Network Computing Research & Analysis

The State Of Java Report:

IBM

11 Starfire Court
Hewlett, NY 11557
Phone: (516) 792-0997
Fax: (516) 792-0996
Web: www.itfa.com

Table Of Contents

About NC.Focus

NC.Focus is a research and analysis firm that concentrates on the Network/Java Computing marketplace. NC.Focus services include the *In The Final Analysis ...* Newsletter, in-depth technical reports, and on-site consulting. For additional inquiries contact J. P. Morgenthal at (516) 792-0997 or electronically at jp@itfa.com. Additional information can also be obtained from their web site at www.itfa.com.

About The State Of Java Report Service

The *State of Java Report* identifies for readers the leading Java solutions, industry trends, and critical factors regarding adoption. With the rapid pace of change in the Java community, it is inevitable that strong solutions today will be overcome by competitors' solutions in a very short time. Furthermore, since the state of Java is in constant flux, these reports focus on those components that should be immediately adopted and those that require a wait-'n'-see approach. *The State Of Java Report* provides the most timely and up-to-date information about Java in the computing marketplace.

Executive Overview

The industry-at-large need not fear Microsoft co-opting Java and splintering write-once run-anywhere. IBM has already demonstrated its ability to secure Java's fate in the enterprise by delivering crucial components, even across Microsoft platforms. The analogy of the difficulty in turning an aircraft carrier (IBM) *vs.* turning a cruise ship (Microsoft) is applicable in this situation. Microsoft has received considerable, and well deserved, acclaim for their rapid turnabout in adopting both the Internet and Java, but few have recognized the same, considerably larger, feat happening inside IBM. Not only has IBM focused its attention on delivering solutions for the Network/Java Computing universe, but it has finally stepped across product and division lines to develop a powerful single-vendor solution.

Java is clearly an integral part of IBM's future, but understanding Java's role in IBM's solutions for distributed computing requires a rich overview of its entire electronic commerce offering. At the heart of IBM's electronic commerce strategy is the Network Computing Framework (NCF). NCF is a set of client and server software that is tightly integrated to provide ease of implementation and various levels of scaling options. Of the NCF, Java is but one small component that will provide the application layer functionality. That is, *Java is the glue that will integrate IBM's heterogeneous computing platform solution.*

With many operating system vendors ramping up considerable Java efforts, IBM needs to define, and clearly state, its marketable differentiation. Digital Equipment, Hewlett-Packard, Microsoft, Novell, and Sun Microsystems will all be competing with IBM across some subsection of a horizontal market. Each of these aforementioned vendors, in addition to IBM, are putting considerable time and energy into establishing their operating system platform(s) as "The Java Platform." Of note, if there is, or will be, a Java platform of choice, Java will have proved itself valuable to consumers by forcing these vendors into a price/performance/value competition.

IBM's number one advantage is its combination of experience, products, and services. Above all else, this combination will entice many enterprise customers to adopt an IBM solution. Distributed computing is just emerging as a dominant force in the enterprise, due in part to advances brought about by Internet awareness. Few organizations can afford the time and effort required to, first, understand this emerging shift in computing, and secondly, to rely on a multiple vendor solution to provide them with their early Internet advances. Of course, the key to success for IBM is to provide a consistent and unbeatable set of services on all of their operating system and hardware platforms. Moreover, IBM must identify for each of their platforms a primary reason for adoption and the level to which it scales.

With such a diverse portfolio of operating systems and hardware, IBM's customers have understandable expectations for a solution relative to a particular platform. Going forward, IBM's customers will be able to choose from OS/2 on Intel, AIX on RS/6000, OS/400 on

AS/400, and OS/390 on S/390. Providing a consistent set of services across this portfolio, inclusive of Java, will yield extreme variances in execution performance. With respect to Java, each of these platforms will enhance the Java computing platform either in services, speed, or both. *Hence, for IBM, Java becomes a platform amplifier, not a platform equalizer*.

A basic market assumption being made today is that all applications written in Java are designed for write-once run-anywhere–not true! Java represents a well-designed object-oriented programming language and standard application programming library; therefore, it is also suitable for platform-dependent application development. That is, why should an organization use one language/API set for write-once run-anywhere and an entirely different language/API set for platform-specific development? There is no clear reason for doing so, except in the cases where performance is crucial and could only be satisfied by lower-level programming languages such as Assembler.

Therefore, a set of Java applications will exist that is specifically designed to run on one particular hardware/operating system platform. With this established, operating system vendors will need to ensure that their Java platforms are 100% compatible with the JavaSoft-defined standard, but that they also provide native access to required operating system and hardware functionality. For the moment, Microsoft Corporation is the leader in providing native integration between the operating system and Java code; however, this lead will not last long. Microsoft's compliance with JavaSoft's standards are waning. Meanwhile, IBM is now working on native functionality for their OS/390, AS/400, AIX, and OS/2 operating systems. The fruits of this labor will be delivered between fourth-quarter 1997 and first-quarter 1998, along with 100% JavaSoft compliant Java virtual machines.

One overriding factor regarding IBM that impressed NC.Focus immensely was IBM Executive Management's enthusiasm and excitement toward this initiative. From the perspective of an organization looking for a vendor to partner with on their Java endeavor, IBM is by far the most advanced in terms of depth of understanding, platform support, but most importantly, commitment. While Sun Microsystems may be the creator and keeper of Java, IBM is the true driving force behind enterprise adoption of this new technology platform. Moreover, the synergy between Sun Microsystems as creator and IBM as an implementor and driver has forged an alliance that will prove to be the single most deterministic partnership for the future of distributed computing.

IBM's Java Offering

A recent survey of IBM's commercial operating systems presents a stronger, more integrated story, due in great part to IBM's Java initiative. Java alone provides a homogeneous computing platform on top of a heterogeneous hardware and operating system environment. IBM enhances this capability of Java by providing a consistent set of homogeneous services to

complement the Java platform. That is, IBM provides access via Java to its MQSeries asynchronous messaging, DB2 database via Java Database Connectivity (JDBC), Encina, Lotus Notes, and CICS transaction processing software. Of this set, only CICS represents a server-side Java initiative; the others are merely designed to provide access for thin-clients.

IBM's Java efforts are not limited to implementing Java virtual machines into IBM operating systems. It extends to leadership of the San Francisco project–a 100% pure distributed Java framework developed to improve interoperability of business applications among a wide array of software vendors' applications. San Francisco is in its early stages of definition, but it has already produced a set of common business objects that operates in both a local and distributed manner; the latter using Java's Remote Method Invocation (RMI) .

IBM is making a clear and definite attempt to provide the best platforms for execution of Java applications. It is also clear that IBM is serious about making write-once run-anywhere operate across IBM operating systems and hardware. *For customers looking toward Java to develop and/or purchase applications that execute across Intel, midrange, and mainframe hardware with reasonable performance, IBM is currently the only organization capable of providing such service.*

IBM's core competency lies in providing a diverse portfolio of operating systems and hardware to meet both the business and technical needs of the enterprise. The following is a high-level summary of the benefits each platform provides and the advantages of Java relative to that platform.

- OS/390 remains one of the most secure and widely-used environments for high-speed mission-critical transaction processing. Its ability to segregate multiple subsystems running simultaneously has provided the groundwork for Internet/Intranet interoperability. Shipped as a standard part of each OS/390 today is a subsystem that implements UNIX services to all running applications. This UNIX subsystem is the foundation upon which Java on the OS/390 is built, providing the Java virtual machine with file, print, X-Windows, and TCP/IP socket-based services. *The OS/390 presents Java with a highly scalable transaction processing platform, while Java presents the OS/390 with an opportunity to exploit its existing legacy data for use with new client/server applications.*

- AS/400 is extremely popular with organizations with limited IT resources. It is mainly used to provide basic operational functionality, such as accounting and inventory, but also to support a large number of users accessing a singular data repository. A common criticism of this platform has always been the continued use of the RPG programming language for development of most of its

applications. RPG resources are becoming scarce in the age of C++ and Java, thereby making it difficult to maintain the existing code base. IBM hopes to overcome this problem fairly soon by providing a native Java 1.1.x virtual machine into the AS/400 by first quarter of 1998 (a 1.0.2 compliant Java virtual machine now exists). *Due to AS/400's unique single-level store architecture, it is the only IBM operating system positioned to become the leading data warehouse for Java objects.*

- AIX, IBM's UNIX System V/BSD compatible operating system, provides a POSIX-compliant operating system designed for high-availability and integration with key IBM server software. AIX is now the most popular UNIX platform for Lotus Notes/Domino and is a leader in World Wide Web server platforms. AIX running on IBM SP2 parallel architecture was responsible for handling 17 million hits a day during the last Olympics. AIX is also the first IBM platform to have a shipping version of the Java development kit version 1.1.2 and a just-in-time (JIT) compiler. This obviously demonstrates that *IBM considers AIX to be a key platform for distributed Java*.

- OS/2 is the only operating system IBM develops for the Intel hardware platform. While early attempts to distribute OS/2 as a desktop operating system for SOHO and home-user markets failed, many large organizations rely on the power of OS/2 to deliver robust mission-critical application support. For example, many automated teller machines (ATM) and airport ticket-agent terminals are based on OS/2. IBM continually supports OS/2 and is working diligently on enabling it for Internet services for its existing OS/2 customer base. OS/2 also plays a prominent role as lower-cost branch servers in industries such as insurance, transportation, and retail banking. In these areas, *OS/2 will provide server-managed client and network computer support to replace aging terminal-based and desktop applications with Web and Java savvy ones.*

After operating systems, IBM holds a market leadership role in middle-tier software solutions. The set of middle-tier software, or middleware, that IBM sells includes: transaction processing monitors, asynchronous messaging and queuing, a CORBA 2.0 compliant object request broker and services, and Lotus Notes/Domino messaging. Because middleware facilitates moving and processing data across all of IBM's diverse operating system offering, it is a critical component of IBM's network computing framework and e-commerce strategy.

Rounding out IBM's Java offering is the Network Computer Framework, or NCF. This framework is a set of server and middleware applications that will provide a consistent set of services to thin-clients. Included in this offering is: Domino Mail, Domino Messaging, DB2

database, and CICS[1] transaction processing. The key to the NCF is that it is completely integrated so that, regardless of platform, an out-of-the-box solution for thin-client/network computing is readily available.

Recently, there has a been a growing trend to integrate transaction processing and object-oriented technology. Leading the way is the Object Management Group's (OMG) Object Transaction Service (OTS), which is a specification for providing transactions over Common Object Request Broker Architecture (CORBA) objects. The OTS specification has also laid the foundation for transactions over Java objects, giving way to the Java Transaction Service (JTS). IBM has been a leader in both of these efforts. Furthermore, when JTS is fully defined, IBM's CICS/Java offering will be JTS compliant, providing secure, robust, and high-scalable transactions to Java applications.

Included among the list of external connectors for the NCF is the MQSeries Java gateway. This is a 100% pure Java solution for accessing IBM's asynchronous messaging solution to control and add queues and data. Notably, MQSeries has the least presence among all IBM server software, with Lotus Domino Messaging services taking a more predominant position. This may be due to the complexity in administering and configuring MQSeries environments. With Domino, IBM can deliver an out-of-the-box solution, but still offer the ability to connect with the more powerful, but more complex MQSeries software.

For customers looking for a single vendor solution for their next millennium distributed computing requirements–IBM presents a hard-to-resist offering.

Survey Of The Competitive Landscape

The conclusions reached in this document were the result of a broad investigation into the services and products offered by vendors with competitive Java initiatives that stretch from the desktop to the server. The ability for a single vendor to provide a robust and scalable solution is critical when deploying solutions based on emerging technologies. More importantly, the single vendor solution must be cohesive throughout its hardware, operating system, and server software offerings.

The group of vendors evaluated for their ability to provide this solution includes IBM, Microsoft Corporation, and Sun Microsystems. Microsoft Corporation is the only company evaluated that does not offer a proprietary hardware platform. Digital Equipment, Hewlett-Packard and Silicon Graphics were also evaluated, but were found to be missing too many pieces to provide a complete single-vendor end-to-end solution.

[1] CICS is the only NCF component that offers Java runtime support.

Java has brought about a whole new awareness of distributed computing possibilities. These possibilities require the cooperation of client and server computing platforms. The client market will be divided between power desktops, from Microsoft and Apple, and network computing devices from a host of vendors. So far, this portion of Java's development has been slowed by inconsistent versions and Java's overall immaturity. Still, the ability of Java to enhance client-side computing has been demonstrated by support for the 1.0.2 version, and will continue to improve with newer releases.

Client software is also hindered by the requirement for Java to work consistently within the Web browser environment. This requirement forces not only the operating system vendor to provide a robust Java virtual machine offering, but also every single Web browser vendor offering a product for a particular platform. Of note, sometimes it just provides a competitive advantage for a Web browser vendor to offer their own Java virtual machine in places where one is offered with the operating system. However, this limitation does not affect the development of server and middle-tier software written in Java.

Server and middle-tier software rely on a Java virtual machine to provide a runtime execution environment for Java applications. This freedom from the Web browser has created more consistency for Java virtual machines in the server environment. Furthermore, it presents a greater opportunity for vendors of server platforms to create a superior environment for Java to thrive in.

IBM, Microsoft, & Sun Microsystems

The three leading vendors capable of providing end-to-end single vendor solutions are represented by IBM, Microsoft, and Sun Microsystems. For purposes of this document, Microsoft represents itself and all vendors reselling Windows client and server operating systems on their hardware; for example Compaq, Data General, and even IBM.

The following is a list of areas that each company was evaluated for:

- Hardware & Operating System Choice
- Capability To Deliver An End-to-End Java Solution
- Ability To Integrate Heterogenous Platforms

IBM offers the most flexibility when it comes to hardware and operating system choice. IBM offers AIX and OS/2 as their departmental computing solution. AIX represents high-availability and scalability with a standard UNIX implementation SP, Symmetric multi-processor (SMP), uniprocessor, and PowerPC architectures. OS/2 represents IBM's ability to deliver 32-bit multitasking on an inexpensive Intel architecture. For enterprise computing, IBM weighs in with the AS/400 and OS/390. The fact that IBM has chosen to turn these latter two platforms into

Java server environments is key to IBM's leadership in the Java space. *Providing Java on these platforms gives users the ability to link to legacy and to also move forward with new distributed computing development*.

Microsoft's Windows platform offering is primarily for Intel-based architectures. Efforts to offer Windows NT on RISC platforms have not been successful. Sun offers just UNIX for their proprietary SPARC architectures and the lower-cost Intel architecture. Also, Sun has recently entered into the enterprise computing platform space, with ultra-scalable UNIX machines. However, the UNIX-only solution is limiting when compared to IBM's OS/390, which can execute MVS, VM, UNIX, and NT applications.

IBM's end-to-end solution offers the most choice. Starting with the client, IBM offers a choice of PowerPC-based network computers, AIX, or OS/2. Here, IBM has a small advantage over Sun Microsystem's solution, because IBM's NC is already in the channel, whereas JavaStations have not yet entered distribution. Furthermore, IBM is a licensee of the JavaOS technology, allowing IBM to match any NC offering from Sun in the future.

Microsoft's client offering consists of Windows NT workstation and Windows 95. Both are relatively weak network stations, but they offer the highest performance Java virtual machine for the client. Microsoft's WinCE platform is starting to meld itself into a useful thin-client solution. At its current pace, it could provide joint Windows and NC functionality for non-power-desktop applications by the end of 1998. This could affect the ability of Sun and IBM to deliver NC solutions to the desktop.

A key ingredient to tying heterogeneous platforms together is the middle-tier software. IBM is a leader with six middleware software solutions: CORBA, DCE, MQSeries, Lotus Notes, DB2, and CICS/Encina. Neither Sun nor Microsoft have as complete of an offering or diversity of platform support for competing products. While individually, better solutions may exist from other vendors for each of these middleware components, combined they are unmatched. That is because IBM offers these products on all four of their operating systems, as well as Windows NT. Futhermore, IBM is in the process of rolling out Component Broker, which promises to simplify development of applications using these middleware pieces.

Even though Sun offers the same operating system on both SPARC and Intel-Solaris, they must still deal with the problems of heterogeneity of hardware platforms. To account for this, Sun presents interoperability via CORBA and their popular Network File System (NFS). Of note, Sun's Network File System (NFS) often competes with and wins competitive analyses against IBM's DCE Distributed File System (DFS). Still, this integration of applications operating over SPARC and Intel requires the synchronous processing of CORBA, or file sharing of NFS. In addition, Sun's UNIX-only offering is limited and must compete with IBM's AIX, where it does not have a sizeable lead.

Microsoft's offering for Windows-based middleware is shaping up quickly. The next version of Windows NT Server 5.0 will include transaction processing, asynchronous messaging, online analytical processing (OLAP), and component functionality through Distributed COM. This will affect sales of competing IBM products on Windows platforms. However, Microsoft's products are forced to interoperate with IBM middleware solutions on AS/400 and OS/390. Also, customers selecting an IBM solution on Windows NT today will not quickly switch to Microsoft's solution without real identifiable benefits.

In the short-term, IBM's OS/390 still provides unmatched scalability. Longer term advances in microcomputer clustering hardware and software will offer users more choice for high-availability and redundancy. The competitive issues here will eventually come down to price and performance, with microcomputer clustering offering the advantages of distributed redundancy. Still, performance and scalability of Java on the OS/390 should exceed the ability of any of the midrange and microcomputer offerings available now and in the near future. Hence, IBM's solution will still provide handling for the greatest number of clients in the foreseeable future.

One final, but important, point is IBM's international presence. Sun and Microsoft are still struggling to build their international stature, yet sales of IBM hardware and software overseas is still on the rise. In fact, the Java Internationalization programming interface was developed and provided by IBM for inclusion into the Java Development Kit, demonstrating their understanding of international computing requirements. For many multi-national multi-geographical corporations, IBM may be the only single-vendor solution available.

Java on the OS/390

In 1995, IBM unveiled a new state-of-the-art operating system that would transform their existing MVS/ESA architecture into an enterprise-wide open systems platform. Not only does it support open system standards such as OSF Distributed Computing Environment (DCE) and Object Management Group (OMG) Common Object Request Broker Architecture (CORBA), but it also executes UNIX applications. The combination of the open systems approach with S/390's traditional strengths, such as database, transaction processing, and batch processing, make OS/390 a powerhouse platform for any enterprise.

OS/390 is highly optimized to run on IBM’s new S/390 Parallel Enterprise Servers–Generation 4. The Generation 4 is an air-cooled CMOS-based architecture–in contrast to previous architectures that required water-cooling–thus, saving both space and energy while providing ten times the processing power of mainframe computers ten years ago. While this hardware platform can still run the more traditional IBM operating systems such as VM/ESA and VSE/ESA, it’s the combination of OS/390 and S/390 that forms a highly-reliable parallel processing architecture for the enterprise. This is serious computing power that will be attractive

to organizations seeking continuous operation 365 days a year, 24 hours a day. Furthermore, this platform has demonstrated the ability to support 5,000 transactions per second and, with clustering, has supported upwards of 150,000 concurrent users.

Considering what it means to be on the Internet, providing constant access to customers and suppliers, the OS/390 is the best choice for highly-reliable, highly-secure, and highly-scalable electronic commerce solution. With the addition of a high-speed Java virtual machine, the IBM mainframe of days old will be transformed into a souped-up content distribution and Web transaction facility. But, before jumping onto the OS/390 Java bandwagon, users should be aware of exactly what Java on the OS/390 entails.

When the first IBM mainframe rolled off the assembly line, there was no ASCII 8-bit character encoding. Instead, these machines used a 7-bit encoding scheme called EBCDIC, which is how the internal processor represented numbers and letters. When incorporating legacy data based on EBCDIC into the newer ASCII-based computing environments, difficulties arise. In fact, this has been one of the more difficult communications problems to overcome in using the mainframe in a heterogeneous client/server environment. Since all client applications use ASCII encoding, it is apparent that the OS/390 is responsible for transparent conversions.

IBM has implemented early versions of the Java virtual machine using only ASCII character encoding. This requires the virtual machine to convert data to EBCDIC in order to make operating system calls. However, the Java Internationalization programming interface–available in the 1.1.x release of the virtual machine–will eliminate further problems regarding this issue. To the developer, this point should be moot unless the conversions are causing considerable performance problems. But those programming native code libraries for the OS/390 will be responsible for converting all strings between ASCII and EBCDIC.

The Java virtual machine also requires the availability of a local hierarchical file system. Java looks for its binaries by converting "." notation separated names into a hierarchical file system path. For example, "java.lang.String" would become "java/lang/String.class." The traditional MVS environment does not use a hierarchical file system, but instead uses the concept of a data set. This can cause mapping problems between the Java virtual machine implementation and MVS data storage. IBM has overcome the file system requirements by using OS/390's UNIX services, which is shipped as a standard component of the OS/390. OS/390 UNIX services provides HFS, Hierarchical File System, which implements a directory structure on top of a single MVS data set. Of note, moving MVS data into these HFS partitions requires the user to ensure that the 'BINARY' copy option is used. This performs an ASCII-based byte-by-byte copy; otherwise, it will be copied as an EBCDIC, thus corrupting the file.

OS/390 also adds X-Windowing functionality. Since the S/390 does not have a single output console, implementing Java's Abstract Windowing Toolkit (AWT) could seemingly be

pointless. However, output from a Java application is directed into an X-Window session. This session can be mapped to an X-Window terminal or into virtual memory. In 1998, IBM will add Remote AWT. This is an extension to the base AWT that allows OS/390 applications to transparently support any Java-enabled platform as an end-user terminal, thus freeing them from the initial restriction to just X-Window terminals.

Three additional initiatives for Java on OS/390 include:

- Access to native functionality, such as CICS, IMS TM/DL1, DB2 via JDBC, MQSeries, and access to record oriented data–for example VSAM (partitioned and sequential).

- Capability to translate Java machine code to native machine executable components, leveraging the existing LE runtime.

- IBM has plans to develop a Java virtual machine on their VM operating environment. Additional details on this initiative were not available, but customers are targeting Java as a replacement for COBOL and PL/1 on this platform.

Overall, IBM offers a powerful Java platform with OS/390 for both end-users and independent software vendors (ISV). For customers, its ability to access both *primitive MVS-based data and to communicate with common desktop clients* finally enables a common development and deployment platform for client/server computing. Furthermore, the processing power of the *OS/390 enables the development of ultra high-speed implementations of the Java virtual machine*. For ISVs, Java on OS/390 opens a new market that may have been previously unattainable due to the complexity and expense of developing for this platform.

Java on the AS/400

Java and the AS/400 make an unbelievable combination for business application development. The AS/400 is well suited for developing core business applications, such as inventory management, accounting, and volume data entry. The key to the AS/400 is in its object-oriented design, which simplifies development of applications that must enter, search, and index large volumes of data. Java is a completely object-oriented programming language, and like the AS/400, it has coherence of objects from creation through destruction. The synergy of these technologies provides a powerful, enterprise-level reliable solution for Internet/Intranet server computing.

Clearly, the AS/400 has a market leadership role in small to mid-sized companies where it provides most, if not all, of the computing applications for daily operation. The AS/400 has an extremely large base of application software specialized for vertical market industries such as

manufacturing, health care, and telecommunications. However, most of these applications are written in RPG or COBOL programming languages. This makes customization and maintenance expensive and hard to find, yet programming languages like C++ have not been quick to catch on these environments. This is most likely due to few RPG programmers being re-trained in C++.

For the AS/400, Java may be just what the doctor ordered. With C++ programming, the developer must learn how to write code specifically for the AS/400. Higher level code frameworks can help alleviate the learning curve, but they do not eliminate it completely. With Java, programmers can develop code on personal computing systems such as Windows and UNIX, but can deploy on the AS/400.

With the release of IBM's AS/400e product line, customers may choose to use an AS/400 for Web server and electronic commerce concerns. Additionally, customers may will find the AS/400 intriguing because the platform now allows for extensibility through resource-rich paths. For example, a small organization with mostly networked PCs may need a highly specialized accounting system. Chances are they will find what they need on the AS/400, but this could require deploying basic green-screen terminals or terminal emulation software throughout the organization. With the advent of a Java virtual machine running on the AS/400, the accounting application can be integrated with a Web browser interface, allowing the accounting staff to update records and run reports with the ease of surfing the Web.

One benefit that the AS/400 offers Java over any and all other operating systems is its single-level store. A single-level store implements a 1:1 mapping between memory and persistent storage. That is, all memory is persistent. All other operating systems will require programmers to serialize, or flatten, objects into a stream format in order to make them persistent. On the AS/400, Java objects can exist intact as full-fledged system objects that can be backed up, restored, replicated, shared, and secured with no additional programming. The increase in performance this can add to a Java application is currently immeasurable.

The AS/400's design can be only described as futuristic. Even the the Java virtual machine has established a default integer size over which it operates, specifically 32-bit. The AS/400 shields applications from this knowledge completely and is capable of executing applications equally well on either a 48-bit or 64-bit architecture without porting or migration efforts. The layer that performs this shielding is known as the technology independent machine interface, or TIMI. The current 1.0.2 technology preview release of the Java virtual machine is built on top of the TIMI layer, thus performance suffers greatly. The official 1.1 release of the virtual machine will be implemented below the TIMI layer allowing for superior control and performance of the Java virtual machine on the AS/400.

Another initiative expected to provide high-performance on the AS/400 is a direct execution static compiler, which will generate RISC machine code as well as Java machine code.

This will allow Java code to directly execute on the AS/400 system without the overhead of interpretation. In addition, high-performance garbage collection–automated cleanup of machine allocated memory–is also being developed for the AS/400 system.

As in the case of the OS/390, the AS/400 is a server platform with no consistent output console. Therefore, some mechanism must exist for implementing Java's AWT library. Under the current 1.0.2 technology preview implementation, AWT functionality is not included; however, AS/400 will also use the Remote AWT capability once it is available.

IBM has not ignored that there are functions on the AS/400 that will not be accessible from the standard Java 1.1 implementation. The AS/400 Toolbox for Java will be a key component of the AS/400 implementation of Java . This is a set of higher level classes and native code libraries for accessing data and core functionality of OS/400. These classes include support for:

- Java Database Connectivity (JDBC) access to AS/400's record-level databases. The AS/400 has a unique ability to allow applications to lock a single record at a time. Most database servers, including Oracle and Sybase, use page locking mechanisms, which lock out a group of records at a time.
- Access to DB2/400 through JDBC.
- Access to a hierarchical file system (a required entry) called the AS/400 Integrated File System (IFS).
- Execution of existing AS/400 applications complete with return values to the Java program.
- The ability to call any non-interactive AS/400 command.
- Access to the AS/400's keyed and sequential data queues. Data queues are one of the most useful native facilities of the AS/400.
- Access to the AS/400's print services inclusive of querying and administering print spoolers, as well as classes to generate SCS printer data stream directly from within a Java application.
- Integration with RPG and CL programming languages.
- Provision for the infrastructure needed to manage sign-on information, to create and maintain sockets connections to the AS/400 services, and to send and receive

data.

- Transparent conversion of data between AS/400 record format and Java objects. The Java-based data description classes provide a method of describing the record-format of a buffer as an object. This object is used to convert and carry the data.

For the past 20 years, the AS/400 has been providing many organizations with key business solutions. However, the AS/400 has yet to reach its maximum utility within the enterprise. With the advances in object-oriented programming, and requirements thereof, the *AS/400 is uniquely poised to become a cornerstone for data storage in the age of distributed object computing*. Combine this with the fact that the AS/400 will natively support Java, also a powerful object-oriented platform, and *IBM has an object storage solution that will rival any of the existing object database technologies and object repositories now in existence*.

The key to success of Java on the AS/400 will be performance of JDBC. Client/Server solutions using Open Database Connectivity (ODBC) with the AS/400 have not been overly successful due to poor performance and large overhead. Another key to success will be a well integrated system for making Java objects long-lived and indexed. That is, the AS/400's primary benefit is that Java objects do not require serialization. However, IBM has not defined how they will escape the natural process of garbage collection as well as provide AS/400 physical file indexes over these Java objects.

Java On OS/2 & AIX

OS/2 and AIX present more traditional departmental operating systems solutions from IBM. Current versions of AIX are being offered on a variety of RS/6000 hardware architectures, and OS/2 is IBM's Intel 80x86 operating system solution. Both operating systems represent strong departmental server platforms, in contrast with their siblings, OS/390 and AS/400, which represent centralized enterprise processing.

The AIX/Java story is straightforward. Java was developed with UNIX in mind, and IBM's UNIX is quickly becoming one of the most popular platforms around. Their RS/6000 architecture is known for its ability to scale extremely well, not to mention its ability to play chess.[2] AIX is IBM's premier Java platform already sporting a 1.1.2 JIT version version of the Java virtual machine with runtime optimizing compiler. Along with Netscape's Web server software, AIX represents a powerful Web server with server-side Java capability. It is also the only IBM platform for which the first release of San Francisco will be available.

[2] IBM's RS/6000 architecture was used to develop Big Blue, the application attributed with beating world famous chess player Kasparov.

Two additional benefits of using Java on AIX include:

- AIX integration with the Common Desktop Environment (CDE). This integration provides access to CDE data types and actions.

- Java on AIX supports AIX's Ultimedia Services (UMS) through native Java classes. UMS provides multimedia capabilities such as video, high-resolution graphics, and audio.

OS/2, unfortunately, may be known more for the rumors of its demise than the size and success of its installed base of users. The OS/2 platform is in use by IBM's top 2,500 customers running mission-critical applications. These users are now looking for ways to extend the current base of applications for Web and Internet/Intranet usage. IBM is supporting them by making the OS/2 Warp 4.0 and OS/2 Warp Server 4.0[3] products Internet savvy machines.

One advantage that OS/2 has in this departmental server space is its internal support for IBM System Object Model (SOM). SOM is still a core component of IBM's CORBA strategy, as well as being the foundation of Component Broker. Therefore, OS/2 is Component Broker ready. The OS/2 team is also providing SOM/Java Bean integration, providing a natural way for Java to integrate with native OS/2 functionality. This feature is akin to Microsoft's support of ActiveX and Java interoperability.

Additional Java Initiatives From IBM

San Francisco

IBM and the San Francisco team just release the 1.1.0 release of software which includes the foundation layer, common business objects, general ledger application, tools, and utilities. Platforms targeted for this first release are Windows NT and IBM AIX. While Java does offer the promise of write-one run-anywhere, and this is a pure Java solution, these two platforms were selected for two key reasons:

1) They have a viable Java virtual machine revision 1.1.2
2) Currently, there is no standard pure-Java application installation facility. Hence, distribution of pure-Java applications still suffers the pangs of platform specific releases. Of note, IBM's research division has made a preview of the pure Java Bean Bag installation facility available on its alphaWorks Web site (www.alphaWorks.ibm.com).

[3] OS/2 Warp 4.0 represents the desktop OS/2 version, while OS/2 Warp Server 4.0 extends from the line of LAN Manager.

The next two target platforms, slated for third quarter 1998, include AS/400 and HP-UX. Again, reliance upon the virtual machine 1.1.2 is a key factor in the time frames for these platform releases.

More information on the San Francisco project can be obtained via the SF Web site (www.ibm.com/java/Sanfrancisco). Also NC.Focus will be watching the SF project closely and reporting on it either in the *In The Final Analysis ...* newsletter or the *State Of Java Report*.

alphaWorks/Taligent

In the past, IBM has not been so forthcoming about the research it was conducting, let alone producing testable products to the general developer community. alphaWorks is a break with this past persona and a move to demonstrate IBM's true research power. Technologies made available on the alphaWorks Web site include:

- A pure Java application installation facility
- Java Bean development tools (Bean Extenders)
- SNA for Java allowing Java-to-Java communications natively over LU 6.2, also known as application-to-application program control (APPC)
- Aglet Workbench–agent technology based upon RMI
- Java-based collaboration tools

Earlier products available via this site included the much-touted WebRunner technology, now featured as part of the Visual Age for Java toolkit. Many of these tools stem from the work of IBM's Taligent division on the CommonPoint Application framework, which was developed to build OpenDoc components. This work has translated effortlessly into Java and Java Beans demonstrating that OpenDoc was not a fruitless endeavor. Starting the lessons learned from OpenDoc now would have delayed the advancement of Java Beans by at least another year. OpenDoc was ahead of its time, and now it seems to be reborn in the form of Java Beans.

Visual Age for Java

IBM's robust Java application development environment is available for AIX and OS/2 platforms as well as Windows 95/NT. Of all Java development tools, IBM's VA for Java leads with the greatest memory requirements–32 megabytes of memory. NC.Focus's early investigations into this excessive memory requirement yielded an unexpected answer. It seems that VA for Java and VA for Smalltalk share the same foundation. While the answer satisfied the

question “Why?” the excessive memory requirement may hinder adoption of VA for Java against other cross-platform Java development tools including SunSoft’s Java Workshop. Of note, this package supposedly supports the ability, albeit unexploited, to generate Java class files from Smalltalk code.

CICS

This paper has mentioned the CICS transaction processing monitor a number of times. It is central to IBM’s electronic commerce platform, and it is one of the most strongly integrated components with Java. CICS will support Java as both a client and as a facility for execution of transactions.

CICS Gateway for Java represents the component that will provide communications from an applet running in a Web browser to a CICS server without having to utilize a Web server. This design allows separation of the Web server and the CICS server so that the Web server will not become inundated with transaction processing requests. By the same token, companies relying on a highly scalable high-speed transaction processing environment do not want it weighted down by requests for Web pages.

Time Frames for Delivery

The following is the planned delivery dates for Java 1.1.x compliant virtual machines on its commercial operating systems.

	Alpha	**Beta**	**OS/Ship**
OS/390	5/97	9/97	10/97
AS/400	N/A	8/97	2/98
OS/2	N/A	6/97	7/97
AIX	N/A	N/A	6/97