



Document Library

Die InterLake GmbH bietet Ihren Kunden in der Document Library Zugriff auf interessante Informationen. Wir stellen Ihnen Inhalte unserer Partner und aus frei zugänglichen Online Quellen zur Verfügung, die mit unserer Tätigkeit und der IT- und Medienbranche zu tun haben. Bitte beachten Sie, dass die Inhalte dem Copyright der jeweiligen Herausgeber unterliegen und diese Inhalte nicht ohne Referenz auf das jeweilige Copyright weitergegeben werden dürfen.

Weitere Informationen zu InterLake und weitere Dokumente finden Sie unter

www.interlake.net

Die InterLake Document Library steht den Nutzern der InterLake.Network zur Verfügung:



InterLake engagiert sich ehrenamtlich beim Münchener IT- und Medienverband FIWM, dem unser Geschäftsführer Sven Slazenger als Vorstandsmitglied angehört, sowie in der Macromedia ColdFusion User Group Central Europe, die unter unserer Leitung seit 1998 ein deutschsprachiges Forum für über 700 Macromedia Entwickler in Deutschland, Österreich und der Schweiz bietet.

Weitere Informationen zu diesen beiden Initiativen erhalten Sie bei Sven Slazenger (slazenger@interlake.net)

InterLake offers its customers and business partners access to an extensive document library. We are offering information from our partners and have also compiled freely available papers from the internet that we consider of value to you. Please be advised that the copyright belongs to the issuer of the information and that you may not distribute this content without referring to these copyrights.

Additional information about InterLake and more documents can be accessed free of charge at

www.interlake.net

The InterLake Document Library is available through the websites of the InterLake.Network:

InterLake is also an active member of the non-profit Munich IT- and Media Association FIWM. With InterLake CEO Sven Slazenger we provide one of the board members of the FIWM. We are also the founders (1998) of the Macromedia ColdFusion User Group Central Europe, the German language forum for Macromedia Developers from Austria, Germany and Switzerland and one of the largest Macromedia Communities worldwide.

For further information please contact Sven Slazenger (slazenger@interlake.net)





J2EE Promises and Customer Experiences

Adding a Productivity Layer to
J2EE Speeds J2EE Benefits

*By Susan E. Aldrich, Senior Vice
President*

January 2003

Prepared for Macromedia, Inc.

Table of Contents

Great Expectations for J2EE	3
J2EE Promise	3
Using J2EE in Today's Environment.....	3
Adding a Productivity Layer to Realize J2EE's Opportunities.....	4
Achieve J2EE Goals with ColdFusion MX.....	5
Introducing ColdFusion MX	5
ColdFusion MX Benefits.....	7
ColdFusion MX Features	8
Conclusion.....	12

Illustrations and Table

Illustration 1. ColdFusion MX Productivity Layer for J2EE.....	6
Illustration 2. Java Training	7
Table A. ColdFusion MX Deployment Environments	8
Illustration 3. ColdFusion MX in J2EE Architecture.....	11

J2EE Promises and Customer Experiences

Adding a Productivity Layer to J2EE Speeds J2EE Benefits

By Susan E. Aldrich, Senior Vice President

Prepared for Macromedia, Inc. by Patricia Seybold Group, Inc.

Netting It Out

Java and J2EE are designed to increase software quality, application reliability, and security and to improve asset re-use, leading many organizations to commit to J2EE as their application platform. But, given today's pressures on IT to produce results daily and deliver value from the investments made during boom times, executives are increasingly impatient to reap J2EE's benefits. There is a gap between expectations and results.

The same delay in achieving productivity occurred when client/server emerged in the late 1980s and when minicomputers emerged in the late 1970s. In both of these cases, fourth generation languages (4GL) emerged to add a productivity layer to the application platform, providing a set of language abstractions and built-in services that shorten developer learning curves and accelerate the development of distributed Web and Internet applications.

Today, Java and J2EE productivity is being extended by new 4GLs, such as Macromedia ColdFusion MX for J2EE, that help organizations investing in J2EE to reduce costs and deliver results significantly faster than achieved by using Java alone for application development.

A key strength of the ColdFusion MX 4GL is the speed of achieving proficiency. Because ColdFusion MX is very easy to learn, developers of all backgrounds can quickly ramp up to participate in J2EE projects. Application services offered by ColdFusion MX, such as rich presentation and back-end integration, boost developer productivity, requiring far less code to be written as compared to Java. ColdFusion MX for J2EE can be deployed on the leading J2EE Application Servers, such as IBM WebSphere and BEA WebLogic, which heightens application reliability and offers seamless management of ColdFusion MX and its applications.

Great Expectations for J2EE

J2EE Promise

Flexibility and Reliability

Java is, without a doubt, the most widely-embraced technology of the Internet age, and J2EE promises to be its most enduring platform. IT developers and managers ardently praise this ideal architecture designed to provide a flexible, secure, and reliable foundation for IT systems of all types. It's no surprise that so many organizations have decided to standardize on J2EE. According to research completed by Fawcette Technical Publications in the fall of 2002, more than 44 percent of new applications are being built using J2EE. That percentage climbs to 51 percent over the next year.

Compared to earlier object-based languages, Java-based applications are more reliable, Java resources more reusable, and business processes are more flexible due to these key features of the J2EE platform:

- Component based model
 - Core runtime services
 - Standardized approaches to integration
 - Easier middleware development
 - Wide portability via the JVM
 - Manageability of the application platform and the applications
-

Bottom Line: Productivity

Yet, despite the great productivity features of Java and J2EE, that pesky development backlog just keeps getting longer, and business executives are not patient. In this tough economy, IT is pressed to make the most of J2EE investments to improve the company financial position. IT executives need ways to retain IT's reputation in the company by delivering on past promises and delivering value every day.

Using J2EE in Today's Environment

Proficiency in Java Is Limited and Costly

In short, many executives have come to realize they are not reaping all the benefits they expected. This gap between expectation and achievement is more onerous the longer it endures.

What has caused the delay in realizing J2EE's benefits? There are two primary reasons.

First, organizations typically don't have enough skilled Java programmers, or, to say it another way, their programmers are skilled in other languages such as Visual Basic, PowerBuilder, Perl, COBOL, ASP, PHP, and HTML. Many programmers are not trained in computer science and lack facility with object-oriented design. These developers are talented and have intimate knowledge of your systems, but they aren't in position to contribute to your J2EE efforts.

J2EE Promises and Customer Experiences

Java has a much shorter learning curve than its object-oriented predecessors, C++ and SmallTalk, but it still isn't easy to pick up. By Gartner's estimate from a May, 2002, study, learning Java takes 10 months full-time, including 60 days of training. The cost of this training averages \$52,000, not counting the developer's salary. Given the high level of investment involved, trained Java developers are aggressively recruited. There is no guarantee your investment won't take another job.

The Right Tool for the Job?

J2EE has addressed the problem of architecture and services for distributed, Web-based applications, but it hasn't solved the problem of speeding up the construction of those applications. This is the second factor in delaying the benefits of J2EE: the nature of Java itself. Java is a low-level system programming language, which means that developers write more lines of code to accomplish tasks as compared to high-level languages. Java and J2EE don't include the built-in services and abstractions that make Web developers productive.

Development teams need the same types of productivity tools for Java as those that sped client/server delivery. Productivity tools can be learned much more quickly than the base language because they provide wizards and guides, produce standardized components, and automate much of the development. For Web-based applications, this is particularly important. Web developers are typically working on presentation (the user interface) and using application data and logic from back-end systems. What they need is automated, high-level, and simple mechanisms for data visualization, search, and dynamic data retrieval.

Tools for Everyone

The variety of programmer backgrounds and IT projects necessitates a variety of tools and approaches. HTML and Perl programmers working on departmental or Web applications tend to have short projects and expect their applications to have a short life. High productivity, ease of hooking-in data and presentation, and quick results are the top requirements for these folks.

The Java, COBOL, and C+ system and infrastructure programmers working on EAI, core business systems, and middleware are working on longer-duration projects and are producing systems that may live for decades. Architecture, portability, and standards are the top requirements for these groups.

By adding a productivity layer to the Java and J2EE platform, you can achieve the quick results the Web developers need, while ensuring that the highly-portable and reliable architecture of J2EE supports the core applications and infrastructure.

Adding a Productivity Layer to Realize J2EE's Opportunities

New Tools for J2EE

A new class of products is emerging that adds a productivity layer on top of J2EE. These products give you the core of J2EE reliability and consistency without the complexity and steep learning curve of Java. Tools in this class bring 4GL-like features to Java, including high-level abstractions to the core J2EE platform, reusable built-in services,

and components that automate the most common tasks involved in Web programming. These integrated tools shorten the developer learning curve and increase developer productivity.

Shorten the Cycle

One of the key results of using these 4GL productivity tools is that application development projects complete more swiftly and produce higher-quality code.

Leverage Existing Skills

Perhaps the most significant result of adding a productivity layer to J2EE is that developers with years of business domain knowledge, but whose skills have prevented them from working on J2EE applications, can now contribute much more efficiently. They can quickly learn to use these new productivity tools and leverage the work of the system programmers without having to learn Java.

4GL Checklist

The list of customer requirements for the 4GL for J2EE touches on development features, ease of adoption, and generation of code that is innately well designed and efficient. Our checklist of 4GL capabilities is as follows:

- **Adoption**
 - Familiar concepts and syntax for developers familiar with Web protocols
 - Rich training and certification resources
 - Strong community of developer support
 - Wizards and Guides for difficult functions
 - Customization of Wizards and Guides to support local practices
- **Development**
 - Data Visualization tools for lists, charts, and other common representations
 - Dynamic data tools for query and presentation from any data source
 - Simple and reliable integration with XML, Web services, and Internet protocols
 - Rich Client presentation tools
 - Automation for commonly-created functions
 - Comprehensive scripting language to allow flexibility and extension of 4GL to work with Java assets
- **Deployment**
 - Well performing, efficient applications
 - Manageability for configuration, performance, deployment

Achieve J2EE Goals with ColdFusion MX

Introducing ColdFusion MX

Development Environment

Macromedia ColdFusion MX is a server scripting environment for creating Internet applications that provide rich presentation and connect to back-end data and applications. It is integrated with popular Macromedia Web development tools, Flash MX and Dreamweaver MX. ColdFusion MX automates and guides common coding activities through a high-level scripting language and easy-to-use component model. While developers do not work in the Java programming language, ColdFusion applications are

J2EE Promises and Customer Experiences

compiled to Java bytecode and run in a J2EE application container. ColdFusion MX is used by Web application developers, professional Web developers, and Macromedia Flash developers. See Illustration 1.

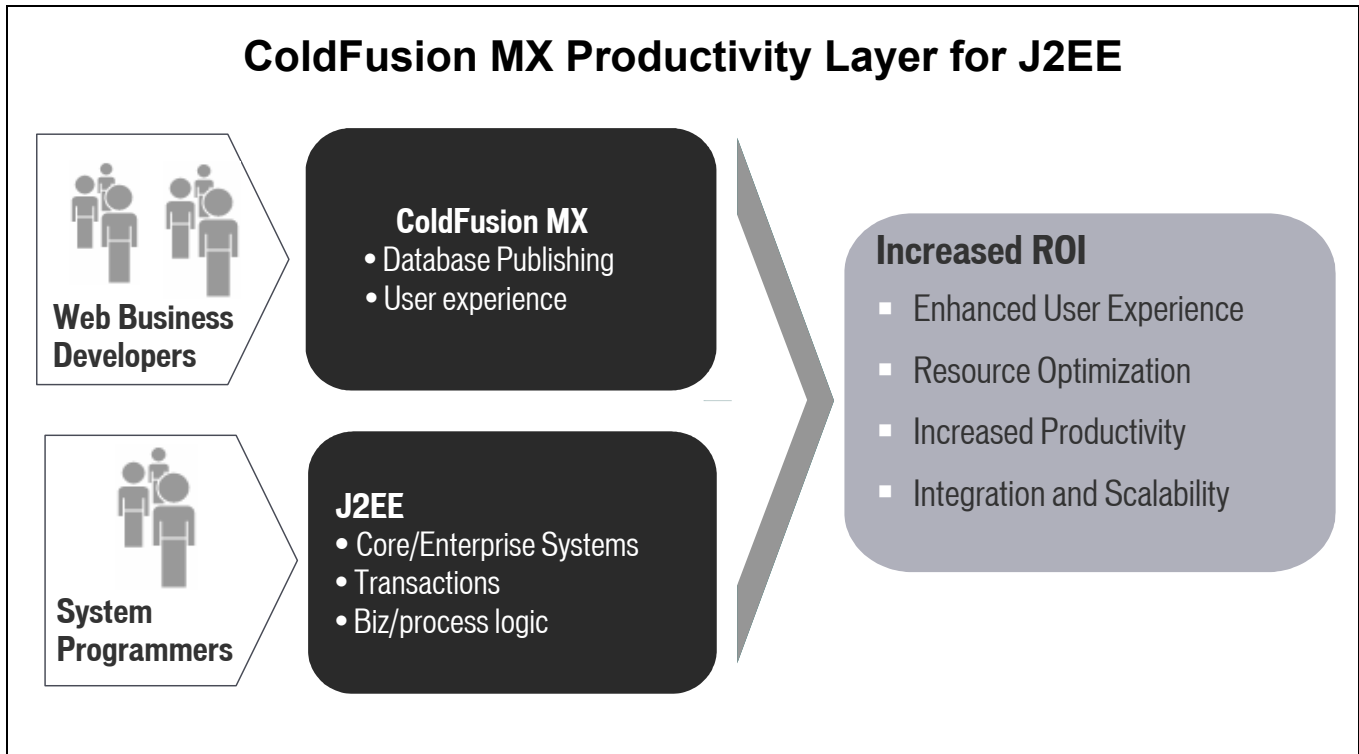


Illustration 1. ColdFusion MX delivers productivity that enables Web and departmental application developers as well as system programmers to contribute to J2EE applications.

Runtime

ColdFusion MX is a Java application that provides the runtime for ColdFusion MX applications, providing services such as database connectivity, full-text search, Web services, and rich presentation. ColdFusion MX can act as a standalone server (just as the original ColdFusion) using an embedded J2EE application server based on Macromedia's JRun technology. ColdFusion MX applications can also be deployed on popular high-end J2EE Application Servers, including IBM WebSphere, Sun ONE Application Server, and BEA WebLogic Server. These ColdFusion MX for J2EE solutions first shipped in September, 2002. In both cases, the core resource management features, such as connection pooling, thread management, and security, are provided by the J2EE platform itself.

Customers Are Winning with This Approach

Some customers are aggressively deploying 4GL for J2EE tools. One company that has been using ColdFusion MX for the internal systems that keep IT running, such as scheduling conference rooms, IT resources, and coverage duty, has recently standardized on WebSphere. It didn't make sense to rewrite the existing applications in Java, especially when they found ColdFusion MX so well suited to the relatively uncomplicated tasks of data presentation, maintenance, and workflow. The combination

of ColdFusion MX and WebSphere allowed this company to consolidate the infrastructure and capitalize on existing investments.

ColdFusion MX Benefits

Far Less Effort to Train

Because ColdFusion MX simplifies development of presentation and integration, it takes less time to learn than Java. Moreover, ColdFusion MX uses languages that are very similar to languages with which Web developers are already familiar. ColdFusion Markup Language (CFML) is a tag-based language with which HTML developers will be very comfortable. ColdFusion also provides more traditional scripting based on the ECMAScript/JavaScript syntax. Bottom line, developers can build and deploy applications on J2EE without knowing Java. A developer new to ColdFusion MX will need one week of training and will be proficient within four months. The comparable Java training is two months, with a total of ten months to reach proficiency. See Illustration 2.

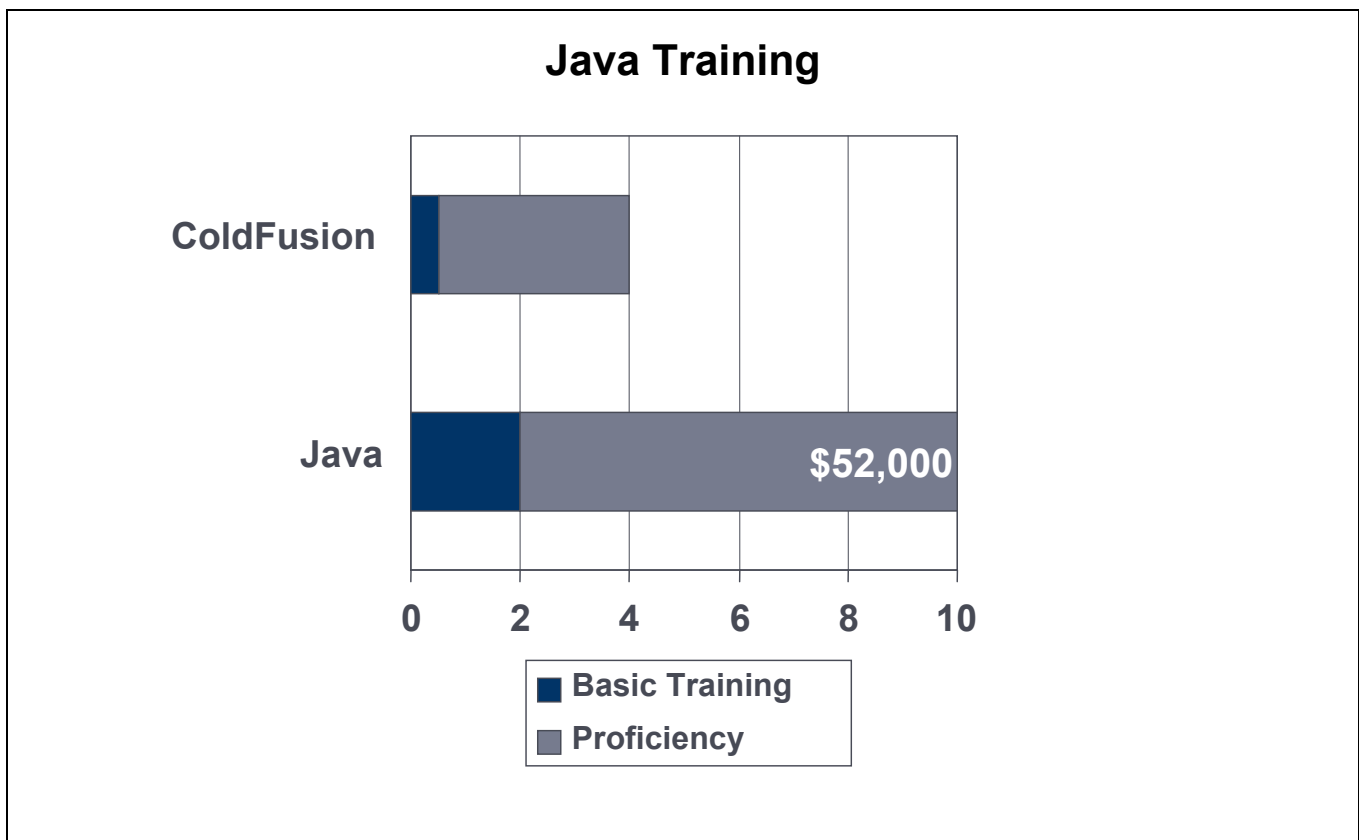


Illustration 2. The time to achieve proficiency and deliver value with Java is 10 months. For ColdFusion MX, it's four months.

J2EE Promises and Customer Experiences

Shorter Project Timeline

The services provided by ColdFusion MX eliminate a significant amount of code that would have to be written by developers working in Java. As a data point, Macromedia created the Java Pet Store reference application using ColdFusion MX and has documented that the ColdFusion MX Pet Store requires 88 percent less code than the Java Pet Store. While this doesn't reduce development time by 88 percent, it does mean that there is less code to review, edit, test, and maintain. As a point of comparison, the ColdFusion MX Pet Store requires 54 percent fewer lines of code compared to the .NET version of the Pet Store. The result of using ColdFusion MX is more efficient development, lower maintenance cost, and more projects in less time.

Suited to Web Development Tasks

ColdFusion MX's loosely-typed scripting, tag-based language, and Web application services were designed specifically to accelerate the construction of Web-based applications. Connectivity to a wide variety of back-end data sources and protocols is provided through high-level abstractions that take advantage of the underlying Java integration APIs. Moreover, by automatically transforming data into native ColdFusion types, ColdFusion greatly simplifies the task of integrating data from a diverse set of back-end resources. Lastly, ColdFusion provides rich presentation services for displaying information to the many types of clients now accessing Internet applications, including browser clients, rich applications built in Macromedia Flash, or Web services that use the SOAP protocol.

ColdFusion MX Features

Deployment Environments

ColdFusion MX for J2EE can be deployed on leading J2EE application servers and operating systems. Table A summarizes the supported configurations. To check for specific versions, you can go to:

http://www.macromedia.com/software/coldfusion/j2ee/productinfo/system_reqs/.

ColdFusion MX Deployment Environments	
J2EE Application Server	Supported Operating Systems
IBM WebSphere Application Server, Versions 4 and 5	Windows, Solaris, Linux, AIX
BEA WebLogic Server, Versions 6 and 7	Windows, Solaris, Linux
Macromedia JRun 4	Windows, Solaris, Linux, Mac OS X (development only)
Sun ONE Web Server 6.02 Sun ONE Application Server 7	Windows, Solaris, Linux

Table A. This table presents the deployment environments for ColdFusion MX.

Web Development

PRODUCTIVE SCRIPTING. ColdFusion MX scripting offers a high-level language that abstracts tasks for user-interface generation, database access, and application integration. The model-to-code translation provided by CFML allows programmers to accomplish application functions with less code.

PRESENTATION SERVICES. To speed the development of presentation interfaces for Web applications, ColdFusion MX includes Verity full-text search, dynamic charting and graphing, automated forms validation, and Macromedia Flash Remoting.

COMPONENTS. Developers can encapsulate standardized presentation services and back-end application integration into ColdFusion Components. The new ColdFusion Components allow developers to create components from virtually any type of source, encouraging reuse and promoting quality. ColdFusion Components are self-documenting, which further encourages reuse. Moreover, ColdFusion MX allows developers to incorporate custom tag libraries and import JSP tag libraries and components into their development environment and work with them in the same way they work with ColdFusion MX tags and components.

DREAMWEAVER INTEGRATION. For developers using Dreamweaver, the integration of Dreamweaver MX with ColdFusion MX provides powerful productivity improvements. These include integrated debugging and access to all of ColdFusion MX's Components, XML, Web Services, ActionScript, and presentation services from Dreamweaver.

Backend Integration

DATABASE. Database access is probably the most common type of back-end interface required by Web applications. ColdFusion MX includes JDBC and ODBC connectivity and supports in-memory queries, dynamic SQL statements, and stored procedures.

DIRECTORY, FILE, AND EMAIL SERVERS. ColdFusion provides built-in integration with the most common Internet protocols, such as HTTP, LDAP, FTP, SMTP, and POP. Using the power of the underlying services provided by the J2EE platform, ColdFusion abstracts these often complex APIs, making it easier for developers to incorporate them in the applications they build.

APPLICATION LOGIC. Access to application logic is the second most common integration requirement. ColdFusion MX, as a Java application, has seamless interfaces with your Java applications. Java resources, such as servlets and JavaBeans, can be invoked from ColdFusion MX pages and can run on the same server as ColdFusion MX. ColdFusion MX's CFOBJECT interface also simplifies invoking COM and CORBA objects from within their applications.

Web Services and XML

Web Services often make use of existing applications, exposing function for reuse by other applications. Using ColdFusion MX, developers can create a Web Service from any ColdFusion, Java, .NET, or COM resource with a single line of code and manage the execution via the embedded SOAP engine. One of the chief values of Web Services is connecting multiple application platforms. ColdFusion MX is unique in enabling developers to combine Windows and Java resources in a single Web Service.

J2EE Promises and Customer Experiences

ColdFusion MX's biggest contribution to Web Services development is its native XML interfaces and services. With ColdFusion MX, developers don't have to get involved with parsers, serialization, mapping, or transformation. These efforts are obviated by ColdFusion MX's built-in XML parser, which converts an XML document to a ColdFusion MX object, a ColdFusion MX tag that automatically generates a valid XML document from text and application variables, and built-in methods for XSLT style sheets and XPath searches.

Security

The Java Application Security Model provides the underlying security infrastructure for ColdFusion MX application security and resource management. ColdFusion MX provides key features for network, application, and server security, and also enables application teams to readily make use of their company's existing security services. The key features for each level of security are as follows:

- **Network Level.** ColdFusion MX supports standard encryption provided by network protocols and Web servers, including HTTPS and SSL.
 - **Application Level.** ColdFusion MX role-based security allows developers to secure component methods and pages and dynamically modify user interface based on the user's role. ColdFusion MX can extract role information from LDAP or NT Directories, as well as databases and other standard repositories. In addition, ColdFusion MX can use existing server and database permissions as the basis for authorizing access.
 - **Server Level.** ColdFusion MX offers sandboxing, a widely-used technique for constraining the databases and directories an application can access. Another approach to server-level security is to create multiple, separate instances of the ColdFusion MX runtime, thus isolating developers from production environments, for example, segregating developers working on a Customer Relationship Management system from developers building a human resources application.
-

Performance, Scalability, and Availability

ColdFusion MX applications are compiled into Java bytecode. This means that ColdFusion MX applications get the advantages of performance optimizations for Java Virtual Machines (JVMs) and that ColdFusion MX avoids the inherent inefficiencies of runtime interpreters used by other 4GLs or scripting environments.

In addition, ColdFusion MX has features to improve performance and scalability, including query- and page-caching, in-memory queries, database-connection pooling, dynamic load-balancing, just-in-time page compilation, and tools for profiling code and identifying bottlenecks. As a result, ColdFusion MX is considerably faster in benchmarks than previous releases of ColdFusion.

Lastly, the move to a J2EE-based architecture has increased the range of deployment options for creating high-availability ColdFusion environments. Taking advantage of the underlying J2EE platform, ColdFusion MX can be deployed into clusters of multiple server processes or a cluster of one or more physical servers. Both approaches have

advantages; clustering of server instances ensures high-availability and predictable performance, while isolating applications in individual servers increases security and reliability. In either approach, automatic server failover, server recovery, and service-level failover ensure the availability of ColdFusion MX, while load-balancing ensures performance. Moreover, ColdFusion MX is one of the few advanced environments that scale in both directions, offering both a very easy-to-install integrated package for the low end, and the high-reliability clustered environment for the high end.

ColdFusion MX in the J2EE Architecture

Because ColdFusion applications are, themselves, Java applications, they can easily incorporate Java assets and components, such as EJBs, servlets, or JavaBeans. As a result, Web developers using ColdFusion MX can leverage the work of system programmers and Java developers to gain access to core systems through well-defined interfaces, while taking advantage of the productivity tools and presentation capabilities provided by ColdFusion MX.

Alternatively, development teams can choose to implement the more complex elements of an application using low-level Java components—taking advantage of the greater control afforded by the Java language or the advanced services provided by an EJB container, while developing the bulk of the application in the high-level ColdFusion environment. The architecture of such a hybrid application is depicted in Illustration 3.

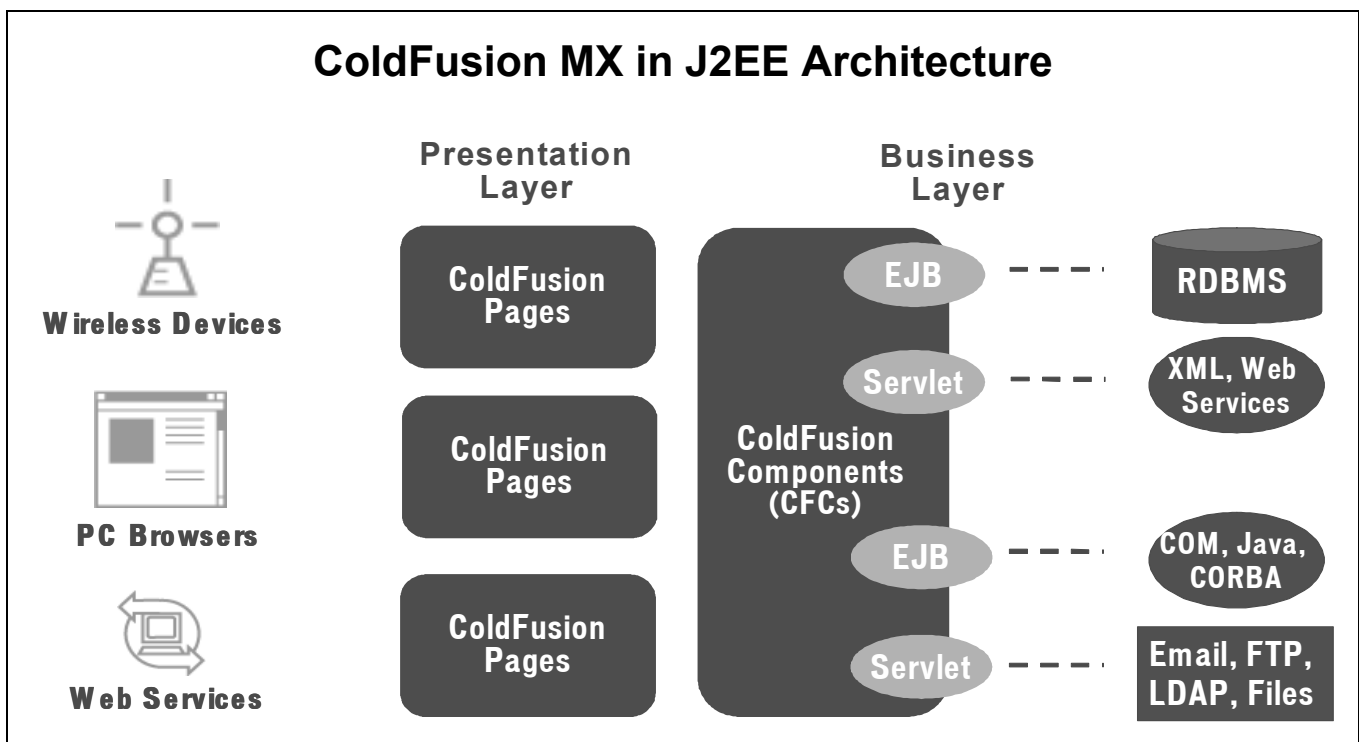


Illustration 3. ColdFusion MX is a J2EE application and thus provides seamless integration with assets developed by programmers using the Java language. As a result, Web developers can leverage the work of other development teams working in Java or choose to implement parts of their applications directly in Java.

Conclusion

Opportunity to Close the Gap

The features of ColdFusion MX offer executives an opportunity to close the gap between their expectations for J2EE benefits and their companies' progress in realizing them. The productivity features for developers let organizations leverage existing skills, turning any programmer into a J2EE contributor. This will be a big boost for IT's J2EE projects.

Because ColdFusion MX deploys on leading J2EE Application Servers, including WebSphere, WebLogic, and Sun ONE, managing ColdFusion MX applications will not impose new burdens on IT operations.

Organizations that already have ColdFusion, and have also standardized on WebSphere, WebLogic, Sun ONE or JRun, will definitely want to take a closer look and see how easy it is to migrate existing and build new applications on J2EE by using ColdFusion MX for J2EE.

Organizations that have J2EE, but are looking to boost productivity while reducing project time and costs, should consider ColdFusion MX for J2EE as a productivity layer on top of J2EE investments.

Take a Closer Look

Your next step is to take a closer look at ColdFusion MX for J2EE. You can get additional product information, a free download, and more to help you evaluate ColdFusion MX solutions by going to http://www.macromedia.com/go/j2ee_promise.