Title:

Integrating Database and World Wide Web Technologies

Authors:

Hongjun Lu
Department of Information Systems and Computer Science
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
Phone: (65)8742978
Fax: (65)7794580
E-mail: luhj@iscs.nus.edu.sg

Ling Feng (contact person)
Department of Information Systems and Computer Science
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
Phone: (65)8746136
Fax: (65)7794580
E-mail: fengl@iscs.nus.edu.sg

Integrating Database and World Wide Web Technologies

Hongjun Lu *Ling Feng*
Department of Information Systems and Computer Science
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
{luhj, fengl}@iscs.nus.edu.sg

Abstract

Integrating database and World Wide Web technologies is another topic where industrial and practical activities lead ahead of academic ones. The purpose of this article is to survey the related activities from database people's view and stimulate the interests among the database community. It covers three aspects. First, the efforts that apply established database techniques to retrieving Web information are summarized. These efforts aim to overcome the inadequacy of file system technology on which the Web is based, so that information can be retrieved easily and quickly from the Web. Second, various approaches to interfacing databases via the Web are discussed, with examples of accomplished projects and commercial products showing recent advances. Finally, some possible extensions to the traditional database techniques are investigated for building fully fledged Web-based database applications.

Keyword Phrases: Internet, the World Wide Web, interfacing databases with the Web, Web-linked database design, Web-based database applications.

1 Introduction

The World Wide Web (known as "WWW" or "Web") is growing at a phenomenal rate. The reason for the Web's success is largely due to its simplicity. It allows users to publish and retrieve information easily via the hypertext interface. Another important feature is its compatibility with other existing protocols, such as gopher, ftp, netnews, and telnet, etc. Moreover, it provides users with the ability to browse multimedia documents in an open environment available on many different platforms, requiring little cooperation between information providers and users.

The current Web is largely based on file system technology, which can deal well with the resources that are primarily static and of read-only nature. However, with the massive growth of resources, the knowledge about information existence and location, as well as the means to its retrieval, have become so bewildering that it becomes difficult for users to manage and use the Web anymore. Under such circumstance, conventional file technology is no longer adequate for undertaking the duties to organize, store and access large amount of information on the Web. Thus, many large Web sites today are turning to database technology to keep track of the ever increasing amount of data.

Database technology has played a critical role in the information management field during the past years. It is very effective at organizing, storing and retrieving data. Adopting these established database techniques, the above Web's scalability problem can be hopefully overcome. On the other hand, the new environment facilitated by the Web also requires rethinking many of the concepts in current database technology. Some traditional database techniques must be extended so as to fit the Web world. In short, the integration of the above two areas will bring us many opportunities for creating advanced information management applications.

This paper addresses three issues regarding such integration: first, what database techniques can be used for the Web; second, how legacy databases is interfaced via the Web; and finally, how traditional database techniques can be extended to adapt to the changing world.

The remainder of the paper is organized as follows. Section 2 discusses some efforts on applying database techniques to retrieving Web information. Section 3 introduces three kinds of approaches to interfacing databases via the Web. In order to build fully fledged Web-based database applications, section 4 discusses some specific problems presented by the nature of the Web, as well as some possible solutions. Section 5 concludes the paper with a brief discussion of future work.

2 Applying Database Techniques to Retrieving Web Information

With the exponential growth of the Web and the diversity of information it offers, using navigation-oriented browsers to find specific information in the Web is becoming increasingly difficult. More often, users are confronted with a large, heterogeneous, and constantly changing network. Except the stepwise exploration, they have no systematic way to retrieve information. In order to facilitate effective Web search, several efforts have been spent in applying database techniques to the Web, including building indexes, extending HTML language, adopting meta-representation mechanism, encapsulating resources into objects, and establishing host query servers.

2.1 Building Indexes

Index is a major facility for improving query efficiency in database systems. It plays a significant role in the traditional information management. Today a number of projects have been engaged in building Web indexes, and some certain indexes are being maintained over the Web. These Web indexes typically fall into two categories: administrator-generated indexes and spider-generated indexes. The former is created by index administrators who manually construct a document containing pointers to Web resources, while the latter is created by Web spiders (also referred to as robots, wanderers, crawlers, or worms), which occasionally scan the Web and build indexes of interesting keywords. Both types of indexes can be useful in locating information by using browsers or other automatic tools.

Among the well-known manually built index servers are Yahoo, the EInet Galaxy, GNN (the Global Network Navigator), and INFOMINE, etc. The main advantage of manual indexing is that Internet resources can be organized in some sort of classificatory schemas which can be alphabetical, chronological, geographical, subject-oriented, or a combination thereof. However, compared with spider-based indexing, manual indexing is much slower and less comprehensive [74]. Thus, many spider-generated indexes are established on the Web today [45]. Some well-known robots are WebCrawler [15], the RBSE (Repository Based Software Engineering) Spider [38], the WWW Worm, and the WISE (WWW Index and Search Engine) [73], Lycos, Alta Vista, Excite, and Infoseek, etc. These systems serve as brokers or directory services which support keyword searching using the index databases. Some of them can perform full- or near full-text indexing while others index only parts of the documents such as URL, titles, headers, weighted words, or anchor hypertexts of the documents. WISE schema lies between the full-text and title-only schemas. Its robot can extract words from page titles, all levels of headings, anchor hypertexts, leading sentence of every list item, and words or sentences in italic and bold face as the index data [73]. A detailed description and evaluation of various Web-based search engines can be found in [35, 52, 68, 62, 51]. Apart from building indexes for textual information, [23] recently extends the multi-media indexing technology

developed by the database community, and implements a simple index schema which can be used to access all types of multimedia objects on the Web.

Although the indexing approach is useful in capturing information on the Web and making them available for queries, it has some drawbacks. The drawbacks include: indexes cannot do well in capturing the hypertext structure of data; existing indexes do not share information; and also they do not provide a common interface [46]. In order to improve the results obtained from index engines that use a server to perform computations in response to information requests, some client-based Web search tools are developed. These tools, unlike the previous ones, perform the computations on the end-user's machine. They do automated navigation, starting from the answer of a major search engine such as WebCrawler or Lycos. Thus, they work more or less like a browsing user, but they are much faster and follow an optimized strategy.

Fish search is one of the most notable client-based search tool [26]. The *depth-first navigation* strategy of fish search complements the breadth-first strategy of WebCrawler in a neat way as WebCrawler knows only about a small fraction of the Web. Starting with a result from Lycos, users are able to find more documents using fish search. This is because Lycos does not use the right words for indexing other relevant documents [25].

2.2 Extending the Language

The Web provides a global universe of resources with hypertext links glued together. It is obviously not sufficient to search information based only on keyword matching. An efficient search facility should also makes use of the information conveyed by the formatting codes. Although HTML identifies structures within a document, it does not allow the semantic content of documents to be specified explicitly. To overcome this problem, one proposal is to extend HTML to include semantic information in a document's mark-up [34]. Three commonly used database concepts, i.e. entity, attribute and relation, are introduced into HTML. Information in the Web pages can be viewed as a collection of entities, having attributes which are connected by relations. This allows elements of a document to be identified with elements of a relational database. Thus database formalism can be imported directly into web pages, which in turn allows client- or server-side applications to extract information directly from them using relational database queries.

Further works are motivated with the objective of providing high-level facilities for presenting, locating, and filtering Web-held information. An SQL-like language which supports effective query processing over the Web is designed and implemented in [47]. It addresses the structure and content of Web sites, together with their various types of associated data. This language integrates new utilities and existing Unix tools (e.g. grep, awk), thus providing advanced display facilities for viewing extracted information. The robot-constructed indexes can also be utilized in the query

processing optimization.

[57] invents techniques to bridge the gap between HTML and SQL (the standard database query language). A general purpose variable substitution mechanism is proposed to conduct cross-language variable substitution between HTML input and SQL query strings as well as between SQL result rows and HTML output. This enables application developers to make full use of the capabilities of HTML for creation of query forms and reports, and SQL for queries and updates. This mechanism has been used in DB2 WWW Connection System to facilitate quick and easy construction of applications that access relational databases through CGI programs.

With dozens of interesting but incompatible inventions from different aspects described above, HTML is overburdened today and at the limit of its usefulness as a way of describing information. While HTML will continue to play a major role for the content it currently represents, many new Web applications require a more flexible and consistent infrastructure. In response to this, the World Wide Web Consortium (W3C) developed a simple dialect of SGML (Standard Generalized Markup Language), called XML (Extensible Markup Language), so as to enable SGML to be served, received, and processed on the Web in the way that is now possible with HTML [14, 11, 31]. XML describes a class of data objects called XML documents and defines the behavior of computer programs that process them. XML documents are made up of storage units, i.e., entities, which contain either text or binary data. Text data are made up of characters, some of which form the character data in the document, and some of which form the markup. Markup encodes a description of the document's storage layout and logical structure. As XML omits some complex and less-used parts of SGML, users can easily define documents and write applications to handle them.

2.3 Utilizing Meta-representation

In order to provide advanced search and browsing capabilities without imposing constraints on information suppliers and creators, several attempts have been made to build meta-representations for the original Web information. Metadata contains information about the type, representation, and location of physical data. It is widely used in multimedia, text and structured databases as an aid in the quest for seamless interoperability.

By maintaining meta-information about the infrastructure as well as user models and preferences, [50] establishes a system that can maximize the quality of information navigation on the Web. A semantic associative search for images is also implemented, based on metadata that represents the user's impression and the image's content [72].

InfoHarness is a system that supports off-line generation of metadata entities to encapsulate heterogeneous raw data, as well as the run-time use of these entities to search and present the encapsulated data [65]. The InfoHarness Repository Definition Language is used to simplify the generation of

metadata. It provides high flexibility in associating typed logical information units with portions of physical data, and in defining relationships between these units. It also provides a number of tools for the automatic generation of metadata [66].

In [36], a multi-layered system that performs cataloging of metadata to help users search for data is also developed. When new data items are received for archive, temporal, spatial, and spectral range information is automatically extracted and populated to a "summary" catalog database. At the same time, more detailed information is extracted and populated to a "program" catalog. Users can locate and obtain data through the summary catalog interface. Then, they can use the detailed program catalogs to analyze the data they have retrieved.

2.4 Encapsulating Resources Into Objects

To enhance the Web's flexibility and extensibility, some projects have brought the object-oriented database techniques into the Web. For example, W3Object model treats Web resources as encapsulated objects which interact with one another through well-defined interfaces [43]. These interfaces and their implementations can be inherited by builders of objects. And methods (operations) inherited can be redefined to better suit applications. Important properties such as concurrency control and persistence can be obtained by inheriting from suitable base classes. Hence, no changes to these resources are required.

[29] provides object-oriented support for the Web, so that data can be explicitly typed and associated with software methods that operate on them. The object-oriented interface to the Web allows for arbitrarily complex interactions and dynamic type extensibility, without requiring modifications to the Web client or server software for each new data type. Object orientation can provide other advantages such as code reuse, implementation hiding, etc.

Based on an Object-Oriented database (OODB), [32] proposes a server which stores documents as objects, the internal structure of which represents the HTML structure. This enables queries on structural elements like headers, hypertext links, quotes, databases, etc. A hypertext prototype "Nestor" for France Telecom is presented in [54]. It combines an OODB with the Netscape navigator through the intermediary of a script language. Recently, [44] even considers storing all the Web server data (e.g., cached data) in an object-relational database so as to alleviate the data retrieval latency problem. Using OODB techniques to organize dynamic structures of the Web, the view of the Web can also be customizable [71].

2.5 Establishing a Host Server

While a number of useful attempts such as indexing and meta-representation are made at finding data, there is still an element of luck involved for finding exact data of one's interest. Usually, users have to spend a lot of efforts exploring the Web in order to search for pieces of information that may lead them to the right direction. To relieve users from the navigation over the Web, a concept of host server was proposed [36]. A host server acts somewhat like a traditional distributed database server. It dispatches users' queries to corresponding Web sites containing the requested information. Each of these Web site servers will process the queries and return the answer to the client directly. Therefore, all that users need to do is to fill in a query form and submit the request to the host server.

3 Interfacing Databases With the Web

To integrate database systems into the Web, one major difficulty is that the HyperText Transfer Protocol (HTTP) is a stateless protocol. It does not support transactions in the general sense. Each request from a Web client is processed individually on a request-response basis. This is quite different from most database applications where a sequence of related requests of retrievals and updates forms a transaction. In this section, we will describe approaches, projects and products to address such problem.

3.1 Approaches

[60, 48] provided good surveys of the current approaches to integrating databases into the Web. Basically, they can be divided into three categories: server-side approaches, client-side approaches, and middle-layer approaches.

3.1.1 Server-Side Approaches

Server-side approaches focus on extending the functionalities of Web servers to access databases via the Web. The methods used include the Common Gateway Interface, Gateways as Stand Alone Servers, Server Side Includes, and Server Application Programming Interface.

. The Common Gateway Interface

Among the server-side approaches, the oldest and most common method is to use the Common Gateway Interface (CGI) and external gateway programs. CGI is a standard for external gateway programs to interface with information servers such as Web servers [3]. A gateway program, also

referred to as a CGI script, is executed at the server site. It is started by the Web server upon a client request. The gateway program receives the parameters supplied by the client, accesses the database through the underlying DBMS, and sends the results back to the server. The server then sends them to the clients who made the request. This approach provides a simple, easy-to-use method of executing programs from within an HTTP server. It can be used to create virtual documents and to interface with servers outside the normal HTTP server. The CGI interface has many advantages, including portability between server softwares, and wide availability of public domain gateway programs and development tools. However, since most gateway programs are external to the server, a process has to be created for each request with database accesses. Creating a process is time-consuming and expensive in terms of the server's RAM, and can also exhaust resources available to server applications. More importantly, most gateway programs are specific and application dependent. There are often the cases that new gateway programs have to be developed for new applications.

. Gateways as Stand-Alone Servers

To overcome the above drawbacks, another method is to establish gateways as stand-alone servers. Most database vendors use this approach to develop high performance Web-based database servers. Since servers can be optimized to interact with their DBMS directly, they can achieve great performance improvements. The major drawback is that such servers are often dedicated to a particular DBMS. Some vendors even require their own browsers to act as clients as well. For an organization with legacy systems from various vendors, it is quite an expensive approach to have different servers and browsers for different systems. In addition, there could be another integration problem for heterogeneous multidatabase access [53, 60, 61].

. Server Side Includes

Server Side Includes is another alternative to connect DBMS with the Web. A server side include consists of a special sequence of characters (tokens) embedded within an HTML document. Before the document is sent from the Web server to the requesting client, it is first scanned by the server for these special tokens. When a token is found, the server interprets the instructions in the token and performs the action accordingly. The resulting data retrieved from the database are then merged into the HTML document, and sent to the client. One of the drawbacks of this approach is that the result obtained is limited to only one single document with all the data in it. Besides, having the server parse documents can be costly, especially in the case of heavy load.

. Server Application Programming Interface

This method augments an HTTP server with an interface known as Server Application Programming Interface (SAPI). SAPI consists of a dynamic-linkable library (DLL) that contains executable

routines required to interface with different legacy information systems. The DLL routines function in the same way as gateway programs. However, unlike the external programs, these routines are loaded in the same address space as the server. Therefore, there is a minimal overhead associated with executing them as there is no additional process creation cost for each request. Furthermore, only those active functions will stay in the memory, as the server can dynamically swap-out non-active ones to free up system resources. As a result, the server performance can be improved [53, 6, 1, 2]. Another feature of SAPI is that it allows pre-processing of requests and post-processing of responses, thus permitting site-specific handling of HTTP requests and responses. SAPI filters can be used for applications such as customized authentication, access, or logging [5]. NSAPI (Netscape Application Programming Interface) and Microsoft ISAPI (Internet Information Server Application Programming Interface) are two examples of such method.

3.1.2 Client-Side Approaches

The server-side approaches mentioned above limit users' interactions to a sequence of selections and inputs. To support more complex applications (e.g., subsequent updates in several database tables that are requested in different forms), another kind of client-side approaches are presented. These approaches enhance the interactivity between clients and DBMSs. The idea is to distribute the application and send it to the client. The client then executes the code locally on the user's machine. Although the existing applications of the DBMSs will probably not fit into the Web, this idea allows rebuilding parts of the user interface on the Web and running them on the client computer. One simple way is to send a compiled program to the user and execute it there. This approach can lead to a better performance and high scalability. By loading the code locally, the load of the remote Web server can be reduced. Local execution can also be useful in cases where it is necessary to interact with a local device for which no cross-network protocol exists (e.g., unlike UNIX displays which can use the X-windows protocol, UNIX audio devices require a special-purpose piece of software to be run locally). However, these approaches require additional access control mechanisms, which typically include authentication procedures, security control, and integrity maintenance on the client computer. Besides having considerable security risks, these approaches are also platform-dependent.

The proliferation of Java programs written for the Web allows for more elaborate interaction with database systems, basically because they allow to overcome the limitations of the stateless HTTP [56]. With a Java-enabled Web browser, the client can make a direct connection to the DBMS server if a Java Database Connectivity (JDBC) is available for this DBMS [41]. In addition to Java, other script languages such as JavaScript can also be interpreted and executed by the client site (Web browser), hence allowing more interactivity and at the same time avoiding communications across the Internet (e.g. for consistency checks of inputs) [60].

3.1.3 Middle-Layer Approaches

While the previous two approaches are concerned with accessing single database from the Web, the middle-layer approaches allow for integration of data from distributed, heterogeneous data sources over the Web [48]. One of the most popular middle layers used today is based on CORBA (Common Object Request Broker Architecture). CORBA, developed by the Object Management Group (OMG), is a standard for distributed objects. It provides the mechanisms by which objects transparently make requests and receive responses. The CORBA ORB is an application framework that supports interoperability between objects, which are built in (possibly) different languages, and running on (possibly) different machines in heterogeneous distributed environments [12]. Based on CORBA, two methods can be developed to access databases via the Web.

. CGI scripts acting as CORBA clients

These CORBA clients running at server sites call a CORBA server which in turn accesses database systems.

. Web clients acting as CORBA clients

A direct communication between a Web client and a CORBA server is established through the IIOP (Internet Inter ORB Protocol) [19].

3.2 Projects and Products

So far, considerable efforts have been spent in integrating databases with the Web. On the market, hundreds of Web application development tools and products are available. In this subsection, we shall describe some of the projects working on the integration of databases and the Web. Commercial solutions by leading database providers like IBM, Oracle, Informix, Sybase, and O2, together with some gateway products from third party vendors like Cold Fusion, NetDynamics, Domino, InterNotes Web Publisher, WebBUILDER, and Web* are also discussed below. More descriptions of products and projects can be found in [9].

3.2.1 Projects

An early project was conducted at CERN in 1992, where an Oracle gateway for SQL SELECT statements was written by Arthur Secret. This gateway allows users to enter an SQL expression as a search text to a Web server. It then formats the results of the query and sends them back as plain text. Secret's work has been extended by Decoux at INRA to support among other things queries using forms, HTML connections between different rows and tables, as well as updates of

data in pre-specified tables [24].

The Translation Server, described in [58, 59], acts as an interface gateway between the textual user interface of a legacy database and the Web server. Each textual service has to depict its schema in a protocol description language. The Translation Server parses this schema and composes Web objects to get the necessary information from the user. The result is translated on the fly and returned as a Web object too, more presumably as an HTML page.

WDB is a software toolset that can tremendously simplify the integration of SQL-based databases into the Web [63]. It is a CGI script written in Perl for NCSA's HTTP Web server. The most noticeable feature of WDB is the possibility to convert data from the database into hyper-text links. As it is possible through WDB to access any database element directly via a Web URL, the entire database can be turned into a huge hyper-text system.

[69] presents an architecture for browsing OODBs over the Web. The architecture consists of a CGI dispatcher script, an intermediate data server, a simple intermediate data format for communicating between these two modules, and generic hypertext interfaces for browsing different database elements.

[30] develops an OMNIS-WWW gateway to access OMNIS digital libraries and office systems for the Web users. The implementation is based on the CGI definition for HTTP server. A CGI program is connected to an HTTP server as well as to several OMNIS database servers. It distributes incoming client requests to OMNIS servers and returns their query results to the HTTP server.

The National Space Science Data Center (NSSDC), located at NASA's Goddard Space Center, presents OMNIWeb, a Web-based system that allows users to produce plots and retrieve data from NASA spaceflight missions. OMNIWeb is constructed of CGI shell scripts, HTML generators, fill-out HTML forms, data listing tools, data subsetting and conversion tools, a graphical data analysis engine, an e-mail handler, and a clean-up script. OMNIWeb not only offers instant access to data but also has the added feature of supporting interactive data analysis [55].

Illustra's Web DataBlade module is a comprehensive toolset for creating Web-enabled database applications that dynamically retrieve and update Illustra database content. Instead of programming complex CGI codes, application pages can be created by using Web DataBlade tags, which are then stored in the database. The Web DataBlade module consists of two core pieces: the WebExplode and WebDriver [40]. With the Web DataBlade module, users can construct simple query front-ends in minutes and Web applications in just hours.

The Netscape Server API (NSAPI) is an enhancement of Netscape server, allowing extension and customization of the core functionality of the Netscape Server. It provides a scalable mechanism for building interfaces between the HTTP server and back-end applications. The NSAPI is designed to solve performance and efficiency problems common to installations that make liberal use of CGI

functionality [6, 4].

In [64], the authors suggest that the Web technology can be used for re-constructing old applications that are wrapped with an obsolete and unfriendly user interface. In this approach, an old application is conceptually wrapped into a container (i.e. translator) which gets its outputs and feeds it with user's inputs. The dialog with the user is mediated by HTML pages, which can present forms or take advantage of features like clickable images. In this way, the Web browsers can be used to provide enhanced capability to software systems, which were created when a graphical user interface was uncommon.

A Web-based Environmental Meta-Information System, WWW-UDK, is developed at FZI [49]. In this system, a UDK database is made available for both the Internet and Intranets using classical Web technology (Web clients, HTTP, CGI scripts) and C programs with embedded SQL statements. Features of WWW-UDK include multiple query modes, an environmental thesaurus, on-line access to the underlying data, multilingual query and result forms, and an on-line help system.

With the growth of the Web, database security becomes a prominent issue for applications such as stock trading, on-line shopping, etc. [27] describes a system that aims to build secure Web accessible database applications. It is composed of three main concepts: the secure loop which provides series of procedures like password checking to verify users at every request; the swap which maintains the state of a user between execution of CGI scripts; and the life-time of an input form which is a modularization of the task that an input form should complete. It is expected that through the use of such systems, platform independent application interfaces can flourish on the Internet.

In contrast to conventional CGI-based approaches, [37] proposes a methodology to develop database applications using Java and HORB (a software tool based on the concept of Object Request Broker). Its objective is to establish a robust Web infrastructure in a corporate environment.

[42] discusses the performance of accessing relational database over the Web. It has implemented a system with a caching mechanism and a Web server-integrated relational database interface. In addition, a relational database viewer is also developed to support the user-friendly interface. The authors test the system through several queries, with the results indicating that system performance can be greatly improved using the cached architecture.

3.2.2 Products

. Products from Leading Database Providers

(1) DB2 World Wide Web Connection

DB2 World Wide Web Connection (DB2 WWW Connection) is a Web server gateway to IBM's DB2 family. It allows developers to build Web applications for DB2 databases using a macro

language, which is a combination of HTML and SQL. HTML can be used for creation of query forms and reports, while SQL for queries and updates against DB2 databases. Meanwhile, DB2 WWW Connection provides a simple cross-language variable substitution mechanism to allow: input data from HTML input forms to be inserted into SQL calls to databases; and SQL query results to be fed back into HTML report forms. Various visual tools may be used to partially generate DB2 WWW Connection macro files. The DB2 WWW Connection runtime engine reads these macro files to generate the appropriate query forms, SQL commands, and report forms.

IBM's DB2 WWW Connection is designed to work with the DB2 family of relational database servers, HTTP Web servers and browsers, and the Internet Connection family of firewalls. In addition, it can also work with IBM middleware products such as DataJoiner, enabling Web access to non-IBM and non-relational data sources. DB2 WWW Connection is able to run on OS/2, AIX, OS/400, MVS/ESA, Windows NT, Sun Solaris, HP-UX, and SCO. It is planned to be available on VM/ESA in the future [13]. Currently, there is no maintenance available for DB2 WWW Connection. Developers need to learn the cross-language variable substitution mechanism before writing macros to build database applications,

(2) Oracle Web Application Server

Oracle Web Application Server is the industry's first transactional Web server with built-in transaction services to enable development of real-time business applications over the Web. It combines HTML-based Web server management tools, development tools and tight integration with Oracle database to support business-critical, transaction-based applications across Netscape and Microsoft HTTP servers. With Oracle's Web Request Broker (WRB) – a multithreaded, multiprocess application server, users can update the existing Netscape or Microsoft Web server to a scalable, reliable, and secure platform for Web-based business applications. By embracing industry standards such as Java and CORBA, Oracle Web Application Server lays a foundation for an open and inter-operable architecture [20].

At the moment, Oracle Web Application Server is only available on Windows NT (X86).

(3) INFORMIX-Universal Web Connect

Based on CGI, Informix-Universal Web Connect integrates Web servers with any Informix database server to let developers create Web applications that incorporate data retrieved from Informix databases. It provides a collection of Informix-specific HTML tag extensions. Users can also create their own customized tag extensions to perform specialized tasks. Unlike other Web solutions which store Web applications as flat files and scripts in the operating system directory, INFORMIX-Universal Web Connect stores all the Web application content including HTML files, application templates, and multimedia content within the Informix database, thus enabling developers to effi-

ciently manage the application as it grows in size and complexity[16].

INFORMIX-Universal Web Connect supports all popular Web servers, browsers and operating systems, and is compatible with other middleware solutions such as NSAPI and ISAPI. As INFORMIX-Universal Web Connect stores the entire Web application content in an Informix database, application developers should have some knowledge on using Informix database management systems, and all documents related to the application must be created in advance. During the execution period, dynamic updates against the application such as templates might not be easily performed, compared with updates against a file in the operating system directory.

(4) Sybase's NetImpact Dynamo

Sybase's NetImpact Dynamo provides an interpretive gateway between a Web server and Sybase SQL Anywhere Professional database. It passes SQL requests to the database, reformats the query results into HTML pages, and returns them to the Web server. By using SQL remote replication technology, both Web sites and databases can be replicated onto a laptop computer and used off-line, bringing complete dynamic contents to users even when disconnected from the Web.

NetImpact Dynamo supports three kinds of instructions that can be embedded into an HTML page: *SQL statements* to enable database queries to be placed into a Web page; *scripts* to include programmatic control (e.g., conditional execution, looping); and *text replacement and host variable macros* to respond to HTML forms and other variables supplied by a Web client. It provides a consistent interface to manage both Web sites and databases by plugging directly into Sybase's database management utility – SQL center [21].

NetImpact Dynamo is a completely HTTP-compatible server. However, it only runs on PC. Similar to DB2 WWW Connection, developers need to understand the embedded macros before they are able to create netImpact Dynamo Web applications.

(5) O_2Web

O_2Web provides Web clients with the ability to browse through hypermedia information stored in any O_2 database. It is made up of three main elements: O_2Web *gateway* (CGI script) to ensure the connection between the Web server and O_2 database; O_2Web *server* to run the queries sent by a Web browser and return the corresponding HTML code; O_2Web *dispatcher* to dispatch and send requests to the best suited O_2Web server in order to preserve load-balancing. O_2Web provides a toolkit to help developers manage Web applications such as HTML form management, image management, session management, and user management (e.g., authorization and role). Web clients access O_2 databases through an association query language OQL.

O_2Web is based on a standard Web server. It runs on workstations like Sun (Sun OS, Solaris), HP (HP-UX), IBM and Bull (AIX), Silicon Graphics and Siemens RW (Irix), SNI RM (Sinix) and

DEC Alpha (OSF1), and PCs like Solaris, SCO and Windows NT. As O_2Web is a package fully integrated into the O_2 environment, it is limited only to O_2 database resources.

. Products from Third Party Vendors

(1) Cold Fusion

Cold Fusion [10] is a Web application development tool that integrates browser, server and database technologies into applications. Unlike other Web application development tools, it uses a page-based application architecture and a server-side markup language (CFML) to allow developers to integrate Internet technologies, create application logic, and control page generation. A Cold Fusion application is in fact a collection of CFML pages. When a page in a Cold Fusion application is requested by a browser, the Cold Fusion Application Server processes the page, executes the CFML, and dynamically generates a Web page which is returned to the browser.

Cold Fusion supports both CGI and standard server API, thus can be directly connected to major Web servers. It works with all ODBC-compliant databases and includes drivers for most major databases. Currently, Cold Fusion Application Server runs on Windows NT and 95.

To interact with Cold Fusion, developers need to learn and write application pages with its tag-based programming language CFML. If providing a more intuitive visual interface, this tool might be used more easily by novice users.

(2) NetDynamics

NetDynamics [18] delivers the solution for developing Web-centric enterprise applications. It combines three components: *a Web application development environment, an application server*, and *WebEXTEND.* NetDynamics Studio is an object-oriented visual development environment that generates Java, SQL and HTML. The NetDynamics application server manages optimized data access, scalability, load balancing, data source connection, and so on. WebEXTEND enables organizations not only to leverage data residing in enterprise databases, but also to integrate with existing PeopleSoft and SAP applications with Java codes so as to create Web-centric business applications.

The platforms that NetDynamics supports include Windows NT/95, Sun Solaris, SGI Irix, HP-UX and IBM AIX, and databases include Oracle, Informix, Sybase, DB2 and other ODBC-compliant databases. NetDynamics is compatible with any CGI-compliant Web server, Netscape NSAPI-enabled Web server, and Microsoft ISAPI-enabled Web server. It is recognized as one of the leading solutions for the delivery of enterprise-wide applications on the Web.

(3) Domino and InterNotes Web Publisher

Domino [8] is a server technology that transforms Lotus Notes into an Internet application server, allowing any Web client to participate in Notes applications. Domino makes it possible to use Notes' rich application development environment to develop, manage and host Web applications. It facilitates Web clients to access dynamic data and applications on a Notes server. That is, Web clients can access a Notes server, search a Notes database, and view contents of Notes databases with Notes navigation capabilities. To run Domino, a Notes Release 4 Server and IP connection is required. Besides, Domino has the same hardware and OS requirements as the Notes Server.

Another product from Lotus is InterNotes Web Publisher [17], which extends Notes' capabilities to the Web, and provides features that the Web is currently lacking of. InterNotes Web Publisher can automatically translate Notes documents into HTML pages, and publish them to the Web. Meanwhile, it captures information from input forms via the Web, incorporating it into Notes business process applications. InterNotes Web Publisher supports Windows NT and IBM OS/2 operating systems. But it requires a Notes server and a standard HTTP server working together.

(4) WebBUILDER

WebBUILDER [22] is an Internet application development tool to build database applications that incorporate dynamic HTML. It uses an enhanced CGI as the interface to the Web server. Through WebBUILDER, business-oriented Internet applications can be built to process transactions, perform ad hoc queries, and manipulate multiple relational databases concurrently. WebBUILDER can run on various platforms, and is compatible with all major Web server software. It provides access to all popular relational databases. WebBUILDER is particularly suited for developers who want to work with multiple relational database management systems from within their Internet applications.

(5) Web*

Web* [7] provides a user-friendly environment for information providers on the Web. It is a programmable CGI script which can be added to an already-installed Web server. The script, implemented in Tcl (Tool Command Language), supports variables and control structures. It formats the HTML parts and results of the executed Tcl scripts according to HTML templates. Using Orbix TM software (an implementation of the CORBA standard), Web* also provides an interface from the script language to Orbix.

Installing Web* is harder than installing an average software package. The reason is that this package only works with an HTTP server. Some aspects of the HTTP server configuration also need to be changed.

4 Building Web-Based Database Applications

Traditional database applications are built based on centralized or distributed databases techniques. Most distributed databases are homogeneous and linked by Local and/or Wide Area Networks. Although techniques developed in integration of heterogeneous databases allow people to access heterogeneous database systems which may be geographically distributed, development of such applications are still rather complex and costly. With the globally available Internet and hardware independent Web browsers, it becomes possible to build Web-enabled database applications. The Internet and the Web free database application developers from the network maintenance. Once a database is linked to the Web, it is accessible from any Internet-linked computer in the world. Moreover, because the Web browsers are avaliable for almost all platforms, they eliminate the needs for designing different database application interfaces across different platforms. Today, more and more researchers and developers are building Web-based database applications [67].

In this section we highlight some issues in developing fully fledged Web-based database applications. The issues addressed include challenges to DBMS technology, Web-linked database design and the design of applications themselves.

4.1 Improving the Scalability of DBMS

The Web constitutes a global information universe. Anyone can access the Web as long as he/she is connected to the Internet. It is often the case that a hot Web site attracts a large number of users daily which usually does not happen in traditional database applications. Furthermore, the number of clients that can be served by a DBMS server is limited. Therefore, Web-based DBMSs face a challenge of scalability. That is, they should be able to support large number of transactions with reasonable response time. For read-only Web documents, the scalability problem may not be critical. However, when databases are open to the Web clients and certain clients, such as the traveling sales-people and executives, are allowed to modify the database, both the design of applications and traditional database techniques need to be reexamined in order to achieve such high scalability.

4.1.1 Reconciling the Transaction Models

For a fully fledged database application, clients should be allowed to update the database on-line. For example, an accepted order should trigger the update of available quantity. Allowing Web-clients to directly update the database poses challenges to the conventional transaction model used by the most current relational DBMSs.

Most DBMSs use the locking mechanism to enforce concurrency control. Under such scheme, write

lock is exclusive. When one transaction is holding the write lock of a data item, other transactions have to wait. Furthermore, for the sake of recovery, a transaction often holds locks until the transaction commits. It has been a well accepted model for debit-credit type of transactions to which most business applications, such as order processing, belong. One of the major characteristics of such transactions is that, a transaction will not hold the lock for a long time. However, in the Web environment, even a simple transaction may hold locks for a period that is long enough to degrade the performance of the system due to the communication delays.

The notion of long duration transactions was introduced when the similar problem was encountered in certain applications such as engineering database systems, where a user may need to hold a data item (e.g. a drawing) and work on it for a long period. Proposals of relaxing the ACID (i.e., Atomicity, Consistency, Isolation, and Durability) properties of traditional transactions have also been proposed. Considering the requirements for developing Web-based applications, reconciling the difference of various transaction models is necessary [33]. In addition to using different degrees of isolation provided by most DBMSs, DBMS may also consider to accept some hints on the property of requests from users. That is, a number of concurrency control mechanisms rather than one hard-coded transaction management strategy can be implemented in one DBMS. For example, a client application can give a hint that it just wants the data even if another transaction is holding a write lock on it. With such a hint, the DBMS transaction manager can retrieve the recent but perhaps a little outdated copy of the required data and deliver it to the Web server.

4.1.2 Pipelining and Storing Query Results

Traditional DBMSs deliver the query results to the user after they complete the execution of the query. For a complex SQL query, the execution time may be quite long. In the Web environment, such a long waiting time may lead to the disconnection between the client and the Web server. One possible strategy is to extend the pipelining concept to the output of a query plan: when the first page of query results is obtained, it will be delivered to the user immediately. While the server transmits the available data over the Internet, the DBMS continues to process the original query concurrently.

To address the problem of possible interrupted connection between the Web server and clients, [24] proposes that the DBMS writes all query results to a temporary table identifiable by the client's session, regardless of whether there is a connection established between the Web server and the client browser from which the query is originated. The application programs retrieve query results from the temporary table instead of the DBMS. In this case, even if the client has been disconnected from the server, the query result is still available which can be accessed by the client or sent to the client via e-mail facilities without re-executing the query.

4.2 Design of Web-Linked Databases

The uniqueness of the Web poses special requirements on the usage of Web-linked databases, hence requiring database designers to develop and employ new strategies to adapt to such an environment. In this subsection, we discuss two strategies, i.e., warehousing Web-linked databases and incorporating hyperlink semantics into database schema.

4.2.1 Warehousing Databases

After a company puts its product database on the Web, not only its own staff but also everyone else in the world can access the most current information about the company's products. In other words, the design of this database should be able to meet different requirements from a large number of users. For instance, traveling sales-people may need to access and update the database, traveling executives may require the most current summarized information of the company, and customers may want to query information on new products. In order to achieve a compromised yet acceptable performance for all kinds of users' queries, the database has to be re-engineered. For this purpose, techniques developed in data warehousing could be very useful. Widom gives a few examples of applications for which warehousing approach is appropriate [70]. Two of them are:

- clients requiring specific, predictable portion of the available information; and
- clients requiring high query performance, but not necessarily requiring the most recent state of the information.

Web-based database applications indeed have such characteristics. Borrowing the concept of data warehousing, a Web-linked database may have a layered architecture. For relational databases, each layer of the database consists of a set of tables. The bottom layer contains the base tables upon which transactions with modifications are executed, more or less like those traditional database systems. The upper layers of the architecture are materialized views of the base tables. The views are designed to meet the performance requirements of users from different categories and provide certain security control features. Note that, those views are materialized as in most data warehouses. Materialized views can provide high query performance but the information contained may be a bit out-of-dated, depending on the view update mechanism used.

Web-based application programs are designed accordingly such that different users or queries are directed to the different layers to access or modify different tables. With carefully designed views, it is hoped that, except transactions from internal users, only a certain percentage of transactions from general users require to access the base relations. Most frequently asked queries can be answered with simple operations on the materialized views. Therefore, the query performance can be improved.

Using materialized views to improve the query performance is somehow similar to the strategies of pre-execution of frequently posted queries and caching large amount of results [67]. However, materialized views can be implemented in a more systematic way. Of course, pre-execution and caching query results can also be used as supplementary strategies if there are such needs.

4.2.2 Incorporating Hyperlink Semantics

Before presented to Web users, data from databases should be converted into hypertext pages so that users can navigate both inside databases and other documents on the Web afterwards. [39] presents a language and a system to construct a hypertextual view (hyperview) of the content of a database. It adopts three approaches to mapping database objects to hypertext components: tuple level mapping, tuple set mapping, and associated tuple mapping. A declarative hyperview definition language is presented to integrate the three approaches, and a prototype translator is implemented to generate a Web hyperview of a relational database.

Instead of leaving such kind of conversion task completely to the additional tools as above, we can also consider introducing hyperlink semantics into the database schema, enabling DBMSs to specify, manage and manipulate hyperlinks of information directly. In this way, large documents can be broken up into smaller, more manageable chunks which are easily read and navigated by Web users. The efforts spent on conversion from data to hypertexts can also be reduced. Moreover, as DBMSs can provide good mechanisms to ensure referential integrity, the annoying problem of broken links between documents on the Web can be hopefully solved as well.

4.3 Design of Web-Based Database Applications

Here, we discuss two issues related to the development of Web-based database applications: retaining good features of traditional DBMS applications, and developing visual Internet interfaces to DBMSs.

4.3.1 Retaining Good Features of Traditional Database Applications

Large number of database applications on daily use are turn-key systems. Along with the development of database technology, such application systems have also developed their own good and unique features which make the database applications so popular. It is desirable to retain such good features in Web-based database applications.

Compared to the current Web-based applications, such as HTML form-based order processing programs, a fully fledged database application supports more comprehensive data entry help and validation. For example, in an order processing system, when a user is required to enter the customer

name, the system can access the database and display a list of current customers for user's selection. After the name is entered, related information such as credit limit will be retrieved and checked. For new customers, the information will be inserted into the database. Such help and validation are important features of successful database applications. On the other hand, current HTML form-based order systems allow Web users to submit requests in HTML forms. However, HTML forms provide less features with respect to data entry help and validation. More importantly, most validation is carried out after the form is submitted to the server. It is obvious that erroneous form will unnecessarily consume both the network bandwidth and server resources.

To validate input values, such as type and range, one can distribute the code for validation to the client side and the client executes the code locally on the client's machine. This is possible but the Web browsers' main feature, platform-independence, will not be available. Furthermore, modifications to the application may often cause the modification of validation procedures. Proper maintenance of the codes at client sides is very difficult and expensive.

With the development of script languages such as JavaScript, we can implement more comprehensive input data validation by embedding JavaScript statements in the HTML documents. The good feature of JavaScript is that it is an interpreted language. The Web browser can interpret and execute JavaScript statements locally so that the feature of platform independence can still be maintained.

Comprehensive help and validation require frequent access to the databases upon which an application is built. As mentioned in the previous section that, with the simple connect-respond-disconnect model of HTTP protocol, such help and validation are almost impossible. It is true that certain data needed for validation can be sent with the HTML form. However, only those relatively static data such as database schema and product codes can be distributed this way. Frequently changed data items such as quantity on hand should always be retrieved when they are required. From this viewpoint, keeping the database connection and providing fast access to the database is very important in developing Web-based database applications.

4.3.2 Developing Visual Interfaces

As front ends, Web browsers provide an Internet interface to DBMSs. Previous approaches let Web users query the database through forms. The middleware software such as CGI script translates the form inputs into SQL queries, passes the queries to the DBMS, and formats the query results as HTML pages. The Web server then returns the HTML pages to the Web browser to display the information to the Web user. However, in real-world applications, browsing and retrieving databases usually rely on complicated queries, and query results may force the user to go over long lists of information. In this case, simply structured forms and HTML pages are apparently

insufficient for Web users to interact with back-end DBMSs.

In order to exert the information presentation power of the Web and improve the human-computer interaction further so that a wide spectrum of users, even those with limited computer experience, can access databases effectively, some more intuitive visual interfaces to Web-linked DBMSs are necessary. These involve the capabilities for forming queries using interactive features like slides (not just forms), and representing the information in 2D or 3D.

Today, there exist some systems that adopt different visual representations and interaction strategies to handle complex information, and some offer sophisticated browsering mechanisms [28]. We can exploit these mechanisms, together with features of other visual query systems, in building Internet interfaces to DBMSs. In addition, we can also consider providing flexibility to Web users in choosing information visualization and query styles that suite their preferences.

5 Conclusion

Nowadays, considerable efforts have been made towards integrating database and World Wide Web technologies. This paper presents a survey of the current activities. It addresses three issues. First, several attempts that adopt database techniques to retrieve Web information are illustrated, including building indexes, extending HTML language, using meta-representation mechanism, encapsulating resources into objects, and establishing host query servers. Second, three approaches to interfacing databases with the Web are discussed. They are server-side approaches, client-side approaches, and middle-layer approaches. Based on one of these methods, many projects and products have been successfully developed. Finally, the paper investigates some possible extensions to the traditional database techniques in order to build fully fledged Web-based database applications. By reconciling DBMS transaction models (e.g., passing hints to the transaction manager), and pipelining and storing database query results, we can improve the scalability of Web-enabled DBMSs. By warehousing databases and incorporating hyperlink semantic into database schema, we can design high-performance Web-linked databases to meet different requirements of Web users. We may also consider retaining good features of traditional applications, and developing more intuitive visual Internet interfaces in the design of Web-based database applications.

Integrating database and World Wide Web technologies is another topic where industrial and practical activities lead ahead of academic ones. With limited resources, it is impossible also not our intention to cover all the activities in this short article. Our purpose is to stimulate the interests among the database community as we feel that there are still quite a number of issues to be addressed by both academic researchers and practitioners [67]. Some of them are listed below:

- Since different Web users may use different capabilities of database systems (e.g., some users

are willing to use active rules while others are not), new mechanisms need to be devised to monitor DBMSs to provide users with their least required facilities.

- As many autonomous users may access Web-linked databases, the security and privacy of information stored in the databases should be highly protected. The development of flexible authentication and authorization mechanisms is especially important for Web-based DBMSs.
- DBMSs will face unreliable data inputs from various users. New methods should be developed for evaluating the reliability of input information.
- With databases open to a wide range of users on the Internet, we should consider integrating service-purchase and charging strategies into DBMSs.

Acknowledgment

We would like to thank anonymous reviewers for their detail comments which enable improving the quality of the paper.

References

[1] A specification for writing Internet server applications (a high-performance alternative to CGI executables). Process Software Corporation, http://www.micrsoft.com /intdev /inttech /isapi.htm, 1995.

[2] Standards documentation – the netscape server API. Netscape Communications Corporation, 1995.

[3] CGI - The Common Gateway Interface. NCSA HTTPd Development Team, http://hoohoo.ncsa.uiuc.edu /cgi, 1996.

[4] Netscape products – netscape communications server. Netscape Communications Corporation, http://www.netscape.com /comprod /netscape_commun.html, 1996.

[5] Publishing dynamic applications. Microsoft Corporation, http://www.microsoft.com /infoserv /docs /program.htm, 1996.

[6] Standards documentation – the NSAPI versus the CGI interface. Netscape Communications Corporation, 1996.

[7] Web*. Concurrent Engineering Research Center (CERC), http://webstar.cerc.wvu.edu, 1996.

[8] All about Notes and Domino. Iris Associates, Inc. http://www.notes.net /about.nsf, 1997.

[9] Ask the SQL Pro, resources useful sites. Inquiry Company, http://www.inquiry.com/techtips /thesqlpro /webdatabase.html, 1997.

[10] COLD FUSION 3.0. Allaire Corporation, http://www.allaire.com /project /interfaces /binder.dbm, 1997.

[11] Common asked questions about the XML. W3C SGML Working Group, http://www.ucc.ie/xml, 1997.

[12] CORBA and OMG information resources. The Object Management Group, http://www.acl.lanl. gov /CORBA /#DOCS, 1997.

[13] DB2 World Wide Web Connection. IBM Corporation, http://www.software.ibm.com /data /db2 /www/, 1997.

[14] Extensible Markup Language XML. W3C Working Group, http://www.w3.org... /TR /WD-xml, 1997.

[15] How WebCrawler works. Excite, Inc. http://webcrawler.com /Help /Aboutme /HowItWorkd.html, 1997.

[16] Informix product line and technology overview. Informix Corporation, http://www.informix.com /informix /products /new_plo /plo1.htm, 1997.

[17] Lotus ships InterNotes Web Publisher release. Lotus Corporation http://www.lotus.com /corpcomm /238e.htm, 1997.

[18] NetDynamics. NetDynamics, Inc. http://www.netdynamics.com, 1997.

[19] OMG, CORBA / IIOP. The Object Management Group, http://www.omg.org /corba /corbiiop.htm, 1997.

[20] Oracle Web Application Server. Oracle Corporation, http://www.olab.com/products /websystem /webserve, 1997.

[21] Sybase SQL Anywhere Professional and the Internet. Sybase Corporation, http://www.sybase.com /products /system11 /workplace prowhite.html/, 1997.

[22] WebBUILDER. VPE, Inc. http://www.vpe.com, 1997.

[23] B. Agnew, Z. Wang, C. Faloutsos, and D. Welch. Multi-media indexing over the Web. In *Proc. of the SPIE - the International Society for Optical Engineering*, San Jose, CA. U.S.A., February 1997.

[24] M. BJÖRN. An interactive relational database gateway with load balancing. In *Proc. of the Asia-Pacific World Wide Web Conference*, Australia, September 1995.

[25] P.M.E. De Bra. Finding information on the Web. CWI Quarterly, http://wwwis.win.tue.nl /~debra /cwi-qw /article.html, 1997.

[26] P.M.E. De Bra and R.D.J. Post. Information retrieval in the World Wide Web: Making client- based searching feasible. In *Proc. of the 1st International Conference on the World Wide Web*, Geneva, Switzerland, May 1994.

[27] R. Buccigrossi, A. Crowley, and D. Turner. A comprehensive system to develop secure web accessible databases. In *Proc. of the WebNet96*, San Francisco, CA, U.S.A., October 1996.

[28] T. Catarci. Interaction with databases. *IEEE Computer Graphics and Applications*, 16(2), March 1996.

[29] B. Chhabra, D.R. Hardy, A. Hundhausen, D. Merkel, J.D. Noble, and M.F. Schwartz. Integrating complex data access methods into the Mosaic / WWW environment. Computer Science Department at the University of Colorado, Boulder, http://dixxy.ncsa.uiuc.edu /SDG /IT94 /Proceedings /DDay /schwartz /schwartz.html, 1994.

[30] A. Clausnitzer, P. Vogel, and S. Wiesener. A WWW interface to the OMNIS /Myriad literature retrieval engine. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[31] R. Cover. The SGML Web page. http://www.sil.org /sgml /sgml.html, 1997.

[32] J.T. de Munk, A.T.M. Aerts, and P.M.E. De Bra. OODB support for WWW applications: disclosing the internal structure of hyperdocuments. In *Proc. of the WebNet96*, San Francisco, CA, U.S.A., October 1996.

[33] D.J. DeWitt. DBMS - Roadkill on the information superhighway. In *Invited Talks, Proc. 21th International Conference on Very Large Data Bases*, Zurich, Switzerland, September 1995.

[34] S. Dobson and V. Burrill. Lightweight databases. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[35] X. Dong and L.T. Su. Search engines on the World Wide Web and information retrieval from the Internet: a review and evaluation. *Online & CD-ROM Review*, 21(2), April 1997.

[36] B.N. Dorland, W.A. Snyder, R.D Jones, S. Heinicke, and D.A. Becker. Finding the data: Using metadata indices to locate and obtain data on the World Wide Web. E.O. Hulbert Center for Space Research, U.S. Naval Research Laboratory and Hughes STX Corporation and Massachussettes Institute of Technology Media Laboratory, http://edsweb.u-strasbg.fr /waw /dorland /dorland-l.html, 1996.

[37] N.N. Duan. Distributed database access in a corporate environment using JAVA. *Computer Networks and ISDN Systems*, 28(7-11), May 1996.

[38] D. Eichmann. The RBSE spider – balancing effective search against web load. In *Proc. of the 1st International Conference on the World Wide Web*, Geneva, Switzerland, May 1994.

[39] G. Falquet, J. Guyot, and I. Prince. Generating hypertext views on databases. In *Proc. of the XIV INFORSID Congress*, Bordeaux, France, June 1996.

[40] J. Gaffney. Illustra's Web DataBlade module. *SIGMOD Record*, 25(1), March 1996.

[41] G. Hamilton and R. Cattell. JDBC: A Java SQL API. Technical report, Sun Microsystems, Inc., 1996.

[42] M. Hatada and H. Endoh. Development of relational database access over WWW. *Transactions of the Information Processing Society of Japan*, 38(2), February 1997.

[43] D. Ingham, M. Little, S. Caughey, and S. Shrivastava. W3Objects: Bringing object-oriented technology to the Web. In *Proc. of the 4th International Conference on the World Wide Web*, Massachusetts, USA, December 1995.

[44] D. Jadav and M. Gupta. Caching of large database objects in Web servers. In *Proc. of the 7th International Workshop on Research Issues in Data Engineering*, Birmingham, UK, April 1997.

[45] S. Kimmel. Robot-generated databases on the World Wide Web. *Database*, 19(1), February 1996.

[46] D. Konopnicki and O. Shmueli. Defining a high level information gathering system for the World Wide Web. Technical report, The Faculty of Computer Science, The Technion - Israel Institute of Technology, 1995.

[47] D. Konopnicki and O. Shmueli. W3QS: A query system for the World Wide Web. In *Proc. of the 21st Conference on Very Large Data Bases*, Zurich, Switzerland, September 1995.

[48] R. Kramer. Database on the Web: Technologies for federation architectures and case studies. In *Proc. of the ACM International Conference on Management Data*, Tucson, Arizona, June 1997.

[49] R. Kramer, R. Nikolai, A. Koschel, C. Rolker, and P. Lockemann. WWW-UDK: A web-based environmental meta-information system. *SIGMOD Record*, 26(1), March 1997.

[50] D.R. Lamas, S.J. Jerrams, and F.R. Gouveia. CAIN: Computer aided information navigation: project description. In *Proc. of the WebNet96*, San Francisco, CA, U.S.A., October 1996.

[51] H.U. Leighton. Performance of four World Wide Web index services: Infoseek, Lycos, WebCrawler and WWW Worm. http://www.winona.msus.edu /is-f .library-f /webind.htm, 1996.

[52] J. Liu. Understanding WWW search tools. Reference Dept. IUB Libraries, http://www.indiana.edu /~librcsd /search/, 1996.

[53] H. Lu and L. Feng. TWINS: Bridging the gap between databases and the Web. National University of Singapore, submitted for publication, May 1997.

[54] Y. Marchand, J.-L. Guerin, and J.-P. Barthes. From a set of technical documents to a hypertext system on the Web. In *Proc. of the WebNet96*, San Francisco, CA, U.S.A., October 1996.

[55] G.J. Mathews and S.S. Towheed. NSSDC OMNIWeb: The first space physics WWW-based data browsing and retrieval system. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[56] P. Naughton. *The JAVA handbook*. Berkeley, Calif. Osborne McGraw-Hill, 1996.

[57] T. Nguyen and V. Srinivasan. Accessing relational database from the World Wide Web. In *Proc. of the ACM International Conference on Management Data*, Montreal, Canada, June 1996.

[58] L. Perrochon. Translation servers: Gateways between stateless and stateful information systems. In *Proc. of the Network Services Conference*, London, November 1994.

[59] L. Perrochon. IDLE: Uniform W3-access to interactive information servers. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[60] L. Perrochon. On the integration of legacy information systems into the World Wide Web. In *Internet-Tutorial of the 1st joint annual meeting of the Gesellschaft für Informatik (GI) and the Schweizer Infomatiker Gesellschaft (SI) (GISI'95)*, Zürich, Switzerland, September 1995.

[61] L. Perrochon. W3 'Middleware': Notions and concepts. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[62] A. Poulter. The design of World Wide Web search engines: a critical review. *Program*, 31(2), April 1997.

[63] B.F. Rasmussen. WDB – a Web interface to SQL databases. http://arch-http.hq.eso.org /bfrasmus /wdb /wdb.html, 1995.

[64] M. Ronchetti, V. D'Andrea, G. Succi, and D. Feltrin. Face lift: using WWW technology for an external reengineering of old applications. In *Proc. of the 3rd International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[65] L. Shklar and C. Behrens. Web access to NASA's remote sensing data through geoharness. Bell Communications Research and Rutgers University, http://athos.rutgers.edu /~shklar /www4 /shklar.html, 1996.

[66] L. Shklar, K. Shah, and C. Basu. Putting legacy data on the Web: A repository definition language. In *Proc. of the 1st International Conference on the World Wide Web*, Darmstadt, Germany, April 1995.

[67] A. Silberschatz, M. Stonebraker, and J. Ullman. Database research: Achievements and opportunities into the 21st century. *SIGMOD Record*, 25(1), March 1996.

[68] N.G. Tomainolo and J.G. Packer. An analysis of internet search engines: assessment of over 200 search queries. *Computers in Libraries*, 16(6), June 1996.

[69] C. Varela, D. Nekhayev, P. Chandrasekharan, C. Krishnan, V. Govindan, D. Modgil, S. Siddiqui, O. Nickolayev, D. Lebedenko, and M. Winslett. DB: Browsing object-oriented databases over the Web. Department of Computer Science, University of Illinois, USA, http://www.physics.uiuc.edu /dimych /paper.html, 1995.

[70] J. Widom. Research problems in data warehousing. In *Proc. 4th International Conference on Information and Knowledge Management*, Baltimore, Maryland, November 1995.

[71] J.J. Yang and G.E. Kraise. An architecture for integrating OODBs with WWW. *Computer Networks and ISDN Systems*, 28(7-11), May 1996.

[72] Y.Kiyoki, T. Kitagawa, and T. Hayama. A meta-database system for semantic image search by a mathematical model of meaning. *SIGMOD Record*, 23(4), December 1994.

[73] B. Yuwono, S.L.Y. Lam, J.H. Ying, and D.L. Lee. A World Wide Web resource discovery system. In *Proc. of the 4th International Conference on the World Wide Web*, Massachusetts, USA, December 1995.

[74] B. Yuwono and D.L. Lee. Search and ranking algorithms for locating resources on the World Wide Web. In *Proc. of the 12th International Conference on the Data Engineering*, New Orleans, Louisiana, February 1996.