MARTE顺序图到TTS4SD的转换

朱梅霞 武继刚
(天津工业大学计算机科学与软件学院 天津 300387)

摘 要 MARTE对UML的顺序图进行了扩充,使其适用于实时系统的建模阶段,但它不能直接用于正确性验证阶段。对象管理组织提出用模型转换的方法将依照MARTE构造的顺序图(记为A)转换成具有完备的验证方法和工具的形式化模型(记为B)。用B表示A的语义可以保证B能够完整且准确地模拟A的行为。提出了形式化模型——TTS4SD,用来描述MARTE顺序图的形式语义,并在此基础上展开了验证。首先给出顺序图的形式定义,把时间变迁系统(TTS)扩充成TTS4SD;然后用TTS4SD描述顺序图的形式语义,并给出从顺序图到TTS4SD的转换算法;最后对TTS4SD展开分析。通过一个实例说明了从顺序图到TTS4SD的转化过程以及基于TTS4SD的验证方法。

关键词 实时系统,顺序图,CCSL,形式化方法,模型转换
中图法分类号 TP301 **文献标识码** A

Approach to Transforming MARTE Sequence Diagram to TTS4SD Models

ZHU Mei-xia WU Ji-gang
(School of Computer Science and Software Engineering, Tianjin Polytechnic University, Tianjin 300387, China)

Abstract The sequence diagram was extended in the MARTE specification for modeling purpose, but it can not be used in the correctness verification stage. The OMG proposes to solve this problem by model transformation techniques: the model A is transformed to a formal model B which is equipped with efficient analysis or verification tools. To describe the semantics of A by model B can guarantee the bi-simulation relation between them. A model named timed transition system for sequence diagram (TTS4SD) was proposed. At first, we offered the formal syntax of the sequence diagram and the TTS4SD, then we described the semantics of the sequence diagram by TTS4SD. Taking the semantics as basis, the checking work was carried out on the TTS4SD. An example was given to describe the above process.

Keywords Real time system, Sequence diagram, CCSL, Formal method, Model transformation

1 引言

建模和验证是设计实时系统的两个重要部分,一组能对系统的动态行为和时间属性进行建模的工具是建模的基础,这些模型不仅要方便设计者使用,而且要容易被用户理解,能使用户和设计者就系统功能达成一致;在验证阶段,设计者对模型的正确性进行验证。统一建模语言UML(Unified Modeling Language)[1]提供的建模工具分别从静态和动态两方面对系统行为进行建模,但UML在时间建模方面能力不足,这主要是由于UML定义的Simple Time类在对系统的时间属性进行建模时的能力是非常有限的。对象管理组织OMG(Object Management Group)指出:UML对系统的时间属性进行建模的能力有待提高,应通过扩充的形式,使其提供的建模工具能对时间属性要求严格的系统进行建模。综合各方观点,OMG发布了MARTE(Modeling and Analysis of Real-time Embedded Systems Specification)[2]用于对实时系统进行建模和分析。

MARTE对UML定义的顺序图进行扩充,使其可对实时系统的时间属性和动态行为进行建模。MARTE对顺序图的扩充体现在两个方面:将消息对应的发送事件和接收事件赋予一个时间区间$[x,y]$,其中$x,y\in\mathbf{R}^+\wedge x<=y$,$\mathbf{R}^+$是非负实数集,时间区间的默认值为$[0,0]$;用CVSL(Clock Value Specification Language)描述事件之间需满足的时间限制(分析阶段待验证的系统需满足的性质)。

在需求建模阶段,设计者通常用顺序图描述系统的动态行为,但在分析阶段,并没有直接对顺序图进行分析的工具。这在一定程度上限制了MARTE在软件设计过程中的应用范围和效率。为此,在性能分析建模阶段,OMG提出用模型转换[3]方法将依照MARTE建模阶段设计的模型A转换成另一种具有完备的分析方法和工具的形式化模型B,然后对B进行分析和精化,以间接完成A的分析和精化工作。迄今,对系统的时间属性进行建模的工具大都是以下3类的变形或扩充:时间Petri网TPN(timed Petri net)[4]、时间变迁系统TTS(timed transition system)[5]和时间自动机TA(time

到稿日期:2012-03-20 返修日期:2012-06-20 本文受国家自然科学基金(61173032)资助。
朱梅霞(1981—),女,博士,讲师,主要研究方向为软件形式化方法、软件建模与验证,E-mail:10848877@pku.edu.cn;武继刚(1963—),男,博士,教授,主要研究方向为软硬件划分技术、并行计算、容错处理器重构技术(通信作者)。

automata)[6]。

文献[7-10]旨在将 MARTE 顺序图转换为 TPN，但他们的转换方法不能保证源模型和目标模型之间的互模拟性。将顺序图转换为 Petri 网的目的之一是用 Petri 网描述顺序图的形式化语义，尤其是在将 MARTE 顺序图转换为 TPN 后，由于 TPN 本身存在两种语义（强语义和弱语义），两者的区别主要在于：当事件的发生时间上限到达时，事件是强制发生还是重新使能？所以在将 MARTE 顺序图转换成 TPN 后，语义描述方式的选择也是关键。在文献[11]中，作者给出了从顺序图到 TPN 的转换过程，但是在分析过程中他们将事件上的时间区间理解成了两个离散的时间点，即事件要么在时间区间的下限时刻发生，要么在时间区间的上限时刻发生，这显然与时间区间的作用相违背。所以在对 MARTE 顺序图进行检测时，不仅要给出从 MARTE 顺序图到形式化模型的转换规则，还要给出恰当的分析方法。

我们注意到 TPN 的语义是用 TTS 描述的，所以本文以 TTS 为目标模型。首先给出 MARTE 顺序图的形式化语法并给出 TTS4SD 的定义，然后用 TTS4SD 从两个方面（事件发生和组合片段转移）描述 MARTE 顺序图的语义，这个语义给出了从 MARTE 顺序图到 TTS4SD 的转换方法。基于此，本文给出的转换方法保证了源模型和目标模型之间的互模拟关系，最后给出从 MARTE 顺序图到 TTS4SD 的生成算法和状态可达图的生成算法，并展开验证。

2 MARTE 顺序图

定义 1 MARTE 顺序图 D 表示为：$D=(C,O,M,E,G,Pro,Fun,Cseq,Eseq)$，其中：

(1)C,O,M,E 分别是组合片段、对象、消息和事件的有限集合。给定一个消息 $m\in M$，我们用 $!\,m$ 和 $?\,m$ 分别代表发送和接收消息 m 的事件。$(!\,m,?\,m)$ 表示消息 m 的发送事件必须在它的接收事件之前发生。另外我们规定 C,O,M 和 E 的两两交集为空。

(2)G 是布尔表达式的集合。

(3)Pro 是顺序图中时间约束的集合。

(4)Fun 是一组函数，$Fun=(\phi,\varphi,\omega,\alpha,\beta,\gamma,\psi)$，其中：

• $\phi:C\to K$，$K=\{seq,alt,opt,par,break,loop,neg,assert\}$，$\phi$ 用来规定组合片段的类型。

• $\varphi:C\to G$ 规定组合片段对应的条件。$\forall c\in C,\phi(c)\in\{par,neg,assert,seq\}$，则 $\varphi(c)=tt$。

• $\omega:C\to C\cup\{\tau\}$ 规定组合片段间的嵌套关系。$\omega(c)=c'$ 表示 c' 嵌套 c；ω 具有如下属性：给定组合片段 c,c',c''，若 $\omega(c)=c'$，则不存在正整数 n 使得 $\omega^n(c')=c''\wedge\omega(c)=c''$，也就是说，若 $\omega(c)=c'$，则 c' 是嵌套 c 的最内层组合片段。$\omega(c)=\tau$ 表示 c 不被嵌套任何组合片段，此时称 c 是极大的，若 $\forall c'\in C,\omega(c')\neq c$，即 c 不嵌套任何组合片段，则称 c 是极小的。用 C_E 和 C_P 表示顺序图的极小组合片段集和极大组合片段集。

• $\alpha:E\to O$ 表示事件和对象间的对应关系，我们规定 $\alpha(e)=o_1\wedge\alpha(e)=o_2\Rightarrow o_1=o_2$。

• $\beta:E\to\mathbf{R}^+\times\mathbf{R}^+$ 表示事件和时间区间的对应关系。给定事件 e，用 $[eft(e),lft(e)]$（其中 $eft(e)\leqslant lft(e)$）的形式表示 e 对应的时间区间，$eft(e)$ 和 $lft(e)$ 分别代表 e 的最小和最大延迟时间。时间区间可以是闭区间或开区间，也可以是半开半闭区间。

• $\gamma:E\to C$ 表示它所处的最内层组合片段。γ 具有如下属性：给定事件 e 和组合片段 c,c'，若 $\gamma(e)=c$，则不存在组合片段 c' 和正整数 n 使得 $\gamma(e)=c'\wedge\omega^n(c')=c$。

• $\psi:E\to\Sigma^*$ 用来给每个事件赋予标号。

(5)$Cseq$ 是一个集合，它的每个元素都形如 (c,c')，这表示 c 应该在 c' 之前判断它里面的算子可否发生，体现在顺序图的图形表示上，即 c 在 c' 的上面。

(6)$Eseq$ 是一个集合，它的每个元素都形如 (e,e') 或 $\langle e,e'\rangle$，其中前者表示 e 须在 e' 之前发生，后者表示 e 既可在 e' 之前发生，也可在其后发生，而且两者可以同时发生。

3 TTS4SD

本节将给出用来描述 MARTE 顺序图的时间迁移系统 TTS4SD 的定义。每个事件都对应着它所处的对象和组合片段；而且，在顺序图中变迁分为 3 类：事件间的离散变迁、组合片段间的离散变迁和连续变迁。事件间的离散变迁和组合片段间的离散变迁从两个方面刻画了系统的动态行为：当条件不变时，在同一组合片段中的事件的发生顺序；当条件改变时，从当前组合片段到条件为真的组合片段之间的转移，或是当前组合片段的最末事件执行结束后，下面该执行的组合片段是哪个？传统的时间迁移系统在描述顺序图的语义方面存在不足，因为它们不能从以上方面全面地描述顺序图的动态行为，为此本文将 TTS 扩充为 TTS4SD，下面给出定义。

3.1 TTS4SD 的语法

定义 2 给定顺序图 $D=(C,O,M,E,G,Pro,Fun,Cseq,Eseq)$，有一个迁移系统 $(TTS_d)\,S_d=(Q_e,q_{0e},T_e,l_{Te},u_{Te})$ 描述 D 的事件之间的迁移关系：

(1)$Q_e\subseteq P(E)$；

(2)$q_{0e}\subseteq Q_e$ 是初始状态的集合；

(3)T_e 是事件变迁的集合，每个变迁 $t_e\in T_e$ 都是 Q_e 上的一个二元关系；T_e 包含两类变迁：

• 延迟变迁 $q_e\xrightarrow{d}{}_eq_{e'}$，$l_t\leqslant d\leqslant u_t$；

• 离散变迁：离散变迁 $q_e\xrightarrow{e}{}_eq_{e'}$。

(4)l_{Te} 是一个集合，$l_{te}\in l_{Te}$ 表示变迁 t_e 的最小延迟；

(5)u_{Te} 也是一个集合，$u_{Te}\in u_{Te}$ 表示 t_e 的最大延迟。

定义 3 给定顺序图 $D=(C,O,M,E,G,Pro,Fun,Cseq,Eseq)$，有一个迁移系统 $(TS_d)\,S_d=(Q_c,q_{0c},T_c)$ 描述组合片段之间的迁移关系，其中：

(1)$Q_c\subseteq P(C)$ 是状态的集合；

(2)$q_{0c}\subseteq Q_c$ 是初始状态的集合；

(3)T_e 是变迁的集合，它是 $Q_c\times G\times C\times Q_c$ 上四元组的集合。

定义 4 TTS4SD S 的形式为 $S=(TTS_d,TS_d)$，其中 TTS_d 是定义 2 定义的基于事件变迁的时间迁移系统；TS_d 是定义 3 中定义的基于组合片段变迁的迁移系统。

3.2 TTS4SD 的语义

TTS 的传统语义是‘紧急’的，若两个或多个事件 $e_x,\cdots,e_y$ 可在状态 q 发生，变迁 t 可以发生当且仅当条件 $\exists e_x\in q$，$t=q\xrightarrow{e_x}q'\wedge u_t\leqslant l_{t'}\wedge t'\in\{\xrightarrow{e}|e\neq e_x\wedge e\in q\}$ 满足，也就是说事件 e_x 所在的变迁可以发生当且仅当在这些使能（即可以发生）事件集合中，e_x 的时间上限必须小于其他事件的时间下限。此语义与 TPN 的强语义相对应，它在描述顺序图的语义

时不合适。顺序图中只有 par 组合片段里的事件可以同时使能，当这些事件使能时，它们的发生顺序是任意的，也就是说这些事件的时间区间规定了该事件自使能到发生需要等待的最短时间和最长时间，只要当前时钟在该事件的时间区间内，该事件就有权发生，而与其他事件的时间区间无关，在 TPN 中，这种语义称为弱语义。本文将 TTS 的强语义改为弱语义。首先定义从 $X\subseteq\Sigma^*\cup\mathbf{R}^+$ 到 $\mathbf{R}^+$ 的映射 $v, v(e)$ 代表自事件 e 使能至发生经历的时间，其中：$\forall x\in\Sigma, \forall d\in\mathbf{R}^+, v(x+d)=v(x)+d$。

定义 5 TTS4SD 的语义：给定 $S=S_{et}+S_{ct}, S_{et}=(Q_e, q_{0e}, T_e, l_{Te}, u_{Te}), S_{ct}=(Q_c, q_{0c}, T_c)$。$S$ 的语义用两组有穷或无穷的序列描述：

(1)S_{et} 中的序列，这些序列可以有穷也可以是无穷。每个序列形如 $\rho_e=(q_{0e},0)\xrightarrow{d_0}{}_e(q_{0e},v_0')\xrightarrow{a_0}{}_e(q_{1e},v_1)\cdots\xrightarrow{d_{n-1}}{}_e(q_{n-1e},v_{n-1}')\xrightarrow{a_n\cdots a_n'}{}_e(q_{ne},v_n)$，其中：

· 形如 $(q,v)\xrightarrow{a}{}_e(q',v')$ 的变迁使能的条件是：$a\in q\wedge l_a\leqslant v(a)\leqslant u_a$，事件 a 发生后各事件的时间归零规则定义为：$v'(a')=\begin{cases}0, & \text{若 } a'\in q'\vee a'=a\\ v(a'), & \text{否则}\end{cases}$。

· 形如 $(q,v)\xrightarrow{d}{}_e(q,v')$ 的延时满足：$\exists t\in T_e, l_t\leqslant d\leqslant u_t\wedge t\in q\wedge\forall x\in q, v'(x)=v(x)+d$。

(2)S_{ct} 上的一组有穷或无穷序列，每个变迁形如 $\rho_c=q_{0c}\xrightarrow{(g_0,c_0)}{}_cq_{1c}\cdots\xrightarrow{(g_{n-1},c_{n-1})}{}_cq_{nc}$。

4 MARTE 顺序图的语义

定义 6 $c.t, c.b, c.h, c.f, \Theta(c)$ 分别表示组合片段 c 的初始组合片段集、终结组合片段集、初始事件集、终结事件集以及需满足的布尔条件，其中：

(1)$c'\in c.t\Leftrightarrow\omega(c')=c\wedge\forall c''\in C, \omega(c'')=c\Rightarrow(c'',c')\notin t(Cseq)$；

(2)$c'\in c.b\Leftrightarrow\omega(c')=c\wedge\forall c''\in C, \omega(c'')=c\Rightarrow(c',c'')\notin t(Cseq)$；

(3)$e\in c.h\Leftrightarrow\gamma(e)=c\wedge\forall e'\in E, \gamma(e')==c\Rightarrow(e,e')\notin Eseq$；

(4)$e\in c.h\Leftrightarrow\gamma(e)=c\wedge\forall e'\in E, \gamma(e')==c\Rightarrow(e',e)\notin Eseq$；

(5)$\Theta(c)=\begin{cases}\varphi(c), & c\in C_P\\ \Theta(c')\wedge\varphi(c), & \omega(c)=c'\end{cases}$；

(6)如果 $c\notin C_E$，则 $c.h=\bigcup_{i=1}^{|c.t|}c.t.i.h, c.f=\bigcup_{i=1}^{|c.b|}c.b.i.f$，用 $E_0=\bigcup_{i=1}^{|C_p|}C_p.i.h$ 表示各组合片段的初始事件集的并集。

定义 7(顺序图的语义) 给定顺序图 $SD=(C,O,M,E,G,Pr,Cseq,Eseq,Fun)$，它的语义用一个面向对象的时间变迁系统 $S=S_{et}+S_{ct}$ 表示，其中 S_{et} 是一个 TTSet，S_{ct} 是一个 TSct，详细定义如下：

(1)$S_{et}=(Q_e, q_{0e}, T_e, l_{Te}, u_{Te})$，其中：

· $Q_e=P(E)$；

· $q_{0e}=E_0$；

· T_e 包含延时和事件发生两类变迁：

(a)延时变迁 $(E_{cur},v)\xrightarrow{d}{}_e(E_{cur},v'), v'=v+d$；

(b)离散变迁：$(E_{cur},v)\xrightarrow{e}{}_e(E_{nx}(e),v'), e\in E_{cur}\wedge\forall e'\in E_{cur}, \varphi(\gamma(e))=tt\Rightarrow(\gamma(e'),\gamma(e))\notin t(Cseq)$。

· l_{Te} 是每个事件对应的时间区间的下限；

· u_{Te} 是每个事件对应的时间区间的上限。

(2)$S_{ct}=(Q_c, q_{0c}, T_c)$，其中：

· $Q_c=P(C)$；

· $q_{0c}=C_0=\{c\mid\exists e\in E_0, \gamma(e)=c\}$；

· T_c 中各元素形如 $C_{cur}\xrightarrow{(g,c)}{}_cC_{nx}(c), g=\varphi(c), \forall c'\in C, \Theta(c)=tt\wedge\Theta(c')=tt$；

· $(c',c)\notin t(Cseq)$。

定义 8 下面给给出定义 7 中有关元素的详细定义。

(1)$en(e)=tt$ 表示 e 使能，$en(e)=tt\Leftrightarrow e\in E_{cur}\wedge\varphi(\gamma(e))=tt\wedge v(\psi(e))\in\beta(e)$。只有当 $en(e)=tt$ 时，$(E_{cur},v)\xrightarrow{e}{}_e(E_{nx}(e),v')$ 才能发生。变迁发生后，$\forall e_x\in E$ 的时间置零策略表示为：

$$v'(e_x)=\begin{cases}0, & \text{若 } e_x\in E_{nx}(e)\vee e_x=e\\ v(e_x), & \text{否则}\end{cases}。$$

(2)$(E_{cur},v)\xrightarrow{e}{}_e(E_{nx}(e),v')$ 中 $E_{nx}(e)$ 的定义如下：

· 若 $e\notin\gamma(e).f$，则 $e_x\in E_{nx}(e)\Leftrightarrow e_x\in E_{cur}\vee(e,e_x)\in Eseq\vee\langle e,e_x\rangle\in Eseq$；

· 若 $e\in\gamma(e).f$，则分两种情况计算 $E_{nx}(e)$；

(a)若对所有的正整数 n，其中 $1\leqslant n\leqslant up, \omega^{up}(\gamma(e))\in C_P, \phi(\omega^n(\gamma(e)))\neq break$，则 $e_x\in E_{nx}(e)\Leftrightarrow(e,e_x)\in Eseq\vee\langle e,e_x\rangle\in Eseq\vee$

$$\begin{cases}(\gamma(e),\gamma(e'))\in t(Cseq)\vee\gamma(e)=\gamma(e')\wedge\gamma(e)\in C_P\wedge\phi(\gamma(e))=loop\\ \qquad\vee\\ \exists n,m\in\mathbf{N}, (\omega^n(\gamma(e)),\omega^m(\gamma(e')))\in t(Cseq)\\ \qquad\vee\\ \phi(\omega^n(\gamma(e)))=loop\wedge\gamma(e)\in\omega^n(\gamma(e)).b\wedge\gamma(e')\in\omega^m(\gamma(e)).y=t\end{cases}$$

(b)若存在正整数 n，其中 $1\leqslant n\leqslant up, \omega^{up}(\gamma(e))\in C_P, \phi(\omega^n(\gamma(e)))=break$，则：

$$e'\in E_{nx}(e)\Leftrightarrow\begin{cases}(e,e')\in Eseq\vee\langle e,e'\rangle\in Eseq\\ \qquad\vee\\ \exists c,e'\in c.h\wedge(\omega^n(\gamma(e)),c)\in t(Cseq)\wedge\\ \quad\forall k,\omega^{n+1}(\gamma(e))\neq\omega^k(\gamma(e')),\\ \text{其中 } 1\leqslant k\leqslant up, \omega^{up}(\gamma(e))\in C_P\end{cases}$$

· $(E_{cur},v)\xrightarrow{e}{}_e(E_{nx}(e),v')$ 发生后，$\forall e_x\in E$ 的时间置零策略是：

$$v'(e_x)=\begin{cases}0, & \text{若 } e_x\in E_{nx}(e)\vee e_x=e\\ v(e_x), & \text{否则}\end{cases}。$$

(3)$C_{nx}(c)$ 的定义为：

· 若 $\phi(c)\neq break$，则 $c'\in C_{nx}(c)\Leftrightarrow$

$$\begin{cases}(c,c')\in t(Cseq)\vee(c=c'\wedge\phi(c)=loop)\\ \qquad\vee\\ \exists n\in\mathbf{N}, \omega^n(c)=c''\wedge\phi(c'')=loop\wedge c\in c''.b\wedge c'\in c''.t\end{cases};$$

· 若 $\phi(c)=break$，则 $c'\in C_{nx}(c)\Leftrightarrow$

$$\begin{cases}(c,c')\in t(Cseq)\vee\omega(c)\neq\omega(c')\\ \vee\\ \exists n\geqslant 2, \phi(\omega^n(c))=loop\wedge c'\in\omega^n(c).t\end{cases};$$

· $C_{nx}(c)=\emptyset\Leftrightarrow\forall c'\in C, (c,c')\notin t(Cseq)$。

5 基于 TTS4SD 的分析

对实时系统开展分析工作要解决的两个关键问题是：如何规定时间的归零法则和定义后续状态，定义 8 给出了这两个问题的解决办法和顺序图 TTS4SD 的生成算法。下面开始讨论顺序图的验证问题。

5.1 判断事件序列的存在性

就顺序图而言，测试旨在判断给定的代表系统具体行为的事件序列或组合片段序列是否满足。若序列较简单，人为就可判断序列是否存在于相应的顺序图中，当序列较复杂时，就必须借助机器的帮助。$Se=t_1e_1t_2e_2\cdots t_ne_n$，其中 $t_i\in\mathbf{R}^+$，$e_i\in E$，$1\leqslant i\leqslant n$ 是否为真可以基于转换得到的 TTSet 进行判断。直观地，Se 为真当且仅当 $t_i\in\beta(e_i)$ 必须为真，其中 t_i 代表 e_i 自使能至发生经历的时间；TTS4et 中存在事件序列 $(\gamma(e_1),e_1,\alpha(e_1))\cdots(\gamma(e_n),e_n,\alpha(e_n))$。算法 1 用于判断 ex 是否存在，其中 $se.f,se.f.t$ 分别代表 se 的第一个事件和自其使能至发生经历的时间。

算法 1 判断 TTS4SDS 中是否存在序列 se

```
输入:S:TTS4SD, se:执行序列;
输出:t:{true,false};
int i=j=k←1; Snew=Snow←∅;Q←Qc
while (i≤|Q|) /*将 se.f 所在的状态标记为'new';*/
  if (se.f∈Q.i) Snew←Snew∪Q.i;
  elsei←i+1;
while(j≤|Snew|)
  char se.c←se.f;
  if se.f.t∉β(se.f) t←false;break; /*跳出循环*/
  else
    se←se−se.f.t−se.f;
    i←1;
    while (i≤|Q|) /*将 se.f 所在的状态标记为'now';*/
      if (se.f∈Q.i) Snow:=Snow∪Q.i;
      else i←i+1
    while(k≤|Snow|)
      if 存在从 Snew.j 到 Snow.k 的变迁(γ(se.c),se.c,β(se.c)))Snew
        ←Snow.k;j←1;break;
      else k←k+1;
    if(k=i+1)j←j+1;
    else j←1;
if (j=|Snew|+1) t←false;
else t←true;
```

5.2 时间约束的满足性

下面讨论另一个问题：如何判断顺序图给定的时间约束是否满足。用 $con(\varphi(e_i),\cdots,\varphi(e_j))$ 表示与 $e_i,\cdots,e_j$ 相关的时间约束。CVSL 指出任何一个约束都可以表示成一组两两相关的时间子约束的集合。验证约束是否满足是将约束和 TTS4SD 作为输入，对 TTS4SD 依照时间和条件的一个遍历过程，最后输出的内容就是验证的结果。验证过程包括 3 部分：给定某约束 $co=con(\varphi(e_i),\cdots,\varphi(e_j))$：

(1)首先确定该约束的起始时间点。Set_e^{co}，E_f^{co}，E_u^{co} 表示与 co 相关的事件集、首事件集和尾事件集，定义如下：

$$Set_e^{co}=\{e_i,\cdots,e_j\},E_f^{co}=\{e\mid e\in Set_e^{co}\wedge(q_e',v(e'))\xrightarrow{e'}{}_e\cdots\xrightarrow{e}{}_e(q_e,v(e))\notin TTSet\}$$

$$E_f^{co}=\{e\mid e\in Set_e^{co}\wedge\forall e\in Set_e^{co},(q_e',v(e'))\xrightarrow{e'}{}_e\cdots\xrightarrow{e}{}_e(q_e,v(e))\notin TTSet\}$$

$$E_u^{co}=\{e\mid e\in Set_e^{co}\wedge\forall e\in Set_e^{co},(q_e,v(e))\xrightarrow{e}{}_e\cdots\xrightarrow{e'}{}_e(q'_e,v(e'))\notin TTSet\}$$

式中，$\psi(e_f),e_f\in E_f^{co}$ 是验证约束 co 的时间起始点。找出从 $e_f\in E_f^{co}$ 到 $e_u\in E_u^{co}$ 的所有执行序列，简单起见，将这些序列记作 $EVseq(c_0)$。

(2)其次定义两个时钟：全局时钟 $GC(e)$ 和局部时钟 $LC(e)$，用于记录事件 e 的发生时间。假设 e 所在的序列是 $Seq=e_0\ll e_1\cdots\ll e_x\ll e$，$GC(e)$ 从 e_0 发生的时间计时，$LC(e)$ 则从 e_x 发生的时间计时。

(3)最后比较按照 TTS4SD 提供的执行序列计算得到的结果与顺序图原先规定的约束之间的关系，如果 TTS4SD 得到的下界小于约束规定的下界或上界大于约束规定的上界，说明设计者给与 co 相关的事件规定的时间区间是不合适的，要么对约束进行修改，要么对时间区间进行修改。

结束语 无论是 TPN 还是 TTS，它们都是外来模型，不是 MARTE 定义的模型。CCSL(Clock Constraint Specification Language)定义的模型可对系统的时间属性进行建模，基于 Observer 技术，还可对 CCSL 模型的正确性进行验证。但与顺序图相比，对用户而言，CCSL 描述的模型较抽象，不利于他们从直观上理解系统的结构和行为。实现从顺序图到 CCSL 模型的转换并证明顺序图与 CCSL 模型的互模拟关系，将在一定程度上扩大 MARTE 在软件设计中的应用范围和效率：用顺序图对系统的动态行为进行建模，使用户和设计者对系统行为达成一致；将顺序图转换成 CCSL 模型进行分析，以保证模型的正确性。所以下一步的工作是实现顺序图到 CCSL 之间的转换并证明转换规则的正确性。

参 考 文 献

[1] OMG. The Unified Modeling Language Superstructure Specification(2 Edition)[M]. February 2009

[2] OMG. A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems(2 Edition)[M]. Nov. 2009

[3] Kozaczynski W, Sandal S. Model Transformation: The Heart and Soul of Model-Driven Software Development [J]. IEEE Software, 2003, 20(5): 42-45

[4] Merlin P M. A Study of the Recoverability of Computing Systems [D]. University of California, Irvine, CA, USA, 1974

[5] Henzinger T A, Manna Z, Pnueli A. Timed transition systems [C]//REX Workshop. 1991: 226-251

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

[7] Bornot S T S, Sifakis J. Modeling Urgency in Timed Systems [C]//COMPOS. 1997: 103-129

[8] Kuo J-Y, Fanjiang Yong-yi, Yang S, et al. Towards the Verification of Scenarios with Time Petri-Nets[C]//COMPSAC. 2000: 503-508

[9] Ehrig H, Heckel R, Baldan P, et al. Compositional semantics for open Petri nets based on deterministic processe[J]. Mathematical Structures in Computer Science, 2005, 15(1): 1-35

[10] Haugen Ø, Halvorsen O, Kobro R. Time Exceptions in Sequence Diagrams [C]//MoDELS. 2006, 131-142

[11] Callou-Bruno G, Maciel P. Mapping UML Sequence Diagram to Time Petri Net for Requirement Validation of Embedded Real-Time [C]//SAC. 2009: 377-381