"THE CYPRUS TREASURE" A web based Multimedia Application for Archeological Treasures

Chrystalla Alexandrou [*], *Achilleas Kentonis* [*],
Andreas Andreou [*], *George Samaras* [*]
and Christos N. Schizas [*]

[*] Multimedia Research and Development Lab Computer Science Department,
University of Cyprus, CYPRUS
E-mail: cschryst, kentonis, aandreou,cssamara, schizas @ ucy.ac.cy}

ABSTRACT
Archaeologist and historians have now the ability not only to use computers and software in order to document and publish their work, but also to access it intelligently and remotely. Recent technological advances in Multimedia and WWW technology made possible the building of network centric applications that can serve their needs. Cultural heritage can be analyzed and expressed in an information rich environment and it can be delivered not only to the scientific community but also via Internet to the general public. This paper proposes a network centric application based on Java applets that via a "parallel" user interface design serves the application requirements in an innovative way.

KEYWORDS: Multimedia, network centric, Java applets, object oriented database systems.

INTRODUCTION

The program "THE CYPRUS TREASURE - a Corpus of Historic Monuments of Cyprus", is a joined project of the Department of History and Archaeology and the Department of Computer Science at the University of Cyprus, which combines historical, archaeological and information systems research and focuses on Cyprus archaeological treasures. The chronological frame is the period from the "Cypro-Classical" era (conventional beginning is the year 475 BC) through 1192 AD, the year which marks the end of the Byzantine epoch and the beginning of the Frankish rule. Among the participants of the project are universities and research centers from Cyprus and abroad.

The objectives of the program are:

1. The creation of a systematic electronic catalogue and the analysis of historical data and antiquities (e.g. Epigraphic documents, numismatic electronic catalogue, etc), those constitute the historical map of Cyprus during the aforementioned periods.
2. The development of "Thesaurus", a multimedia network-centric application for the presentation of a detailed catalogue of all the architectural monuments and antiquities (epigraphic documents, ceramics, coins, portable icons, scripts, travelers notes and diaries,

mosaics, frescos, architecture and sculpture) within their geographical and topographical frame.

3. The selection of all relative excerpts of the Greek, Latin, Syriac, Armenian, Arabic and Hebrew narrative sources and the creation of an interface of their data.

The "Thesaurus" project aims at developing a platform independent WEB application that supports the storage, manipulation and presentation of the archaeological treasures of Cyprus. Recent technological advances in Multimedia and WWW technology made possible the development of such applications that can be available on the Internet not only to the scientific community but also to the general public.

The vision of a platform independent language became a reality with the creation of the Java platform, that is a fundamentally new way of computing, based on the power of networks and the idea that the same software should run on many different kinds of computers. That has a tremendous impact on related technologies. Among these innovative technologies Java applets provide the technical foundation for platform independent applications provided by Internet.

BACKGROUND

Java Applets

Java applets [**i**] are Java programs that can be included in an HTML page, much in the same way an image is included. When an Internet user uses a Java-compatible browser to view a page that contains a Java applet, the applet's code is transferred to his/hers system and executed by the browser. That means that the Java platform provides the ability to download behavior from the server to a client through the network. Consequently the clients may gain intelligence they did not already have without complicated installation procedures. Also build-in facilities like Remote Method Invocation (RMI) [**ii**] provide easy access to objects (chunks of application data and functionality) to resign on a remote computer without ever being aware that they are remote objects. This results to the selection of the best object for each job that resigns on the best platform on which that object runs. Consequently this provides flexibility that is essential in the development of distributed systems. Java applets can contain Java Beans, the Java components that can be manipulated visually and provide build-in application functionality without any effort to produce it.

The applets created the **Network Centric Computing** [**iii**] instead of desktop computing. Users connected to the network with machines that run a Java enable browser can connect via a URL to server and run a servers application without pre-installing anything.

NETWORK CENTRIC COMPUTING

Java's unique capabilities, working with other Internet-based technologies, make possible the creation of distributed object-oriented applications that exist and function independently of any particular desktop architecture. Consequently, Java offers a development environment for an inherently more flexible and dynamic computing model.

Most common client/server applications today are based on the desktop-centric model of computing. In other words, applications are designed with specific client architecture. These desktop-centric applications must be preinstalled onto client PCs. To make any changes, new version or upgrades on a client-by-

client basis must be installed. The World Wide Web (WWW) and the Internet provide the basis for a technology that changes all that. Platform-independent tools and standards such as HTML and TCP/IP and java let Web developers to build a single "application" without worrying about operating systems or hardware platforms. At the same time, users do not need to be concerned with the type of server being accessed.

Together with its object-oriented nature, Java's dynamic download capability also means that when new data come online, corresponding Java code is also deployed. That code knows where to find the data, and how to interact with it. If the information moves or its format changes, the Java code is simply updated. The bundling of data and associated Java programs into objects is what makes network-centric computing possible.

Software development organizations create and maintain different versions of software for different platforms, leading to a tremendous development and deployment cost. Java applets install themselves just in time, on the fly, and de-install themselves when they are no longer needed. Hence, the concept of upgrading application software is eliminated. This is the important advantage that the network-centric Java applets have over desktop-centric applications. Java based network-centric applications, allow organizations to build and customize business applications that reach a wider audience much more quickly and with fewer support issues.

Since Java is an object-oriented language it provides all the build-in benefits of the object-oriented languages like decoupling abilities, build-in modularity, encapsulation and information hiding and polymorphism.

It is true that network-centric computing due to its distributed nature is also more complex because it must locate and then successfully transmit and launch several applets across several different servers and nodes across the network. This added complexity, however, does not diminish Java's ability to provide the traditional advantages of client/server systems; it merely enhances the application developer's ability to distribute processing across a network.

SYSTEM REQUIREMENTS

Within the information systems context the main goal of "Thesaurus" is the development of a network centric application that supports the storage, manipulation and presentation of archaeological treasures. The system in order to provide the flexibility and advanced functionality that the project requires the following functional requirements have been specified:

Cross-platform availability: "Thesaurus" as a network-centric application should operate on any computing platform. The same applied on the content that should be presentable on any kind of platform. To accomplish this, Java was an obvious tool since it can be deployed on most computing platforms without the expense of building and maintaining separate versions for each platform.

Extensibility: The collection and cataloguing of all the archaeological treasures of Cyprus is a huge project. The application should be extensible in order to support schema evolution of the system. This affects the whole application as well as the database design. The selection of the applet-based system allows the creation of thin clients that they do not require any pre-installation and consequently no re-installation, due to the system evolution. That means that a change on the content

repository that is based in Cyprus will automatically appeared on a users machine in Canada the next moment that the user will be connected to "Thesaurus" side. Also the repository of the content is huge and even with today's media, as DVD and CD ROMS the frequent versioning will have a tremendous cost.

Easy to use: There is a need for intuitive user interface (UI) that makes it easy for anyone to use the application. The target group of "Thesaurus" is archaeologist, researchers, students as well as general public that may have only limited computing experience. Searching a huge database for specific information should not demand extensive abilities in Boolean algebra.

Modularity: The application has to be modular so users can add functionality, as they need it. There is no need for the user to have a thick client with functionality that is rarely used. Modular architecture allows the complete application to exist only on the server and specific features to be downloaded to users only as needed.

Compactness: Each application module has to be designed subject to current network characteristics (low bandwidth, absence of good compression algorithms etc). It has to be compact in order to enable the quick delivery over the network.

User Focused: Categorization of the application features and functionality in different levels of user utilization. This will facilitate the compact design. The main goal of the system is to enhance the user with functionality that is impossible to have by using conventional methods and tools. Cross-referencing using hypertext links, powerful search engines and search reuse will provide enhanced functionality.

Search reuse: Many searchers duplicate previous searchers yet the information is lost and new searchers have to re-invent the wheel. The application should support the reuse of searches in order to take advantage of other searchers effort to retrieve a specific piece of information.

Richness in context: After all is an information repository. The richness of archeological context requires the extensive use of multimedia technology.

The nature of "Thesaurus" data characteristics (high volume, data evolution, static and dynamic expressions, timed media etc) and the multilevel nature of the user target group (researchers, students, public etc), justify the decision of a network-centric application with the following additional characteristics:

(a) Multimodal user interface design that will enable a simple and easy to use UI in the form of a visual search engine with extensive use of multimedia.

(b) The application must cope with the vast amount of related information and the disadvantages that WWW exhibits today (e.g. low bandwidths, absence of good compression algorithms, etc.). The deployment of a multimedia aware repository will allow flexibility and additional features regarding the access to multimedia data and consequently the access time of this data.

(c) An Object Oriented implementation of the data repository in the form of OODBMS that will allow flexibility in the applet based design. The application is object oriented and the use of an Object Oriented Database allows the creation of a Thin-Client application. In this kind of applications each object is

responsible for keeping a record of its state and functionality in the database.

THE APPROACH

The development team of this project is composed of two sub-teams: (a) the archaeologist and historians team that is carrying out research on cataloguing of all the archaeological treasures and (b) the information systems team that is responsible for the development of the system. Both sub-teams meet under JAD sessions where they plan the different phases update each another of the progress of the currently executed phase and also discuss and decide their interface.

Most of the electronic repositories for archaeological treasures use a free text description, which in the best case contain hyperlinks and pictures of each antiquity. Some other systems are built using the relational database approach without any support to multimedia data that are stored separately in other sources like videodisks etc. Both approaches have major disadvantages: the former one is rather limited in manipulating the information stored. Free text searching is very time consuming and not effective. On the other hand relational databases are unable to represent complex relationships, the need for polymorphism in data retrieval and the schema evolution that characterize the system.

An object oriented approach will be more suitable **iv** since archaeological systems require wide range of types because of the significance in the details of the data representation that the object orientation can serve better via inheritance, composition and polymorphism. Also multimedia object oriented systems can store under the same system scheme different media types leading to a more accurate and specific application. That's why an important task of this project was the selection of a database system that supports the system requirements.

SELECTION OF A DATABASE SYSTEM

There are many different kinds of database systems available, from pure relational database systems to pure object oriented database systems. Relational database systems revolutionized database management systems in 1980s, and object oriented programming languages are revolutionizing software development in 1990s. These systems seem to have complementary strengths. Relational database systems are good for managing large amounts of data; object oriented programming languages are suitable for expressing complex relationships among elements that combine code and data (objects). Relational databases serve well data retrieval but perform poorly in data manipulation; on the other hand OO programming languages are excellent at data manipulation but they provide very little or no support of data persistence and retrieval.

The obvious solution should have been a hybrid system combining relational and OO in order to manage large amounts of data with complex relations. Unfortunately, the two models are fundamentally different and integrating the two is not an easy job. Relational database systems are based on two-dimensional tables in which each item appears as a row. Relationships among data are expressed by comparing the values stored in these tables. The object model is based on the tight connection among code and data, flexible data structures, hierarchical relationships among data types and references. The simulation of all these using two-

dimensional tables of the relational database is not trivial.

Relational databases and object oriented programming languages have fundamentally different data models and they demand two different data models for each application. The creation of the class hierarchy, since there is no build in feature for handling it, requires the decision of how each class should be stored in the database. Because the relational model does not support the most basic aspects of the object model, the interface to database is complex. A complex interface is bad enough, but when this interface has to be reflected in every single object then it becomes much more difficult to maintain. The reason is simple: the relational model was never developed to store objects and objects were not designed to be stored in two-dimensional tables.

The **Object Oriented Databases** provide support to all major characteristics of the object model:

Encapsulation: An object can have private and protected parts. An object read from the database has the same state as the one it had when it was stored. This will serve the need for persistence from the usage of Java.

Inheritance: Classes, which are derived from other classes, can be stored with one operation. The database system is aware of the class hierarchy and acts accordingly. The usage of hierarchy in the antiquities gives a lot of flexibility in the system design. Inheritance decreases structure duplication and minimizes relationships. For example, an outdoor temple is a temple with additional characteristics and behavior. In the design the class Outdoor temple is a descendant of the class temple.

Polymorphism: Objects can be read from the database without knowing the complete data type. For example storing an outdoor temple can be retrieved when a query for all temples is running.

Object Identity: There is no need for the programmer to maintain the relationship between database objects and objects in memory.

References among objects: Automatic resolution of pointer references and their representation in the database.

The Archaeologist sub-team in their cataloguing process has decided to create categories of antiquities and to define a structure for each category that called "passport structure". By defining the passport structure there is an implicit and directly available way for locating this piece of information. It can be seen as the coordinates of each element in the information space. For example the coin's passport structure is:

Coins = {Coin Manufacturer, Coin material, Issued by, Coin Subdivision, Collection, Chronological period, Manufactured date}

In the case of the data model of the "Thesaurus" there are complex relationships between the different passport types, long hierarchies and need of polymorphism support. The Java applet model that supports the functional layer is fully object oriented without persistence support. Java as a programming language does not provide query facilities and does not support transaction semantics. Java objects are not persistent and vanish upon the termination of the program that creates them. Java alone, therefore, is not sufficient to build our multimedia application, which deal with large volumes of complex data. That is why the selection of a multimedia object oriented database system simplified the whole development.

Additional to complexity, performance is another key issue for the selection of multimedia object oriented database system. Mapping objects to and from a relational format by joining tables imposes significant performance penalty. As the object model increases in complexity, the number of tables needed to represent it increases geometrically. More tables' means more joins and relational system performance quickly degrades. On the other hand, object databases have no such performance limitations in dealing with complex objects. They also have much greater performance due to their ability to smart-cache. Smart caching eliminates server communication overhead if the object is already in the cache.

Looking for the appropriate database systems we have evaluated different object oriented database systems. We have decided to adopt Computer Associates Jasmine since it is a robust multimedia object-oriented database management system with build-in Web connectivity. The integration of Jasmine and Java provides a platform capable of meeting "Thesaurus" requirements of Web-enabled multimedia and multi-platform application.

6. THE ARCHITECTURE

A client machine connected to Internet via a browser can access the "Thesaurus" application by connecting to the Web server. The Web server accommodates the server part of the application, which is connected with one or many database servers.

CA Jasmine provides different models for system architecture. Two, three or multi-layer architecture is possible to be implemented. And the bigger advantage is that the application can be written in such a way that the decision between the two-tier and three-tier architecture can be delayed until runtime by the use of different constructors.

The client server communication can be achieved either by using the client–server communication protocol of Jasmine or the Remote Method Invocation (RMI) of Java.

"Thesaurus" implements a three-tier architecture (figure 1). The requirement for thin client determines the three-layer architecture.

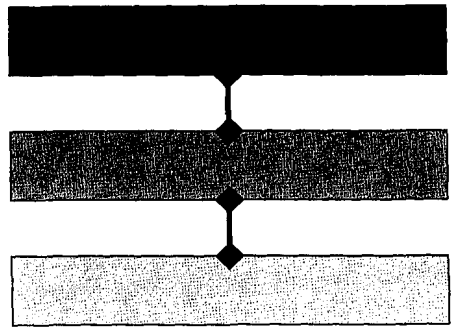

Figure 1

box 1. Client Layer
box 2. Application Layer
box 3. Database Layer

USER INTERFACE DESIGN

This application is a part of a more general attempt to create user centric systems with the use of multimedia and WWW. The basic idea is to avoid the windows button-oriented interface and the serial way of thinking for the creation of queries. This is achieved by using a visual representation of a "parallel" way of thinking. Statistics have shown that even advance users are unable to construct complicate queries leading to a "serial" way of searching by defining one attribute each time. In the case of extremely high volume repositories this is time consuming. The UI goal is to facilitate the user to

construct multilevel queries (parallel queries) with a visual approach of the user interface. Keeping the user's intent and the machine's interpretation the same, is probably the most challenging task in this visual design [**v**]. This parallel approach will enable the users to search in depth and width as well.

A parallel Navigation

The user interface of any system is the pivoting locus of the whole environment. It comes before any graphic design. It is to secure a non-stop communication between computer and human environment (and what ever comes with it) [**vi**]. Thus, its design must be simple, effective and based on the concept of human interaction.

Nowadays, there is an overflow of information that leads to information futility. Thus, the only choice left is to move into a parallel and simultaneous (if possible) way of data mining. In order to offer these parallel ways we have to display them simultaneously on one screen without posing any confusion or acting against the legibility of the information on the screen. The human eye is trained to identify high volume of information provided simultaneously. Taking advantage of this ability and also the recent technological advances in multimedia systems, by defining the screen as our visual space, we can create a UI that with just one screen will be able to control a bigger portion of information space.

The design aims to a parallel representation of data mining of structured information by the use of different metaphors that will appear on the screen based on user interaction. The goal is to break the linear or serial thinking in a visual way that would be well and clearly received by the user so he or she will interact with the specific system in a parallel way.

If all the elements were having the same passport structure then the information space would have been defined by a k-dimensional space where k is the number of attributes in the passport structure. Due to the fact that each category has different passport structure we have to define a superset of all the passport structures with n attributes. Each attribute of this super-passport-structure defines a dimension of the information space. The problem is to find a simple way to represent visually the n-dimensional information space. In our case this n was very big and that added a lot of complexity.

The number of axes theoretically can be infinite and reach any infinite-dimensional information space. It is proven that visually any number of axes, which is greater than five [**vii**] will distort the conception of parallelism of the user. That was the reason why we have decided to move to an opposite approach; from maximum to minimum, from superset to subset. There were tree major attributes in all the passport structures(figure2): a) **Time** (periods- like Hellenistic), b) **Kind** (sculpture, architecture, coins etc) and c) **Place** (different toponyms of Cyprus). The minimization of the attributes had as a result the simplicity of the representation problem since a three dimensional space is easily understood and represented. The above three attributes are considered as the main *information axes.*

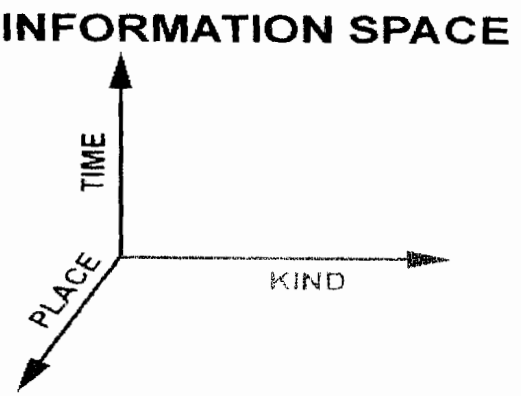

Figure 2

The UI design should provide a parallel way to traverse these three axes in order to give the opportunity to the user to specify a value or range on these axes for locating the needed piece of information. These three axes presented in UI will help to bound the information space in smaller spaces where the search engines and other constructions can easily handle. In our design the three axes are displayed at the same time on the main window of the application (figure 3) and upon a user request the 3-dimensional system will expand to k_1 axes by the use of popup windows (figure 4). This facilitates traversal in information depth without moving to a separate screen.

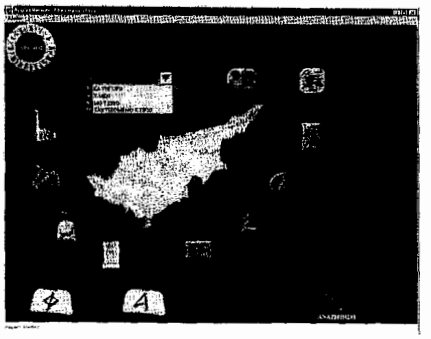

Figure 3

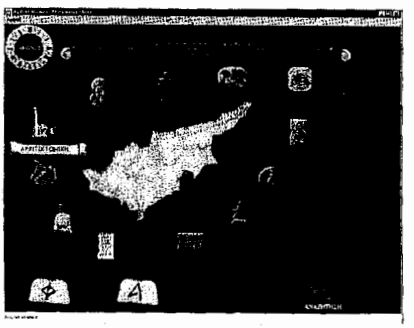

Figure 4

The main idea is to provide user navigation in width and depth, in parallel though the same screen.
The way metaphors are placed determines the thinking mode the user will be in, when he or she will face the screen. In the center of the screen there is a map of the island. This is used in order to navigate in the "place" axis. Since it is impossible to display all the toponyms in such a small map, right clicking in the area around the place the user wants to choose, a popup window will appear displaying a zoomed image of the area. The user can right click in order to zoom further in the same popup window. Left mouse click always selects or deselects a place. Clicking anywhere outside the popup window will cause the flying away of the popup window. User selection will cause a change in the color of the map in the selected area. As the number of selected places increase the display color darkens. Around the map we have the axis "Kind", where the major "kind" categories are displayed. By right clicking on any of the category icons, popup menus appear with subtypes. Selection will have again a visual effect on the screen: a frame will appear around the initial icon. On the right top corner the time axis is displayed. Right clicking on it will display a time line where the user can zoom on it or he/she can select an area. Again a frame around the time icon will change color depending on the size of the user selection. (The color becomes darker when the range grows).
When the user has completed his/her selection then he/she has to press the "search" button on the right of the bottom line of the screen. The system constructs a query based on user selections and the results are displayed in a form of a list in a new application screen. Selection of an item on the list will cause the displaying of relevant information on another screen (figure 5).

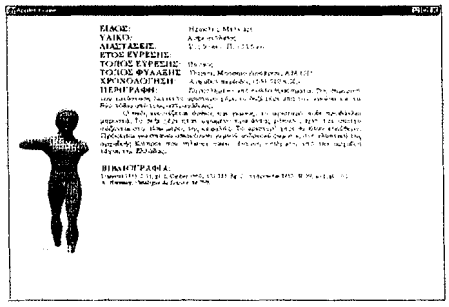

Figure 5

The goal of the application is to give the researcher one tool, which would be simple, rich in information and effective for his/hers research. At the same time the researcher could lay back and print or export or save the path he followed in order to get to this information, save anything that is with in his/hers interest with just a mouse or a touch screen.
Similarly the time axe that is displayed on the top of the screen can be expanded into the different chronological periods (figure 4) and the user by clicking can select for his search one or multiple periods.
Another important issue that is cover in the UI design is the ability to save and recall a query. The moment that the user has specified with the visual interface his search criteria by selecting the kind/time/space in any depth (from all to nothing) automatically the system builds a query that is displayed at the bottom of the screen. This will help the frequent users to modify or reform a query without going through the visual forming of query. Any changes in the query sentence will take affect on the visual representation automatically. At any moment the user has the ability to save the query on his disk or on the

server (depending on his willingness to share his query) by clicking the appropriate button. On the same way the user can recall a query he/she has previously save, or a query stored on the server, and even modify it according to his/her needs.

INVENTORY SUPPORT APPLICATION

A web application has been developed for the inventory support. Archaeologists, according to their authorization access level, are able to input new content and modify existing content remotely. The support application follow the same architecture gives the ability to archaeologists to input text, maps, video, or any other multimedia data. Using the web the archaeologists can update the database even from the excavations using his/her mobile computer.

FURTHER WORK.

Our team is working on the integration of a GIS (Geographic Information System) that will be integrated in the system. For now the GIS system is working as a separated module. Starting from the Cyprus map the user can click and proceed in depth into the space information and even see for example the mosaics in an archaeological space. For the moment two very big areas have been inserted into the GIS system with great detail.

The next step is to integrate the GIS system into the "Thesaurus" system. Applet architecture enable the replacement of the map of Cyprus of the main screen with the GIS system. The space that is cover today with the map of Cyprus will be taken be the GIS applet that will display the GIS map of Cyprus. When the user click on GIS space the GIS application will take control and by sending messages using RMI to the "Thesaurus" application the query will be formed as before.

CONCLUSIONS

The WWW, the Java platform and the recent advances in multimedia technology enable the building of a new generation of applications in areas where the media plays an important role. Presentation and cataloguing of the cultural heritage is an important obligation that we owe to ourselves and also to the ones that they will come After all, culture is for sharing and the better you present it, the more people will share it.

"Thesaurus" is a project for cataloguing and presenting a part of the cultural heritage of Cyprus. The use of the Java platform aims to a network centric application that the thin client part will be delivered dynamically with the use of Java applets. The Object Oriented database system offers the ideal database system to support the object paradigm of the application part, in a more effective way conclusively because it is built to maintain objects in an object oriented way.

The UI design is based on the abstraction of the attributes (characteristics) of the information in a three dimensional system, the visual representation of which provides the backbone for the UI. UI is designed with the same philosophy as the whole application. It provides context and functionality "on demand". The initial screen is very simple, it displays only the three basic axes: time, place and kind. User interaction has as a result the appearance of additional information axes that help the user to locate the information that is interested more efficiently.

ACKNOWLEDGEMENTS

The authors would like to thank the whole team of the "Thesaurus" project for their participation in forming this mental framework and their insight knowledge about the end users needs and expectations. Special thanks to the project coordinator Evangelos Chrisos, professor of History and his assistant Dr. Stathis Raptou for their contribution . Last but not least the authors like to thank the undergraduate students of the department of computer Science of the University of Cyprus Elias Nisiotis, Andreas Palekis and Stella Nisiotou for their significant help during the development of the prototype system.

REFERENCES

1. Ed Anuff. "Java Sourcebook", Wiley Computer Publishing, New York, 1996

2. T. b. Downing, "RMI: Developing Distributed Java applications with Remote Method Invocation and Object Serialization", IDG Books Worldwide, California, 1998.

3. S. Shirmohammadi, J.C. de Oliveira and N. Georganas, " Applet-Based Multimedia Telecollaboration: A Network-Centric Approach", IEEE Multimedia, p 64, April -June 1998,

4. C. Alexandrou, G Papadopoullos, "A multimedia system for archaeological scenes", Proceedings of ED-MEDIA, p. 734, Graz Austria, 1995.

5. Crimi, C., A. Guercio, G. Pacini, G. Tortora, and M. Tucci, "Automating Visual Language Generation," IEEE Transactions on Software Engineering, vol. 16, no. 10, pp. 1122-1135, October 1990.

6. Pounti K. "Disturbance in Human Computer Interaction", p5, January 1995

7. Achilleas Kentonis "Screen Topography", Multimedia Solution International magazine, May 1996