6.7 JAVA AND THE WORLD WIDE WEB: THE RIGHT CHOICE FOR INTERACTIVE SYSTEMS

William Hibbard *, John Anderson and Brian Paul
Space Science and Engineering Center
University of Wisconsin - Madison

1. INTRODUCTION

Seven years ago at this conference we presented a paper entitled *Unix and X Windows: the Right Choice for Interactive Systems* [Hibbard, Santek, Voidrot-Martinez, Kamins and Vroom, 1990]. The paper argued that Unix, X Windows, TCP/IP, C and Fortran were useful tools for building Earth science software systems because they provided a uniform software environment across a wide variety of hardware platforms. This uniformity has made it easier to port software between hardware vendors and between hardware generations than was possible with the vendor-proprietary systems that preceded Unix. We have demonstrated the utility of Unix with implementations of McIDAS [Santek, Hibbard, Voidrot-Martinez, Kamins, and Vroom, 1990; Santek, Whittaker, Young, and Hibbard, 1991], Vis5D [Vis5D Web Page] and VisAD [VisAD Web Page]. However, as we and other system developers have discovered, the uniformity of Unix environments is imperfect. Furthermore, the most widely used operating system is Windows, which is incompatible with Unix and X.

2. PLATFORM INDEPENDENCE

Now the object-oriented Java programming language [Java Web Page] and the World Wide Web are providing a uniform software environment across virtually all hardware platforms and operating systems. Sessions about the Web at this conference for the last two years have illustrated the point that the Web is a system building tool rather than just a passive way of organizing data on the Internet. Version 1.0 of the Java programming language was released during 1996 and makes the Web a complete platform-independent system development environment. Java defines a platform-independent computing environment in the following two ways:

1. Java compilers translate Java source code into Java byte codes that are interpreted on a precisely-defined Java Virtual Machine (JVM). The JVM is implemented on each hardware platform and provides hardware independence.
2. Java includes standard Application Programming Interfaces (APIs) for all commonly used operating system services, including file and network I/O, user interface, and multiple execution threads. These APIs are implemented on each operating system and provide operating system independence by making it possible to write Java programs that do not call native operating system services.

* Corresponding author address: William Hibbard, SSEC, 1225 W. Dayton St., Madison, WI 53706; email address: whibbard@macc.wisc.edu

3. HIDING THE NETWORK

Most scientists now use email, ftp, gopher and the Web as part of the normal work day. These tools bring great benefits, but users need to know lots of details about moving data between physical computers and between different software systems.

The Web has become popular because it enables a very easy user interface for retrieving passive information on the Internet. The user interface hides network details from users. Server scripts add some programmability to the Web but lack flexibility and place the computing burden on the server. Now Java adds programmability to the Web that is more flexible and efficient. In particular, the Java Distributed Object Model and its associated Remote Method Invocation (RMI) system provide access to remote objects in a way that is almost indistinguishable from access to local objects [RMI Web Page]. RMI will make it relatively easy to create software that accesses data and invokes other software via Web links. Whereas most distributed computations currently consist of 2-sided communications between client software and server software, distributed objects naturally support computations that run on many computers at once in myriad little client-server exchanges (i.e., when an object invokes a method of a remote object, the invoking object plays the client role and the remote object plays the server role). This is the scenario envisioned in the phrase "the network is the computer". In this scenario the network will be invisible in the same way that the inside details of computers are invisible to current users.

NCSA Habanero and Sun's planned Java Media Share API will make it relatively easy to build collaborative user interfaces to software systems [Habanero Web Page; Java Media Web Page]. Specifically, the same user interface will be displayed at different workstations and user control events (e.g., mouse clicks on various widgets) from these workstations will be routed back to the application. Tools like RMI, Habanero and Java Media Share will make it easier to create software systems that relieve users of having to know about the existence of multiple physical computers and their network connections.

4. PROMOTING BETTER PROGRAMMING

The genius of the Java programming language is largely in the way that difficult implementation issues are addressed in the basic design of the language. In addition to enabling universal access to software, platform-independence will also effectively eliminate continuing maintenance to adapt to hardware and operating system changes.

Java is a small language that provides expressive power through a few orthogonal elements rather than through a bloated set of features. The language eliminates several types of errors that plague complex systems:

1. Pointer errors are syntactically impossible.
2. Memory leaks are impossible because of the use of garbage collection.
3. Array bounds are checked on all array accesses.

Java provides sophisticated exception handling that enables developers to easily build error handling into their software. The Java language includes special types of comments for embedding documentation directly in class and method definitions, and a utility program for generating a linked set of Web documents from these embedded comments.

5. MISSING PIECES

There are currently a few pieces missing from the Java system development environment but these are being filled in. Specifically:

1. Efficient compilers. Java compilers translate Java source code into Java byte codes that are interpreted on a precisely-defined Java Virtual Machine, at about 1/30 the speed of C++ (estimates vary). However, many companies are developing "just in time compilers" that translate Java byte code into native code on various computing platforms. Sun is also working on processor chips that execute Java byte codes directly.
2. Reliable security. Security holes are still being discovered. However, these are being fixed as they are found. We are confident that Java security will become reliable.
3. 3-D graphics. The Java 3-D graphics API has not been defined yet but a spec will probably be published by the time of this AMS conference and implementations are promised during 1997 [Java Media Web Page]. Sun and its partners have said that the 3-D API will be compatible with the Virtual Reality Modeling Language (VRML) that is being defined as a graphical alternative to the Hyper Text Markup Language (HTML) that is used to define almost all Web pages.
4. Imperfect platform independence. Small incompatibilities are being discovered, particularly in the Java user interface API. We believe that the use of Java applets embedded in millions of Web pages will force compatibility.

6. CONCLUSION

The telephone, television and radio are examples of existing ubiquitous communications networks. Many vendors compete to sell user appliances and communications links for these networks, yet they are completely compatible with each other because the market demands compatibility. The same will happen with the Internet. Java and the World Wide Web are important steps in this direction, and their use is in the long term best interests of the Earth science community. The real question is how quickly we adopt these standards as the basis for our systems. We are demonstrating our faith by rewriting our VisAD system in Java [Hibbard, Anderson, and Paul, 1997].

7. REFERENCES

Habanero Web Page.
http://www.ncsa.uiuc.edu/SDG/Software/Habanero

Hibbard, W., D. Santek, M-F. Voidrot-Martinez, D. Kamins, and J. Vroom, 1990; UNIX and X windows: the right choice for interactive systems. Preprints, Conf. Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology. Anaheim, Amer. Meteor. Soc., 162-163.

Hibbard, W., J. Anderson, and B. Paul, 1997; A Java and World Wide Web implementation of VisAD. Preprints, Conf. Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology. Long Beach, Amer. Meteor. Soc.

Java Web Page.
http://java.sun.com/

Java Media Web Page.
http://java.sun.com/products/apiOverview.html

RMI Web Page.
http://chatsubo.javasoft.com/current/rmi/index.html

Santek D., W. Hibbard, M-F. Voidrot-Martinez, D. Kamins, and J. Vroom, 1990; A UNIX and X windows implementation of McIDAS. Preprints, Conf. Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology. Anaheim, Amer. Meteor. Soc., 164-166.

Santek, D., T. Whittaker, J. Young, and W. Hibbard, 1991; The implementation plan for McIDAS-AIX. Preprints, Conf. Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology. New Orleans, Amer. Meteor. Soc., 177-179.

Vis5D Web Page.
http://www.ssec.wisc.edu/~billh/vis5d.html

VisAD Web Page.
http://www.ssec.wisc.edu/~billh/visad.html