J2EE 在网站建设中的应用

Building Websites Based on J2EE

李真文　杨学良

（中国科技大学研究生院计算机系　北京100039）

Abstract　This paper presents J2EE(Java 2 Enterprise Edition)in history, architecture, running units and application interfaces. Then it introduces the architecture of a typical website building on J2EE. The paper concludes with pros and cons in building websites based on J2EE.

Keywords　Java, J2EE, EJB, JDBC, JSP

1　引言

Java 在1995年由 Sun 公司作为一种编程语言推出，其"编写一次、到处运行"的平台可移植性广受瞩目，一度被业界作为 Microsoft Windows 之外的重要选择。但是，人们后来发现，Java 提供的桌面应用功能非常有限，下载速度慢，Java 虚拟机没有优化，运行性能差强人意，Java 因此遭受冷落。

与此同时，Internet 和电子商务大潮开始席卷全球。跟上这个浪潮唯一的办法就是将传统应用变成基于 Web 的多层(n-tier)体系结构，以满足大型门户网站和电子商务网站必须具备的规模可伸缩性、平台可移植性和事务可管理性的要求。而多层体系结构的核心是应用服务器，它主要担当两个角色：组件协同工作的组织者和连接传统异构环境的中间件。这样，应用服务器标准成为热点，软件巨头都想成为该标准的制定者。其中，Microsoft 推出 DNA(现在处于 Beta 阶段的 Net Framework 的前身)，OMG(Object Management Group)推出 CORBA，以及 Sun 公司推出 J2EE(Java 2 Enterprise Edition)。Microsoft 的 DNA 局限自不必说，采用 DNA 必须要以使用 Windows 为前提；OMG 的 CORBA 本来可以作为工业标准，可是没有及时赶上 Internet 和电子商务的大潮。

虽然 Java 作为桌面应用表现不尽人意，但是，Sun 公司从来没有停止过拓展 Java 的应用领域。1997年4月，Sun 公司开始开发 Java Enterprise Platform。通过开放的 Java 社区进程(JCP)，Sun 推出了一组标准扩展 Enterprise Java API，这就是 J2EE 的原型。经过近2年的发展，于1999年6月，在 Sun 公司发布 Java 2的时候，发布 J2EE，从而将 Java 从一个编程语言变成了全功能的服务器端应用开发环境。J2EE 一出台，立即得到业界的强劲支持，成为大型门户网站和电子商务网站建设非 Microsoft 的不二选择。从1999年下半年至2000年年底，美国新锐网站大量采用 J2EE 来构造就充分说明了 Java 已经再次唤起了人们的高度重视。

2　体系结构

J2EE 从 Java Enterprise Platform 起源，版本不断升级，技术不断演化，体系结构当然就不断庞大。本文讨论的是文[4]给出 J2EE 1.3版本的体系结构，参见图1。从图中可以看出，J2EE 平台包含下面3个方面：

1. 运行单元。J2EE 通过提供 Java 小程序容器、Java 应用程序容器、Web 容器、EJB 容器，为 Java 小程序、Java 应用程序(JSP 和 Servlet)、Web 应用程序、EJB 组件提供运行环境。而 Java 小程序、Java 应用程序(JSP 和 Servlet)、Web 应用程序、EJB 组件是 J2EE 的运行单元，是基于 J2EE 平台的开发人员着眼点。

2. 应用接口。J2EE 通过提供 JMS、JAAS、JTA、JavaMail、JAXP、JDBC、Connector 等技术来作为与其他应用和系统集成的接口。图中画出了 JDBC 与数据库的接口。

3. 基础平台。随 Java 2一同推出的除了 J2EE 以外，还有另外两个版本：J2SE(Java 2 Standard Edition)和 J2ME(Java 2 Micro Edition)。其中，J2SE 主要用于 Java 小程序和 Java 应用程序(客户端应用)的编写；J2ME 用于各种嵌入设备的开发。尽管在 J2EE 中从来不提 J2SE，但事实上 J2EE 构筑在 J2SE 之上。

3　运行单元

虽然 J2EE 平台通过作为基础的 J2SE 支持 Java 小程序和 Java 应用程序的开发，但它们主要在客户端运行，不在本文讨论范围之内，本文仅介绍在服务器端运行的 EJB、Servlet 和 JSP。

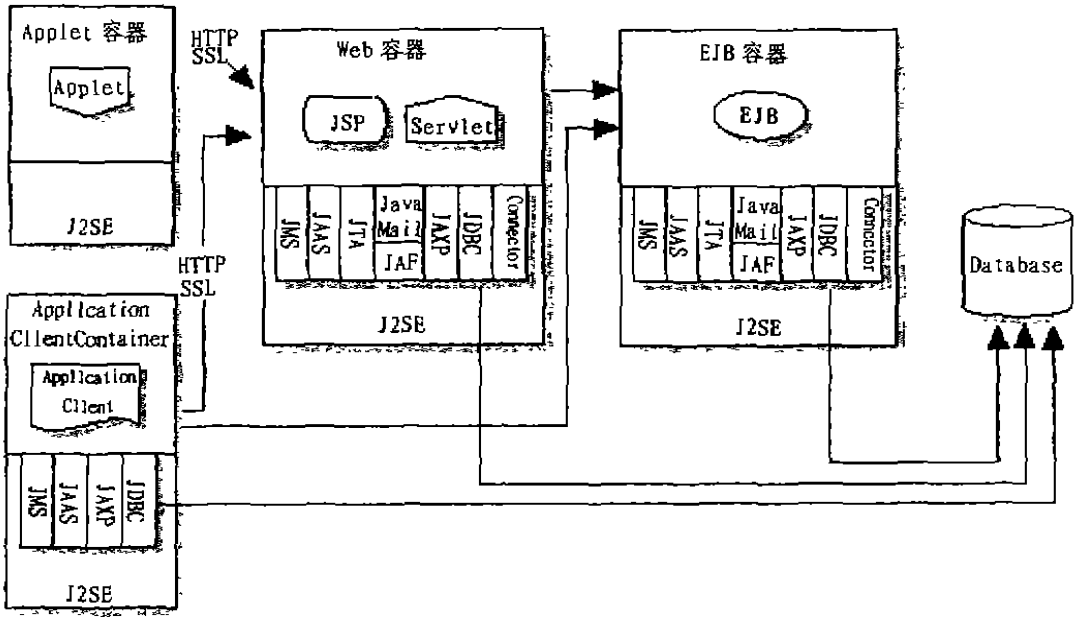

图1 J2EE 体系结构

3.1 EJB

EJB(Enterprise JavaBeans)是 J2EE 平台的组件技术，是 J2EE 的核心。EJB 是用 Java 编写的服务器端组件，用于实现企业应用的商业逻辑。EJB 分为两类：会话 EJB 和实体 EJB。前者用于应用逻辑实现；后者用于数据存储和状态跟踪。

在 EJB 之前，Sun 公司已经推出 JavaBeans。通过比较 EJB 与 JavaBeans，可以更清楚知道 EJB：

·都是组件模型：JavaBeans 用于可视化地在开发工具中操纵组件，EJB 用于可移植地在服务器中部署组件。

·都可以用于服务器端开发：JavaBeans 在用于服务器端开发时需要设计整个服务器服务框架，而 EJB 不用。

·事件和属性：JavaBeans 包含事件和属性等特征，EJB 则没有事件，因为 EJB 通常不发送和接受事件，同样也没有属性——属性定制并不是在开发时进行，而是在运行时(实际上在部署时)通过部署描述符来描述。

每个 EJB 至少包含下面代码：Remote 接口、Home 接口和 EJB 类。Remote 接口定义了 EJB 客户调用的方法，该方法在 EJB 类中实现：

```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;
public interface Converter extends EJBObject{
  public double dollar ToYen(double dollars)throws Remo-
      teException;
  public double yenToEuro(double yen)throws RemoteEx-
      ception;
}
```

Home 定义 EJB 客户创建、查找或删除 EJB 的方法。下面 ConverterHome 接口包含一个创建方法，它返回 Remote 接口类型的对象：

```
import java.io.Serializable;
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;
public interface ConverterHome extends EJBHome{
  Converter create()throws RemoteException,CreateExcep-
      tion;
}
```

下面例子是一个会话 EJB ConverterEJB。实现了 Remote 接口定义的 dollarToYen 和 yen ToEuro 2个方法：

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
public class ConverterEJB implements SessionBean{
  public double dollarToYen(double dollars){
    return dollars*121.6000;
  }
  public double yenToEuro(double yen){
    return yen*0.0077;
  }
  public ConverterEJB(){}
  public void ejbCreate(){}
  public void ejbRemove(){}
  public void ejbActivate(){}
  public void ejbPassivate(){}
  public void setSessionContext(SessionContext sc){}
}
```

3.2 Java Servlet

Java Servlet 是最早的 Java 服务器端编程技术，推出当时的目的是用于解决服务端 Java 编程问题，而后成为 JSP 的重要基础：JSP 就是 Java Servlet API 的扩展，并且在执行机制上，JSP 被编译为 Java Servlet 后才被执行的。所以，Java Servlet 在 J2EE 平台中的地位已经演变为低层框架支持，新型的 J2EE 应用已不

推荐 Java 程序员直接编写 Java Servlet 来构筑网站。

3.3 JSP

JSP(JavaServer Pages)是 J2EE 平台的页面编程技术,通过将 JSP 脚本嵌入到 HTML 页面中,就可以快速开发查询数据库和处理 HTML 表单的动态页面。很多中小型的基于 J2EE 的网站,由于开发成本的原因,很有可能没有采用 EJB、Servlet,但是,肯定采用了 JSP,因此使得 JSP 成为 J2EE 中使用最广泛的技术。

在 HTML 中嵌入可执行代码用于服务器端处理的技术并不时新。早在 JSP 之前,Microsoft 就推出了 ASP(Active Server Pages):在 HTML 页面中嵌入 VBScript(或者 JavaScript)的代码,用于服务器端产生动态内容。但是,JSP 比 ASP 走得更远:每一 JSP 页面在第一次访问时自动编译成 Servlet 类文件,而 ASP 页面每次访问都得重新解释执行,使得 JSP 从执行机制上就比 ASP 优越。现在处于 Beta 阶段的 Microsoft. Net Framework,又借鉴了 JSP 的设计思路,也将在新版 ASP. Net 中通过引入新型的、类似于 Java 的 C# 语言(或者改进的 VB 语言),采用与 JSP 相同的执行机制。

下面是一个简单的 JSP 例子,用于显示服务器当前日期:JSP 代码镶嵌在"<%"和"%>"之间:

```
<html>
<head>
<title>JSP 示例页面</title>
</head>
<body>
<h1>JSP 日期显示示例</h1>
<h2>
<% response.setHeader("Refresh",5);%>
当前日期是<%=new Date()%>.
</h2>
</body>
</html>
```

4 应用接口

J2EE 的应用接口在 J2EE 发展过程中不断演化:已有技术不断升级,新技术不断添加。下面介绍的是 J2EE 1.3版本中的各种应用接口技术。

4.1 JDBC

JDBC(Java Database Connectivity)是 J2EE 平台的老成员,也是应用最广泛的技术之一。JDBC 提供 J2EE 平台中数据库接口标准,用于提供了访问各种关系数据库的统一接口,并提供了更高层的数据库工具和接口的通用基础。

本 JDBC 例子假设有 Cloudscape(Informix 的纯 Java 数据库,是随 J2EE 发布的示例数据库系统,http://www.cloudscape.com)数据库 PhoneBook,其包含表 CONTACT_TABLE,并具有 NAME 和 PHONE 两个字段。下面代码首先装入 Cloudscape JDBC 驱动程序,并请求驱动程序管理器获取 PhoneBook 数据库连接。通过该连接,构造 Statement 对象,并用它来执行简单 SQL 查询。最后,迭代结果集中的所有项目,并把 NAME 和 PHONE 字段写到标准输出。

```
import java.sql.*;
public class JDBCExample
  public static void main(String args[])
    try
       Class.forName("COM.cloudscape.core.
            JDBCDriver"),
        Connection conn = DriverManager getConnection
          ("jdbc:cloudscape:PhoneBook");
      Statement stmt=conn createStatement();
    String sql="SELECT name,phone FROM CONTACT_
      TABLE ORDER BY name";
        ResultSet resultSet=stmt.executeQuery(sql);
        String name;
        String phone;
        while(resultSet.next()){
        name=resultSet.getString(1).trim();
        phone=resultSet.getString(2) trim();
        System out.println(name+","+phone);
        }
      }
  catch(Exception e)
        e.printStackTrace();//处理异常
    }
  }
}
```

4.2 JNDI

JNDI(Java Naming and Directory Interface)是 J2EE 平台的老成员,也是应用最广泛的技术之一。JNDI 提供命名和目录服务技术,它为多种命名和目录服务提供了统一的访问方式,这些服务包括 LADP、DNS、NIS、NDS、RMI 和 CORBA。

4.3 JTA

JTA(Java Transaction API)是 J2EE 平台中的事务管理技术,用于确保与复杂事务资源的互操作性,这些事务资源包括事务应用程序、资源管理程序、事务处理监视程序和事务管理程序。由于这些组件可以由不同厂商提供,JTA 提供了一个开放、标准的访问这些事务资源的途径。

4.4 JavaMail

JavaMail 是 J2EE 平台中的邮件技术,是构造 Java 邮件和信报应用程序、与平台和协议无关的框架,

4.5 JMS

JMS(Java Message Service)是 J2EE 平台中的工作流技术,用于提供企业信报服务。企业信报服务包括可靠的排队、出版和订阅通信,以及各种推-拉技术。

4.6 RMI-IIOP

RMI-IIOP 提供了 OMG 的工业标准 Internet Inter-Orb Protocol(IIOP)之上的 Java RMI API 实现。通过它,可以开发出客户和服务器之间的远程接口,并用 Java 和 Java RMI API 实现它们。

4.7 JAAS

JAAS(Java Authentication and Authorization Service)用于验证用户和对访问实施控制。通过实现标准的插入式验证模块(Pluggable Authentication Module,PAM),以及扩展 J2EE 的访问控制机制,使之支持基于用户的访问控制。并且,JAAS 还提供与非 J2EE 组件(比如 CORBA)的互操作性。

4.8 JAXP

JAXP(Java API for XML Parsing)是J2EE 1.3版本中的新成员，用于阅读、操作和产生XML文档。由于JAXP支持工业标准SAX和DOM，J2EE开发人员可以在选择不同XML分析器而不用修改任何代码。由于XML是电子商务计算时代的页面描述语言，开发人员因此可以快速、容易地构造支持XML的电子商务应用。

4.9 Connector

Connector是J2EE 1.3版本中的新成员，是J2EE平台与异构企业信息系统(EIS)互连的体系结构标准。EIS包括ERP系统(比如SAP R/3)、大型机事务处理系统(比如IBM CICS)以及传统应用和非关系数据库系统。

5 基于J2EE做网站开发

由于J2EE的规模可伸缩性、平台可移植性和事务可管理性，使得J2EE成为大中型门户网站和电子商务网站首选平台。本文以建立一个典型的电子商务网站为例，说明采用J2EE构造网站的典型结构。该网站是一个网上商店，体系结构参见图2。

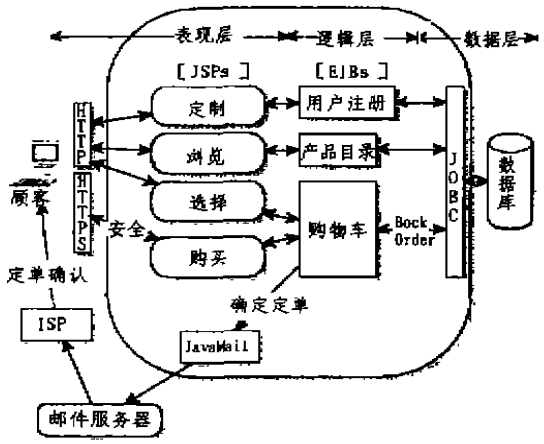

图2 网上商店的体系结构

限于篇幅，详尽的网上商店实施方案不在本文讨论范围之内，本文主要讨论图2演示的主要模块：

1. 表现层。定制、浏览、选择、支付等功能直接与用户打交道，功能经常需要调整，属于表现层。在J2EE中，表现层用JSP来实现，以便可以快速、容易地开发和修改。

2. 逻辑层：用户注册、产品目录，购物车等功能涉及公司的营销策略，功能要求比较稳定，并要具有相当的安全性，属于逻辑层。在J2EE中，逻辑层用EJB来实现。

3. 数据层。J2EE与数据库打交道都通过JDBC来实现，逻辑层中需要永久保存的信息需要通过JDBC存放到数据库中。

4. 支付处理。电子商务网站需要解决安全支付的问题，这样，必须通过加密的HTPPS协议来传输有关支付的信息，比如信用卡类型、号码、过期时支付金额等。

5. 定单处理。网上书店不仅要提供通过JSP页直接支付，还要提供通过Email系统进行订购处理。J2EE通过Java Mail来构造所有有关邮件通信的应用。

实现在线商店的体系结构与实现别的企业应用的体系结构相似，这些企业应用包括客户关系管理(CRM)、供应链管理(SCM)等。

结束语 采用J2EE体系结构做网站开发，可以构造出完全不依赖平台的网站，并且支持分布式事务处理，是大型商务网站和门户网站的首先构造平台。相对于业界的另一选择Microsoft.Net Framework，选择J2EE就意味着选择应用服务器的灵活性：可以随意选择J2EE应用服务器。

严格说来，Sun公司出台的J2EE仅仅是基于Java的中间件的企业标准，开发者可以选择J2EE自带的作为原型的J2EE实现。而事实上根本没有必要—市场上还有若干非常优秀的J2EE应用服务器，包括Bea的WebLogic、IBM的WebSphere、Allaire的JRun等。由于这些产品以J2EE为标准，开发者编写的J2EE应用可以在不修改任何代码的情况下，可以运行在各应用服务器上。

但是，任何事物都有其两个方面。笔者认为，J2EE目前仍然具有下列劣势：

·JSP还是太难。相对来讲，JSP在所有脚本编程语言中(Perl、VBScript、JavaScript、PHP)还是太难，仍然需要足够的Java语言的知识，由于没有完善的开发工具，那些只会HTML/XML的Web内容作者将被拒之门外。

·第三方服务器端组件还不多。相对于国际市场上大量Microsoft ASP组件来讲，可用的第三方EJB组件、JSP置标库还不多。

参考文献

1 The Source for Java Technology. Available at: http://www.javasoft.com/, Sun Microsystems, December 20, 2000

2 Java Emerges As Server-Side Standard, Alan Radding, May 2000. Available at: http://www.informationweek.com/767/java2.htm

3 Java 2 Platform, Enterprise Edition—Ensuring Consistency, Portability, and Interoperability, Anne Thomas, Patricia, Seybold Group, Sep. 1999

4 Java 2 Platform Enterprise Edition Specification, v1.3, Sun Microsystems, Oct. 2000

5 Develop n-tier applications using J2EE, Steven Gould, December Java World. Available at: http://www.javaworld.com/javaworld/jw-12-2000/jw-1201-weblogic_p.html

6 Java 2 Enterprise Edition Developer's Guide Version 1.2.1, Sun Microsystems, May 2000

7 Simplified Guide to the Java 2 Platform, Enterprise Edition, Sun Microsystems, July 2000