# 基于协议的实时构件行为一致性验证

张振领[1] 贾仰理[1] 谢圣献[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一致性验证,可以提高实时构件的可复用性和系统的正确性、可靠性。分析了时间行为协议TBP(Timed Behavior Protocol)及其它学术界和工业界常用的时序行为形式化描述方法,对实时构件替换理论进行了讨论,给出了基于时间行为协议的构件一致性验证算法并对其进行了分析。

**关键词** 实时构件,时间行为协议,形式化描述,一致性验证
**中图法分类号** TP311 **文献标识码** A

## Protocol Based Real-time Component Behavior Consistency Verification

ZHANG Zhen-ling[1] JIA Yang-li[1] XIE Sheng-xian[1] LI Zhou-jun[2]
(School of Computer, Liaocheng University, Liaocheng 252059, China)[1]
(School of Computer Science & Engineering, Beihang University, Beijing 100191, China)[2]

**Abstract** The formal specification and consistency verification of complex real-time component systems' behavior can efficiently improve the systems' reusability, correctness and reliability. This paper analyzed the timed behavior protocol and the other mainstream formal specification methods of real-time behavior used in academia and industry. Based on the analysis we gave the substitution theory and the consistency verification algorithm based on timed behavior protocol, which can support complex real-time component based systems' development.

**Keywords** Realtime component, Timed behavior protocol, Formal specification, Consistency verification

## 1 引言

信息物理融合系统(Cyber-Physical System,CPS)在航空航天、医疗、交通管理等领域有着广阔的应用前景。这些领域的共同特点是对系统的质量、可靠性等高可信性质要求特别高,系统失效造成的生命、经济损失将不堪设想。因此,文献[1]指出,信息物理融合系统必须满足可靠性和安全性,其行为必须具备可预测性和自适应性,其操作必须具有可靠性和可验证性。在实际应用中,这类系统大多是实时构件系统,具有应用规模庞大、复杂性高等特点,而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会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。从构件开发、选择、组装等环节保障这类复杂实时构件系统在功能和性能上的正确性与可靠性,防止灾难的发生,是当前复杂信息物理融合系统所面临的主要挑战之一。

在这些实时系统的开发、组装中,经常需要从构件库选取构件或开发新构件,这些都需要分析、判断或验证目标构件是否满足要求,即系统构件在语法、语义上是否符合需求,从而使得参与组装的内部子构件之间在语法和语义上相容,避免组装可能出现的错误。另外,在构件系统维护及升级过程中,也涉及构件可替换性问题。这些都需要对实时构件行为的一致性进行验证。

基于形式化描述和验证方法对复杂信息物理融合构件系统的行为进行一致性验证,可以有效地提高系统的可复用性、正确性、可靠性与安全性。形式化描述使用具有严格数学语法和语义定义的语言刻画软硬件系统及其性质,具有简明、无二义性、精确清晰等特点。在形式化描述的基础上,可以进行形式化验证,即建立软件系统及其性质的关系,分析系统是否具有所期望性质,从而对系统的一致性、相容性等各种可信性质进行自动分析和验证。实现自动分析、验证实时构件行为及组合是否存在错误、描述是否一致,在构件组合、精化、构件检索、复用(替代)等方面具有重要应用意义。

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,对时间行为协议TBP及各种实时行为形式化描述方法进行了分析,且对基于时间行为协议的构件可替换性验证理论进行了分析和研究,并给出了基于时间行为协议的构件行为一致性验证算法。

本文第2节介绍了时间行为的形式化描述方法;第3节讨论了基于时间行为协议的实时构件一致性验证理论,给出了基于时间行为协议的构件行为一致性验证算法;最后总结

到稿日期:2011-07-19 返修日期:2011-12-17 本文受国家自然科学基金项目(90718017),山东省自然科学基金项目(ZR2011FL023),山东省软科学项目(2010RKE16007),山东省高校智能信息处理与网络安全重点实验室(聊城大学)资助。
张振领(1977—),女,硕士,讲师,主要研究方向为形式化方法、智能信息处理,E-mail:zhangzhenling@lcu.edu.cn;贾仰理(1976—),男,博士,主要研究方向为形式化方法等;谢圣献(1957—),男,教授,主要研究方向为网络与信息安全等;李舟军(1963—),男,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。

全文并指出了下一步的工作。

## 2 时间行为的形式化描述

### 2.1 时间行为协议

文献[2]利用行为协议来描述复杂构件各个层次上的行为，给出了所描述构件提供(provide)或请求(require)的原子事件的(方法调用)的所有可能序列。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"!"表示发出(emit)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 ↑}。行为协议定义简单，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，即得到了广泛关注。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。

文献[3]在 SOFA 构件模型行为协议语法的基础上，提出了时间行为协议 TBP(Timed Behavior Protocol)的概念。TBP 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。在该模型下，事件的时间是单调递增的无界实数。通过设置时钟变量，可分析与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。

时间行为协议引入了时间事件的概念来描述实时构件发送、接收、请求、响应情况的时间属性，文献[3]给出了其具体表示形式和对应的含义。另外，简单的时间行为协议可以仅包含一个事件。为了表示构件复杂的行为，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：

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

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。更多时间行为协议的知识参考文献[3]。

### 2.2 其他时间行为描述方法

常见的时间行为描述方法包括基于进程代数的方法[4-6]、基于自动机的方法[7-9]、基于 Petri 网的方法[10]和基于逻辑的方法[11-13]等。例如，基于进程代数的方法对进程代数(CSP，CCS 和 PI 演算)做了时间域上的扩充，使用 Timed CSP、Hybrid CSP、HyPA 等来对时间系统行为进行了建模。基于自动机的方法则是在有限自动机基础上加入时间的约束机制，如时间自动机(Timed Automata)、Duration 自动机、时间接口自动机、时间 I/O 自动机等。

与时间行为协议相比，这些时间系统行为的形式化建模语言复杂、软件开发人员难以掌握，从而影响了它们的推广和使用。时间行为协议能够无缝融合到构件模型，并且具有语法和语义简单、能方便软件开发人员掌握的特点。

## 3 基于时间行为协议的实时构件一致性验证

文献[3]详细介绍了时间行为协议的相关概念，对基于时间行为协议的实时构件的相容性验证进行了研究。本文在此基础上，探讨了基于时间行为协议的实时构件的一致性验证问题。这是因为在实际应用中，经常需要对构件行为进行比较分析，判断待用构件是否在行为上和查找或需求的构件一致，即进行构件行为的一致性分析。另外，有时需要分析不同层次上时间行为协议之间的一致性。例如，在 SOFA 构件模型中自底向上分别描述了接口行为协议、Architecture 行为协议、Frame 行为协议。在构件进化中，经常需要对处于不同层次上的构件行为是否一致进行分析和验证。

### 3.1 时间行为协议的一致性

构件的一致性也分为两个层次：型构一致性和行为一致性。型构一致性是指两个构件所发送/接受的消息的参数类型、个数完全匹配。在传统构件替换方法中，往往通过检查构件提供服务和请求服务的关系来判断构件的型构一致性。例如，如果构件 $A$ 提供的服务 $Pro_A$ 是构件 $B$ 提供的服务 $Pro_B$ 的子类型，而构件 $B$ 请求服务 $Req_B$ 是构件 $A$ 提供服务 $Req_A$ 的子类型，那么构件 $A$ 可替换构件 $B$。因为没有考虑构件提供服务和请求服务的调用次序，即没有考虑构件服务调用时体现出来的行为，这种使用型构层次接口的子类型关系来判断构件可替换性有着很大的局限性，并不能保证构件在运行时刻安全地替换另外一个构件[14]，因此构件行为的可替换性判定非常必要。

对于构件行为的一致性定义，可以借助时间行为协议的一致性定义给出。

文献[2]通过比较构件行为的符号集和语言来判断构件的一致性。构件 $A$ 可以替换构件 $B$，必须满足以下定义。

构件可替代性判定定义：

如果构件 $A,B$ 的行为协议满足下面的要求，则 $A$ 构件可以替换 $B$ 构件。

(1) $S_{B,prov}\subseteq S_{A,prov}$，$S_{A,req}\subseteq S_{B,req}$

(2) $L_B/S_{B,prov}\subseteq L_A/S_{A,prov}$

(3) $L_B/S_{B,prov}\mid S_{A,prov}\mid L_A/S_{A,ext}\subseteq L_B/S_{B,ext}$

其中，$L$ 为语言，$S$ 为符号表，/为限制算子，$|S|$ 为调整算子，详见文献[2]。

在上述第一个条件中，要求 $A$ 构件提供的服务(方法)至少和 $B$ 一样多，需求的服务也至多和 $B$ 一样多，即 $A$ 尽可能提供的方法比 $B$ 多，而需求的方法比 $B$ 少。

仅仅满足这一条件并不能判定 $A$ 构件可以替换 $B$ 构件，例如：

时间行为协议 $P$=? $a[t_1][t_2]$;? $b[t_3][t_4]$

时间行为协议 $Q$=? $b[t_3][t_4]$;! $a[t_1][t_2]$

$P$ 和 $Q$ 在事件集$\{a,b\}$上提供和需求的事件是一样的，但要求的执行顺序不同，显然不能相互替换。

因此，还需要满足后面的条件。

第二个条件要求在各自的提供事件集合上，$A$ 至少提供那些 $B$ 能够提供的行为迹。

最后一个条件要求 $A$ 的外部行为在其像 $B$ 一样提供服务时，其行为和 $B$ 的外部行为一样。

文献[2]给出的例子如下：

两个构件 $A$ 和 $B$，构件 $A$ 的行为协议产生行为迹集合(语言)如下：

$L(A)=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

在环境 $E$ 下，构件 $B$ 行为协议产生行为迹集合(语言)如下：

$L(B)=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

显然，第一个条件满足。

对于第二个条件，

$L_A/S_{A,prov}=L_B/S_{B,prov}=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

第三个条件，

$L_B/S_{B,prov}\mid S_{A,prov}\mid L_A/S_{A,ext}=L(A)$

$L_B/S_{B,ext}=L(B)$

显然也满足。因此按照上述判定方法，构件 $A$ 可以替换构件 $B$。

再看一个例子：

$L(A)=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle, \langle ?\ dbAcc.\ Insert\uparrow\rangle\}$

在环境 $E$ 下构件 $B$ 行为协议的行为迹如下：

$L(B)=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle, \langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

$A,B$ 的符号表分别如下：

$S_{A,prov}=\{?\ dbSrv.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow, ?\ dbAcc.\ Insert\uparrow\}$

$S_{B,prov}=\{?\ dbSrv.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\}$

$S_{A,req}=\{!\ dbAcc.\ Query\uparrow\}$

$S_{B,req}=\{!\ dbAcc.\ Query\uparrow, !\ dbAcc.\ Insert\uparrow\}$

则显然，第一个条件 $S_{B,prov}\subseteq S_{A,prov}, S_{A,req}\subseteq S_{B,req}$ 满足。

第二个条件，

$L_A/S_{A,prov}=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle, \langle ?\ dbAcc.\ Insert\uparrow\rangle\}$

$L_B/S_{B,prov}=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

显然，$L_B/S_{B,prov}\subseteq L_A/S_{A,prov}$，因此第二个条件也满足。

第三个条件，

$L_B/S_{B,prov}\mid S_{A,prov}\mid L_A/S_{A,ext}=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

$L_B/S_{B,ext}=\{\langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Query\uparrow, !\ dbSrv.\ Insert\downarrow\rangle, \langle ?\ dbSrv.\ Insert\uparrow, !\ dbAcc.\ Insert\uparrow, !\ dbSrv.\ Insert\downarrow\rangle\}$

显然也满足。因此按照上面定理的判定方法，构件 $A$ 可以替换构件 $B$。

但这显然会产生错误。因 $B$ 构件的请求事件 $!\ dbAcc.\ Insert\uparrow$ 对应的应答事件 $?\ dbAcc.\ Insert\uparrow$ 应该在其环境 $E$ 中存在，$A$ 构件提供的响应事件明显会造成冗余冲突。

因此，必须改进上述定理。

首先，给出补事件的概念。

**定义 1**(补事件) 事件 $a$ 的前缀"!"换为"?"或"?"换为"!"，则称 $!\ a[t]$ 和 $?\ a[t]$ 互为补事件，记作 $R(!\ a[t])=?\ a[t], R(?\ a[t])=!\ a[t]$。特别，对于内部动作事件 $\tau, R(\tau)=\tau$。

**定义 2**(构件可替代性判定) 如果构件 $A,B$ 的时间行为协议满足下面的要求，则 $A$ 构件可以替换 $B$ 构件。

(1) $S_{B,prov}\subseteq S_{A,prov}, S_{A,req}\subseteq S_{B,req}$

(2) $L_B/S_{B,prov}\subseteq L_A/S_{A,prov}$

(3) $L_B/S_{B,prov}\mid S_{A,prov}\mid L_A/S_{A,ext}\subseteq L_B/S_{B,ext}$

(4) $S_{A,prov}\cap R(S_{B,req})=\emptyset$，或 $S_{B,req}\cap R(S_{A,prov})=\emptyset$

如果将两个时间行为协议中对应的时间事件中的时间信息作为事件的参数，放在型构层次上进行一致性验证，即不在行为层次上验证时间信息，则上面的改进验证理论直接可以用于时间行为协议的一致性验证。

### 3.2 一致性验证

一致性验证可以用于验证构件组装或维护中一个构件能否替换另一个构件。另外，在构件精化设计中经常需要对构件行为逐步求精，并分析不同层次构件行为的一致性。

我们可以根据两个时间行为协议形成的迹的关系来判断它们的可替代性关系。

首先给出下面的相关定义。

**定义 3**(时间行为协议语义等价) 设 $A$ 为事件集合，$P$ 和 $Q$ 为时间行为协议，如果两个时间行为协议产生的语言相同，则这两个时间行为协议在事件集 $A$ 上语义等价。

下面给出时间行为协议的补的概念。

**定义 4**(时间行为协议的补) 将一个时间行为协议中所有的"!"换为"?"，所有的"?"换作"!"(即动作取补)，则形成的时间行为协议称为原时间行为协议的补。

例如：

对于一个 DBServer 构件的 Frame 定义，包括其时间行为协议如下：

```
frame DBServer{
provides:
IDatabase db;
protocol:{
? db. start[1];(? db. add ‖ ? db. get ‖ ? db. remove)* ;? db. stop}
};
```

其时间行为协议取补则为：

```
protocol:{
! db. start[1];(! db. add ‖ ! db. get ‖ ! db. remove)* ;! db. stop}
```

**定义 5**(时间行为协议的互补关系) 若在事件集 $A$ 上，时间行为协议 $P$ 产生的所有行为迹和 $Q$ 产生的所有行为迹都为互补的行为迹，则称 $P$ 和 $Q$ 在事件集 $A$ 上为互补的时间行为协议。

显然，时间行为协议 $P$ 及其补协议为互补的时间行为协议。

时间行为协议互补关系的意义：时间行为协议的补可以看作一个新的时间行为协议，它代表了原时间行为协议的外

围环境的行为。如图1所示，$Q$为时间行为协议$P$的补，它的输入接口为$P$的输出接口，输出接口是$P$的输入接口，接口上相应方法的性质(emitted、accepted)亦相反。

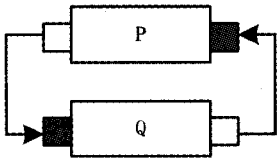


图1　时间行为协议的补图示

利用时间行为协议的补和互补时间行为协议来定义时间行为协议的一致性。

**定义6**(时间行为协议一致性)　对于时间行为协议$P$和$Q$，如果在事件集$A$上，$P/Q$和$Q/P$的补为互补关系，则称在$A$上$P$和$Q$行为一致。

**定义7**(时间行为协议一致性判定)　对于时间行为协议$P$和$Q$，如果在事件集$A$上($A=S_{P,prov}\cup S_{P,req}\cup S_{Q,req}\cup S_{Q,prov}$)$P/Q$和$R(Q)/R(P)$的所有可能的同步交叠路径$L(P\nabla_t\{X\}R(Q))$及$L(Q\nabla_t\{X\}R(P))$都不包含错误事件，则称在$A$上$P$和$Q$满足一致性。

这样，在文献[3]相容性判定算法的基础上，给出一致性验证算法。

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一致、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)对协议P和Q分别取补，按下面的步骤验证P和R(Q)、Q和R(P)的相容性

(3)生成P和R(Q)及Q和R(P)组合可产生的所有迹

(4)l<－true;

(5)while l==true;

(6) if 存在tc∈T(c)非终止(|tc|≥100)，则输出divergence错误，l<-false;

(7) else 遍历两子协议组合可产生的所有迹T(c)，

(8)　任意tc∈T(c)

(9)　while tc未终止

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

(11)　then

(12)　if $[t_1]$，$[t_2]$存在重叠区域

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

(14)　else 输出路径，报时限badactivity错误，l<－false;

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

(16)　then 输出路径，报bad activity错误，l<－false;

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

(18)　then 输出路径，报no activity错误，l<－false;

(19)tc←tc′，$tc'\in T(c)$ goto(9)

(20)if l==false then 输出 no consistency

else 输出 consistency

对于时间行为协议的一致性验证算法，首先需要将其中一个时间行为协议$P$或$Q$取补，然后逐一地检验该时间行为协议的补$R(P)$或$R(Q)$和另一协议$Q$或$P$的组合产生的所有行为迹。如果行为存在divergence、badactivity或者no activity错误，则认为两协议行为不满足一致性。

如果时间行为协议组合后产生的迹不包含任何错误事件，则可以判断$P$和$R(Q)$及$Q$和$R(P)$代表的两个构件行为相容，$P$和$Q$满足一致性；否则给出错误信息，基于此信息可以得出两个协议不满足一致性的原因。

可以用构件行为一致性判定算法查找符合语义要求的构件，或者用于判定是否可以用一个构件去替换另一构件。

利用时间行为协议的一致性的判定定义，给出构件行为一致性的概念。

**定义8**(构件行为一致性的判定)　设两个构件$C1$和$C2$，它们的时间行为协议分别为$P$和$Q$，在事件集$A$($A=S_{P,prov}\cup S_{P,req}\cup S_{Q,req}\cup S_{Q,prov}$)上，如果$P$、$Q$满足一致性，则在$A$上构件$C1$和构件$C2$满足一致性。

显然，一致性关系是自反的。对于任何时间行为协议$P$，$L(P\nabla_t R(P))$的所有行为迹显然不会包含任何错误信息。

## 4　时间行为协议一致性验证算法的应用

下面以一个数据库实时构件行为的可替换性判定为例，简要说明时间行为协议一致性验证算法的应用。对于一个数据库服务构件$C1$的Frame定义，其时间行为协议如下：

$P=?\,db.add\{!\,tm.begin[t_1]\uparrow;!\,tm.commit\uparrow\}$

另一数据库服务构件$C2$的Frame时间行为协议记为

$Q=?\,db.add\{!\,tm.begin[t_1]\uparrow;(!\,tm.commit\uparrow+!\,tm.rollback\uparrow)\}$

按照一致性验证算法，对时间行为协议$P$取补为

$R(P)=!\,db.add\{?\,tm.begin[t_1]\uparrow;?\,tm.commit\uparrow\}$

对时间协议$Q$取补则为

$R(Q)=!\,db.add\{?\,tm.begin[t_1]\uparrow;(?\,tm.commit\uparrow+?\,tm.rollback\uparrow)\}$

计算$P$和$R(Q)$及$Q$和$R(P)$组合可产生的所有行为迹，可用语言分别描述为

$L(P\nabla_t\{X\}R(Q)):\tau db.add\{\tau tm.begin[t_1]\uparrow;\tau tm.commit\uparrow\}$

$L(Q\nabla_t\{X\}R(P)):\tau db.add\{\tau tm.begin[t_1]\uparrow;(\tau tm.commit\uparrow+badactivity)\}$

显然，$P$和$R(Q)$组合不会出现错误信息，而$Q$和$R(P)$组合会报出badactivity错误。这是因为当构件$C2$按时间行为协议$Q$发出事务回滚要求时，$R(P)$给出的行为无法响应该请求。

分析构件的可替换性。显然，构件$C2$(行为协议为$Q$)可以替换构件$C1$(行为协议为$P$)，反之则不行。

**结束语**　以实时构件系统为代表的各种信息物理融合系统在各种高可靠需求领域发展迅速，对这些系统进行形式化描述和验证具有重要意义。分析了时间行为协议TBP及其它学术界和工业界常用的时序行为的形式化描述方法，给出了基于时间行为协议的构件一致性验证算法，并对其进行了分析，在此基础上讨论了实时构件替换理论。在今后工作中，将进一步开展构件实时行为建模、验证实用工具的开发。

## 参考文献

[1]　李建中. 信息物理融合系统(CPS)的概念、特点、挑战和研究进展[C]//2009年中国计算机科学技术发展报告. 2010:1-17

(下转第142页)

Grasshopper[13]构建系统的各个 Agent，电影领域本体采用已有的电影本体[14]。

实验过程中，系统对 236 个注册用户使用 968 部电影服务进行个性化主动服务，分别测试服务检索和主动推送性能。服务检索性能主要通过查全率和查准率两个指标进行刻画；主动推送性能主要通过推送多样性来评价，即对任意两个用户，计算系统推荐的 20 个电影服务中不相同服务的数量，其除以 20 得到每两个用户之间的推送多样性，最后计算所有两两用户多样性的均值。实验结果如图 5 所示。

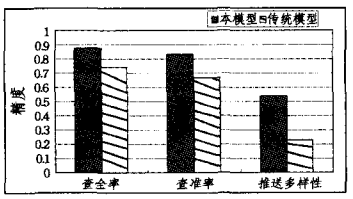


图 5　实验结果对比图

从实验结果可以得到：①针对服务检索，本模型的服务查全率和查准率与传统的主动服务结果相比有明显的提高，检索服务质量明显提升；②针对主动推送，本模型的推送多样性比传统服务的两倍还高，说明模型对个性化服务的质量有显著提升。

**结束语**　Web 服务的前进方向就是提供智能的主动服务和个性化服务。本文针对经典主动服务模型的缺陷，提出了基于本体和多 Agent 的个性化主动服务模型，引入领域本体和个性化 Agent 集，在推与拉的协作运行机制下，力图给用户提供方便快捷的个性化主动服务。原型系统的实验结果表明，本模型能够提供良好的服务性能。在下一步的研究工作中，将重点对模型运行中的服务定制算法做进一步研究，并进行更多的性能测试。

## 参 考 文 献


[1] Gauch S, Chaffee J, Pretschner A. Ontology-based personalized search and browsing[J]. Web Intelligence and Agent System, 2003, 1(3/4): 219-234

[2] 李家清. 个性化信息服务方式与策略研究[J]. 现代情报，2006，9：45-48

[3] 王汝传，徐小龙，黄海平. 智能 Agent 及其在信息网络中应用[M]. 北京：北京邮电大学出版社，2006

[4] 徐小龙，赵昌耀，耿卫健，等. 一种基于智能 Agent 的科技文献快速协作推送机制[J]. 计算机科学，2011，38(4)：249-253

[5] 王艳侠，董东，康振科. 基于 CBR 和 MADM 的多 Agent 推荐系统[J]. 计算机工程与设计，2011，32(1)：157-161

[6] Vedran P, Maja M, Ignac L, et al. Agent-based framework for personalized service provisioning in converged IP Networks[J]. Lecture Notes in Computer Science, 2009, 59(07): 83-94

[7] 张尧学，方存好. 主动服务：概念、结构与实现[M]. 北京：科学出版社，2005

[8] Ganesan P, Garcua-molina H, Widom J. Exploiting hierarchical domain structure to compute similarity[J]. ACM Transaction on System, 2003, 21(1): 64-93

[9] Anderson J R. A spreading activation theory of memory[J]. Journal of Verbal Learning and Verbal Behavior, 1983, 22: 261-295

[10] 陈文宇，张忠全，向涛，等. 基于相似度的语义 Web 服务发现技术研究[J]. 电子科技大学学报，2010，39(6)：896-899

[11] Dong Wen-li, Hu Jian-hua. Test Method for BEPL-Based Web Service Composition Based on Data Flow Analysis[J]. Journal of Software, 2009, 20(8): 2102-2112

[12] 余正涛，宋丽哲，樊孝忠. 基于本体的个性化领域信息服务[J]. 计算机工程，2005，31(5)：22-24

[13] IKV++. Grasshopper[EB/OL]. http://www.Grasshopper.de/, 2010-01-16

[14] Yevs R, Samer A, Mark S, et al. The Music Ontology[C]// Proceedings of the International Conference on Music Information Retrieval(ICMIR). 2007


(上接第 128 页)


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

[3] 贾仰理，张振领，李舟军. 构件行为协议实时性扩展及相容性验证[J]. 计算机科学，2010，37(10)：143-1147

[4] Reed G M, Roscoe A W. A timed model for communicating sequential processes[J]. Lecture Notes in Computer Science Automata Languages and Programming, 1986, 22(6): 314-32

[5] He Ji-feng. From CSP to Hybrid Systems[C]// Roscoe A W, ed. A Classical Mind, Essays in Honour of C. A. R. Hoare, International Series in Computer Science. Prentice Hall, 1994: 171-189

[6] Bergstra J A, Middleburg C A. Process algebra for Hybrid Systems[J]. Theor. Comput. Sci., 2005, 335(2/3): 215-280

[7] Alur R, Dill D L. A theory of timed automata[J]. Theory Computer Science, 1994, 12(6): 183-235

[8] Kaynar D K, Lynch N, Segala R, et al. The Theory of Timed I/O Automata[R]. Cambridge MA: MIT Laboratory for Computer Science, November 2004

[9] Lynch N, Segala R, Vaandrager F. Hybrid I/O automata[J]. Information and Computation, 2003, 185(1): 105-157

[10] Thacker R A, Jones K R, Myers C J, et al. Automatic Abstraction for Verification of Cyber-physical Systems[C]// Proc of ICCPS 10. 2010: 12-21

[11] 李晓山，周巢尘. 时段演算综述[J]. 计算机学报，1994，17(11)：842-850

[12] 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

[13] Wang Han-bo, Zhou Xing-she, Dong Yun-wei, et al. Modeling timing behavior for cyber-physical systems[C]// Proc of Int Conf on Computational Intelligence and S oftware Engineering. Piscataway, NJ: IEEE. 2009: 1-4

[14] 陈波. 基于软件体系结构的构件模型和语言研究[D]. 长沙：国防科技大学，2007