

基于智能家居的一致性模型融合技术研究

包晓安¹, 林 辉¹, 周建平², 桂 宁¹, 孙献策¹, 张 娜¹

(1. 浙江理工大学信息学院, 杭州 310018; 2. 海盐盛迪电子科技有限公司, 浙江嘉兴 314300)

摘 要: 在智能家居的自适应软件设计中,往往采用传统的模块构建技术。但由于其适应逻辑存在着低复用性、高复杂度等问题,很难验证模块组合后的正确性及有效性。为了满足开发过程中用户的增量性需求,针对适应逻辑的组合难题,提出把部分行为模型的形式化方法引入到适应行为的描述中。通过三值逻辑 KMTS 模型描述语言,提出一致性模型的判断方法以及融合算法,以提供明确的适应模型在线融合支持。最后通过智能家居中的一个模型实例来分析并验证融合后的适应逻辑。

关键词: 智能家居; 模型融合; 适应逻辑; KMTS; 一致性模型

中图分类号: TP311.5 **文献标志码:** A

0 引 言

随着智能家居不断地进入到人们的生活中,智能设备的自适应技术也越来越受到关注。在当前的自适应软件设计中,主要是采用专用的适应模块构建方式。这种方式是由多个模块共同实现用户需求,当用户需求或是环境发生变化后,以模块替换甚至模块的重新组合来完成整体功能的设计。然而,由于原有软件模块与需求模块的适应逻辑,往往因设计目标或者描述语言的不同,使得适应逻辑的组合设计显得非常困难。一些知名研究者在适应逻辑组合上已经有深入的研究,如 Maurel 等^[1]提出了基于投机的融合策略,该策略的核心是假设各个适应逻辑不会产生冲突,在这种假设下各个适应方案进行直接地叠加。Sicard 等^[2]提出了一种支持软件自修复的自适应软件框架,在该框架中,两种不同的适应目标是通过设计一种特别的适应算法来进行融合,以消除可能存在的冲突。但是,这个专用的算法在两个不同的适应逻辑中间构成了强耦合机制,使得适应模块在不同上下文中很难被重用。接钧靖等^[3]实现了一种基于质量运行时动态评估的自主构

件的自适应机制,通过对自主构件的性能监控,以评估自适应策略的优劣并进行调整,但没有涉及如何构建适应逻辑过程。这些关于多模型融合的自适应软件开发模式,大致分为三大类:简单融合、专用算法融合方案和基于收益函数的融合方案。由于这些方案通常是采用经验性的解决方式,因此很难验证组合后模块的正确性及有效性。

为了更好地解决模块组合后适应逻辑的验证性问题,本文借助了部分行为模型(partial behaviour model)^[4]的数学工具,提出将部分行为模型的形式化方法引入到自适应软件适应行为的描述中,使得适应逻辑组合的过程形式化。通过精确地描述建模中未知以及可重配置的信息,从而使得模型融合的实现及其验证成为可能。把不同的自适应软件模块看成是不同的适应模型,以部分行为模型的不断融合来满足增量性的需求。在形式化描述的基础上,设计部分适应模型之间的一致性判定方法,以及适应逻辑融合算法,为一致性模型的融合提供可靠支持。通过构建相应的模型融合实例,抽象的模型分析以及实际的开发应用,实现了融合结果的正确性及有效性验证。

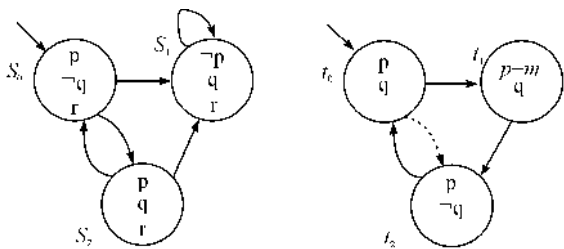
收稿日期: 2014-04-28

基金项目: 国家自然科学基金资助项目(61202050, 61379036); 浙江省自然科学基金资助项目(Y13F020175); 浙江理工大学 521 人才培养计划资助项目; 浙江省钱江人才计划资助项目(2013R10015); 浙江省新苗计划项目(2013R406070)

作者简介: 包晓安(1973-),男,浙江东阳人,教授,主要从事自适应软件及软件测试等方面的研究。

1 形式化描述语言的选择和分析

为了支持适应模型增量性的融合以及解决融合过程中所产生的冲突,适应模型必须支持对当前未知信息的明确建模。传统的建模语言 kripke structure,如图 1(a)所示,使用 kripke 架构定义了所有的状态和转换,并不支持可能的状态转换或求精。为了更好地支持增量式的软件开发模式,一些研究者提出部分行为模型的概念,把每一个需要进行融合的模型都作为系统整体的一个部分,通过不断融合多个部分模型来严格描述整体行为,并判断其特性。它对“分而治之”^[5]的开发模式有着良好的支持,并逐渐成为行为模型形式化研究的热点。目前出现了许多基于部分行为模型的形式化语言和架构,如部分标记变迁系统(partial labelled transition systems, PLTSs)^[6], multi-valued state machines^[7]等。如图 1(b)使用了基于三值逻辑的 KMTS (kripke modal transition system),增加了可能的状态转换(虚线表示)和可能的状态($p=m$ 意味着 p 的状态未知)支持,可以有效地提供对未知和可重配信息的描述,并且一些非确定的描述可以在行为模型设计中逐步求精或者多部分模型融合过程中逐步去除,从而为适应行为模型的融合描述提供了良好支持。



(a) 基于 kripke 架构的形式化描述 (b) 基于 KMTS 的形式化描述

图 1 基于状态机的形式化描述

KMTS 是一个元组 $(S, s_0, R^{\text{must}}, R^{\text{may}}, L, AP)$, 定义了所有的状态和转换,包括可能的状态和转换。其中 S 是状态集, s_0 是初始状态, $R^{\text{must}} \subseteq S \times S$ 表示状态之间的确定性关系, $R^{\text{may}} \subseteq S \times S$ 表示状态之间的可能性关系。而 $L \rightarrow 3^{AP}$ 表示一个三值标记功能,以 $t(\text{true})$ 、 $f(\text{false})$ 、 $m(\text{maybe})$ 三值标记状态中的每个事件是否发生。 AP 则表示一个模型中事件的集合,在模型融合时,以 AP_u 表示融合模型的事件总集。

用 KMTS 形式化描述语言对图 1(b)进行如下定义:

a) $S = \{t_0, t_1, t_2\}$, 其中 t_0 是初始状态;

b) $R^{\text{must}} = \{(t_0, t_1), (t_1, t_2), (t_2, t_0)\}$;

c) $R^{\text{may}} = \{(t_0, t_1), (t_1, t_2), (t_2, t_0), (t_0, t_2)\}$;

d) $L(t_0) = \{(p, t), (q, t)\}$, $L(t_1) = \{(p, m), (q, t)\}$, $L(t_2) = \{(p, t), (q, f)\}$;

e) $AP = \{p, q\}$

KMTS 描述时,令 $R^{\text{must}} \subseteq R^{\text{may}}$, 转移关系 R^{must} 用实线箭头表示,而 $R^{\text{may}} \setminus R^{\text{must}}$ 用虚线箭头表示。

在自适应软件开发中,用户需求以及自适应环境往往会随时发生变化,模型之间的关系很难被预先判断,从而影响模型的融合策略以及融合效果。本文所研究的模型融合过程,是针对一致性模型之间,并且基于不变的环境以及需求,以保证模型融合过程中的稳定性。对于需要进行融合的两个适应模型,首先需要确定模型间的关系。由于设计的目标和设计角度不同,不同模型可能采用不同的描述语言,很难判断模型间关系^[8]。本文统一所有模型的形式化描述语言为 KMTS,以基于状态机的运行方式综合判断模型间是否具有相关性,而在相关模型中,根据模型间有无冲突又可区分为一致性与非一致性。其次,针对模型间关系,从而采取不同的融合策略^[9]来比较并且融合这些相关模型。在一致性模型融合时,关键在于确定相同性质的状态以及转移关系,为融合行为提供切入点。最后我们需要对融合结果进行分析,以确保模型融合是否实现了融合目标。

2 适应行为模型融合

模型融合是模型之间的最小共同求精 (least common refinement, LCR)^[10]的过程。在模型融合之前,需要对各个模型采用 KMTS 描述。若模型之间的 KMTS 语言描述无关,则可认为模型融合时不产生干扰,产生的 LCR 唯一,即作为模型融合结果。

若模型之间的 KMTS 语言描述相关时,在模型融合过程中,将产生多个无法相互比较的极小共同求精。为此,本文进一步研究了获得最小共同求精的方法。

2.1 相关模型的统一描述

对于需要进行融合的适应模型,都是内在不完整的模型。这些模型都只描述整个系统某些极少的事件。例如,设一个系统中有两个模型 K_1 、 K_2 , KMTS 形式化描述如下:

$K_1: \{S_1 = \{s_0, s_1\}; R^{\text{must}} = \{(s_0, s_1)\}\}$;

$R^{\text{may}} = \{(s_0, s_1)\}; L(s_0) = \{(p, t)\}, L(s_1) = \{(p, t); AP = \{p\}\}$ 。

$K_2: \{S_2 = \{u_0, u_1, u_2\}; R^{\text{must}} = \{(u_0, u_1), (u_1, u_2)\}; R^{\text{may}} = \{(u_0, u_1), (u_1, u_2)\}; L(u_0) = (q, f), L(u_1) = (q, f), L(u_2) = (q, f); AP = \{q\}\}$ 。

其中系统的事件总集 $AP_u = \{p, q\}$ 。为了比较并融合这些不完整的模型,需要统一它们的描述。模型描述的统一可以分为以下两个部分。

a) 行为统一。在单一的 KMTS 模型描述中,所有的状态转移都是确定的,此时 R^{must} 与 R^{may} 是等同的。每一个模型都只单独地描述其所涉及到的状态。但在统一的描述中,每一个模型必须描述所有与之相关或者可能相关的状态。因此需要增加这些未被描述的状态以及状态之间可能的转移关系。在 K_1 中,为了与 K_2 中的状态以及转移关系相对应,修改其 KMTS 部分描述为 $\{S_1 = \{s_0, s_1, s_2\}; R^{\text{must}} = \{(s_0, s_1)\}; R^{\text{may}} = \{(s_0, s_1), (s_1, s_2)\}\}$ 。

b) 事件统一。对于模型中的每一个状态,需要把其中的事件与事件总集 AP_u 进行比较并进行调整,使得每一个状态的事件与 AP_u 中的事件完全相同。而对于所补充的事件,其属性都以 m 赋值,表示事件可能发生。通过在 K_1 、 K_2 中分别增加事件 q 和 p ,则修改的部分 KMTS 描述为:

$K_1: L(s_0) = \{(p, t), (q, m)\}, L(s_1) = \{(p, t), (q, m)\}, L(s_2) = \{(p, m), (q, m)\}; AP = \{p, q\};$

$K_2: L(u_0) = \{(p, m), (q, f)\}, L(u_1) = \{(p, m), (q, f)\}, L(u_2) = \{(p, m), (q, f)\}; AP = \{p, q\}。$

2.2 一致性模型的判断以及融合条件

设有两个模型 K_1 、 K_2 ,在统一 KMTS 描述后,对 K_1 和 K_2 模型进行一致性关系判断。现定义一致性关系标记符 $\sim \subseteq S_1 \times S_2$,设 s 与 u 分别表示 K_1 与 K_2 中的状态,若 $s \sim u$,则需满足以下几个条件:

a) $\forall p \in AP_u \cdot (L_1(s, p) = t \Rightarrow L_2(u, p) \neq f) \wedge (L_2(u, p) = t \Rightarrow L_1(s, p) \neq f);$

b) $\forall s' \in S_1 \cdot R_1^{\text{must}}(s, s') \Rightarrow \exists u' \in S_2 \cdot R_2^{\text{may}}(u, u') \wedge s' \sim u';$

c) $\forall u' \in S_2 \cdot R_2^{\text{must}}(u, u') \Rightarrow \exists s' \in S_1 \cdot R_1^{\text{may}}(s, s') \wedge s' \sim u'。$

其中 p 表示 AP_u 中的事件, s' 、 u' 分别表示模型 K_1 、 K_2 中的状态。以上模型一致性的判断是具有递归效应的,若 $s \sim u$,则 s 与 u 状态中的所有事件都保持一致性,并且它们的后继者也保持一致性。因此,若是对于模型的初始状态 s_0 、 u_0 ,满足 $s_0 \sim u_0$,则可以推出 $K_1 \sim K_2$,即两模型保持一致性。

在统一模型 KMTS 描述以及确定模型之间满足一致性后,以 $K_1 + K_2$ 表示模型 K_1 与 K_2 的融

合,结果用一个元组 $(S_+, (s_0, u_0), R_+^{\text{must}}, R_+^{\text{may}}, L_+, AP_1 \cup AP_2)$ 表示,并满足以下几个条件:

a) $S_+ = \{(s, t) | s \sim u\};$

b) $R_+^{\text{must}} = \{((s, u), (s', u')) | (R_1^{\text{must}}(s, s') (R_2^{\text{may}}(u, u')) \vee (R_1^{\text{may}}(s, s') (R_2^{\text{must}}(u, u')))\};$

c) $R_+^{\text{may}} = \{((s, u), (s', u')) | (R_1^{\text{may}}(s, s') (R_2^{\text{may}}(u, u')))\};$

d) $\forall p \in AP_u \cdot L_+((s, u), p) = L_1(s, p) \nabla L_2(u, p)。$

其中 ∇ 表示信息量的并集,本文设定事件的属性值 t 与 f 所包含信息量均大于 m ,则有 $t \nabla m = t, f \nabla m = f$ 。在两个一致性的状态进行融合时,状态中的每一个事件都要进行综合考虑,并赋予融合后的属性值。对于一致性状态间的转移关系,需区分确定性转移和可能性转移,便于融合之后进行简化。

2.3 模型融合算法

对于两个相关的适应模型,在融合前经过判断并确定满足一致性后,分析对应的融合条件,则本文基于部分行为模型的一致性模型融合的流程如图 2 所示。

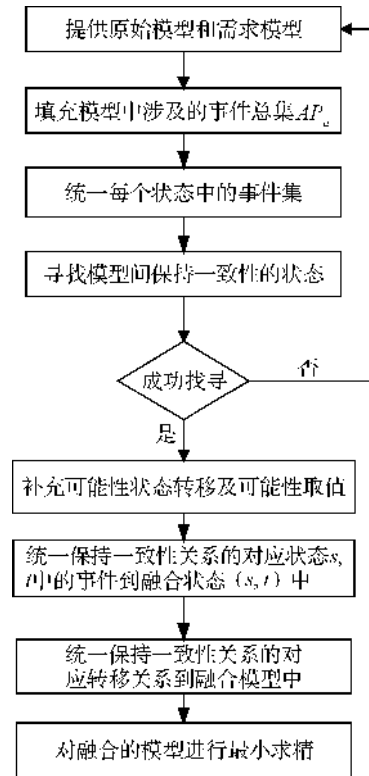


图 2 行为模型融合流程

在融合过程中,主要关键步骤为:

a) 对原有模型和需求模型进行广度优先搜索,统一状态中的事件集,使得每个状态都包含 AP_u 中的所有事件。对于状态中新增的事件 p ,以 m 表示

其真值。

b) 以原有模型的初始状态 s_0 为出发点, 遍历需求模型的所有状态, 找出与之保持一致性的某个状态, 反之亦可。若都未成功找寻, 则说明两模型并非一致性, 无法使用本文的融合方法。若找寻成功, 则补充两模型之间未形成对应的状态, 并以虚线箭头表示为可能性转移关系。

c) 统一保持一致性关系的对应状态 s, t 中的事件到融合状态 (s, t) 中, 以及保持一致性的对应转移关系到融合模型中。

d) 对融合模型进行最小求精, 分析状态中真值为 m 的事件, 修改真值为 t 或是 f 。模型融合完成后, 再对照相应的适应逻辑进行验证。

3 实例研究与分析

3.1 模型融合过程实例研究

以智能家居中一个简单的适应模块组合为例, 整个控制系统由模块“数据采集”和“窗帘控制”共同构成。自适应窗帘采用感应控制, 当室外光照强度大于设定值时, 窗帘自动打开可实现室内照明, 反之则自动关闭。窗户玻璃采用特殊材料, 可隔绝室外热量传递, 但是为了方便室内人员观看室外风景, 用户增加了自适应窗户的需求, 通过感应人窗距离从而实现窗户的打开与关闭。因此, 在控制策略上需

要发生调整, 以适应窗帘以及窗户的共同控制需求。

窗帘控制模型中有两个事件, 即电源以及窗帘是否开启。以符号 Power 表示电源开启, \neg Power 表示电源关闭; Curtain 表示窗帘打开状态, \neg Curtain 表示窗帘关闭。KMTS 描述如图 3(a) 所示。对于窗户控制需求模型, 同理, 用 Window 表示窗户打开, \neg Window 表示窗户关闭。在窗帘打开的前提下, 只要室内有人距离窗户距离小于设定值时, 窗户便能自动打开, 反之窗户则处于关闭状态, 该需求模型 KMTS 描述如图 3(b) 所示。

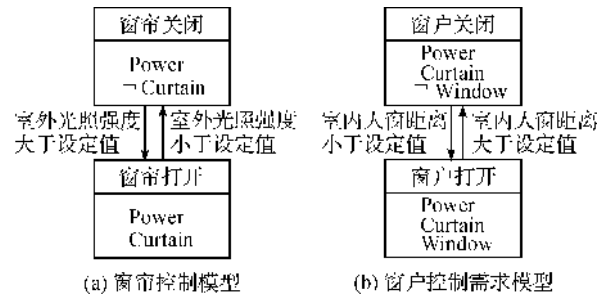


图3 单一的 KMTS 描述

但在图 3(a) 中各个状态中, 只包含事件 Power 以及 Curtain, 缺少了需求模型中的事件 Window, 同时也缺少了对于感应人窗距离后所可能发生的状态转移。因此可以把它补充完整, 如图 4(a) 所示。同理, 图 4(b) 为窗户控制需求模型的统一 KMTS 描述。

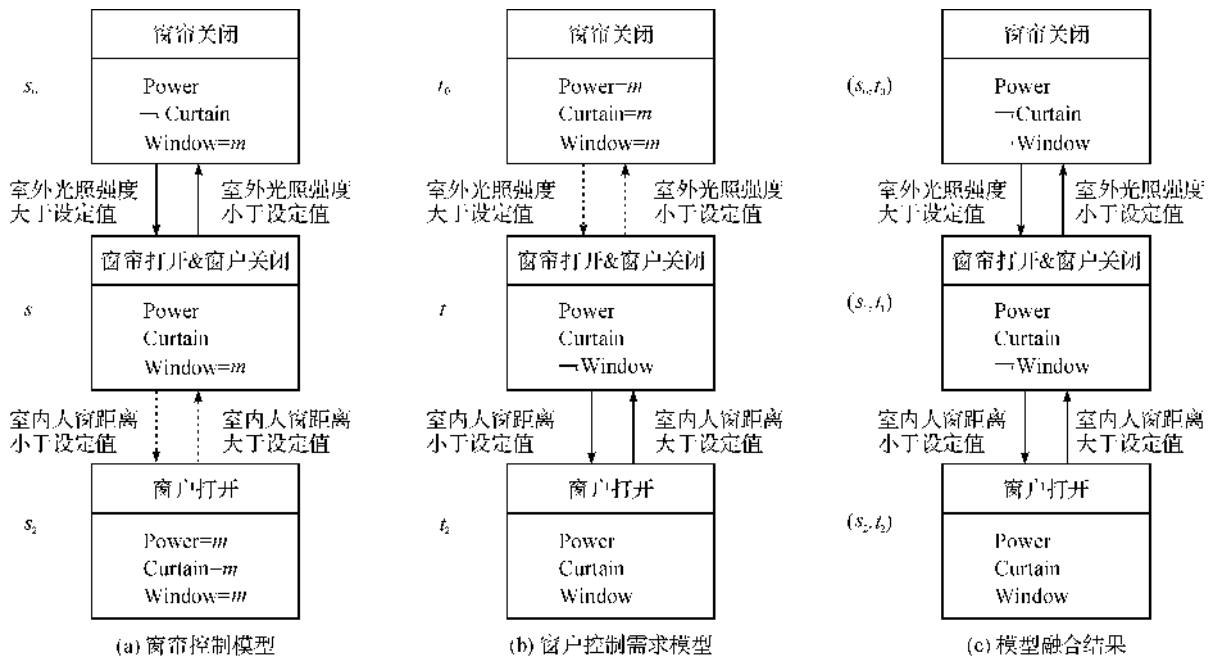


图4 统一的 KMTS 描述

在统一窗帘控制模型以及窗户控制需求模型的 KMTS 描述后, 根据前文所给出的适应模型融合算

法, 判断两模型的一致性关系, 从而进行模型融合, 融合结果如图 4(c) 所示。

3.2 模型融合开发展示

以安卓平台为开发工具,对本文实例进行简单开发,界面如图 5 所示,主要把整个系统分为三个部分:a) 显示区,可显示窗帘、窗户事件,并以红、绿指示灯表示关闭、开启状态。在现实生活中,只有显示区是呈现在人们眼前的。b) 感应区,由各类传感器

组成,以指示灯显示感应状况。例如窗帘传感器开启时,可以感应窗外的光照强度;同理,窗户传感器可以感应室内的人窗距离。c) 控制区,可以根据感应区产生的反馈,采用固定的适应逻辑,以控制相应事件的开启与关闭。控制区处于整个系统的控制中枢,对于系统的成功运作起着至关重要的作用。



图 5 实例图

图 5(a)表示了整个系统只需要实现窗帘的控制,控制区固有的适应逻辑就是根据窗帘感应器的状态实现对窗帘的控制。在适应逻辑不变的情况下,窗帘控制与其相应的感应器是保持一致的。当增加自适应窗户的需求时,如图 5(b)所示,若未进行模型融合或者融合模型不正确时,控制区的适应逻辑是不正确的。因此,即使窗户传感器感应到人窗距离小于设定值时,也不能使得对应的窗户打开,其旁边的指示灯仍然是红色的。只有适应逻辑进行正确的改变,整个系统的控制功能才能发生作用,使得窗帘与窗户均能开启或关闭,如图 5(c)所示。

在实际操作过程中,把窗帘与窗户抽象成两个部分行为模型,分别进行建模。通过第二节的融合算法,若两者能够相互融合,则它们的适应逻辑也会同时发生融合。完整融合之后的适应逻辑便能够正确并且有效地控制整个系统。

3.3 融合结果分析

融合模型必须在理解模型结果的前提下,从约束和场景两方面进行分析。

在模型中,每一个事件都可以看成是其行为表现,而融合的模型在事件上必定有所关联。窗帘和窗户的模型融合,都要保证电源的开启。同时,只有在窗帘打开的状态下,才能正常地实现窗户的打开与关闭。但是对于融合模型的每个状态,包含的事

件数有最低和最高约束。一般来说,很少可以包含所有事件。那么,对于融合模型中所有取值为 m 的事件,如果不能从相连事件中确定它的确切真值,则需要从状态中去除。该实例中,在模型融合后,窗帘关闭状态中的 Window 真值为 m ,由前提设定中可知窗帘关闭时不实现窗户的自适应控制,因此可分析得到它的确切真值是 f ,应该保留。同时,对于相同事件,在同一个状态中也不应该重复出现。

原始模型和需求模型,对应不同的场景。如窗帘控制模型,为了满足房间内的照明需求。而窗户控制需求模型,则为了方便室内人员观看室外风景。从场景分析中发现,窗户可以实现自适应开关的前提是窗帘已经打开。因此,当模型融合之后,融合结果也必须符合两者的相关关系。

4 结 论

在智能家居中的自适应软件开发过程中,模块组合可以有效解决复杂的建模问题。然而,由于专用模块适应逻辑存在着高复杂度、低复用性等问题,很难验证其组合后适应逻辑的有效性和正确性。针对此问题,本文通过将部分行为模型的形式化方法引入到自适应软件的适应行为的描述中,从系统的局部行为来分析适应逻辑的动态组合过程,使得模型融合形式化,并为融合结果的验证提供了有力基础。本文采用的 KMTS 描述语言,增加了对可能行

为和可能状态的支持,可以有效地提供模型对未知和可重配置信息的描述。因此,当两 KMTS 模型具有一致性时,它们的融合便可以看作生成最小共同求精 LCR 模型的过程。本文提出的一致性模型判断以及融合算法也为融合模型的验证提供了有效依据。

本文所研究的模型融合方法,主要是针对一致性的适应模型之间。并且,在基于固定需求的模型融合时,需保持自适应环境的稳定。我们将进一步针对非一致性模型之间的融合以及多变环境的适应情况进行研究。

参考文献:

- [1] Maurel Y, Diaconescu A, Lalanda P. Creating complex, adaptable management strategies via the opportunistic integration of decentralised management resources[C]//Proceedings of the 2009 International Conference on Adaptive and Intelligent Systems. Washington: IEEE Computer Society, 2009: 86-91.
- [2] Sicard S, Boyer F, De Palma N. Using components for architecture-based management: the self-repair case [C]//Proceedings of the 30th International Conference on Software Engineering. New York: ACM, 2008: 101-110.
- [3] 接钧靖, 史庭训, 焦文品, 等. 自主构件自适应策略的在线定制及动态评估[J]. 软件学报, 2012, 23(4): 802-815.
- [4] Uchitel S, Alrajeh D, Ben D S, et al. Supporting incremental behaviour model elaboration[J]. Computer Science-Research and Development, 2013, 28(4): 279-293.
- [5] Gui N, Florio V D, Blondia C. Transformer: an adaptation framework with contextual adaptation behavior composition support[J]. Software Practice & Experience, 2013, 43(8): 937-967.
- [6] Uchitel S, Kramer J, Magee J. Behaviour model elaboration using partial labelled transition systems[J]. ACM SIGSOFT Software Engineering Notes, 2003, 28(5): 19-27.
- [7] Redondo R P D, Pazos A J J. Reuse of verification efforts and incomplete specifications in a formalized, iterative and incremental software process[C]//Proceedings of the 23rd International Conference on Software Engineering. Toronto: IEEE Computer Society, 2001: 801-802.
- [8] Nejati S, Sabetzadeh M, Chechik M, et al. Matching and merging of variant feature specifications [J]. Software Engineering, IEEE Transactions on, 2012, 38(6): 1355-1375.
- [9] 易立, 赵海燕, 张伟, 等. 特征模型融合研究[J]. 计算机学报, 2013, 36(1): 1-9.
- [10] Fischbein D, Uchitel S. On correct and complete strong merging of partial behaviour models [C]//Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering. New York: ACM, 2008: 297-307.

Research on Consistency Model Fusion Technology Based on Smart Home

BAO Xiao-an¹, LIN Hui¹, ZHOU Jian-ping², GUI Ning¹, SUN Xian-ce¹, ZHANG Na¹

(1. The School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China; 2. Haiyan Shengdi Electrical Technical Co., Ltd., Jiaxing 314300, China)

Abstract: In the self-adaptive software design of smart home, the traditional module construction technology is often adopted. However, due to its low reusability and high complexity in adaptive logics, it is very difficult to confirm the correctness and effectiveness of module combination. In order to meet users' incremental demand in development process, this paper proposes to introduce formalization method of partial behavioral model into the description of adaptive behavior based on combination difficulty of adaptive logics. Through three-valued logic KMTS model description language, this paper proposes judgment method and fusion algorithm of consistency model to provide clear online fusion support for adaptive model. Finally, this paper takes a smart home model as an example to analyze and confirm the adaptive logics after fusion.

Key words: smart home; model fusion; adaptive logics; KMTS; consistency model

(责任编辑: 陈和榜)