



July 2003

ColdFusion MX and the J2EE Architecture

© 2003 Macromedia Inc. All rights reserved.

The information contained in this document represents the current view of Macromedia on the issues discussed as of the date of publication. Because Macromedia must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Macromedia, and Macromedia cannot guarantee the accuracy of any information presented after the date of publication.

This Guide is for informational purposes only. **MACROMEDIA MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.**

Copyright 2003 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo, ColdFusion, the ColdFusion logo, Dreamweaver, UltraDev, Flash, JRun, HomeSite, Macromedia Generator, and Macromedia Spectra are trademarks or registered trademarks of Macromedia, Inc. Other brands may be trademarks or registered trademarks of others.

Macromedia, Inc • 275 Grove Street • Newton • MA • 02466

www.macromedia.com • info@macromedia.com • (617) 219-2000

Table of Contents

Table of Contents	3
Strengths and Weaknesses of the J2EE Platform.....	4
J2EE: A Powerful Application Platform	4
J2EE as a Rapid Web Development Platform	5
Summary.....	7
ColdFusion MX and the J2EE Platform	8
ColdFusion MX: Rapid Web Development on J2EE.....	8
ColdFusion MX: Meeting the Needs of Enterprise IT	9
Summary.....	11
ColdFusion MX: Technical Overview.....	12
ColdFusion MX Architecture.....	12
Developing ColdFusion Applications.....	14
Integrating ColdFusion with External Applications	16
Summary.....	17
Increasing ROI with ColdFusion MX.....	18
Optimizing Development Teams	18
Leveraging Open Standards.....	18
Enhancing the J2EE Infrastructure	18

Strengths and Weaknesses of the J2EE Platform

J2EE: A Powerful Application Platform

When it was first released in 1995, few could have predicted the enormous impact that Java™ technology would have on corporate IT and the high technology industry. During over eight years of development, Java has evolved from a web curiosity into a strategic technology foundation for IT departments worldwide. For instance, in a recent survey conducted by Giga Information Group, nearly 75% of large enterprises identified the Java 2 Enterprise Edition (J2EE) as a strategic technology platform.¹

Many reasons underlie the increasing adoption of J2EE as a strategic platform for application development, but a few of the most commonly cited include:

The Java Programming Language

As an object-oriented (OO) programming language, Java offers a powerful tool for constructing complex applications. At the same time, because it simplifies some of the constructs employed by earlier OO languages such as C++ or Smalltalk, Java presents a shorter learning curve and is less error-prone.

Rich Runtime Services

Developers building for the J2EE platform enjoy access to an array of runtime services that support applications, including low-level services such as thread management and database access to higher-level functions such as security and transaction support. By supplying a common set of infrastructure services, the J2EE platform has helped many companies reduce costs and simplify integration. With a standardized infrastructure, companies can consolidate their server investments and reduce management costs as well as more easily integrate their diverse applications using a common set of APIs and component interfaces.

Multi-Vendor Support

In addition to Sun, Java's inventor, dozens of vendors provide products that support Java technology, including BEA, Hewlett Packard, IBM, Macromedia, Oracle, Sun, Sybase, and many more. Moreover, these companies also participate in the evolution of the Java platform, submitting enhancements to the Java Community Process and voting on whether to include them in future versions of the specification.

As a result of this wide support and open evolution, companies investing in the J2EE platform are less susceptible to the ups and downs of the technology market and the fortunes of one particular technology vendor. More importantly, by basing applications on a widely adopted standard, they have greater access to complementary products and technologies.

¹ "IT Trends 2002, Midyear Update: Software Infrastructure for Application Integration and Deployment," Giga Information Group, April 2002.

J2EE as a Rapid Web Development Platform

While the J2EE platform provides a powerful general-purpose foundation for modern applications, web application development teams have additional requirements that are specific to the types of projects on which they work and the technologies used in them. The following section highlights some of these differences by describing a fictional (though all too realistic) intranet development scenario.

Canton Stewart - An Intranet Development Scenario

Canton Stewart (a fictional company) provides integrated logistics services to Midwestern manufacturing companies. With over 4,000 employees spread out among 40 different locations, the company relies heavily on its IT infrastructure to facilitate communication between offices and to coordinate the pick-up, warehousing, and delivery of its cargo.

Recognizing the vital importance of their IT infrastructure to its continued growth, Canton Stewart has recently standardized on the J2EE platform. The head of IT believes this move will provide an open infrastructure upon which the company can build out its application portfolio and better manage its rapidly growing business. As a result of this choice, all systems must begin moving toward the corporate standard.

One of the company's important systems is its intranet. Despite its small size, the intranet team has been very successful in delivering new applications to the business - including a wildly popular corporate portal, a customer service application that provides real-time access to shipment data, and an online training system for new employees. A key component to their success has been the ability to respond quickly to the needs of the business. Their typical project time is less than four months.

Bob, the intranet team manager, has extensive experience running development projects for both client/server and web systems. He has two developers on his team and another member focused primarily on design and layout of web pages. Neither developer has a traditional computer science background, but both have worked on a variety of web projects using scripting technologies like ASP, ColdFusion, Perl, and CGI.

The new corporate J2EE standard presents Bob's team with a difficult challenge. While the consolidation around J2EE will make it easier for Bob's team to integrate with the company's back-end systems, none of his team has any experience developing Java applications. Bob can request budget to send his developers to training, but he is concerned it will take them a long time to master the technology. Moreover, they still have to deliver a new reporting system to the sales team, a travel request system for operations, and an inventory management system for the fleet maintenance department.

As if these transitional issues were not enough, Bob's biggest concern is that the move to J2EE will permanently slow down his development team. While Java, JSP and servlets provide a powerful foundation on which they can develop their applications, they require a lot more low-level coding than the scripting technologies his team is accustomed to using in their projects. To address this concern, Bob has asked his team to research additional tools and technologies they might use to maintain their high level of productivity.

Supporting Rapid Application Development

As the Canton Stewart scenario illustrates, the J2EE platform does not necessarily meet the needs of all development shops. For many teams, maintaining productivity is of vital importance. To successfully meet the needs of his growing company, Bob's team must deliver new applications within weeks, not months - especially in a tight economy where resources are more constrained than ever. Only by adopting technologies that support rapid application development (RAD) can the IT organization hope to fulfill the ever-increasing demand for new applications that help automate the flow of business.

Yet while the J2EE platform provides the foundation for robust Internet applications, it does not by itself support the kind of RAD development his team needs. This is primarily the result of its reliance on low-level APIs and the Java programming language. While easier to learn and use than C++, Java is still a system programming language, also known as a third-generation language (3GL). As such, Java does not offer the RAD capabilities companies enjoyed with an earlier generation of client/server tools.

The fourth-generation languages (4GLs) of the client/server era enabled rapid development in two ways: by encapsulating the most common development tasks within high-level scripting languages and by providing tightly integrated development tools that automated routine development tasks. To fully take advantage of the opportunities presented by Internet applications, companies require additional development paradigms that take advantage of the powerful J2EE infrastructure yet enable rapid delivery of Internet projects through higher-level tools and languages.

Providing Specialized Services

The types of applications that Bob's team is working on (portals, reporting, and online training) highlight another important aspect of web development. In addition to the general requirements of any development platform - data access, basic i/o, and error handling - there are many aspects of web application development that are unique to the Internet domain.

For instance, there are a number of higher level services that have become common to Internet sites and applications. Virtually every web site provides a search interface so that users can quickly locate the information they need, and data visualization (charts, graphs, etc.) is rapidly becoming a common feature of browser-based business applications.

A more recent requirement is the desire to integrate dynamic data and back-end business logic with rich presentation technologies like Macromedia Flash. By providing a rich user interface, developers can create more responsive and usable applications. This is particularly true for instructional applications such as e-learning, but valid for more line-of-business oriented applications as well.

These types of specialized services are not provided by the J2EE platform. While it is possible to assemble them from best-of-breed suppliers or through custom development, doing so increases the cost and complexity of development. To remain productive and to manage costs, development teams increasingly expect that these services be provided as an integrated part of the environment.

Enabling a Wide Range of Developers

Lastly, the Canton Stewart scenario dramatizes a vital management issue. In an effort to simplify, IT departments are moving to standardize their application infrastructure. Yet while many have chosen the J2EE platform as their standard, few can afford the significant time and money it takes to train all of their developers in the complexities of the Java platform. According to the Gartner Group, it can take up to 10 months for a skilled Visual Basic developer to become proficient in Java, a transition that can cost more than \$52,000 per developer in direct costs and lost productivity.²

Particularly for developers building less complex applications - online publishing, database reporting, and data entry applications - the cost and time required to master the Java platform are simply not justified. These teams require development technologies that enable them to quickly become productive and deliver results, without incurring unmanageable costs. At the same, these technologies must also provide room for growth so that developers can expand their skills and meet the requirements of more complex projects should the need arise.

Summary

While the J2EE platform provides a powerful general-purpose foundation for many types of applications, it does not meet all of the needs of web development teams. Achieving productivity in web development requires a specialized development platform. Such a platform must be tailored to the skills and needs of the developers building the applications. It must support the unique requirements of web development by providing specialized services for Web applications. And lastly, it must meet the needs of IT management, enabling the IT organization to support the business through the timely and cost-effective delivery of new solutions.

² "The Cost and Risk of Application Development Decisions," Gartner Group, May 2002.

ColdFusion MX and the J2EE Platform

With the release of ColdFusion MX, Macromedia is delivering the type of web development solution needed by companies that have standardized on the J2EE technology platform. For the first time, developers can enjoy the productivity and built-in services that are the hallmark of ColdFusion while taking full advantage of the power and flexibility of the J2EE platform.

ColdFusion MX: Rapid Web Development on J2EE

Macromedia ColdFusion is a proven leader in rapid web application development. Used in more than 75 of the Fortune 100, recent surveys show that ColdFusion is one of the top two web development technologies at use in companies today.³ With the help of ColdFusion, companies like Allied Office Products, Bank of America, Boeing, Fodors, Hertz, Jaguar, and Simon & Schuster have built customer service applications, online publishing systems, e-learning solutions, business reporting applications, business-to-business extranets, and much more.

With the release of ColdFusion MX, these same benefits are available to IT organizations that have standardized on the J2EE platform – whether it is Macromedia JRun, IBM WebSphere, Sun ONE, or BEA WebLogic. ColdFusion MX enables companies to extend their existing Java servers by adding the ability to develop and deploy ColdFusion applications on their J2EE server environment.

With ColdFusion MX, companies can enjoy the benefits of rapid web application development while still simplifying integration and consolidating server infrastructure. The end result is a higher return on investment through increased flexibility and responsiveness as well as reduced costs of development and administration.

Rapid Server Scripting

Developers working in ColdFusion MX build applications using the ColdFusion Markup Language (CFML), a highly evolved scripting language designed specifically for web development. CFML encapsulates the most common web programming tasks in a high-level, tag-based scripting language, insulating developers from the mundane, repetitive tasks often associated with web programming, such as building output loops, interfacing with low-level web protocols, or working with session and form data. As a result, developers have less code to write overall, and can focus the majority of their time on building the business logic of the application, as opposed to low-level plumbing.

Macromedia's industry-leading web development tool, Dreamweaver MX, provides the other essential part of this rapid development solution. Tightly integrated with ColdFusion MX, Dreamweaver MX combines the best of visual page design with robust code editing and application assembly. Working with Dreamweaver MX, ColdFusion developers can quickly develop and debug their applications and easily access reusable components and web services from the server.

³ "Application Infrastructure Report," Yankee Group, October 2001

Integrated Services

In addition to the ColdFusion scripting environment, ColdFusion MX also provides a suite of integrated services that address the specific needs of web applications. For example, developers working in ColdFusion MX can take advantage of the integrated Verity™ full-text search engine to provide high-performance searches of both document and database content. In addition, developers can use the built-in charting and graphing engine to produce professional quality business charts on demand or on a scheduled basis.

ColdFusion also provides integrated support for standard Internet protocols, including server-side HTTP requests, sending and retrieving email, parsing and generating XML, and working with web services. By providing high-level interfaces for these protocols, ColdFusion makes it easy to exchange data or transfer information with remote systems - without having to get lost in the low-level details of individual Internet protocols or integrate third-party components by hand.

Lastly, ColdFusion includes integrated support for Macromedia Flash Remoting, a revolutionary new technology for building rich Internet applications. Using Flash Remoting, developers building Macromedia Flash content or desktop applications can easily request data or invoke business components on the ColdFusion server. As a result, ColdFusion MX makes it easy to build applications that combine the reach of the Internet with rich client-side logic on the desktop.

Together, these integrated services provide a powerful foundation for rapidly assembling high-impact web applications that take advantage of the underlying J2EE platform but also provide innovative capabilities that are specifically tailored to the needs of web applications.

Approachability and Extensibility

Because ColdFusion MX utilizes a high-level scripting language, users familiar with any programming or scripting language - JavaScript, PHP, ASP, or Visual Basic - can quickly master ColdFusion and begin building applications for the J2EE platform. A typical developer can learn the basics of ColdFusion in about a week and become proficient in less than two months.

Yet while ColdFusion is easy to learn, it also provides powerful programming capabilities for more advanced developers, including a component model for creating reusable application objects, facilities for custom error handling, as well as a role-based security framework. Moreover, developers can easily extend the ColdFusion environment with Java servlets, JSPs, and EJBs as well as integrate closely with web services, COM objects, or CORBA components.

By combining an easy to learn and highly productive scripting environment with a powerful suite of built-in services, ColdFusion MX provides the ideal environment for companies seeking rapid web development capabilities for the J2EE platform.

ColdFusion MX: Meeting the Needs of Enterprise IT

In addition to its benefits in development, ColdFusion MX also provides the performance, manageability, and security enterprises need from their applications. ColdFusion MX also cleanly integrates with a company's

existing enterprise infrastructure, allowing companies to leverage their investment in existing assets while building new applications.

Integrating with Enterprise Systems

ColdFusion MX can be deployed on Macromedia JRun 4, a fully J2EE-compliant Java application server, or on any of the leading third-party servers, allowing companies to leverage their investment in this powerful infrastructure. In addition, ColdFusion MX supports a variety of enterprise operating systems, including Windows, UNIX, and Linux.

ColdFusion MX also provides built-in connectivity to variety of back-end systems, including enterprise databases, web servers, and LDAP directories as well as email, FTP, and file servers. For integrating with legacy or packaged applications, ColdFusion MX supports the leading component interfaces (COM, JavaBeans, and CORBA) and provides APIs for developing custom extensions in Java or C++. ColdFusion MX can also take advantage of any legacy integration services that are part of the application server on which it is installed. As a result, developers working with ColdFusion can easily access a variety of back-end systems and deliver all of the data and functionality their users need.

Delivering High Performance and Availability

A frequent concern with developing applications using scripting is performance. Many scripting languages such as ASP, Perl, PHP, and Tcl are interpreted at runtime, resulting in significant overhead. This is not the case with ColdFusion.

Once they have been fully deployed, applications developed in ColdFusion are compiled into Java bytecode, just like applications developed completely in Java. As a result, ColdFusion takes full advantage of the performance optimizations available with the latest Java virtual machines (JVM), such as the HotSpot™ technology from Sun or the high-performance JVMs available from IBM and others.

Complementing this underlying infrastructure, ColdFusion MX also provides its own performance-enhancing features, including query and page caching, support for in-memory queries, and built-in facilities for profiling code and identifying bottlenecks.

As a result of optimization for the J2EE platform, ColdFusion MX delivers significant performance gains over previous releases of ColdFusion. Internal benchmarks demonstrate a performance increase of up to 150% compared with ColdFusion 5.

Moreover, because ColdFusion MX is deployed within a standard J2EE container, the entire ColdFusion environment is under the control of the application server administrator, who can start and stop the application as well as control access to server resources, such as memory space or processors. ColdFusion MX can also take advantage of the high-availability features of the application server, including isolating applications in separate server processes as well as load balancing among server processes or among separate physical servers.

Ensuring Application Security

Finally, ColdFusion MX includes powerful resources for ensuring applications provide the appropriate level of security. With full support for

SSL encryption between server and client as well as between servers, ColdFusion can protect sensitive data as it crosses the network. In addition, ColdFusion MX provides a flexible role-based security framework that enables developers to leverage existing database or LDAP directories to authenticate and authorize users before they gain access to applications.

Lastly, ColdFusion provides integrated server sandboxing - allowing administrators to host multiple applications on a single ColdFusion server while restricting their access to other resources. Sandboxes can restrict access to virtually any server resource, including databases, directories, ports, and even individual language elements or tags.

Summary

With a highly productive and easy to learn scripting environment, integrated development tools, and a wealth of built-in services - all running atop the powerful J2EE platform - ColdFusion MX provides a powerful foundation for developing web applications. Using ColdFusion, companies can accelerate web development on the J2EE platform while providing the security, performance and reliability required by today's applications.

ColdFusion MX: Technical Overview

The following sections provide a technical overview of ColdFusion MX when deployed in the J2EE configuration. The first section explains how ColdFusion MX can be deployed on a J2EE application server, while the subsequent sections explore the internal architecture of a ColdFusion application as well as the facilities available for integrating with external assets, such as Java and COM components, XML data, or web services.

ColdFusion MX Architecture

Deploying ColdFusion MX on a J2EE Server

ColdFusion MX is a standards-based J2EE application. When deployed in its server configuration, ColdFusion MX includes an embedded J2EE runtime. This configuration provides users with simplified installation and management, while taking advantage of many of the J2EE environment's strengths.

However, the full benefit of the ColdFusion MX architecture is realized only in the J2EE configuration. Here, ColdFusion MX takes full advantage of the J2EE architecture and the runtime services provided by the Java application server on which it is installed (see Figure 1).

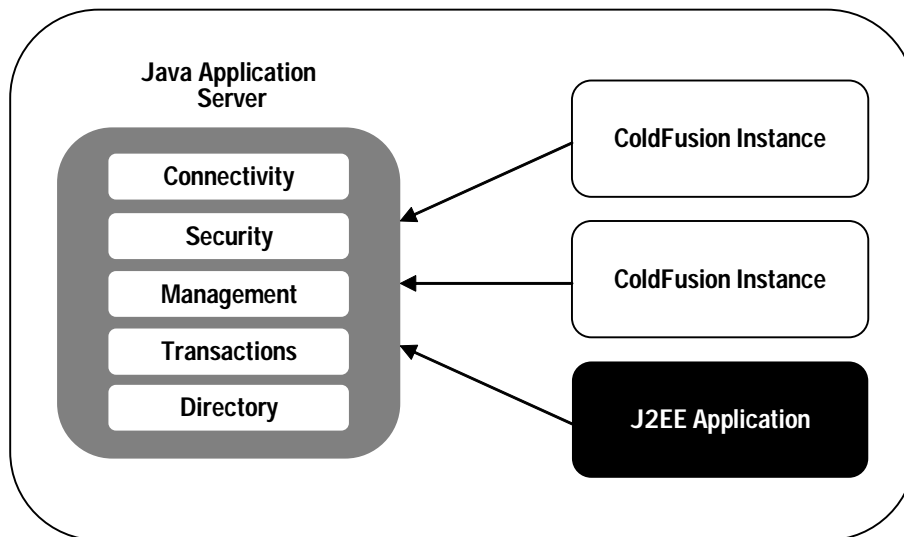


Figure 1. ColdFusion MX takes advantages of the same resources as other Java applications running on the application server.

For instance, in the J2EE configuration, administrators can create multiple instances of ColdFusion on a single server, thereby separating applications into separate processes. This not only reduces the possibility of errors in one application affecting another, but also increases security by ensuring that developers do not have unauthorized access to data being managed by other applications.

ColdFusion MX Enterprise includes the full infrastructure needed to run ColdFusion in the J2EE configuration, including a full version of Macromedia JRun, a fully J2EE-compliant Java application server. However, organizations can also choose to deploy ColdFusion on third-

party servers such as IBM WebSphere, BEA WebLogic, or the Sun ONE Application Server.

ColdFusion MX Runtime Services

Figure 2 outlines the internal architecture of the ColdFusion MX runtime environment. At its heart is the ColdFusion language runtime. This piece of server infrastructure both compiles ColdFusion applications into Java bytecode as well as provides runtime services for managing their execution. For example, the ColdFusion runtime provides services for interacting with common Internet protocols such as HTTP, SMTP, POP, and FTP as well as services for handling data transformation and sorting.

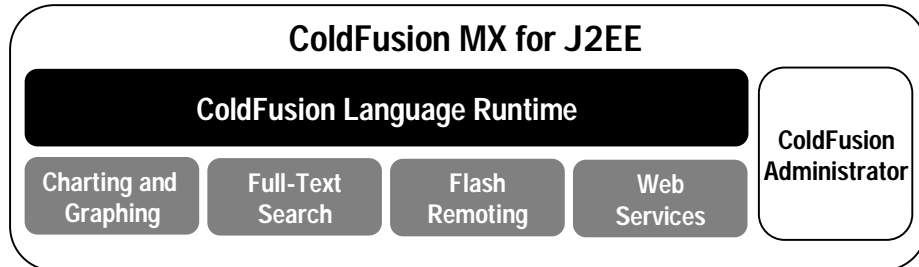


Figure 2. ColdFusion MX includes the ColdFusion language runtime, which compiles ColdFusion applications into Java bytecode, as well as several built-in services frequently used in web and rich Internet applications.

In addition to the lower-level services in the ColdFusion language runtime, the ColdFusion MX environment includes a number of high-level services that support ColdFusion applications. These include a charting and graphing engine, full-text search provided by Verity K2, as well as an embedded web services engine based on Apache Axis.⁴ All of these services can be accessed directly from the ColdFusion scripting environment through high-level tag interfaces that make them easy to use within any ColdFusion application.

Finally, ColdFusion MX includes an easy to use browser-based administrator. This utility allows authorized users to configure the ColdFusion runtime environment itself, including adjusting security settings, creating new data sources,⁵ archiving applications, and setting up application monitoring. The ColdFusion administrator does not interfere with configuration settings managed by the Java server administrator.

Deploying ColdFusion MX Applications

Similar to a web site or a Java web application, a ColdFusion application consists of one or more files that are stored within a specified set of directories on the server. Deploying a new application or updating an existing one is as simple as copying the files to the appropriate server directory. The ColdFusion MX environment automatically the new files and compiles the new or altered pages and components.

⁴ Macromedia is a key contributor to the Apache Axis project, an open source web services engine built to work with the Simple Object Access Protocol (SOAP).

⁵ The ColdFusion administrator enables developers to create, modify or delete data sources using the built-in type IV JDBC drivers. A ColdFusion application can also access data sources created by the application server administrator, but these sources cannot be modified from within the ColdFusion administrator.

When a user requests a web page that is part of a ColdFusion application, the web server passes the request to the web container containing the ColdFusion runtime. This interaction is identical to the processing of applications built with JavaServer Pages (JSP) or Active Server Pages (ASP).

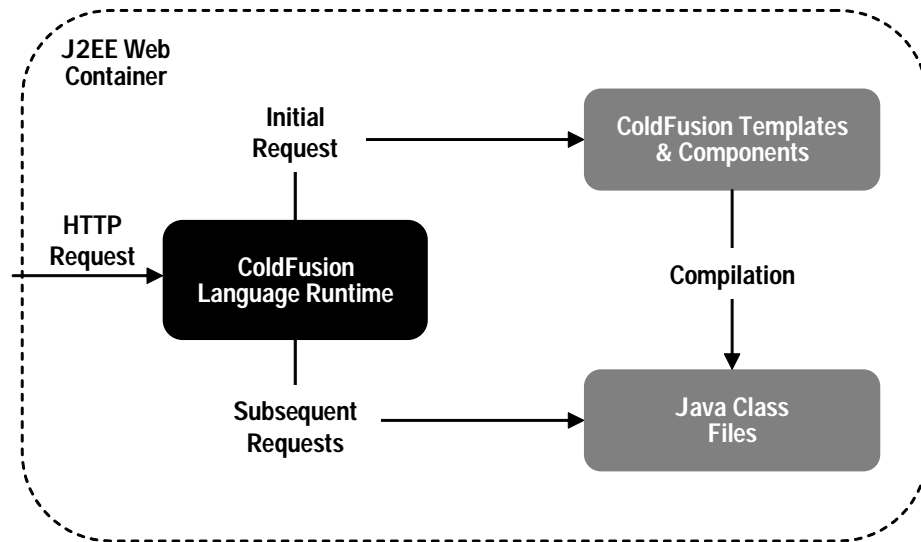


Figure 3. HTTP requests are passed by the web server to the J2EE web container running ColdFusion MX. Upon the initial request, the corresponding template or component is compiled into Java bytecode, which is then cached to memory/disk for use with all subsequent requests.

As pictured in Figure 3, after the initial request, the ColdFusion page is compiled into Java bytecode, at which point it is executed and then cached to memory and disk.⁶ All subsequent requests for that page are processed by the cached bytecode, unless an update to the page forces the page to be recompiled.

Developing ColdFusion Applications

In terms of physical architecture, a ColdFusion application is similar to a web site or a JSP application. Developers store applications as files on the server, and users access these files via a URL. Also like JSP, when a ColdFusion application is deployed, its files are compiled into Java bytecode and then executed. By contrast, PHP applications are interpreted each time the page is requested from the server.

While developers can create the source files for a ColdFusion application using any text editor, Macromedia Dreamweaver MX provides the ideal development tool for building ColdFusion applications. Dreamweaver MX combines a robust code editor with powerful visual design tools. It also provides complete support for ColdFusion development, including code completion, property inspectors, tag editors, a component browser, integrated debugging, and built-in help.

⁶ It is also possible to precompile all pages and components in an application as they are being deployed through the use of a batch utility provided by Macromedia.

ColdFusion Templates and ColdFusion Components

There are two primary types of files in a ColdFusion application. ColdFusion templates or pages (which have a *.cfm extension) are typically used for the presentation layer and generally contain both HTML for formatting as well as CFML for generating dynamic output, such as the results of a database query.

Developers also use CFML to create ColdFusion Components (which have a *.cfc extension). As in other languages, components encapsulate application logic and expose it through high-level methods. ColdFusion Components can be called by other ColdFusion templates or components, or their methods can also be accessed via web services or Macromedia Flash Remoting. The ColdFusion runtime generates the necessary proxies and automatically handles the necessary data marshalling between these the SOAP protocol and components. Like servlets or EJBs, ColdFusion Components are typically used to implement the business logic layer of an application and provide a powerful tool for reusing application logic within an application and across different applications.⁷

Together, ColdFusion templates and components provide an easy-to-learn yet powerful framework for rapidly developing web applications. While beginning developers can easily get started creating simple applications, ColdFusion also provides advanced developers with the infrastructure they need to create complex applications with a robust and maintainable design. For example, Figure 4 illustrates how to implement a Model-View-Controller (MVC) architecture using ColdFusion MX.

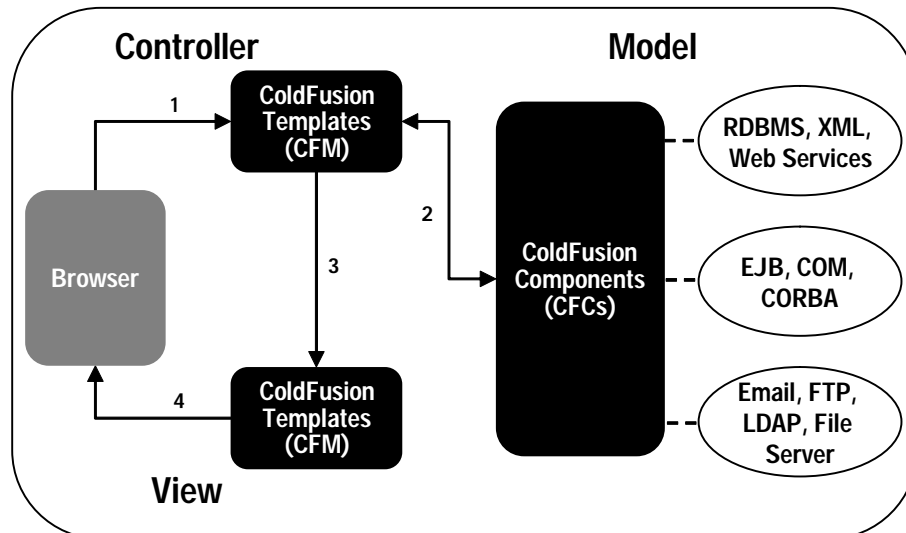


Figure 4. Implementing an MVC Architecture in ColdFusion. Users send requests to a ColdFusion template (CFM) acting as the controller (1). The controller invokes the appropriate methods on the application's components, which represent the model layer and insulate the controller from the underlying data layer (2). Once the action is complete, the controller forwards control to another template (3). This view template then generates the appropriate output (HTML, XML, Macromedia Flash, etc.) so that the user can see the results (4).

⁷ In addition to components, developers can also extend ColdFusion by creating custom tags and user-defined functions. These are typically used for more granular utility functions.

Integrating ColdFusion with External Applications

While ColdFusion MX can be used as a self-contained environment for developing complete web applications, it also provides a number of mechanisms for integrating with external systems and components, such as Java, COM or CORBA components, XML, and web services.

Integrating ColdFusion Applications with JSPs/Servlets

As discussed previously, ColdFusion MX applications are executed within a J2EE web container. As a result, ColdFusion applications provide very strong integration with JSPs and servlets, making it easy to build hybrid applications that combine servlets and JSPs with ColdFusion pages and components.

For example, a ColdFusion page can include a servlet or JSP or forward requests to a JSP or servlet; similarly, servlets and JSPs can include CFML pages or forward control of a request to them. Using this capability, developers can reuse existing assets or choose different technologies for different sections of the site - yet still easily share application data (e.g., form variables or data stored in the request, session, application scopes) between the pages and components.

Java tag libraries provide another method of integrating ColdFusion with assets developed in Java. Similar to custom tags in ColdFusion, Java tag libraries provide a means of encapsulating custom logic in a high-level tag interface. Because it is itself a Java application, ColdFusion MX allows developers to import Java tag libraries and reuse them within a ColdFusion application. As a result, developers can take advantage of the many free libraries available from the Java community as well as collaborate with Java developers within their organization that have built JTL interfaces into their reusable components or services.

Integrating with Java, COM and CORBA Components

In addition to integrating with web technologies, ColdFusion MX also provides rich connectivity with higher-level business components. Using a simple tag-based interface, developers can work with JavaBeans, Enterprise JavaBeans, COM components, or CORBA objects. ColdFusion allows developers to instantiate object instances, get and set properties, invoke methods, as well as store objects in the application or session scope.

Working with XML Data

While working with component interfaces enables tight integration with existing assets, ColdFusion also supports more loosely coupled forms of integration. XML and web services are rapidly emerging as the standard mechanism for integrating heterogeneous systems as well as exchanging data between companies.

ColdFusion MX provides robust support for working with XML data. The built-in XML parser in ColdFusion MX automatically processes XML and converts it into a native ColdFusion object, allowing developers to work with XML data just like any other complex data type in ColdFusion. This makes it easy to search XML documents for specific information, display XML data in HTML or Macromedia Flash, or apply transformations based on XSL style sheets. ColdFusion MX also provides native support for

XPath expressions, allowing developers to quickly search through complex XML data to extract specific information.

ColdFusion MX also makes it easy for developers to generate XML output. Building XML documents with other programming environments typically requires an understanding of the Document Object Model (DOM) and its associated API. Developers must execute multiple function calls to create nodes, set their values, and then serialize the object to generate an XML string. By contrast, ColdFusion MX provides a single tag that automatically generates a valid XML document from text, application variables, and the output of CFML code embedded within its body. As a result, developers can quickly and easily exchange data with other applications using standards-based XML.

Working with Web Services

Building on the success of XML, web services are rapidly becoming the preferred means of application integration, either within companies or between business partners. ColdFusion MX makes it easy to expose application logic as web services as well as to invoke web services created with other technologies. As discussed previously, any method on a ColdFusion component can be exposed as a web service, simply by setting a parameter. The ColdFusion MX runtime automatically publishes the WSDL needed by web services clients and handles all of the marshalling of data between ColdFusion and the SOAP protocol.

Invoking a web service from within ColdFusion is equally straightforward. Developers simply provide the location of its WSDL description and specify the appropriate method name and arguments. Underneath the covers, ColdFusion generates the client proxy and handles the data marshalling and interactions with the SOAP protocol.

Summary

ColdFusion MX delivers a flexible, high productivity environment that also provides tight integration with the enterprise infrastructure. The ColdFusion Markup Language (CFML) and integrated services accelerate web development by encapsulating low-level details in a high-level scripting syntax and providing easy access to powerful services like full-text search, charting, and Macromedia Flash Remoting. At the same time, ColdFusion MX takes full advantage of the underlying J2EE platform, providing high performance and reliability as well as rich extensibility via custom tags, components, JSP/servlets, EJBs, COM components, web services, and XML.

Increasing ROI with ColdFusion MX

As the previous sections have shown, ColdFusion MX enables companies to accelerate their web development initiatives with a highly productive scripting language, integrated tools and a powerful suite of built-in services. As a result, companies can experience the benefits of rapid web application development while taking advantage of the powerful Java platform.

Optimizing Development Teams

Despite its many strengths the Java language and J2EE platform do not meet the needs of all developers. By providing rapid web application capabilities on a powerful J2EE infrastructure, ColdFusion MX enables companies to better optimize their development resources.

With ColdFusion MX, building a well-architected J2EE application no longer requires advanced knowledge of Java and its underlying APIs. As a result, business application developers can quickly become productive and build well-designed applications - while still enjoying the performance, management and integration benefits of a J2EE infrastructure. This not only makes these developers more productive but also allows the organization to focus its most advanced resources on high-end applications that demand their skills.

ColdFusion MX also enables web developers to more easily participate in large-scale development projects. One of the challenges of developing web applications in the J2EE architecture is integrating the work of back-end Java developers with the user interfaces built by front-end designers. By providing tag-based access to back-end Java resources, ColdFusion MX makes it possible for front-end developers to integrate their designs with the business components developed by the system-level Java programmers. As a result, both sides can spend less time integrating their work and more time delivering new functionality.

Leveraging Open Standards

ColdFusion MX is built on the open J2EE platform and provides rich integration with enterprise technologies, including databases, directories, COM, CORBA, XML, and web services. By supporting a variety of back-end technologies, ColdFusion enables companies to fully leverage their existing investments in enterprise technology. In addition, because ColdFusion MX is built on the widely supported J2EE platform, companies enjoy access to a wider range of complementary technologies, not to mention protection from vendor lock-in.

With the endorsement of leading J2EE vendors, including IBM and Sun, and the support of the worldwide Macromedia development community, ColdFusion provides a sure way to increase the return from an investment in the J2EE platform.

Enhancing the J2EE Infrastructure

In addition to its benefits in ease of use and productivity, ColdFusion MX also provides a rich suite of built-in services that lower the cost and complexity of developing Internet applications on the Java platform. As a

result, developers can easily incorporate the services they need into their applications - without having to purchase and integrate search technology, charting, and other add-on services from different third-party sources.

Moreover, by enabling companies to build and deploy both complex Java-based applications as well as web-based ColdFusion applications on a single J2EE server, ColdFusion MX allows companies to consolidate their server infrastructure and minimize the cost of new hardware and ongoing management. The result for the business is obvious. With fewer servers to manage and a single point of control, companies can reduce their overall infrastructure costs and spend more of their resources on finding new ways to serve the business needs of their organization.