

特定领域建模方法和代码生成的研究

卫红春, 贾笑蕊

(西安石油大学 计算机学院, 陕西 西安 710065)

摘 要: 特定领域建模(Domain-Specific Modeling, DSM)提高了软件的抽象水平, DSM 用模型表示用户的需求, 然后通过生成器, 直接把模型中的信息提取出来并转换成代码, 实现了代码的自动生成. 研究了特定领域建模语言、生成器和领域框架, 总结了 MetaEdit+ 下实现 DSM 的一般过程及其使用的相关技术, 并介绍了代码生成的具体过程. 基于 MetaEdit+, 实现了电子万年历的元建模和领域建模, 并通过提出的深度优先遍历的代码生成算法, 实现了电子万年历代码的自动生成.

关键词: 特定领域建模; 元模型; MetaEdit+; 生成器; 深度优先遍历; 电子万年历

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2013)09-0154-04

The Research of the Approach and Code Generation Based on Domain-Specific Modeling

WEI Hong-chun, JIA Xiao-rui

(Institute of Computer, Xi'an Shiyou University, Xi'an 710065, China)

Abstract: Domain-Specific Modeling (DSM) raises the abstract level of the software, it represents the user requirements by models, and with generators, information is directly extracted from the models and transformed into code, the code can be automatically generated. Based on the research of domain-specific modeling language, generator, domain framework, summarizes the general process of implementing DSM under the tool of MetaEdit+, related technologies, and introduces the process of code generation in detail. Based on MetaEdit+, the metamodeling and domain-specific modeling of Electronic calendar are implemented, and with the algorithm of the depth-first traverse proposed, code of the electronic calendar is generated automatically.

Key words: DSM; meta model; MetaEdit+; generator; depth-first traverse; electronic calendar

1 引言

特定领域建模(DSM)的出现, 为软件开发提供了一种新的模型驱动开发方法, 它能够大幅度提高软件开发效率并简化软件开发过程. DSM 从两个方面提高了软件的开发效率. 首先, DSM 解决方案直接使用特定问题领域中的概念和规则, 提高了软件的抽象级别. 第二, 通过生成器能直接生成可运行的代码^[1].

在 DSM 中, 特定领域建模语言(Domain-Specific Modeling Language, DSML)的定义, 包括领域

概念、规则和到代码的映射都是由领域专家完成的. 然后软件开发人员根据规则引导下的概念创建模型及代码的自动生成. 这样, 开发人员可以更高效地开发应用程序. 使用 DSM, 人们可以把关注点放到寻找用领域概念表示的解决方案, 而不是代码. 在过去十年中, 采用 DSM 能够提高五到十倍的软件开发效率^[2].

MetaEdit+^[3]是 MetaCase 公司开发的一款特定领域建模工具. 它提供了良好的可视化图形界面, 集成了元模型和模型, 并提供和其它开发环境(如 Eclipse, Visual Studio)的集成. 本文研究了特定领

域建模工具 MetaEdit+ 下的特定领域建模和代码生成,并以电子万年历为例详细阐述了特定领域建模的具体设计过程,使用深度优先遍历算法实现代码的自动生成。

2 DSM 开发流程

利用 MetaEdit+ 开发 DSM 的基本流程如图 1 所示。

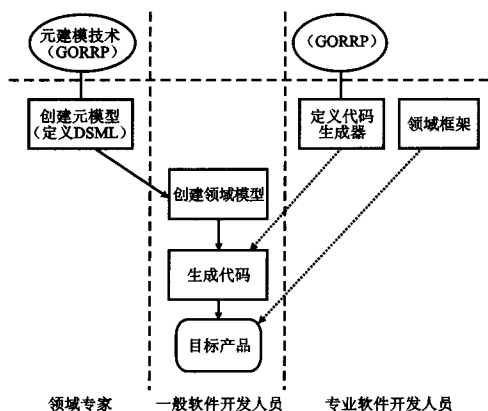


图1 特定领域建模的基本流程

(1)由领域专家创建元模型。元模型^[4]是对特定领域建模语言(DSML)的描述,主要包括定义语言概念和规则。

(2)为元模型绘制具象的符号表示。利用 MetaEdit+ 提供的符号编辑器或引入已存在的元素符号。

(3)专门的软件开发人员定义代码生成器,定义模型元素和代码之间的映射。

(4)普通软件开发人员使用领域专家定义的 DSML 创建领域模型。

(5)结合领域框架,实现目标产品。领域框架提供了生成的代码和底层平台之间的接口。生成器生成的代码一般不能单独运行,还要使用领域框架提供的额外组件才能运行。

2.1 创建元模型

元模型是对模型的抽象,用来描述模型的一组规则,它提供了比模型更高级别的抽象,更接近现实问题,用来描述元模型的语言称为元-元模型。2002 年 OMG^[1](对象管理组织)提出了 4 层建模框架,如图 2 所示。

MetaEdit+ 提供了基于 GORRP^[2](Graph、Object、Role、Relationship、Property)的元建模。由图 2 可见,DSM 是基于元模型的,属于元模型层,一般的编程语言中定义的类等处于模型层,DSM 提供了比

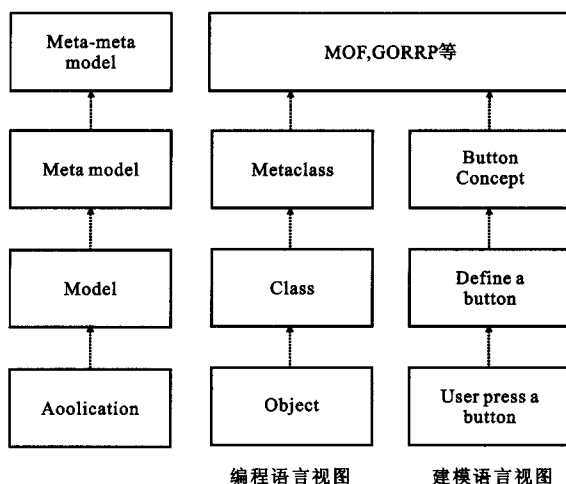


图2 实例化层次结构图

一般编程语言更高的抽象度。DSML 主要把问题域中的概念映射为建模语言中的对象、关系、属性或角色等,领域中的关键概念映射为建模语言对象,其它概念则映射为属性等。特定领域关键概念的提取是由领域专家完成的,建模概念的来源^[5]主要有:

- (1)物理产品结构;
- (2)待建系统的外观;
- (3)产品变化空间;
- (4)生成目标。

根据问题领域的具体情况选择合适的建模概念来源,提取领域关键概念,进行元模型的创建。

2.2 定义代码生成器

MetaEdit+ 提供了专门的生成器定义语言 MERL,用来为对应的 DSML 定义相应的生成器,生成器的思想是遍历模型,提取模型中的信息,并把它转换为目标平台代码这部分工作是有专门的软件开发人员实现的。

要创建代码生成器,首先清楚要生成何种代码,然后选择生成的方法,生成器的构造方法^[1]有基于模板的生成器、基于爬虫的生成器和基于访问者的生成器等,一般生成脚本文件和配置文件选择基于模板的生成器,基于爬虫的生成器适合导航模型中的元素路径并输出对应的代码。构建和测试生成器的一般步骤可归结为:

(1)首先创建一个示例模型,编写想要的输出代码,将其作为第一个生成器;

(2)修改生成器,直到可以生成和模型对应的正确的代码输出;

(3)精简生成器,即把生成器分解为多个独立的小生成器,便于重用,如果需要输出某个生成器对应的代码,直接把该生成器作为子生成器调用即可;

(4)修改模型中的元素,调试生成器,重复步骤(2)~(4),直到为示例模型生成正确的输出;

(5)结合实际例子,运行生成器,得到正确的输出,完成生成器的定义。

一般,定义生成器的规则如下:

(1)访问模型数据 生成器访问基于语言元模型的模型. 可以从一个特定的根元素开始导航,查找特定的对象类型,或根据模型中不同的关系类型和连接类型开始导航。

(2)提取模型数据 在模型中导航时,生成提取设计数据并把它和领域框架结合起来. 代码生成器从模型中只能提取元模型中提供的信息. 最简单的情况,每个建模元素产生固定的代码,包括建模者输入的值. 生成器也可以通过分析模型元素的组合提取数据. 比如,连接模型的关系,元素的子模型,或者模型间的其它联系。

(3)转换模型为输出代码 在模型中导航时,为了生成代码,把访问的数据结合起来. 在此,生成器为输出增加了其他的信息,例如,和框架代码集成或调用底层的目标环境和库。

2.3 创建领域模型

一旦定义了元模型,团队中的其他人员就可以利用元模型创建领域模型,最后再结合代码生成器和领域框架实现最终可运行的目标产品,由于DSML和生成器都是为特定领域专门定制的,所以可以实现从模型到代码的生成^[6]。

3 电子万年历的设计

3.1 定义 DSML

首先创建元模型,通过待建系统的外观来捕获电子万年历中的领域概念. 提取到的主要概念有:按钮(button),状态(state[calendar]),显示时间和日期的区域等;另外除了我们直接看到的显示部分,还有控制万年历运行的逻辑部分. 把电子万年历的实现分为显示部分和逻辑部分. 显示部分和逻辑部分组成电子万年历模型. 经分析,定义两种建模语言,CalendarModel和CalendarApplication. CalendarModel描述万年历的组成,由显示和逻辑部分组成,CalendarApplication主要描述万年历的具体逻辑实现. 分别建立领域元模型,设置绑定约束关系,并对元模型进行符号化,表1是提取电子万年历中主要对象和关系的相关说明。

3.2 创建模型

使用上述定义的领域建模语言创建对应的模

型,即电子万年历的逻辑应用程序模型 Electronic-Calendar 和软件模型 Calendar。

表 1 电子万年历中主要的对象和关系及其说明表

符号表示	名字	相关说明
	CalendarModel	电子万年历模型,由显示部分和逻辑部分组成。
	Displays	显示部分,由显示时间、日期、星期的区域和按钮组成。
	LogicalCalendar	万年历逻辑部分,指向电子万年历逻辑应用图。
	Action	动作,对相关事件的处理。
	State[Calendar]	状态,主要有编辑年、月、日、时、分、秒、等。
	Button	按钮,对时间、日期的操作,如 Set、Up、Down 等。
	Transation	关系,从一个状态到另一个状态之间的转换。

电子万年历分为两部分实现:显示部分和逻辑部分. 图3的左下方是电子万年历的显示部分. 其中包括三个显示时间的区域,三个显示阳历日期的区域,一个显示农历日期的区域和一个显示周几的区域. 右下方是电子万年历的逻辑部分,它由一个指向逻辑应用程序的引用组成。

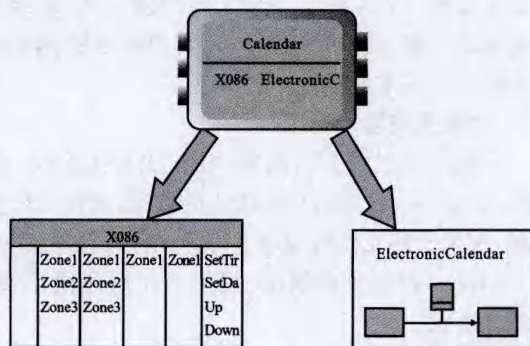


图 3 电子万年历模型图

电子万年历的逻辑实现模型主要由两部分组成:上部分是对时间的显示及处理,下部分是对阳历

日期的显示和处理过程。

3.3 定义生成器

有了模型捕获所有的静态和动态逻辑方面以及框架提供所需的底层支持,代码生成器可以产生功能完全且可执行的输出代码^[5]。在电子万年历例子中,生成器不仅仅能生成和模型对应的C#代码,它还负责创建和运行构建脚本,并确保领域框架可用。生成按以下步骤运行:

- (1)生成用于编译和执行代码的批处理脚本。
- (2)生成用于领域框架组件的代码。
- (3)生成的代码作为目标平台的测试环境被编译和执行。

电子万年历DSM解决方案的顶层是Calendar-Model建模语言,同时它也构成了顶层的生成器。顶层的生成器如图4所示。箭头表示对子生成器的调用,执行顺序是深度优先,从左到右。

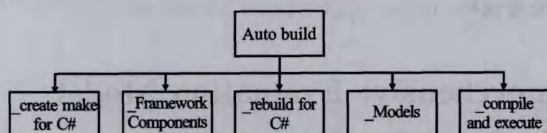


图4 顶层生成器框架

顶部是主生成器“Auto build”。“Auto build”的角色和许多编程语言中的“main”函数相似:初始化整个生成过程。“_Framework Components”为领域框架组件输出预定义的C#代码,“_create make for C#”子生成器的基本任务是创建可执行的脚本,这些脚本负责生成代码的编译和执行。“_Models”子生成器的职责是为电子万年历模型、和万年历逻辑应用程序处理代码生成。“_compile and execute”执行以上生成过程步骤中产生的脚本。

低层生成器架构(为CalendarApplication建模语言设计的生成器)如图5所示。

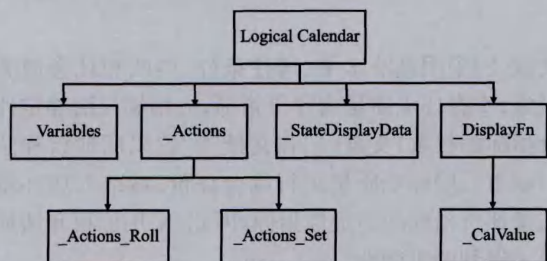


图5 底层生成器框架图

通过底层生成器产生电子万年历的逻辑应用程序代码,结合领域框架和顶层生成器中产生的批处理脚本和对代码的编译执行文件,最终实现了电子

万年历的设计,运行界面如图6所示。

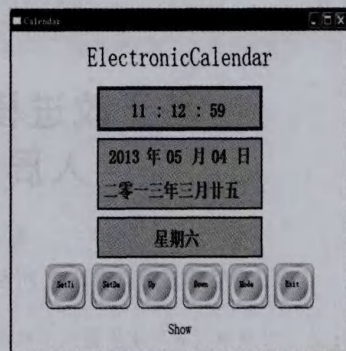


图6 运行界面

4 结束语

本文介绍了开发DSM的基本过程以及代码的自动生成的一般步骤,在代码生成器中运用了深度优先算法,实现对模型元素的有序遍历,并以电子万年历为例,经过测试,无需人工编写代码,利用生成器,可以实现从模型到目标产品的实现。DSM提供了比一般编程语言更高的抽象水平,从而减少了开发软件的时间,为软件开发提供了一种新思路。

参考文献:

- [1] Kelly S, Tolvanen J-P. Domain-Specific Modeling: Enabling Full Code Generation[M]. Wiley: IEEE Computer Society Press, 2008.
- [2] 刘辉,麻志毅,邵维忠. 元建模技术研究进展[J]. 软件学报, 2008, 19(6): 1317-1327.
- [3] MetaCase, MetaEdit+ Workbench 4.5 SR1 User's[EB/OL]. [2012-11-25] <http://www.metacase.com/Support/45/manuals/2009>.
- [4] 杨兴平,苏桂平,王瑞芳,等. 特定领域建模与代码生成的研究与实现[J]. 计算机系统应用, 2009(4): 100-103.
- [5] 周金根. MetaModelEngine: 元模型引擎开发思路[EB/OL]. (2010-07-28) [2013-05-19] <http://www.cnblogs.com/zhoujg/archive/2010/07/28/1786155.html>.
- [6] Tolvanen JP, Kelly S. Defining domain-specific modeling language to automate product derivation: Collected experiences[C]//Proc Of the Software Product Lines: The 9th Int'l Conf(GPCE 2005). 2005:198-209.

作者简介:

卫红春 男,(1956-),教授。研究方向为软件工程、软件建模。

贾笑蕊 女,(1987-),硕士研究生。研究方向为特定领域建模。