

Preface

The Java** programming language is the language of choice for Internet-based applications that are designed to run unchanged on many, different computer platforms. Java's powerful capabilities are used more and more frequently to lower the cost of programming for diverse platforms while making the resulting products available to a wide range of customers and users. IBM uses the Java language to develop many of its application products for the Internet and intranets.

This issue features a variety of Java-based and Java-oriented environments, products, and tools from IBM and its Lotus Development Corporation subsidiary. Among the nine papers, topics range from an introduction to IBM's Java initiatives, to middleware for network programs implemented with Java, and to relationships between Java and other software systems. We are indebted to K. D. Gottschalk of the IBM Network Computing Software Division in Research Triangle Park, North Carolina, for his initiative and significant effort in developing and coordinating this issue.

Gottschalk presents an overview on how IBM is using Java in applications, infrastructure, and tools. A description of the architectural structure of IBM's implementations of Java is followed by discussions of the approaches used for components, server platforms, and distributed objects. Tools for Java and the importance of security are also discussed.

The paper by Brackenbury et al. describes how the IBM Enterprise Server for Java specification provides a standard model and set of services by using Enterprise JavaBeans** to meet current and future middleware requirements. The specification assumes a three-tier model for applications, consisting of a client tier, a Java application server tier, and a data and resource manager subsystem tier.

Bayeh describes a recently announced IBM product, the WebSphere Application Server*. The architecture for this server has its foundation in the Network Computing Framework and was developed to allow easy deployment of client/server applications on the Web. Because Web applications are becoming more complex and dynamic, and since Java provides many benefits as an application programming language, Java is the programming language of choice for Web applications. To better serve this environment, the WebSphere Application Server is based on Java technology.

As Java technology matures, security is a prime focus of many development efforts. Initially Java was a tool used in conjunction with Web browsers for small downloaded applications, or applets. This use prompted development of the first security measures to protect against the downloading of hostile code. As Java has evolved to encompass other uses, Java security has likewise evolved. The paper by Koved et al. describes how security has been handled from the early uses of Java up to the present, with a discussion of security requirements and future directions.

The San Francisco* project allows independent software vendors to concentrate on their own unique programming solutions by providing them with a common foundation, which they can specialize and extend to produce platform-independent software. Rubin, Christ, and Bohrer explain why San Francisco was written almost entirely in Java and what had to be considered as the product was developed.

Network computers allow applications to be downloaded on demand. Since there is only one persistent copy of these applications, maintenance and upgrade tasks are greatly simplified. The eSuite** product from Lotus is an integrated set of downloadable applications developed for network computers and personal computers. In his paper, Briggs de-

scribes how eSuite was designed to take advantage of a network-centric environment, making use of Java, the Internet, and intranets.

VisualAge* for Java assists programmers in rapidly developing Java applications by providing visual interfaces for developing, combining, and testing components. Chamberland, Lymer, and Ryman describe this set of tools and explain how they can be used effectively. Code generated with this product supports the JavaBeans** component model, allowing programmers to create components that can be reused in other tool environments.

In Java the validity of array references must be checked at run time. In their paper, Midkiff, Moreira, and Snir present results of their work to develop techniques that minimize the number of run-time tests that must be performed for Java array reference operations. They describe various optimization techniques and some experimental results. Although developed for use with Java, these techniques can be applied to other programming languages.

For distributed computing to be used successfully, especially with the development of very small smart devices, communication among all devices must be of the highest quality. Network middleware is the software layer that will enable such quality to be obtained. A Tuplespace system, which acts like a global communication buffer, can be used for delivering data in distributed computing networks. Wyckoff et al. describe T Spaces, a system based on the Tuplespace system, implemented in Java, that allows any program or system service to be available to any other program or service in a network.

The next issue of the *Journal* consists of papers on a variety of subjects, including IBM's Component Broker, improving existing measurement frameworks, performance of teams and tools in software development, and the thirtieth anniversary of IBM's Information Management System (IMS*).

Gene F. Hoffnagle
Editor

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Sun Microsystems, Inc. or Lotus Development Corporation.