计算机科学 2002Vol. 29№. 4

面向嵌入式实时软件系统需求工程环境——SREE*)

Embedded Real-Time Software-Oriented Requirements Engineering Environment — SREE

舒风笛　毋国庆　王　敏

(武汉大学软件工程国家重点实验室计算机科学系　武汉 430072)

Abstract The paper presents the embedded real-time software-oriented requirements engineering environment — SREE, which has been developed by us recently. It involves the whole process of software requirements engineering, including the definition, analysis and checking of requirements specifications. The paper first explains the principles of the executable specification language RTRSM and illustrates it with an example. Subsequently, through comparison with Statemate, the paper introduces the main functions of SREE, illustrates the methods and techniques of checking requirements specifications, especially how to perform simulation execution, combining prototyping method with RTRSM and animated representations. At last, the characteristics of SREE and our future research are discussed.

Keywords Embedded real-time software, Requirements specifications, RTRSM, Prototyping method, Simulation execution

1　前言

软件开发的需求阶段需要严格定义被开发系统的需求规格说明书(Software Requirement Specifications,SRS)。对于软件开发来说,SRS 中的错误将导致开发成本、开发时间的增加,甚至是开发过程的失败,所以,SRS 的正确和可靠极为重要。但由于许多因素,如问题的复杂性、设计人员交流障碍、用户对问题陈述的不完全和不一致以及需求易变性等,需求分析变得十分复杂。为有效地解决这些问题,人们主要围绕着需求分析模型和语言及 CASE 技术和工具进行了相关的研究,引入形式化技术和 CASE 工具的需求工程正不断向着需求工程自动化的方向发展。

作为一类特殊软件系统,嵌入式实时系统软件(Embedded Real-time Software,ERS)具有复杂的动态行为、严格的时间限制和极高的可靠性要求,且与外部环境进行通讯。由于传统的结构化分析、设计方法不能处理 ERS 的动态行为,因此人们以形式化和图形化为主要研究方向,提出了不少新方法,其中具有代表性的有:基于逻辑的形式化方法 RTL[1],ASTRAL[2]等,基于有穷自动机的 Statecharts[3]、Statemate[4]、UML[5,14]和 RSML[6]等。这些方法各有其特点[15],其中不足的主要有:(1)某些需求描述语言或模型的语法和语义过于接近传统的程序设计语言,缺乏高度的抽象能力。(2)除 Statecharts、SCR 和 RSML 等外,一些需求描述语言或模型过于复杂,影响了 SRS 的易理解性和可审查性。(3)许多方法的基于模拟执行的检测只强调了实时系统动态行为的检测,避开了对其它重要特征:时间限制和数据流的检测。

为此,我们设计出采用状态图和模板表示需求,提供数据计算和属性传递的可执行的需求描述语言 RTRSM[7,16],并提出了与该模型相关的原型化检测方法[17]。该语言用类似于 Statechart 的层次有穷状态机来描述系统的动态行为,并将属性附加到状态和事件中。我们已实现了运行在 Windows 平台下的面向 ERS 的需求工程环境 SREE(Software Requirement Engineering Environment)。它是帮助软件开发人员(主要是系统分析员)对 ERS 进行需求分析的辅助工具集,具有图形界面,可用于实时系统的规格说明、分析、设计和文档编制,自动集成并打印输出符合某些标准的 ERS 的 SRS。它具有较强的规格说明检测功能,除一致性、可达性和确定性等检查外,还可进行严格的执行和模拟,检测重要的动态性质,这对保证最终生成的规格说明书的质量和可靠性都极为重要。本文将通过对 SREE 和 Statemate 进行比较来详细说明 RTRSM 及 SREE。

2　可执行的需求描述语言 RTRSM

描述实时系统需求的模型不仅需要具有黑盒性、一致性、简明性、无二义性等一般系统需求模型应具有的性质,还应能有效描述时间限制,提供数据计算功能,并能描述 ERS 与外部环境间的接口以反映其相互间的通讯。

与 Statechart 一样,RTRSM 使用的也是基于状态机模型,其语言为可执行的,通过描述状态的层次和并行关系,把对复杂系统的描述分解为对若干较为简单的子系统的描述。它们都是一种可视化的形式化方法,即语言本质上是可视化的,依赖于少数图形参数;同时,又具有形式化语义,给每个特征(包括图形的和非图形的)以精确、无二义性的语义。对于实时系统,这意味着它应能准备直观而全面的规格说明书,可在计算机辅助系统的帮助下,在任何阶段被分析、模拟和调试。但与 Statemate 的描述语言不同,用 RTRSM 表示的 SRS 实际上由状态图和多个模板组成。

2.1　状态转换图、模板与规则

RTRSM 的状态图与 Statechart 类似,具有超状态、与/或状态、广播机制,状态转换为源状态与目的状态间的连接,包括触发事件、卫士条件和操作(action),其语义也基本上与 Statechart 类似。但 RTRSM 的状态图也有许多不同之处:

(1)引入了变量和事件属性。与 Statechart 中的全局变量不同,我们的变量属于某个模板,且可以被其子模板继承。不同模板变量可同名,若未标明所属模板名,则该变元为从它使用模板起,最近的上层模板(包括其本身)的变元。事件具有 Statechart 中所没有的属性,并能进行传递,可作为只读变量在卫士条件和操作中被使用。

(2)状态图上只标明了转换的触发事件,其卫士条件和操作由专门的编辑器编辑存储到相应的模板中,从而不会因为

*)本研究工作得到国家自然科学基金和高等学校博士点专项科研基金资助。

复杂的卫士条件和操作使得状态图变得拥挤，使之简洁、清晰。Statechart 中的操作为变量的赋值或事件的布尔表达式，而 RTRSM 中的操作不仅包括上述情况，还可广播事件，向外部环境发送控制命令，这为 SRS 的模拟执行和动画演示提供了基本设施。Statechart 中的操作不仅是一种转换的执行结果，而且在进入或转出一个状态时，也可以执行相应的操作。在 RTRSM 中，操作只与转换相关，对于与状态相关的操作都放入相应的转换中。这是因为我们认为，按照先执行操作，后进入目的状态的语义，将操作与转换相对应足以处理进入或转出状态时所要执行的操作，而且可使状态图更为简洁。

RTRSM 的一个模板对应于层次状态机中的某个状态机，其表示形式为表格，包含：模板名、父模板名、初始状态、内部状态说明、状态图、输入接口、功能说明、性能和限制、内部属性和变元定义、内部公式和函数定义以及对应状态图的规则集等一些状态图中无法说明的信息。用户完成的状态图被自动转换为规则集，作为系统内部的形式化表达方式。用户既可不必理解这些规则的语法和语义，也可通过菜单直接激活规则转换子系统来浏览系统中的所有规则或在每个模板中查看该模板的所有规则。在确定的语义解释下，状态图与转换后的规则集等价[18]，故查看规则也可作为需求检查的一种方式。规则集是模拟执行的形式化基础，RTRSM 通过对规则的解释执行来实现 SRS 的执行。一般情况下，一条转换对应一条规则。规则的形式为：

Sour，e(att(e))⇒ Dest，WHEN cond，DO action

其含义为：当 source 为系统的当前状态，触发事件 e 发生且卫士条件 cond 为真时，执行 action(动作)部分，然后转到目的状态 Dest，其中 att(e)表示事件 e 的属性集。规则中的 action 包含一些赋值语句或可用软件部件或可执行模块实现的公式或函数，故我们可利用已有的软件部件或可执行模块来支持需求规格说明的模拟执行。

由于 ERS 具有严格的时间限制，如它与外部环境通讯所受的时间限制，因此在描述其需求时必须考虑这一问题。我们将这些限制信息包含在规则的 action 部分中。实时系统的时钟装置可通过中断方式来产生人为事件(如超时事件)。需求描述模型提供如下函数送入时间量以启动时钟工作：

SetTimer(t，e)：规定的时间 t 被送入时钟装置中，期待的事件必须在时间 t 内发生，否则时钟装置将发生中断(即产生超时事件 e)，然后转入其它处理。

ClearTimer(n)：清除时钟 n 中的剩余时间以避免人为事件的发生。其中，n 为函数 SetTimer(t，time-out)的返回值。

关于通信问题的描述，主要包含两类：并发状态机间的相互通信和状态机与外部环境间的相互通信。前者采用广播机制，所有其它并发子状态机都接收到消息，由接受者自己判断执行状态转换或抛弃无用事件；后者提供专用函数 SendControl(C)来描述从状态机向控制部件发送的信息 C，而控制部件向状态机发送的信息则直接抽象为事件来看待。总之，在 RTRSM 中：

(1)一个模板对应于一个状态机，使系统的功能可相对独立，并保证了系统内部数据的封闭性和属性的可继承性。

(2)模板填写内容形式化要求并不十分严格。模型的简单性较好。

(3)通过模板可将系统或子系统的功能和性能描述及其它信息集成为一体。

(4)把 FSM 变换为形式化的规则表示，增强了模型的严密性，且可通过规则来模拟实时系统的动态行为，是满足可执行性的重要保证。

(5)提供的函数形式能较好地处理系统与外部环境间的通讯和时间限制问题。

表 1　温控系统模板信息

模板名	Temperature-Control	On	AirConditioner	Door
上层模板名	无	Temperature-control	On	On
输入接口	Run：控制台(启动系统运行) Stop：控制台(停止系统运行)		Normal(t)：温度传感器(室温正常) High(t)：温度传感器(室温偏高) Low(t)：温度传感器(室温偏低) Time_out：超时事件	D_Close：房门传感器(房门关上) D_Open：房门传感器(房门打开) Time_out：超时事件
性能和限制	系统能正常运行	系统处在运行状态中	空调正常能工作	能根据房门的情况控制空调并发地协同工作
初始状态	Halt		Off	Close
内部状态说明	Halt：系统处于停止状态 On：系统处于启动状态	AirConditioner：并发状态机 On 的空调子状态机 Door：并发状态机 On 的房门子状态	Off：空调处于停止状态 Cold：空调处于制冷状态 Heat：空调处于加热状态	Close：房门处于关闭状态 Open1：房门处于打开状态 Open2：房门打开已超过一个规定时间
内部属性和变元			T：温度阈值	DoorOpen：房门打开为1，否则为零
内部公式和函数			SendControl(命令)：向动画显示发送命令	SendControl(命令)：向动画显示发送命令
规则集	见图2			
功能说明	控制系统停止和运行	并发超状态机 On，包含系统运行时的两个控制对象	根据温度传感器上的温度高低和房门的开关情况启动空调装置制冷或制热	判断房门的开关状态，在房门开启时间达到一定时间时发送超时事件停止空调的运作

2.2 实例说明

为便于理解，本节将通过温控系统实例来说明如何应用我们提出的需求描述模型生成室内温度控制系统(简称温控系统)的SRS。

在温控系统中，系统自动控制空调装置进行制冷或制热以调节室内温度。系统最初处于Halt状态，当操作员按下"Run"按钮后，系统开始运行。当室温高于或低于规定的正常范围(22℃～26℃)且门关闭时，系统启动空调装置制冷或制热。同时，该系统也将根据房门的开关情况(由某一传感器指示)控制空调装置工作：当房门打开且超过系统规定的时间限制，系统将停止空调装置工作。当操作员按下"Stop"按钮后，整个系统将被强行停止运行。我们约定：(1)启动该控制系统运行前，房门关闭，空调未运行。(2)空调装置制冷或制热时，房门被打开但在规定的时间内关闭，不影响空调工作。

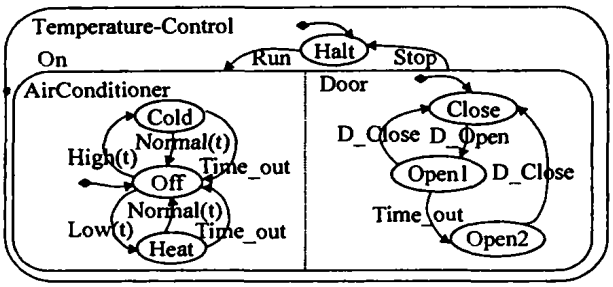

图1 温控系统的状态图

系统的控制对象：空调和房门是相对独立的。我们用相互并发的两个超状态AirConditioner和Door来分别对应这两个对象。空调和房门，及其相互间的并行关系是系统描述的主要部分。图1为该系统的状态图，其中附加在有向弧上的名字是事件名。温控系统的需求规格说明包括4个模板：最上层根模板、空调模板、门模板以及一个表达空调和门组成并发关系的模板。各模板的详细内容如表1所示。

从上述模型介绍和实例说明中可看出运用上述模型产生的需求规格说明书由状态图和模板组成，状态图能直观地表达状态间的层次和并发关系，而模板能表达状态图中一些不能包含的信息，如变元信息、性能限制等。一个状态机对应于一个模板，各个模板包含的规则集与它描述的状态机等价，规则集是状态图的形式化表示[18]。

```
模板 Temperature-Control 的规则集为：
  Temperature-Control，____⇒Temperature-Control. Halt
    这是一条默认状态转换
  Halt，Run⇒On Cond b<=0 Action a=5345+54353;‖b=b-34‖
  Halt，Stop⇒on Action b=a%c
模板 On 的规则集为：
  On，____⇒AirConditioner‖Door
模板 AirConditioner 的规则集为：
  AirConditioner，____⇒AirConditioner. Off 这是一条默认状态转换
  Off，High(t)⇒AirConditioner. Cold Cond High. t>26 Action Send-
    Control(cold);‖T=26
  Off，Low(t)⇒AirConditioner. Heat Cond low. t<22 Action Send-
    Control(heat);‖T=22
  Cold，Normal(t)⇒AirConditioner. Off Cond T>normal. t Action
    SendControl(stop);‖T=Normal. t
  Cold，Time_out⇒AicConditioner. Off Action SendControl(stop)
  Heat，Normal(t)⇒AirCondtioner. Off Cond T<Normal. t Action
    SendControl(stop)
  Heat，Time_out⇒AirConditioner. Off Action SendControl(stop)
模板 Door 的规则集为：
  Door，____⇒Door. Close 这是一条默认状态转换
  Close，D_Open⇒Door. Open1 Action a=on. a+10
  Open1，D_Close⇒Door. Close
  Open1，Time_out⇒Door. Open2
  Open2，D_Close⇒Door. Close
```

图2 温控系统规则集

3 SREE的整体结构与功能

SREE是基于RTRSM的辅助需求定义和分析的集成工具环境，主要包括：需求说明编辑、规则自动生成、需求检查和生成四大功能部分。系统整体结构如图3所示。

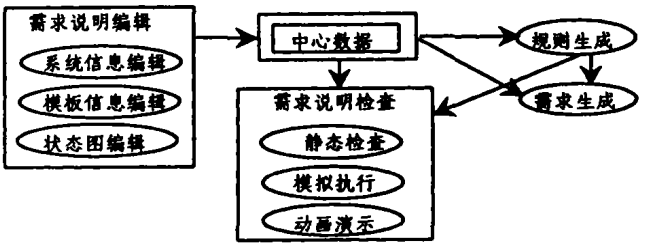

图3 系统整体结构图

系统信息编辑器、模板编辑器和层次状态图编辑器被集成到一个一致、可协同工作的用户界面下，以鼠标或键盘驱动，使用相当方便。如图4所示，用户正在编辑装配系统的状态图，并准备进入模拟执行子系统检查该状态图。系统信息编辑器用于输入SRS中与目标系统整体相关的各个方面，包括：系统名称、编制单位、编制名称、概述、系统总体概要、详细需求、运行环境和索引等，其中详细需求包括性能要求和限制、实时控制系统功能需求、需求详细描述、输入和输出、故障处理等，其中运行环境包括设备环境、软件支持环境、接口等，除需求详细描述中的模板集和状态图外，大都使用文本和表格输入。与Statemate中的状态图编辑器一样，层次状态图编辑器具有作图、编辑、选择功能，并在用户绘制状态图时进行图形语法的初步检查，比如一个状态转换必须起源且终止于某个状态。模板编辑器用于编辑模板包含的各项信息，如图5。它能自动将层次状态图中的超状态和模板对应起来并保持超状态和模板数据的一致性，使用户不必考虑两者间数据的同步更新问题。随着用户与上述编辑器的交互，所得到的信息被逐步存储，并由规则转换子系统将状态图转换为形式化的规则集，供进一步的修改、设计和需求检查、验证及SRS的生成。该子系统使用户不必阅读形式化符号和规则，并确保状态图信息与规则集信息一致。

SREE提供了较为强大的检查功能，使用户能分析、调试编辑得到的SRS。根据检查方法的不同，需求说明检查可分为：需求检查、模拟执行和动画演示三个子系统，详见4.3节。除此之外，与Statemate不同，SREE在进行模拟执行前，还要对SRS中转换的卫士条件和操作及变元、属性的定义等进行语法和部分语义检查，若有错误则显示错误信息，以便用户修改。

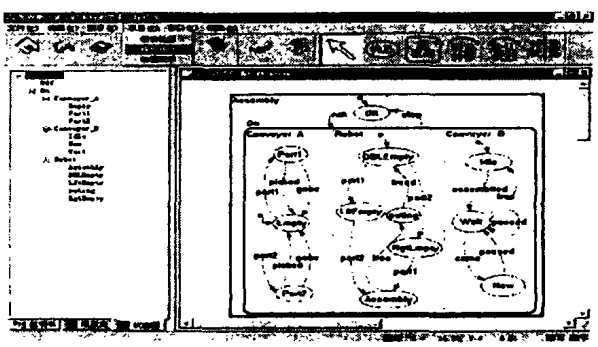

图4 需求工程编辑界面

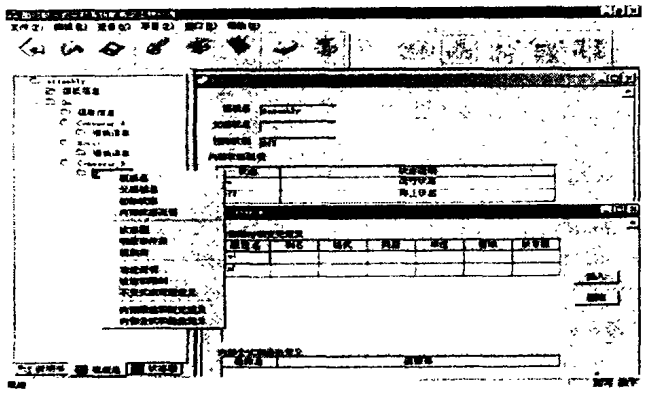

图5 模板编辑视图

当用户完成所有的需求分析后，就可以运行需求规格说明书生成子系统来产生最终的SRS。该系统输出SRS符合国家军标的标准。

4 需求规格说明的检测

SRS中的错误若不能及时发现、将导致开发成本、时间的增加，甚至是灾难性后果，因此对其进行全面检测，发现其中错误并验证是否满足系统用户的要求极为重要且必不可少。对SRS的有效检测必须结合其所使用的需求描述模型或语言的特点。SREE使用的需求描述语言为RTRSM，其检测内容可归结为一致性、完全性和安全性检测。对于RTRSM，一致性包含两方面，一是在无并行情况下，任一事件的发生不能引发多个转换或规则同时执行，导致在某一时刻状态机中出现多个当前状态；二是在并行情况下，任何两个转换或规则在执行过程中不允许发生冲突。完全性指每一个可能发生的事件及其所引发的响应都必须被明确说明。安全性指实时系统不但能在规定的时间内处理正常事件，也必须能处理非正常事件(如干扰等)。关于ERS的需求规格说明检测方法的研究已有不少[8~13]，下面，我们将结合RTRSM的特点和上述内容来探讨如何将形式化分析和模拟执行相结合来检测基于RTRSM的SRS。

4.1 形式化的需求检查

与Statemate中基于穷举的动态测试不同，SREE对一致性和完全性的检查采用的是静态的形式化分析方法，主要通过规则的解释来进行，具体包括：

(1)可达性检查：判断是否每一状态都存在一条从初始状态开始的路径可以到达。

(2)初值检查：检查每一个模板是否都有确定的初始状态。

(3)非确定性检查：对状态图中存在的非确定性状态转换进行检查，包括两种情况：一事件触发多个以当前状态为源状态的转换同时执行；两个结束同一状态的转换同时执行，且其目的状态既不并行，也不相同。

(4)并行冲突检查：检查并行状态机中的状态转换是否相互冲突，即在并行状态下，两个由同一事件触发的并行执行的状态转换中可能存在变元的读写冲突。

(5)连通性检查：判断指定的两个或多个状态间是否可能存在相互连通的路径。

(6)路径检查：从初始状态起，根据输入事件判断是否存在可能的路径。

Statemate中动态穷举的方法对于复杂系统而言是不可行的，即为局部可行；而我们采用的静态检测方法假设转换的卫士条件为永真，且未考虑其操作，所以虽然全局可行，但不全面，所检测到的路径都只为可能路径。为此，我们正在探讨构图覆盖的检测方法，它将和已有静态方法相结合，实现对重点子系统的充分检查。

4.2 模拟执行

模拟执行即使用一模拟控制语言来执行允许用户模拟待开发系统的环境，并交互式或批处理地执行该SRS，为一种原型化的检测方法。SREE的模拟执行是在通过一定的语法和部分语义检查后，根据状态图和模板信息模拟执行层次状态图的动态行为，以对该需求说明进行检查，发现其中的语法语义错误并对有关信息加以显示输出，帮助用户判断该SRS是否满足要求。由于RTRSM的可执行性能较好地支持原型化方法，且作为原型的SRS是用RTRSM书写的，故修改原型时能避免麻烦和不一致性。模拟执行的形式化语义基础为文[16]中提出的基于控制流和数据流的动态执行模型DERTS，它保证了模拟执行在语义基础上的可验证性。

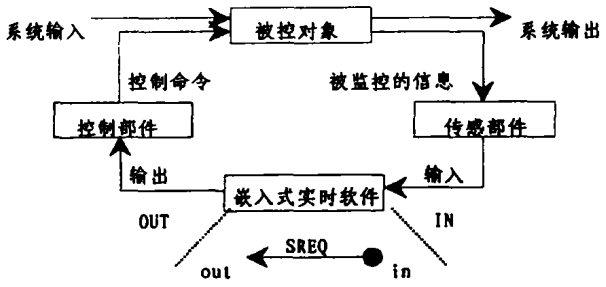

图6 嵌入式实时系统的模型

实时系统的系统模型[6]如图6所示，包括被控对象(CB)，传感部件(SC)，控制部件(CC)和ERS四部分。SC将从CB获取的监控信息作为输入传送给ERS，后者进行相应处理，并将处理结果作为控制信息传送给CC，再由CC发出控制命令控制CB的动作。由于ERS与外部环境(如传感部件、控制部件等)紧密相关，为在模拟执行ERS的SRS时减少对SC和CC部分的依赖，我们可再将ERS分为：in，out和SREQ，其中in是将实际输入IN通过某种转换得到的ERS可接受的输入；out是将模拟执行中产生的结果转换为CC所需的实际输出信息，SREQ是代表ERS的需求规格说明。这样，SRS的模拟执行可独立于实际外部环境，且可获得与实际控制过程相同的结果。此外，SC和CC的变化可通过in和out反映到SREQ中，使得不必等到SC和CC正式形成后再修改SREQ和实际的软件系统。基于上述考虑的模拟执行在脱离实际环境的情况下，既可达到实际执行的结果，又是有意义的。故在建立ERS的需求规格说明时首先需定义系统的边界和识别外部环境提供的所需信息，即in和out，从而可在不考虑硬件的情况下，使用模拟的输入信息来检测ERS的功能。

SREE中模拟执行的实现方法与Statemate的不同：后者是由从SRS转换得到的代码来控制的，先自动代码转换，由SRS到能被连接到真实或模拟的目标环境的高级编程；而RTRSM是基于分层状态机，且以规则的形式表示状态图，故其生成的SRS的操作语义可通过规则的执行，由分层状态机给予解释，即模拟执行ERS的SRS实际上就是执行一系列规则。通常，ERS以任务(task)作为其运行的基本单位，系统的运行可通过一组相互作用的任务来实现，且它们可动态改变系统的执行状态。在模拟执行中，我们把一个模板作为一个

任务，每个模板任务的执行是根据该任务的当前状态、卫士条件和发生事件，执行相应的动作，然后产生下一当前状态。当任务对应的事件未发生时，则该任务处于等待状态。

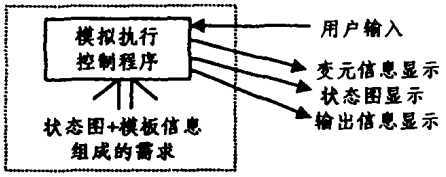

图7 模拟执行控制过程

模拟执行子系统为用户提供了一种调试方式，既可交互式运行，即每一步都要求由用户确定事件及其属性值；也可以批处理的方式运行，即用户一次设定好初始状态和属性值及事件序列后，系统自动运行。该子系统由模拟执行控制(SC)和输入/出部分组成。SC 根据接收到的来自外部的事件和数据激活某些模板任务；输入部分负责接收用户的输入，包括变量的赋值和事件及其属性值；输出部分则显示执行情况、输出执行结果。与 Statemate 不同，我们没有设置专门的数据库存储执行结果，而是除了在状态图中用特殊颜色或闪烁等方式显示系统所处的当前状态、正在发生的状态转换等信息外，更新模板变元的当前值，并在信息输出窗口以表格形式显示执行过程中的各种状态转换信息和错误信息，如：执行了的转换，转换执行过程中发生的错误、触发事件和向控制部件发送的控制命令，某些转换未执行的原因等，并对满足条件但未执行的转换予以说明以帮助用户判断是否存在非确定状态转换。这些执行信息比 Statemate 所提供的更为丰富。模拟执行控制过程如图7所示。

我们在 SREE 中还实现了与动画演示程序的接口，该接口能与动画演示程序共同作用以状态图和动画相结合的形式生动地展现系统的动态行为，使得用户能更直观、更形象地判断已经过静态需求检查和动态模拟执行的需求说明书是否完全符合该系统的要求。该部分由模拟执行程序和用户制作的动画演示程序共同组成。模拟执行程序从动画演示程序获得事件，通过模拟执行控制程序的处理，一方面在状态图中显示系统所处的当前状态和正在发生的状态转换，同时向动画演示程序发送控制命令，使动画演示程序执行相应的操作，从而用状态图和动画界面来共同展现系统。我们已设计实现了温控系统、装配系统和火电厂锅炉炉水自动防垢除垢系统等多个实例的动画演示。

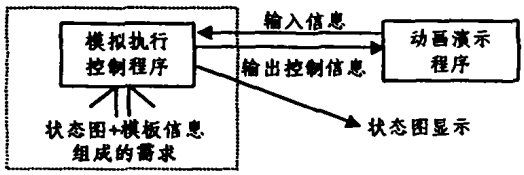

图8 SC 在动画演示中的作用

SC 不仅用于模拟执行子系统中，而且也可重用于路径检查和动画显示子系统中。与图7相比，路径检查不需输出变元信息；SC 在动画演示程序中的控制作用如图8所示。该子程序可理解为规则的解释器，所以可把规则自动生成器理解为可由该解释器解释执行的代码自动生成器。在用户输入需求说明书，并进行全面检查后，只需再提供输入、输出转换接口，将从实际环境中采集到的控制子系统所需的现场信息转换为 SC 所能接收的输入，并将 SC 产生的输出，即处理结果作为控制信息，转换为实际环境所能接收的控制命令。这样，由需求说明书和 SC，则可实现对现场的控制。模拟执行控制程序加上用户输入的状态图和模板消息组成的需求实际上构成了嵌入式实时软件的原型。

结束语 我们已开发出面向 ERS 的需求工程开发环境 SREE。该环境简单，易于使用，不但能用于定义实时嵌入式软件的需求、产生 SRS，还可把它作为模拟执行的原型，与直观、可视化效果良好的动画演示相结合，对其进行较为全面的检查。利用该环境，我们已实现了对室温自动控制系统、机器人自动装配系统和火电厂锅炉炉水自动防垢除垢控制系统的需要规格说明进行检测的实验，且效果较为理想。

与同样基于自动机模型的 Statemate 一样，我们提出的需求描述模型和检测方法及其实现环境有如下特点：

(1)生成的规格说明书为可执行的，支持原型化方法，为可视化的形式化方法；

(2)需求规格说明的描述和各种检测构成一个整体，具有较强的分析能力，包括一致性、确定性检测等以及原型化的模拟执行。把 SRS 作为原型，进行可视化的模拟执行，展现系统重要方面，并能对错误直接加以修改。所提供的对 SRS 进行分析、检测的工具为其它一些建模工具，如 UML 所不具备的。

我们工作的独特之处在于：

(1)SREE 采用的是面向过程的需求描述方法，该方法通常主要是研究系统 I/O 转化的方式，对数据本身及控制方面不十分重视，而 SREE 通过引入事件属性，增强变元的使用，提供在转换的操作中发送控制信息的函数，从一定程度上增强了对数据和控制的描述和处理功能。

(2)SREE 中规格说明书的具体表现形式为模板和状态图，用户只需输入状态图及模板中的一些状态图中无法表示的信息，加上 SREE 对数据和控制方面的描述功能，它能较为充分地描述整个系统的各重要方面，与 Statemate 相比，更易于使用，且减少冗余，不用考虑数据一致性问题。此外，将状态图自动转换为形式化的、具有部分推理功能的规则，为系统提供了实现分析、检查和验证的精确和严格的语义基础。

(3)用于实现检测的环境，仅使用了可执行的需求描述语言和输入/出接口等，不需要过于复杂的支持原型化方法的环境，保证了模拟执行的简单性和易使用。

(4)通过提供 SRS 与外界环境间的模拟接口，不仅真正实现了动画演示功能，为用户提供了可视性较强的验证方式，而且使 SREE 的模拟执行程序加上用户输入的状态图和模板信息实际上构成了一个 ERS 原型。

(5)由于实现检测是通过解释环境内部的规则，因此在规则中所使用的一些计算和处理可以与软件部件化研究相结合，例如可用软件部件来实现规则中的执行动作部分，探索从需求规格说明到代码生成的可行途径。

建立合适的需求规格描述语言和模型，既具有较好的描述能力，也能被大多数用户接受，而且能对其描述的需求规格说明进行自动分析来发现错误或运用原型化的方法来形象地展现系统的重要方面，从而产生高质量的需求规格说明，甚至直接实现从需求到代码的理想飞跃，这一直是需求工程研究的目标。我们虽进行了一些有益的探索，但离实现该目标还相差甚远。我们希望在现有研究的基础上进一步完成如下工作：

(下转第14页)

$-1\}$.其中 E_i 为同态加密函数。

3. 每个投票者分别用投票中心 C 和投票中心 C 的公钥加密 $X_i^{(1)}$ 和 $X_i^{(2)}$。

4. 投票中心收到 $X_i^{(1)}$ 和 $X_i^{(2)}$ 后分别用 E_1 和 E_2 对其加密，与前面投票者公布出来的数字比较，验证其真实性。

5. 每个投票中心 j 把所有的 $X_i^{(j)}$ 加起来模 q 得到 t_j，公布结果，而最终的投票结果是 $T=t_1+t_2$ 在这个过程中每个投票者都可通过计算：

$$E_j(t_j)=\prod_j E_j(X_i^{(j)})$$

来验证投票中心是否正确计票。

这种协议可以很容易扩展到 m 个投票中心，具体的扩展方法见文[2]。在此协议中只要有一个投票中心是诚实的，选票的安全性就可以得到保证。对此协议的攻击需要计算群上的离散对数问题，为了提高安全性，我们还可以使用椭圆曲线上的离散对数。

存在的问题及展望 现在已经有许多关于电子投票的各种特色方案的提出，但是没有一个方案能够满足电子投票的所有安全要求，尤其是在防止投票中心伪造选票这一方面，因此利用安全多方计算的投票协议越来越受到人们的关注，成为目前研究的主流。我们还可以看到无论是利用匿名信道或是安全多方计算的投票协议都要用到零知识证明，而即使把交互式的零知识证明转化为非交互式的，在投票的过程中我们依然面对着较大的通信量和计算量，而对于投票者来说除保密性、匿名性等要求之外，简单方便是用户最希望得到的。因此如何简化用户的投票过程，降低投票协议进行的通讯和计算成本，也是电子选举协议研究中不可忽视的问题。

随着对电子投票方案研究的深入，在国外已经出现了一些实验用的电子投票系统，比如 California 的 Voting By Mail[5]，由 MIT 研究实现的 EVOX[6]，Princeton 大学的校园选举系统，Washington 大学的 Sensus 系统等。其中 EVOX 和 Sensus 系统都是基于前文提到的 FOO 投票协议设计并实现的，不过这些系统的实现都是基于匿名信道的，同样具有对选举的组织机构依赖性过大的缺点，因此基于安全多方计算的电子选举系统有待于我们的开发和实现。可以预见，在不久的将来，随着密码学和网络技术的发展，我们可以研究并实现出真正实用、方便的电子选举系统。

参 考 文 献

1 Fujioka, Okamot, Ohta. A Practical Secret Voting Scheme for Large Scale Elections

2 Kazue Sako, Joe Kilian. Secure Voting Using Partially Compatible Homomorphisms

3 Benaloh J C, Tuinstra D. Distributing the Power of a Government to Enhance the Pirvacy of Voters

4 Fiat A, Shamir A. How to Prove Yourself: Practical Solutions to Identification And Signature Problems

5 Herschberg M. Secure Electornic Voting Using The World Wide Web

6 DuRette B W. Multiple Administrators For Electronic Voting

7 Benalh J C, Turnstra D. Receipt-Free Secret-Ballot Elections

8 Park C, Itoh K, Kurosawa K. All/Noting Election Scheme and Anonymous Channels

9 Sako K, Kilian J. Receipt-Free Mix-Type Voting Scheme - A Practical Solution to The Implementation of a Voting Booth

10 Brassard G, Chaum D, Crepeau C. Minimum Disclosure Proofs of Knowledge

（上接第8页）

（1）从理论方面完善上述的需求描述及检测方法和已研制出的需求工程环境，并且也打算将此环境应用于更多、更复杂的实时嵌入式软件实例中，以使我们的需求描述及检测方法和环境更具有实用性和发挥较大的作用。

（2）将部件化的方法和技术运用到模拟执行中去，严格定义出规则的 DO 部分的语法，解释调用嵌入的组件，探索从需求到代码的可行途径。

参 考 文 献

1 Jahanian F, Mok A K. Safety Analysis of Timing Properties in Real-Time Systems. IEEE Trans. On Software Eng., 1986, 12(9): 890～904

2 Porisini A C, Ghezzi C, Kemmerer R A. Specification of Real-time System Using ASTRAL. IEEE Trans. on Software Eng., 1997, 23(9): 572～598

3 Harel D. Statecharts: A Visual Approach to Complex Systems. Science of Computer Programming, 1987, 8(3): 231～274

4 Harel D, et al. Statemate: A Working Environment for the Development of Complex Reactive Systems. IEEE Transactions on Software Engineering, 1990, 16(4): 403～413

5 Douglass B P. Designing Real_Time Systems with UML. http://www.embedded.com/98/9803fe2.htm.

6 Leveson N G, et al. Requirements Specification for Process Control Systems. IEEE Trans. on Software Eng., 1994, 20(9): 689～707

7 Wu Guoqing, Xiao Haifeng, Zheng Peng, et al. Specifying Requirements of Real-Time System with Rules and Templates. Wuhan University J. of Natural Sciences, 2000, 5(5)3: 278～284

8 Thompson J M, et al. Specification-Based Prototyping for Embedded System. Software Engineering Notes, 1999, 24(6): 163～179

9 Berzing V, Luqi, Yehudai A. Using Transitions in Specification-Based Prototying. IEEE Transaction on Software Engineering, 1993, 19(5): 436～452

10 Kramer B, Luqi, Berzins V. Compositional Semantic of a Real-Time Prototyping Language. IEEE Transaction on Software Engineering, 1993, 19(5): 453～477

11 Stocks P, Carrington D. A framework for specification-based testing. IEEE Trans. on Software Eng., 1996, 22(11): 777～793

12 Atlee J M, Gannon J. State-Based Model Checking of Event-Driven System Requirements. IEEE Trans. on Software Eng., 1993, 19(1): 25～39

13 Heimdahl M P E, Levison N G. Completeness and Consistency in Hierarchical State-Based Requirements. IEEE Transactions on Software Engineering, 1996, 22(6)

14 刘超，张莉编著. 可视化面向对象的建模技术——标准建模语言 UML 教程. 北京航空航天大学出版社, 1999

15 卢梅，李明树. 软件需求工程——方法及工具评述. 计算机研究与发展, 1999, 11(11)

16 毋国庆，胡春丽，蔡持峰，何峰. 面向嵌入式实时系统的需求模型. 计算机工程与科学, 1999(A1)

17 胡春丽，毋国庆，何峰，等. 从状态图到规则的转换. 小型微型计算机系统, 2001(6)

18 毋国庆，朱立松，王敏，蔡持峰. 嵌入式实时系统的软件需求测试. 软件学报(待发表)