基于MARTE模型的系统可靠性预测

柴叶生[1,2] 朱雪阳[2] 晏荣杰[2] 张广泉[1]

(苏州大学计算机科学与技术学院 苏州 215000)[1]

(中国科学院软件研究所计算机科学国家重点实验室 北京 100190)[2]

摘 要 系统的可靠性是系统的重要非功能属性之一。传统的可靠性分析在系统开发结束后进行,可能会发现由于系统开发早期的架构设计不合理而导致的问题,这时再修改系统架构并重做后继开发步骤,将会浪费大量人力和物力。如果能在开发的早期阶段,在系统模型层面进行分析并预测,则可以尽早地发现系统可靠性方面的问题并将其修复。UML是一种通用的、标准化的建模语言,MARTE是UML在嵌入式实时系统领域的扩展。提出了基于MARTE模型的系统可靠性预测方法,该方法考虑的MARTE模型包括用例图、活动图、部署图。先将MARTE模型转换为马尔可夫决策过程网络模型,再利用概率模型检测工具PRISM进行分析,得到系统可靠性的预测结果。实例研究表明,所提方法不仅能够预测系统可靠性的最大值和最小值,还能通过调整各个资源的可靠性值,考察其对系统可靠性的影响,为设计人员的进一步工作提供参考。

关键词 系统可靠性,UML,MARTE模型,马尔可夫决策过程

中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.12.018

MARTE Models Based System Reliability Prediction

CHAI Ye-sheng[1,2] ZHU Xue-yang[2] YAN Rong-jie[2] ZHANG Guang-quan[1]

(Department of Computer Science and Technology, Soochow University, Suzhou 215000, China)[1]

(State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)[2]

Abstract System reliability is an important qualitative attribute of systems. System reliability analysis based on models may find problems and solve them at the early phase of development. UML profile for MARTE is an extension of UML in the domain of real-time and embedded systems. We presented a method for reliability prediction of MARTE models. The MARTE model considerd by the method includes a use case diagram, a deployment diagram and a set of activity diagrams. A MARTE model is transformed into a network of Markov decision process, which is then analyzed by the model checking tool PRISM. We obtained the estimation of system reliability by analyzing the resulting model. A case study demonstrates that, by analyzing models with various resource reliability, our method can further reveal the impact of the reliability of each resource on the system.

Keywords System reliability, UML, MARTE model, Markov decision process

1 引言

随着计算机技术的发展,仅功能正确的软件已不能满足用户的需求,用户对软件的非功能属性如时间性质、软件可靠性等的要求越来越高。系统的可靠性是在规定的条件和时间下系统无故障运行的概率[1],是系统的重要非功能属性之一。传统的可靠性分析在系统开发结束后进行,可能会发现由于系统开发早期的架构设计不合理导致的问题,这时再修改系统架构并重做后继开发步骤,将会浪费大量人力和物力。系统的可靠性预测是发生在系统开发的早期阶段,通过已知或预测的数据(如资源的可靠性),在系统模型层面进行分析,可以尽早地发现问题并修复。

UML[2](Unified Modeling Language)可以从多个角度来描述系统的结构、行为特征,已经成为事实的工业标准。MARTE[3](Modeling and Analysis of Real Time and Embedded system)规范是UML的一个扩展(UML profile),用于嵌入式系统开发的建模与分析。MARTE模型由UML和MARTE规范组成。由于MARTE模型缺乏严格的语义,因此很难直接对其进行性能分析。形式化方法以严格的形式化模型为中心,具有精确的语义,因此,基于MARTE模型的性能分析方法通常需要将MARTE模型转化为形式化模型,再通过对形式化模型进行分析得到系统的安全性、可靠性或能耗的估算值。

在UML模型的系统可靠性分析方面,已有大量的相关

到稿日期:2015-02-11 返修日期:2015-04-28 本文受江苏省自然科学基金(BK2011281),苏州市应用基础研究计划(SYG201241)资助。

柴叶生(1988-),男,硕士,主要研究方向为MARTE模型、系统可靠性和压力测试,E-mail:chaiyesheng@126.com;朱雪阳(1971-),女,博士,助理研究员,主要研究方向为嵌入式系统设计和形式化方法;晏荣杰(1977-),女,博士,助理研究员,主要研究方向为实时系统模型及验证;张广泉(1965-),男,教授,CCF高级会员,主要研究方向为网络软件工程、形式化方法、云计算和物联网(CPS)。

研究工作。文献[4]采用例图、部署图和序列图建立性能模型，然后通过贝叶斯方法进行系统可靠性预测。文献[5]将 UML 模型中的场景和消息序列图转化成带标签的迁移系统，然后将带标签的迁移系统转换成马尔可夫链模型，最后通过文献[6]中的基于马尔可夫链分析方法进行系统可靠性的预测。文献[7]将 UML 结构视图中的类图和对象图转换成时间 Petri 网，然后采用基于时间 Petri 网的分析方法得到系统可靠性。文献[8]将标注可靠性信息的用例图、序列图、活动图和构件图转换成马尔可夫链，方便系统可靠性的分析。

本文提出了基于 MARTE 模型预测系统可靠性的方法。该方法将 MARTE 模型转换成马尔可夫决策过程(Markov Decision Process，MDP)[9]模型，再借助概率模型检测工具 PRISM[10]分析系统可靠性的最大值和最小值。相对于以往的相关方法，该方法考虑了不确定性特征，预测系统可靠性的最大值和最小值，并分析资源可靠性值的变化对系统可靠性最大值和最小值的影响。

2 MARTE 模型和马尔可夫决策过程模型

2.1 MARTE 模型

本文方法采用的 MARTE 模型包括用例图、活动图、部署图和 MARTE 构造型(stereotype)，构造型包括≪PaStep≫和≪gaExecHost≫。下面以图 1 所示酒店预订系统为例介绍本文中用到的 MARTE 模型的主要构成。

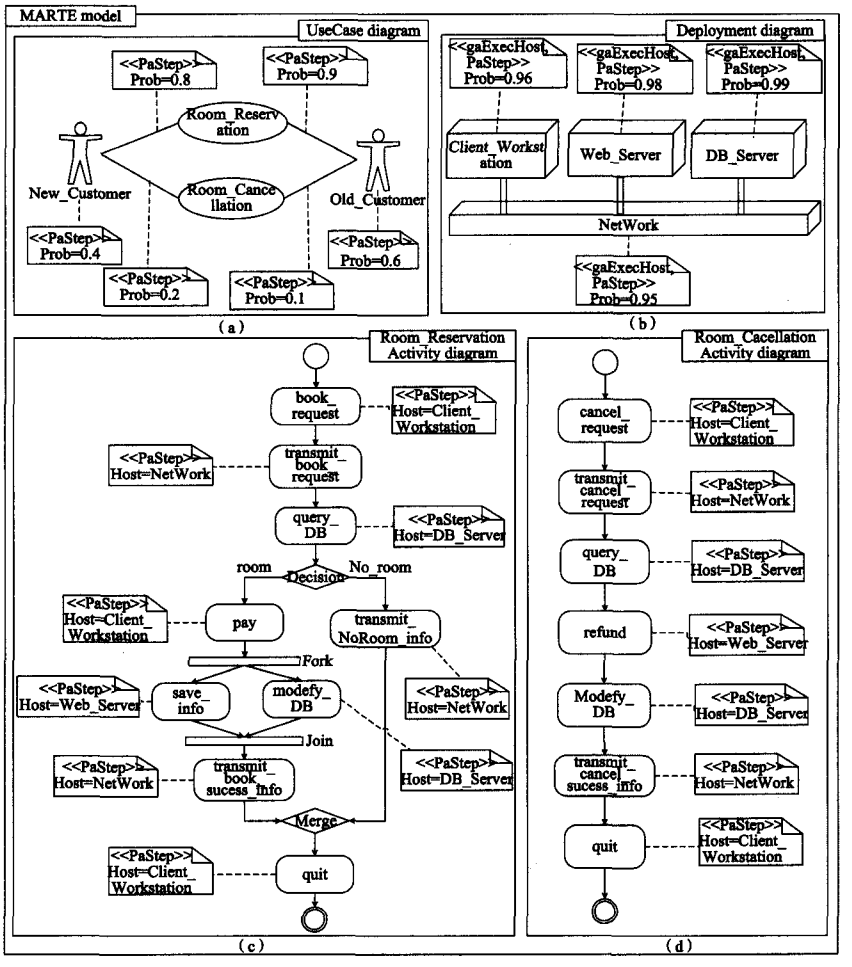

图 1 酒店预订系统

用例图描述系统的用户和系统的交互情况，由角色、用例和关联线组成。每个角色都有≪PaStep≫构造型，通过其 prob 标签值赋予这个角色使用系统的概率。角色和用例之间的关联都有≪PaStep≫构造型，通过其 prob 标签值赋予角色执行用例的概率。用例的具体功能由相应的活动图实现。当用例图中有 n 个角色，m 个用例时，满足：$\sum_{x=1}^{n} q_x = 1$，其中 q_x 是以第 x 个角色使用系统的概率。对于第 x 个角色，需要满足 $\sum_{y=1}^{m} p_{xy} = 1$，其中 p_{xy} 表示第 x 个角色执行第 y 个用例的概率。如图 1(a)所示，角色 *New_Customer* 使用系统的概率为 0.4，该角色选择执行用例 *Room_Reservation* 的概率为 0.8。

部署图描述软件系统中资源的拓扑结构，由资源和连接线组成。每个资源结点都有≪PaStep≫和≪gaExecHost≫构造型，通过≪PaStep≫的 prob 标签值描述资源的可靠性即资源无故障执行的概率。如图 1(b)中 NetWork 资源的可靠性值为 0.95。

活动图描述系统功能的动态行为、活动之间的顺序关系。活动图中的每个活动结点都有≪PaStep≫构造型，通过其 Host 标签值将其部署到部署图的资源上，表示活动由其相应的资源来执行。如图 1(c)中活动 *book_request* 部署到资源 *Client_Workstation*。活动图各元素的具体含义见 2.2 节。

2.2 马尔可夫决策过程模型

定义 1 马尔可夫决策过程(Markov Decision Process，MDP)[9]是一个四元组 $M=(S, s_0, Act, \delta_M)$，其中 S 表示状态

的有限集合，$s_0 \in S$ 表示初始状态，Act 表示动作有限集合，$\delta_M: s \times Act \times S \to [0,1]$ 表示状态间的概率迁移函数，满足 $\sum_{s' \in suc(s,a)} \delta_M(s,a,s') = 1$，其中 $s \in S, a \in Act, suc(s,a)$ 是状态 s 和动作 a 的后继状态集合。

图 2 是一个简单的 MDP 实例，其中 $S=\{q_0,q_1,q_2\}$，$s_0=q_0$，$Act=\{a_0,a_1,a_2,b_0\}$。状态 q_0 的概率迁移函数 $\delta_M(q_0,a_0,q_1)=0.4$，$\delta_M(q_0,a_0,q_2)=0.6$，$\delta_M(q_0,b_0,q_0)=1.0$。状态 q_0 所关联的动作不止一个，体现了 MDP 的不确定性特征。

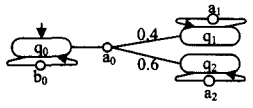

图 2　一个简单的 MDP 实例

PRISM[10]是一款开源的概率模型检测工具，支持对马尔可夫链、马尔可夫决策过程等概率模型的分析。PRISM 模型由模板和全局变量组成，模板由局部变量和命令组成。命令的基本形式为：$[Actio]guard \to prob_1 : update_1, \cdots, prob_n : update_n$，其中，$prob_j$ 表示迁移的概率，满足 $\sum_{1 \leqslant j \leqslant n} prob_j = 1$，$Action$ 表示动作标签，$guard$ 是局部变量或者全局变量的逻辑表达式，$update_j$ 是对局部变量或者全局变量的值的更新，其中 $1 \leqslant j \leqslant n$。图 3 给出一个 PRISM 模型示例，由模板 $M1$、$M2$ 和全局变量 e、g 组成。$M1$ 模板由 5 个命令和局部变量 q 组成，$M2$ 模板由一个命令和全局变量组成。

```
mdp
global e:bool init false; global g:bool init false;
module M1
q:[0..6] init 0;
[]q=0 ->0.4:(q'=1)+0.6:(q'=2);//a0
[]q=0 ->1.0:(q'=5);//a1
[]q=1 ->1.0:(q'=3)&(e'=true);//a2
[]q=2 ->1.0:(q'=4)&(g'=true);//a3
[]q=5 ->1.0:(q'=6)&(e'=true);//a4
endmodule
module M2
p:[0..1]init 0;
[]p=0 & e=true ->1.0:(p'=1)&(e'=false);//b0
endmodule
```

图 3　PRISM 模型示例

在 PRISM 模型的图形化描述中，局部变量的不同取值分别对应到图形中状态，命令对应图形中状态的迁移，并且命令中全局变量的条件与更新对应到图形中迁移边的卫士条件和赋值。图 4 是图 3 所示模型的图形化表示。图 4 中模型 $M1$ 和 $M2$ 分别对应图 3 中模板 $M1$ 和 $M2$。图 4 中的状态 p_i 和 q_j 分别表示图 3 中当前局部变量 $p=i$ 和 $q=j$ 的状态。迁移上的卫士条件 $e=T$ 和赋值 $e:=F$，表示当全局变量 e 为 true(T)时，迁移发生并且将 e 赋值为 false(F)。图 4 中模型 $M1$ 和 $M2$ 之间通过全局变量 e 实现通信。模型 $M1$ 中动作 a_2 或者 a_4 的迁移发生，将全局变量 e 赋值为 true(T)，然后模型 $M2$ 中动作 b_0 的迁移发生到达状态 p_1。$P_{max}=?$ 和 $P_{min}=?$ 分别是 PRISM 工具中最大概率和最小概率的关键字，F 为时序算子。通过 PRISM 公式 $P_{max}=?[F\ p_1]$ 和 $P_{min}=?[F\ p_1]$ 分别表示图 4 中模型 $M2$ 到达状态 p_1 的最大概率和最小概率，对应图 3 中局部变量 p 最终为 1 的最大概率和最小概率。图 4 模型到达状态 p_1 的最大概率值为 1，对应的路径为：$q_0 \to q_5 \to q_6$，$p_0 \to p_1$，最小概率值为 0.4，对应的路径为：$q_0 \to q_1 \to q_3$，$p_0 \to p_1$。我们将利用这两个公式来计算系统可靠性的最大值和最小值。

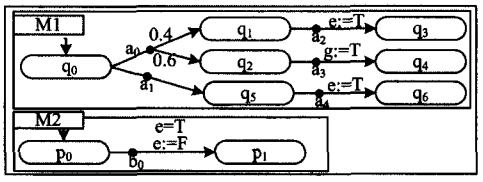

图 4　PRISM 模型的图形化示例表示

在不引起歧义的情况，将 PRISM 模型的图形化表示称为 MDP 模型。本文在未做特殊说明的情况下，全局变量的初始值为 false(F)。

3　模型转换

MARTE 模型到 MDP 模型的转换，包括用例图到 MDP 模型转换和活动图到 MDP 模型转换。MARTE 模型中活动结点分别被部署到部署图中的资源上，活动结点的转换规则中包括从部署图中提取其相应资源的可靠性信息，因此我们不对部署图单独做转换。

3.1　用例图到 MDP 模型转换

用例图描述用户使用系统的模式。由 n 个角色、m 个用例和($n \times m$)个关联组成的用例图用 MDP 模型表示的行为语义如图 5 所示。

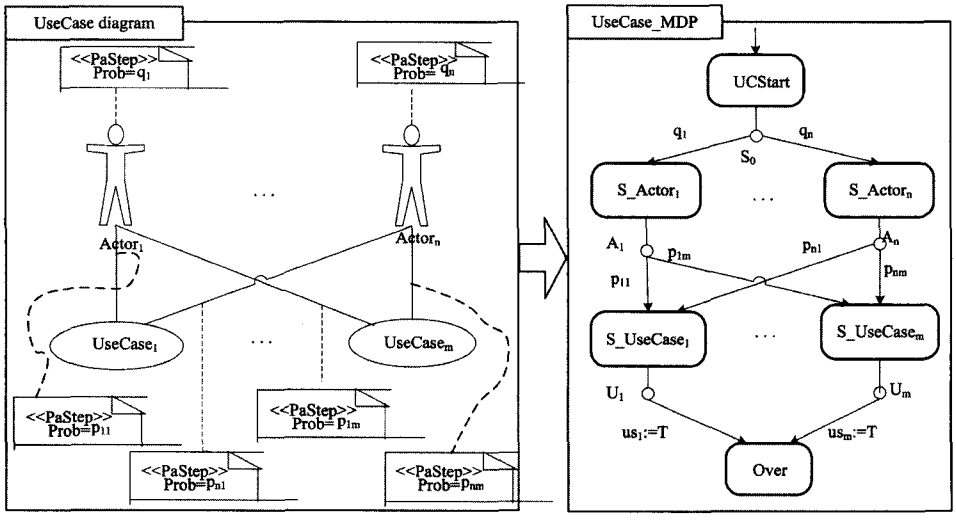

图 5　用例图到 MDP 模型转换规则

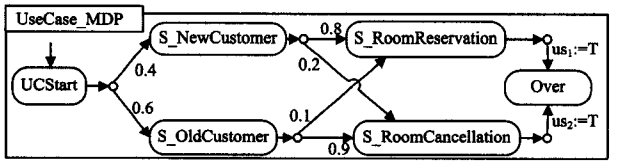

图 6　图 1 用例图转换的 MDP 模型

根据 2.1 节用例图的介绍，用户与系统的交互过程分为两步。第一步：根据角色的概率分布选择使用系统的角色；第二步：根据角色执行用例的概率分布选择被执行的用例。第一步的语义转换为模型 *UseCase_MDP* 中初始状态 *UC-Start* 到状态 $S\_Actor_1,\cdots,S\_Actor_n$ 迁移，概率分别为 $\delta_M(UC\text{-}Start,S_0,S\_Actor_1)=q_1,\cdots,\delta_M(UCStart,S_0,S\_Actor_n)=q_n$，其中，状态 *UCStart* 表示用例图的开始执行，用例图中角色 $Actor_1,\cdots,Actor_n$ 转换为状态 $S\_Actor_1,\cdots,S\_Actor_n$。第二步的语义转换为模型 *UseCase_MDP* 中状态 S_Actor_j 到状态 $S\_UseCase_1,\cdots,S\_UseCase_m$ 的迁移，概率分别为 $\delta_M(S\_Actor_j,A_j,S\_UseCase_1)=p_{j1},\cdots,\delta_M(S\_Actor_j,A_j,S\_UseCase_m)=p_{jm}$，其中，用例图中的用例 $UseCase_1,\cdots,UseCase_m$ 转换为状态 $S\_UseCase_1,\cdots,S\_UseCase_m$，$j\in[1,n]$。模型 *UseCase_MDP* 中的状态 $S\_UseCase_j$ 到状态 *Over* 的迁移将全局变量 us_j 赋值为 true，用来与对应的活动图的 MDP 模型（参见 2.2 节）通信，触发活动图的开始执行。图 1 中的用例图转换的 MDP 模型如图 6 所示。

3.2　活动图到 MDP 模型转换

本文的活动图包含图 7 中所示的元素。活动图到 MDP 模型转换的思路为：将初始结点（Init Node）、合并结点（Merge Node）、分叉结点（Fork Node）和汇合结点（Join Node）的后继工作流分别转换为模型 *Init_MDP*、*Merge_MDP*、*Fork_MDP* 和 *Join_MDP*，并且模型之间的通信通过全局变量实现。活动图到 MDP 模型转换规则如图 7 所示。

初始结点的转换如图 7(a)所示。该规则生成模型 *Init_MDP*，初始状态为 *ADStart*，当全局变量 us_j 为 true 时执行其后继迁移，同时将 us_j 赋值为 false。活动图与用例图的通信通过全局变量 us_j 实现。用例图通过将 us_j 置为 true 来触发第 j 个用例所对应的活动图。

活动结点的转换如图 7(b)所示。活动结点通过构造型≪Pastep≫将其部署到资源图的资源上。由于资源的运行可能出现故障，因此生成了状态 *Fault*，表示执行中出现故障。当资源 *resource* 可靠性值为 p_1 时，状态 A 到达后继状态的概率为 p_1，到状态 *Fault* 的概率为 $1-p_1$。

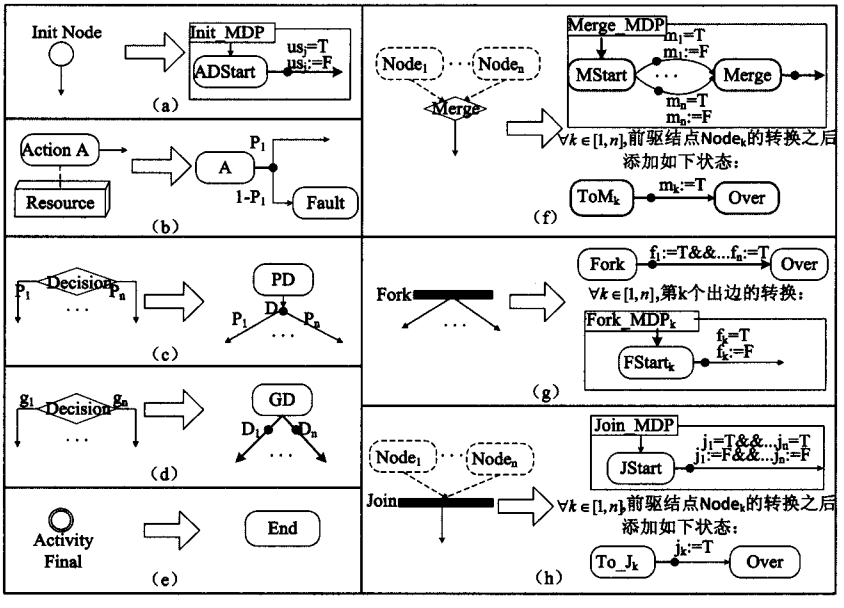

图 7　活动图到 MDP 模型转换规则

概率选择结点的后继工作流中的概率选择特征与 MDP 模型中动作的概率迁移特征相同，因此其转换规则如图 7(c)所示。

在不同的情形下，条件选择结点会选择不同的后继工作流执行，具有不确定性的特征，因此，将其转换为 MDP 模型中多个动作之间的不确定性选择，如图 7(d)所示。

活动结束结点表示活动图的执行结束，因此将活动结束结点转换成状态 *End*，如图 7(e)所示。

合并结点描述其多个前驱工作流的其中一个执行结束后，执行其后继工作流。合并结点的转换如图 7(f)所示。该规则生成模型 *Merge_MDP*。根据合并结点的入边个数 n，状态 *MStart* 和状态 *Merge* 之间有 n 个迁移，并且每个迁移当全局变量 m_k 为 true 时执行，同时将 m_k 赋值为 false。前驱结点 $Node_k$ 可以是活动、概率选择、条件选择、合并、分叉和汇合中任意一种类型的元素。前驱结点 $Node_k$ 的转换之后会添加状态 ToM_k，并且状态 ToM_k 的后继迁移将全局变量 m_k 赋值为 true 之后，到达状态 *Over*。全局变量 m_k 实现合并结点的前驱工作流转换得到的 MDP 模型和后继工作流转换得到的 MDP 模型之间的顺序执行，其中 $k\in[1,n]$。图 1 中的合并结点的转换见图 8 中模型 *Merge_MDP*、模型 *Init_MDP* 中状态 ToM_1 和模型 *Join_MDP* 中状态 ToM_2。

分叉结点描述其多个后继工作流的并发执行。分叉结点的转换如图 7(g)所示。该规则生成了状态 *Fork*，并且其后继的迁移将全局变量 $f_1,\cdots,f_n$ 赋值为 true 之后，到达状态 *Over*。分叉结点的第 k 个出边的转换生成模型 $Fork\_MDP_k$，初始状态为 $FStart_k$，并且其后继迁移当全局变量 $f_1,\cdots,f_n$

全部为 true 时执行，同时将 $f_1,\cdots,f_n$ 全部赋值为 false。全局变量 f_k 实现分叉结点的前驱工作流转换得到的 MDP 模型和后继工作流转换得到的 MDP 模型之间的顺序执行，其中 $k\in[1,n]$。图 1 中的分叉结点的转换见图 8 中模型 *Init_MDP* 中状态 *Fork*、模型 $Fork\_MDP_1$ 和模型 $Fork\_MDP_2$。

汇合结点描述其多个前驱工作流全部执行结束后，执行其后继工作流。汇合结点的转换规则和合并结点的转换规则类似，如图 7(h)所示。图 1 中的汇合结点的转换见图 8 中模型 *Join_MDP*、模型 $Fork\_MDP_1$ 中状态 ToJ_1 和模型 $Fork\_MDP_2$ 中状态 ToJ_2。

根据图 7 的转换规则，模型 *Init_MDP*、*Merge_MDP*、*Fork_MDP* 和 *Join_MDP* 未包含由概率选择、条件选择、分叉和活动等结点转换得到的状态、动作和边。根据活动图中结点和边之间的相连关系，得到转换得到的状态、动作和边之间的相连关系，然后遍历不属于任何 MDP 模型的状态、动作和边，并将它们添加到相应的 MDP 模型中。

根据以上转换规则和处理，图 1 活动图 *Room_Reservation* 转换得到的 MDP 模型如图 8 所示。

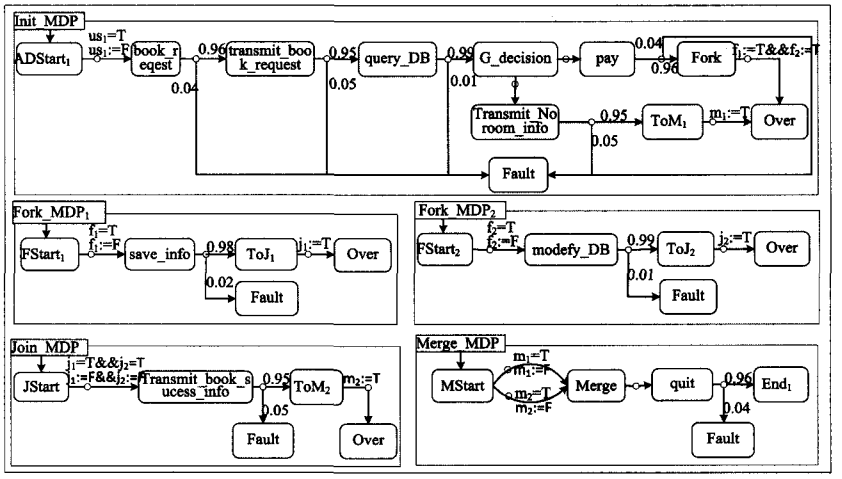

图 8　图 1 活动图 Room_Reservation 转换的 MDP 模型

4　模型分析

对于具有 n 个角色、m 个用例的 MARTE 模型，其系统可靠性定义[4]为：每个用例 $UseCase_j$ 被执行的概率乘以该用例所对应的活动图 $Activity\_diagram_j$ 无故障运行的概率，再求和，即

$$R(System)=\sum_{j=1}^{m}(p(UseCase_j)\times p(Success(Activity\_diagram_j))) \quad (1)$$

根据 3.1 节用例图到 MDP 模型的转换规则，$p(UseCase_j)$ 等于模型 $UseCase\_MDP$ 中从初始状态 *UCStart* 到达状态 $S\_UseCase_j$ 的概率。根据 3.2 节活动图到 MDP 模型的转换规则(见图 7(a))可知，活动图的开始结点对应模型 *Init_MDP* 中初始状态 *ADStart*。由图 7(b)可知，活动图运行过程出现故障，对应到 MDP 模型到达状态 *Fault*。由图 7(h)可知，活动图运行到结束结点，对应到 MDP 模型到达状态 *End*。因此，$p(Success(Activity\_diagram_j))$ 等于从状态 $ADStart_j$ 执行，最终到达状态 End_j 的概率。由于模型 $UseCase\_MDP$ 和模型 *Init_MDP* 之间通过全局变量 us_j 通信，模型 $UseCase\_MDP$ 中状态 $S\_UseCase_j$ 的后继迁移将全局变量 us_j 赋值为 true，当全局变量 us_j 为 true 时模型 *Init_MDP* 中初始状态的后继迁移执行。因此，$p(UseCase_j)\times p(Success(Activity\_diagram_j))$ 等于从状态 *UCStart* 开始执行，最终到达状态 End_j 的概率。只考虑 MARTE 模型中活动图 $Activity\_diagram_1,\cdots,Activity\_diagram_m$ 之间是互斥执行的情况，所以转换得到的 MDP 模型最终到达状态 $End_1,\cdots,End_m$ 也是互斥的，因此系统的可靠性等于从状态 *UCStart* 开始执行，最终到达状态 $End_1,\cdots,End_m$ 中一个状态的概率。

当活动图包含条件选择结点时，选择不同的分支执行得到的系统可靠性值不同。通过计算系统可靠性的最大值和最小值来评估系统的可靠性范围。这可通过将 MARTE 模型按以上讨论转换的 MDP 模型以及式(2)和式(3)作为输入，利用 PRISM 计算而得。通过 PRISM 式(2)和式(3)分析转换得到的 MDP 模型，分别得到系统可靠性的最大值和最小值。

$$P_{max}=?[F(End_1\,|\,End_2\,|\cdots End_m)] \quad (2)$$

$$P_{min}=?[F(End_1\,|\,End_2\,|\cdots End_m)] \quad (3)$$

5　实验

在 FMPAer(Formal Models-based Performance Analyser)[11]工具集中实现了本文的方法。借助于建模工具 Papyrus[12]建立 MARTE 模型，然后运用模型转换语言 ATL[13]将 MARTE 模型转换为 MDP 模型，并遍历 MDP 模型生成相应的式(2)和式(3)，最后利用开源的概率模型检测工具 PRISM 计算系统可靠性的最大值和最小值。

针对图 1 的酒店客房预订系统，在当前系统的架构、操作规范和资源可靠性($R(ClientWorkstation)=96\%$，$R(Web\_Ser)=98\%$，$R(DB\_Ser)=99\%$，$R(NetWork)=95\%$)下，通过本文的方法预测系统可靠性的最大值和最小值分别为 82%和 77%。在系统设计阶段，分析单个资源的可靠性变化对系统可靠性的影响，可以为提高系统的可靠性提供更多参考依据。将资源 *NetWork* 的可靠性值从 90%变化到 100%，系统可靠性变化如图 9 所示，其中具有圆圈标记的实线为最大值，具有圆圈标记的虚线为最小值。图 9 也显示了其他资源的可靠性变化对系统可靠性的影响。由图 9 可得，资源

(下转第 91 页)

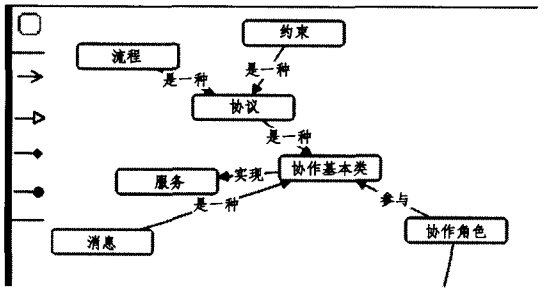

图 4　示例本体的概念图展示效果

结束语　提供一套具有较高可用性且符合语义网诸多语言标准的本体编辑器是本体研究领域的重要实现工作。已有的诸多本体编辑器均提供了强大的本体编辑与推理等功能，但它们仍然需要在展示本体概念图的实现机制上更加贴近实际用户的需求，并且在概念图内容正确性、概念图局部化显示以及轻量级的前后端数据传输方面体现其优势。

本文提出一个应用于协同本体编辑平台的本体概念图展示过程，该过程包含一系列步骤：本体读取、局部本体抽取、本体数据转换、图形模型转换和图形展示。这些步骤分别应用于建模平台的服务器端与客户端，涉及了抽取算法，并且建立在模型转换等通用技术的基础之上。

当然，本体概念图的展示在许多方面仍然具有扩展与改善的潜力，它们可作为我们今后的工作内容。这些潜在的改进包括扩展对数据属性（DatatypeProperty）和限制（Restriction）的展示、改善概念图的自动布局，以及实现选择性隐藏以支持概念过多的情况下的图形简化等诸多方面。

参 考 文 献

[1] Berners-Lee T, Hendler J, Lassila O. The Semantic Web[J]. Scientific American, 2001, 284(5): 28-37

[2] Protégé[OL]. http://protege.stanford.edu

[3] OntoStudio [OL]. http://www.semafora-systems.com/en/products/ontostudio/

[4] Dijkstra E W. On the Role of Scientific Thought[M]// Selected Writings on Computing: A Personal Perspective. Springer New York, 1982: 60-66

[5] mxGraph[OL]. http://www.jgraph.com/mxgraph.html

（上接第 86 页）

Web_Ser 的可靠性变化对系统可靠性最大值和最小值的影响比较小，资源 *NetWork* 和 *ClientWorkstation* 的可靠性变化对系统可靠性最大值的影响比较大，资源 *ClientWorkstation* 的可靠性变化对系统可靠性最小值的影响比较大。

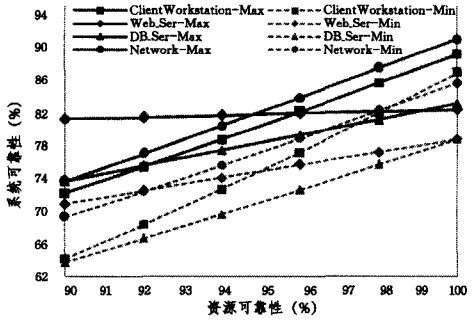

图 9　图 1 模型的系统可靠性预测

结束语　本文提出了基于 MARTE 模型的系统可靠性预测方法。该方法不仅能够预测系统可靠性的最大值和最小值，还能通过调整各个资源的可靠性值，考察其对系统可靠性的影响，为设计人员的进一步工作提供参考。下一步将在现在的方法上考虑资源的故障传播性和资源的冗余优化问题。

参 考 文 献

[1] Gokhale S S, Trivedi K S. Analytical models for architecture-based software reliability prediction: A unification framework [J]. IEEE Transactions on Reliability, 2006, 55(4): 578-590

[2] Booch G, Rumbaugh J, Jacobson I. Unified Modeling Language User Guide[M]. New York: Addison Wesley, 2005

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

[4] Cortellessa V, Singh H, Cukic B. Early reliability assessment of UML based software models[C]// Proceedings of the 3rd International Workshop on Software and Performance. Rome, Italy: ACM, 2002: 302-309

[5] Genaina N R, David S R. Sebastian Uchitel. Reliability Prediction in Model-Driven Development[C]// Proceedings of the 8th International Conference on Model Driven Engineering Languages and Systems. Berlin: Springer, 2005: 339-354

[6] Cheung R C. A User-Oriented Software Reliability Model[J]. IEEE Transactions on Software Engineering, 1980, 6(2): 118-125

[7] Majzik I, Patariczа A, Bondavalli A. Stochastic dependability analysis of system architecture based on UML Models[J]. Architecting Dependable Systems, 2003, 2677: 219-244

[8] 刘毅，麻志毅，何啸，等. 一种从 UML 模型到可靠性分析模型的转换方法[J]. 软件学报，2010，21(2)：287-304
Liu Yi, Ma Zhi-yi, He Xiao, et al. Approach to Transforming UML Model to Reliability Analysis Model[J]. Journal of Software, 2010, 21(2): 287-304

[9] Forejt V, Kwiatkowska M, Norman G, et al. Automated Verification Techniques for Probabilistic Systems[C]// Proceedings of 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM 2011). Bertinoro, Italy, Springer, 2011: 53-113

[10] Kwiatkowska M, Norman G, Parker D. PRISM 4.0: Verification of Probabilistic Realtime Systems[C]// Proceedings of 23rd International Conference on Computer Aided Verification. Berlin: Springer, 2011: 585-591

[11] FMPAer[OL]. http://lcs.ios.ac.cn/~zxy/tools/fmpaer.htm

[12] Gérard S, Dumoulin C, Tessier P, et al. Papyrus: A UML2 Tool for Domain Specific Language Modeling[C]// Proceedings of Model-Based Engineering of Embedded Real-Time Systems. Germany, Springer, 2010: 361-368

[13] Jouault F, Kurtev I. Transforming models with ATL[C]// Proceedings of the 2005 International Conference on Satellite Events at the MoDELS. Italy, Springer, 2005: 128-138