
JAM Stack

An Open-source Architecture for Enterprise Applications

Javad K. HESHMATI ♦ 30-4-2006 ♦ 16 Pages

Dixite
Av. Louise 179, P.B.: 3, 1050 Brussels ♦ www.dixite.com

Preface

Introduction

Today, more and more developers want to write distributed transactional applications for the enterprise and thereby leverage the speed, security, and reliability of server-side technology. If you are already working in this area, you know that in the fast-moving and demanding world of e-commerce and information technology, enterprise applications must be designed, built, and produced for less money, with greater speed, and with fewer resources than ever before Eric Armstrong. Java, JBoss, Apache, MySQL (JAM) provides solutions for implementing distributed transactional applications that are *reliable, scalable, secure* and *affordable*. What is JAM?

JAM is an open-source stack that includes *Java & JBoss* (§1.1), *Apache* (§1.2) web server, its servlet container, and *MySQL* (§1.3) database. It offers an architecture that is best suited for an enterprise distributed environment.

Why JAM based architecture is important to your Enterprise?

1. *Lower Cost of Ownership*: Open source products typically not only have lower license costs, but also significantly reduce the operation costs that make up the total cost of ownership.
2. *Reliability and Performance*: A large community of users try and test open source software across a range of platforms before it is certified for production. When bugs are found, they are fixed quickly. Extensive public review as well as rapid development iteration has made open source software more reliable than closed source.
3. *Security*: Open source software is out there in the open, a large community of users try it and pin-point security issues. As a result open source software is more secure and suffer fewer vulnerability attacks than proprietary software.

Scope

The scope of this document is to cover the *Java based reference architecture* of [Dixite](#), that is JAM. Each component of the stack is briefly described, but a complete account of the individual components of the JAM stack is beyond the scope of this document.

Document Structure

Chapter *JAM Stack* (§1) covers the *JAM stack* and its components.

Chapter *JAM Architecture Components* (§2) describes the *JAM architecture* for a typical enterprise application.

Chapter *Further Information* (§3) provides a list of online resources about JAM components.

Chapter *FAQ* (§4) provides answers to some Frequently Asked Questions (FAQ) with regard to open source software.

Document Conventions

When you read this document, certain words are represented in different fonts, typefaces, sizes, and weights. This highlighting is systematic; different words are represented in the same style to indicate their inclusion in a specific category. The types of words that are represented this way include the following:

Element	Style
External link	<i>Dixite Website</i> ↔ www.dixite.com Or Dixite Website
Reference	the canonical name (§section number) is used to refer to sections.
Important	<i>emphasized</i>

CONTENTS

CHAPTER 1	JAM Stack	5
1.1.	Java & JBoss	5
1.1.1.	JBoss	5
1.2.	Apache	6
1.2.1.	Tomcat	6
1.3.	MySQL	6
CHAPTER 2	JAM Architecture Components	7
2.1.	Client Tier	7
2.1.1.	Client Components	8
2.2.	Web Tier	8
2.2.1.	Web Components	8
2.3.	Business Tier	9
2.3.1.	Business Component	9
2.4.	Database Tier	9
2.5.	Component Communication	9
CHAPTER 3	Further Information	10
CHAPTER 4	FAQ	11

CHAPTER 1

JAM Stack

Figure 1.1, shows the JAM stack. This chapter describes the JAM stack by describing each individual component of the stack.

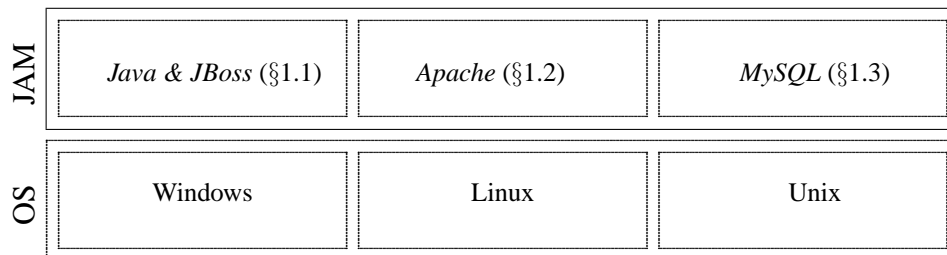


Figure 1.1: JAM Stack Components

1.1 Java & JBoss

The [Java](#) programming language is a general purpose, concurrent, class-based, object-oriented language. From its introduction to the world in 1994 to the current day, Java has been used in myriad of ways to implement various types of systems. Over time, Java has become the chosen platform for programming servers.

1.1.1 JBoss

As the acceptance and adoption of Java on server side became more established, and the demand for Web-centric applications rose, a new breed of infrastructure applications (applications servers) emerged. Applications servers such as [JBoss](#) provide the basic infrastructure required for developing and deploying multi-tiered enterprise applications.

1.2 Apache

Apache web server has been the dominant web server since 1996.

1.2.1 Tomcat

[Tomcat](#) is the Apache Servlet container that is considered as the reference implementation of servlet engines. A *servlet* is a generic web server extension that can be loaded dynamically to extend the functionality of the server. Servlets are commonly used with web servers where they take the place of Common Gateway Interface (CGI) scripts. Unlike CGI, which uses multiple processes to handle separate requests, servlets are handled by separate threads within the web server process. This means that servlets are more *efficient* and *scalable*. Another advantage of servlets is that they are portable: both across operating systems as with Java and also across web servers.

1.3 MySQL

MySQL database has grown the world's most popular open source database with over six million installations worldwide. MySQL serves as the *data store* in our reference architecture.

JAM Architecture Components

Figure 2.1, shows the JAM architecture. As shown, JAM is a *multi-tiered* architecture based on Java 2 Enterprise Edition (J2EE). This chapter describes each individual tier of the JAM architecture.

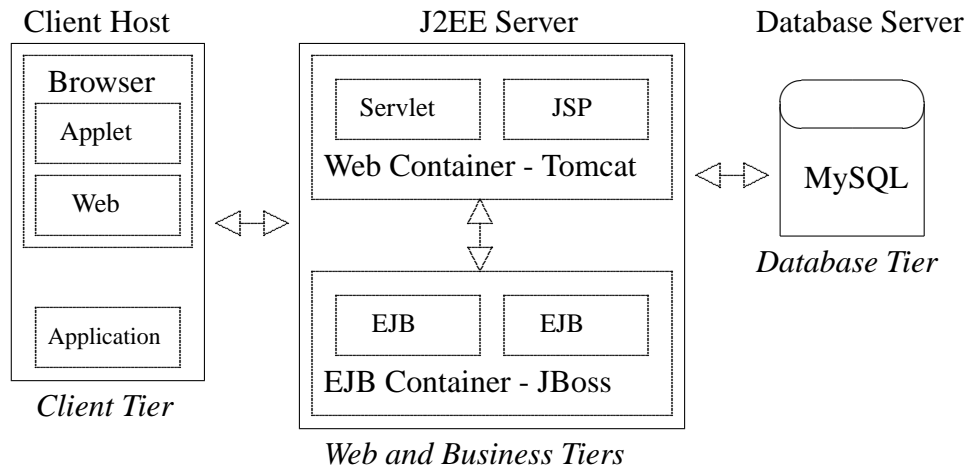


Figure 2.1: JAM Architecture

2.1 Client Tier

The client tier interacts with the user and displays information from the system to the user. J2EE supports different types of clients, including web (HyperText Markup Language (HTML)) client, Java applets and applications.

2.1.1 Client Components

Applet

Applets are small Java programs that can be embedded within HTML pages and downloaded and executed by a web browser. Because executing code from random websites presents a security risk, Java goes to great lengths to ensure the integrity of the program executing and to prevent it from performing any unauthorized tasks.

Web Clients

A web client consists of two parts:

1. Dynamic web pages containing various types of markup language (HTML, XML, and so on), which are generated by web components running in the web tier.
2. A web browser, which renders the pages received from the server.

A web client is sometimes called a thin client. Thin clients usually do not query databases, execute complex business rules, or connect to legacy applications. When you use a thin client, such heavyweight operations are off-loaded to enterprise beans executing on the J2EE server, where they can leverage the security, speed, services, and reliability of J2EE server-side technologies.

Application Clients

An application client runs on a client machine and provides a way for users to handle tasks that require a richer user interface than can be provided by a markup language. It typically has a Graphical User Interface (GUI) created from the Swing or the Abstract Window Toolkit (AWT) API, but a command-line interface is certainly possible.

2.2 Web Tier

The Web tier generates the presentation logic and accepts user responses from the presentation clients, which are typically HTML clients, Java applets and other Web clients. Based on the received client request, the presentation tier generates the appropriate responses to the client request that it receives. In the J2EE platform, *Servlet* (§2.2.1) and *JSP* (§2.2.1) in a Web container implement this tier.

2.2.1 Web Components

J2EE web components are either servlets or pages created using JSP technology (JSP pages). Servlets are Java programming language classes that dynamically process requests and construct responses. JSP pages are text-based documents that execute as servlets but allow a more natural approach to creating static content. Eric Armstrong

Servlet

Java Servlets offer fast, powerful, portable replacement for CGI scripts. Servlets execute within the server's process space and they persist between invocations, which gives them tremendous performance benefits over CGI programs.

JSP

The CGI approach for developing Web-centric applications is resource intensive and does not scale well. With the introduction of Java Server Page (JSP) technology, Java developers had an elegant and efficient mechanism to write web-centric applications that generates dynamic content.

2.3 Business Tier

This tier handles the core business logic of the application. The business tier provides the necessary interfaces to the underlying business service components. The business components are typically implemented as Enterprise Java Bean (EJB) components with support from EJB container that facilitates the component life cycle and manages *persistence*, *transactions*, and the resource allocation.

2.3.1 Business Component

Business code that solves or meets the needs of a particular business domain such as banking, retail, or finance, is handled by enterprise beans running in the business tier.

EJB

EJB is one of the prominent technologies in the J2EE platform. The EJB provides a standard for developing reusable Java server components that run in an application server such as *JBoss* (§1.1.1). The EJB specification provides a vendor independent programming interface for application servers.

2.4 Database Tier

This tier is responsible for the enterprise information systems, including database systems and enterprise resources.

2.5 Component Communication

The client communicates with the business tier running on the J2EE server either directly or, as in the case of a client running in a browser, by going through JSP pages or servlets running in the web tier.

CHAPTER 3

Further Information

For more information about JAM components and architecture, refer to the following:

JAM	Topic & URL
J	<i>Java</i> ↗ java.sun.com
J	<i>JBoss</i> ↗ www.jboss.com
J	<i>J2SE</i> ↗ java.sun.com/j2se
J	<i>J2EE</i> ↗ java.sun.com/j2ee
J	<i>Blueprints</i> ↗ java.sun.com/j2ee/blueprints
J	<i>EJB</i> ↗ java.sun.com/j2ee/products/ejb
J	<i>Servlet</i> ↗ java.sun.com/j2ee/products/servlet
J	<i>JSP</i> ↗ java.sun.com/j2ee/products/jsp
J	<i>JDBC</i> ↗ java.sun.com/j2ee/products/jdbc
A	<i>Web Server</i> ↗ http.apache.org
A	<i>Tomcat</i> ↗ jakarta.apache.org
M	<i>MySQL</i> ↗ www.mysql.com

Table 3.1: Websites and URL References

CHAPTER 4

FAQ

This section is provided in order to answer some FAQ and place the JAM stack in the context of the open source software.

What is open source ?

According to Open Source Initiative (OSI), open source can be defined as follows:

Open source promotes software reliability and quality by supporting independent peer review and rapid evolution of source code.

Why is open source software important ?

The initial attraction that most users have to open source software is the cost of ownership and quality. In addition, open source program allow developers to study the source code, improve it (if required) and use it in their own programs. Contributors from around the world contribute to the open source community by offering translation, documentation, and bug reports.

What are Dixite contribution to the open source community ?

Dixite engineers and staff participate in open source projects; a good example of such projects is *SilkPage* ↗ silkpage.markupware.com. In addition, Dixite contributes to the open source community by contributing to open source foundations such as Free Software Foundation (FSF) and Apache Software Foundation (ASF).

What is the Dixite added value on JAM ?

At Dixite we have an extensive experience with all of the JAM components. We implemented the first Java based project at DG Fish of the European Commission in 1999. Since then we have built an extensive knowledge-base on each individual component of JAM. In addition,

we have access to a wealth of support materials, toolkits, middleware and software provided by the open source community as well as our partners. Table ?? lists the main components of the JAM stack as well as our *key partner* and a *selected client* for whom we have implemented a JAM based solution.

Component	Key Partner	Selected Client
<i>Java & JBoss</i> (§1.1)	Sun Microsystems	European Commission
<i>Apache</i> (§1.2)	MarkupWare	EuroLand
<i>MySQL</i> (§1.3)	MySQL	Auxipress

Table 4.1: Quick Reference to Dixite Partners

Document Information

This document was typeset by the author using \LaTeX ¹.

Revision

Revision	Date	Modified Sections
0.1	26-06-2005	Initial Revision

Table 4.2: Document Revision History

Related Documents

Title	URL	Description
SPPOSS	www.dixite.com/docs/spposs	Software Production Process Using Open Source
LAMP Stack	www.dixite.com/docs/lamp	An Open-source Architecture for Web Based Applications

Table 4.3: Related Documents

Copyright

This document can be freely redistributed according to the terms of the GNU Free Documentation License (GFDL). To learn more about GFDL, visit the following *URL* ↗ www.gnu.org/copyleft/fdl.html.

¹*LaTeX* ↗ www.latex-project.org

Glossary

Applet Applet some text ...

LAMP Linux, Apache, MySQL, PHP

SPPOSS Software Production Process using Open Source Software is [Dixite](#) approach to software development. *URL* ↗ www.dixite.com/docs/spposs

XML Extensible Markup Language Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere. *URL* ↗ www.w3.org/XML

HTML HyperText Markup Language XHTML 2 is a general-purpose markup language designed for representing documents for a wide range of purposes across the World Wide Web. To this end it does not attempt to be all things to all people, supplying every possible markup idiom, but to supply a generally useful set of elements. *URL* ↗ www.w3.org/MarkUp

HTTP HyperText Transfer Protocol The underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands.

GNU GNU's Not UNIX *URL* ↗ www.gnu.org

FSF Free Software Foundation *URL* ↗ www.fsf.org

ASF Apache Software Foundation *URL* ↗ www.apache.org

OSI Open Source Initiative *URL* ↗ www.opensource.org

FAQ Frequently Asked Questions

JSP Java Server Page

JDBC Java Database Connectivity

EJB Enterprise Java Bean

J2EE Java 2 Enterprise Edition

J2SE Java 2 Standard Edition

CRE Customer Relation

CMS Content Management System

JAM Java, JBoss, Apache, MySQL

CGI Common Gateway Interface The CGI defines a way for a web server to interact with external content-generating programs, which are often referred to as CGI programs or CGI scripts.

SSL Secure Socket Layer SSL is a technology which allows web browsers and web servers to communicate over a secured connection. This means that the data being sent is encrypted by one side, transmitted, then decrypted by the other side before processing. This is a two-way process, meaning that both the server AND the browser encrypt all traffic before sending out data.

GUI Graphical User Interface

AWT Abstract Window Toolkit

BIBLIOGRAPHY

Eric Armstrong, Jennifer Ball, Stephanie Bodoff: The J2EE 1.4 Tutorial. NR, 2005, ISBN
0-7357-0921-2