浙江工业大学

本科毕业设计外文翻译

（2012 届）

论文题目　基于 WEB 的 J2EE 的信息系统的方法研究

作者姓名　[单击此处输入姓名]

指导教师　[单击此处输入姓名]

学科(专业)　软件工程(专升本)1001

所在学院　计算机科学与技术学院

提交日期　[时间]

基于 WEB 的 J2EE 的信息系统的方法研究

摘要：本文介绍基于项目的 Java 开发框架背后的概念，并介绍它如何用于 IT 项目开发。因为有许多相同设计和开发工作在不同的方式下重复，而且并不总是符合最佳实践，所以许多开发框架建立了。我们已经定义了共同关注的问题和应用模式，代表有效解决办法的工具。开发框架提供：（1）从用户界面到数据集成的应用程序开发堆栈；（2）一个架构，基本环境及他们的相关技术，这些技术用来使用其他一些框架。架构定义了一个开发方法，其目的是协助客户开发项目。

关键词：J2EE　框架　WEB 开发

一、 引言

软件工具包用来进行复杂的空间动态系统的非线性分析越来越多地使用基于 Web 的网络平台，以实现他们的用户界面，科学分析，分布仿真结果和科学家之间的信息交流。对于许多应用系统基于 Web 访问的非线性分析模拟软件成为一个重要组成部分。网络硬件和软件方面的密集技术变革[1]提供了比过去更多的自由选择机会[2]。因此，WEB 平台的合理选择和发展对整个地区的非线性分析及其众多的应用程序具有越来越重要的意义。现阶段的 WEB 发展的特点是出现了大量的开源框架。框架将 Web 开发提到一个更高的水平，使基本功能的重复使用成为可能和从而提高了开发的生产力。

在某些情况下，开源框架没有提供常见问题的一个解决方案。出于这个原因，开发在开源框架的基础上建立自己的项目发展框架。本文旨在描述是一个基于 Java 的框架，该框架利用了开源框架并有助于开发基于 Web 的应用。通过分析现有的开源框架，本文提出了新的架构，基本环境及他们用来提高和利用其他一些框架的相关技术。架构定义了自己开发方法，其目的是协助客户开发和事例项目。

应用程序设计应该关注在项目中的重复利用。即使有独特的功能要求，也有常见的可用模式使用，这使得它们的设计和开发能重用。本文介绍了一个“自定

义”框架，这个框架用来定义能被开发者使用的相同的应用问题和定义设计模式。这个框架，我们将称之为某某开发框架，提供了一套模式和工具，建立了行业最佳实践，使之适合常见的应用问题。它提供了一个从表示到集成的应用程序开发堆栈。本文阐明了这些应用问题和模式，工具和行业最佳实践。某某开发框架可以根据各种项目的需求进行定制。它的开发和配置是基于诸如 Struts、Spring、Hibernate 和 JUnit 之类的各种框架和工具。

二、 开发框架的主要技术

2.1 代码和配置的层与层之间的分离

Web 应用程序有各种设计问题，如表现，商业逻辑，数据存取和安全性。不同的代码层的分离设计有如下几个方面的优势，如：便于维修，实施良好设计模式的能力，选择专门的工具的能力和具体问题的解决技术。将一个项目进行层与层之间的分离导致了这些层之间的依赖关系。例如，一个简单的使用案例，它涉及数据的输入和查询通常必须整合表示，业务逻辑和数据访问以达到所需的功能[3] 。因此，必须有一个明确的策略来管理这些依赖关系。开发框架包括设计模式，可复用的代码和配置文件，使开发框架尽可能地容易的被使用。这一框架使用 Spring 的反向控制来管理相依。Spring 框架[4]提供了一种方法整合各层成为一个应用项目。它通过 Spring 应用上下文来完成这一目标，这是一个对象之间管理依赖策略。Spring 使用的依赖注入和拦截技术介绍如下。

我们所写的代码依赖于使用的对象。它负责创建这些对象。这可能导致紧耦合的，但我们希望我们的代码是松散耦合。依赖注入是一个技术，可以帮助我们实现这一目标。依赖注入是反向控制（IOC）的一种形式[5]。当应用程序使用依赖注入时，代码将变得更加清洁和容易。这就是松耦合，从而更容易配置和测试。开发框架使用了多个 Spring 应用背景文件来定义层与层之间的依赖关系。方法拦截是面向方面编程（AOP）概念[6]。Spring AOP 方法拦截是通过 JDK 动态代理来实现的。开发框架使用 Spring AOP 来管理问如交易管理和性能监测等问题。

开发框架包括两个不同的部分：代码和配置。代码位于一个特定的应用层，并侧重于某一特定条件中的应用解决方案。这可能要与数据库交互，或将数据显示给屏幕。配置将应用的各个层联系在一起。从代码中分离出配置使我们能够独立管理配置，使我们在同一代码基础上方便的进行不同的配置。例如，数据访问

对象（DAO）知道它是使用 JDBC 通过数据源来连接一个数据库的，但它不知道该数据源是如何实现的。它可能是一个 Java 命名和目录接口（JNDI 上下文或是来自驱动程序。它可以指向远程数据库或本地数据库。无论数据来自何处，DAO 执行操作数据源的方式是相同的。同样，服务对象可能依赖于 DAO ，但不知道 DAO 是如何实现，可能通过 Hibernate，直接的 JDBC ，或 Web 服务。互动服务对象与 DAO 有相同的方式，而不管 DAO 的实现。

Spring 通过 Spring 应该上下文来管理我们的应用程序的整个配置，这些配置是一些 XML 文件。我们可以在一个文件中定义应用的环境。然而，我们可以在较小的文件中定义它来简化配置管理。这样的应用环境文件的逻辑集合组成了一个被称之为配置集的完整的应用配置。

开发基于 Java 的企业应用的标准配置是在一个框架的配置中设置使用如数据源和 JNDI 的资源的外部资源。这种类型的配置有些时候可能带来如下问题：（1）尚未加载完全的数据库。开发人员可能要测试某些类型的数据的显示，但如果基础数据尚未完成，他们将无法做到这一点。（2）服务或 DAOs 可能还未被开发。整合未完成的服务或 DAOs 可能阻碍发展的进程。

这些问题降低了生产力。开发框架已从它的代码中分散其配置，我们可以针对开发使用有选择的配置集。这可以减轻我们对外部系统的可用性的担心，这对于解决开发的中间需求是不相关的。

开发框架定义了两种配置集：默认和独立。我们还可以自定义应用，来增加基于我们项目需要的额外配置集。默认配置使用在 JNDI 中的定义的数据源来连接数据库。它完全使用了应用服务和 DAOs 。独立的配置设置对开发而言是最灵活的。此配置集：（1）使用 DriverManagerDataSource 连接到任何本地安装的数据库或开发数据库；（2）使用 Spring 的 DataSourceTransactionManager 作为本地事务管理；（3）利用充分开发应用服务和 DAOs；（4）充分利用 Spring 应用上下文在应用服务器以外进行运行和测试。

开发框架通过它的应用上下文进行配置。应用上下文被定义一个或多个 XML 文件。一个配置集是定义一个应用上下文的一套 XML 文件。配置集包括两部分：服务和网络。该服务定义了整合过程中的 DAOs 和资源。一个配置不能同时完成这些部分。

开发框架配置集通过被 Spring 称之为 bean 映射上下文组合到一起，这些映射在 beanRefContext.xml 和 applicationContextMapping.properties 中定义。beanRefContext.xml 文件定义所有的配置的服务部分。此文件位于的 src /服务项目的配置目录下。应用上下文之间共享也是通过这个目录下的配置来实现的。此外，各配置有自己的子目录，其中包含自己的特定配置。例如服务和 DAOs 通过配置集来共享，而支持服务（如数据源）则属于子目录。 XML 文件在应用程序通过使用<bean>标记来定义 Spring bean。Spring bean 是一个 Java 对象和通过应用上下文来初始化。

2.2 类及其关系

利用开发框架，在一个典型项目中有如下的代码和配置：（a）Action，ActionForm 类和 validation.xml 文件；（b）服务接口和实现类；（c）DAO 接口和实现类；（d）以上这些的关系管理。当我们开始我们例子的开发时，我们必须认识到所有这些类和他们的关系的重要性。

2.3 测试技术

测试应是项目开发过程中的一个不可分割的组成部分的。使用开发框架建立的应用程序，单元测试是指只测试服务或集成层的单一类。表现层（Action 类）不执行单元测试。这种测试的目的是保证每个类的行为封装与预期一致。项目中的单元测试是基于 JUnit 框架的[7]。与单元测试不同，集成测试需要测试代码之间的相互依赖性。这种测试的目的是以确保各个不同的类（不同的开发者开发的）整合在一起时也能想期望一样的运作。在功能测试过程中，重点是采用不同的场景进行功能的测试。典型的功能测试包括在业务层用不同的数据进行类的测试。

为了执行不同类型的测试，项目在开发过程中必须是测试可测试的。下面列出的可测试项目的一些基本特性。（1）开发单元的简单和集成测试。我们可以在没有数据源，或排队的情况下执行单元测试。当然，我们也能模拟相依赖代码而进行测试。（2）有易于进程各种模拟测试场景的功能测试。（3）在整个生命周期中方便重新运行所有测试。（4）从应用代码中清楚的分离出测试代码来。

精心计划应用的各个设计问题，如表示，服务和数据访问对于可测试的应用是非常重要的。应用程序编码以 get 方法、set 方法、变量等开始。单元测试是是其他任何测试方法的基础。开发框架设计的便利的可测试应用开发的方法：提

供测试模板类来帮助建立单元测试，使应用更易于配置以适应测试需求。单元测试可以运行像任何 JUnit 测试。默认的专门开发的“建设脚本”提供了一个任务来运行单元测试。这个任务部署的 EAR 文件，可以单独运行。

2.4 页面表示设计

开发框架采用 Struts 框架和 JavaScript 来实现页面，并提供可扩展用于另外项目的额外功能。当使用 Struts 框架进行发展，首先，我们在 web.xml 配置 Servlet Action；然后在 struts-config.xml 中配置 action mapping，form bean 和 local forwards；最后我们在 validation.xml 配置验证规则。

这种建立应用程序的方法在开发框架中已经发生了改变，开发人员不必要直接编辑 config.xml 或 validation.xml 文件。相反，我们通过 XDoclet 注释直接在 Action 和 Action Form 类中直接配置。这些信息在运行 Ant 脚本时翻译插入 struts-config.xml 和 validation.xml 文件中。

有两种需要验证的类型：数据格式验证和业务逻辑验证。数据格式验证最好在表示层进行，而业务逻辑验证最好的在服务层进行验证。在业务层发生的业务逻辑错误，必须通过抛出异常进行处理。

以下是表现层的设计目标：（1）每个 JSP 文件只有一个 Action 类和 ActionForm 类。一个单一动作类必须处理一个单一的页面；（2）使用 XDoclet 定义依赖和验证规则；（3）开发人员应该避免或尽量减少使用 sessoin 对象，因为它阻碍了可扩展性。

开发框架提供了一个默认的 Action 模板类，其中包含解决上述设计目标的方法。以下是典型的开发 web 页面所需要的代码：（1）创建一个带有称之为“actionType”默认隐藏域的 JSP 文件，用于处理在页面上发生的用户行为。(2)创建一个扩展模板 Action 类的新的 Action 类。我们必须使用 XDoclet 配置 ActionForm 和服务类之间的关联。然后，我们就应该针对隐藏域“action Type”中的值来建立具体的处理用户动作方法。最后，我们根据需要给这个 Actoin 类访问权限。这就是 Spring 的配置文件所做的工作。(3)创建一个新的 ActionForm 类，并用 XDoclet 注释指定验证规则。

一旦 JSP，Action 和 ActionForm 创建完成，就必须运行 Ant 脚本来重新生成“struts-config.xml”文件。

2.5 数据库访问

通过框架建立的应用程序支持直接使用 JDBC 和 Hibernate 框架将数据持久化到关系数据库中。应用程序通过 Spring 上下文文件进行配置。直接使用 JDBC 的 DAOs 必须继承 Spring 框架中的 JdbcDaoSupport.java 类。同样，使用 Hibernate 的 DAOs 必须继承 Spring 框架的 HibernateDaoSupport.java 类。

2.6 通过注释进行配置

开发框架使用 Spring 框架维持代码之间依赖。一些相依（例如 Action 和 ActionForm ）在“struts-config.xml”中配置，而另一些（例如服务和 DAO）在 Spring 应用上下文文件（applicationContext.xml）中配置。在一个团队中这些配置文件可以被开发者共享。就这是为什么在这些配置上可能发生版本冲突。开发框架提供了一种新的有效的办法，使用特别注释来定义这些依赖。通过使用这些注释，配置变得更加简单和相互冲突也可以避免。

三、 开发框架中的服务

开发框架提倡使用 plain-old-java-object（POJOs）实现业务逻辑。业务逻辑必需申明为接口。所有服务的实现必须实现一个或多个业务接口当有业务规则验证错误服务层抛出业务异常这是推荐的。开发框架采用基于 Spring 框架的事务处理方法[8]。这是通过面向方面编程（AOP）实现的。

开发框架提供了从服务接口(与应用程序的业务逻辑联系)分离部署接口(与服务消费者联系）的良好实践。在 WDSL 中部署接口是 Java 接口表示服务的外在表现。实现这个接口的类必须代表实现服务实现接口的类的需要。这样可确保所有的业务逻辑是保持在一个正确的地方。服务接口是一种 Java 接口，代表宝物逻辑。在大多数情况下，部署接口包含从服务接口而来的一个方法子集。

Apache Axis 1.2.4 Web 服务框架是目前 Web 服务的标准。开发 Web 服务，有两种不同的方法[9]：contract first 和 contract last。Contract first 与 contract last 的区别 WDSL 首先被创建还是从代码中生成。

当服务消费者或供应商的外部供应商时，他们可以使用不同的技术实现 WEB 服务（他们可以使用.NET 不是 Java）[10]时，Contract first 办法对开发 WEB 服务是一个很好的做法。

四、 中间层的集成

与外部资源的整合有多种技术，如数据库和 Web 服务。开发框架使用在逻辑层称之为“整合”层的技术。这一层的设计目标是：（1）通过 JDBC 或 Hibernate 进行的数据库访问必须封装在数据存取对象（DAO）中。（2）Web 服务应尽可能简单。（3）所有的外部数据格式转换到应用程序域对象应限于这一层。（4）在这个层单元测试类应用做的简单。

开发框架支持使用 Hibernate 和直接的 JDBC 调用访问关系数据库。使用 Spring 框架的模板类：JdbcTemplate 和 HibernateTemplate 是推荐的。当直接使用 JDBC 访问关系型数据库，建议应用程序的 DAOs 继承自 Spring 框架的 JdbcDaoSupport。JdbcTemplate 类管理访问数据库（例如 PreparedStatement）的资源。开发框架使用应用程序配置文件将 datasource 插入到 DAOs。当使用 Hibernate 访问关系数据库，通过应用程序配置文件将 Hiberate SessionFactory 注入到应用程序的 DAOs。

五、 开发生命周期

开发框架的重点在一个开发团队中可以明确定义角色及其相互作用的结构上。三种角色描述如下。这些角色的相互作用是整个应用项目成功的关键。（a）前端的开发人员专注于 JSPs，Action/ActionForm 类和外部 Web 服务。（b）服务的开发者专注于开发应用的服务和整合这些服务中的不同部分。（c）项目集成者主要专注开发的集成文件，如 DAOs 或消费型 Web 服务。

发展中的一个基本问题是在其依赖组件没有准备好或不可用时如何开发和集成的代码。开发框架通过以声明式注入“模拟对象”这种结构来解决这个问题，并在开发生命周期的过程中用实际对象取代模拟对象。由于我们的应用是通过不同的配置集进行配置的使这成为了可能。该框架使团队能测试开发过程中的一个组成部分。这使编写和运行 JUnit 测试成为了可能。框架专注于测试应用服务和他们的依赖性。应用部署在一个单一的 Enterprise Archive（EAR）文件中。Ant 脚本生成此 EAR 文件，并可以手动运行或定期调度。建议在创建 EAR 之前运行所有的 JUnit 测试。

六、 总结

本文作者概述了 J2EE 开发框架。作者讨论了 J2EE 项目中重要的架构问题，技术和发展步骤。这些资料来自实际的项目经验，是为了帮助开发人员构建基于

J2EE 系统和设计自己的框架。然而，这仅仅是冰山的一角，短短的文章不足以详细描述 J2EE 在科学和企业应用，特别是基于 Web 的非线性分析仿真软件的潜在影响。

参考文献：

[1] N.Hritonenko, Yu.Yatsenko.Creative destruction of computing systems: Analysis and modeling, Journal of Supercomputing, 38(2006), 143-154.

[2] Yu.Yatsenko, N.Hritonenko.Network economics and optimal replacement of agestructured. IT capital, Mathematical Methods of Operations Research, 65(2007), 483-497.

[3] A.S. Boranbayev, Reference Architecture for Web Applications, Reports of the National Academy of Science of the Republic of Kazakhstan, 5 (2007), 18-26.

[4] The Spring Framework official website: http://www.springframework.org/.

[5] Martin Fowler discusses details of the dependency injection pattern and how Spring.

[6] injects dependencies: http://www.martinfowler.com/ Spring reference manual's chapter on Aspect Oriented Programming with Spring: http://static.springframework.org/spring/docs/2.0.x/reference/aop.html#aop-introductiondefn.

[7] JUnit framework: http://www.junit.org/.

[8] Spring Framework official website - Chapter 9, Transaction management: http://static.springframework.org/spring/docs/2.0.x/reference/transaction.html.

[9] A.S. Boranbayev, Optimal Methods for Java Web Services, News of the National Academy of Science of the Republic of Kazakhstan, 5 (2007), 38-43.

[10] Spring Framework official website. Chapter 2: http://static.springframework.org/spring-ws/sites/1.5/reference/

html/why-contract-first.html.

[11] ……