一个基于下推自动机的 Web 测试自动执行器*)

贾晓霞[1] 刘 昶[1] 吴 际[1] 柳永坡[1,2] 金茂忠[1] 刘 超[1]

(北京航空航天大学计算机学院 北京 100083)[1] (大庆石油学院计算机与信息技术学院 大庆 163318)[2]

摘 要 随着Web应用系统的广泛应用,对其质量要求也越来越高。如何进行有效的测试以保证Web应用系统的质量是值得关注的研究问题。支持测试用例自动执行的测试执行器(Test runner)是其中的一个热点和难点。文章研究并提出了基于下推自动机的Web测试自动执行器,并实现了其原型系统。该执行器支持测试用例的自动执行并给出测试结果报告,并通过Web应用测试实例验证了其有效性。

关键词 测试自动执行器,下推自动机,Web测试

A Novel Web Test Runner Based on Pushdown Automation

JIA Xiao-Xia[1] LIU Chang[1] WU Ji[1] LIU Yong-Po[1,2] JIN Mao-Zhong[1] LIU Chao[1]

(School of Computer Science and Technology, Beijing University of Aeronautics and Astronautics, Beijing 100083)[1]

(School of Computer and Information Technology, Daqing Petroleum Institute, Daqing 163318)[2]

Abstract With the wide spread of Web-based application systems, there have been rising demands on the quality of such systems. Accordingly, valid testing approach to guarantee the quality of Web-based application systems has become a subject of intense research, in which area, automated test runner to support automatic testing execution is currently a hotspot and a tough problem. Proposed and analyzed in this paper is a novel Web test runner based on pushdown automation, and a prototype of the Web test runner is implemented. The Web test runner is able to execute the testing cases automatically and present the testing reports, its performance is studied and its validity is proved via practical testing routines for Web applications.

Keywords Test runner, Pushdown automation, Web testing

随着网络技术的发展,Web应用系统在电子商务、教育、娱乐、电子政务、医疗、金融等行业得到了越来越广泛的应用,对其可靠性和质量的要求也越来越高。而测试是质量保证的主要手段,如何对Web系统进行有效的测试是一个热门的研究问题。

目前对Web应用测试的研究非常多,从是否深入实现代码可分为白盒测试和黑盒测试。白盒测试包括分析源代码进行内容检查、页面链接检查等[1,2];对源代码进行单元测试和集成测试[3];测试Web页面中的Java Applet或Active X控件等[4];基于HtmlUnit测试框架进行测试等[5]。黑盒测试则包括功能测试、性能测试、负载测试、兼容性测试等[6~8]。

Web功能测试是Web测试中非常重要的一部分[9,10]。功能测试时的测试用例为Web操作序列,手动执行该操作序列费时费力,如何实现其自动执行是个有价值的研究问题。本文提出了基于下推自动机的测试自动执行器,支持测试用例的自动执行并给出测试结果报告。

本文第1节介绍了Web测试自动执行方法的相关研究;第2节提出了基于下推自动机的Web测试自动执行器(以下称为测试自动机);第3节给出了Web应用系统测试实例;最后总结了全文。

1 相关研究

当前对Web应用测试自动执行方法的研究非常多,本节主要介绍以下几个方面:

Oliver等提出的方法使用测试图(test graph)表示测试用例。测试图中的节点表示测试块(test block),每个测试块同一个Web操作关联。测试用例便表示为这样一个测试块的序列(对应于测试图中的一条路径)。测试用例的自动执行是通过一个tracer完成的。tracer执行某测试块时调用其相关操作的实现,并记录响应信息和操作结果[10]。

Yang等[11]构建的Web测试框架包括对测试用例的开发、管理、执行及结果判断等。测试执行系统采用脚本技术完成,利用脚本语言丰富的语法支持Web应用操作,包括录入form所需数据,验证Web服务器的响应等。

文[6,12~14]采用"录入-回放"工具记录用户定义的测试场景,并依据其产生脚本(浏览器执行序列),浏览器可以自动执行这些脚本进行回归测试。但Web结构改变时,已有测试场景的鲁棒性就会受到挑战。VeriWeb[15]支持测试场景的自动发现,部分弥补了前述不足。

从上述介绍可以看到,基于一定的测试用例格式,很多方法支持测试用例的自动执行。本文提出的基于下推自动机的测试自动执行器(以下称为测试自动机)是其中的一种。下一节首先介绍测试自动机原型的数据流图和采用的测试用例格式,然后在介绍下推自动机理论的基础上,提出测试自动机。

2 Web测试自动机

Web应用测试用例体现为Web操作序列,测试自动机便是设计用来自动执行这些操作序列,并根据程序规格说明判断系统对操作序列的响应是否正确的。Web应用的操作主要包括HTTP协议中标准的"GET"请求和"POST"请求,浏

*)基金项目:国家自然科学基金资助项目(60373016),国家863计划项目(2004AA112030)。贾晓霞 博士生,主要研究方向:软件测试技术,软件工程。刘 昶 硕士。吴 际 博士,讲师。柳永坡 博士生。金茂忠 教授,博导。刘 超 教授。

览器中的“填写表单”、“回退”和“重置表单”等。Web 应用测试用例是由这些操作构成的操作序列，该操作序列类似于一段没有跳转的顺序执行代码。测试用例的特点决定了对它的自动执行可由自动机方便地完成。

本文提出的测试自动机依据测试用例的操作步骤执行动作语义。对 Web 应用程序，自动机利用 HTTP 协议与 Web 服务器通讯，模拟浏览器来代替用户完成各种操作，从而完成对语义动作的执行。操作完成后，测试自动机将 Web 服务器的响应(response)同测试用例中指定的操作结果进行核对，从而判断测试用例是否通过。图 1 为测试自动机原型的数据流图。

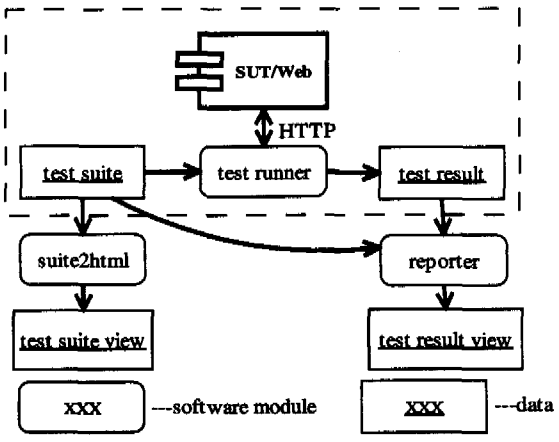

图 1　测试自动机原型的数据流图

图 1 中虚线框表示的是原型中的核心部分，test runner 为测试自动机，其输入为测试用例集。测试自动机采用 HTTP 协议来模拟与被测软件间的交互，并输出测试结果。除此之外，原型系统采用 HTML 形式显示测试用例集，得到测试集视图(test suite view)；而测试结果视图(test result view)中，既包括测试结果报告(由 reporter 根据 test result 生成)，也包括相应采用的测试集。下面介绍测试用例的格式。

2.1　测试用例格式

测试自动机是测试用例的“使用者”，本节从数据使用者的角度介绍测试用例。测试用例的生成方法非常多，本文采用的测试用例是基于测试模型生成的，该测试模型采用扩展的 UML 活动图建立[16]。图 2 中的类图表示测试用例集的结构。

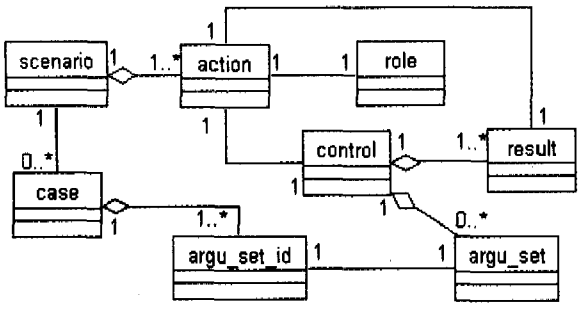

图 2　测试集的类图

从图 2 可以看到：

• 测试用例由测试场景及驱动该测试场景的参数集组成。对同一测试场景，不同的参数集构成不同的测试用例；

• 测试场景由操作序列构成；任一操作都由某个角色完成；控制表示操作的具体动作；每一操作都有一个或多个操作结果；操作可能包含参数。

另外，图 2 中：

• 角色是操作的执行者，也就是系统中的某个用户；

• 操作结果是通过对网页特征的描述给出的，目前讨论的特征包括网页是否包含链接；是否包含指定名称的链接；包含的链接数目等；

测试自动机每次的输入为一操作序列，序列中的每个操作都包含其角色、所需参数及操作结果。测试自动机运行该操作序列，并在每次操作完成后，检查其结果是否同预期一致，并给出测试结果报告。

2.2　下推自动机

本节回顾下推自动机的相关概念[17]，在其基础上作者建立了测试自动机。

定义 1(下推自动机，pushdown automation，PDA)　M 是一个七元组：

$M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，其中

Q：状态的非空有穷集合。$\forall q\in Q$，q 为 M 的一个状态(state)

Σ：输入字母表(input alphabet)

Γ：栈符号表(stack alphabet)。$\forall A\in\Gamma$ 称为一个栈符号

Z_0：$Z_0\in\Gamma$ 为开始符号(start symbol)，也称为栈底符号

q_0：$q_0\in Q$ 是 M 的开始状态(initial state)，也称为初始状态

F：$F\subseteq Q$ 是 M 的终止状态集合，简称为终态集。

δ：状态转移函数(transition function)

$\delta: Q\times(\Sigma\cup\{\varepsilon\})\times\Gamma\to 2^{Q\times\Gamma^*}$，

对 $\forall(q,a,Z)\in Q\times\Sigma\times\Gamma$，$\delta(q,a,Z)=\{(p_1,\gamma_1),(p_2,\gamma_2),\cdots,(p_m,\gamma_m)\}$ 表示 M 在状态为 q，栈顶符号为 Z 时，读入字符 a，自动机有选择地将状态变为 p_i，并将栈顶符号 Z 弹出，将 γ_i 中的符号从右到左依此入栈，输入头前进一个符号。

对 $\forall(q,Z)\in Q\times\Gamma$，$\delta(q,\varepsilon,Z)=\{(p_1,\gamma_1),(p_2,\gamma_2),\cdots,(p_m,\gamma_m)\}$ 称为 M 进行了一次 ε 移动(空移动)，表示 M 在状态为 q，栈顶符号为 Z 时，无论输入字符是什么，自动机有选择地将状态变成 p_i，并将栈顶符号 Z 弹出，将 γ_i 中的符号从右到左依此入栈。

定义 2　设有 PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，则 M 用终态方式接受的语言为：

$L(M)=\{\omega\mid q_0,\omega,Z_0)\vdash^*(p,\varepsilon,\beta),p\in F,\beta\in\Gamma^*\}$

本文提出的测试自动机是一个以终态方式接受操作序列(字符串)的下推自动机。

定义 3　设 PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，$\forall(q,\omega,\gamma)\in(Q,\Sigma,\Gamma^*)$ 称为 M 的一个及时描述(instantaneous description)。它表示：M 处于状态 q，ω 是当前还未处理的输入字符串，且 M 正注视着 ω 的首字符，栈中的字符串 γ，γ 的最左符号为栈顶符号，最右符号为栈底符号，较左的符号在栈的较上面，较右的符号在栈的较下面。

2.3　测试自动机

基于上述下推自动机的定义，可以将测试自动机设计为一个 PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，其中

$Q=\{(0,0),(1,0),(1,1)\}$，集合 Q 中的 3 个状态表示自动机的两个状态变量(page(页面)，form(表单))是否为空。如(1,0)表示 page 非空，form 为空的一个自动机状态；

$\Sigma=\{a,b,c,d\}$ 表示目前讨论的操作，包括：a：post 操作，提交表单；b：get 操作，点击链接；c：fill 操作，填写表单；d：

reset 操作，重置表单。

$\Gamma=\{A,Z_0\}$，A 表示一个历史网页

$q_0=(0,0)$

$F=\{(1,0),(1,1)\}$

状态转移函数如下，其中 $\beta=A^* Z_0$

$\delta((0,0),b,Z_0)=\{((1,0),Z_0)\}$

$\delta((1,0),a,\beta)=\{((1,0),A\beta)\}$

$\delta((1,0),b,\beta)=\{((1,0),A\beta)\}$

$\delta((1,0),c,\beta)=\{((1,1),\beta)\}$

$\delta((1,0),d,A\beta)=\{((1,0),\beta)\}$

$\delta((1,0),e,\beta)=\{((1,0),\beta)\}$

$\delta((1,1),a,\beta)=\{((1,0),A\beta)\}$

$\delta((1,1),b,\beta)=\{((1,0),A\beta)\}$

$\delta((1,1),c,\beta)=\{((1,1),\beta)\}$

$\delta((1,1),d,A\beta)=\{((1,0),\beta)\}$

$\delta((1,1),e,\beta)=\{((1,0),\beta)\}$

测试自动机的状态迁移过程表示测试用例被执行的过程。每次迁移完成后自动机根据操作结果进行判断，当操作结果同预期不一致或当自动机遇到不可接受的操作序列而拒绝后续字符时，可能出现一个失效，此时的自动机状态就是失效现场。

测试结束后给出的测试结果报告包括基本测试信息（测试用例通过数，失效数，测试起止时间），失效现场（失效类型，失效现场的网页快照，堆栈中的历史网页）。

下一节通过 Web 应用实例来说明利用自动机进行的测试。

3 实例

本文实现的测试自动机是一个 Python[18] 程序，和 Web 服务器的通讯通过类库 wwwsearch[19] 完成，利用类库 wwwsearch 和 BeautifulSoup[20] 对返回的网页进行分析。本节采用的实例是基于 Web 的图书借阅系统，图 3 为系统中图书借阅的活动图。

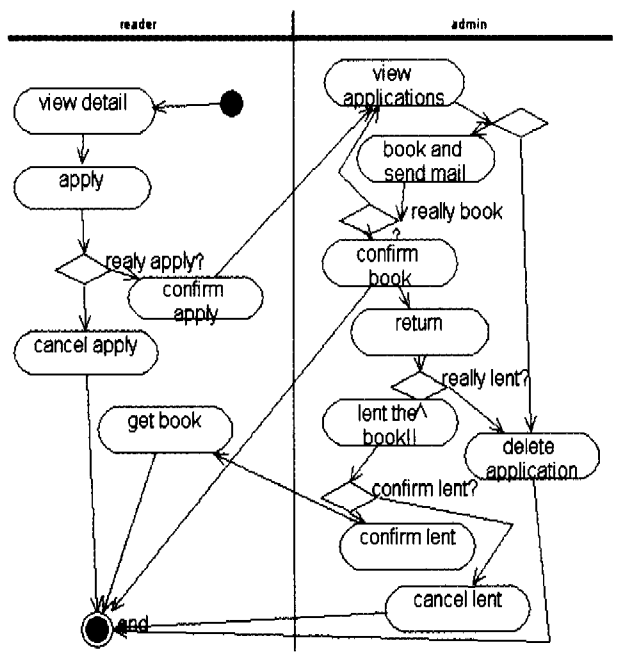

图 3 基于 Web 的图书借阅系统中图书借阅活动图

图 3 中，两个泳道（reader 和 admin）分别表示普通用户和管理员；普通用户的活动包括申请借书（apply），确认申请（confirm apply），取消申请（cancel apply）及借书成功（get book）等。而管理员的活动则包括预定并给用户发送邮件（book and send mail），同意借书（lent the book），确认借书（confirm lent）等。

图书借阅系统中一个典型的测试场景是用户登录，用户登录成功后才能进行各种操作。下面以其为例子，说明自动机的执行过程。

3.1 测试自动机执行实例

用户的登录过程可用表 1 的测试场景表示。

表 1 用户登录过程的一个测试场景

动作名称	角色	操作类型	预期操作结果
打开主页	Sail	Get(b)	result [1]
填写用户名和口令	Sail	fill((c)	result [2]
登录	Sail	post(a)	result [3]

表 1 中的操作结果规定了场景中每个动作的预期响应，见表 2。

表 2 表 1 中测试场景对应的预期操作结果

ID	Form	Link	Link caption	Form count	Link count
result [1]				0	0
result [2]	用户名，口令			2	0
result [3]		http://safejava:8080/lib/reader/mybook.jsp	个人图书	−1	6

表 2 中，Form 表示网页包含的指定名称的 Form，Link 表示包含指定地址的链接，Link caption 表示包含显示名称为指定字符串的链接，Form count 和 Link count 是页面包含的 Form 和 Link 的个数，−1 表示该次测试未关注该项目。从表 2 可以看到，result [2]表示当前网页包含两个 Form，其显示的名称分别为用户名和口令；result [3]表示当前页面包含六个链接，其中一个链接的显示名称为“个人图书”，地址为 http://safejava:8080/lib/reader/mybook.jsp 。下面介绍测试自动机对表 1 中测试场景的执行过程。

测试自动机的起始状态是(0,0)，堆栈 Z_0，“及时描述”为 $((0,0),bca,Z_0,)$，依据转移函数 $\delta((0,0),b,z_0)=\{((1,0),z_0)\}$，自动机的及时描述变为 $((1,0),ca,Z_0)$；自动机进行“get”操作来“打开主页”，状态变量 page 获得一个页面作为当前网页。接着自动机依据测试场景中给定的预期操作结果对该页面进行检测。若在该过程中访问不到页面，或者对页面的检测不通过，自动机认为出现异常，从而停止接受后续字符（操作）。

上述操作结果正确时，自动机接着处理字符‘c’，依据转移函数 $\delta((1,0),c,\beta)=\{((1,1),\beta)\}$，自动机的及时描述变为：$((1,1),a,Z_0)$；自动机依据测试用例中给定的参数组填写表单，状态变量 form 被赋值。这时自动机注视字符‘a’，依据转移函数 $\delta((1,1),a,\beta)=\{((1,0),A\beta)\}$，自动机将状态变量 page 的值压入堆栈，进行“post”操作——提交表单，状态变量 page 被赋值为 post 操作的响应网页，变量 form 被清空。如果这个过程中响应的网页正常，且通过了检测，则自动机完成接受字符串，此时状态为 $(1,0)\in F$，所以自动机接受字符串“bca”。

测试自动机处理这个测试场景的演变过程，用自动机的及时描述表示为表3。

表3 测试自动机的及时描述及采用的转移函数

及时描述	转移函数
$((0,0),bca,Z_0,)$	$\delta((0,0),b,Z_0)=\{((1,0),Z_0)\}$
$((1,0),ca,Z_0,)$	$\delta((1,0),c,\beta)=\{((1,1),\beta)\}$
$((1,1),a,Z_0,)$	$\delta((1,1),a,\beta)=\{((1,0),A\beta)\}$
$((1,0),\varepsilon,AZ_0,)$	

通过测试自动机，测试用例中定义的操作可被自动机接受并作用于被测软件，从而完成软件的测试执行，得到测试结果。

3.2 典型测试用例

上一节利用用户登录的例子，说明了测试自动机的执行过程。以下是上述Web借阅系统中图书借阅的典型测试用例。

表4 典型测试用例

活动	角色	动作	参数集	操作结果
view detail	sail	[get]		result[10]
apply	sail	[get]	(isbn, 781268sdjfka2)	result [5]
confirm apply	sail	[post]		result [6]
view application	admin	[get]		result [12]
book and send mail	admin	[get]	(isbn, 781268sdjfka2) (id, sail) (action, sendmail)	Result [14]
view application	admin	[get]		result [12]
book and send mail	admin	[get]	(isbn, 781268sdjfka2) (id, sail) (action, sendmail)	result [14]
view application	admin	[get]		result [12]
book and send mail	admin	[get]	(isbn, 781268sdjfka2) (id, sail) (action, sendmail)	result [14]
confirm book	admin	[post]		result [18]
teturn	admin	[get]		result [23]
lent the book	admin	[get]	(action, approval) (isbn, 781268sdjfka2) (id, sail)	result [21]
confirm lent	admin	[post]		result [25]
get book	sail	[get]		result [4]

操作结果格式同表2，具体内容限于篇幅不再赘述。该测试为测试自动机可以接受的成功测试用例。

3.3 失效报告

当操作结果同预期不一致或自动机因某些原因拒绝接受后续字符时，则可能出现一个失效，此时失效报告给出失效发生时的场景，也就是自动机当时的状态。表5为一测试失效现场。

表5 一个失效现场

活动	指向的链接地址/当前网页	操作结果
view detail	http://safejava:8080/lib/reader/detail.jsp? isbn=781268sdjfka2	result [10]
apply	http://safejava:8080/lib/reader/apply.jsp? isbn = 781268sdjfka2&-categoryid =201	result [5]
confirm apply	http://safejava:8080/lib/reader/apply.jsp	result [6]
view application	http://safejava: 8080/lib/admin/borrowrequest. jsp	result [12]
book and send mail	http://safejava: 8080/lib/admin/permitborow. jsp	result [14]
confirm book		result [19]
error pages	64. html	Err: check error

上述失效产生的原因是测试自动机没有实现对应用程序初始状态的自动复位，测试用例循环执行时，对同一本书第二次进行预定(book)，管理员进行确认时出现失效。表格最后一行的64. html指向一个本地保存的页面，该页面为失效发生时Web服务器返回的页面，表示失效现场。

3.4 测试集运行结果

测试结果信息包括测试基本信息和失效报告。测试基本信息包括：测试用例通过数，失效数及测试的起止时间。而失效报告指的是失效发生时自动机的及时描述，包括失效类型，如检测不通过或网络异常等；失效现场的网页快照，也就是当前网页的内容；堆栈中的历史网页，以备用户参考。下面是某测试集的运行结果实例。

表6 某测试集的运行结果

项目	值
测试场景	16
测试用例	22
测试执行时间	57秒
失效数目	5

具体的测试场景及测试用例，限于篇幅，无法一一赘述。对发生失效的测试用例，给出类似表5的失效现场。

上面只是给出了一些初步的测试结果信息，后续将在其基础上进行扩充，以提供更丰富的结果信息。

总结 本文基于下推自动机理论研究并提出了Web测试自动机，已实现的Web测试自动机原型支持Web测试用例的自动执行。测试用例中的每一操作被执行后，自动机记录对该操作的响应，并将结果同预期结果进行比较，从而判断该测试用例是否失效，并给出测试结果报告。测试自动机描述的测试用例执行过程非常易于理解。

当然，Web应用系统是非常复杂的软件系统，本文并未能够考虑其所有特征。并且，本文讨论的Web系统的操作类型及对操作结果的描述还不够丰富，但下推自动机良好的扩展能力及本文提出的基于下推自动机的灵活的体系结构使得进一步的扩展变得可行。如何支持更丰富的Web操作，对更精细的操作结果进行描述及提供更详尽的测试结果报告，都

是后续工作需要完善的。

参 考 文 献

1 Netmechanic. http://www.netmechanic.com
2 Web site garage. http://websitegarage.Netscape.com
3 Di Lucca G A, Fasolino A R, Faralli F, De Carlini U. Testing Web applications. In: Proceedings, International Conference on Software Maintenance, Oct. 2002. 310～319
4 Sun Microsystems. SunTest suite. http://www.sun.com/suntest
5 HtmlUnit: htmlunit.sourceforge.net
6 Rational robot. http://www.rational.com/products/robot/index.jsp
7 Sitetools. http://www.softlight.com/sitea/index.asp
8 Technovations load testing products. http://www.technovations.com/home.htm
9 Conallen J. Modeling Web Application Architectures with UML. Communications of the ACM, 1999,42(10)
10 Niese O, Margaria T, Steffen B. Automated Functional Testing of Web-Based Applications. QualityWeek Europe 2002. Brussels, Belgium. Applied computing. Nicosia, Cyprus. 2002. 1662～1669
11 Yang Ji-Tzay, Huang Jiun-Long, Wang Feng-Jian, et al. Constructing an Object-Oriented Architecture for Web Application Testing. Journal of Information Science and Engineering 18, 2002. 59～84
12 E-tester. http://www.rswsoftware.com/products/etester index.shtml
13 Silk test. http://www.segue.com/html/s solutions/silk/s family.htm
14 Watchfire enterprise solution. http://www.watchfire.com/solutions/wes.asp
15 VeriWeb: Automatically Testing. Dynamic Web Sites. Juliana Freire. http://www-db.bell-labs.com/juliana. Bell Labs
16 刘昶. 面向交互式软件的测试模型设计和测试用例生成:[北京航空航天大学硕士论文]. 2005,3
17 蒋宗礼, 姜守旭. 形式语言与自动机理论. 清华大学出版社, 2003. 235～257
18 van Rossum G. Python Programming Language. Python. Orgnization, 1990～2005. http://www.python.org
19 Lee J J. Python Bits. SourceForge. 2004.5. http://wwwsearch.sourceforge.net
20 Richardson L. We called him Tortoise because he taughtus. crummy dot com. 2004.10. leonardr@segfault.org

(上接第 268 页)

图中,属于较大的。从开始发现算法到输出抽象后的序列图,用时在 1～2s 之间,时间等待完全在可接受的范围内。

对于消息层次关系的发现和抽象,也进行了相关实验。以 2523 号进程内交互序列图为示例,由于篇幅所限,我们只给出了抽象后的最终结果,如图 6 所示。

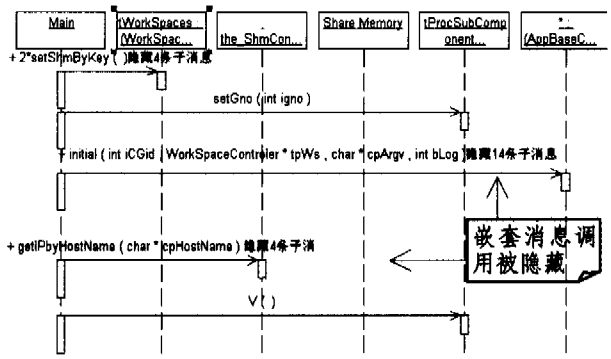

图 6 层次抽象后的 2523 序列图(局部)

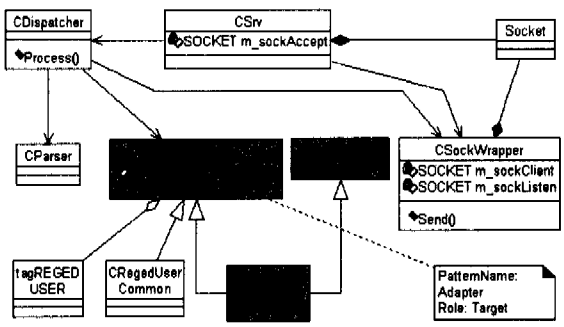

图 7 注册服务系统的服务器端发现 Adapter 模式

图 7 是一个设计模式自动发现的例子,目标系统是一个以客户端/服务器模式进行注册服务的系统。我们找到了 Adapter 模式的一个实例,为了方便用户观察,系统重新绘制类图,通过着色处理把模式突出显示出来,并对模式中的核心类加上 note 标记说明。

结论 作为逆向工程工具 XDRE 系统的一个功能模块,本文提出并实现了交互重复模式的自动发现和交互层次的自动恢复方法,以及基于类图的设计模式发现,并实现了以此为依据对动态剧情的抽象,为用户提供了在序列图上交互地进行剧情抽象的支持。用户可以从系统较低层的行为构造更高设计层的行为模型,在不同的抽象层次上观察系统的动态行为,验证、更新最初的设计模型,认定需要再工程的部分。从使用者的角度来看,该功能模块的时间迟延性、正确性和可用性等都达到了要求。

在已完成的工作基础上,今后的工作将集中在通过对系统的行为特征分析,识别设计模式,使得系统具有自动发现动态剧情中潜在设计模式的能力。

参 考 文 献

1 Ernst M D. Static and Dynamic Analysis: Synery and duality. In: ICSE Workshop on Dynamic Analysis, (Portland, OR), May 9, 2003. 24～27
2 Richner T, Ducasse S. Recovering high level views of object-oriented applications from static and dynamic information. In: Proceedings of International Conference on Software Maintenance, Oxford, UK, IEEE CS Press, 1999. 13～22
3 Systä T. Static and Dynamic Reverse Engineering Techniques for Java Software Systems: [Ph D Dissertation]. Dept of Computer and Information Sciences, Unversity of Tampere, May 2000
4 李青山, 陈平. 利用改进遗传算法恢复高层架构. 软件学报, 2003,14(7):1221～1228
5 李凡, 陈平. 逆向工程中 UML 序列图的自动生成与抽象技术. 计算机科学, 2004(12)
6 陈平. C3I 系统应用软件逆向工程开发工具研究任务申请书. 本项目课题组, 2001
7 张广红. UML 序列图的逆向自动生成与剧情抽象:[硕士论文]. 西安:西安电子科技大学, 2003
8 Kramer C. Design recovery by automated search for structural design patterns in object-oriented software. In: Proceedings of the Third Working Conference on Reverse Engineering, 1996. 208～221
9 陈平. iCALL 呼叫中心应用支撑平台低层通信模型设计文档. 西安电子科技大学软件工程研究所, 2000