

分析模式在嵌入式系统建模中的应用研究

段盛^{1,2}, 彭蔓蔓², 李仁发², 李仲生²

(1 湘南学院计算机系, 郴州 423000; 2 湖南大学计算机与通信学院, 长沙 410012)



摘要: 面向对象建模语言 UML (Unified Modeling Language) 已广泛用于嵌入式系统建模, 但它在嵌入式实时系统建模时存在模型重用性不高、概念模型形式化复杂和状态图对时间约束方面的建模功能不强的问题, 针对这些问题, 提出了对象分析模式与实时规范模式相结合的新思想, 并给出了使用对象分析模式和实时描述模式进行嵌入式系统建模及模型分析、验证的方法, 然后介绍了基于 UML 的模式驱动的嵌入式系统建模及分析的实现步骤, 最后以一个实际应用进行了详细说明。

关键词: 模式; 嵌入式系统; 建模; UML; 实时系统; 模型验证

中图分类号: TP311.11

文献标识码: A

文章编号: 1004-731X (2007) 14-3346-04

Application Study of Analysis Patterns in Embedded system Modeling

DUAN Sheng^{1,2}, PENG Man-man², LI Ren-fa², LI Zhong-sheng²

(1. Department of computer and science, Xiang nan college, Chenzhou 423000, China;

2. School of Computer and Telecommunication, Hu nan university, Changsha 410012, China)

Abstract: Object oriented modeling language UML (Unified Modeling Language) is applied abroad for embedded system modeling. But UML exists problems that model does not reuse and concept model is formalization complicated and Statechart does not Model well to time constraint. Concerning these problems, a new idea was proposed that object analysis patterns combined with real-time specification patterns, and a method that used object analysis patterns and real-time specification patterns to model for embedded system to analyze and validate for the model was proposed. UML-based embedded system modeling of pattern-driven and analysis approaches were introduced. An application example using the above approach was illustrated.

Key words: patterns; embedded system; modeling; UML; real-time system; model verification

引言

随着嵌入式系统在各个领域的广泛应用, 嵌入式系统变得越来越复杂。因而, 嵌入式系统分析阶段的工作—建模及模型分析、验证也变得越来越重要。

目前对于复杂嵌入式系统建模一般使用统一建模语言 UML (Unified Modeling Language)^[1], UML 作为面向对象设计技术的代表, 在系统设计中已经获得了广泛的承认, 并在多个领域中有成功的应用。然而, UML 对嵌入式系统建模时存在两个主要不足: 一是 UML 不是形式化描述语言, 不能直接对其模型进行模拟验证; 二是 UML 状态图对时间约束的建模能力不强。另外, 目前的建模方法还普遍存在模型重用性不高的问题。

当前国内解决 UML 建模时两个问题的方法主要有: 用 UML 建立概念模型, 再将它转换为某形式化模型, 然后再模拟验证。比如石柯等^[2]提出的用 UML 建模、再用其开发的可执行 UML 扩充子集进行模拟验证; 对 UML 进行时间

扩展以描述时间信息, 如赖明志、尤晋元等^[3]提出了利用 UML 扩展机制, 构造时间自动化机, 对 UML 状态图进行时间扩展的方法。但是这些方法没有通用的支撑工具, 需要有开发的特殊工具进行形式化和模型模拟验证, 工作复杂, 难度大, 且没有解决模型重用性问题。

国外较早就有人提出了分析模式的概念, 如 Douglass 的 ROPES^[4], 能用于软件开发的建模, 但是重设计而轻分析; 近年来, 国外有人将分析模式引入到嵌入式系统的分析阶段, 如 S. Konrad 等提出的对象分析模式^[5], 能完整地描述嵌入式系统的模型及分析, 但是对时间约束信息描述能力不强; 文献[6-7]能很好地描述时间信息, 但不能完整地进行系统建模及模型分析、验证。

1 模式驱动的建模与分析思想概述

针对上述问题, 本文提出了一种使用对象分析模式与实时描述模式结合起来进行基于 UML 的嵌入式建模及模型分析、验证的方法 (称为模式方法), 其思想是: 构建模式模板, 具体建模时根据系统需求选取模板中适当的分析模式, 利用结构模式的类/对象图模板引导建立系统的 UML 类/对象图, 利用行为模式的状态图、序列图模板建立系统的 UML 状态图、序列图, 进而建立系统 UML 初始概念模型, 再用文献[8]的形式化方法和 3 章定义的语义规则进行形式化转换, 用工具 MINERVA^[9]进行错误自动分析, 得到可视化的含结构错误的结构图或行为错误的行为轨迹, 并容易地纠正检测到的错

收稿日期: 2006-12-25 修回日期: 2007-03-13

基金项目: 国家自然科学基金资助项目 (60673061)

作者简介: 段盛 (1964-), 男, 湖南隆回人, 副教授, 硕士生, 研究方向为嵌入式系统, 数据库; 彭蔓蔓 (1964-), 女, 湖南长沙人, 副教授, 博士生, 研究方向为低功耗技术; 李仁发 (1957-), 男, 湖南长沙人, 教授, 博士, 博导, 湖南大学计通院院长, 研究方向为嵌入式计算, 无线网络, 网络与数字媒体; 李仲生 (1967-), 男, 湖南邵阳人, 讲师, 博士生, 研究方向为信息处理, 视屏处理。

误;最后,使用模型检测器 SPIN^[10]对模型进行验证。

2 对象分析模式与实时描述模式

在这一章,将定义用于描述模式属性的语义规则和用于实时系统的三个时序逻辑映射,再建立模式模板,然后简介模式的分类及各个字段。

2.1 语义规则定义

参照 Dwyer 符号和 Gamma^[11]规则,定义如下用于描述分析对象模式的语义规则。

∧: 表“与”(and) 操作

∨: 表“或”(or) 操作

¬: 表“非”(not) 操作

→: 表“暗示、包含”(implication) 操作

U: 表“强直到”(Strong-Until) 操作

W: 表“弱直到”(Weak-Until) 操作

□: 表“总是”(always)

◇: 表“最终”(eventually)

2.2 模式模板

根据从嵌入式开发者得到的建模文档库及文献[5-6],对其进行分析和抽象,建立模式模板(见表1),模板中涵盖了所有建模所需要的通用性模式,每个嵌入式开发者建模时都可根据需要进行选择、扩充。表1中包含了7个对象分析模式与5个实时规范模式。

表1 嵌入式实时系统中的对象分析模式和实时规范模式

模式类别	模式名称	模式用途描述
1	结构	用户接口
2	结构	触发器-传感器
3	结构	控制器分解
4	行为	处理组件
5	行为	检测器-修正器
6	行为	故障处理器
7	行为	通信链接
8	时间持续	最小持续时间
9	时间持续	最大持续时间
10	时间周期	循环约束
11	实时次序	响应约束
12	实时次序	不变性约束

2.2.1 模式分类

根据模式的作用的和性质,将模式分为对象分析模式和实时模式,而对象分析模式又分为结构模式和行为模式两类:

结构模式: 捕获系统静态结构属性,用来识别对象,抽取对象公共属性成为类,捕获类之间的关系;

行为模式: 表示对象/类的动态行为,用来定义相互作用的对象的本质行为,建立对象相互关联机制。

实时模式: 用来捕获和表述实时系统中的时间约束信息。

2.2.2 模板中模式简介

模式名称 是为了表示模式,给模式取的名字,它简洁描述了模式的本质。

模式类别 指模式是结构还是行为模式,如处理组件是一个行为模式。

模式意图 模式要解决的问题的简要描述,如处理组件是对发生的故障进行响应。

模式动机 使用此模式的目的和目标,说明如何使用模式中的类/对象来解决问题的特定情景,用用例和用例图来描述。

时序逻辑映射 用于实时模式的字段,为了能在 UML 图中表达实时系统的时间约束信息及后面的形式化描述的需要,使用了三个时序逻辑 MTL (Metric temporal logic)、TCTL(Timed computational tree logic)、RTGIL(Real-time graphical interval logic),MTL 和 TCTL 分别描述离散和连续时序逻辑,RTGIL 是时序逻辑的图示表示法。定义从模式到三个时序逻辑的映射如图1:

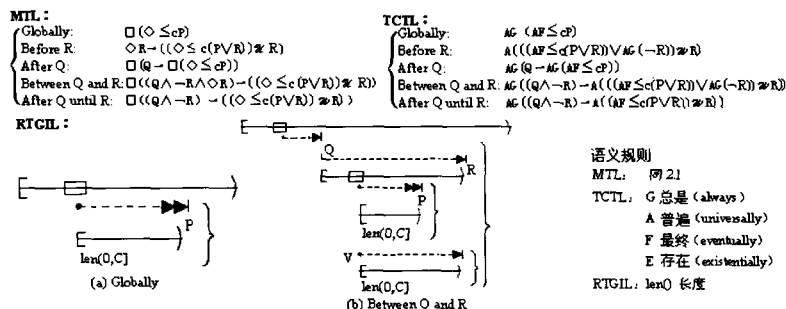


图1 时序逻辑映射

图 1 映射规则对五种可能的时间范围: 全局 (Globally)、在 R 之前 (Before R)、在 Q 之后 (After Q)、在 Q 与 R 之间 (Between Q and R)、在 Q 之后直到 R 发生 (After Q until R) 定义了映射, 可以根据需要任意选择一种时序逻辑映射规则, 对 RTGIL 只图示了 Globally 和 Between Q and R 两个范围。

结构 用 UML 类图来表示类及类之间的联系, 用户接口模式、触发器-传感器模式、控制器分解模式属于结构模式。

行为 用 UML 状态图和次序图来表示模式某一特定时刻类/对象交互及动态行为, 处理组件模式、检测器-修正器模式、故障处理器模式、通信链接模式属于行为模式。

参与者 描述模式中的类/对象以及它们各自的职责。

协作 描述参与者如何通过类/对象交互来实现各自的职责。

约束 描述在模式被应用后必须满足的属性, 包含操作约束和实时时间约束。

3 基于对象分析模式和实时描述模式的模型构造与检测

这一章, 将结合应用实例, 详细介绍如何使用模式方法进行嵌入式系统建模及模型分析、验证。

3.1 电子监控与告警系统 (简称 ESCAS) 简介

系统由一个系统处理器、一个主单元处理器及与主单元相连的多个传感器、多个告警器组成。传感器由热传感器和图像传感器组成, 探测监控区内的活动情况。系统要求当发生紧急情况时, 被自动激活: 当发现入侵者, 告警器产生警告声音或警告光来警告入侵者并将信息送给系统处理器, 当发现火警, 产生警告声音并通知系统处理器。

3.2 用于电子监控与告警系统的对象分析和实时模式

下面分析系统需求需要使用哪些对象分析模式和实时约束模式。

3.2.1 结构模式

使用结构模式模板的 UML 类图模板来引导建立 ESCAS 的 UML 类图。

1 控制器分解模式。为了能对 ESCAS 清晰地进行分析, 将系统根据其职责分解为各个组件, 使用此模式, 它不具体描述系统的功能。在 ESCAS 中, 中央处理组件是核心, 控制着传感器和触发器, 用户接口的职责是控制指示器将系统状态显示给用户, 系统不使用通信链接 (不是分布式系统)。控制器分解模式捕获各个组件。

2 触发器-传感器模式。为了细化控制器分解模式捕获的传感器 (Sensor)、触发器 (Actuator) 功能, 使用触发器-传感器模式, 用 Sensor1 传感热信息, 用 Sensor2 传感图像信息, 当有告警信号时, 触发器激活告警器, 产生声音警报或光警报, 并驱动指示灯显示 (DriverDisplay), 将系统状态显示给用户。

3 用户接口模式 (UserInterface)。应用这个模式细化被控

制器分解模式捕获的用户接口组件, 在 ESCAS 中, 用户接口只控制指示器 (Indicator), 显示驱动器 (DriverDisplay), 用指示灯表示告警信息指示给用户。

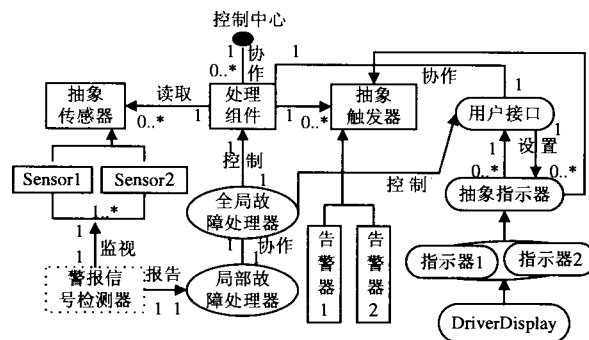


图 2 被对象分析模式引导得到的 ESCAS UML 类图

图 2 显示了使用结构模式模板的 UML 类图模板指导建立的 ESCAS 初始 UML 类图, 其中用矩形表示触发器-传感器模式类, 用圆角矩形表示用户接口模式类, 用黑体矩形表示处理组件类, 用椭圆表示故障处理器模式类, 用虚线矩形表示检测器模式类。

3.2.2 行为模式

使用行为模式模板的 UML 状态图和次序图模板来引导建立 ESCAS 的 UML 状态图和序列图。

1 处理组件模式 (ComputingComponent)。是系统控制与处理中心, 使用处理组件来进行操作控制与处理。模式有三种操作状态: 空闲、读取、处理, 其状态图见图 3 (要求循环读取 Sensor 中信息, Periodic<=1s)。

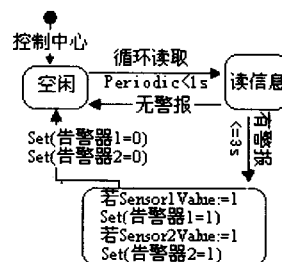


图 3 处理组件的 UML 状态图

2 故障处理模式。应用此模式进行故障的处理, 含全局故障处理 (GlobalFaultHandler) 和局部故障处理 (LocalFaultHandler), 局部故障处理将故障信息发送给全局故障处理, 全局故障处理控制处理组件进行处理行动, 同时控制用户接口, 将警告信息指示给用户 (均要求 3s 内完成)。

3 检测器-修正器模式。处理组件必须监视传感器 (Sensor) 的状态, 以便做出相应的处理行动, 为了减轻处理组件的负担, 使用检测器 (Detector) 监视 Sensor, 当它检测到报警信息后, 通告给局部故障处理器, 局部故障处理器再报告给全局故障处理器, 全局故障处理器控制处理组件采取相应的行动, 整个过程必须在 3s 内完成。

上述行为的协作次序图见图 4。

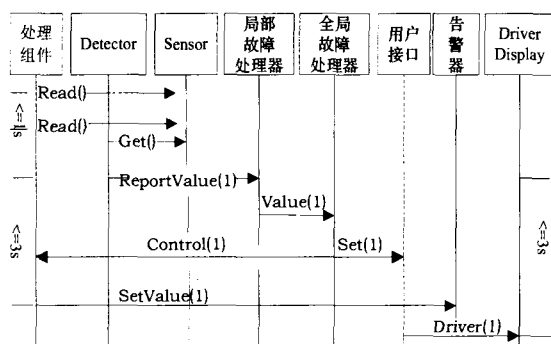


图 4 ESCAS UML 次序图

3.2.3 实时模式

使用实时描述模式表达时间约束条件。

1 循环约束实时规范模式。因为系统要时刻监控 Sensor 的情况, 处理组件要不断地查询 Sensor 以获取警报信息, 要求每隔 1s 钟就要查询一次, 因此使用循环约束 (Bounded Recurrence) 实时规范模式来描述此循环约束事件。

2 响应约束实时规范模式。系统要求当检测到警报到后, 处理组件和用户接口做出响应行动时间不超过 3s, 使用响应约束 (Bounded Response) 表述。

3.3 ESCAS 基于 UML 的概念模型

根据前面建立的基于 UML 的 ESCAS 类图 (图 2)、行为模式的状态图模板和序列图 (图 4)、实时模式描述的时间约束信息, 建立 ESCAS 基于 UML 的带时间信息概念模型图 5。在图 5 中, 处理组件是系统的核心, 它从传感器读取信息值, 还能设置告警器的值, 检测器监视传感器, 一旦有警报, 就报告给局部故障处理器, 局部故障处理器再报告给全局故障处理器, 全局故障处理器控制处理组件和用户接口采取适当的行动, 处理组件将警报信息传给用户接口并激活它, 用户接口控制 DriverDisplay 将警报信息以警报灯的形式指示给用户。

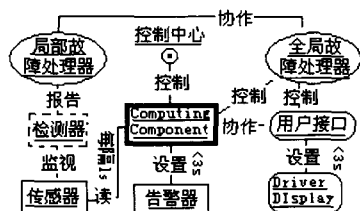


图 5 被抽象的带时间信息的 ESCAS UML 对象图

3.4 模型约束分析与检测

在利用对象分析模式和实时描述模式建立了系统初始概念模型后, 需要对模型进行分析和验证, 通过对系统需求分析, 根据图 5 和 3.2 的分析可知处理组件模式、触发器-传感器模式、故障处理模式、用户接口模式、循环约束实时规范模式、响应约束实时规范模式结合起来的 10 个约束条件满足系统需求。为了能对模型进行分析和验证, 用 Hydra^[1], 并利用 2.1 和图 1 定义的语义规则和映射规则, 生成形式化的可执行规范

描述 (系统被捕获的对象属性用队列语义经过通道进行通信) 来可视化的分析错误并予以纠正, 然后用 SPIN 检测器进行验证。下面例示其中 4 个实验中对约束条件的分析和验证情况:

1 循环约束实时描述模式。处理组件每隔 1s 循环读取 Sensor 信息, 设定时间粒度为 100ms (下同), 用 in(operationRead)表读取状态, 认为每 1s 多于读取 1 次是正确的, 这个属性适合循环约束实时规范模式在全局范围内的 MTL 规则。

形式化描述:

$$\square(\Diamond \leq 10(\text{in}(\text{operationRead}))) \quad (1)$$

实验结果: 用 SPIN 检验(1), 在经过 368236 次转换, 存贮 311512 种状态, 占用 32.468M 内存, 耗时 18.26s 后, (1)通过验证。

2 处理组件模式约束。当警报消息发送给处理器组件, 激活其中适当的告警器装置后, 处理组件激活适当的告警器, 用 1 表示告警器激活。

形式化描述:

$$\square((\text{Send}(\text{ComputingComponent.Activate}(1))) \rightarrow \Diamond (\text{告警器.DriverValue}==1)) \quad (2)$$

实验结果: 用 SPIN 检验(2), 在经过 386527 次转换, 存贮 202896 种状态, 占用 26.225M 内存, 耗时 15.13s 后, (2)通过验证。

3 用户接口模式约束。当警报消息发送到用户接口激活其中适当的警示灯装置后, 所有相应的指示器被激活, 用 1 表示激活。

形式化描述:

$$\square((\text{Send}(\text{UserInterface.Activate}(1))) \rightarrow \Diamond (\text{DriverDisplay.ActivateValue}==1)) \quad (3)$$

实验结果: 用 SPIN 检验(3), 在经过 346423 次转换, 存贮 202889 种状态, 占用 26.225M 内存, 耗时 15.10s 后, (3)通过验证。

4 响应约束实时规范模式。从局部故障处理器被警报通报, 到处理组件激活适当的告警器, 要求在 30 个时间单元(3s)内完成, 这个属性适合响应约束实时规范模式在全局范围内的 MTL 规则。

形式化描述:

$$\square((\text{Send}(\text{GlobalFaultHandler}(1))) \rightarrow \Diamond \leq 3 (\text{告警器.DriverValue}==1)) \quad (4)$$

实验结果: 用 SPIN 检验(4), 在经过 546133 次转换, 存贮 504762 种状态, 占用 40.245M 内存, 耗时 25.28s 后, (4)通过验证。

4 结论

从上应用实例可看出, 模式方法直观、描述简便, 为了与当前典型的嵌入式建模方法深入比较, 进行了几种建模方法所化时间的量化实验: 将建模者分为三组, 每组二人, 分别使用 VHDL、UML 和模式方法进行描述, 实验系统是 ESCAS 系统。

(下转第 3364 页)

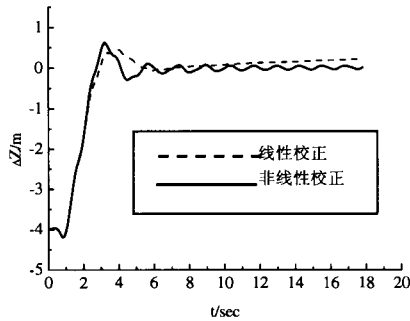


图 13 非线性校正弹道比较 (水平面)

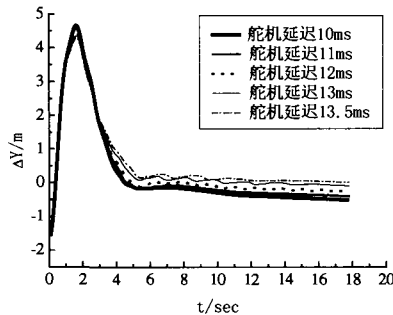


图 14 改进后的弹道在铅垂面内的误差曲线

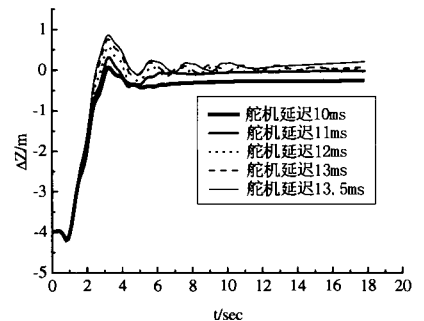


图 15 改进后的弹道在水平面内的误差曲线

3 结论

本文从分析某型反坦克导弹制导回路的设计缺陷入手,改进设计了纵向通道和横向通道的控制参数,大胆引入了非线性控制方法,仿真结果表明,本文的设计不仅能解决个别导弹系统右偏问题,而且具有较高命中精度,为该型反坦克导弹制导参数的改进设计奠定了理论基础。

参考文献:

[1] 钱杏芳. 导弹飞行力学 [M]. 北京: 北京理工大学出版社, 2002.

- [2] 万春熙. 反坦克导弹设计原理 [M]. 北京: 国防工业出版社, 1981.
- [3] 李颖, 朱伯立. Simulink 动态系统建模与仿真基础 [M]. 西安: 西安电子科技大学出版社, 2004.
- [4] 宋振铎. 反坦克制导兵器论证与试验 [M]. 北京: 国防工业出版社, 2003.
- [5] 齐占元. 超近程反坦克导弹武器系统设计与仿真研究 [D]. 北京: 北京理工大学, 2004: 56-74.
- [6] Zhu Boli, Yang Shuxing. On the Elastic Vibration Model for High Length-Diameter Ration Rocket with Attitude Control System [J]. Journal of Beijing Institute of Technology (S1004-0579), 2003, 3(12): 269-272.

(上接第 3349 页)

实验结果是: VHDL 小组的建模时间是模式方法小组的 2.92 倍、文字描述量 4 倍左右, 并发生 2 个错误, 难理解, 描述量大; UML 小组是模式方法小组的 1.26 倍、文字量相近, 但 UML 模型需特殊支撑工具进行形式化转换和模型分析、验证, 工作复杂, 工作量大, 且状态图对时间约束的建模能力不强。模式方法能利用结构模式 UML 类图模板和行为模式 UML 对象图/序列图模板简易地得出系统的 UML 概念模型, 并根据文中定义类似 Gamma 规则的语义规则和实时映射, 很容易地得出带时间信息的可执行的形式化描述模型, 进而利用 UML 的形式化框架, 容易地转换为 Promela 格式, 这样能使用 Spin 进行模型检测和验证, 并且应用对象分析模式和实时描述模式相结合进行嵌入式建模及模型分析, 在嵌入式实例化建模时可根据系统需求对模式模板进行选择、裁剪、扩充, 模式模板对每个嵌入式建模通用, 可快速、高效地进行模型重用, 提高了模型的复用性, 解决了 UML 模型分析与检测难和 UML 时间约束的建模能力不强的问题。但是这个方法不能对自然语言进行分析, 需要对模型和时间约束信息规范化表示后才能进行分析。下一步的工作将研究如何自动分析 UML 模型的自然语言属性及 UML 图的时间信息, 以进一步简化建模工作。

参考文献:

[1] G Booch, J Rumbaugh, I Jacobson. The Unified Modeling Language User Guide [M]. Massachusetts: Addison-Wesley, 1999.

- [2] 石柯, 阳富民, 胡贵荣. 基于 UML 的嵌入式系统模型验证机制的研究 [J]. 计算机工程与应用, 2001, 37(23): 111-113.
- [3] 赖明志, 尤晋元. 使用时间化自动机形式化带有时间扩展的 UML 状态图 [J]. 计算机应用, 2003, 23(8): 4-6.
- [4] B P Douglass. Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks and Patterns [M]. Massachusetts: Addison-Wesley, 1999.
- [5] S Konrad, B H C Cheng, L A Campbell. Object analysis patterns for embedded systems [J]. IEEE Transactions on Software Engineering (S0098-5589), 2004, 30(12): 970-992.
- [6] S Konrad, B H C Cheng. Real-time specification patterns [C]// Proceedings of the 27th International Conference on Software Engineering, St. Louis, MO, USA: ACM, 2005: 372-381.
- [7] S Konrad, B H C Cheng. Automated Analysis of Natural Language Properties for UML Models. [J/OL]. (MAY, 2006) [2006-12-20]. <http://www.cse.msu.edu/~konradsa>.
- [8] W E Mcumber, B H C Cheng. A General Framework for Formalizing UML with Formal Languages [C]// Proc. IEEE Int'l Conf. Software Eng. (ICSE 2001), Michigan: IEEE, 2001: 433-442.
- [9] LA Campbell, B H C Cheng, W E Mcumber, et al. Automatically Detecting and Visualizing Errors in UML Diagrams [J]. Requirements Eng. J. (S0947-3602), 2002, 7(4): 264-287.
- [10] G J Holzmann. The Model Checker SPIN [J]. IEEE Trans. Software Eng. (S0098-5589), 1997, 23(5): 279-295.
- [11] E Gamma, R Helm, R Johnson, et al. Design Patterns: Elements of Reusable Object-Oriented Software [M]. Massachusetts: Addison-Wesley, 1994.