

带有数据约束的信息物理融合系统的建模方法

卜星辰,曹子宁,胡名光

(南京航空航天大学 计算机科学与技术学院,江苏 南京 211106)

摘要:信息物理融合系统由具有连续性的物理系统和具有离散性的信息系统组合而成,是一种复杂的混成系统。针对现有的建模方法不能够直接对信息物理融合系统进行建模,采用基于模型的体系结构建模方法,对信息物理融合系统的不同部分采用不同的建模方法进行建模,将不同的建模语言进行相应的转换整合成一个完整的系统。AADL建模语言提供了较为全面的图表和结构,适合用来对系统的体系结构进行建模,针对于其无法描述概率迁移的问题,扩展了概率行为附件。作为AADL的补充,Modelica语言可以利用微分代数方程对物理系统的连续动态性进行建模。根据AADL与Modelica的映射规则,对AADL的属性集进行相应的扩展,将Modelica建立的模型转换为AADL模型,使得信息系统和物理系统融合。在Modelica与AADL建模的基础上,采用Z规范对信息物理融合系统交互过程中产生的大量数据进行形式化的约束。

关键词:信息物理融合系统;Modelica;AADL;Z规范;概率离散性;连续行为特性

中图分类号:TP301

文献标识码:A

文章编号:1673-629X(2020)10-0079-07

doi:10.3969/j.issn.1673-629X.2020.10.015

A Modeling Method of Cyber-physical System with Data Constraints

BU Xing-chen, CAO Zi-ning, HU Ming-guang

(School of Computer Science and Technology, Nanjing University of
Aeronautics and Astronautics, Nanjing 211106, China)

Abstract: The cyber-physical system (CPS) is a complex hybrid system composed of continuous physical system and discrete information system. Existing modeling methods are not able to directly model the CPS. With the model-based architecture modeling method, different modeling methods are used to model different parts of the CPS, and then different models modeled by different modeling languages are transformed into a unified system model. The AADL modeling language provides a comprehensive diagram and structure that is suitable for modeling the architecture of the system and the probabilistic behavioral annex is extended to address the problem that the AADL cannot describe probabilistic transition. As a complement to AADL, the Modelica language can model the continuous dynamics of a physical system using differential algebraic equations. To merge the AADL model and Modelica model, the attribute set of AADL is expended to convert the model established by Modelica into AADL according to the mapping rules of AADL and Modelica. Based on the combination of Modelica and AADL, the Z specification is used to formally constrain the large amount of data generated during the interaction of CPS.

Key words: cyber-physical system; Modelica; AADL; Z specification; probability transition; continuous dynamic

0 引言

信息物理融合系统(cyber-physical system, CPS)^[1]是一种集成的动态系统,系统同时具有离散动态性的计算单元和连续动态性的物理系统。CPS将物理系统的连续变化引入到计算单元的离散状态变迁中,物理系统中变量的演变可以触发计算单元中的离散事件,而计算单元中的离散事件也可以改变物理系

统中变量的变化模式,这种系统间的相互关联性极大地增加了对CPS建模与验证的难度。现有的技术还不能够直接对CPS进行建模来检查系统的一致性和连续动态性。

因此,需要一种新的建模方法——能够同时刻画系统的离散性和连续性。现有的建模语言中,AADL^[2-3]作为嵌入式系统的建模语言,不仅能够对系

收稿日期:2019-10-13

修回日期:2020-02-18

基金项目:国家航空科学基金(20185152035,20150652008);中央高校基本科研业务费(NJ2019010)

作者简介:卜星辰(1995-),男,硕士研究生,CCF会员(88989G),研究方向为形式化方法;曹子宁,教授,博导,研究方向为形式化方法、人工智能。

统的体系结构进行详细刻画,还能够支持属性集和行为附件的扩展。文献[4]扩展了不确定附件对系统中的不确定性进行建模,文献[5]提出了混成附件对混成系统的连续行为进行建模。而 Modelica^[6-8]作为面向对象的物理系统的建模语言,能够利用微分(代数)方程对物理系统的连续性进行详细刻画。采用 AADL 对信息系统建模,采用 Modelica 对物理系统进行建模,然后再将所建立的模型通过连接机制连接起来,能够充分刻画 CPS 的连续性和离散性。该方法是一种基于模型的体系系统^[9-10]的建模方法,体系系统中每个子系统都可以独立运行,子系统需要与别的子系统进行相应的交互,来满足指定的系统需求^[11]。文献[12]通过定义 Modelica-AADL 的接口将两者结合起来,使得物理系统和计算过程相融合。文献[13]根据 Modelica 与 AADL 的映射关系,将 AADL 模型转换为 Modelica 模型。Z 规范为软件系统的规格说明提供了一套严密的数学描述方法,主要用来对软件系统的需求、功能、规格等进行正确的描述和验证。文献[14-15]采用 Z 规范对系统中产生的数据及状态变迁进行数据约束。

文中使用 AADL、Modelica 分别对 CPS 中的信息系统和物理系统进行建模。针对 AADL 无法描述状态间的概率行为事件,提出了带有通信机制的概率行为附件。Modelica 语言利用微分方程对物理系统的连续动态性进行建模。根据 AADL 与 Modelica 的对应关系,对 AADL 的属性集进行相应的扩展,将 Modelica 建立的模型转换为 AADL 组件,使得信息系统模型可以和物理系统模型进行组合,并使用 Osate 对 CPS 中的延迟时间进行分析。在 AADL-Modelica 建模的基础上,采用 Z 规范对系统交互过程中产生的数据进行形式化的约束。

1 AADL 扩展的概率行为附件

CPS 与人们的日常生活紧密联系,然而在现实中很多系统都不是精确无误的,或多或少地存在一些不确定的影响因素。针对现有 AADL 还不能够对信息系统中的概率行为进行建模的问题,本节通过定义附件子语言的方式提出了一个轻量级的拓展语言——概率行为附件。

为了明确概率行为附件的使用方法,采用 EBNF (Extended Backus Naur Form) 范式来定义概率行为附件的语法。在 EBNF 中,关键字通过粗体字表现出来,可替换元素通过“|”进行分隔,同一组元素用“()”包围,可选择的部分通过“[]”进行覆盖,“{}+”和“{}*”分别表示所包含集合中的一个或多个元素和零个或多个元素。概率行为附件语言主要由 variables、

states 和 transition 三大模块组成。

```
Probability Annex ::= { * *
    variables { variables_declaration } +
    states { states_declaration } +
    transition { transition_declaration } +
    * * }
```

1.1 variables 模块

概率行为附件内的变量及其数据类型在 variables 模块内进行声明,用来刻画该组件在某一时刻的性质。变量的语法定义如下所示:

```
Variables_declaration ::= variable_identifier | , variable_identifier |
    * : data_componet_classifier_reference
```

数据类型由分类器进行引用并指派给适当的 AADL 数据组件。引用的外部数据组件必须与当前组件处在同一个包下,或者使用关键字 with,将另一个包的数据组件导入到当前的组件范围内。

1.2 states 模块

states 模块用来定义系统中出现的状态集合和系统的初始状态。状态的声明包括初始状态声明和状态集合声明。状态的语法定义如下所示:

```
States_declaration ::= initial_state_declaration, states_declaration
initial_state_declaration ::= Initial_state_identifier; initial state;
states_declaration ::= { state_identifier } + ; states;
```

1.3 transition 模块

transition 模块定义状态间的变迁关系,一个状态如果满足相应的条件将会跳转到相应的后继状态,但跳转到的后继状态具有不确定性。状态变迁的语法定义如下所示:

```
transition_declaration ::= state - [ guard ] -> prob : state [ action ] { +
    prob : state [ action ] } *
```

其中 guard 表示状态发生变迁所要满足的条件,prob 表示当前状态跳转到下一个状态的概率约束,prob 的取值范围在[0,1]之间,所有可能跳转到的后继状态的 prob 值相加为 1。状态变迁表示如果当前状态满足相应的条件,将在所有的后续状态中以一定的概率挑选一个状态进行跳转,同时执行相应的动作[action]。

guard 条件可以是当前状态内变量的布尔表达式,也可以是当前状态执行的输入或输出动作。guard 的语法定义如下:

```
guard ::= data_expression | control_communication
data_expression ::= data_communication and bool_expression
```

guard 可以描述两种不同的状态变迁场景,第一种是因为状态内变量的值满足谓词条件约束,从而引发状态上的变迁。信息系统中连续变量的值都是通过数

据传输端口从物理系统中获得,所以 guard 中的 data_expression 由 data_communication 和 bool_expression 作与运算 (and) 得到, data_communication 表示从相应的数据端口接收数据,当接收到的数据值使得变量的谓词条件约束 Bool_expression 为真时,状态发生迁移。这种情况下发生的状态迁移是一种概率选择事件,即状态变迁时可能会有多个后继状态的选择,跳转到相应状态的可能性大小即为对应的概率值。

第二种发生状态变迁的场景是因为当前状态执行了发送或接收动作,从而引发了状态上的变迁。执行动作带来的变迁同样是一种概率选择事件,从当前状态跳转到的后继状态可能有多个,且跳转到相应状态的可能性大小即为对应的概率值。

布尔表达式 (Boolean_expression) 由布尔表达式通过二元运算符与 (and)、或 (or) 和一元运算符非 (not) 组合而成。比较关系由数学表达式结合了标准的比较运算符 (=, <, >, <=, >=, !=) 组合而成。布尔表达式主要用来描述信息系统中发生状态迁移时的变量谓词约束。布尔表达式的语法定义如下:

```
Boolean_expression ::= ture | false | Boolean_expression |
Boolean_expression { and Boolean_expression } + | Boolean_expression { or Boolean_expression } + | not Boolean_expression
| comparison
Comparison ::= [ numeric_expression comparison_symbol numeric_expression ]
Comparison_symbol ::= = | < | > | <= | >= | !=
Numeric_expression ::= numeric_term | numeric_term -
numeric_term | numeric_term / numeric_term | numeric_term mod
numeric_term | numeric_term + numeric_term |
numeric_term * numeric_term
Numeric_term ::= [ - ] ( numeric_literal | variable_identifier )
numeric_literal ::= integer_literal | real_literal
```

状态变迁时可以给跳转到的后继状态指定相应的执行动作 action, action 动作主要用于对状态中的变量进行重赋值。action 的语法定义如下:

```
Action ::= action { , action } *
Action ::= assignment
Assignment ::= variable_identifier = ( integer_number | real_number | boolean )
```

1.4 信息系统中的交互建模

信息系统不仅与物理系统有着广泛的交互,还与其余的信息系统有着密切的联系。因此对信息系统与其余系统之间的通信行为进行准确建模是对信息系统进行建模的重要组成部分。AADL 组件的通信依赖于组件类型声明中的端口。对于数据信号的传输,使用数据端口 (data port) 进行建模;对于控制信号的传输,采用数据事件端口 (event data port) 进行建模。相互

通信的端口在互补方向上是成对出现的。端口和端口通信的语法定义如下:

```
Event_port ::= event_port_identifier { , event_port_identifier }
* : data_componet_classifier_reference
Data_port ::= data_port_identifier { , data_port_identifier } * :
data_componet_classifier_reference
Data_communication ::= data_port_identifier? ( [ variable_identifier ] )
Control_communication ::= event_port_identifier ( ? ! )
( [ variable_identifier ] )
```

(? !) 分别表示从端口输入/输出信号,信息系统的端口主要用来接收物理系统中的变量值,所以信息系统中的数据端口只有接收动作。而信息系统不仅可以发送控制信号给物理系统,还可以与别的信息系统进行控制信号上的交互,所以信息系统中的控制端口可以执行输入/输出动作。

2 Modelica 向 AADL 的转化

CPS 中的物理系统具有连续变化的行为特性,使用 Modelica 可以很方便地利用数学方程对物理系统中的连续变化进行建模,通过将所建立的模型编译成方程组,对方程组进行求解可以得到系统的仿真结果。

2.1 Modelica 与 AADL 的对应关系

Modelica 是一种面向对象的建模语言,主要构成元素是类和对象。AADL 中的组件可以分为类型声明和实现两个部分,一个组件类型声明可以拥有多个组件实现。AADL 中的组件类型和组件实现与 Modelica 中的类和对象一一对应。同时 AADL 与 Modelica 都支持继承机制,这为两种建模语言的相互转换奠定了基础。

Modelica 主要用来对物理系统进行建模,而物理系统相当于计算机系统硬件设备,因此将 Modelica 中的类转换为 AADL 中的设备 (device) 类型声明,实例化后的对象转化为设备对应的实现;在 Modelica 中数据信息的交互主要是通过 connector 来实现,而 AADL 中的 feature 可以用来描述组件之间的数据和事件的交互端口,同时能够刻画数据/事件的传输方向,这两者可以进行相互转换;Modelica 中的变量可以通过 AADL 中的参数 (parameter) 表示;Modelica 中的函数 (function) 可以用 AADL 中的子程序 (subprogram) 来实现,同时函数中的参数可以用 AADL 中的参数进行标识;Modelica 自身支持的多种不同的数据类型,在 AADL 中可以使用 datatype 对各种不同的数据类型进行定义,能够满足相互转化的需求。而 Modelica 中常用的建模元素常量和方程在 AADL 中没有相对应的对象,因此通过对 AADL 的属

性集进行扩展引入新的属性和属性类型,使得 AADL 和 Modelica 能够相互转换。Modelica 模型与 AADL 模型的对应关系如表 1 所示。

表 1 Modelica 模型与 AADL 模型的对应关系

Modelica 模型	AADL 模型
Package	Package
Class、model	Device
Nested class	Subcomponent
Connector	Port、portgroup
Equation	Property extension
Variables	Property extension
Connect()	Connection
Function	Parameter、subprogram
Constant	Property extension
Initial value	Property extension
Basic datatype、record、array	Data、datatype

2.2 AADL 扩展的属性集

对于 AADL 中没有的对应转换元素,采用 AADL 的内置数据类型 aadlstring 对属性集进行扩展。扩展的属性集再通过 applies to 子句添加到硬件组件上。扩展的 AADL 属性集合如下所示:

```
property set Modelica_property is
Variables:aadlstring applies to (device);
InitialVariable:aadlstring applies to (device);
ConstantVariables:aadlstring applies to (device);
ConstantValues:aadlstring applies to (device);
Equation:aadlstring applies to (device);
end Modelica_property;
```

3 水箱进出水系统的实例

3.1 使用 AADL 对水箱控制系统建模

本节选择水箱系统进行研究,如图 1 所示,该系统由控制器、水箱和进出水管道组成。其中,控制器相当于信息系统,而水箱及进出水管道相当于物理系统,两者通过端口紧密联系。

控制器通过数据接收端口 wl 接收物理系统发送的水位值 h , 控制器对 h 进行判断: 当 h 下降到 100 时, 控制系统将会通过事件数据端口 cc 给水箱发送开闸放水的控制信号 on, 水箱系统接收到控制信号后, 立即打开阀门注水; 当 h 上升到 150 时, 控制系统将会发送关闭进水的控制信号 off, 整个系统将会把 h 控制在 $[100, 150]$ 的区间范围内。但因为控制系统存在接触不良或者线路老化等问题, 控制系统有 20% 的可能不能够顺利地对水箱进行控制, 出现故障时, 控制系统将会跳转至错误模式。

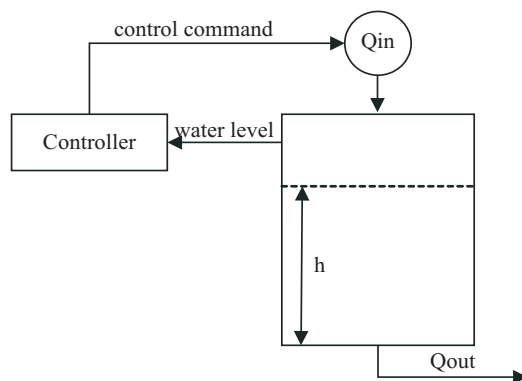


图 1 水箱进出水系统

使用 AADL 对水箱系统中的控制系统进行建模, 如下所示:

```
process controller
features
cc:out event data port;
wl:in data port;
end controller;

process implementation controller.impl
annex Probability_behavior { * *
variables
v:WLCS::ValveStatus
h:WLCS::WaterLevel
states;
inactive:initial state;
low,high,inflow,error: states;
transitions;
inactive - [ wl? (h) & h=100 ] -> 0.8;low{v = on} +
0.2:error;
low - [ cc! (v) ] -> inflow;
inflow-[ [ wl? (h) & h= 150 ] ] -> 0.8;high{v = off}
+ 0.2:error;
high - [ cc! (v) ] -> inactive;
* * };
end controller.impl;
```

3.2 使用 Modelica 对水箱物理部分建模

将水箱及进出水管道看作是 CPS 中的物理系统。本例定义的常量 Q_{in} 、 g 、 π 和 r 分别表示进水值、重力值、圆周率和排水管的半径。水位值 h 的变化采用微分方程进行刻画, 在进水阀门关闭的情况下, 水位值以 $-\sqrt{2\pi r^2 gh}$ 的速度进行下降, 当水位降到一定界限时, 控制系统将会打开进水开关, 这时水位以 $Q_{in} - \sqrt{2\pi r^2 gh}$ 的速度进行上升。当水位达到一定高度后, 控制系统又将会发送控制信息关闭进水开关。

使用 Modelica 对水箱系统中的物理部分进行建模, 如下所示:


```

model watertank
import Modelica.Blocks.Interfaces.BooleanInput;
import Modelica.Blocks.Interfaces.RealOutput;
BooleanInput cc=true;
RealOutput wl;

Real h=150;
parameter Real Qin=0.07;
parameter Real g=9.8;
parameter Real pi=3.14;
parameter Real r=0.0254;
equation
  when cc==true then
    h=Qin-pi*r*r*1.414*g;
  end when;
  when cc==false then
    h=-pi*r*r*1.414*g;
  end when;
end watertank;

```

3.3 Modelica 模型向 AADL 模型的转化

根据对应关系,可以将 Modelica 模型转化为扩展了属性集的 AADL 组件。转化后的模型如下所示:

```

package watertank
public
with Modelica_property;
device watertank
features
  cc :in event data port;
  wl :out data port;
properties
  modelica_property::Variables =>"h";
  modelica_property::InitialVariables =>"150";
  modelica_property::ConstantVariables =>"Qin,g,pi,r";
  modelica_property::ConstantValues =>"0.07,9.8,3.14,0.0254";
  modelica_property::Equation =>
    "whencc == true then
      h = Qin - pi * r * r * 1.414 * g;
    end when;
    whencc == false then
      h = -pi * r * r * 1.414 * g;
    end when;";
end watertank;
end watertank;

```

3.4 系统的组合

将物理系统转化后的 AADL 组件与控制系统的 AADL 模型进行组合,系统内的 AADL 组件连接如图 2 所示。

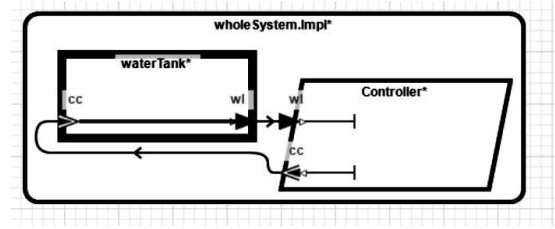


图 2 系统内部的组件连接图

水箱系统对应的体系结构模型如下所示:

```

system wholeSystem
end wholeSystem;

system implementation wholeSystem.Impl
subcomponents
  waterTank:device watertank::watertank.impl;
  Controller:process controller::controller.impl;
connections
  conn1:port waterTank.wl -> Controller.wl;
  conn2:port controller.cc -> waterTank.cc;
flows
  on_end_to_end;end to end flow
  waterTank.on_flow_src -> conn1 -> Controller.on_flow_path -> conn2 -> waterTank.on_flow_sink | latency => 12 ms..12 ms;|;
end wholeSystem.Impl;

```

3.5 系统的流延迟分析

3.5.1 添加系统的流规范

在系统的 AADL 体系结构模型上,可以通过给每个 AADL 组件添加属性,利用现有的 AADL 模型分析工具 Ostate 对系统的体系结构进行流延迟分析、资源分配分析、实时调度分析和安全性分析。

本节对系统的流延迟进行了分析:检测从水箱的水位值达到阈值后到接收到控制系统的控制信号并改变运行状态所要花费的时间。为了验证系统的流延迟,需要为控制系统和物理系统的 AADL 模型分别添加流规范,每个流规范都被赋予了一个延迟值。

物理系统的流规范如下所示,表示物理系统发送数据信号的流延迟为 2 ms ~ 3 ms,物理系统接收控制信号的流延迟为 2 ms ~ 3 ms。

```

on_flow_src;flow source wl | latency => 2 ms .. 3 ms;|;
on_flow_sink;flow sink cc | latency => 2 ms .. 3 ms;|;

```

信息系统的流规范如下所示,表示信息系统对接收到的数据信号进行判断并将控制信号发送给物理系统的流延迟为 3 ms ~ 4 ms。

```

on_flow_path;flow path wl -> cc | latency => 3 ms .. 4 ms;|;

```

水箱控制系统对进水闸进行操作的完整路径,是从源组件 watertank 再到 watertank 的数据信号的流动,在上一小节 wholeSystem 系统中说明了这种流动。

此外,模型 wholeSystem 还为该过程定义了一个 12 ms 的延迟。该数据可以从系统的要求中获取。

3.5.2 系统的流分析

利用流延迟分析工具 Ostate,检测水箱内水位值达到阈值后能否在规定时间内操作水闸。图 3 说明了此次流分析的运行结果。水位值达到阈值之后打开注水开关的操作总延迟时间区间为 7 ms ~ 10 ms,小于系统要求的 12 ms。

Description
▼ ⚠ Warnings (4 items)
⚠ on_end_to_end: Jitter of actual latency total 7.00..10.0ms exceeds expected end to end latency jitter 12.0..12.0ms
⚠ on_end_to_end: Jitter of specified latency total 7.00..10.0ms exceeds expected end to end latency jitter 12.0..12.0ms
⚠ on_end_to_end: Minimum actual latency total 7.00ms less than expected minimum end to end latency 12.0ms (faster actual)
⚠ on_end_to_end: Minimum specified flow latency total 7.00ms less than expected minimum end to end latency 12.0ms (better)
▼ i Infos (1 item)
i Maximum actual latency total 10.0ms is less or equal to expected maximum end to end latency 12.0ms

图 3 顶层的端对端流分析结果

4 Modelica、AADL 上的数据约束

CPS 中的信息系统与物理系统在交互的过程中将会产生大量的数据,为了能够更好地对交互过程及状态变迁中的变量进行形式化的约束,在 AADL - Modelica 建模的基础上,采用 Z 语言对模型中的状态及状态变迁进行形式化的描述。

Z 语言中主要有两种模式:状态模式和操作模式,分别表示状态空间和在状态空间上进行的操作。使用 Z 语言描述状态空间如下所示:

state
$x_1 : S_1; \dots; x_n : S_n$
$\text{Inv}(x_1, \dots, x_n)$

水平线上是对变量的声明, x_1, \dots, x_n 表示状态中所包含的所有变量,变量的取值表示当前所处的状态; S_1, \dots, S_n 表示变量 x_1, \dots, x_n 的取值范围;水平线下是对变量的谓词声明,相当于变量的限制条件且这些限制条件必须为真,同时这些限制条件在相应的操作模式下也必须为真。

Z 语言描述的操作模式如下所示:

Transition
$x_1 : S_1; \dots; x_n : S_n$
$x'_1 : S'_1; \dots; x'_n : S'_n$
$i_1 ? : T_1; \dots; i_m ? : T_m$
$o_1 ! : U_1; \dots; o_k ! : U_k$
$\text{Pre}(i_1 ?, \dots, i_m ?, x_1, \dots, x_n)$
$\text{Inv}(x_1, \dots, x_n)$
$\text{Inv}(x'_1, \dots, x'_n)$
$\text{Op}(i_1 ?, \dots, i_m ?, x_1, \dots, x_n, x'_1, \dots, x'_n, o_1 !, \dots, o_k !)$

$i_1 ?, \dots, i_m ?$ 表示输入变量,描述信息系统或者物

理系统接收到的信号或数据; $o_1 !, \dots, o_k !$ 表示输出变量,描述信息系统或者物理系统到达某一状态发出的信号或数据; $\text{Pre}(i_1 ?, \dots, i_m ?, x_1, \dots, x_n)$ 表示发生状态迁移的前置条件; $\text{Op}(i_1 ?, \dots, i_m ?, x_1, \dots, x_n, x'_1, \dots, x'_n, o_1 !, \dots, o_k !)$ 表示从状态 $\text{Inv}(x_1, \dots, x_n)$ 迁移到状态 $\text{Inv}(x'_1, \dots, x'_n)$ 所做的操作。

CPS 是由物理系统和信息系统组合而成的系统,其本身是一个状态变迁系统。本节将对上述的水箱系统进行形式化的说明。水箱信息系统中的状态 inactive、状态 low 以及从状态 inactive 跳转到状态 low 的 Z 语言描述如下:

状态 inactive:

Inactive_state
$\text{water_level}?: \mathbb{R}$
$\text{control_command}:\text{boolean}$
$\text{state}:\{\text{inactive}, \text{low}, \text{high}, \text{inflow}, \text{error}\}$
$\text{water_level} \geq 100$
$\text{state} = \text{inactive}$
$\text{control_command} = \text{false}$

状态 low:

Low_state
$\text{water_level}?: \mathbb{R}$
$\text{control_command}:\text{boolean}$
$\text{state}:\{\text{inactive}, \text{low}, \text{high}, \text{inflow}, \text{error}\}$
$\text{water_level} = 100$
$\text{control_command} = \text{true}$
$\text{state} = \text{low}$

状态 inactive 到状态 low 的状态迁移:

Low_state
$\text{water_level}?: \mathbb{R}$
$\text{state}, \text{state}': \{\text{inactive}, \text{low}, \text{error}, \text{inflow}, \text{high}\}$
$\text{control_command}, \text{control_command}': \text{Boolean}$
$\text{waterlevel} = 100$
$\text{state} = \text{inactive}$
$\text{state}' = \text{low}$
$\text{control_command} = \text{false}$
$\text{control_command}' = \text{true}$

5 结束语

对 CPS 的建模与验证是一个复杂的问题,文中基于模型的体系结构建模方法,将 CPS 中的信息系统和物理系统分开建模。使用 AADL 对信息系统进行建模,使用 Modelica 对物理系统进行建模。针对信息系统中的概率行为事件,提出了轻量级的拓展语言——概率行为附件;根据 Modelica 与 AADL 的对应关系,对 AADL 拓展相应的属性集,使得 Modelica 模型转化为相应的 AADL 组件成为可能。将 Modelica 模型转换后的 AADL 组件与信息系统的 AADL 模型组合形成一个完整的系统模型。利用模型分析工具 Osate,可以对系统的整体架构、流延迟以及可调用性进行分析。最后再在 AADL-Modelica 建模的基础上,使用 Z 规范对 CPS 中产生的大量数据进行形式化的数据约束。采用 AADL 与 Modelica 相结合的方法对 CPS 进行建模是一个可取的方法,该研究也为系统后续的形式化分析与验证打下了基础。

参考文献:

- [1] GARIBALDO F, REBECCHI E. Cyber-physical system[J]. *AI & Society*, 2018, 33(3): 1-13.
- [2] 董云卫, 王广仁, 张凡, 等. AADL 模型可靠性分析评估工具[J]. *软件学报*, 2011, 22(6): 1252-1266.
- [3] 杨志斌, 皮磊, 胡凯, 等. 复杂嵌入式实时系统体系结构设计与分析语言: AADL[J]. *软件学报*, 2010, 21(5): 899-915.
- [4] BAO Y, CHEN M, ZHU Q, et al. Quantitative performance evaluation of uncertainty-aware hybrid AADL designs using statistical model checking[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017, 36(12): 1989-2002.
- [5] AHMAD E, LARSON B R, BARRETT S C, et al. Hybrid annex: an AADL extension for continuous behavior and cyber-physical interaction modeling[J]. *ACM Sigada Ada Letters*, 2014, 34(3): 29-38.
- [6] 吴义忠, 吴民峰, 陈立平. 基于 Modelica 语言的复杂机械系统统一建模平台研究[J]. *中国机械工程*, 2006, 17(22): 2391-2396.
- [7] FRITZSON P. Introduction to modeling and simulation of technical and physical systems with Modelica[M]. [s. l.]: Wiley-IEEE Press, 2011.
- [8] 黄华, 周凡利. Modelica 语言建模特性研究[J]. *机械与电子*, 2005(8): 62-65.
- [9] JAMSHIDI M. System of systems engineering - new challenges for the 21st century[J]. *IEEE Aerospace and Electronic Systems Magazine*, 2008, 23(5): 4-19.
- [10] ADJUNCT C A S S E. An open systems approach to system of systems engineering[M]//System of systems engineering. [s. l.]: John Wiley & Sons, Inc., 2008.
- [11] LIN Y, SHAO L, ZHU Z, et al. Wireless network cloud: architecture and system requirements[J]. *IBM Journal of Research and Development*, 2010, 54(1): 4:1-4:12.
- [12] 张福高, 曹雪岳. MA 建模的概率混成自动机转换方法研究[J]. *计算机技术与发展*, 2019, 29(2): 19-22.
- [13] FENG S, ZHANG L. A mechanism of transforming architecture analysis and design language into Modelica[J]. *Lecture Notes in Electrical Engineering*, 2014, 277: 1117-1123.
- [14] 李国拯, 高正. 基于带数据约束实时系统的互模拟检测方法[J]. *计算机技术与发展*, 2016, 26(1): 6-9.
- [15] 倪水妹, 曹子宁, 李心磊. 带数据约束实时系统的模型检测[J]. *计算机科学*, 2014, 41(5): 254-262.

(上接第 78 页)

- [4] 孙亮. 基于移动网络信道抗干扰的信息服务多路并发技术[J]. *电子世界*, 2016(17): 90.
- [5] 黄春明. TD-SCDMA 并发业务问题分析与优化[J]. *移动通信*, 2015, 39(9): 53-57.
- [6] VAN DOMELEN D R, LYLES R H. Formulas and web application for designing a biospecimen pooling study to compare group means[J]. *Epidemiology*, 2020(1): 98-102.
- [7] CELEBIOGLU A, UYAR T. Metronidazole/Hydroxypropyl- β -Cyclodextrin inclusion complex nanofibrous webs as fast-dissolving oral drug delivery system[J]. *International Journal of Pharmaceutics*, 2019, 572: 70-78.
- [8] KOSHIMIZU S, NAKAMURA Y, NISHITANI C, et al. TRANSNAP: a web database providing comprehensive information on Japanese pear transcriptome[J]. *Scientific Reports*, 2019, 9(1): 23-38.
- [9] ANITHA M. Sexism and sexual harassment in medicine: unraveling the web[J]. *Journal of General Internal Medicine*, 2019(2): 11-27.
- [10] XIE Guiyan, XIA Mengxuan, MIAO Yaru, et al. FFLtool: a web server for transcription factor and miRNA feed forward loop analysis in human[J]. *Bioinformatics*, 2019(4): 21-32.
- [11] BLAMIRE S J, SELLERS W I. Modelling temperature and humidity effects on web performance: implications for predicting orb-web spider (*Argiope* spp.) foraging under Australian climate change scenarios[J]. *Conservation Physiology*, 2019, 7(1): 69-76.
- [12] 李科伟. 互联网中高并发技术架构实践[J]. *数字通信世界*, 2019(3): 65-66.
- [13] GUL Khan Safi Qamas, 王鹏, 罗森林, 等. 一种高并发网络 Web 应用技术研究[J]. *信息网络安全*, 2017(12): 29-35.
- [14] 侯艳君. 流媒体集群并发技术在视频点播系统中的应用[J]. *传播力研究*, 2018, 2(21): 249.
- [15] 刘文建, 邓思胜, 丁华祥, 等. 基于 CORS 位置云服务的高并发技术研究[J]. *全球定位系统*, 2018, 43(4): 67-72.