COMBINING DATABASE TECHNOLOGY WITH THE WORLD WIDE WEB FOR TELE-TEACHING ENVIRONMENTS

P. KUTSCHERA

University of Stuttgart, Institute for Parallel and Distributed High-Performance Systems (IPVR), Breitwiesenstr. 20-22, 70565 Stuttgart, Germany

Abstract

Todays most popular information system, the World Wide Web (WWW), has a great potential to be used within a tele-teaching environment, because of its availability on a great number of different hardware platforms, the appealing graphical interface of the browser and the huge framework of resources that is world-wide on-line accessible. But, due to its well-known limitations like the missing navigational aid during the access of on-line resources world-wide, the existence of invalid links, the limited search capabilities, etc. the World Wide Web lacks some of the features an efficient and user-friendly tele-teaching environment should provide. Thus, the approach presented in this paper is to combine the features of the World Wide Web with the functionality of database systems to overcome these limitations.

1 Introduction

The world we are living in today is changing at a great pace and knowledge becomes obsolete soon. This is the reason, why everyone has to learn new things all of the life and keep the knowledge up to date. Unfortunately, not everyone has the time or possibility to attend courses in school or other institutions. Preferably, everyone should be able to attend courses at whatever time he or she wants to. Thus, there is an increasing need for efficient and better learning environments using telecommunication services which are accessible from school, the job or at home. An environment that meets the prerequisite to make a huge framework of resources world-wide on-line accessible and provides the means for user interaction as well as support for multi-media content is the World Wide Web (WWW), one of the most popular information systems today. Its popularity is based on the free availability of the software (server and browser) on almost every hardware platform, the appealing graphical interface of the browser and the simplicity to make existing as well as new information on-line available. In a tele-teaching environment, the World Wide Web can be used to announce courses, make all of the course material including additional cross references on-line available and to provide interaction for course registration, exercises or even animated tutorials. Although the World Wide Web has a great potential to be used within a tele-teaching environment, it comprises some limitations like the well-known "Lost in Hyperspace" syndrome, the existence of invalid links, the limited search capabilities, etc. that make the World Wide Web not suitable for an efficient and easy-to-use tele-teaching environment. However, in combining the features of the World Wide Web with the functionality of database systems, like the easy and standardized access and administration of different type of data, the flexibility to combine different data sources within a presentation, the ability of pre-processing of the data as well to guarantee their consistency during access, tele-teaching environments can be developed that provide world-wide on-line access to different teaching material and overcome the limitations of the World Wide Web as well.

2 Using the WWW for Tele-Teaching: Pros and Cons

In the following section we want to explain, why it is a good idea to use the World Wide Web as a basis for a tele-teaching environment, what kind of limitations exist and how some of them can be addressed by using database technology.

The World Wide Web software is not only available on almost every hardware platform and for every operating system, even a lot of good quality free software exists for non commercial use. Based on its hyper-exponential growth, the Web is a kind of a standard for on-line presentations of all kind of information including text, graphics, audio and video clips. Documents are formatted according to the HTML (Hypertext Markup Language) standard, are easy to develop and can be integrated into the Web very easily. The concept of hyperlinks that is part of HTML gives an author the opportunity to interlink the own documents to different resources all over the world. In a tele-teaching environment, hyperlinks can either be used as a means to structure a course into sections or subsections or to denote cross references to additional material for further readings. The Web even provides some basic form of interaction through the use of the CGI (Common Gateway Interface) together with HTML forms. These capabilities can be used to develop multiple choice tests or fill-out forms for exams.

In summary, using the World Wide Web as a basis for a tele-teaching environment is a reasonable choice, but the following limitations of the World Wide Web still exist:

- The interaction between the user and the WWW server is based on the stateless HTTP (Hypertext Transfer Protocol) request/response protocol. Thus, no context is preserved between different invocations and the user has to retain this kind of information by himself. However, in a tele-teaching environment it is important to monitor the users interactions to gather some results about the course efficiency as well as the user's effort.
- The World Wide Web provides the user with a navigational interface along the existing hyperlinks. While in principle, it is a very elegant way to link related information together, it has been observed that users often feel disoriented after following a few hyperlinks ("Lost in Hyperspace"). This phenomenon results from a missing navigational aid similar to a table of contents in a book or a guided tour through the course material. Another problem related to hyperlinks is that the World Wide Web does not guarantee their consistency. Thus, encountering an invalid hyperlink leads to frustration of the user or incomprehension of this particular situation. A tele-teaching environment should enforce link consistency as well as additional navigational aids for the user.
- Although there are many different resources available on the World Wide Web, it is very difficult to locate them. Currently, either a directory service such as Yahoo or one of the existing search engines such as AltaVista, Web Crawler, etc. [19] can be used. While this approach is feasible for searching the entire Web, in a tele-teaching environment this method causes to much of an overhead. However, a user should be able to query a site to find all of the relevant material about a certain topic.
- On the World Wide Web it is easy to integrate new material and thus, everyone can become an author. But, this also implies that the layout of every document is based on the author's preferences. A tele-teaching environment, however, should present all of the course material in a uniform fashion (fonts, colors, etc.) to ease the comprehension.
- Resources on the World Wide Web are managed by the Web server and the access rights are based on the operating systems capabilities. Thus, external users have in general no possibilities to insert, modify or delete resources. In a tele-teaching environment, however, users should be able to create new material as well as delete or modify existing material. Thus, a user management is needed to assign different access rights to different types of users for all of the available resources.

After describing the limitation of the World Wide Web in regard of tele-teaching environments, the next section presents two new Web developments that enable the development of interactive applications like graphical frontends, chat tools, applications for cooperative work, etc. which could be part of tele-teaching environment. Next, we give an overview about related work, before we introduce our concepts and describe how the limitations of the Web can be addressed by using database technology. Finally, the realization of these concepts is presented in more detail and give an outlook on future work.

3 New Web Developments

Using the standard CGI interface for validation of HTML forms input as well as to perform simple computations, these operations have to be performed on the server side by the CGI script and cause an increased load factor of the server. Thus, the goal by introducing the *JavaScript* scripting language by Netscape Communication Corporation was to reduce the load of the Web server by enhancing the capabilities of the browser to execute these tasks itself. JavaScript extends standard HTML forms with event handling capabilities through callbacks, extended GUI features and the functionality of a simple programming language.

The *Java* Programming Language[1] was developed at Sun Microsystems Inc. to be used for programming on the Internet. Thus, Java programs are compiled into a platform independent bytecode that can be transfered over the Internet and executed at a remote site. A special kind of Java programs, the so-called applets, can be included into HTML documents and executed by any browser with a built-in Java interpreter. Java is a full-fledged object-oriented programming language with network programming, threads and GUI support. Currently, many extensions for Java including Java database access (JDBC[10]) have been announced and will be fully available at the end of next year.

4 Related Work

Obviously, our proposal touches many different topics that range from database access and graphical frontends for navigational access to full fledged tele-teaching applications. It is neither possible nor meaningful to make a full account of all those ramifications here. Thus, we chose to refer to some research results that we found significant, but this reflects only our personal view and does not claim to be complete.

The same limitations like those inherent in the World Wide Web have already been encountered in other hypermedia systems in the past[13]. Thus, the approach that led to the development of the so-called *second generation hypermedia systems* to overcome these limitations is similar to ours. One of these systems is Hyper-G[13], a client/server based system that has been developed at the IICM of Graz University of Technology. In contrast to the Web, Hyper-G treats hyperlinks as bidirectional objects which are stored separately from the document types (audio, video, text, image, etc.) that may contain them. Documents as well as hyperlinks are stored in an object-oriented database system. Thus, Hyper-G is able to guarantee consistency of the hyperlinks by keeping track of the changes in the document structure. Furthermore, the link information is used to guide the user while navigating a Hyper-G site. Besides multi-language support and hierarchical user management, Hyper-G offers search facilities to query a document by keywords or title as well as full-text retrieval. For the interaction between client and server, Hyper-G uses a connection oriented protocol. Although, it is not possible for Hyper-G to access the Web directly, gateways between Hyper-G and the Web already exist. In summary, our goal is to enhance the Web with the same features as proposed for second generation hypermedia systems, but instead of developing a completely new system, we chose to combine standard components (Web software and databases) to overcome the current limitations of the Web.

Database interfaces that enable *World Wide Web based access* by using CGI scripts to access the database exist for almost every database system (see [9, 14] for example). But, most of these interfaces can only be used to access "traditional" data (relations or objects). The only exception is the Illustra Web Datablade[4] which provides access to other types of information like images, video and audio clips as well. All of these approaches use vendor-specific HTML tags to include database queries directly into HTML documents.

Only a few approaches exist that use database systems within a Web environment. In [16] a database system is used to store link information to guarantee *local consistency of the existing hyperlinks*. Another approach that is described in [17] stores the information included in the META-tags of HTML documents in a database to provide keyword search capabilities.

Tele-teaching itself is a very active area of research[15] and many different applications are currently developed or ported to the World Wide Web[5, 8, 12, 18]. These applications range from course-specific solutions to frontend tools that provide navigational aid[11].

5 Concepts

Our goal is to combine the World Wide Web and database systems in a way that all of the available resources on the World Wide Web (text, images, audio, video, Java applets, etc.) as well as additional meta data are stored and managed by the database system. This design has the advantage that the database system can be used for pre-processing of the data prior to their retrieval, insertion, modification, etc. to obtain additional meta data. Thus, these meta data can be used to provide the following services:

User management: Prior to accessing information in a tele-teaching environment, a user has to register to the system. During the registration phase the access rights of a user for the different resources are determined and a unique user id is assigned. Every time, a user wants work with the environment, he or she has to identify himself or herself during a login procedure. After the login, the corresponding user id is retrieved and will be part of every user interaction with the system. Thus, the user's interactions with the system can easily be monitored to gather access statistics and information about the user's efforts. These statistics can be used to evaluate the quality of the existing material and as a feedback for the user about his or her progress.

Search facilities: Prior to storing HTML documents in the database additional meta data like keywords, headings, captions, etc. will be extracted. Thus, the built-in query language of the database system can not only be used for full-text or keyword searches, but also to search specific parts of the documents very efficiently.

Consistency of hyperlinks: The consistency of the hyperlinks for all of the documents that reside in the local database can be guaranteed by extracting all of the relevant link information during insertion of a HTML document and storing them separately in the database. To keep this information up to date, all of the documents' changes have to be monitored and the link information has to be updated accordingly. Prior to retrieving a document, the validity of the existing hyperlinks has to be checked. If invalid hyperlinks exist, they have to be removed during the retrieval of the document.

Navigational aid: Additional link information can be provided by annotating all of the existing hyperlinks with type attributes (cross reference, chapter, exercise). This information can be used together with the document structure by frontend tools to provide navigational aid.

Uniform document layout: A uniform layout of the course material can be enforced by storing document types (templates) for every course or different parts of a course in the database. Prior to the insertion of a new document or the modification of an existing document, the layout of the document has to be checked against the corresponding document type. If there is a mismatch, the operation is rejected and the document has to be modified accordingly.

Views: It is likely, that some information which is available on the Web already exists in a different representation and may even be stored in the database. Thus, it should be possible to include references to data located in the database into a HTML document. By the way, storing information independent of the presentation format has the advantage that it is possible to have multiple presentation formats (views) without the need to replicate the presented information. Furthermore, changing the presentation format means no longer changing multiple documents independently, but simply modifying a single presentation template. In a tele-teaching environment, these views can be used to display information for different kind of users in a different fashion.

In summary, we have to admit that the concepts presented here are very similar to the design goals of second generation hypermedia systems, but instead of building an completely new system, we chose to combine standard components, namely World Wide Web software and a database management system, to overcome the limitations of the Web.

6 Realization

The following section describes how the World Wide Web server accesses resources in the database and what kind of meta data is needed to realize the concepts of the previous section.

6.1 Database access via World Wide Web

Information that is located in a database system can not be accessed as easily by the Web server as information that is stored in the local file system, because no URL notation for referencing resources in the database currently exists. Developing a World Wide Web based access to database systems, the following four different approaches have to be considered which all differ in regard of efficiency and the necessary modifications to the World Wide Web software that have to be made:

1. The standard method to access external data sources is to use CGI scripts for the interaction with the database system. Using this method, the CGI script has to transmits the users' queries to the database system, receive the query results and include them into HTML documents.
2. The Web browser itself could be used to directly access the database system without any interaction with the Web server. This approach is used by the JDBC interface[10] that supports database access for Java applications and in Java-based CORBA[3] implementations.
3. Another candidate for accessing the database system is the Web server itself. This approach can either be implemented by modifying the software of the Web server or by using an extensible Web server like SunSoft's Jeeves[6] or Jigsaw[2], a Web server developed by the World Wide Web Consortium.
4. Finally, database access can also be achieved by introducing a specialized protocol between the browser and the Web server. This approach leads to the modification of the Web browser as well as the Web server software.

Although, using CGI scripts is the standard method to access external data sources on the World Wide Web, we do not use this type of access due to the overhead involved in the start up of a CGI script for every database request. Since we do not want to make any changes to the existing World Wide Web software to ensure usability in many different environments, we chose the following approach:

- An extensible World Wide Web server is used to access the database directly. This is achieved by introducing a special form of URL notation, a so-called "database URL", to reference data that is located in the database system. Thus, the World Wide Web server itself connects to the database transforms the "database URL" into a database request, receives and processes the results before sending the resulting HTML page back to the browser. This approach should not only be more efficient than using CGI scripts, but it also offers an migration path from using the local file system to store all of the resources to the use of a database system.

- Searching the database as well as inserting, deleting and modifying of resources is accomplished by using the browser to directly access the database. This is an efficient way to access the database without the need to interact with the Web server.

Thus, we propose the following special form of URL notation ("database URL") to access a database system. The syntax is given using a BNF-like form:

```
<Database URL> = http://<Server Name>[:<Port>]/<Database Ref>
<Database Ref> = /(db)/<Databases>[/<Entities>][/<Attributes>][?<Query Params>]
<Attribute>    = <Entity Attribute> | <Bytecode Ref> | <Presentation>
<Bytecode Ref> = <Entity Attribute>/<Class name>
<Presentation> = <Entity Attribute>(<Entity Attributes>)
```

Although, the syntax presented here is only a small part of the overall definition, this should be enough to illustrate the similarities and differences compared to the regular URL syntax. In general, a "database URL" like any other URL consists of a protocol type, a server name, an optional port number and a path name. The prefix *(db)* of the path name denotes a reference to the database and is used by the Web server to distinguish between resources located in the database and the local file system. The main idea by developing this URL notation was that a hierarchical naming scheme can be applied as well to denote resources in a database, because a database has a hierarchical structure as well. Information in every database system is partitioned into different databases, every database consists of many different entities and an entity consists of a set of attributes. Thus, the main difference between an URL denoting a file name and our proposed "database URL" is that multiple data sources, as indicated by the **s** in ⟨*Database***s**⟩, ⟨*Entitie***s**⟩ and ⟨*Attribute***s**⟩ can be combined for retrieval. Multiple data sources are referenced by concatenating the appropriate resource names with the ampersand. A "database URL" is able to reference different attribute types like text, images, video, audio (⟨*Entity Attribute*⟩), Java bytecode (⟨*Bytecode Ref*⟩) or presentation formats (⟨*Presentation*⟩) respectively. The definition of the ⟨*Query Params*⟩ in our proposal is identical to the same term in the CGI specification. These parameters denote name-value pairs that can be concatenated by an ampersand and can be used to restrict the result set of a database reference.

Instead of a long theoretical discourse, we prefer to illustrate the usage of our "database URL" with a few examples:

Accessing a single source of data: Information stored in a single entity type like the HTML document that contains section 2.2 of the course "Programming in C" can be retrieved using an URL like */(db)/wwwdb/courses/section?section_nr="2.2"&course_name="Programming in C"*.

Accessing multiple data sources: Information that is distributed across different entity types like a summary of all section headings of courses about database systems can be retrieved by an URL notation like */(db)/wwwdb/sections&courses/courses.name&sections.name?course.id=section.id &course.description="%database systems%"*.

Using presentation formats: Sometimes it is necessary to provide a unique presentation layout for all of the information that belongs to a certain topic or to display information using a particular form of presentation. An example use of a presentation format can be demonstrated with an URL like */(db)/wwwdb/teachers&layouts/layout.lecturers(teachers.name, teachers.image)?teachers.topic= "computer science"* which uses the "lecturers" presentation format to display the name and image of every lecturer that is specialized in computer science.

After this short introduction, it should be clear in which way our proposed "database URL" can be used. Using this kind of notation, it is easy to enhance existing HTML documents with database references and make these information accessible world-wide.

In general, a "database URL" can be used in an HTML document like any URL notation. Because we felt that the inclusion of query results into HTML documents should be as easy as the inclusion of images, we extended the HTML syntax with a special INCLUDE-tag (Syntax: ⟨**include** ⟨*Database URL*⟩ ⟩) to provide this functionality.

6.2 User management

Prior to accessing our tele-teaching environment, every user has to register to the system specifying a user name and a password. During the registration phase, a unique user identification is assigned and the user's access rights are determined. When the registration has been completed the meta data about the existing users in the *UserTable* ⟨*name, password, userId*⟩ has been updated, the user has been assigned to the appropriate group of users (*GroupTable* ⟨*groupId, userId*⟩ and the access rights of the user have been determined and stored in the *AccessRights* ⟨*entity, attribute, access, accessType, id, authType*⟩ data structure. A user's access rights determine which types of operations are restricted/allowed (based on the *accessType* attribute) on the different entity attributes. Access rights can either be specified on a group or on a user level (based on the *authType* attribute) and have to be checked prior to the retrieval, insertion, modification and deletion of every resource.

After the registration phase, the user can log in to the tele-teaching environment by specifying his or her user name and password. Thus, the Web server has to retrieve the meta data about the existing users, check the user's authorization and obtain the identification of the user. The identification of the user should be part of every interaction between Web server and browser to be able to gather access statistics for every user. To be able to store the user's identification as context information between successive HTTP requests, we chose to use Netscape's cookie mechanism. A cookie is a piece of context information that the Web server includes into the header of a HTTP response, the browser stores locally and includes into the header of any successive HTTP request. Thus, the Web server is able to extract the user identification out of the HTTP header and to store it along with the user's request, the current date and time in the database. Based on this information several types of access statistics can be collected. Data that is frequently requested by many different users is very likely more relevant than other parts of the data. Information that is accessed several times from the same user is most likely too difficult to comprehend. Based on the access times, it is possible to find out, how many time a user uses for his or her studies. These statistics can be used to restructure or change a course to increase the efficiency of the course material.

6.3 Search Facilities

The use of existing World Wide Web search engines in a tele-teaching environment would not be a reasonable choice. These engines perform a full-text, depth-first search along the hyperlink structure of the existing HTML documents until a certain search depth has been reached. Using the database system of our tele-teaching environment, the built-in query language of the database system can be used to provide keyword as well as full-text search capabilities. A full-text search can be performed by searching the stored HTML documents.

Additional search capabilities can be provided by storing additional meta data in form of the *DocumentInfo* ⟨*documentId, tag, contentType, content*⟩ structure in the database. This kind of meta data can be used to store some of the documents' structural information like headings, captions of figures and tables as well as keywords describing the contents of the documents. Based on these information and the general ability of database systems to provide efficient value based retrieval, it is possible to give the users better response times and a more flexible search facility compared to the existing search engines of the World Wide Web.

6.4 Consistency of Hyperlinks

Every user of the World Wide Web has already encountered the situation of a broken link where the requested document could not be accessed. To guarantee the consistency of the existing hyperlinks an additional data structure is needed where all of the relevant link information are stored as well as monitoring all of the changes that affect the validity of the links included in the documents. A data structure called the link table of the following form *LinkTable* ⟨*source, target, linkType, status, newLocation*⟩ has to be used to store all of the relevant link information. For every link, the source document where the link resides, the target where link refers to as well as the link type are stored. It has to be mentioned that the term "link" in our context is not only used to denote hyperlinks, but is used to denote all other kind of references like images, Java bytecode, etc. as well. The monitoring of changes that affect the existing link structure can be performed automatically by the database system using triggers. Based on the different types of operations, the following actions have to be performed:

Insertion of a new document: All of the relevant link information have to be extracted and stored in the link table.

Deletion of a resource: All of the rows in the link table that contain the deleted resource as its source can be deleted. Additionally, all of the links that contain the deleted resource as its target have to be marked invalid.

Modification of a document: In general, a modification comprises deletions as well as insertions. Thus, the appropriate operations have to be performed.

Changing the location of a resource: All of the rows in the link table containing the resource's old location as its source have to be updated according to the new location. Rows containing the resource's old location as its target have to marked invalid and the new location has to be added.

All of the information of the link table has to be used prior to retrieving a HTML document residing in the database to check if the links are still valid. Normally, this will be the case and the Web server accesses the resource and sends it without further modifications to the browser. If invalid links exist, the resource has to be modified accordingly and the invalid links have to be replaced. Replacing an invalid link means in case of a link that references a resource that has been deleted to remove this link from the document. If a resource has changed its location, the reference in the document has to be modified according to the resource's new location that has been stored in the link table. It has to be noted that these changes of the document structure during runtime will probably only be necessary for a short period of time until the author or administrator of a resource has been contacted by e-mail and the existing inconsistencies have been removed.

6.5 Navigational Aid

Most of the information that is needed to provide the user with a navigational aid through the course material is already available. The structural parts of a document are stored in the *DocumentInfo* and the hierarchical structure of the course material can be obtained by using the *LinkTable*. By extending HTML's HREF-tag with the so called *context attribute* that has to be added to the link table as well, a hyperlink can additionally be annotated according to its use as a cross reference, structural link, a reference to an exercise, etc. Thus, a frontend tool (navigator) may use this kind of information to display the structural parts of a course, all of the existing hyperlinks as well as the user's current location within the course material.

Another type of navigational aid can be provided by using guided tours[7]. Guided tours are paths through the course material that have been recommended by the authors. To provide an easy adaption in case of changes in the course material these kind of decorations should be stored independent of the HTML documents that are part of a guided tour. Thus, the meta data that could be used to store information about guided tours may be structured as *GuidedTours* ⟨*tourId, docId, prevDocId, nextDocId, firstDocId*⟩ where every document of a guided tour along with its predecessor, successor as well as the starting document is stored. During the retrieval of a document that belongs to a guided tour these additional decorations have to be included during runtime.

6.6 Uniform Document Layout

In our view, it is a very important issue to provide an uniform layout of the entire course material, because the comprehension of the user is much better when all of the structural parts (definitions, summaries, overviews, etc.) of a lecture use the same layout. Thus, we propose to store document templates along with the structural parts of a course as well. Prior to an insertion of a new document or the modification of an existing document, the document structure has to be checked against the document template. Documents that are not formatting according to the template are rejected and have to be changed according to the generated error messages. An example for such a templates are document type descriptions (DTDs) in the Structured General Markup Language (SGML) that are used to check the validity of a document's layout. Thus, our proposal is to structure this kind of meta data similar to the *DocumentTemplates* ⟨*entity, attribute, template*⟩ structure and use it to enforce an uniform layout of all HTML documents that are stored in the specified entity's attribute. We are convinced that enforcing a uniform layout of the course material leads to an improved efficiency of the learning process.

6.7 Views

In a tele-teaching environment, it is necessary to develop course material for users with different types of backgrounds. Thus, the same data should be presented differently without the need to store it multiple times. This is what presentation formats or views are used for. Presentation formats (see section 6.1) are HTML documents that itself contain references to resources residing in the database as well as variable references that correspond to parameters that are passed to the presentation formats. Thus, presentation formats can be used to provide a unique layout for the same type of data as well as a means to store the presented information independent of the presentation's layout. So, changing the format of a presentation means simply modifying a presentation format instead of changing multiple documents independently.

7 Conclusion and Future Work

This paper presented the underlying concepts of a tele-teaching environment combining the World Wide Web and database technology to overcome some of the World Wide Web's current limitations. A prototype implementation that will serve as a testbed to show the feasibility of our concepts and as a basis for further development is currently under way.

It is our main goal to use our prototype implementation (which is nearly finished for a lecture to make the course material on-line available for our students. The students' experiences will then be used for the evaluation of the prototype with respect to robustness and performance as well as missing features. Additionally, we plan to develop some frontend tools which enable an easy administration of the course material.

References

[1] Arnold, K., Gosling, J.: *The Java Programming Language*. Reading et. al.: Addison-Wesley, 1996.

[2] Baird-Smith, A.: Jigsaw Overview. Available at http://www.w3.org/pub/WWW/Jigsaw/.

[3] Duan, N.N.: Distributed Database Access in a Corporate Environment Using Java. *Computer Networks and ISDN Systems* **28(7-11)**, 1149–1156, 1996.

[4] Gaffney, J.: Illustra's Web DataBlade Module. *SIGMOD Record* **25(1)**, 105–112, 1996.

[5] Goldberg, M.W., Salari, S., Swoboda, P.: World Wide Web - Course Tool: An Environment for Building WWW-Based Courses. *Computer Networks and ISDN Systems* **28(7-11)**, 1219–1231, 1996.

[6] Gosling, J., et. al.: Jeeves. Available at http://java.sun.com/products/jeeves/index.html.

[7] Hauck, F.J.: Supporting Hierarchical Guided Tours in the World Wide Web. *Computer Networks and ISDN Systems* **28(7-11)**, 1233–1242, 1996.

[8] Hubler, A.W., Assad, A.M.: CyberProf: An Intelligent Human-Computer Interface for Asynchronous Wide Area Training. *World Wide Web Journal* **1(1)**, 231–238, 1995.

[9] Hunter, A., Ferguson, R.I., Hedges, S.: SWOOP: An application generator for ORACLE/WWW systems. *World Wide Web Journal* **1(1)**, 185–194, 1995.

[10] Hamilton, G., Cattell, R.: JDBC: A Java SQL API. Available at http://splash.javasoft.com/jdbc/.

[11] Jones, K.L.: nif-T-nav: A Hierarchical Navigator for WWW Pages. *Computer Networks and ISDN Systems* **28(7-11)**, 1345–1354, 1996.

[12] Lai, M.C., Chen, B.H., Yuan, S.M.: Toward a New Educational Environment. *World Wide Web Journal* **1(1)**, 221–230, 1995.

[13] Maurer, H.: *HyperWave: The Next Generation Web Solution*. Reading et. al.: Addison-Wesley, 1996.

[14] Nguyen, T., Srinivasan, V.: Accessing Relational Databases from the World Wide Web. *SIGMOD Record* **25(2)**, 529–540, 1996.

[15] Norman, D.A., Spohrer, J.C.: Learner-Centered Education. *Communications of the ACM.* **39(4)**, 24–49, 1996.

[16] Pitkow, J.E., Jones, R.K: Supporting the Web: A Distributed Hyperlink Database System. *Computer Networks and ISDN Systems* **28(7-11)**, 981–991, 1996.

[17] Schickler, M.A., Mazer M.S., Brooks, C.: Pan-Browser Support for Annotations and Other Meta-Information on the World Wide Web. *Computer Networks and ISDN Systems* **28(7-11)**, 1063–1074, 1996.

[18] Schroeder, U., Tritsch, B., Krjerriem-Jasnoch, A.: A Modular Training System for Education in the WWW Environment. *World Wide Web Journal* **1(1)**, 239–250, 1995.

[19] Tuthill, W.R.: Don't Get Caught in the Web: A Fieldguide to Searching the Net. *Proc. COMPCON'96*, Santa Clara, February 25-28, 1996, 77 – 83, Los Alamitos: IEEE Computer Society Press, 1996.