基于数据模型驱动的业务组件自动生成*)

尹彦均　李亚芬　王　普
(北京工业大学电子信息与控制工程学院　北京 100022)

摘　要　为加快 Web 信息系统开发，减少软件开发人员的重复工作，实现系统设计到代码的快速转化，本文提出了一个基于数据模型驱动开发的方法，实现了由数据模型到业务组件的自动生成。从分析 J2EE 结构入手，确定了生成的目标代码，讨论了数据模型中描述业务组件信息的内容，定义了该数据模型结构，设计了基于模型驱动的代码生成器，生成了业务组件中各类代码。最后介绍了一种基于模型驱动开发模式，该模式改变了传统的手工编码方式，提高了软件开发速度。

关键词　模型驱动，数据模型，代码生成器，J2EE，Web 应用快速开发

Data Model Driven for Automatic Generation of Business Component

YIN Yan-Jun　LI Ya-Fen　WANG Pu
(College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100022)

Abstract　Towards quicken the development of Web information system, reduce the software developer's repeating work, implement the fast translation from system design to code. The paper brings forward a method of model-driven development based on data model, and achieving the automatic generation from data model to business component. In this paper, starting with analyzing the J2EE architecture, ascertaining the target code generated, discussing contents of data model used for describing the information of business component, then making a definition on the structure of data model, following designing the code generator to implement model-driven development and generating all the code in business component. At last introducing a development pattern based on model-driven, this development pattern change the traditional manual coding, improve the speed of software development.

Keywords　Model-drive, Data model, Code generator, J2EE, Fast Web-based application development

1　引言

随着近几年 Web 技术的不断发展，出现了许多优秀的 Web 应用框架，如 Struts、Spring、Hibernate 等，降低了 J2EE 的开发难度，提高了 Web 信息系统的开发速度。但涉及底层业务组件开发时，仍无法摆脱 J2EE 中大量重复代码的编写，严重地影响了软件开发速度。代码自动生成是解决这一问题的理想方法，可以实现系统设计成果到代码实例的快速转化，满足现在信息化建设的需求。

目前，基于模型驱动[1]的代码自动生成是一种比较流行开发方式，其中 OMG 组织提出的模型驱动架构 MDA[2]，定义了软件自动生成的宏伟蓝图。但 MDA 中涉及的模型过多、模型间转换过于复杂，还不能普遍用于商业性开发。本文则提出了更为实际的模型驱动方法，基本思想是通过建立描述业务组件的数据模型，结合 J2EE 开源框架，由代码生成器实现数据模型到 J2EE 中业务组件的自动生成。使开发人员仅通过编辑业务组件的数据模型，就可以快速地生成业务对象、数据库访问逻辑以及业务操作的大量代码，提高了软件开发速度。

2　基于开源框架的 J2EE 体系结构

本文采用开源框架 Struts、Spring 和 Hibernate 作为 J2EE 体系结构的一种轻量级实现。其中 Struts 负责 Web 展现层，Spring 框架实现整个应用的配置和管理，Hibernate 完成数据库 ORM 映射工作。这种分层设计有助于系统软件模块的解耦，实现软件项目分工开发。同时，也有利于代码自动生成的设计。代码生成器只要针对各层次的特点，分别生成 J2EE 各层所需要的源代码即可。由于本文仅涉及业务组件的代码自动生成，因此前端展现层设计不再赘述。下面主要讨论 J2EE 中业务组件开发中所涉及的层次。

2.1　域模型层 Domain Layer

域模型，即业务对象，承担各层之间数据通信，域模型层中的对象关系描述了系统业务。本文用 POJO[3]作为域模型，保存业务对象的数据，它是一种简单的 Java 对象，不用实现一些特殊的接口，只包括对象字段属性和相应的 setter/getter 方法。它

*)基金项目：北京市自然科学基金资助项目(4052010)。尹彦均　研究生，主要研究方向为快速开发 Web 应用，代码生动生成等；李亚芬　高工，硕士生导师，主要研究方向：Web 应用基础研究、快速开发 Web 应用，Web 组件技术等；王　普　教授，博士生导师，主要研究方向：计算机应用，数据库技术，网络技术等领域。

保持了良好的面向对象设计风格，便于测试和业务重用。在该层中，根据系统中的业务对象创建 POJO 域模型。

2.2 数据访问层 Dao Layer

在数据访问层，引入了 DAO(Data Access Object)模式，即数据对象访问模式，实现了业务逻辑与数据访问逻辑相分离，使该层中仅包含数据访问逻辑，供业务服务层调用。本文借助 Hibernate 提供的 ORM 框架，实现了对数据库中的数据 CRUD 操作的封装。在该层中，设计了域模型的数据访问接口 Dao 以及对应 Hibernate 实现类 DaoHibernateImpl。

2.3 业务服务层 Service Layer

业务服务层采用了 Facade 门面模式，为用户定义统一的入口点，作为业务组件对外界提供的服务接口。在该层中，业务逻辑是系统实现的核心，但具体的业务逻辑通常导致代码的高度耦合，该层借助 Spring 容器可以很好地实现代码逻辑的解耦，使系统中组件的配置更加灵活，提高了扩展性。在该层中，定义了与 Web 展现层交互的服务接口 Service 和具体的业务逻辑实现类 ServiceImpl。

3 数据模型的设计

在基于模型驱动开发中，模型设计是整个过程的关键，它是实现模型到计算机语言转换的首要问题。设计良好的模型可以使用软件开发和维护的过程全部围绕模型编辑进行，并与具体的 J2EE 平台实现无关，便于移植。本节中为自动生成业务组件，从面向对象角度设计和分析了业务对象模型，定义了一种数据模型结构[4]。

3.1 数据模型描述的内容

在传统的信息系统开发中，一方面要用面向对象技术来设计业务对象模型和进行关系型数据库设计，另一方面还要建立二者间的映射转换关系，增加了开发和维护的工作量。为解决面向对象结构与面向关系结构间的技术鸿沟，在数据模型中要提供对业务对象、业务对象关系的信息，同时提供相应的数据表及表间关系的映射描述，实现从面向对象结构推导出关系型数据结构。

(1)业务对象的描述

业务对象描述包括对象名称，对象属性等信息，其中业务对象映射为数据表，属性映射为数据表字段。

(2)业务对象间关系的描述

针对业务对象的特点，对象间关系包括关联关系和继承关系的描述。关联关系包括一对一、一对多和多对多关系，其中对象间多对多关系需要用关系表来描述；面向对象中的继承关系映射为关系型数据表有三种策略：

- 父类没有物理表，而每个子类都对应一张数据表；
- 主扩展表结构；
- 父类、子类描述在同一张数据表中。

3.2 数据模型结构

根据上述分析数据模型中描述的内容，定义了图 1 所示的数据模型结构来存储信息。

在图 1 中以 UML 图的形式描述了数据模型结构，其中用元素来描述业务对象信息，而用元素属性来描述数据库映射信息。该数据模型采用 XML 文档来描述业务模型中的元数据，作为代码生成器的输入源，通过 XML 解析器按模型结构可以在内存中建立一个 DOM 树模型，方便数据的读取。

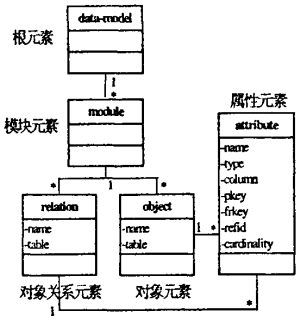

图 1 数据模型结构的 UML 描述图

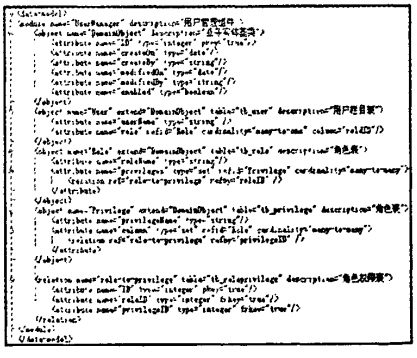

图 2 描述用户管理组件的数据模型文档

图 2XML 文档描述了用户管理组件模型，定义了用户、角色和权限 3 个业务对象和一个业务对象公用基类，并描述了业务对象之间的关联关系和继承关系，刻画了组件信息，为下一步生成业务组件提供了数据模型。

4 业务组件的生成

为实现数据模型到业务组件快速转换，需要引入代码生成器[5]来自动生成业务组件。图 3 中描绘了基于数据模型驱动的代码自动生成模型。

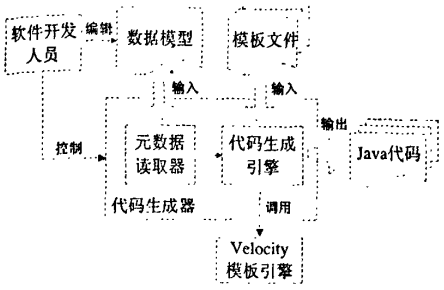

图 3　基于数据模型驱动的代码自动生成模型

在图 3 中，代码自动生成模型由四部分组成，包括数据模型、模板文件、代码生成器、控制部分。其中代码生成器在生成目标代码时，需要调用 Velocity[6]来完成模板中可变数据部分的替换工作。

这种基于数据模型驱动的开发流程是，软件开发人员首先分析业务组件需求，如业务组件中的对象、对象间关系等，然后编辑数据模型将业务对象表达出来，作为代码生成器的输入源。建立好模型文档后，就可以控制代码生成器读取模型文件，并抽取出元数据，根据加载的模板文件，由代码生成引擎将元数据和模板文件绑定在一起来生成目标代码，并最终生成整个业务组件。当业务发生变更后或需要维护时，软件开发人员只要修改数据模型就可以重新生成业务组件即可。

4.1　模板设计

模板是最终生成代码的原型，根据模板可以生成规范、可用的代码。良好的模板设计，包含了软件设计和开发人员多年的经验，应用大量的模式，封装了代码的最佳实践，确保了生成代码的品质，并可以生成大部分的业务逻辑。由于要生成 J2EE 各层次所需的目标代码，本文设计了 5 种代码模板，如表 1 所示。

表 1　代码生成器中所用到模板文件

模板所在层	模板名称	模板说明
域模型层	Pojo.vm	POJO 业务对象模板
数据访问层	Dao.vm	业务对象的数据访问接口模板
	DaoHibernateImpl.vm	基于 Hibernate 的数据访问层接口实现模板
业务服务层	Service.vm	服务层接口模板
	ServiceImpl.vm	服务层接口实现模板

其中，POJO 模板中主要包括业务对象的属性定义以及相应的 setter/getter 方法；在数据访问层的模板中，设置了对数据表的 CRUD 操作的固定逻辑，并保留了可变化信息；在业务服务层的模板中，定义了服务接口的调用方法。通过这些模板的设计，为自动生成代码提供了原型。

4.2　代码生成器设计

代码生成器包括两部分，即元数据读取器和代码生成引擎，分别完成数据准备和代码生成的工作。

4.2.1　元数据读取器

元数据读取器主要负责读取数据模型的 XML 描述文件，并在内存中生成 DOM 树模型，为代码生成引擎的输入源。其处理算法过程分为两步：

第一步：检查数据模型文档的语法、结构组织是否无误，验证录入的元数据类型是否正确。如检查出文档内容不符合文档规定，则抛出异常信息；

第二步：读取业务模型描述文件，抽取出其中的元数据，并加载到内存中，以待访问。

4.2.2　代码生成引擎

代码生成引擎设计，采用了 Factory 工厂模式，定义一个代码生成器的接口 Generator，及 5 个具体代码生成类，负责生成 J2EE 中的 POJO、Dao、DaoImpl、Service 和 ServiceImpl 代码。在使用时，由工厂类 GeneratorFactory 获得某一具体的代码生成器实例，生成特定的源代码。其类图如图 4 所示。

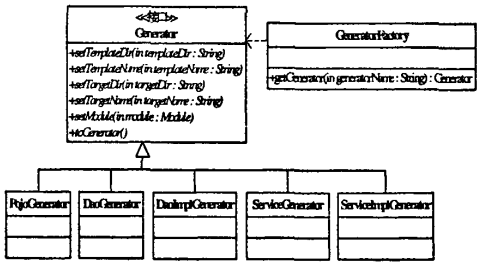

图 4　代码生成引擎类图

在图 4 中，Generator 类定义了代码生成引擎的接口，主要的接口声明有：

public void setTemplateName (String templateName)；//设置模板文件

public void setModule(Module module)；//设置业务数据模型

public void toGenerate()；//将业务数据与模板绑定，生成源代码文件

5　基于模型驱动开发的实际应用

在实际的 J2EE 项目开发中，使用 Ant 项目构

建工具来引入代码生成器，只需定义好 build. xml 构建文件，就可以帮助开发人员自动完成工程目录的建立，配置代码生成器所需的数据模型和模板文件，控制和操作代码生成器来生成、编译源代码，并打包成业务组件等。同时，Ant 还可以调用 xDoclet 工具，根据 POJO 源代码中的属性声明生成 Hibernate 的映射文件，以及数据表 Scheme。

基于上述的开发模式，如果要开发信息系统中一个用户管理的业务组件，只需要用 XML 文档描述出其数据模型，然后利用 Ant 调用代码生成器和 xDoclet 就能生成完整的业务组件。对于其中的一个用户对象，就会一次性生成 User、UserDao、UserDaoHibernateImpl、UserManagerService、UserManagerServiceImpl、user. hbm. xml、user. sql 等 7 个文件。可见，Ant 工具、xDoclet 工具和代码生成器结合的这种模型驱动开发模式改变了传统的手工编码方式，提高了软件开发速度。

结论 本文分析了 J2EE 体系结构的特点，确定了生成业务组件的目标代码，定义了一个数据模型的结构，它完整地描述了业务组件中对象及对象关系的信息，并设计出基于模型驱动的代码生成器来实现由数据模型到业务组件的自动转换。最后介绍了结合 Ant 等工具进行模型驱动开发的模式，从而实现自动生成 J2EE 中各层次的源代码。

这种基于模型驱动的业务组件生成方法在实际应用中非常有效，在大规模工程项目中效果尤为明显，减轻了软件开发的工作量，加快了 Web 信息系统开发的速度。但本文中对于数据模型的编辑还仅限于手工方式，在涉及业务对象关系描述时比较复杂，对开发人员要求较高。今后，还需要不断弱化该数据模型编辑的难度，提供可视化的方式来生成该文档，使用这种模型驱动开发方式变得更为简单、快速。

参 考 文 献

1 徐长梅. 基于数据驱动操作模式的 J2EE 应用自动化生成方法. 长沙大学学报，2006，20(5)：67～71
2 王继瑞. 基于 J2EE 平台的代码生成器. 山东：山东大学，2006
3 Richardson C. POJOs IN ACTION. USA：Manning，2006. 12～14
4 Jakob M，Schwarz H，Kaiser F，et al. Towards an Operation Model for Generated Web Applications. ICWE'06，July 2006
5 Herrington J. CODE GENERATION IN ACTION. USA：Manning，2003. 32～25
6 The Apache Software Foundation. The Apache Velocity Project. http://velocity. apache. org/

(上接第 212 页)

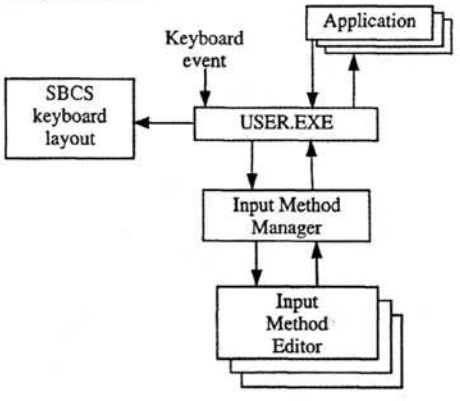

Fig. 2 Communication between the IME and applications

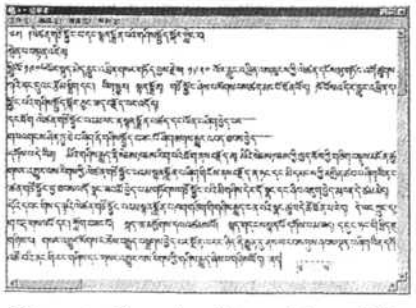

Fig. 3 The composition and candidates windows of Tibetan IME

Tibetan Web Server：www. tibetaninfo. com

5 Tibetan Truetype Pont (TTF)

The Unicode standard is used in TrueType fonts as the main character identification method. In principle TrueType fonts may be encoded with different standards，but in Windows Unicode is always used. Tibetan coding is based on the ISO/IEC 10646-1Tibetan. TTF font file be consists of three parts：offset table，table directory and other tables. TTF is able to describe not only English character but also Chinese and Tibetan character. So Tibetan font employs the TTF with Unicode. The Tibetan TTF is created by the font tool of Fontographer and Fontlab.

6 Conclusion

The construct and its characteristics of Tibetan information interchange system on Internet are described. The methods of design and implementation of Tibetan information interchange system are proposed. The design theory and implementation for Tibetan information system interface，Tibetan IME and Tibetan TTF have been discussed，and the Tibetan information interchange system has been realized. Tibetan Web Server has been created.

References

1 Petzold C. Windows Programming. Microsoft Press，1999
2 David J，Scot Wingo. Visual C++6. 0 Programming. Microsoft Press，1999
3 Microfost. Win32 Multilingual IME Application Programming Interface，Win NT SDK
4 Microfost. Win32 Multilingual IME Overview for IME Development，Windows NT SDK
5 International Standard ISO/IEC10646-1 Second Edition，Information technology- Universal Multiple Octet Coded Character Set(UCS)，2000
6 The Unicode consortium. http://www. Unicode. org
7 Hu Yuxiao，Ma Shaoping，Xia Ying. Input method Implement based on IMM-IME. Computer Engineering and application，2002(1)：117～124
8 Microsoft Corp. TrueType Font Files. Microsoft Corporation Press，1993
9 MSDN Library Visual Studio 6. 0，Microsoft Corporation

基于数据模型驱动的业务组件自动生成

作者： 尹彦均， 李亚芬， 王普
作者单位： 北京工业大学电子信息与控制工程学院 北京 100022