Thomas Frankewitsch, M.D., Ulrich Prokosch, Ph.D.
Department of Medical Informatics and Biomathematics
University of Muenster, Germany

ABSTRACT
Multimedia plays a major role in medicine. Databases containing images, movies or other types of multimedia objects are increasing in number, especially on the WWW. However, no good retrieval mechanism or search engine currently exists to efficiently track down such multimedia sources in the vast of information provided by the WWW. Secondly, the tools for searching databases are usually not adapted to the properties of images. HTML pages do not allow complex searches. Therefore establishing a more comfortable retrieval involves the use of a higher programming level like JAVA. With this platform independent language it is possible to create extensions to commonly used web browsers. These applets offer a graphical user interface for high level navigation.

We implemented a database using JAVA objects as the primary storage container which are then stored by a JAVA controlled ORACLE8 database. Navigation depends on a structured vocabulary enhanced by a semantic network. With this approach multimedia objects can be encapsulated within a logical module for quick data retrieval.

INTRODUCTION
In medicine multimedia objects (e.g. images, animated images, videos, sounds) play an important role not only in the diagnostic process, but also as teaching material for educating medical staff and students. On the diagnostic side PACS (Picture Archiving and Communication Systems) are established to store - usually radiological - image slices or image series.1 For a standardized communication between different manufacturers' PACS the American College of Radiology and the National Electrical Manufacturers Association (ACR-NEMA) released DICOM, the Digital Imaging and Communications in Medicine standard.2 DICOM is an interface definition which allows the transfer of pictures without loss of information between different systems. Specific DICOM Viewers support the manipulation of images to show different views and to rearrange an examination.

A DICOM store may be seen as a container which stores the radiological image as well as associated patient data and technical information concerning the data acquisition.

Although DICOM displays great advantages for the communication and online presentation of images especially for radiology, it is not very useful for storing more general types of multimedia objects, such as "normal" colored images like photographs, movies or sound-tracks. These multimedia data require additional and different information content.

On the other hand, numerous image databases aimed at educating medical students have been published on CD-ROM.4,5 Additionally the world wide web (WWW) has become more important to the publication of educational materials. The WWW-standards, like the Hypertext Mark-up Language (HTML) of the World Wide Web Consortium (W3C), are designed to be platform independent. Browser technology enables the integration of different information sources.

Therefore it is not surprising that around the world many image databases embedding medical subjects exist.6,8 The main dilemma, however, is to provide efficient retrieval mechanisms for those databases allowing to find and reuse images "in a timely, cost effective way".3 Although there are many so called search-engines or spiders, which do nothing but index HTML-pages, these do not really support efficient ways of image retrieval based on medical characteristics. The basic problem lies within the underlying algorithms of the common search engines: Some are programmed to analyze the header-sections of HTML-pages, some to read the whole page for an indexed storage of the textual contents and the last group is manually indexed by humans according to the information of the web-page-publishers.

Furthermore, finding good multimedia materials on the WWW is a twofold problem: Firstly, finding a certain database within the vast of information available on the world wide web and secondly, navigating through such a database to localize particular multimedia objects which might be useful for a particular

1091-8280/99/$5.00 © 1999 AMIA, Inc. 765
purpose. Up until now none of the systems we have found on the WWW provides a flexible navigation interface allowing efficient retrieval of interesting images from the database. The quality of an information retrieval system is determined in large part by the effectiveness of the selection and assignment of thesaurus terms to descriptions represented in the system in order that searchers may easily retrieve the required images. The image databases we have found on the world wide web mostly have a very narrow focus and were developed to support single medical specialities, e.g. ophthalmology, radiology, surgery or anatomy. Thus, the navigation and retrieval interface of those databases is very much adapted to the respective clinical speciality and offers no general speciality-independent access possibility. It is often necessary therefore to know the vocabulary of those specialities exist to find an image within their database.

Most of those databases are structured according to diagnoses (e.g. diseases) or methods of image acquisition. Whereas the method of image acquisition is pretty much standardized classification schemes based on medical diagnosis are usually not standardized at all. Every speciality and every institution have basically applied their own local classification terminology (sometimes at least mapped to ICD-9 or ICD-10).

Navigation on a higher, abstract level is usually not implemented. Most of the databases use pure HTML-navigation, i.e. links to other HTML-pages. Some use query interfaces where simple interactively formulated queries are processed by SQL-database-engines. Only within a few database active JAVA-supported navigation tools have been implemented. However, a platform-independent, highly object-oriented programming language, is supported by almost any browser. The program code, called applet, is executed within the environment of the browser on the local client computer. JAVA contains many features supporting efficient internet-applications. For example it implements socket-based communication, direct database access via JDBC, object serialization and compressed streaming of data and objects using the ZIP-compression format.

In this paper we investigate the suitability of a pure JAVA extension to a browser (applet) as a user interface for navigating a multimedia object database, in combination with a JAVA-storage model for multimedia objects.

**METHODS**

1. **Dictionary**

In contrast to most existing image databases, navigation and retrieval in our multimedia objects database are solely based on pure sequential lists of index terms (e.g. diagnosis). To establish a more comfortable "navigational" approach we have implemented a controlled vocabulary which in the current first release based on anatomical names. This is additionally structured within a semantic network to index the objects in the multimedia database. This semantic network structure provides the basis for establishing navigation structures through hierarchies of objects or finding semantically related objects from a completely different conceptual context (e.g. therapies associated with diseases).

2. **JAVA - Objects**

JAVA objects have been defined for the storage of any multimedia object added to the database. The actual images, videos, sounds, etc. are routed through a compressing class in order to save storage space. All classes encapsulating data are descendants of a common virtual basic JAVA object (ObjectWrapper). This root class defines a generic interface for data input and data access as well as basic private methods (e.g. getCreationDate, getAuthor) needed in all further classes.

Two branches are derived based on this root, one to store the actual multimedia objects (ObjectWrapperDataLive) and one to collect these JAVA objects in a database (ObjectWrapperDatabaseOracle). The hierarchy of these defined JAVA-objects is shown in figure 1.

![Hierarchy of JAVA-Objects](image_url)

Using the polymorphism of this object oriented programming language it is then possible to create finally the actual working instances of the respective JAVA objects.
In the first branch, additional information describing the multimedia objects is stored in three types of vectors (public information, private information and generated information). These vectors are themselves constructed by adding objects describing particular characteristics of the multimedia object (e.g. author’s name, creation date, access control). Those information objects are structured as variable length records in an attribute-value-structure: the internal information about the author, the type and the name are defined as JAVA-string-objects and the expression is a valid JAVA-object. This provides a very flexible data structure open to further expansion. For an example see figure 2.

On the other branch the ORACLE database descendant (ObjectWrapperDatabaseORACLE) of the wrapper objects has the function to finally store the objects in a relational database management system. In our application this is an ORACLE 8 database implemented on an UNIX-server. This encapsulation ensures that with changing the RDBMS environment only the ORACLE-specific "ObjectWrapperDatabaseORACLE" has to be changed.

Combining the basic rules of object oriented programming – encapsulating data, inheritance, polymorphism – it is easily possible to implement virtual objects responsible for storing multimedia data.

Since the above object classes only provide data storage management methods, they can not yet be executed directly, but have to be finally activated from an application environment (e.g. applet/servlet, see below)

On the administrator side a “factory” has been programmed, allowing the creation of the basic storage objects, the selection of images, and the input of user data.

On the end-user side two separate applications have been implemented: an applet for the user interface (started from any browser) and a servlet. The applet displays the dictionary's semantic network tree and allows navigating this tree. The servlet implements the database descendant of the wrappers and establishes the communication between the database and the web-front-end.

3. SQL and XML

The interfaces of the objects are grouped according to administrator functions and end-user functions. The administrator has to add, change and edit objects. The end-user can only view objects and their logical dependence. The basic interaction is shown in figure 3.

The administrator interface has the interface-methods "executeQuery" and "putObject" implemented. The end-user interface only allows the method "executeQuery".

The predefined interface "executeQuery" uses an elementary SQL-String for the parameter allowed; therefore a minimal SQL-parser was implemented.

The result of "executeQuery" is a pure JAVA object. Depending on the query, the result can be an array of bytes - the multimedia-object itself or a JAVA string. The mapping has to be done by the calling thread. If the answer is a string, then it is formatted in XML-style which provides both textual and structural information.

4. Applet-Servlet-Communication

Because of security restrictions, only the servlet is allowed to directly connect to the database. Applets can only communicate with the calling server. Furthermore, if the database access is performed by the applet all basic information about the servers address and access data would become public. Since it is easy to recompile applets, this would be an unacceptable lack of security.

The applet opens an http-connection to the servlet using the http-commands 'post' and 'get'. The interaction between server, browser and servlet is shown in figure 4. Steps 3, 4 and 6 are repeated according to the user’s selections in the graphical user interface. Step 5 is only necessary if the image is not stored within the database itself.
DISCUSSION

1. Dictionary

One of the basic requirements of successful navigation and object retrieval within a database is the vocabulary. It is not in the scope of this article to fully describe the semantic network design of our vocabulary (this will be published in a separate article).

In contrast to other WWW-image databases, such a semantic network approach allows the definition of associations between vocabulary terms. Thus, if a user is unable to find a particular entry, he can use the network structure to navigate through the causal relationships. For instance, navigating along the "is a" relationship allows to step through a hierarchy of objects: "tibia" "is a" "distal skeletal bone" and the "fibula" "is a" "distal skeletal bone". Going from "tibia" to "distal skeletal bone" it is easy to find "fibula" - without needing to know the expression in advance. In future more complex associations will be added, e.g. linking diseases to recommended therapies. Following such relationships may allow to find images associated with a particular therapeutic regimen which may be recommended for a specific disease category.

2. JAVA - Objects

With JAVA the code created can be used on different platforms. There is no difference whether the servlets are installed on UNIX-Server or NT-Server. Only the local runtime-environment must also be installed, the JAVA virtual machine.

Even though JAVA 2 currently is only supported by a few browsers, we are optimistic that by the time our system will have its full capabilities. JAVA 2 will be generally supported. Furthermore, even today, JAVA 2 can already be run within every browser, if a freely available plugin is installed.

While the DICOM serves special purpose in radiology images the use of pure JAVA-objects serves many advantages. Firstly, the data are stored within a blackbox. For the front-end users it is not primary of interest how the internal structure of data-storage is managed. The main and daily question is how to gain the information. If a program wants to get data, it has only "to ask" the object using the predefined interfaces. The data are stored in a "living" environment. Therefore the items of attributes describing images can easily be updated. The end-user does not need to know where the specific information is collected.

The next interesting point is that these objects are stored within an object of their own super-class. The name of the procedures for storing are the same, but the database-version simply uses an RDBM-System in the background. There seems to be the problem of dual data-version simply uses an RDBM-System in the background. There seems to be the problem of dual data-version simply uses an RDBM-System. The user does not even need to know, whether an actual multimedia object is directly stored within the JAVA-object itself or if it is even located on a completely different server and the object only contains a link to the relevant URL. This objective may be elementary because of copyrights on images which may not allow the duplication of images and importation of them into our own database, whereas linking to the relevant storage address is allowed. Thus, while one's own multimedia objects may be stored directly within the object, for "third party" objects only the descriptive information and a link address to the actual object may be stored within the JAVA-object. In this case the servlet functions as a proxy-server, which combines information from different sources within its own dictionary.

3. SQL in - XML out

Using a platform independent programming language enables a wide reuse of the source code. Therefore, the user-interface must be standardized. Although 'public interfaces' are the only way to retrieve the data, the interface should also be standardized.

The standard retrieval language in database technology is the structured query language (SQL). Thus, SQL was selected as the syntax for describing "retrieval questions" and communicating these to the database engine. For the reverse communication XML was chosen, since XML allows to describe structure and content within one message.

4. Applet - Servlet - Communication

The applet-servlet-communication guarantees user-controlled access to the media-objects because only
the servlet can unwrap the inner part of the information, load the image and serve it back the calling browser-applet communication partner. Furthermore, servlets can generate and instantiate JAVA classes of data models (e.g. lists, trees) which can be streamed over the network – invisible for the browser - to the applet, which can present these models using its own panel.

CONCLUSION

Apart from textual information the multimedia part of the WWW will expand further within the next few years. This will be powered by the increasing speed of data-transfer and by the production of faster central process units. While in most existing image databases, basic browser technology is commonly used, a flexible graphic user interface provides more comfort for navigating in a semantic network structure. JAVA programs (applets) can do this on different browsers and different operating systems.

The meta-information of multimedia objects (e.g. title, comments or key-words), as well as the multimedia objects or their link-addresses are well stored within JAVA-objects, which are kept in an ORACLE database. Database access is enabled and controlled by a server-side JAVA program (servlet) which provides the high flexibility of this programming language.

Because of the full JAVA realization platform independence is achieved.

Furthermore with the current release of JAVA 2 it is possible to implement tools to analyze the inner objects and create new meta-information beyond human scope.

On the other side there is the possibility of using the JAVA 2 capabilities of a corba interface to offer information on multimedia objects and the objects themselves with this new standard.

References

12. WebPath: The Internet Pathology Laboratory, Department of Pathology, University of Utah, Salt Lake City. Available: http://www-medlib.med.utah.edu/WebPath/webpath.html (03/09/1999)