# 基于 Petri 的分布式实时嵌入式软件合理性分析*)

陈丽琼　邵志清　王秀英　范贵生

(华东理工大学计算机科学与工程系　上海 200237)

**摘　要**　合理的模型是保证分布式实时嵌入式(DRE)软件可靠性的关键。提出了分析 DRE 软件模型的合理性方法。该方法基于带抑制弧的时间 Petri 网(ITPN),采用自顶向下的策略对功能模块及其通信过程分别建模,并利用 Petri 网的合成运算形成整个应用的 ITPN 模型。在确保系统实时性的前提下,给出软件模型合理性的形式化定义及其判定定理。最后以实例说明该方法的可行性。

**关键词**　分布式实时嵌入式软件,Petri 网,建模,合理性,验证

## Soundness Analysis of Distributed Real-time and Embedded Software Based on Petri Nets

CHEN Li-qiong　SHAO Zhi-qing　WANG Xiu-ying　FAN Gui-sheng

(Department of Computer Science and Engineering, East China University of Science and Technology, Shanghai 200237, China)

**Abstract**　The soundness of the model is the key issue to guarantee the reliability of distributed real-time embedded (DRE) software. Reasonable determination method of DRE software model is given in this paper. The method is based on time Petri nets with inhibitor arc, using top-down strategy to respectively model functional modules and the communication process. Exploiting the synthesis operations of Petri nets to form the ITPN model of system. Under the premise to meet real-time property, the formal definition and judgment theorem of soundness of distributed real-time embedded software are presented. Finally, an example is given to explain the feasibility of the method.

**Keywords**　Distributed real-time embedded software, Petri nets, Model, Soundness, Verification

## 1　引言

随着计算机技术和网络技术的高速发展,分布式实时嵌入式(Distributed Real-time Embedded,DRE)系统已经被逐步应用于各个领域[1]。典型的应用包括飞行器和航天器的控制系统、汽车电子系统、工业过程控制系统、电站控制系统、危重病人的生命维持系统以及通信系统等关键任务、高安全性领域[2]。而基于网络技术的分布式系统具有有效的资源共享、优越的性能价格比、良好的可伸缩性及支持系统容错等显著优点。但是,如果无法满足 DRE 系统设计要求的逻辑正确性和时间正确性,将造成重大的财产损失,甚至人身伤害,导致灾难性的后果[3]。因此,如何保证分布式实时嵌入式系统的可靠性已成为目前 DRE 系统设计的迫切需要和新的挑战[4]。

目前,分布式实时嵌入式软件设计的理论、方法和技术已经取得了一些有价值的成果。文献[5]采用维也纳分析方法(Vienna Development Method,VDM)的时间扩展 VDM++对 DRE 软件进行规约,最后运用 VDM 验证工具来验证系统的性质。但是,与其它形式化方法相比而言,VDM 对于开发人员可能较难理解和掌握。文献[6]针对分布式非抢占式实时嵌入式系统,基于时间自动机(Timed Automata,TA)提出一个形式化验证框架。由于在 TA 模型中隐含了全局时钟的存在,不适于对分布式系统的建模。文献[7]针对如何减少 DRE 系统使用空间问题,基于 Petri 网提出了用双层设计结构来满足系统的服务质量 QoS。但是该文章从组件配置的角度出发,没有考虑软件模型。文献[8]基于模型驱动的方法,提出一种树层服务器/客户端模型,利用该模型生成自适应的 QoS。而模型驱动方法本身还不成熟。文献[9]分析了 CORBA 构件模型(CORBA Component Model,CCM)的总体构架,提出了一种支持分布式实时嵌入式软件开发的构件模型 Z-CCM,但是该方法没有分析系统的结构合理性。文献[10]提出多层分布式资源管理框架,并分析带宽分配和处理器使用率问题。该方法没有考虑显式考虑系统的实时性。

但是这些研究都没有涉及软件模型的合理性分析,实际上有必要在验证之前保证模型的合理性,避免在后期发现错误而造成更大的损失。本文针对 DRE 软件的通信机制,提出用带抑制弧的时间 Petri 网(Time Petri Nets with Inhibitor Arc,ITPN)来描述 DRE 软件模型;通过引入抑制弧扩展时间 Petri 网的模拟能力,全面刻画 DRE 模块之间的通信机制;将各个组成部分和通信过程的 ITPN 模型组合成整个应用的模型;最后结合时间 Petri 网的状态类图技术给出了软件模型合理性的形式化定义,以分析和判定 DRE 软件模型的合理性。

本文结构如下:第 2 节给出相关工作;第 3 节详细说明 DRE 软件的 ITPN 模型的构建过程;第 4 节分析软件模型的正确性;第 5 节实例分析;最后是结论和下一步的研究方向。

## 2　ITPN 模型

带抑制弧的时间 Petri 网是通过在时间 Petri 网中引入抑制弧,来控制或限定 Petri 网的变迁引发序列,描述系统事件

*)本课题得到国家自然科学基金(60373075)、上海市科技发展基金(06dz15004-1)的资助。**陈丽琼**　博士研究生,主要研究领域为软件形式化、Petri 网应用、嵌入式软件设计;**邵志清**　教授,博士生导师,主要研究领域为软件开发与验证方法;**王秀英**　博士研究生,主要研究领域为入侵检测技术;**范贵生**　博士研究生,主要研究领域为 Petri 网应用、Web 服务。

之间的优先关系。ITPN 作为时间 Petri 网的子类，是一种直观的图形建模工具和一种具有丰富数学基础的形式化模型，非常适合描述具有并发、异步和分布式特征的系统。

**定义 1** 带抑制弧的时间 Petri 网 $\Sigma$ 是一个六元组 $(P, T, F, I_h, M_0, R^s)$，其中：

(1) $P=\{p_1, p_2, \cdots, p_n\}$ 是一个有限库所集，$n\geqslant 0$；

(2) $T=\{t_1, t_2, \cdots, t_m\}$ 是一个有限的变迁集，$m\geqslant 0$，并且 $P\cup T\neq\Phi$，$P\cap T=\Phi$；

(3) $F:(P\times T)\cup(T\times P)\rightarrow N^*$，$N^*$ 为非负整数集，$F$ 称为关联函数；

(4) $I_h$ 是抑制弧集合：$I_h\subset P\times T$，且 $I_h\cup F=\Phi$；

(5) $M_0: P\rightarrow N^*$，$M_0$ 称为初始标识；

(6) $R^s: T\rightarrow N^*\times(N^*(\infty)$ 是变迁的相对触发间隔，$R^s(T_i)=[\alpha_i^s, \beta_i^s]$，其中 $\alpha_i^s$，$\beta_i^s$ 分别表示变迁 $T_i$ 的最早触发时间和最迟触发时间。若 $R^s(T_i)=[0,0]$，则称 $T_i$ 为瞬间变迁。

在 DRE 软件建模中，软件的功能由一系列的子任务关联而成，这些子任务映射为 ITPN 的变迁；任务间消息的传递由库所及其中的令牌来表征；某时刻各库所中令牌的分布状况称为 ITPN 的标识，在标识 $M$ 下，库所 $p$ 中的令牌数量记为 $M(p)$；变迁的最早触发时间和最晚触发时间则分别表示 *DRE* 中任务的释放时间和截止时限；任意 $x\in(P\cup T)$，集合 $\cdot x=\{y\mid y\in(P\cup T)\wedge(y,x)\in F\}$ 和 $x^{\cdot}=\{y\mid y\in(P\cup T)\wedge(x,y)\in F\}$ 分别对应于 $x$ 的输入和输出；标识和变迁的有效触发域构成系统的状态。

**定义 2** 设 $\Sigma=(P, T, F, I_h, M_0, R^s)$ 为带抑制弧的时间 Petri 网，$ST=(M, I)$ 为 $\Sigma$ 的一个状态，$\Sigma$ 有如下触发规则：

(1) 对于变迁 $t_i\in T$，如果 $\forall p\in P$，使得 $M(p)\geqslant F(p,t_i)$，$(p,t)\notin I_h$ 且 $M(p)=0$，$(p,t)\in I_h$，则称 $t_i$ 在标识 $M$ 下有发生权（记作 $M[t_i>$），所有在标识 $M$ 下有发生权的变迁的集合记为 $ET(M)$。

(2) 对于变迁 $t_i\in ET(M)$，$T$ 表示进入状态 $ST$ 的时刻，$t_i$ 在 $T+\theta$ 时刻触发，如果对于 $\forall t_k\in ET(M)$，有 $\alpha_i\leqslant\theta\leqslant\min(\beta_k)$，则 $t_i$ 的触发称为有效触发。

该定义保证了任何变迁的触发都不会影响其他变迁的可调度性，保证了整个应用的实时性。

针对时间 Petri 网状态无限多的问题，Berthomieu[11] 最早提出了时间 Petri 网的状态类概念。

**定义 3** $\Sigma=(P, T, F, I_h, M_0, R^s)$ 为带抑制弧的时间 Petri 网，满足下列条件的二元组 $C=(M, D)$ 称作 $\Sigma$ 状态类：

(1) $M$ 为 $\Sigma$ 的一个标识；

(2) $\forall t_i\in ET(M)$，$D(t_i)=\begin{cases}\alpha_i\leqslant t_i\leqslant\beta_i\\ t_i-t_j\leqslant R_{ij}\end{cases}$ 是在标识 $M$ 下有发生权的变迁的有效触发域。

一个状态类是从初始状态出发，经过一系列共同的变迁触发而达到的所有状态的集合。通过状态类，可将无限的状态空间转化为有限的状态类空间，使 ITPN 的状态空间遍历成为可能。

**定义 4** 设 $\Sigma=(P, T, F, I_h, M_0, R^s)$ 为带抑制弧的时间 Petri 网，$C=(M, D)$ 为 $\Sigma$ 的一个状态类，从状态类 $C$ 有效触发有发生权的变迁 $t_i$，得到一个新的状态类 $C'=(M', D')$，$M'$，$D'$ 分别按如下规则得到（设 $\uparrow M(p_k)=M(p_k)-F(p_k, t_i)$，$\forall p_k\in P$）：

(1) 由于 $t_i$ 的触发使用了某些令牌，并产生了新的令牌：$\forall p\in P, M'(p)=M(p_k)-F'(p,t_i)+F'(t_i,p)$

(2) 下面根据 $M'$ 下使能的变迁 $t_j$ 与 $t_i$ 的关系分三个步骤计算 $D'$（详见文献[11]）：

① 插入新的有发生权的变迁的触发域：$\alpha_i=\alpha_j^s\wedge\beta_i=\beta_j^s$，$t_j\in((ET(M')-ET(M))\cup(ET(M')\cap ET(M)-ET(\uparrow M))$

② 消去因为 $t_i$ 的触发而没有发生权的变迁 $t_f$ 的触发域：

$$D'(t_j)=\begin{cases}\alpha'_j=\max(\alpha_j, \alpha_f-r_{fj})\\ \beta'_j=\max(\beta_j, \beta_f+r_{jf})t_k, t_j\in(ET(\uparrow M)\cap ET(M'))\wedge\\ \quad t_f\in(ET(M)-ET(\uparrow M))\\ r'_{jk}=\min(r_{jk}, r_{if}+r_{jk})\end{cases}$$

③ 对于一直都是有发生权的变迁：

$$D'(t_k)=\begin{cases}\alpha'_j=\max(0, -r_{ij}, \alpha_i-\beta_i)\\ \beta'_j=\min(r_{ji}, \beta_j-\alpha_i)t_k, t_j\in(ET(\uparrow M)\cap ET(M'))\\ r'_{jk}=\min(r_{jk}, \beta'_j-\alpha'_j)\end{cases}$$

综上所述，从初始状态类 $C_0$ 出发，通过不断有效触发有发生权的变迁将改变系统中的令牌分布，持续产生新的状态类，由此建立一个状态类空间。该空间可表示为一个有向图（状态类图），以状态类为顶点，父类顶点到子类顶点之间存在一条以对应的触发变迁来标注的有向边。利用该图，我们将无限的状态空间转化为有限的状态类空间。

## 3 构建模型

本节下面采用自顶向下的策略，将系统按功能划分成若干通信子系统。建模的具体实现步骤如下：

步骤 1 设计系统的组成部分

分布式实时嵌入式系统由各自独立的嵌入式设备模块组成，每个模块负责一定的功能，具有一定的自治性，但又依赖于系统其它模块的计算的结果[12]。按照功能划分一下系统的各个组成部分（任务），画出各个任务之间的关系图。如图 1 所示，系统有五个任务协同完成整个功能，其中任务 $T_5$ 的完成代表整个应用的完成。最后根据实际需要给出各个任务的时间约束。

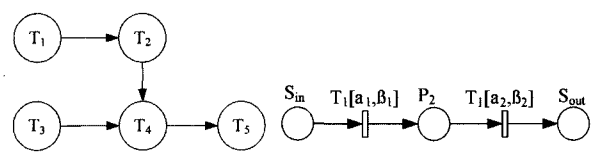


图 1 任务间的数据依赖




图 2 顺序关系的 ITPN 模型

步骤 2 描述语言的定义

本文的建模过程使用进程代数的一种扩展来描述任务/子任务之间的关系，形成表达式。基本的结构类似于语言 PA。下面将任务和子任务通称为对象，如果 $X$ 和 $Y$ 都是对象，则可以通过算子构造出更复杂的对象。在应用算子的过程中，可以用括弧来表达作用的顺序。

算子 $>$ 表示顺序：$Z=X>Y$ 也是对象，表示对象 $X$ 和 $Y$ 顺序执行。顺序算子的优先级最高。

算子 $+$ 表示选择：$Z=X+Y$ 也是对象，表示从对象 $X$ 和 $Y$ 选择一个执行。

算子 $||$ 表示并行：$Z=X||Y$ 也是对象，指对立地完成对象 $X$ 和 $Y$，$Z$ 结束当且仅当 $X$ 和 $Y$ 都结束。

算子 * 表示循环：$Z=X*Y$ 也是对象，指循环执行 $X$ 后再顺序执行 $Y$。

步骤 3　构造子系统的网模型

将任务理解为独立对象的集合，描述对象可能处于不同的状态，以及对象在每个状态可能经历的事件。每个对象都有一个初始态和一个结束态。在确定所有关键部分和优先约束后，通常一个任务可以划分为多个子任务。子任务的时间约束由任务的释放时间和截止时限决定，以保证子任务的执行不会超过任务的截止时限。最后采用步骤 2 的描述语言将子任务间的关系描述为一个表达式。

在 ITPN 模型中，顺序关系表示：如果一个变迁的引发直接导致另一个变迁可以被实施，则称这两个变迁的发生关系是顺序的，如图 2 所示：变迁 $T_1$ 执行后，才能引发 $T_2$。

为了构造并行，选择和循环对象结构，分别需要两对基本组件，即 AND-split 和 OR-split，AND-join 和 OR-join，如图 3 所示。其中在对 OR-split 建模时添加了两个瞬时变迁，这是因为复杂的对象模型都可以由这些基本的结构复合构成。$T_1$ 和 $T_2$ 的截止时限可能不一样，如果直接作选择，根据有效触发的定义则会一直执行截止时限小的那个任务，造成另一个任务的“饥饿”状态。

结合上述的基本模型，为每个任务表达式构造一个子网，每个子网有一个初始库所和一个终止库所，初始库所没有输入边，终止库所没有输出边，其他的库所都既有输入边也有输出边。这样构造出来的网一旦到达终止状态，那么网就结束了。

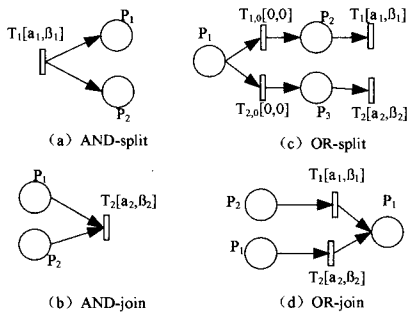


图 3　基本组件的 ITPN 模型

步骤 4　构造通信协议的网模型

在 DRE 软件中，各功能模块单元间是通过通信协议进行可靠、有效的通信。本文中主要是对静态非抢占式分布式实时嵌入式系统进行建模，所以相应的通信协议选择时间驱动协议(Time-Triggered Protocol，TTP)。

子任务间的通信抽象成顺序关系。这里只考虑任务之间的通信过程。有下面几个特点：发送完消息，任务继续它的工作；任务只能在收到完整的消息以后才继续；点对点通信，且信道可靠；有持有总线令牌的任务才可以执行发送消息；通信耗费的时间由对应的变迁表示。

TTP 的总线访问控制策略是时分多址(Time-Division Multiple-Access，TDMA)，图 1 示例对应的通信过程建模如图 4 所示，本文中的任务时间单位(Task Time Units，TTU)是 0.05s。其中假设 $S_i$，$M_i$，$C_i$ 和 $T_{c,,i}$ 分别表示任务 $i$ 的抽象、发送消息子任务、总线令牌存放位置和总线令牌传送。总线令牌在任务间循环传递。每个任务的总线令牌的保留时间是 1TTU，如超时仍无消息需要传送，则系统自动触发 $T_{c,,i}$ 将总线令牌传送至下一个任务。只有持有总线令牌的任务才可以执行发送消息操作 $M_i$，将消息发送到目的任务(如图 4 中 $M_1$ 发送消息给的目任务是 $T_2$)。由于 $T_5$ 的结束代表系统的结束，因此该任务没有发送消息到其他任务。根据实际情况如果任务 $T_i$ 有消息要发送，则不能执行 $T_{c,,i}$，所以图中 $S_i$ 和 $T_{c,,i}$ 之间有一条抑制弧。

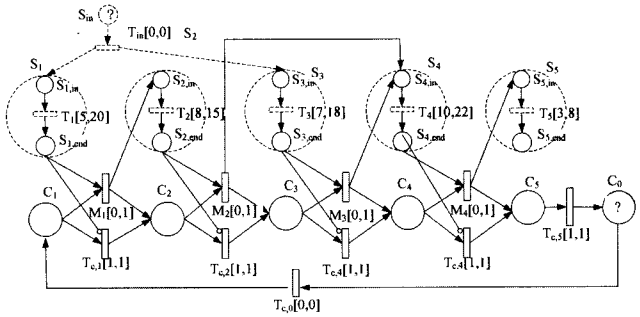


图 4　整个应用的 ITPN 模型

步骤 5　整个应用的 ITPN 模型

在 DRE 软件的 ITPN 模型中，令牌表征了资源。本文假定在任务执行过程中无内存限制，所以不考虑内存资源。此外，由于本模型中各个功能模块是通过消息来完成控制功能的，因此，我们可以将消息定义为网内的资源，以改善 Petri 网中无任何形式控制的状况，从而有力地支持了系统的合成。至此，DRE 软件中任务的结构性质、时间约束和通信机制都可以由 ITPN 来进行精确建模。在此基础上，采用 Petri 网的合成运算合并各个子网提供的公共接口，形成 DRE 软件的 ITPN 模型，如图 4 所示。对每个任务添加一个输入库所 $S_{i,in}$ 和一个输出库所 $S_{i,out}$ 作为任务 $i$ 与其他任务通信的公共接口，变迁 $T_i$ 表示任务 $i$ 的抽象，可以根据实际需要适当扩展。对于整个应用添加一个输入库所 $S_{in}$，变迁 $t_{in}$ 是产生系统的初始条件。

## 4　模型合理性分析

**定义 5**　设 $\Sigma=(P,T,F,I_h,M_0,R^s)$ 为带抑制弧的时间 Petri 网，$RG(\Sigma)=(V,E)$ 是 $\Sigma$ 对应的状态类图，$C_0$ 分别是 $\Sigma$ 的初始状态类，则 $\Sigma$ 具有如下性质：

(1)有界性：$\forall p\in P$，$\exists K\in N$，使得 $\forall C(M,D)\in V$ 有 $M(p)\leqslant K$，称 $\Sigma$(在状态类 $C_0$)有界的，如果 $K=1$，则称 $\Sigma$(在状态类 $C_0$)是安全的；

(2)活性：$\forall t\in T$，$\forall C\in V$，$\exists$ 触发序列 $\sigma$，$C'=(M',D')$，使得 $C[\sigma>C' \wedge M'[t>$，称 $\Sigma$ 是活的。

通过对 ITPN 的构建，可知 ITPN 模型的正确性、有效性的关键在于不存在结构上的问题。基于分布式实时嵌入式软件的本身特性定义一个合理的 ITPN 模型应该满足的最基本的需求：

(1)每个 ITPN 初始状态和终止状态均唯一；

(2)每个变迁/库所都在一条从库所 $S_{in}$ 到 $S_{out}$ 的路径上；

(3)在任何情况下，系统将最终结束，在结束的时候，系统到达终止状态；

(4)整个过程中没有死锁，即任何一个任务或子任务都有被调用的可能性。

这四条需求能够保证软件模型在结构上不存在任何错误，ITPN 能正常启动、运行和结束。根据以上四条需求，结合状态类图可以给出 ITPN 合理性(soundness)的形式化定义。

**定义 6** 设 $\Sigma=(P,T,F,I_h,M_0,R^s)$ 为带抑制弧的时间 Petri 网，$\Sigma$ 是合理的，当且仅当 $\Sigma$ 满足下面条件：

(1)任意可达状态类 $C\in V$，存在触发序列 $\sigma$，使得：$C[\sigma>C_{end}$

(2) $\forall t\in T,\exists C=(M,D)\in V$，使得：$M[t>$

从定义 6 可以看出，条件(1)映射到状态类图为图中任意一个节点，在该节点到节点 $C_{end}$ 之间肯定存在一条路径，该条件保证了 ITPN 模型最终能够到达结束状态；条件(2)要求网中不存在无死的变迁，即每个变迁都在状态类图中出现过。由于在构造模型的过程中已经保证了每个模型的输入库所和输出库所唯一，因此其初始状态和终止状态也是唯一。

下面给出判定 ITPN 合理性定理。首先引入内部网的定义：

**定义 7** 设 $\Sigma=(P,T,F,I_h,M_0,R^s)$ 为带抑制弧的时间 Petri 网，六元组 $\Sigma^*=(P,T,F,I_h,M_0,R^s)$ 称为 $\Sigma$ 的内部网，其中：$P^*=P;T^*=T\cup\{t^*\};F^*=F\cup\{(S_{in},t^*)=(t^*,S_{out})=1\};I_h{}^*=I_h;M_0{}^*=M_0;R^{s*}=R^s\cup\{R^{s*}(t^*)=[0,0]\}$

**定理 1** 一个带抑制弧的时间 Petri 网 $\Sigma$ 模型是合理的，当且仅当其内部网 $\Sigma^*$ 是活的且有界的。

证明：(1)必要性(⇒)反证法：假设 $\Sigma^*$ 不是活的或是无界的，则

往证 $\Sigma$ 是合理性⇒$\Sigma^*$ 不是活的不成立

又∵$\Sigma^*$ 不是活的，有下面两种情形：

情形 1：$\forall C\in V,\exists t\in T,\exists$ 触发序列 $\sigma,C'=(M',D')$，使得 $C[\sigma>C'\wedge\neg M'[t>$，这与定义 6(2)式矛盾

情形 2：$\forall t\in T,\exists C\in V,\exists$ 触发序列 $\sigma,C'=(M',D')$，使得 $C[\sigma>C'\wedge\neg M'[t>$，这与定义 6 (1)式矛盾

∴$\Sigma$ 是合理性⇒$\Sigma^*$ 是活的成立

往证 $\Sigma$ 是合理性的⇒$\Sigma^*$ 是无界的不成立

∵假设 $\Sigma^*$ 是无界的

∴$\forall K\in N,\exists C=(M,D)\in V,\exists p\in P$，使得：$M(p)>K$

∴$\exists t\in p^\cdot$，$t$ 可在状态类 $C$ 上至少连续触发两次

∴$\exists$ 触发序列 $\sigma_1,\sigma_2$，使得 $C[\sigma_1>C'\wedge M',D')\wedge C'[\sigma_2>C''(M'',D''),M'(S_{out})=1,M'(p)\neq0,M''(S_{out})=2$，这与 ITPN 模型的终止状态类唯一矛盾

∴$\Sigma$ 是合理性⇒$\Sigma^*$ 是有界的成立

综上，SCTPN 是合理性保证了内部网的活性和有界性，必要性得证。

(2)充分性(⇐)由 $\Sigma^*$ 的活性可知，定义 6(2)显然满足。因此只要证明 $\Sigma$ 满足定义 6(1)。

往证 $\Sigma^*$ 是活的且有界的⇒$\Sigma$ 满足 6(1)

∵$\Sigma^*$ 是活的

∴对于 $t^*$，$\forall C\in V,\exists$ 触发序列 $\sigma$，使得 $C[\sigma>C'(M',D'),M'[t^*>$

又∵$S_{out}$ 是 $t^*$ 的唯一输入库所，即终止状态类是唯一的

∴$C'=C_{end}$，即 $\forall C\in V,\exists$ 触发序列 $\sigma$，使得 $C[\sigma>C_{end}$，即定义 6(1)成立。

因此，$\Sigma$ 是合理，充分性得证。

综上所述，一个 $\Sigma$ 模型是合理，当且仅当其内部网 $\Sigma^*$ 是活的和有界的。

**定义 8** 一个 DRE 软件对应的 ITPN 模型是合理的，当且仅当：

(1)各个功能模块对应的 ITPN 模型是合理的；

(2)整个软件的 ITPN 模型对应的内部网是活的且有界的。

## 5 实例研究

为了更好地描述分布式实时嵌入式软件的 ITPN 模型的建模过程和合理性判定，本文使用地铁无人驾驶系统(Driverless Metro System，DMS)为例来介绍上述建模和分析过程。根据实现功能可划分为五大任务模块(如图 1 所示)，分别为导航数据采集($T_1$)、任务管理和决策($T_2$)、运行控制数据采集($T_3$)、运行控制($T_4$)和反馈回路控制($T_5$)。其中任务 $T_5$ 的完成代表 DMS 的一次运行控制数据返回，整个应用的模型则如图 4 所示。

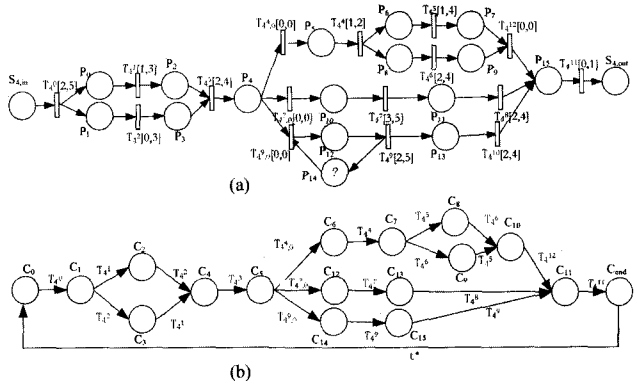


图 5 $T_4$ 的 ITPN 模型及内部网的状态类图

地铁运行控制 $T_4$ 的大概工作流程为：任务初始化($T_4{}^1$)，对地铁运行状态量进行检测及通讯解码($T_4{}^2$，$T_4{}^3$)，由地铁运行状态测量值判断地铁所处的运行状态及是否存在故障并做出控制决策($T_4{}^3$)，并根据不同的控制决策执行不同的计算($T_4{}^4$ 到 $T_4{}^{10}$)，最后输出($T_4{}^{11}$)。所以任务 $T_4$ 的表达式为：$T_4=T_4{}^1>(T_4{}^2\parallel T_4{}^3)>((T_4{}^4>(T_4{}^5\parallel T_4{}^6))+(T_4{}^7>T_4{}^8)+(T_4{}^9>T_4{}^{10}))>T_4{}^{11}$，对应的 ITPN 模型如图 5(a)所示。采用 Petri 网的可达图分析算法构造飞行控制任务对应的内部网的状态类图，如图 5(b)所示。该内部网有 16 个可能状态类，$C_0$，$C_{end}$ 分别表示其初始状态类和终止状态类，图 5(a)中任意的一个变迁，从图 5(b)中任意一个状态类出发，都存在一条路径使得系统到达该变迁可以有发生权的状态，而且所有状态都是安全的。所以可以判定地铁运行控制任务对应的 ITPN 模型是合理的。同理可以其他任务进行建模和验证。最后再根据定义 7 判定整个应用的合理性。

**结束语** 本文主要提出用带抑制弧的时间 Petri 来对分布式实时嵌入式软件进行建模和形式化验证的方法。该方法具有以下优点：(1)具有模块化功能和高度的可重用性。(2)具有严格的数学基础，易于对所建立的模型进行分析和验证。(3)在验证了系统的合理性后，下一步的工作主要有下面两个方面：(1)完善该方法，考虑系统的资源调度问题；(2)开发相应的工具对 DRE 软件进行模拟。

## 参 考 文 献

[1] Lardieri P, Balasubramanian J, Schmidt D C, et al. A Multi-layered Resource Management Framework for Dynamic Resource Management in Enterprise DRE Systems. Journal of Systems and Software, 2007, 80 (7): 984-996

[2] Tambe S, Balasubramanian J, Gokhale A S, et al. MDDPro: Model-Driven Dependability Provisioning in Enterprise Distributed Real-Time and Embedded Systems// Fourth Annual International Service Availability Symposium. New Hampshire, 2007

(下转第 299 页)

静态分布信息检索任务规划主体 PlanAgent：它是分布任务规划主体 TPA 的子类，它的主要功能和作用是：(1)接受用户输入的远程网络节点名称；(2)接受用户输入的分布检索任务描述；(3)根据远程网络节点名称和分布检索任务描述形成分布检索任务规划 dprtpm；(4)根据分布检索任务规划 dprtpm 创建分布信息检索主体 RetrievalAgent：，并启动它运行。PlanAgent：也兼做当前平台的主体移动服务器 MS 和分布信息检索主体的 SponsorAgent：。

在安装有检索工作流移动主体的任何一个网络平台上(任何异构的网络平台都可以安装检索工作流移动主体)安装分布检索任务规划主体后，它就能负责检索移动主体的发送、接收与运行。

### 4.3 基于 Z39.50 的分布式检索在复合型图书馆系统中的实现

最终实现的 Web-OPAC 子系统是 Internet 环境下的一个全文信息检索系统。其体系结构图如图 3 所示。

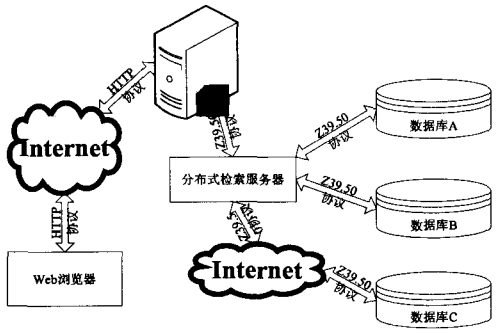


图 3 Web-OPAC 子系统的体系结构

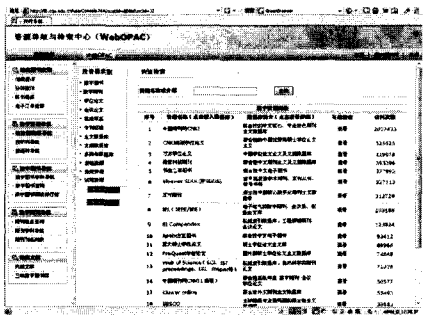

图 4 Web-OPAC 子系统主界面

图中 Web 浏览器负责接收和初步处理，并发送客户输入的查询请求，并基于 http 协议通过 Internet 传送到 web 服务器上，在 Web 服务器上进行信息处理转换成符合 Z39.50 协议规则的信息，再传递给本地或远程的检索服务器。检索服务器接受用户请求并执行检索。最后把请求处理后的结果送给 Web 浏览器端的客户。在 ADLibSys 系统中的最终展现的 Web-OPAC 如图 4 所示。

**结束语** 我们研究了复合型数字图书馆的特性，对各种信息检索技术进行比对，最终选择了基于移动主体的分布式检索方式，这种检索方式能够从增强客户端事务处理能力、增强远程 Server 的事务处理能力和功能的特性化以及增强分布计算的自治性三个方面扩展分布检索的灵活性和高效性，并将这种分布检索方式基于 Z39.50 标准协议，实现了高效、准确、灵活、快捷的跨库统一检索子系统 Web-OPAC，使跨库检索的准确率和速度得到很大的提高。复合型数字图书馆的信息检索系统中依然有许多问题有待我们进一步去研究和开发，效率、准确度、灵活性有待更进一步提高；信息检索也将向着更加智能化和人性化方向发展。

## 参 考 文 献


[1] 张大苹. 论复合图书馆的信息资源建设和用户服务[J]. 情报资料工作，2004(1)：34-36

[2] 杨晓江，张福炎. 基于 Z39.50 的联机书目检索服务[J]. 软件学报，1999，10(8)：8-24

[3] 李振龙. Web 信息检索的技术分析与发展策略研究[J]. 计算机科学，2006，33(4)：181-184

[4] 赵文清，高伟. 移动 Agent 在分布式信息检索中的研究[J]. 计算机工程与应用，2006，42(25)：170-172

[5] 周斌，刘波. Z39.50 协议的原理及其在分布式检索中的应用[J]. 计算机工程，2002，28(9)：275-277

[6] Nasraoui O，Rojas C. From static to dynamic web usage mining：Toward scalable profiling and personalization with evolutionary computation// the Proc. Workshop Inf. Technol. (Invited Paper). Rabat，Morocco，2003

[7] Uehara M，Sato N. Information Retrieval based on Temporal Attributes in WWW Archives// Proceedings of the 2005 11th International Conference on Parallel and Distributed Systems (ICPADS05)

[8] Marcus1 A，Lucia A D，Hayes J H. Information Retrieval Based Approaches in Software Evolution // 22nd IEEE International Conference on Software Maintenance (ICSM'06)


(上接第 280 页)


[3] Madl G，Abdelwahed S，Abdelwahed S. Modelbased Analysis of Distributed Real-time Embedded System Composition// Proceedings of the 5th ACM international conference on Embedded software. New Jersey：International Workshop on Embedded Systems. 2005：371-374

[4] Schmidt D. MDE4DRE：Model-Driven Engineering for Distributed Real-time and Embedded Systems// 13th IEEE Real-Time and Embedded Technology and Applications Symposium. Washington，2007

[5] Marcel V，Gorm L P，Jozef H. Modeling and Validating Distributed Embedded Real-Time Systems with VDM++ // Proceedings of Formal Methods. Heidelberg：Springer-Verlag，2006：147-162

[6] Shankaran N，Balasubramanian J，Schmidt D，et al. A Framework for (Re) Deploying Components in Distributed Realtime and Embedded Systems// Proceedings of the 2006 ACM Symposium on Applied Computing SAC '06. New York：ACM Press，2006

[7] Liu Shih-hsi，Bryant B，Gray J，et al. Two-level Assurance of QoS Requirements for Distributed Real-time and Embedded Systems// Proceedings of the ACM Symposium on Applied Computin. New Mexico：ACM Press，2005：903-904

[8] Yuan You-wei，Yan La-mei，Guo Qing-ping. The Efficient QoS Control in Distributed Real-Time Embedded Systems. Embedded Software and Systems，Springer Berlin：Heidelberg，2005 (3605)

[9] 吴斌，叶绿，吴朝晖. 一种分布式实时嵌入式软件的构件模型 Z-CCM. 计算机工程与应用，2005，41：40-44

[10] Shankaran N，Koutsoukos X，Schmidt D，et al. Hierarchical Control of Multiple Resources in Distributed Real-time and Embedded Systems，Real-Time Systems Journal// Special Issue on Best Papers at Euromicro Conference on Real-Time Systems (ECRTS'06). 2006

[11] Berthomieu B，Diaz M. Modeling and verification of time dependent systems using time Petri nets. IEEE transactions on Software Engineering，1991，17 (3)：259-273

[12] Pop P，Eles P，Peng Z，et al. Analysis and optimization of distributed real-time embedded systems// Proceedings of the 41st annual conference on Design automation DAC '04. New York：ACM Press，2004：593-625