第35卷 第18期 计 算 机 工 程 2009年9月
Vol.35 No.18 Computer Engineering September 2009
·软件技术与数据库· 文章编号：1000—3428(2009)18—0048—03 文献标识码：A 中图分类号：TP311

基于模型驱动架构的决策支持系统开发方法

宋旭东[1]，徐连鹏[1]，刘晓冰[2]
(1. 大连交通大学软件学院，大连 116028；2. 大连理工大学管理学院，大连 116024)

摘 要：针对当前决策支持系统开发所面临的系统复杂度高、扩展性差等问题，提出一个基于模型驱动架构的决策支持系统的开发方法，给出一个基于模型驱动框架的决策支持系统开发框架，并通过一个具体的开发实例说明如何应用该框架进行决策支持系统开发。应用表明，该方法可有效地缩短开发周期，降低系统复杂度，提高开发效率，同时保证实施的可行性和扩展性。

关键词：模型驱动架构；决策支持系统；模型驱动；开发方法

Development Approach of Decision Support System Based on Model Driven Architecture

SONG Xu-dong[1], XU Lian-peng[1], LIU Xiao-bing[2]
(1. Software Institute, Dalian Jiaotong University, Dalian 116028; 2. School of Management, Dalian University of Technology, Dalian 116024)

【Abstract】 In order to solve the existing problems of the complexity of the system, expansion of poor issues and so on, this paper presents an approach based on Model Driven Architecture(MDA) for the development of decision support system, and proposes a framework based on MDA. A case study is provided to exemplify the benefits of the approach. Practice proves that the method effectively shortens the development cycle and improves the efficiency of development.

【Key words】 Model Driven Architecture(MDA); decision support system; model driven; development approach

1 概述

决策支持系统(Decision Support System, DSS)是指具有辅助决策功能的高级计算机信息系统[1]。它可以及时、有效地为决策者分析、解决复杂的决策问题，从而提高管理决策水平，促进管理决策的科学化。因此，DSS 的研究与应用有着重要的理论意义和重大的实用价值。由于决策支持系统包含了众多的关键技术，因此在开发过程中存在很多困难，比较显著的问题有：系统复杂度高，复用率低，结构灵活性差，可扩展性差，不能快速适应变化，系统之间的有机集成难以实现等难题。

模型驱动体系架构(Model Driven Architecture, MDA)是对象管理组织(Object Management Group, OMG)发布的一个软件开发框架[2]，目的是将业务和应用逻辑与底层平台技术分离开来。当前，已有一些将 MDA 应用到各个领域的开发应用研究，文献[3]提出了一个基于 MDA 开发数据仓库的框架，解决了传统的数据仓库开发方法的异构性问题；文献[4]提出了一种基于 QVT 的模型转换框架，在一定程度上消除了模型转换技术的异构性针对决策支持系统开发过程中存在的难题。通过借鉴上述方法和技术，本文将 MDA 的思想与决策支持系统的特点相结合，提出一种基于 MDA 的决策支持系统开发方法。

2 基于 MDA 的决策支持系统的开发框架

2.1 决策支持系统体系结构

笔者设计了一个含有 4 层结构的决策支持系统(图 1)：展现层，逻辑层，控制层，数据层。分层的体系结构可以避免系统部件的耦合，降低系统的复杂性，增强系统的可扩展性、可复用性和可维护性。

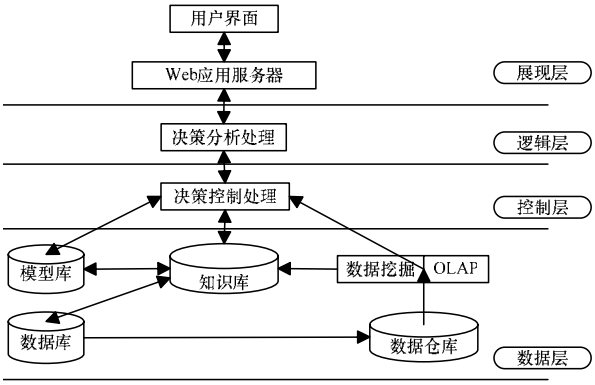

图 1 DSS 体系结构

下面是各层结构的详细描述：

(1)展现层：负责系统的界面表现，负责控制页面的流转和对业务服务的调用，并控制页面和业务服务之间的数据传递。

(2)逻辑层：对决策的属性和功能进行逻辑控制，体现决策规则；提供对决策数据、决策知识和决策信息的各项处理，满足决策的功能需求；确保对决策数据和决策信息展现的有效性和合理性。

(3)控制层：负责对决策的处理过程进行分解、调度和控

基金项目：国家自然科学基金资助项目(70572098, 70471056)；大连IT教师基金资助项目(大财企[2008]281号)
作者简介：宋旭东(1969－)，男，副教授，主研方向：数据仓库，决策支持系统；徐连鹏，硕士；刘晓冰，教授
收稿日期：2009-03-14 **E-mail：**xudongsong@126.com

制；负责对数据、模型和知识 3 个部件进行有机集成，根据实际决策问题的规模和复杂程度决定是采用单个主体辅助决策，还是采用 2 个或是 3 个主体相互结合的辅助决策。

(4)数据层：为各项决策应用提供数据服务，它包括数据库、模型库、知识库、数据仓库、OLAP 以及数据挖掘。通过数据服务层，能够成功地收集、分析、理解决策知识和决策信息，并以此做出相应决策。

2.2 基于 MDA 的决策支持系统框架

MDA 的体系中包含 4 类模型，分别对应软件生命周期中的 4 类模型：计算独立模型(Computer Independent Model, CIM)对应需求模型，平台无关模型(Platform Independent Model, PIM)对应分析模型，平台相关模型(Platform Specific Model, PSM)对应设计模型，以及 CODE 对应实现模型。MDA 的核心思想是抽象出与实现技术无关、完整描述系统的平台无关模型(Platform Independent Model, PIM)，针对不同实现技术制定变换定义；通过制定映射规则，利用转换工具将 PIM 转换成与具体实现技术相关的 PSM；最后，再通过转换工具将 PSM 自动转换成 CODE。

针对本文给出的 DSS 体系结构，提出了一个基于 MDA 的 DSS 开发框架(图 2)，应用这个框架进行 DSS 开发，能够有效解决目前应用传统软件开发方法带来的系统复用率低、不能快速适应变化等问题。

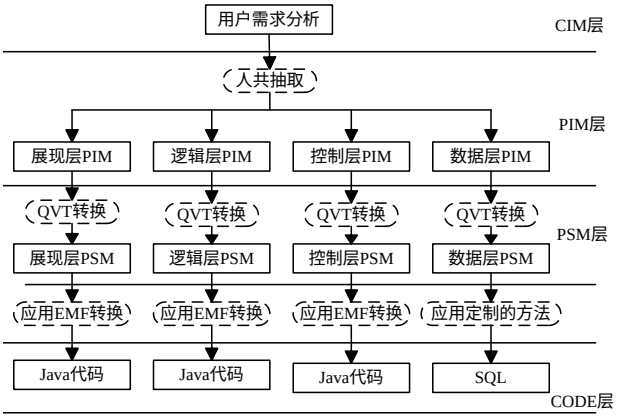

图 2 基于 MDA 的 DSS 开发框架

这个框架自上而下分为 4 层，分别对应 MDA 中的 4 种模型：

(1)CIM 层：当前应用 MDA 进行的系统开发，CIM 模型由于需要领域专家、客户专家等人员的参与，因此这层模型并没有一个通用的标准，往往以实际的用户需求分析文档作为 CIM。

(2)PIM 层：本层所建立的模型力图体现 DSS 的功能特性，而不考虑使用的技术的细节。本文使用了 UML(Unified Modeling Language)profile 来描述 PIM。UML 是由 OMG 推荐使用的 PIM 建模语言。UML 中语义归约和实现归约的分离，以及良好的可扩展性，都是其作为 PIM 建模语言的优点。而且 UML 在目前的系统开发中使用也较为普遍，使用其定义 PIM 具有良好的通用性。

(3)PSM 层：本层模型是与具体使用的平台相关的。在本文给出的 DSS 体系结构中，展现层、控制层和逻辑层的 PSM 是基于 EMF(Eclipse Modeling Framework)的 ecore 模型，在下文给出的开发实例中，PSM 是基于 EMF 的 ecore 模型，数据层 PSM 使用 XML(eXtensible Markup Language)描述。

(4)CODE 层：这一层模型实际上就是可执行的代码。同样的，这一层也是与具体使用的平台相关的。

这 4 层模型已经确立，接下来需要定义转换规则来使其实现模型的自动转换：

(1)CIM 到 PIM：由于实际的某些用户需求只能通过文字来描述清楚，因此从 CIM 到 PIM 的转换主要以人工抽取的方式来实现，这样才能保证 PIM 对客户需求的完整性和准确性。

(2)PIM 到 PSM：这一步转换是较为关键的一步，这一步转换必须确保 PIM 中所有信息都能够在生成的 PSM 中得以体现。OMG 的 QVT(Query View Transformation)规范可以保证这一步转换的完整性[5]。同时，使用 QVT 规范定义的转换，具有良好的可读性和可维护性。本文应用了 IBM 的支持 QVT 规范的模型转换框架(Modeling Transformation Framework, MTF)，定义基于 QVT 的映射规则，并实现 PIM 到 PSM 的转换。在本文给出的 DSS 体系结构中，数据层 PIM 到 PSM 的转换借鉴了文献[6]中的方法。

(3)PSM 到 CODE：在本文给出的框架中，展现层、控制层和逻辑层的 PSM 是基于 EMF 的 ecore 模型，他们的转换直接使用 EMF 即可实现。对于数据层的 PSM 到 CODE 的转换，使用自行开发的方法将 XML 转换为 SQL(Structured Query Language)。

使用该框架进行 DSS 开发具有很多优势：

(1)高效性：整个决策支持系统可以由定义好的 PIM 通过定制的转换规则，自动生成可执行代码，提高了开发效率，降低了开发成本。

(2)复用性：PIM 使用 UML profile 定义，是与平台无关的模型，在遇到类似的系统开发时，可以直接使用这些模型。

(3)可移植性：如果需要对使用的 DSS 进行改动，例如改变其使用的应用服务器或数据库系统环境，所建立的 PIM 不需要改动，因为这些模型是与平台无关的。开发人员只需针对改动的部分，重新定义转换规则，生成新的 PSM 及 CODE，而不用对整个 DSS 进行修改，从而具有很好的可移植性。

(4)可扩展性：随着系统的使用，用户往往会提出新的需求或对原有需求的变更。开发人员针对这些要求，定义新的 PIM 或修改相关的 PIM，并定义或更改转换规则，生成相应的 PSM 及 CODE，即可满足用户的需求。

(5)业务性：使用该框架进行开发的开发人员可以将主要的精力放在概念模型，即在 PIM 的建立上，而无需过多地关注技术细节，从而可以使设计的系统更好地满足用户的需求。

3 基于 MDA 的决策支持系统的开发实例

以某钢铁企业的决策支持系统中的数据挖掘算法管理模块为例，给出具体的开发方法和步骤。

步骤 1 建立 CIM

系统的开发从需求分析开始，对此模块(CIM)需求描述如下：

(1)可以向系统注册新的挖掘算法。

(2)算法需要分类，以便于管理。

(3)可以对算法进行查询、修改、删除等维护操作。

这部分模型由需求分析文档来体现。

步骤 2 抽取 PIM

根据上述的需求，采用人工的方式，抽取出 PIM。利用 UML profile 来描述，如图 3 所示，其中一些无关的模型细节

被省略。

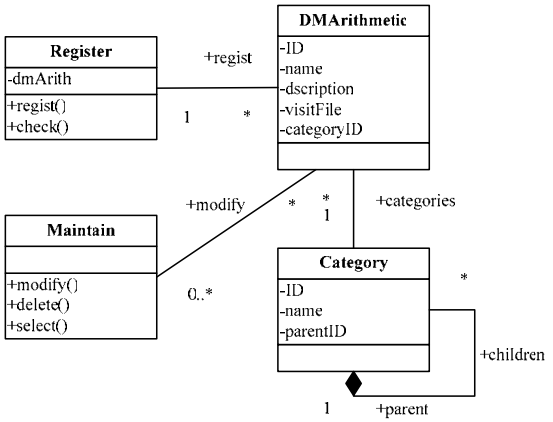

图 3　平台无关模型(PIM)

这个模型中含有 4 个类：

(1)DMArithmetic(挖掘算法类)：描述系统中的挖掘算法，每个算法通过唯一 ID 进行标识，并且每个挖掘算法必须属于某个分类。

(2)Register(注册类)：负责向系统注册新的挖掘算法，检测要注册的算法是否符合系统要求。

(3)Maintain(维护类)：负责执行对挖掘算法的查询、修改、删除等维护操作。

(4)Category(类别类)：是挖掘算法的分类，每个分类可能含有父级分类(通过 parentID 标识)。

步骤 3　生成 PSM

应用 QVT 规范来定义映射规则，实现 PIM 到 PSM 的转换。如下就是从 PIM 到 PSM 的映射规则片断，即实现 UML 模型到 ecore 模型的映射规则：

```
relate uml2ecore( emf:ResourceSet uml, emf:ResourceSet ecore){
    umlfile2ecore( over uml.resources, ecore )
}
relate umlfile2ecore( ecore:EResource uml, emf:ResourceSet ecore){
    model2ecore( over uml.contents, ecore )
}
relate model2ecore( uml:Model model, emf:ResourceSet ecore){
    umlpkg2resource( over model.packagedElement, over ecore.resources )
}
relate umlpkg2resource ( uml:Package pkg, emf:Resource Extension ecoreResource when equals(ecoreResource.fileExtension, "ecore"))
        when equals(pkg.name, ecoreResource.name){
    umlpkg2ecore( pkg, over ecoreResource.contents )
}
```

本文使用 IBM 的 MTF 来定义并执行此映射规则。执行上述规则，首先会调用其入口方法 uml2ecore，然后执行这个方法中的 umlfile2ecore 方法，再依次执行各个方法体的内容，直到结束。

应用这个映射规则，即可将 UML 模型转化为 ecore 模型，如图 4 所示。

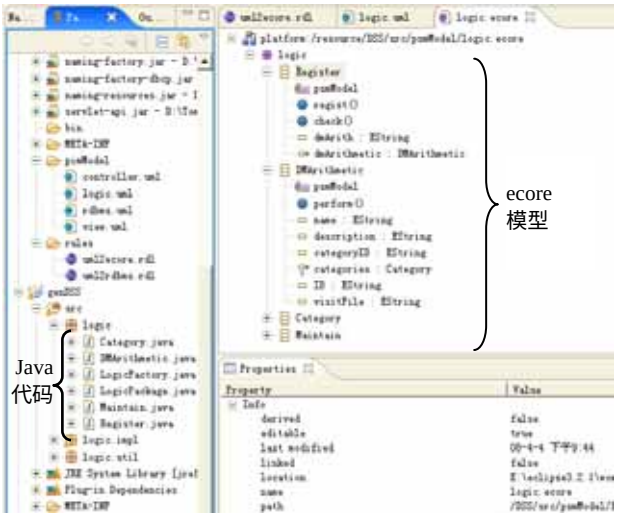

图 4　由 ecore 模型生成的代码

步骤 4　生成 CODE

步骤 3 生成的 PSM 是基于 EMF 的 ecore 模型，再使用 EMF 代码生成的工具 Java 发射器模板 JET(Java Emitter Templates)，可以直接将 ecore 模型转化为可执行的 Java 代码(图 4)。

以上 4 个步骤详细描述了如何利用本文给出的基于 MDA 的 DSS 开发方法进行开发。使用该方法进行开发，侧重点是模型的建立，制定转换规则，这些模型使用 UML profile 来描述，与平台无关，具有很好的通用性。在开发类似的系统时，可以直接使用这些模型，从而在降低系统开发成本的同时，极大地提高了开发的效率。

4　结束语

目前，MDA 已经在很多领域得到应用，并取得了很好的效果。应用 MDA 进行系统开发，软件开发人员就可以集中精力在系统的设计上，而无需关注实现细节，从而显著地提高软件开发的效率。本文给出了一个应用 MDA 进行决策支持系统开发的方法，并成功应用于某钢铁集团的决策支持系统开发中。对如何提高这种方法的通用性，以适应其他领域的开发，是下一步的研究工作。

参考文献

[1] 李志刚. 决策支持系统原理与应用[M]. 北京: 高等教育出版社, 2005: 46-56.

[2] OMG. MDA Guide Version 1.0.1[EB/OL]. (2003-06-01). http://www.omg.org/cgi-bin/apps/doc?formal/03-06-01.pdf.

[3] Mazon J N, Trujillo J. An MDA Approach for the Development of Data Warehouses[J]. Decision Support Systems, 2008, 45(4): 41-58.

[4] 王学斌, 王怀民, 吴泉源, 等. 一种模型转换的编织框架[J]. 软件学报, 2006, 17(6): 1423-1435.

[5] OMG. QVT Specification[EB/OL]. (2005-11-01). http://wwsw.omg.org/docs/ptc/05-11-01.pdf.

[6] Mazon J N, Trujillo J. An MDA Approach for the Development of Data Warehouses[J]. Decision Support Systems, 2008, 45(4): 41-58.

编辑　任吉慧