一种面向信息物理融合系统安全攸关场景的测试用例自动生成方法

姜 鹏 陈 鑫 李宣东
(南京大学计算机软件新技术国家重点实验室 南京 210046)
(南京大学计算机科学与技术系 南京 210046)

摘 要 对信息物理融合系统中的安全攸关场景进行有效的测试是提高系统安全性的重要手段。如何对安全攸关场景进行建模以完整准确地刻画系统行为,如何有效地生成测试用例以提高测试覆盖度、降低测试成本,是面向安全场景测试技术需要解决的核心技术问题。现有的场景建模与测试用例生成技术缺少对信息物理融合系统重要特性的描述和处理,其生成的测试用例不能满足系统安全攸关场景的测试需求。围绕信息物理融合系统的安全攸关场景建模以及测试用例自动生成方法展开研究,为UML活动图扩充了外部事件驱动机制和时间特性描述机制,以满足对安全攸关场景建模的需要;并研究了基于场景模型自动生成测试用例的方法。

关键词 信息物理融合系统,安全攸关场景,测试用例自动生成
中图法分类号 TP311.56 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.11.024

Method to Automatic Testcase Generation toward Safety Critical Scenarios of Cyber-physical Systems

JIANG Peng CHEN Xin LI Xuan-dong
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210046, China)
(Department of Computer Science and Technology, Nanjing University, Nanjing 210046, China)

Abstract Effectively testing safety-critical scenarios is an important means to improve security of cyber-physical systems. How to model safety critical scenarios so that the behavior of systems can be completely and precisely described, and how to effectively generate test cases so as to enhance the test coverage and reduce testing cost, are two key technical problems that must be solved in safety-critical scenarios' testing. Existing scenario modeling and test case generation techniques lack the support to describing and treating the important features of cyber-physical systems, thus they can't generate test cases satisfying the testing requirements of the safety critical scenarios. We studied modeling and test case generation methods to safety critical scenarios in cyber-physical systems. A method that satisfies the requirement for modeling safety-critical scenarios by extending UML activity diagram with external event driven mechanism and timeliness characterizing mechanism was proposed. Then based on the scenario models, we studied a method to automatically generate test cases.

Keywords Cyber-physical systems, Safety-critical scenarios, Automatic testcase generation

1 引言

信息物理融合系统(Cyber Physical System,CPS)是指物理系统与计算机系统、通信系统深度交联,通过网络化计算扩展和增强物理系统能力的新一代系统[1]。典型的CPS系统包括高可信医疗系统、交通控制与交通安全管理系统、航空航天控制系统以及国家基础设施中的各种核心控制系统等。这些CPS系统很多时候也是安全攸关系统(Safety-Critical System),系统的故障或失效会导致重大的生命财产损失。

国际标准组织为安全攸关电子电气系统整个生命周期活动制订的通用标准(IEC61508)[6]中,将安全定义为"不存在不可接受的风险"。在工程实践中,通常要求对系统中的所有安全攸关场景进行完整的测试,充分覆盖场景中系统各种可能的执行路径,以确认系统在各种情况下都能够满足系统安全规范的要求。因此,研究安全攸关系统的测试方法,尤其是对系统中的安全攸关场景进行有效测试的方法对于保证安全攸关系统的安全性具有至关重要的意义。

对安全攸关场景进行测试面临的主要挑战是如何有效地生成测试用例对安全攸关场景进行充分的覆盖。同时,由于这类安全攸关系统规模大、复杂度高,也需要研究如何自动地生成测试用例以降低测试的开销提高测试的效率。生成有效测试用例的前提是对安全攸关场景规约进行准确的描述。简单依靠使用自然语言描述的规约在给定时间内达到可接受的测试覆盖率是非常困难的[12]。模型是对复杂系统进行描述

到稿日期:2013-09-16 返修日期:2013-11-16
姜 鹏(1987-),硕士生,主要研究领域为软件工程、软件测试、形式化方法,E-mail:jiangpeng.nju@gmail.com;陈 鑫(1975-),男,博士,讲师,主要研究领域为软件工程、形式化方法;李宣东(1962-),男,博士,教授,博士生导师,主要研究领域为软件工程、形式化方法、模型检验。

和分析的有效工具，如何选取和设计恰当的模型，对 CPS 系统中各种安全攸关场景规约进行完整准确的描述，是首先需要解决的问题。在构造各种安全攸关场景的模型后，如何有效地生成测试用例，以对场景中各种可能的执行进行充分的覆盖，特别是为降低测试的开销，能够自动地生成测试用例，也是需要关注的关键问题。

基于以上分析，本文提出了一种为 CPS 系统中安全攸关场景规约进行建模并基于此模型自动生成测试用例的方法。安全攸关场景建模对 UML 活动图模型进行了扩充，添加了外部事件驱动机制，以及时刻、时段(持续时间)和时间约束的描述机制，以满足安全攸关场景规约建模的需要。本文研究了基于此场景模型自动生成测试用例的方法。其基本思想是首先遍历模型，找出场景中各种可能的执行路径，然后从场景模型中提取路径上的事件和时间，构造出约束系统，并对其求解得到相应的控制序列。在这一序列的控制下，系统将按照场景规约中指定的路径执行。这一序列就是一个有效的测试用例。

本文第 2 节介绍了安全攸关场景的建模方法，并以一个列车控制系统中的安全攸关场景为例进行说明；第 3 节给出了针对扩展后模型自动化生成测试用例的方法；第 4 节是本文工作与现有相关工作的比较；最后对本文进行总结，并给出进一步工作的介绍。

2 安全攸关场景建模

模型是对复杂系统进行描述和分析的有效工具，如何选取和设计恰当的模型，对 CPS 系统中各安全攸关场景规约进行完整、准确的描述，是首先要解决的问题。现有的针对 CPS 系统的研究表明：CPS 系统具有高并发性；时间性质在系统中具有绝对重要的地位[1]。支持系统中异构子系统、外部事件引起中断、时间性质等的刻画，是对 CPS 系统中安全攸关的场景进行建模的基本技术需求。

UML 作为事实的工业标准，被工业界广泛使用。其中，UML 活动图模型常被用于对系统中各个子系统的工作流和相互间的交互进行建模。

我们的建模语言选择 UML 活动图模型进行扩展，以满足 CPS 系统特点带来的建模需求。我们的扩展主要从两方面入手：1)如何表示活动过程中，由外部事件引起中断的情况；2)如何对时刻、持续时间、时间约束这 3 个时间相关特性进行刻画。对于第一个方面，UML 活动图模型通常假定整个过程中没有外部事件引发中断的情况发生。从测试的角度出发，我们需要刻画出模块之间的事件信息的传递。在外部事件发生时，系统会发生相应的活动间的转换，我们在活动图中相应的转换上标注出事件内容，其格式为“〈event：事件内容描述〉”。对于第二个方面，有关时间特性的刻画，尽管 UML 可以应用于大多数面向对象、基于组件的建模方法，但是该语言对于时间而言缺少一些可以量化的表达方式。我们采用添加时间标识的方式引入时间概念，并以这些标识间的约束来描述系统需要满足的时间特性。首先，在选取的关键活动的入边或者出边处，以“@t_i”的格式标注，表示该活动的开始或结束在 t_i 这一时刻；而一个活动的持续时间，则可以用形如“t_j-t_i”的格式表示，其中 t_i 与 t_j 分别表示该活动的开始和结束时刻；对于时间约束，我们在模型中提供区域集中放置，可用于表示整个场景全局中需要满足的一些约束。

为了更好地对我们的模型进行分析，我们采用类 Petri 网的语义对扩展之后的模型进行如下定义。

定义 1(一个带时间标记的活动图，Timed Activity Diagram)　D 是一个多元组 $D=(A,Tr,T,G,E,F,a_I,a_F,TC)$，其中

- $A=\{a_1,a_2,\cdots,a_i\}$是活动的一个有穷集合；
- $Tr=\{tr_1,tr_2,\cdots,tr_j\}$是转换的一个有穷集合；
- $T=\{t_1,t_2,\cdots,t_k\}$是时间点的一个有穷集合，t_i 用于表示活动 a_j 的起始或者结束的时刻；
- $G=\{g_1,g_2,\cdots,g_j\}$是卫士条件的一个有穷集合，且在对应的转换上(无卫士默认为 true)；
- $E=\{e_1,e_2,\cdots,e_m\}$是事件的一个有穷集合，且 e_i 位于跨泳道的转换 tr_j 上；
- $F\subset(A\times Tr)\cup(Tr\times A)$是一个有穷的流关系；
- $a_I\in A$ 是初始节点；$a_F\in A$ 是终止节点。仅有一个转换 tr 满足$(a_I,tr)\in F$；对任意转换 $tr_i\in Tr$，满足$(tr_i,a_I)\notin F\wedge(a_F,tr_i)\notin F$。
- $TC=\{tc_1,tc_2,\cdots,tc_l\}$是关于时间的不等式的一个有穷集合。

为更形象地说明带时间标识的活动图，下面我们用一个例子进行说明。我们准备描述的是列车进站时与屏蔽门控制相关的场景。这是列车控制系统中的一个安全攸关场景，其描述的业务流程是列车进站后，正常情况下先打开列车门，接着打开屏蔽门，然后上下客，接着关闭屏蔽门，最后关闭列车门；但如果在列车开门后，屏蔽门在 10s 内未能成功打开，则会触发警报，需要进行应急处理。

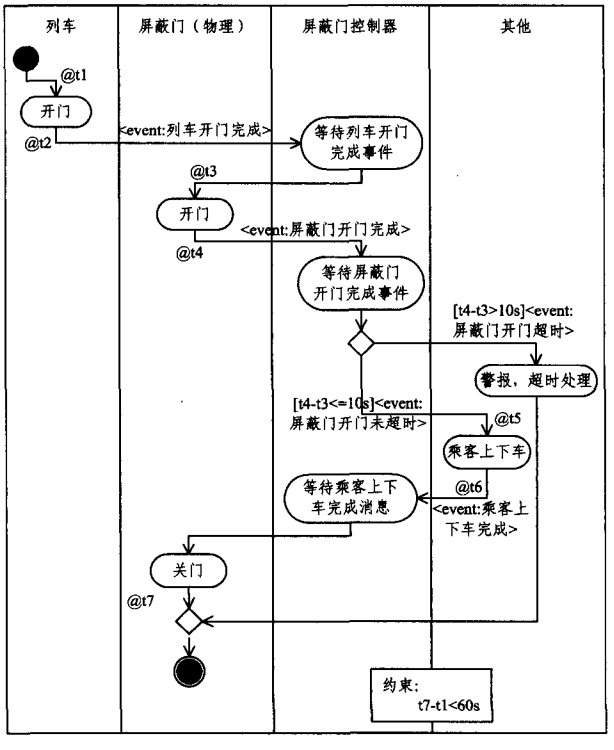

图 1　列控系统中列车进站场景模型

对本场景建立出的模型如图 1 所示。直观地看，该模型与 UML 活动图模型有很大相似之处：含 4 个泳道，分别表示场景中的 4 个主体，分别代表 3 个子系统(列车、屏蔽门物理部分、屏蔽门控制器)和外部环境(其他)；有初始节点和终止

节点，分别表示场景的开始和结束；用圆角方框表示活动，有向箭头表示转换；用菱形框表示判断节点。同时该模型又体现了我们所做的扩展：在事件触发引起的转换上，有对事件内容的表示，如列车开门活动完成后，发出“列车开门完成”信号至屏蔽门控制器。在节点之前或之后，有形如“@t1”的标识，这是我们所做的时间标识；模型中右下方约束方框用于集中放置全局时间约束，图中 t7－t1＜60s 表示整个场景时间跨度不超过 60s。

3 测试用例自动生成

带时间标识的活动图中从起始节点开始到终止节点的任意一条由活动与转换交替组成的执行序列，我们称之为一条路径。需要注意的是，当路径中出现循环的执行序列片段时，其执行序列可以衍生出无数个，为保证测试用例相对完整的同时降低复杂度，我们引入了简单路径的概念，就是在出现循环片段时，对该片段做只执行一次处理。

给出路径和简单路径的定义如下：

定义 2 在一个带时间标记的活动图 $D=(A,Tr,T,G,E,F,a_I,a_F,TC)$ 中，一个转换 $tr\in Tr$ 可能从状态 μ 触发，当且仅当满足如下两个条件：

- $tr\in enabled(\mu)$；
- $(\mu-\bullet tr)\cap tr\bullet=\emptyset$。

转换 tr 从状态 μ 触发后得到的新状态为 $\mu'=(\mu-\bullet tr)\cup tr\bullet$，记为 $\mu'=fire(\mu,tr)$。

D 中的一个路径定义为：

$$\sigma=\mu_0\xrightarrow{\tau_0}\mu_1\xrightarrow{\tau_1}\cdots\cdots\xrightarrow{\tau_{n-1}}\mu_{n-1}$$

其中，$\mu_0=\{a_I\}$，$\mu_n=\{a_F\}$，且对于任意的 $i(1\leqslant i\leqslant n)$，有 $\mu_i=fire(\mu_{i-1},\tau_{i-1})$。如果 σ 中不存在重复的执行，即满足条件：对于任意的 $\tau_i,\tau_j(0\leqslant i<j<n)$ 中的任意转换 $tr\in\tau_i$ 和 $tr'\in\tau_j$，有 $\bullet tr\cap\bullet tr'=\emptyset$，则 σ 是一个简单路径。

通过以上定义，我们可以将安全攸关场景的场景覆盖问题转化为模型上的路径覆盖问题。通过覆盖模型中尽量多的可执行路径即可完成场景的覆盖，提高系统的安全性。

我们的思路主要分为两个步骤：第一步，依据对安全攸关场景建立的模型，分析出其中的简单路径；第二步，通过分析各路径，给出控制信息，以及预期的观察结果，即可得到测试用例。对于这两个步骤，以下分别给出算法及解释。

对于给定的带时间标记的活动图模型，我们使用一种深度优先(DFS)算法来计算出相对充分的路径。由于循环的存在，会导致路径中的某些活动多次出现。特别对于物理信息融合系统而言，系统的高度复杂性很可能因分支和循环而带来路径爆炸问题，因此我们以简单路径覆盖[11]作为路径覆盖准则。

算法 1：对判定和并发进行组合，最多循环一次，深度有限遍历活动图，获取活动图上线程执行的所有简单路径就可以生成路径集合。其总体思想是对条件分支和并发分支进行完全组合，至多循环一次，采用带回溯的深度优先方法遍历活动。在算法中，*currentStateSet* 表示某一时刻 *diagram* 中活动集合 A 的一个子集；*enabledTrans* 是指从当前活动集合 *currentStateSet* 中可以发生的转换。

算法 1 简单路径集合生成算法

输入：待时间标记的活动图 diagram

输出：简单路径集合 returnPaths

```
1. DECLARE tranStack, stateStack, enabledStack
2. DECLARE currentStackSet, newStateSet, selectedTrans
3. CurrentStactSet=diagram. getStartState();
4. DO BEGIN
5.   stateStack. push(CurrentStateSet);
6.   enabledStack=transitions can start from currentStackSet;
7.   IF endState is in currentStateSet
8.     path=new Path;
9.     path. addState(stateStack);
10.    path. addTran(tranStack);
11.    returnPaths. append(path);
12.    ROLLBACK:
13.    IF tranStack is empty
14.      RETURN returnPaths;
15.    ELSE
16.      stateStack. pop();
17.      currentStateSet=stateStack. top();
18.      tranStack. pop();
19.      enabledTrans=enabledStack. top();
20.      enabledStack. pop();
21.    ENDIF
22.  ELSE
23.    DO BEGIN
24.      selectedTrans = select a tran, then remove it;
25.      IF(selectedTrans is empty)
26.        GOTO ROLLBACK;
27.      ENDIF
28.      newStateSet=CurrentStateSet－SelectTrans. PreStateSet
                    +selectTrans. PostStateSet;
29.      IF(newStateSet has appeared in stateStack twice)
30.        selectAnother=true;
31.      ELSE
32.        selectAnother=false;
33.        tranStack. push(selectedTranIDs);
34.        enabledStack. push(enabledTranIDs);
35.        currentStateSet=newStateSet;
36.      ENDIF
37.    END WHILE selectAnother==true;
38.  ENDIF
39. END WHILE true;
```

算法 1 将安全场景测试中的场景覆盖问题转化为路径覆盖问题，并给出应用遍历的思想在带时间标记的活动图上找出所需的路径的相关算法。

获得简单路径集合之后，针对集合中每一条简单路径，我们分别生成对应的测试用例。首先，明确该场景中被测系统(System Under Test，SUT)的边界，测试关注的某一子系统或多个系统组成的集合位于边界之内，其他子系统或人员等外部环境位于边界之外。在生成的测试用例中，应当包含 3 个部分：SUT 边界之外在特定时间、满足特定约束情况下传递给 SUT 的控制信息；在特定时间点对 SUT 的观察结果；关于时间上所需满足的特定的约束。本文给出对测试用例的定义如下：

定义 3 一个测试用例 *case* 是一个多元组 $case=(input,timeCons,observed)$，其中

• $input=(s_m,e_n,t_1)\rightarrow(s_{m+1},e_{n+1},t_2)\rightarrow\cdots\rightarrow(s_{m+k},e_{n+k},t_k)$是一个输入序列，$(s_m,e_n,t_1)$表示在 t_1 时刻，消息发送者 s_m 向被测系统发送消息 e_n；

• $timeCons=\{tc_1,tc_2,\cdots,tc_o\}$是关于全局的时间约束的集合，其中 $tc_1,tc_2,\cdots,tc_o$ 为时间约束。

• $observed=(r_m,e_n,t_1)\rightarrow(r_{m+1},r_{n+1},t_2)\rightarrow\cdots\rightarrow(r_{m+k},e_{n+k},t_k)$是一个观测到的结果序列，$(r_m,e_n,t_1)$表示在 t_1 时刻，可以观察到被测系统向消息接受者 r_m 发送了消息 e_n。

上述定义中观测结果序列表示生成的测试用例在给定的满足约束的输入之下得到的预期结果。

算法 2 以简单路径为输入，从起始节点开始，对简单路径上位于被测系统边界上的转换进行判定，如该转换是由被测系统外指向被测系统，则将其添加到输入序列；如该转换由被测系统指向被测系统外，则将其添加至观测结果序列。同时，搜集沿该简单路径所需满足的约束，其包含的类型有以下 3 种：该简单路径上的卫士条件；全局时间约束中与该路径相关的部分；该路径上活动先后带来的时间约束。通过对这些约束进行求解，可得到一族测试用例。

算法 2 测试用例集生成算法

输入：给定简单路径 path，形如 $a_0tr_0a_1tr_1a_2tr_2\cdots a_{n-1}tr_{n-1}a_n$；全局时间约束集合 TC，其中每一个时间约束是一个关于时间点的线性不等式，形如 $t_m-t_n\leqslant d$，其中 t_m，t_n 为时间点，d 是一个表示时间长度的常数。

输出：对应的一族测试用例集合 testCases，其中每个用例 testCase 形式如定义 3 中所示。

```
1. FOREACH(a_i in path)
2.   IF(a_i.swimlane∉SUT && a_i.next.swimlane∈SUT)
3.     input.queueIn(a_i.swimlane,event on tr_i,t_i);
4.     trans.add(tr_i);
5.   ENDIF
6.   IF(a_i.swimlane∈SUT && a_i.next.swimlane∉SUT)
7.     observed.queueIn(a_i.swimlane,event on tr_i,t_i);//此处 t_j 表示观测到该事件的时刻
8.     trans.add(tr_i);
9.   ENDIF
10. ENDFOREACH
11. FOREACH(tr_i in trans)//guards 产生的约束
12.   testCase.constraints.add(tr_i.guards);
13. ENDFOREACH
14. FOREACH(tc in TC)//TC 中的全局时间约束
15.   testCase.constraints.add(tc);
16. ENDFOREACH
17. FOREACH(t_i in path.timepoints)//该路径上事件发生先后的时间约束
18.   testCase.constraints.add(generateConstraint(t_i≤t_i+1));
19.   ENDFOREACH
20.   RETURN testCase;
```

为更直观地解释测试用例和算法 2，我们仍以上文提到的列控系统中的列车进站场景举例说明。本场景中有两条简单路径，分别用于表示屏蔽门在 10s 内是否正常打开的两种情形。以屏蔽门在 10s 内正常打开的情形为例，分析该情形下的测试用例及其生成过程。表 1 所列的输入输出消息序列表示沿指定路径运行，与被测系统（屏蔽门控制器）之间进行的所有交互。

表 1 输入输出消息序列

事件类型	事件 sender	事件 receiver	事件内容	时间
Input	列车	屏蔽门控制器	〈event:列车开门完成〉	t2
Output	屏蔽门控制器	屏蔽门（物理）	〈event:屏蔽门允许开门〉	t3
Input	屏蔽门（物理）	屏蔽门控制器	〈event:屏蔽门开门完成〉	t4
Output	屏蔽门控制器	其他	〈event:屏蔽门开门未超时〉	t5
Input	其他	屏蔽门控制器	〈event:上下客完成〉	t6
Output	屏蔽门控制器	屏蔽门（物理）	〈event:屏蔽门允许关门〉	—

根据算法 2，由简单路径中初始节点出发，向后依次对活动间转换进行判定，将被测系统边界上的输入类型事件按照先后顺序存放为输入序列，将输出类型事件按照先后顺序存放为观测序列，同时搜集该简单路径上 3 种类型的约束：1）卫士条件，该例子中为 $t4-t3\leqslant 10s$；2）全局时间约束，该例子中为 $t7-t1\leqslant 60s$；3）事件发生先后顺序带来的时间约束，包括 $\{t1<t2,t2<t3,t3<t4,t4<t5,t5<t6,t6<t7\}$。然后对 3 种约束进行合并并求解。最后，以输入序列为输入，以观测节点为预期结果，并且满足约束的一族结果即为我们所生成的测试用例。

4 相关工作

随着测试技术的发展，通过构造系统的行为模型、借助工具进行场景测试方面的工作已有一些相关工作。场景测试方面的工作通过使用用户定义的输入来引导系统的运行，结合基于模型的测试策略，限定模型在一个相关的子集中执行，完成其在各自领域的测试工作。但据我们所知，将场景测试技术应用于安全攸关的 CPS 系统中的研究工作还比较匮乏。

Frédéric Dadeau 等[10]实现了一个名为“jSynoPSys”的工具，该工具使用 B machine 建立的模型描述执行场景，以正则表达式语法描述了子系统的操作，以及场景展开时系统所需达到的中间状态。然后基于 BZ-Testing-Tools 引擎对模型进行有界测试生成。该项工作应用了将用例的自动化生成工作转化为约束求解问题这一思想，使用 B machine 作为建模机制，难以被工业界广泛使用。

Kalou Cabrera Castillos 等[7]提出了一种基于场景的测试方法，该方法针对 UML/OCL 行为模型，使用正则表达式语法描述场景，描述了组成场景和约束模型执行的操作序列，并使用商用测试工具 Test Designer[8]实现该方法。同样，该方法相对于采用图形化的 UML 或类 UML 技术进行建模的工作而言，难于推广。

Eda Marchetti 等[9]将场景测试分别应用于医疗和人力资源方面两个实例研究中。该工作使用 UML Sequence Diagram 进行建模，通过定义消息序列、分析可能的子用例、定义设置分类、确定分支选择、确定选择间的约束关系、生成测试流程的算法，生成测试用例。该工作针对两个实例的集成测试方面，而本文工作提出的方法更具有通用性，根据模型描述场景的大小，可用于单元测试、集成测试、系统测试等各种情形。

Percy Pari Salas 等[13]提出一种软件测试的自动化方法

（下转第 161 页）

Joint Conference on INC, IMS and IDC, 2009(NCM'09). 2009: 214-219

[10] SEI. CMMI for Development, Version 1.2-Improving Processes for Better Products[S]. SEI, CMU, 2006

[11] Sujin C, Dae-Kyoo K, Sooyong P. ReMo: A recommendation model for software process improvement[C]//2012 International Conference on Software and System Process (ICSSP). IEEE Press, Zurich, 2012: 135-139

(上接第 127 页)

并将其应用于异步系统(asynchronous systems)中。他们的工作使用事件结构作为形式化描述方式,该工作将控制事件和可观察事件区分开来,通过提取出的仅含有控制事件的序列来生成测试用例。该工作通过一些扩展机制,在现有测试工具基础上完成自己的功能并将其应用于真实系统中。该工作对于控制事件与可观察事件的区分与本文有相似之处,但其应用领域为纯粹的软件行业,显然不能满足高并发、高安全性要求的 CPS 系统。

在安全攸关的 CPS 系统中应用基于场景的测试技术需要充分考虑 CPS 系统与传统系统,如软件、嵌入式系统等之间的区别,突出 CPS 系统的特征以及面临的新的挑战。

CPS 系统研究的先驱 Edward A. Lee[1]将 CPS 系统与当前的嵌入式系统作了比较,他强调了 CPS 系统具有与生俱来的高并发性,时间性质在 CPS 系统中具有绝对重要的地位,不可忽略。Frank Mueller[2]指出安全问题、时间相关分析和软错误保护是 CPS 系统研究面临的 3 大挑战。Lui Sha 等[4]将实时系统的抽象建模,CPS 系统的健壮性、防危性和安全性,系统服务质量组合,将系统工程化方法研究与系统可信作为 CPS 研究的主要挑战。Stankovic 等[5]认为相关的验证与检验方法仍是 CPS 系统研究的重点,其中形式化的建模与设计方法是重要的研究方向。

本文工作参考上述文献所述的 CPS 系统的特点,在模型构造和用例生成阶段充分考虑系统的时间特性,应用路径覆盖的思想达到对场景的覆盖,保证相对充分的测试,从而提高 CPS 的安全性。

结束语 本文立足于现有的 CPS 中安全攸关场景建模与测试用例生成技术上存在的不足,提出了一种基于 UML 活动图扩展的建模语言,并提供对此类模型进行测试用例自动化生成的算法。

本文工作的意义如下:

(1)在 UML 活动图的基础上,根据安全攸关场景建模的新需求,提供新的建模支持,尤其是在时间特征方面的支持。

(2)采用对安全攸关场景模型生成测试用例的过程,来帮助系统确定受控设备的范围,系统内部、系统与外部环境相互作用可能存在风险的点。进一步提高系统的安全性。

(3)测试用例的生成过程实现了自动化,极大减少了手工创建测试用例容易造成的遗漏,同时降低了人力成本。

参 考 文 献

[1] Lee E A. Cyber Physical Systems: Design Challenges[C]//Proc. 11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. 2008: 363-369

[2] Mueller F. Challenges for Cyber-Physical Systems: Security, Timing Analysis and Soft Error Protection[C]//National Workshop on High Confidence Software Platforms for Cyber-Physical Systems(HCSP-CPS). Nov 2006

[4] Lui Sha, Gopalakrishnan S, Liu Xue, et al. Cyber-Physical Systems: A New Frontier[C]//Proc. 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing. 2008: 1-9

[5] Lee S J A, Mok I, Rajkumar A, et al. Opportunities and Obligations for Physical Computing Systems[J]. IEEE Computer, 2005, 38(11): 23-31

[6] IEC 61508 series[OL]. http://www.iec.ch/functionalsafety

[7] Castillos, Cabrera K, Botella J. Scenario based test generation using test designer[C]//2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2011

[8] Utting M, Legeard B. Practical Model-Based Testing-A tools approach[J]. Elsevier Science, 2006, 550

[9] Marchetti, Eda, Schilders L, et al. Scenario-based testing applied in two real contexts: Healthcare and Employability[C]//2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2011

[10] Dadeau, Frédéric, Tissot R. jSynoPSys-a scenario-based testing tool based on the symbolic animation of B machines[J]. Electronic Notes in Theoretical Computer Science, 2009, 253(2): 117-132

[11] Wang Lin-zhang, et al. Generating test cases from UML activity diagram based on gray-box method[C]//Software Engineering Conference, 2004. 11th Asia-Pacific. IEEE, 2004

[12] Ryser J, Berner S, Glinz M. On the State of the Art in Requirements-based Validation and Test of Software[M]. Universität Zürich, Institute für Informatik, 1998

[13] Salas, Pari P, Krishnan P. Automated software testing of asynchronous systems[J]. Electronic notes in theoretical computer science, 2009, 253(2): 3-19