模型驱动的EJB构件测试建模研究

邓 雄, 常创业, 吴 际, 金茂忠, 刘 超
(北京航空航天大学计算机学院, 北京 100083)

摘 要: 基于对Web系统的特点研究, 提出了一个全面测试Web系统的框架. 并基于模型驱动的软件测试思想(MDT), 重点研究了此框架中的中间业务逻辑层的测试建模方法: 定义了一个平台相关的测试模型(PSTM) ——EJB构件测试模型; 并给出了对它的模型复原算法和模型一致性检测方法. 该测试建模方法, 从软件构件的层次, 实现了EJB静态结构和动态行为的建模, 并通过模型一致性检测实现EJB代码的潜在缺陷检查. 这一测试模型将为测试用例和测试数据的生成提供有力的支持.

关键词: EJB; 构件测试; 测试建模; 模型驱动测试; 构件识别; 平台相关测试模型; 模型复原; 模型检查
中图分类号: TN311.5 文献标识码: A 文章编号: 0372-2112 (2006) 12A-2467-06

Research on Model Driven Test Modeling of EJB Component

DENG Xiong, CHANG Chuang-ye, WU Ji, JIN Mao-zhong, LIU Chao
(*School of Computer Science and Engineering, Beihang University, Beijing* 100083, *China*)

Abstract: Based on the study on Web-based systems, this paper presents a systematic testing framework for Web-based systems. And focused on test modeling of business logic layers of Web-based systems, a EJB test model, one kind of platform specific test models(PSTM) according to model driven testing(MDT), is defined, as well as the approach of its model recovery and model checking are provided. The test modeling deals with the static structures and dynamic behaviors of EJB from the component level and describes a mechanism of code failure detection based on its model checking, which will provide a good support for test cases and test data generation.

Key words: enterprise JavaBean (EJB); component testing; test modeling; model driven testing (MDT); platform specific testmModel (PSTM); component identification; model recovery; model checking

1 引言

随着互联网技术的飞速发展, 基于互联网的各种应用应运而生. 由于Web系统的分布、动态、异构、交互式的特性, 较好的满足了全球范围内的信息共享, 而使得Web系统在各行各业发挥越来越关键和重要的作用, 但是有报告显示实际的Web系统的质量并不能令人满意[1], 这样对于Web系统的质量保障就自然被提上关键议事日程.

由于Web系统具有分层体系结构, 包含大量构件和框架[16], Web的开发也经常受到频繁变更的需求、短时限等因素的制约, 使得Web系统测试成为艰巨的任务. 而Web相关的研究工作目前主要集中在Web site的设计和开发上, 例如Web site开发中的多语言、开发模型、图形表示的研究[2,3]. 对于Web系统测试的研究却不多: 部分工具通过捕捉/回放的方式支持Web系统的功能测试[15]; 少数的研究集中在Web系统的结构测试上[6,8]; 文献[9, 10]中利用UML或者扩展的UML对Web系统进行测试建模; Liu[11,12]等扩展了传统的数据流测试技术来测试Web系统; kallepalli[13]及Tonella[14]等人分别从统计的角度研究Web系统测试.

我们针对Web系统测试的诸多开放问题之一——测试建模Web系统展开工作, 首先研究了J2EE为代表的Web系统, 提出了一个Web系统测试的框架, 然后给出了一个用于模型驱动测试的平台相关测试模型——EJB测试模型及其建模方法, 最后给出了以后研究的展望.

2 Web系统、EJB、模型驱动方法概述

2.1 Web体系架构及测试概述

Web系统是以分布式计算、多层应用模型和中间件等技术为基础的多层体系结构. 以下以J2EE为例说明Web系统的体系结构, 如图1.

Web系统在逻辑上分为3个部分: Client、Server、Enterprise Information System(EIS). Client是Web系统的发起方, 典型的代

收稿日期: 2006-09-07; 修回日期: 2006-12-18
基金项目: 国家863高技术研究发展计划重点项目(No.2004AA119030)

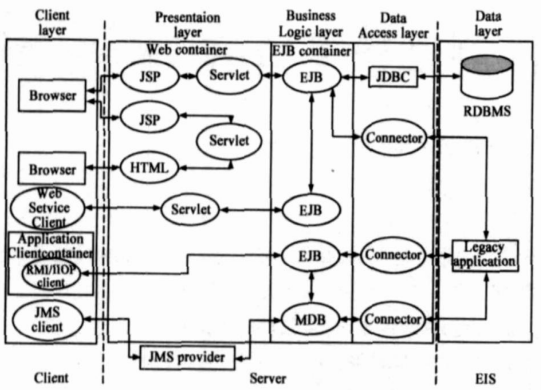

图 1 J2EE 框架示例

表是基于浏览器的 Web 页面访问; Server 是一个 Web 系统的实现者和响应者, J2EE 利用 JSP、Servlet、EJB、JDBC 等技术实现完整的 Web 系统; EIS 是 Web 系统中数据的集散地, 目前主要是利用关系数据库存储各种数据. 为实现 Web 系统, Server 的逻辑又可以细分为 3 层: (1) 表示层: 处理用户请求, 格式化数据信息; (2) 业务逻辑层: 实现领域相关的业务逻辑, 事务管理等, 具体的功能一般由运行于 EJB 容器中的 EJB 构件实现; (3) 数据访问层: 持久化业务对象, 一般由 JDBC 实现数据从数据库中读出或向数据库中写入.

2.2 EJB 概述

EJB 是运行在服务器端的一种构件, 由容器(EJB Containrer) 来管理. 根据规范[9], EJB 的实现具有严格的要求, 以实体 Bean 为例. 实现一个实体 Bean, 至少需要具备两个接口(Home 接口、Object 接口), 一个 Bean class、一个可选的主键类和一段 XML 部署描述.

(1) Home 接口必须直接或者间接继承自 javax. ejb. EJBHome 或 javax. ejb. EJBLocalHome. 接口中只能声明以下形式的方法: createXXX( )、findXXX( ) 和业务逻辑方法, 并且至少声明 createXXX( ) 和 findXXX( ) 中的一种.

(2) Object 接口必须直接或者间接继承自 javax. ejb. EJBObject 或 javax. ejb. EJBLocalObject. 接口中声明了自定义的业务逻辑方法, 并将在 Bean class 中实现.

(3) Bean class 必须要实现 javax. ejb. EntityBean 接口以及声明的 7 个回调方法: setEntityContext( )、unsetEntityContext( )、ejbActivate( )、ejbPassivate( )、ejbLoad( )、ejbStore( )、ejbRemove( ). Bean class 还必须实现与相应的 Home 接口和 Object 接口中声明的所有方法.

(4) 必须利用部署描述符定义 EJB 的部署信息: 包括 EJB 的内部结构和它的外部依赖性(例如 Home 和 Object 接口); EJB 之间的关联; 以及安全和事务属性等.

上述 Home 接口、Object 接口、Bean 类和主键类为单一的 Java 类或接口, 我们将其称为构件核.

2.3 模型驱动的测试方法概述

软件开发思想的演进经历了从毫无模型定义的单一编码(code only) 阶段、改进建模符号提供代码可视化(code visualization) 的代码模型阶段、描述系统的架构和设计与代码模型之间双向交互的双向工程(round-trip engineering) 几个阶段. 随着对软件模型的理解的加深, 以模型为中心的开发方法(model only) 将顺应趋势成为软件开发的主流. 在这一方法中, 模型成为关注的焦点, 软件架构与业务逻辑相分离, 可以灵活的定义和演进, 功能构件也能够重用. 在这样的形势下, MDA 应运而生.

模型驱动测试(MDT) 将这一思想应用于软件测试中, 使得软件测试以平台无关测试模型(PITM)、平台相关测试模型(PSTM)、测试代码生成(TCG, Test Code Generation) 等基本要素为核心, 实现测试需求、测试用例、测试数据、测试运行的描述, 以及 SUT 与测试模型的自动转换等, 如图 2[18]. U2TP[20] 是扩展自 UML2.0 描述测试模型的领域无关的标准建模语言.

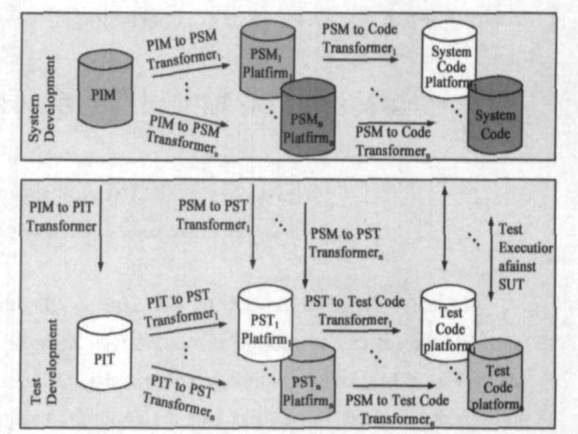

图 2 MDA & MDT

3 Web 系统测试框架

我们认为对于 Web 系统的测试和分析, 至少需要: (1) 测试表示层的页面: 包括链接检查, 页面内容测试, 多语言测试, cookie 过期检查等; (2) 中间业务逻辑层的测试及分析: 结构分析、内部和外部的交互关系分析, 业务功能测试等; (3) 数据测试: 对数据访问(读写等) 功能测试、数据的一致性验证等; (4) 多层次的组合功能测试和结构分析.

由此, 我们可以给出一个相应的系统的测试方案:

(1) 测试表示层页面: 由于表示层是 Server 与 Client 的接口层, 我们应该从两方面对它测试. 首先, 从 Client 端, 即从 Web 系统使用者的角度分析最终的 HTML 页面, 验证页面的内容, 语言和 Cookie 等. 这种方法比较直观, 适于功能测试, 其局限在于只能看到最终产生的页面, 对于动态脚本文件的分析无能为力, 这种分析更适于功能测试页面的表单, 表格, 语言等页面内容; 其次, 白盒的分析表示层的页面代码结构, 这种方法能较全面的了解 Web 构件的构成及交互, 但由于脚本语言的多样性给分析带来巨大的不便, 同时由于缺乏 Client 端的支持而使得测试用例的执行和结果验证十分困难. 因此, 我们首先从 Server 端, 尽可能复原代码结构, 根据代码的覆盖率指导测试场景生成; 然后从 Client 端, 利用对页面的不同参数输入和操作, 实例化测试场景为不同的测试用例, 并执行用例以验证结果.

(2) 中间业务逻辑层: 由于涉及复杂的业务逻辑, 需要考

虑性能及功能两方面的测试. 首先白盒分析中间层代码入手, 获取代码静态结构和动态行为: 静态结构可以用 UML 类图、构件图描述, 动态行为则以 UML 状态图、顺序图、活动图表示. 代码结构的分析保证测试充分性(例如各种覆盖) 和指导测试用例的生成(例如单元测试和集成测试用例), 动态行为的分析实现运行性能的监视和追踪.

(3) 数据测试: 由于一般的 Web 系统的数据的存取由专门的程序库(例如 JDBC) 实现, 对它们的测试不在我们的考虑范畴内, 我们更希望利用对数据库中数据的分析来辅助 Web 系统的功能测试. 从 3 方面考虑: (*a*) 实现数据库测试状态的可视化察看和记录; (*b*) 辅助测试数据的生成: 利用数据库的当前及历史状态信息, 给功能测试的测试数据的生成一定指导(例如以数据库中已经存在的 Web 用户作为添加 Web 用户操作的测试数据) ; (*c*) 增强 Web 系统期望结果的验证能力, 根据数据库状态的分析, 辅助验证期望结果(例如删除 Web 用户测试用例运行后, 察看数据库中是否仍然存在该用户的信息来验证删除操作是否正确).

(4) 多层次的组合功能测试及结构分析: 以各层之间的接口交互分析为基础实现测试的组合和多层次运行过程监控和追踪.

本文的研究目的就是利用模型驱动测试的方法, 对基于 EJB 实现的中间业务逻辑建模, 以指导测试用例的开发、测试数据的生成、测试结果的验证.

4　EJB 构件的测试建模

EJB 是一个用于开发和部署服务端应用的领域相关的构件模型规范, 它简化了中间件的开发, 方便的支持具有可伸缩(scalable)、多用户的企业级应用, 支持事物处理、数据库连接和构件定制等功能[19]. 由于开发模型的着眼点在于指导软件开发, 例如代码生成, 而测试模型的目的是为了指导软件测试, 例如测试用例、测试数据的生成、覆盖率的度量、性能跟踪等, 二者虽然是对同一事物的描述, 但是由于描述的不同目的和角度, 而使得二者不能等同. 如果将 EJB 的构件模型直接应用于 EJB 应用的测试, 对于软件测试的帮助并不大, 因此实现 EJB 的测试建模非常必要, 而目前对于 EJB 的研究却主要集中在对开发模型本身的评估[4, 5] 和应用[21, 22] 上. 另外, 由于 PSM 的大量存在和应用的广泛(例如 COBAR、DCOM/ COM + 等), 对相应的 PSTM 的研究也就显得十分必要.

在我们的研究中, 定义 EJB 测试模型的目的有如下三方面: (1) EJB 代码分析, 理解 EJB 静态结构和动态行为, 进而帮助理解 Web 系统业务逻辑, 支持测试用例和测试数据建模, 实现程序度量; (2) 追踪和监控 EJB 的运行轨迹, 进而了解程序的运行性能, 并辅助提高测试充分性. (3) 测试模型一致性检测, 发现 EJB 实现中的潜在代码失效.

4.1　EJB 测试模型

EJB 规范中明确定义了 EJB 的组成及实现细节, 但在实际开发中, 根据需要 EJB 的代码实现却有一些细节变动. 例如 Home 接口实现往往不是直接继承自 javax. ejb. EJBHome[17], 而是通过将此标准接口置于一个继承链的顶端实现; 又如, 在 EJB 构件之间往往存在一些粘合类[22] 或基础设施类, 也应该属于测试的范畴; 再如, 基于不同部署描述, 相同的 EJB 代码对外的实际功能表现不同, 这样可能存在“冗余” 或“缺失” 的 EJB 代码实现. 此外, 我们还需要了解 EJB 之间的交互关系等动态信息. 为此, 我们给出了一个从结构和行为两方面描述的 EJB 构件的测试模型, 如图 3~ 8 以 Entity Bean 为例描述了这一测试元模型.

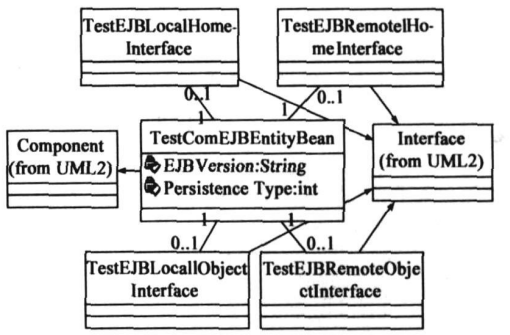

图 3　Entity Bean 测试元模型

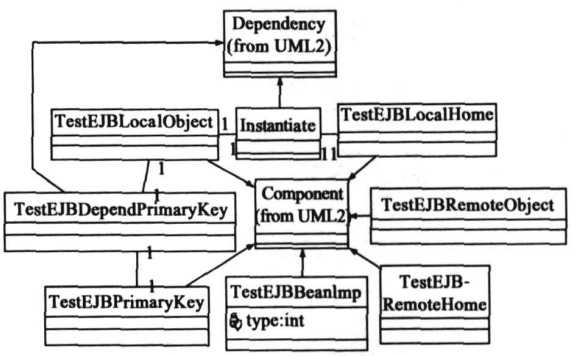

图 4　Entity Bean 测试元模型

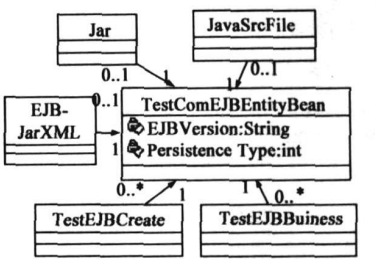

图 5　Entity Bean 测试元模型

静态结构上, EntityBean 测试构件(TestComEJBEntityBean) 由 Home 构件(TestEJBHome)、Object 构件(TestEJBObject)、Bean 实现构件(TestBeanImp) 和可选的主键构件(TestEJBPrimaryKey) 组成; TestComEJBEntityBean 提供了对外的访问接口, 分别是 TestEJBRemoteHomeInterface、TestEJBRemoteObjectInterface、TestEJBLocalHomeInteface、TestEJBLocalObjectInterface; TestComEJBEntityBean 具有两类操作: create 方法(TestEJBCreate) 和业务逻辑方法(TestEJBBusiness); EJB 的测试构件模型必须与 EJB 实现的源代码或者部署描述文件对应, 因此 TestComEJBEntityBean 包含了 JarSrcFile、Jar、EJB-JarXML 等 Artifact. EJB 之间具有两种交互关系: 调用关系(TestEJBReference) 和消息驱动关系

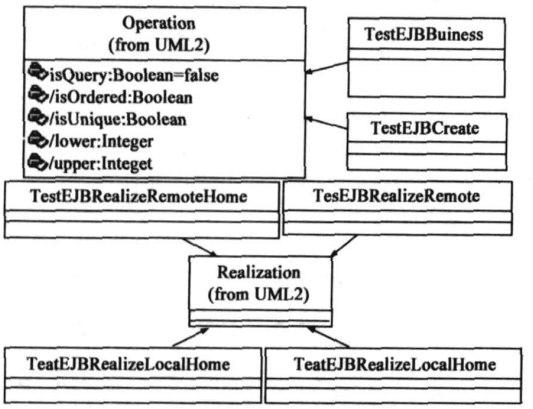

图 6 Entity Bean 测试元模型

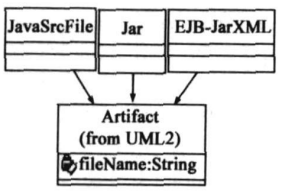

图 7 Entity Bean 测试元模型

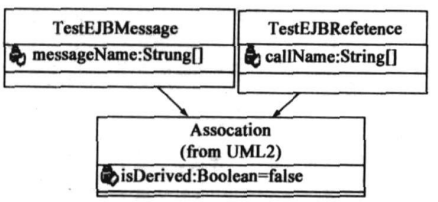

图 8 Entity Bean 测试元模型

(TestEJBMessage).

动态行为上,我们利用了 EJB 固有的生存周期(例如 BMP 的生存状态周期图)和执行的顺序图(例如状态 Bean 的顺序图)来追踪和监视 EJB 的执行过程.

上述测试元模型中的 TestEJBRemoteHomeInterface、TestEJBRemoteObjectInterface、TestEJBLocalHomeInterface、TestEJBLocalObjectInterface 等部分是在 EJB 构件核的基础上,集成基础设施类、粘合类、相应的父类或父接口等构成的子构件.由此,我们提供一个较好的解决前述种种问题的 EJB 构件测试模型.

4.2 EJB 测试模型识别

EJB 构件测试模型的识别是指从 EJB 的实现文件中逆向发现潜在的 EJB 构件测试模型的过程. EJB 的实现文件包括独立的两部分:源代码和部署文件. 表 1~2 给出了从源代码中发现测试模型的算法,以 BMP 为例.(以下的 Bean Class、Home 接口、Object 接口、主键类等均指构件核)

表 1 BMP 静态结构复原算法

(1) 解析源代码查找实现了 javax.ejb.EntityBean 的类或其子类,如果存在则表明它可能是一个 BMP 的 Bean Class,否则退出.
(2) 分析该 Bean Class,验证其是否实现了七种回调方法和 ejbFindByPrimaryKey() 方法,其中,ejbCreateXXX() 可以被 ejbFindXXX() 方法取代. 这八种方法必须在该类或其父类中实现,如果缺少一种则退出.
(3) 根据 ejbCreateXXX() 或 ejbFindXXX() 的方法名,遍历源代码查找声明了 createXXX() 方法或 findXXX() 方法,并且继承了 javax.ejb.EJBHome(或 EJBLocalHome)的接口. 如果存在,则说明找到了该 BMP 的 Home 接口. 如果未能找到,则退出.
(4) 在上述 Home 接口中,检查 createXXX() 方法和 findByPrimaryKey() 方法的返回值类型是否一致,不一致则退出,一致则记录此类型名. 接着遍历源代码查找此类型是否为继承了 javax.ejb.EJBObject(或 EJBLocalObject)的接口. 如果是,则说明我们找到了该 BMP 的 Object 接口,否则退出.
(5) 在上述 Home 接口中,记录 findByPrimaryKey() 方法的参数的类型,遍历源代码去查找此类型是否为实现了 java.io.Serializable 的类. 如果是,则认为找到了主键类,否则退出.
(6) 如果 Bean Class、Home 接口、Object 接口、主键类都已经成功地被识别出来,还必须确保 Home 接口、Object 接口和主键类中声明的方法都在 Bean Class 实现,否则退出.
(7) 为了复原成前述的测试模型,还需要继续将与各个构件核关联的基础设施类,粘合类,父类,其他实现的接口等组合起来,以共同构成的相应的子构件(见 4.1).

表 2 EJB 构件交互关系的复原算法
(前提是我们已获取了所有 EJB 的信息)

(1) 进入给定的 EJB_B 的代码中,查找并记录 EJB_A 的 Home 接口和 Object 接口出现的所有语句,分别记为 SH 和 SO
(2) 层次遍历所有 SH 语句中调用的方法,如果调用了 lookup() 方法或者 PortableRemoteObject.narrow() 方法,记顶层方法返回变量为 V.
(3) 查看 V 是否调用了 EJB_A 的 Home 接口中定义的 createXXX() 或 findByPrimaryKey() 方法,如果否,则退出.
(4) 层次遍历所有 SO 语句中调用的方法,如果有一处调用了 EJB_A 的 business 方法,则认为在 EJB_B 中调用了 EJB_A. 如果没有找到,则退出.

4.3 EJB 测试模型的一致性检测方法

(1) 对于实体 bean 我们可以检查它的持久化域是否与程序中定义的相对应.

源代码:

```
……
public abstract String getNo();
……
```

部署描述:

```
< cmp-field>
  < field-name> no< /field-name>
< /cmp-field>
```

(2) 对于实体 bean 和会话 bean,我们通过 ejb 之间的调用

关系来进行一致性检查.

源代码:

```
Context ctx= new InitialContext( ) ;
Object
result= ctx. . lookup(" java: comp/ env/ ejb/ Example") ;
ExampleHome home= (ExampleHome) result;
Example object= home. create( ) ;
```

部署描述:

```
< ejb- ref- name> ejb/ Example< ejb- ref- name>
< home> ExampleHome< / home>
< remote> Example< / remote>
```

(3) 对于各种 ejb, 我们还可以从他们所用到的资源属性进行一致性检查.

源代码:

```
Context ctx= new InitialContext( ) ;
javax. sql. DataSouce
ds= ctx. lookup(" javax: comp/ env/ jdbc/ ejbPool") ;
```

部署描述:

```
< res- ref>
< res- ref- name> ejbPool< / res- ref- name>
  < res- ref- type> javax. sql. DataSouce< / res- ref- type>
< / res- ref>
```

(4) 对于各种 ejb, 我们还可以从他们所用到的环境变量进行一致性检查.

源代码:

```
Context ctx= new InitialContext( ) ;
String
tax= ( String) ctx. . lookup(" java: comp/ env/ Example/ r") ;
```

部署描述:

```
< env- entry>
  < env- entry- name> Example/ algor< / env- entry- name>
  < env- entry- type> java. lang/ String< / env- entry- type>
< / env- entry>
```

5　结论及展望

鉴于 Web 系统的广泛应用, 研究了 Web 系统的特点, 提出了一个 Web 系统测试的框架, 并基于模型驱动的测试方法, 实现了 EJB 构件的测试模型定义和识别. 由于 EJB 的实现具有一定的灵活性, 下一步的工作, 将进一步提高识别算法的识别率; 并研究基于测试模型的测试用例和测试数据的生成; 并考虑建立相关的测试充分性准则来衡量测试的充分性.

参考文献:

[1] Elbaum S, Rothermel G, Karre S, et al. Leveraging user- session data to support Web application testing[J]. IEEE Transactions on Software Engineering, 2005, 31(3): 187- 202.

[2] T Isakowitz, E A Stohr, P Balasubramanian. RMM: a methodology for structured hypermedia design[J]. Comm of the ACM, 1995, 38(8): 34- 44.

[3] J Conallen. Building Web Applications with UML[M]. Beijing: Science Press, 2003.

[4] A Liu, L Bass, M Klein. Analyzing Enterprise Java Beans Systems using Quality Attribute Design Primitives[R]. Pennsylvania USA: Carnegie Mellon University, 2001.

[5] Yan (Jenny) Liu, Ian Gorton, Anna Liu, Shiping Chen. Evalurating the scalability of enterprise javaBeans technology[A]. Proc of Int Symp on Software Engineering Conference, 2002[C]. NW Washington, DC USA: IEEE Computer Society, 2002. 74- 83.

[6] C- H Liu, D C Kung, P Hsia, C- T Hsu. Structural testing of Web applications[A]. Proc of Int Symp on Software Reliability Engineering, 2000[C]. California: IEEE Computer Society, 2000. 84- 93.

[7] Di Lucca G A, Fasolino A R, Faralli F, et al. Testing Web applications[A]. Proc of the International Conference on Software Maintenance[C]. Washington, DC, USA: IEEE Computer Society, 2002. 310- 319.

[8] Ricca F, Tonella P. Analysis and testing of Web applications[A]. Proc of International Conference on Software Engineering[C]. NW Washington, DC USA: IEEE Computer Society, 2001. 25- 34.

[9] Conallen J. Modeling Web application architectures with UML[J]. Communications of the ACM, 1999, 42(10): 63- 70.

[10] Di Lucca G A, Penta M D. Considering brower interaction in web application testing[A]. Proc of Web Site Evolution[C]. NW Washington, DC USA: IEEE Computer Society, 2003. 74 - 81.

[11] Kung D C, Liu C H, et al. An object- oriented Web test model for testing Web applications[A]. 24th International Computer Software and Applications Conference[C]. Washington DC, USA: IEEE Computer Society, 2000. 537- 542.

[12] Liu C H, Kung D C, et al. An object- based data flow testing approach for Web applications[J]. International Journal of Software Engineering and Knowledge Engineering, 2001, 11(2): 157- 179.

[13] Kallepli C, Tian J. Measuring and modeling usage and reliability for statistical Web testing[J]. IEEE Transactions on Software Engineering, 2001, 27(11): 1023- 1036.

[14] Tonella P, Ricca F. Statistical testing of Web applications[J]. Journal of Software Maintenance and Evolotion, 2004, 16(1- 2): 103- 127.

[15] IJ Nino et al. Carena: a tool to capture and replay Web navigation sessions[A]. Proc of End- to- End Monitoring Techniques and Services, 2005[C]. Washington DC, USA: IEEE Computer Society, 2005: 127- 141.

[16] 程洪等. 基于 J2EE 体系的 Web 应用框架整合[J]. 计算机工程, 2005, 31(20): 96- 98.
Cheng H, et al. Web application framework composition based

on J2EE[J]. Computer Engineering, 2005, 31(20): 96- 98. (in chinese)

[17] ObjectWeb Forge. Lobomz plugúin[EB/OL]. http://forge.objectweb.org/projects/lomboz, 2006-12-15.

[18] Ina SchieferdeckeIna. TTCN-3 Test Generation from System Models[EB/OL]. http://www.win.tue.nl/ipa/activities/springdays2006/Schieferdeckerslides.pdf, 2006-12-15.

[19] Sun Microsystems. Java 2 Enterprise Edition Platform White Papers[EB/OL]. http://java.sun.com/j2ee/white/index.html, 2006-12-15.

[20] FOKUS. U2TP[EB/OL]. http://www.fokus.gmd.de/u2tp. 2006-12-15.

[21] Y Liu, H C Cunningham. Mapping component specifications to enterprise JavaBeans implementations[A]. Proc of Proceedings of the ACM Southeast Conference[C]. Alabama, USA: ACM Press, 2004. 177- 181.

[22] Steffen Goebel, Michael Nestler. Composite component support for EJB[A]. Proceedings of the winter international symposium on Information and communication technologies[C]. Cancun, Mexiko: Trinity College Dublin, 2004. 1- 6.

作者简介:

邓 雄 男, 1981 年生于四川巴中, 北京航空航天大学计算机科学与工程学院软件工程研究所博士生. 研究方向为程序分析、软件工程、软件测试、搜索引擎技术.
E-mail: Johnny_dx@126.com

常创业 男, 1983 年生于陕西, 硕士研究生, 研究方向为软件测试.

刘 超 男, 教授, 博导, 主要研究方向为软件工程、面向对象技术.

吴 际 男, 讲师, 主要研究方向为软件测试.

金茂忠 男, 教授, 博导, 主要研究方向为软件工程、软件质量保证、编译技术.