iWebDB - Web Site Administration based on Object-Relational Database Technology

Henrik Loeser and Norbert Ritter

University of Kaiserslautern, Department of Computer Science,
P.O. Box 3049, D-67653 Kaiserslautern,Germany
email: {loeser | ritter}@informatik.uni-kl.de

Abstract

Since the emergence of the World Wide Web in the year 1990, the number of accessible Web servers, HTML pages, and, thereby, the amount of accessible data have increased dramatically. Consequently, many people around the world are engaged in maintaining a Web server and its content. While creating HTML pages is getting more and more simple, the administration of a Web site is still a time consuming and demanding task. Available tools for Web Content Management (WCM) only cover some of the relevant aspects of the whole task; an integrated, fully functional solution is missing. Therefore, Web administrators still have to cope with many problems, such as ensuring validity of available documents or keeping hyperlinks consistent. In this paper, we present administrative components of iWebDB, an integrated WCM solution, based on object-relational database technology. By extending an object-relational database system (ORDBS) with WCM functionality, many tasks of WCM, such as validity checks and consistency checks for local hyperlinks, can be delegated to the DBS and, thereby, automated. Therefore, iWebDB achieves more effective and easier administration of a Web server and its content by exploiting the extensibility property of ORDBSs.

1. Introduction

Since the emergence of the World Wide Web (WWW or Web) in the year 1990, the number of accessible Web servers, HTML pages (Web pages), and, thereby, the amount of accessible data have increased dramatically. Private users, educational institutions, public authorities, and enterprises are providing information and, more or less useful, applications on their own Web pages. Thus, today many people around the world are in charge of maintaining a Web server and its content. And it can be expected that the number of persons engaged in this task will continue to increase.

While, due to a wide variety of authoring tools, the creation and layout of HTML pages is getting more and more simple and can be done by almost everyone, the administration of a Web site is still a time consuming and demanding task. Available tools for Web Content Management (WCM) only cover some of the relevant aspects of the whole task; an integrated, fully functional solution is missing. Therefore, Web administrators still have to cope with many problems, such as ensuring validity of documents, i. e., management of document availability and expiration, or avoiding invalid links.

On the other hand, due to the amount of Web pages to be managed, people have begun to use database systems (DBSs) for storing their pages and templates, as well as for creating pages dynamically. Currently, the utilized systems only serve as repositories for HTML documents. Further functionality can not be provided because the integration of such functionality into a standard DBS is rather difficult or even impossible. With the advent of object-relational database management systems (ORDBMSs) developers now can extent the database system by so-called user-defined functions (UDFs) that are executed within the server. In addition, user-defined types (UDTs) and index structures can be created; external data can be accessed via SQL.

Using OR technology, we build an integrated system for WCM, *iWebDB* [10]. We extended an ORDBMS to handle HTML documents as first class citizens and treat them like data of built-in types. iWebDB offers WCM functionality integrated into the DBS, such as consistency checks of local hyperlinks, extraction of search information, automated management of released/obsolete documents, simple version management, etc. Thus, many tasks of WCM can be delegated to the DBS and automated. Therefore, iWebDB achieves more effective and easier administration of a Web server and its content.

In this paper, we emphasize on administrative aspects of iWebDB, our integrated WCM system. First of all, starting from an analysis of current problems in WCM, we want to motivate a catalog of requirements (Section 2). Section 3 gives an overview of the iWebDB software architecture introducing its major components. Concentrating on administrative aspects, we discuss those components responsible for the integrated management of HTML documents and for the access to (from the viewpoint of the ORDBMS) externally stored documents as well as the administration interface in more detail in Section 4. This discussion will especially show, how iWebDB can fulfill the requirements given in Section 2. Section 5 covers implementation aspects. A comparison of iWebDB with other WCM approaches follows in Section 6. Finally, Section 7 concludes the paper and gives an outlook to further work.

2. Web Content Management Requirements

In this section, we want to identify requirements specific to WCM. For that purpose, we first consider the current situation w. r. t. big Web sites and corresponding problems. This discussion will lead us to a catalog of requirements.

2.1 Current Situation and Problems

Today, Web sites contain hundreds or even thousands of HTML documents. Usually, many persons or interest groups, e. g., departments, are involved in providing information and authoring Web pages. In addition, there is a multiplicity of CGI programs, for example supporting DB access, showing server statistics, or communicating with a locally available search engine. Therefore, setting up and maintaining such a Web site as well as coordinating all resources and providers is a hard and time-consuming job. Especially, if additional requirements regarding consistency of hyperlinks, page layout, and validity of pages, etc. exist, this task is extremely demanding.

Most of the HTML documents existing at a Web site contain hyperlinks to other locally held pages and document elements, e. g., images. Due to modifications performed by different persons resp. groups, and the amount of documents, it is very difficult to keep all hyperlinks up-to-date and to prevent invalid links. There are so-called link checkers, helping to detect invalid links. However, these tools have some restrictions. On one hand, they exclusively start checking with the root document, leaving document trees not referenced by the main tree untouched. On the other hand, they only run periodically or after considerable modifications, so that the documents remain unchecked for certain periods of time. Therefore, the problem of inconsistent hyperlinks cannot sufficiently be solved by using traditional link checkers.

Another difficult task is the maintenance of document validity. Often, a set of documents has to be published simultaneously, e. g., product information and detailed description. To manage this situation, administrators resp. information providers have to copy their documents all at once into the Web server file system, often into different directories. Additionally, some pages are newsworthy only for a small period of time. As soon as these documents get outdated, they should be removed in order to avoid waste of resources and to guarantee topicality. In order to handle these requirements adequately, administrators have to inspect the document base for outdated documents periodically, or special-purpose mechanisms, e. g., alerters, are to be applied.

Besides a topic index many Web sites maintain a search engine helping to find local documents. Nevertheless, documents of interest may not be found because index data is gathered and updated only periodically and documents can exclusively be reached via hyperlinks. Furthermore, before recording a newly created/published document in an index the corresponding provider must be contacted. Obviously, keeping index data topical is a general problem to be faced.

On corporate Web sites, a homogeneous page layout is wanted to establish corporate identity. Therefore, templates or style files are used to create new documents. Nevertheless, due to the multiplicity of information providers and daily changing Web site content, it is rather difficult to observe compliance with internal style guides.

Regarding the problems and difficulties mentioned in this section, it becomes obvious that Web site administration is a very demanding task.

2.2 Requirements

As motivated in the previous section, there are still several unsolved problems to be faced by Web site administrators and information providers. Starting from this discussion, we give a list of properties, which, as we think, an adequate approach to WCM must have:

- *Consistency*: When inserting a new document or updating an existing one, local hyperlinks must be checked. The insertion/modification must be rejected or at least the user must be notified, if the new document (state) contains hyperlinks referencing non-existent documents.
- *Integrated management of documents/elements*: HTML is not the only document type to be maintained at a Web site (XML [4] is coming), e. g., Postscript files. Consequently, functions for managing documents of other types are to be provided, too. Moreover, Web documents contain embedded elements, e. g., images, audio files, etc. Therefore, the WCMS should be capable of managing all document and element types.
- *Validity enforcement*: In order to simplify the publication of a collection of related documents and to control removal of outdated documents there must be some kind of *validity management*. This requires users to associate documents with publishing and expiration dates, so that validity can be maintained automatically.
- *Support for local search based on topical index information*: A search engine must provide effective access to locally managed and valid documents. Corresponding index information must be topical and complete in the sense that access to all currently valid documents, which are accessible by public, is supported.
- *Simple administration*: Web servers are file-based, administration of users/groups as well as most Web authoring tools are file-based, too. Therefore, on one hand, work with file-based tools must be supported, and, on the other hand, administration should be at least as simple as working at the operating system level.
- *Dynamic creation of documents*: Often, requests cannot be satisfied by delivering static documents, but must be answered by accessing (business) data stored in a DBS and dynamically creating HTML pages. Hence, access to business data must be supported, and document templates supporting dynamic creation of HTML pages must be managed.
- *Homogeneous layout*: Usually, on an enterprise Web site, all documents have to follow a given layout style in order to achieve corporate identity. Overhead for controlling compli-

ance of documents with given style conventions should be minimized.

- *Version management*: In order to manage several states of a single document, e. g., for documenting evolution or for automatically switching from one document version to another one at a certain point in time, version and configuration management services are needed.

3. iWebDB

In this section, we will give an overview of iWebDB, our integrated WCMS, and its components. iWebDB is based on an ORDBS [9, 12]. Using OR features like UDT, UDF, and external data access, iWebDB is designed to meet the requirements mentioned above by offering a single integrated system. As shown in Fig. 1, iWebDB consists of several components, which will be discussed later on. Some components are considered to be extension modules of the DB server. Depending on the DBMS vendor, such a module is called DataBlade (Informix), Extender (IBM), or Cartridge (Oracle). iWebDB uses this (extensibility) approach in order to achieve easier administration. Furthermore, it allowed us to develop and test iWebDB components independently, and, thereby, to shorten development times.

Since we want to emphasize the administration aspect in this paper, the following section will discuss those components of iWebDB specifically contributing to this aspect in more detail. First of all, however, we want to introduce each of the components of iWebDB by giving a short description, respectively.

- *iWebDB/HTML*: Providing all base types and tables for WCM as well as corresponding management functionality, iWebDB/HTML is the base component of iWebDB. Most components rely on this module and extend its functionality.
- *iWebDB/ED (external data)*: This module helps to ease Web site administration by allowing to integrate externally stored files into iWebDB. File system data accessible via Web protocols (“file:”, “ftp:”, “http:”) can be made available in document formats provided by iWebDB/HTML, and, afterwards, can be queried using SQL. Thus, external data can seamlessly be integrated into the system leading to a considerable simplification of Web site administration.
- *iWebDB/eXtract*: To enhance the search capabilities, iWebDB/eXtract offers additional functions for analyzing documents. Furthermore, functionality is provided for storing extracted data in a special index structure being exploited by a search engine. To implement this index structure, the OR feature of user-defined index structures is used. iWebDB/eXtract contains functions extracting text marked by logical and physical HTML tags. Additionally, functions to index Postscript and PDF (Portable Document Format) files are provided.
- *iWebDB/MP (macro processor)* is an extensible macro processor provided as a library. This can be used to easily realize a fully functional Web database gateway as well as to build other applications, e. g., iWebDB/DG. Using the macro processor, DBs can be accessed, SQL statements and functions be executed as well as variable processing be performed. iWebDB/MP supports access to several DBs while composing a single document.
- *iWebDB/DG (document generator)*: Based on iWebDB/MP, this module can be used to generate a set of template-based HTML documents. Utilizing triggers for generator invocation, based on a control file with instructions for document naming, template parametrization, etc., a single document or a set of documents can be created or updated at a given event.
- *iWebDB/SM (site manager)*: As a tool for site management, iWebDB/SM offers a graphical user interface (GUI) to all information providers of a Web site. Using iWebDB/SM, all tasks concerning installation and maintenance of Web documents can easily be performed.

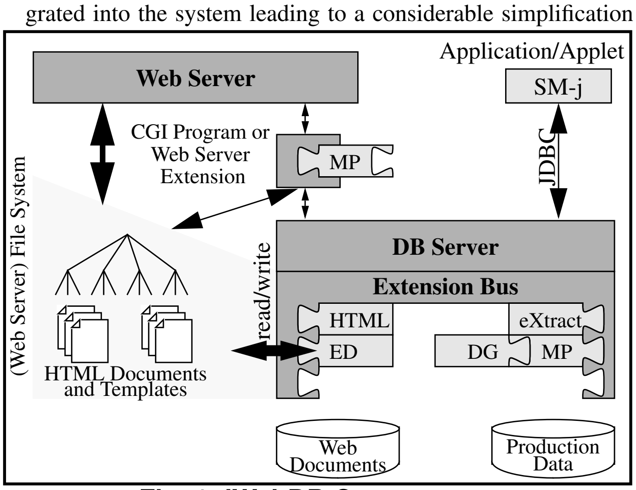

Fig. 1: iWebDB Components

4. Administrative Components

After the overview of all iWebDB components, we, now, want to concentrate on those modules specifically supporting Web site administration (iWebDB/HTML, iWebDB/ED, iWebDB/SM).

4.1 iWebDB/HTML

Acting as the iWebDB base component, iWebDB/HTML has to provide all types, tables, and base functions needed for storing documents and (embedded) document elements in the DB. For this purpose, iWebDB/HTML offers types and functions simulating a versioned file system.

Data Types

The central data type provided by iWebDB, *document,* is responsible for recording all administrative information, e. g., file name, owner, content type, etc.

Each instance of document type is associated with an instance of path and content. While path provides a basis for controlling resp. simulating file system directories, content is utilized for all categories of documents and embedded elements. As illustrated by statement 1 in Fig. 2, the document content (page) can be easily fetched using the file name and the path. Considering the last-update attribute (fdate), the names and paths of all documents owned by user “loeser” and updated within the last 10 days can be retrieved, if the file type is HTML and the document contains the word *Publications* in one of the headings (statement 2).

Functions

In order to analyze and manage stored documents, iWebDB/HTML provides several functions. Most of these functions manage HTML documents. An example is the function *headings* as used by statement 2. Among others, iWebDB/HTML provides functions to extract the title, the headings, address data, hyperlinks, or images. In order to speed up search, results from function calls can be stored in separate attributes or within a special table, depending on system capabilities. This functionality, however, together with other functions targeting different document types and images is provided by iWebDB/eXtract.

```
1. SELECT page
   FROM doctable
   WHERE path='/AG-Haerder'
   AND fname='LoeserD.html';

2. SELECT path || fname
   FROM doctable
   WHERE fdate > CURRENT YEAR TO DAY
   - INTERVAL(10) DAY TO DAY
   AND owner='loeser' AND type='html'
   AND 'Publications'
   IN headings(page);

3. INSERT INTO ftpDocs
   SELECT * FROM docs
   WHERE type='gif' AND path LIKE
   '/AG-Haerder%';

4. CREATE VIEW dbisDocs
   OF TYPE document AS SELECT *
   FROM docs
   WHERE path LIKE '/AG-Haerder/%'
   OR path='/'
   AND fname LIKE 'dbis%';
```

Fig. 2: iWebDB: SQL Examples

The above mentioned functions for the extraction of "metadata" from HTML documents are used to perform consistency checks. Hyperlinks are extracted, in order to check existence and validity of referenced local documents and elements. Consistency checks can be performed at different events. Using insert triggers, a user who wants to check-in an HTML document can be warned, if the page contains "dangling" links. Furthermore, documents can be examined after modifications. Another possibility is to check the documents before publishing them on the Web server.

4.2 iWebDB/ED

As already mentioned previously, iWebDB/ED provides access to external data sources. These are seamlessly integrated into the WCMS by providing their data in relational form and making it accessible via SQL. Sources are specified using the URI (*Uniform Resource Identifier*) notation, e. g., "file:/space/web" for the local web space. iWebDB/ED supports the protocols "file:", "ftp:", and "http:" to access a local files system, an FTP server, or a Web server, respectively. Due to the seamless integration of an external data source into the DBS environment, it is possible to access externally stored data in the same way as DB data. By using statement 3, e. g., all GIF images can be copied from the DB to an FTP server.

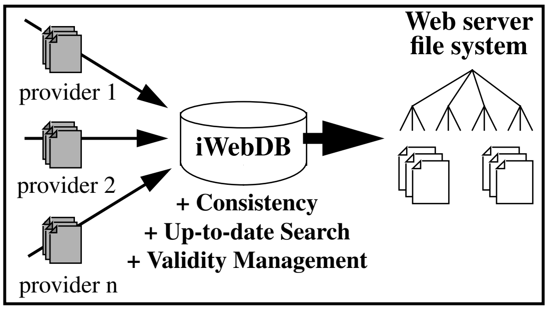

Fig. 3: Web Site Administration

The mechanisms introduced so far are well suited for adequate Web site administration. As depicted in Fig. 3, iWebDB can be used as a specialized repository for Web content. Providers check in their documents into iWebDB and, at proper points in time, the documents are published at the Web server.

Obviously, adequate security mechanisms are needed to ensure that information providers manipulate only their own documents and pass them to iWebDB with well-formed file names and paths. This problem is solved using the DB view concept. To each information provider a view with path and file name restrictions is assigned (see Fig. 2, statement 4) associated with certain access permissions.

Thus, each information provider may only modify its own documents and has to create new ones in a clearly defined name space. In the given examples, the users associated with view dbisDoc may only manage documents with path "/AG-Haerder/" and corresponding subdirectories, or the root path with file names prefixed "dbis". This approach allows to store all documents in the same table facilitating global document management. Moreover, using DB roles for information providers, a fine granular access control can be offered. Thus, secure sharing of a path among providers can be easily realized.

As already mentioned above, publishing documents at Web servers can be performed by iWebDB automatically. At publishing time additional provisions may be made to speed up subsequent document delivery by the Web server. So-called SSI (*Server Side Includes*) directives, usually responsible for dynamically, i. e., on request, supplementing the requested document by headers and footers, etc., can be (automatically) materialized already at publishing time. Dependencies between documents and document elements to be included can also be maintained automatically. For that purpose triggers are used ensuring that documents are republished as soon as a document element to be included, e. g., a header, has been modified.

4.3 iWebDB/SM

The graphical tool kit iWebDB/SM, the iWebDB site manager, facilitates the daily tasks of the site administrator and the information providers. The most important functions implemented by this module are the check-in of single documents as well as of document sets, the publication, and consistency control of hyperlinks.

Checkin is supported by providing a tree-based, group-dependent document view. Thus, the structure of documents stored within the DB appears to be the same as in a file system. The set of documents presented to a specific user depends on that user's role, i. e., the user's membership in particular administration groups. In addition to the Web pages stored within the DB, documents from an accessible data source, e. g., the local file system, are presented in a separate area of the GUI. This view relies on iWebDB/ED, so that both document areas can be accessed in the same way by using SQL queries. A Web page can be copied via "drag'n drop" from the file system to iWebDB. Automatic publication may be controlled by information providers by specifying release and expiration dates.

For simplified permission management, iWebDB/SM provides administrative tools for Web site administrators. Using these tools, groups and users can be added, removed, and modified, e. g., new privileges can be granted to a user or a group. Each group is associated with one or more path expressions indicating the name space in which a group may publish documents. Among other attributes, a group record stores the view name, e. g., "dbisDocs", so that this view can be altered whenever a group's name space is modified.

5. Implementation

Currently, a first version of iWebDB is implemented using Informix IDS/UDO (Informix Dynamic Server with Universal Data Option, [6]). While integrating functionality as C-based UDFs works almost fine, problems arise from limited modeling capabilities. This is a problem of all (commercially) available ORDBMS, because there is no commonly accepted definition of the term "object-relational". Consequently, there is no system providing all object-relational features with full functionality. For instance, apart from Illustra (no longer distributed), none of the available systems offers a reference type. As a workaround, we employed traditional relational primary/foreign key associations.

iWebDB/HTML provides its types as so-called *opaque types* and *row types*. That is, a type is made public using *support functions* or is defined based on available (built-in) types, respectively. While, for the first prototype, functions are implemented in C, it is planned to migrate to a Java-based implementation.

The implementation of iWebDB/ED is based on the VTI (*Virtual Table Interface*) of IDS/UDO, an interface simulating the relational scan operator for transforming, e. g., external data into table structures in order to connect them to database processing. Values/tuples that qualify for a given condition, are provided by iWebDB/ED row by row to the DB server.

The tool iWebDB/SM is designed as a hybrid Java program, i. e., usable as applet as well as application. In addition, a pure HTML solution is planned that will be based on iWebDB/MP. Thus, a GUI-based tool with sophisticated management functionality that is employable on any Web browser is offered. The Java program uses JDBC to access iWebDB. While the data for one document tree is based on the internally managed Web pages, the other document tree is dynamically filled up with data retrieved through iWebDB/ED. Thus, the same data structures as well as data retrieval functions can be used for composing the content of both GUI windows.

To speed up query evaluation, iWebDB/eXtract adds a (full) text search index to iWebDB. The index structure is implemented utilizing the VII (*Virtual Index Interface*) of IDS/UDO. VII defines the functions to be implemented and determines how the query optimizer can be guided to exploit the index. The provided secondary access method can be used to index an entire document as well as to realize a functional index, i. e., an index based on the result values of a given function.

6. Related Work

The system presented in this paper targets problems of Web Content Management. This pertains a relatively new, but already fast growing market with many products and systems under development. There are both research as well as commercial systems. In the following, we first compare iWebDB with other research projects, then we deal with commercially available systems.

There are projects for Digital Libraries and for the management of semi-structured data, e. g., SGML (*Standardized General Markup Language*) documents. In addition, many Web search engines, Web database gateways, and tools for Web page management exist. So what is new? Of course, iWebDB not only targets one aspect of WCM as done by many other projects. Furthermore, iWebDB tries to cope with the entirety of all problems arising while maintaining a Web site and its content. This is achieved by integrating all of the necessary functionality into an ORDBS. Thus, on one hand, Web documents are treated as first class citizens. On the other hand, the documents are not only stored, but can be intelligently managed and queried. Note that this is achieved by extending a standard ORDBS, so that potentially any ORDBS can be turned into an "iWebDB".

Analogous to iWebDB, WebOQL [2] is an integrative approach, too. However, it only covers document content and structure analysis. Document management as performed by iWebDB is not targeted. The systems ARA-

NEUS [3] and STRUDEL [5] aim at management of document components and "business" data to compose HTML pages. These are generated based on templates before runtime. Dependencies between documents and document components are maintained using a graph-based approach. While iWebDB, compared to ARANEUS and STRUDEL, is more flexible concerning document management and offers more functionality, e. g., integration of external data sources, search index, etc., both other systems are superior regarding static Web page generation.

With growing demands for WCM solutions, ORDBS vendors have started to build WCMSs. For instance, iSite [7] of Informix, a system just announced, is also built on top of IDS/UDO. It offers a facility to store documents and build dynamic Web sites, but lacks link checking and sophisticated search functions. Furthermore, WCMSs of other vendors exist, e. g., DynaBase [8] and Coldfusion [1] of Allaire. While DynaBase utilizes Objectstore, an object-oriented DBS, as document repository, Coldfusion relies on relational DBSs. Coldfusion is primarily designed to dynamically create Web pages based on document templates (analogous to iWebDB/MP) and lacks sophisticated WCM functionality, e. g., link checking, and document indexing, DynaBase is a real WCMS. It meets most of the requirements listed in Sect. 2, and offers even more functions than the iWebDB prototype, e. g., enhanced document versioning. The main difference between DynaBase and iWebDB is the system architecture. iWebDB extends a given ORDBS and integrates its components, DynaBase offers the functionality by its clients and application servers.

7. Conclusions and Outlook

In this paper, we have presented iWebDB, an integrated DB for Web Content Management. Today, most Web sites are managed using a file-system-based solution leading to a couple of problems, e. g., inconsistency, permission problems, outdated index data, and, therefore, increased administration efforts. By utilizing object-relational DB technology, the first time, HTML documents are treated as first class citizens of an DBS, so that management and administration can be performed at the DBS interface, i. e., by using SQL. In contrast to many other systems, iWebDB offers an integrated solution. Web site administrators and information providers benefit from data management facilities offered by DBS technology. Besides other features, iWebDB guarantees consistency of local hyperlinks leading to less user complains and higher Web site quality. In addition, users as well as administrators profit from increased search capabilities offered by iWebDB.

Currently, the iWebDB prototype consists of 6 components, most of them realized as DB server extensions, so-called DataBlades. In this paper, we have concentrated on administrative aspects and, therefore, detailed iWebDB/HTML, iWebDB/ED, and iWebDB/SM, which are the components especially contributing to effective administration. These components provide the foundation for WCM, iWebDB/HTML by defining data types and checking consistency, iWebDB/ED by integrating external data sources, and, last but not least, iWebDB/SM offering a "easy-to-use" GUI-based administration tool.

The current iWebDB prototype handles Web pages well, but lacks document versioning. For the future, it is planned to integrate versioning mechanisms from SERUM [11], a framework for repositories also based on ORDBMS technology. Regarding versioning, it is planned to meet the WebDAV [13] requirements to support the upcoming distributed authoring and versioning standard. Thus, iWebDB users will benefit from enhanced document management capabilities and seamless integration of other tools with the iWebDB approach. Another topic of future work is to support XML [4] documents. For that purpose, the analysis functions provided by iWebDB/HTML and iWebDB/eXtract must be adapted to new tags resp. syntax.

References

[1] *Coldfusion White Paper, Version 4.0*, Allaire Corporation, http://www.coldfusion.com, 1998.

[2] Arocena, G., Mendelzon, A.: *WebOQL: Restructuring Documents, Databases and Webs*, in: Proc. of ICDE'98, Orlando, Florida, February 1998.

[3] Atzeni, P., Mecca G., Merialdo, P.: *Design and maintenance of data-intensive Web sites*, in: VI Intl. Conference on Extending Database Technology (EDBT'98), Valencia, Spain, March 1998, pp. 436-450.

[4] Bray, T., Paoli, J., Sperberg-McQueen, C. M.: *Extensible Markup Language (XML) 1.0, W3C Recommendation 10-February-1998*, W3C, http://www.w3.org/TR/1998/REC-xml-19980210, February 1998.

[5] Fernandez, M., Florescu, D., Levy, A., Suciu, D.: *A Query Language and Processor for Web-Site Management*, in: Proc. of the Workshop on Semi-structured Data, Tuscon, Arizona, May 1997.

[6] *INFORMIX-Universal Server Documentation*, Informix Software, Inc., http://www.informix.com/, March 1997.

[7] *Content Management Solutions with Informix iSite - A Technical Overview*, Informix Software Inc., http://www.informix.com/informix/whitepapers/, 1998.

[8] *DynaBase - Dynamic Web Publishing System*, Inso Corporation, http://www.inso.com/dynabase/, 1998.

[9] Kim, W.: *Object-Relational - The unification of object and relational database technology*, UniSQL White Paper, 1996.

[10] Loeser, H.: *iWebDB - An integrated Web Database on basis of object-relational database technology,* (in German), in: Proc. of BTW'99, Freiburg, Germany, March 1999.

[11] Mahnke, W., Ritter, N., Steiert, H.-P.: *Towards Generating Object-Relational Software Engineering Repositories*, in: Proc. of BTW'99, Freiburg, Germany, March 1999.

[12] Stonebraker, M.: *Object-Relational DBMSs - The Next Great Wave*, Morgan Kaufman, 1996.

[13] Stein, G., Whitehead, E., Best, M.: *WebDAV Papers*, http://www.webdav.org/papers, 1999.