PSC2GS:一个基于属性序列图的监控器生成工具

余 俊[1] 张鹏程[1,2] 周宇鹏[1] 刘宗磊[1]
(河海大学计算机与信息学院 南京 211100)[1] (南京大学计算机软件新技术国家重点实验室 南京 210093)[2]

摘 要 在开放和动态环境下,系统或环境的不安全的运行时变化可能为整个系统的正确执行埋下隐患,可能最终导致软件失效。基于监控器的软件运行时验证技术已经成为开放环境下侦测软件失效行为的基本方法,该工具采用了一种基于博弈论的从Property Sequence Charts(属性序列图)中自动生成监控器的方法。监控器被赋予多值语义:满足、无限可控、系统有限可控、系统紧急可控、环境有限可控、环境紧急可控以及违例。监控器可以提供足够的信息用来预测系统失效。正文中将描述一个名为"PSC2GS"的工具,该工具具有设计属性序列图、基于属性序列图生成博弈结构、基于博弈结构生成Aspect Oriented Programming(面向方面编程)代码(监控器)等一系列功能。PSC2GS提供的完全图形化的前端接口使软件设计者可以不用处理任何特殊的文本或者逻辑公式。

关键词 运行时验证,监控器,属性序列图,面向方面编程
中图法分类号 TP315.69 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.09.015

PSC2GS:A Tool for Generating Monitors Based on Property Sequence Charts

YU Jun[1] ZHANG Peng-cheng[1,2] ZHOU Yu-peng[1] LIU Zong-lei[1]
(College of Computer and Information, Hohai University, Nanjing 211100, China)[1]
(State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing 210093, China)[2]

Abstract In open and dynamic environments, unsafe run-time changes of system or environments may compromise the correct execution of the entire system. It may eventually lead to the occurrence of software failures. Run-time verification techniques based on monitors have become the basic means of detecting software failures in open environments. This tool proposes an automated approach to generate monitors from property sequence charts based on game theory. The monitors are interpreted in multi-valued semantics: satisfied, infinitely controllable, system finitely controllable, system urgently controllable, environment finitely controllable, environment urgently controllable, and violated. The monitors can provide enough information to help the system to take measures for failure prediction or prevention. This paper described a novel tool called 'PSC2GS', which is able to design property sequence charts, generate game structure based on property sequence charts and generate Aspect Oriented Programming codes (the Monitor) based on game structure. The tool chain provides a completely graphical front-end which can make software designers do not have to deal with any particular textual and logical formalism.

Keywords Run-time verification, Monitor, Property sequence charts, Aspect oriented programming

1 引言

近年来,由于SOA(Service-Oriented Architecture)这类新范式的出现,软件密集型系统的环境正在从静态、封闭、可控向动态、开放、不可控发展。这种开放环境下的软件系统行为很容易受到其内部框架以及来自外部环境输入的影响,从而导致软件失效[1]。

传统的设计时验证技术只能保证软件系统的行为满足封闭或者可控环境下需要的属性,不能完全考虑到系统运行时不停变化的环境以及系统的执行状态。运行时验证的主体思想是生成一个可以检查运行时行为是否满足系统所需要属性的监控器,这些属性用一种特定的语言表示[2]。属性的验证大多集中在基于逻辑规范和基于场景规范两种方法。基于逻辑的形式通常比基于场景的形式具有更强的表达能力,但是基于场景的方法提供了在工业实践中被广泛采用的图形规范形式。

设计一个高效的监控器必须要遵循以下标准:(1)监控器必须是自动生成的,不能有人为干预来仲裁是否违例。(2)监控器必须在不影响表达能力的情况下尽可能简单。(3)监控器不可以占用系统太多资源,尽可能少地影响系统的性能。

到稿日期:2013-12-25 返修日期:2014-01-16 本文受国家自然科学基金(61202097,61202136),中国博士后基金(2012T50489,2011M500897),教育部博士点专项基金(20120094120009)资助。

余 俊(1990-),男,硕士生,主要研究领域为软件建模、分析和验证技术,E-mail:yujunhhu@gmail.com;张鹏程(1981-),博士,副教授,CCF高级会员,主要研究领域为软件建模、分析和验证技术,E-mail:pchzhang@hhu.edu.cn(通信作者);周宇鹏(1990-),硕士生,主要研究领域为软件模型验证;刘宗磊(1990-),硕士生,主要研究领域为基于服务的软件工程。

(4)监控器应该尽早地检测到潜在的失效,甚至能够预测一些系统的失效,提供足够的信息来阻止以及恢复失效。

与本文相关的工作主要有两方面:第一,从一种规范语言生成监控器。比如,用 Java PathExplorer 将 LTL(Linear Temporal Logic)公式转化成使用 Büchi 自动机[3]。但是,这些方法大多基于传统语言的两值语义而没有考虑多值语义,违背了标准(2);第二,将博弈理论应用于规范语言的语义和合成。曾经有人在合成算法中扩展应用过博弈理论,他们将 LSC(Live Sequence Chart)验证看成系统和环境之间的博弈[4]。合成问题变成了系统和环境之间的一种博弈,这种博弈以寻求胜利战略为最终目的。而监控不同于合成,合成只检查战略的初始状态,监控则要检查每个交互的信息。

为了遵循这些标准,本文以博弈论以及基于场景的验证提出了一种自动生成监控器的新方法。博弈论模型已经被广泛地应用在开放系统中,以控制问题的分析与解决。在之前的工作[5,6]中我们提出博弈论观点并用于开放环境下系统组件之间以及系统组件和其相应的环境之间交互的验证,而且,博弈论模型也能提供监控器的多值语义。因此,本文重点研究基于现有的图形化规范形式——属性序列图和博弈理论对软件系统进行运行时监控,并开发相应工具。

2 方法

2.1 理论基础

属性序列图是一种基于场景的可视化语言,容易理解并具有较强的表达能力。属性序列图中的基本元素如图 1 所示。

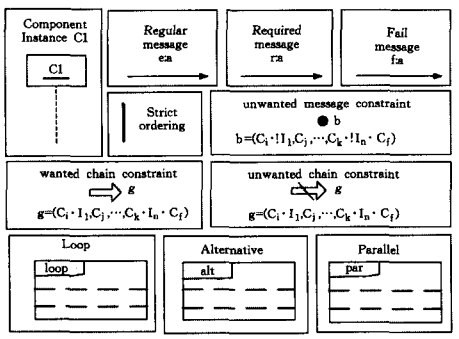

图 1 属性序列图图形元素

根据属性序列图的语义将消息分为 3 类:正则消息(e:)、强制消息(r:)、错误消息(f:),并定义了严格操作(strict),而没有定义严格操作的就是松散顺序。在属性序列图中可对消息添加限制,如:unwanted message constraint、unwanted chain constraint、wanted chain constraint 等等,这些限制都包含 past 和 future 两类。属性序列图还可以用来表示并(parallel)、循环(loop)、交替(alternative)等操作。有关 PSC 的详细描述参看文献[7,8]。

博弈结构(Game Structure,GS)在语句结构上跟普通的自动机比较相似,但是不同的是,博弈结构具有多值语义,可以提供更详细的监控结果并且提供足够的信息以实现阻止和恢复失效[1]。而自动机一般只有 true 和 false 两个值。博弈结构的状态和消息分别可以分为系统可控和环境可控。本文生成的博弈结构的状态根据系统或者环境、可控或者不可控,分为 7 类来解释 PSC 的语义:满足、无限可控、系统有限可控、系统紧急可控、环境有限可控、环境紧急可控以及违例。

Aspect Oriented Programming(面向方面编程)[9]可以通过预编译方式和运行期间动态代理实现在不修改源代码的情况下给程序动态地添加功能。AOP 代码就是前文提到的"监控器",AOP 代码是根据 GS 所提供的信息生成的。本文利用 AOP 技术可以在不对原有系统代码做任何修改的前提下对各个组件或服务添加监控器,该监控器可以捕捉到系统内部以及系统内部与外部环境之间的信息交互,判断系统的行为是否"合法",从而可以对整个系统的运行时监控甚至对系统失效行为做出预测、阻止和恢复[10]。

2.2 框架概述

PSC2GS 的框架如图 2 所示,整个流程分为如下几个步骤:

(1)设计属性序列图:软件设计者根据开放式系统各个组件之间以及内部组件与外部环境之间的交互在 PSC2GS 的 PSC Editor 中画出软件系统中各组件以及内部组件与外部环境之间交互的图形化流程。对各个消息加上必要的限制以及对相应的消息加上需要的操作。

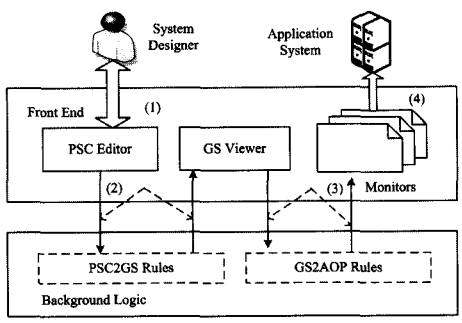

图 2 PSC2GS 软件框架

(2)生成博弈结构:根据(1)所设计的属性序列图将可视化的软件系统运行流程转化成文本形式的博弈结构,博弈结构形式类似自动机但是在状态和消息的表示上略有不同。博弈结构包含了自动化生成监控器的基本信息,比如:消息的 source 与 target,消息内容以及 source 端发送消息所调用的方法,target 端接受消息所调用的方法、消息类型等等。

(3)生成 Monitor:根据(2)中生成的博弈结构,生成相应的监控器。本文的策略是针对系统中每个产生交互的组件或服务都生成一个单独的监控器。

(4)对系统进行监控:对每个组件的监控器进行必要的修改,使之可以在系统中顺利运行从而达到在不修改原有系统的情况下对原有系统进行监控。

2.3 软件实现

针对本文现有的工作,我们开发了原型工具——PSC2GS。PSC2GS 基于 Eclipse Rich Client Platform(RCP)平台开发,其中包含了 120 多个 java 类以及 20000 行左右的代码。开发过程采用了 Eclipse Graphical Editing Framework (GEF)编程模型。

图 3 展示了该工具的用户交互主界面。这是一个简单的字典查询系统的例子,可以实现本地和远程两种查询方式。PSC2GS 拥有以下组件和接口:导航窗口、PSC 编辑器、GS 窗口、属性窗口、控制台等等。导航窗口用来管理和显示打开的项目和文件。PSC 编辑器为用户提供人为设计属性序列图的调色板窗口以及画布,其中调色板窗口为用户提供各种可以

拖放的属性序列图元素。Lifeline 以及 message(system 或 environment)被选中后它们的每个属性都显示在属性窗口。GS 窗口会显示 PSC 编辑器中属性序列图生成的博弈结构，博弈结构用文本的形式显示。最后，控制台窗口会显示将属性序列图转化成相应博弈结构所花费的时间以及由博弈结构生成的监控器雏形亦即与博弈结构相对应的 AOP 代码，非正常情况下也会提示工具运行过程的部分提示或错误信息。

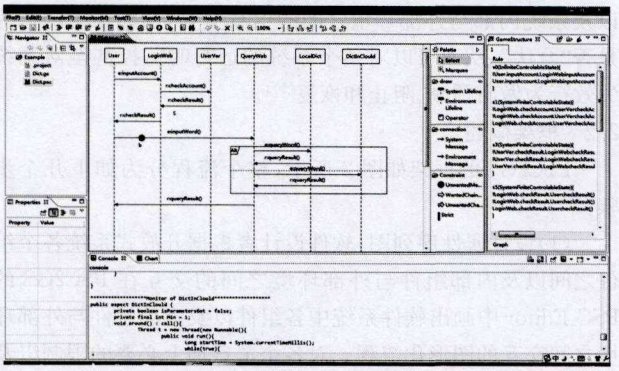

图 3　PSC2GS 主体窗口

3　实例

本节将以一个简单的字典查询实例 Dict 来解析 PSC2GS 的具体应用。在此只简单讲解对 Dict 系统添加监控器的过程，详细的过程参照本文在 sourceforge. net 上的一篇博文[1]。

Dict 是用 OSGi(Open Service Gateway Initiative)实现的一个字典查询系统。整个系统由以下几个 Bundle 组成：User，DictQuery，DictQueryWeb，LocalDictQuery，RemoteDictQuery，UserVer，LoginWeb。其中 DictQuery 的功能只是向外提供接口，因此不参加整个系统交互。

首先，根据 Dict 系统交互过程将整个字典查询流程设计成属性序列图。步骤如下：

(1)用 PSC2GS 画出各个 bundle；(2)根据交互过程画出各个 bundle 之间可能传递的 message；(3)填写完整 message 的类型、内容以及参数等等；(4)画出特定的 message 所需要的 constraint；(5)对于需要并行、交替或者循环的 message 加上相应的 operator，结果如图 4 所示。

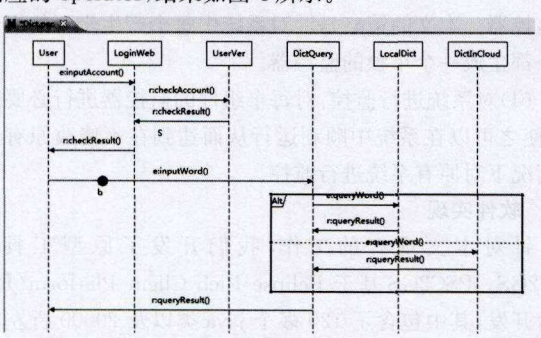

图 4　Dict 查询系统交互

其次，将属性序列图转化成 Dict 查询系统相应的博弈结构。在博弈结构生成之后，可以立即生成 Dict 系统监控器，如之前所述，OSGi 应用中的每个 bundle 对应一个单独的 monitor，因此在 Dict 系统中会相应地生成 6 个 monitor。这是一个从设计属性序列图到生成监控器的完全自动化的流程。如果我们直接拥有某个系统的博弈结构而没有属性序列图也是可以生成监控器的，将博弈结构拷贝到 GS 窗口后直接生成 AOP 代码。

在监控器生成之后本文需要对其中某些代码做一些必要的改动，比如相应的 jar 包的导入以及一些 AOP 切入点的指定等等，然后将其加入原有项目之中(要想系统可以顺利运行 AOP 代码，Eclipse 需要安装一些插件以及一些设置[2])。

最后，运行 Dict 系统，需要强调的是一直到最后一步本文都没有对原来的 Dict 查询系统的代码做任何修改。只要了解交互过程中各个 bundle 发送以及接受消息所调用的方法，就可以忽略系统的具体代码实现。加了监控器之后的 Dict 系统的正常的查询过程运行结果如图 5 所示。

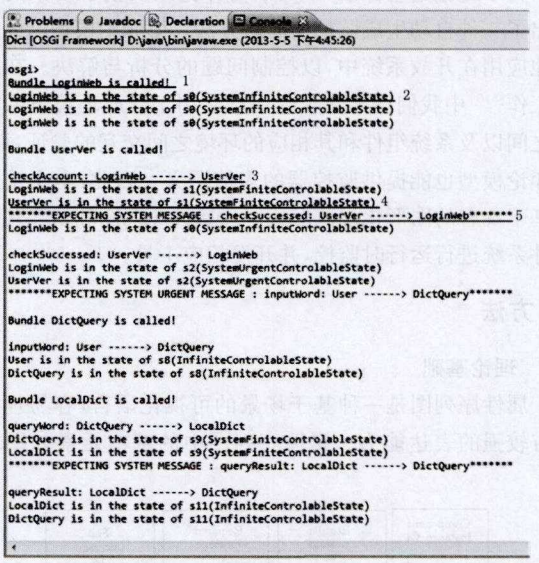

图 5　添加了 Monitor 之后 Dict 运行结果

显然，正常情况下监控器会打印出系统相关行为的相关组件状态，如果系统出现失效也会打印出失效的状态。打印出的状态充分体现了基于博弈结构的监控器多值语义优点。

系统运行时若 bundle 被调用，则系统会打印出一条提示信息，如图 5 中 1 所示。同时，监控器每隔一秒钟会打印一次该 bundle 的状态。图中 2 所指的 LoginWeb bundle 重复打印 3 次是人为设置的，这里为了突出效果让主线程休眠了 3 秒。当系统中组件之间有消息传递时，监控器会打印出消息内容以及消息发送方和接收方，如图中 3 所示。当消息传递成功之后，消息的发送方和接受方的状态都会随之改变，此时监控器会打印出相关 bundle 的最新状态，如图中 4 所示。系统处于新状态时，监控器会打印出系统当前状态正常情况下跳转至下一状态所需的 Message(包括 message 内容以及发送方和接收方)，如图中 5 所示。

结束语　本文描述了基于博弈理论根据软件系统的属性序列图生成具有多值语义的监控器的工具 PSC2GS。我们可以利用监控器精确地监控软件系统行为。基于博弈结构的监控器甚至提供了足够的信息来预测潜在失效，并且对失效进行阻止或恢复。

PSC2GS 基于属性序列图的监控器生成器将 PSC 转化成

[1] 有关 Dict 项目的详细过程参照：http://sourceforge. net/p/psc2gs/wiki/Manual%20on%20PSC2GS%20with%20Dict%20Example/

[2] 基于 Equinox 的 Aspect 项目部署：http://www. eclipse. org/equinox/incubator/aspects/equinox-aspects-quick-start. php

博弈结构从而自动生成 AspectJ 代码。整个过程无需其他的中间框架，比如 JavaMOP[11] 或者 tracematches[12]。

本文对 PSC2GS 进行详细描述是为了展示如何从系统的属性序列图得到博弈结构，从而生成监控器，最终对目标系统进行监控的全部过程。本文中的方法提供图形化界面，从而避免了处理特殊文本或者逻辑公式。

然而，现阶段由于信息量的限制，本文自动生成的 AOP 代码在直接使用之前还需要一定程度的人为修改。尽可能提高所生成监控器的自动化程度将是我们需要解决的问题之一，也是将来的工作重点。因此，我们将继续开发并提高该工具来帮助软件工程师满足其对软件系统的监控需求。与此同时，我们正考虑利用'PSC2GS'产生的监控器在大型 OSGi 应用上进行监控，并测试其对系统的性能影响。并且基于本文已有工作我们将来会尝试将这套理论以及工具应用于移动端 android 平台和云平台。

参 考 文 献

[1] de Alfaro L, Stoelinga M. Interfaces: A game-theoretic framework for reasoning about component-based systems[J]. Electr. Notes Theor. Comput. Sci., 2004, 97: 3-23

[2] Zhang X, Leucker M, Dong W. Runtime verification with predictive semantics[C]// NASA Formal Methods. 2012: 418-432

[3] Giannakopoulou D, Havelund K. Automata- based verification of temporal properties on running programs[C]// ASE 2001. IEEE Computer Society, 2001: 412-416

[4] Harel D, Segall I. Synthesis from scenario-based specifications [J]. J. Comput. Syst. Sci., 2012, 78(3): 970-980

[5] 张鹏程，李宣东，李雯睿. 基于博弈论的开放环境下场景规约监控语义[J]. 中国科学信息科学，2013

[6] Zhang P, Yu J, Li W, et al. Game-based monitors for scenario-based specifications[C]// The 18th International Conference on Engineering of Complex Computer Systems (ICECCS2013). 2013: 264-267

[7] PSC Project. PSC Web site [OL]. http://www.di.univaq.it/psc2ba, 2005

[8] Autili M, Inverardi P, Elliccione P P. A scenario based notation for specifying temporal properties [C]// The 5th SCESM06, ICSE06. Shanghai, 2006

[9] The AspectJ Project. AspectJ Web site [OL]. http://eclipse.org/aspectj/

[10] Ehlers R, Finkbeiner B. Monitoring realizability[C]// RV 2011. 2011: 427-441

[11] Jin D, Meredith P O, Lee C, et al. Javamop: Efficient parametric runtime monitoring framework[C]// ICSE. 2012: 1427-1430

[12] Allan C, Avgustinov P, Christensen A S, et al. Adding trace matching with free variables to aspectj[C]// OOPSLA. 2005: 345-364

(上接第 70 页)

服务的发展。但该原型系统尚有不足和未完善之处，如提高服务标注准确率、提高服务组合准确度和效率等。这也是我们下一步努力的重点和方向。

参 考 文 献

[1] Papazoglou M P, Van Den Heuvel W J. Service oriented architectures: approaches, technologies and research issues [J]. The VLDB journal, 2007, 16(3): 389-415

[2] Newcomer E, Lomow G. Understanding SOA with web services (independent technology guides) [M]. Addison-Wesley Professional, 2004

[3] Benini L, De Micheli G. Networks on chips: A new SoC paradigm [J]. Computer, 2002, 35(1): 70-78

[4] Brown C W, Nyarko K. Software as a Service (SaaS) [J]. Cloud Computing Service and Deployment Models: Layers and Management, 2012: 50

[5] 杜宗霞，怀进鹏. 主动分布式 Web 服务注册机制研究与实现[J]. 软件学报，2006, 17(3): 454-462

[6] 廖祝华，刘建勋，刘毅志，等. Web 服务发现技术研究综述[J]. 情报学报，2008, 27(2): 186-192

[7] 兰明敬. 语义聚集的 P2P 服务组织模型 [J]. 计算机科学，2013, 40(4): 78-82

[8] Chen Shi-zhan, Feng Zhi-yong, Wang Hui. Building the Semantic Relations-Based Web Services Registry through Services Mining [C]// IEEE/ACIS International Conference on Computer and Information Science. Shanghai, 2009: 736-743

[9] Liang Qi-xuan, Chen Shi-zhan, Feng Zhi-yong. Application Services Relation Tracing to Automated Web Service Compositon [J]. Applied Mathematics & Information Services, 2013, 7(1): 243-251

[10] Zheng Hui-yuan, Zhao Wei-liang, Yang Jian. Qos Analysis for Web Service Composition[C]// IEEE International Conference on Services Computing. 2009: 235-242

[11] Wang Hui, Feng Zhi-yong, Sui Yang, et al. Service Network: An Infrastructure of Web Service [C]// IEEE International Conference on Intelligent Computiong and Intelligent Systems. Shaihai, 2009: 303-308

[12] Wang Hui, Feng Zhi-yong, Chen Shi-zhan, et al. Constructing Service Network via Classification and Annotation [C]// 5th IEEE International Workshop on Service-Oriented System Engineering. Nanjing, China, 2010: 69-73

[13] 陈世展. 服务网络：基于语义和社会化关系的 Web 服务计算基础设施 [D]. 天津：天津大学，2009

[14] 王辉. 面向互联网的 Web 服务基础设施构建和应用 [D]. 天津：天津大学，2011

[15] 王辉，冯志勇，陈世展. 基于服务网络的服务关系挖掘 [J]. 计算机应用研究，2010, 8(27): 2962-2966

[16] Guo Y, Chen S, Feng Z. Composition Oriented Web Service Semantic Relations Research [C]// 2011 International Joint Conference on Service Sciences(IJCSS). IEEE, 2011: 69-73