面向嵌入式软件开发的UML到Simulink模型转换方法

郭 鹏[1,2] 李亚晖[1,2] 孙 磊[3] 蔡晓乐[1]

(中航工业西安航空计算技术研究所 西安 710065)[1]

(机载弹载计算机航空科技重点实验室 西安 710065)[2] (北京航空航天大学计算机学院 北京 100191)[3]

摘 要 模型驱动开发及其关键技术模型转换是近年来软件工程领域研究的热点。在嵌入式软件开发早期,不仅需要对设计模型进行静态分析,更需要对其进行动态仿真,验证系统设计的正确性。如何把设计模型和仿真模型无缝连接起来是工业部门亟待解决的问题。深入调研了UML和Simulink模型转换研究现状,详细分析了模型驱动开发中模型转换的相关技术,提出了一种UML到Simulink的模型转换方法,设计了UML元模型、Simulink元模型,撰写了UML元模型到Simulink元模型的映射规则。最后选取自动驾驶仪系统的飞行控制软件作为案例,验证了该方法的正确性。该方法能实现UML和Simulink两种异构模型同构化,提高嵌入式软件开发效率,丰富并且完善模型驱动开发,也为飞行控制系统、高速铁路控制、机载航电系统等嵌入式软件开发提供了技术支持。

关键词 模型驱动开发,模型转换,元模型,ATL,UML,Simulink

中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.2.042

UML Model to Simulink Model Transformation Method in Design of Embedded Software

GUO Peng[1,2] LI Ya-hui[1,2] SUN Lei[3] CAI Xiao-le[1]

(AVIC Xi'an Aeronautics Computing Technique Research Institute, Xi'an 710065, China)[1]

(Aviation Key Laboratory of Science and Technology on Airborne and Missile-borne Computer, Xi'an 710065, China)[2]

(School of Computer Science and Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100191, China)[3]

Abstract Model driven development and its key technique model transformation are research hotspot of software engineering in recent years. At the early stage of embedded software development, design model not only requires static analysis, but also needs dynamic simulation, verifying correctness of system design. How to transform design model to simulation model is a serious problem to industrial department. This paper surveyed model transformation research status, analysed related model transformation techniques of model drive development, proposed a model transformation method from UML to Simulink, built UML meta-model and Simulink meta-model, designed a set of mapping rule between UML meta-model and Simulink meta-model. Finally, this paper validated technique and method correctness using automatic flight control system as antitype. The method makes two isomerism models homogeneous, improving the efficiency of embedded software development, enriching MDD technique, and providing technique support for embedded software development, such as automobile control system, express control system, and avionics system.

Keywords Model driven development, Model transformation, Meta-model, ATL, UML, Simulink

1 引言

对象管理组织(OMG)提出了一种新的软件开发方法和理论,即模型驱动开发(MDD)。模型驱动开发把软件的开发过程提升到模型的高度,模型成为软件开发的核心,通过对模型进行建模、转换和精化来驱动软件的开发。其核心思想是分离软件需求规格和特定平台的实现,使系统的功能不依赖于不同平台上的具体实现,最终缩短软件开发时间,适应不断变化的需求[1]。

根据描述场景和问题领域的不同,模型驱动开发中的模型分为计算无关模型(CIM)、平台无关模型(PIM)、平台相关模型(PSM)。平台无关模型到平台相关模型转换是当前模型驱动开发的研究热点,也是模型驱动开发的关键技术[2],其中广泛使用UML作为平台无关模型的建模语言。著名的UML建模工具包括IBM公司的Rational Software Architecture、Rhapsody,Atego公司的Artisan Studio,Ellidiss公司的Stood,开源软件Papyrus等。现如今越来越多的工业部门使用这些商业工具和开源工具来完成UML设计模型,并且这种趋势越来越明显[3],这些工业部门包括洛克西德马丁公司、空客公司、波音公司、西门子公司等。

嵌入式系统中,实时性、可靠性、安全性是关键的质量属性,如果不能满足这些关键属性,系统可能遭受毁灭性的后果。UML/MARTE(Modeling and Analysis of Real Time and Embedded Systems)可以在设计阶段对实时性、可靠性、安全

到稿日期:2014-12-03 返修日期:2015-04-30 本文受航空科学基金项目(2013ZC31005)资助。

郭 鹏(1987-),男,硕士,助理工程师,主要研究方向为嵌入式系统建模与仿真,E-mail:nwpu062475@163.com;李亚晖(1976-),男,博士,高级工程师,CCF会员,主要研究方向为嵌入式系统安全和网络安全协议;孙 磊(1989-),男,硕士生,主要研究方向为软件工程和软件仿真;蔡晓乐(1991-),男,硕士生,主要研究方向为计算机存储。

性等非功能属性建模，进而采取静态分析验证的方法证明系统设计的正确性。但是许多重要指标仅仅通过前期的静态分析是不够的，需要采取动态仿真方法获取更多的数据，进而对仿真结果进行分析，更进一步验证 UML 设计的正确性。

虽然有些商用 UML 工具支持仿真，例如 Rhapsody，但是在连续动态行为建模和离散状态行为建模领域，它与专门的仿真软件还是有不小的差距。另一方面 UML 是一种通用的建模语言，缺少对嵌入式领域的强力支持，因此采用专门的仿真软件可以弥补 UML 动态行为建模方面的不足。其中 Simulink就是一款功能强大的仿真工具，支持在交互式、图形化环境中对动态系统进行仿真和分析。更重要的是，针对嵌入式的领域特点，Simulink 提供了许多具有特殊功能的工具包，以方便软件开发人员使用，例如 Aerospace Blockset 提供火箭、飞机的作动器、飞行参数、飞行环境模型，Communication System 支持对通信总线建模，Control System 支持对控制系统建模等[4]。

现阶段，大多数软件工程人员采取手工方式完成从 UML 设计模型到 Simulink 仿真模型的转换。随着嵌入式软件复杂程度越来越高，功能越来越强大，对实时性、可靠性、安全性的要求越来越高，手工转换方式带来的弊端也越来越明显。一是耗费大量人力资源，现阶段对于工业部门而言，人力成本越来越高，已经成为工业部门的一项巨大开支；二是不同的软件开发人员从相同的 UML 模型能得到不同的 Simulink 仿真模型，不能保证模型的一致性；三是手工转换容易产生错误，特别是年轻的软件工程师，由于缺少相关工程经验，更易出错。如何把 UML 模型和 Simulink 模型无缝衔接起来，成为工业部门一个亟待解决的难题。

针对上述问题，本文重点研究 UML 设计模型到 Simulink 仿真模型的自动转换。本文运用 MDD 相关技术，使用 UML 建立 PIM 模型，使用 Simulink 建立 PSM 模型，通过 ATL(Atlas Transformation Language)模型转换引擎完成 UML 到 Simulink 的自动转换。本文第 1 节介绍了 MDD 开发方法，进而引出 UML 和 Simulink 模型转换的需求；第 2 节介绍国内外研究现状；第 3 节研究 ATL 模型转换引擎的相关技术，包括 ATL 模型转换框架、KM3(kernel metametamodel)语言、ATL 语言；第 4 节研究 UML 模型到 Simulink 模型的转换方法，涉及 UML 元模型和 Simulink 元模型的建立、UML 元模型和 Simulink 元模型的映射规则；第 5 节选取自动驾驶仪的自动飞行控制软件作为案例，验证模型转换方法的正确性；最后总结本文，并对未来工作提出展望。

2 研究现状

Matlab/Simulink 在信号处理建模方面具有强大优势，UML 能以一种统一视角涵盖系统架构、设计和验证的全过程，为了充分结合这两种建模语言的优势，Yves Vanderperren、Wim Dehaene[5]提出了两种 UML 和 Matlab/Simulink 的联合方法。第一种方法是 UML 和 Simulink 的联合仿真，通过一个中间的工具，例如 Extessy AG 的 Exite 工具，联合 Sinumlink 模型和 Studio 或者 Rhapsody 的 UML 模型，但这对 UML 和 Simulink 的时间同步提出了较高要求；第二种方法是基于一种公共可执行语言的模型集成，例如 C/C++语言。在 Real-Time Innovation 或者 GeneralStore 集成平台中，联合 Matlab Compiler 或者 Real-Time Workshop 产生的 C/C++代码和 UML 生成 C++代码。这两种方法的主要问题是过分依赖第三方工具的支持，前者依赖工具保证两种模型的同步，后者依赖开发平台减轻代码生成的难度。

针对嵌入式系统中不同学科类型需要多种工具和建模语言的联合使用的情况，瑞典皇家理工学院的 Carl-Johan Sjöstedt、Martin Torngren 等研究了从 Simulink 到 UML 的模型转方法[6]。重点关注连续时间模型和离散时间模型，实现 Simulink 静态模块到 UML 组合结构图的映射，以及 Simulink 行为模型到 UML 活动图的映射，最终使用 ATL 转换语言形成一个 Simulink 到 UML 的模型转换的雏形。但是该研究存在一些弊端，生成的活动图不能运行，也不能产生仿真结果，现阶段仅仅能追踪先前的需求。类似地，日本东京城市大学的 Tatsuya Kamiyama、Takahiro Soeda 研发了一款从 Simulink 到 UML 的模型转换工具[7]，该工具从 Simulink 模型中的子系统模块自动生成对应的类或者对象，同时手工添加非功能属性，最终将完成控制逻辑设计到软件设计的过渡，该工具应用于汽车控制领域。日本学者与瑞典学者的研究目标不同，日本学者先用 Simulink 完成控制模型，通过仿真验证控制模型的正确性，在此基础上，通过模型转换实现软件设计模型。

国内主要研究 UML 和 Simulink 的协同仿真，南京航空航天大学的刘兴华博士使用 Simulink 模型嵌入 SysML 模型的方式实现两者之间的协同仿真[8]。具体来说，Simulink RTW 生成 Simulink 的 C++代码，Rhapsody 生成 SysML 的 C++代码，最终两种代码集成在 Rhapsody，这种方法类似于文献[5]中的第二种方法。

综上所述，目前 UML 和 Simulink 的模型转换主要有两种技术手段：代码集成和模型映射。代码集成的劣势很明显，随着系统规模越来越大，UML 和 Simulink 两种异构模型融合的复杂度也将急剧增大，并且代码集成的复用性差。而 UML 和 Simulink 的模型映射是未来的趋势，但是国内外研究缺少对相关模型转换理论的深入研究，例如 MDD 相关技术缺少对元模型的设计，主要局限于特定的语言和特定的平台，不具有通用性。因此，需要以 MDD 为理论基础，研究模型转换框架，建立 UML 和 Simulink 元模型不仅仅能够完成 UML 到 Simulink 模型的转换，也为以后 AADL 到 UML、AADL 到 SCADE 模型转换提供技术支持。

3 ATL 模型转换引擎

法国 INRIA 的 ATLAS 研究组织提出了一个模型驱动开发平台，即 AMMA(ATLAS model management architecture)。AMMA 提供 KM3、TCS(Textual Concrete Syntax)、ATL 等工具集，支持(元模型)建模、模型转换、模型编织和模型管理[9]。

3.1 ATL 模型转换框架

ATL 模型转换引擎是基于 AMMA 平台的 KM3、TCS 以及 ATL 实现的，其结构层次类似于 OMG 的 MOF(Meta-Object Facility)，但比 MOF 结构更加简单、实用[10]。图 1 显示 ATL 模型转换框架，整个框架分为 3 层：M3 元元模型层、M2 元模型层和 M1 模型层，其中元元模型解释说明元模型，元模型解释说明模型。具体来说，KM3 是整个模型转换引擎的元元模型，通过 KM3 自身定义。源元模型和目标元模型也是根据 KM3 元元模型定义的。源模型根据源元模型建

立，目标模型根据目标元模型建立。ATL 模型依赖 ATL 语言，实现源元模型、目标元模型的映射。最后符合源元模型的源模型作为 ATL 模型转换引擎的输入，自动生成符合目标元模型的目标模型，从而实现源模型和目标模型两种异构模型的同构化。

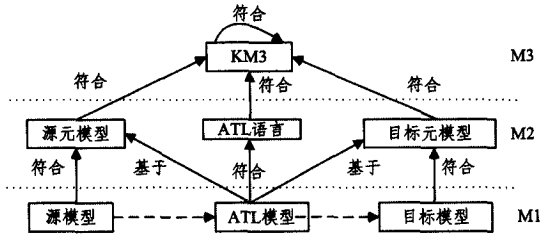

图 1　ATL 模型转换框架[10]

3.2　KM3

KM3 是一种轻量级的文本元模型定义语言，它的语义清晰，适合创建和修改元模型。同时 KM3 的实用性比较强，主要强调类和关系[11,12]。

KM3 元模型结构中，KM3 元模型由一系列 Package(包)构成，一个 Package 由一些抽象 ModelElement(模型元素)实体组成。ModelElement 是一个元模型，拥有 name 属性，其他元素都继承于 ModelElement。ModelElement 的子类包括 Enumliteral(枚举值)、Classifier(分类器)、Annotation(注释)、DataType(数据类型)、TypedELement(类元)等。Annotation 由一系列 AnnotationLink(注释关联)组成。Enumeration(枚举)至少包括一个 Enumliteral。DataType(数据类型)和 Class(类)继承于 Classifler。Classifler 和 Class 至少包括一个 TypedELement 类型元素。TypedElement 定义了 lower、upper、isOrdered 和 isUnique 属性。StucturalFeature(结构特征)和 Operation(操作)继承于 TypedElement。ModelElement 各个元素结构及其关系如图 2 所示。

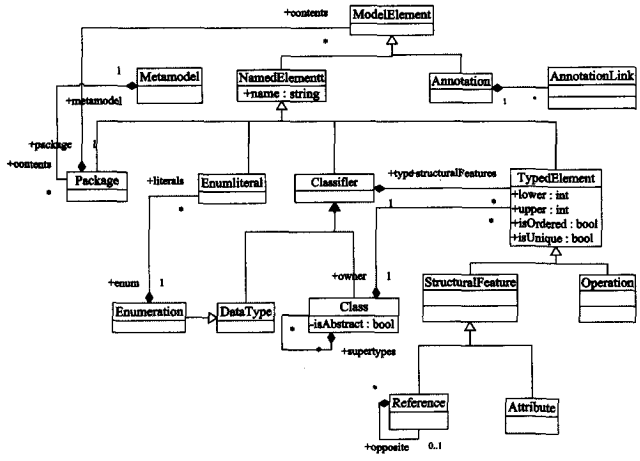

图 2　KM3 的元模型结构

3.3　ATL

ATL 是 ATLAS、INRIA & LINA 和 Nantes 大学共同开发出来的一种符合 OMG QVT 的解决方案。它是元模型和文本语法的模型转换语言。在模型驱动开发中，ATL 为软件工程师提供了一系列从源模型到目标模型的转换方法[13]。

ATL 语言为 ATL 开发人员提供不同的 ATL 单元，包括 ATL 模块、ATL 查询和 ATL 库 3 种类型。ATL 模块包括 Header section、Import section、Helpers、Rules，帮助软件工程人员设计源模型到目标模型的转换方法。ATL 查询由一个模型到原始类型值的转换组成，ATL 查询可以认为是一个计算一系列源模型原始值的操作，ATL 查询的最主要用途是从源模型中产生文本输出。ATL 单元的最后一个类型是 ATL 库，提供了一系列可以被不同 ATL 单元(模块、查询、库)调用的 ATL helper。

4　UML 到 Simulink 模型的转换方法

本文选用 Simulink 模型作为仿真模型、UML 模型作为设计模型，使用形式化方法实现两种异构模型的转换。分别设计 UML 数据类型、活动图、状态机元模型和 Simulink 数据类型、控制流、Stateflow 元模型，在元模型的基础上使用 ATL 语言实现模型间的映射。

4.1　UML 元模型

4.1.1　UML 数据类型

在 UML 中数据类型类似于类，但其是一个特殊的分类器，它的实例通过值来识别。UML 的 DataType 继承于 Classifier，派生两个子类 PrimitiveType 和 Enuneration。其中 PrimitiveType 包含 4 个子数据类型：Boolean、Integer、UnlimiteNatural和 String[14]。这里重点关注预先定义的数据类型 PrimitiveType，同时根据嵌入式领域的特点扩展了 Real 和 Datetime 两个数据类型。具体 UML 数据类型元模型如图 3 所示。

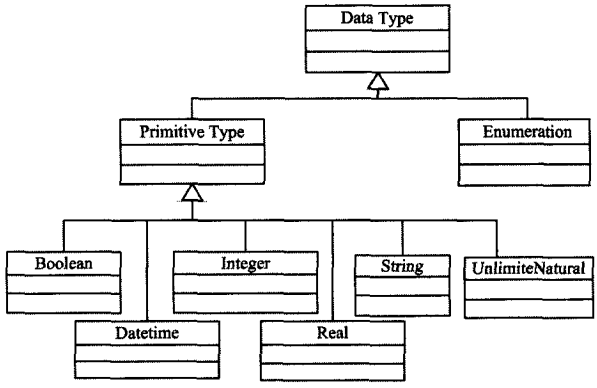

图 3　UML 数据类型

4.1.2　UML 活动图元模型

UML 的活动由活动图表示，它是一幅由控制流和对象流连接起来的活动节点的图。活动节点代表嵌套的活动、动作数据单元以及控制构造，活动节点的流对活动中的控制和数据流建模。活动具有输入和输出参数，分别代表提供给活动执行所需要的值以及执行活动所需要的值。UML 活动图元模型十分复杂，这里仅仅表示其中主要的元素。根据《OMG Unified Modeling LanguageTM (OMG UML)，Superstructure》版本 2.4.1[14]，UML 活动图的元模型如图 4 所示。

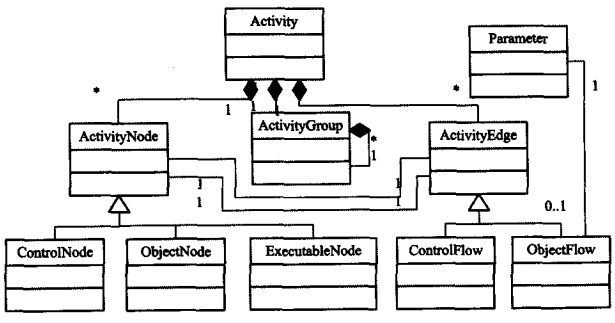

图 4　UML 活动图元模型

4.1.3 状态机元模型

UML 状态机是一个类/对象可能经历的所有历程的模型图。状态机由对象的各个状态和连接这些状态的转换组成。每个状态对一个对象在其生命期中满足某条件的一个时间段建模。当一个事件发生时，它会触发状态间的转换，导致对象从一种状态转化到另一种新的状态。当转换开始时，会发生与转换关联的某种效果(动作或者活动)[14]。

UML 状态机的简化元模型如图 5 所示。一个状态机由仅有的一个顶级的状态机元素、一系列状态元素和转移元素组成。其中虚状态、状态、终止状态继承顶点状态，初始状态、连接状态、分叉状态继承虚状态，顶点状态由执行活动、进口活动、出口活动组成，转移由监护条件、触发条件、效果组成，顶级状态和转移存在对应关系，一个顶级状态对应一个或者多个转移。

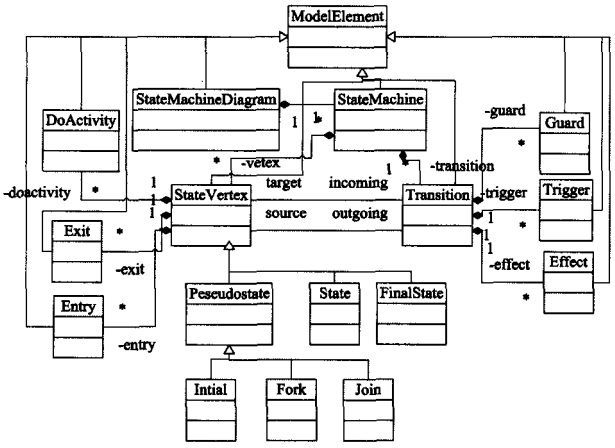

图 5 UML 状态机元模型

4.2 Simulink 元模型

4.2.1 Simulink 数据类型

不同于 UML，Simulink 不需要类型声明。一些模块(例如 Unit Delay 和 SUM)自身没有类型约束。相反，这些模块的数据类型继承于输入数据的类型，但是 Simulink 拒绝类型错误。Simulink 自定义的 9 种数据类型包括：boolean、double、single、int8、uint8、int16、uint16、int32 和 uint32[15]。在 ATL 模型转换引擎中，根据需要使用 KM3 语言定义 Simulink 数据类型元模型，如图 6 所示。

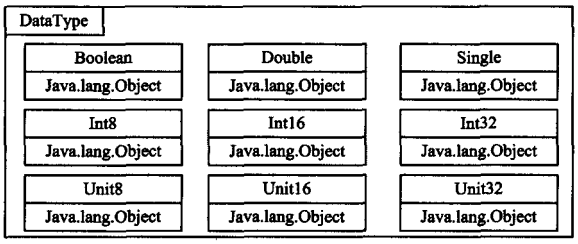

图 6 Simulink 数据类型

4.2.2 Simulink 元模型

Simulink 拥有丰富的预定义模型库和一系列相关的建模规则。图 7 仅仅是 Simulink 元模型的一部分，其中 SimulinkObject 是 Simulink 元模型的基类，Libraty 表示 Simulink 的模型库，每一个 Library 包括多个模块，模块对应于 Simulink 元模型中的类 Block，模块(Block)关联相关的参数(Parameter)。一个特殊的模块是 Subsystem，由 Block、Port、Line 组成。通道(Line)连接模块的输入端口(Inport)和输出端口(Outport)。这些 Simulink 元素组成 System[16]。

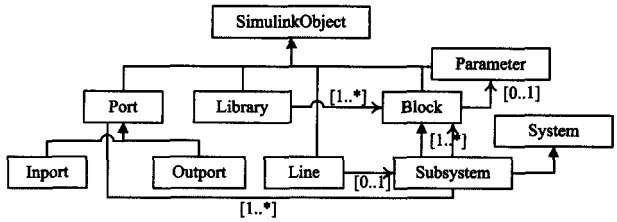

图 7 Simulink 元模型

4.2.3 Stateflow 元模型

Stateflow 是 Matlab/Simulink 中重要的组成元素，Stateflow 能够对基于 FSM(有限状态机)理论的事件驱动系统进行建模和仿真，也可以对复杂逻辑系统进行建模和仿真。UML 有严格的语法、语义，然而 Stateflow 不同，需要对其元素仔细分析，从而创建 Stateflow 元模型[16]。

Stateflow 的基本元素包括：State Machine(状态机)、Chart(图块)、Diagram(框图)、State(状态)、Entry、During、Transition(转移)、event、condition、condition_action、transition_action等。本文设计出 Stateflow 的元模型，如图 8 所示。SimulinkElement 是所有元素的基类，StateMachine 由 Diagram 组成，Diagram 由 State 和 Transition 组成，ExclusiveState 和 ParalleState 是 State 的子类，State 包含 DoActivity 和 Entry，同时 Transiton 包含 Condition、Trigger、Effect。

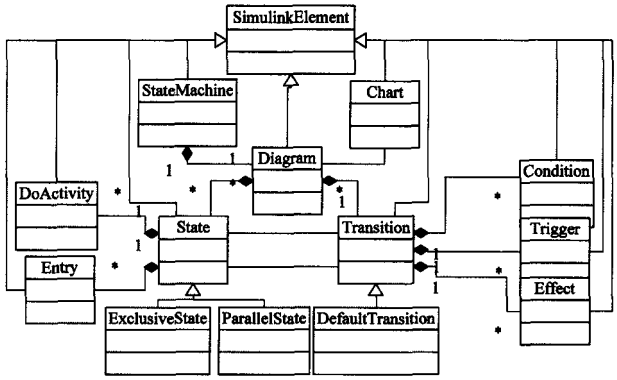

图 8 Stateflow 元模型

4.3 UML 状态机与 Stateflow 映射关系

4.3.1 数据类型映射

UML 的数据类型包括枚举类型和基本类型，其中枚举类型不适合 Simulink，因此不参与映射。基本数据类型的 UnlimiteNatural 是一种技术规定，在规定多重性时使用，显然不符合 Simulink，因此也不参与映射。具体的数据类型映射关系如表 1 所列。

表 1 UML 和 Simulink 数据类型映射关系

UML 数据类型	Simulink 数据类型	解释
Boolean	boolean	直接映射
Integer	Int32	UML 的 integer 映射为 Simulink 取值范围最大的 int32
Real	double	UML 的 Real 代表实数，直接映射为 Simulink 的 double
DataTime	double	DataTime 直接映射为 double

4.3.2 活动图映射

UML 活动图提供丰富的表示法表示一系列活动。活动图与 Simulink 的控制流程图非常相似，由于 UML 活动图元

模型过于复杂，这里选择其中的关键元素进行转换。UML 活动图到 Simulink 控制流的转换方法分为两步：第一步是 UML 活动图基本元素到 Simulink 基本元素的映射，如表 2 所列。这里对嵌入式领域使用频率较高的两个控制节点 Derivation 和 Join 进行了映射，分别对应 Simulink 信号路由模块的 Demux 和 Mux 模块。第二步是在基本元素映射的基础上，对基本元素的内部信息进行映射。一个 UML 活动图的基本元素内部包含丰富的信息，包括基本元素的唯一标示符 ID、name 以及它所关联的其他元素。以 ActivityNode 到 Block 的转换为例，需要把 ActivityNode 的 type、id、name、incoming、outgoing 等信息映射到 Simulink 的 block。

表 2　活动图和 Simulink 映射关系

活动图	Simulink	解释
Activity	System	一个活动图映射为 Simulink 的系统
ActivityEdge	Line	活动图的边映射为 Simulink 的通道
ActivityNode	Block	活动图的节点映射为 Simulink 的模块
Parameter	Parameter	活动图的参数映射为 Simulink 的参数
Derivation	Demux	活动图的控制节点-“派生”映射为 Simulink 信号路由模块-Demux
Join	Mux	活动图中的控制节点-“连接”映射为 Simulink 信号路由模块-Mux

4.3.3　状态机映射

由 4.1.3 节 UML 状态机元模型和 4.2.3 节 Stateflow 元模型可知，二者具有较高的相似性，都是由状态、转移、状态动作、时间、效果、条件等元素组成。依据 UML 状态机和 Stateflow 的元模型，设计 UML 状态机和 Stateflow 的映射规则，如表 3 所列。映射的步骤与 4.3.2 活动图映射步骤类似。

表 3　UML 状态机和 Stateflow 的映射规则

序号	映射关系	解释说明
1	ModelElement→SimulinkElement	ModelElement 是 UML 状态机所有元素的基类，对应地设计 Stateflow 所有元素的基类 SimulinkElement。ModelElement 映射到 SimulinkElement
2	StateMachineDiagrams→StateMachine	对于状态机这个词，UML 和 Simulink 的理解略有偏差。UML 的 StateMachineDiagram 由 StateMachine 组成。Simulink 的 StateMachine 由 Chart 组成。因此 UML 的 StateMachineDiagram 与 Simulilnk 的 StateMachine 含义基本相同
3	StateMachineDiagram→Chart	与 2 不同，这里的 StateMachineDiagram 指单个状态图，对应到 Simulink 中的 Chart
4	StateMachine→State	UML 的 StateMachine 由 State、Transition 等组成，Simulink 的 State 中也包含嵌套 State、Transition 等元素
5	State→State	与 U2S4 不同，这里 UML 的 State 指非嵌套的状态。UML 和 Simulink 中的 State 含义相同，可以直接映射
6	Transition →Transition	UML 和 Simulink 中的转移含义基本相同，直接映射
7	DoActivity→DoActivity	UML 和 Simulink 中的持续活动含义基本相同，直接映射
8	Entry → Entry	UML 和 Simulink 中的进口活动含义基本相同，直接映射
9	Condition →Condition	UML 和 Simulink 中的条件含义基本相同，直接映射
10	Effect→ Effect	UML 和 Simulink 中的效果含义基本相同，直接映射
11	Trigger→Trigger	UML 和 Simulink 中的事件含义基本相同，直接映射

5　实例分析

本节使用自动驾驶仪中的飞行控制软件作为案例，说明如何利用 UML 到 Simulink 转换引擎完成从 UML 模型到 Simulink 模型的转换，同时验证本文提出方法的正确性和有效性。

5.1　自动飞控软件简介

自动驾驶仪的基本功能是控制飞机的飞行，并能在无须飞行员操纵飞机的情况下使飞机按预定航线飞行。自动驾驶仪控制回路如图 9 所示。其中自动飞控软件是自动驾驶仪的神经中枢，必须能实时采集外部信号源数据、对数据进行交叉传输、表决监控、控制律计算、实时输出控制律计算结果，辅助飞行员控制飞机的飞行[17,18]。

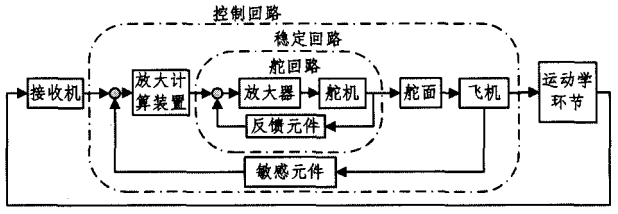

图 9　控制回路

5.2　自动飞控软件建模

使用 IBM 的 Rational Software Architecture 对自动飞控软件进行建模。根据自动驾驶仪的功能建立系统的活动图，如图 10 所示。根据自动驾驶仪当前的故障类别，自动驾驶仪的状态分为正常状态、瞬时故障状态、伪正常状态、永久故障状态。其中根据故障的危险程度，永久状态又分为第一类永久状态、第二类永久状态、第三类永久状态，具体的状态机如图 11 所示。

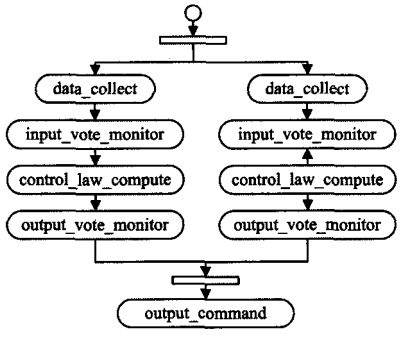

图 10　自动飞控软件活动图

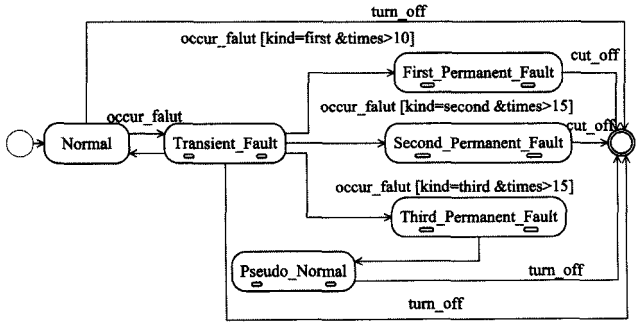

图 11　自动飞控软件状态机

5.3　UML 到 Simulink 模型的转换

利用 Eclipse 工具完成 UML 到 Simulink 模型转换引擎，添加 KM3 语言设计的 Simulink 元模型(. ecore)、UML 元模

型(.ecore)以及 ATL 语言撰写的 UML 到 Simulink 模型转换规则。在上述基础上,输入 5.2 节 Rational Software Architecture 实现的 UML 源模型(.xmi),UML 到 Simulink 模型转换引擎自动生成 Simulink 目标模型(.xmi),再进一步使用 Matlab 脚本程序对目标模型进行解析,生成 Simulink 可以识别的 mdl 文件。根据应用程序需求和仿真周期次数多(例如机载故障一般在 10^{-9},因此仿真周期数大于 10^{9} 次),使用 Matlab 的工作空间和 S 函数存储和处理系统所需要的外部激励,包括 Sensor、control crank、Pilot control console 和 actuator 等外设信息。最终形成 Matlab/Simulink 模型,如图 12、图 13 所示。

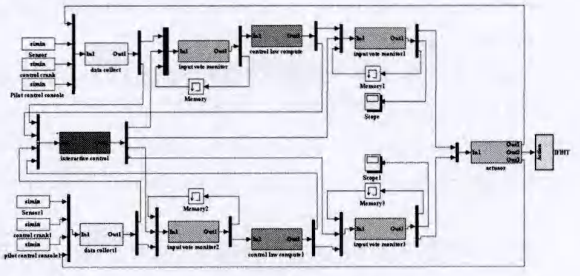

图 12 Simulink 模型

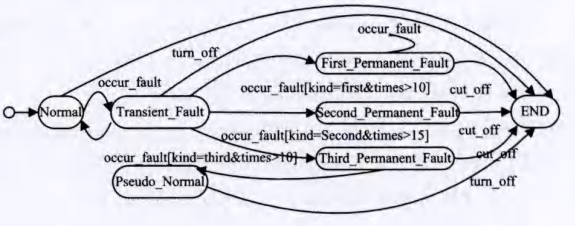

图 13 Stateflow 模型

5.4 仿真分析

案例中的 UML 设计模型引入相关可靠性、实时性建模方法,由于篇幅限制,不对上述建模方法进行讨论。此处的仿真和分析评估,不是针对案例中系统的业务逻辑,而是针对设计模型中的可靠性、实时性建模方法,这里主要以实时建模方法为例。

对模型转换引擎生成的 Simulink 模型进行必要修改,根据案例原型特点在 300 个周期内注入不同的外部激励,同时对系统可能产生的故障进行排列组合,随机注入到不同的周期内,记录每一个操作的时间、各个任务的时间、各个分区的时间和总时间。统计系统在故障模式下不同时间的波动情况。如果时间的波动在系统时间约束的范围内,则证明相关建模方法正确,否则错误。如图 14 所示,在仿真的第 60 个周期,仿真时间强烈抖动,并达到了峰值 15ms,但是峰值都在系统时间约束范围内。仿真结果证明了相关实时性建模方法的正确性,也说明了本文的模型转换方法的正确性。

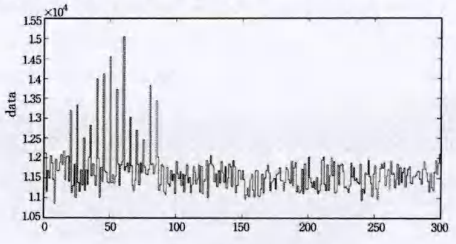

图 14 分区 1 的时间统计图

结束语 模型转换作为模型驱动开发的关键技术,这些年来已成为工业界和学术界的研究热点。特别是嵌入式领域,迫切希望实现 UML 模型和 Simulink 模型的相互转换。本文以 UML 模型到 Simulink 模型的转换为需求,设计 UML 数据类型、活动图、状态机元模型和 Simulink 数据类型、模型、Stateflow 元模型,运用 ATL 语言完成 UML 元模型和 Simulink 元模型的映射。最终实现 UML 模型到 Simulink 模型的自动转换。这些研究成果丰富并且完善了模型驱动开发,也为汽车控制、高速铁路控制、机载航电系统等嵌入式软件开发提供了技术支持。

本文未来的工作主要集中在两方面:1)在第二章模型转换的相关理论的基础上,设计模型总线,实现 AADL 到 Simulink、AADL 到 SCADE、AADL 到 UML 等模型转换;2)继续完善 UML 到 Simulink 模型转换,补充 UML 时序图、类图到 Simulink 模型的转换,同时延伸 Stateflow 模型到 C 代码的自动生成。

参 考 文 献

[1] Object Management Group (OMG). MDA guide version1.0.1 [EB/OL]. (2003-06-01)[2014-07-11]. New York: Object Management Group. http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf.

[2] Qi Tie-lin. Research and Implementation on Model Transformation Method in MDA Model Transformation Platform[D]. Beijing: Beijing University of Technology, 2010(in Chinese)
戚铁林. MDA 模型转换平台中模型转换方法的研究与实现[D]. 北京:北京工业大学,2010

[3] Larman C. UML 和模式应用[M]. 李洋,译. 北京:机械工业出版社,2013

[4] 李颖. Simulink 动态系统建模与仿真[M]. 西安:西安电子科技大学出版社,2009

[5] Vanderperren Y, Dehaene W. From UML/SysML to Matlab/Simulink: Current State and Future Perspectives[C]// Proceedings of the conference on Design, automation and test in Europe. Munich, Germany, 2006

[6] Shi J, Törngren M, Servat D, et al. Combined Usage of UML and Simulink in the Design of Embedded Systems: Investigating Scenarios and Structural and Behavioural Mapping[C]// The 4th workshop of Objectoriented Modeling of Embedded Real-time Systems. Paderborn, Germany, 2007

[7] Kamiyama T, Soeda T, Yoo M. A Simulink to UML Transformation Tool for Embedded Control Software Design[J]. International Journal of Modeling and Optimization, 2012, 2(3): 197-201

[8] Liu Xing-hua, Cao Yun-feng, Wang Biao, et al. Flight Control System Conceptual Prototype Design Based on SysMl and Simulink[J]. Journal of University of Electronic Science and Technology of China, 2011, 40(6): 888-891(in Chinese)
刘兴华,曹云峰,王彪,等. 基于 SysML 与 Simulink 的飞控系统概念样机设计[J]. 电子科技大学学报,2011,40(6):888-891

[9] The Eclipse Foundation. ATLAS Transformation Language [EB/OL]. (2003-06-07) [2014-07-11]. http://wiki.eclipse.

org/ATL/Tutorials

[10] The Eclipse Foundation. The Eclipse Modeling Framework Overview [EB/OL]. (2011-12-03) [2014-07-11]. http://www.eclipse.org/emf.

[11] ATLAS group LINA & INRIA Nantes. ATLAS Transformation Language (ATL) Home Page [EB/OL]. (2011-12-03) [2014-07-11]. http://www.eclipse.org/atl/documentation/old/ATL_User_Manual[v0.7].pdf

[12] Jounult F, Bezivin J. KM3: A DSL for metamodel specification [M]// Gorrieri R, Wehrheim H, eds. Proc. of the 8th IFIP Int'l conf. on Formal Methods for Open Object-Based Distributed System. Berlin: Springer-Verlag, 2006: 171-185

[13] Jouault F, Allilaire F, Bezivina J, et al. ATL: A model transformation tool [J]. Science of Computer Programming, 2008, 72 (2): 21-29

[14] Object Management Group (OMG). UML2.0 Infrastruture-Specification[EB/OL]. (2009-04-01) [2014-07-11]. New York: Object Management Group. http: www.omg.org.does/ptc/03-09-15.pdf

[15] Zhang Tian, Jouault F, Attiogbe C, et al. MDE-Based Mode Transformation: From MARTE Model to FIACRE Model[J]. Journal of Software, 2009, 20(2): 214-233(in Chinese)
张天, Jouault F, Attiogbe C, 等. 基于 MDE 的异构模型转换: 从 MARTE 模型到 FIACRE 模型[J]. 软件学报, 2009, 20(2): 214-233

[16] Han Zhen, Zhao Quan-xiang. Logic Modeling and Application of Under Water Control System [J]. Automation Application, 2011, 11(3): 23-27(in Chinese)
韩振, 赵全香. 水下控制系统逻辑建模与应用[J]. 自动化应用, 2011, 11(3): 23-27

[17] Gao Jin-yuan, Jiao Zong-xia, Zhang Ping. The Plane Telex Control System Active Control Technology[M]. Beijing: Beihang University Press, 2005: 34-65(in Chinese)
高金源, 焦宗夏, 张平. 飞机电传操纵系统主动控制技术[M]. 北京: 北京航空航天大学出版社, 2005: 34-65

[18] Wang Yong, Liang De-fang. Civilian Aircraft Fly-By-Wire Flight Control System [J]. Aeronautic Standardization & Quality, 2008, 227(5): 24-28(in Chinese)
王永, 梁德芳. 民用飞机电传飞行控制系统初探[J]. 航空标准化与质量, 2008, 227(5): 24-28

(上接第 178 页)

[2] Goyal V, Pandey O, Sahai A, et al. Attribute-based encryption for fine-grained access control of encrypted data[C]// Proceeding of the 13th ACM Conference on Computer and Communications Security (ACM CSS 2006). New York: ACM New York, 2006: 89-98

[3] Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption[C]// IEEE Symposium on Security and Privacy 2007 (SP 2007). Piscataway: IEEE, 2006: 321-334

[4] Cheung L, Newport C. Provably secure ciphertext policy ABE [C]// 14th ACM Conference on Computer and Communications Security (ACM CSS 2007). New York: ACM, 2007: 456-465

[5] Lbraimi L, Tang Q, Hartel P, et al. Efficient and provably secure ciphertext-policy attribute-based encryption schemes [C]// Information Security Practice and Experience. Berlin: Springer Berlin Heidelberg, 2009: 1-12

[6] Kapadia A, Tsang P P, Smith S W. Attribute-based publishing with hidden credentials and hidden policies [C]// Proceedings USA of 14th Annual Network & Distributed System Security Symposium (NDSS 2007). San Diego, California, USA: NDSS, 2007: 179-192

[7] Boneh D, Waters B. Conjunctive, subset, and range queries on encrypted data[C]// 4th Theory of Cryptography Conference. Berlin: Springer Berlin Heidelberg, 2007: 535-554

[8] Nishide T, Yoneyama K, Ohta K. Attribute-based encryption with partially hidden encryptor-specified access structures[C]// 6th International Conference on Applied Cryptography and Network Security. Berlin: Springer Berlin Heidelberg, 2008: 111-129

[9] Müller S, Katzenbeisser S. Hiding the policy in cryptographic access control [M]// Security and Trust Management. Berlin: Springer Berlin Heidelberg, 2012: 90-105

[10] Xu R, Wang Y, Lang B. A Tree-Based CP-ABE Scheme with Hidden Policy Supporting Secure Data Sharing in Cloud Computing[C]// 2013 International Conference on Advanced Cloud and Big Data (CBD). Piscataway: IEEE, 2013: 51-57

[11] Lai J Z, Deng R H, Li Y J. Fully secure ciphertext-policy hiding CP-ABE[C]// 7th International Conference on Information Security Practice and Experience. Berlin: Springer Berlin Heidelberg, 2011: 24-39

[12] Waters B. Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions[C]// Advances in Cryptology-CRYPTO 2009(009). Berlin: Springer Berlin Heidelberg, 2011: 619-636

[13] Emura K, Miyaji A, Nomura A, et al. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length[C]// 5th International Conference on Information Security Practice and Experience. Berlin: Springer Berlin Heidelberg, 2009: 13-23

[14] Zhou Z B, Huang D J. On efficient ciphertext-policy attribute-based encryption and broadcast encryption extended abstract [C]// Proceedings of the 17th ACM Conference on Computer and Communications Security. New York: ACM, 2010: 753-755

[15] Rao Y S, Dutta R. Recipient anonymous ciphertext-policy attribute based encryption[C]// 9th Internatonal Conference on Information Systems Security. Berlin: Springer Berlin Heidelberg, 2013: 329-344

[16] Zhang M, Wang X A, Yang X, et al. Efficient Predicate Encryption Supporting Construction of Fine-Grained Searchable Encryption[C]// 2013 5th International Conference on Intelligent Networking and Collaborative Systems (INCoS). Piscataway: IEEE, 2013: 438-442

[17] Padhya M, Jinwala D. A Novel Approach for Searchable CP-ABE with Hidden Ciphertext-Policy [C]// 10th International Conference on Information Systems Security. Berlin: Springer Berlin Heidelberg, 2014: 167-184