query

The current query, as it was parsed by the system, is displayed. This way users can keep up with query changes if they are refocusing their query.

Word stemming

Keywords that appear in a simple search query are stemmed using the Porter algorithm. The Topic Map contains stemmed keywords (which were stemmed with the same algorithm). Metadata searches also use this stemming algorithm. Word stemming should be enabled by default.

Combination of search methods

In order to provide better search results, a combination of “smart search”, “metadata search” and “full-text search” might be useful. This will be considered after evaluation

Search options for users

When submitting a search, users can indicate if they want to perform a “smart” or a “full-text” search. Users can also indicate if their query needs to be processed in a case sensitive manner or not. The default values are to perform smart search and to use case insensitive queries. When smart search is disabled, a full-text search is performed in the background, but the web of concepts will still be shown (although not used to resolve the query, it can still be useful for navigation).

Automatic keyword extraction (AKE)

Keywords in the Key_list metadata field have been extracted automatically using data mining.

AKE on queries, allowing natural language queries, is part of WP3. Integration with this prototype is also WP3 work, since it is related to dynamic update of knowledge layer. When new articles arrive, keywords have to be extracted.

Navigation

Two ways of navigating between news concepts/subjects are supported:

- Hierarchical subject view
- Relational concept view

Both views are extracted from the same Topic Map containing semantically related concepts and keywords from the WordNet database.

Relational concept view: "Web of concepts'

This function gives the user a relational view on concepts relevant to the current query. The following functions are possible:

1. Refocus the current query:
    - Using semantically related concepts (more general, more specific, associated).
    - Using different word senses of the query keywords. This allows users to redefine/refine their query if one of the query keywords has different possible meanings (e.g. bank as a financial institution or as a piece of furniture).
    - Remove concepts from the query.
2. Navigate between different concepts relevant the query
3. For each concept, the following information is shown:
    - Semantic explanation of the concept
    - All keywords related to the concept
    - Related concepts according to different word senses of the query keyword.
    - Related concepts according to semantic relations from WordNet.

Hierarchical subject view:

In this function the Daily Telegraph subject reference is shown as a *browseable* tree of news subjects. Clicking on a subject will submit a metadata search on the Subject metadata field (for the Daily Telegraph, this is mapped to their field Section). If no results: metadata search also on Key_list metadata field.

The hierarchical subject view is obtained from the XTM database. Clicking a subject in this view invokes a metadata search on the Subject field.

(In the WP3 prototype the IPTC reference will be used. In that case a translation between IPTC subjects and sections used by the different newspapers will be necessary.)

Keyword extraction

See B.2.2.4 AKE (Automatic Keyword Extraction)

Multilingual user interface

All text appearing in the user interface is defined in an initialisation XML file, so that multiple languages are supported for the UI. This means that items in the user interface such as directions to the user and other words and phrases explaining the functioning of the prototype, are set in a configurable manner, so that they can be shown in multiple languages.

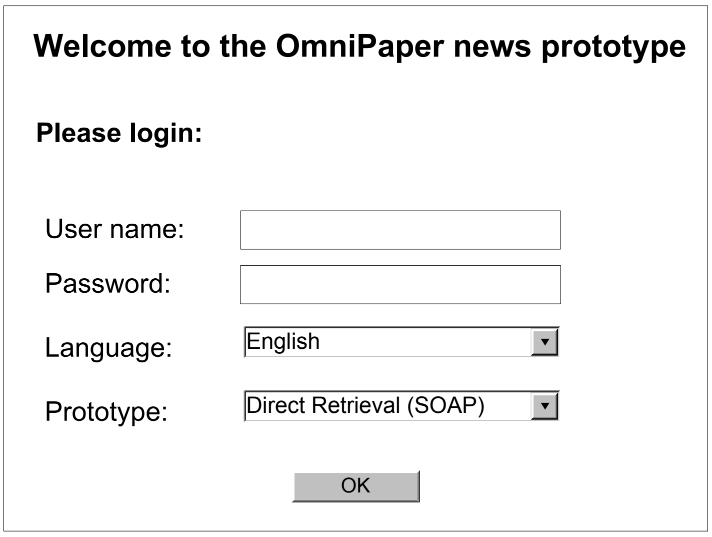
Note that queries and newspaper articles are still in English for this prototype. Multilingual querying will be provided by the WP3 prototype.

User Interface design

Login screen

Before the prototype can be used, the user must login using a user name and a password. At this stage, the user can choose which prototype he or she wants to use (SOAP, XTM, RDF, Automatic Test Engine or Final WP2 prototype) and which language he or she prefers for the interface of the prototype.

Figure 15: Login screen



Welcome to the OmniPaper news prototype

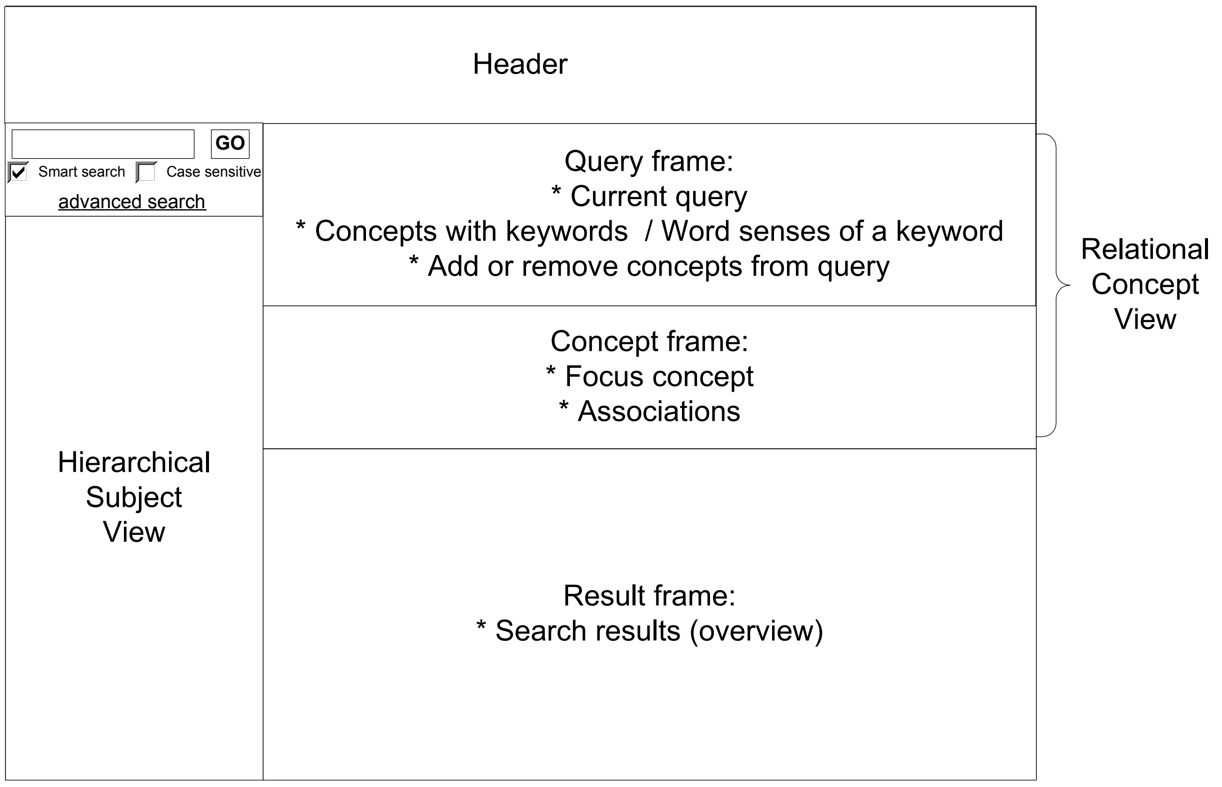
Please login:

User name:	
Password:	
Language:	English
Prototype:	Direct Retrieval (SOAP)

Next to the search frame, where simple and advanced searches can be submitted, there is a hierarchical subject view frame on the left side which allows browsing through news subjects. The relational concept view is created in a textual way using two parts: a concept frame and a query frame. The concept frame shows the focus concept and its related concepts; the query frame shows the query and the concepts found in this query and it allows redefining the current query. The result frame finally shows the search results.

Screen layout

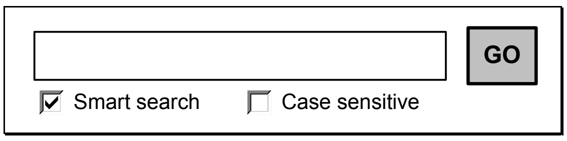
Figure 16: screen layout



Simple search

The simple search box is the same as in previous prototypes, but two options are added: smart search or not and case sensitive or not (see 0: Search options for users).

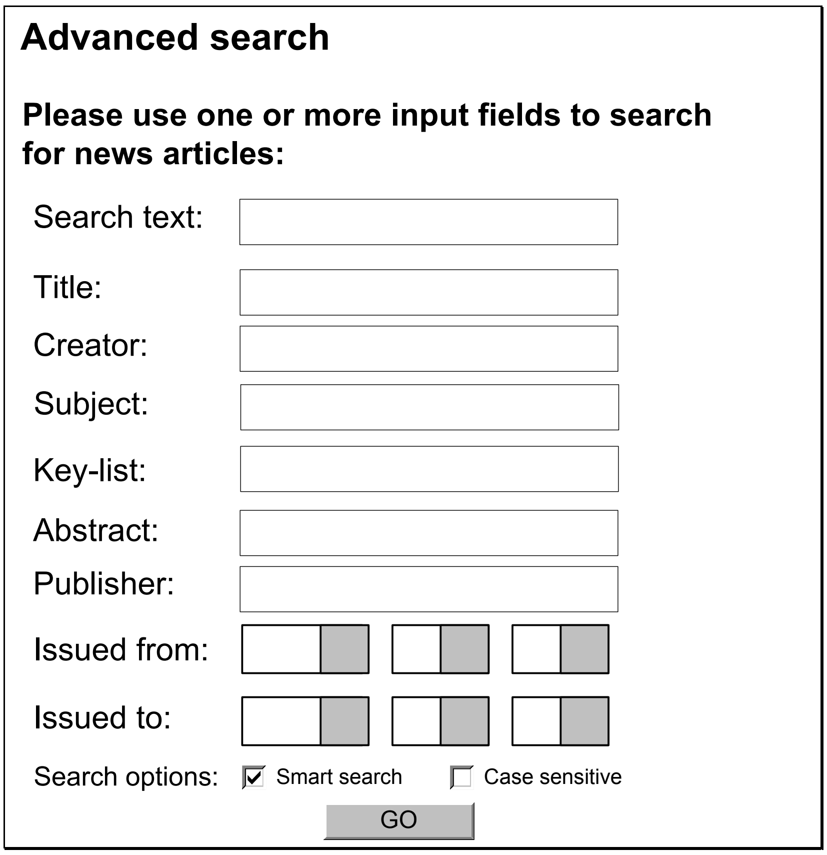
Figure 17: simple search user interface design



Advanced search

In the advanced search the simple search input field is extended with a number of additional input fields for limiting the search to certain metadata values. For the fields Issued from and Issued to, a drop down list of years, months and days is used in such a way that it is impossible to enter an incorrect date (e.g. 31 February 2003). In future versions this might be replaced with a real calendar, which is more user friendly.

Figure 18: advanced search user interface design

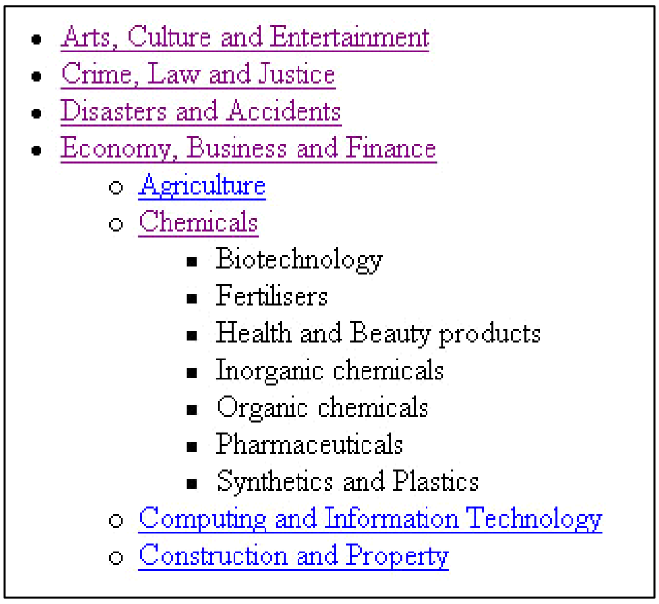


Advanced search

Please use one or more input fields to search for news articles:

Search text:	
Title:	
Creator:	
Subject:	
Key-list:	
Abstract:	
Publisher:	
Issued from:	
Issued to:	
Search options:	[✓ Smart search] Case sensitive

Figure 19: Hierarchical subject view



- Arts, Culture and Entertainment
- Crime, Law and Justice
- Disasters and Accidents
- Economy, Business and Finance
    - Agriculture
    - Chemicals
        - Biotechnology
        - Fertilisers
        - Health and Beauty products
        - Inorganic chemicals
        - Organic chemicals
        - Pharmaceuticals
        - Synthetics and Plastics
    - Computing and Information Technology
    - Construction and Property

Hierarchical subject view

The user can navigate through news categories using the hierarchical subject view. Figure 19 depicts an example subject view (based on the IPTC subject reference).

Relational concept view

This view consists of two frames: the query frame and the concept frame.

In the **query frame** the current query is shown as it was processed by the system. This means that the Boolean operators and brackets that might have been added by the query engine are shown. For example: the query phrase | belgian police force | will result in belgian OR police OR force.

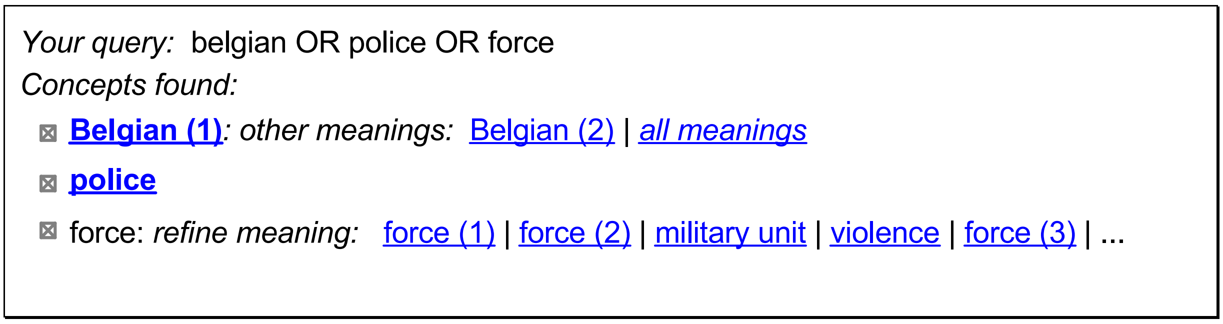
The different keywords found in the query are shown. Three possibilities exist:

1. No word sense ambiguity exists for the query keyword: the corresponding concept is shown in bold.
2. Word sense ambiguity exists for the query keyword but the preferred meaning of the word has been chosen (either by the user or by the word sense disambiguation system): the preferred concept is shown in bold. The other meanings of the keyword are also shown and can be clicked. If such a word is clicked, this meaning of the keyword is selected and made bold.
3. Word sense ambiguity exists for the query keyword and no preferred meaning of the word has been chosen: the keyword is shown. All possible meanings of the keyword are also shown and can be clicked. If such a word is clicked, this meaning of the keyword is selected and made bold.

In the latter case, the displayed search results will relate to all possible meanings of the keyword. Upon clicking of a certain meaning, the result set is limited to articles that are only related to the clicked meaning. The user can always go back to the bigger result set by clicking the link "all meanings" for that keyword.

For helping the user to refine the keyword into a certain word meaning, a semantic explanation of a certain word meaning is displayed in a tool tip upon mouse over.

Figure 20: query frame in relational concept view



Your query: belgian OR police OR force

Concepts found:

⊠ **Belgian (1)**: *other meanings:* Belgian (2) | *all meanings*

⊠ **police**

⊠ force: *refine meaning:* force (1) | force (2) | military unit | violence | force (3) | ...

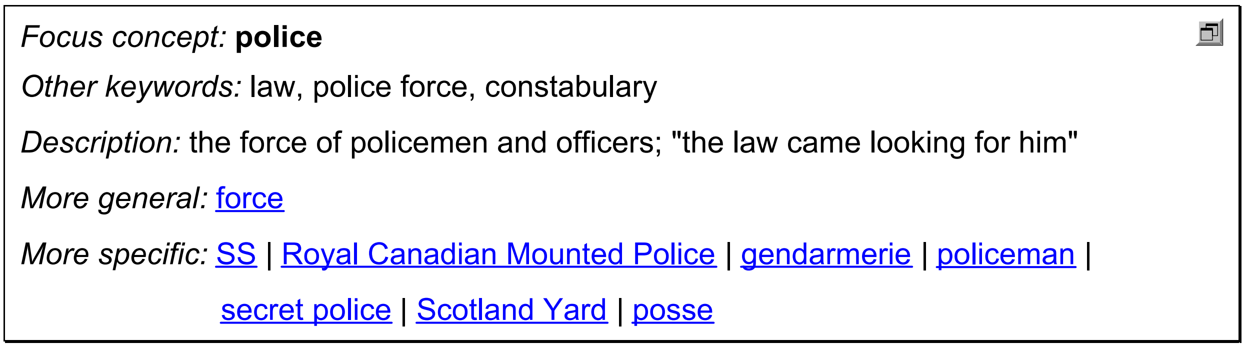
The X icon in front of each concept allows the user to remove that concept from the query. Clicking on a concept causes the concept frame to take this concept as the focus concept.

In the **concept frame** more information about the focus concept is shown. Next to the semantic explanation of the concept, the semantically related concepts are shown (relations "more general", "more specific" and "associated"). With these related concepts the user can broaden or narrow his or her query.

Two possibilities exist for displaying the concept frame. It can be displayed as a separate frame that refreshes when a user changes the focus concept in the query frame or it can be displayed as a tool tip

- like pop-up screen that appears upon mouse over of a concept in the query frame. The top right button allows switching between these two views. When this button is clicked the concept frame disappears, giving its space to the result frame. The concept frame is then displayed as a pop-up tool tip upon mouse over of a concept in the query frame. The query frame now gets the same top right icon that allows going back to the previous view.

Figure 21: concept frame in relational concept view



Focus concept: **police**

Other keywords: law, police force, constabulary

Description: the force of policemen and officers; "the law came looking for him"

More general: force

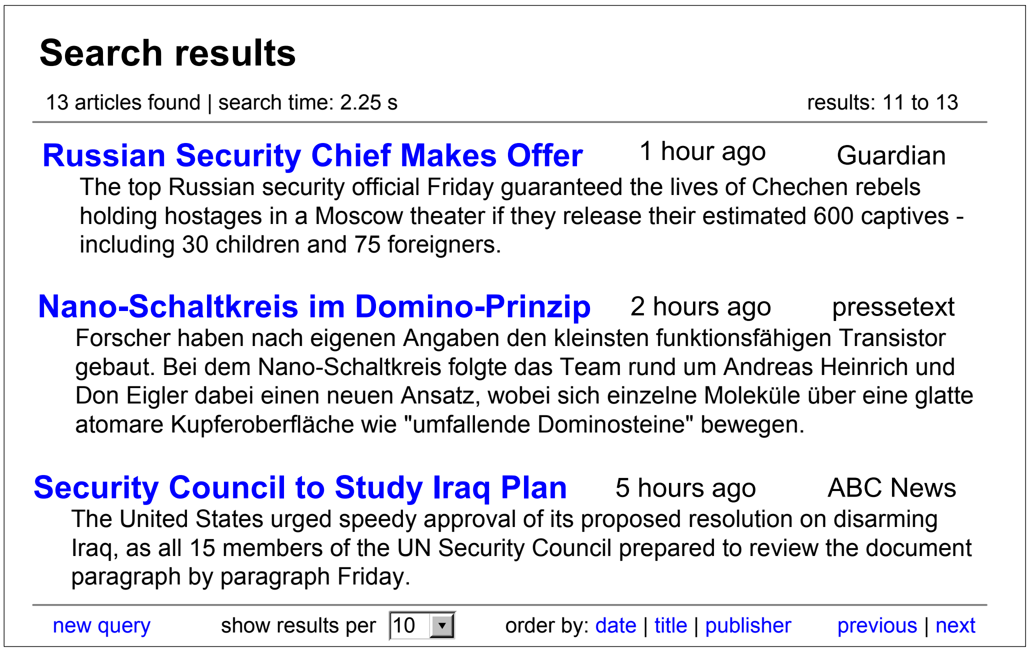
More specific: SS | Royal Canadian Mounted Police | gendarmerie | policeman | secret police | Scotland Yard | posse

Search results

Search results are shown in a limited-length list of items. The user can modify the number of items on one “search result page” using a drop-down list. The search result page contains the following items: a title, the number of articles found, the search time and the beginning and end of the current result page. The title of a “normal” search result page (i.e. results of a simple or advanced search) is “Search results”. The title of a hierarchical subject view page is the name of the clicked news subject.

Below the page header containing this information, the results are shown using their title, date, publisher and abstract. If no abstract is present, the first 200 characters of the article content are shown. Each title is a hyperlink to the content of the article.

Figure 22: search results



Search results

13 articles found | search time: 2.25 s results: 11 to 13

Russian Security Chief Makes Offer 1 hour ago Guardian

The top Russian security official Friday guaranteed the lives of Chechen rebels holding hostages in a Moscow theater if they release their estimated 600 captives - including 30 children and 75 foreigners.

Nano-Schaltkreis im Domino-Prinzip 2 hours ago pressetext

Forscher haben nach eigenen Angaben den kleinsten funktionsfähigen Transistor gebaut. Bei dem Nano-Schaltkreis folgte das Team rund um Andreas Heinrich und Don Eigler dabei einen neuen Ansatz, wobei sich einzelne Moleküle über eine glatte atomare Kupferoberfläche wie "umfallende Dominosteine" bewegen.

Security Council to Study Iraq Plan 5 hours ago ABC News

The United States urged speedy approval of its proposed resolution on disarming Iraq, as all 15 members of the UN Security Council prepared to review the document paragraph by paragraph Friday.

new query show results per 10 order by: date | title | publisher previous | next

At the bottom of the page, a number of hyperlinks allow navigation through and re-filtering of the search results:

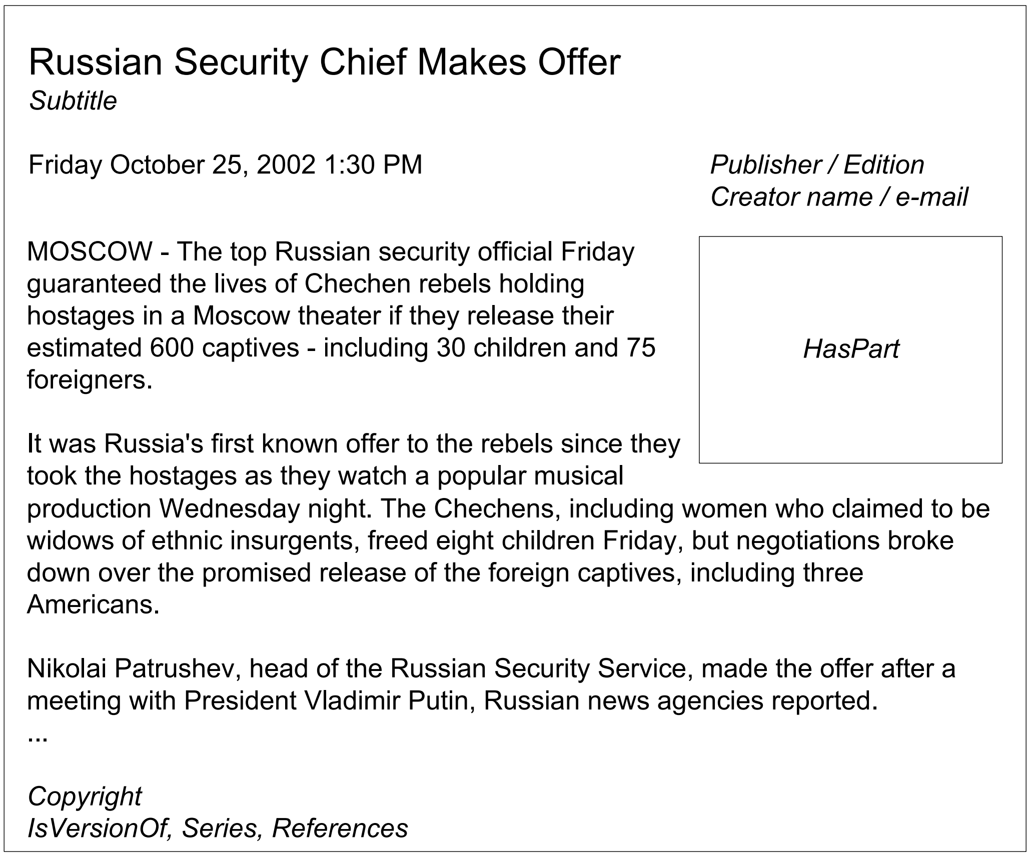
- The link “new query” points to the empty query form (this can be either the simple search or advanced search query form).
- A drop-down list allows changing the number of results per result page. When a different value is selected, the page is automatically refreshed with this new value. In this case, the result page is reset to the first one.
- The links “date”, “title” and “publisher” allow re-ordering of the results. When re-ordering, the result page is also reset to the first one.
- Previous and next point to the previous and next (if any) result page

Article display

The following meta-data fields are shown (if available): Title, Subtitle, Creator, Publisher, Copyright, Edition, Issued, HasPart (picture or graphic), IsVersionOf, Series, References and Spatial. In Figure 23, the non-available fields are written in italic.

Next to the meta-data, the article content is shown. If the news article is part of a search result, the words or sentences matching the query string(s) are indicated with a different colour.

Figure 23: Article display



Russian Security Chief Makes Offer

Subtitle

Friday October 25, 2002 1:30 PM

Publisher / Edition
Creator name / e-mail

HasPart

MOSCOW - The top Russian security official Friday guaranteed the lives of Chechen rebels holding hostages in a Moscow theater if they release their estimated 600 captives - including 30 children and 75 foreigners.

It was Russia's first known offer to the rebels since they took the hostages as they watch a popular musical production Wednesday night. The Chechens, including women who claimed to be widows of ethnic insurgents, freed eight children Friday, but negotiations broke down over the promised release of the foreign captives, including three Americans.

Nikolai Patrushev, head of the Russian Security Service, made the offer after a meeting with President Vladimir Putin, Russian news agencies reported.
...

Copyright
IsVersionOf, Series, References

B.2.3 Overall Knowledge Layer Prototype (WP3)

The final prototype has been implemented based on the bottom-up developments during the early phases of the OmniPaper project. The end users have direct access to the prototype which has been developed mainly in Workpackages 3 (Overall Knowledge Layer) and 5 (User Interface Layer).

B.2.3.1 System Requirements Analysis and Specifications

The purpose of this prototype is similar the Distributed Information Retrieval prototype, but some functions are added for integration and multilingual capabilities. Users can search with this prototype for news originating from 3 archives (My News, Mediargus and pressetext) in 7 different languages (English, French, German, Spanish, Dutch, Catalan, Portuguese). In this prototype users can perform queries in their own language, yielding results in all 7 languages present in the archives.

The base functions are limited to: simple search, advanced search and relational concept view.

Another important change is the data set: while the Local Knowledge Layer prototype contained a fixed data set of news articles, this prototype contains a dynamic set of news articles, continuously updated by the local archives.

This prototype is used by the OmniPaper consortium and developers during development, testing and demonstration. It is also used by a test group that will evaluate the different prototypes. Further it can be used by the wider public as a demonstration of the OmniPaper work.

B.2.3.2 Functional specification

In view of the business plan decisions taken by the consortium the base functions of the prototype has been limited to the functions directly aimed at the later business phase of the project. The OmniPaper business strategy is to create an additional service to existing news providers and news clipping services. The core functions of this service are:

→ Smart search:

- Semantically enhanced
- Multilingual query

→ Web of concepts as a tool for query navigation and refinement

In a later business phase, an additional function will be the extension of news provider's content with other provider's content using the OmniPaper news exchange (or news update) architecture. Since this function is not part of the "OmniPaper core" and since its final usage is uncertain, it will only be designed theoretically and not included in the real implementation of the WP3 prototype.

The functions of the WP3 prototype are:

→ **Smart search** in news articles:

- Simple search: only one search box
- Advanced search: search box and metadata fields for limiting the search

→ Navigation:

- Web of concepts: show concepts relevant to the current query
- Hierarchical subject view: show predefined news subjects in a hierarchical way (like a tree of news subjects)

→ Article retrieval: show the full text of the news article

The following functions have been added in relation to the WP2 prototype:

→ Multi-archive search. This means that the SOAP search functions have to send out requests to multiple archives and combine these results in the OmniPaper user interface.

→ Multilinguality of knowledge layer

- Multilingual SOAP: the SOAP search methods need to be redefined so that they support multilingual search operations. The central OmniPaper system will typically translate the query and send the different translations to the local archives.
- Automatic language detection (query and articles) ("language identification"): both for use on queries and news articles, the WP3 prototype will contain a language detection module that automatically identifies in what language the query or article is written. This will also have effect on the Language metadata item.
- Adaptation of multilingual Topic Map database (EWN): for the Topic Map search to support multilingual querying, the Topic Map database needs to be redefined and filled with EuroWordNet linguistic information. Further the use of the Inter-Lingual Index and Top Level Concepts mechanisms needs to be taken into account. Since the EWN

database contains a limited set of languages, **the WP3 language support is limited to: English, Spanish, German, French and Dutch.**

  - Multilingual query and navigation through Topic Map

- → Dynamic update of knowledge layer
  - Maintenance of the link DB and Topic Map DB: these databases are linked to each other, so when anything changes, related information possibly has to be updated too. Frequent rescanning of the databases probably is necessary for keeping all information consistent.
  - News feed: handling of new articles. When a local archive sends out the SOAP message NewsUpdate to indicate the arrival of new articles, the OmniPaper system has to respond in a fast and appropriate way, so that it can include the new article as soon as possible in the system.

Other prototype enhancements are:

- → Topic Map-based query expansion: the Topic Map DB can be used for "pure Topic Map" search, where the links between a topic and a news article is used for getting search results. However, WP2 tests have shown pour search performance for the pure Topic Map search. Therefore the Topic Map will be used for semantic query expansion (and query translation) on top of full-text search.
- → Graphical version of Web of concepts: the web of concepts can mainly be used for two purposes. First: as a "query guide", where the user can refine his query. Second: as a navigation tool, where the user can browse to other concepts related to his query. The aim of a graphical version is to enhance the user experience and make the use of the web of concepts more intuitive.
- → Update of local archive information: when a local archive sends the SOAP message MetaDataFeedback to the OmniPaper system, this system has to respond by sending back the latest meta data corresponding to the requested articles.

B.2.3.3 External interfaces

This is identical to the Distributed Information Retrieval Prototype description.

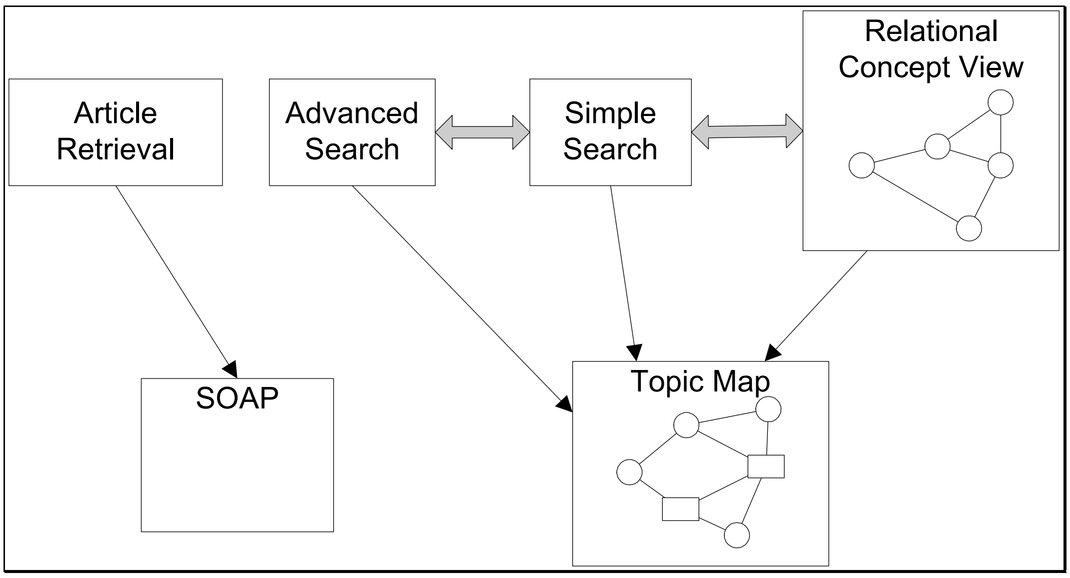
B.2.3.4 Context diagram

This diagram is identical to the Distributed Information Retrieval Prototype diagram.

B.2.3.5 System diagram

The functional specifications defined in B.2.3.2 can be summarised into four main system functions: article retrieval, advanced search, simple search and relational concept view. Figure 24 depicts how these functions relate to the different parts of the prototype (SOAP and Topic Maps).

Figure 24: system diagram



B.2.3.6 Functional Analysis

In this section all system functions are described in detail.

Article retrieval

This function retrieves a news article from the news provider's database using SOAP and displays this article in HTML format.

Search

This function allows users to search for news articles in the predefined data set. The idea behind the prototypes is:

- Querying is used to narrow the search, in such a way that not too many results are shown and that the relevance ranking makes sure that the most relevant results are really shown on top.
- Navigation is used for widening the search into more general concepts, for re-focusing the query into other areas or for filtering/narrowing the search results to more specific areas.

Simple search

In the simple search users can type a query in a single input field and execute the query using the enter key or the "GO" button.
The following scenario is used when a user types in a query:

- The language of query is detected (or indicated by the user)
- The query is translated to all available languages
- The query is semantically expanded (in all available languages) to include synonyms of query words
- The "enhanced" queries (in all available languages) are forwarded to the news providers

Advanced search (metadata query input fields):

With this function users can perform more sophisticated queries using one or more metadata fields to limit their query. Possible metadata fields: Title, Subject, Key-List, Abstract, Publisher, Creator, Issued-From, Issued-To. Two last fields are searching on the metadata field Issued. Fields are combined by default using the AND operator.

For the advanced search, three possibilities exist:

4. The user only types a query in one or more of the metadata fields: a metadata search is performed.
5. The user types a query in the “Search text” field and also in one or more of the metadata fields: a simple search is performed and the search results are filtered using metadata search.
6. The user types a query in the “Search text” field and nothing in the metadata fields: a simple search is performed.

Limited search period

Since the WP3 data set is extended to all articles of all news providers a default search operation is limited to articles from the last 30 days (or less if necessary). If the user wants to find older articles the advanced search has to be used.

Search results

This function displays a list of search results. For each article in the result list the following metadata fields are displayed: Title, Issued, Language, Abstract and Publisher. Results are ranked by relevance with the most relevant result on top of the result list. On top of the result list, the following information is displayed: the number of search results, the search time. Search results can be ordered by Title, Issued, Publisher or Relevance.

Query processing

This function processes the query that the user has typed into a query that can be understood by the search engine. A “correct” query consists of a number of keywords that can be combined using the following Boolean operators: AND, OR. Keywords and operators are separated using a space. Keywords separated with only a space are combined using the OR operator. The “ sign can be used to indicate exact query strings or keywords that consist of multiple words. Brackets can be used to logically separate parts of a query. The inclusion (+) or exclusion (-) signs can be used to indicate if a certain keyword must or may not appear in the resulting articles.

When a query is processed a query syntax tree is built.

Display of query

The current query, as it was parsed by the system, is displayed. This way users can keep up with query changes if they are refocusing their query.

Search options for users

When submitting a search, users can indicate:

- In what language they are searching. The entire user interface will be shown in that language.
- In what language(s) they want to see search results. This means that only news articles are shown that comply with that language(s).

Navigation: “web of concepts”

(Since it is not in line with the business plan decisions, the hierarchical subject view is not supported anymore in the WP3 prototype.)

The web of concepts gives the user a relational view on concepts relevant to the current query. The following functions are possible:

- Refocus the current query:
  - Using semantically related concepts (more general, more specific, associated).
  - Using different word senses of the query keywords. This allows users to redefine/refine their query if one of the query keywords has different possible meanings (e.g. bank as a financial institution or as a piece of furniture).
- Remove concepts from the query.
- Navigate between different concepts relevant the query
- For each concept, the following information is shown:
  - Semantic explanation of the concept
  - All keywords related to the concept
  - Related concepts according to different word senses of the query keyword.
  - Related concepts according to semantic relations from WordNet.

Multilingual support

User interface

All text appearing in the user interface is defined in an initialisation XML file, so that multiple languages are supported for the UI. This means that items in the user interface such as directions to the user and other words and phrases explaining the functioning of the prototype, are set in a configurable manner, so that they can be shown in multiple languages.

Web of concepts

Since the web of concepts is regarded as being an integral part of the user interface it is displayed in the same language as the overall user interface language. So words appearing in this web are always in the chosen UI language, even if the original query words are in another language.

Query

The query can be in any supported language (English, German, Spanish, Dutch or French). Normally the language identification software will try to automatically induce the language from the words appearing in the query. If this fails, the user interface language will be used as query language.

News articles

All news articles and their metadata remain in the article’s original language.

A link to an online automatic translation engine will be provided for each article, so that a (rough) translation can be done into the user’s language.

B.2.3.7 Data set

Where the WP2 data set was a limited set of 1881 English news articles published in September 2002, the WP3 set has been extended to a dynamic multilingual set: at one specific period in time, the WP3 data set has been be completely reset and since then, the system contains a dynamic set of news articles, obtained using the dynamic update functionality.

Next to this data set, also a test set will be used in order to test the multilingual information retrieval performance of the prototype.

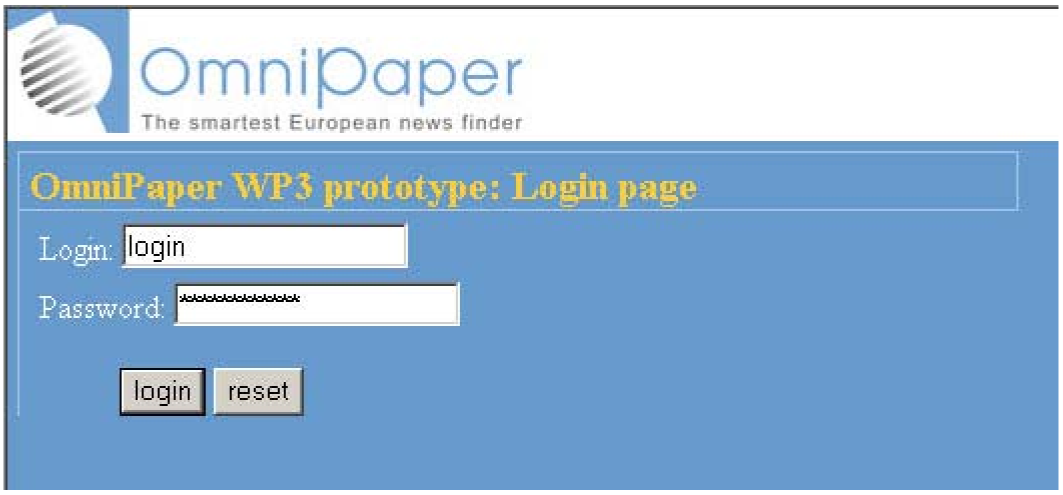
B.2.3.8 User Interface design

The user interface design for the WP3 prototype is completely changed in comparison with the previous prototypes. The changes made are based on user input and developer experiences. In WP6 the user interface will be evaluated thoroughly so it can evolve to an ergonomic and easy-to-use interface.

Login screen

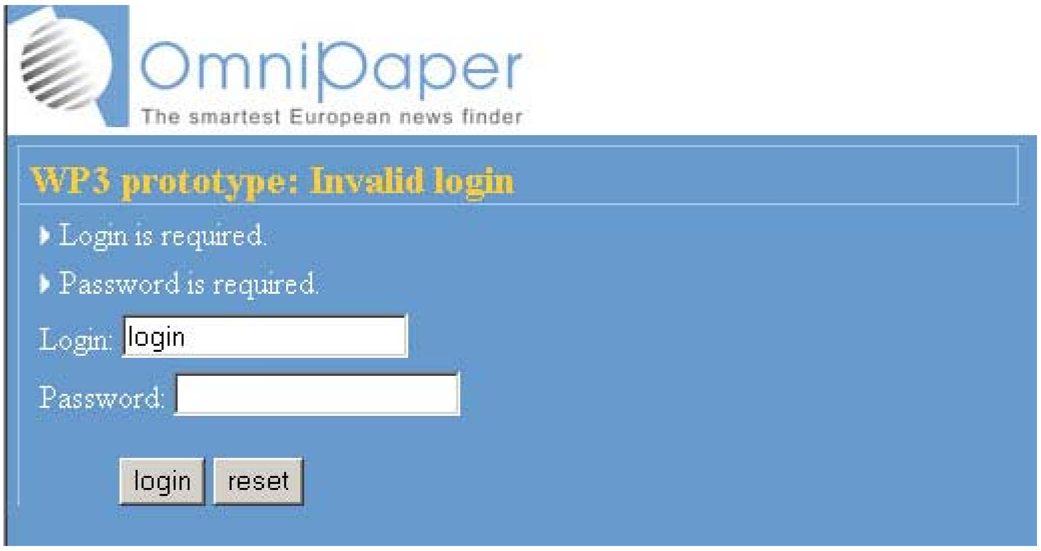
Before the prototype can be used, the user must login using a user name and a password.

Figure 25: Login screen


OmniPaper
The smartest European news finder
OmniPaper WP3 prototype: Login page
Login: login
Password: ************

The validation plugin in the framework ensures that a login and password must be provided before authentication with the user database can occur. Failed validation and/or authentication are directly displayed to the user.

Figure 26: Login screen with failed validation/authentication


OmniPaper
The smartest European news finder
WP3 prototype: Invalid login
- Login is required.
- Password is required.

Login: login
Password:

When the login information is correctly validated and authenticated, the user is redirected to the simple search screen.

Simple search screen

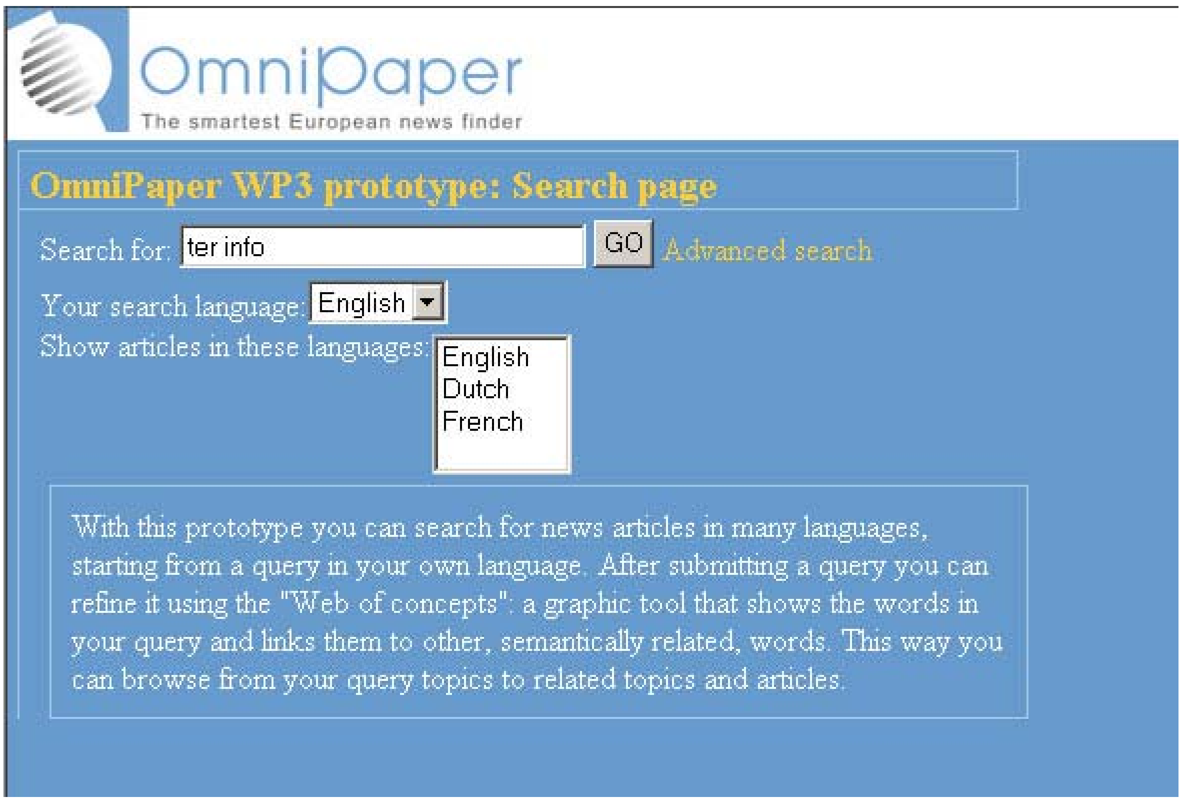
The simple search box is quite different as in previous prototypes. A search can only be performed successfully when the query, the search language and the article language fields are provided.

A search is considered to be valid when:

- The query field is not empty
- At least one article language is specified
- A search language is specified

The screen contains a link to the advanced screen, which is described in the next section. A short description about the system usage is located at the bottom of the screen.

Figure 27: default/ simple search screen


OmniPaper
The smartest European news finder
OmniPaper WP3 prototype: Search page
Search for:
ter info
GO
Advanced search
Your search language:
English
Show articles in these languages:
English
Dutch
French

With this prototype you can search for news articles in many languages, starting from a query in your own language. After submitting a query you can refine it using the "Web of concepts": a graphic tool that shows the words in your query and links them to other, semantically related, words. This way you can browse from your query topics to related topics and articles.

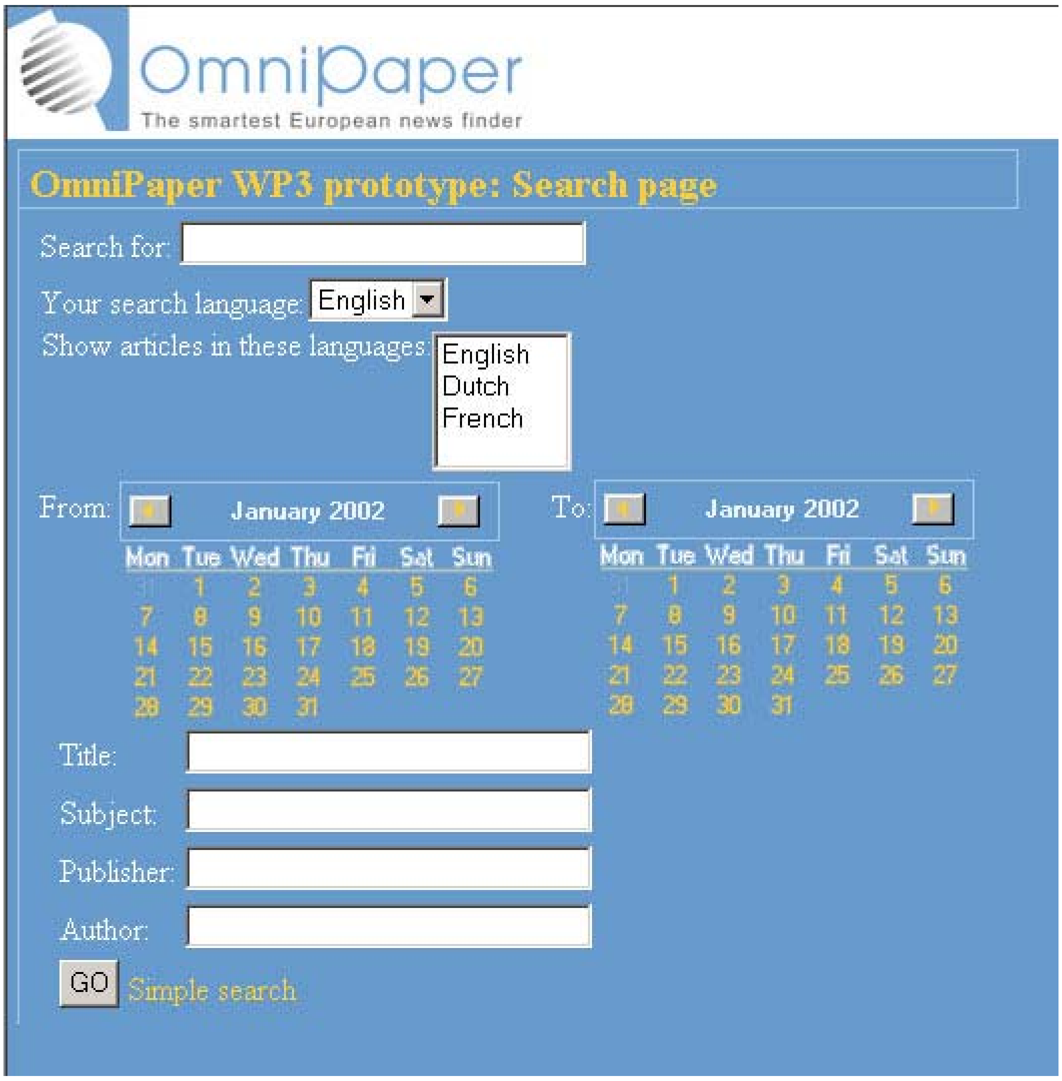
When the search information is submitted to the OmniPaper system successfully the user is redirected to the result screen. Otherwise, errors are displayed in the same way as at the login screen.

Advanced search screen

In the advanced search the simple search input fields are extended with a number of additional input fields for limiting the search to certain metadata values. For the fields Issued from and Issued to, a calendar is created in such a way that it is impossible to enter an incorrect date (e.g. 31 February

2003). Just like the default search, validation of the fields is executed before performing the actual search.

Figure 28: Advanced search screen



OmniPaper WP3 prototype: Search page

Search for:

Your search language: English

Show articles in these languages: English Dutch French

From: January 2002 To: January 2002

Mon	Tue	Wed	Thu	Fri	Sat	Sun
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Title:

Subject:

Publisher:

Author:

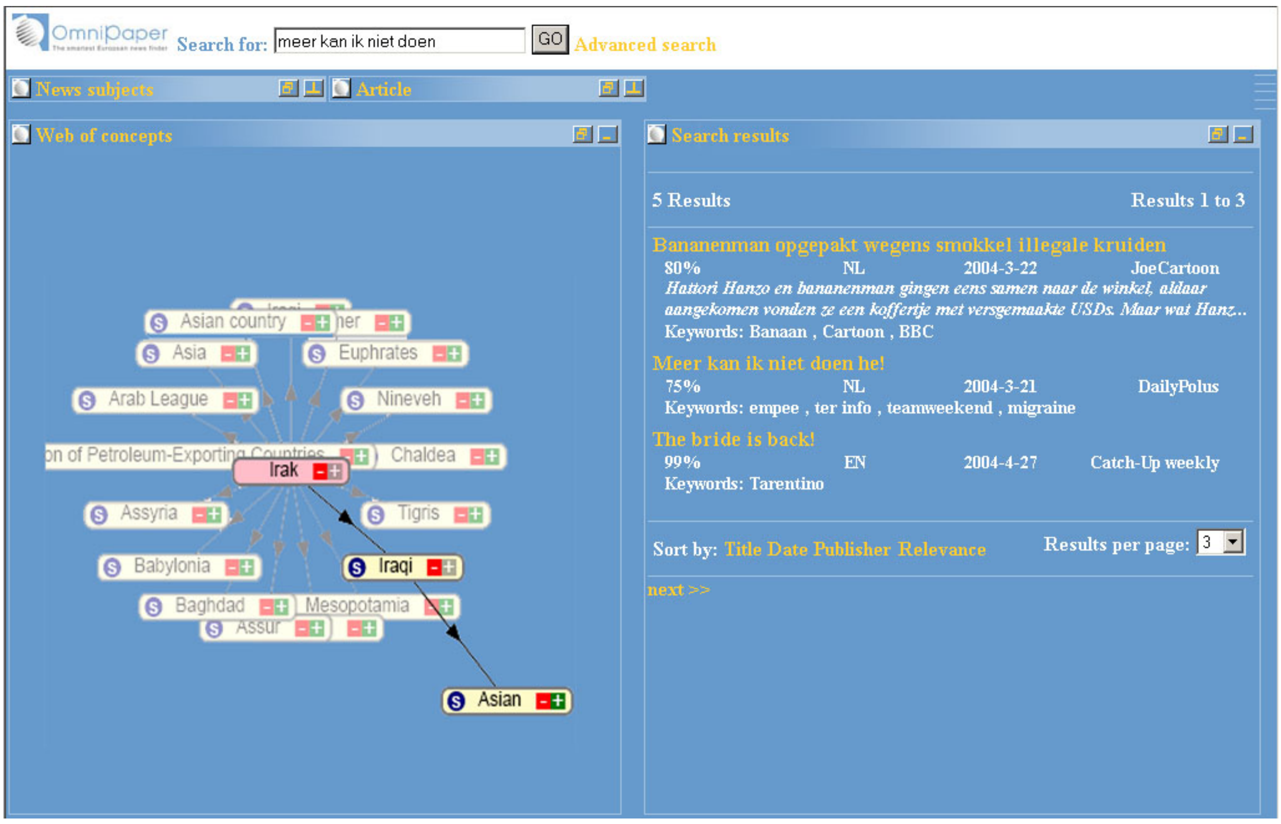
GO Simple search

Result screen

The result screen is divided in four parts, also called views:

- Header view: Standard view, available on top of all result pages.
- (Relational) Concept view: An SVG presentation of concepts is displayed in this view.
- Result list view: The articles linked to the objects are displayed as a result list.
- Article view: The complete article itself is shown.

Figure 29: default result screen



5 Results Results 1 to 3

Bananenman opgepakt wegens smokkel illegale kruiden
80% NL 2004-3-22 JoeCartoon
Hattori Hanzo en bananenman gingen eens samen naar de winkel, aldaar aangekomen vonden ze een koffertje met versgemaakte USDs. Maar wat Hanz...
Keywords: Banaan , Cartoon , BBC

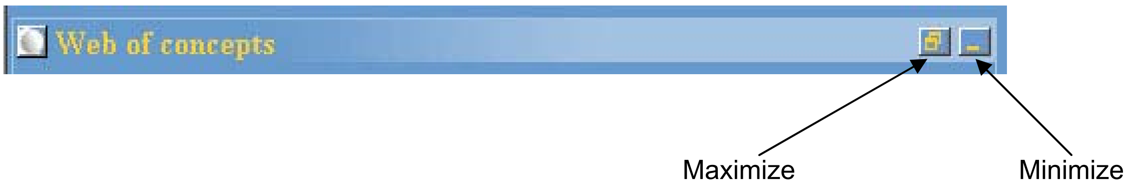
Meer kan ik niet doen he!
75% NL 2004-3-21 DailyPolus
Keywords: empee , ter info , teamweekend , migraine

The bride is back!
99% EN 2004-4-27 Catch-Up weekly
Keywords: Tarentino

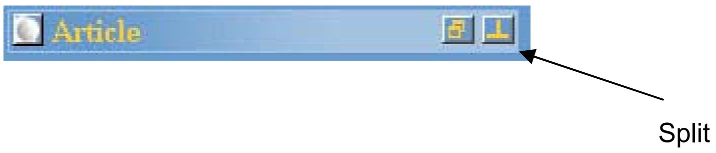
Sort by: Title Date Publisher Relevance Results per page: 3

next >>

Every view can be minimized and/or maximized over the screen.



Splitting the screen in 2 parts is also possible:



The behaviour of the screens, either at the minimize, maximize or split action is logically decided by the system. In this way it is possible to have a whole set of logical views available for the user.

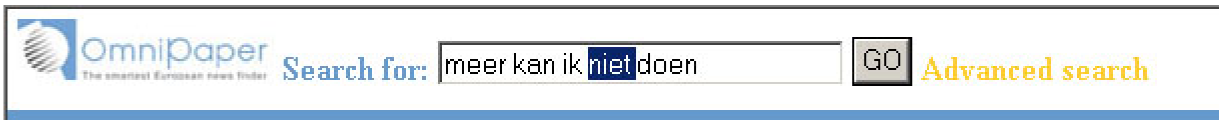
Search view

In this view it is possible to perform a new search without returning to the full screen simple search page. It is not possible to change either the article languages or the search language. During the search action the system uses those values from the lastly performed (full screen) simple search. In this way it is easy to quickly alter your query without having to specify the language options time after time.

A link to the advanced search is available. When activating the link the advanced screen is displayed, from this page it possible to perform an advanced search or to go back to the simple search screen (full screen).

This view also contains a link to an extensive help page (not implemented yet).

Figure 30: Search view



Relational concept view

Text version

This view consists of two frames[3]: the query frame and the concept frame.

In the **query frame** the current query is shown as it was processed by the system. This means that the Boolean operators and brackets that might have been added by the query engine are shown. For example: the query phrase “belgian police force” will result in “belgian OR police OR force”.

The different keywords found in the query are shown. Three possibilities exist:

4. No word sense ambiguity exists for the query keyword: the corresponding concept is shown in bold.
5. Word sense ambiguity exists for the query keyword but the preferred meaning of the word has been chosen (either by the user or by the word sense disambiguation system): the preferred concept is shown in bold. The other meanings of the keyword are also shown and can be clicked. If such a word is clicked, this meaning of the keyword is selected and made bold.
6. Word sense ambiguity exists for the query keyword and no preferred meaning of the word has been chosen: the keyword is shown. All possible meanings of the keyword are also shown and can be clicked. If such a word is clicked, this meaning of the keyword is selected and made bold.

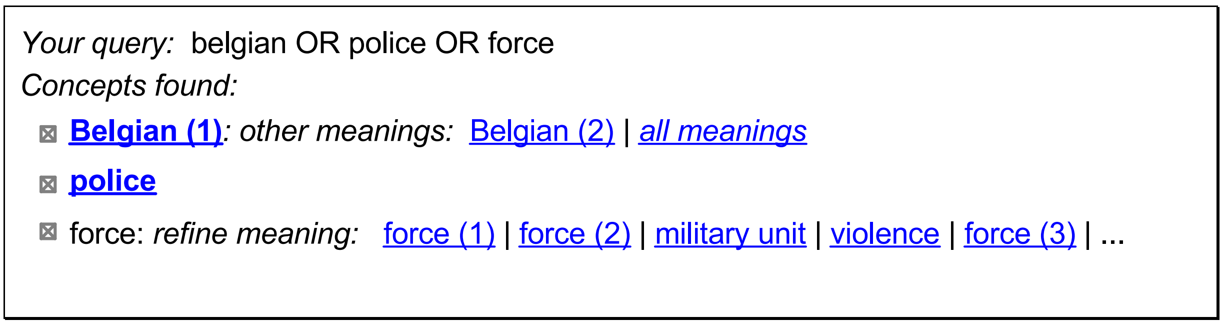
In the latter case, the displayed search results will relate to all possible meanings of the keyword. Upon clicking of a certain meaning, the result set is limited to articles that are only related to the clicked meaning. The user can always go back to the bigger result set by clicking the link “all meanings” for that keyword.

A “meaning” can be represented in several ways, in order of preference: by showing the domain label, by showing the top concept or by showing the keyword (for more information about domain labels and top concepts, see [EWN], p9-10). If words exist that have the same lowest level domain, their concepts can be grouped together as only one concept. At this moment, however domain labels only exist for the computer terminology domain. Domain information could be added according to the ITPC subject reference in order to use this feature in this prototype. If domain information is not present, the system will use the top concept ontology of EuroWordNet. This is a hierarchy of language-independent concepts, reflecting important semantic distinctions. If different candidate concepts belong the same top concept, the system will show the keywords of the concept, along with a index to distinguish equal keywords.

In order to help the user to refine the keyword into a certain word meaning, a semantic explanation of a certain word meaning is displayed in a tool tip upon mouse over.

[3] In this section, a “frame” should be understood as a “display entity”, which does not necessarily correspond to a HTML frame. It could also mean a cell in a table or a separate JSP page.

Figure 31: query frame in relational concept view



Your query: belgian OR police OR force

Concepts found:

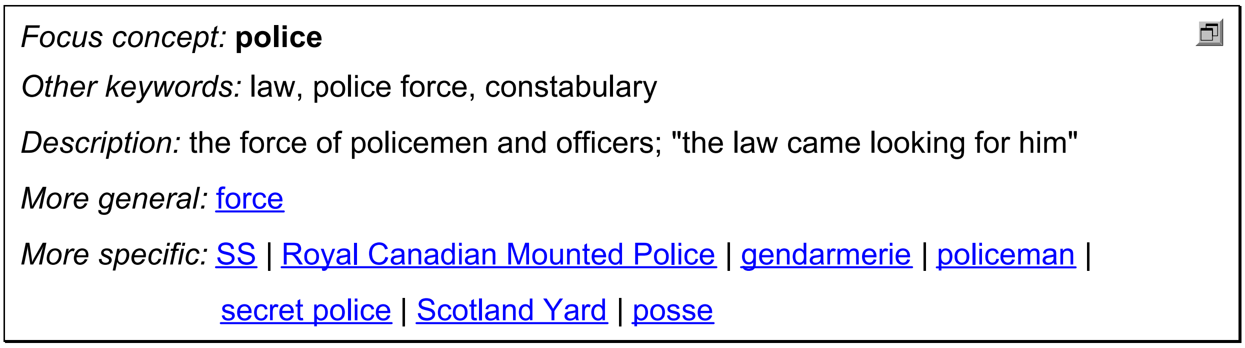
- ☒ **Belgian (1)**: *other meanings:* Belgian (2) | *all meanings*
- ☒ **police**
- ☒ force: *refine meaning:* force (1) | force (2) | military unit | violence | force (3) | ...

The X icon in front of each concept allows the user to remove that concept from the query. Clicking on a concept causes the concept frame to take this concept as the focus concept.

In the **concept frame** more information about the focus concept is shown. Next to the semantic explanation of the concept, the semantically related concepts are shown (relations “more general”, “more specific” and “associated”). With these related concepts the user can broaden or narrow his or her query.

Two possibilities exist for displaying the concept frame. It can be displayed as a separate frame that refreshes when a user changes the focus concept in the query frame or it can be displayed as a tool tip - like pop-up screen that appears upon mouse over of a concept in the query frame. The top right button allows switching between these two views. When this button is clicked the concept frame disappears, giving its space to the result frame. The concept frame is then displayed as a pop-up tool tip upon mouse over of a concept in the query frame. The query frame now gets the same top right icon that allows going back to the previous view.

Figure 32: concept frame in relational concept view



Focus concept: **police**

Other keywords: law, police force, constabulary

Description: the force of policemen and officers; "the law came looking for him"

More general: force

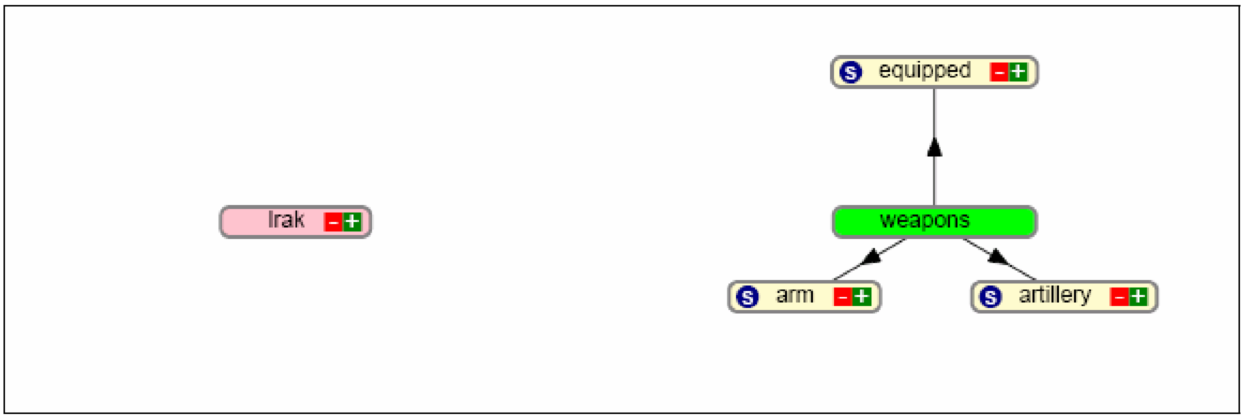
More specific: SS | Royal Canadian Mounted Police | gendarmerie | policeman | secret police | Scotland Yard | posse

The information that is needed to build the query frame and the concept frame is delivered in XML format.

Graphical version

The different keywords and concepts found in the query are shown. Keywords and concepts are represented by rounded rectangles with buttons explained later on. Keywords are green, unfocused concepts are yellow; concepts that are brought into focus are pink. Keywords and concepts use a word to distinguish them from other topics. The word displayed for a keyword is the keyword itself, in case of a concept the word used in the view is the preferred keyword of the concept.

Figure 33: SVG view representing the query "Iraq OR weapons"



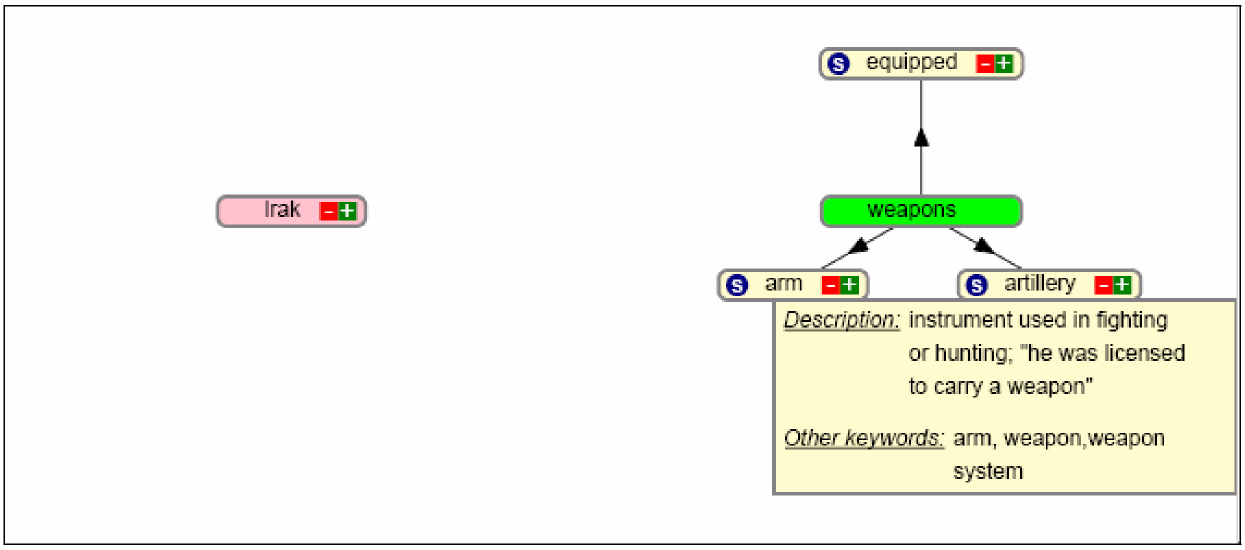
Two possibilities exist:

1. No word sense ambiguity exists for the query keyword: the query keyword is represented directly in the view by its corresponding concept. Because this concept represents a query keyword it is therefore considered a focus concept and coloured pink. An example for this situation is depicted in Figure 33 for the query keyword “Iraq”.
2. Word sense ambiguity exists for the query keyword and no preferred meaning of the word has been chosen: the keyword is shown in green. All possible meanings of the keyword represented by yellow concepts are also shown around the keyword and can be clicked. An example for this situation is depicted in Figure 33 for the query keyword “weapons”. If such a concept is clicked, the green keyword is replaced by this yellow concept and the other related concepts are removed. This yellow concept is a new focus concept and therefore the colour is changed to pink.

In the latter case, the displayed search results will relate to all possible meanings of the keyword. Upon selecting a certain meaning represented by a concept, the result set is limited to articles that are only related to the selected meaning.

In order to help the user to refine the keyword into a certain concept, a semantic explanation of the meaning represented by the concept is displayed in a tool tip upon mouse over. This explanation consists of a short textual description and other keywords for this concept (see Figure 34).

Figure 34: Tool tip for the concept "arm"



Description: instrument used in fighting or hunting; "he was licensed to carry a weapon"

Other keywords: arm, weapon,weapon system

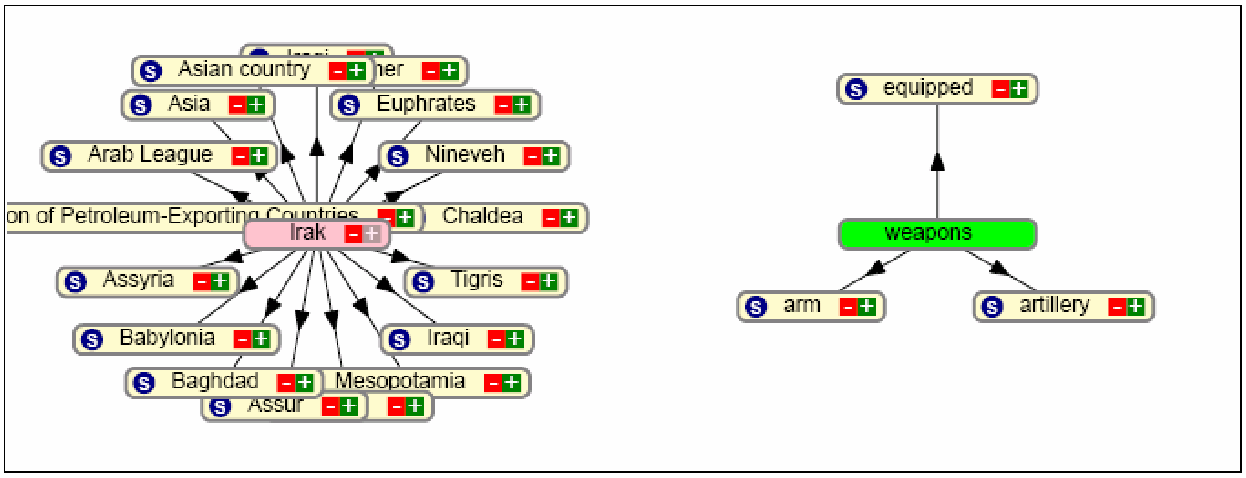
The expand button (rectangle with a plus sign) at the right of the each concept name allows the user to expand that concept to its semantically related concepts (relations “more general”, “more specific” and “associated”). Related concepts are drawn on an imaginary circle around the concept (see Figure 35). With these related concepts the user can broaden or narrow his or her query. Expanding a topic does not change the current query. It offers a way to the user to explore semantically related concepts. Every concept can be expanded. A concept “created” by expanding a query keyword or query concept can in turn be expanded and so on. A grey “plus sign” button means that the concept has no more relations than the ones shown and therefore can not be expanded anymore.

To avoid as much as possible overlapping concepts when a topic is expanded, some measures are foreseen:

- The radius is calculated in function of the number of associations.
- The circle around the concept is divided into equal radial sectors based on the number of relations. The sectors occupied by related concepts already drawn for a previous expansion of another topic are marked busy. The new related concepts are drawn in the remaining free sectors.

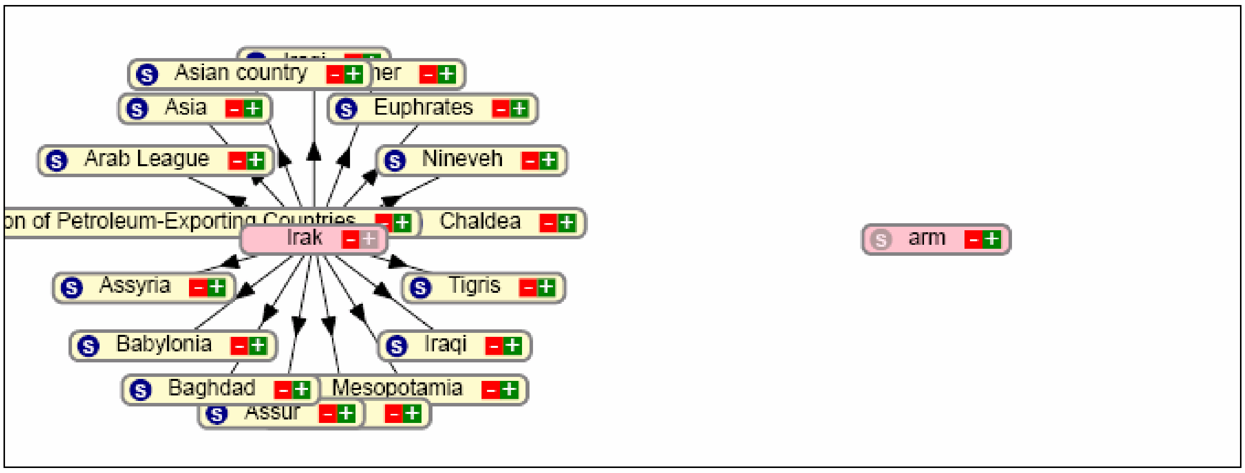
The remove button (rectangle with a minus sign) at the right of the each concept name allows the user to remove that concept from the query. Topics expanded by this topic and having only relations with this topic are also removed.

Figure 35: Expanding the concept "Iraq"



Clicking on the select button (circle with an “S”) at the left of the each concept name causes the system to take this concept as the focus concept (see Figure 36). Concepts not expanded by this concept are removed, this concept moves to the location of the previous focus concept and is shown in pink.

Figure 36: Result after selecting the concept "arm" to disambiguate the query keyword "weapons"

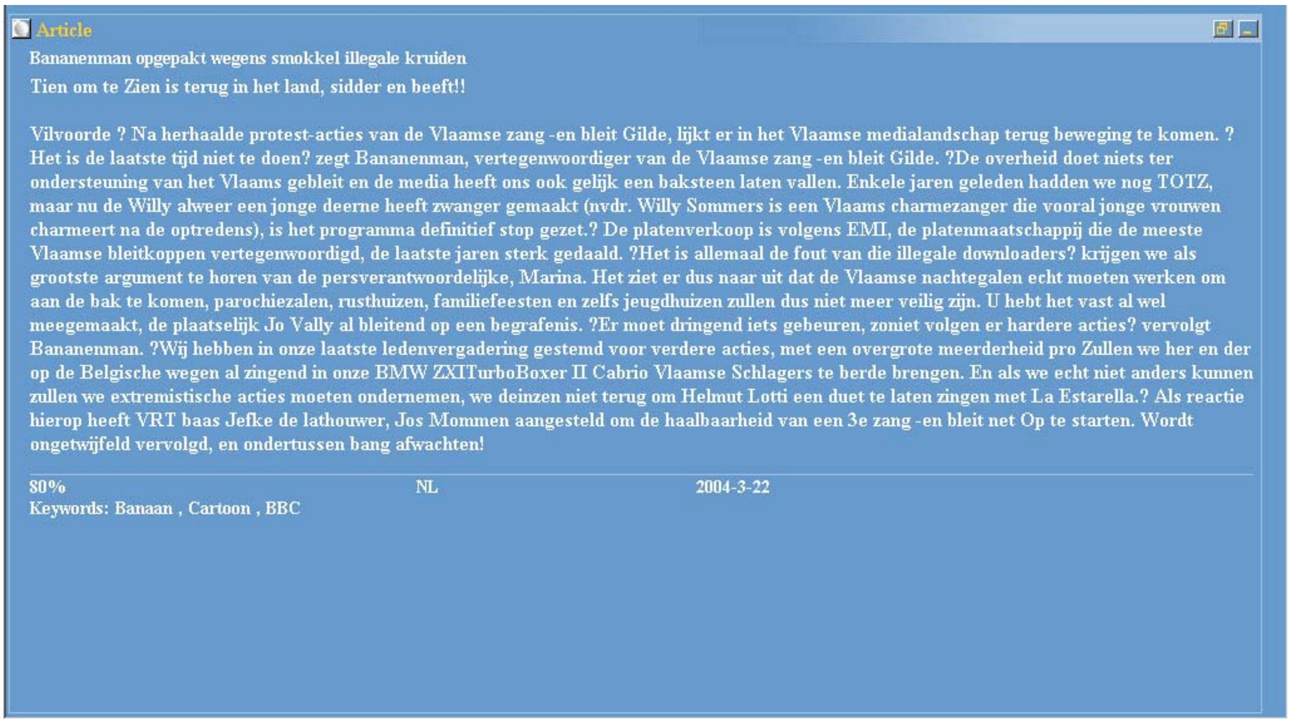


Concepts and topics can be dragged and dropped with the same ease as in classic Windows applications. Just catch the rectangle of the chosen topic by pressing down the mouse button, hold the button down while moving and release the button on the desired position.

Article view

Once an article is selected from the result list view, the article can be read. The full article is displayed to the user with its matching title, and a few metadata fields which were also available in the result list screen.

Figure 37: Article view



Bananenman opgepakt wegens smokkel illegale kruiden

Tien om te Zien is terug in het land, sidder en beeft!!

Vilvoorde ? Na herhaalde protest-acties van de Vlaamse zang -en bleit Gilde, lijkt er in het Vlaamse medialandschap terug beweging te komen. ? Het is de laatste tijd niet te doen? zegt Bananenman, vertegenwoordiger van de Vlaamse zang -en bleit Gilde. ?De overheid doet niets ter ondersteuning van het Vlaams gebleit en de media heeft ons ook gelijk een baksteen laten vallen. Enkele jaren geleden hadden we nog TOTZ, maar nu de Willy alweer een jonge deerne heeft zwanger gemaakt (nvdr. Willy Sommers is een Vlaams charmezanger die vooral jonge vrouwen charmeert na de optredens), is het programma definitief stop gezet.? De platenverkoop is volgens EMI, de platenmaatschappij die de meeste Vlaamse bleitkoppen vertegenwoordigd, de laatste jaren sterk gedaald. ?Het is allemaal de fout van die illegale downloaders? krijgen we als grootste argument te horen van de persverantwoordelijke, Marina. Het ziet er dus naar uit dat de Vlaamse nachtegalen echt moeten werken om aan de bak te komen, parochiezalen, rusthuizen, familiefeesten en zelfs jeugdhuizen zullen dus niet meer veilig zijn. U hebt het vast al wel meegemaakt, de plaatselijk Jo Vally al bleitend op een begrafenis. ?Er moet dringend iets gebeuren, zoniet volgen er hardere acties? vervolgt Bananenman. ?Wij hebben in onze laatste ledenvergadering gestemd voor verdere acties, met een overgrote meerderheid pro Zullen we her en der op de Belgische wegen al zingend in onze BMW ZXITurboBoxer II Cabrio Vlaamse Schlagers te berde brengen. En als we echt niet anders kunnen zullen we extremistische acties moeten ondernemen, we deinzen niet terug om Helmut Lotti een duet te laten zingen met La Estarella.? Als reactie hierop heeft VRT baas Jefke de lathouwer, Jos Mommen aangesteld om de haalbaarheid van een 3e zang -en bleit net Op te starten. Wordt ongetwijfeld vervolgd, en ondertussen bang afwachten!

80% NL 2004-3-22
Keywords: Banaan , Cartoon , BBC

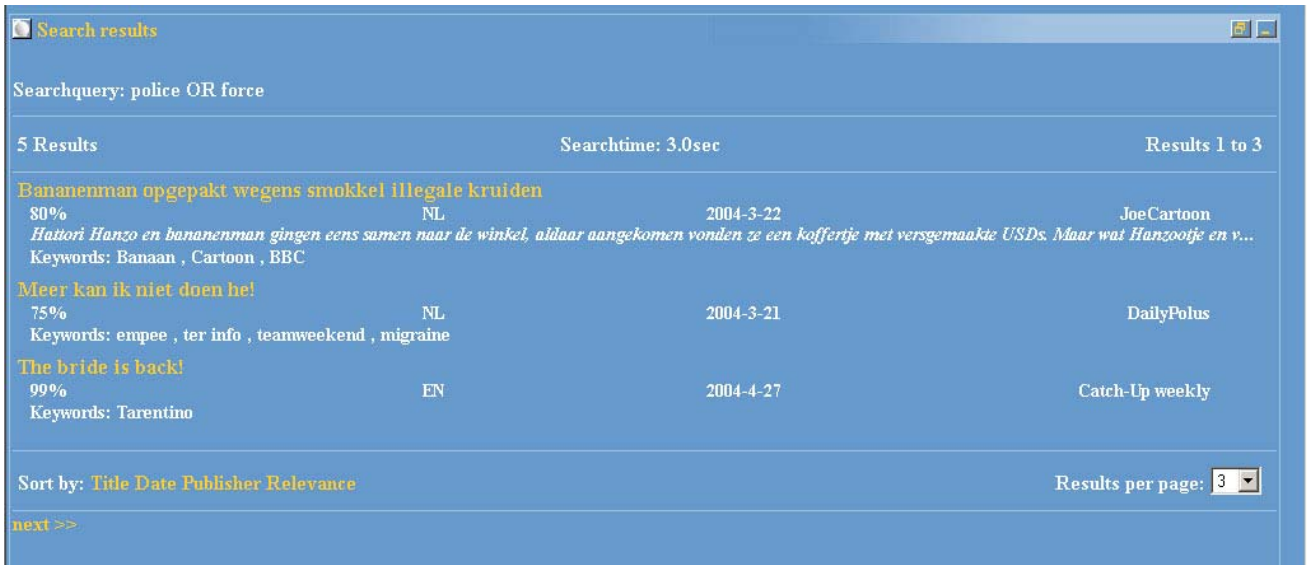
Result list view

In the result list view the current query is shown as it was processed by the system. This means that the Boolean operators and brackets that might have been added by the query engine are shown. For example: the query phrase “belgian police force” will result in “belgian OR police OR force”.

Search results are shown in a limited-length list of items. The user can modify the number of items on one “search result page” using a drop-down list. The search result page contains the following items: a title, the number of articles found, the search time and the beginning and end of the current result page.

The results are shown using their title, date, publisher, relevance and abstract. If no abstract is present, the first 150 characters of the article content are used as abstract and shown. Each title is a hyperlink to the content of the article. Also in the result list is a list of keywords with each article.

Figure 38: result list view



Search results

Searchquery: police OR force

5 Results Searchtime: 3.0sec Results 1 to 3

Bananenman opgepakt wegens smokkel illegale kruiden
80% NL 2004-3-22 JoeCartoon
Hattori Hanzo en bananenman gingen eens samen naar de winkel, aldaar aangekomen vonden ze een koffertje met versgemaakte USDs. Maar wat Hanzootje en v...
Keywords: Banaan , Cartoon , BBC

Meer kan ik niet doen he!
75% NL 2004-3-21 DailyPolus
Keywords: empee , ter info , teamweekend , migraine

The bride is back!
99% EN 2004-4-27 Catch-Up weekly
Keywords: Tarentino

Sort by: Title Date Publisher Relevance Results per page: 3

next >>

At the bottom of the page, a number of hyperlinks allow navigation through and re-filtering of the search results:

- A drop-down list allows changing the number of results per result page. When a different value is selected, the page is automatically refreshed with this new value.
- The links “Date”, “Title”, “Publisher” and “Relevance” allow re-ordering of the results. When re-ordering, the result page is also reset to the first one. The ordering is not limited to one single field: when one order parameter is selected, duplicates are again ordered by predefined secondary parameter(s). The order hierarchy of those parameters and the ordering methods are set by the system but can easily be changed.
- “previous” and “next” point to the previous and next (if any) result pages

B.2.4 Final prototype for smart access to European newspapers (WP5)

This prototype is being described in detail in the document “Final prototype for smart access to European newspapers”, which is publicly available from the OmniPaper website on www.omnipaper.org.

B.2.5 Prototype cross-testing

B.2.5.1 WP2: SOAP, XTM, RDF and Final WP2 prototypes

Goals

The purpose of the testing processes is to establish a testing framework for objective, statistical testing of the WP2 prototypes.

The goal of testing the prototypes developed in WP2 is twofold: in the first place it is to be conceived as a means for detecting flaws, bugs, inconsistencies, bottlenecks,… in the existing prototypes.

A second goal is to learn the strengths and weaknesses of each of the prototypes in order to draw strategic conclusions towards the development of the final WP2 prototype. Some prototypes might perform significantly better than others on certain types of queries and this information can help to make decisions during the integration of the different prototypes.

Comparison Criteria

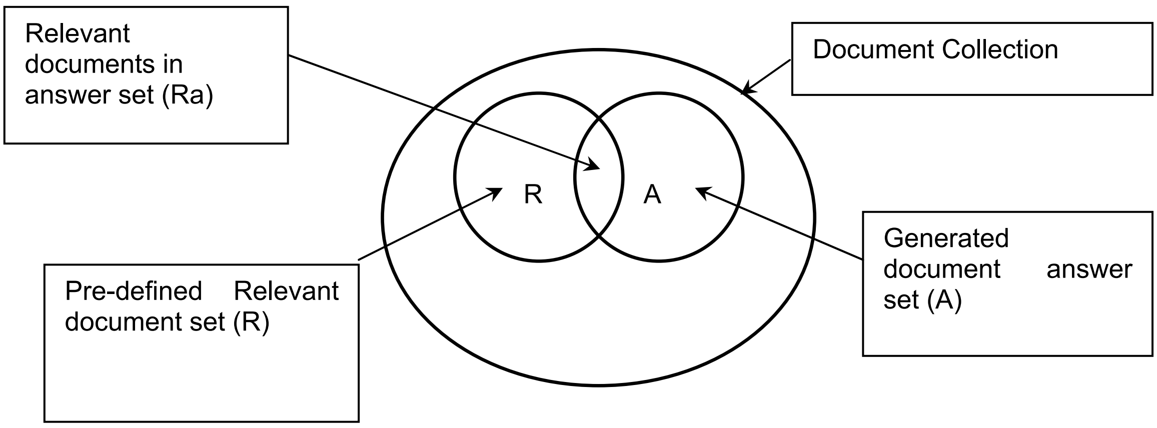
The prototypes have been tested and compared as much numerically as possible in order to measure the pure efficiency and effectiveness of the technologies used. This kind of numeric measurement is possible by the following criteria:

- Relevancy
- Response time
- Data size

On the other hand, some aspects cannot be assessed by the numerical comparison but are still important for the evaluation. Specific features of the technologies and usability are that kind of criteria. For them, observational study will be used with a simple questionnaire for the searcher.

Relevancy

Relevancy is normally measured by recall and precision which are traditional measurement method in Information Retrieval (IR). They evaluate the quality of retrieved documents as follows.


Relevant documents in answer set (Ra)
Document Collection
R
A
Generated document answer set (A)
Pre-defined Relevant document set (R)

Recall --- A measure of the ability of a system to present all relevant items

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved (Ra)}}{\text{Number of relevant documents in collection (R)}}$$

Precision --- A measure of the ability of a system to present only relevant items

$$\text{Precision} = \frac{\text{Number of relevant documents retrieved (Ra)}}{\text{Total number of documents retrieved (A)}}$$

This method is applicable to the comparison between the Direct Retrieval prototype and the metadata prototypes. However, the relevancy of the metadata prototypes can be described in slightly a different way.

$$\text{Metadata search recall} = \frac{\text{Number of relevant metadata documents retrieved from metadata database}}{\text{Number of relevant metadata documents in metadata collection}}$$

$$\text{Metadata search precision} = \frac{\text{Number of relevant metadata documents retrieved from metadata database}}{\text{Total number of metadata document retrieved}}$$

Since metadata describes about each article using a set of elements, these metadata documents correspond to the original articles. In other words, each article will have one metadata document which summarises and represents the article. Therefore, although the metadata prototypes use metadata documents to search, the pre-defined relevant document set (original documents) for each query will be the same thing: a set of relevant metadata documents correspond to a set of relevant (original) documents. This can be identified by the unique identifier of the document.

The query to be used for the comparison between the Direct Retrieval prototype and the metadata prototypes is a set of keywords and optionally one or more metadata elements such as date, subject and so on. The relevant answer set (the collection of relevant documents for each query) must be pre-defined to measure the relevancy. Detailed preparation of the test set will be described in the next section.

It might be interesting to set the different number of relevant keywords for the “key-list” metadata element for the metadata prototypes in order to see which number (e.g. 5, 10, 15,…etc.) will give the highest relevancy. In this case, we are examining not only metadata description technologies but also the efficiency of extraction, weighting and clustering of keywords.

Response time

Response time measures the elapse time between a search engine initiated a query for the system and send a result to that query. This means network time should not be included to the response time. Each prototype system must have a function to return this time.

The assumption is the shorter the time is taken (with high relevancy), the better the system is.

Data size

Data size will be compared to see how much extra bytes each system uses for the same set of metadata. The less bytes the system uses (with good performance), the better the system is.

Data size is the criterion which will be used only to compare the RDF prototype and the XTM prototype as the Direct Retrieval prototype holds no metadata by itself.

Summary

Here is the summary of which prototypes each comparison criterion compares.

Recall and Precision: SOAP vs. RDF; SOAP vs. XTM; RDF vs. XTM

Response time: SOAP vs. RDF vs. XTM in recall and precision comparison

RDF vs. XTM in metadata search

Data size: RDF vs. XTM

Test collection

The test collection consists of:

- A collection of documents (test data set)
- A set of example information requests (topics and queries)
- A set of relevant document for each example information request

The **collection of documents** is a set of news articles in the form of XML. We have defined the common test data sets which have been provided by each news content provider. For the test in WP2, only English articles (1881 articles of Daily Telegraph provided by MyNews) were used. The number of documents needs to be considerably large. However, considering that our prototypes are still relatively small, and the time is constrained for relevance judgment, we kept the number of articles as that of published in one month (September 2002).

The multilingual aspect of the news articles is one of the important issues in the OmniPaper project. However, at the Local Knowledge Layer, each archive is treated individually and the search will not cross different archives. Therefore, we focused on only one archive with English articles as our test bed.

The **set of example information requests** is about 20 topics with 2 or 3 queries for each topic to be searched in the defined test data collection. They can be broad or narrow topics, and form a representative range of real user needs over the document collection (news articles). This means that topics and queries were created by the experts on the demands of typical news-searchers and were based on real user queries on the My News system in the period of September 2002.

For each topic, a **set of relevant documents** was manually pre-defined. Normally, they were defined by the person who created the topic/query. The existing search system helped to develop the set.

However, it is dangerous to depend too much on MyNews search engine because the Direct Retrieval prototype uses it. Therefore the relevance judgment was done as much manually as possible, using different tools.

Example of a test topic:

```
<Topic>
      <Title>Bush’s speech on terrorist attack</Title>
      <Description>President Bush’s speech on the terrorist attack on
      the 11th of September in 2001 to the United Nations on
      anniversary. </Description>
      <Narrative>The relevant documents would include the transcript
      of his speech. The documents should report when and where the
      speech was made</Narrative>
      <Keywords>Bush, speech, terrorist attack, 11th of September,
      United Nations</Keywords>
      <Metadata> Date: 2002-09-11</Metadata>
</Topic>
```

Relevance judgment (answer set preparation)

Relevance judgement was critically important for the evaluation of the prototypes. Therefore, a pre-testing was needed. Ideally, the average performance of the systems on the topics should be neither too good nor too bad. Depending on the results of pre-testing, some modification to the query were needed to avoid retrieval of no relevant documents or only relevant documents. Those topics that have roughly 20 to 100 hits in the pre-test may be used.

Instruments

During the project, the need for a dedicated, statistical testing tool has arisen and the consortium has decided to invest a considerable amount of time in the development of an automatic testing engine (ATE). The ATE is meant to facilitate and encourage testing and retesting of the prototypes, so that the time invested in its development will be regained afterwards. Also, it provides a uniform way to conduct tests, not only with respect to the layout of the test reports, but also – which is much more important – with respect to the algorithms used to calculate the results. This algorithm- or method-invariance is a very important aspect in creating comparable and compatible tests.

The ATE also has some drawbacks to be taken into consideration when drawing conclusions based on automatically generated test reports. The main problem is that the person conducting the test relies on the correctness of the generated reports. Any interpretation of the results can only be valid insofar the reports are correct. Even worse, when the test reports are not reliable, actions taken based on these results can result in a misconception of the prototype’s inner working or in a loss of valuable time while trying to tune the wrong parameters or redesigning algorithms that don’t need to be redesigned.

In order to ensure the uniformity of the test sets, a number of queries and manually extracted answer sets has been created, based on a test set of 1881 Daily Telegraph articles, dating from September 2002. This predefined query and answer set has been used throughout the whole testing phase. Although this a feasible approach, it is not really an objective one. For example, the structural form a query can depend on the its creator’s experience with search engines or the level in which he or she masters the English language,… Moreover, the answer sets have been created by skimming the database on possible relevant articles, but there is no guarantee that all relevant articles have been found. Even worse, relevance is a subjective measure and what is relevant to one person could not be relevant to another. It can however be expected that this subjectivity will be somehow filtered out by taking into account a large number of result sets created by a number of different persons. On the other side, the chance of missing a relevant article cannot be reduced to zero, but it will become smaller when more time is invested in searching the database. Of course, a compromise will have to be made at a certain point.

In conclusion, the most important test instrument still remains a critical human brain: the person conducting the test should always be aware of the issues mentioned above and keep a critical mind while interpreting results.

Timetable and test results

Only comparative tests are discussed in this section. During the continuous test phase, a number of tests have been conducted that have resulted in solving a bug in some prototype. These tests have will not be discussed here, only comparative results for the debugged version are considered.

- *Full test 1*: The first test with all three prototypes together. XTM-R was tested a bit later, because of some bug that had to be solved.
    - RDF version 07 (2003-07-07)
    - SOAP version 1.1 (2003-07-07)
    - XTM-R version 1.3 (2003-08-07)
- *SOAP test*: Comparison between old, slow SOAP prototype and the fast version with ranking.
    - SOAP version 1.1 (2003-07-07)
    - SOAP version 2.0 (2003-09-23)
- *XTM JDBC test:* Comparison between regular XTM and the same XTM prototype, but with a new JDBC driver.
    - XTM-RQA version 1.3 (2003-09-11)
    - XTM-RQA version 1.4 (2003-09-12)
- *XTM AKE test:* Comparison with new keywords provided by UPM.
    - XTM-R version 1.3 (2003-08-07)
    - XTM-R version 1.5 (2003-10-16)
- *XTM stemming test:* Comparison of different stemming variants.
    - XTM-R version 1.5 (2003-10-28)
    - XTM-RA version 1.5 (2003-10-28)
    - XTM-RQ version 1.5 (2003-10-28)
    - XTM-RQA version 1.5(2003-10-28)

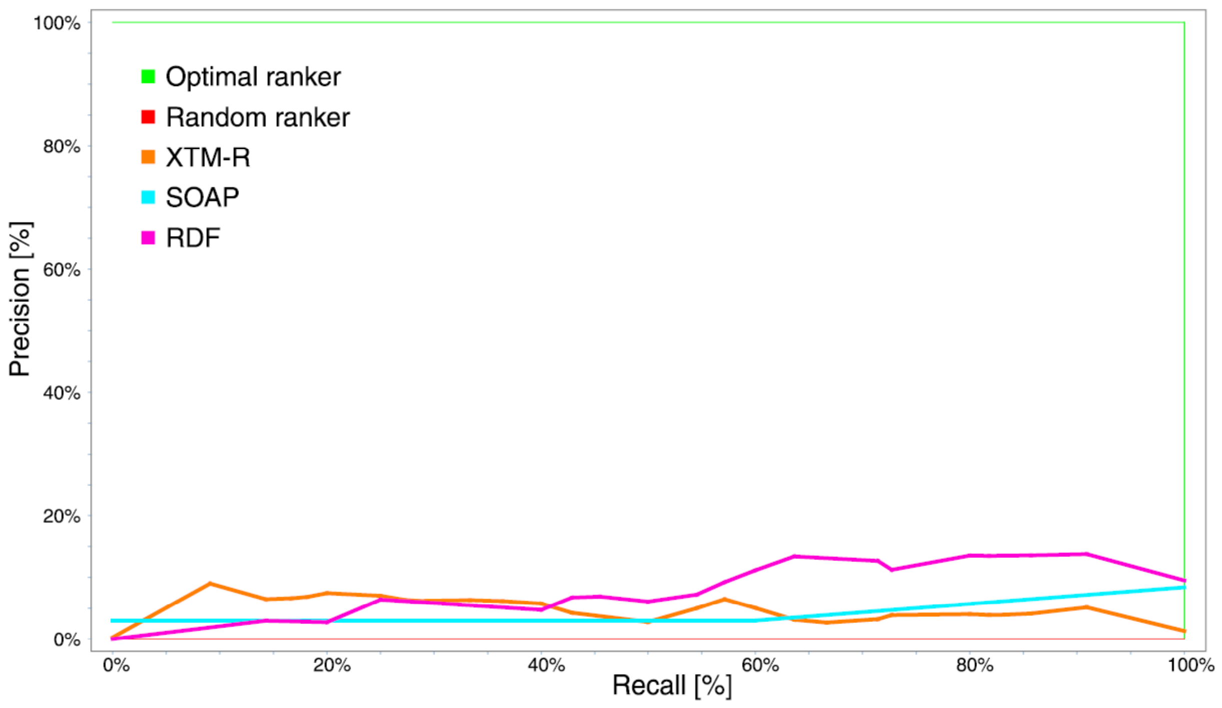
Full test 1

Figure 39 shows the recall-precision and time graphs of the three main prototypes. They are both averaged curves, taking into account 45 predefined queries clustered into 18 topics. It can be noticed that the precision for all prototypes is very low. This is partly due to the fact that all prototypes have the tendency to return lots of results, of which only a small fraction is relevant. However, for a good ranking algorithm, it is expected that the top results contain a considerable amount of relevant results. This means that high precision for low recall values should be seen on the graph. A quick inspection of Figure 39 (top) learns that this is not the case at all. This fact must be interpreted with care however: a more detailed study of the results on a per query basis shows that there exists a great variety in the quality of the results. For RDF 22 out of 45 queries do not return any results at all, for SOAP 20 out of 45 and for XTM 14 out of 45. These results have a considerable influence on the averaged recall-precision graph. The fact that XTM has less queries that have no results, is partly due to its stemming and to the fact that words found in the topic map will be expanded to synonymous words that do return results. If this is indeed the case, then one could wonder if the WordNet approach does have any effect at all, because overall results don't seem to improve for XTM.

Figure 39 (bottom) shows that the processing time for SOAP is an order of magnitude larger than for the other prototypes. This is because the old version of the SOAP prototype fetches the results one by one, thus sending a SOAP message for each result. This is also the reason why the number of results was restricted to 100.

Another strange consideration is that SOAP itself does not perform any better than the other prototypes, although SOAP does full-text search and thus should be more complete. Even more, in order to create the manual results sets for the predefined queries, a full-text search engine has been used, so it seems strange that is does not perform better. A explanation for this behaviour is that the SOAP prototype is a classifier, which means that all results are needed in order to come up with a reliable summary of the accuracy. As mentioned above, this was not the case as only a maximum of 100 results was allowed. A second reason will be explained in the next section.

Figure 39: Recall-precision diagram (top), Time graph (bottom) of XTM, RDF and SOAP prototypes



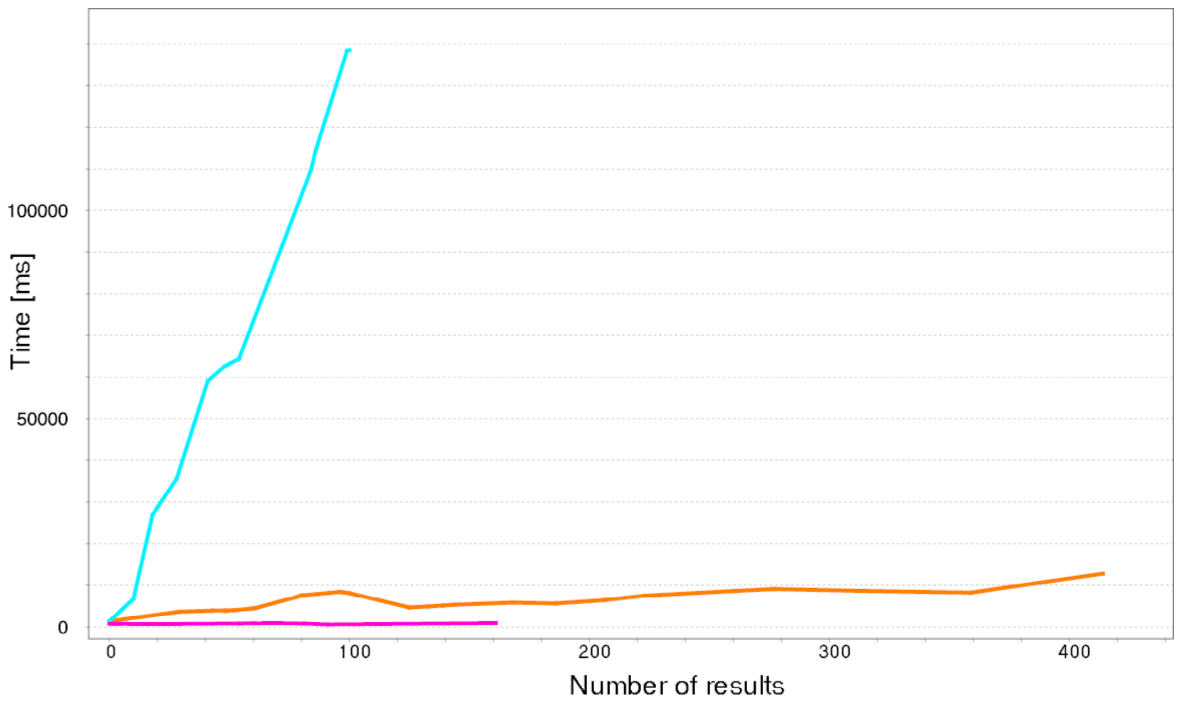
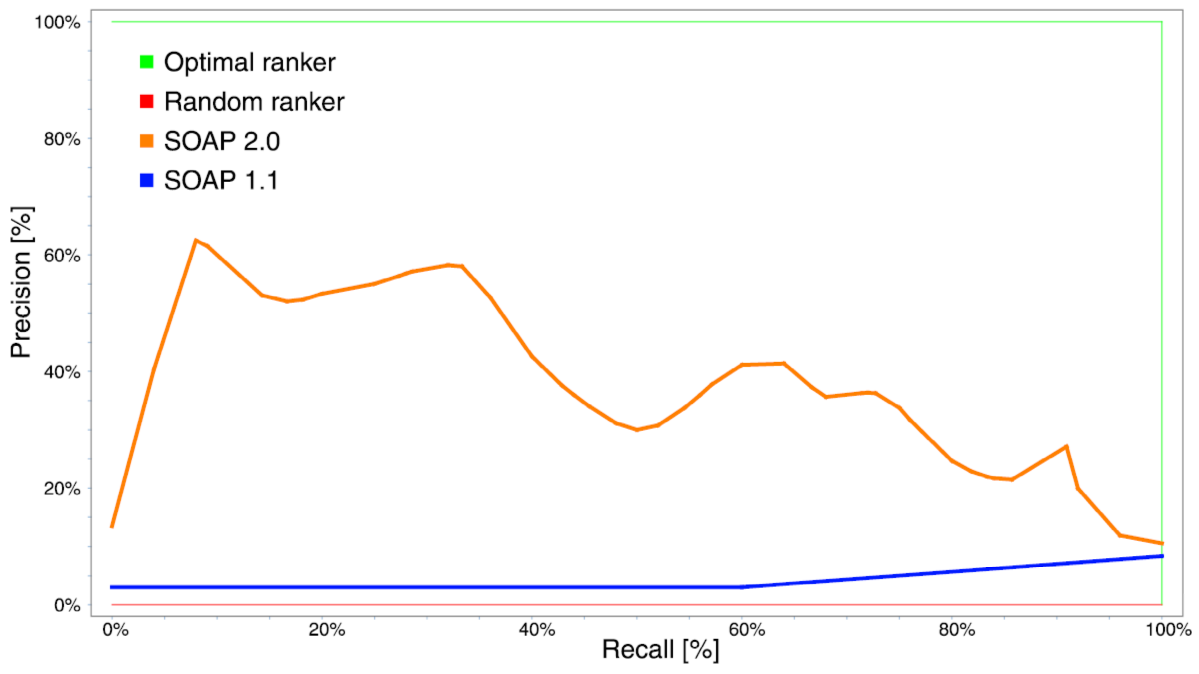
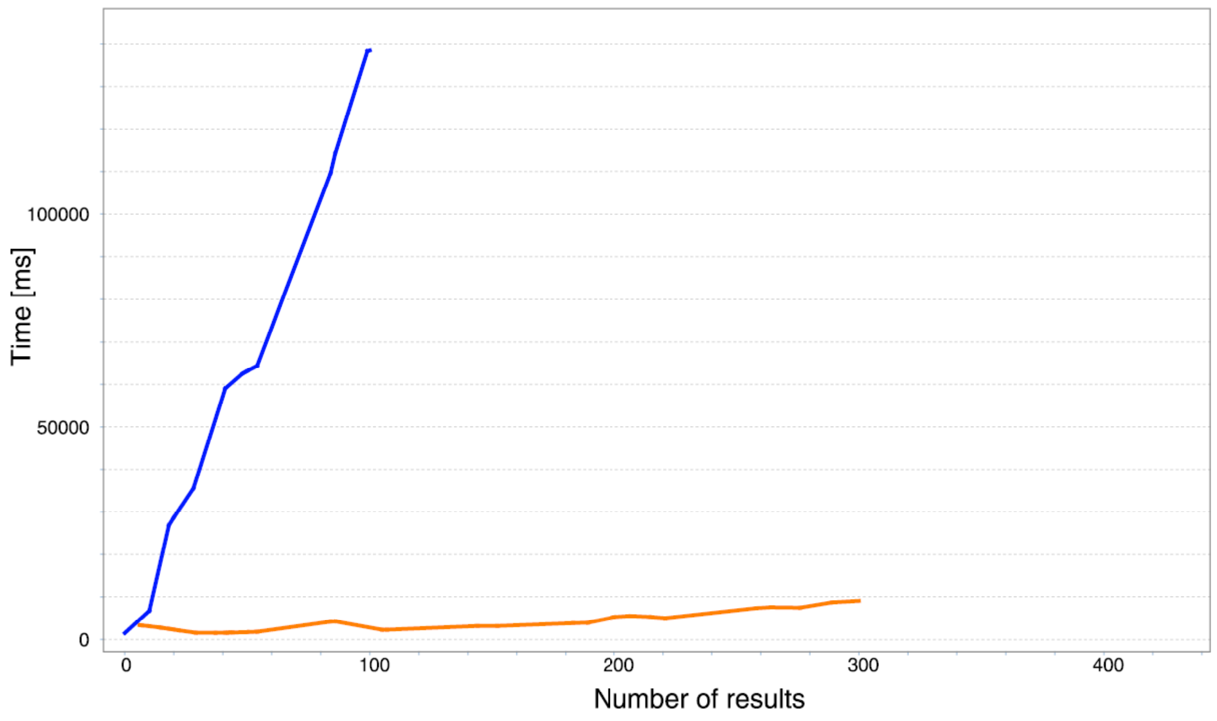


Figure 40: Recall-precision (top) and processing time graphs (bottom) for SOAP prototypes.





SOAP test

In order to increase the speed of the SOAP prototype, the definition of the SOAP messages has been redefined so that bulk messages are allowed now. This means that only one SOAP message, containing all results is returned now, which has a great impact on the processing time. As can be seen in Figure 40, the processing time for the new SOAP prototype is reduced to the same order of magnitude as the RDF and XTM prototypes.

- When using SOAP for distributed information retrieval, make sure the number of exchanged SOAP messages is as low as possible.

Another innovation in version 2 of the SOAP prototype is the introduction of ranking. The ranking algorithm of the full text search engine of the article database (Authonomy db) has been used for this. This has an enormous effect on the recall-precision, not only because the ranking algorithm works better than the original classifier, but also because this search engine has been used to find the manual result sets in the first place. Nevertheless, this curve should be the benchmark for the OmniPaper prototypes although we are not comparing them with pure full-text search anymore, but with the search engine if the Authonomy database.

XTM JDBC test

At the development stage of the XTM prototype, not much attention has been paid to the time efficiency of the algorithms. This has lead to the consequence that the XTM prototype is somewhat slower than the SOAP prototype at this moment. This test shows that an update of the third-party JDBC driver used to connect to the topic map database has lead to a 15% gain in processing time (see Figure 41). Also other database optimizations have been considered and will be implemented in the final WP2 prototype, with an expected gain of another 30%.

XTM AKE test

This test shows the improvement of the accuracy of the XTM prototype due to a new version of the AKE algorithm (see Figure 42). The average F-value has doubled to about 23% and only 6 queries do not return any results anymore. This graph shows that the quality of the XTM prototype depends on all parts of the search process. The success of the approach is highly, but not exclusively dependent on the quality of the AKE process.

- When keywords are used for searching, the quality of the automatic keyword extraction process has an important influence on the quality of the search system.

Figure 41: Processing time for new and old JDBC driver

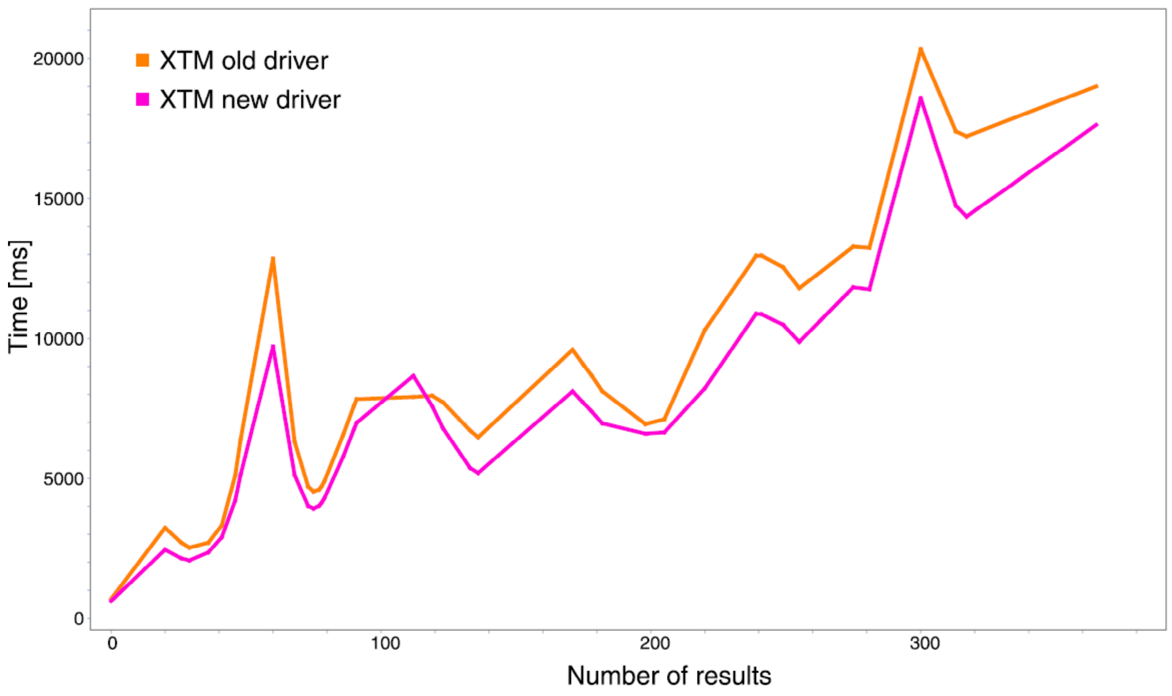
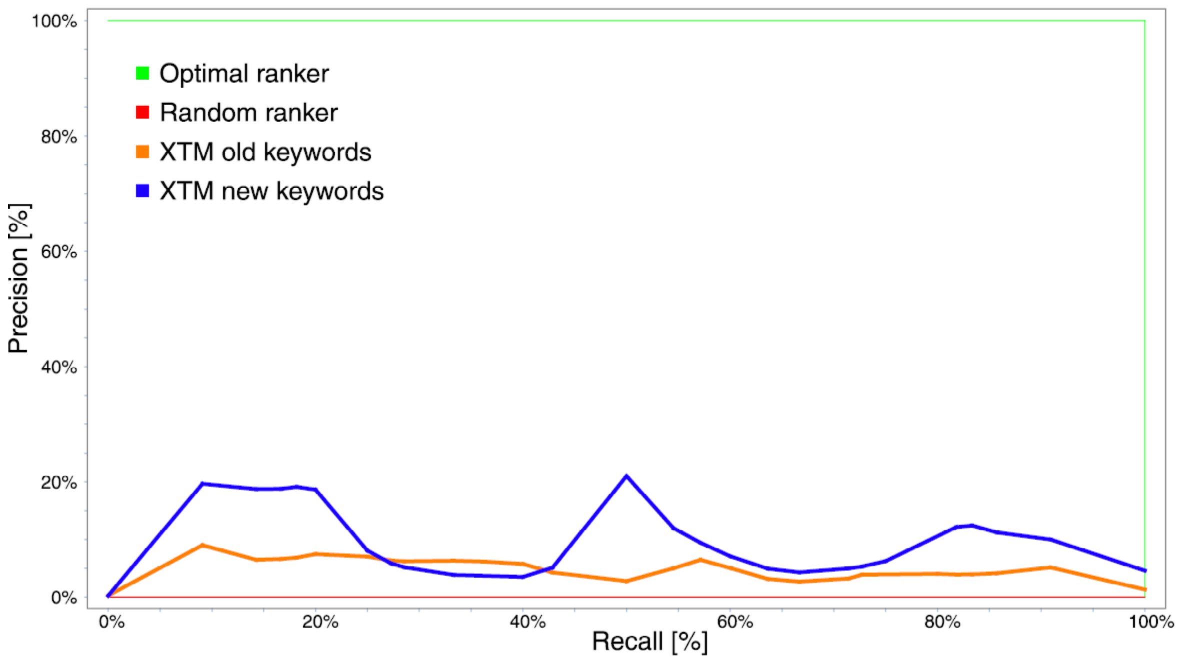


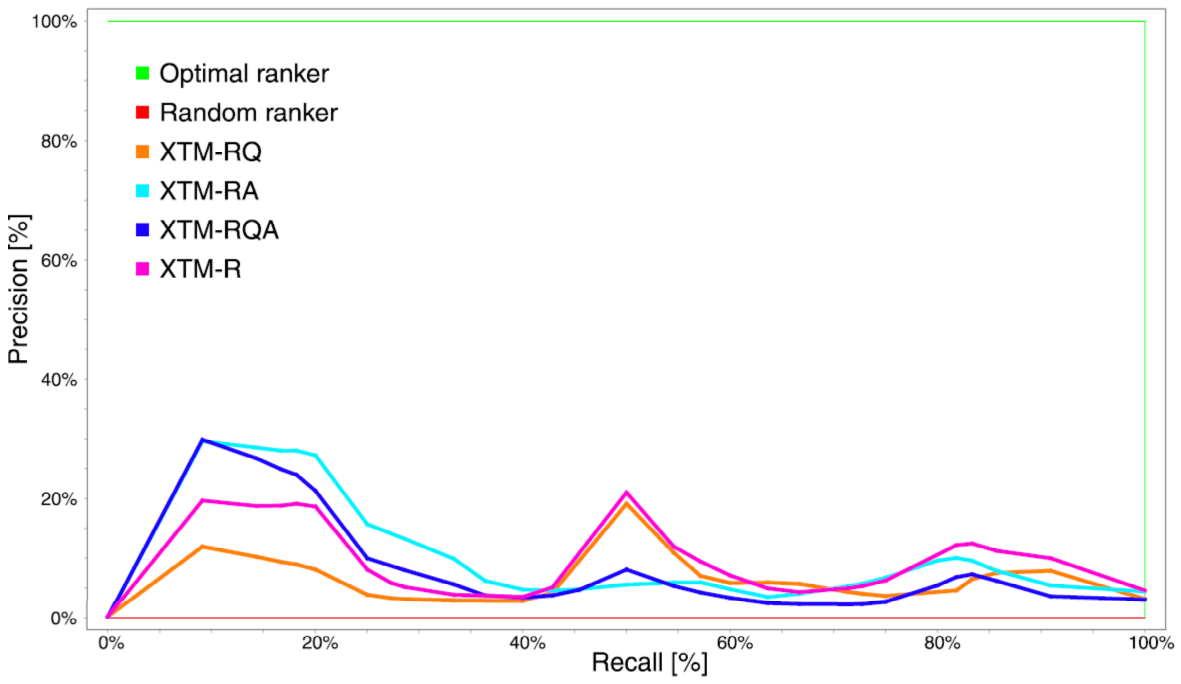
Figure 42: Recall-precision graph for XTM prototype with old and new keywords



XTM stemming test

Figure 43 shows the effect of the different variants of stemming. XTM-R stands for no stemming at all, the characters A end Q denote stemming of the extracted article keywords and stemming of the query words respectively. It can be seen that stemming of the article keywords alone leads to a small gain, but the variability in the different topics and queries to too high to judge the significance of this increase.

Figure 43: Recall-precision graph for different stemming variants of the XTM prototype



B.2.5.2 WP2: AKE subsystem

Purpose

AKE Subsystem has been tested using a news collection supplied by content providers involved in OmniPaper project. This collection is formed by 1.881 English news articles published during Sept. 2002. Each article is described in XML, including some metadata and the average document length (stopwords removed) is 250 words.

There are two basic configuration parameters that must be taken into account for test and evaluation purposes. These parameters are:

❒ *Stemming:* If stemming is applied words will be represented by a canonical form, identified by the word stem. This would lead to a grouping of words into the same representative stem, reducing dictionary size.

- *Frequency Thresholds (FT):* As mentioned in previous sections, keywords are selected according to the word DF into the whole collection. FTs are crucial in determining vector dimension and keywords quality. Tests ran for evaluating the system consider the following FTs:
  - *5% - 90% of total documents in collection*. This threshold has been established considering empirical results obtained by previous research in the field. This previous works stated that these Frequency Thresholds are dependent on the document collection being studied, but a good starting point would be a threshold between 10% and 90%. In this experimental work, a minimum Frequency Threshold of 10% turned to be inefficient because some documents were not assigned any keyword.
  - *0% - 100% of total documents in collection*. In this test no keyword selection is applied according to DF. It will be interesting to evaluate differences in processing times in the future.
  - *5% - 100% of total documents in collection.*
  - 0% - 90% of total documents in collection.

Set-up

Tests have been ran over a computer with an Intel Pentium III 800 MHz processor with 256 MB of RAM

Test results

Results of performance evaluation obtained for the different executions carried out with the system are summarised in Table 1. Results show some interesting facts:

- When applying stemming, the total number of dictionary entries is considerably reduced, as well as processing time. Considering execution times and storage requirements, stemming leads to a more efficient system. It will be necessary to prove if the values of traditional quality measures for Information Retrieval systems, like recall and precision, are better when stemming is applied.

- Stemming process leads to a greater number of keywords per document but a smaller number of total keywords in the document collection. This effect is due to variations in words frequency distribution. With stemming, words are grouped under the same stem, so that more stems surpass minimum FT than simple words in the no stemming approach. However, there are less stems in total than simple words.

- The more restrictive FT is Minimum Frequency Threshold. If figures from 5% - 90% and 5% - 100% thresholds are compared, differences in average keywords per document and total keywords in collection are very similar. On the contrary, comparison between 5% - 90% and 0% - 90% thresholds show greater differences. These results could point out some deficiencies on tokenization process, as a lot of different words are appearing in very few documents. This point will be checked according to word lists obtained from the tokenizer.

- For the AKE search prototype stemming leads to a more efficient system.
- Recall and precision are better when stemming is applied.

Stemming Applied					
Frequency Threshold	Aver.Keyw. per Docs.	Total Keyw. in collection	Total Dic. Entries	Processing time (s)	
				Indexing Time	Vector Construc.
5% - 90%	72,4003	726	29.684	179	111
0% - 100%	167,7363	29.684	29.684	172	161
5% - 100%	73,3615	727	29.684	177	112
0% - 90%	166,7751	29.683	29.684	199	170
Stemming NOT Applied					
Frequency Threshold	Aver. Keyw. per Doc.	Total Keyw. in collection	Total Dic. Entries	Processing time (s)	
				Indexing Time	Vector Construc.
5% - 90%	50,0138	537	42.265	199	104
0% - 100%	178,1637	42.265	42.265	208	172
5% - 100%	50,9761	538	42.265	199	105
0% - 90%	177,2015	42.264	42.265	200	173

Table 1. Basic experimental results for Keyword Extraction Subsystem

B.2.5.3 WP2: RDF prototype testing

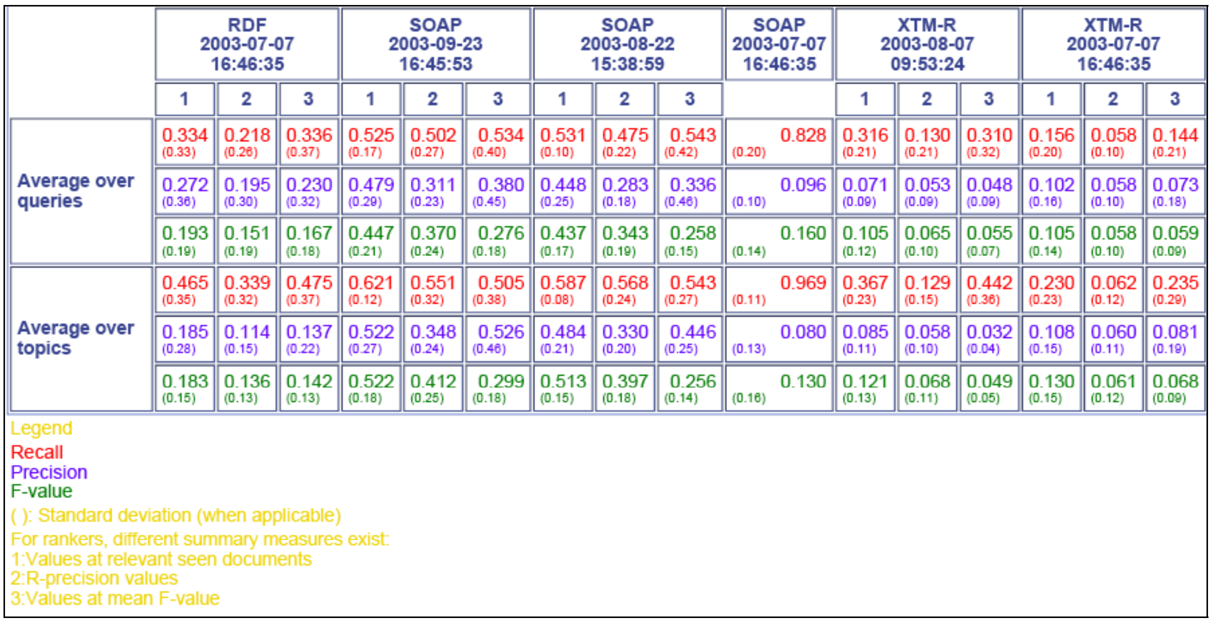
Relevancy

Figure 44 shows the Relevancy test results for all prototypes built within WP2. Summarizing:

a. Regarding both precision and recall, the RDF prototype behaves better than XTM in all tests.

b. Regarding precision, the RDF prototype behaves better than the first SOAP prototype and worse than the other two.

c. Regarding recall, the RDF prototype behaves worse than all SOAP prototypes.



	RDF 2003-07-07 16:46:35			SOAP 2003-09-23 16:45:53			SOAP 2003-08-22 15:38:59			SOAP 2003-07-07 16:46:35	XTM-R 2003-08-07 09:53:24			XTM-R 2003-07-07 16:46:35		
	1	2	3	1	2	3	1	2	3		1	2	3	1	2	3
Average over queries	0.334 (0.33)	0.218 (0.26)	0.336 (0.37)	0.525 (0.17)	0.502 (0.27)	0.534 (0.40)	0.531 (0.10)	0.475 (0.22)	0.543 (0.42)	0.828 (0.20)	0.316 (0.21)	0.130 (0.21)	0.310 (0.32)	0.156 (0.20)	0.058 (0.10)	0.144 (0.21)
	0.272 (0.36)	0.195 (0.30)	0.230 (0.32)	0.479 (0.29)	0.311 (0.23)	0.380 (0.45)	0.448 (0.25)	0.283 (0.18)	0.336 (0.46)	0.096 (0.10)	0.071 (0.09)	0.053 (0.09)	0.048 (0.09)	0.102 (0.16)	0.058 (0.10)	0.073 (0.18)
	0.193 (0.19)	0.151 (0.19)	0.167 (0.18)	0.447 (0.21)	0.370 (0.24)	0.276 (0.18)	0.437 (0.17)	0.343 (0.19)	0.258 (0.15)	0.160 (0.14)	0.105 (0.12)	0.065 (0.10)	0.055 (0.07)	0.105 (0.14)	0.058 (0.10)	0.059 (0.09)
Average over topics	0.465 (0.35)	0.339 (0.32)	0.475 (0.37)	0.621 (0.12)	0.551 (0.32)	0.505 (0.38)	0.587 (0.08)	0.568 (0.24)	0.543 (0.27)	0.969 (0.11)	0.367 (0.23)	0.129 (0.15)	0.442 (0.36)	0.230 (0.23)	0.062 (0.12)	0.235 (0.29)
	0.185 (0.28)	0.114 (0.15)	0.137 (0.22)	0.522 (0.27)	0.348 (0.24)	0.526 (0.46)	0.484 (0.21)	0.330 (0.20)	0.446 (0.25)	0.080 (0.13)	0.085 (0.11)	0.058 (0.10)	0.032 (0.04)	0.108 (0.15)	0.060 (0.11)	0.081 (0.19)
	0.183 (0.15)	0.136 (0.13)	0.142 (0.13)	0.522 (0.18)	0.412 (0.25)	0.299 (0.18)	0.513 (0.15)	0.397 (0.18)	0.256 (0.14)	0.130 (0.16)	0.121 (0.13)	0.068 (0.11)	0.049 (0.05)	0.130 (0.15)	0.061 (0.12)	0.068 (0.09)

Legend
Recall
Precision
F-value
( ): Standard deviation (when applicable)
For rankers, different summary measures exist:
1:Values at relevant seen documents
2:R-precision values
3:Values at mean F-value

Figure 44 - Relevancy tests' results for RDF, SOAP and XTM prototypes

Regarding precision, once that:

the RDF prototype only uses the 20 most relevant keywords in a text

the precision formula is

$$\text{Precision} = \frac{\text{Number of relevant documents retrieved (Ra)}}{\text{Total number of documents retrieved (A)}}$$

– This ratio should be high (very close to 1, if not 1). Being low, means the prototype retrieved many documents that were not relevant for the query. But, as only 20 keywords are used per document, this seems to mean that those keywords were not the more relevant ones.

Regarding recall, once that:

the RDF prototype only uses the 20 most relevant keywords in a text

the recall formula is

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved (Ra)}}{\text{Number of relevant documents in collection (R)}}$$

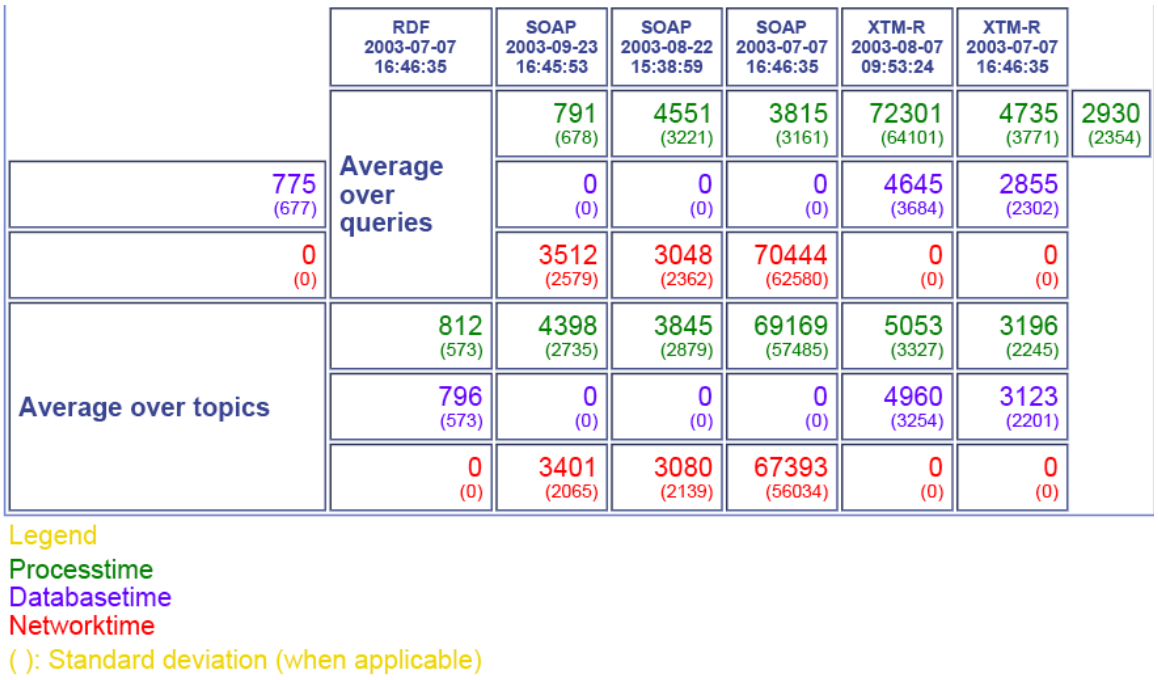
The fact that the ratio is low means that many of the relevant documents were not retrieved by the system. Taking into account the prototype looks for keywords in the key-list metadata element, this means the keywords entered in the query were not found (and should have) in the key-list metadata element for many relevant documents.

This analysis points to the fact that either the relevancy of keywords (and its ranking) for each article is not as well calculated, as it would be needed or that the number of keywords (20) is not the optimal number, or both.

Thus, if the more relevant keywords for a document are well calculated and if an optimal number of keywords is found, this prototype might improve its performance in relevancy terms in a very significant way.

Timing

Figure 45 shows the response time for each prototype. Network time will not be taken into account on this analysis.



		RDF 2003-07-07 16:46:35	SOAP 2003-09-23 16:45:53	SOAP 2003-08-22 15:38:59	SOAP 2003-07-07 16:46:35	XTM-R 2003-08-07 09:53:24	XTM-R 2003-07-07 16:46:35
Average over queries		791 (678)	4551 (3221)	3815 (3161)	72301 (64101)	4735 (3771)	2930 (2354)
		775 (677)	0 (0)	0 (0)	0 (0)	4645 (3684)	2855 (2302)
		0 (0)	3512 (2579)	3048 (2362)	70444 (62580)	0 (0)	0 (0)
Average over topics		812 (573)	4398 (2735)	3845 (2879)	69169 (57485)	5053 (3327)	3196 (2245)
		796 (573)	0 (0)	0 (0)	0 (0)	4960 (3254)	3123 (2201)
		0 (0)	3401 (2065)	3080 (2139)	67393 (56034)	0 (0)	0 (0)

Legend
Processtime
Databasetime
Networktime
( ): Standard deviation (when applicable)

Figure 45 - Timing tests' results for RDF, SOAP and XTM prototypes

Table 3 shows the total timing numbers (not including network time) for all prototypes. The RDF prototype behaves much better than any of the others, spending about 1/3 of the time of the last SOAP prototype and 1% of the time of the first one.

Table 3 - Total timing tests' results for each prototype

	RDF 2003-07-07	SOAP 2003-09-23	SOAP 2003-08-22	SOAP 2003-07-07	XTM-R 2003-08-07	XTM-R 2003-07-07
Average over queries	1566	4551	3815	72301	9380	5785
Average over topics	1608	4398	3845	69169	10013	6319

B.2.5.4 WP3 prototype testing

Purpose

The goal of the WP3 prototype tests is to assess the performance of the query expansion system as compared to: 1) the original SOAP search without the expansion and 2) the XTM and RDF search prototypes developed in WP2.

Comparison criteria are the recall and precision rates, the ranking behaviour (recall/precision ratio) and search time.

Set-up

For these tests the Automatic Testing Engine is used, which was originally developed in WP2. This engine has been integrated to work with the WP3 prototype. As document set the same set of 1881 English articles was used, to allow comparison with the other test results. Also the test set created in WP2, containing queries and reference answer sets, was reused. This means that for all prototypes (both from WP2 and WP3) the same dataset (WP2) is used. This is very important for allowing conclusions.

The tests were run in Leuven (on 5 October 2004) and the document set was served by the SOAP server of My News using their underlying Autonomy search engine and the WSDL version 2.2 (which is also used by the WP3 prototype). Queries are only put in English and not translated to other language, as the document set is also monolingual English. So this test does not evaluate the multilingual search performance of the prototype.

Test results

Test 1: WP3 variants with different MaxNbResults

The first test is a cross-comparison of different variants of the WP3 prototype. Each variant uses a different value for MaxNbResults, which means that this is the maximum number of results that each archive is allowed to return. If this value is 20 for example, each archive only returns the 20 most relevant search results. “Most relevant” here means to the current query and in comparison to their own archive. Tested values for MaxNbResults are 10, 25, 50, 100, 200 and 300.

Relevancy

This table shows that a higher number of retrieved results steadily increases the recall rate, but decreases the precision, which can be expected. The F-value is a measure for the optimal combination between recall and precision. This table shows that the best F-values are obtained at values of 25 and 50 MaxNbResults.

Table 4: relevancy summary for WP3 test[4]

	WP3	**WP3**	**WP3**	**WP3**	**WP3**	**WP3**
MaxNbResults:	**10**	**25**	**50**	**100**	**200**	**300**
Recall	0.28	0.39	0.43	0.54	0.60	0.61

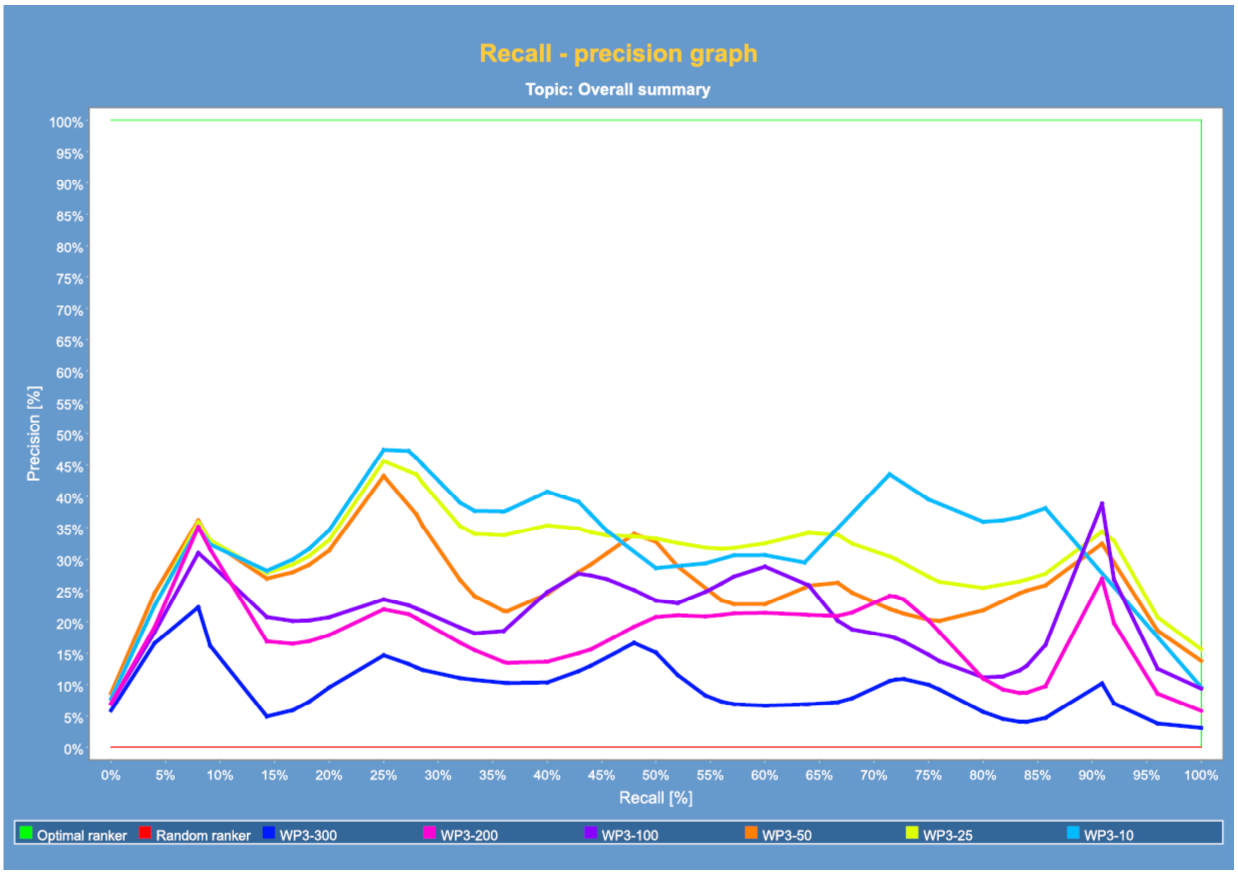
[4] values are averages over queries and values at relevant seen documents

Precision	0.46	0.39	0.36	0.27	0.20	0.16
F-value	0.32	0.35	0.35	0.28	0.21	0.17

- The higher the maximum number of retrieved results, the higher the recall rate, but the lower the precision.
- For the OmniPaper search system the best relevance was obtained with maximum number of results of 25 to 50.

Ranking (recall/precision)

Figure 46: recall/precision graph for WP3 prototype variants



The figure above shows the recall / precision comparison for the WP3 prototype variants with different MaxNbResults values. For this test set it shows that the ranking behaviour is not substantially different in quality across the different MaxNbResults values. For none of the values of MaxNbResults the precision decreases substantially towards the higher recall values.

➢ Limiting the maximum number of retrieved results does not have a substantial influence on the ranking behaviour

Timing

Database time is the time needed by the database to process and respond to SQL statements. The database in WP3 only consists of the EWN ontology; therefore it is only used for semantically expanding the queries. Network time is all time needed between sending out a SOAP request and receiving a SOAP response. So in practice this is the sum of 1) the network travel time of the request, 2) the time needed for the archive to respond and 3) the network travel time of the response. Process time is the total time minus the database time and the network time. In practice this means the time needed by the java web application to process the query and the results.

Table 5: timing summary for WP3 test[5]

	WP3	WP3	WP3	WP3	WP3	WP3
MaxNbResults:	**10**	**25**	**50**	**100**	**200**	**300**
Database time (s)	2.62	2.52	2.47	2.48	2.47	2.4
Network time (s)	1.5	1.71	1.94	1.99	2.82	4.15
Process time (s)	0.37	0.46	0.56	0.88	1.42	1.11
Total time (s)	**4.49**	**4.69**	**4.97**	**5.35**	**6.71**	**7.66**

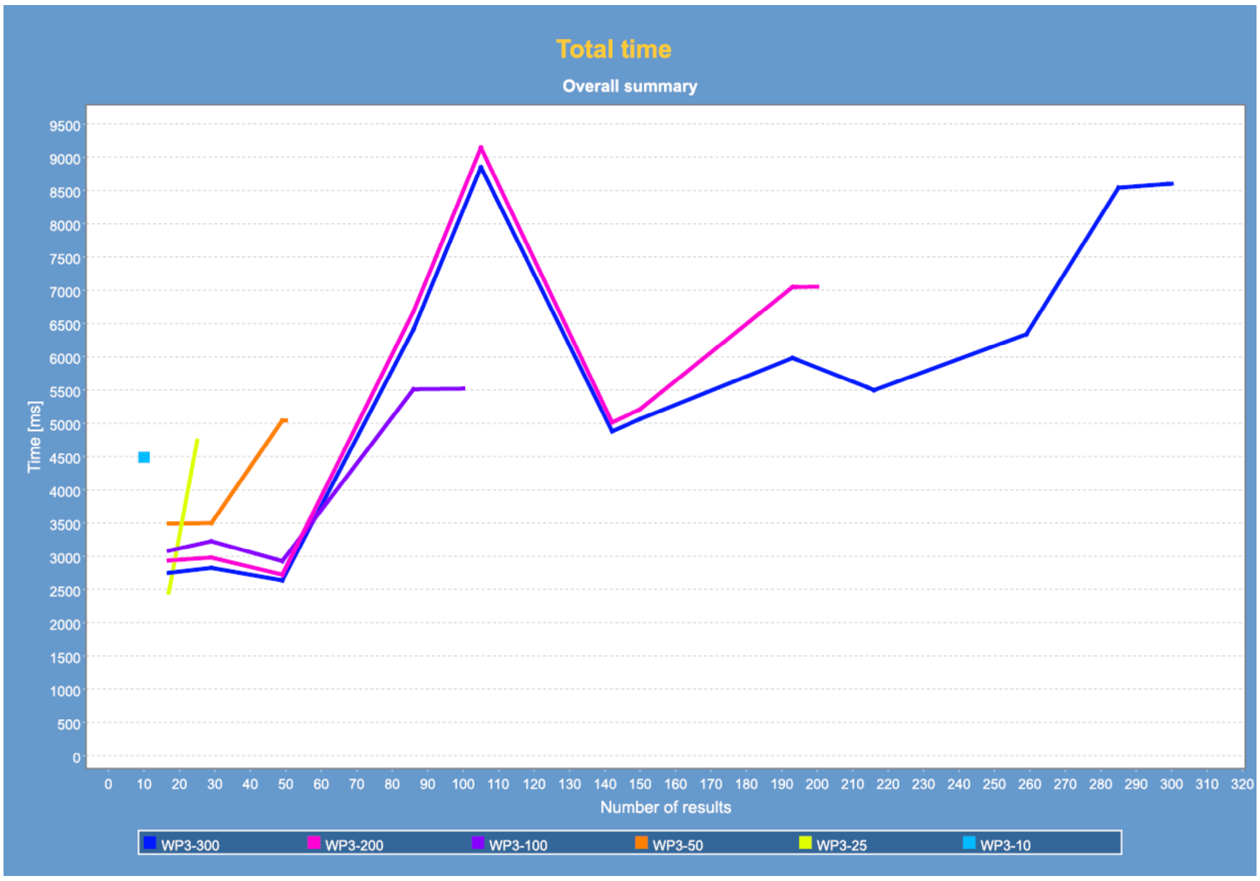
As can be expected the average total time needed for each query depends on the number of results returned: the more results the more time needed to gather them. The table above shows that this is mostly caused by the network delay: when more results are returned (by SOAP) the SOAP response becomes larger and thus it needs more time to be transferred over the network. Also the local archives need more time to respond. Secondly the process time also increases together with a higher number of results, because more results have to be processed. The database time does not differ significantly because the same database actions are done each time.

- The total time needed to process a query depends on the number of results returned: the more results the more time needed to gather them.

The graph below shows the total time needed in relation to the number of search results. The striking rise at about 100 search results should be interpreted as a coincidence related to the rather small size of the test set. In general terms the progress of the total time in terms of the number of search results is a quite linear increase.

[5] values are averages over queries / scale = seconds

Figure 47: timing graph for WP3 prototype variants



Test 2: SOAP, RDF, XTM, WP3

This second test is a cross-comparison of the following prototypes:

→ WP2 SOAP prototype, test ran on 2003-09-23 at 16:45:53

→ WP2 RDF prototype, test ran on 2003-07-07 at 16:46:35

→ WP2 Topic Map prototype (version XTM-RA) test ran on 2003-10-28 at 16:40:17

→ WP3 prototype (using MaxNbResults = 25), test ran on 2004-10-05 at 14:12:32

Relevancy

The table below shows that keyword-based search (RDF and Topic Maps) in general has poor results in recall. The reason is that a lot of results are not retrieved because only extracted keywords can be found. So there is a high dependence on the keyword extraction performance. For the Topic Maps prototype also the influence of the used WordNet ontology is important: only articles are found that 1) have relevant extracted keywords and 2) have keywords that are known in WordNet.

In this test full-text search (SOAP) is performing best, both in terms of recall and precision. This fact led to the decision to use SOAP (with query expansion) in the WP3 prototype.

Table 6: relevancy summary for SOAP, RDF, Topic Maps and WP3 test[6]

	WP2 SOAP	WP2 RDF	WP2 Topic Maps	WP3-25	WP3-300
Recall	0.53	0.33	0.36	0.39	0.61
Precision	0.48	0.27	0.17	0.39	0.16
F-value	0.45	0.19	0.20	0.36	0.17

When we compare the performance of the SOAP prototype (fulltext seach without query expansion) with the performance of the WP3 prototype variants (fulltext search with semantic query expansion), the following conclusions can be made:

– At lower MaxNbResults values, semantic query expansion does *not* make full-text search engines perform better in terms of recall and precision. Both recall, precision and F-value are lower in the WP3 prototype with MaxNbResults = 25.

– At the highest MaxNbResults value (which is better to compare the WP3 prototype with the SOAP prototype because the SOAP prototype did not use a MaxNbResults parameter: it received *all* results), a different situation can be seen. The table shows a higher recall for the WP3-300 prototype variant than for the SOAP prototype. However the average precision is substantially lower for the WP3-300 prototype because the query expansion generates too much noise (non-relevant results). This causes also the F-value to be very low.

➢ Ontology-based search scores much lower on both recall and precision than pure full text search

➢ Keyword-based search in general has poor results in recall

➢ The search performance of this kind of search method relies greatly on the quality of the keywords used

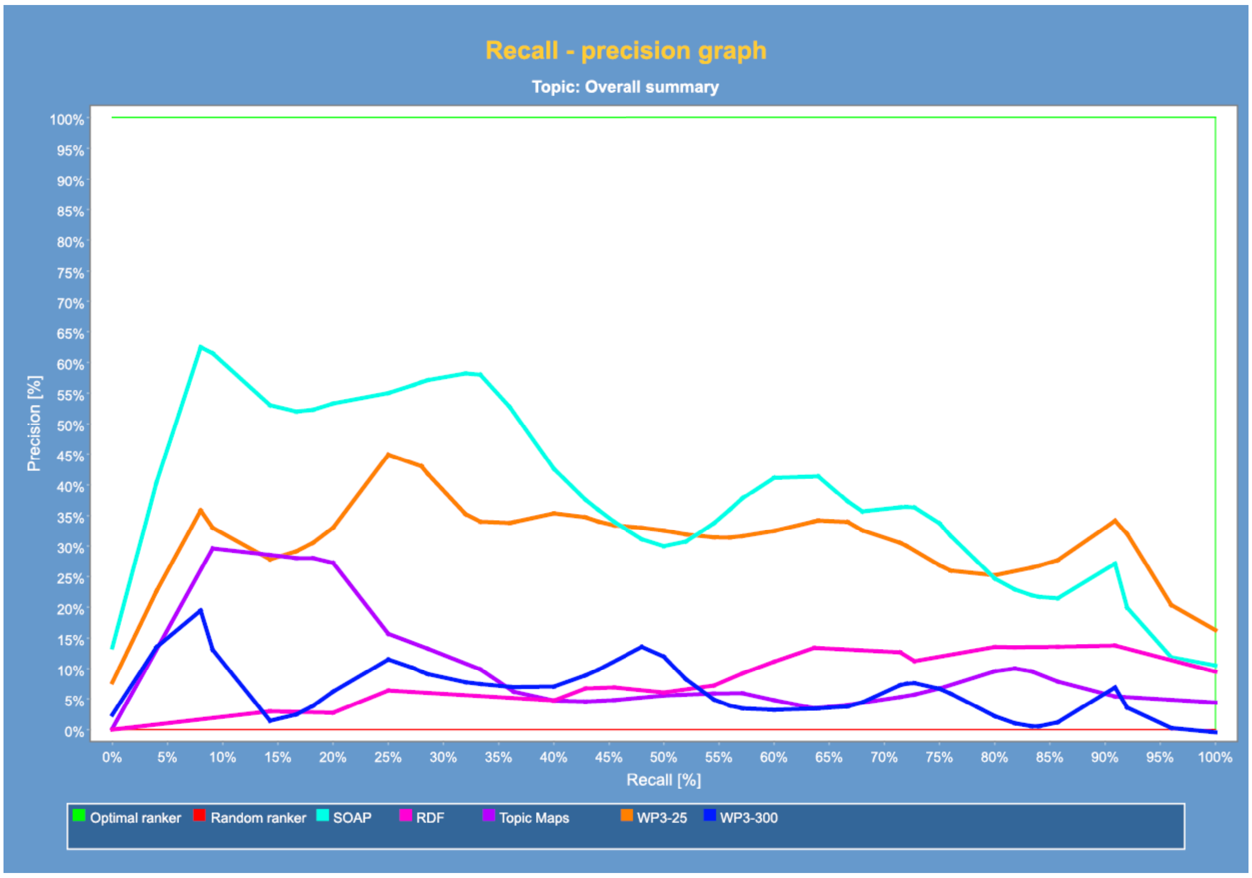
➢ Semantic query expansion slightly increases recall but dramatically decreases precision because of the *noise* generated by the expansion

Ranking (recall/precision)

In relation to this last conclusion the ranking behaviour of the different prototypes is shown in the graph below. A comparison between the SOAP prototype and the two WP3 prototype variants shows that semantic query expansion does not make full-text search engines perform better in terms of ranking. For both WP3 prototype variants the precision remains practically equal across the different recall values.

Another interesting fact is that the Topic Map prototype performs quite well in terms of ranking, knowing that this prototype uses the same semantic query expansion algorithm as the WP3 prototype. The only difference is that the Topic Map prototype uses the extracted keywords and weights for calculating the relevance rate. This might mean that the AKE system helps in improving the ranking of search results.

[6] values are averages over queries and values at relevant seen documents



- Semantic query expansion does not make full-text search engines perform better in terms of ranking
- The use of keyword weights in keyword-based search systems can improve the quality of relevance ranking

Timing

Table 7: timing summary for SOAP-RDF-Topic Maps-WP3 test[7]

	WP2 SOAP	WP2 RDF	WP2 Topic Maps	WP3-25	WP3-300
Database time (s)	0	0.78	2.30	2.55	2.4
Network time (s)	3.40	0	0	2.00	4.15
Process time (s)	1.15	0.01	0.06	0.55	1.11
Total time (s)	**4.55**	**0.79**	**2.36**	**4.90**	**7.66**

The table above allows the following conclusions:

[7] values are averages over queries / scale = seconds

- The RDF prototype performs best in terms of total response time. It stores all metadata of all articles in the data set in a native RDF database. This database system keeps a lot of information in the memory of the server; therefore querying is very fast.
- The use of SOAP (in SOAP and WP3 prototypes) increases the total response time due to network traffic and network delay.
- The more results returned by SOAP the more network time is needed.
- The use of a linguistic database for query expansion (in the Topic Map and WP3 prototypes) increases the database time because of heavy query operations (with lots of joins) needed to do the query expansion.

➢ Keyword-based distributed information retrieval systems have the lowest response time if a central metadata database is used.

➢ The use of a linguistic database for query expansion increases the time needed for retrieving query results.

B.2.5.5 WP5: Final prototype testing

This section describes the tests performed on the final prototype. The test data set used is the WP2 data set, but the search system is that of the final prototype.

Goal of these tests is to assess what influence semantic query expansion and AKE ranking have on the recall and precision of the search engine. Four prototype variants will be cross-compared:

→ Final prototype with semantic query expansion ON and AKE ranking OFF

→ Final prototype with semantic query expansion OFF and AKE ranking ON

→ Final prototype with semantic query expansion ON and AKE ranking ON

→ Final prototype with semantic query expansion OFF and AKE ranking OFF

At the time of writing this document the tests have not been finished yet. Test results and evaluation will be included in a later update of this document.

B.2.6 User evaluation

Two of OmniPaper's prototypes have been evaluated by end users: the Distributed Information Retrieval Prototype and the Final prototype for smart access to European newspapers. The evaluation results of both evaluation actions are included in the deliverable D6.2 Evaluation and Demonstration Report, of which a public version is available on the OmniPaper website.

B.2.7 Summary overview of search methods

In the different prototypes a number of search methods have been developed and cross-compared:

- **Full-text search**: searches for the exact words, appearing in the query, in the full contents of the documents. This method is used by the WP2 SOAP prototype and is also adopted in the later prototypes developed in WP3 and WP5. This method can be combined with Semantic query expansion and/or AKE ranking, as has been done in the final WP5 prototype.
- **Ontology-based search**: searches for the exact words, appearing in the query, and also for related words in a network of words and semantic relations (coming from WordNet or EuroWordNet). The

links between the words in the ontology and the underlying documents are based on the AKE process. This search method is used in the WP2 XTM prototype and in the combined WP2 prototype.

- **Keyword search**: searches for the exact words, appearing in the query, in a list of the top N keywords from the underlying documents. These keywords are provided by the AKE process. This search method is used in the WP2 RDF prototype.

- **Vector Space Model**: creates a vector of keywords, appearing in the query, and matches this with a vector of keywords, extracted from the underlying documents using the AKE process. This search method is used in the AKE search prototype, developed in WP2 and WP3.

Next to these search methods also some supplementary features have been developed and tested:

- **Semantic query expansion**: this feature expands the words appearing in the query to semantically related words. Semantic relations are taken from the linguistic resource WordNet or EuroWordNet. This feature has been used in many prototypes: WP2 XTM, WP2 combined, WP3 and WP5. While in the WP2 prototypes it was used in combination with the ontology search method, in WP3 and WP5 it has been combined with full-text search (query words are replaced by a set of related words, combined with the OR-operator).

- **Query translation**: this feature also uses relations from the EuroWordNet linguistic database, but this time to translate query words to related words in other languages. It is used by the WP3 and WP5 prototypes to allow cross-language information retrieval.

- **Stemming**: this feature reduces a word to its basic form (stem). It is used by the AKE process, by the WP2 XTM prototype and by the WP2 combined prototype. Both keywords, extracted from documents, and query words have been stemmed.

B.2.8 Summary overview of developed prototypes

Table 6 below gives an overview of all developed prototypes. It mentions for each prototype what kind of search method is used, how the query is being processed, in what part of the information the search is being performed (“search target”) and how relevance ranking is done. It further shows what resources are being used: (E)WN (WordNet or EuroWordNet linguistic database[8]), AKE (Automatic Keyword Extraction) and MDDB (MetaData DataBase). Finally it briefly summarises the advantages and disadvantages of each prototype.

[8] WordNet in case of WP2 prototypes (= monolingual English); EuroWordNet in case of WP3 prototype (= multilingual)

Table 8: Summary overview of developed prototypes

Prototype	Search method	processed query	search target	uses (E)WN	uses AKE	uses MDDB	Ranking	Advantages	Disadvantages
SOAP (WP2)	Fulltext search	original query	fulltext	no	no	no	depends on local archive ranking	- very good recall and precision (if underlying engine is also good)	- Different ranking methods cannot be combined - Different Search me-thods in the background - As slow as the slowest provider
RDF (WP2)	Keyword search using top N keywords	original query	Top N keywords	no	yes	yes	yes, based on keyword weights	- fastest prototype	- highly depending on the AKE performance
XTM (WP2)	keyword search with query expansion	expanded query	Top N keywords	yes	yes	yes	yes, based on extended Boolean model	- best response to user refinements	- poor recall and precision performance (only using top N keywords; completely depending on AKE for searching)
AKE search (WP2)	vector space model	query vector	article vectors	no	yes	yes			

Prototype	Search method	processed query	search target	uses (E)WN	uses AKE	uses MDDB	Ranking	Advantages	Disadvantages
AKE workbench (WP2)	This prototype is not a search prototype but an auxiliary prototype to the AKE prototype. As such it cannot be compared to the other prototypes.								
Distributed Information Retrieval Prototype (combined WP2)	keyword search with query expansion	expanded query	Top N keywords	yes	yes	yes	yes, based on extended Boolean model	- best response to user refinements	- poor recall and precision performance (only using top N keywords; completely depending on AKE for searching)
Overall Knowledge Layer Prototype (WP3)	fulltext + query expansion + query translation	expanded query	fulltext	yes	no	no	depends on local archives ranking mechanisms	- improved performance - can be combined with user feedback	- Different ranking methods cannot be combined - Different Search methods in the background - As slow as the slowest provider
Final Prototype (WP5)	fulltext + query expansion + query translation	original query OR expanded query	fulltext	yes	yes (for unified rankin g)	no	yes, using AKE	- improved performance - unified ranking - can be combined with user feedback	- Different Search methods in the background - As slow as the slowest provider

B.2.9 Open technical issues and possibilities for future work

This section covers a number of technical issues encountered during the project and that have fallen out of the project's scope. Some of them are interesting opportunities for future work in other research projects.

B.2.9.1 XTM format limitations

In one of the WP2 prototypes (Topic Map prototype) an XTM[9] information structure has been used for capturing the WordNet ontology containing keywords, concepts and semantic relations between concepts. This prototype used a pure ontology-based search algorithm:

- Each query term is looked up in the ontology (keywords)
- From the found keywords the corresponding concepts are looked up
- From the found concepts the related concepts are looked up (using the synonymous semantic relations)
- From all related concepts the keywords are looked up to find the related articles
- Articles are retrieved using a link between each keyword and its related news articles. This relationship between a news article is found using Automatic Keyword Extraction, a data mining process that extracts the most important keywords for each article and that attaches a weight factor to each keyword (more weight = more important keyword for that article). These weight factors are used by the prototype to calculate the relevance of an article for the current query.

During the development of this prototype initially the XTM format was used to capture the WordNet ontology. This means that the original WordNet source file (an ASCII text file using a WordNet-specific data format) has been transformed to an XTM file (using an intermediate XML file format to cover the differences in information structure between the WordNet ASCII format and the XTM format).

This XTM file was then loaded by the prototype software to get all the ontology information. During this process the software loads all the XTM data into in-memory objects. The problem there was that the XTM file was too large to be loaded into memory, because the total in-memory object would be 5-8 times the size of the original XTM file. Even with a Gigabyte memory server this led to memory overload problems. Therefore the development team has decided to create a relational database structure and to load all the WordNet information to this database. This means that the XTM format has not been used during the rest of the project; however the Topic Map paradigm has been adopted in this relational database until the end of the project.

➢ Topic Map structures can be easily stored using a relational database model.

➢ The use of the XTM format for capturing Topic Maps can become problematic when using large amounts of information like the WordNet ontology.

➢ Use a relational or other database when using large amounts of information that need to be captured in a Topic Map structure

[9] XML Topic Maps, see http://www.topicmaps.org/xtm/1.0/

B.2.9.2 Concept clustering

The notion of concept clustering came to the forefront during the conceptualisation of the ontology search prototypes (WP2 XTM, WP2 combined). Tests revealed that ontology-based search scores much lower on both recall and precision than pure full text search[10]. One of the main reasons for the low precision is the noise generated by the query expansion algorithm based on the ontology relationships. In the ontology search prototype articles are matched to user queries by comparison between the article's extracted *keywords* and the keywords from the query. Since an article can contain up to 100 keywords, it would be interesting if we could decrease the number of occurrences of an article in the ontology by combining information about the article's keywords into a reduced number of concepts.

Another consideration is that when articles are attached to keywords in the ontology, the user's impression of controlling the disambiguation is partly elusive, because the article keywords themselves are not disambiguated. For example, an article with the keyword `bush` might as well be about the president of the United States as about the vegetation in Australia. When the user selects one of the offered concepts (i.e. George W. Bush or the bushes), the system will behave smartly in the sense that it will also look for the other keywords that are expressions of the selected concept. For example, when user selects the president, the OmniPaper system will also look for articles containing the keywords `George Bush`, `President Bush`, `George Walker Bush`, `George W. Bush` and `Dubyuh`[11]. This expansion is a good feature, but it does not exclude the articles containing the word `bush` in the vegetative sense from the result list. This is where the concept clustering comes in: when articles are enriched with descriptive concepts instead of mere keywords, results will become more accurate because words that are used in the wrong context/sense can be filtered out now.

Technically this means that we would like to attach articles to concepts in the ontology instead of *keywords*.

During the project a first trial has been made for creating a concept clustering algorithm. This trial has been documented and supporting Java software has been developed. This concept clustering algorithm however is still in a very premature stadium. Lots of refinements will be necessary in order to make it really useful for search enhancement – refinements that fell outside the scope of the OmniPaper project. Therefore, the main conclusion is that it might still be too difficult to successfully implement such an algorithm. This is partly due to the fact that the quality of the results highly depends on the quality of the keyword extraction process, which is for the current purposes high enough, but maybe not for use in a concept clustering algorithm.

A second issue is the quality of the (Euro)WordNet from which candidate concepts are chosen: the associations that are contained in the WordNet are *semantic* associations. What would be more efficient for concept clustering is to have *contextual* associations that relate words that seem to co-occur often. This is because keyword disambiguation is in fact nothing more than the determination of the context in which the document is to be situated. Regardless of the previous considerations tough, some remarks on the current algorithm have been made already:

– In the weight normalization step, too much articles have weight 100%. Therefore, no differentiation can be made between the top keywords of a document. This problem can be solved by changing the properties of the model keyword.

– The algorithm should be forced to select only one of the possible concepts of a keywords. In other words disambiguation should be enforced. In the current version, it is often seen that the top concepts

[10] For the tests performed the recall was 0.53 for the full text search and 0.36 for the ontology based search. Precision was 0.48 for the full text search and 0.17 for the ontology based search.

[11] Dubyuh is the Texan pronunciation of the letter W, Bush's middle initial and is used to indicate the president in a pejorative sense. Try it in Google.

are in fact concepts that belong to the same keyword, so that the purpose of the clustering algorithm, disambiguating the article keywords in order to enhance the search results, cannot be fulfilled.

– There are lots of parameters to be tuned. This has not been investigated because the default values were chosen without any experimental backup.

– The algorithm uses only full keywords. It does not include the stemmed keywords. Stemmed keywords could lead to better results, but this would certainly require that more contextual information is available, because stemmed keywords generally have more candidate concepts that regular keywords.

– The algorithm is based on the supposition that a document's concepts are likely to be concepts of its top keywords or combinations of those. However, no attempt has been made to find candidate concepts that are combinations of top keywords.

– Although no special attention has been paid to performance, it can be foreseen that it could be very hard to efficiently speed up the clustering process. The main obstacle is the uniform cost search algorithm, which has an exponential time increase in function of the maximal cost allowed. Moreover, it requires lots of database actions for which it may be hard to define a efficient stored procedure.

All these considerations show that there is a lot of room for additional research on this matter.

➢ Clustering of keywords into concepts can probably increase the search quality of ontology-based search systems, but more research is needed to verify this.

B.2.9.3 SOAP speed optimisation

In the first SOAP prototype (WP2) the delays caused by the SOAP communication was a true bottle neck for the response time of the search system. Even after applying several major performance improvements to the SOAP interface definition and implementation, the delays caused by the SOAP communication was still responsible for 40% to 54% of the total time needed to process a search request.

Several improvements could be made to the SOAP communication between the central system and the distributed archives:

– When the current prototype requests information from multiple archives, it requests this information in an asynchronous way. This means that the second request is only sent after the first response has been received. In other words: the total time needed to gather a result is equal to the sum of the time needed by the different archives. With three archives this is does not have an unacceptable impact, but with more archives it will become a real problem. **Multi-threading** at the central server could help in solving this performance issue, by sending all information requests to the different archives simultaneously.

– Since the redefinition of the SOAP interface the number of exchanged SOAP messages has been reduced to an absolute minimum. Still the use of SOAP causes some overhead on the exchanged information; XML is known to increase redundancy (for example because of start and end tags). **Compression** of the exchanged SOAP messages could decrease the amount of bytes exchanged over the Internet and thus speed up the communication.

➢ When using a multi-archive information retrieval system with SOAP, synchronous sending of SOAP requests to the different archives will be the best solution in terms of speed.

➢ Compression of the exchanged SOAP messages can decrease the amount of bytes exchanged over the Internet and thus speed up the communication.

B.2.9.4 AKE workbench

The AKE (Automatic Keyword Extraction) workbench is a tool that allows manual verification of automatically extracted keywords and keyword vectors to improve the quality of the automatic keyword extraction processes as well as the overall information retrieval quality.

However the tool has been used by the researchers during development time to (manually) improve the AKE process, its original goal was more challenging. The original goal of this tool was the automatic improvement of the AKE process based on the manual verification of extracted keywords and keyword vectors. However, the AKE process has turned out not to be able to be tuned automatically based on this user input. This issue could be the topic of further investigation by other projects.

Also during the project's lifetime it became clear that the (daily) work of manually verifying automatically extracted keywords would create a heavy load on the costs of a future OmniPaper service.

Aside from these considerations it is clear that the AKE workbench also can be used in other content areas where keywords and keyword vectors can be automatically extracted from textual documents.

B.2.9.5 Prototype cross-testing

As documented in the Blueprint (D4.1) a lot of cross-tests have been conducted between different information retrieval prototypes and prototype variants. All these tests have been using the same test set and data set. As data set a collection of 1881 English news articles has been used (from The Daily Telegraph), all published in September 2002. As test set a manually created collection has been used containing about 45 queries and their reference answer sets. Since these identical sets have been used for the cross-tests, results are comparable.

Next to this testing effort the project has also been involved in the CLEF[12] work, mostly with the AKE search prototypes. The main advantages of this work are that a very big test set could be used (provided by CLEF), that this test set was multilingual and that the test results could be compared to search prototypes from other projects participating in CLEF. Thanks to the size of the test set scientifically sounder conclusions can be derived. A disadvantage for OmniPaper is that the test results obtained from the CLEF work cannot be compared with the test results described in the previous paragraph.

A discrepancy came to light between the different evaluation methods of information retrieval prototypes. While statistical tests proved that the use of an ontology did not increase the relevance of search results (neither in ontology search nor in full text search with semantic query expansion), the user feedback shows that many users find the ontology very helpful as a query refinement tool. Therefore the need for combined evaluation methods arises: methods that do not only take into account the objectively measurable statistical performance of an information retrieval process, but also the subjectively experienced user satisfaction with the information retrieval quality.

- ➢ The use of an ontology can be very helpful for users as a query refinement tool
- ➢ There is a need for combined information retrieval evaluation methods, taking into account not only the statistical performance, but also the user satisfaction.

B.2.9.6 Automatic Test Engine

The testing engine has been used to cross-test the various WP2 prototypes, some WP3 prototype variants and the WP5 (final) prototype. It could be used in many other projects, if adapted to their

[12] Cross Language Evaluation Forum, see http://clef.isti.cnr.it/

specific needs. It could even be developed further as a project on its own to become a tool used by many information retrieval projects.

The main goal of such a project would be to develop an open source platform for the automated assessment of the information retrieval performance of search engines. Open for use and further development to scholars, researchers and developers, this tool will facilitate and speed up the evaluation process of their information retrieval prototypes and software.

As has been shown during the last decade, the provision of massive data test sets has allowed information retrieval research to make rapid progress. Especially the existence of joint evaluation programmes, such as TREC, NTCIR and CLEF, has enabled rich comparisons between technologies, resulting in solid research results. Where evaluation methods and metrics are quite standard, no common *evaluation tool* exists yet. By providing a platform that can be used by researchers to evaluate their technologies using these existing test sets, this project could fill in this gap and thus help to further stimulate information retrieval research.

The platform should allow two ways of testing. First, the statistical testing module calculates well-known evaluation metrics such as recall, precision, F-value and search time. Second, the user evaluation module allows on line gathering and processing of user feedback using more “soft” metrics such as user satisfaction and usability.

Key features and advantages of the evaluation platform:

- ➢ Integrate statistical testing and user-based evaluation in such a way that the platform can automatically process big data sets as well as user feedback for testing the same prototype.
- ➢ Testing is faster than with using manual tools. Overnight testing is possible.
- ➢ Variants and versions of information retrieval software can be cross-evaluated quite easily, allowing more rapid development cycles.
- ➢ Test reports including tables and graphics can be generated automatically. Statistical results and summaries can be created on-the-fly.
- ➢ An on line testing environment allows collaborative testing by researchers from different disciplines and situated in multiple countries. A multi-platform off line variant is also part of the system.
- ➢ If the tool is used by a lot of researchers, test results from different researchers are easier to compare.

Research topics for this project could include:

- ➢ Research on existing evaluation methods and metrics for information retrieval (for both statistical and user-based testing).
- ➢ Comparison of statistical testing (the Cranfield paradigm) and user-based evaluation: advantages and drawbacks of each paradigm. Proposal of ways to overcome these drawbacks by combining both evaluation methods.
- ➢ Comparison of existing test data sets regarding their contents and structure.
- ➢ Development of a standard XML format for test data sets to be used by the platform.
- ➢ Definition of the platform architecture. This architecture must be flexible and easily adaptable so that new evaluation metrics can be included.
- ➢ Development of testing tool components and other auxiliary tools.
- ➢ Definition of an API for programmatic access to the components of these tools.

- ➢ For information retrieval projects the use of a test environment for automated cross-testing of search systems is essential.

B.2.9.7 Other content areas

Although a lot of efforts have been spent by the consortium to test the developed systems in other content areas, like in the CLEF work, additional work would be useful to test the system in very different areas.

One example is web search. The developed search system – using ontology-based semantic query expansion, query translation and uniform relevance ranking using automatic keyword extraction – could for example be tested with the Google SOAP interface. If a certain multilingual document set (for example a CLEF or TREC set) would be put online, a very interesting experiment could be conducted. The same set could be searched using the Google SOAP interface without any additions and using the same interface but with semantic query expansion and/or with AKE relevance ranking. Also a user interface could be created that allows plain Google searches but with the addition of a query refinement tool like in the final OmniPaper prototype. This work looks very interesting and promising, also to allow sounder statistical testing, but fell outside the scope of the OmniPaper project.

➢ The OmniPaper search system could be used on top of other existing search engines (full text or others).

B.2.9.8 Linguistic resources

As a linguistic resource enabling multilingual queries, EuroWordNet (EWN) has proven to be very useful for this project. Furthermore this resource has been used as a basis for semantic query expansion and query refinement.

During the project however also limitations of the current EuroWordNet database came to light that have had consequences on the project developments and that will have consequences on future exploitation:

– For the query disambiguation tool (SVG Web of Concepts) often EWN has proven to be much **too fine-grained**. Most concepts appearing in EWN have a very specific meaning, so that a lot of ambiguity appears when a users searches for a specific word. For example the English word "vote" has 7 possible meanings (=concepts) in EWN: *vote* as "a body of voters who have the same interests", *vote* as "the opinion of a group as determined by voting", *ballot* as "a choice that is made by voting", *franchise* as "a legal right guaranteed by the 15th amendment to the US constitution", *to cast a vote*, *to vote for* and *voter turnout* as "the total number of votes cast". However, for most queries in the area of news these kind of very specific meanings are not required at all. As a consequence the user is often confronted with too many words to choose from when he wants to disambiguate a query word using the Web of Concepts.

– As a possible solution for the previously mentioned problem the EWN **top concepts** or **top-level domains** could be used. Top concepts are a set of general concepts that exist in all languages and to which more specific concepts can be linked. Top-level domains are context areas in which a certain word can have a specific meaning. The OmniPaper Web of Concepts could use this feature for making query term disambiguation more user-friendly, by showing only the top concepts or the top-level domains related to the query term instead of all related concepts. For the query term *bush* for example, the current Web of Concepts shows 5 related concepts: *bush* as "a large wilderness area", *bush* as "provide with a bushing", *shrub* as "a low woody perennial plant usually having several major branches", *chaparral* as "dense vegetation consisting stunted trees or bushes" and *pubic hair*. If instead top-level domains would be used for example only domains like *nature* or *biology* would appear. For users this could be a much more usable way for disambiguating a query term. The problem is that EWN does not provide sufficient top level domains or top concepts to be useful for the project.

– Another problem lies within the nature of the EWN project. As a European research project it has been successful in the creation of a multilingual wordnet for many European languages. After its funded project phase however the **continuation** of the EWN work has been dispersed across the different participating organisations. As a consequence many parts of EWN have not been updated recently and the addition of other languages does not happen in a coordinated way. Since the area of news requires a very up-to-date linguistic resource, this approach does not suffice. Therefore the consortium pleads for a **sustained European funding effort** for creating an EWN-like linguistic resource for all European languages that could be used in many applications and research projects.

– EWN does not contain a lot of **proper names**. Only countries, major cities and main historic person names are included. Since queries for news very often contain proper names this gap in EWN is a real problem for a project like OmniPaper. If an additional linguistic resource would be available that contains up-to-date information about all kinds of proper names that can appear in the news, this resource could be used for query refinement, disambiguation and for solving the problem of misspelled names. The project consortium however has not been able to find such a resource.

➢ There is a need for a EWN-like linguistic resource for all European languages that could be used in many applications and research projects.

B.3 User Interface Guidelines

This section provides user interface guidelines for integrating the OmniPaper multilingual search into (general) web interfaces. These guidelines should help developers to design effective search interfaces that are useful, easy and pleasant to use. Special attention is paid to multilingual usage, which includes input as well as output in multiple languages.

This section considers different preconditions and requirements when integrating the search interface into existing or planned systems and user interfaces with different archives and design constraints.

In order to meet the various needs and preconditions, user interface guidelines are described for different user interface units or modules. Typically a search interface is composed of different “standard” modules that can be chosen and combined according to the respective conditions. Guidelines provide arguments and recommendations to help with the necessary decisions in the design process.

This section covers elements, wording and positioning of the proposed user interface modules as well as screen layouts for the arrangement of the modules. Queries, required user input, search fail strategies, and implications of multilingual search are discussed.

Wording suggestions are only given in English, but might also serve as examples for interfaces in other languages.

Also in this section, detailed Guidelines are marked within this text by a yellow background to support quick perception (light box if a black and white printer is used).

B.3.1 Characteristics of Multilingual Search

B.3.1.1 General

In a multilingual search, both input and output of a search can be in multiple languages. This brings considerable advantages for the users, but also challenges for the user interface design. For OmniPaper, multilingual search means:

- Articles in various languages can be searched.
- Users can submit queries in any supported language.

A multilingual search will most probably lead to different results than a search in only one language. Those results may partly be desired, partly not intended, but welcome, and partly unwanted.

- If the system is able to translate query phrases, the result may contain articles in various languages.
- Names and certain other query words (e.g. “computer”) may return results in different languages.
- Some queries may deliver results in more than one language with different meanings (e.g. the English word “gift” has a different meaning in German, and that is “poison”.) Therefore, parts of the results are not what the user intended to find.

B.3.1.2 Queries

To put the focus on the users, it is useful to look at the queries they may submit. A study by Jakob Nielsen revealed an average query length of 2.0 words, which corresponds to other studies [5]. According to Google Zeitgeist and Yahoo's Buzz Log[13], where you can see listings about popular search terms, the most frequent queries consist of one or two words. A large portion of those popular queries is made up of proper names.

The user's queries in OmniPaper may consist of:

- One or more words in one language.
- One word in different languages (e.g. "gift OR geschenk"). In this case it is clear that the user does not mean the German word "Gift". This information should be recognized by the system.
- More than one word in different languages. The user might want them to be translated and applied on single articles.
- A word meant in one language, not aware that this word has a different meaning in other languages.

B.3.1.3 Use Cases

An important question is what the users probably want to accomplish. Those cases should be detected and cared for by the system. Users probably want:

- Find articles in the same language as the query. This might be the case most frequently.
- Find articles in more languages, but put the query in only one language. This is a major advantage of a multilingual search that is capable of translating queries.
- Find articles in all languages they understand.
- Find articles in certain languages they understand.
- Find articles in a certain language, although the search phrase might exist in different languages.
- Find articles in one language, although the search phrase can be translated.

B.3.1.4 User Requirements

User interfaces for multilingual search should support the use cases mentioned above and help the users in accomplishing their tasks. As Hansen et al. [2] found in their field study, the following user requirements apply to cross-language information retrieval:

- Search multiple languages at the same time
- Change query language in same search session

[13] Googe Zeitgeist - Search patterns, trends, and surprises according to Google
http://www.google.com/press/zeitgeist.html
Yahoo's Buzz Log – Search Spikes and Trends http://buzz.yahoo.com/

- Support multilingual queries
- Support queries with compound names and phrases
- Use lexical and morphological tools, such as synonym lists
- Combine Boolean and ranked retrieval
- Filter results by language, genre, date, or other features
- Create user-specific dictionaries and term lists

Since the user's tasks can be considered to be an important factor for search behaviour, this question of task domains should also be integrated in our considerations about the Omnipaper search interface. Elaine G. Toms et al. investigated the effects of task domains on search [12]. The results were design requirements for the task domains consumer health, research, shopping and travel. For Omnipaper will mainly be used for research tasks, we will have a closer look at the design requirements for this domain in order to draw conclusions for the Omnipaper interface [12]:

a) the ability to specify the level of information (general overview, detailed, scientific)

b) the ability to specify desired information formats (journal articles, newspapers, statistics, etc.)

c) a quicker and more effective way to evaluate the content of a website from the hitlist

d) support for the research process of beginning from a general overview and working towards progressively more detailed information through progressive filtering

B.3.1.5 Tasks

In order to serve the user's needs as accurately as possible, it is necessary to know what the user really intends to find, which language his input is, what his native language is and which languages he understands. From the user's needs, the following tasks can be extracted, that have to be cared for by the system:

- Interpretation of the query input:
  Detection of the language of the search phrases, regarding language and meaning, in order to get the best results possible. Interpretation of multiple search phrases in one query. Those search phrases could be in different languages. Unintended results caused by different meanings of a phrase in different languages should be avoided. The language of the query can also serve as an indicator for the result display, because users mostly want results in the language of their query.

- Translation of the search phrase:
  When search phrases are translated into multiple languages, contrary to a simple full text search, multilingual archives can be searched with a query in only one language. The user does not need to care about language issues.

- Languages of results presented to the user:
  Users might probably not want to get results in languages they do not understand, but they also might want to get results in more than one language. They might want to get results in all languages they understand, or in certain languages of their choice.

Those tasks can partly be handled by the system behind the surface, without the user's notice. For other parts, the user's input is useful or necessary. Those requirements have to be taken into account when designing the user interface.

The decision of which choices should be taken by the system and which choices should be left to the user depends on the capabilities of the system, and the necessity or desirability for the user to decide and chose. For the user it is on the one hand an additional effort, on the other hand it might be desirable to control their search process and results.

- Interpretation of the query input:
  Concerning the query words, it is of advantage if the system tries to guess the language so that the user does not have to specify the language for each query. An input element may be provided for setting the general query language for a user or for optionally specifying the language for one certain query. However, if guessing the language of a query works well, there is no need to provide an optional input element for it.

- Translation of the search phrase:
  The translation of query is entirely up to the system.

- Languages of results presented to the user:
  For the results, it is possible to provide suitable default settings, incorporating all accessible information about the user, such as the language of the query, if known, the languages of former queries, browser language etc. Further details about estimating the user's preferred languages are discussed later.
  However, in order to meet the user's requirements exactly, it is necessary to provide language settings and filtering options.

B.3.1.6 Estimating Preferred Languages for Results

The user's settings provide exact information about his preferences, but if no user input is available, the system should be able to provide reasonable default settings for a certain user.

The query language, if identifiable, can serve as an indicator for the user's language preferences. Conclusions can also be drawn from former inputs of the user, browser versions and other sources.

In order to estimate the preferred languages, a reasonable combination of the following indicators should be found. We recommend applying the named criteria in the order shown below.

Preferred languages for articles in a certain search can be the languages:

1. of the **search phrase**, if known.
   However, the search phrase does not always allow conclusions about the preferred language, e.g. technical phrases or names ("computer", "gift", "George Bush")

2. of **former queries** submitted by a user, if they provide better information about the language than this query, provided that personalization is possible.

3. of the **web interface**, if this applies (results in this language exist). The language of the web interface can be different from the search language, but it can be changed by the user and is therefore a better indicator than the language of the user's system.

4. of the **user's browser**, if this applies (results in this language exist)
   The language of the user's system can be different from the user's native language.

5. of the **web site** containing the OmniPaper search. A majority of the users for instance of a Spanish news agency may be Spanish and prefer their results to be in Spanish.

6. with the **most results**. This is a weak indicator, and it is only available *after* submit. Therefore it can not be displayed in the Advanced Search form.

B.3.2 User Interface Modules

In this section we follow a modular approach to describe the parts of OmniPaper user interfaces. Four main modules can be identified:

- Simple Search
- Advanced Search
- Search Results
- Detailed Article Display

The minimum version of an OmniPaper search interface consists of one search module (Simple Search or Advanced Search) and the Search Results module. The Advanced Search module and the Search Results module consist of sub modules. Part of them are necessary, others are optional.

In the following chapters, those modules, elements and sub modules are described, and recommendations are given for building reasonable, functional and usable search interfaces. The guidelines also cover the positioning of modules and interrelation between those modules.

B.3.2.1 Simple Search

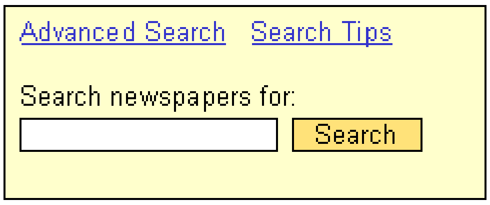


Figure 48: Example of a Simple Search module.

The Simple Search module provides easy access to search functionallity. The main focus should not be on a plentitude of search options, but on simplicity. Advanced features should be left to the Advanced Search module.

- Search should be available on every page.
- If an Advanced Search is provided, keep the Simple Search as simple as possible.
- Make sure that users have sufficient information about the search.
- Avoid confusions with a common site search.
- If it may be uncertain, which queries are supported, give users unobtrusive information about it.

Wherever the Omnipaper search is used, there must be the information that Omnipaper search is a multilingual search in newspapers. The users must know what they can search, what queries they can enter and what they will get. If this is not clear from the context, it should be communicated by a short description preceding the search elements. It is not sufficient to include an explanation on the “Search Help” page.

Elements of Simple Search

Simple Search should consist of the following elements:

- Query input field
- Label of input field
- Submit Button
- Link to Search Tips
- Link to Advanced Search

Optional elements:

- Link to language settings for results: “Set result languages”. A link to a language settings dialogue may be provided, if no separate module for language settings is offered. This option should be omitted if there is an Advanced Search with this option (see Language Selection for Simple Search).

➢ Abstain from providing a link to language settings in the Simple Search, if there is an Advanced Search that provides language settings.

Query Input Field

Length: Whereas in the pre-web era users were most likely to enter single-word queries, they use substantially more two-word queries and longer queries today. [6] Simple Search is mostly used for one or two word queries, but if the options are available, advanced users can also enter long queries using logical operators and other options.

➢ If there is enough space available, ensure that an average query is entirely visible in the input field.

➢ It is recommended to make it at least 25 characters wide [4], so the input will be entirely visible in most cases.

➢ The length of input should not be restricted.

Another positive effect of wide input fields is that they encourage users to enter more query words, which usually leads to more precise results [6].

Default: empty

Label of Input Field

A label of the input field is not obligatory, if the wording on the submit button gives enough information. Nevertheless, the label might give valuable information.

Wording:

➢ Use the most common wording in your language. (In English: “Search”)

➢ Make clear what the user can search. If any confusion with a common site-search or a search in the whole web is possible, it is recommended to chose an expression like “Search newspapers for”, "Search news archives for", "Search European news archives for" or similar, to avoid misunderstandings about the scope of the search.

- If an Advanced Search module is shown alternatively to the Simple Search module in the same place, use clear terms to distinguish them, e.g. "Simple Search" and "Advanced Search".
- If users change between search modules by selecting links, e.g. "Advanced Search" or "Simple Search", the titles of the referred modules should be conform.

- "**Search**", "**Find**":
  In English, the most common label for Simple Search fields is "Search". It is recommended to follow such wording conventions, because they operate as key words that catch the user's attention when scanning the site for a search function. Other wording might make it more difficult to find the search function quickly.

- "**Search newspapers for**", "**Search news for**", "**Search news archive for**":
  If any confusion with a common site-search or a search in the whole web is possible, it is recommended to chose an expression like "Search newspapers for", "Search news archives for", "Search European news archives for" or similar, to avoid misunderstandings about the scope of the search. This choice depends on the context.

- "**Simple Search**", "**Simple Search**":
  If an Advanced Search module is provided, it might be useful to distinguish the different search modules by using distinct terms like "Simple Search" or "Simple Search" and "Advanced Search".

  If users change between those search modules by selecting for example a "Simple Search" or an "Advanced Search" link, the title of the referred modules should be named accordingly to allow for a clear feedback to the user's action.

Submit Button

- Take care that the button looks like a button.
- Design it consistently with other buttons on the site.

Wording:

- Use the most common wording in your language. (In English: "Go")
- If the key word "Search" (or equivalent in other languages) is not used in the label, use it for the submit button.

It is recommended to use the term "Search" (or the equivalent key word in other languages) anywhere in the Simple Search module. If it is not used in the label (or there is no label for the input field), it should be used for the submit button. Otherwise, the most common wording for submit buttons of a Simple Search is "Go". It is recommended to follow such conventions.

Link to Advanced Search

- Unless the Advanced Search is rarely used, a link to it should be placed in the Simple Search module.

If an Advanced Search module is provided, a link to it could be included in the Simple Search module, placed in another position on the screen or offered in the results page.

If users are looking for Advanced Search, they will most probably find the Simple Search (that should be present on every screen) first, and then try to find a link to Advanced Search there. Therefore it is most probably the best way to place this link in the Simple Search module. On the other hand, each additional element makes the Simple Search more complicated. Therefore, it might be considered to offer it only on the results page. In this case, it is necessary to execute a Simple Search to reach the Advanced Search.

The decision where to place the Advanced Search link depends on the frequency of use. If the Advanced Search is often used, a direct and easy to find link to it should be provided. If it is rarely used, it might be offered only on the Search Results page.

Wording: “Advanced Search”

Adjustment of Elements

- ➢ When integrating new modules in your site, follow existing styles and conventions.
- ➢ Take care that elements are consistently designed.

The elements of the Simple Search module can be arranged in different ways. There might be conventions of an existing web site that have to be considered here, such as titles, title bars of modules, help buttons or links, etc.

Elements should be designed consistently, for instance all links should have the same styles and be clearly recognizable as links. The same applies to buttons, labels etc.

Label of input field:

- ➢ Labels should generally be placed on the left of the input field, following the reading direction (in western cultures).
- ➢ Longer labels may also be placed above the input field, left-aligned.

Submit button:

- ➢ The submit button should be right of the input field.

Links to Advanced Search, Search Help, Language Settings:

- ➢ Make sure that all links clearly look like links.
- ➢ Consider
- ➢ Group similar links to support a clear layout.

Those links could be presented in one line that might be placed:

- Above the search elements or
- Below the search elements

There might be other good ways to arrange those links. The link to search help, for instance, might be a button showing a question mark, that could be placed in the top right corner of the module, the link to Advanced Search might be placed on top, besides the title, while the language settings might be placed at the bottom of the module.

Positioning & Space

- ➢ Search should be available on every page.
- ➢ The Simple Search module should be placed in the upper part of the screen.
- ➢ If Advanced Search is displayed, Simple Search should be hidden to avoid confusions.

Generally, it is recommended to have a search module present on every page. In cases where the search is not of high importance (which is usually not true for Omnipaper search), there might be only a link to a search page.

If the Advanced Search module is displayed, the Simple Search module should be hidden to make clear where the user should enter his input, unless there are good reasons to keep it in place.

The Simple Search module should be placed in the upper part of the screen. There are two main alternatives:

- Simple Search is permanently visible in the same position on each page.
- Simple Search and Advanced Search are alternatively shown in the same place.

This decision depends on the screen layout, the space available for the display of search and result modules and the minimal dimensions of each module.

- ➢ If possible, provide Simple Search in a fixed position on every page.
- ➢ The best and most common place for a Simple Search is in the top right corner.

Simple Search may be integrated in the header of the web site, displayed in the content area or in a column beside the content area.

Header:

- ➢ If there is enough space to accommodate the Simple Search module, it should be placed in the header.
- ➢ In the usual cases where the logo is placed on the left of the header, the Simple Search module should be positioned on the right.

Content area:

The search module might be displayed in the content area when selected via a user interface element. This implies that an additional step is necessary for the user to start a search, which should be avoided.

- ➢ Avoid additional steps to reach the search, unless the search is very rarely used.

Column beside the content area:

- If displayed in a column beside the content area, the Simple Search module should be positioned in the upper part of the column in a prominent place.

Query

The user's query might consist of only one word, but Simple Search might offer a much wider range of options. Those options would be mainly used by expert and returning users of the search. Search help should be offered covering the use of those options.

- Simple Search should support queries containing:
- More than one word, separated by blanks: These should be connected with the operator AND.
- Phrase search: Phrases should be indicated using " " (e.g. "Robert Altman")
- Including and excluding via +, -
- Logical operators: Simple Search should support the input of logical operators AND, OR, NOT
- Additional options:
- Criteria like language, publisher (e.g. "lang:english")

B.3.2.2 Advanced Search

Advanced Search offers more search options than Simple Search. All possibilities that users have in Simple Search by entering logical operators or other options should be represented as form elements in Advanced Search, additional options may be offered.

Nevertheless, options should be presented in a way that users understand. Users should not be forced to choose between options they do not understand [4], for instance options related to differences in the internal search process.

We have to bear in mind that average users are poor at using advanced search or Boolean operators. Jakob Nielsen therefore recommends that advanced search should not be offered from the start page since users might use it wrong [5]. However, for Omnipaper's purpose advanced search will be of higher importance and usefulness than for an average web site, and advanced search options are very common for research in libraries and newspaper archives. Nevertheless, advanced search should not become a substitute for a simple search that does not deliver satisfying results.

Whereas Simple Search should be kept simple, Advanced Search should provide a sufficient set of options, so that users can determine their results as exactly as they want to.

- Options should be chosen to meet user requirements. Not every possible option is useful for the users.

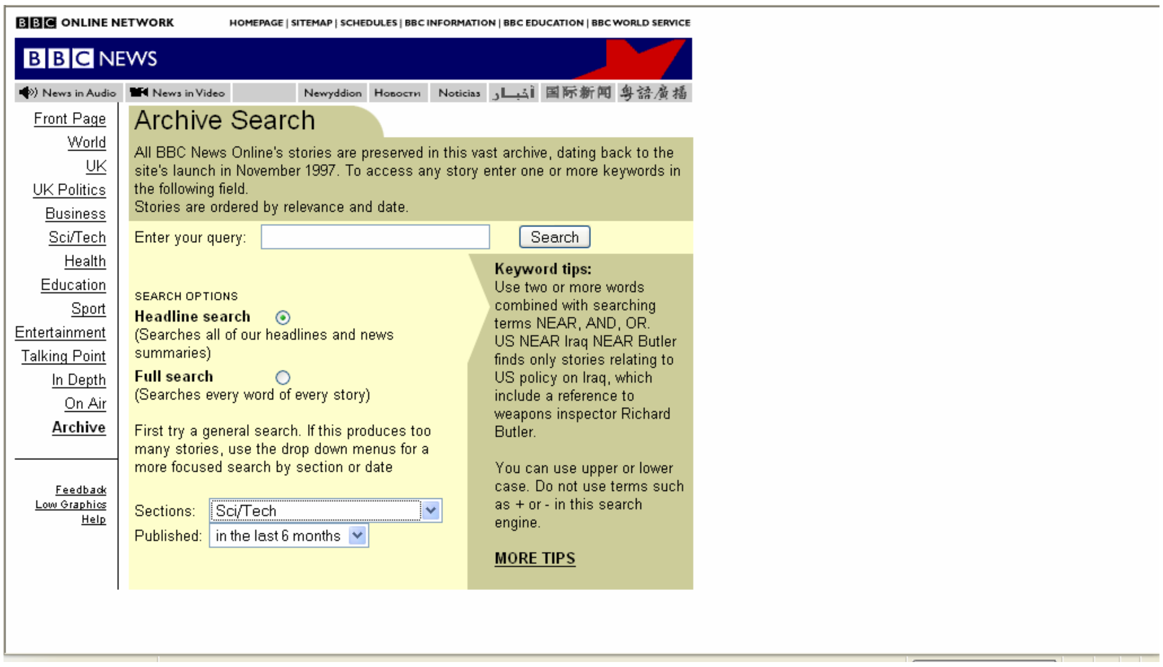
Examples

The following screenshots show examples for advanced search pages of news providers in order to give an insight and discuss their practise.

BBC News has a rather simple search with only a few, but well chosen options. The query term is entered into one single field, just as in a simple search. Additionally, users have the options to specify where the search terms should be contained (in the entire text or only in the headline). A section can be chosen (“World”, “UK”, “Business” etc.) and the timeliness can be specified by choosing options from a dropdown list (“in the last day”, “in the last week” etc.). The options cover the most important attributes for news search in a reasonable manner.

The default values are set to useful values: Sections is set to “Any”, Published is set to “in the last 6 months”. Although “entire archive” is also available, the limitation to 6 months is a good choice for news.

Advanced tips for logical operators are also provided on the same page. Only part of the searchers will use operators in their query, but since this is an advanced search and the information parts are set off from the functional parts by using a different background-colour, this practise works quite well. However, many users may not be able to understand and use the operators correctly, and a mask for entering the ALL-, ANY- etc. terms separately might support non-expert users better.



Archive Search

All BBC News Online's stories are preserved in this vast archive, dating back to the site's launch in November 1997. To access any story enter one or more keywords in the following field.
Stories are ordered by relevance and date.

Enter your query:

SEARCH OPTIONS

Headline search [✓]
(Searches all of our headlines and news summaries)

Full search
(Searches every word of every story)

First try a general search. If this produces too many stories, use the drop down menus for a more focused search by section or date

Sections: Sci/Tech
Published: in the last 6 months

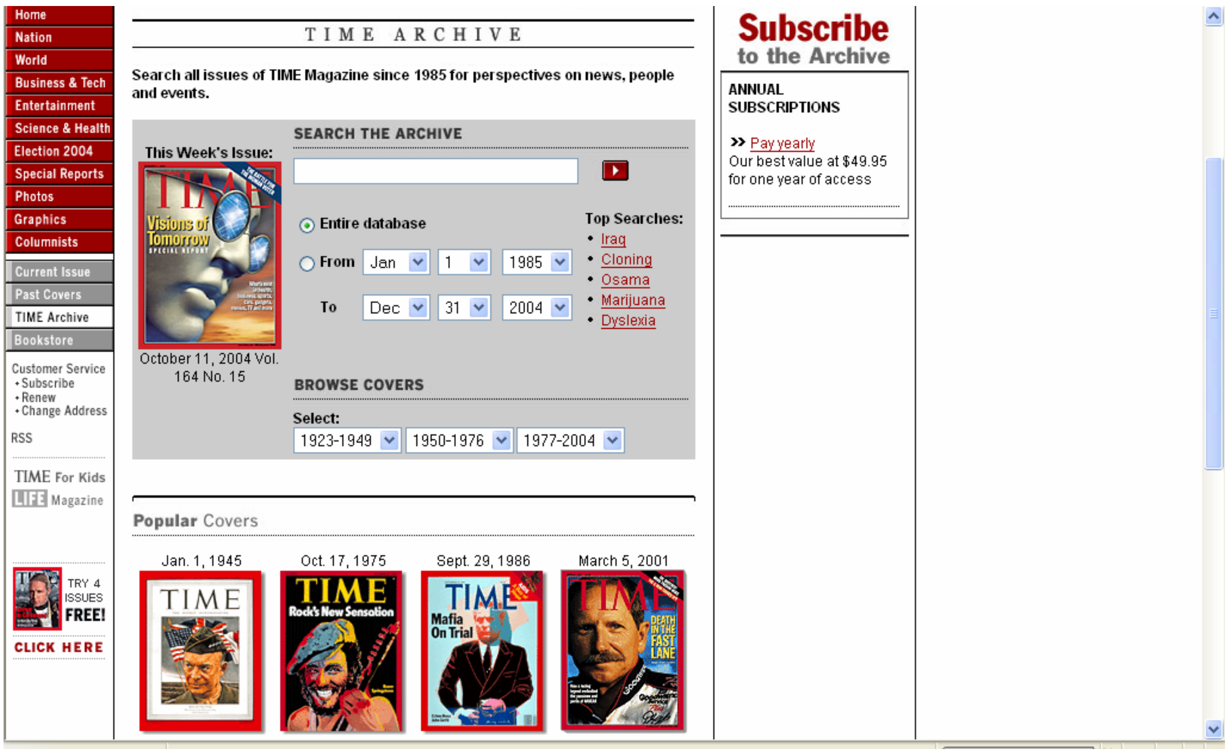
Keyword tips:
Use two or more words combined with searching terms NEAR, AND, OR.
US NEAR Iraq NEAR Butler finds only stories relating to US policy on Iraq, which include a reference to weapons inspector Richard Butler.

You can use upper or lower case. Do not use terms such as + or - in this search engine.

Figure 49: BBC News advanced search page [14]

The time magazine offers a more “popular”, entertainment-oriented search. The simple search is supplemented by an option to define a time-span and a list of the most popular query terms. People who search information on one of these popular terms may use those links without having to type their own query. Besides the archive search, users may also browse the magazine covers from 1923 on. The advanced search options may not be sufficient for expert use, but work well for average users.

[14] http://news.bbc.co.uk/hi/english/static/advquery/advquery.htm



TIME ARCHIVE

Search all issues of TIME Magazine since 1985 for perspectives on news, people and events.

This Week's Issue: October 11, 2004 Vol. 164 No. 15

SEARCH THE ARCHIVE

[✓ Entire database]

From Jan 1 1985

To Dec 31 2004

Top Searches: Iraq, Cloning, Osama, Marijuana, Dyslexia

BROWSE COVERS

Select: 1923-1949 1950-1976 1977-2004

Jan. 1, 1945 Oct. 17, 1975 Sept. 29, 1986 March 5, 2001

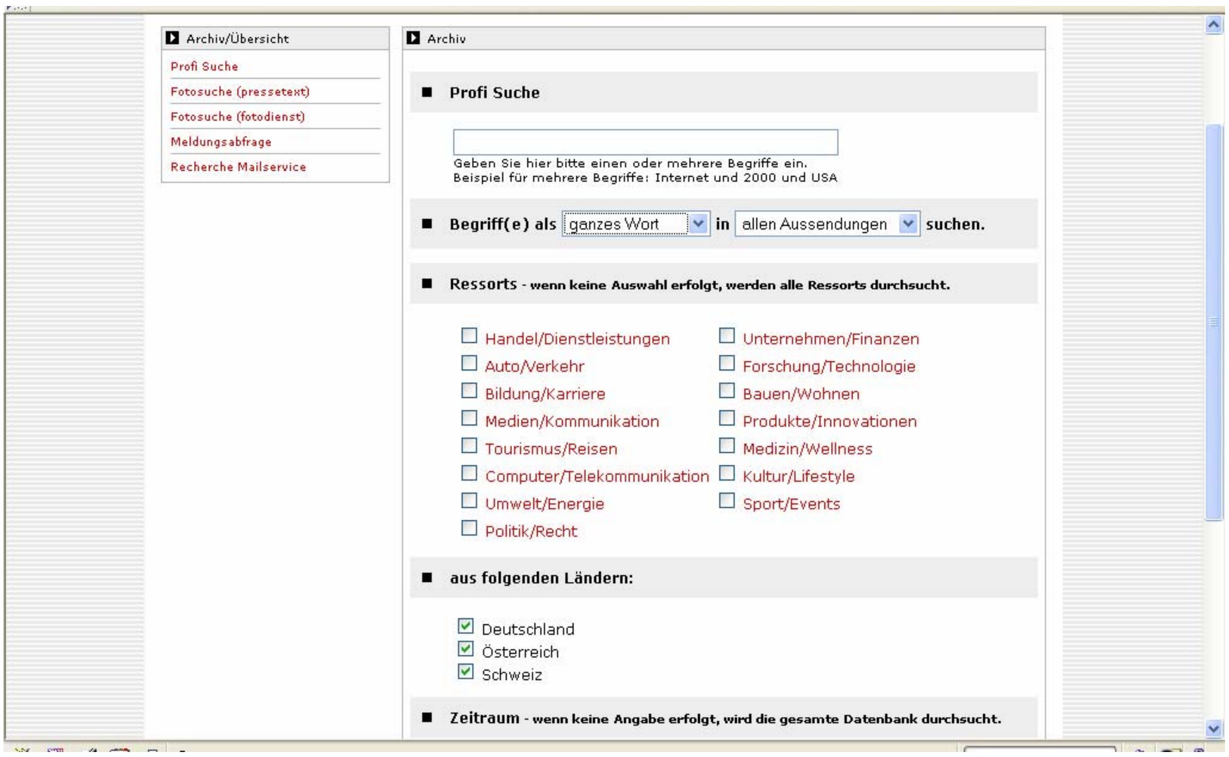
ANNUAL SUBSCRIPTIONS

Pay yearly
Our best value at $49.95 for one year of access

Figure 50: Time Online Edition advanced search page [15]

The Austrian news provider pressetext.austria offers a comprehensive search page for professionals with various options (sections, scope, countries, time span). The options are chosen carefully and presented in a concise way. However, it is not clear how logical operators should be used, since the only example shows several words connected with an AND-connection. No further help is provided.

[15] http://www.time.com/time/magazine/archives/advanced



Archiv/Übersicht

- Profi Suche
- Fotosuche (pressetext)
- Fotosuche (fotodienst)
- Meldungsabfrage
- Recherche Mailservice

Archiv

■ Profi Suche

Geben Sie hier bitte einen oder mehrere Begriffe ein.
Beispiel für mehrere Begriffe: Internet und 2000 und USA

■ Begriff(e) als ganzes Wort in allen Aussendungen suchen.

■ Ressorts - wenn keine Auswahl erfolgt, werden alle Ressorts durchsucht.

☐ Handel/Dienstleistungen	☐ Unternehmen/Finanzen
☐ Auto/Verkehr	☐ Forschung/Technologie
☐ Bildung/Karriere	☐ Bauen/Wohnen
☐ Medien/Kommunikation	☐ Produkte/Innovationen
☐ Tourismus/Reisen	☐ Medizin/Wellness
☐ Computer/Telekommunikation	☐ Kultur/Lifestyle
☐ Umwelt/Energie	☐ Sport/Events
☐ Politik/Recht	

■ aus folgenden Ländern:

☑ Deutschland
☑ Österreich
☑ Schweiz

■ Zeitraum - wenn keine Angabe erfolgt, wird die gesamte Datenbank durchsucht.

Elements of Advanced Search

- Title
- Link to Search Help
- Search options
- Submit button

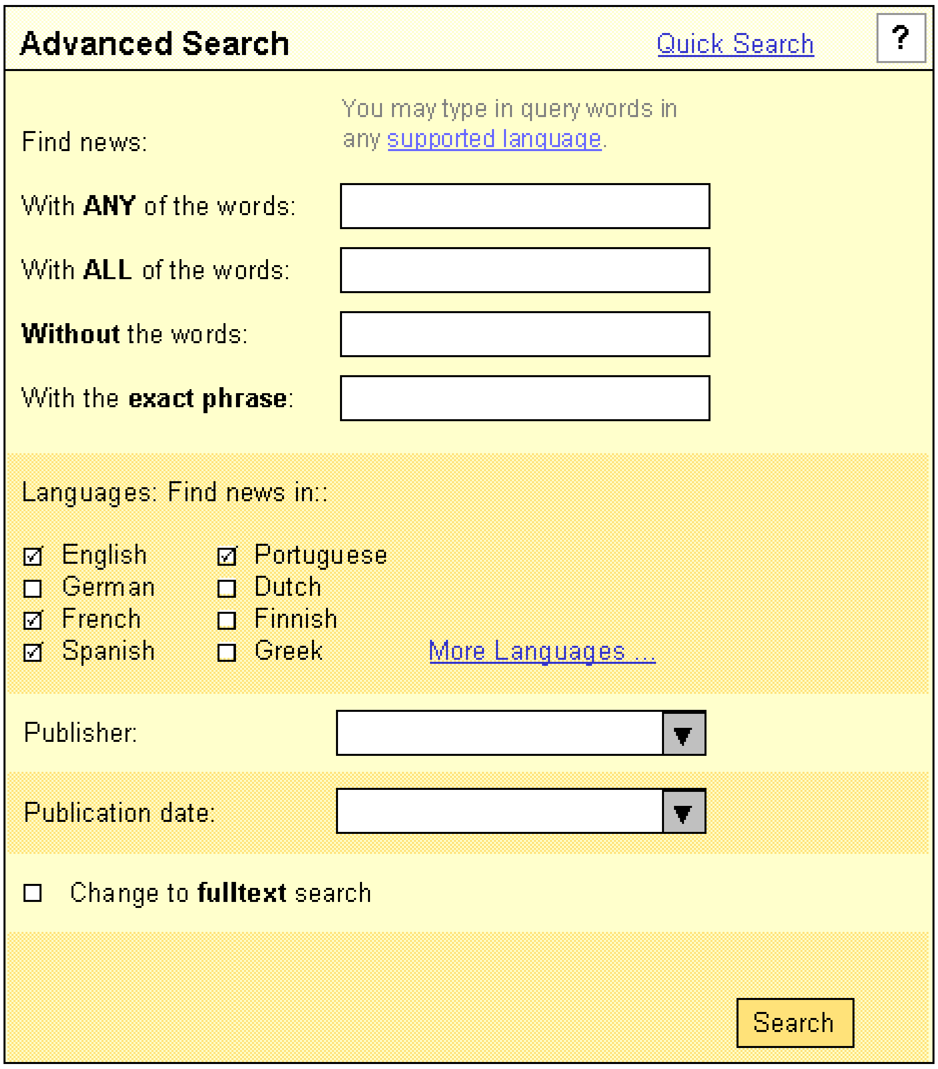
Optional elements:

- Link to Simple Search

➢ Provide a link to Simple Search, if the Simple Search module is not permanently visible on the site.

Title

As explained in Elements of Simple Search, it is recommended to use clear terms to distinguish the search modules, e.g. "Simple Search" and "Advanced Search". Jakob Nielsen recommends to "use an intimidating name like 'Advanced Search'" [4] to deter novice users, because advanced features like Boolean search are often misunderstood.


Advanced Search [Quick Search] [?]

Find news: You may type in query words in any supported language.

With **ANY** of the words:	
With **ALL** of the words:	
Without the words:	
With the **exact phrase**:	

Languages: Find news in::

☑ English	☑ Portuguese	
☐ German	☐ Dutch	
☑ French	☐ Finnish	
☑ Spanish	☐ Greek	More Languages ...

Publisher: ▼

Publication date: ▼

☐ Change to **fulltext** search

[Search]

Figure 51: Example of an Advanced Search module.

Search Options

- Chose default values that do not restrict the results, unless the results can be extended after the search.
- Check the form inputs and give clear and helpful error messages.

Search options are organized in a form. Default values should generally not be restricting the results, unless the results can be extended later (filtering). Inputs should be checked when the form was submitted, and detected errors should be communicated via helpful error messages. Users should be able to correct wrong inputs easily.

- The following options should be available in Advanced Search:
- Logical operators AND, OR, NOT
- Phrase Search

- ➢ Language selection
- ➢ Search only Title, Abstract, Keywords
- ➢ Publication Date: Age of articles and / or
- ➢ Publication Date: Timespan from – to
- ➢ Region
- ➢ Publisher
- ➢ Change to full text search

Logical Operators and Phrase Search

The Advanced Search should offer input of query words with **logical operators** AND, OR, NOT and phrase search. The order of these options should be chosen regarding the frequency of use. Usually, if no further explanation is provided, e.g. in a Simple Search box, more words are linked by the operator AND (“ALL of the words”). This should be the first option. The second frequent is the phrase search, the next could be the OR operator (“ANY of the words”) and the least frequently used is the NOT operator (“Without the words”).

- ➢ The order of logical operators and phrase search should be chosen according to the frequence of there use:
- ➢ AND (“With ALL of the words”)
- ➢ Phrase Search (“With the exact phrase”)
- ➢ OR (“With ANY of the words”)
- ➢ NOT (“Without the words”)

UI elements: long input fields, labels, which should be left of the input fields, and a title. The wording should be for instance “Find articles”.

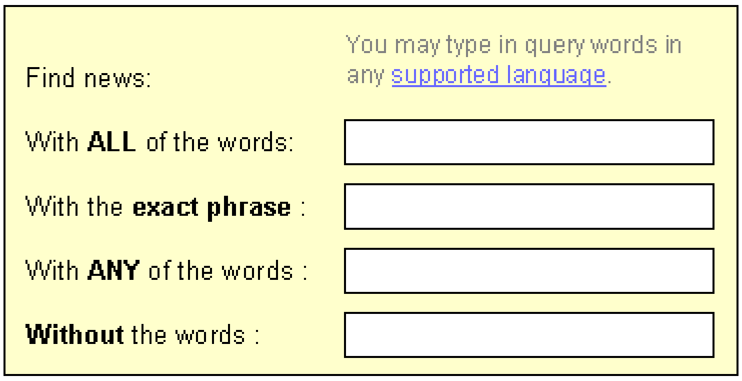
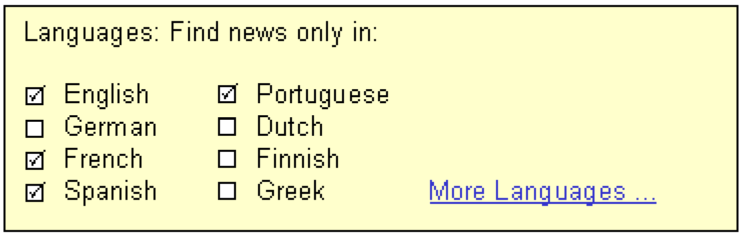

Find news:
You may type in query words in any supported language.
With **ALL** of the words:
With the **exact phrase** :
With **ANY** of the words :
Without the words :

Figure 52: Example: Logical operators and phrase search.

Default: empty

Language selection


Languages: Find news only in:

☑ English ☑ Portuguese
☐ German ☐ Dutch
☑ French ☐ Finnish
☑ Spanish ☐ Greek More Languages ...

Figure 53: Example: Language selection.

Default: previous settings

- If technically possible, save the user's language settings.
- If the user already set his languages, use the previously saved settings.

Change to full text search

UI elements: check box with label e.g. "Change to fulltext search"
Default: not checked

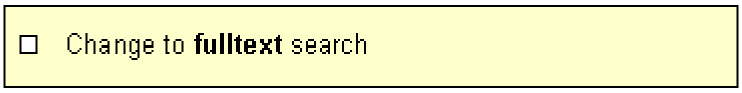

☐ Change to **fulltext** search

Figure 54: Example: Change to fulltext search.

Search only in Title, Abstract, Keywords

UI elements: check boxes
Default: not checked

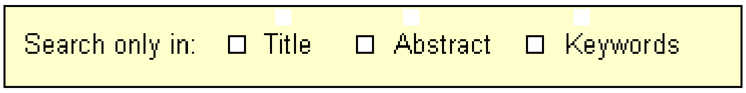

Search only in: ☐ Title ☐ Abstract ☐ Keywords

Figure 55: Example: Search only in Title, Abstract or Keywords.

Publication date: Age of articles, up-to-dateness

UI elements: Drop down list labelled e.g. "Age of articles"

The values could be e.g. All, 1 day, 1 week, 1 month, 6 months, 1 year
Default: All

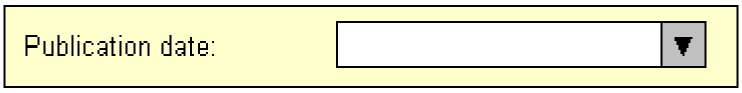

Publication date: ▼

Figure 56: Example: Publication date – age of documents.

Publication date: timespan from – to

UI elements: Separate drop down lists for day (if necessary), month and year for both "from" and "to" date. The date format could be: YYYY Month DD for the english interface version. For other language interfaces this should be adopted to common date formats. If the user selects a month, the day can be set automatically, e.g. to "01", to avoid demanding too much input from the user. Wording: e.g. "Publication date"

Default: Empty

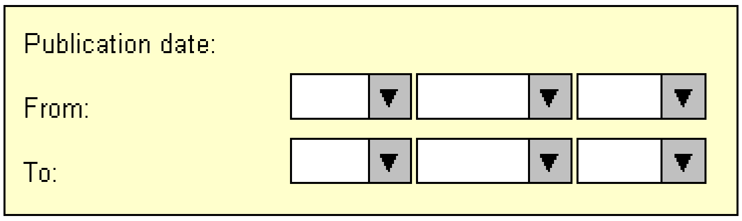


Figure 57: Example: Publication date – age of documents.

If both up-to-dateness and timespan are offered, they should both be applied, which can lead to empty results. For this case the input should be checked and a warning should be shown.

Region

UI elements: Dropdown lists and labels.

Default: Empty

Advanced Search might provide the option to search for articles from certain geographical regions. (Articles *about* certain regions can be found by entering them as query words.)

The design of this module depends on the geographical area represented in the search and the desired division into smaller regions. If there are more regions than can be displayed in one menu, the selection should be broken up into two or more reasonable steps. Depending on the first selection, the corresponding values for the next selection should be loaded. As long as the previous selection is not made, the next selections should be disabled.

- If there are too many regions to display in one menu, the selection should be made in two or more reasonable steps.

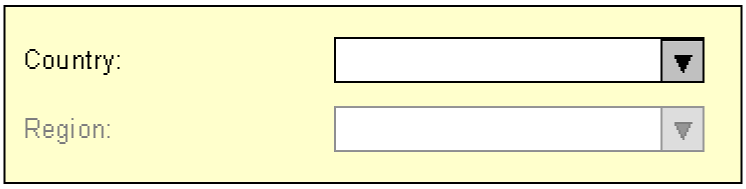


Figure 58: Example: Country and Region. Second option disabled.

Publisher

UI elements: Dropdown list and label.

Default: Empty

Articles of one certain publisher may be searched by selecting the publisher in a dropdown list (as shown in the example below).

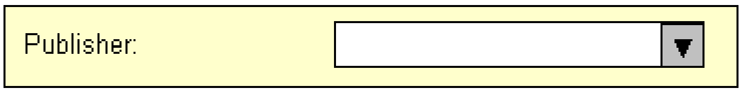


Figure 59: Example: Publisher.

- If there are only a few publishers represented in the search, they should be presented using checkboxes.

The advantage of this solution is, that more than one publisher can be selected at a time.

Submit Button

- The submit button should be placed in the bottom right corner of the module.
- If the search form is longer than will be visible on an average screen, the same button should also be placed in the top right area of the module.

A second submit button in the top right area of the module is recommended for users who only need options in the top of the search module, in order to avoid unnecessary scrolling.

The wording should follow the conventions in a language, in English this would be “Search”.

Adjustment of Elements

For the design of form elements, the following guidelines should be considered:

- Labels as well as form elements should be left-aligned.
- Whitespace should be carefully used as a means for structuring the form.
- Generally, a precise wording of the required information is important.

In order to obtain a clearly structured and easy to use search form, the grouping and order of the options should be deliberate:

- Options should be visually grouped in logical units, such as query input and logical operators, language settings, parametric search (publication date, region and publisher).
- Concerning the order of the options, take into consideration the sequence of tasks in a search process and the frequency of use (most frequent up).

Positioning & Space

The display of Advanced Search needs a certain width, due to the form elements it contains. Therefore the Advanced Search should be displayed in the "content area" of the screen or in a column of sufficient width beside the content area.

- ➢ Displayed the Advanced Search module in the "content area" of the screen or in a column of sufficient width beside the content area.

Scrolling might be necessary to view the bottom end of the module, and can be tolerated. Nevertheless, if there are too many options and the form gets too long, ways should be considered to shorten the form, for example by introducing expandable sections.

- ➢ If the module is too long to allow viewing the entire form in a comfortable way, a solution like expandable sections should be found.

Sections that can be expanded and collapsed can be for instance: Language settings, parametric search (publication date, region and publisher).

- ➢ The user interface elements must sufficiently indicate that a section can be expanded.
- ➢ Appropriate wording must be found to communicate to the user what lies behind the collapsed section.

B.3.2.3 Search Result

The search result module displays a search summary, lists the results and offers further options for manipulating the results, such as sorting and filtering options, or related links.

Guidance for query reformulation should be provided since average users are bad at redefining their query if the first attempt does not succeed. As Jakob Nielsen found in a study on e-commerce sites [5], success of a search decreased progressively with the second and third attempts to reformulate the query.

In Nielsen's study, almost half of the users gave up after one unsuccessful attempt. This implies that searches should return valuable results at the user's first attempts if any possible, and suggestions that help users forming better queries should be provided.

In a log file analysis on sequences of query reformulations, Soo Young Rieh [9] found three types of query reformulation:

- Content (specification, generalization, replacement with synonym, parallel movements)
- Format (term variations, operator usage or error correction) and
- Resource reformulation (e. g. article, picture, URL)

Most query reformulations (over 80 %) performed by users are content reformulations. Nearly half of them consist of parallel moves, e.g. "air canada" → "united airlines" → "alaska airlines", which is in fact no reformulation for the reasons of an unsuccessful previous attempt, but a range of queries to find related but different contents.

The second largest part (31.5 % of the content reformulations, about 25 % of all query reformulations) is made up of specifications. This means that users often start with rather general terms and then try to get more specific with further attempts, e.g.:

“no prescription” → “no prescription” viagra → “no prescription” viagra xenical → “no prescription” viagra xenical phentermine

Almost 14 % of all query reformulations account for generalizations, e.g.:

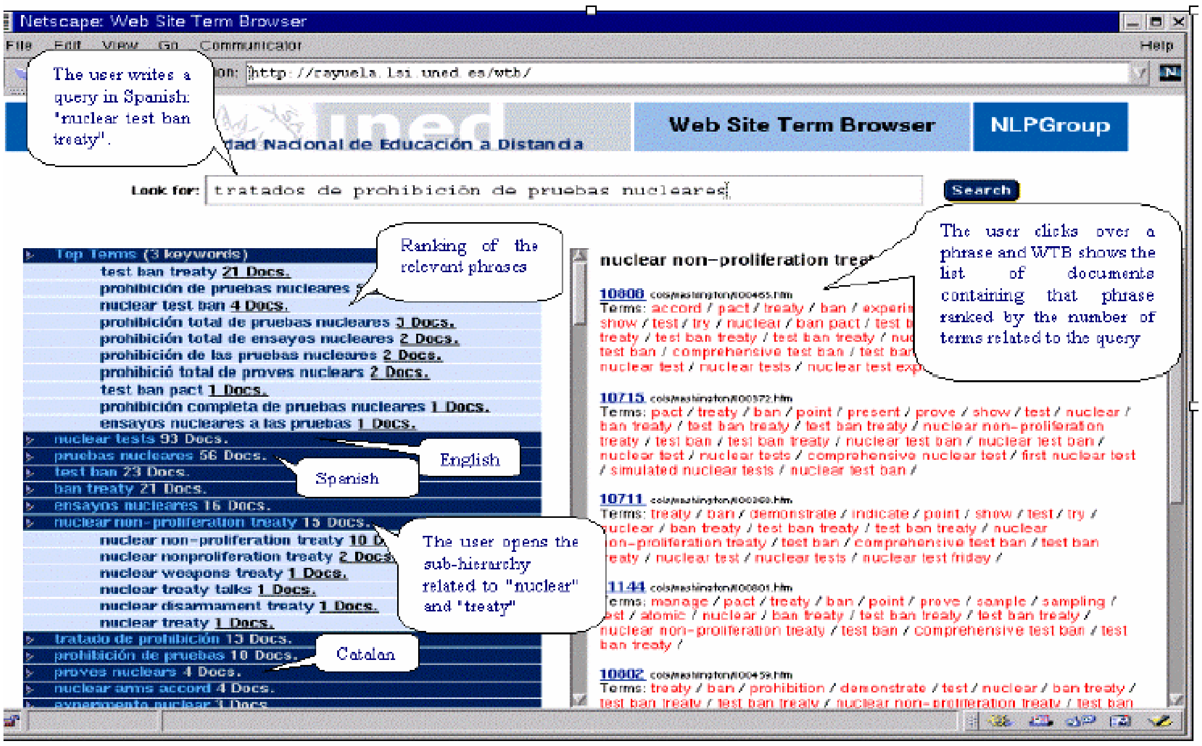
liberal feminism → types of feminism → definitions of feminism

This kind of knowledge can be useful for search designers in order to provide useful suggestions to help users reformulate their queries. The findings suggest that concentrating on more specific, more general and related terms will be most needed. Nearly one third of the format reformulations consist of error corrections. This emphasizes the need of a spell checking functionality.

Other Approaches

The display of results recommended in this document is based on a conventional hitlist with the additional option to filter and manipulate results. Besides this well-known kind of display for search results, several other methods have been developped and will be presented here briefly:

Anselmo Peñas et al. [8] complemented the known list view with a “browse by phrases” interface that allows users to view documents with certain phrases, in the search language as well as in translations. This is an innovative approach to interactive CLIR (Cross-language Information Retrieval). However, the interface was not tested and evaluated with users.


Netscape: Web Site Term Browser
File Edit View Go Communicator Help
http://rayuela.lsi.uned.es/wtb/
dad Nacional de Educación a Distancia
Web Site Term Browser
NLPGroup
Look for:
Search
The user writes a query in Spanish: "nuclear test ban treaty".
Ranking of the relevant phrases
The user clicks over a phrase and WTB shows the list of documents containing that phrase ranked by the number of terms related to the query
English
Spanish
The user opens the sub-hierarchy related to "nuclear" and "treaty"
Catalan

tratados de prohibición de pruebas nucleares

Top Terms (3 keywords)
test ban treaty 21 Docs.
prohibición de pruebas nucleares
nuclear test ban 4 Docs.
prohibición total de pruebas nucleares 3 Docs.
prohibición total de ensayos nucleares 2 Docs.
prohibición de las pruebas nucleares 2 Docs.
prohibició total de proves nuclears 2 Docs.
test ban pact 1 Docs.
prohibición completa de pruebas nucleares 1 Docs.
ensayos nucleares a las pruebas 1 Docs.
nuclear tests 93 Docs.
pruebas nucleares 56 Docs.
test ban 23 Docs.
ban treaty 21 Docs.
ensayos nucleares 16 Docs.
nuclear non-proliferation treaty 15 Docs.
nuclear non-proliferation treaty 10 D
nuclear nonproliferation treaty 2 Docs
nuclear weapons treaty 1 Docs.
nuclear treaty talks 1 Docs.
nuclear disarmament treaty 1 Docs.
nuclear treaty 1 Docs.
tratado de prohibición 13 Docs.
prohibición de pruebas 10 Docs.
proves nuclears 4 Docs.
nuclear arms accord 4 Docs.
experimento nuclear 3 Docs.

nuclear non-proliferation treat

10808 cols/washington/00465.htm
Terms: accord / pact / treaty / ban / experi
show / test / try / nuclear / ban pact / test b
treaty / test ban treaty / test ban treaty / nuc
test ban / comprehensive test ban / test ban
nuclear test / nuclear tests / nuclear test exp

10715 cols/washington/00372.htm
Terms: pact / treaty / ban / point / present / prove / show / test / nuclear / ban treaty / test ban treaty / test ban treaty / nuclear non-proliferation treaty / test ban / test ban treaty / nuclear test ban / nuclear test ban / nuclear test / nuclear tests / comprehensive nuclear test / first nuclear test / simulated nuclear tests / nuclear test ban /

10711 cols/washington/00369.htm
Terms: treaty / ban / demonstrate / indicate / point / show / test / try / nuclear / ban treaty / test ban treaty / test ban treaty / nuclear non-proliferation treaty / test ban / comprehensive test ban / test ban treaty / nuclear test / nuclear tests / nuclear test friday /

1144 cols/washington/00801.htm
Terms: manage / pact / treaty / ban / point / prove / sample / sampling / test / atomic / nuclear / ban treaty / test ban treaty / test ban treaty / nuclear non-proliferation treaty / test ban / comprehensive test ban / test ban treaty /

10802 cols/washington/00459.htm
Terms: treaty / ban / prohibition / demonstrate / test / nuclear / ban treaty / test ban treaty / test ban treaty / nuclear non-proliferation treaty / test ban

Figure 60: Browsing by Phrases interface, Anselmo Peñas et al.

Edward Suvanaphen’s and Jonathan C. Roberts’ approach [11] was to visualize the relationships between the results of multiple reformulated queries. If at all, those relationships are usually only implicitly learned by the users. In this experiment those relations were made visible to aid users in judging the relevance of documents. This was mainly accomplished by offering a “summary view” that contained only results of more than one search attempt.

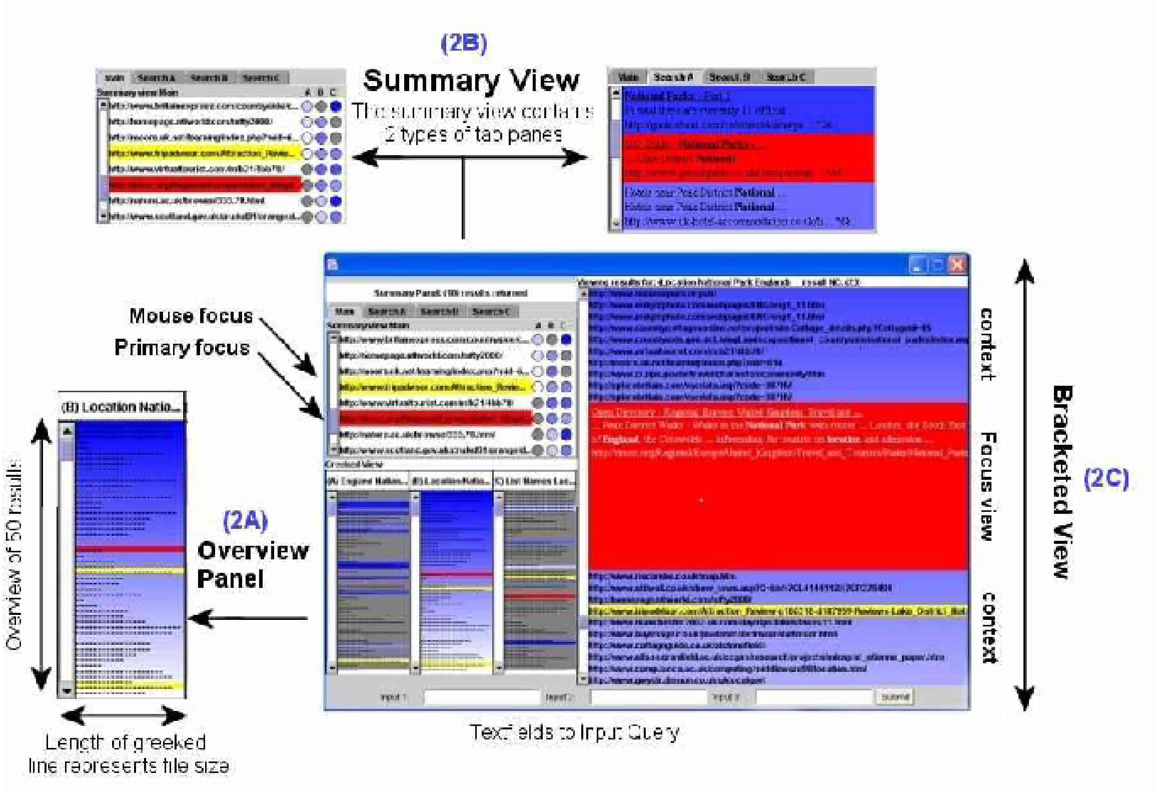


Figure 61: Multiple Search Result Visualization, Suvanaphen, E. and Roberts, J. C.

Their user tests showed that this summary view helped users to identify relevant results more effectively.

William C. Ogden and Mark W. Davis investigated ways to improve cross-language text retrieval with human interactions [7]. Their approach was to support searching documents in languages the researchers cannot understand, but want to select for human translation and further analysis. Therefore Ogden et al. try to allow choosing documents without reading.

They evaluated a thumbnail view with query term highlighting. Compared with a display of document titles, this interface proved to enhance the process of selecting documents, but had little effect on the general performance for tasks that required document reading.

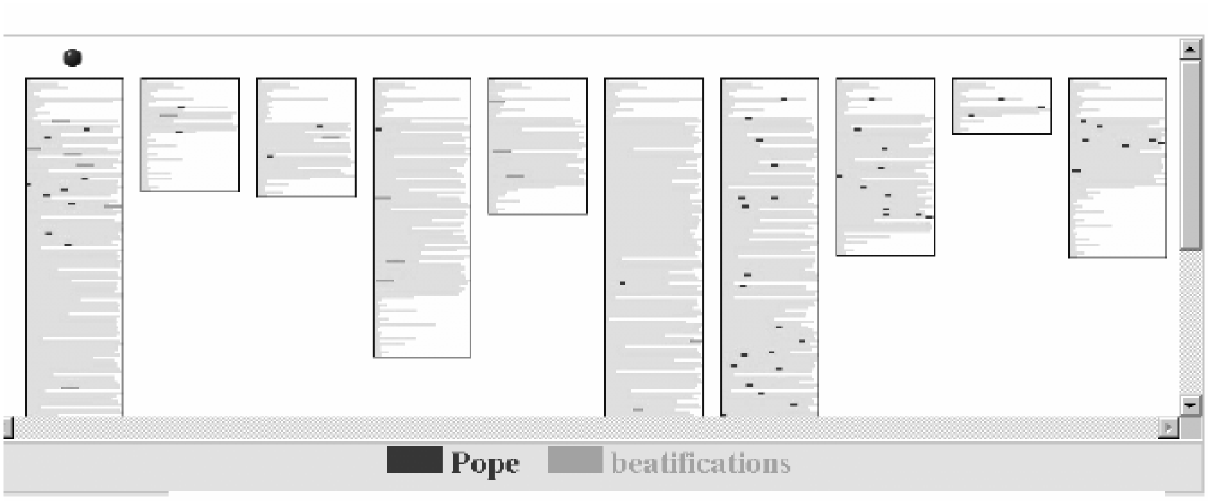
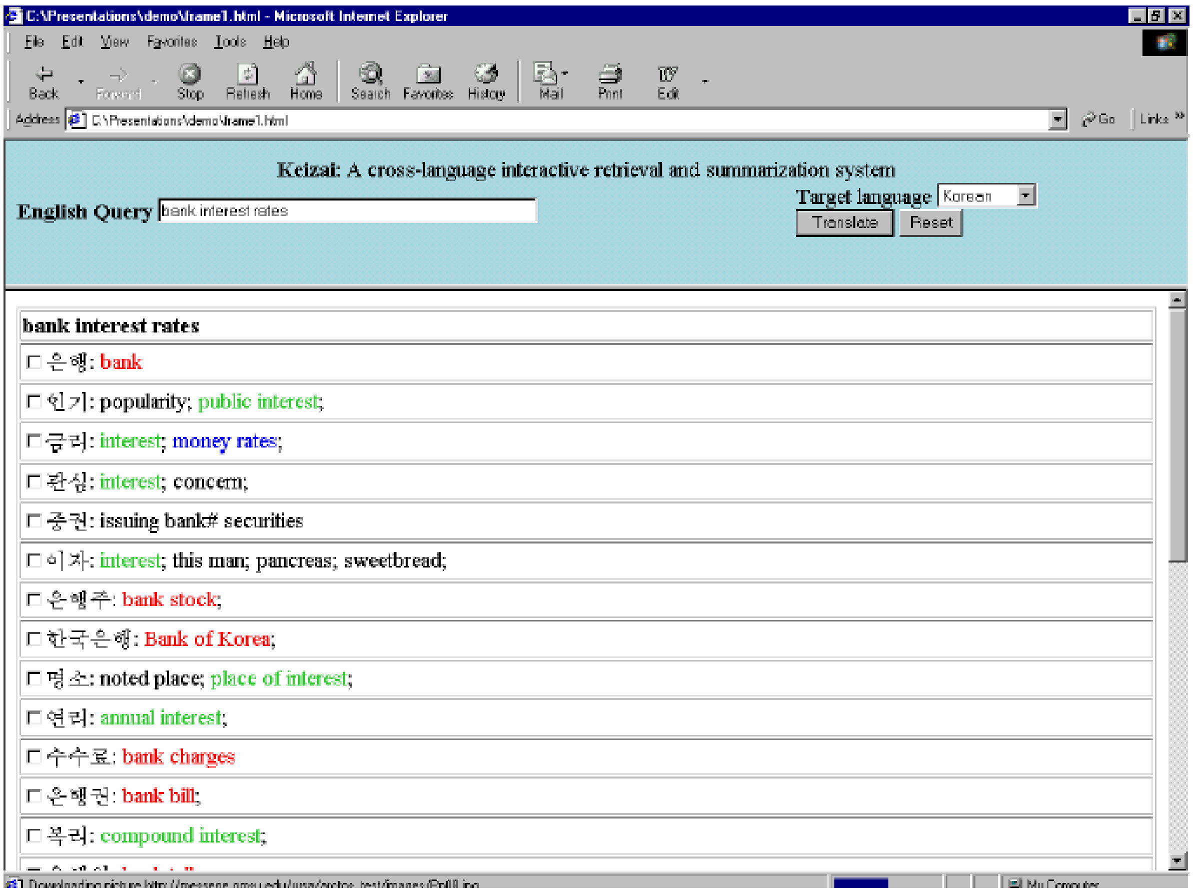


Figure 62: Thumbnail view with query term highlighting, William C. Ogden and Mark W. Davis

With the Keizai project, that brings together much of their previous findings, William C. Ogden and Mark W. Davis aimed at providing a cross-language text retrieval system that allows users to search and accurately judge Japanese and Korean news documents. Users enter a query in English and then select the foreign language terms by English definitions that most accurately match their original query, without the need for morphological analysis and segmentation.


C:\Presentations\demo\frame1.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address C:\Presentations\demo\frame1.html
Go Links
Translate Reset

Keizai: A cross-language interactive retrieval and summarization system

English Query: bank interest rates

Target language: Korean

bank interest rates

- ☐ 은행: bank
- ☐ 인기: popularity; public interest;
- ☐ 금리: interest; money rates;
- ☐ 관심: interest; concern;
- ☐ 증권: issuing bank# securities
- ☐ 이자: interest; this man; pancreas; sweetbread;
- ☐ 은행주: bank stock;
- ☐ 한국은행: Bank of Korea;
- ☐ 명소: noted place; place of interest;
- ☐ 연리: annual interest;
- ☐ 수수료: bank charges
- ☐ 은행권: bank bill;
- ☐ 복리: compound interest;

Figure 63: Demo of the Keizai interface, William C. Ogden and Mark W. Davis

Abdelali et al. [1] developped a complex multilingual information retrieval tool based on these studies, incorporating result display with color coded query term highlighting and translation options.

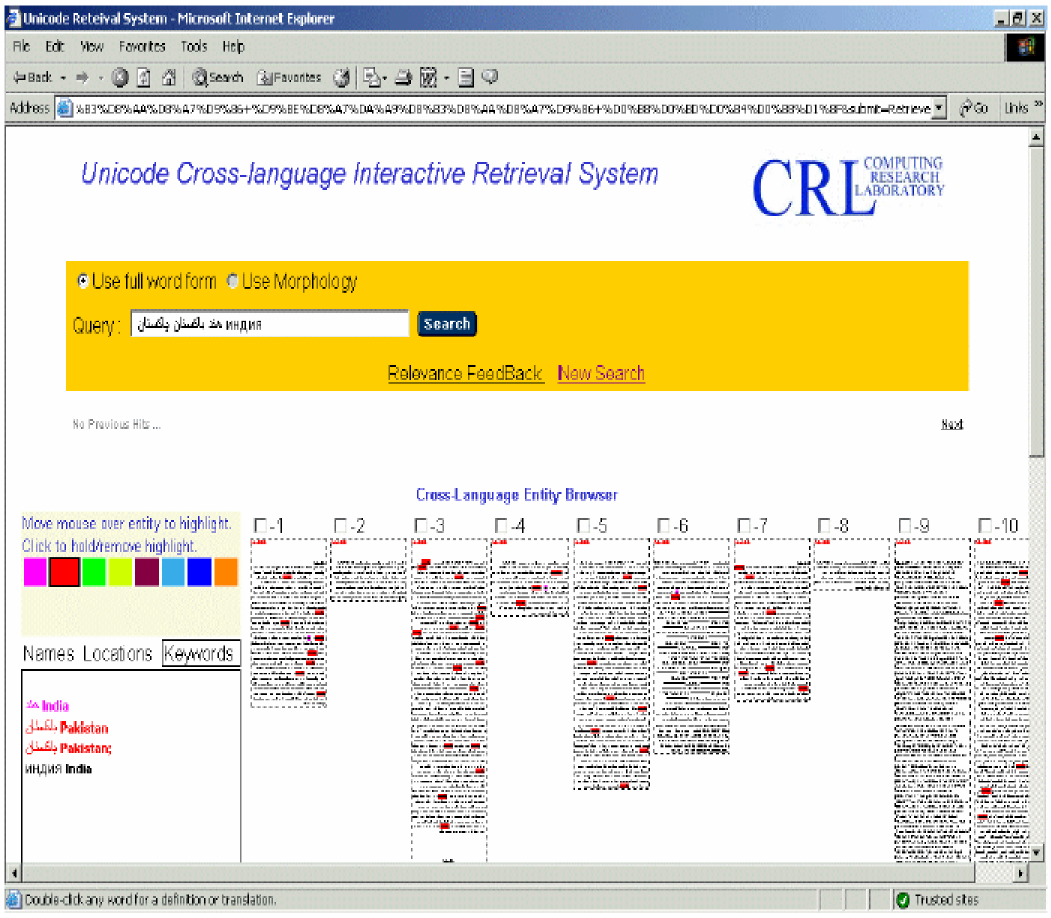
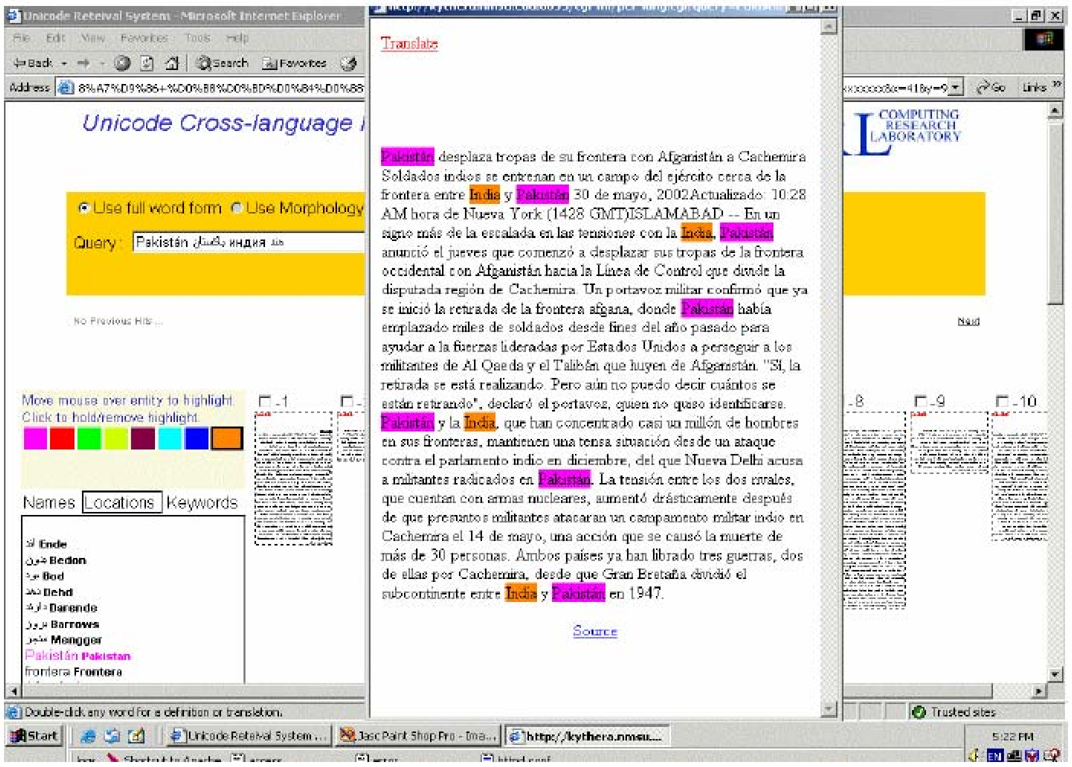

Unicode Reteival System - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address
Go Links
Unicode Cross-language Interactive Retrieval System
CRL COMPUTING RESEARCH LABORATORY
Use full word form Use Morphology
Query:
Search
Relevance FeedBack New Search
No Previous Hits...
Next
Cross-Language Entity Browser
Move mouse over entity to highlight.
Click to hold/remove highlight.
Names Locations Keywords
India
Pakistan
Pakistan;
индия India
-1 -2 -3 -4 -5 -6 -7 -8 -9 -10
Double-click any word for a definition or translation.
Trusted sites

Figure 64: Thumbnail view with color coded query term highlighting, Abdelali et al.



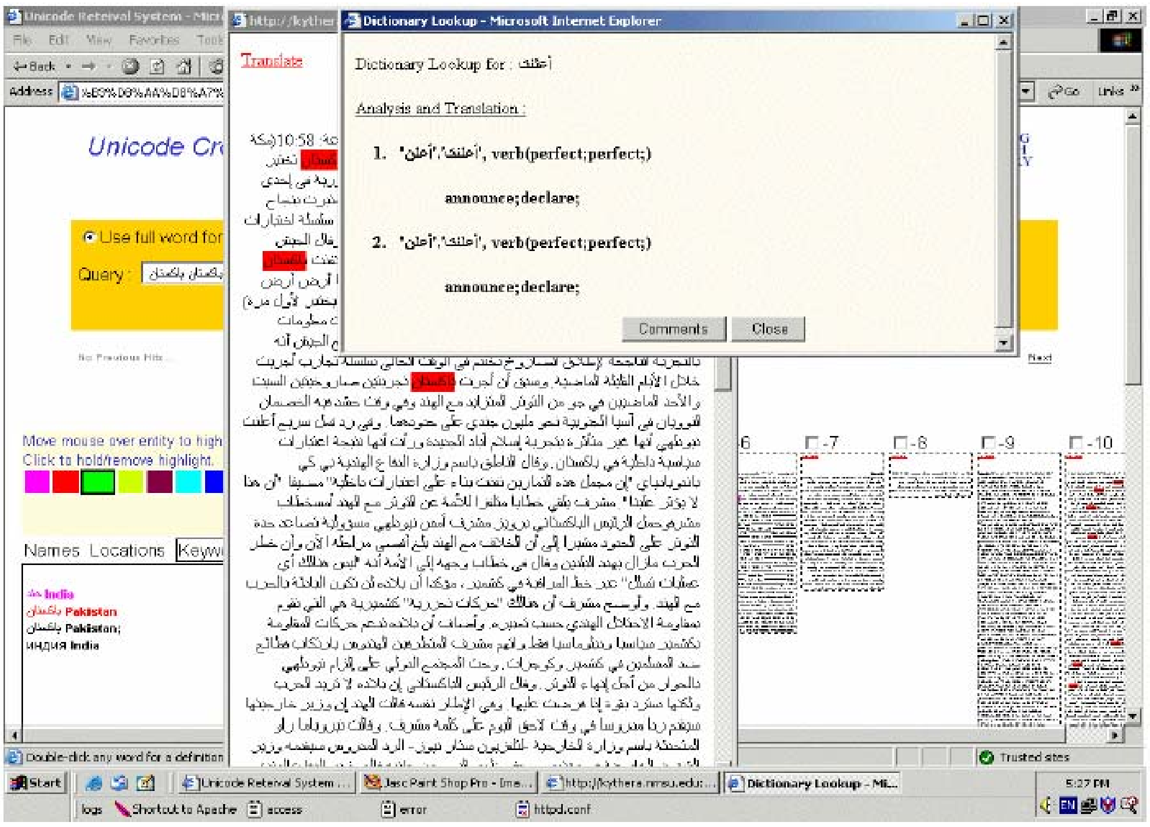
Translate

Pakistán desplaza tropas de su frontera con Afganistán a Cachemira Soldados indios se entrenan en un campo del ejército cerca de la frontera entre India y Pakistán 30 de mayo, 2002Actualizado: 10:28 AM hora de Nueva York (1428 GMT)ISLAMABAD -- En un signo más de la escalada en las tensiones con la India, Pakistán anunció el jueves que comenzó a desplazar sus tropas de la frontera occidental con Afganistán hacia la Línea de Control que divide la disputada región de Cachemira. Un portavoz militar confirmó que ya se inició la retirada de la frontera afgana, donde Pakistán había emplazado miles de soldados desde fines del año pasado para ayudar a la fuerzas lideradas por Estados Unidos a perseguir a los militantes de Al Qaeda y el Talibán que huyen de Afganistán. "Sí, la retirada se está realizando. Pero aún no puedo decir cuántos se están retirando", declaró el portavoz, quien no quiso identificarse. Pakistán y la India, que han concentrado casi un millón de hombres en sus fronteras, mantienen una tensa situación desde un ataque contra el parlamento indio en diciembre, del que Nueva Delhi acusa a militantes radicados en Pakistán. La tensión entre los dos rivales, que cuentan con armas nucleares, aumentó drásticamente después de que presuntos militantes atacaran un campamento militar indio en Cachemira el 14 de mayo, una acción que se causó la muerte de más de 30 personas. Ambos países ya han librado tres guerras, dos de ellas por Cachemira, desde que Gran Bretaña dividió el subcontinente entre India y Pakistán en 1947.

Source

Unicode Retrieval System - Microsoft Internet Explorer
Unicode Cross-language
Use full word form
Use Morphology
Query: Pakistán باكستان индия هند
No Previous Hits ...
Move mouse over entity to highlight
Click to hold/remove highlight
Names
Locations
Keywords
Pakistán Pakistan
frontera Frontera
Next
COMPUTING RESEARCH LABORATORY
Double-click any word for a definition or translation.
Trusted sites
Start

Figure 65: Document view with query term highlighting, Abdelali et al.



Dictionary Lookup - Microsoft Internet Explorer

Dictionary Lookup for : أعلنت

Analysis and Translation :

1. 'أعلن','أعلنت', verb(perfect;perfect;)

   announce;declare;

2. 'أعلن','أعلنت', verb(perfect;perfect;)

   announce;declare;

Comments Close

Figure 66: Arabic document view with word translation, Abdelali et al.

These approaches all contain valuable ideas to ease the user's search process, but they also make the search interface or result display, respectively, more complicated. The way to display search results should be individually chosen regarding the user requirements, search domain etc. Further research is necessary to evaluate new innovative approaches like the ones mentioned above.

Display of Results in More Than One Language

In multilingual searches, the search results may contain documents in different languages. Those languages may already be selected by the user, for instance if the Advanced Search module offers language selection or general language preferences for the entire site were set.

If this is not the case, the results may contain documents in languages that the user does not understand and does not wish to have amongst his results. To avoid this, it is reasonable to give the user the possibility to control the languages after the search was conducted. This can be done by asking the user before displaying the results, offering the possibility to filter the results or to display the results sorted by language.

Results in more than one language can be displayed in different ways:

All results on one page. This seems a good possibility if a user already set his language preferences to more than one language and will be content with a result set consisting of articles in different languages. Moreover, filtering and sorting options may be provided.

A **dialogue screen** might be displayed, where the user can chose between the result languages. This implies that the user has to take an additional step to get to his results, which should be avoided.

An advantage of such a dialogue screen could be that the user can be lead to set his preferences. If those preferences are saved and used in further search processes, this dialogue screen needs to be shown only once. Nevertheless, the user must have the possibility to change those preferences later on.

Results in **one language** can be displayed, while access to results in other languages can be provided as links above the results, together with the search summary. This option should be preferred to the previous one, since no additional choice is necessary. The language displayed first should be chosen using reasonable criteria as discussed in B.3.1.6 Estimating Preferred Languages for Results.

Recommendations:

- If the user already selected his preferred languages for the results, all results should be displayed together in one result list.
- Otherwise, if the user probably understands one certain language, but does not wish to view results in all the languages found, results in one language should be displayed and the option to choose other languages should be provided.
- If the user probably wants to view all results in one result list, display all results together, but provide the possibility to choose certain languages.

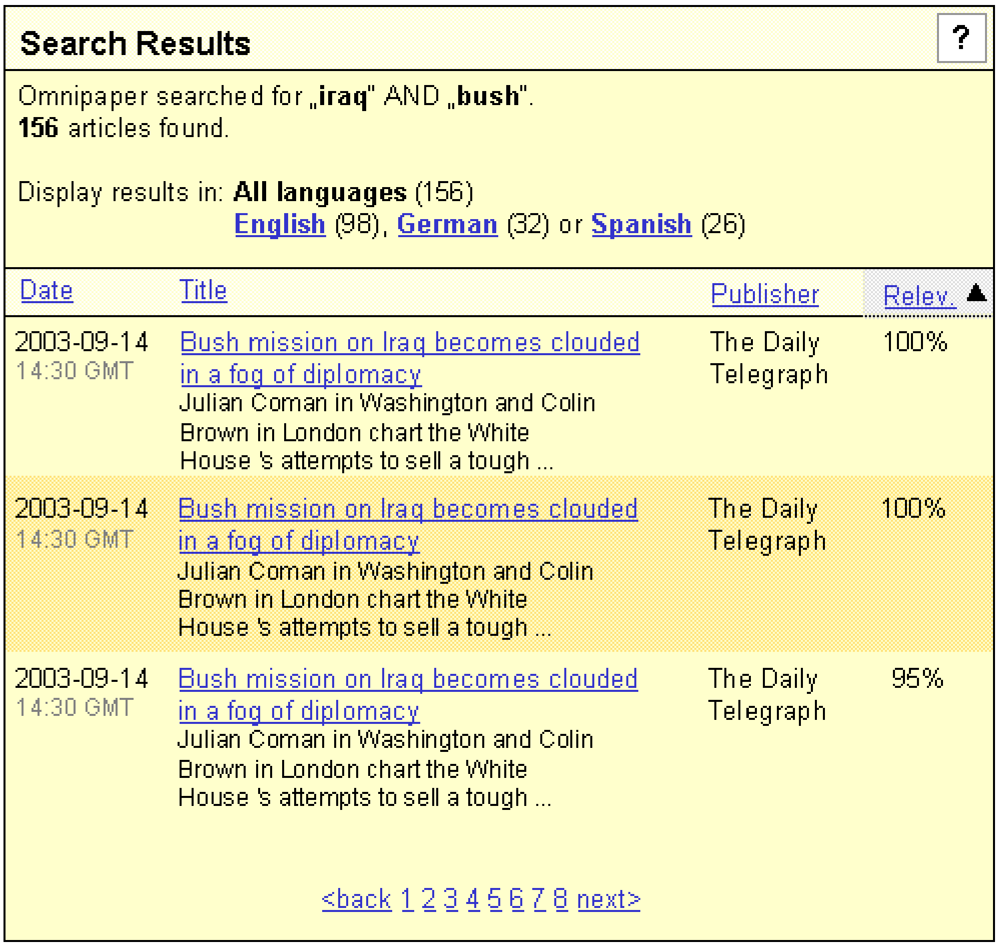
Elements of the Search Result Module

The search result module may consist of the following elements:

- Search summary
- Link back to search again (unless search form elements are visible)
- List of results (about 10 per page)
- Navigation elements for browsing through multiple result pages

Optional elements:

- Categories or concepts used in the search
- Translation option (for result list)
- Sorting options
- Related links
- Result manipulation (filtering options, specially language filtering, and search within the results)
- Query manipulation (options for refining or expanding the search, spell checking)



Search Results ?

Omnipaper searched for „**iraq**" AND „**bush**".
156 articles found.

Display results in: **All languages** (156)
English (98), **German** (32) or **Spanish** (26)

Date	Title	Publisher	Relev. ▲
2003-09-14 14:30 GMT	Bush mission on Iraq becomes clouded in a fog of diplomacy Julian Coman in Washington and Colin Brown in London chart the White House 's attempts to sell a tough ...	The Daily Telegraph	100%
2003-09-14 14:30 GMT	Bush mission on Iraq becomes clouded in a fog of diplomacy Julian Coman in Washington and Colin Brown in London chart the White House 's attempts to sell a tough ...	The Daily Telegraph	100%
2003-09-14 14:30 GMT	Bush mission on Iraq becomes clouded in a fog of diplomacy Julian Coman in Washington and Colin Brown in London chart the White House 's attempts to sell a tough ...	The Daily Telegraph	95%

<back 1 2 3 4 5 6 7 8 next>

Figure 67: Example: Search Results with search summary, language filtering and sorting

options.

Search Summary

A common mistake in designing search result pages is not to include the search criteria. The search summary reduces the memory load by showing the user what he was looking for, helps to interprete the search results and possibly also to adjust the query in order to get better results. [4]

The search summary could contain:

- A summary of query words and search parameters
- The number of found articles
- Languages of found articles
- Categories or concepts used in the search

Example search summary:

”Omnipaper searched for ‘**EU budget**’. **2,300** results in **English** and **Spanish**.”

Link to Search Again

- If the search form is not visible, there should be a link to the search. Wording suggestion: “Search Again”.

List of Results

A major problem that users often have with search facilities is interpreting the search results [10]. Users have to find out which of the results are relevant to them as quickly as possible. It must not be necessary to open each link to see what is behind. Enough information should be provided to enable fast scanning of the results. Therefore, the full title and a description, if available, should be displayed for each result. The description should be short and precise, if possible, to avoid overstraining users with too much text.

As Jakob Nielsen says “Users almost never look beyond the second page of search results” [5]. Therefore, search results should be ranked by relevance to the user and all the most relevant results should be shown on the first page.

- The list of results should provide **enough information** to ensure that users can judge which articles are useful for them and which are not, without opening each link.
- If **sorting options** for several properties are provided, the list should be organized as a table with the sorting functions in the first row. This contributes to a clear structure of the presented data.

The list of results should contain the following information:

- Full **title of article**, emphasized (e.g. larger font as the continuous text, bold), as a link to the article

- Below the title an **abstract** of the article should be shown, if available. This abstract might be the beginning of a longer abstract, abbreviated to a certain length. The description should give enough information about the article, but it should not take too much space, recommended are 2 or 3 lines.
- For results of a fulltext search, the **phrases containing the query words** should be displayed instead of the description. The query words or their translations, respectively, should be highlighted (e.g. bold or different color).
- If **subject classifications** of articles are available, display the classification information below the description, in a different style than the continuous text (e.g. in a different color)
- In the line below the description, **additional information**, such as publisher, date, region and a link to the media website, if available, may be shown.

Also general guidelines for the implementation of lists should be taken into account:

- Every entry is a link itself (the headline) and additionally a "more…" link at the end of the abstract shall be provided.
- The headline should be bold to support the user scanning the entries.

Navigation Elements to Browse Through Multiple Result Pages

In case there are more results than displayed on a single page, there must be navigation elements for browsing through the result pages. Usually those elements consist of links to go back and forth as well as numbered links for going directly to a certain page. Those elements are usually placed at the bottom of the screen, but they could be additionally offered above the result list.

If there are a high number of result pages, the numbered links shown may be limited to a certain number before and after the currently viewed page. For instance, if page number 25 is currently viewed, links from page 20 to page 30 may be shown.

The links to go back and forth could be named "Previous" and "Next". They should be placed left and right of the numbered page links.

Those elements could look as shown here:

<back 1 2 3 4 5 6 7 8 9 10 next>

- Use big click-ssensible areas for the paging mechanisms. Don't force the user to finetune his mouseicon onto a small icon.

Sorting Options

- Sorting options should be displayed in a row above the list of results.

For each column there should be a link or button that sorts the result list by the respective property.

- If useful, the additional option to sort up or down may be provided.
- In this case, the sorting direction should be visualised using an arrow symbol.

➢ When a sorting link or button is hit for the first time, the most reasonable sorting direction should be chosen, e.g. most recent results first (instead of oldest results first) or most relevant first (instead of least relevant first).

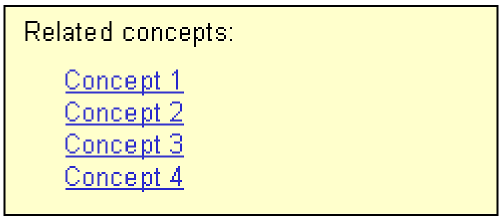
The following sorting options may be provided:

- Sorting by relevance
- Sorting by date
- Sorting by title
- Sorting by publisher
- Sorting by category
- Etc.

Default: By default the list should be sorted by relevance, based on the search parameters, most relevant results first.

Related Links

Related links, topics, categories or concepts may be offered above or beside the result list. This may be combined with possible links to results in other languages.



Related concepts:

Concept 1
Concept 2
Concept 3
Concept 4

Figure 68: Example: Related links – here: related concepts.

Filtering Options

Filtering means to manipulate the search results by certain criteria. These criteria may be for instance:

- Language
- Categories
- Concepts

Theoretically, every search parameter can also be used for manipulating the results after the search was conducted.

If filtering is useful and which filtering options should be provided has to be decided carefully by taking into consideration what the user can do in the respective search modules, and what makes sense to change afterwards on the search results. For instance, if the Simple Search module is used and does not offer any language selection options, it is useful to provide this as a filtering option.

There are specially options that are not known before the search was conducted. For instance concepts that were used during the search process can be offered for filtering the results.

Filtering by Language

Filtering by language can be offered using the language filtering module, which is discussed in chapter The Language Filtering Module.

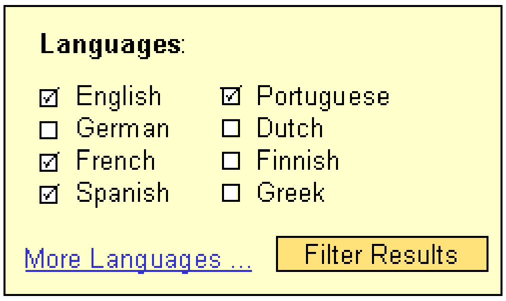


Figure 69: Example: Language filtering.

➢ If there are only two or three languages, language filtering can be offered using text links above the results list.

For instance:

”Omnipaper found **3900** results for ‘**gift**’.
Show only results in **English** (2.300) **German** (400) or **Dutch** (1.200)”

Or:

“Omnipaper found **2300** results for ‘**gift**’ in **English**.
We also found results in **German** (400) or **Dutch** (1.200)”

➢ In this example, if one certain language is selected, a link to display all languages again (“All Languages”) should be shown.

Filtering by Category

If the user’s query is translated or assigned to categories or concepts, or keywords are used in the search process, this information may be presented to the user in order to make the search process more transparent and to provide additional information and options.

A filtering option for categories could be offered if it is possible to unify the categories of all available articles in one scheme, and if there is a manageable number of categories (about 7). Examples for categories are: Economy, Sports, International, Local, Politics etc.

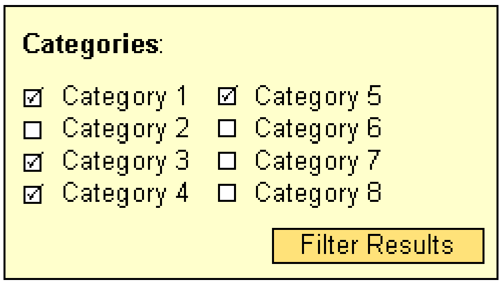


Figure 70: Example: Category filtering.

UI elements: checkboxes and labels of checkboxes.

Default: All checked

Search within Results

A module might be offered to search within the results. A link may be provided in the search result module that leads to a new page.

- It must be clear to the user, what the scope of this search is. Therefore it is necessary to provide a search summary and a short description

For example:

“Omnipaper searched for ‘**soccer**’ – 2350 articles found
Search within these results:”

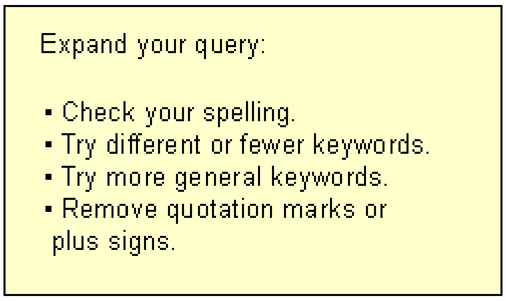
The search form for the search within results may be a Simple Search form with only one input field, or a more complex form as in the Advanced Search module.

Search within results means for the user that he must keep in mind two queries. It might be easier for the user to review and change the search phrases and parameters in the search form. Therefore, if the user goes back to the search form, the previous query inputs should be saved.

- The last query inputs in a search form should always be saved to enable editing them without repeating all the selections.

Expansion

- Expansion suggestions may be provided as a separate module beside the result list.
- If the search returned no results, the expansion suggestions should be displayed below the search summary to be more obvious.


Expand your query:

▪ Check your spelling.
▪ Try different or fewer keywords.
▪ Try more general keywords.
▪ Remove quotation marks or plus signs.

Figure 71: Example: Expanding suggestions.

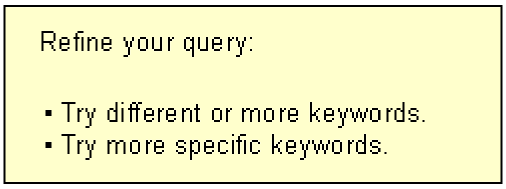
Suggestions for expanding the search may be provided beside the search results as a standard module, or only in case the query returns no or only a few results. In this case suggestions for expanding the search could be displayed below the search summary, because they are more important then.

Suggestions for expanding the search should be:

- Check your spelling.
- Try different or fewer keywords.
- Try more general keywords.
- Remove quotation marks or plus signs.
- If the language for results can be set outside the search form, it should be suggested to check those settings:
  ”Check your language settings.”

If possible, the system may provide suggestions of more general query words or hints, about which specific search parameters to change (e.g. Search not only in titles, but in entire articles). Whenever possible, those specific recommendations should include a direct link to the particular search.

Refining Suggestions


Refine your query:

▪ Try different or more keywords.
▪ Try more specific keywords.

Figure 72: Example: Refining suggestions.

- Refining suggestions should be provided as a separate module beside the result list.
- If the query returns an exceptionally high number of results, refining suggestions should be placed above the result list to be more apparently visible.

Suggestions for refining the query should be:

- Try different or more keywords
- Try more specific keywords
- The system should also provide a list of links to related, more specific categories or phrases to chose from, if such information is available.

Spell Checking

- If the system detects a possible spelling error, links to a search with similar words should be displayed below the search summary

Example: “Did you mean multimodal?”

Search Fail Message

- If no results were found, the search summary should contain a search fail message.

For example: “Sorry, Omnipaper found no results for ‘multimodal’”

- If possible, this should be complemented by helpful suggestions. These could be:
- Spell checking
- Expansion suggestions
- Related links (topics, concepts)

- If the system knows that the found results are of low relevance, the user should also be informed, to avoid irritations about results that seem to have little to no relation to what the user was looking for.

Adjustment of Elements

- Elements and options of high importance should be placed above the result list

For example the search summary, spell checking and search fail messages and suggestions.

- Options with effect on the result list should have a visual connection to the list.

For example the sorting options are ideally placed directly above the results.

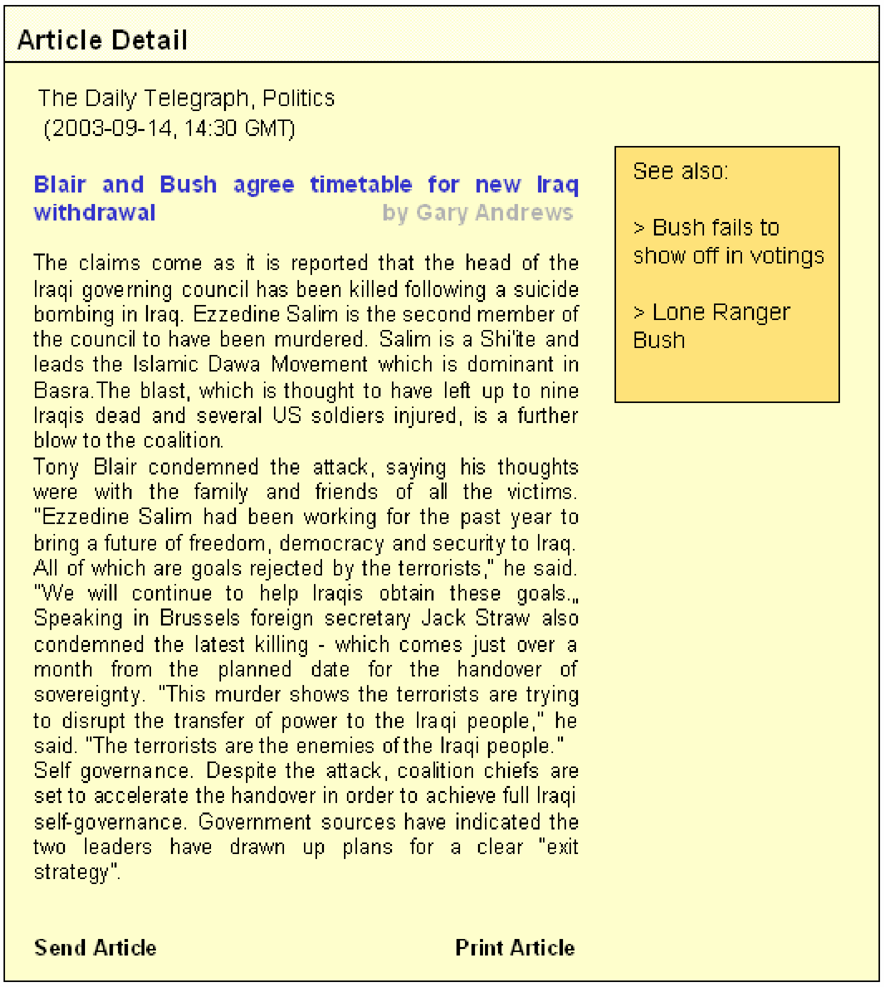
- Further options such as related links should be placed beside the result list, to account for their character as alternatives.

Positioning & Space

- The search result module should be displayed in the “content area” of the screen.

The width of this area should usually be the width of the screen (browser window) minus vertical navigation bars. The display of the search result module needs a certain width, due to the amount of text it contains.

B.3.2.4 Detailed Article Display



Article Detail

The Daily Telegraph, Politics
(2003-09-14, 14:30 GMT)

Blair and Bush agree timetable for new Iraq withdrawal by Gary Andrews

The claims come as it is reported that the head of the Iraqi governing council has been killed following a suicide bombing in Iraq. Ezzedine Salim is the second member of the council to have been murdered. Salim is a Shi'ite and leads the Islamic Dawa Movement which is dominant in Basra.The blast, which is thought to have left up to nine Iraqis dead and several US soldiers injured, is a further blow to the coalition.
Tony Blair condemned the attack, saying his thoughts were with the family and friends of all the victims. "Ezzedine Salim had been working for the past year to bring a future of freedom, democracy and security to Iraq. All of which are goals rejected by the terrorists," he said. "We will continue to help Iraqis obtain these goals.„ Speaking in Brussels foreign secretary Jack Straw also condemned the latest killing - which comes just over a month from the planned date for the handover of sovereignty. "This murder shows the terrorists are trying to disrupt the transfer of power to the Iraqi people," he said. "The terrorists are the enemies of the Iraqi people."
Self governance. Despite the attack, coalition chiefs are set to accelerate the handover in order to achieve full Iraqi self-governance. Government sources have indicated the two leaders have drawn up plans for a clear "exit strategy".

See also:

> Bush fails to show off in votings

> Lone Ranger Bush

For the detailed display one of the most important questions is if the article is displayed integrated in the website or in a new window. Both solutions have pro and cons and support different interaction styles.

Displaying the article in the content supports a more browsing-oriented perception of the website and invites for online reading and looking for related content. Opening a new window for an article "isolates" it from the website. Thereby the connection to the original source is enforced.

Elements of Detailed Article Display

The Article display consists of the following elements:

- Article Metadata (Source, Date and time, title, etc.)
- Content
- Links to related articles

- Print-Option

Optional Elements

- Display of other relevant metadata (e.g. author, subject classification)
- Add to binder
- Send/download as pdf option
- Linking of Keywords in content of article
- Back to top – Link
- Disclaimer & copyright notice

Metadata Display

In the detailed view all relevant metadata shall be displayed.

- ➢ Use different format and color consistent for differentiating the type of metadata and to indicate relevance.
- ➢ Titles or links in capital letters only should be avoided, as they reduce legibility.

Content Display

Articles may contain hypertext links if appropriate but by providing too many hyperlinks its easier for the users to loose the structural context of the side. Also the flow of reading can be disturbed by hyperlinks within the text.

- ➢ Use Textlinks within the content sparely in favour of the overall readability of the text.
- ➢ Font sizes should not be smaller than 9 – 9.5 pt at a screen resolution of 1024 x 768 pixels, and not smaller than 8 – 8.5 pt at a resolution of 800 x 600 pixels.
- ➢ Concerning contrasts, text is best to read with black font colour on a white background.
- ➢ The contrast of embedded links and text should be clearly recognizable but not disturb the reading flow of the user.
- ➢ Provide a Back to Top - Link if the page length exceeds the current window height.

Links to related articles

Providing links to related articles (e.g. the same keywords) can be a valuable add on for the user when used properly.

- ➢ Display suggested links in a way they to not disturb the reading of the main article but are clealy visible. Good positions are to the right or below the actual article.
- ➢ Provide enough information in the description so the user can decide if it is worth following.

Print

- ➢ This option shall open a new browser window which displays a print preview.
- ➢ Providing the user with different options (with/out pictures, big or small font) also might be helpful.
- ➢ Only content (article) shall be printed, not the navigation elements.

B.3.2.5 Different Language Settings

Language settings might be needed for different purposes or tasks during the search process:

- Selecting the **web interface** language
- Specifying the language of **query inputs**:
- Settings for the **search results** and **filtering**:

Web Interface Language

The language of the web interface itself should be chosen by language negotiation [for detailed information on this topic, see W3C, 0]. Additional options for the user to manually change this pre-setting should be offered.

Automatic language negotiation works with the user's browser configuration. Users may set their preferred languages for web sites in their browser and prioritize them. This information is sent to the server as part of the HTTP request, thus telling it which language version of the site should be delivered.

If the user does not set any language, browsers mostly specify the language of its user interface as the preferred language. This means that language negotiation should lead to reasonable results in most cases:

- The user's preferred language
- If not set: The user interface language of the browser he uses

Nevertheless, we have to consider that people may use browsers in an internet café somewhere in the world that is set to a foreign language, or that no one of the preferred languages is available. Then a default language should be chosen. Therefore it is necessary that users have the possibility to change the web interface language manually.

For this purpose there may be separate user interface elements on the web site or a language selection screen where the user has to select an interface language before reaching the actual website.

Such a language selection screen ensures that users chose their desired language for the interface, whereas the possibility to change it later on might be overlooked. On the other hand it is certainly annoying for returning users to be forced to make the same language selection over and over again. If this way is chosen anyway, it is necessary to save the user's settings (for instance through a technical solution like cookies or user data stored on the server) to avoid unnecessary annoyance.

Cookies or other methods should be used to care for the "stickiness" of the once set language, so that users do not have to select their language over and over again.

- ➢ Use language negotiation in combination with controls for the user to select a language manually.

- Use a technique for storing the language setting (e.g. cookies).
- Do not pose a language selection screen before the actual website, unless you have very good reasons to do so.
- The language selection elements must be clear to users who do not understand the current user interface language.
- Language negotiation should be provided on each page. This is particularly important for pages that can be bookmarked or accessed directly, not via the home page.

Query Input Language

The user could be provided with the possibility to specify the language of his query inputs. This might be done in the Advanced Search module. In the ideal case, this should not be necessary, because the system would be able to handle the language negotiation of the query input, but for certain inputs, for instance proper names or internationally used professional terms, it may be useful to have such an option. Details on this option are discussed in Advanced Search.

- If any possible, the user should not be bothered with specifying the language of his query inputs.

Language of Search Results and Filtering

If searches are conducted over articles in different languages, the user should be able to specify in which languages he wants to get results. This might be done before the query is submitted, but the system might also offer the possibility to filter the results after the search was executed. It is reasonable to store these settings permanently, but allow the users to change them whenever they wish to.

- Give users the possibility to select in which languages their results should be.
- If the option to select languages is not given within the search, provide the possibility to filter the results by languages.

The Advanced Search module should provide options to set the preferred languages for the results of a certain query (Advanced Search).

For Simple Search there might also be the possibility to make a language selection. This can be attained through a link to a language selection dialogue. Additionally, there might be the option to specify languages in the query string in the input field (see also “Simple Search”).

It is recommended to provide a language filtering option for the results, since language selection is an important feature for multilingual search, in particular if the languages can not be set in the search module (e.g. for Simple Search).

Association of Language Setting and Filtering Modules

For language settings as well as for the filtering options, a similar interface module is required. It needs the same set of languages to chose from, and the selected languages should be the same that the user selected before submitting the search**.** The language filtering module should be available whenever the user views his search results.

The similarity and related functionallity of these modules suggests to combine them into one module. Advantages of this solution would be that the user knows and recognizes this module as the overall element for language selections. It might be on the same position in the user interface, which adds to this positive effect. Furthermore, the user understands easier, that settings made in this module are not only temporary (e.g. for one search), but stay the same until he changes them, and are valid for both search and filtering operations. The aim is for each user to have his or her set of languages that he uses most often and that contains all the languages he knows.

The problem of such a combined module lies in the different functionallity of the language selection for search on the one hand and filtering on the other hand: For search, language selection is one of several search parameters that will be accounted for when submitting the search. For filtering, the selection of languages triggers an action, namely a manipulation of the search results.

The second problem that has to be taken into consideration is the necessary integration of the module into the respective tasks: search and manipulation of results, Simple Search as well as Advanced Search. The user has to see the relation to his tasks and know the effect that the module has on them. This might not be given, if the language selection is designed as a separate module in a fixed place on the screen.

Because task orientation, and therefore the integration of the language options into the respective modules, obtains priority, the following solution is recommended:

- ➢ Design a language selection module and a language filtering module that look the same as far as possible (same order and alignment of languages, same styles), and integrate them into the respective tasks, e.g. among the search options of Advanced Search or in a right column among the result manipulation options.

In this solution, differences between the modules are possible and might help to make the funktionallity clear to the user. For instance, the titles can be different and more precise, and the Language Filtering module should provide an action button that triggers the filtering action.

B.3.2.6 The Language Selection Module

The Language Selection module is to be integrated into or placed next to the respective search modules. Depending on the conditions of the search modules and the number of available languages, different designs are recommended.

Language Selection for Simple Search

- ➢ The Simple Search module should provide the possibility to enter language options in the query input field.

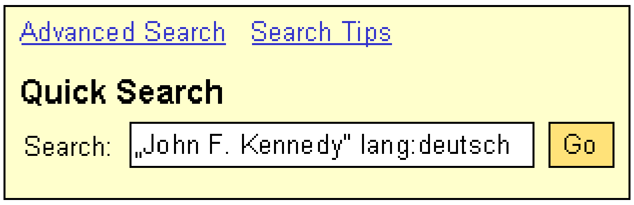


Figure 73: Example: Language option within the query string.

This selection is only valid for this certain query.

- ➢ The Simple Search module does not need separate UI elements for language settings, particularly if an option to filter the results is provided.

- ➢ If there are good reasons to provide a separate UI element for language settings, do not give them a too prominent representation in the module. A simple link to the settings should be sufficient.

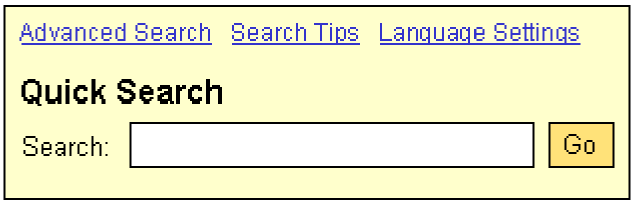


Figure 74: Example: Language settings link.

The language settings can open in a popup window or in the content area of the screen. They should be designed according to language settings for the Advanced Search module. The settings should be stored permanently and used for all search modules and for filtering.

Language Selection for Advanced Search

The design of language selection depends on the number of languages available for the search.

- ➢ If technically possible, save the user's language settings.
- ➢ If the user already set his languages, use the previously saved settings.

2 or 3 Languages

- ➢ If the total number of languages available for the search is very small (not more than two or three), the language selection can be offered as a dropdown menu in the Advanced Search module.

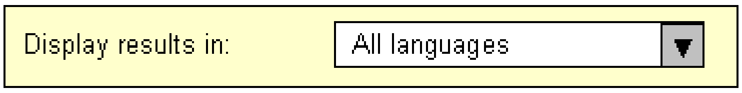


Figure 75: Example of simple language selection.

In this case, the user has the options to display one language or all languages at a time.

- ➢ Multiple selections are not possible in this design. For multiple selections, the next design should be chosen (see section Up to 8 Languages).

Up to 8 Languages

For up to 8 languages, all languages can be visible at the same time in the language selection module. The languages applied to a search can be chosen from this set of languages.

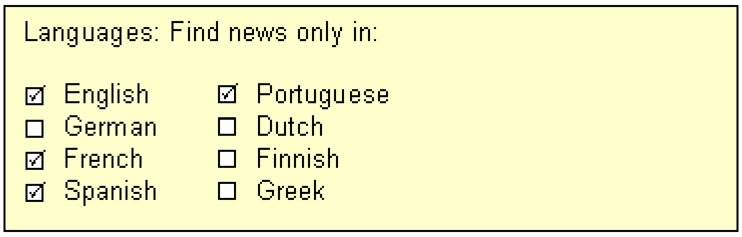
The user interface elements should display the languages to chose from and allow multiple selections. These requirements are met by either checkboxes or multiselection lists.

From a usability point of view, multiselection lists are not recommended for this purpose, because users have no clear overview of the selected items, specially if not the whole list is visible at once. Moreover, if multiple selections require the use of CTRL or Shift keys, additional explanation is necessary, for not all users know how to use them.

Checkboxes need more space, but give a good overview of the selected items.

➢ It is recommended to design a language selection module consisting of a group of checkboxes, that should be available whenever the user wants to conduct a search.

The language selection module allows the user to select in which languages he wants to get results. The user may select one or more languages out of a set of languages, and change this selection whenever he wants to. This language selection will have an effect on the displayed results whenever a search is executed. These settings should be stored and used for other tasks such as filtering the results.


Languages: Find news only in:

[✓ English]	[✓ Portuguese]
German	Dutch
[✓ French]	Finnish
[✓ Spanish]	Greek

Figure 76: Example of a language selection module.

More than 8 Languages

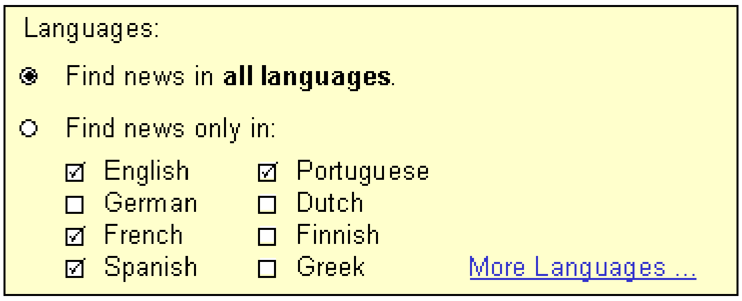
➢ If the total number of languages is higher than can be displayed as a set of checkboxes (more than 8 languages) the selection should be done in two steps.

For more than 8 languages, the choice should be made in 2 steps.

1. The user can select a set of languages by chosing from a list containing all available languages (read more in Language Preferences). These preferences should be permanently stored for each user and can be changed whenever the user wants to.

2. These languages are presented as a set of checkboxes, as shown above (Up to 8 Languages). From this set the user can select or unselect languages for certain queries. The set can be changed whenever the user wants to.

➢ Since the set of languages is restricted to a certain number of languages, the option to search all languages can be included.


Languages:

[✓ Find news in **all languages**.]

Find news only in:

[✓ English]	[✓ Portuguese]	
German	Dutch	
[✓ French]	Finnish	
[✓ Spanish]	Greek	More Languages ...

Figure 77: Example: Language selection module with option “all languages”.

➢ Before the user sets his preferences for the first time, the language selection module should contain a user specific set of default languages.

These default languages should be:

- The user's browser language
- Web interface language
- Languages of former queries of the user, if available

Elements of the Language Selection Module

In the following the Language Selection module for up to 8 languages is described in detail. It consists of:

- Title
- Goup of checkboxes
- Labels for checkboxes

Optional elements:

- A link to change the selected set of languages, if the language selection module does not contain all available languages: e.g. "Change Languages" or "More Languages …" Wordings like "Select Languages" or "Chose Languages" are not recommended in order to avoid confusions with the selection of the checkboxes.
- Radiobuttons to chose between "Find documents in all languages" and "Find documents in selected languages: …".

Title

Titles for the Language Selection module could be for instance:

- Search for articles in the following languages:
- Languages: Search for articles in:

➢ The title should contain the key word "Languages" and make clear what the module does.

"Languages: Search for articles in" contains the key word "Languages" at the beginning and therefore allows the user to understand quickly what the element is about.

Group of Checkboxes

Depending on the number of languages (not more than 8), the checkboxes should be aligned in 1 or 2 colums and 3 or 4 rows.

To enforce the feedback of a selection, the labels of the checkboxes may be displayed in an emphasized style.

Default:

- If the user already set his language preferences, these settings should be displayed by default.
- Otherwise all languages should be checked by default.

Labels for Checkboxes

- The Labels should be placed after the checkboxes.

They could be in the language of the web interface: e.g. English, German, Spanish, or in the respective language: e.g. English, Deutsch, Español.

- The labels should be in the language of the interface to be consistent.

B.3.2.7 The Language Filtering Module

The language filtering module allows the user to filter the results by language. The design of this module depends on the number of languages available for the search.

2 or 3 Languages

- If there are only two or three languages, language filtering can be offered using text links above the results list.

For instance:

”OmniPaper found **3900** results for ‘**gift**’.
Show only results in **English** (2.300) **German** (400) or **Dutch** (1.200)”

Or:

“OmniPaper found **2300** results for ‘**gift**’ in **English**.
We also found results in **German** (400) or **Dutch** (1.200)”

- In this example, if one certain language is selected, a link to display all languages again (“All Languages”) should be shown.

More than 3 Languages

- For more than 3 languages, the Language Filtering module should be designed like the Language Selection module (B.3.2.6 Language Selection Module).
- The functionality should be the same, except that the action, the filtering process, is triggered by an action button.

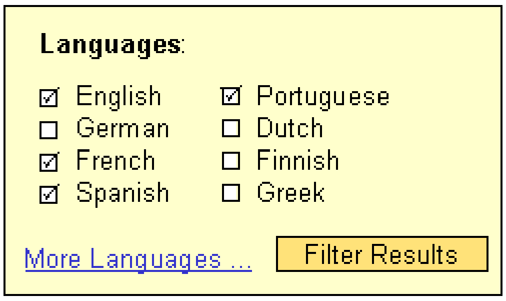


Figure 78: Example: Language filtering.

The user may select or deselect languages out of a set of languages. This set of languages should be the same as the one displayed in the Advanced Search module. Using this language filtering module may restrict or expand the search. The settings should be stored and used for other tasks such as new searches.

As in the language selection module, there should be the possibility to change the given set of languages, if there are more languages available than shown in the module.

Contrary to the language selection module, the language filtering module should contain a button to submit the changes. Since applying the changes will require a new search process and therefore lead to time delays, it is of advantage not to submit each selection at once, but let the user conduct several selections or de-selections and submit them with the hit of a button.

Elements of the Language Filtering Module

The language filtering module consists of:

- Title
- Group of checkboxes
- Labels for checkboxes
- Submit button

Optional elements:

- Radiobuttons to chose between “Find documents in all languages” and “Find documents in selected languages: …".
- A link to change the selected set of languages, if the language selection module does not contain all available languages: e.g. “Change Languages” or “More Languages …” Wordings like “Select Languages” or “Chose Languages” are not recommended in order to avoid confusions with the selection of the checkboxes.

Title

Titles for the Language selection module could be:

- Display articles in the following languages:
- Languages: Display articles in

- The title should contain the key word “Languages” and make clear what the module does.

“Languages: Search for articles in” contains the key word “Languages” at the beginning and therefore allows the user to understand quickly what the element is about.

Group of Checkboxes

- The group of checkboxes should be the same as in the language selection module.

Default: The same languages are selected as in the language selection module before the search.

Labels for Checkboxes

- The labels for the checkboxes should be the same as in the language selection module.

Submit Button

- The submit button should be in the bottom right corner of the module.
- The wording of the submit button should precisely describe the action triggered by the button, e.g. “Filter Results”

Positioning & Space

The Language selection module should be placed close to the list of results, coordinated with the other modules for manipulating the search results, for instance above the results or in a column on the right side of the results.

B.3.2.8 Language Preferences

- Language Preferences should be permanent settings that are saved at least for the time of one session, better is to find a way to store them permanently (e.g. as a cookie or in a user profile on the server).

Depending on the importance and frequency of use, this module might be accessible via a link in a menu, links in the search modules or the results module. The design of such a module depends on the total number of languages to chose from. The languages can be chosen from a multiselection list or a group of checkboxes.

The language preferences module described in the following is intended for the selection of several languages out of a high number of languages (more than eight). It provides the possibility to chose a set of languages out of all the languages available for the search. This set of languages can be shown in the Language Selection module and the Language Filtering module. It can be opened from the Language Selection module or the Language Filtering module as a popup window or in the content area.

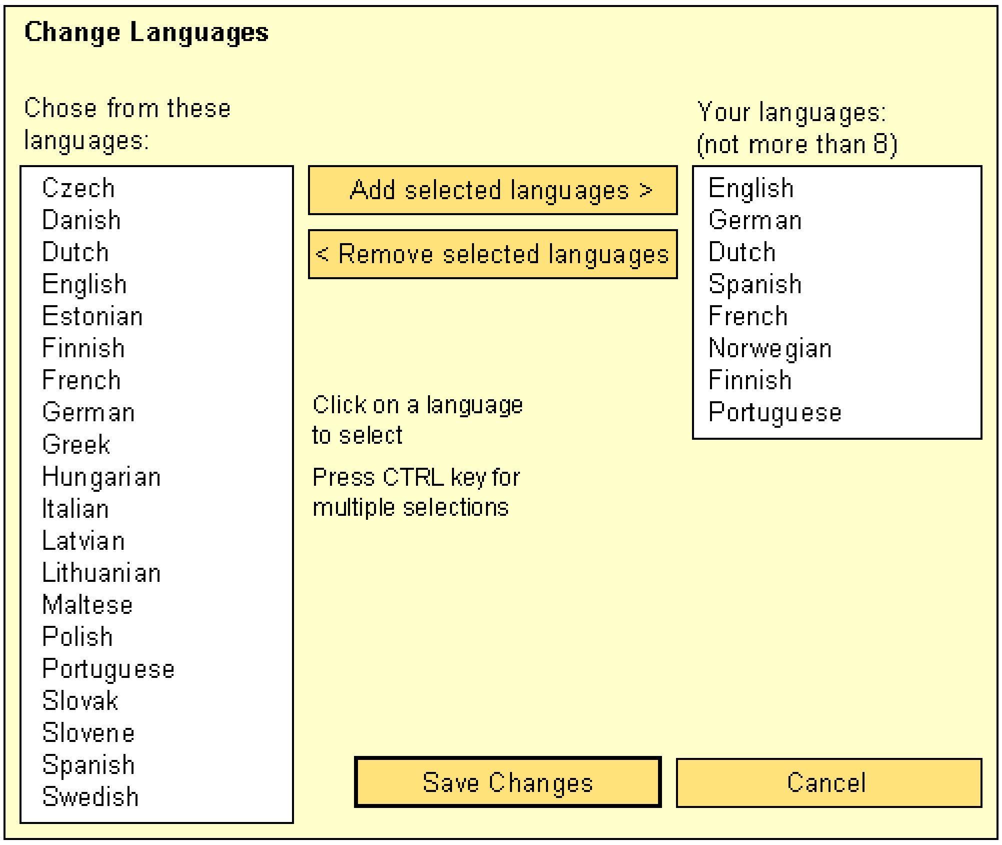


Figure 79: Example: Language Preferences.

Elements of the Language Preferences module

- List of all available languages
- List of selected languages
- Button for adding languages
- Button for deleting languages from the list of selected languages
- Button for finishing the task
- Button for cancelling the task

List of All Available Languages

➢ All available languages should be contained in a multiselection list.

➢ All or at least a large part of the languages should be visible at the same time.

Wording: The label should be “Chose from these languages:” or similar. “(Press CTRL key for multiple selections)” may be added if this applies.

Adjustment: This list should be placed on the left side of the module.

Default: none selected

List of Selected Languages

- ➢ The list of selected languages should be a multiselection list.
- ➢ By default, it contains all languages that are currently available in the language selection module and the language filtering module.

Wording:

The label should be “Your languages” or similar.

“Selected Languages” should not be used to avoid confusions with selecting items from the menues. The number of languages that can be chosen should be indicated, e.g. “(not more than 8)”

Adjustment: This list should be placed on the right side of the module.

Default: none selected

Button for Adding Languages

This button adds selected (marked) languages from the list of all languages to the list of selected languages.

- ➢ The added languages should be removed from the list of languages to chose from.
- ➢ Give the user feedback, if he tries to select more than the maximum number of languages, e.g. In a warning dialogue.

Wording: “Add selected languages”.

To make the function clear, additional arrows can be used.

Adjustment: It should be placed between the two lists.

Button for Removing Languages From the List of Selected Languages

This button removes selected (marked) languages from the list of selected languages. Removed languages should be added to the list of available languages. It should have the same style as the Button for adding languages.

Wording: “Remove selected languages”.

Adjustment: This button should be close to the list of selected languages, it might be left of the list or below the list.

Button for Finishing the Task

This button submits the selections, so that the changes take effect in the language selection module and the language filtering module. It closes the popup window or leads back to the previous screen, respectively.

Wording: “Save changes”

Default: By default this button should have the focus.

Adjustment: This button should be placed at the bottom of the module.

Button for Cancelling the Task

This button cancels the task. It closes the popup window or leads back to the previous screen, respectively.

Wording: “Cancel”

Adjustment: This button should be placed right of the button for finishing the task.

B.3.3 References

1. Abdelali, A., Cowie, J., Farwell, D. and Ogden, W., *UCLIR: a Multilingual Information Retrieval Tool*. In: Inteligencia Artificial N. 22 Vol VIII/2004. Madrid, 2004.

2. Hansen, P. et al, *User-Centered Interface Design for Cross-Language Information Retrieval*. Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. ACM Press, 2002.

3. Johnson, J., *GUI bloopers: don'ts and do's for software developers and Web designers.* Academic Press, San Francisco, 2000, pp. 342-343.

4. Nielsen, J., *Designing Web Usability: The Practice of Simplicity.* New Riders Publishing, Indianapolis, 2000, pp. 227-233.

5. Nielsen, J., *Query Reformulation, Advanced Search, First Page Results. Jakob Nielsen's Alertbox, May 13, 2001.* http://www.useit.com/alertbox/20010513.html

6. Nielsen, J., *Top Ten Guidelines for Homepage Usability. Jakob Nielsen's Alertbox, May 12, 2002*. http://www.useit.com/alertbox/20020512.html.

7. Ogden, W. C., Davis, M. W., *Improving Cross-Language Text Retrieval with Human Interactions*. 33rd Hawaii International Conference on System Sciences-Volume 3, Maui, Hawaii, 2000.

8. Peñas, A. et al., *Browsing by Phrases: Terminological Information in Interactive Multilingual Text Retrieval*. First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'01), Roanoke, Virginia, 2001.

9. Rieh, S. Y., *Patterns and Sequences of Multiple Query Reformulations in Web Searching: A Preliminary Study.* Proceedings of the 64th ASIST Annual Meeting, 38, 2001, pp. 246-255.

10. Spool, J. M., *Web Site Usability. A Designer's Guide.* Morgan Kaufmann Publishers, San Francisco, 1999, pp. 53-56.

11. Suvanaphen, E., Roberts, J. C., *Explicit verses Implicit: An Analysis of a Multiple Search Result Visualization*, Information Visualisation, Eighth International Conference on (IV'04), London, 2004.

12. Toms, E. G. et al., *The effect of task domain on search*. Proceedings of the 2003 conference of the Centre for Advanced Studies conference on Collaborative research. IBM Press, 2003.

W3C, Internationalization: FAQ: When to use language negotiation.
http://www.w3.org/International/questions/qa-when-lang-neg

PART C APPLICABILITY IN OTHER DOMAINS

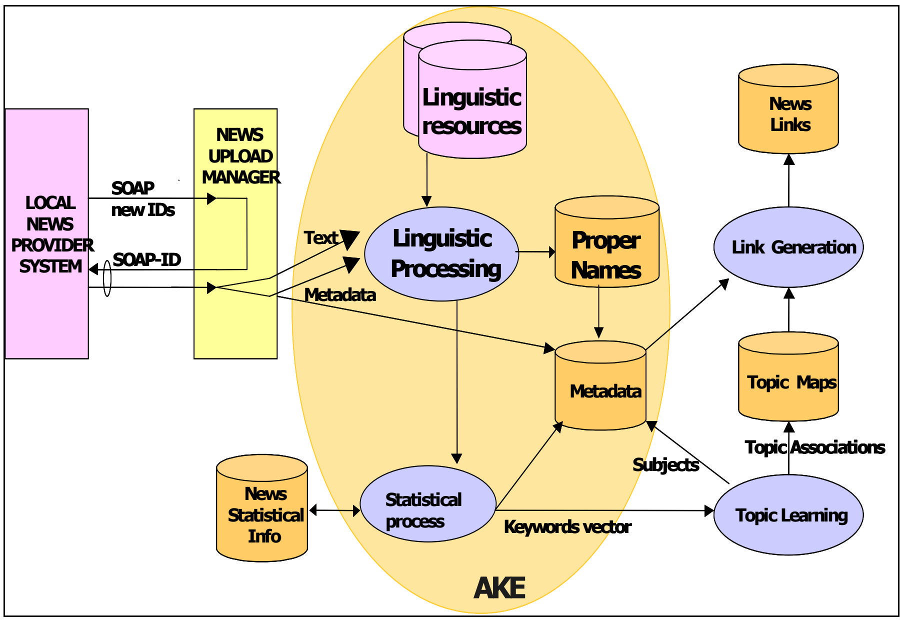
Several of the conclusions and guidelines described in this document are applicable to other content domains than news. This part first describes the work done by the OmniPaper consortium in CLEF (Cross-Language Evaluation Forum) throughout the project duration. The second section includes information about related projects that have proposed an architecture which is very similar to OmniPaper, like ERDDS and MIND.

C.1 OmniPaper at the Cross-Language Evaluation Forum

Among the functionalities provided by the OmniPaper service there is an on line indexing function, devoted to the reordering of the articles retrieved for a user search according to the likelihood of satisfying the query stated by the user.

Several component functionalities are provided by natural language processing. In fact the stop words identification (words without semantic relevance for the search process), proper names identification (to be taken into account as special words to carry out the search) are some of them.

Figure 80: AKE process



Currently, there are two main research topics focused on improving information retrieval technology. The first one concerns the user and his interaction with an information retrieval system, especially the issues of how to specify a query and how to interpret the answer provided by the system. The second one is related to the characterization of documents and how it affects the information retrieval process. Focusing on the second one, there are three main trends:

- **Semantic approaches** that try to implement some degree of syntactic and semantic analysis of queries and documents; this involves reproducing in a certain way the understanding of the natural language text.
- **Statistical approaches** that retrieve and rank documents according to the match of documents-query in terms of some statistical measure.
- **Mixed approaches** that combine both of them trying to complement the statistical approach with semantic approaches by integrating natural language processing (NLP) techniques, in order to enhance the representation of queries and documents and, consequently, to produce adequate levels of recall and precision.

In the OmniPaper project the third approach is adopted: first, the statistical approach is considered, and then semantic techniques complement the statistical framework through some kind of syntactic and semantic processing performed on the news and user queries, but in a shallow way (not for understanding the text). This approach requires having available linguistic resources for every language involved in the project, that is, semantic approaches are language and domain dependent.

The Automatic Keyword Extraction (AKE) process in the OmniPaper project make use of different NLP resources: stemmers for reducing words that differ only by suffixes to the same root, taggers for knowing the POS tags of words in documents and queries. Also other resources are going to be included as proper names heuristics for recognizing entities (personal, geographical and institutional names), semantic resources (such as EuroWordNet) for enriching query and document terms, and syntactic patterns for detecting multiword terms.

C.1.1 Keyword Recognition for document characterization

Two different approaches can be considered when trying to configure the relevant terms of a document; in the simplest approach, one term is one word and words are often preprocessed (they are stemmed, stopwords are dropped, etc). Clearly, this approach is language dependent.

In a most sophisticated approach, not only single words are considered as indexing terms, but also, a phrase can be a term. A phrase is "an indexing term that corresponds to the presence of two or more single word indexing terms", 0, although we consider two or more single words that can be or not indexing terms. The notion of phrase can be considered in a syntactical or a statistical way, 0. In the syntactical notion, techniques used for detecting the presence of phrases in the text will be NLP-oriented. On the other hand, if a statistical notion is preferred, methods based on n-gram (a window formed by a fixed number of characters that is sliced through the text character by character), 0, or on the co-occurrence of two words in the text, should be applied. This statistical notion has the advantage of language independence, as it does not require preprocessing of words and elimination of stopwords.

Whatever the types of terms are, numeric weights are commonly assigned to document and query terms. The "weight" is usually a measure of how effective the given term is likely to be in distinguishing the given document from other documents in the collection (between 0 and 1 if it is normalized). Weights can also be assigned to the terms in a query.

Some approaches found in the literature to decide what are the keywords of a document are: 0 recommend that terms found repeatedly in a document were appropriate for indexing and based the weights ranking by means of term frequency relative to frequency in the overall corpus;0 argued that words found in the document under study, but rarely in other documents, were important and developed the inverse document frequency (idf) as a term score; 0 combined the idf with the in-document frequency by taking their product as a measure of term importance.

Statistical frameworks break documents and queries into terms; these terms represent the population that is counted and measured statistically. In information retrieval tasks, what the user really wants is to retrieve documents that are *about* certain concepts and these concepts are described by a set of keywords. Of course, a document may deal with multiple topics. Generally speaking, the set of terms that describe a document is composed of all the words (or phrases) of the document except stopwords; optionally, these *significant* words could be stemmed. Moreover, not every word is used for indexing a document: usually, a filtering method is performed in order to select the most adequate, that configure the keywords of a document.

In the vector space model documents are represented as a set of keywords extracted from the documents themselves. The union of all set of terms is the set of terms that represents the entire collection and defines a "space" such that each distinct term represents one dimension in that space. Since each document is represented as a set of terms, this space is the "document space".

In the document space, each document is defined by the weights of the terms that represent it (user queries are represented in the same way), that is, the vector dj = (wd_{j1}, wd_{j2} , …., wd_{jm}) where *m* is the cardinality of the set of terms and wd_{ji} represents the weight of term *i* in document *j*.

The most successful and widely used scheme for automatic generation of weights is the **"term frequency * inverse document frequency"** weighting scheme, abbreviated "tf * idf"; this is the approach used in the OmniPaper project.

The expression used for the term frequency could be the number of times a word appears in the document normalized to allow for variation in document size. The term frequency (TF) for term *i* in document *j* is $tf_{ij} = n / maxtf_j$ where *n* is the number of occurrences of term *i* in document *j*, and $maxtf_j$ is the maximum term frequency for any word in document *j*.

The inverse document frequency (IDF) for word *i* is $idf_i = \log_2(N/ n_i)+1$ where *N* is the total number of documents in the collection and n_i is the number of documents in the collection where the word *i* appears.

Computing the weight of a given term in a given document as tf*idf implies that the best descriptors of a given document will be terms that occur a good deal in the given document and very little in other documents. Similarly, a term that occurs a moderate number of times in a moderate proportion of the documents in the collection will also be a good descriptor. Of course, there are variations to this scheme.

Could this approach be considered as a valid one? As it is well known, Zipf law empirically stated that not every word that appears in a document is helpful for characterizing a document: only words with a good discrimination capability are useful to build a document characterization. Words that are present in all documents do not allow to recognize a subset of documents in the collection. On the contrary, if a word appears only in one document, it only discriminates one document in the entire collection, so it is not useful to characterize a document subset in the collection.

As a matter of fact, the final objective pursued by the vector components is to categorize documents in the collection according to the query stated by the user. Vector components should thus be selected with this purpose in mind. Zipf law stablished a frequency threshold used for selecting some document words that will become vector components. In a generic document collection, if a word appears in more than 10 per cent of documents and in less than 90 per cent of documents, then that word should be used as a document keyword. Our tests have tried different thresholds, as presented in Section 5.

This weighting schema assumes the existence of a static collection of documents on which each query formulated by a user is applied. For instance, what happens if a new document is added to the set?, that is, if there is not a fixed collection of documents. In this case, every idf measure should be recalculated, and index term selection according to the selected frequency threshold should be performed again.

Usually, it is possible to provide a training set of typical documents for which idf frequencies can be calculated. This implies the assumption that all the subsequent documents received by the system will have the same "statistical properties" as the training set 0; the alternative is to update the training set regularly.

Our approach to this is an incremental one. The idf measure is updated with each new document; subsequent document components will therefore be computed with actual values, but the components previously computed become progressively obsolete. Under the above assumption, this obsolescence might not be a serious problem. However, to be safe, all vector components are recomputed at selected points in time. In this way, the document collection is incrementally maintained updated.

As final remark, it seems to be agreed upon that the occurrence frequency of a word in the document under study should not be predicted by observing the remainder in order to consider that word a good keyword. Among the most recent algorithms for extracting keywords are KEA, 0 treats keyphrase extraction as a machine learning problem and the work of 0 about using phrases for browsing multilingual documents.

C.1.2 AKE Description

This section is not available in the public version of this document.

C.1.3 AKE evaluation

The evaluation was performed in three steps. First round evaluation was done with a locally test: 1881 news were provided by MyNews and several partners working together in order to produce the set of queries and the identification of the correct answer. Second evaluation was performed using the UPM experience at the CLEF forum (Aug 2004). Finally the third step was done locally when integrated in the final prototype as a ranking service .

C.1.3.1 AKE first evaluation step

AKE Subsystem has been tested using a news collection supplied by content providers involved in OmniPaper project. This collection is formed by 1.881 English news articles published during Sept. 2002. Each article is described in XML, including some metadata and the average document length (stop words removed) is 250 words.

There are two basic configuration parameters that must be taken into account for test and evaluation purposes. These parameters are:

- *Stemming:* If stemming is applied words will be represented by a canonical form, identified by the word stem. This would lead to a grouping of words into the same representative stem, reducing dictionary size.

- vector dimension and keywords quality. Tests ran for evaluating the system consider the following FTs:

- *Frequency Thresholds (FT):* As mentioned in previous sections, keywords are selected according to the word DF into the whole collection. FTs are crucial in determining *5% - 90% of total documents in collection*. This threshold has been established considering empirical results. In this experimental work, the minimum *a priori* FT of 10% turned to be ineffective because some documents were not assigned any keyword.

    - *0% - 100% of total documents in collection*. In this test no keyword selection is applied according to DF. It will be interesting to evaluate differences in processing times in the future.

    - *5% - 100% of total documents in collection.*

    - *0% - 90% of total documents in collection*.

Results of performance evaluation obtained for the different executions carried out with the system are summarised in the following. Tests have been ran over a computer with an Intel Pentium III 800 MHz processor with 256 MB of RAM.

When applying stemming, the total number of dictionary entries is considerably reduced, as well as processing time. Considering execution times and storage requirements, stemming leads to a more efficient system. It will be necessary to prove if the values of traditional quality measures for Information Retrieval systems, like *recall* and *precision,* are better when stemming is applied.

Stemming process leads to a greater number of keywords per document but a smaller number of total keywords in the document collection. This effect is due to variations in words frequency distribution. With stemming, words are grouped under the same stem, so that more stems surpass minimum FT than simple words in the no stemming approach. However, there are less stems in total than simple words.

The more restrictive FT is Minimum Frequency Threshold. If figures from 5% - 90% and 5% - 100% thresholds are compared, differences in average keywords per document and total keywords in collection are very similar. On the contrary, comparison between 5% - 90% and 0% - 90% thresholds show greater differences. These results could point out some deficiencies in the tokenization process, as a lot of different words are appearing in very few documents. This point will be checked according to word lists obtained from the tokenizer.

C.1.3.2 Cross-Language Evaluation Forum

The Cross-Language Evaluation Forum is an annual conference whose main objective is to constitute a reference framework to evaluate and compare multilingual (cross-language) information retrieval systems and approaches. The Forum is organized as a context among research groups which are working in the information retrieval area. The organization provides all participant groups with a document collection including over 1.5 million documents in 9 different languages (Spanish, English, French, German, Italian, Finnish, Swedish, Russian and Chinese) and also proposes a set of topics: structured statements of information needs from which queries are extracted and which are then searched in the document collection. The main goal is to evaluate and compare the different systems by performing relevance assessments with the aim to create a community of researchers and developers studying the same problems and to facilitate collaborative initiatives between groups.

CLEF offers a series of evaluation tracks to test different aspects of cross-language information retrieval system development: monolingual, bilingual and multilingual information retrieval, image search (ImageCLEF), mono- and cross-language information retrieval on structured scientific data (GIRT), interactive cross-language information retrieval (iCLEF), multiple language question answering systems (QA-CLEF) and cross-language spoken document retrieval (CL-SDR).

The MIRACLE (Multilingual Information RetrievAl at CLEF) team is a joint effort of different research groups from two universities and one private company, with a strong common interest in all aspects of information retrieval and a long-lasting cooperation in numerous projects. Different experiments were submitted to the CLEF 2003 campaign main track, in the context of monolingual (Spanish, English, German and French), bilingual (from Spanish and French to English and from Italian to Spanish) and multilingual-4 (French, English, German and Spanish languages) tasks.

Our approach focuses on the mixed approach combining statistical and linguistic resources. Techniques vary from automatic machine translation, strategies for query construction, relevance feedback to topic term semantic expansion using WordNet. The main aim behind the MIRACLE participation is to compare how these different retrieval techniques affect retrieval performance. We also participated in ImageCLEF track (for a description of our work and obtained results see 0).

ImageCLEF 0 is a pilot experiment first run at CLEF 2003, which consisted on cross-language image retrieval using textual captions. A collection of nearly 30,000 black and white images from the Eurovision St Andrews Photographic Collection was provided by the task coordinators. Each image had an English caption (of about 50 words). Sets of 50 topics in English, French, German, Italian, Spanish and Dutch were also provided. Non-English topics were obtained as human translations of the original English ones, which also included a narrative explanation of what should be considered relevant for each image. The proposed experiments were designed to retrieve the relevant images of the collection using different query languages, therefore having to deal with monolingual and bilingual image retrieval (multilingual retrieval was not possible as the document collection was written only in one language).

C.1.3.3 Evaluation of the Automatic Keyword Extraction Module with CLEF test sets

In a previously mentioned, the Automatic Keyword Extraction module is based on the Vector Space model and, to support the process of selecting the document words that are going to be used for characterization purposes, statistical Natural Language Processing techniques are described. In particular, techniques implemented in the AKE prototype are: stemming, automatic proper noun detection and bigram detection. So, for evaluation purposes, three parameters where identified, one for each technique, and used to define the experiments to be run. These experiments are summarized in Table 9. The document collection used in the evaluation process was a set of 56.000 articles, approximately, taken form the Glasgow Herald. This collection is part of one of the document collection provided by the CLEF organization and the set of queries and relevance judgements used to evaluate the AKE prototype was also part of this collection.

	Stemming?	**Proper Name?**	**2-Word Gram**	**Query Section**
Exp1-3	No	No	No	Title Title + Description Title + Descr. + Narrative
Exp4-6	Yes	No	No	Title Title + Description Title + Descr. + Narrative
Exp7-9	No	Yes	No	Title Title + Description Title + Descr. + Narrative
Exp10-12	No	No	Yes	Title Title + Description Title + Descr. + Narrative
Exp13-15	Yes	Yes	No	Title Title + Description Title + Descr. + Narrative

Table 9 Description of experiments defined for AKE evaluation with the CLEF dataset

Obtained results for these experiments are shown in the figures bellow. The queries used for evaluation had a structure divided in three fields: a title, with some words about the main subject of the query; a description, a longer description for the query subject and a narrative, where two or three paragraphs are provided to give a detailed description of the query. Figure 81 depicts the results when only the title of the query is used to pose the search. Figure 82 shows results when the title and the description fields for the query are applied and Figure 83 shows the situation when all three query fields are used.

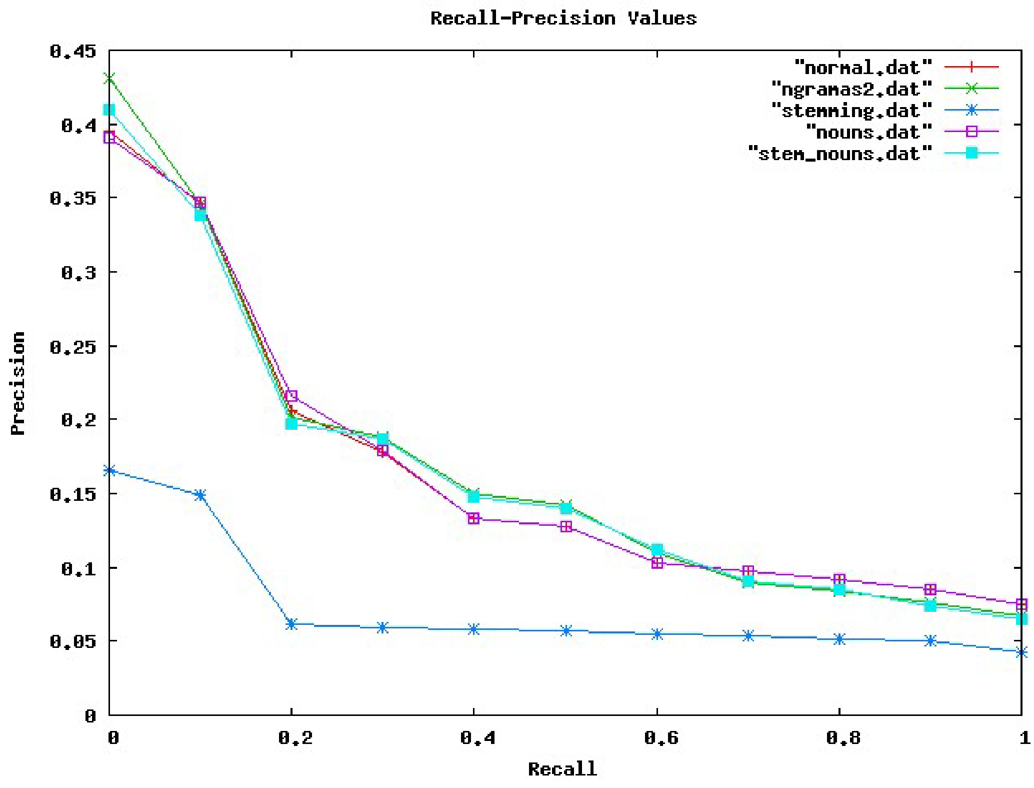


Figure 81 Evaluation results when the Title field of the query is used

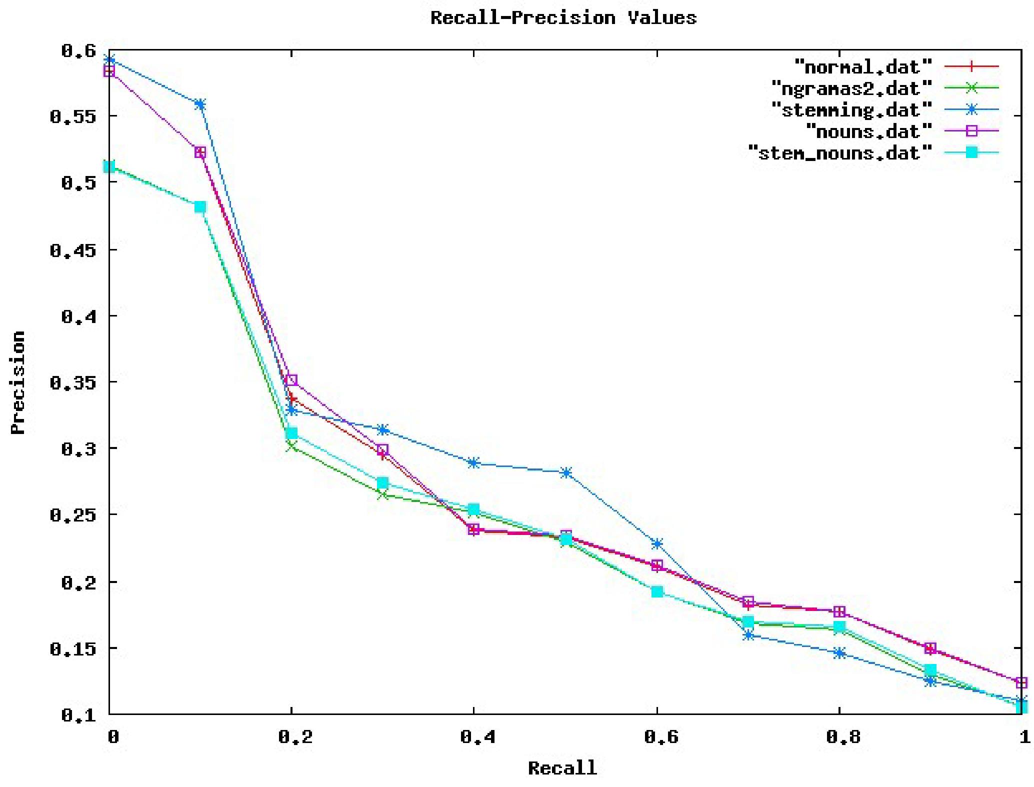


Figure 82 Evaluation results when the Title and Description fields of the query are used

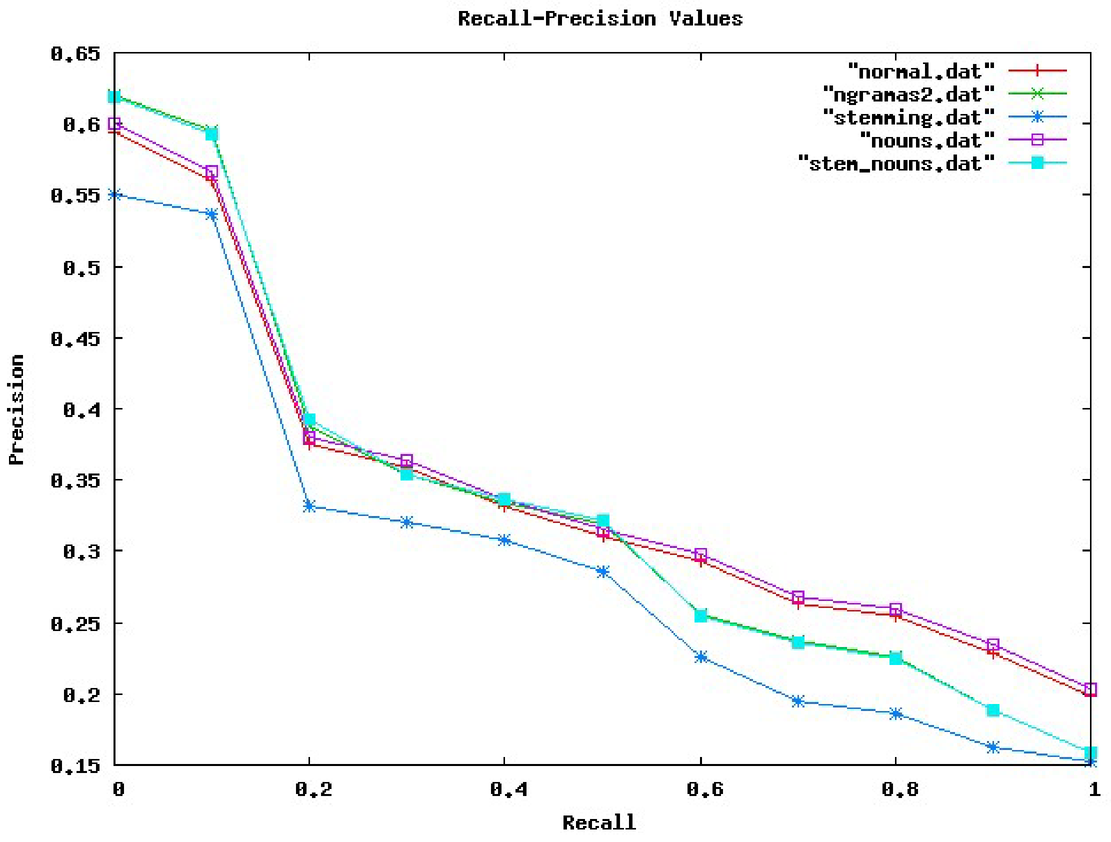


Figure 83 Evaluation results when the Title, Description and Narrative fields of the query are used

Some conclusions can be made taking into account these graphs. First, the use of long queries improves precision and recall values. As can be seen, when both title and description fields are used, precision gets the maximum value. It is worth mentioning that increasing the query length adding the narrative fields do not produce a substantial benefit. Second, the use of stemming improves results when medium size queries are posed to the system. These results strengthen previous experiments where stemming has been applied 0,0. A similar conclusion can be drawn from experiments where proper noun detection has been applied. In this situation, it worth the value to further analyze the use of more refined techniques, like entity recognition systems. Finally, if attention is paid to the use of statistical techniques like n-grams, no improvement is done when bi-gram detection is activated. Further evaluation would be needed to discard the use of this kind of technique.

- The use of long queries improves precision and recall values
- The use of stemming improves results when medium size queries are posed to the system

Information Retrieval techniques applied in the OmniPaper development have also been tested in CLEF 2004 campaign 0, 0, which workshop was held in Bath, from 15 to 17 September. In this issue, new configuration of several statistic and linguistic techniques have been tested, including multilingual aspects and working with languages using character sets such as Cyrillic for Russian.

Apart from the use of CLEF test sets, an article collection, along with several queries and relevance judgements, has been developed by the OmniPaper consortium. This test set was built using 1.881 articles provided by MyNews Online, one of the companies involved in the project. Figure 84 shows results obtained by the AKE prototype using this test set.

As can be seen, recall and precision values obtained with this evaluation data are very small. This can be due to the process followed to build the test set, where a very small collection is used and the relevance judgements provided are biased by the judge who made them.

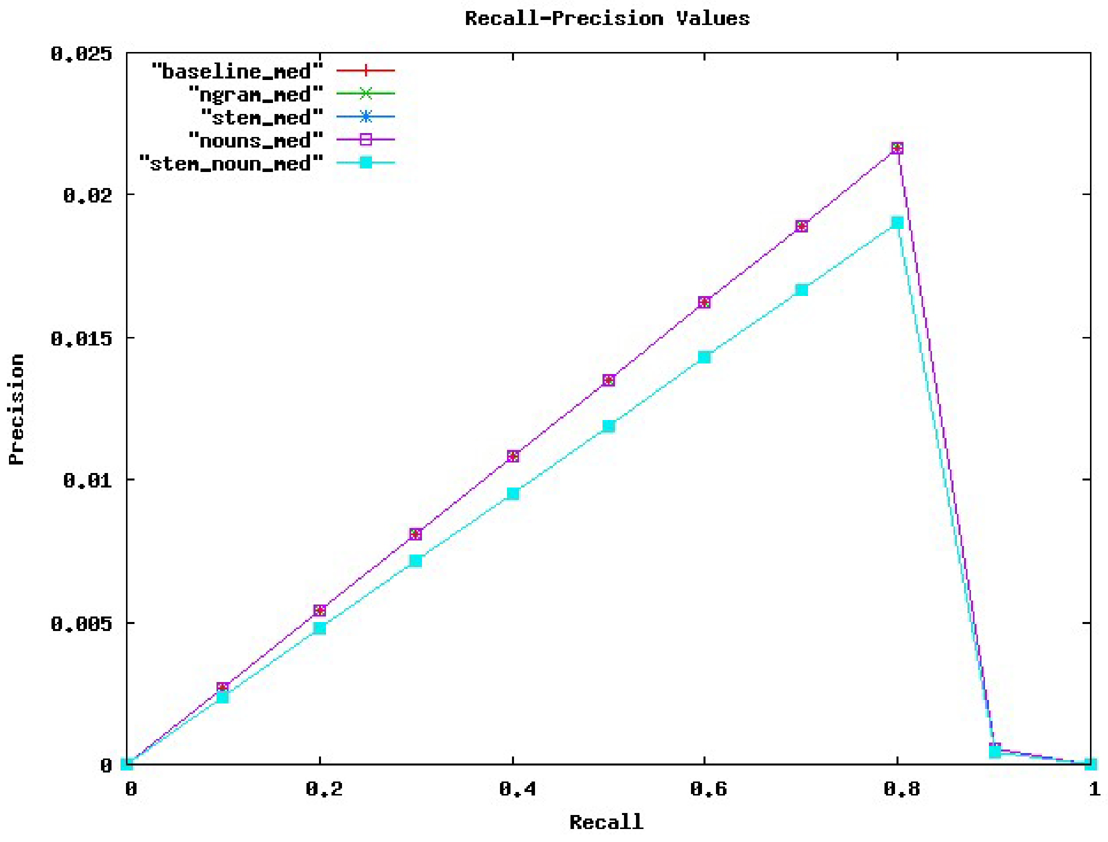


Figure 84 Evaluation results when the OmniPaper developed test set is applied

C.1.4 Natural language Processing

Expected advances in the processing of speech and written language are both crucial to allow a (nearly) universal access to the on-line information and services. On the other hand, as the importance of information extraction functionalities increases in systems that helps users in daily life, the human language technology is needed to the management of the big amount of on-information either in public domain or commercially. This situation is shared by the IR so it can be identified three main research lines to improve the current information retrieval technology:

a) The first one concerns the user and his interaction with an information retrieval system, especially the issues of how to specify a query and how to interpret the answer provided by the system.

b) The second one is related to the characterization of documents and how it affects the information retrieval process.

c) The third one is devoted to prove the benefits of using linguistic resources during the user consult or to the search itself.

Our approach consists in the development of a framework with statistical and linguistic techniques in order to evaluate the benefits of different processes defined using both kinds of techniques.

Up to know our work has been related with the statistical approach for AKE development and the integration of simple linguistic techniques to complement the statistical framework. The kind of morphological and semantic processing performed in a shallow way (not for a complete understanding the text) on the news and user queries, is done using linguistic resources for different languages (English, Spanish, German and French at the project).

The update of the different databases of the system containing documents, metadata from the documents, topic maps, and others required the coupling of the used statistical-based techniques with:

a) available linguistic resources for every language involved in the project as

- morphological taggers or analyzers in order to allow multilingual process,
- stemmers for reducing words that differ only by suffixes to the same root or just into a sequence of a fixed number of letters,
- taggers for identify the Part Of Speech (POS) tags of words in documents and queries,
- syntactic analyzers or segmenters for phrase identification or multi-words as “the rights of the child”
- semantic lexicons and heuristics for proper names and entity recognition.

b) lexical ontologies that are language and domain dependent for the incorporation of semantic and pragmatic processes.

Also not available linguistic resources has been developed in the project and included in the complete processes to improve the search as an entity recognizer (personal, geographical and institutional names or multiwords) and a phase identifier. Both are currently in the evaluation step to calculate the precision and the degree of enhancement of the search.

In the following, different works related with the OmniPaper techniques used at the implemented AKE are described.

C.1.4.1 Linguistic techniques and available resources

A Natural Language Processing system (NLP) has as generic goal to translate a source representation into a final representation that will be integrated in other system with special functionality [1]. The development of these systems must have into account the following aspects: what are the features the system must hold?, what are the type of texts the system works on?, what is the required functionality? (what are the processes that drive to this functionality?, how are these processes to be linked?, how each of them is carried out?), what is the system knowledge to perform its functionality? (how the knowledge pieces are related?, how is the knowledge represented?). All these questions are answering through the life cycle phases of a NLP system (analysis, design and implementation) and its study helps to design the knowledge model we present.

NLP research mainly has grown from symbolic or statistical systems approaches in computer science or linguistics departments, motivated by a desire to understand cognitive processes and therefore, the underlying theories from linguistics and psychology. Practical applications and broad coverage have been poorly worked until the later ten years when a growing interest in the use of engineering techniques allows new challenges in the field.

In real applications approaches based in complex grammars become to be difficult to maintain and reuse, so current applications employ simple grammars (different kinds of finite-state grammars to efficiently processing) and even some approaches do away with grammars and use statistical methods to find basic linguistic patterns.

The pre-processing analysis builds a representation of the input text. It consists in the analysis of chains of symbols by parser that maps each word to some canonical representation using different techniques as morphological analysis, morphological disambiguation, sallow parsing and others. This canonical representation is, sometimes, a linguistic one as the lexical root, the lemma (*body* for *bodies*), a stem by suppressing letters (*bodi* for *bodies*) or any other description domain dependent (less usual).

Stemming (reduction to a string) is usually associated with simple algorithms which do not do any morphological analysis. For example the Porter Algorithm (1980) [2], essentially:

- remove plurals, -ed, -ing,
- changes the terminal 'y' by 'i' when there's another vowel in stem,
- maps double suffixes to single ... –isation,
- deals with -ic, -full, -ness, take off -ant, -ence,
- remove -e if word > 2,
- …

The selection of available stemmers for languages different from English is not a simple task. The Porter algorithm successfully substitutes a morphological analysis for English (indexing purposes) but, it is not easy to find simple and successful algorithms for more inflectional languages with a morphology only slightly more complex than English. Porter and other stemming algorithms that do not extract the base form of words via morphological analysis, may fail to identify inflectional variants of terms in languages with a morphology only slightly more complex than English and may fail to relate synonym words and others given that they does not distinguish different meanings of the same word (lexical ambiguity is ignored).

A morphological analysis of a word form produces a set of possible base forms with associated inflectional information [3]. For each occurrence of a word form in context, a POS (Part-of-Speech) tagger discriminates which of these base forms is more likely in the context. A typical set of tags could be: CC (coordinating conjunction), CD (cardinal number), DT (determiner), NN (noun, singular or mass), etc. The disambiguation because homonyms, multiple functions of affixes or uncertainty about suffix and word boundaries, are solved by rule based and probabilistic approaches. The accuracy of hybrid taggers for English has remained constant from 94, whereas the rule-based disambiguator of Voutilainen has obtained the best results (99.7%).

The main morphological problem is the disambiguation of morphological alternatives, when the same morpheme may be realised in different ways depending on the context, dealing with the valid arrangements among stems, affixes and parts of compounds. Another difficult task concerns the management of the ambiguity related to the ability of appropriate uses of words in context by manipulation of syntactic or semantic properties of words.

For English, which is by far the most explored language, the morphology is simpler than for other languages. English Part-of-Speech tagging may help for example, identifying meaningful phrases, even most of languages are using a traditional approaches assuming that word use extensibility can be modelled by exhaustively describing the meaning of a word through closed enumeration of it senses. There is strong evidence that the mechanisms that govern lexical knowledge are related with word sense and several attempts have been made to develop a dynamic approach to polisemy (when there are multiple senses for the a word) and to create new aspects of word use [4].

One of the first available lexicons organized into a semantic map of words was WordNet®, a very large thesaurus created at Princeton University [5,6]. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different semantic relations[16] link the synonym sets, not definitions of meaning.

EuroWordNet (EWN) [6] is also available (through a licence agreement), a multilingual lexical database that includes several semantic relationships among words in English, Spanish, German and Italian languages. EWN is structured as a top concept ontology that reflects different explicit opposite relationships (v.g., animate, inanimate) and it can be seen as a representation of several vocabulary semantic fields. Moreover, it contains a hierarchy of domain tags that relate concepts in different subjects, for instance, sports, winter sports, water sports, etc.

EWN is a European initiative that has been developed by informatics to include linguistic expertise about words. It has been used in vary different applications. It is important to say that the direct use of the synonyms to expand the user queries in IR systems has always fail in precision and recall [7,8]. A new technique for the EWN help in semantic processes needed at the Information Extraction (IE) from multi-modal documents should be investigated. In OmniPaper project we are trying to find an aggregation method for determine the category of news.

A standard assumption in computationally oriented semantics is that knowledge of the meaning of a sentence is a function of the meaning of its constituents (Frege 1892). The modes of combination are largely determined by the syntactic structure and valid inferences from the truth conditions of a

[16] Synonymy - the related word is similar to the entry, as (PIPE, TUBE).

Antonymy - the terms are opposite in meaning, as (WET, DRY).

Hyponymy - one term is under or subordinate to the other, as (MAPLE, TREE).

Metonymy - one term is a part of the other, as (TWIG, TREE).

Troponymy - one term describes a manner of the other, as (WHISPER, SPEAK).

Entailment - one term implies the other, as (DIVORCE, MARRY).

sentence. But in real-life applications the statement of interpretation becomes extremely difficult especially when sentences are semantically but not syntactically ambiguous. For this reason in most applications the interpretation of sentences are given directly into an expression of some artificial or logical language from where an interpretation is inferred according to the context. Also this intermediate level of representation is needed, (for explicit reference into the representation) in order to capture the meanings of pronouns or other referentially dependent items, elliptical sentences or sentences ascribing mental states (beliefs, hopes and intentions).

Although some natural language processing tasks can be carried out using statistical or pattern matching techniques that do not involve deep semantics, performance improves if it is involved. But, for most current applications the predictive and evidential power of a general purpose grammar and a general control mechanism are insufficient for reasonable performance. The alternative is to devise grammars that specify directly how relationships relevant to the task may be expressed in natural language. For instance a grammar in which terminals stand for concepts, tasks or relationships and rules specify possible expressions of them could be used. Current approaches [9,10] are based in partial or shallow parsing[17]. These partial parses are further used in pragmatic issues in order to find an adequate context dependent interpretation. The problems in this approach are overcome by extending the resources with explicit models of the linguistic phenomena or by designing more robustly the linguistic analysis, but only before to get an insufficient performance again.

C.1.4.2 Keywords

Two different approaches can be considered when trying to configure the relevant terms of a document [12]; in the simplest approach, one term is one word and words are often preprocessed (they are stemmed, stopwords[18] are dropped, etc). In a more complex approach, not only single words are considered as indexing terms, but also, a phrase can be a term. A phrase is “an indexing term that corresponds to the presence of two or more single word indexing terms”, [13], although we consider two or more single words that can be or not indexing terms.

The notion of phrase can be considered in a syntactical or a statistical way, [14]. In the syntactical notion, techniques used for detecting the presence of phrases in the text will be NLP-oriented. On the other hand, methods based on n-grams[19] [15], or on the co-occurrence of two words in the text, should be applied. This statistical notion has the advantage of language independence, as it does not require preprocessing of words and elimination of stopwords.

Whatever the types of terms are, numeric weights are commonly assigned to document and query terms. The “weight” is usually a measure of how effective the given term is likely to be in distinguishing the given document from other documents in the collection (between 0 and 1 if it is normalized). Weights can also be assigned to the terms in a query.

---

[17] The term of *shallow syntax or parsing* refers to a less complete analysis that the output from a conventional parser to annotate texts with superficial syntactic information. It may identify some phrasal constituents, such as noun phrases with indication about neither their internal structure nor their function on the sentence. Also it can be identify the functional role of some words as the main verb and its direct arguments.

Shallow parsing normally works on top of morphological analysis and disambiguation to infer as much syntactic structure as possible from the morphological information and word order configuration. Work remains to be done on the integration of shallow with deeper analysis to solve co-ordination and ellipsis phenomena as well as in interfacing morphological descriptions with lexicon, syntax and semantics in a maximally informative way.

[18]Stopwords are words not relevant for information retrieval using statistical technologies (e.g. articles, prepositions, conjunctions, etc.)

[19] N-gram processing is a technique based on a window formed by a fixed number of characters (N) that is sliced through the text,

In defining what a keyword is, in bibliography different studies:

[16] recommend that the terms found repeatedly in a document are appropriate for indexing, so the weights can be ranking by means of term frequency relative to frequency in the overall corpus;

[17] argue that words found in the document under study, but rarely in other documents, are important, so the use of the inverse document frequency (idf) as a term score is appropriate;

[18] propose the combination of the idf with the in-document frequency by taking their product as a measure of term importance.

Statistical frameworks break documents and queries into terms; these terms represent the population that is counted and measured statistically. In information retrieval tasks, what the user really wants is to retrieve documents that are *about* certain concepts and these concepts are described by a set of keywords. Of course, a document may deal with multiple subjects. Generally speaking, the set of terms that describe a document is composed of all the words (or phrases) of the document except stopwords (this is idea of the so-called full-text search); optionally, these words could be stemmed[20].

Moreover, not every word is used for indexing a document: usually, a filtering method is performed in order to select the most adequate words, which configure the keywords of a document. This is the approach taken in the vector space model, described below.

C.1.4.3 Vector space model and AKE

In the vector space model documents are represented by a set of keywords extracted from the documents themselves. The union of all set of terms is the set of terms that represents the entire collection and defines a “space” such that each distinct term represents one dimension in that space. Since each document is represented as a set of terms, this space is the “document space”.

Could this approach be considered as a valid one? It was empirically stated that not every word that appears in a document is helpful for characterizing a document: only words with a good discrimination capability are useful to build a document characterization. Words that are present in all documents do not allow to recognize a subset of documents in the collection. On the contrary, if a word appears only in one document, it only discriminates one document in the entire collection, so it is not useful to characterize a document subset in the collection.

As a matter of fact, the final goal pursued by the vector components is to categorize documents in the collection according to the query stated by the user. Vector components should thus be selected with this purpose in mind. A frequency threshold is going to be use for selecting the words that will become vector components. In a generic document collection, if a word appears in more than 10 per cent of documents and in less than 90 per cent of documents, then that word should be used as a document keyword. Our tests have tried different thresholds. This weighting schema assumes the existence of a static collection of documents on which each query formulated by a user is applied. So, what happens if a new document is added to the set?, that is, if there is not a fixed collection of documents. In this case, every idf measure should be recalculated, and index term selection according to the selected frequency threshold should be performed again.

Usually, it is possible to provide a training set of typical documents for which idf frequencies can be calculated. This implies the assumption that all the subsequent documents received by the system will have the same “statistical properties” as the training set [19]; the alternative is to update the training set regularly. OmniPaper approach is an incremental one. The idf measure is updated with each new document; subsequent document components will therefore be computed with actual values, but the

[20] A keyword can be used for IR in different forms. For example *bodies* can be transformed into (a) lexical stem *bodi*, (b) stem *bod* (suppressing letters) (c) lemma *body*, (d) phrase *the body*.

components previously computed become progressively obsolete. Under the above assumption, to be safe, all vector components are recomputed at selected points in time. In this way, the document collection is incrementally maintained updated.

As final remark, it seems to be agreed upon that the occurrence frequency of a word in the document under study should not be predicted by observing the remainder in order to consider that word a good keyword. Among the most recent algorithms for extracting keywords, KEA [20] proposes the key-phrase extraction as a machine learning problem and the work of [8] propose the use of phrases for browsing multilingual documents.

C.1.5 References

The Association of Computational Linguistic
http://www1.cs.columbia.edu/~acl/home.html
ELRA (European Language Resources Association): http://www.icp.grenet.fr/ELRA
ELSNET (European Network in Language and Speech): http://www.elsnet.org
The Spanish association for NLP (SEPLN) http://www.sepln.org/
The Porter Stemming Algorithm page maintained by Martin Porter.
www.tartarus.org/~martin/PorterStemmer/
Free Translation, www.freetranslation.com
Ergane Translation Dictionaries, http://dictionaries.travlang.com
Xerox http://www.xrce.xerox.com/competencies/content-analysis/toolhome.en.html
Cole, R, Mariani, J., Uskoreit, H., Zaenen, A. and Zue, V. editors. *Survey of the State of the Art in Human Language Technology*. Studies in Natural Language Processing, Cambridge University Press, 1997
G.A. Miller. "***WordNet**: A lexical database for English*". Communications of the ACM, 38(11):39—41,1995.
WordNet http://www.cogsci.princeton.edu/~wn/
Eurowordnet: building a multilingual database with wordnets for several European languages. http://www.illc.uva.nl/EuroWordNet/
The Global WordNet Association http://www.globalwordnet.org/
García-Serrano A., Martínez P. and Ruiz A.**,** *Knowledge Engineering contribution to the enlargement of the quality in web-browsing.* IEEE International Workshop on Natural Language Processing and Knowledge Engineering (NLPKE) Tucson (Arizona), 2001.
Lopez-Ostenero, F., Gonzalo, J., Peñas, A. and Verdejo, F. Interactive Cross-Language Searching: phrases are better than terms for query formulation and refinement. In C.Peters, M.Braschler, J.Gonzalo, M.Kluck, editors, Advances in Cross-Language Information Retrieval, CLEF 2002. LNCS 2785, Springer-Verlag, 2003
P. Martínez y A. García-Serrano. *The Role of Knowledge-based Technology in Language Applications Development.* Expert Systems With Applications Journal, Vol. 19, pp. 31-44, 2000, Elsevier Science.
Paziencia M. T. (ed.) Information Extraction in the web era, LNAI 2700, 2003.
García-Serrano A., Martínez P. and Rodrigo L. *Adapting and extending lexical resources in a dialogue system*, Proceedings 39th Annual Meeting and 10th Conference of the European Chapter of the Association for Computational Linguistics (Workshop on Human Language Technology and Knowledge Management), Morgan Kaufmann Publishers, Toulouse, France 2001
Karen Sparck Jones y Peter Willet*, "Readings in Information Retrieval"*, Morgan Kaufmann Publishers, Inc. San Francisco, California, 1997.
Lewis D., *Representation and learning in information retrieval*. Technical Report UM-CS-1991-093. Department of Computer Science, Univ. of Massachusetts, Amherst, MA
Cohen, J. Highlights: Language and domain independent automatic indexing terms for abstracting. JASIS, 46(3), (1995), 162-174.
Chade-Meng, T., Yuan-Fang W., Chan-Doo, L. *The use of bigrams to enhace text categorization*, Information Processing and Management 38, pag. 529-546, 2002.
Luhn, H. A statistical approach to the mechanized encoding and searching of literary information. IBM Journal of Research and Development,1 (1957),309-317.
Sparck Jones, K. *Index term weighting*. Informa. Storage and Retrieval, 9 (1973), 619-633.

Salton, G., Yang,C. On the specification of terms values in automatic indexing. Journal of documentation, 29 (1973), 351-372.

Greengrass, E. Information Retrieval: A survey, November (2000).

Jones, S., Paynter, G. W. Automatic Extraction of Document Keyphrases for Use in Digital Libraries: Evaluation and Applications. JASIS, 53(8), (2002), 653-677.

Baeza-Yates and Riveiro-Meto, Modern Information Retrieval, Addison Wesley, 1999

Mendes Pereira T. S. , Baptista A.A., *The OmniPaper metadata rdf/xml prototype implementation* ELPUB Conference 2003

DAEDALUS – Data, Decisions and Language, S.A. www.daedalus.es

J.L. Martínez, J. Villena Román, J. Fombella, A. García-Serrano, A. Ruiz, P. Martínez, J.M. Goñi, J.C. González, Evaluation of MIRACLE approach results for CLEF 2003, Working Notes of CLEF Workshop, Trodheim, Norway, August 03.

José M. Goñi-Menoyo, José C. González, José L. Martínez-Fernández, Julio Villena-Román, Ana M. García-Serrano, Paloma Martínez-Fernández, César de Pablo-Sánchez and Javier Alonso-Sánchez, : MIRACLE's Hybrid Approaches to Bilingual and Monolingual Information Retrieval, Working Notes of CLEF 04 Workshop, Bath, UK, September 04.

José L. Martínez-Fernández, Ana García Serrano, Julio Villena, Víctor David Méndez Sáez, Santiago González Tortosa, Michelangelo Castagnone and Javier Alonso: MIRACLE at ImageCLEF 2004, Working Notes of CLEF 04 Workshop, Bath, UK, September 04.

Clough, P., Sanderson, M.: The CLEF 2003 Cross Language Image Retrieval Task. Working Notes for CLEF 2003 Workshop (Trondheim, Norway), Vol 1 (2003).

Villena, J., Martínez, J.L., Fombella, J., García Serrano, A., Ruiz, A., Martínez, P., Goñi, J.M., González, J.C.: *Evaluation of MIRACLE results for Image* CLEF 2003. Working Notes for the CLEF 2003 Workshop (21-22 August, Trondheim, Norway), Vol 1 (2003).

CLEF http://clef.iei.pi.cnr.it:2002/

Hernández J. Z., Garcia-Serrano A. *Intelligent e-assistance: a multiparadigm approach*, eChallenges IOS Press, 2003

Semantic Web http://www.w3.org/2001/sw/

Van Hemel S., Paepen B., Engelen J.,*Smart search in newspaper archives using topic maps* ELPUB Conference 2003

Peñas, A. Website Term Browser: Un sistema interactivo y multilingüe de búsqueda textual basado en técnicas lingüísticas. PhD Thesis, Dep. Lenguajes y SI, UNED, 2002.

C.2 Architectural Reuse for distributed information retrieval

This section includes information about related projects that have proposed an architecture which is very similar to OmniPaper.

C.2.1 ERDDS

The main objective of the ERDDS (The European Radiological Digital Data System) project is to collect and reuse digital information available from the Radiology Department throughout the EU, in order to define a referral European Radiological Classification system, in both medical and administrative domain, and to set up a European radiological digital data integration framework.

Medical Imaging did not experience, so far, a real breakthrough, mostly due to the lack of commonly accepted European radiology classification and taxonomy systems.

ERDDS addresses this relevant limiting factor and will:

- Make available, disseminate and support exhaustive information in the radiological sector, through its portal, with a simple and easy-to-use interface
- Define a unified European radiology framework, based on a glossary and the classification of Medical Imaging services
- Facilitate the creation and the growth of a European teleradiology sector.

C.2.1.1 ERDDS project goals and objectives

The ERDDS project objectives are to exploit the available digital contents on the Radiology Department throughout the EU, to define a European Radiological Classification, to be taken as a reference in both medical and administrative domain, and to set up a European radiological digital data integration framework.

This result will:

- Support integration and re-use of public sector medical knowledge, in order to facilitate the development of pan-European radiological digital data collections for professionals, policy makers and citizens.
- Make available, disseminate and support information and data for benchmarking, evaluation and standardization of radiology workflow and performances, including fares comparison among different EU healthcare systems
- Define a unfired European radiology framework, based on a glossary and a classification of Medical Imaging services, with a standardized, multilingual and integrated approach to support the RIS (Radiological Information System) , and HIS (Hospital Information System)
- Facilitate the creation and the growth of a European teleradiology sector, enabling contacts and providing services to public institutions and citizens, among different European Countries.

- Create an European data collection of public sector medical data information including benchmarking, evaluation and standardization of radiology workflow and performances.

The main activities in building a distributed framework of integrated radiology information services include:

- Data analysis, including legislations, administrative rules, radiological classification and radiology workflow applied in the different participant national health system. This activity will lead to detailed definition of data categories, data types and their use in the different step of radiology workflow;

- User requirements definition, focused on the definition of the needs and requirement which can be developed on the basis of the data, information and knowledge available;

- Analysis of business and operational model and analysis of market potential: this analysis will be done both in the private and in the public sector, and will allow a detailed specification of the services to be provided, including the operational and business model to be followed;

- IT infrastructure design and implementation: in this phase the design and the implementation of the IT infrastructure needed to support the service model defined in the previous phases will be carried out;

- Test and Demonstration: this phase will demonstrate the results of the project and will allow its evaluation, in a real context, and the detailed definition of the exploitation activities which are planned to be performed after the project termination.

- Awareness and dissemination is key to the success of the project. Project awareness and dissemination is managed directly by radiology scientific societies involved, through a dissemination initiatives such as European workshops and information days

- Project management activities needed to carry on the project will be performed following the guidelines defined, moreover proper methodologies will be adopted in order to assure high quality standards.

C.2.1.2 Related approach in ERDDS and OmniPaper

Experts from the ERDDS group and from OmniPaper met on different occasions to exchange information about technical details in the application of different problem areas.

During the kick-off meeting of the eContent projects in Luxembourg OmniPaper discovered from the presentation details of ERDDS that there are major overlapping in the architecture description of the distributed information retrieval and collection approach and offered their approach to distributed information retrieval for integrating news archives.

Early architecture descriptions of ERDDS show close connections to the OmniPaper approach and the technical team of OmniPaper has initiated an information exchange with the ERDDS team. Since ERDDS has started mid of 2004, results of OmniPaper can be valuable for the technical progress of the IT-integration work in ERDDS.

C.2.1.3 Project details and information

Partners

- IMS - Istituto di Management Sanitario Spa (IMS) (I)
- Società Italiana di Radiologia Medica (SIRM) (I)
- Sociedad Espanola de Radiologia Med (S)
- Faculty Hospital Motole, Radiological Department (FN Motol) (CZ)
- Softeco Sismat Spa (I)
- Sadiel S.A. (S)
- Societè Francaise de Radiologie, (SFR) (F)
- Hopital Europèen George Pompidou, Departiment de Radiologie (HEGP) (F)
- Fundation Hospital Alcorcon, Area Diagnostico por Imagen (FHA), Alcorcon (S)

Contacts :

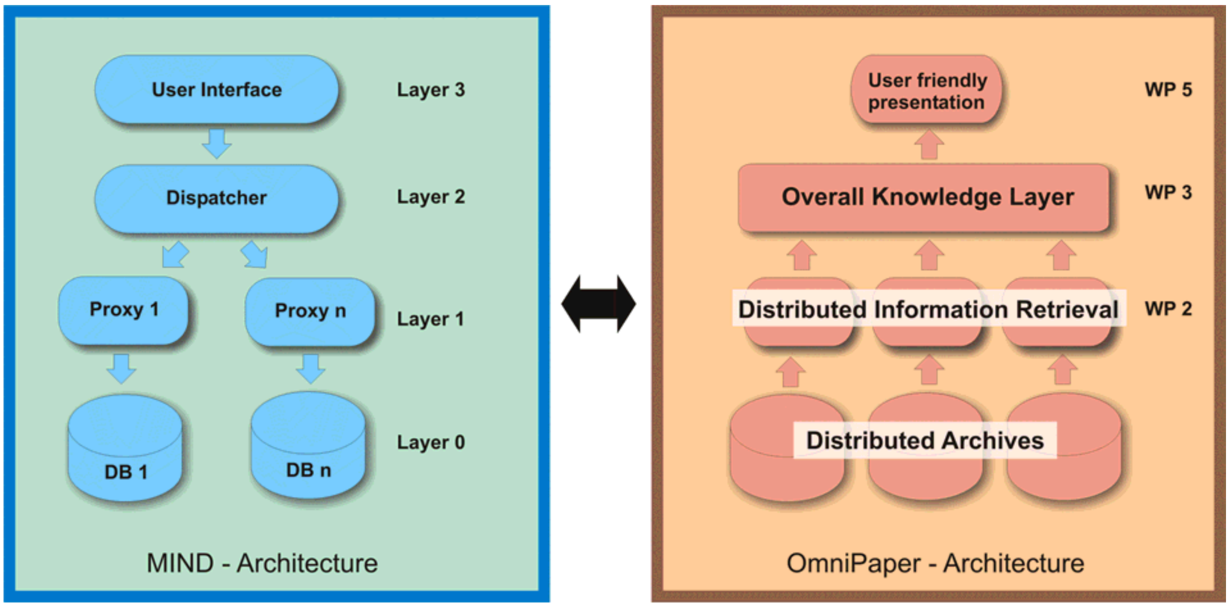
Dr. Enrico Morten
Softeco Sismat SpA
Phone: +39 10 6026 328
Fax: +39 10 6026 350
email: enrico.morten@softeco.it

C.2.2 Project MIND

The consortium investigated ongoing research activities in relevant fields of the OmniPaper project. Especially within a concertation meeting for European and US- Projects in the field of library applications, organised by the DELOS network and held on March 25 and 26, 2002 in Rome, valuable input was collected by analysing related projects. A list of relevant and interesting projects can be found in the annex. One of the projects closest related to the concept of “OmniPaper” is the “MIND-Project”.

The MIND approach provides an end-to-end solution for federated digital libraries which cover most of the problematic issues that are identified also in the OmniPaper project. Information retrieval techniques, retrieval quality and non-co-operating libraries with heterogeneous query interfaces are the research focus. OmniPaper extends the research work of the MIND project in exploiting a special application area, the digital news libraries.

The MIND project comes up with a similar architecture and self-defined names for the involved components. The following figures demonstrate the close relationship:



The top level is the „user interface", corresponding to OmniPaper's User friendly presentation.

Figure 85: Comparison of the proposed architecture of the projects "MIND" and "OmniPaper"

The intermediate level is called „dispatcher" and corresponds to our overall layer components (knowledge layer)

The bottom elements are called „proxies" and correspond to our „local layer components", providing individual access to the existing databases.

The main functionality within the MIND – concept is provided by the „smart proxies". "Local" components are integrated to reflect local functionalities and specifics of existing heterogeneous databases. In order to integrate the distributed information retrieval, MIND proposes the following concepts:

Query transformation:

the query might be rewritten with respect to the local news archive.

Similar to our approach in OmniPaper the "local" components have tasks that specifically match to the environment at the local database to be included. This environment involves data types, access methods, retrieval functionality, metadata extraction, etc.

Resource selection:

According to the query, a subset of the resources is selected with respect to query costs and quality. Information retrieval from distributed heterogeneous databases may vary in access costs. Costs include attributes as time to finish database access, query duration, result translation, keyword extraction, network connectivity, etc.

Data fusion:

the weights of the retrieved documents have to be normalized for optimum retrieval quality, corresponding results in multiple languages need to be identified.

Especially in the area of newspapers and news archives OmniPaper needs to identify appropriate measures of relevance and proximity of particular news articles in relation to other articles and documents from heterogeneous databases from different cultural, legal and ethnical sources.

About using different types of metadata (DC and RFC 1807 were mentioned). Data fusion after searching.

Uses a similar architecture as OmniPaper. MIND deals with automatic resource gathering, heterogeneity, multimedia, resource selection and data fusion in federated digital libraries.

http://www.mind-project.net, http://www.mind-project.org