Visualization in a Mobile WWW Environment

Alberto B. Raposo
Dept. of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (UNICAMP) - Campinas, SP, Brazil
E-mail: alberto@dca.fee.unicamp.br

Luc Neumann
Dept. Mobile Information Visualization
Computer Graphics Center (ZGDV) - Darmstadt, Germany
E-mail: neumann@zgdv.de

Léo P. Magalhães[1, 2]
[1] Computer Science Dept. - Computer Graphics Lab.
University of Waterloo - Waterloo, Ontario, Canada
[2] DCA - FEEC - UNICAMP - Brazil
E-mails: lpini@cgl.uwaterloo.ca, leopini@dca.fee.unicamp.br

Ivan L. M. Ricarte
DCA - FEEC - UNICAMP - Brazil
E-mail: ricarte@dca.fee.unicamp.br

Abstract: The facility of access to information in the World-Wide Web (WWW), the expanding availability of information technology, and the recent developments in the handling of multimedia data are all important steps towards a Global Information Infrastructure (GII) accessible to anyone, anywhere in the world. However, in order to achieve this accessible infrastructure, one should consider the aspects related to efficient communication. These aspects are addressed by this paper. A mobile WWW rendering application using VRML 2.0 (Virtual Reality Modeling Language) is introduced, the related problems are pinpointed and approaches to overcome them are proposed. We have developed an application that filters VRML scenes to render only parts selected by the user. It improves the interactive visualization within a mobile environment and is a further small step towards the GII.

Introduction

The emergence of the WWW expanded the global information space, integrating services offered by the Internet and other networks. However, the number of users is increasing rapidly and the information they need is becoming more complex, requiring new ways of communication and interaction. The challenge is to reach the Global Information Infrastructure (GII) [Gershon et al. 96], a step beyond the WWW with more efficient interaction and globally accessible.

The technology of mobile communications is becoming increasingly integrated into the WWW. Due to this rapidly expanding technology, mobile users with portable computers are capable of accessing information anywhere and at anytime.

In general, a mobile WWW application uses mobile devices such as Personal Digital Assistants (PDAs) or notebooks to access remote WWW services. The required connection to the stationary WWW server can be established with communication facilities provided by the mobile device itself or with a cellular phone. The scenario is illustrated in [Fig. 1].

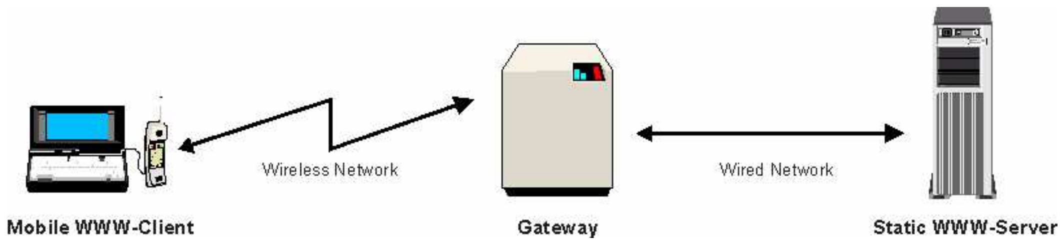


Figure 1: Mobile Application Scenario

This new communication style significantly affects the requirements of WWW applications and faces new challenges [Forman & Zahorjan 94], [Satyanarayanan 96]. One challenge is the design and implementation of highly interactive applications handling non-textual bulky information.
The main problems in handling such applications in a mobile environment are the resource constraints of the mobile data terminal and the narrow bandwidth of wireless wide-area networks. Thus, sophisticated concepts for the management and optimal utilization of resources within a mobile system are needed.
In this paper we address the remote access of VRML 2.0 worlds [VRML 97] from a mobile client, typically a laptop-like environment. In Section [Challenges of Mobile Computing] we present the peculiarities of mobile computing and its challenges. Our solutions are introduced in [Application Scenario and Solutions]. In [Conclusions and Future Work] we come to the conclusions and propose future improvements on this work.

Challenges of Mobile Computing

A mobile environment is characterized by at least the following properties [Neumann et al. 96]:

- Limited resources of the mobile devices in terms of storage, battery power, memory and processing power relative to non-portable devices.
- Low bandwidth, low reliability and high costs of wireless narrowband wide-area networks (WANs). Typical bandwidths of currently available cellular systems are 9.6 Kbps (GSM - Global System for Mobile Communication) or 19.2 Kbps (CDPD - Cellular Digital Packet Data), insufficient for multimedia applications.
- Imbalance of resource availability between the mobile device and the stationary servers.
- Users have no fixed location and can move during a connection.

The introduction of interactive applications - such as a WWW application - used within a mobile environment will only be successful when the requirements arising from the handling of non-textual bulky data (e.g., graphics, animation) can be fulfilled.
A simple solution might be to apply all the well-known mechanisms for a distributed application in a wired network also for a mobile application. However, they are designed for a higher bandwidth and richer resources at the end device. It will work, but the requirements in terms of throughput, delay, jitter, or response time will not be fulfilled. Therefore, it is a challenge to develop appropriate techniques such as compression, progressive refinement, and previewing, for a mobile environment to solve these problems.
In general, solutions address two main aspects related to the specific properties or limitations of a mobile environment.

- The transferred amount of data has to be as small as possible. This requires that parts of the application data be processed and stored on the client side.
- The use of local resources such as processing power and storage space should be the least possible. The client provides a presentation front end and communicates frequently with the server that processes computing intensive parts of the application.

It can be easily seen that solutions for one aspect are counterproductive for the other aspect. A solution providing an appropriate trade-off between both aspects would overcome the most serious problems - namely the narrow bandwidth and limited resources - of a mobile environment.

Application Scenario and Solutions

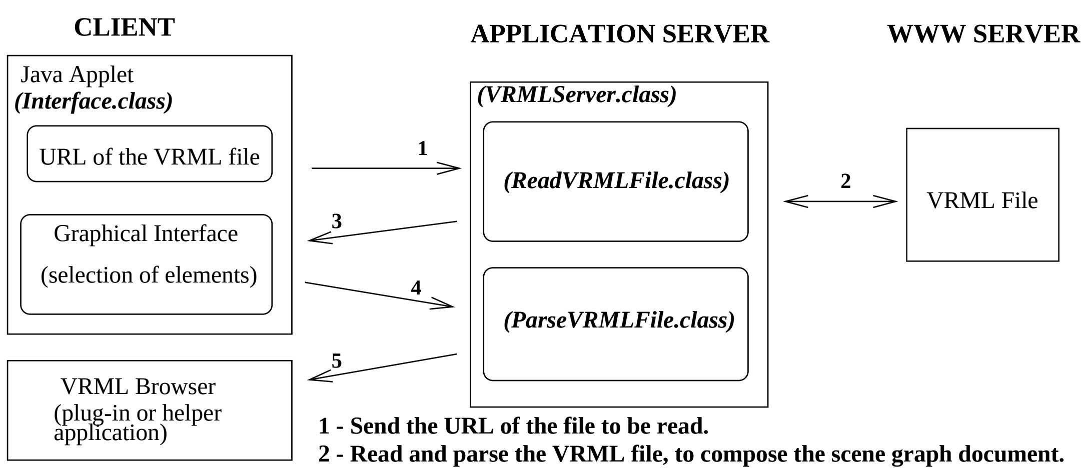
We focus on a rendering scenario where mobile users connected to the Internet via a wireless communication channel request for the visualization of a certain VRML world. The description of the world is stored somewhere on an information server. This description will be retrieved, rendered, and presented on a mobile client. However, the narrow bandwidth and the restricted processing power of the mobile data terminal require intelligent strategies to enable an interactive handling of the rendered scenes. Simply stated, the straightforward approach of retrieving the scene description and rendering it on a mobile client is not an adequate solution, especially if we think of complex scenes.

The solution we adopted in our application is a kind of data filtering technique. In a further improvement of the application, we propose the use of server resources in the rendering process, which will decrease the resource utilization in the client side. This enables a good balance between bandwidth and resource requirements in the mobile data terminal and ensures a good quality of service.

Reducing the Scene Complexity

Rendering is notably a process that requires a large amount of processing power. Due to the limited resources of mobile devices, techniques are necessary to simplify that process. One straightforward approach is to filter the data to be transmitted and to render only the parts of the scene that are actually of interest.

In order to demonstrate this strategy, we have developed an application capable of selecting the elements (i.e., geometric objects, light sources, and cameras) of a remote VRML world, which are going to be rendered and visualized in the mobile client [Raposo et al. 97]. This application runs in a mobile terminal (as a Java applet [Campione & Walrath 97], downloaded via an HTML page), which is connected to an application server. This server is capable of reading the VRML world (located in any WWW-server), parsing it and sending its hierarchical structure to the client. The client provides a user interface adapted to the hierarchical structure, enabling the users to select the elements of the world they want to see. The selected elements are then sent to the application server, which can parse the original VRML file and extract from it only the desired elements. This valid “sub-VRML” world is sent to the client, which can finally render the scene. This approach is illustrated in [Fig.2].



1 - Send the URL of the file to be read.
2 - Read and parse the VRML file, to compose the scene graph document.
3 - Send the scene graph document (with the elements’ hierarchy).
4 - After the selection, send the document with the desired elements of the scene.
5 - Send the "sub-VRML".

Figure 2: Strategy to visualize only a subset of a VRML 2.0 world.

The process begins by calling the application using any Java-enabled WWW browser. At this moment, a connection is established between the client and the application server. The next step is to send the URL of a VRML world to the application server (arrow 1 in [Fig. 2]). The application server will then connect with the

server of the VRML file, read and parse it (arrow 2 in [Fig. 2]). The first output of the application server is a small document (called *scene graph document*), representing the hierarchy of the elements of the VRML world, which will be sent to the client (arrow 3).
Based on the received document, the applet creates an interface, which allows the users to select the desired elements of the world. After this selection, a new document is sent to the application server, describing the selected elements (arrow 4 in [Fig. 2]). Using this new document, the application server can create a new VRML world from the original one, by extracting only the desired parts of it. This final sub-VRML world is then sent to the client (arrow 5), that visualizes the results using any VRML browser connected to the Web browser (as a plug-in or helper application).

Object Selection

Although only textual data are transmitted when the scene is completely rendered in the client, the narrow bandwidth of the wireless connection requires additional data reduction, especially for complex worlds. Our solution asks the user to build a sub-VRML world to be sent to the client. This sub-VRML file becomes smaller if the user selects fewer elements to be visualized. By presenting only the elements selected, which simplifies the rendering process, the utilization of the resources on the client side is reduced.
The application has also the interesting capability of mixing elements of different VRML worlds. In this way, the user can read several worlds, selecting and combining elements from each of them.
[Fig. 3] shows the interface of our application. In the top of the interface window there is a place for the URL of the VRML world and below, two buttons (*Read it*) allowing to read a new world. The first one (*add*) maintains the previously selected elements in the next visualization, while the second (*new*) removes all elements before loading the new world. The large text area in the middle contains the script used in the transmission between client and server. (This script is showed for demonstration purposes only - the user does not need to deal with it.) Below the script area there are buttons to choose the appropriate elements and to visualize the result of the selection. In the figure, the *Cameras* button has been pressed and the user can select among the several cameras of the world.

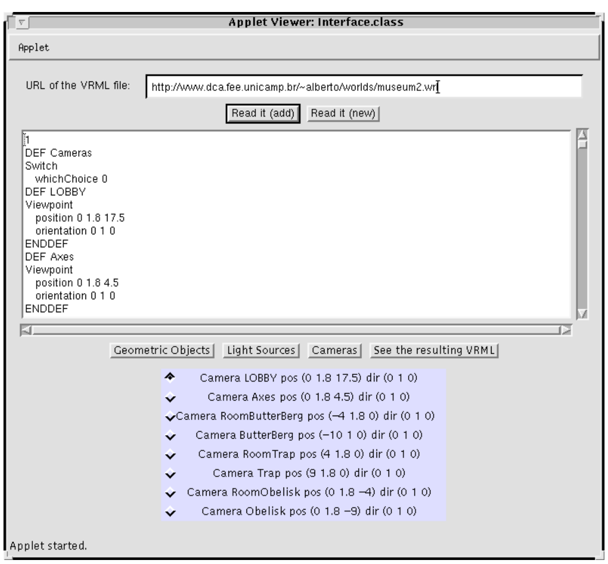


Figure 3: Interface of the developed application.

Task Distribution

In this section we present an approach to optimize the rendering process of the previous sections by the distribution of the rendering tasks among the available resources of the mobile client and the stationary servers.

This will improve the rendering process within the limits of the available resources. The main approach is to use the knowledge about the application semantic data and the environment resources to distribute the tasks.
The establishment of such an architecture is based on two main aspects: the introduction of a semantic content header for the scene description and the application of Java-based Object Request Brokers (ORBs) to integrate the distribution architecture into the WWW.
For the distribution of the different tasks to the most suitable resources we propose an architecture with a *Resource and Task Manager* (*ResTaMan*) [Neumann et al. 96] that distributes and controls the rendering process using application semantic information. A VRML-analyzer reads the semantic header and the world description. The semantic header can be regarded as metainformation about the scene supporting the identification of subtasks to be distributed. Here, the filter introduced in the previous section may be used to extract the relevant parts of a scene for the different renderers. In combination with an ODP-Trader [ODP Trading 97], which is a yellow page service knowing the properties of the environment resources, a good utilization of the rendering resources on the mobile client and in the fixed network can be achieved. Finally, an image composer and synchronizer is necessary to display the result.
Java ORBs are used for the integration of *ResTaMan* in the WWW environment. This enables the WWW Browser to gain access to arbitrary CORBA-based services. A Java applet is downloaded to the WWW client and serves as CORBA client accessing services using the Internet Inter-ORB Protocol (IIOP) [OMG 95]. In our approach it contacts the *ResTaMan* object that represents a metaserver from the viewpoint of the client. The entire rendering application consists of a set of interworking objects playing together via an ORB. For the transmission of images or image sequences we use a stream-oriented connection between the Applet and *ResTaMan* that may be established with an IP socket connection. Thus, for the bulk data transfer the stream connection is used and the control operations are transmitted via the IIOP. [Fig. 4] illustrates the integration of our architecture with the WWW.

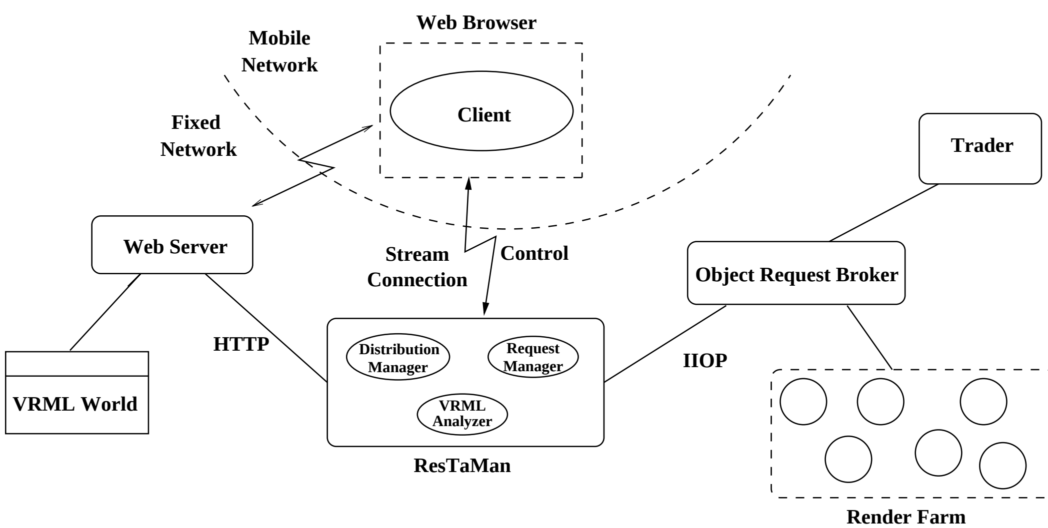


Figure 4: Integration of the Distributed Rendering Architecture into WWW.

Because VRML 2.0 also describes interactive worlds, we need a distributed event mechanism to send information about the occurrence of a specific event in the client to the appropriate remote object. CORBA introduced an *Event Notification Service* allowing objects to register their interest for specific events or to inform interested parties about the occurence of events. This service will be used in our environment to handle events and to inform the respective rendering object.

Conclusions and Future Work

There is a growing need for tools supporting the interactive access, manipulation, and visualization of distributed multimedia information to realize the vision of “all information at your fingertip”. One great

challenge is the improvement of the accessibility in the WWW using mobile devices. The main problems are the limited resources on the mobile device and the narrow bandwidth of the wireless link.
In this paper we presented solutions aiming to provide a good trade-off between bandwidth and resource requirements. We developed an application enabling the user to select elements of a VRML world and proposed an architecture for the adaptive distribution of tasks in a mobile environment. Although our solutions have been developed aiming a mobile environment, they can be used with any other type of WWW client (e.g., dialup).
The main advantage of our approach is the application of interactivity to achieve an efficient transmission. In order to filter the data or to reduce the complexity of the scene (efficient transmission), the user selection is needed (interactivity). In the current prototype, this interactivity still requires some knowledge of VRML by the users, but the goal is to use semantic information (defined by the author in the *semantic content header*) to achieve as much transparency as possible in the process. In this way, the users would be able to select objects by the role they played in the world (as defined by the author). Another possibility is to have the author assigning levels of priority to objects in the world, with a high level to objects which are essential to the scene and a low level of priority to details and textures, for example. In this case, the users would have only to set up to which level of priority they are willing to wait.

References

[Campione & Walrath 97] Campione, M., and Walrath, K. (1997). *The Java Tutorial - Object-Oriented Programming for the Internet*. The Java Series. Addison-Wesley. (http://java.sun.com/nav/read/Tutorial/index.html).

[Forman & Zahorjan 94] Forman, G.H., and Zahorjan, J. (1994). The Challenges of Mobile Computing. *IEEE Computer*, 27(4), 38-47.

[Gershon et al. 96] Gershon, N., Brown, J.R. et al. (1996). Computer Graphics and Visualization in the Global Information Infrastructure (special report). *IEEE Computer Graphics and Applications*, 16(2), 60-75.

[Neumann et al. 96] Neumann, L., Zhang, J., and Strack, R. (1996). Concepts for the Efficient Handling of Multimedia Data within a Mobile Environment. MOMID Deliverable 3, Computer Graphics Center (ZGDV).

[ODP Trading 97] *Information Technology - Open Distributed Processing - Trading Function - Part 1: Specification. Draft Rec. X.950-1 ISO/IEC DIS 13235-1*. (1997). (http://www.dstc.edu.au/AU/research_news/odp/trader/standards.html).

[OMG 95] Object Management Group (OMG). (1995). The Common Object Request Broker: Architecture and Specification; Revision 2.0. OMG TC Document.

[Raposo et al. 97] Raposo, A.B., Neumann, L., Magalhães, L.P., and Ricarte, I.L.M. (1997). Efficient Visualization in a Mobile WWW Environment. Technical Report DCA 001/97, DCA - FEEC - UNICAMP. (ftp://ftp.dca.fee.unicamp.br/pub/docs/techrep/1997/index.html).

[Satyanarayanan 96] Satyanarayanan, M. (1996). Fundamental Challenges in Mobile Computing. *15th ACM Symposium on Principles of Distributed Computing, 1996*, Philadelphia, PA. (http://www.cs.cmu.edu/afs/cs/project/coda/Web/docs-coda.html).

[VRML 97] *The Virtual Reality Modeling Language Specification, ISO/IEC DIS 14772-1*. (1997). (http://vrml.sgi.com/moving-worlds/).

Acknowledgements

This paper is a result of ProBrAl 002/94, a cooperation project between UNICAMP and the TH-Darmstadt, sponsored by CAPES (Brazil) and DAAD (Germany). We appreciate the support granted by these institutions. We would like to thank FAPESP, CNPq, and the Computer Science Dept. (University of Waterloo) for the support of some authors, and also José M. De Martino, Dr. Rüdiger Strack and Jian Zhang for helpful pointers and clarifying discussions.