J2EE 平台上的所见即所得的 HTML 电子表格工具的设计*)

肖 舒 赵建华 李宣东 郑国梁
(南京大学计算机科学与技术系 南京 210093)

摘 要 本文介绍了一个 MDA 软件开发工具的设计思想和实现工具。该工具可以根据 MDA 中的平台无关模型的信息,由用户通过所见即所得的方式来编辑 HTML 表格。用户编辑了 HTML 表格框架后,可以基于 PIM 中的 ER 模型来定义表格中各个单元格的值。用户在编辑的过程中就可以直接看到表格的最终显示。这个工具和我们研究的其它两个工具协同使用,可以高效地把 PIM 转换成为 J2EE 平台上的应用程序。

关键词 MDA,J2EE,模型转换,表示层

A Design Tool for Dynamic WYSWYG Form on J2EE Platform

XIAO Shu ZHAO Jian-Hua LI Xuan-Dong ZHENG Guo-Liang
(Department of Computer Science and Technology, Nanjing University, Nanjing 210093)

Abstract This paper introduces the design idea and implements tool of a MDA software development tool. The usage of the tool is that users can define and edit HTML table in a "what you see is what you get" manner according to information specified in platform independent models. When the edit work of the framework of the HTML table has finished, users can define the value of every cell according to the ER models of the PIM. Users could see the final display of the HTML table during their edit process. This tool cooperates with the other two tools developed by my research group, leading to automatic and efficient transformation from platform independent model to application on J2EE platform.

Keywords MDA,J2EE,Model transformation, Presentation layer

1 引言

MDA[1]是 OMG 提出的新型软件开发方法。MDA[2]软件开发活动的重心在于建立用于描述业务逻辑的平台无关模型。而具体平台上的实现可以通过模型转换工具由 PIM 自动生成。我们研究的总体目标是把 PIM 转换成为 J2EE 平台上的应用系统。J2EE 平台上的应用具有三层结构:数据库层、业务逻辑层、表示层。我们研究的工具组合的功能是完成 PIM 到数据层和业务逻辑层的转换,并支持表示层的开发。PIM 中并没有规定数据如何向用户展示。因此,软件设计人员需要另行输入表示层信息。而为了保证 MDA 开发的效率,我们需要把这些表示层信息和原有的 PIM 信息有机结合起来。

我们研究的工具组合的主要功能是实现从用 EDOC[3]描述的 PIM 模型到被选作 PSM 的 J2EE 平台应用之间的自动化模型转换。我们的工具组合分为三个模块:第一个模块是实体定义模块,实现从 EDOC ER 模型到关系数据库和 EJBEntityBean 的模型转换。第二个模块是业务逻辑定义,实现 EDOC 业务过程模型(Business Process Model, BPM)和 UML 状态图到 EJBSessionBean 和 Web 模型的转换。我们的工具作为第三个模块,实现了 Web 模块中 HTML 表格的自动生成。

工业界有不少支持自动生成 HTML 表格的工具。有些工具侧重于简化表格样式的定义工作,可以方便地改变行距、列距、表格样式、单元格样式和数据样式。另一些则侧重于与数据库的连接,如 ODBiC[4]。ODBiC 通过写 SQL 语句和简单的"脚本"语言可以动态地把数据插入到网页中去,通过对 Web 浏览器上 Web 窗体元素的操作可以实现对数据库的更新和插入操作,但是它并不是一个 MDA 工具。

2 相关技术介绍

2.1 MDA

MDA 的主要特点是将系统的结构/功能与具体的技术实现相分离,以提升软件开发的抽象层次。MDA 中最重要的概念是模型。模型是对系统的功能、结构和行为的表示。根据抽象层次的不同,MDA 将模型区分为平台无关模型(Platform Independent Model, PIM)和平台相关模型(Platform Dependent Model, PSM)。MDA 模型转换的中心任务是实现 PIM 到 PSM 的自动转换。实现自动化的模型转换需要工具的支持,目前工业界比较成熟的工具的代表是 AndroMDA。

EDOC 规约是 ISO RM-ODP(Reference Model of Open Distributed Processing)[5]运用于企业规模的系统后得到的,目标是指导企业分布式计算系统的建模。作为 OMG 组织发布的第一个领域建模标准,EDOC 规约是 MDA 框架的重要组成部分。OMG 于 2004 年正式发布了 EDOC 规约的 1.0 版本,包括 Enterprise Collaboration Architecture (ECA) Specification, Metamodel and UML Profile for Java and EJB Specification, Flow Composition Model (FCM) Specification, UML Profile for Patterns Specification, UML Profile for En-

*)本文的研究工作受到国家自然科学基金(批准号 60203009,60233020)、江苏省自然科学基金(批准号 BK2003408)和国家 973 项目(批准号 2002CB312001)的资助。肖 舒 研究生,主要研究领域为软件工程;赵建华 教授,研究领域为形式化方法、软件工程、程序设计语言;李宣东 教授,博士生导师,研究领域为形式化方法、模型检验;郑国梁 教授,博士生导师,研究领域为软件工程、软件开发环境。

terprise Collaboration Architecture Specification, UML Profile for Metaobject Facility (MOF) Specification 和 UML Profile for Relationships Specification 共 7 套规范。以上规范为分布式计算系统的建模提供了元模型基础、组件体系结构、中间件技术规范、工作流规范等全方位的支持。

2.2 J2EE

J2EE 是用于建立服务器方应用程序的一种系统平台。它的整体架构是一个多层结构，包括用户层、Web 层、业务层和 EIS 层。

用户层用来与用户交互，并把来自系统的信息显示给用户。Web 层产生表示逻辑，并接受客户端的用户反馈，这些用户通常是 HTML 客户端。该层由 JSP 或 Servlet 来实现。业务层处理系统的业务规则，一般由 EJB(Enterprise JavaBeans)来实现。EIS 层主要包括后台数据库系统、遗产系统、企业资源计划系统等。

3 工具总体概述

3.1 与其他工具的调用关系

该工具是我们的 MDA 研究小组研发的 MDA 系列工具中的一个。另外两个工具是 PIM 的实体模型定义以及转换工具和 PIM 的业务逻辑模型定义与转换工具。本文论述的这个工具和 PIM 的实体模型定义以及转换工具之间的关系如图 1 所示，其中 EJB 和实体定义文件都是由实体定义工具自动生成的，数据查询管理器是由本工具根据实体定义文件自动生成的。首先，工具使用者通过数据实体定义工具定义 Web 应用中使用的实体，并把信息导出到文件中，生成相应的 EJB 代码和数据库描述文件。然后本工具从实体文件中读入实体定义信息，进行相应的界面设计和数据与界面元素的关联。切换到 Debug 状态时，通过数据查询管理器调用 EJB 代码动态获得数据。

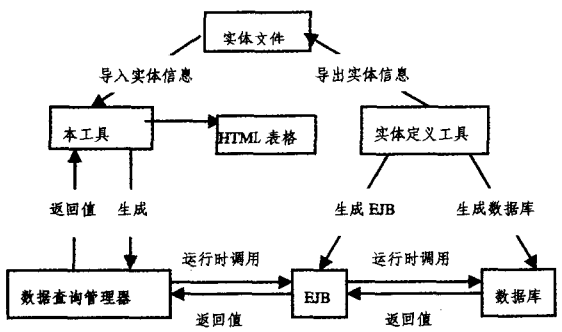

图 1

3.2 工具概述

本工具是 J2EE 平台上的所见即所得的 HTML 表格编辑和生成工具的设计。用户可以根据 PIM 中定义的实体-关系模型来定义表格中各个单元中将显示的数据。为了实现所见即所得的目标，该工具本身就是一个 Web 应用。该工具定义了两种状态：Edit 状态和 Debug 状态。处于 Edit 状态时，用户可以进行表格结构的编辑以及表格数据的定义。处于 Debug 状态时，工具调用 EJB 代码动态地获得数据。此状态下工具的使用者可以看到最终得到的 HTML 表格。这两种状态之间可以进行切换，当用户出于 Edit 状态时，如果想预览一下用户最终得到的 HTML 表格，就可以切换到 Debug 状态下；在 Debug 状态下，用户如果对当前的效果不满意，可以切换到 Edit 状态下进行修改工作。

本工具具有如下优点：首先，本工具在定义表格结构时是没有任何限制的，可以定义出任何结构的表格。其次，本工具实现了所见即所得的功能。再次，本工具是完全自动的，不需要工具的使用者对得到的 HTML 表格进行任何的后续修改。

4 工具的数据模型和总体框架

4.1 HTML 表格模型

4.1.1 结构模型

HTML 表格由若干单元格嵌套构成。我们使用树型的数据结构来表示用户编辑的 HTML 表格。在这个树形结构中，叶子节点代表了原子单元格，而非叶子节点代表了一个由它的子单元格组合而成的复合单元格。每个原子单元格可以设置宽度值和高度值。设置完原子单元格的高度和宽度后，要对该原子单元格的兄弟结点和父结点进行相应的调整，这种调整一直向树形结构的根部扩散，直到根节点。树形结构是通过对单元格的拆分来实现的(5.3)，通过拆分单元格和单元格大小的调整可以得到任何结构的表格。

4.1.2 数据模型

表格中显示的数据分为静态的和动态的。静态的数据即字符串常量，在 Edit 和 Debug 状态下显示相同的值。动态的数据是通过调用本工具自动生成的查询管理器(5.1)获得的，首先需要定义实体实例 EntityName，其次要定义相应实体实例的 Find 函数，最后为 Find 函数中的参数设置参数值。

4.2 总体框架

本工具主要由四个模块构成：实体导入模块、表格结构编辑模块、表格数据定义模块、表格保存模块。

实体导入模块：它主要用来获得实体信息和自动生成查询管理器的代码。

表格结构编辑模块：它提供两种功能：拆分和合并单元格、单元格大小的设置。由这两个功能的复合可以得到任意结构的表格。

表格数据定义模块：它由两个子模块组成：实体实例的定义和两种单元格值的定义(常量以及实体实例的属性)。

表格保存模块：提供两种表格保存方式：仅保存 HTML 表格以及保存所有的定义信息。

5 设计和实现技术

5.1 查询管理器

查询管理器是本工具根据实体定义工具提供的实体定义信息自动生成的，其中封装了 EJB 方法的调用框架，可以直接调用远程服务器的 EJB，并得到返回结果。

如果一个表格使用了多个不同类型的实体，那么生成的代码运行时就必须访问多个不同的 EJB 实体 Bean。由于本工具适用于所有的表格定义，因此不能直接访问这些 Bean 的 Find 函数或者 FindByPrimaryKey 函数。为了解决这个问题，本工具为每一个表格生成一个数据查询管理器。在查询管理器中枚举了所有的实体类型以及这种实体类型在每个查询函数下访问 EJB 时的不同情况。当根据实体定义去获得实体实例的时候，只要向数据查询管理器提供定义中说明的实体类型和相应的查找函数的名称，以及相关参数，查询管理器就会根据这些信息完成对相关 EJB 的函数调用，并返回相应的值。

5.2 URL Button

为了实现所见即所得的目标，该工具本身就是一个 Web 应用。而 Web 页面中定位单元格比应用程序中定位单元格要困难得多。在应用程序中，定位一个单元格是非常方便的，只要得到鼠标所处的位置的横坐标和纵坐标，判断它处于哪个单元格就可以了。但是在 Web 应用中，定位一个单元格的工作变得比较困难，无法根据鼠标所处位置的坐标来进行定位。因此本工具中引入了 URL Button。所谓 URL Button 是指在每个单元格中加入一个 Button，并为这个 Button 赋一个唯一的 id。当要对某个单元格进行操作时，通过在页面之间传递相应 URL Button 的 id，使得工具可以判断是对哪一个单元格进行的操作。值得注意的是，原子单元格的定位和复合单元格的定位略有不同。由于原子单元格的 URL Button 在 Edit 状态下是显示出来的，所以只需直接点击 URL Button 即可。复合单元格的定位稍复杂一点，需要首先选中它的子孙节点中的一个原子单元格的 URL Button，然后通过若干次定位到父单元格的操作来实现。本文论述的工具中，还提供了定位到左兄弟单元格、右兄弟单元格和第一个子单元格的功能，使得用户可以更加方便地定位到任意一个原子单元格或复合单元格。

5.3 表格结构的定义

本工具采用拆分单元格的方法来定义表格的结构。在进行结构编辑之前，是一个单一的单元格。用户可以根据三种方式拆分单元格：横向不变拆分，纵向不变拆分和纵向可变拆分。通过多次拆分并设定每个单元格的大小，用户可以定义任何形式的表格。横向不变拆分和纵向不变拆分是在水平和垂直方向上将一个单元格拆分成一组固定个数的多个子单元格，子单元格的数量由工具的使用者输入。其表格变化和数据结构的改变如图 2 所示。纵向可变拆分是在垂直方向上将一个单元格拆分成一组不固定的多个子单元格，子单元格的数量根据切换到 Debug 状态时的实际情况来动态确定。例如一个单元格是纵向可变拆分的，如果在运行时数据查询管理器返回的的实际数据包含 n 个实体，那么该单元格就会包含 n 个子单元格。其表格变化和数据结构的改变如图 2 所示。我们使用树形结构来记录表格的结构。在这个树形结构中，叶子节点代表了原子单元格；而非叶子节点代表了复合单元格。这种表格结构定义方式配合单元格大小的调整可以得到任何结构的表格。

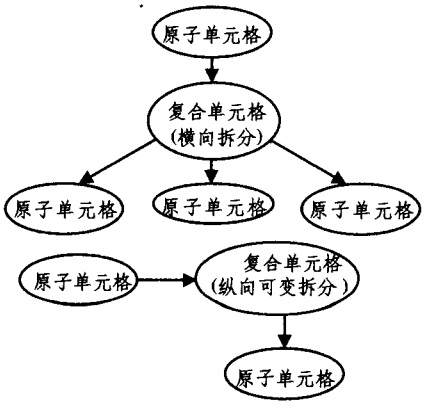

图 2

5.4 表格数据的定义

实体实例的定义。定义了表格的基本结构后，工具的使用者还需要指定表格的各个单元格中所显示的值。本工具需要使用实体定义文件中定义的实体信息。每个实体至少包含一个 FindByPrimaryKey 函数以及若干个 Find 函数。FindByPrimaryKey 函数的所有参数刚好组成实体的关键字属性，该函数的返回值是一个 EJB Remote 接口类对象。而 Find 函数的返回值是一个由 EJB Remote 接口类对象组成的集合。当查找函数为 FindByPrimary 时，相应的操作为单值实体定义；而查找函数为 Find 类型时，相应的操作是一个多值实体定义。实体实例的定义总是和某个单元格相联系。其中单值实体可以和任意类型的单元格关联，多值实体只能和纵向可变单元格关联。处于 Debug 状态时，纵向可变拆分的子单元格的个数就等于该多值实体定义中实体实例的个数。

单元格值的定义。在完成实体实例的定义之后，就可以进行单元格的定义了。单元格值分为两种类型：常量和实体实例的属性。常量的定义比较简单，直接输入字符串常量就可以了，Edit 状态和 Debug 状态下显示的相应的单元格值都为该字符串常量。当单元格值是一个实体实例的属性时，在 Edit 状态下，它显示的格式为 EntityName. Attribute，其中 EntityName 是实体实例的名称，而 Attribute 是属性名称。在 Debug 状态下，该单元格的值就是 EntityName 对应的实体实例的 Attribute 属性的值。

5.5 表格的保存

本工具提供了保存表格定义与编辑的中间结果的功能，用户可以导入中间结果，并在中间结果的基础上继续进行表格的定义和编辑工作。本工具提供了两种保存 HTML 表格的方式：只保存 HTML 表格本身和保存完整的表格定义信息两种。在第一种情况下，将表格保存为一个 HTML 页面，表格中的数据将不能再进行修改。在第二种情况下，保存了完整的定义信息，包括由实体定义文件得到的实体信息以及树形结构中各个节点完整的定义信息，使用这种保存方式保存的结果可以再次导入到本工具中进行再次的编辑和定义工作。

总结与进一步工作 本文介绍了一种 J2EE 平台上所见即所得的 HTML 表格编辑和生成工具的设计。首先用户定义 HTML 表格的结构形式，然后定义 HTML 表格中的各个单元格应该显示什么数据，最后就可以自动生成 HTML 表格了。我们工具的优点在于首先可以定义任意结构的 HTML 表格，其次可以通过对 Web 浏览器上 Web 窗体元素的操作实现对 EJB 调用，从而实现对数据库的查询，再次实现了所见即所得的功能，使得用户随时可以看到 HTML 表格的最终显示。本文论述的这个工具作为我们 MDA 研究小组的一系列工具中的一个组成部分，可以和其他两个工具相结合，提供更强大的功能。

本工具中表格的行和列之间还是相对孤立的，在以后的工作中将进一步建立行和列之间的联系，如纵向可变单元格的显示可按某个属性值进行排序，还可以引入统计数据等。

参 考 文 献

1 Object Management Group. MDA Guide. Object Management Group, 2003. www. omg. org
2 Object Management Group. Model Driven Architecture. Object Management Group, 2001. www. omg. org
3 Enterprise Distributed Object Computing. http://www. omg. org/technology/documents/formal/edoc. htm
4 Open Database Internet Connector. www. odbic. com
5 ISO. ISO/IEC 10746-1: Information technology - Open Distributed Processing-Reference model. ISO, 1998