基于MDA的实时软件资源建模与模型转换的方法

吉 鸣 黄志球 祝 义 王珊珊 沈国华
(南京航空航天大学信息科学与技术学院 南京 210016)

摘 要 模型驱动体系结构(MDA)是一种以模型为中心的软件开发框架,其本质是元建模与模型转换。提出了一种基于MDA的实时软件资源建模与模型转换的方法。首先通过元建模抽象出包含资源信息的MARTE元模型以及价格时间自动机的元模型;然后利用模型转换语言ATL对MARTE元模型和价格时间自动机元模型构造转换规则,通过将对应的实例模型进行相互转换,实现在MDA下MARTE模型到价格时间自动机模型的转换;最后通过形式化工具UPPAAL对模型转换结果进行形式化验证。实例分析表明了该方法的可行性与有效性,它能够提高实时软件资源建模的可信性。

关键词 模型驱动体系结构,元建模,MARTE,模型转换,价格时间自动机

MDA-based Method on Resource Modeling and Model Transformation of Real-time Software

JI Ming HUANG Zhi-qiu ZHU Yi WANG Shan-shan SHEN Guo-hua
(Department of Information Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract Model Driven Architecture (MDA) is a model-centric software development framework. Its nature is meta modeling and model transformation. In this paper, a MDA-based method on resource modeling and model transformation of real-time software was proposed. This method first abstracts MARTE meta-model which contains some resource information and Priced Timed Automata (PTA) meta-model through meta-modeling. Secondly it uses ATL model transformation language to transform instance models from MARTE model to PTA model. The transformation is to construct transformation rules for MARTE meta-model and PTA meta-model. Finally, the result is formal verified through formal tool UPPAAL. The case shows the feasibility and effectiveness of the method which can improve the reliability of resource modeling of real-time software.

Keywords MDA, Meta modeling, MARTE, Model transformation, Priced timed automata

1 引言

随着航天、通信、军事等行业的迅速发展,嵌入式实时系统在当今社会得到越来越广泛的应用。与传统的非实时系统相比,嵌入式实时系统有更强的时间约束,因而对系统的一系列非功能属性如可靠性、可维护性有着更高的要求[1]。如何高效、低成本地开发高质量的软件产品,一直是计算机软件领域重点关注和研究的问题。模型驱动体系结构(Model-driven Engineering Architecture, MDA)[2]这一概念正是在这样的背景下逐渐受到重视。它的核心思想是一种以模型为中心的软件开发指南,其中以UML(Unified Modeling Language)[3]建模应用最为广泛。

UML是一种软件建模语言,它强调建模的灵活性和实用性,支持对系统静态和动态的多个侧面进行建模。但是它在实时系统建模过程中存在一定的局限,缺乏对系统非功能属性(如时间、资源等)方面的描述。时间是对系统动态行为的建模,而资源则表示系统执行的外部环境,它是对系统活动的执行环境进行建模。根据资源是否能够多次存在,可将其分为两类:可重用资源和不可重用资源。其中可重用资源是指被占用的任务对象释放后可重复使用的资源,如系统内存、设备等;不可重用资源是指被占用的任务对象释放后不可重复使用的资源,如能耗、能量使用等。MARTE(Modeling and Analysis of Real-Time Embedded systems)[4]是OMG组织2007年发布的建模实时和嵌入式系统的正式规范,它是UML的profile[5],弥补了UML在非功能属性建模上存在的不足(缺乏时间和代数表达式的语法,不能对时间、资源等非功能属性建模),已得到Papyrus[6]建模工具的支持。在MARTE中,也将资源建模分为两类:"硬件"和"软件"建模,分别对应资源中的可重用资源和不可重用资源。本文旨在对系统的不可重用资源系统能耗进行建模。

无论是UML还是MARTE,都缺乏精确的语义,因而不能对其进行形式化的分析和验证,难以满足实时系统高可靠

到稿日期:2010-09-06 返修日期:2010-12-03 本文受国家高技术研究发展计划(863计划)(2009AA010307),南京航空航天大学基本科研业务费专项科研项目(NS2010108)资助。

吉 鸣(1986—),男,硕士,主要研究方向为软件工程、形式化方法,E-mali:jimingonly@126.com;黄志球(1965—),男,博士,教授,主要研究方向为软件工程、知识工程;祝 义(1976—),男,博士,讲师,主要研究方向为软件工程、形式化方法;王珊珊(1963—),女,硕士,副教授,主要研究方向为软件工程。

性需求。形式化方法(formal method)[7]是一种为软件和硬件系统提供精确规约、设计和验证的严格的数学技术和工具，可以用在软件生命周期各阶段，如自动机(Automata)、进程代数(Process Algebra)、Petri 网等。将系统的 UML 模型转换为形式化模型进行验证的方法，是软件工程领域研究的热点。Vieri Del Bianco 等在扩展了时间和约束的 UML 基础上，提出从 UML 状态图到时间自动机的映射，并通过工具对其进行验证[8]；Simona Bernardi 等人则将 UML 的时序图和状态机图转换成 Petri 网模型并进行验证[9]。

传统的模型转换多是 Ad-hoc[10]式，因为是基于模型层的转换，所以转换关系和转换规则难以重用，而基于元模型[16]的模型转换有效地解决了以上问题。ATL(ATLAS Transformation Language)[11,16]是 ATLAS(INRIA&LINA)提出的 MDA 下的模型转换语言，它是一种指定元模型和文本具体语法的模型转换语言，通过定义元模型层的转换规则实现模型之间的转换，目前已在 AMMA[12]平台下实现。然而，现有的基于 MDA 的模型转换大多是针对系统的功能进行建模，对于系统的非功能属性考虑得相对较少。本文在 MDA 下研究基于元模型的模型转换，通过引用 MARTE 资源元模型以及建立价格时间自动机(Priced Timed Automata，PTA)[13,17]元模型，并定义 UML 元模型到价格时间自动机元模型的 ATL 转换规则，来实现 UML 模型到价格时间自动机模型之间的转换，最后利用 UPPAAL CORA[14]对转换结果进行形式化验证。

本文第 2 节简要介绍基于 MDA 的模型转换框架，在 MDA 的基础上提出 MARTE、价格时间自动机的具体概念，然后说明 ATL 模型转换框架；第 3 节则介绍 MARTE 资源元模型和建立价格时间自动机元模型，并在定义的元模型层上进行 ATL 转换规则映射；第 4 节通过具体的飞机着陆问题进行建模，并对模型转换结果进行验证；最后是总结。

2 基于 MDA 的模型转换框架

2.1 模型驱动体系结构

模型驱动体系结构(MDA)是对象管理组织(Object Management Group，OMG)[15]提出的以模型为中心的软件开发框架性标准。它把软件开发过程转变成在一组具有不同视角或抽象层次的模型之间的转换和精化过程，从而大大提高了软件的质量和可靠性，因而受到了学术界和工业界的广泛认可。MDA 与传统的软件开发生命周期类似，也可以分为需求分析、设计、实现、测试和发布几个阶段，每个阶段都是一个迭代的过程。不同的是，模型在软件开发过程中体现了最重要的作用，而模型转换是实现软件开发的关键。MDA 软件开发过程中存在两大类型的模型转换：纵向转换和横向转换[2]。纵向模型转换是指相邻的不同抽象层次模型之间的转换，也就是所谓的元模型到模型、模型到代码的转换。其中，元模型层的主要职责是为了描述模型层而定义的一种“抽象语言”，是对模型层的进一步抽象；而模型则是对代码的抽象。横向模型转换是指同一抽象层次的模型之间的转换，主要用于异构模型之间。虽然目前横向模型转换的相关研究不多，但是异构模型转换如本文的 MARTE 到价格时间自动机的模型转换旨在提高软件的可靠性和质量，将非形式化的模型转换成形式化的模型加以验证，因而意义重大。

2.2 MARTE

MARTE 作为 UML 的 profile，是对嵌入式实时系统的建模与分析规范，它弥补了 UML 在非功能属性上建模的不足。从模型层次上看，和 UML 一样，MARTE 属于元模型层。MARTE 中的概念主要分为基础(foundation)、建模(modeling)和分析(analyzing)3 个部分。其中基础模型包括实时和嵌入式系统中的一些基本概念，如时间、资源等和一系列核心元素，它是整个 MARTE 规范的核心。设计模型提供较高抽象层的建模元素，用于对并发和实时的活动主体和行为建模，如响应并发和实时调用的服务和动作等。分析模型封装了对系统性能质量以及调度的建模元素，主要针对非功能属性的具体建模，如能耗、能量等。

2.3 价格时间自动机

价格时间自动机(PTA)是在自动机的基础上扩展了时间和代价的元素，它是对系统进行建模的形式化描述方法。一个价格时间自动机是一个七元组(L, l_0, C, A, E, I, P)。其中，L 表示节点集合；$l_0 \in L$ 表示初始节点；C 表示时钟集；A 表示一组动作的集合；$E \subseteq L \times A \times B(C) \times 2^C \times L$ 表示在两个节点之间的边的集合，它包含 3 个元素：一个动作(a)、一个卫士条件(guard)和一个时钟重置集(r)；$I: L \to B(C)$ 表示一组节点不变量(invariant)，是对状态的约束；$P: L \cup E \to N_0$ 表示边上或节点上的价格。对于复杂的系统，一个 PTA 不能表示，因而就出现了时间自动机网(NPTA)，它是 PTA 的集合，其定义就是 PTA 的组合。本文采用 UPPAAL CORA 作为 PTA 的验证工具。

2.4 ATL 模型转换框架

ATL 是 ATLAS(INRIA&LINA)提出的 MDA 下的模型转换语言，依托于 AMMA 平台。AMMA 提供了一系列的工具如 KM3 (Kernel MetaMetaModel)[13]、TCS(Textual Concrete Syntax)[13] 以及 ATL (ATLAS Transformation Language)[12]来解决 MDA 下的建模问题。其中，KM3 是一种轻量级的文本元模型定义语言，它有清晰的语义，为异构模型提供共同的元元模型的根；TCS 则定义了模型到文本的转换桥；而 ATL 是基于元模型的模型转换语言，它声明源模型和目标模型之间的元素，以命令式的指令表达元素之间的映射关系。图 1 描述了 ATL 模型转换框架。M1、M2、M3 分别表示模型层、元模型层和元元模型层。源模型(Source Model)遵从其元模型 MMs 的定义，同理目标模型(Target Model)遵从其元模型 MMt 的定义，而 ATL 则是在元模型层定义转换规则，将 MMs 的元素映射到 MMt 上，从而实现低层模型的自动转换。其本身就是一个元模型，遵从 ATL 的 Transform Mode 则是 ATL 的具体实例。而所有的元模型都由共同的元元模型 KM3 作为根。

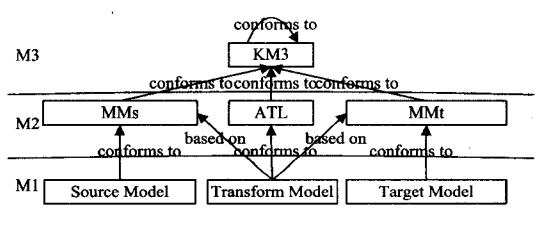


图 1 ATL 模型转换框架

3 元建模及模型转换

第 2 节介绍了基于 MDA 的模型转换框架，同时简要说明了 MARTE、价格时间自动机以及 ATL 模型转换的基本概念。结合本文的具体工作，这一节将在元模型层上给出具体

的 MARTE 资源元建模、价格时间自动机元建模以及 ATL 元模型之间的映射规则。

3.1 MARTE 资源元建模

MARTE 相对于 UML 的最大优势就在于它可以对 UML 所不能描述的非功能属性如时间、资源进行建模。下面对资源相关元素和系统行为相关元素进行元建模。

3.1.1 资源元素元模型

MARTE 一般资源建模主要封装在 GRM 包中，它定义了资源元素、资源服务、资源使用等资源相关的元模型，并且定义了它们之间的关联关系。本文主要通过 MARTE 资源元建模，提取出与资源使用相关的元模型，从而在模型层对系统能耗(Energy)进行建模，这是 UML 所不能描述的。图 2 给出了 MARTE 资源元素的元模型。其中，Resource 表示一个资源元素，它有一个或多个 ResourceService(资源服务)，每个 ResourceService 都能被泛化为 Acquire、Release 等服务。而一个 Resource 可以实例化 0 个或多个 ResourceInstance(资源实例)。Resource 也定义了时钟元素 Clock。ResourceUsage 表示资源使用，它可以使用 0 个或多个资源，而 UsageTypeAmount 则表示某类资源使用的数量，其中 energy 定义了能耗，它的类型为 NFP_Energy，在实时系统中，它表示单位时间内的能耗量，而总能耗量则是 energy 和 Clock 实例的乘积(总能耗=单位时间能耗 * 时间)。

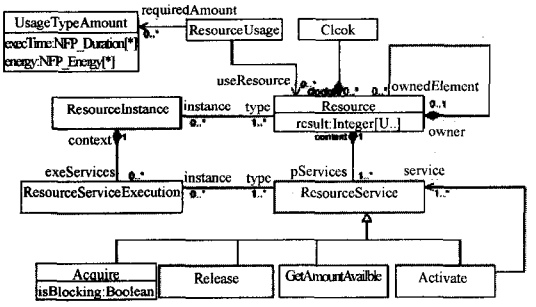


图 2 MARTE 资源元素元模型

3.1.2 行为元素元模型

MARTE 中行为建模主要封装在高层应用建模包(High-Level Application Modeling，HLAM)中。图 3 简要设计了表示系统活动行为的模型。其中，RtUnit 和 PpUnit 表示系统的活动对象(active object)。RtUnit(real-time unit)可看作一个自动执行的资源，同时处理不同的消息，并且拥有并发和行为控制器来根据当前状态和消息的当前执行约束管理消息约束。RtUnit 分两种，由 isMain 属性标识，一个应用程序必须有一个主 RtUnit(isMain = true)作为主控单元。PpUnit(protected passive unit)表示 RtUnit 之间的共享信息，它定义并发策略并由 RtUnit 调用。RtAction 和 RtBehavior 分别表示原子行为和组合行为，RtService 代表可调度资源的行为，RealTimeFeature 描述行为的实时约束。

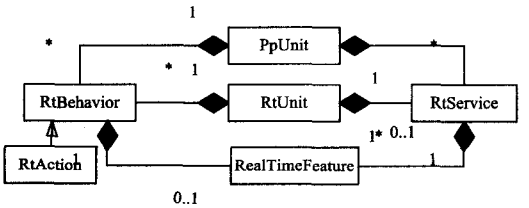


图 3 行为元素元模型

3.2 价格时间自动机元建模

价格时间自动机作为软件形式化建模语言，有着丰富的语义定义，用数学符号描述。如今有很多对实时系统的自动机进行建模和验证的工具，其中最著名的是 UPPAAL，它对时间自动机进行建模和验证，进而推出的 UPPAAL CORA 则是对 UPPAAL 的扩展，主要是对价格时间自动机进行建模和验证。UPPAAL CORA 在价格时间自动机定义的基础上，扩展了它的共享整型变量、紧急通道等。然而价格时间自动机只有形式化定义，要对其进行元建模，最终实施模型转换，就必须将其放在和 MARTE 相同的环境下，实现 MARTE 和价格时间自动机的同构化。图 4 定义了价格时间自动机的元模型，它是与 MARTE 元模型同构的，都遵从于 UML 的 KM3 元元模型。

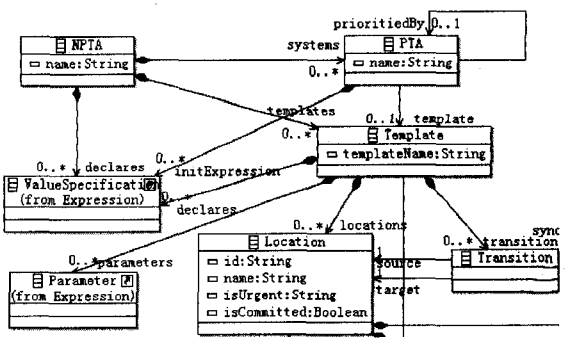


图 4 价格时间自动机元模型

价格时间自动机(PTA)组成了价格时间自动机网(NPTA)，它关联 0 个或 1 个模板(Template)，所有的 PTA 都是由 Template 表示的。一个模板包含若干节点(Location)、变迁(Transation)和参数(Parameter)，Location 用 id 作为唯一标识，用 isUrgent 和 isCommitted 注明它的类型：普通节点、紧急节点或委托节点，Initial 表示初始节点，是 Location 的泛化。Transition 表示边，用 source 和 target 指向源和目标节点，它包含 4 种表达式，分别是选择表达式(SelExpression)、同步表达式(SynExpression)、布尔表达式(BoolExpression)和赋值表达式(AssignExpression)。在表达式中，SynExpression 表示具体的同步操作，是触发变迁的动作，定义为通道(Channel)类型；BoolExpression 既可表示 Location 的 invariant(状态的时钟约束)，也可表示 Transition 的 guard(变迁的触发条件)；AssignExpression 既可以表示边上的代价，对应于变迁瞬间完成的系统能耗，也可以表示节点上的代价，对应于处在某一状态时间内单位时间的能耗率；Parameter 指明了自动机模板定义的参数。

3.3 ATL 映射规则

ATL 模型转换是基于图 1 中的模型转换框架进行的，通过 ATL 模型转换语言，在元模型层定义映射规则，从而在模型层实现自动的转换。本文用 MARTE 对系统资源(如能耗)建模，同时要表示实时系统的动态行为，因而除了将 MARTE 相关元素映射成为 PTA 之外，还将用 UML 描述的交互图、状态机图和活动图分别映射成为 PTA。

3.3.1 MARTE 相关元素映射

表 1 给出了 MARTE 相关元素如时钟、资源(能耗)等元素到价格时间自动机元素的映射关系，相关的映射元素在图 2、图 3 和图 4 中给出。

表 1　MARTE 相关元素到价格时间自动机的映射

MARTE	价格时间自动机	解释
Resource	Template	Resource 表示实时的活动对象，它不能自动执行，映射成为一个 Template
Resource-Instance	PTA	ResourceInstance 表示 Resource 的实例化，映射成为一个 PTA
ResourceS-ervice	SynExpression	ResourceService 表示 Resource 提供的服务，可以是同步信道，映射为 SynExpression
Resource Usage	Expression	ResourceUsage 表示对一个资源使用的请求，映射成一个 Expression
UsageType Amount	AssignExpres-sion	UsageTypeAmount 中的 energy 表示能耗，映射成为 PTA 中代价的 AssignExpression
Clock	Clock	MARTE 的时钟元素映射为 PTA 的时钟元素
RtUnit	Template	RtUnit 表示一个活动对象，是一个可自动执行的资源，映射为一个 Template
RtService	SynExpression	RtService 表示可调度资源 RtUnit 的行为，映射为 SynExpression
TimedCon-straint	BoolExpression	TimedConstraint 表示 MARTE 对于实时系统的时间约束，映射为 BoolExpression

3.3.2　交互图映射

UML 中的交互视图描述了执行系统功能的各个角色之间相互传递消息的顺序关系，它包含顺序图和协作图。前者侧重于时间顺序，而后者侧重于角色之间的关系。本文旨在抽取实时系统的动态行为，更偏重于时间顺序，因此采用顺序图描述系统角色的交互。图 5 表示 UML 交互图元模型，具体映射关系见表 2。

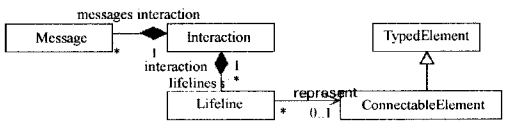


图 5　UML 交互图元模型

表 2　UML 交互图到价格时间自动机的映射

交互图	价格时间自动机	解释
Interaction	NPTA	Interaction 表示系统的交互场景，是各个相互交互的角色的集合，映射为 NPTA
Lifeline	PTA	Lifeline 表示参与活动的角色，每个角色都可以向另一角色发送消息，映射为 PTA
Ty-pedElement	Template	TypedElement 表示具体定义的类型，它是活动图或状态机图代表的类型，映射为 Template
Message	SynExpression	Message 表示活动角色发出的消息，映射为 SynExpression

3.3.3　状态机图映射

状态机视图通过对类对象的生存周期建立模型来描述对象随时间变化的动态行为，它是表示状态和状态之间转换的图，可以对并发行为进行建模。图 6 给出了 UML 状态机图元模型，具体映射关系见表 3。

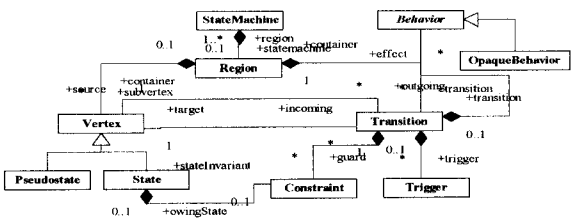


图 6　UML 状态机图元模型

表 3　UML 状态机图到价格时间自动机的映射

状态机图	价格时间自动机	解释
State-Machine	Template	StateMachine 表示状态机，它含有 1 个或多个 Region，映射为 Template
Pseudostate	Initial	Pseudostate 表示状态机的初始状态，映射为 Initial
State	Location	State 表示状态机的状态，映射为 Location
Transition	Transition	Transition 表示引起状态机状态变化的边，映射为 Transition
Constaint	Expression	Constaint 表示状态或者变迁的约束条件，映射为 BoolExpression（Location 的 invariant 或 Transition 的 guard）或 AssignExpression（Location 的代价率或 Transition 的代价）
Trigger	SynExpression	Trigger 表示触发变迁的动作，映射为 Syn-Expression
Opaque-Be-havior	AssignExpres-sion	OpaqueBehavior 表示变迁触发后引起的动作，映射为 AssignExpression，如时钟置零

3.3.4　活动图映射

活动图是一种特殊形式的状态机，用于对计算流程和工作流程建模，它描述了一组顺序的或并发的活动。活动图包含活动状态，表示过程中命令的执行或工作流程中活动的进行。图 7 给出了 UML 活动图元模型，具体映射关系见表 4。

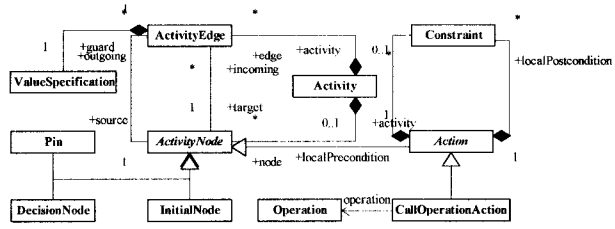


图 7　UML 活动图元模型

表 4　UML 活动图到价格时间自动机的映射

活动图	价格时间自动机	解释
Activity	Template	Activity 表示一组活动状态，映射为 Template
InitialNode	Initial	InitialNode 表示活动图的初始节点，映射为 Initial
Decision-Node	Location	DecisionNode 表示活动图的节点，映射为 Location
Action	Transition	Action 表示活动图的活动状态，映射为 Transition
Activi-tyEdge	Expression	ActivityEdge 表示节点的约束条件，可以表示节点的能耗率，映射为 Expression（Location 的 invariant 或 Location 的 cost）
Constraint	Expression	Constraint 表示 Action 的前置条件和后置条件，也可表示 Action 的能耗，映射为 Expression（Transation 的 guard、Transation 后的动作，如时钟置零或 Transation 的 cost）
CallOpera-tionAction	SynExpression	CallOperationAction 表示触发变迁的动作，映射为 SynExpression。

4　用例研究

在第 3 节中给出了 MARTE 资源相关元素元建模、价格时间自动机元建模以及根据元模型的 ATL 映射规则，本节将结合具体的“飞机着陆”问题，说明如何通过 ATL 将 MARTE 模型转换到价格时间自动机模型并验证其正确性。

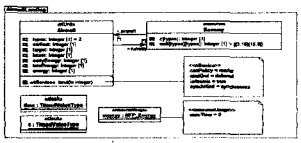
4.1　问题描述

飞机着陆问题（Aircraft Landing）被提出来是为了解决飞机间的调度问题，以达到能耗使用最小的目的。问题描述是：给定一组不同类型的飞机（Aircraft），要求其在有限的跑道（Runway）着陆。每架飞机有目标登陆时间（target）、最早时

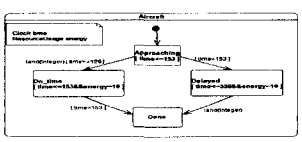
间(earliest)和最晚时间(latest)。当飞机提前着陆时,它必须加速因而使用更多的燃料;当飞机晚点着陆时,它又必须使用燃料来维持在空中的徘徊等待。所有的额外消耗就是能耗。而由于飞机尾流原因,不同类型的飞机必须有时间限制,让同一跑道上后来的飞机等待,不同类型的飞机的等待时间不同。

4.2 MARTE 建模

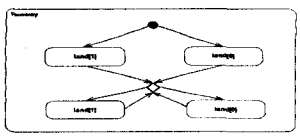
针对飞机着陆问题,结合 3.1 节的 MARTE 资源元素和行为元素元模型,对飞机着陆问题进行 MARTE 建模。图 8 给出了用 Papyrus 建模工具描述的飞机着陆问题的模型。其中(a)为 MARTE 元素的类图,它定义了系统所需要的类和类之间的关系。系统中有两类对象:Aircraft 和 Runway,它们分别是 MARTE 的 RtUnit 对象和 Resource 对象,并发处理不同的消息。在 Aircraft 中,types 是 Aircraft 的属性,表示飞机的类型,默认为两种。earliest、target、latest 分别表示最早、目标和最晚到达时间,earlyEnergy 和 lateEnergy 分别表示早到和延迟的能耗。Aircraft 有一个操作 land(Integer),它是一个 RtService 对象,表示对于不同类型的飞机的着陆请求,其中参数为 Integer 类型,表示飞机类型,〈〈rtService〉〉标注了一些动态属性初始化信息。在 Runway 中,c[types]数组表示不同类型的飞机使用跑道的时钟变量,wait[type][type]则表示不同类型的飞机互相等待的时间。在类图中,有两个 Clock 类型变量,分别为 time 和 c,time 为飞机的到达时钟,c 为飞机使用跑道的时钟。还有一个 resourceUsage 类型变量 energy,它表示飞机的能耗,使用完在等待时间内释放。(b)、(c)、(d)分别是简化的 Aircraft 的 UML 状态机图建模、Runway 的活动图建模以及系统的顺序图建模。



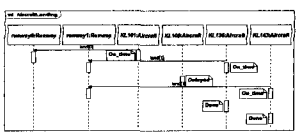
(a)MARTE 元素类图



(b)UML 状态机图



(c)UML 活动图



(d)UML 顺序图

图 8 飞机着陆 MARTE 模型

4.3 模型转换

ATL 是 Eclipse 环境下的一个插件,转换过程需要 4 个文件:(1) MARTE. profile. uml,它是 UML 的 profile 文件,本文仅使用了 3.1 节介绍的资源、行为元模型对输入模型建模;(2) NPTA. ecore,它是价格时间自动机网的元模型文件,在 3.2 节给出;(3) MARTE2NPTA. atl,它是用 ATL 语言描述的 ATL 元模型层的转换规则文件,具体映射规则见 3.3 节;(4) AircraftLanding_MARTE. xmi,是 MARTE 飞机着陆问题的资源模型文件,利用 papyrus 自动生成 xmi 文件,格式是 xmi(也可是 uml),具体模型在 4.2 节给出。图 9 显示了 ATL 模型转换的配置图,左边是工程文件目录,包括上述 4 个文件。图中,元模型 UML2 和 NPTA 分别是 uri:http://www. eclipse. org/uml2/3. 0. 0/UML(Eclipse 下的 UML2 模型库)和 NPTA. ecore,而输入模型 AircraftLanding_MARTE. xmi 遵从于 UML2 元模型,并且使用 MARTE profile,输出模型 AircraftLanding_PTA. xmi 则遵从于 NPTA 元模型。

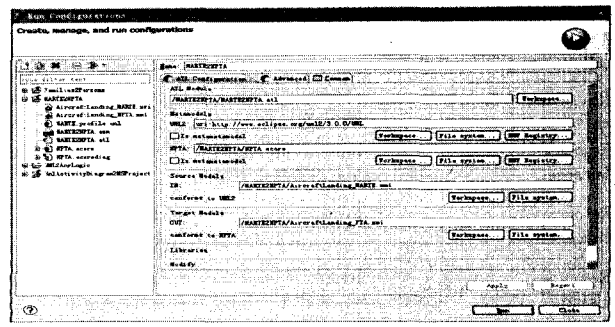


图 9 ATL 模型转换配置

转换的结果也为 xmi 文件,图 10 分别给出顺序图、状态机图和活动图到价格时间自动机的转换结果。在图 10 中,遵循元模型的转换规则,ATL 转换程序将 MARTE 模型自动转换成为 PTA 模型。根据映射规则,各个元素的映射结果显而易见。对于 UML 的顺序图,通过 ATL 映射规则转换成 NPTA,它是一系列 PTA 的集合,分别用 name 标识,如图中的 KL101 表示的是某架飞机的名字,而 runway0 则表示一条跑道;对于状态机图,将 Aircraft 建模的 UML 状态机图转换成 PTA 的 template,如状态机图中的 energy 就被转换成为 PTA 中的 cost,状态变迁条件被转换成 PTA 中的 guard 等等;对于活动图,将 Runway 建模的 UML 活动图转换成为 PTA 的 template,如活动图中的活动状态 land 被转换成为 PTA 中的同步表达式 land(int type)。

```
1、<NPTA name=" AircraftLanding ">
2、<system xsi:type=" PTA"
name= "KL101 " type="//@template.0"/>
3、<system xsi:type=" PTA"
name= "runway0 "
type="//@template.1"/>
4、...
5、<template
name=" Aircraft" >...</template>
6、<template
name=" Runway">...</template>
7 、 <declare xmi:type=" Comment"
content=" const int types = 2;int
c[types];int wait[types][types]"/>
8、...
9、<NPTA>
```

```
1、<templates name="Aircraft">
2、<declare name="time"
datatype="//@datatype.0">...
3、< initial id="id0"
name="Approaching"/>
4、<locations id="id1" name="On_time">
5、<cost expression=cost"=10">
6、<invariant expression="time<=153">
7、</locations>
8、<transitions source="//@locations.0"
target="//@locations.1" >
9、<synchronization expression="land(int
type)" isIn="false"/>
10、<guard expression="time>=129"/>
11、</transitions>
13、</template>
```

```
1、<template name="Runway">
2、<declare xmi:type="Comment"
content=" int c[types];int
wait[types][types]">
3、< initial id="id0" name="Init"/>
4、<locations id="id1"
name="IdelAndInUse"/>
5、<transitions source="//@locations.0"
target="//@locations.1" >
6、<synchronization
expression="land(int type)"
isIn="true"/>
7、<action expression="c[1]=0"/>
8、</transitions>
9、...
10、</template>
```

图 10 ATL 模型转换结果

4.4 形式化验证

本文的目的是将非形式化的 UML 模型转换成为形式化的 PTA 模型并加以验证。4.3 节通过 ATL 模型转换得到了飞机着陆问题的 PTA 模型,现在将生成的 PTA 模型的 xmi 文件转换成 xml 文件作为模型验证工具 UPPAAL 的输入,验证其正确性。图 11 给出了 UPPAAL 输入 AircraftLanding 的 PTA 模型。UPAAL 采用 CTL 表示系统形式化规约,本文只验证最终 Done 状态的可达性。通过对生成的 PTA 模型进行形式化验证,表明系统模型满足形式化规约。

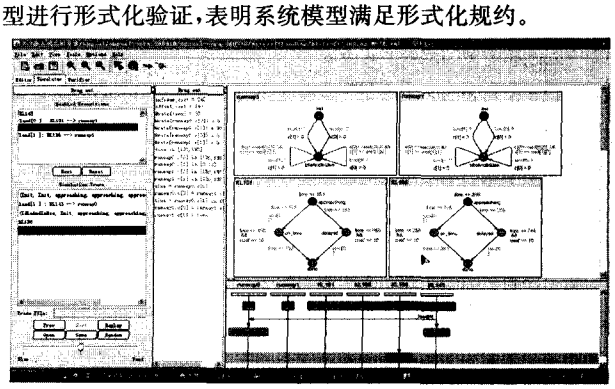


图 11 UPPAAL AircraftLanding PTA 模型

结束语 本文在 MDA 的背景下,对实时系统的资源属

性进行 MARTE 元建模，再利用 ATL 转换工具在元模型层将 MARTE 模型转换为价格时间自动机模型，实质上这是一种基于元模型的模型转换方法。进而通过具体案例进行模型转换，对转换后的结果使用 UPPAAL 工具验证其正确性和可靠性。然而，在嵌入式实时系统中，会涉及硬件的相关信息，MARTE 提供专门的包来对系统硬件如内存、设备等可重用资源建模，而在加入代价后，PTA 也提供了很多算法来解决最小代价问题，这都是本文下一步要研究的内容。

参 考 文 献

[1] 王涛，张伟良，葛宁，等. 嵌入式实时系统及其在通信系统中的应用[J]. 电子应用技术，2001，27(11)

[2] 张天，张岩，于笑丰，等. 基于 MDA 的设计模式建模与模型转换[J]. 软件学报，2008，19 (09)：2203-2217

[3] OMG. Unified modeling language: Superstructure v2.0 [EB/OL]. http://www.omg.org/docs/formal/05-07-04.pdf，2005

[4] OMG. UML Profile for MARTE, Beta 2 [EB/OL]. http://www.omg.org/cgi-bin/doc? ptc/2008-06-08，2008

[5] Fuentes-Fernández L, Vallecillo-Moreno A. An Introduction to UML Profiles[J]. The European Journal for the Informatics Professional，2004

[6] CEA LIST Team. Papyrus (open source tool for graphical UML2 modeling) [EB/OL]. http://www.papyrusuml.org/，2007

[7] Clarke E M，Wing J M. Formal methods: state of the art and future directions[J]. ACM Computing Surveys，1996，28(4)：626-643

[8] Del Bianco V，Lavazza L，Mauri M. Model checking UML specifications of real time software[C]// Proceedings of the 8th IEEE International Conference on Engineering of Complex Computer Systems. 2002：203-212

[9] Bernardi S，Donatelli S，Merseguer J. From UML sequence diagrams and statecharts to analysable petri net models[C]// Proceedings of the 3rd International Workshop on Software and Performance. 2002：35-45

[10] Czarnecki K，Helsen S. Classification of Model Transformation Approaches[J]. Generative Techniques in the Context of Model-driven Architecture，2003

[11] Jouault F，Allilaire F，Bézivina J，et al. ATL：A model transformation tool[J]. Science of Computer Programming，2008，72 (2)：31-39

[12] Jouault F，Bézivin J，Barbero M. Towards an advanced model-driven engineering toolbox[J]. Innovations in Systems and Software Engineering，2009，5(1)：5-12

[13] Behrmann G，Larsen K G，Rasmussen J I. Optimal Scheduling using Priced Timed Automata[J]. ACM Sigmetrics Performance Evaluation Review，2005，32(4)：34-40

[14] Behrmann G，David A，Larsen K G. A Tutorial on Uppaal[J]. Formal Methods for the Design of Real-Time Systems，2004

[15] Object Management Group. OMG[EB/OL]. http://www.omg.org/

[16] Jouault F，Kurtev I. Transforming Models with ATL [C]// Satellite Events at the MoDELS 2005 Conference. 2006

[17] Rasmussen J I，Larsen K G，Subramani K. Resource-optimal Scheduling Using Priced Timed Automata[J]. Tools and Algorithms for the Construction and Analysis of Systems，2004

[18] 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

(上接第 116 页)

常用码书，因此对码书的优化影响不大。

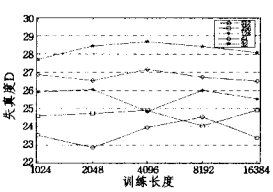


图 1 失真度与不同大小基础码书的关系

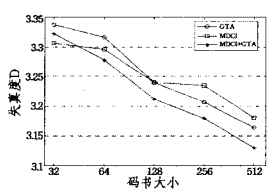


图 2 失真度与码书大小关系

5.2 平均谱失真

谱包络失真(Spectral Distortion，SD)是比较常用的一种客观评价量化器性能的标准。第 n 帧的 SD 表示为

$$SD_n = \frac{1}{\pi}\left\{\int_0^{\pi}\left[20\log_{10}\frac{|H(\omega)|}{|\hat{H}(\omega)|}\right]^2 d\omega\right\}^{1/2} \quad (dB) \qquad (7)$$

式中，$H(\omega)$和$\hat{H}(\omega)$分别为输入和输出语音的短时线性预测分析谱。

平均谱失真如下式所示。其中，M 为测试矢量的总数。

$$ASD=\frac{1}{M}\sum_{m=1}^{M} SD_m \qquad (8)$$

平均谱失真度与基础码书大小的关系如图 3 所示。虽然各种算法都没有满足语音质量的客观评价标准，但是从文献[6]中可知，这里谱包络失真值及谱泄露值对评价一种码书的好坏也有着直接的影响，其数值的大小仍能反映出码书的质量。

由图 3 可以看到，由所需码书的 32 倍大小的基础码书得到的最终码书，其平均谱失真也是最小或者次小的。图 4 中也可以看到，从平均谱失真的角度看，也是除 32 大小的码书外，采用新算法生成的平均谱失真均较小。

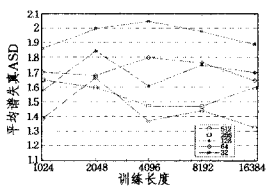


图 3 平均谱失真与不同大小基础码书的关系

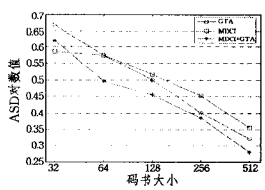


图 4 平均谱失真与码书大小关系

结束语 本文基于贪婪树及初始码书间距最大算法提出了一种新的算法，该算法结合了贪婪树算法计算量低的特点，并克服了初始码书间距最大化算法不具代表性的缺点。结果显示，用贪婪树算法生成的基础码书大小为最终所需码书大小的 32 倍时，所得码书的失真度及平均谱失真都是最低或次低的，性能相对于原算法均有所改进。

参 考 文 献

[1] 李碧，林土胜，廖亮. 初始码字间距最大化的矢量量化码书设计算法 [J]. 小型微型计算机系统，2009，30(4)：780-783

[2] Katsavounidis I，Jay Kuo C-C，Zhang Zhen. A New Initialization Technique for Generalized Lloyd Iterarion[J]. IEEE Signal Processing Letters，1994，1(10)：144-146

[3] 梁彦霞，杨家玮，李烨. 一种低速率多模 MBE 线性预测语音编解码器[J]. 西安电子科技大学学报，2010，37(3)：417-422

[4] Riskin E A，Gray R M. A Greedy Tree Growing Algorithm for the Design of Variable Rate Vector Quantizers[J]. IEEE Trans Signal Proc，1991，39(11)：2500-2507

[5] 李弼程，文超，平西建. 两个优于分裂法的初始码书设计方法[J]. 中国图象图形学报，2000，5(A)(1)：48-51

[6] 梁彦霞，杨家玮，李烨. 线谱频率参数的快速、低存储矢量量化[J]. 吉林大学学报：工学版，录用待发表