第37卷 第17期 计 算 机 工 程 2011年9月
Vol.37 No.17 ComPuteR EngineeRing September 2011

·软件技术与数据库· 文章编号：1000—3428(2011)17—0029—03 文献标识码：A 中图分类号：TP311.5

网构软件演化的业务一致性验证方法

包书勇，王忠群

(安徽工程大学计算机与信息学院，安徽 芜湖 241000)

摘 要：为提高网构软件的可信性，提出一种网构软件演化的业务一致性验证方法。基于接口自动机对由XYZ/ADL描述的系统进行语义解释，定义XYZ/ADL到接口自动机的转换规则，给出检验系统业务一致性的3个规则，结合实例给出业务一致性的检验过程。通过模型检测器Spin证明该方法能够验证网构软件演化的业务一致性。

关键词：网构软件；动态演化；业务一致性；接口自动机；体系结构描述语言

Business Consistency Verification Approach of Internetware Evolution

BAO Shu-yong, WANG Zhong-qun

(School of Computer and Information, Anhui Polytechnic University, Wuhu 241000, China)

【Abstract】Addressing the trustworthiness of internetware, this paper proposes an approach to verify business consistency of internetware evolution. The approach gives the semantic interpretation of system described by XYZ/ADL and defines the transformation rules from XYZ/ADL to interface automata, and presents three rules for verifying system business consistency. An example is incorporated to instantiate the application of these rules. Spin model-checker is used to prove that the approach can be used to verify the business consistency of internetware evolution.

【Key words】internetware; dynamic evolution; business consistency; interface automata; Architecture Description Language(ADL)

DOI: 10.3969/j.issn.1000-3428.2011.17.008

1 概述

基于 Internet 网络环境的开放、动态以及用户使用方式的个性化，要求网构软件应能根据用户的要求或者感知运行环境的变化，并按照可信度指标等进行合适的调整和演化[1]。对此，一个重要挑战是：如何对演化后系统各部件之间业务的一致性进行有效的分析和验证，以提高系统在自适应性方面的可信性[2-4]。

文献[2]基于Petri-net对系统实现中的构件功能行为以及构件交互行为进行建模，推导出系统行为。根据行为一致性规则对定义的系统行为进行验证，保证构件演化时系统行为的一致性。文献[3]用基于场景的UML顺序图模型来描述具有多功能的组合系统规约，在网构软件的系统模型层次，使用带资源语义信息的接口自动机对构件的行为进行形式化建模，使用资源接口自动机来描述构件实体的行为，检验组合系统的所有行为是否满足给定的资源约束以及检验指定的系统行为是否满足资源约束。文献[4]提出的动态演化的实施方案是通过对演化命令的检查，保证系统新的集成行为不会偏离原来的系统。

本文基于接口自动机给出了检验网构软件系统演化时的业务一致性验证规则，结合实例给出了验证，并且使用模型检测器Spin对其正确性进行了证明。

2 网构软件自适应与XYZ/ADL

体系结构描述语言XYZ/ADL[5]能够显式地描述构件、连接件以及体系结构配置，并且能够提供相应的工具支持。XYZ/ADL中基本的体系结构概念有构件、连接件、端口、角色、配置、体系结构，以及粘接操作和绑定操作等。其中最基本的概念是构件和连接件。构件与连接件的详细定义参看文献[5]。

图1为一个网上书店系统，包含出版商(Publisher)、读者(Reader)以及书店(Shop)3个部分。

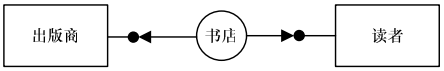


图1 网上书店系统

书店起交易中介的功能，将出版商的商品信息传给读者，将读者的订单传给出版商。将书店建模为连接件，读者以及出版商建模为构件。每个部分将自己的工作流程或者期望的合作对象的业务流程以端口或角色的形式显示。书店所期望的出版商以及读者的行为方式分别建模为各个角色，出版商和读者与书店打交道时的行为方式建模为相应构件的端口。

Shop支持2种工作模式：(1)直接订购，若成功给出ok消息，若失败给出fail消息；(2)在线预定模式，允许顾客先预定，再确认，且允许在线预订过程中因网络延时等原因超时而及时取消本次预订购。其基于XYZ/ADL的系统描述如下：

```
%SYSTEM BookShop==[
%COMPONENT Publisher==[
%PORT Publish==CHAR;[PortBehavior].
%COMPUTATION==[Computation Specification]. ]
%COMPONENT Reader==[
%PORT buy==CHAR; [PortBehavior].
%COMPUTATION==[Computation Specification]. ]
%CONNECTOR Shop==[
```

基金项目：安徽省自然科学基金资助项目(070412058)

作者简介：包书勇(1985－)，男，硕士研究生，主研方向：软件体系结构，分布式计算；王忠群，教授

收稿日期：2011-03-25 **E-mail：**zzqq.wang@gmail.com

```
%ROLE seller==CHAR; [RoleBehavior].
%ROLE buyer==CHAR;[RoleBehavior]
%GLUE==[Interact Protocol] ]
%COMPOSITION==[PB1:Publisher;
                RD1:Reader;Shop: Shop]
%ATTACHMENTS==[PB1.Publish#Shop.seller;
                RD1.buy#Shop.buyer; ]]
```

3 XYZ/ADL 的基于接口自动机的语义解释

用接口自动机[6]来对 XYZ/ADL 描述的系统进行语义解释。构件的端口、计算，连接件的角色和交互协议，以及系统的行为都可以用相应的接口自动机来描述刻画。

3.1 连接件(Connector)的语义

定义 1 具有角色 $R_1, R_2,\cdots,R_n$ 和交互协议 Glue 的连接件所对应的接口自动机为：$Glue||(R_1||R_2||\cdots||R_n)$。其中，Glue、$R_i(i=1,2,\cdots,n)$为交互协议与角色对应的接口自动机。

3.2 连接关系(Attachments)的语义

定义 2 Con 为含有角色 $R_1,R_2,\cdots,R_n$ 和交互协议 Glue 的连接件。具有连接关系 $Con.R_i\#P_i(i=1,2,\cdots,n)$的系统所对应的接口自动机为：$Glue||(P_1:R_1||R_2||\cdots||P_n:R_n)$。其中，$P_i:R_i(i=1,2,\cdots,n)$表示的是用端口 P_i 替换角色 R_i。

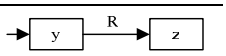
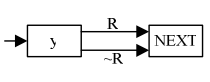
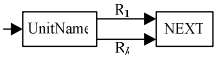
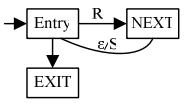
3.3 系统(System)的语义

定义 3 若系统定义了 m 个构件 $Com_1, Com_2,\cdots,Com_m$，构件 Com_i 的计算规范为 $Comp_i$。声明了 n 个连接件 $Con_1, Con_2,\cdots,Con_n$，$\forall Con_i$ 有 k_i 个连接关系(Attachments)，则系统所对应的接口自动机为：$(\|_{i1,2,\cdots,m} Comp_i) \| (\|_{j1,2,\cdots,n} Glue_j)$。

3.4 XYZ/ADL 到接口自动机的转换

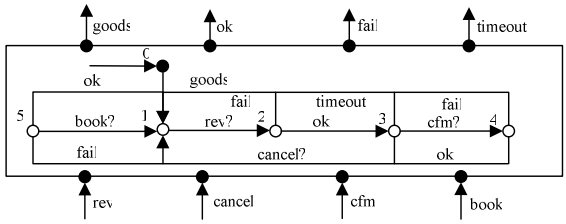
基于 XYZ/ADL 与接口自动机的基本元素，给出两者之间的转换规则，如表 1 所示。

表 1 XYZ/ADL 与接口自动机的关系映射

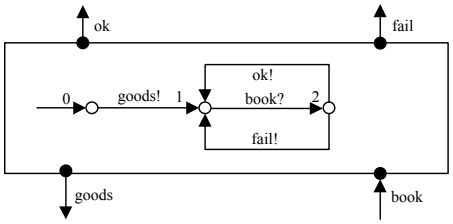
XYZ/基本结构	XYZ/E 语句描述	接口自动机描述
XYZ/BE	LB=y∧R=> $O(Q∧LB=z)	
通道命令：输入	LB=y∧R=> ChNm?x∧$OLB=z	
通道命令：输出	LB=y∧R=> ChNm!x∧$OLB=z	
选择	LB=y∧R=>$O ($Q_1$∧LB=NEXT| Q_2∧LB=NEXT)	
选择元	UnitName:!![R_1|>Q_1, R_2|>Q_2,…,R_k|>Q_k]	
循环	*[LB=Entry∧R=> ($OLB=NEXT\|$O=EXIT); LB=Label{|S|}$OLB=Entry]	

由表 1 所示的转换规则可知，通道命令的输入输出元素即为相应的接口自动机的输入输出接口元素，可将基于 XYZ/ADL 的构件描述转换为对应的接口自动机。接口自动机的状态转换图可由通道的行为描述来进行映射。

对于图 1 所示的网上书店系统，根据上述转换规则，以及构件、连接件、连接的语义解释，连接件的角色 seller，构件 Publisher 的端口 Publish 所对应的接口自动机描述如图 2 所示。其中，图 2 书店系统连接件 Shop 的角色 seller 支持直接订购和预定 2 种模式，且在预订模式中仅支持：在读者在线预订中途，如发生超时事件，则及时发出超时通知，要求立即返回重新进行预订。由图 2(b)可知，出版商 Publisher 只提供直接订购功能。



(a)角色 seller 自动机描述



(b)端口 Publish 自动机描述

图 2 角色 seller 与端口 Publish 的自动机描述

4 构件演化的业务一致性验证

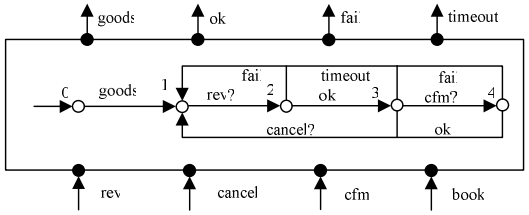
4.1 构件演化中业务不一致问题

在运行一段时间后，客户要求网上书店系统(图 1)提供在线预定图书，故系统必须修改演化，使得构件 Publisher 具备在线提交订单功能，但必须考虑网络延时造成的订单提交的超时问题。对于在线预订图书，如何处理其超时问题存在 2 种方案：

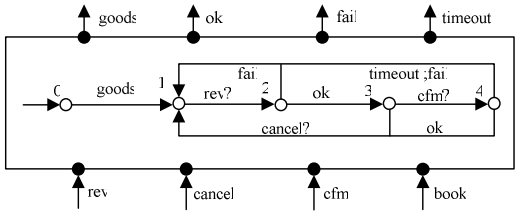
(1)在预订过程中途发生应答超时，则及时发出超时通知 timeout，且取消此次订购要求重新预订。

(2)在中途发生超时事件时，只在最后确认的时候和 fail 消息一起发出此次预订购发生超时现象。

对此，2 种演化方案的 Publish 端口分别如图 3 所示。但是，出版商构件 Publisher 只允许在预订中途发生超时事件，并及时发出超时通知，而非在最后确认的时候和 fail 消息一起发出超时通知。因此，Publisher-2 的业务流程与书店期望的业务流程(图 2(a))不一致。如果采用方案 2，将导致构件与连接件之间业务逻辑不一致。



(a)端口 Publish-1 自动机描述



(b)端口 Publish-2 自动机描述

图 3 端口 Publish-1 与端口 Publish-2 的自动机描述

4.2 构件演化中系统业务一致性验证规则

文献[6]定义了接口自动机上的一种新的精化关系：2/3 精化关系($\preceq_b^{2/3}$)，使得精化关系不仅可以保持相容的性质，还可以保持死锁(Deadlock)的性质。

构件的计算规范(Computation)将各个端口连接为一个整体，形成完整的构件行为描述。因此，计算要与各个端口保持业务上的一致，即所有端口可以正常工作的环境，用该构件的计算替换以后，系统仍能正常工作。

由定义 1，端口和角色之间要满足一致性关系：角色可正常工作的环境，用相应的端口进行替换后系统仍可正常工作。要保证整个组合系统不产生死锁，就要保证每个连接件不产生死锁，体现为交互协议和角色之间的相容性。为此，要保证演化后的系统满足端口与计算、端口与角色的业务一致性要求，下面给出业务一致性的验证规则：

(1)具有计算 Comp 和端口 $P_1,P_2,\cdots,P_n$ 的构件，下列关系成立：$\mathrm{Comp}\preceq_b^{2/3}\mathrm{P}_1\|\mathrm{P}_2\|\cdots\|\mathrm{P}_n$。当此条件满足时，称计算与端口是一致的。

(2)具有连接关系(Attachments)的端口 Port 和角色 Role，下列关系成立：$\mathrm{Port}\preceq_b^{2/3}\mathrm{Role}$。当此条件满足时，称端口与角色是一致的。

(3)对于任意具有角色 $R_1,R_2,\cdots,R_n$ 和交互协议(Glue)的连接件，接口自动机 $\mathrm{Glue}|\,|(\mathrm{R}_1\|\mathrm{R}_2\|\cdots\|\mathrm{R}_n)$ 应该是相容和无死锁的，即任意连接件所对应的接口自动机都应是相容和无死锁的。当此条件满足时，则称该连接件的角色与交互协议是相容的。

如果演化后的系统满足检验规则 1、规则 2 和规则 3，那么在连接件中将角色用相应的端口替换以后所得到的系统 $(\mathrm{C}'=\mathrm{Glue}\|(\mathrm{P}_1:\mathrm{R}_1\|\mathrm{P}_2:\mathrm{R}_2\|\cdots\|\mathrm{P}_n:\mathrm{R}_n))$，以及将这些替换后的所有端口用其所在构件的计算进行替换以后得到的系统 $(\mathrm{C}''=\mathrm{Glue}\|\mathrm{Comp})$ 也是相容且无死锁的。这样系统的业务逻辑则具有一致性。

基于上述检验规则，要保证网上书店实例的演化是可信的，具有业务一致性，需要对以下问题进行检验：

(1)读者、出版商的实际业务流程(即构件的计算)是否与他们的端口描述相一致。

(2)出版商的端口是否与角色 seller 一致，读者的端口是否与角色 buyer 一致。

(3)连接件 shop 是否是相容和无死锁的。

限于篇幅，假定 Reader 的计算与端口是一致的，且端口 buy 与角色 buyer 业务一致，构件 Publish-1、Publish-2 的计算与它们的端口一致。只对角色 seller 与它连接的端口的一致性、连接件 shoP 是否相容和无死锁进行验证。对上述 2 个方案分别进行验证如下：

对于方案 1，其所示的接口自动机与 seller(图 2(a))存在着这样一个 2/3 交替模拟 $\preceq_b^{2/3}=\{(i',i)\,|\,i\in[0,1,\cdots,4]\}$，因此 $\mathrm{publish}-1\preceq_b^{2/3}\mathrm{seller}$。连接件 shop 接口自动机，以及用端口替换角色后，再用计算替换端口而得到的自动机 $\mathrm{Glue}(\mathrm{Comp}_r\|\mathrm{Comp}_p)$，接口自动机 Glue||(seller||buyer) 相容且无死锁，且整个系统自动机也是无死锁的。故方案 1 满足上述检验规则，可实施；对于方案 2， 由于找不到关系 $\preceq_b^{2/3}$，因此 Publish-2 不能 2/3 精化 seller，方案 2 不满足上述检验规则，不能够被实施。

4.3　基于 Spin 的系统检验的证明

为了验证上述检验规则的正确性，用模型检验工具 Spin[7] 对系统模型进行验证。根据已有的转换规则，将 4.1 节给出的 2 种演化方案得到的系统分别进行转换，保存为 Bookshop.abp、Bookshop2.abp。

在 2 个系统中，用 assert(Ak== TimeOut)来验证当 Reader 的发送超时事件时，是否能从 Publisher 那里得到超时通知 TimeOut。使用 Spin 检验得到的验证结果如图 4 所示(限于篇幅，选取部分数据罗列)。

```
State-vector 68 byte, depth reached 24, errors: 0
       39 states, stored
       11 states, matched
       50 transitions (= stored+matched)
        1 atomic steps
hash conflicts:         0 (resolved)
```

(a)Bookshop.abp 检测结果

```
$ spin -a bookshop2.abp
bookshop2.abp:61:2: warning: no newline at end of file
channel name: ch6
        2 msg parameters sent, 1 expected
spin: line  20 "bookshop2.abp", Error: too many pars in send
```

(b)Bookshop2.abp 检测结果

图 4　使用 Spin 的验证结果

从图 4(a)的检测结果可看出，系统没有错误产生，说明了 assert 语句判断的条件在整个模拟过程中是满足的，保证了 Reader 可以在适当的时间接收到 TimeOut 消息，并且系统没有出现无效的循环。

而从图 4(b)中也可看出，Reader 要在最后才同时收到 fail 和 TimeOut 消息，系统由于无法识别这样的消息而产生编译错误，因而这个系统是不合理的。所以，检测结果说明了上述检验规则的正确性。

5　结束语

本文在网构软件的 SA 层次上，基于接口自动机对 ADL 进行了语义解释，并给出了两者之间的转换规则。用接口自动机来描述构件业务行为以及构件间的业务组合。在系统演化时，将基于 XYZ/ADL 描述的演化后的系统由映射规则转换为相应的接口自动机，然后对由接口自动机描述的系统进行一致性规则检测，从而检验演化后的系统是否满足业务一致性要求，以保证系统演化行为的可信性。同时，使用 Spin 对检测规则的正确性进行了验证。但本文工作目前尚缺乏相应的工具支持，下一步将构造相应的分析验证工具，以实现系统业务一致性验证的半自动化。

参考文献

[1] 梅　宏, 黄　罡, 兰　灵, 等. 基于体系结构的网构软件自适应方法[J]. 中国科学 E 辑: 信息科学, 2008, 38(6): 901-920.

[2] 罗　毅, 李兴宇, 关连伟, 等. 构件演化中的系统行为一致性的研究[J]. 计算机科学, 2008, 35(1): 266-270, 300.

[3] 胡　军, 黄志球, 曹　东, 等. 网构软件的资源自适应性的形式化分析与验证[J]. 软件学报, 2008, 19(5): 1186-1200.

[4] 朱　庆, 王小平, 薛小平, 等. 基于构件的网构软件系统动态演化[J]. 计算机工程, 2010, 36(1): 55-57, 60.

[5] 朱雪阳, 唐稚松. 基于时序逻辑的软件体系结构描述语言 XYZ/ADL[J]. 软件学报, 2003, 14(4): 714-720.

[6] Wen Yanjun, Wang Ji, Qi Zhichang. 2/3 Alternating Simulation Between Interface Automata[C]//Proc. of the 7th International Conference on Formal Engineering Methods. Manchester, UK: [s. n.], 2005.

[7] Bell Labs. Spin Main Page[EB/OL]. (2010-05-10). http://www.spinroot.com.

编辑　任吉慧