# BINGO! and DAFFODIL: Personalized Exploration of Digital Libraries and Web Sources*

Martin Theobald[1] and Claus-Peter Klas[2]

[1] Max-Planck Institut für Informatik, Saarbrücken
martin.theobald@mpi-sb.mpg.de
[2] University of Duisburg
klas@uni-duisburg.de

**Abstract.** Daffodil is a digital library system targeted at strategic support of advanced users during the information search process. It provides user-customizable "stratagems" for exploring and managing digital library objects with meta data annotations over a federation of heterogeneous digital libraries. Bingo! is a focused crawler that learns how to gather thematically relevant documents from the Web and Deep-Web sources. This paper presents a coupling architecture for Daffodil and Bingo! that allows advanced users to explore digital libraries and Web sources in a comprehensive and coherent way. Starting from a user's interest profile in Daffodil, Bingo! is instructed to find thematically similar documents on the Web, leading to high-quality recommendations that reach beyond the information that can be directly found in digital libraries. Our experimental studies demonstrate that this coupling does indeed lead to a powerful tool suite that improves the precision and recall compared to conventional library and Web search mechanisms.

## 1 Introduction

Daffodil is a digital library system targeted at *strategic support* during the information search process [12]. For advanced users, high-level search functions, so-called "stratagems", provide powerful information-exploration functionality beyond today's digital libraries. In particular, Daffodil can handle federations of heterogeneous digital libraries (DLs) through appropriate wrappers (e.g., for DBLP, ACM digital library, etc.). For example, a user can associate her interest profile with a set of digital library objects (DLOs for short), typically journal or conference publications, which are organized into a folder hierarchy. These DLOs are automatically enriched by meta data annotations such as authors, titles, publication year, etc., which Daffodil extracts from the underlying DLs. However, Daffodil cannot directly explore arbitrary Web sources, for which no structured service interface is known a priori. For example, it cannot find publications that are stored on authors' homepages.

* Funded by the German Science Foundation (DFG) as part of the research project "Classification and Intelligent Search on Information in XML (CLASSIX)"

The work presented in this paper aims at enhancing DAFFODIL to reach out for arbitrary Web data including Deep-Web sources where the data resides behind a structured portal interface. At the same time, the high search precision and comfortable feedback mechanisms that DAFFODIL provides on DLs should be maintained also for Web sources. To this end, we have coupled DAFFODIL with the BINGO! focused crawler, an advanced toolkit for information portal generation and expert Web search. In contrast to standard search engines which are solely based on precomputed index structures, a focused crawler interleaves crawling, automatic classification, link analysis and assessment, and text filtering. A crawl is started from a user-provided set of training data and aims to collect comprehensive results for the given topics. A particularity of BINGO! is that it uses semi-supervised learning techniques and ontological background information to overcome the insufficiencies of initial training data for improved precision. Moreover, BINGO! incorporates topic-specific information portals (e.g., external search engines or portals of large research institutes) that cannot be directly crawled.

The coupling architecture for combining the benefits of DAFFODIL and BINGO! is depicted in Figure 1. The two subsystems communicate using multi-agent and Web Service protocols. A typical scenario starts with a user putting together one or more folders of annotated DLOs, for example, the ones shown in the DAFFODIL box for topics "database core technology (DB core)", "Web information retrieval (Web IR)", and "workflow management". These DLOs point to documents within digital libraries, and in some cases may include links to the document content itself. For further exploration of these topics and acquiring new recommendations, these DLO folders can be sent to BINGO!, where they are converted into appropriate training data for the classifier and seeds for the focused crawler. The DLO meta data helps to generate queries to Deep-Web portals which cannot be directly crawled; this step makes use of an ontology to map, for example, a meta data attribute "subject" to "category", "genre", "topic", etc. Finally, both the crawled and positively classified Web documents and the portal-extracted results are returned to DAFFODIL as recommendations and are presented through an assessment and feedback GUI to the human user. These steps and their order are depicted in Figure 1 by the thick numbered arrows (the numbers correspond to the workflow described in section 6). Note that the figure shows all steps of one "handshake" between DAFFODIL and BINGO!; the entire loop can be iterated multiple times, either fully automatically or after obtaining relevance feedback by the human user.

The contributions of this paper are twofold:

- We show how to couple different technologies for information search and organization, one geared for digital libraries and one for Web sources. The two systems that underlie our work, DAFFODIL and BINGO!, are paradigmatic; our considerations can be carried over to other toolkits of these flavors.
- The combination of DAFFODIL and BINGO! provides advanced users with significant added value. From the DL viewpoint, the recall is substantially improved by considering Web sources such as homepages of individual re-

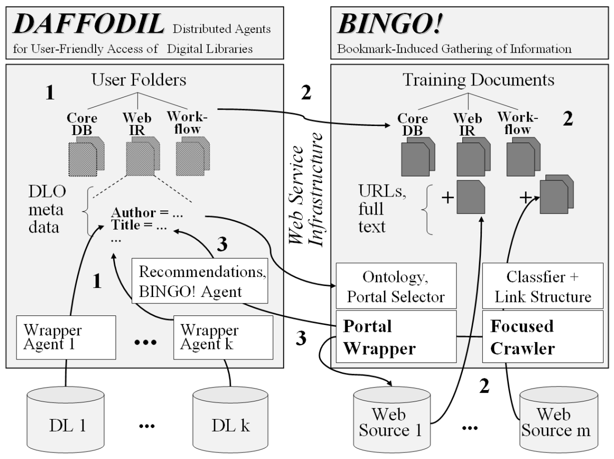

DAFFODIL Distributed Agents for User-Friendly Access of Digital Libraries
BINGO! Bookmark-Induced Gathering of Information
User Folders
Core DB
Web IR
Work-flow
DLO meta data
Author = ...
Title = ...
Recommendations, BINGO! Agent
Wrapper Agent 1
Wrapper Agent k
DL 1
DL k
Web Service Infrastructure
Training Documents
URLs, full text
Ontology, Portal Selector
Classfier + Link Structure
Portal Wrapper
Focused Crawler
Web Source 1
Web Source m
1
2
3

**Figure 1.** Coupling Architecture for DAFFODIL and BINGO!

searchers or entire institutes. The focusing mechanism of BINGO! largely avoids the danger of information overloading, compared to conventional Web search mechanisms, and helps keeping the precision sufficiently high. From the Web search viewpoint, the meta data annotations provided by DAFFODIL are crucial for generating meaningful queries against Deep-Web portals with multi-parameter interfaces (e.g., the advanced search form of the DBLP Trier). Also, the meta data is vital for high precision of recommendations, and DAFFODIL can additionally exploit relevance feedback for further improvements.

## 2 Related Work

Most related work on recommendation is based on the usage of already available information items [31] from a precomputed index. But most lack the problem of having only sparse data from the user of the digital library, since they usually rely on Web interfaces only. Other account on human collaboration to generate recommendations [15, 25, 26]. Inspired by the organization of the human brain, the Hebbian approach [16] introduces a set of algorithms that try to model the finely-graded, continuous associations between documents that trace the users' constantly changing focus of interest while browsing a document collection. These associations are measured through co-activation values to calculate document similarities, which in turn can be used to cluster similar documents. DAFFODIL, in combination with BINGO!, is the first approach that combines semi-structured meta data requests with focused crawling techniques to build a personalized recommendation index over a unified view on Deep-Web *and* traditional Web contents.

## 3 Daffodil

Daffodil[1] is a federated DL system that offers a rich set of functions across a heterogeneous set of DLs (see [12] for a description of the architecture). The current prototype integrates over 10 DLs (e.g. Achilles, Citeseer, DBLP, etc.) in the area of computer science, together with other information sources (e.g. Google, Scirus, Ispell, etc.). Since different DLs may contain various pieces of information about the same publications, the federation yields important synergies by combining these information in high quality detailed data for the user.

For structuring the functionality, we employ the concept of high-level search activities for strategic support as proposed by Bates [3, 4]. Based on empirical studies of the information seeking behavior of experienced library users, Bates distinguishes four levels of search activities, called *moves*, *tactics*, *stratagems* and *strategies*. The higher levels base on the lower levels, so *tactics* usually relay on one or more *moves*, and so on.

Based on these levels, *strategic support* during the information search process is the fundamental concept implemented within Daffodil. High-level search functions, based on the stratagem level, implement this strategic support for the user and provide functionality beyond today's digital libraries. To our knowledge, Daffodil is the first implementation of Bates' ideas.

To open the complex level of *strategies* for the user, we divided the task into phases (workflow), called the *Digital Library Life Cycle* [24] outlined in Figure 2. It is based on five phases, from *discovering* information resources, *retrieving* the information, *collating* the found information in a structured personal library, *interpreting* the found information through cognitive and collaborative processes, to *re-presenting* the new conceived information into new information. The workflow usually starts from the discovering through the re-presenting phase, but, of course, switching between the phases will occur often. Every phase consists of different *stratagems*, which can be combined to reach the information goal.

### 3.1 Daffodil: Bases for Recommendation

Daffodil's high-level search activities, as outlined above, have been designed in close accordance with the *WOB* model [20] as a range of tools that are integrated into a common workspace environment. The goal of Daffodil's graphical user interface is to provide an environment for retrieval, searching, and browsing tasks, as well as collation, organization and re-usage of the retrieved information in a user-friendly way. On the desktop (see Figure 3), the set of available stratagems is represented as a set of tools. This design offers a wide range of synergies, starting from the information sources up to the visualization, whereby an optimal, strategy-supported information search process is presented to the user. Furthermore, the synergies are extended through a tight integration of these tools, e.g., by using Drag&Drop mechanisms or links to external information sources. In [14] we discuss the user interface of Daffodil.

[1] Distributed Agents for User-Friendly Access of Digital Libraries

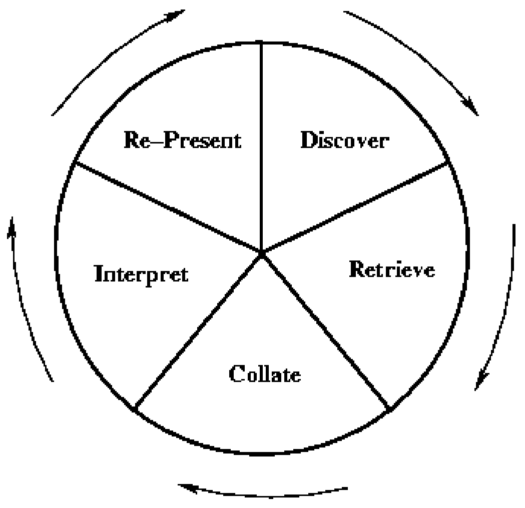
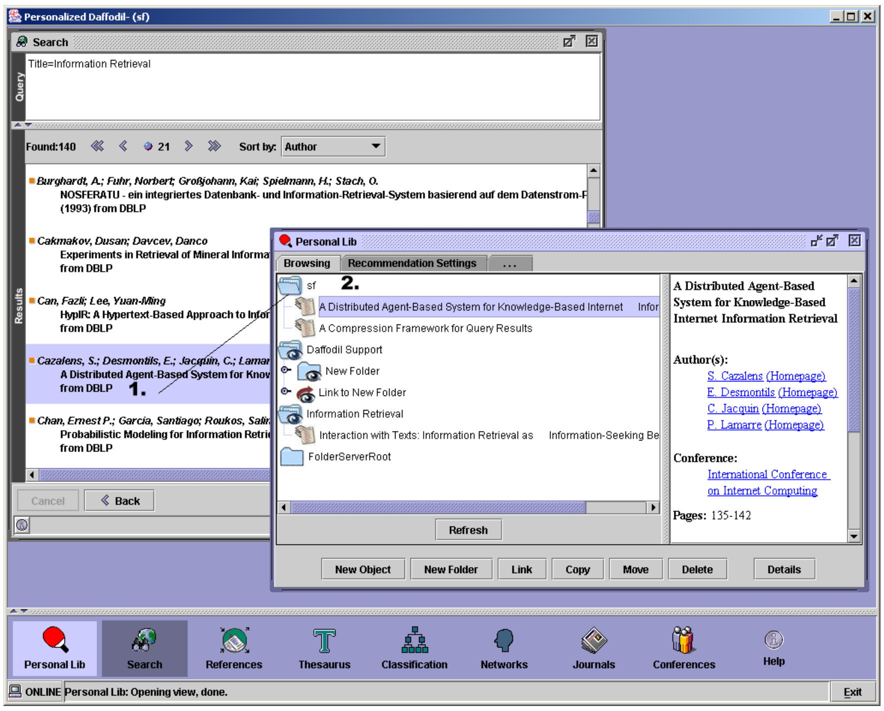


**Figure 2.** The Digital Library Life Cycle

So far DAFFODIL includes more than 10 different integrated tools. For *personalization*, we have integrated a *personal library tool* on the desktop, which supports individuals as well as groups. This tool allows for storing DL objects — documents, authors, journals, conferences as well as query formulations — in arbitrary folders (via the standard Drag&Drop interaction). This is the major connection to BINGO!. Usually starting from a list of bookmarks, BINGO! is now backed by structured meta data information, since DAFFODIL is able to extract annotations like title, authors, abstracts, or classification keywords. The information is classified by the user in folders, describing precisely the information need. In Figure 3 the lower right window shows the current visualization of the library. On the left side an arbitrary folder structure can be created by the user. Any DLO can be saved within these folders. The detail of a DLO is shown in the right half of the tool. In the case of web pages, an external browser is used. To enable BINGO!'s Web recommendation service the user only has to choose a context menu entry on that folder.

Other recommendation services are already integrated into DAFFODIL. In [13] we describe some concepts for recommending all kinds of digital library objects through collaborative services, based on a large number of users and user groups; for this purpose, we exploit the stored information in the digital library, instead of relying on users' rating (empirical evidence from similar applications suggests that only little rating activity should be expected). Based on these ideas, we developed a matchmaking algorithm allowing a new DAFFODIL user to find related users or groups.



Title=Information Retrieval

Found:140 21 Sort by: Author

Burghardt, A.; Fuhr, Norbert; Großjohann, Kai; Spielmann, H.; Stach, O.
NOSFERATU - ein integriertes Datenbank- und Information-Retrieval-System basierend auf dem Datenstrom-F
(1993) from DBLP

Cakmakov, Dusan; Davcev, Danco
Experiments in Retrieval of Mineral Informa
from DBLP

Can, Fazli; Lee, Yuan-Ming
HypIR: A Hypertext-Based Approach to Infor
from DBLP

Cazalens, S.; Desmontils, E.; Jacquin, C.; Lamar
A Distributed Agent-Based System for Know
from DBLP

Chan, Ernest P.; Garcia, Santiago; Roukos, Salin
Probabilistic Modeling for Information Retrie
from DBLP

sf
A Distributed Agent-Based System for Knowledge-Based Internet Infor
A Compression Framework for Query Results
Daffodil Support
New Folder
Link to New Folder
Information Retrieval
Interaction with Texts: Information Retrieval as Information-Seeking Be
FolderServerRoot

A Distributed Agent-Based System for Knowledge-Based Internet Information Retrieval

Author(s):
S. Cazalens (Homepage)
E. Desmontils (Homepage)
C. Jacquin (Homepage)
P. Lamarre (Homepage)

Conference:
International Conference on Internet Computing

Pages: 135-142

ONLINE Personal Lib: Opening view, done.

**Figure 3.** Collating documents into the personal library

## 4 Bingo!

### 4.1 Focused Crawling for Expert Web Search

In contrast to a search engine's generic crawler (which serves to build and maintain the engine's index), a focused crawler is interested only in a specific, typically small, set of topics [7]. The topics of interest may be organized into a user- or community-specific hierarchy. The crawl is started from a given set of seed documents, typically taken from an intellectually built topic directory (e.g., *bookmarks*), and aims to proceed along the most promising paths that stay "on topic" while also accepting some detours along digressing subjects with a certain "tunneling" probability. Each of the visited documents is automatically classified into the crawler's hierarchy of topics to test whether it is of interest at all and where it belongs in the taxonomy [5, 9]. The outcome of the focused crawl can be viewed as the index of a personalized information service or a thematically specialized search engine.

The Bingo![2] [28, 29] engine is designed to address two major problems in information organization and search:

[2] Bookmark-Induced Gathering of !nformation

1. Starting with a reasonable set of seed documents that also serve as training data for the classifier, a focused crawl can populate a topic directory and thus serves as a largely automated *information portal generator*.
2. Starting with a set of keywords or an initial result set from a search engine (e.g., from a Google query), a focused crawl can improve the recall for an advanced *expert query*, a query that would take a human expert to identify matches and for which current Web search engines would typically return either no or only irrelevant documents (at least in the top ranks).

## 4.2 System Architecture

The BINGO! focused crawling toolkit consists of the following six main components that are depicted in Figure 4.

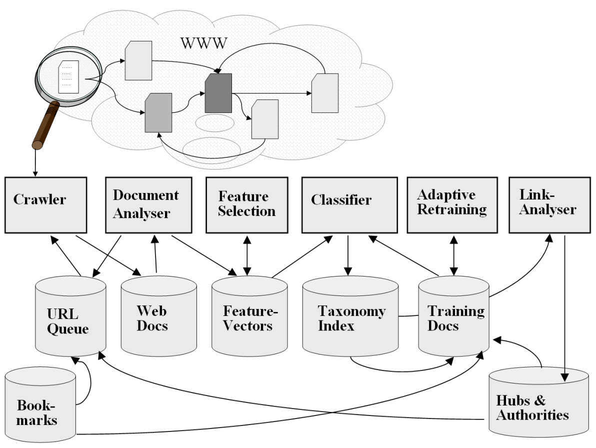


**Figure 4.** The BINGO! architecture and its main components

**Crawler.** The crawler processes the links in the URL queue using multiple threads. For each retrieved document, the crawler initiates content-specific handlers that depend on the document's MIME type (e.g., text/html, application/pdf, etc.) and then invokes the classifier on the resulting feature vector. Once a crawled HTML document has been successfully classified, BINGO! extracts all *href* links and adds them to the URL queue in a canonical form. The priority of new links in the topic-specific crawler queues is set according to the classification confidence of the parent document. All extracted terms, links, and the available crawler meta data such as URL, title, classification results, etc., is stored in our local database index for later retrieval.

**Document Analyzer.** BINGO! computes feature vectors for documents according to the standard bag-of-words model using stop word elimination, stemming, and $tf * idf$-based term weighting [2, 22]. We consider our local document database as an approximation of the corpus for *idf* computation and recompute it lazily as the corpus grows.

**Feature Selection.** The feature selection aims to detect the most characteristic features among the training documents for each given topic in comparison to the opposing topics at each level of the topic tree. These are the features that are selected in the input vectors of both the trainer and classifier, while the remaining features are considered as noise, either because the are frequent across the competing topics (in the manner of a stop word), or because the are very infrequent for each of the topics (and therefore not relevant, e.g., typos).

We use the *Mutual Information* (MI) [22] measure to build topic-specific feature spaces. This technique, which is a specialized case of the notions of cross entropy or Kullback-Leibler divergence [2, 22], is known as one of the most effective methods [35, 36] that is slightly in favor of rare terms (i.e., the ones with a high *idf* value), which is an excellent property for classification.

**Classifier.** We chose Support Vector Machines (SVM) [6, 18, 33] to model the classification task in the vector space. The SVM algorithm is a learning algorithm that consists of a supervised training phase for a *binary set of topics*. In the optimization phase a separating hyperplane is computed such that it maximizes the margin between a set of positive and negative feature vectors in the $m$-dimensional feature space. The classification step then simply computes the algebraic sign of the scalar product between this hyperplane and a test document's vector, where the sign denotes on which side of the hyperplane the test vector is determined to be. The absolute value of the result yields a measure for the *classification confidence* in form of the perpendicular distance of the test vector to the separating hyperplane. The *hierarchical, multi-class classification* problem for a tree of topics is solved by training a number of binary SVM's, one for each topic in the tree.

Focused crawling over the Web inherently involves a precision-recall trade-off. Two principle focusing strategies are provided with regard on the user's demand for best possible precision or recall: "strong" and "soft" focus. The "strong" focus pursues only links from documents that are successfully classified into *the same topic* as one of their predecessors. This strategy is useful to keep the crawler closely "on track" and to improve the crawler's precision during a deep Web search while skipping potentially valuable hub pages which do not belong to any topic-of-interest themselves, but might contain links to a relevant page (e.g., a homepage overview at the DBLP Trier, that is not itself devoted to the topic "database core technology", but might contain links to authors' homepages working on that specific topic would be skipped). In contrast, the "soft" focus mode pursues links from *all* documents without regard to the classification results and provides higher recall. In the first case, BINGO! additionally traverses links from off-the-topic documents up to a small threshold depth (typically set to depth 1 or 2 from the last topic-specific document) to tunnel through topic-unspecific "welcome" or "table-of-content" pages before again reaching a thematically relevant document.

**Topic Distillation.** The BINGO! engine applies Kleinberg's link analysis method, coined HITS [19], iteratively to each topic in the taxonomy tree. Analogously to Google's Page Rank, the HITS algorithm uses the documents adjacency matrix $A$ of the crawled Web graph to exploit principle Eigenvectors. While Page Rank uses $A$ directly, HITS divides the task into the approximation of the principle Eigenvectors of the matrices $A^TA$, and $AA^T$ respectively, and thus produces a ranking for the best *hubs* (link collections that point to multiple good authorities) and the best *authorities* (high quality resources which are referenced by many good hubs).

**Adaptive Retraining.** Building a reasonably precise classifier from a small, user-specific training set is a challenging task. To address this problem, BINGO! introduces a two phase, semi-supervised training approach. The *learning phase* serves to automatically identify the most characteristic documents of a topic, coined *archetypes*, among the bookmarks' neighbor documents from a initial shoal crawl in strong focusing mode. Starting by the user's training set, these archetypes are determined among the neighbor documents with the highest classification confidence and highest authority score to be promoted for retraining an extended classifier. The *harvesting phase* then switches to soft focus and serves to effectively process the user's information demands with improved crawling recall. In the case where user-feedback is available, this automatic approach can of course be improved by human input, e.g., when a DAFFODIL user marks certain results as bad and implies that these documents are used as explicit negative training examples in the next run.

## 5 Integration of BINGO! and DAFFODIL

Recall from Section 1 that our goal is to feed user-specific DLO folders, maintained in DAFFODIL, to the BINGO! focused crawler in order to detect additional relevant publications and information sources on the Web and in Deep-Web portals. For example, the homepage of a scientist whose publications match the user interests that are manifested in some DLO folder or a particularly relevant book that is advertised in a bookshop portal or on a university course Web page would be an excellent recommendation that our combined system should point out to the user.

In some cases, the DLO's provide links to full-text sources (e.g, a PDF file) or an author's homepage; in many cases, however, the full-text sources are not available to DAFFODIL. Without Web links, however, neither a traditional HTML based crawler can be started, nor can we train a document classifier without extracting features from full-text.

The meta data itself may yield a short description for a DLO and thus specify a user's topic of interest, but typically, these descriptions are too sparse to yield sufficiently dense feature spaces for classifying arbitrary Web contents. To overcome this digital library-centric viewpoint and to be able to explore more background information from the Web, the BINGO! crawling toolkit is extended by a deployment interface for its own portal wrappers, which are chosen from a combination of digital libraries and Web portals such as DBLP Trier, Achilles, etc., and search engines like Google, Yahoo!, and Exite. By using these external databases, BINGO! is able to resolve *virtual links* between meta data and Web contents and can thus continue a search over various Deep-Web sources in a traditional crawling manner.

### 5.1 Web Service Infrastructure

For service federations such as the coupling of DAFFODIL and BINGO!, some form of middleware is needed, and these days the method of choice is Web Service technology [1, 21]. This technology includes XML as the data exchange format and SOAP [27] as communication protocol (on top of HTTP) for remote procedure calls and messaging. Service interfaces are described in the Web Service Description Language (WSDL) [34] and registered in a Universal Description, Discovery and Integration (UDDI) [32] registry. The Web Service infrastructure for the integration of DAFFODIL and BINGO! can be broken down into the following three parts:

A. The communication between DAFFODIL and BINGO! to invoke user requests and return Web recommendations.
B. The communication between BINGO! and its portal-specific wrappers.
C. The communication with additional services, e.g., the ontology server.

This Web Service infrastructure allows the seamless integration of the two self-sustaining DAFFODIL and BINGO! servers, where arbitrary clients can connect to the DAFFODIL server by executing a Java Applet and authenticating at the DAFFODIL server. The DAFFODIL server, in turn, forwards recommendation requests to BINGO! using a simple XML schema for the DLOs along with a user id and a timestamp. These requests are queued by the BINGO! server until an idle crawler instance is found and a new recommendation workflow for the respective user folder(s) can be invoked. As the crawling process is finished, newly-found recommendations are asynchronously sent back to the DAFFODIL server, where the user can pick up her personal recommendations at the next log-in.

### 5.2 Portal Queries for the Deep-Web

A great part of the information on the Web is stored in non-HTML data sources, mostly relational databases, which are connected to some server application logic that dynamically generates Web pages. These information repositories are hosted by so-called *Web portals.* Most existing portals do not support Web Services, but they provide HTML form based access to an internal search engine. In BINGO! we have integrated such servers by generating Web Service wrappers, which are hosted by a dedicated server application. This application, called the *Service Mediator*, is responsible for the generation and invocation of wrapped services. When the focused crawler (or some other client of the mediator) obtains a reference to a Web Service hosted by the Service Mediator, it retrieves the corresponding WSDL description from the local UDDI and generates an appropriate SOAP Message to invoke the service. The translation layer of the Service Mediator interprets the SOAP message, activates the corresponding wrapper classes, and invokes the necessary methods to communicate via HTTP with the portal site to invoke the internal search engine of the portal.

**Portal Wrapping and Query Processing.** New portals can be deployed automatically in our system. When a new URL that points to a form page is supplied, our *wrapper generator tool* [17] is invoked which applies heuristic rules for generating a WSDL entry on the fly. In this generating mode, the system tries to parse the target Web site and extract form fields. Typically, labels and internal names of the form fields are used as parameter names for the WSDL description. For example, highlighted text next to form fields is promoted as the attribute name of that form field, and the given type that this form field requires determines the corresponding attribute type (e.g., an enumeration type for fields with a pull-down menu).

The WSDL entry for the new Web Service is then added to our local UDDI registry. A small Java class is automatically generated from the HTML form page and compiled under our Web server's servlet engine such that it transforms the SOAP call containing the values for the respective WSDL parameters into an HTTP posting to the form submission URL which invokes the portal's search engine. The class is deployed as the target of this Web Service and returns the portal's response URL with regard to the submitted form values for each call. These result pages can contain links to static Web pages or dynamically generated Deep-Web sources that can now be reached by a crawler.

**Query Generation for Heterogeneous Portals.** The Web Service mediator approach helps with unified shipping of queries to portals, but does not provide the queries themselves. Rather BINGO! needs to generate meaningful queries from the DLO meta data that it receives from DAFFODIL To this end, we have integrated "Semantic Web" technologies [11] in the form of ontological knowledge on the users' topics of interest which is exploited for two purposes: 1) the selection of appropriate portals upon receiving a DLO entry, to be discussed in the next subsection, and 2) the generation of meaningful query parameters to be filled into a portal's Web Service (or its form fields, respectively). The main difficulty is to cope with heterogeneous portals (on the same topic(s) but with different terminologies and interfaces).

Architecturally, we treat ontology sources like any other data source that we interact with and integrate it by means of a Web Service interface. Our ontology server [17, 30] provides unified access to ontology data sources like WordNet [10] and OpenCyc [23]. In addition to the ontological taxonomy (e.g., hypernyms/hyponyms or holonyms/meronyms relations) provided by WordNet, these relationships between different ontological concepts carry weights to quantify concept similarities.

When BINGO! receives a DLO, it needs to match the attribute names in the DLO with the parameter names provided by a Web-Service-wrapped portal. Obviously, the diversity of attribute names often makes an approximate match based on ontological similarity the only viable choice by mapping each DLO attribute to its (most similar) correspondent form field. This "approximate join" between a DLO and a portal's query parameters is illustrated in the following example: Suppose the join attribute is an author's name, for example with the value "John Doe", divided into the two attributes *first name* and *last name* with values "John"' and "Doe" in DAFFODIL's meta data. We want to connect to a form page providing the descriptions *author*, *title*, and *year*, only. The ontology service is now able to tell us that the semantic concept capturing the meaning of *first name* is in fact more similar to the concept capturing *author* than to the concepts representing *title* or *year*. Analogously, we are able to match *last name* to the form field for *author*, too, and correctly fill out the *author* field of the form with the value "John Doe".

**Automated Portal Selection.** DLO's may substantially vary in coverage, too; e.g., some objects provide abstracts, keywords, or a publisher information. The similarity scores for possible matches between a DLO and the candidate portals are aggregated into an overall similarity score between DAFFODIL's DLO's and the WSDL description of each portal, providing us with a *ranking* for query-specific portal selection. This way, we are able to choose only the top $k$ matching portals for each request, i.e., the portals with the highest ontological similarity between the attribute names of its form fields and a given DLO and skip weak matches to save expensive portal requests.

## 5.3 Result Page Segmentation

Most Web portals are designed for a human user, many of them with commercial background. Result pages of portal requests often contain only a minor part of query-relevant document links, while the major part of the page consists of navigational links or advertisements.

To avoid crawling these off-the-topic links, we first segment the result page, i.e., we are looking for the most relevant subtree in the DOM [8] structure of the HTML document. In the body element of each document $d$, we choose the *densest subtree* that maximizes the ratio of the weighted sum of the number of query matches and the amount of links to the overall amount of terms in the subtree using the same feature vectors as the classifier. For each node $s_i$ in $d$ with feature vector $\boldsymbol{f_i}$ and a given query vector $\boldsymbol{q}$, we maximize the $density(\boldsymbol{s_i}, \boldsymbol{q}) := \frac{\alpha\ (\boldsymbol{f_i} \times \boldsymbol{q}) + \beta\ links(s_i)}{|\boldsymbol{f_i}|}$, where the weights $\alpha$ and $\beta$ can be adjusted to fit each registered portal's layout. Typically, the interesting link collection in the result page is rooted at a table (*<table>*) or list (*<DL>, <OL>, or <UL>*) element of the HTML-DOM. If there is no further tree structure contained in the body element, or if there are no term matches to the query, the root node of the result page is returned.

### 5.4 Recommendation Ranking

The ranking of returned Web recommendations has to reflect a user's demand for high quality content and provide a precise description of her topic-of-interest. The classifier and the topic distillation components of BINGO! provide the criteria to measure the quality of a document. We use the product of the non-normalized, positive classification confidence $svmscore_{d_i}$ and link-analysis-based authority $authscore_{d_i}$ as a measure for ranking the documents that are returned to DAFFODIL as recommendations, i.e., $score_{d_i} = (1 + svmscore_{d_i})(1 + authscore_{d_i})$. To reflect the user's preference for a non-HTML resource (e.g., PDF), the score for these document's can be boosted by an additional factor.

## 6 The Web Recommendation Workflow

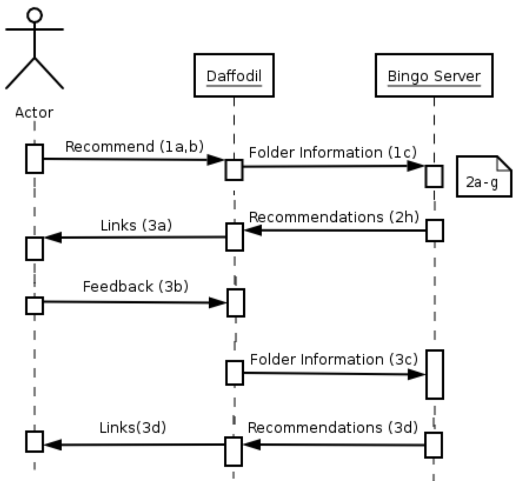


**Figure 5.** Communication: BINGO! DAFFODIL

In this section we describe the complete workflow that the combined DAFFODIL-BINGO! system (recall from Figure 1) executes to provide the digital library user with high-quality Web recommendations. The workflow itself is illustrated in more detail in Figure 5.

1. (a) The DAFFODIL user (actor) searches publications by sending multiple requests to DAFFODIL's wrapper agents and organizes the DLOs in a personal folder structure that represents her topics-of-interest.

   (b) One or more folders are marked for a recommendation request. The information about the request is stored into the Daffodil backend database, along with a timestamp.
   (c) The complete folder information including all DLOs is sent to the Bingo! server, where the actual recommendation process takes place.
2. At this time, an initial Web Service call to the Bingo! server is executed. Multiple user requests are simply queued in the FIFO (First-In-First-Out) principle by Bingo! until an idle crawler instance is available.
   (a) To identify crawler seeds and the classifier's training documents, Bingo! sends multiple Web Service requests to its registered portal wrappers selecting only the best matching portals for each DLO, i.e., the ones with the highest ontological similarity between a portal's form fields and the DLO's meta data attributes.
   (b) Each meta data attribute (e.g., author, title, keywords) is mapped to a portal's form field using a precomputed configuration table. To improve efficiency, ontology lookups and the similarity computation take place only once during an initial portal administration phase.
   This static mapping of DLO attributes to form fields saves expensive and redundant requests to the ontology server (but in situations where the DLO attribute names may themselves change frequently, we could as well compute the mapping dynamically, at some higher cost).
   (c) The result page of each portal request is segmented to identify a query-relevant subtree, which is selected to yield the links for further processing. These Deep-Web links including the links already provided by Daffodil (e.g., pointers to an author's homepage) are added to the crawler's queue.
   If a full document URL is part of a DLO provided by Daffodil, this document is fetched and put into the classifier's training base. Otherwise, the portal result pages are filtered to find the actual full-text source in an appropriate format (e.g., PDF). If the portal results do not yield the full-text link, a *virtual document* extracted from the DLO's attribute values (e.g., author(s), title, abstract, and keywords) is generated to extend the training base.
   (d) The feature selection and the SVM trainer are invoked using all successfully resolved full-text sources and the virtual documents derived from the DLO's.
   (e) If negative user feedback for a DLO has been provided in a previous iteration, this item is used as an explicit counterexample for training and helps to eliminate similar documents in the current run.
   (f) The focused crawler is started on all meta data links and the links from the segmented portal results with a maximum depth of 2 or 3 for fast results. A depth of more than 3 can be requested, too, by using the tunneling strategy (see section 4.2) in the "strong focus" mode.
   (g) The crawler stores all documents in a database maintained by Bingo!. When the crawl terminates, Bingo! analyzes the crawl result's link structure using the HITS algorithm.

(h) The top ranked recommendations (see section 5.4) are sent back to the Daffodil server via a second Web Service call.

3. (a) The links are then provided in a recommendation tool on the desktop for the user.
   (b) The user views the recommended links and stores an interesting subset into her personal library, typically into the same folder where the recommendation process started from. This information is interpreted as implicit *positive* feedback. By default, if no user action is taken for a recommended item, this is interpreted as *negative* feedback.
   (c) After an arbitrary timeout, Daffodil can proactivly trigger new recommendations from Bingo!, providing this time not only the folder information, but also the feedback data.
   (d) Bingo! can then create a more accurate recommendation for the user's profile which is presented again. Of course, only formerly unseen links are presented in the new rounds. Thus, iterative feedback can gradually refine the system's representation of the user's interest.

## 7 Experiments

Our preliminary experimental setup reflects a typical single-folder recommendation request concerning the topic *workflow.* We searched for recent research publications on workflow management using Daffodil's user interface and manually chose the 12 papers depicted in Table 1 (with a focus on Michael Gillmann's and Jeannine Weissenfels' work on the Mentor-Lite project [W3,W4], workflow performance [W1,W2,W9], and some mixed workflow research papers [W5,W6,W7,W8,W10, W11,W12]). The documents were marked for a Bingo! recommendation and submitted to the Bingo! server via SOAP.

Bingo! was configured to use wrappers for the following portals: HCIBIB, Google advanced search, DBLP Trier, Springer advanced search, Achilles, and NCSTRL. Crawling parameters like maximum crawling depth, maximum amount of visited documents, and the number of recommendations could be customized by the Daffodil user. We chose a crawling depth of 4 and limited the crawl to 5000 documents where the top 20 were to be returned as recommendations.

Portal requests where initiated to the top 3 matching portals for each DLO, and this resulted in resolving two full-text links to PDF resources (both found by Google) in addition to the 5 full-text links that were specified in the meta data itself. The additional *href* links from these portal results where added to the crawler queue. Without any full-text documents (i.e., the meta data values for the DLO's alone), the initial feature space for the workflow folder would have contained only 201 distinct terms, which would have led to a much weaker classifier compared to the 2324 distinct terms gathered from DLO's and the full-text documents (after stemming and stop word removal). These feature spaces were reduced by selecting the 400 top ranked features based on the MI measure. We built the SVM classifier using 14 counterexamples from various Yahoo! topics as a coarse filter and started the crawler. Since it is difficult to measure exact

| No. | Author | Title | Year |
|---|---|---|---|
| W1. | M. Gillmann, J. Weissenfels, A. Kraiss, G. Weikum | Performance Assessment and Configuration of Enterprise-Wide Workflow Management Systems | 1999 |
| W2. | M. Gillmann, G. Weikum | Benchmarking and Configuration of Workflow Management Systems | 2000 |
| W3. | M. Gillmann, J. Weissenfels, A. Kraiss, G. Weikum | Performance and Availability Assessment for the Configuration of Distributed Workflow Management Systems | 2000 |
| W4. | J. Weissenfels, M. Gillmann G. Shegalov, W. Wonner | The Mentor-Lite Prototype: A Light-Weight Workflow Management System | 2000 |
| W5. | A. Alessandra, G. Michaelis | A Light Workflow Management System Using Simple Process Models | 2000 |
| W6. | M. Gillmann, G.Shegalov, G. Weikum, | XML-enabled workflow management for e-services across heterogeneous platforms | 2001 |
| W7. | H. Zhuge, T. Cheung | A timed workflow process model | 2001 |
| W8. | P.Grefen | Transactional Workflows or Workflow Transactions ? | 2002 |
| W9. | B. Reiner, E. Siegel, J.A. Carrino | Workflow optimization: current trends and future directions | 2002 |
| W10. | P. Hung, K. Karlapalem | A Secure Workflow Model | 2003 |
| W11. | B. Kiepuszewski, A.H.M. ter Hofstede | Fundamentals of Control Flow in Workflows | 2003 |
| W12. | C. Seggewies, et al | Soarian–workflow management applied for health care | 2003 |

**Table 1.** A user-specific DAFFODIL folder for *workflow*

recall values from crawling a virtually unlimited resource like the Web, we settled for measuring the precision at top 20 based on the binary user ratings shown in Table 2.

| No. | URL | BINGO! Rating | Key Resource | Feed-back |
|---|---|---|---|---|
| 1. | `http://www.e-workflow.org` | 3.86 | 0 | ⊖ |
| 2. | `http://www.waria.com` | 3.76 | 0 | ⊖ |
| 3. | `http://www.wfmc.org/standards/docs/Glossay_German.pdf` | 3.32 | 0 | |
| 4. | `http://www.wfmc.org/information/handbook2003.htm` | 3.05 | 0 | |
| 5. | `http://www.wfmc.org/pr/CDROM_2001.pdf` | 2.95 | 0 | ⊖ |
| 6. | `http://www.wfmc.org/information/info.htm` | 2.91 | 0 | ⊖ |
| 7. | `http://www.usc.edu/dept/ATRIUM/Papers/PDI.pdf` | 2.89 | 1 | ⊕ |
| 8. | `http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf` | 2.88 | 0 | |
| 9. | `http://www.wfmc.org/pr/Workflow_Handbook_2001.pdf` | 2.88 | 0 | |
| 10. | `http://www-dbs.cs.uni-sb.de/~gillmann/Publications/Demo-ICDE.pdf` | 2.86 | 1 | ⊕ |
| 11. | `http://www.wfmc.org/information/Workflow-An_Introduction.pdf` | 2.75 | 0 | |
| 12. | `http://www.wfmc.org/standards/docs/Stds_diagram.pdf` | 2.74 | 1 | ⊕ |
| 13. | `http://www-dbs.cs.uni-sb.de/~gillmann/Publications/Demo-SIGMOD.pdf` | 2.68 | 1 | ⊕ |
| 14. | `http://www.informatik.uni-stuttgart.de/ipvr/as/ projekte/poliflow/IPVR/adaptive_workflows.html` | 2.68 | 1 | |
| 15. | `http://www.wfmc.org/information/awards.htm` | 2.68 | 0 | ⊖ |
| 16. | `http://www.wfmc.org/standards/docs/tc003v11.pdf` | 2.63 | 0 | |
| 17. | `http://www.dfki.uni-kl.de/~aabecker/Freiburg/Final/Wargitsch/Wargitsch.pdf` | 2.63 | 1 | |
| 18. | `http://www.dfki.uni-kl.de/~elst/papers/kmdap_frodo_eval_submitted.pdf` | 2.55 | 1 | ⊕ |
| 19. | `http://www.csd.uch.gr/~hy565/Papers/overview_workflow_management.pdf` | 2.48 | 1 | |
| 20. | `http://www-dbs.cs.uni-sb.de/projekte/workflow_de.htm` | 2.37 | 1 | |
| | | **Precision** | **0.45** | |

**Table 2.** Top 20 BINGO! Web recommendations for *workflow* for the initial user folder.

Fortunately, all the top recommendations were correctly classified into the major topic workflow, which itself unfolds into two subtopics *workflow research* and *e-business/e-commerce*. Since the focus in the choice of the DLO's is clearly on the research subtopic, we restricted positive user rating to *key resources for workflow research* only, yielding a more conservative goal for measuring precision. Positive (⊕) and negative (⊖) feedback was manually provided for a subset of these documents in order to refine the classifier for workflow research in the next iteration. Negative feedback was used to extend the counterexamples for the classifier as a second-level, fine-grained filter between workflow research and e-business, while positive feedback extended the training set with the DLO's. To find more key resources with the refined classifier, the crawling depth was extended to 5 and limited by 10000 documents in the second run.

| No. | URL | BINGO! Rating | Key Resource |
|---|---|---|---|
| 1. | http://awareness.ics.uci.edu/~rsilvafi/wonder/ISADS99/isads99.pdf | 1.95 | 1 |
| 2. | http://www.ai.sri.com/~swim/resources/SOA-workflow.html | 1.80 | 1 |
| 3. | http://www-dbs.cs.uni-sb.de/~gillmann/Publications/ConfigTool-EDBT.pdf | 1.79 | 1 |
| 4. | http://awareness.ics.uci.edu/~rsilvafi/wonder/SBRC99/sbrc99.pdf | 1.79 | 1 |
| 5. | http://www.informatik.uni-stuttgart.de/ipvr/as/projekte/ apricots/atma96.pdf | 1.78 | 1 |
| 6. | http://www.research.ibm.com/journal/sj/361/leymann.html | 1.72 | 1 |
| 7. | http://www.cs.colorado.edu/~skip/proclets.pdf | 1.70 | 1 |
| 8. | http://www.ifi.unizh.ch/dbtg/Projects/TRAMs/trams.html | 1.64 | 1 |
| 9. | http://www.ifi.unizh.ch/dbtg/Projects/EVE/eve.html | 1.62 | 1 |
| 10. | http://www.almaden.ibm.com/cs/exotica/wfmsys.pdf | 1.61 | 1 |
| 11. | http://dis.sema.es/projects/WIDE/Documents | 1.58 | 1 |
| 12. | http://ccs.mit.edu/klein/cscw98/paper08/ | 1.54 | 1 |
| 13. | http://www.computer.org/proceedings/Hiccs2/0001/00010198Babs.htm | 1.51 | 0 |
| 14. | http://www.cs.toronto.edu/~avigal/nsfhtml/nsfhtml.html | 1.47 | 1 |
| 15. | http://www.jeffsutherland.org/oopsla97/schulze.html | 1.44 | 1 |
| 16. | http://osm7.cs.byu.edu/ER97/workshop4/ls.html | 1.33 | 1 |
| 17. | http://www.informatik.uni-ulm.de/dbis/f&l/forschung/ workflow/ftext-adept_e.html | 1.33 | 1 |
| 18. | http://www.jeffsutherland.org/oopsla98/kuechler.html | 1.32 | 1 |
| 19. | http://www.ifi.unizh.ch/groups/dbtg/Projects/SEAMAN/seaman1.html | 1.22 | 0 |
| 20. | http://ccs.mit.edu/klein/cscw98/paper07/ | 1.57 | 1 |
| | | **Precision** | **0.9** |

**Table 3.** Top 20 BINGO! Web recommendations for *workflow* after one feedback iteration.

Table 3 shows the positive calibration effect of the user feedback that successfully filtered out the e-business documents. Precision increased to 0.9, still accepting key resources for workflow research only. Only one document was incorrectly classified into workflow, and one was not a key resource. Both runs together yielded an overall micro-average precision of 0.675, which could probably be further improved by additional iterations. Most of the links found in the second run were excellent pointers to formerly unseen Web resources for workflow research, which demonstrates the significant benefit of feedback-based retraining and link analysis. Particularly interesting is the mixture of recommendations to scientific project homepages and full-text links to related research papers.

To compare our recommendation system to existing search engines, we manually entered the title of each DLO into Google and evaluated the "Find Similar Documents" search function. While Google typically provided very high precision in the search for known entities (typically the top were always relevant), only very few previously unseen key resources were found for such a specialized query, due to the nature of a query-driven search engine (among the top 5 the same resource often appeared multiple times in different forms, e.g., a Citeseer page, a research paper, the first author's homepage, etc.). Only for one DLO the "Find Similar Documents" option returned an interesting new key resource, namely, the OPERA[3] project as a result of the query with the title of the Mentor-Lite [W2] DLO. Citeseer offers a similar search function, but it is limited to digital

[3] http://www.inf.ethz.ch/department/IS/iks/research/opera.html

publications and does not give any pointers to Web resources such as project or scientists's homepages. So unless a user is able to specify the target of her search with much better accuracy or is willing to start with a vague topic query and then manually click through large link collections, it is complex to obtain convincing recommendations from current search engines. The combined DAFFODIL - BINGO! system was much superior to Google's or Citeseer's standalone services in this regard.

## 8 Conclusion

In this paper we have described the coupling of DAFFODIL and BINGO!, two advanced tools for information search and organization in digital libraries and Web sources, respectively. For DAFFODIL, BINGO! adds the power of context based search functionality and access to Deep-Web portals. For BINGO!, DAFFODIL enables personalization with access to semistructured metadata. The resulting combination provides substantial added value to advanced users such as scientists or students. Our future work includes a more comprehensive evaluation of the efficiency and effectiveness of the combined DAFFODIL-BINGO! system. In particular, the influence of relevance feedback will be studied in more depth.

## References

1. G. Alonso, F. Casati, et al. Web Services. *Springer Verlag*, 2003, ISBN 3540440089.
2. R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. *Addison Wesley*, 1999.
3. M.J. Bates. Idea tactics. *Journal of the American Society for Information Science* 30(5), 1979.
4. M. J. Bates. Where should the person stop and the information search interface start? 26(5), 1990.
5. Y. Batterywala, S. Chakrabarti. Mining Themes from Bookmarks. *ACM SIGKDD Workshop*, 2000.
6. C.J.C. Burges. A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 1998.
7. S. Chakrabarti, M.v.d Berg, and B. Dom. Focused crawling: A new approach to topic-specific Web resource discovery. *WWW Conference*, 1999.
8. Document Object Model (DOM). http://www.w3.org/DOM/
9. S. Dumais and H. Chen. Hierarchical Classification of Web Content. *ACM SIGIR Conference*, 2000.
10. C. Fellbaum. WordNet: An Electronic Lexical Database. *MIT Press*, 1998.
11. D. Fensel, C. Bussler, Y. Ding, et al. Semantic Web Application Areas. *NLDB Workshop*, 2002.
12. N. Fuhr, N. Gövert, and C.-P. Klas. An agent-based architecture for supporting high-level search activities in federated digital libraries. *ICADL Conference*, 2003.
13. N. Fuhr, N. Gövert, and C.-P. Klas. Recommendation in a collaborative digital library environment. Technical report, University of Dortmund, Germany, 2001.
14. N. Fuhr, C.-P. Klas, A. Schaefer, and P. Mutschke. Daffodil: An integrated desktop for supporting high-level search activities in federated digital libraries. *ECDL Conference*, 2002.

15. J.L. Herlocker, Joseph A. Konstan, Al Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. *ACM SIGIR Conference*, 1999.
16. F. Heylighen and J. Bollen. Hebbian Algorithms for a Digital Library Recommendation System *DCAL Workshop*, 2002.
17. J. Graupmann, M. Biwer, and P. Zimmer. Towards Federated Search Based on Web Services. *BTW Conference*, 2003.
18. T. Joachims. The Maximum-Margin Approach to Learning Text Classifiers. *Ausgezeichnete Informatikdissertationen 2001*, D. Wagner et al. (Hrsg.). GI-Edition - Lecture Notes in Informatics (LNI), Köllen Verlag, Bonn, 2002.
19. J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5), 1999.
20. J. Krause. Graphische Oberflächen für das Textretrieval im Rahmen des WOB-Modells. Skriptum, Universität Koblenz, Institut für Informatik, 1997.
21. F. Leymann. Web Services: Distributed Applications Without Limits. *BTW Conference*, 2003.
22. C.D. Manning and H. Schuetze. Foundations of Statistical Natural Language Processing. *MIT Press*, 1999.
23. OpenCyc, The Open Source Version of Cyc technology. http://www.opencyc.org
24. A. Paepcke. Digital libraries: Searching is not enough - what we learned on-site. *D-Lib Magazine* 2(5), 1996.
25. M.E. Renda and U. Straccia. A personalized collaborative digital library environment. *ICADL Conference*, 2002.
26. B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Reidl. Item-based Collaborative Filtering Recommendation Algorithms. *WWW Conference*, 2001.
27. Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/SOAP/
28. S. Sizov, M. Theobald, S. Siersdorfer, and G. Weikum. BINGO!: Bookmark-Induced Gathering of Information. *WISE Conference*, 2002.
29. S. Sizov, G. Weikum, et al. The BINGO! System for Information Portal Generation and Expert Web Search. *CIDR Conference*, 2003.
30. M. Theobald, R. Schenkel, and G. Weikum. Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data. *WebDB Workshop*, 2003.
31. M.B. Twidale, D.M. Nichols, and C.D. Paice. Browsing is a collaborative process. *Information Processing and Management* 33(6), 1997.
32. Universal Description, Discovery and Integration (UDDI) 2.0, http://www.uddi.org
33. V. Vapnik. Statistical Learning Theory. *Wiley, New York*, 1998.
34. Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl
35. Y. Yang and O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. *ICML Conference*, 1997.
36. Y. Yang. An Evaluation of Statistical Approaches to Text Categorization. *Journal of Information Retrieval* 1(1/2), 1999.