Full text of:

Hall, H. & Read, T. (1996). Java: an explosion on the Internet. In D.I Raitt & B. Jeapes (Eds.), *Proceedings of the 20th International Online Meeting,* London, December 3-5 1996 (pp. 49-55). Oxford: Learned Information.

Keywords: application development; information retrieval; Java; network computing; programming languages.

Abstract

Summer 1995 saw the release, with considerable media attention, of draft versions of Sun Microsystems' Java language and the HotJava browser. In the past few months Java has been heralded as the latest "killer" technology in the Internet explosion. Licensing terms have been agreed between Sun and numerous companies including Microsoft, IBM and Netscape.

Java is set to change the way that people interact with Intra/Internet. Discussed in this paper are:

- how a domestic interface development tool became an Internet product;
- the Java environment and related products: HotJava; Java Virtual Machine; JavaOS and JavaChips;
- Java's strengths as a developer and user tool: platform independence; programming quality; object-orientation; security; commercial promise; application integration;
- Java's competitors: ActiveX and Inferno;
- Java applications: cybertrade; corporate information flow; information services specific projects.

A glossary of terms is provided at the end of the paper. All trademarks and tradenames are acknowledged.

1. Introduction: Domestic interface developer to Internet dream product

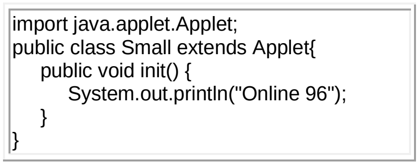
Java was born as Oak, a programming language to be used for the development of user interfaces for consumer electronic devices such as video recorders, telephones and products of Video on Demand (VOD). Whilst anticipated growth of the VOD market has been slow to materialise, interest in the Internet has surged. Oak became Java and, with little active marketing, software developers rushed to adopt it to write their Intra/Internet applications. The attendance record of 6000 delegates per day at JavaOne, the first Java developers' conference held in San Francisco's Moscone Centre at the end of May 1996, demonstrates an unprecedented eagerness for such a young product. For information professionals curious about Java enthusiasm, this paper outlines the programming environment's characteristics and complementary products; its strengths as a software development tool; competitor products and online applications. Inevitably any discussion of this kind necessitates the use of technical terms and acronyms, and for this reason a glossary is appended to the paper.

2. Java environment and products

Since the release of draft versions in Summer 1995 the Java environment has been the subject of an immense amount of media coverage. Several other complementary Sun products which use the Java name have also been developed, and subsequently discussed in the computing press. These include the HotJava browser; Java Virtual Machine; JavaOS and JavaChips. To the casual observer the inter-relationships of these products are difficult to distinguish. A discussion of this "Java jigsaw" serves to put the pieces together in a meaningful fashion.

2.1 Java programming environment

The main product, the Java programming environment, is a programming language (just like BASIC, C or FORTRAN), with a number of additional tools. In its raw form the Java language is unlikely to interest the creator of a library web page or an average Internet end user. Computer enthusiasts may take a different view. Tools for creating the small applications in Java known as applets include SunSoft's Java Workshop, Cafe produced by Symantec, a product of Borland called Latte and Microsoft's Jakarta. Figure 1 shows a short Java program that prints out the phrase "Online 96".



```
import java.applet.Applet;
public class Small extends Applet{
    public void init() {
        System.out.println("Online 96");
    }
}
```

Figure 1 - A Java program

2.2 HotJava

HotJava is a browser. Written in Java, this browser permits on-screen display of the applets. Applets can be loaded from both the user's local disk and, more appropriately, across the Intra/Internet. The HotJava application can be considered as a framework into which new units can be dynamically plugged. These units are designed to provide additional functionality for purposes that might not have been considered when HotJava was originally conceived. The user need not be concerned about how the HotJava browser operates, nor worry about compromising the security of their machine.

2.3 Java Virtual Machine

The Java Virtual Machine is software specifically designed to run programs written in Java. Unlike operating systems such as MS-DOS, the Java Virtual Machine creates a "padded cell" environment for the programs. This prevents virus-like attack of the user's computer.

2.4 JavaOS

The Java Virtual Machine works with the operating system. Although environments such as Microsoft Windows, MacOS or UNIX can provide basic functionality for the computer hardware in use, they are not necessarily available for all types of computer such as Personal Digital Assistants (PDAs) like the Apple Newton, and TV set top boxes for VOD purposes. The JavaOS meets the need to support the Java Virtual Machine by providing very low level hardware routines: threading; memory management; input and output and networking. It can also make provision for higher level class libraries. The overall effect is to provide an identical platform for an application or applet written in Java. Figure 2 gives a diagrammatic interpretation of the JavaOS architecture.

2.5 JavaChips

JavaChips offer a low cost, open, alternative to the standard processors used in PCs (for example Intel) and workstations (such as SPARC, Alpha and PowerPC). A large proportion of the JavaOS and Java Virtual Machine is implemented by the JavaChip in hardware. This improves the performance of Java applications/applets since the programs do not need to be interpreted by software: they are executed directly. Cheap processor power paves the way for low cost systems such as the network computing devices already previewed by Oracle and Sun.

3. Software developer and user dilemmas - Java's responses

The extent to which a programming environment minimises the software developer's difficulties in the design of applications helps determine the rate of adoption of any new product. Similarly there are criteria of importance to those who want to use the technology to provide services, and to the end-user of applications, regardless of language. Java's potential can be assessed against the common problems of platform dependence; programming quality; procedural programming; security and inter-operability.

3.1 Java's platform independence: "Write once and run anywhere... safely"[1]

The platform for which a computer application is written depends on a number of factors. Most important is consideration of the potential user group: the size of the projected market and competing products within the identified market. Once a decision has been made to write for PC, MacIntosh or UNIX users, further choices face the developer: the Application Programmatic Interface (API) to be employed; the computer language to use; the object model to adopt. Each of these secondary decisions may limit the market again, or at least necessitate a recompilation or porting exercise to ensure that the application runs across multiple platforms. It is little wonder that less time than the user might hope is devoted to quality assurance and testing.

The biggest benefit of an application written in Java is that it can run on any major platform. The Java Virtual Machine described above provides the same functionality to Java applications regardless of hardware configuration. Windows NT, Windows 95, Windows 3.11, MacOS 7.x, Solaris, HP-UX, AIX, O/S2 and various network devices are all "Java-compatible". This represents a huge potential market for the software developer, and instead of putting effort into locating a potential client-base, the real priorities of software development emerge: the application's functionality, user interface and quality. The chore of learning and keeping up to date with programming languages for each different environment is eliminated.

3.2 Java promotes quality: reduced chance of program error

Complex and huge volumes of computer code generate modern day computer applications. The smallest error has the potential to cause a machine to crash. C code, on average, contains one error every 55 lines (Ref 1).

Certain features common to other computer languages have been deliberately left out of Java. For example, there are no pointers, no goto statements and no need to manage memory explicitly since "garbage collection" is automatic. By removing these, and a number of other unnecessary language constructs, the threat of programming error is greatly reduced.

3.3 Java and object-orientation: new, richer applications

Until the recent adoption of object oriented technologies programmers have been obliged to "recreate" or "rework" earlier code in order to progress a new project. This is analogous to scientists being repeatedly forced to reprove the basic tenant of their subject prior to taking their work any further. With such constraints Einstein might have given up and never produced his General Theory of Relativity!

Java is totally object-oriented so that developers can re-use applets and class libraries that have been developed by themselves and others to create new and richer applications.

3.4 Java security: cynical software, download limits and digital signatures

Downloading programs from the Internet is regarded as a risky business. Users fear viruses, worms and Trojan horses. Ultimately security has to reach the highest level possible without making a system unusable.

With this in mind the Java Virtual Machine, which runs Java applications, has been designed to be sceptical of any code that the user executes. It assumes the worst, checks the integrity of the byte-codes and passes them through a security algorithm. This is to ensure that none of the basic rules set in the specification are by-passed before a program is executed.

Applications downloaded from the Intra/Internet are prevented by the Java Virtual Machine from performing many basic functions, such as reading from or writing to the hard disk, on the user's machine. Trusted applications or applets installed locally are exempt. Any applet performing in an MS DOS virus-like manner would normally be obstructed.

Security scares relating to Java have been attributed to implementation errors within the Java Virtual Machine, or very specific circumstances such as situations where cyberthugs have persuaded their gullible prey to save and run unsafe programs locally. Security is under constant review in an attempt to close any loop-holes that might be exploited. Digital signatures will give users an additional assurance that download requests are met with the genuine article.

3.5 Java exploits the commercial promise of the network

To a new PC user landing a personal machine on the desktop is the first priority. Subsequent network connections are a bonus. In the UNIX world networking and a client-server approach is a facility that has been taken for granted, and exploited by Java technologies.

In many cases Java expects that the classes or components that constitute an applet or application will be loaded from a remote server. The security scheme prevents the application from then communicating with any other system arbitrarily. Remote method invocation (RMI), however, allows interaction with the distant application. One major benefit of this is that little code needs to be downloaded. Another is that service providers are able to hide the actual implementation of what they offer. Examples of commercial uses of networked Java applications include the calculation of insurance policy payments or the execution of online searches based on remote user input. The commercial organisation protects its intellectual property rights to methods in use by limiting downloadable code to algorithms necessary to complete the transaction.

3.6 Java and application integration: "the network is the computer"[2]

Computer users rarely ditch older equipment in favour of wholesale migration to a new system. Rather, they cope with piece-meal integration of existing "legacy" hardware and software. Software developers recognise that user needs are determined to a degree by older configurations. The capacity of software applications to integrate on an equal footing with the de facto standard tools of the native environment is key to product success. The volume of developers who adopt the tools as their primary development environment is also important. Currently there are three main approaches to integration.

The current de facto standard for integration on the desktop PC is Microsoft's Object Linked Embedding (OLE). This allows users to drop data into applications (OLE drag and drop); link or embed data from one application into another (OLE documents); automate data and function shipping between documents (OLE automation) and supplement the functionality of an application with custom extensions (OLE Custom Controls). The main thrust of this approach is to synthesise business solutions by bolting together "component" software on the desktop using a binary application interface standard (Ref 2).

In the UNIX environment and beyond the picture is rather different. The network is the focus for application integration. The major players, such as IBM, Hewlett-Packard, Digital and Sun, conform to the Object Management Group's Common Object Request Broker Architecture (CORBA). Each company implements its own version of this standard. For example Sun uses NEO Workshop, which includes a CORBA compliant Object Request Broker (ORB). Applications components, written in any computer language and to this standard, can then be glued together across the network and upgraded or re-engineered as appropriate.

Straddling the gap between these two approaches are products such as Iona Technologies Ltd's Orbix. Through use of Interface Definition Language (IDL) remote objects are described. The Orbix tools then create an OLE automation server for the PC. This can be called from applications such as Microsoft Excel via Visual Basic. This approach allows considerable freedom to integrate information found from a variety of sources across the Intra/Internet.

Developers are striving to break down their software into smaller, more manageable pieces and thereby move away from single monolithic applications. If the interfaces are well defined and conform to standards then users should be able to choose to use and integrate "best of breed" software packages to fulfil their business requirements. The arrival of the Intra/Internet and dynamically extensible products such as Netscape v2.0 and the HotJava Browser have broken a potential Microsoft monopoly in this arena.

Microsoft has been moving in the component software direction since the release of OLE v1.0, but until recently it has been firmly focused on the desktop only. Commercial agreements with Digital, Software AG and others are intended to make OLE/COM available on multiple platforms and, through the use of Distributed Computing Environment/Remote Procedure Calls (DCE/RPCs), network ready. In September 1993 IBM and Apple announced their equivalent technology called OpenDoc and DSOM which provides similar functionality in a multi-platform environment. This was an attempt to break the Microsoft stranglehold, but despite a number of supporters the project results have not met with much visible success or publicity (Ref 3).

When Java was released in mid-1995 both source code and early access products were made freely available on the Internet for both commercial and academic review. It would now appear that Java has leap-frogged OpenDoc as the main competitor to OLE/COM as the preferred option for component software development. For it to succeed on the desktop it needs to provide peer level integration with Microsoft Internet Explorer as the potential vehicle (or "container" in the industry jargon) of choice. Java Beans fits the requirement of supporting OLE automation, drag and drop and OLE controls. This defines an API for application developers. The completed applets integrate tightly with the environment in which they are executed.

A whole range of new opportunities for the implementation of a corporate desktop results from these developments. It is possible that over the coming years the browser becomes the container for application delivery with information and data being drawn from both the Intra and Internet seamlessly into a single interface. Just as the IBM PC supplanted the IBM 3270 terminal as the industry standard desktop, so may the low-cost network computing device slowly erode the dominance of the "Wintel" PC. This will be welcome news in organisations where computer administration costs are burdensome. A device that is simply plugged into the network is cheaper to service than a suite of PCs (Ref 4).

4. Java's competitors

Developments in this sector of the information industry race ahead at speed. Alliances are formed and products launched, leapfrogged, re-packaged or dropped with alarming haste. The two other major players are Microsoft's ActiveX and Inferno from Lucent Technologies.

4.1 ActiveX

An early Java competitor casualty was a Microsoft Internet Studio product (Blackbird). Now Microsoft's efforts are concentrated on the ActiveX framework. At the same time

Microsoft has also committed to embedding the Java Virtual Machine in their Windows products such as Internet Explorer 3.0.

4.2 Inferno

Inferno from Lucent Technologies (part of Bell Laboratories) sets out to meet the gap in the TV set top box (for VOD), PDA and Internet market. Inferno comprises a programming language called Limbo; a virtual machine named DIS and a communication protocol called Styx. It has already been ported to a number of platforms including Windows NT, Windows 95 and UNIX. The success of Inferno may well depend on both its inherent cross platform functionality and its ability to tightly integrate with native environments.

5. Java applications

Java and the Intra/Internet have the potential to change the way people work and obtain information, products and services over the networks.

5.1 Purchase of goods online

One of the biggest opportunities that the Internet offers is the option to sell and purchase services directly online. Those on offer at present are relatively mundane. For example, wine can be bought directly from a supermarket chain in response to a static Internet page advertisement. The platform independence of Java applications means a much enlarged potential market for products. Multimedia animations may be used simply as a gimmick to attract attention to wares for sale, or as an essential component of marketing, for example a video clip can demonstrate product use. Payment may be made using cybercash, an API of the Java Wallet. Nortel has demonstrated how intelligent agents such as software applets can be used in new mobile telephony applications to provide "all singing, all dancing" user devices (Ref 5).

A route to market that involves much lower overheads than traditional options may be established. This could squeeze the market for established firms and in turn give rise to a new breed of provider of the Yahoo ilk (Ref 6). With only a Java'tised home page as the shop window the old maxim of "Caveat Emptor" may well be in order as there will be little to distinguish the major players from the small time outfits.

5.2 Improved access to corporate information resources

The leading relational database management system (RDBMS) vendors such as Informix, Oracle and Sybase are already racing to provide Java and Internet-ready versions of their tools. In addition to this the development of Java Database Connectivity (JDBC), an API for relational database access similar in style to Microsoft's Open Database Connectivity (ODBC), will facilitate the development and deployment of applications that access corporate databases. Quicker and simpler access to information improves customer service, lowers costs and shortens the period to launch products and services.

5.3 Java applications and information services provision

For the information professional Java has been described as "yet another tool developed outside the library community that will nonetheless have a huge impact on information services" (Ref 7). Responses to a Net query made in December 1995 demonstrated that American librarians are interested in the benefits that the technology can bring to service provision (Ref 7).

One of the most obvious areas for application of Java in the library context is for information retrieval (IR). Applets have already been implemented for some IR purposes. Examples include an applet called Sketch and Fetch that offers an interface between a journal and a chemical database, which was announced in April 1996 (Ref 8); staff at Cambridge University Press have written code to allow hierarchical searching over subject areas (Ref 9). The Edinburgh Engineering Virtual Library (the EEVL Project based at Heriot-Watt University and funded as part of the British eLib

Programme) uses Java as one of the tools for building a gateway to high quality web resources in engineering. Java was chosen for constructing a customised web server in preference to C++ because of its ready availability and ease of use. At the time of writing Java applets are being developed to support the searching and browsing options that EEVL provides. They will allow a free floating window containing the options so as to save users the trouble of reverting to the search page within the browser (Ref 10). This approach could be applied to OPAC design to improve interface and search capabilities. Applets incorporating help routines or to set and save user profiles and previous searches improve service to the library catalogue user. National Semi-conductor built a product catalogue of some 30,000 items with Java that satisfies customers on various platforms quickly (Ref 11).

Java may become an important tool in user education. Multimedia tutorials can aid in developing information skills such as competence in database searching (Ref 12).

It is relatively easy to suggest applications for a new technology in the library environment, and the examples given show that with adequate resources these can be brought to fruition. However, information services personnel face a number of difficulties in integrating technical innovations, and these can limit the adoption and exploitation of new technology which, in ideal circumstances, could serve user information needs. Java might be regarded as a perplexing "Just-Too-Fast" development (Ref 13) built up by excessive media hype, and library staff in particular may feel cautious in exploring its potential. As Crawford and Gorman (Ref 14) caution "... librarians have been urged to use almost every one of these innovations immediately to protect their libraries from becoming irrelevant" and the result has been library stock rooms littered with equipment defunct after six months of use. Many (often public) cash-poor library services may be obliged to sit and observe developments, unable to afford investment in equipment or training despite enthusiasm for what looks likely to become part of the standard set of tools to access information objects on the network.

6. Conclusions

In the computer industry marketing matters, and any hype certainly helps focus attention on a product. Sometimes, however, it frustrates those who want to keep up to date with developments, yet find it difficult to penetrate beyond the hyped messages. At this stage it can be concluded that Java may not be relevant to everybody who follows the computing press or navigates the Internet. However, in the future few will avoid its influence. What Java does offer now is a greater choice of tools for software development for several platforms, easing developer decisions on market penetration and choice of programming language. End users can benefit from the enhancements Java brings to web page design. In the field of library and information science the most obvious applications are in information retrieval. Long term, Java's legacy is likely to lie in the shift towards true network computing.

References

1. Klein, Robert (1996) "Doing the Java jive" *New electronics 27* February 1996, 39-40.

2. "The Microsoft object technology strategy: component software" paper available from http://www.microsoft.com.

3. "OpenDoc vs. OLE 2.0 Superior by design - a developer's view" paper available from http://www.ibm.com.

4. Gartner Group (1993) Do you know what your PC really costs? More than $40,000. *Industry service 9* (12) 1993.

5. Sugarbroad, Ian "Using intelligent agents in new mobile applications". Paper presented

at JavaOne, San Francisco, May 1996.

6. Kehoe, Louise and Paul Taylor, "The online challenge flowers", *Financial Times* 14 March 1996.

7. Schneider, Karen G (1996) "Java: the third wave of the Internet". *American libraries 27* (2) 1996, 60.

8. Rzepa, Henry (1996) "Java Applet: Eqn Viewer and standards in Java applets". Message to lis-elib@mailbase.ac.uk 3 April 1996.

9. Bursell, Mike, Cambridge University Press. Personal communication 17th June 1996.

10. Kirby, Richard, EEVL Technical Officer, Heriot Watt University. Personal communication 13th June 1996.

11. Blake, Paul (1996) "Javascript versus Internet Studio" *Monitor* 180 February 1996, 14-16.

12. Mulder, Craig (1996) "Introducing Java to the library world" *MLA* news 283 March 1996, 11.

13. Schneiderman, R. Anders (1996) "Just-too-fast: why the Internet changes more rapidly than is good for us" *Enode* 1(1) 5 June 1996.

14. Crawford, Walt and Michael Gorman (1995) *Future libraries: dreams, madness and reality*. ALA editions: Chicago, 1995.

Footnotes

[1]Dr Alan Baratz, JavaOne Conference, Moscone Centre San Francisco, 29th May 1996.
[2]Sun Microsystems tag line since 1982.

Glossary

ActiveX	Microsoft's competitor of Java
Alpha	Digital chip family
API	Application Programmatic Interface
Applet	Programs created in Java
Blackbird	Microsoft Internet Studio product, now defunct.
Cafe	Symantec's development environment for writing Java applications
CISC	Complex Instruction Set Computer
Class library	Set of software components for a particular function
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
DCE/RPC	Distributed Computing Environment/Remote Procedure Call
DIS	Virtual machine component of Inferno
DSOM	Distributed System Object Model
HotJava	Internet browser created in Java
IDL	Interface Definition Language
Inferno	Lucent Technology's competitor of Java
Internet Explorer	Virtual machine component of ActiveX
IR	Information Retrieval
Jakarta	Microsoft's development environment for writing Java applications
Java	Sun Microsystems Intra/Internet programming environment

Java Beans	A Java API that provides peer level integration with OLE
Java Virtual Machine	A "padded cell" environment for the running of programs written in Java
Java Wallet	An API for Internet commerce
Java Workshop	SunSoft's development environment for writing Java applications
JavaChips	Pico, micro and ultra Java CPU specifications
JavaOne	The first Java developers' conference held in San Francisco, May 1996
JavaOS	Operating system to support the Java Virtual Machine
JDBC	Java Database Connectivity
Latte	Borland's development environment for writing Java applications
Limbo	Inferno's programming language
NEO workshop	A software development product including Sun's implementation of CORBA
Oak	Java's ancestor, a programming language for domestic appliance interfaces
ODBC	Open Database Connectivity
OLE	Object Link Embedding
OpenDoc	Apple/IBM strategy for component software
ORB	Object Request Broker
Orbix	Iona Technology's MS Windows/CORBA integration product
PDA	Personal Digital Assistant (e.g. the Apple Newton)
PowerPC	IBM chip architecture
RDBMS	Relational Database Management System
RISC	Reduced Instruction Set Computer
RMI	Remote Method Invocation
Solaris	Sun's UNIX operating system
SPARC	Scaleable Processor ARChitecture
Styx	Inferno's communication protocol
TV set top box	Control panel for VOD applications
VOD	Video on Demand
Wintel	Windows/Intel