

科技计划项目管理系统的领域框架研究

潘俊俊, 徐小良

(杭州电子科技大学计算机学院, 杭州 310018)

摘 要: 针对科技计划项目管理系统缺乏标准化、规范化的问题, 设计一个灵活、可扩展、松耦合的轻量级 Web 框架。通过领域分析与抽象, 提出基于生命周期的全过程管理领域模型。以 Spring 框架为核心, 整合 Struts2、JPA、JBPM3 构建分层的体系架构。构建具有良好扩展性、可维护性与移植性的领域框架。实际应用证明该框架在规范科技管理流程、提高系统开发效率等方面具有一定效果。

关键词: 项目管理; 领域模型; 框架; J2EE 架构

Research on Domain Framework for Science and Technology Plan Project Management System

PAN Jun-jun, XU Xiao-liang

(Computer School, Hangzhou Dianzi University, Hangzhou 310018)

【Abstract】 For problems of standardization in the field of science and technology plan project management system, this paper researches and designs a flexible, scalable, loosely coupled lightweight Web framework. Based on analysis and abstraction of the areas, it proposes management model with a whole life-cycle process, builds a J2EE stratified architecture that takes Spring framework as the core, and integrates Struts2, JPA and JBPM3. In the basis of two points, a domain framework is constructed, which has good scalability, maintainability and portability. The practical applications of the framework show that it has a significant effect on normalization of technology management processes and improving efficiency of system development.

【Key words】 project management; domain model; framework; J2EE architecture

1 概述

随着信息化的发展, 科技项目管理的信息化系统建设规模越来越大, 经费投入与从业人员不断增加, 业务逻辑也日益繁杂, 非常需要利用信息化手段进行项目管理。但是科技计划项目包含内容众多, 知识创新性、项目差异性大, 随着政策变更导致的需求变化也比较大, 项目管理缺乏标准和科学的过程控制。目前, 我国科技管理系统的研发存在着开发周期长、重复开发、柔性差等问题。领域框架作为一个系统的全部或部分复用设计, 是为开发人员定制的应用系统的骨架。因此, 构建一个科技项目管理业务标准化, 支持动态扩展和快速开发的轻量级领域框架成为目前需要迫切解决的问题。本文按照领域工程 3 个阶段(领域分析、领域设计、领域实现)^[1], 分别进行领域模型的分析、领域体系架构的设计和领域框架的设计实现。

2 领域建模

领域模型是领域中各系统共同需求的描述, 是领域分析活动的输出结果, 为系统开发提供可复用的软件需求规约^[2]。本文运用 UML 建模语言和一种面向对象的领域建模方法^[3]对科技计划项目管理领域进行建模, 以便领域分析向实现阶段平滑过渡。

2.1 领域角色模型

领域模型注重多方互动, 组织间的协同工作, 积极争取更多的技术外溢效应、组织多元参与联动和资源整合。领域模型主要涉及的组织结构包括申报单位、申报单位的归口部门、科技主管部门、专家组等几个部分, 具体如图 1 所示。

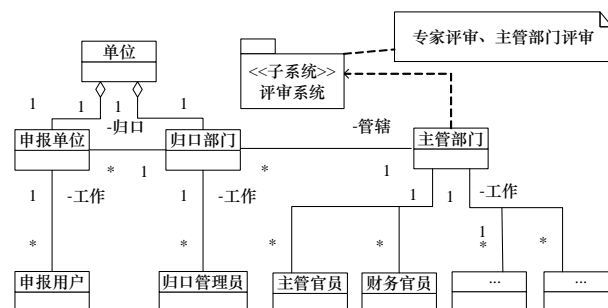


图 1 组织角色示意图

各个组织结构的角色主要有: 项目申报用户, 申报用户单位所在归口部门管理员, 科技计划项目主管部门人员(可能包括项目官员、财务官员、相关部门人员及领导), 项目评审专家。

各个角色之间的关系主要有: 申报用户依托申报单位发起项目申请; 归口部门负责项目申请的初步审核并逐级上报合格申请至科技主管部门; 科技主管部门通过评审子系统完成项目立项。

2.2 领域功能模型

领域功能模型是领域模型在功能需求方面的体现, 说明系统主要行为主体的具体活动。科技计划项目管理领域主要

基金项目: 浙江省重大科技专项基金资助项目(2008C11102)

作者简介: 潘俊俊(1984—), 男, 硕士研究生, 主研方向: 软件体系架构, 中间件技术; 徐小良, 副教授、博士

收稿日期: 2010-02-10 **E-mail:** justinchinazj@gmail.com

涉及项目、合同、中期检查、验收的填报、审核和管理等活动用例，如图 2 所示。模型中系统管理员具有系统配置管理的功能，系统配置包括用户管理、项目审批流程、项目计划类别、不同版本的合同和码表等配置。

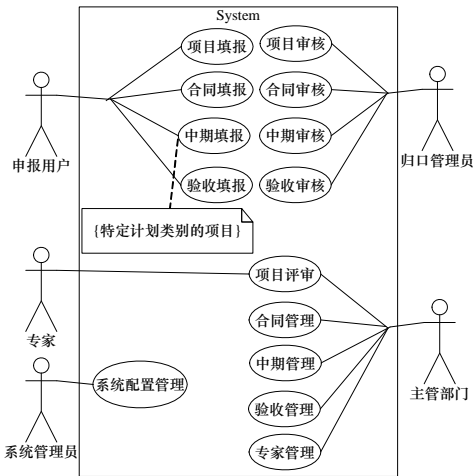


图 2 领域功能模型

2.3 领域业务模型

在运用系统工程的分析思路对我国科技计划体系及管理模式的现状进行分析的基础上，提出基于项目生命周期全过程管理^[4]的领域业务模型。全过程管理的业务模型不仅包括项目申报、立项评审和结题验收，在项目过程管理中还根据计划类别的具体特点、要求实行阶段性中期检查和评审，从质量、进度、财务支出等方面跟踪控制。因此，业务模型主要包括申报管理、专家评审、合同管理、中期检查管理、项目验收管理几大功能模块。如图 3 所示。其中每个“审核”活动都包含“审核通过”和“审核退回”两部分。主管部门的评审由评审子系统进行处理，评审子系统可包含专家评审、主管部门内部评审以及其他方式的项目评审。业务模型可将项目计划类别作为临界条件选择性执行阶段性中期检查。

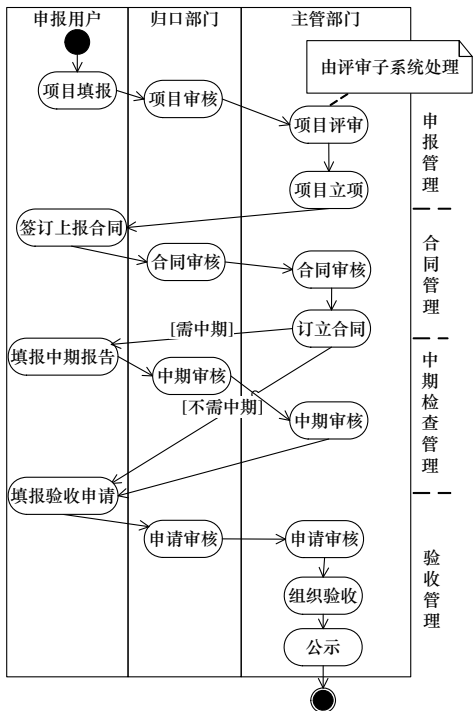


图 3 领域业务模型示意图

3 领域体系架构

领域架构主要借鉴 J2EE 架构的分层思想、面向接口驱动、面向切面编程等思想，搭建一个各层之间松度耦合的轻量级架构。

3.1 架构层次

在传统应用系统中，业务流程在系统中没有独立的层次体现，通常隐藏在应用系统的业务逻辑模块中。领域框架把业务流程从系统的业务逻辑层分离出来作为一个独立的架构层次，从而使传统的 J2EE 3 层架构变成了 4 层框架：表现层，业务流程层，业务逻辑层，持久层，如图 4 所示。

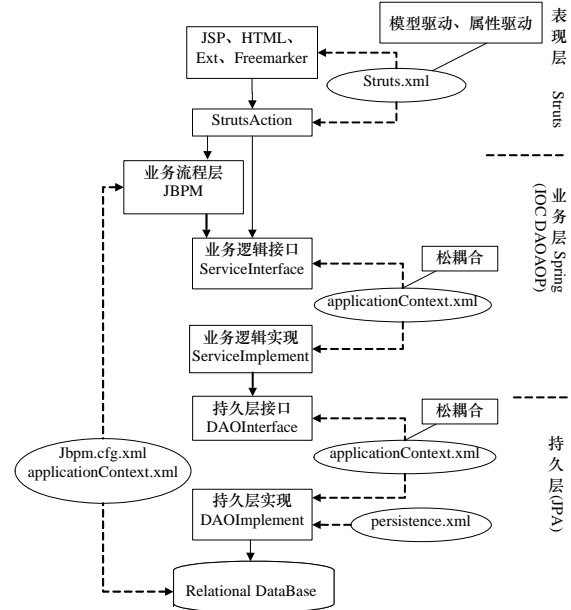


图 4 体系架构

领域框架以 Spring 为核心框架，通过对 Struts2 框架的整合，形成一个基于 MVC 模式的 Web 层；以 JPA 作为 Java 的持久化解决方案，解决了 ORM 框架之间不兼容的问题，使得系统具有良好的可移植性；选用 FreeMarker 充当模板支持，可以在任何场景下将数据模型生成标准输出；同时表现层前端的 Ext 作为 Ajax 框架，可以极大的增加用户体验。

架构利用 Web 应用描述符 web.xml、struts 框架的部署描述符 struts-config.xml 和 Spring 框架的应用上下文 application Context.xml 等配置文件，利用 Spring 的依赖注入 DI 或称控制反转 IoC 机制，将本来由应用程序本身主动控制的调用等逻辑转变成由外部配置文件来被动控制来启动 Spring 容器、获取应用上下文、整合持久层和为其他层提供业务服务对象调用接口，使各层协同工作。

3.2 业务流程

在引入工作流之前，系统的业务逻辑层通过多个业务对象的协作实现，业务对象之间耦合性比较强。引入工作流后，业务逻辑层由一些与流程无关的业务组件构成，流程实例通过对象 Adapter(适配器)模式调用这些流程独立的业务对象来实现一个业务流程需要完成的工作。JBPM3 屏蔽不同业务过程的差异，为系统提供了对业务过程统一管理的平台。

3.3 持久化

框架业务数据持久化采用 JPA 实现。JPA 作为统一持久层框架的标准数据操作接口，不依赖于任何特定的框架实现，也不依赖于特定的关系数据库，可以大大改善系统的移植性

和扩张性。同时使用 Spring 提供 EntityManager 来管理 JPA, 可以完全屏蔽 JPA 实现框架差异。JPA 透明的对象持久化和结合 JUnit 实现的自动建表功能是领域框架能够实现快速开发,具备通用性、可伸缩性、动态扩展性的关键。JBPM3 工作流引擎流程相关数据采用 Hibernate 框架实现对象关系映射。

Spring 同时支持编程式事务和声明式事务, 声明式事务可以分离业务逻辑和事务管理逻辑, 利用了 AOP 机制, 无侵入性, 具备良好的适应性。所以, 本框架选用了声明式事务管理。

工作流引擎负责工作流相关数据的访问、处理、持久化, 业务逻辑单元负责业务相关数据的访问、处理、持久化。这样的策略使得业务逻辑层可以应用任何合适的持久化方式, 而不受业务流程层持久化解决方案的影响^[5]。

3.4 Spring服务

框架的业务层通过 Spring 的声明事务特性实现事务封装, 业务层的 POJO 和逻辑组件都受管于 Spring 的 Application Context IoC 容器, 实现细粒度的依赖注入和层次间的松耦合。同时 Spring 的 ORM 支持和 JPA、Hibernate 的集成, 可复用已经存在的映射文件和标准的 Hibernate SessionFactory 配置, 无缝整合逻辑层和持久层。

4 领域框架设计与实现

4.1 模块设计

根据图 3 所示的领域业务模型可知, 框架功能覆盖了科技计划项目管理的整个生命周期, 包括了项目申报, 合同填报等模块。框架各模块的设计都是遵循图 4 所示的领域体系架构来实现的, 下面以项目申报模块为例进行设计。如图 5 所示, 功能模块基于 IoC 模式的领域服务注入实现层次间的松耦合。在表现层 ApplicationAction 通过获取业务层服务 IApplicationService 实现申请书信息保存和更新的过程中, 服务的创建点从应用程序转移到了基于 IoC 控制的 Spring 框架中, 由 Spring 自动装配(Assemble)服务的实现对象给服务使用者, 从而实现了服务使用者和服务提供者的解耦。

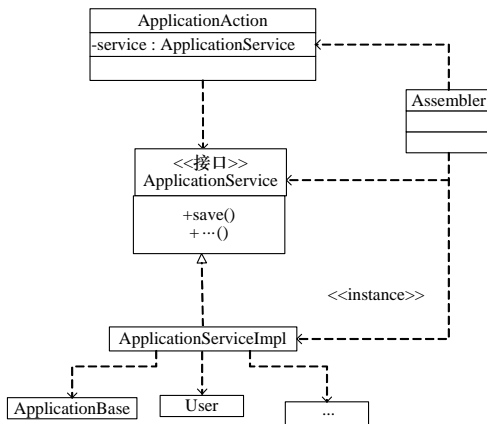


图 5 项目申报模块构建图

4.2 业务实体

域对象用于衔接业务层和持久化层, 同时它的托管对象被传递到表现层用于界面生成。本框架的域对象层由业务实体表示, 包括实体和实体关系。系统首先设计基础实体类 Base Bean, 类中封装了系统实体所共有的属性, 申请书、合同、用户等其他实体都泛化自基础实体类。不同版本的申请书和合同都需从各自的基础实体类 ApplicationBase 和 Contract Base 派生新类。实体间的继承采用 JOINED 策略, 这种设计

方式符合低耦合高内聚的原则, 支持多态查询, 同时方便利用实体监听器更新实体信息。JPA 的类继承和多态是框架实现扩展的关键技术。比较 XML 配置方式, Annotations 方式能够简化开发过程, 明晰业务逻辑过程, 因此, 框架选用 Annotations 标注实现实体间的关系映射, 如图 6 所示。

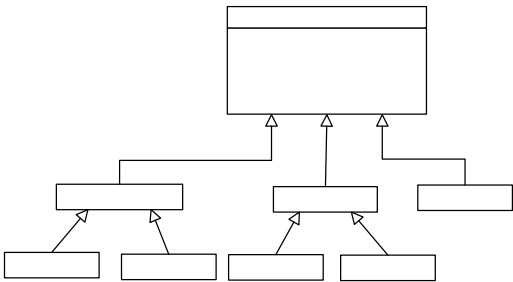


图 6 实体类结构图

4.3 业务逻辑

框架将业务逻辑层分成 Service 层与 DAO 层。Service 层为服务提供层, DAO 层即数据访问层。框架使用 Strategy、Adapter 等模式思想来设计业务逻辑组件。DAO 层的 IBase DAO 类封装公共的 DAO 操作接口, 由 BaseDAOImpl 实现。Service 层的基类 IBaseService 类封装各模块的公共接口, 并由实现基类 BaseServiceImpl 实现, 其他各模块的服务类都从服务基类派生, 服务实现类都从服务实现基类派生。Service 层和 DAO 层通过对象适配模式实现层次衔接。以项目申报模块为例进行说明, 如图 7 所示, IApplicationService 类在继承服务基类的基础上, 可定义新的业务方法和重载服务基类的业务方法, 而接口实现类 ApplicationServiceImpl 则在继承服务实现基类基础上具体实现新定义或者重载的业务方法。

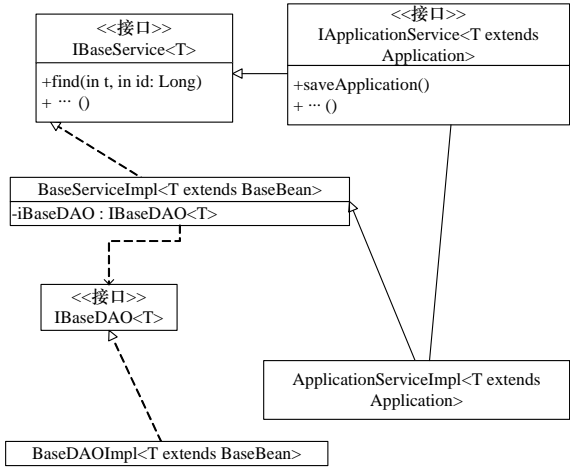


图 7 项目申报服务结构图

5 领域框架应用

基于本框架已经开发了浙江省科技厅网上申报管理系统、路桥科技局项目申报管理系统等。浙江省科技厅网上申报管理系统主要包括项目申报、专家评审、合同管理、中期检查管理、验收管理、项目审计管理等几大功能模块, 系统在领域框架的基础上, 扩展构建了十几种项目计划类别和多种版本的合同, 角色方面包括了多层的归口部门管理员、科技厅业务处和计划处及厅领导等相关人员; 在流程方面, 系统各模块的工作流允许动态配置。系统在实际应用中运行稳定, 有效适应了需求变化。

(下转第 279 页)