Okapi in TIPS: The Changing Context of Information Retrieval

Murat Karamuftuoglu, Fabio Venuti

Centre for Interactive Systems Research
Department of Information Science
City University
{hmk, fabio}@soi.city.ac.uk

Abstract. In this paper the changing context of information retrieval is discussed by examining the role of the Okapi text retrieval system in the "Tools for Innovative Publishing in Science" (TIPS) project. TIPS project poses a number of new challenges for end-user probabilistic retrieval, including field-based searching, Web access, and integration with other software. These and some other problems involved in designing a sophisticated Web-based best match retrieval system are highlighted. The architecture of the implemented Okapi Web system, its integration with the other systems that comprise the TIPS portal and design and evaluation of the user interface are discussed.

1 Introduction

In this paper we discuss the changing "context" of information retrieval by examining the role of Okapi in the EC funded "Tools for Innovative Publishing in Science[1]" (TIPS) project [1].

Okapi is the name given to a family of experimental retrieval systems that have been developed over the last two decades[2]. It is based on the Robertson-Sparck Jones probabilistic model of searching [4]. Okapi started life as an online library catalogue system and since has been used as the basis for real services to groups of users in various contexts. The TIPS project uses Okapi as the main retrieval tool to index and retrieve full text papers in High Energy Physics (HEP) area and to serve the information needs of that scientific community. The overall aim of the TIPS project is to provide a unified, desktop-like access to various services and tools to be used by the HEP community [1]. While Okapi has been the subject of a considerable amount of research, the TIPS framework presents significant new challenges.

The next section describes the overall structure of a typical Okapi system and reviews briefly the past Okapi research in relation to the TIPS project. The following section discusses in some detail issues that surround the implementation of the Web-based Okapi retrieval service in TIPS. In this section the architecture of the Okapi Web system, its integration with the other systems that comprise the TIPS portal and

[1] Information Society Technologies Program: IST-1999-10419
[2] For an overview of past Okapi research see [2,3]

design and evaluation of the user interface are discussed. The final section summarises the main points examined in the paper.

2 Okapi and the Changing Context of Retrieval

A typical Okapi system involves (Fig. 1):

- A search engine, the BSS (Basic Search System), which provides efficient low level functionality for weighting and ranking searches, and also for full Boolean and pseudo-Boolean (proximity) operations. Term weighting and document ranking is based on the Robertson-Sparck Jones probabilistic model of searching [4, 5].
- Indexing routines for indexing and inputting text documents. These accept raw text files and create a database in a form suitable for searching.
- Various interface systems that provide different representations of the underlying BSS functionality.
- Query Model layer, which manages the interaction between the user interfaces and the BSS and supporting additional functions such as incremental query expansion and passage retrieval.

BSS and indexing routines run on Solaris & Linux for Sun and Intel architectures.

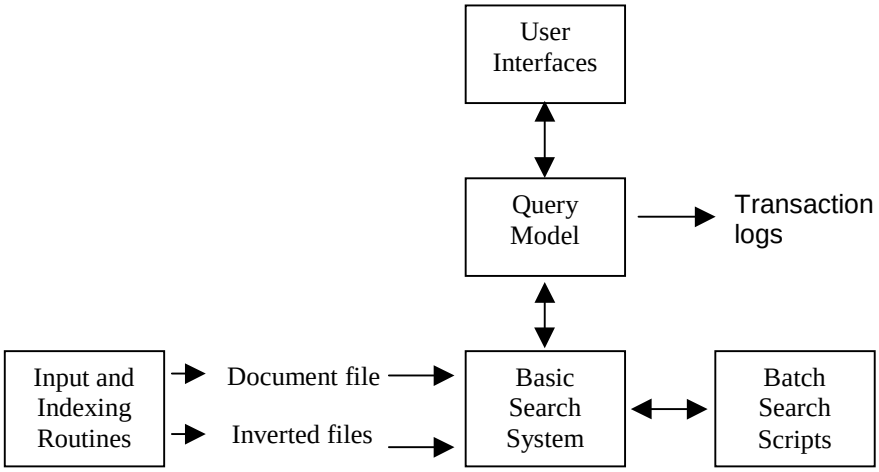

Fig. 1. Overview of Okapi architecture.

A typical Okapi search session involves the following:

User:
- Enters search terms.

System:
- Pre-processes, parses and stems the search terms to remove capitals, hyphens, punctuation and similar linguistic devices to convert the user input to the standard form used in indexing the documents in the databases. After input terms are pre-processed they are parsed to remove stop words and identify common phrases. The remaining terms are stemmed and weighted and document score are calculated. A ranked list of matching documents is presented to the user.

User:
- Provides relevance judgements feedback on the documents (relevance feedback).

System:
- Selects terms from the relevant documents, expands the query and performs a new search based on the expanded query. Term selection is based on the model described in [5].

There are four main sources of data used in query term weighting in Okapi [5]:

- Collection frequency - Terms which occur in only a few documents are likely to be more useful than ones occurring in many
- Term frequency - The more frequently a term appears in a document, the more important it is likely to be for that document
- Document length - A term that occurs the same number of times in a short document as in a long one is likely to be more important to the short document than it is to the long one
- Relevance information - The more frequently a term appears in relevant documents, the more important it is likely to be for that query

All these elements are combined to give a weight for each term-document combination, and then all these term weights are combined to give a total score for how well the document matches the query.

Okapi was set up specifically to provide an environment in which ideas could be tested in live settings that involve real users and information needs. Although some of the Okapi projects involve laboratory type experiments, as exemplified by the involvement in the TREC program, the main focus of Okapi-based projects has been to explore the inherent interactive nature of the retrieval process.

User-oriented Okapi research [2,3] in the past decade concerned with information seeking behaviour of users given different representations of the underlying probabilistic retrieval model at the user interface level. The past Okapi experiments involved a database of short bibliographic records (INSPEC) and/or library catalogues. The systems allowed users to enter “free-text” queries, which were matched against an index derived from the titles and abstracts of the records. Similarly the query expansion was based on the same index. These systems, hence, are basically free-text search tools that did not implement searching on specific fields

(e.g. author) or combination of fields (e.g. title and author). Users in these experiments were students and other members of the City University. Although they were all studying/researching in the general area of IS&T, they constituted rather a heterogeneous group with different levels of knowledge, skills and interest in the subject. Laboratory experiments in the context of TREC (both purely automatic runs and interactive experiments with human subjects as users) used full-text databases of various kinds, including newspaper and newswire stories. Again these were treated in a purely "free-text" fashion, with no field searching.

Involvement in the TIPS project poses a number of new challenges for end-user probabilistic retrieval: The document collections used in the project contain the full-texts of the articles as well as bibliographic information (or metadata) associated with the records, such as title, author names, date of publication, assigned keywords (controlled and/or free-text), and abstract. The source documents to be indexed are in TeX format that contain mathematical formulae and special symbols as well as plain (ascii) text. We have a research community to serve, who are experts in a particular area of Physics, namely HEP. The system will need to have client-server architecture and will be accessible on the World Wide Web using standard Web browsers. There will be *many* concurrent users of the system at any time. Finally, Okapi needs to work with and be integrated into other software developed by our partners in the project. These points are discussed further below.

Table 1. Changing context of the retrieval process.

Past	**Present**
Bibliographic (plain text)	Full-text (TeX format)
Free-text searching	Field searching
User community mainly novice	User community mainly experts
LAN-based	Web-based (HTTP)
Client and sever tightly coupled	Client and server separated
Few concurrent accesses	Many concurrent accesses
Stand-alone	Integrated with other software

3 Okapi in Tips

3.1 System Architecture

One of the main challenges in implementing a Web version of Okapi is that, HTTP is a stateless protocol. Between the requests sent to the Web server the client-server connection is broken, so information about the search history is not automatically maintained. However, one of the main defining characteristics of a probabilistic

retrieval system is that relies on the history of interactions between the user and the system within a given search session to optimise its performance. In particular it needs to store the current query state and documents selected as relevant by the user.

So the most important technical problem to be addressed in the Web environment is how to reconcile the Web's essentially "stateless" mode of operation with the tight browser-server integration and session continuity considered essential for an effective Okapi implementation. There are several different technologies available to maintain session continuity. Rapid evolution of the Web means that several new tools are added to the available set all the time, and selection of any one of the methods at a given moment may turn out to be non-optimal.

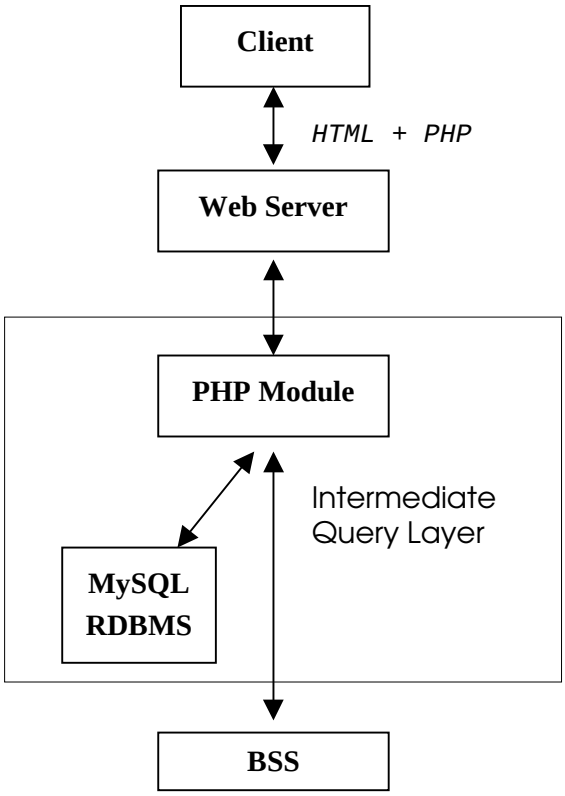

Fig. 2. Prototype Okapi Web System.

Fig. 2 illustrates the architecture of the Web-based system implemented. In this system, the client communicates with the Web server by means of PHP code embedded in a standard HTML document. Communication between the Web server and BSS is achieved via an Intermediate Query Layer (IQL), and session continuity is

maintained using a relational database holding data about the current retrieval state. IQL interacts with the BSS to perform various low-level BSS operations used in searching and retrieving documents and provide additional functions for query manipulation and expansion, plus transaction logging. The Web browser acts as a *thin* client, most of the processing are done at the server-end. The system can service simultaneous requests from multiple clients, maintaining the current query state for each of them.

The combination of PHP and relational database management system (RDBMS) to hold session information has proved to be a practical solution in the time-scale of the current project. PHP scripts are relatively simple to write, they are portable, all the standard browser functionality provided via the normal HTTP interface is retained and the resulting client is lightweight. PHP supports XML, which is used by the portal into which Okapi is integrated (see 3.2 below).

3.2 Integration with the Portal

The above described system is integrated into the TIPS portal, which is based on the Apache Software Foundation's JetSpeed groupware [6]. All portal content is stored in XML format. XML documents are dynamically rendered and converted to client-dependent formats such as HTML, PDF and WML (for mobile devices), at the server end (see a separate paper on the portal in the proceedings of the workshop).

The Retrieval Assistant and Advanced Search. One of the sub-systems of the portal is the Information Retrieval Assistant which is a rule-based expert system developed by our project partners. The retrieval assistant employs a relatively complex rule base to "reason" about the search process; it monitors user actions and gives contextual suggestions to improve the search outcome. The suggestions aim to help the user exploit the various resources available to them, and to resolve specific problematic situations (see a separate paper on the retrieval assistant in the proceedings of the workshop).

A distributed architecture is developed to establish the communication between Okapi and the retrieval assistant. The communication is one-way and consists of messages sent by Okapi to the retrieval assistant, which is called by means of three java servlets: one for the creation of a new session, one for the analysis of user's actions within the session and one for the deletion of the current session. Whenever a user interacts with the search interface, the PHP module of the Okapi system processes the requests and saves temporary data into the relational database. At the same time Okapi calls the proper servlet and passes the message regarding the user's actions (e.g.: search queries, document judgements, etc.) and the results obtained (e.g.: number of postings for each search term, list of documents retrieved, etc.) as a list of parameters to the servlet itself. Hence the retrieval assistant can monitor the user's activity and send suggestions directly to the portal to be displayed to the user. The two systems operate as independent 'black-boxes', neither need to know about the internal structure of the other. All information needed by both the systems to conduct an interactive retrieval session is stored in the relational database (cf. Fig. 2).

The same relational database serves for system evaluation and user behaviour research.

Communicating with the Portal. Contrary to communication with the retrieval assistant, communication with the portal is two-ways: Okapi receives data from the portal (e.g.: a query), processes it and sends back the results to the portal (e.g.: the list of documents retrieved). For this task the XML-RPC protocol [7] has been chosen. Okapi is seen by the portal as a set of functions remotely called through the web. The remote call from the portal is received in the form of an HTTP post by the web server where Okapi is installed. The web server then routes the request to the XML-RPC server that is part of the PHP module of the Okapi system. The XML-RPC server parses the request written in the XML-RPC syntax and calls the proper PHP function (for instance, the function okapi.SearchDb that performs a search given a query and a username). The function processes the data passed as parameters and sends back the results to the XML-RPC server, which prepares and sends back the appropriate response to the portal.

3.3 User Study

A small-scale user study was carried out to evaluate an earlier version of the system. The study involved a small number of handpicked expert users of HEP document collections. Its purpose was to improve the usability of the system based on the feedback elicited from this focus group. The subjects performed their searches on an indexed sample of about 4000 full-text arXiv documents. Five expert users in High Energy Physics were asked to participate in the experiment. All five users perform daily online searches that fall into two main categories:

- Filtering or monitoring: the user searches for new relevant documents in an established area of interest. The area is precisely defined (e.g.: *"localised particles on branes in extra-dimensional models"*). This kind of search is performed daily and often more than once per day. Users know exactly the keywords commonly used in the area and the most interesting authors, and demonstrate a high degree of expertise and confidence when making relevance judgements.
- Starting a new search to explore a new area of interest, often following leads suggested by other colleagues. The documents in most demand are review and "top cited" articles.

The five users were first given a short introduction to the system by the experimenter. They were then asked to perform a search on a topic concerning their research interests and give their comments. Their actions were logged. The experimenter took note of the comments made by the subjects and intervened whenever necessary to clarify a user action or comment.

The comments gathered from the focus group are used to improve the design (functionality as well as the user interface) of the system. Specific results drawn from the interviews and observations made by the experimenter are summarised below (more detailed discussion could be found in [8]). Users expressed their wish to:

- Search in document ID, full-text, abstract, title, author name, date and keywords fields.
- Search by phrases as well as single keywords.
- Restrict the search to some specific arXiv sub-area (category) and possibly combine them (e.g.: search in astrophysics and hep-experiment).
- See detailed information about the documents in the hit list, including, the names of the authors.
- See the number of citations the document received by a document. (The need to see in the hit list whether or not the document has been published in a referred journal was also expressed).
- See in the document record the reference list and a list of other papers that cites the document.
- Recover a past search session, with queries, results and relevant documents.
- Find similar documents to those chosen as relevant.

3.4 The User Interface

Based on the feedback elicited from the experimental study a new interface is designed which is comprised of the following main elements:

Query Frame. Fig. 3 illustrates the user interface of the Okapi search system. The system allows users to enter single words or phrases delimited by double quotes to describe their information need. The query language includes the "+" and "-" operators to include or exclude documents that contain the marked term(s) from the search results. The system assigns a high positive weight when a term is marked by "+" sign and inversely a negative weight when marked by the "-" sign.

There are currently two collections available in Okapi for retrieval purposes. First one, arXiv[3], is a collection of pre-prints in the HEP and related areas. The other, Journal of High Energy Physics (JHEP)[4], is an electronic journal in the same subject domain. The fields searchable in these two collections are: document ID, full-text, title, author, abstract, date (of submission in arXiv, publication in JHEP) and keywords. Keywords are assigned by authors, so not all documents contain them.

For field-based searching the weight of a term in a given field is calculated on the basis of its frequency in that field. Weights of all matching terms for a given

[3] arXiv is divided into four areas: Physics, Mathematics, Nonlinear Sciences, Computer Science. The physics area is divided into twelve sub-areas, of which six are of interest to the HEP community. As of April 2002, arXiv contained about 150,000 full-text papers in these six HEP related sub-areas. Number of new papers submitted to arXiv wisas about 2500 per month. arXiv is available from http://xxx.lanl.gov/ and a number of mirror sites. The Web site (excluding the mirrors) receives over 100,000 daily (in weekdays) connections. Detailed usage statistics about the arXiv collection can be found in [9] and at http://arXiv.org/todays_stats

[4] JHEP is a small database (a few hundred) of full text papers. The JHEP site and its mirrors receive roughly 650 paper download requests per week. More detail about JHEP can be found in [9].

document are combined to calculate the document score as described in section 2. When more than one field is used in the search, the sets of documents that match the search terms in a given field are retrieved for each field separately and the final hit list is created by merging the sets using the Boolean AND operator. Users can search simultaneously in arXiv and JHEP or separately in one of the collections. Users can also limit their searches to one of the sub-areas (categories) of arXiv.

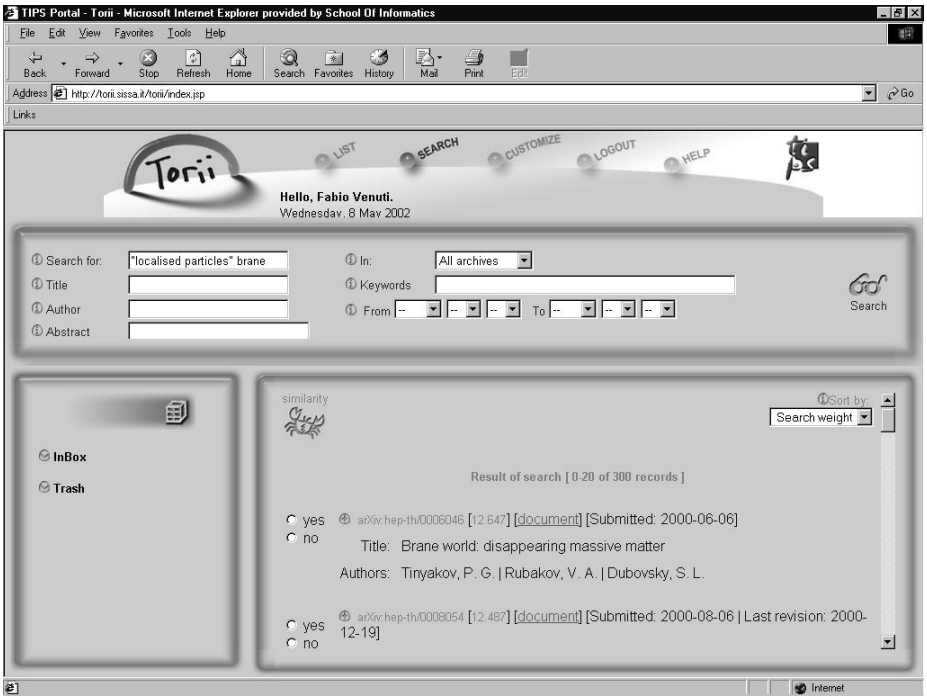

Fig. 3. The Search Window.

Hit List. Document ID, title, author name, and date fields are displayed in the hit list (Fig. 3). By default the documents in the hit list are sorted in descending order of their Okapi scores. The user could also choose to order the list by document ID, impact factor, or quality score. The latter is assigned by other users of the system (see a separate paper on the quality control tools in the proceedings of the workshop). Clicking a title in the hit list brings up the document record (see below). In the hit list next to each document entry there is a binary relevance feedback option in the form of yes/no radio buttons. Documents that are marked with yes are used to expand the user's current query in the manner described in section 2. The user could then repeat the search with the new expanded query using the similarity search option (see further below).

Document Records. The information shown in the document record window includes the bibliographic details and abstract of the document. It is possible to view the reference list given in the document. One can also view other documents that cite the document in this window. It is also possible to evaluate the quality of the document filling a form. The user can save the document to a personal folder and/or use it to modify her/his profile(s). The user profile(s) is used by the document filtering system see a separate paper on the portal in the proceedings of the workshop). There is also a link in this window to the source document in various formats.

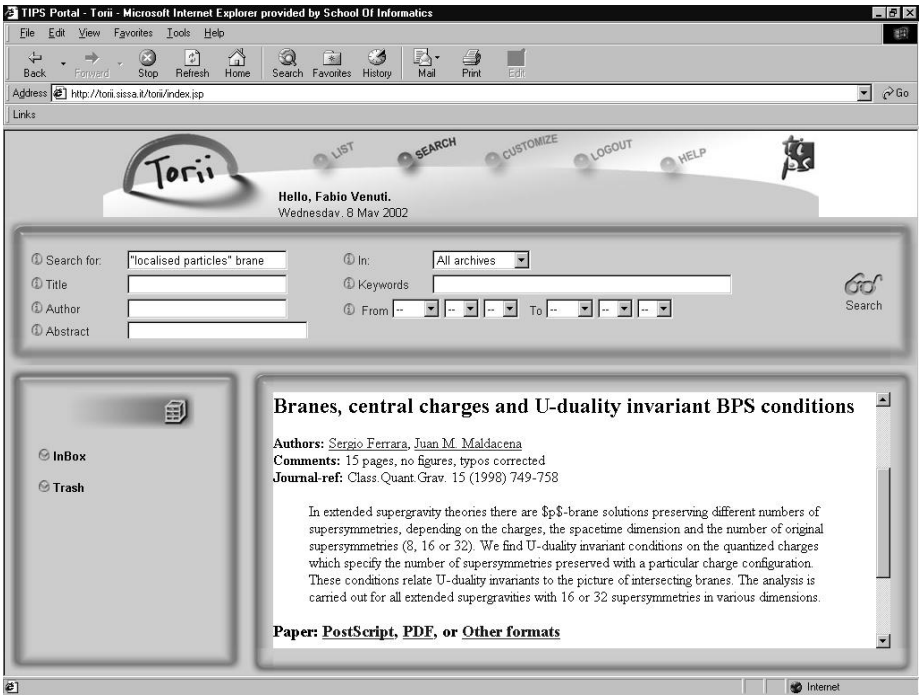

Fig. 4. A Document Record.

Similarity Search. After two positive relevance judgements[5] the user can activate the "Similarity Search" option[6]. This triggers another search using the expanded query,

[5] The more documents judged relevant, the better the term selection performance of the probabilistic formula becomes. A single positive relevance judgement is not enough for effective term selection.

[6] The "Similarity search" label is used in the user interface instead of the more accurate, but perhaps less intuitive label of "Relevance Search".

comprising terms extracted from the full text of relevant documents. Use of field-based criteria in the original query is disregarded at this stage, as are multi-word phrases and the "+" and "-" operators: the search is based solely on single word stems selected from the full text of relevant documents. The problems involved maintaining all user criteria during the query expansion process are discussed in detail elsewhere [8].

History and Judged Documents. A search "History" function has been implemented (but not yet integrated into the portal at the time of writing this paper) to backtrack to a previous state of the search process within a given search session. To keep it simple and avoid a too cumbersome interface we save only the original user query and the result of the last similarity (relevance) search – if the user made use of this option – for each of the queries within a single search session. Results of intermediate searches are not saved. The user queries are represented in the History view by the terms submitted by the searchers. A user can submit any number of queries by filling in the search form and pressing the submit button in a given search session. Each search request made in this way is considered as an independent query. A search session starts when the user submits a query for the first time and terminates when s/he logs out from the portal (or timed out). It will also be possible to save the complete search session (or any part of it) before logging out from the portal.

4 Conclusion: The Future

Some of the problems involved in designing a sophisticated Web-based retrieval system and solutions implemented are discussed. Our experience from our involvement in the TIPS project showed that creating a user-friendly interface while maintaining the power of the underlying system remains to be one of the main challenges of information retrieval research. The HEP community with geographically distributed user population who depend on document retrieval to support their day to day activities proved to be a highly fertile ground to study the complexity of the information retrieval problem.

As well as providing challenges, such an environment provides new opportunities to tackle the information retrieval problem from new directions. We are barely starting with the TIPS project to appreciate the collaborative and social nature of the retrieval process. A substantial amount of theoretical and empirical research is needed in this direction to understand the social contexts of the retrieval process and how to harness the potential offered by information networks to enable people to discover, create and share information.

References

1. TIPS project home page. Available at http://tips.sissa.it
2. Special issue of Journal of Documentation 53 (1), 1997.
3. Okapi projects home page. Available at

http://web.soi.city.ac.uk/research/cisr/okapi/okapi.html
4. Robertson, S.E. and Sparck Jones, K. Relevance weighting of search terms. *Journal of the American Society for Information Science 27*, 1976, 129-146.
5. K. Sparck Jones, S., Walker and S.E. Robertson, A probabilistic model of information retrieval: development and status. University of Cambridge Computer Laboratory Technical Report no. 446, 1998. Available at http://www.ftp.cl.cam.ac.uk/ftp/papers/reports/#TR446
6. JetSpeed project home page is available at http://jakarta.apache.org/jetspeed/site/index.html
7. XML-RPC home page. Available at http://www.xmlrpc.com/
8. Karamuftuoglu, M., Jones, S., Robertson, S., Venuti, F., Wang, X. Challenges posed by web-based retrieval of scientific papers: okapi participation in TIPS. *Journal of Information Science 28*, 2002, 3-17.
9. TIPS Information Resources. Available at http://tips.sissa.it/docs/UR-R1/UR-R1-IR.ps