第 38 卷 第 4 期
Vol.38 No.4

计 算 机 工 程
Computer Engineering

2012 年 2 月
February 2012

• 软件技术与数据库 •　　文章编号：1000—3428(2012)04—0055—03　　文献标识码：A　　中图分类号：TP311.12

基于用户定义规则的软件平台研究与设计

肖立中，刘云翔，胡　婷，吴雁林

(上海应用技术学院计算机科学与信息工程学院，上海 200235)

摘　要：软件中的逻辑规则和数据多，对其定义和修改的工作量大。为此，提出一种基于用户定义规则的软件平台。从业务逻辑规则的角度出发，结合软件需求变更的特点，以数据字典、知识仓库和解释器为核心，以软件业务规则灵活化为主要目的，实现不同数据来源和类型规则的管理和解释功能，设计随需应变的软件平台。实验结果表明，该平台的开发和维护效率均较好。

关键词：用户定义规则；数据字典；知识仓库；解释器；工厂方法；软件平台

Research and Design of Software Platform Based on User-defined Rule

XIAO Li-zhong, LIU Yun-xiang, HU Ting, WU Yan-lin

(School of Computer Science & Information Engineering, Shanghai Institute of Technology, Shanghai 200235, China)

【Abstract】There are lots of rules and data in the logic of software, whose definition and modification are consuming. To overcome the above problems, a software platform based on user-defined rule is proposed in the paper, which can manage and interpret rules in different styles and data from different resources. Aiming at flexibility of software business rules, the platform thinks of rules of business logic, combines the characteristics of changing of software demands and orients modules including data dictionary, knowledge base and interpreter. Through the above research, a software platform convenient for demand changing is implemented which improves the efficiency of development and maintenance for software.

【Key words】user-defined rule; data dictionary; knowledge base; interpreter; factory method; software platform

DOI: 10.3969/j.issn.1000-3428.2012.04.018

1　概述

随着软件产业的飞速发展，应用软件的开发周期要求越来越短，原有的单一业务应用开发方式，即定制化、套件化、逐个对象实现的方式，已成为应用开发的瓶颈。因此，从 20 世纪 90 年代中期开始，软件基础架构平台的兴起以及业务基础软件平台的诞生，使得新的软件平台产业正在悄然而迅速地形成。

在快速开发方面，各企业开发的软件平台注重功能的组装，但针对以下问题没有给出解决方案：

(1)各种平台虽然可以实现功能组装，但是还需要用户的充分参与，用户只是给出需求功能描述，功能组装还要依靠软件人员来完成，在复杂系统中，软件人员由于不具备领域知识，组装会变得非常困难。另外，对应用软件业务逻辑的修改还停留在源代码级别，后期的测试工作量非常大，并且软件质量得不到保证。在本文平台中，用户可以直接在应用软件外定义业务逻辑，应用软件本身不需要任何修改，实现了用户定义，并且减少了软件测试的工作量，提高了软件的质量。

(2)软件重用方面，目前大多数平台主要集中在服务或组件的重用上，粒度较大，对其内部的逻辑规则重用没有考虑。由于使用环境和业务需求的变迁，服务或组件本身的逻辑规则需要经常发生变化，但其中的操作并不需要改变，由于这些逻辑规则的变化导致服务或组件的进化和重新配置的代价是很高的。

本文提出一种软件平台，用户只要遵循平台定义的语法规则，自行维护“知识仓库”与“数据字典”，就能维护软件流程规则的运行，达到规则升级而软件不需改写的目的。

2　软件平台设计

随需应变、数据共享和减少维护成本是搭建本文软件平台的目标，平台的概念模型如图 1 所示。

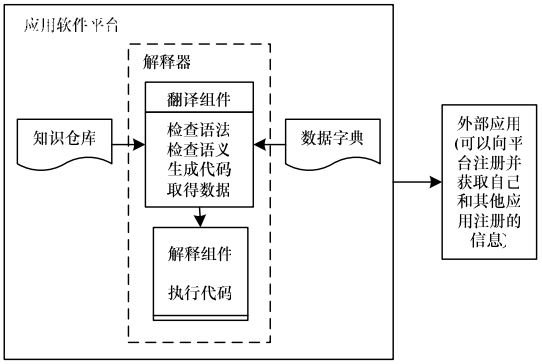

图 1　软件平台概念模型

2.1　平台的组成

本文平台是由数据字典(data dictionary)、知识仓库(knowledge base)和解释器(interpreter)的相互调用和整合，并配以 B/S 结构组成的。以下给出三者的作用：

(1)数据字典

数据字典用来存储用户的自定义数据。在应用软件中，

基金项目：上海应用技术学院计算机科学与技术重点学科基金资助项目；上海应用技术学院校内科技发展基金资助项目(KJ2011-03)

作者简介：肖立中(1981－)，男，副教授、博士，主研方向：网络异常行为检测，软件工程；刘云翔，教授、博士；胡　婷，讲师、硕士；吴雁林，本科生

收稿日期：2011-08-04　**E-mail：**lyexp@163.com

有很多数据由于环境和需求的原因而经常变更，如果在程序中将其写定，这些程序就会面临经常改写，而将这些数据定义在程序之外，软件可以很容易进行维护。

数据字典中包含数据来源注册器，用于管理数据来源。使得数据字典中能够定义不同来源的数据，包括数据库、XML、文本、设备等的信息。

(2)知识仓库

应用软件中有很多规则，定义这些规则需要充分了解领域知识，并且它们会随着需求的变迁而修改，它们往往由多种数据构成，无法在数据字典中直接定义，知识仓库的目的就在于存放用户定义的这些规则。

(3)解释器

本文的解释器主要完成对用户在数据字典中自定义数据和知识仓库中自定义规则的解释，得出所需的结果。解释器能够对数据和规则进行词法分析、句法分析和语义分析，得出能够被计算机识别的数据结果集。

2.2 平台的工作原理

平台的工作原理描述如下：软件中的规则并不定义在软件中，而是定义在知识仓库中，当软件执行过程中需要获取规则时，便由解释器到知识仓库中读取规则，并进行解释。规则中的数据也不定义在软件中，而由用户定义在数据字典中或由知识仓库中的规则得出，因此，当解释器遇到数据时，首先在数据字典中查找是否有相应的数据定义，如果有，便对数据进行解释，获得对应的值，否则，到知识仓库中查找相应数据的规则定义，并对该规则进行解释，如此反复，直到得到相应的数据。解释器从数据字典和知识仓库中获取数据的流程如图 2 所示。

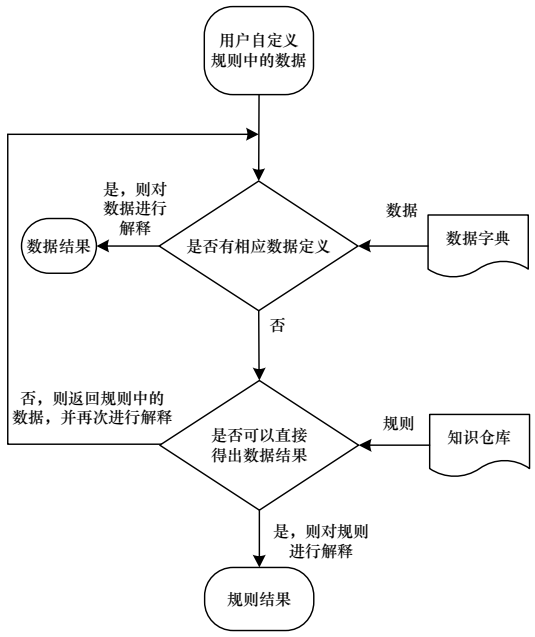

图 2 软件平台工作流程

3 软件平台实现

3.1 平台实现技术

平台的实现技术主要包括以下 4 项内容：

(1)基于 S2SH 的框架整合

S2SH，即 J2EE 项目中的 Struts2+Spring+Hibernate 三大框架的整合。平台核心组件是由 S2SH 框架进行整合开发而成的，它的主要功能是管理数据字典和知识仓库，并利用解释器对规则进行解释。

(2)基于工厂方法模式的业务处理

设计模式(design pattern)是一套被反复使用、知名度很高的、经过分类编目的、代码设计经验的总结。设计模式中非常重要的原则为“开-闭”原则[1-2]，它的含义为：一个软件实体应当对扩展开放，对修改关闭，即在设计和开发一个模块时，应使其可以在不被修改的前提下被扩展。

工厂方法[3]模式为设计模式中的一种，它定义为：定义一个用于创建对象的接口，让子类决定实例化哪一个类，工厂方法使一个类的实例化延迟到其子类进行。由于解释器需要解释来自不同数据来源的数据及不同类型的规则，因此需要多个解释器，同时，为了适应数据和规则的变化，解释器应该是很容易扩展的，即符合“开-闭”原则，工厂方法模式正是解决这一问题的良方。

(3)基于 JSON 的数据交换格式

JSON[4-5](JavaScript Object Notation)是一种轻量级的数据交换格式，基于 JavaScript(Standard ECMA-262 3rd Edition-December 1999)的一个子集。JSON 采用完全独立于语言的文本格式，但是也使用了类似于 C 语言家族的习惯，这些特性使 JSON 成为理想的数据交换语言。

基于 JSON 的数据交换格式在软件平台的设计中主要运用在以下 2 处：

1)核心解释器。解释器所解释的内容是由 JSON 组成的字符串表达式，表达式的格式是根据不同的数据来源和规则设定的，新扩展的解释器只需要根据表达式所设置的格式对其进行翻译和处理，便可以获得所需要的数据。

2)与外界的数据交换。解释器最终将解释的结果重新封装成 JSON 的数据交换格式，从而与外部数据交换。客户端发送了请求后，收到的是由软件平台发送回来的 JSON 数据。

(4)基于 ExtJs 的 RIA 应用技术实现

ExtJs 是用来开发 RIA(Rich Internet Application)的 AJAX 应用，主要用于创建前端用户界面，是一个与后台技术无关的前端 AJAX 框架。ExtJs 提供了许多易用的 Web 控件，可以被用于快速地开发 Web 应用程序的前端用户接口。

3.2 平台架构设计

本文软件平台架构具有 5 个层面，并配置信息安全管理体系，以保证信息的安全和完整性，平台的架构设计如图 3 所示。

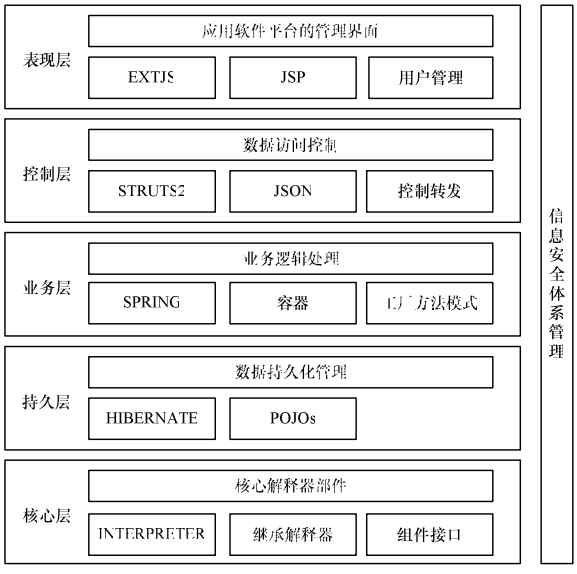

图 3 软件平台架构

在图 3 中，表现层主要负责软件平台的管理界面，配合 ExtJs 技术实现 RIA 的应用，由于与 Struts2 的整合，因此以 Jsp 作为输出页面。

控制层主要负责数据的访问控制，对发送来的请求进行筛选和处理，将错误请求转交给错误处理程序，将可执行的请求转交给业务层处理。这主要是运用了 Struts2 框架的技术，在业务层处理完请求后，发送相应的页面返回给表现层，数据交换方面则是用到了 JSON 技术。

业务层的职责是接收控制层转发的请求并处理，该层使用 Spring 容器管理业务 Bean，增强了业务间的交互性，也减少了业务间的耦合程度。同时启用了 Spring 基于注解的模式，有效地减少了配置的代码量，增强了系统的可控性。

持久层的功能是对数据进行持久化管理，运用 Hibernate 框架技术，对数据库进行管理，整合 Spring 后，将相关的配置工作转交给 Spring 容器负责管理，减少了配置数据库的代码量。

最底层为核心层，前面的技术和功能都是在为该层的调用提供服务。该层的关键是解释器，完成对数据和规则的解释，解释器符合“开-闭”原则，对扩展解释器开放，对原有解释器的修改关闭。

4 软件平台实验

平台由 Java 作为开发语言，主机操作系统宜采用 Linux，应用程序服务器可使用 Tomcat、JBoss、GlassFish 等基于 J2EE 平台的服务器，数据库服务器使用 MySQL。

平台的功能主要包括 5 个内容：数据来源类型，数据来源，数据字典，知识仓库和解释器的定义。通过以上操作，用户可以把应用软件中需要的规则和数据定义在平台中，而不是定义在应用程序中，可以方便地增加和修改，实现用户定义，而不必修改应用程序本身。

在实验中，模拟了来自某台计算机设备的监控信息和数据库 2 种数据来源。数据来源的信息分别如下：

(1)数据来源 1

IP 地址：192.168.1.102；

端口(port)：80；

应用程序名称(appName)：WebApp；

命名空间(namespace)：monitorInfo；

Struts Action(action)：json；

(2)数据来源 2

数据库：MySQL；

连接字符串(jdbcUrl)：jdbc:mysql://localhost:3306/appdemo；

用户名(username)：root；

密码(password)：123456；

设计的 JSON 表达式为：

{"ip":"localhost","port":"8084","appName":"WebApp/","namespace":"monitorInfo/","action":"json"}

设计 JSON 表达式为：

{"jdbcUrl":"jdbc:mysql://localhost:3306/appdemo","username":"root","password":"123456"}

对于数据来源 1，平台返回了监控信息。

对于数据来源 2，本文在数据字典中又进行了如表 1 所示的定义。

表 1 一组数据字典项

编号	名称	说明	类型	内容
1	basicsalary	基本工资	2	"{"sql":"select BasicSalary from Salary"}"
2	gwsalary	岗位津贴	2	"{"sql":"select gwSalary from Salary"}"
3	zhusu	住宿费	2	"{"sql":"select zhusu from Salary"}"

在表 1 中，“类型”为 2 表示输入为数据来源 2。通过以上定义，平台可返回如表 2 所示的数据结果。

表 2 与表 1 对应的数据结果

编号	结果
1	{"results":3,"data":"[{\"basicsalary\":\"2000\"},{\"basicsalary\":\"3000\"},{\"basicsalary\":\"5000\"}]"}
2	{"results":3,"data":"[{\"gwsalary\":\"30\"},{\"gwsalary\":\"40\"},{\"gwsalary\":\"100\"}]"}
3	{"results":3,"data":"[{\"zhusu\":\"500\"},{\"zhusu\":\"800\"},{\"zhusu\":\"2000\"}]"}

从表 2 可以看出，该数据库有 3 条记录。对应数据字典中的以上数据可以在知识仓库中定义如表 3 所示的规则。

表 3 知识仓库中的一条规则

编号	名称	说明	规则
1	salary	实得工资	"{"formula":"#\{basicsalary\}+#\{gwsalary\}-#\{zhusu\}"}"

以上规则表示：实得工资=基本工资+岗位津贴−住宿费。

通过知识仓库解释器可得出规则结果：

["1530.0","2240.0","3100.0"]

可见，针对数据库中的 3 条记录，得出了“salary”的结果，同时“salary”还可用于其他的规则定义中。

5 结束语

针对软件业务规则实现灵活化的目的，本文研究并实现了用户定义规则的软件平台，完成了不同数据来源数据、不同类型规则的管理和解释，实现了用户自行定义规则的目的，通过实验证明，该平台能够解释不同数据来源的数据和用户定义的规则，提高了开发和维护的效率。另外，对数据库的复杂规则和对字符的规则定义的解释是今后的研究内容。

参考文献

[1] Shalloway A, Trott J R. 设计模式精解[M]. 熊 节, 译. 北京: 清华大学出版社, 2005.

[2] Elaasar M, Briand L C, Labiche Y. A Metamodeling Approach to Pattern Specification[C]//Proc. of the 9th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems. Berlin, Germany: Springer-Verlag, 2006: 484-498.

[3] 伍 星, 王 茜. 设计模式在 HTML 解析器中的应用[J]. 计算机工程, 2005, 31(2): 89-90.

[4] Aziz A, Kollhof J K. JSON-RPC 1.1 Specification[EB/OL]. (2006-08-07). http://json-rpc.org/wd/JSON-RPC-1-1-WD-20060807.html.

[5] 黄 强, 王 薇, 张晓梅, 等. 基于 JSON 和 IoC 的 AJAX-RMI 插件[J]. 计算机工程, 2009, 35(19): 71-74.

编辑 任吉慧