构件行为协议实时性扩展及相容性验证

贾仰理[1] 张振领[1] 李舟军[2]

(聊城大学计算机学院 聊城 252059)[1] (北京航空航天大学计算机学院 北京 100191)[2]

摘 要 对复杂实时构件系统行为进行形式化描述和相容性验证,可以有效提高系统的正确性、可靠性。分析了学术界和工业界的主流构件模型及常见时间行为的形式化描述方法,对构件行为协议BP(Behavior Protocol)进行了扩展,提出了时间行为协议TBP(Timed Behavior Protocol),分析了构件组合中常见的相容性错误类型,给出了基于时间行为协议的构件组合相容性验证算法。TBP应用简洁、方便、易于验证。结合具体例子给出了应用示例。

关键词 构件,行为协议,时间行为协议,形式化描述,相容性验证

中图法分类号 TP311 **文献标识码** A

Real-time Extension of Component Behavior Protocol and its Compatibility Verification

JIA Yang-li[1] ZHANG Zhen-ling[1] LI Zhou-jun[2]

(School of Computer Science & Technology, Liaocheng University, Liaocheng 252059, China)[1]

(School of Computer Science & Engineering, Beihang University, Beijing 100191, China)[2]

Abstract Formal specification and compatibility verification of complex real-time component systems' behavior can efficiently improve the systems' correctness and reliability. This paper analyzed the mainstream component models using in academia and industry and the common formal specification methods of component timed behavior. Based on the analysis we extended component behavior protocol and presented the timed behavior protocol(TBP) to model components' real-time behavior. Common compatibility error types in component composition were analyzed and the compatibility verification algorithm based on TBP was given. The timed behavior protocol is simple and convenient to apply and verify. An application example was introduced.

Keywords Component, Behavior protocol, Timed behavior protocol, Formal specification, Compatibility verification

1 引言

基于构件的软件开发方法(CBSD)利用已有的构件来组装新的软件系统,可以有效提高软件开发效率,降低开发费用,提高软件的质量和可靠性[1]。因此,构件技术在软件开发中的应用越来越广泛,并逐渐渗透到实时系统的开发中[2,3]。当前,已有越来越多的软构件系统部署应用到航空航天、军事过程控制、指挥通讯、交通管理等实时控制领域。这类生命攸关和安全攸关领域的实时构件系统对正确性、可靠性等可信性质要求较高,一些情况下系统如果无法在要求的时间内完成规定动作,就可能引发巨大的灾难。但是,这类系统往往具有应用规模庞大、复杂性高等特点,系统各组成构件之间往往交互频繁,时序行为复杂,将它们组装到一起时经常出现构件间动态行为不相容、行为协议冲突等各种难以预料的错误。如何有效地对复杂实时构件系统进行描述和验证,尽可能早地发现错误,提高系统的可信性,是我们面临的一个重要问题。

构件技术和形式化描述与验证方法的结合是实现高可信实时构件系统的重要途径。形式化方法是关于在计算系统的开发中进行严格推理的理论、技术和工具,它主要包括形式化描述技术和形式化验证技术。形式化描述使用具有严格数学定义语法和语义的语言刻画软件系统及其性质,具有简明、无二义性、精确清晰等特点。形式化验证是在形式化描述的基础上建立软件系统及其性质的关系,即分析系统是否具有所期望性质的过程。当前,模型检验等形式化验证方法具有自动化程度高、验证速度快、使用方便等特点,已引起广泛关注。越来越多的国内外研究人员正在将形式化描述和检验技术应用到复杂构件系统的设计、分析和验证中[4-6]。

实时构件系统对实时性的特殊要求,使得它在构件模型、系统设计、实现等方面与其它构件不同。本文针对实时构件系统的快速发展及形式化描述和验证的需求,对主流构件模型及各种实时行为形式化描述方法进行了分析,在对SOFA构件模型采用的行为协议[7]进行实时性扩展的基础上提出了时间行为协议TBP,并给出了基于时间行为协议的构件行为相容性验证算法。TBP具有良好的形式化基础,语法语义定义完整,对带时间的并发进程表达清晰简洁,可以在构件系统

到稿日期:2009-10-26 返修日期:2009-12-25 本文受国家自然科学基金项目(90718017,60473057),博士学科点专项科研基金项目(20070006055)资助。

贾仰理(1976—),男,博士,讲师,主要研究方向为形式化方法、高可信软件技术等,E-mail:jyl@cse.buaa.edu.cn;张振领(1977—),女,硕士,讲师,主要研究方向为智能信息处理等;李舟军(1963—),男,博士,博士生导师,主要研究方向为形式化方法、信息安全、智能信息处理等。

的分析设计中提供构件行为实时属性的形式化描述，减少开发人员对系统行为理解上的歧义，并可以为复杂时序行为的形式化验证提供基础。相容性验证提供了形式化手段来对构件行为进行分析，以便尽早发现设计错误，提升系统质量和可靠性。

本文第 2 节介绍了构件模型及时间行为的形式化描述方法；第 3 节介绍了构件行为协议，并对其进行了实时性扩展；第 4 节分析了构件行为的常见相容性错误，给出了基于时间行为协议的构件组合相容性验证算法；第 5 节给出了应用实例；最后总结全文并指出了下一步的工作。

2 构件模型及时间行为的形式化描述

通常，构件行为的形式化描述和验证应以构件模型为基础。构件模型定义构件的本质特征及构件间的关系，是构件技术的核心和基础。只有基于构件模型的行为描述和验证方法才能为大量实际系统的开发应用提供坚实的支持。实时系统的特点要求我们对构件模型进行扩展，在构件模型中提供能够描述构件实时行为的形式化描述方法。

2.1 构件模型及构件行为的形式化描述

构件模型一般从型构层次(Signature Level)和行为层次(Behavior Level)上给出构件的形式化描述信息。型构层次上的描述信息会给出构件服务所对应的方法名、参数类型和返回结果类型等内容。行为层次上的描述信息则主要关注构件接口上提供服务和请求服务之间的合法次序关系，即构件组合、交互行为等内容。

目前，学术界经过多年的努力，在通用构件模型的研究方面取得了一定进展。描述构件的代表性模型主要包括 3C，Wright，Darwin，SOFA，Fractal，Reo 以及青鸟等，而工业界的通用构件模型则包括 OMG 的 CORBA，Microsoft 的 COM/DCOM 和 Sun 的 JavaBean，EJB 等。目前工业界构件模型对构件的描述还停留在语法层次上，而且大多仅仅描述型构层次上构件提供的服务，而构件对外请求的服务则被隐藏在代码的实现细节中，在方法学层面、语义层面以及适应性层面的支撑还有一定的欠缺，普遍缺乏对构件行为的形式化描述。

学术界构件模型如 Wright，Darwin 等给出了构件行为的形式化描述。例如，Wright[8] 运用通信顺序进程(Communicating Sequential Processes，CSP)代数的子集并对其进行了一定的扩充来描述构件系统的交互行为，但缺乏行为组装推导机制，存在描述复杂等特点；北京大学研制的青鸟构件模型[9,10]中使用操作规约、接口等概念对构件行为进行定义，如在操作规约中利用前后置断言的形式给出操作的功能规约，利用入接口和出接口给出构件的对外行为描述；SOFA，Fractal 则利用构件行为协议对构件行为进行描述，使对构件系统行为的验证转化成了对行为协议的验证。行为协议采用类正则(Regular-like)语言表示，并借鉴了 CSP，CCS 中的一些操作，简单有效，对形式化规约和验证支持较好。这些通用构件模型虽然很好地规约了构件自身所应有的功能属性，但是缺乏对实时环境中实时构件所额外附有的时间约束特征的语义描述机制，无法用于实时系统的建模。

为此，许多研究人员对实时构件进行了研究，提出了一系列的实时构件模型，如实时 CORBA，PECOS，Koala，SaveCCM，CAmkES 等。但实时 CORBA 仍然没有构件行为的形式化描述机制，而 PECOS，Koala 等则是面向特定应用领域(消费电子、汽车制造)的实时构件模型，无法作为通用实时构件模型的推广和使用。

一些学者也提出了一些通用实时构件模型，如文献[11]给出了一个用于构建可信实时反应系统(RTRS)的构件模型。构件由构件模版实例化后得到，构件模版由结构和协议两部分组成，结构的框架(frame)部分从黑盒角度定义构件，给出了构件的接口类型、接口上的服务的集合以及每个服务需要的参数的数据类型等信息；结构的体系结构(architecture)部分定义构件的内部层次结构，是接口之间连接的抽象。协议部分定义构件的数据约束、时间约束等特性信息。该构件模型侧重实时反应系统的时间特性和安全特性的描述，用一个 10 元组来描述构件，表述复杂、可操作性差、缺乏实用工具的支持。

当前，虽然对基于构件模型的实时行为的研究相对较少，但已经出现了很多描述时间行为的方法和工具。这些方法和工具虽然没有整合到常见构件模型中去，但可以单独用来对实时系统进行建模和验证。有些文献也把它们看作一种不完善的构件模型[11]。

2.2 时间行为形式化描述方法

我们按建模方法将常见时间行为描述方法分为以下几类。

(1)基于进程代数的方法

进程代数是关于通信并发系统的代数理论的统称，包括 CSP，CCS 和 PI 演算等。这些代数理论都使用通信而不是共享存储作为进程之间相互作用的基本手段，表现出面向分布式系统的特征。为了描述时间系统，一些研究者对进程代数做了时间域上的扩充，提出了 Timed CSP，Timed CCS，ET-LOTOS(Enhanced Timed LOTOS)，Duration Calculus 等来对时间系统行为进行建模。Timed CSP[12] 语言是在 Hoare 的 CSP 基础上加入时间相关操作而形成的一种形式化语言，包含了事件的精确计时，能够对实时的并发系统进行描述。Timed CSP 语义模型从两方面扩展了 CSP：一是采用 CSP 的进程操作符，并在更加具体的、实时层次上重新解释；二是引入延迟、超时和实时中断等特殊操作来精确地描述时间行为。

(2)基于自动机的方法

有限自动机是最为重要的一种形式化描述与验证技术，是很多形式化方法的基础。它直观性强，可实现与其它形式化方法的组合和转换。时间自动机(Timed Automata)[13]是有限自动机的一种扩展，它在状态机的基础上加入了时间的约束机制，用于时间系统的形式建模。它是 R. Alur 和 D. Din 首先提出来的，一直是一种用于描述时间系统的标准模型。一个时间自动机具有有限个“位置”(Location)和多个实型值时钟，所有的时钟同步，记录了上次重置后流逝的时间。由 TA 的边，即位置的转换来重置时钟，因此时钟实际上记录的是相对于重置操作后流逝的时间。

基于自动机的方法还包括使用 Duration 自动机、时间接口自动机、时间 I/O 自动机等来对构件行为进行建模。

(3)与自动机类似的表示方法有时间迁移系统(Timed Transition Systern)[14]。时间迁移系统是对转换加以限制，将每个转换都绑定一个转换上限和下限时间，从而将时间引入系统模型。时间的上下限信息给出了转换发生的时间要求，

用来保证转换不会过早也不会太迟发生。时间迁移系统模型和时间自动机模型可以存在转化关系。

(4)基于时序逻辑的方法

时序逻辑又称时态逻辑、时间逻辑,它是由模态逻辑将时间因素引入而形成的广义模态逻辑。如果一个给定系统的所有计算都是一个时序公式的模型,我们可以说该系统满足该时序性质。时序逻辑可分为两类:线性时序逻辑和分支时序逻辑。线性时序逻辑将时间设想为一个线性的序列 $\delta: s_0, s_1, s_2 \cdots$;分支时序逻辑则允许时间序列出现分支,时间表示可以不断向前或向后分叉;常见的 CTL,CTL * 等均属分支时态逻辑。另外,它们的各种扩充,如 TPTL,TCTL,LTLC 等都可以更好地提供实时信息的描述。

例如,XYZ/E[15]是由中科院软件研究所唐稚松等人提出的基于时序逻辑的语言,该语言能够准确地描述各种复杂系统。对于带有实时要求的复杂系统,XYZ/E 的实时扩展语言——实时时序逻辑语言 XYZ/RE 可以对系统进程和整个实时系统进行准确描述。

关于时间系统的建模语言还包括时间 Petri 网、基于 UML 的时间统一建模语言 TUML(Timed UML)、TCOZ(Timed CSP+Objeet Z)等。但是,鉴于进程代数、时间自动机、Petri 网等形式化方法本身的复杂性,我们迫切需要一种能够融合到构件模型中、语法和语义简单,能方便软件开发人员掌握的实时构件行为形式化描述方法。

3 构件行为协议及其实时性扩展

3.1 行为协议

大多数基于进程代数的构件模型并没有真正大面积地应用到实际的软件开发中,原因在于这些构件模型使用的 CSP,Pi 演算等进程代数形式化手段过于复杂,很难被工业界所接受。SOFA 构件模型则使用行为协议来描述构件各个层次上的行为,行为协议是一种类正则(Regular-like)语言,给出了所描述构件提供(provide)或请求(require)的原子事件(方法调用)的所有可能有限序列。它定义简单、易于阅读和书写,并且支持行为描述的逐步求精,因此一经提出立刻得到了广泛关注。

行为协议由事件标记和操作符组成。事件标记用于表示构件中方法的调用,操作符用于构造复杂的行为协议。事件的表示由接口名、方法名、事件类型组成。事件类型由符号"!","?","τ","↑","↓"表示,其中事件前缀"!"表示(emit)发出事件,"?"表示接收(accept)事件,"τ"表示内部事件;事件后缀"↑"和"↓"分别表示"请求(request)"和"响应(response)"。例如,数据库构件接收调用、依次启动日志和事务处理行为可表示为? db. start↑{! logger. start↑;! tm. init↑}。更多行为协议的知识请参考文献[7]。

3.2 行为协议的实时性扩展

SOFA 构件模型采用的行为协议虽然具有简单、易于掌握等优点,但由于行为协议定义时未考虑时间因素,因此行为协议对时间因素的描述能力先天不足,难以满足对实时系统、嵌入式和分布式系统的描述和验证要求。

在行为协议语法的基础上引入时间概念,对一些事件绑定时间限制属性,并引入与时间相关的操作符。将扩展的行为协议称为时间行为协议 TBP(Timed Behavior Protocol)。TBP 中的时间模型是稠密时间模型,在该模型下,事件的时间是单调递增的无界实数。设 t 是时钟变量,t 的值表示流逝的时间值。通过时钟变量,可测量和比较不同事件发生的时机,并可对时钟变量进行赋值、复位等操作。

3.2.1 带时间的事件标记

在系统形式化描述中,为了方便事件时间属性的详细描述(发送、接收、请求、响应情况的时间属性),我们给出带时间的事件表示形式。常见的带时间的事件具体表示形式和对应含义如下:

! interface. method[t_1]↑ 表示调用方构件在时刻 t_1 内发出接口 interface 上的 method 方法的调用请求;

? interface. method[t_2]↑ 表示接收方构件在时刻 t_2 内接收到接口 interface 上的 method 方法的调用请求;

! interface. method[t_3]↓ 表示接收方构件在时刻 t_3 发出接口 interface 上的 method 方法的调用响应;

? interface. method[t_4]↓ 表示调用方构件在时刻 t_4 内接收到接口 interface 上的 method 方法的调用响应;

由带时间的事件标记表示的行为协议,我们称为时间行为协议。

3.2.2 TBP 操作符语法

最简单的时间行为协议仅包含一个事件,为了表示构件复杂的行为,我们可以定义操作符来构造更复杂的时间行为协议。

设 A,B 为时间行为协议,t 为时间,TBP 语法子集的巴克斯范式定义如下:

$$P ::= \text{Reset}(t) \mid \text{Idle}(t) \mid \text{Stop} \mid \text{Skip} \mid \text{Error} \mid P;Q \mid P + Q \mid P * \mid P \triangleright_t Q \mid P \mid Q \mid P \mid \mid Q \mid P/G \mid P \cap_x Q \mid$$

式中,Reset,Idle 等是特殊类型的事件,Reset(t) 表示时钟变量 t 的复位事件;Idle(t) 表示一段时间的流逝;Stop 表示终止事件;Skip 表示跳过事件;Error 表示错误事件。其它 TBP 操作符的含义简单总结如表 1 所列。

表 1 TBP 操作符对应操作

类别	语法	对应操作
基本操作	A;B	顺序
	A+B	非确定性选择
	A *	循环
	$A \triangleright_t B$	超时
高级操作	A|B	与并行
	A||B	或并行
	A/G	限制
组合操作	$A \cap_x B$	组合

对于系统行为描述中不需要给出时间信息的事件,我们仍然可以使用不带时间的事件标记。

时间行为协议组合、并行等操作符的操作语义与 Timed CSP 类似,参见文献[16]。

4 基于时间行为协议的构件组合相容性验证

相容性验证是对处于同一层次上的具有连接关系的两个或多个组合构件的交互行为进行验证,又称为横向验证,用于发现构件组合中的错误。

4.1 常见相容性错误

为了方便查错,我们首先需要仔细分析构件组合中各种常见的错误,并对这些错误进行归纳分类。

文献[17]给出了SOFA行为协议组合中常见的几种错误类型，我们根据时间行为协议的特点进行了扩充。

(1)无效活动(bad activity)：构件时间行为协议 P 发出请求(调用)行为，而构件时间行为协议 Q 或没有相应的响应行为，或响应行为事件与请求事件不能满足时效性要求。

例如，对于时间行为协议 $P=!\ a[1]\{?\ m[1]\}+!\ b[1]\{?\ n[1]\}$、时间行为协议 $Q=?\ a[1]\{!\ m[1]+!\ n[1]\}+?\ b[1]\{!\ n[1]\}$，当它们在集合$\{a\uparrow, a\downarrow, b\uparrow, b\downarrow, m\uparrow, m\downarrow, n\uparrow, n\downarrow\}$上组合时，时间行为协议 Q 在处理完 a 事件后，发出 n 事件调用请求，P 时间行为协议没有相应的响应行为。

对于时间行为协议 $P=!\ a[2]\{?\ m[1]+!\ b[1]\}$、时间行为协议 $Q=?\ a[1]\{!\ m[1]+?\ b[1]\}$，当它们在集合$\{a\uparrow, a\downarrow, b\uparrow, b\downarrow, m\uparrow, m\downarrow, n\uparrow, n\downarrow\}$上组合时，时间行为协议 Q 要求在1个时间单位内接收到 a 方法调用，而时间行为协议 P 最迟在2个时间单位内发出 a 事件调用请求，这使得请求事件和响应事件因为不满足时限性要求而出现错误。

无效活动错误本质上是一种安全性(Safety)错误。

(2)停止推进：构件时间行为协议 P 和 Q 组合时出现都不能发出事件调用，同时又不能结束的情况，这时时间行为协议进入STOP状态。

例如，对于时间行为协议 $P=?\ m[1];?\ n[1]$、时间行为协议 $Q=!\ m[1];?\ a[1]$，当它们在集合$\{a\uparrow, a\downarrow, m\uparrow, m\downarrow, n\uparrow, n\downarrow\}$上组合时，时间行为协议 Q 在处理完 m 事件后，无法推进下去。

停止推进本质上是一种死锁(Deadlock)错误。

(3)发散(divergence)：时间行为协议 P 或 Q 组合时都可以发出事件请求，但存在无法终止的行为。

例如，时间行为协议 $P=(!\ a[1];?\ b[1])^*$ 和时间行为协议 $Q=?\ a[1];(!\ b[1];?\ a[1])^*$，它们组合后产生的路径为$\langle \tau a[1]\uparrow, \tau a[1]\downarrow, \tau b[1]\uparrow, \tau b[1]\downarrow, \tau a[1]\uparrow, \cdots\rangle$。

发散错误本质上是一种活性(liveness)错误。

(4)冗余：时间行为协议 P 发出事件请求，同时有 Q,R 两个时间行为协议可以接收该请求事件，我们认为这时候产生构件组装冗余错误。

4.2 相容性验证

设 X 为行为事件集合，对于时间行为协议 P 和 Q，定义 $L(P\ \nabla_t\{X\}Q)$ 为动作集合 X 上 P 和 Q 的所有可能的同步交叠路径的集合。同步是指对于每一个 $m\in X$，任意形式为 $?\ m[t]\uparrow$ 或 $?\ m[t]\downarrow$ 的事件都会等待相应的调用事件 $!\ m[t]\uparrow$ 或 $!\ m[t]\downarrow$，并组合为 $\tau m\uparrow$ 或 $\tau m\downarrow$ 的形式。$L(P\ \nabla_t\{X\}Q)$ 形成的路径可能包含前面介绍的几类常见错误。

构件的重要用途是用来组装产生新的复合构件。构件的组装不是构件行为的简单罗列，而是构件之间行为的交互，交互中构件间存在大量行为不相容等错误。因此，构件行为的相容性验证非常必要。

我们给出基于时间行为协议的相容性验证算法思想如下。

算法输入：时间行为协议 P,Q，事件集合 X

算法输出：两个时间行为协议是否相容，存在的错误

(1)读入两个子时间行为协议 P,Q，将事件分别归类 $S_{P,prov}$，$S_{P,req}$，$S_{Q,prov}$，$S_{Q,req}$

(2)生成两个子协议组合可产生的所有迹

(3)if 存在 $tc\in T(c)$ 非终止($|tc|\geqslant 100$)，则输出 divergence 错误

(4)else 遍历两子协议组合可产生的所有迹 $T(c)$

(5)任意 $tc\in T(c)$

(6)while tc 未终止

(7) if 存在迹片断 $tc_1=!\ m[t_1]\uparrow,?\ m[t_2]\uparrow$ 或 $tc_1=!\ m[t_3]\downarrow,?\ m[t_4]\downarrow, m\in X$

(8) then

(9) if $[t_1],[t_2]$ 存在重叠区域

(10) then 组合为 $\tau m[t]\uparrow$ 或 $\tau m[t]\downarrow$

(11) else 输出路径，报时限 badactivity 错误

(12) if 任意 $m\in(S_{P,prov}\cup S_{Q,prov})\wedge m\in X$ 没有组合

(13) then 输出路径，报 bad activity 错误

(14) if 任意 $m\in(S_{P,req}\cup S_{Q,req})\wedge m\in X$ 没有组合

(15) then 输出路径，报 no activity 错误

(16)$tc\leftarrow tc', tc'\in T(c)$ goto(6)

上面的相容性验证算法的基本思路是查找组合迹中是否存在发散行为(迹长度大于等于100)，或有哪些事件发出而没有收到响应、哪些事件发出延迟超出了接收事件要求的延迟等错误状态。如果时间行为协议组合后产生的迹包含错误事件，则可以判断两个构件行为存在相容性错误，不能组合。

5 基于实时行为协议的应用

我们对BP在语法上进行了扩充，增加了与时间相关的操作，解决了行为协议对实时性支持的不足。TBP语言具有良好的形式化基础，表达清晰简洁，易于学习和掌握，适合对实时并发构件系统行为建模。

下面以一个锅炉压力控制系统[18]为例，说明经过实时扩展后行为协议简单而强大的描述能力。该系统由压力感应器、压力监视器、压力控制器、报警器4个子构件组成：压力监视器能够每18ms从压力感应器采集一次压力数据，并在2ms内向控制器发出降温或升温请求。控制器在接收到信息后1s内发出响应信息。系统如果出错，则在1ms内报警。

系统控制流程如图1所示。

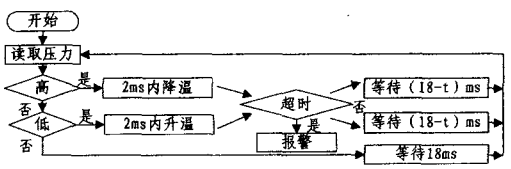

图1 锅炉控制系统流程

各构件行为分别描述如下(为了方便输入，将'↑'和'↓'分别用'~'和'#'表示)：

(1)压力感应器时间行为协议(Pressure_inductor)

? pressure[18]~;! pressure#;

(2)压力监视器时间行为协议(Pressure_monitor)

(! pressure[18]~;? pressure#;

(

! low[20]~▷$_2$(Reset(t);(! alert~;? alert#)[1]);? low[21]#;Idle(18−t)

+! high[20]~▷$_2$(Reset(t);(! alert~;? alert#)[1]);? high[21]#;Idle(18−t)

+Idle(18);

)

)*;

其中，t 为压力监视器向控制器发出降温或升温请求需要的时间，$t<2$。

(3)压力控制器时间行为协议(Pressure_controller)

? low~[20];! low#[21]+? high[20]~;! high#[21];

(4)报警器时间行为协议(Alert)

(? alert^;! alert#)[1]或者表示为? alert[1]。

前两个构件在{pressure}事件集合上组合后,其时间行为协议为:

```
(τpressure[18]^;τpressure#;
(
! low^[20]▷2(Reset(t);(! alert^;? alert#)[1]);? low[21]#;Idle(18-t)
+! high^[20]▷2(Reset(t);(! alert^;? alert#)[1]);? high[21]#;Idle(18-t)
+Idle(18);
)
)*;
```

4个子构件组合成一个构件(系统/子系统),其时间行为协议为:

```
(τpressure[18]^;τpressure#;
(
τlow^[20]▷2(Reset(t);(τalert^;τalert#)[1]);τlow[21]#;Idle(18-t)
+τhigh^[20]▷2(Reset(t);(τalert^;τalert#)[1]);τhigh[21]#;Idle(18-t)
+Idle(18);
)
)*;
```

该锅炉压力控制系统中存在着时间限制等信息,如果不对行为协议进行实时扩展,则无法对系统行为进行有效的描述。在系统形式化描述的基础上,我们可以方便地根据相容性算法检测系统的各种错误。

结束语 本文针对复杂实时构件系统的快速发展及形式化描述和验证的需求,对主流构件模型及各种实时行为形式化描述方法进行了分析,在对SOFA构件模型采用的行为协议进行实时性扩展的基础上提出了时间行为协议TBP并给出了基于时间行为协议的相容性验证算法。TBP具有良好的形式化基础,语法语义定义完整,对带时间的并发进程表达清晰简洁,可以在构件系统的分析设计中提供构件行为实时属性的形式化描述,减少开发人员对系统行为理解上的歧义。基于TBP模型,可以对构件组合的相容性进行验证,本文为此提供了理论基础。我们将进一步开展构件实时行为建模、验证实用工具的开发实现研究。

参 考 文 献

[1] Szyperski C, Gruntz D, Murer S. Component-Software: Beyond Object-oriented Programming(Second Edition)[M]. New York: ACM Press, Addison-Wesley, 2002: 12-18

[2] Alagar V, Mohammad M. A component model for trustworthy real-time reactive systems development[C]// Formal Aspects of Component Software(FACS'07). Sophia-Antipolis, France: ENTCS, Elsevier, Sep. 2007: 1-15

[3] Moller A, Akerholm M, Fredriksson J, et al. Evaluation of component technologies with respect to industrial requirements[C]// Proceedings of the 30th EUROMICRO Conference(EUROMICRO'04). Los Alamitos, CA, USA: IEEE Computer Society, 2004: 56-63

[4] Xie Fei, Browne J C. Verified systems by composition from verified components[J]. ACM SIGSOFT Software Engineering Notes, 2003, 28(5): 277-286

[5] Jezek P, Kofron J, Plasil F. Model Checking of Component Behavior Specification: A Real Life Experience[J]. Electronic Notes in Theoretical Computer Science, 2006, 160: 197-210

[6] Kofron J. Checking Software Component Behavior Using Behavior Protocols and Spin[C]// Proceedings of the 2007 ACM Ssymposium on Applied Computing. New York: ACM Press, 2007: 1513-1517

[7] Plasil F, Visnovsky S. Behavior Protocols for Software Components[J]. IEEE Transactions on Software Engineering, 2002, 28(11): 1056-1076

[8] Allen R, Garlan D. A Formal Basis for Architectural Connection[J]. ACM Trans. Software Eng. and Methodology, 1997, 6(3): 213-249

[9] 潘颖,赵俊峰,谢冰. 构件库技术的研究与发展[J]. 计算机科学, 2003, 30(5): 90-93

[10] 谢冰,杨芙清. 青鸟工程及其Case工具[J]. 计算机工程, 2000, 26(11): 76-78

[11] Alagar V, Mohammad M. A component model for trustworthy real-time reactive systems development[C]// Formal Aspects of Component Software(FACS'07). Sophia-Antipolis, France: ENTCS, Elsevier, Sept 2007: 1-15

[12] Reed G M, Roscoe A W. A timed model for communicating sequential processes[J]. Theoretical Computer Science, 1988, 58(13): 249-261

[13] Alur A W, Dill D L. A theory of timed automata[J]. Theoretical Computer Science, 1994, 126(2): 183-235

[14] Henzinger T A, Manna Z, Pnueli A. Temporal proof methodologies for timed transition systems[J]. Information and Computation, 1994, 112(2): 273-337

[15] 郭亮,唐稚松. 基于XYZ/E描述和验证容错系统[J]. 软件学报, 2002, 13(5): 913-920

[16] Schneider S A. An operational semantics for timed CSP[J]. Information and Computation, 1995, 116(2): 193-213

[17] Jezek P, Kofron J, Frantisek P. Model Checking of Component Behavior Specification: A Real Life Experience[J]. ENTCS, 2006, 160(6): 197-210

[18] 黄靖,卢炎生,徐丽萍. RTCS:一种具有精确语义的实时构件描述机制[J]. 计算机科学, 2005, 32(8): 205-208

(上接第129页)

中去以及如何更好地协调好人机结合问题等,都需要进行长时间的深入研究。量化涉及到大量专业知识和数学知识,本文对其中的部分问题进行了初步研究,其中的大量细节工作还需进一步研究。

参 考 文 献

[1] D'Ambrosio B, et al. Security Situation Assessment and Response Evaluation(SSARE)[C]// DARPA Info. Survivability Conf. & Expo. 2002

[2] D'Ambrosio B, Takikawa M, Upper D. Representation for Dynamic Situation Modeling IET report[R]

[3] Hall D L. Mathematical Techniques in Multisensor Data Fusion[Z]. Artech House, Inc., 1992

[4] Hall D L, Llinas J. An Introduction to Multisensor Data Fusion[C]// Proceedings of the IEEE. vol 85, January 1997

[5] 汪渊,蒋凡,陈国良. 一种基于安全案例推理的网络安全分析方法[J]. 小型微型计算机系统, 2003, 24(12): 2082-2085

[6] 汪渊,蒋凡,陈国良. 基于图论的网络安全分析方法研究与应用[J]. 小型微型计算机系统, 2003, 24(10): 1865-1869