基于 SysML 的平台无关模型转换研究

褚长勇 任栩生 孙安程
(杭州电子科技大学机械工程学院 浙江 杭州 310018)

摘 要 图形化系统建模语言 SysML 可以通过建立不同的模型,从多个角度反映整个系统的结构特征。但是从系统的需求出发,使用 SysML 建立系统的多种模型图比较繁琐,效率低下;模型图之间相互关联,存在信息重叠。提出基于 UML Profile 通用扩展机制下平台无关模型间的转换方法:通过在扩展文件中建立模板,导入到模型图中实现对模型元素额外的扩充和限制。以此为基础,分别设计需求图及用例图的模板,采用 ATL 模型转换语言实现 SysML 需求图到用例图、用例图到序列图的转换。通过一个实例验证该方法的可行性。该方法不仅可以明确模型元素之间的关系,而且有效提高建模效率。

关键词 系统建模语言 信息重叠 平台无关模型 模型转换

中图分类号 TP3 - 0 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2018.12.002

PLATFORM INDEPENDENT MODEL TRANSFORMATION BASED ON SYSML

Chu Changyong Ren Xusheng Sun Ancheng
(*School of Mechanical Engineering*, *Hangzhou Dianzi University*, *Hangzhou 310018*, *Zhejiang*, *China*)

Abstract The graphical system modeling language (SysML) can reflect the structural characteristics of the whole system from multiple perspectives by establishing different models. However, from the requirements of the system, it is tedious and inefficient to use SysML to build multiple model diagrams of the system, and the model diagrams are interrelated and there is information overlap. This paper proposed a transformation method of platform independent models based on UML Profile universal extension mechanism. Additional extensions and restrictions of model elements were achieved by creating templates and importing them into the model diagram. On this basis, we designed the template of requirement diagram and used case diagram respectively, and realized the transformation from SysML requirement diagram to use case diagram and use case diagram to sequence diagram by ATL. The feasibility of the method was verified by an example. This method can not only clarify the relationship among the model elements, but also improve the modeling efficiency.

Keywords System modeling language Information overlap Platform independent model Model transformation

0 引 言

在信息技术高速发展的今天,人们需要开发的系统日益复杂,而且牵涉到的领域也越来越广。系统建模语言 SysML 针对系统工程领域中设计与建模的特点,提供了可视化、图形化的系统建模支持,广泛应用于复杂系统建模[1]。SysML 包含九种模型视图,侧重于不同角度反映整个系统的结构特征。模型图之间相互关联,互为补充。但是现阶段系统建模主要依赖于工程师的个人能力水平,不仅建模效率较低,而且也无法避免人为错误导致的模型错误、不一致等问题。

由于不同模型图反映的模型信息存在重叠、冗余的情况,因此可以对模型信息进行扩展,加强模型之间的联系,通过建立模型之间的映射关系实现模型转换达到快速开发建模的目的。模型驱动架构 MDA(Mod-

收稿日期:2018 - 05 - 15。国家自然科学基金项目(51005065)。**褚长勇**,副教授,主研领域:基于模型的系统工程。**任栩生**,硕士生。**孙安程**,硕士生。

el Driven Architecture)是对象管理组织 OMG(Object Management Group)于 2001 年提出的软件系统开发方法。对 MDA 而言,模型不再是一种辅助工具,而是开发过程的产品,其核心即为模型转换。平台无关模型 PIM(Platform Independent Model)和平台相关模型 PSM(Platform Specific Model)是 MDA 中用于开发工作的计算模型。MDA 中 PIM 到 PIM 的转换实际上是模型之间的增强、过滤和精化。目前 MDA 更多地关注在 PIM 到 PSM 的转换上,忽视了 PIM 的精确性对系统需求的影响。因此,为了实现快速、精确建立系统的 SysML 模型,提出一种基于 UML Profile 通用扩展机制的模型转换方法:根据模型之间的关联关系,分别定义 SysML 需求图及用例图的模型扩展版型(Stereotype),通过元模型间的映射方法实现 SysML 需求图到用例图、用例图到序列图的转换。

1 国内外研究现状

文献[2]提出一种模型映射的九元组,并给出序列图到状态图转换的形式化定义,但该方法没有实现模型的自动生成,需要分段分析并进行整合。文献[3]采用规范化的描述语言,设计了一种用例图到序列图的半自动转换方法,但这种方法不但在描述语言中抽取信息时存在混淆错误的情况,还需额外设计模型图中元素的格局布置,准确度和自由度较低。文献[4]给出一种递归对象模型(ROM)到 SysML 模型的转换方法,通过分析系统需求设计其 ROM 模型,ROM 模型实际上也是规范化描述语言的图形化表示,通过分析 ROM 模型元素间得关系实现 ROM 到 SysML 的模型转换。文献[2-4]对于建立 SysML 模型图之间的映射、实现模型转换给出了参考思路:通过分析模型图的用途、元素组成,可以构建模型元素间的映射关联,以实现模型转换的目的。

文献[5]提出以扩展需求图为基础,实现其到用例图及状态图的转换。但其扩展没有依据 SysML 模型图之间的关联关系,只是将所有扩展信息加入需求图中,导致需求图过于臃肿,并且适用范围很窄。文献[6]采用业务流程建模标记(BPMN)方法,通过在计算无关模型(CIM)层次创建包含更多信息的业务模型,研究实现 CIM 到 PIM 的转换,与文献[5]类似,同样使用扩展源模型的方式来实现。因此,在建立模型间映射关联的基础上,通过扩展源模型所包含的信息、约束、关联关系是实现模型转换、达到快速开发建模目的的有效方法。

2 模型转换的扩展 Profile 设计

2.1 需求图到用例图的转换

SysML 需求图是由 UML 类图扩充而来,是 SysML 中用于沟通以文字形式记录的系统需求的一种主要媒介。建模者通常会创建需求图来表示需求之间的可跟踪性,以及从需求到系统结构和行为的可跟踪性,包括包含、跟踪、继承、改善、验证等关系[7]。SysML 用例图可以显示各种类型的元素和关系,说明系统提供的服务信息,以及需要服务的利益相关者的信息。在用例图中通过创建用例描述系统的功能性需求,包括基础用例、内含用例(include)和扩展用例(extend)。因此,根据需求图与用例图内容之间的关联关系,将需求图中的功能性需求部分通过模型转换得到用例图是可行的。

根据这种思想,首先需要使用 UML Profile 设计需求图的扩展版型(Stereotype),使其能够包含用例图中的 Actor、Include 以及 Extend 等元素,这是实现需求图到用例图转换的基础。如图 1 所示,在功能性需求扩展版型中:TargetUseCase 用于判断该需求是否为所需的功能性需求;TargetUser 表示需求所服务的使用者,同时通过枚举的方式列出使用者;IncludedBy 表示用例的内含关系;ExtendedFrom 和 ExtensionPoint 则表示用例之间的扩展关系。最后,将其导入到所建立的需求图中。

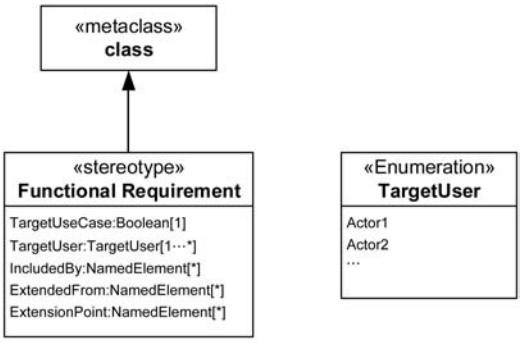

图 1 需求图的扩展 Profile

2.2 用例图到序列图的转换

SysML 中序列图可以表达系统动态行为信息,说明随着时间推移而发生的行为和事件的序列。通过生命线元素,为系统行为中的参与者建模,然后使用生命线之间的消息,为参与者之间的交互建模[7]。序列图主要包含的模型元素有:消息发送者、消息接收者以及消息序列。用例图是用来描述系统功能的,从面向对象的角度来看,系统的功能是由一组对象通过互相发送消息来完成,而序列图则是通过这样的对象和信息

来描述系统的动态行为。因此,序列图是描述用例的事件流场景[8]。

基于用例图与序列图的关联关系,同样使用 UML Profile 设计用例图的扩展版型(Stereotype),使用例图中每一个用例都可以完整表达其内部的事件流,如图 2 所示。首先,为了确定要进行模型转换的目标用例,设置 TargetUseCase 来进行判断,对布尔值为 true 的用例进行转换操作;其次,由于序列图中可能存在用例外部的"触发者"通过消息激活序列图事件流,所以设置多重性为[0,1]的 Trigger 作为触发者,TMessage 作为触发消息,TReceiver 作为接收消息的生命线元素表达这种情况;最后,通过发送者序列(SenderSequence)、接收者序列(ReceiverSequence)以及消息序列(MessageSequence)来表达生命线元素之间的消息交互。从纵向来看,消息的发送者、接收者和消息本身组成了一个消息传递的三元组,该三元组在横向上按照从左到右的方式对消息传递按照时间先后进行排序,从而可以得到消息传递的顺序。

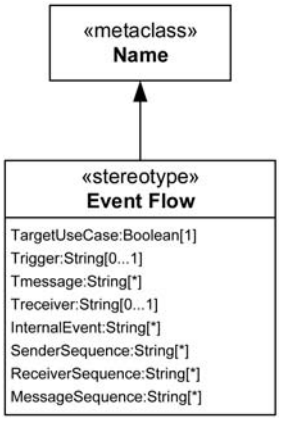

图 2　用例图的扩展 Profile

3　基于扩展 Profile 模型转换实现过程

3.1　ATL 模型转换语言

ATL(Atlas Transf ormation Language)[9-10]是 ATLAS、INRIA&LINA 和 Nantes 大学共同开发的一种符合 OMG 的 QVT 解决方案,它基于 EMF(Eclipse 模型框架),通过 EMF 来描述其元模型和模型。从本质上来说,ATL 是一种描述式和命令式混合的编程语言。一方面 ATL 的描述式可以简化源模型和目标模型元素之间的映射,另一方面命令式可以弥补描述式在映射关系上的不足[11]。

本文在 Eclipse 平台上通过插件 Papyrus 分别对需求图、需求图扩展 Profile 以及用例图扩展 Profile 进行建模。在此基础上,通过在 ATL 插件中使用 ATL 模型转换语言编写转换代码来实现模型转换。

3.2　需求图到用例图转换的 ATL 规则实现

首先,需定义模型(Model)层面的转换规则,由需求模型经 ATL 转换生成用例模型,作为容器为后续元素层面的模型转换奠定基础。规则可用如下伪代码来描述:

```
if exist RequirementModel && contain Profile then
    create UseCaseModel
```

在元素层面,将经 Profile 扩展的需求图中包含的元素分别对应转换为用例图中的执行者、用例、执行者与用例之间的关系以及用例之间的关系。在遍历需求图实例的基础上,转换规则如下:

(1) 若关键词 TargetUser 存在,则将其包含的元素转换为用例图中的 Actor,即执行者;

(2) 判断 TargetUseCase 设置的属性值,若为 true,将该实例转换为用例图中的 UseCase,即用例;

(3) 若实例中 TargetUser 存在并且 TargetUseCase 为 true,则在用例图中创建执行者与用例之间的关系(Association);

(4) 若需求图中 IncludedBy 存在,则将其包含的实例作为源端用例,该需求本身作为目标用例,建立用例之间的内含关系(include);

(5) 同理,若需求图中 ExtendedFrom 存在,则将其包含的实例作为目标用例,该需求本身作为源端用例,在目标用例中创建对源端用例的接口 ExtensionPoint,实现用例之间的扩展关系(extend)。

下列伪代码描述了元素之间的对应转换关系,将其放入模型层中实现需求图到用例图的转换过程:

```
traversal instances in RequirementModel
//创建执行者
if exist TargetUser then
    create Actor
    Actor. name = TargetUser. name
//创建用例
if TargetUseCase is true then
    create UseCase
    UseCase. name = TargetUseCase. name
//创建 Association
if exist TargetUser && TargetUseCase is true then
    create Association
    Association. from = Actor
    Association. to = UseCase
//创建内含关系
if exist IncludedBy then
    create include
    include. from = UseCase[ target ]
    include. to = UseCase[ self ]
```

```
//扩展关系与内含关系类似,不再赘述
……
```

3.3 用例图到序列图转换的 ATL 规则实现

在模型层面,与需求图到用例图的转换类似,由用例模型转换生成序列模型作为容器。与用例图不同的是,序列图还需在模型中创建一个交互(Interaction)面板,用于放置后续的模型元素。伪代码如下:

```
if exist UseCaseModel && contain Profile then
    create SequenceModel
    create Interaction
```

元素层面上,需由用例包含的事件流信息转换得到序列图中生命线、消息序列等元素。其转换规则设计如下:

(1) 设置布尔值属性关键词 TargetUseCase,若值为 true,则将该用例作为转换时序图的目标用例,获取该用例所包含的事件流信息;

(2) 若用例中外部触发者 Trigger 存在,将该元素转换为序列图中的 LifeLine,即生命线元素;

(3) 用例的触发消息中,以 Trigger、TMessage 以及 TReceiver 作为触发消息的三元组,判断三者的值属性转换得到序列图的触发消息;

(4) 若用例的内部事件 InternalEvent 存在,将其包含的所有元素均转换为生命线元素;

(5) 用例内部事件之间的消息交互,同样采用消息三元组 SenderSequence、MessageSequence 以及 ReceiverSequence 通过模型转换得到序列图中生命线元素之间的消息序列;

(6) 规定 TMessage 和 MessageSequence 消息交互的属性值首位以数字序号开头,用以判别消息交互的序列,以便模型转换后序列图的消息布局。

实现该模型转换的伪代码如下:

```
//判断是否为目标用例
if TargetUseCase is true then
    //创建触发者生命线
    if exist Trigger then
        create LifeLine
        LifeLine. name = Trigger. name
        //若存在触发者,则必定有触发消息
        //创建触发消息
        create LifeLine
        LifeLine. name = TReceiver. name
        create Message
        //触发消息必定在消息序列的首位
        Message. name = "0 " + TMessage. name
        Message. from = LifeLine[Trigger]
        Message. to = LifeLine[TReceiver]
    //创建内部事件生命线
    if exist InternalEvent then
        for i 0 to InternalEvent. length by 1 do
        create LifeLine
        LifeLine. name = InternalEvent[i]. name
    //创建内部消息
    for j 0 to MessageSequence. length by 1 do
        create Message
        //内部消息序列从 1 开始排序
        Message. name = "(j +1) " +
                MessageSequence[j]. name
        Message. from = SenderSequence[j]. name
        Message. to = ReceiverSequence[j]. name
```

4 基于扩展 Profile 的模型转换实例

本章选取电动汽车充电桩软件控制系统作为案例,用以验证上文研究的模型转换方法可行性。首先,通过分析充电桩软件控制系统的需求[12],在 Eclipse 平台下采用 papyrus 插件建立系统的需求图,并将其扩展 Profile 导入,得到扩展的系统需求图,如图 3、图 4 所示。

图 3 需求图导入扩展 Profile

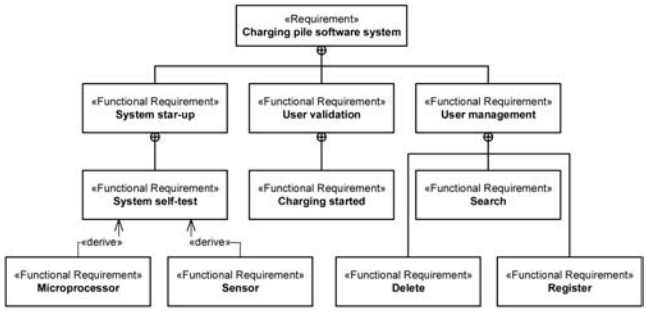

图 4 电动汽车充电桩软件控制系统需求图

同样在 Eclipse 平台下,通过 ATL 模型转换插件根据转换算法编写转换代码实现需求图到用例图的模型转换,如图 5 所示。

图 5 系统 ATL 模型转换程序界面

在执行 ATL 转换程序时配置其输入、扩展 Profile 以及输出得到电动汽车充电桩软件控制系统的用例图,如图 6、图 7 所示。

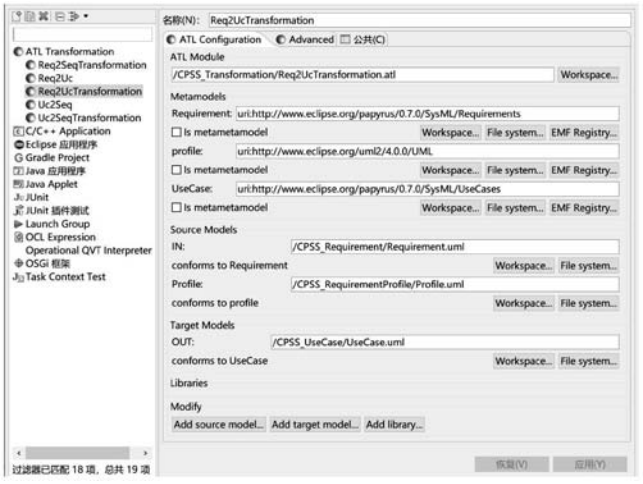

图 6　系统 ATL 模型转换程序配置信息

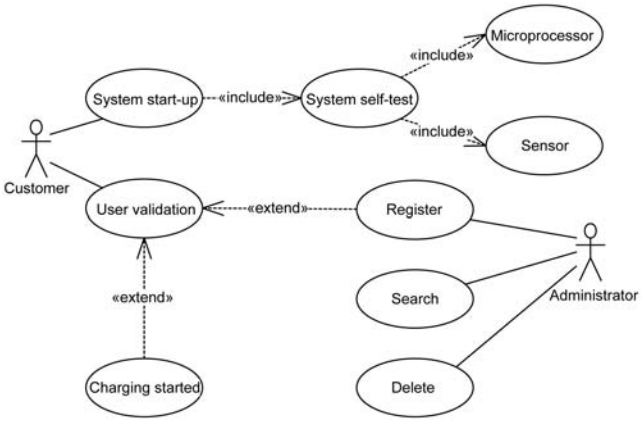

图 7　电动汽车充电桩软件控制系统用例图

同理,在得到的用例图中导入其扩展 Profile,根据 ATL 转换规则,得到电动汽车充电桩软件控制系统各个用例的序列图。以“用户验证”为例,其转换后的序列图如图 8 所示。

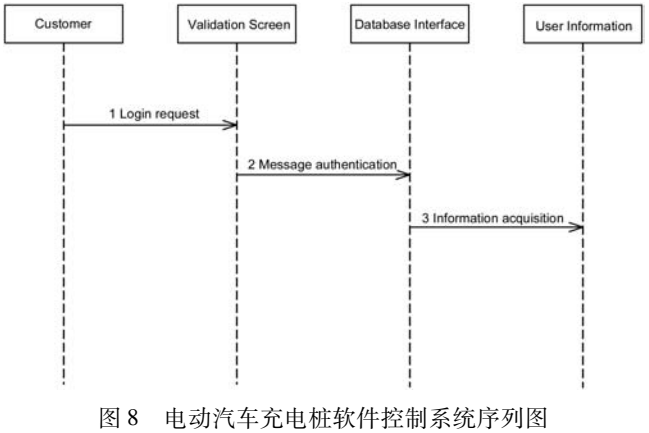

图 8　电动汽车充电桩软件控制系统序列图

5　结　语

针对目前系统开发需求复杂、领域交叉导致建模效率低、人为错误多等问题,本文提出一种基于扩展 UML Profile 的模型转换机制。在根据系统需求建立 SysML 需求图的基础上,建立需求图到用例图以及用例图到序列图的扩展 Profile。然后根据扩展 Profile 包含的系统信息使用 ATL 模型转换语言实现需求图到用例图、用例图到序列图的转换。最后,通过以电动汽车充电桩软件控制系统为例,验证了该方法的可行性。下一步,我们将研究关于序列图消息序列排序更好的表达方式,并实现用例图到序列图中组合片段的转换,建立更加完善的 Profile 扩展信息。

参　考　文　献

[1] 王松锋,熊选东,张亮忠,等. 基于有色 Petri 网的 SysML 序列图的分析与验证[J]. 计算机应用研究,2012,29(9):3341－3347.

[2] 史耀馨, 崔萌, 李宣东,等. 基于 MDA 的 UML 模型转换技术——从顺序图到状态图[J]. 计算机工程与应用, 2004, 40(13):40－45.

[3] 辛永博. 基于 MDA 的 UML 模型转换研究[D]. 西安:西安电子科技大学,2009.

[4] Wan W, Cheong H, Li W, et al. Automated transformation of design text ROM diagram into SysML models[J]. Advanced Engineering Informatics, 2016, 30(3):585－603.

[5] Chang C H, Lu C W, Yang W P, et al. A SysML based requirement modeling automatic transformation approach[C]// Proceedings of the 2014 IEEE 38th International Computer Software and Applications Conference Workshops. IEEE Computer Society, 2014:474－479.

[6] Rhazali Y, Hadi Y, Mouloudi A. Model transformation with ATL into MDA from CIM to PIM structured through MVC [C]//Proceedings of the 6th International Symposium on Frontiers in Ambient and Mobile Systems, 2016: 1096－1101.

[7] Delligatti L. SysML 精粹[M]. 侯伯薇,朱艳兰,译. 北京:机械工业出版社,2015:151－161.

[8] 陈磊. 用例图到顺序图转换的研究[D]. 西安:西安电子科技大学,2009.

[9] Jouault F, Kurtev I. Transforming Models with ATL[C]// Proceedings of the 2005 international conference on Satellite Events at the MoDELS. Springer-Verlag, 2005:128－138.

[10] PIERSW. M2M/Atlas transformation language (ATL)[EB/OL]. (2010). http://wiki.eclipse.org/ATL.

[11] 郭鹏,李亚晖,李明娟,等. 基于 ATL 引擎的 UML 到 Simulink 模型转换方法研究[J]. 航空计算技术,2015,45(2):129－134.

[12] 魏国,商慧杰,朱春波,等. 电动汽车交流充电桩系统设计[J]. 现代电子技术,2012,35(21): 124－126.