JavaServer Pages™ Technology White Paper

A Simplified Guide

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
1 (800) 786.7638
1.512.434.1511

Please
Recycle

Contents

CHAPTER 1

Executive Summary

The Internet was once full of Web sites hosting static pages (“brochure-ware”) or simple forms at best. Now it’s an interactive environment for transacting daily business, from shopping to trading stocks to interacting with suppliers, in a personalized and dynamic setting.

Today, the tools and products to build dynamic, Web-based applications are still maturing. Traditionally, companies used CGI applications to generate dynamic content for Web pages. But that solution hasn’t scaled well to support complex functionality and growing numbers of concurrent users.

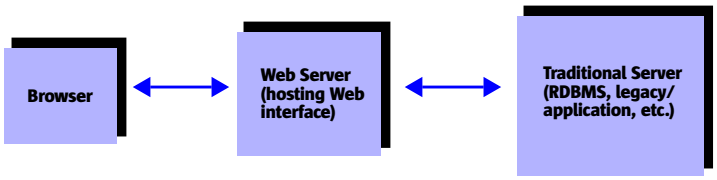
JavaServer Pages™ (JSP) technology provides a highly scalable method for creating dynamic content for the Web. As part of the Java™ family of APIs, JSP technology shares the Write Once, Run Anywhere™ benefits of the Java platform, with easy access to a broad range of Java APIs. JSP technology enables a tiered development methodology that lets organizations leverage internal programming expertise to create applications that are fast to deploy and easy to maintain.

CHAPTER 2

Multitier Application Architectures

The growth and acceptance of the Internet in both businesses and homes is changing the face of many industries — and the information systems that support them. From new .com companies to brick-and-mortar establishments, businesses everywhere are finding new ways to leverage the power of the Internet.

Software developers have been quick to realize the possibilities of Web-based clients in application architectures. With a browser on virtually every desktop, companies can deploy a multitier architecture in which Web servers act as a middle tier, managing interactions with Web-based clients.



A Web-based client architecture may have three or more layers. This multitier architecture provides many benefits over a traditional (two-tiered) client/server architecture.

- Installing and deploying the user interface is virtually instantaneous – only the Web interface in the middle tier needs to be updated.
- Because the application itself is server-based, users always access the most up-to-date version.
- Without a “thick” client interface, it is easier to deploy, maintain, and modify applications – no matter where the client is located.

These benefits explain the growing popularity of the multitier architecture, and why almost every client/server application provider has retooled or is retooling to support Web-based clients.

Development Tools and Methodologies Are Maturing

Companies building and deploying applications on this model are faced with an application environment that is still maturing. A number of different technologies — ranging from traditional CGI scripts to JSP technology — are available today to build the interactive, “customer-facing” component of these applications. The challenge is selecting an application architecture and component design that meets the evolving user needs (whether they be customers, partners, or internal staff) as well as the enterprise’s own IT requirements.

Users Have Heightened Requirements

Internet users have heightened expectations for application availability and reliability. They want to be able to access applications at any time of day or night to perform a wide variety of tasks online. They expect up-to-date information and fast response times.

To support these requirements, application providers need high-performance, highly reliable applications that can be updated easily. They need applications that can scale to support large numbers of users, and that can interact with vital business systems.

Solutions Must Meet Demanding Enterprise Software Requirements

The organizations that are building and maintaining these applications also have stringent requirements when selecting the architectures, products, and tools for creating Web-based applications.

- The development platform must support fast application deployment and rapid updates.
- The application must be easy to maintain using minimal developer resources. Many organizations face a shortage of qualified Web developers and need to protect the developers they already have.
- Finally, the organization needs to retain the ability to adopt new tools or technologies as needed, so the development environment should not close out options. With new tools, systems, and information sources appearing nearly every day, there is a risk to selecting a solution that leaves the organization entirely at the mercy of a single vendor – even if that vendor is the market leader.

CHAPTER 3

JSP Technology: The Next Evolution of Servlets

JSP technology is a means for creating dynamic Web-based content using server-side (middle-tier) processing. JSP simplifies the process of creating these dynamic pages by separating the application logic from the page design and encapsulating logic in portable, reusable Java components.

JSP technology has evolved from the powerful servlet technology. (Servlets are Java technology-based, server-side applications.) JSP extends the servlet technology in many ways, making it easier and faster to build, deploy, and maintain server-side applications that communicate with Web-based clients.

The following sections describe where JSP technology fits in the Java family of products, how JSP can simplify the creation and maintenance of dynamic pages, and how these pages fit into more complex, multitier applications.

Overview of the Java APIs

The name Java is attached to many things, so it's worth cutting back to fundamentals for a moment and defining where JSP technology fits in the Java technology family of products.

A Java programming environment consists of the *Java programming language*, a *Java language compiler* (to compile the Java programs), and a *Java virtual machine* (to run the programs). The power of Java technology for cross-platform development lies in this model —– the compiler creates platform-independent executables (Java bytecode) that run in the Java virtual machine, which generates the actual, machine-specific instructions to execute.

Write Once, Run Anywhere

The underlying design goal driving all past and future Java development is articulated in the slogan, Write Once, Run Anywhere. The concept behind the technology is that any Java program can run on any Java virtual machine — without platform-specific modification. Because Java virtual machines are available for almost every platform, this is the closest thing possible to a universal programming environment.

The Java APIs

The Java programming language consists of classes and interfaces, accessible through defined APIs (application programming interfaces), distributed in a tree structure of class libraries. The Java core classes include the primary APIs. Extensions to the language are implemented through Java standard extensions.

JavaServer Pages Technology

JSP technology is implemented as a Java API; the JSP API is part of the Java 2 Platform, Enterprise Edition (J2EE). Because pages created with JSP actually compile into servlets, they build on and extend the servlet API as well.

Because JSP technology is part of the J2EE platform, expect to see more servers and tools supporting this API. For more information on the latest version as well as a listing of the Web servers, application servers, and tools that support it, please go to *java.sun.com/products/jsp/*.

JSP Extends Servlets

In a sense, JSP technology does not provide new core technologies — everything that can be done with a JSP page, can also be done by writing a servlet. Servlets have access to the same set of Java APIs as JSP. Pages created with JSP technology are, in fact, compiled into servlets, so they cannot be capable of anything inherently different.

What JSP pages do, however, is enable a different, more efficient development methodology and simplify ongoing maintenance. This is because JSP technology truly separates the page design and static content from the logic used to generate the dynamic content.

Empowering Page Authors

Other methods for creating dynamic content require Web developers (with programming expertise) to embed the fixed page design and content into a script or a program (such as a CGI program). Average, HTML-literate page authors cannot easily edit and manipulate the content of the page without being familiar with the scripting language used.

With JSP technology pages, however, the logic itself is embedded in a standard page (HTML or XML). Page authors can use familiar tools to create and edit pages, and simply embed calls to the necessary application components where needed. All they need to know is how to invoke the logic — a programmer can be responsible for building and maintaining the logic components.

From a page author's perspective, the JSP page is a functionally-enhanced version of something they already know.

```
<JSP set up text>
<HTML tags<
<HTML tags>
text
.....
<customized JSP tag>
text
.....
<HTML tags>
text
......
<JSP tag calling JavaBean>
text
```

Benefits to Development Cycles

JavaServer Pages technology enables efficiencies in development and maintenance processes alike:

- Developers focus on the core components called by the pages. These components may be implemented through JavaBeans™ components or custom JSP tag libraries. Multiple applications can reuse the same components.
- Page authors can change the look and feel of a dynamic page, embed new content, update copyright information, change product names – all without involving the developer or interfering with the function of the page. Developers do not have to be involved in any routine page maintenance activities unless they involve application logic itself.

Not only does this division translate into faster application deployment, it makes the ongoing maintenance of these applications much easier by dividing the maintenance tasks themselves into content and logic tasks. In a sense, JSP technology keeps the code out of the content, and the content out of the code.

Custom Tag Libraries Distribute Business Logic

JSP technology supports a unique feature called Custom JSP Tags. Using this feature, a developer using the Java language could create a custom tag library of commonly-used functions, then distribute the information to a wide range of page authors. Instead of being responsible for every page's look and feel, the programmer is responsible solely for the application logic.

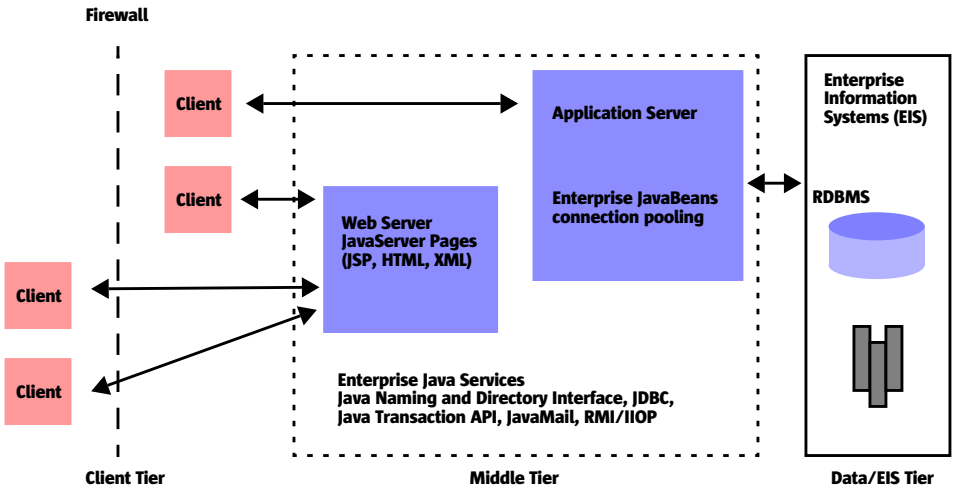
Custom tag libraries provide developers with an easy way to distribute sophisticated application functionality for reuse to a wide range of page authors. Because these tags are resolved in the server and not exposed to clients, they do not introduce client-side dependencies.

CHAPTER 4

JSP Technology in the Multitiered Architecture

JSP technology can be used to create simple, interactive pages that report the time of day or track visit-counts to a page. But this technology also scales well to takes its place in an enterprise-level, multitier application architecture.

Pages created with JSP technology are the server-based, “customer-facing” interface for multitier applications, and provide a flexible, scalable environment for deploying these applications.



Client Tier

On the client side, JSP pages communicate with Web-based clients, including HTML-based browsers. The page manages the browser interaction, sending and receiving requests and information, typically in HTML format.

Middle Tier

JSP pages may reside in almost any Web server or application server. The pages have access to the wide range of Java APIs and services, including JDBC™ technology for database access, JavaMail™ software for Java technology-based e-mail applications, Java Transaction API for transaction services, and so on.

Data / EIS Tier

Enterprise JavaBeans™ components in the middle tier can manage access to enterprise resources in the data / EIS tier, handling security, authorization, transaction integrity, connection pooling, and data caching. Enterprise JavaBeans and their containers encapsulate the complex logic required for high-performance, highly secure enterprise applications.

Flexibility and Adaptability

The beauty of the JSP architecture is its flexible, adaptable design. The back-end data sources are independent from the application logic, which itself is independent from the browser presentation.

For example, it is easy to add new JDBC technology-compliant data sources to an existing application infrastructure. Likewise, new Web-based applications can be built that access existing resources, without reworking the original data sources. Logic that is built once and encapsulated in beans or Enterprise JavaBeans components can be leveraged for many different applications. And the rich and evolving family of Java APIs means that there are continually expanding options for application integration.

The Competitive Landscape

JSP technology is not the first approach for enabling interactive Web pages — rather, it is the next step in the progression of products supporting Web-based clients.

Some of the earlier methods include CGI programs, the mod_perl plug-in for the Apache Web Server, and Microsoft Active Server Pages (ASP). For further discussion of these technologies and their differences from JSP technology, please go to *java.sun.com/products/jsp/*.

The JSP technology surpasses these previous methods in two fundamental areas:

- *Portability* – Pages built with JSP technology are portable across platforms and servers, and work with portable, reusable components.
- *Easier Maintenance and Development* – Because the page design is truly separate from the application logic, JSP enables tiered development and maintenance tasks, so page authors and developers can focus on specific areas of interest without requiring the others' help.

CHAPTER 5

Summary

JSP Technology Powers Dynamic Content on the Web

JSP technology builds on the strength of the Java family and the multivendor Java community, extending the core capabilities of the Java platform to create powerful, flexible, and easy-to-maintain dynamic Web pages. JSP technology inherits all of the benefits of the Java language, including platform- and server-independence, a modular and reusable component architecture, and access to the rich family of Java APIs (including JDBC, JavaMail, and Java Transaction Service).

Using JSP technology, organizations can meet their users' needs for up-to-date, high-performance, and reliable Web-based applications, while addressing the overall organizational needs for long-term architectural decisions:

- *Vendor Independence* – Many different Web and application servers support JSP technology, and more are added every day. The JSP specification is the product of the Java Community Process (JCP), an open process used by Sun since 1995 to develop and revise Java technology and specifications in cooperation with the international Java community. The result is a specification, built with the input and support of the major vendors in the market, that does not lock companies into any one vendor's solutions.
- *Portability* – JSP technology is portable across platforms and servers alike, building on the Write Once, Run Anywhere philosophy of the Java language. Not only are the pages themselves portable, but they can access Java technology-based components (beans, customized JSP tags) that are also reusable and portable across platforms.
- *Flexibility* – The multitier architecture using JSP technology is inherently flexible and adaptable. Developers can change data sources, access external data if necessary, and implement new security or authorization methodology – all without affecting "customer-facing" applications. Similarly, new applications can be built without reworking the infrastructure to support them. This enables organizations to adopt new technologies or applications as their business needs change.

- *Low Cost of Ownership* – JSP technology lets page authors handle the ongoing maintenance of dynamic pages without requiring developers to be involved in changes that do not involve application logic. Likewise, JSP custom tag libraries make it easy for developers to distribute advanced application functionality to a wide variety of page authors. And, as part of the Java family, JSP technology enables businesses to leverage Java language expertise, JavaBeans components, and existing Java technology-compatible components or data sources.

CHAPTER 6

JSP Deliverables

Sun's standard practice for a Java API is to provide the API specification with documentation and a reference implementation, so developers and companies can refer to a code implementation as they develop their own. Sun provides the following deliverables for JSP technology:

- TheJSP Specification – This document defines the application programming interface. The specification is a useful source document for JSP syntax.
- TheJSP Reference Implementation – Sun licensed the JSP and Servlet Reference Implementation to the Apache Software Foundation. This project, called Tomcat@Jakarta, is freely distributable and licensed directly from Apache. Tomcat is already pre-integrated with the Apache Web Server, a popular Web server. For those not using the Apache Web Server, most other Web servers are well on their way to shipping support for JSP technology (see *java.sun.com/products/jsp/ for more information*).

Competitive Landscape

To understand how JSP technology fits in the progression of products supporting dynamic content, it is worth describing a few of the alternatives.

- CGI(Common Gateway Interface) – CGI programs were an early solution for simplifying dynamic content. CGI programs are typically written in C or in Perl, a freely-distributed programming language. Still prevalent in Web-based applications, they can experience severe performance problems when scaling to support high-volume access.
- Mod_perlis a plug-in for the Apache Web Server that integrates the Perl programming language with the Web server, so programmers can write Web server extensions in Perl. As a replacement for the CGI interface, mod_perl addresses some of the limitations of CGI. This is a powerful solution, but it still has the programmers developing the pages themselves, and is tied closely to the Apache Web Server.

- Microsoft Active Server Pages (ASP) technology is closer in spirit to JSP technology than the others because its goal is to simplify the page development process. ASP is essentially limited to the Microsoft IIS Web Server on a Windows platform. (The object model imposes platform restrictions. For more information, see *java.sun.com/products/jsp/*.) ASP relies heavily on Basic-based scripting languages, which tend to be less scalable and more difficult to maintain over time than a component-based approach.

Portability Across Platforms and Servers

JSP technology, as part of the Java family, is designed to adhere to the Write Once, Run Anywhere credo. Pages created with JSP technology can run on any server, on any platform. Components (such as JavaBeans or tag libraries) developed on one platform can run easily on another.

JSP is unique in this capacity — none of the other solutions have this degree of platform-independence and cross-platform portability. This frees businesses from depending on a single vendor for product solutions or development directions.

Scalability

Scalable performance is critical for Web pages that handle a significant volume of users. CGI pages have a significant weakness when scaling to handle many concurrent users. A CGI application creates a new process for each access, creating significant performance problems in high-volume sites.

JSP pages are compiled once when first invoked and remain in memory; this provides better scalability for high-volume sites than the CGI approach.

Ease of Development, Deployment, and Maintenance

Creating CGI programs and mod_perl scripts are tasks belonging to programmers familiar with the programming languages. The same can be said of writing beans or custom tag libraries — it requires familiarity with the Java language.

But writing and maintaining JSP pages that *call* these custom components is a task that can be delegated to page authors familiar with HTML or XML. In other words, only the JSP technology really enables a tiered development methodology leveraging each person's skills (described above) in addition to a tiered architecture.

Some of the other differences are summarized in the table below. For a more direct comparison of the technologies, please see *java.sun.com/products/jsp/*.

	CGI/Perl	**Mod_Perl**	**ASP**	**JSP**
Web server	Any Web server	Apache Web Server	Microsoft IIS or Personal Web Server	Any Web server, including Apache, Netscape, IIS today
Portable across platforms/servers	No	No	No	Yes
Reusable, modular code	No	No	No	Yes
Scripting language	C, Perl	Perl	VBScript, JScript	Java
Memory leak protection	Yes	No	No	Yes
Supports concurrent access without separate processes	No	Yes	Yes	Yes

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303

1 (800) 786.7638
1.512.434.1511

http://java.sun.com/products/jsp

December 1999