基于构件的Web应用框架

吴光亮　徐茂生

（北京交通大学电子信息工程学院　北京100044）

摘　要　Web应用基础是以文档为中心，面向对象的成熟软件方法无法有效应用到Web应用的开发中。本文提出了一种多层次Web应用框架，抽象出了一个企业业务相关的用户框架层，使它可以同时支持文档和对象两类模型，有效地提高了复用度。最后从开发角度给出了一个合理的开发流程模型。

关键词　Web应用框架，基于构件的软件

Component-Based Framework for Web Application

WU Guang-Liang　XU Mao-Sheng

(School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing 100044)

Abstract　Because the basic of Web application is document, it is difficult to apply object-oriented well-established software development methods to Web applications. A multi-tier Web application framework is presented in this paper, in order to support both document and object model, one enterprise-dependent user framework is modeled. By using the model in designing, it can enhance effectively reusability. From a point of view of different development rules, a reasonable development flow based on the Web application framework is proposed at last.

Keywords　Web application framework, Component-based software

越来越多的企业应用转移到Web方式的应用平台上，用Web来统一整个企业的应用已经成为一个重要的趋势。然而，最初Web只是作为一种信息平台引入的，Web应用围绕着以文档为中心，其基础是抽象能力低的资源模型[1]，因此成熟的软件开发方法无法被应用到B/S开发过程中。这使得目前Web应用的开发和维护在很大程度上依赖于开发者个人的知识和经验，当应用的规模扩大时，相应的风险和成本也随之增长。

针对上述问题，本文提出了基于构件的Web应用框架结构，对页面层次和业务逻辑层次的不同实体进行建模，并利用XML实现了实体之间交互。与传统的应用框架相比，本文的应用框架使动态网页的实现更加容易，实现了页面显示与逻辑的松散耦合，提高了复用度。

本文第1节在分析Web平台基础上，介绍Web应用的一般体系结构。第2节提出一种多层次Web应用框架，抽象出一个企业业务相关的用户框架层，用户框架层分为通用Web视图框架和企业业务相关的构件模型，进而给出框架的实现方式。从开发的角度，给出一个合理的开发流程模型。最后给出结论。

1　Web应用的体系结构

Web应用从最初的提供文档为中心的信息导航服务到现在支持不同规模的分布式计算，经历了三种不同的Web应用方式。从技术实现和运行模式角度划分，可以分为脚本模式、面向对象模式和对象模式。

脚本模式通过脚本以超文本方式描述动态页面的内容和处理逻辑，当接收到客户请求时，应用服务器首先搜索相应的源文件，然后在服务器端解释该源文件中的脚本，最后将脚本解释器产生的结果汇编后返回给Web服务器。通过扩展接口或服务器端脚本语言可动态地产生响应页面。这种模式可重用度低，且缺乏集成历史遗留系统以及事务处理的支持能力。

面向对象模式介于脚本模式和对象模式中间的过渡，其主要特点是使用面向对象语言编写脚本。例如：在JSP和Servlets中使用Java语言进行编码。相对纯脚本模式，该模式的可重用性较好，但没有相应的规范，不提供统一的接口规范，其使用范围和移植性受到了限制。

对象模式支持分布对象模型，支持组件重用，可维护性高。与面向对象模式相比，对象模式应用服务器遵循相应的标准和规范，其中较突出者主要有两大类：J2EE和.Net。J2EE由SUN公司1999年底提出，J2EE为事务性Web应用的开发、部署、运行和管理提供一个完整的底层框架和一套标准的规范，在不同的J2EE应用服务器之上的应用组件可以互

吴光亮　硕士生，主要研究领域为sensor network, multi-agent。

操作，移植的风险和代价小。微软的.Net 构建在 Windows DNA 技术基础上，提供一系列企业级应用服务，为部署、管理和建立基于 XML 和 Web 的应用构筑了基础框架。

目前基于 Web 的应用大多建立在对象模式的 Web 平台之上，但是由于 Web 应用(Web 视图)的基础是抽象能力低的资源模型，因此传统的 C/S 的开发框架没法直接迁移到 B/S 模式。因此把 Web 视图与业务逻辑一块考虑的 Web 应用框架的开发显得尤其紧迫。

2 基于构件的 Web 应用框架

基于构件的软件开发(component-based software development，CBSD)是解决企业信息系统开发和维护问题的最新成果，近几年业界构件标准(如 EJB，COM＋，CORBA 等)的成熟进一步促进了 CBSD 的发展和应用。在 Web 应用开发中，设计者可以把应用中不同层次的对象抽象成构件，合理的框架不仅能促进并行开发，还可提高复用度，降低成本。

2.1 Web 应用框架

J2EE 服务器实现了 Servlet，JSP，EJB，JTA，JTS，JMS，JAXP，RMI-IIOP，JNDI，JCA，Java Mail 和 JAF 等一套标准的规范，并提供一个完整的底层框架，是 Web 应用的理想平台。

企业决策者并不想在 J2EE 平台上直接构建企业应用，而是希望在 J2EE 基础框架之上，建立一个新的面向企业应用的框架结构，这个层次称为用户框架层，企业的具体业务应用都可以建立在用户框架层上，最终达到利用一种多层次框架结构统一所有业务应用的目的，这样就可以大大提高复用程度。如图1所示的多层次框架中，有三个不同粒度的层次，分别是基础框架层，用户框架层，业务逻辑层，自底向上其业务相关性越强。

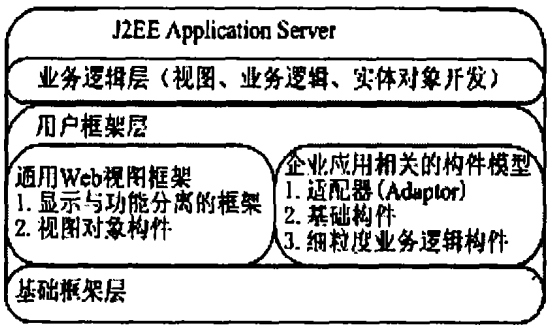

图1 多层框架结构 multi-tier framework

1)基础框架层：包含了 J2EE 应用服务器的基础服务，比如持久性，安全性，事务，负载均衡，监控，日志，应用集成，认证/权限/角色管理，Web 容器，EJB 容器等等。通过这些服务提供一个可以解决业务无关问题的基础框架。2)用户框架层：它是建立在基础框架层上，面向某种企业应用的框架结构，是对这种企业应用的共性的抽象。这里又包含了两类，即通用 Web 视图框架和企业应用相关的构件模型。3)业务逻辑层：业务逻辑层在通用 Web 视图框架基础上完成 JSP、Severlet、HTML 的实现；在企业应用相关的构件模型基础上实现商业逻辑，这样用户框架层的构件模型就得到复用，可以大大缩短开发周期。

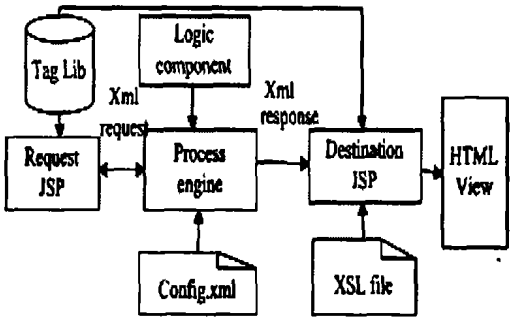

图2 通用 Web 视图框架

2.1.1 通用 Web 视图框架 在基于 J2EE 的 Web 应用中，经常看到 Java 代码与 html 强耦合在一起，这样就要求页面设计员具有界面设计和程序两项职能，同时程序审美性差、维护困难。因此大部分 Web 应用框架都要求视图显示与功能模块的分离，迄今已经有多种面向界面类的开发框架。Apache 的 struts 项目实现在 Web 应用中的 MVC 模型：1)Model 模型对象：是应用程序的主体部分。2)视图(View)对象：是应用程序中负责生成用户界面的部分。3)控制器(Controller)对象：是根据用户的输入，控制用户界面数据显示及更新 Model 对象状态的部分。Struts 不仅实现了在结构上功能模块和显示模块的分离，同时它还提高了应用系统的可维护性、可扩展性、可移植性和组件的可复用性；它的优点是结构清晰，缺点是结构复杂，抽象这三种模型对象很费时，特别是具有大量动态页面显示时。Webview 项目[4]致力于寻找一条 Web 技术与对象技术相结合的途径，基于 WebView 的 Web 应用框架实现了对象视图与源对象之间的分离，具有将不同类型的对象视图映射为相应的 HTML 实现的能力。

图2是本文要提出的通用 Web 视图框架，Tag lib 是自定义标签库，它包含了大量的可重复利用的标签，标签是由 JSP 引擎解析执行的，最终开发者需要做的是类似引用普通标签那样在 JSP 页面里嵌入到适当的位置，给标签赋予它要求的参量。

Process engine 是框架运行的核心，它的输入请求具有 XML 格式文档，请求可以定义为例1样式的 XML 格式，其中 trans 标签表示一次交易，每笔

交易由 request 和 response 两部分组成。

例1:xml request 的定义:

```
〈trans id="query * "〉
  〈request〉
      〈attribute1〉value1〈/ attribute1〉 …… …… 〈at-
      tributen〉value〈/attributen〉
  〈/request〉
〈/trans〉
```

根据请求的交易 id,然后 process engine 查找 conf. xml 配置文件,找到对交易的动做类(action class),然后执行调用这个类的对应的方法。本文假设所有的逻辑组件都提供一致的调用接口:xmldata action::execute (xmldata),函数执行完毕会返回 xml 格式(xmldata)的执行结果,执行结果可以是查询得到的一组相关信息,或者是简单的标识执行成功与否。

例2给出 conf. xml 的配置结构,trans 节点标点表示一次交易,requst 节点定义了本次请求要调用的对象、方法、参量。response 节点定义了业务逻辑执行完毕后,最终的 JSP 动态显示页面。例3给出了业务逻辑执行结果的一般结构,具体的属性需视具体应用而定。

例2:Config. xml 的定义:

```
〈configuration〉
    〈trans id="query * "〉
        〈request〉
            〈bject name="one actionclass"/〉
            〈method name="execute"/〉
            〈param name="xmldata" value=xmlrequs-
            t/〉
        〈/request〉
        〈response〉
            destination_page name="show_result. jsp "
            dir=" "/〉
        〈/response〉
    〈/trans〉
〈/configuration〉
```

例3:xml response 的定义:

```
〈trans id="query * "〉
    〈response〉
      〈attribute1〉value1〈/ attribute1〉 …… …… 〈at-
      tributen〉valuen〈/attributen
    〈/response〉
〈/trans〉
```

在 conf. xml 配置文件中的 response 节点下有个 destination_page,它定义了本次业务完成后要重定向页面,它要显示的数据为交易执行的结果,视图样式由 XSL 样式文件定义。例4利用自定义标签和 XSL 样式文件把 XML 结果数据显示成动态显示。

例4:数据的动态显示(show_result. jsp)

```
〈html〉〈body〉
〈xsl: transform xsl ="/source/xsl/customer/showbasicin-
fo. xsl"〉 Request. GetXMLData()
〈/xsl:transform〉
〈/body〉〈/html〉
```

通用 Web 视图框架对 Web 应用的框架进行了规范,例如视图与逻辑、视图与数据真正做了到分离的原则、基于 XML 数据的交互机制、基于自定义标签的页面逻辑封装等,这些特点使得基于这种框架二次开发的应用具有良好的扩展性。

2.1.2 企业应用相关的构件模型 企业相关的构件模型包含了以下三个方面:1)面向第三方系统和遗留系统的适配器(adaptor),通过适配器实现在新系统中对遗留系统的访问;2)面向企业应用的基础构件,比如加解密算法、数据交换格式、工作流引擎、报表/打印、特定行业/特定领域的模块等等;3)面向企业应用的细粒度构件,即可重复利用的原子业务逻辑,在业务逻辑层开发者可以组合这些原子业务逻辑形成复杂的业务逻辑。显然这三个方面构件都是针对某个具体企业需求的框架结构,这些构件必须在业务层开始编码前完成设计和实现,然后就可以在之后的开发周期中重复使用这些构件。

2.2 开发者角色

在基于 Web 应用框架的软件项目中,开发者有下面几个角色。项目经理:负责规划项目进度,控制成本;框架设计师:需求分析,软件架构,技术风险控制;美工:设计静态页面;JSP 程序员:动态页面,JAVA Bean 以及自定义标签编写;EJB 程序员:商业逻辑应用。

在已有成熟的应用框架的平台上开发时,过程模型跟直接在应用服务器上开发没有两样;如果不存在用户框架层,开发团队需要根据企业的特点来建立用户应用框架,然后再在用户框架上进行二次开发。用户框架层的开发不可能一次完成,在业务逻辑层的开发过程中,新的问题和需求也会导致用户应用框架的修改,多次修改后,最终形成一个稳定的开发框架。

从开发框架的不稳定可以看出,采用这种面向用户框架构建企业应用时,它的开发流程是有别于通用 Web 应用的。下面只描述流程各阶段不同于通用方法的地方:

1)在需求分析阶段,框架要求能够确定企业应用相关的构件模型,哪些第三方系统或者遗留系统需要适配器?系统需要哪些特定构件?哪些细粒度的业务逻辑可以抽象出来?这个阶段不需要实现每个构件,但必须清楚哪些构件要抽象出来。框架设计师和项目经理需要做好这些工作。

2)在设计阶段,页面设计员需要设计 XSL 格式的界面原型;框架设计师完成企业应用相关的构件模型设计与实现;EJB 程序员对商业逻辑对象进行详细设计时,要考虑构件模型的存在;JSP 程序员在通用 Web 视图框架基础上设计界面对象。在这个阶段,必须完成用户框架层实现,否则在代码实现阶段会妨碍业务逻辑的调试。

3)在实现阶段,如果开发人员发现用户框架层的不足之处,可以逐步丰富和完善,但必须控制修改

(下转第193页)

络结构为7—10—1，而预测成交量的网络结构为9—18—3。其收市指数及成交量序列的预测结果见图1、2所示。

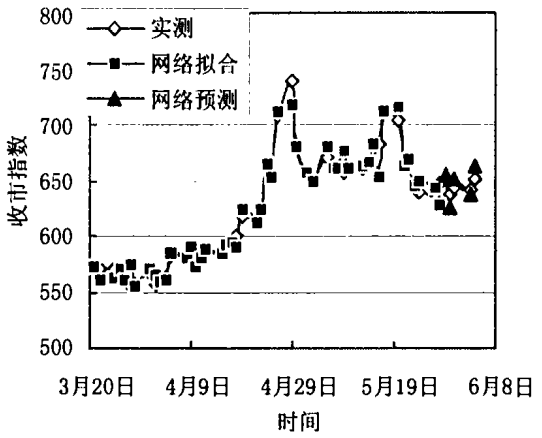

图1 股市收市指数实测与预测结果

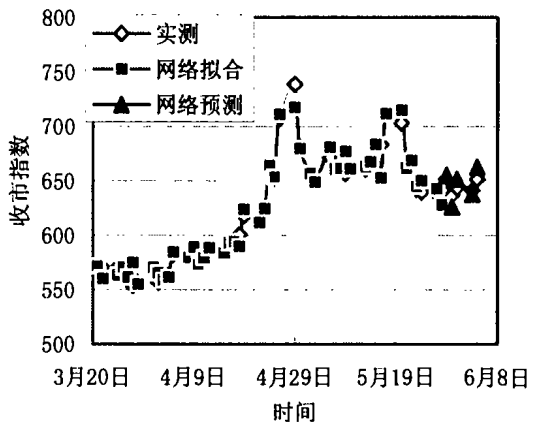

图2 股市成交量实测及预测结果

由上图1、2可见，本文提出的进化神经网络模型进行股市预测不但拟合效果良好，而且预测结果也很理想，是进行股市预测的一种较理想的方法。

结语 股票市场是一个受众多因素影响的非线性动态系统，对其演化规律进行预测对于证卷投资及市场管理均具有重大的理论及现实意义。本文通过对股市动态演化的数学描述，提出了一种进行股市预测的进化神经网络方法，并用沪市1996年的一段时间序列进行了方法验证，结果证明所提方法不但拟合效果好，而且预测结果也很理想，是进行股市预测的一种较理想的方法。

当然，任何模型、任何方法都只是从某些方面来反映或描述事物的特征，不可避免地具有一定的局限性。而且，由于我国股市建立时间比较短，市场法则法规尚不完善，会计制度、审计制度及上市公司信息批露等方面与国际惯例相差甚远。股市的不成熟常常使得国家政策、人为因素、有关消息及其他因素极易导致股市的不规则突变，这大大增加了技术预测的难度。因而，技术预测必须与基本面、政策面、消息面、资金面等相结合才能更有效地规避操作风险。同时，我们相信，传统的基本分析和技术分析与非线性科学、现代高新技术相结合将是证券投资分析方法发展的必然趋势。

参 考 文 献

1 吴奉，陈宏立. 经济预测与决策方法. 广州：暨南大学出版社，1991
2 陈茂申. 股市技术分析. 合肥：安徽人民出版社，1995
3 高玮. 非线性时序建模预测方法探讨. 清华大学学报，2000，40(S1)：6～10
4 高玮. 非线性信号预测模型研究. 计算机研究与发展，2000，37(S)：254～258
5 Refenes A N，Zapranis A，Francis G. Stock performance modeling using neural networks：a comparative with regression model. Neural Network，1994，7(2)：375～388
6 Schomeburg. Stock Price Predication Using Neural Networks：An Empirical Test. Neurocomputing，1991，2(1)：34～38
7 赵 建，邹永革，黄 炯. 基于神经网络的股市预测. 计算机研究与发展，1996，33(9)：692～697
8 刘新勇，贺江峰，孟祥泽，等. 基于神经网络的股市预测. 南开大学学报，1998，31(3)：39～44
9 Gao Wei. Study on new evolutionary neural network. International Conference on Machine Learning and Cybernetics，2003. 1287～1292

（上接第173页）

的规模。因为应用框架分割了模块之间的耦合性，各个开发者只需要专注于各自的领域就可以了，有效地提高了开发效率。软件的规模越大，框架带来的优越性也越明显。

如前面所述，在设计阶段就完成的一个应用框架不会是完善的，但是早期合理的抽象才会使应用框架具有普适性，才能避免在实现阶段或者 QA 阶段时底层框架的频繁大规模修改。因此评价 Web 应用框架是否成功，大约要从两方面着手，一方面是复用度，另一方是应用框架的历次修改对开发造成的影响。

结论 本文提出了针对 J2EE 平台的多层应用框架，抽象出了一个企业业务相关的用户框架层，把用户框架层分为通用 Web 视图框架和企业业务相关的构件模型，进而给出了框架的实现方式。从开发角度说明了本应用框架模型给应用开发流程带来的变化，框架本身不但可以提供良好的复用性，而且使开发流程更顺畅。

参 考 文 献

1 Gellersen H-W，Gaedke M. Object-Oriented Web application development. IEEE Internet Computing，1999，3(1)：60～68
2 Hoque R，Sharma T. Programming Web Component. New York：McGraw-Hill，1998
3 Johnson RE，Foote B. Designing reusable classes. Journal of Object-Oriented Programming，1988，1(2)：22～35
4 张波，冯玉琳，黄涛. 基于对象视图模型 WebView 的 Web 应用框架. 软件学报，2002，13(10)：1985～1990
5 王柏，王红熳，邹华. 分布计算环境. 北京：北京邮电大学出版社，2000