维普资讯 http://www.cqvip.com

一种基于 Web 软件集成测试的建模方法*)

胡 蓉 缪淮扣 刘焕洲

(上海大学计算机工程与科学学院 上海 200072)

摘 要 本文给出了一种 Web 软件集成测试的建模方法。该方法通过分析 Web 应用程序体系结构,对 Web 应用划分,用分层的有限状态机对 Web 应用进行行为建模,通过采用基于有限状态机的导航模型来指导测试人员进行 Web 集成测试;采用 UML 扩展的模型对 Web 应用组件间的交互建模,通过建立基于 UML 的组件依赖模型并提供相应的测试用例生成规则来提取 Web 应用的测试用例。

关键词 Web 建模,UML 扩展,测试模型,有限状态机

An Approach to Modeling Web Applications Integration Testing

HU Rong MIAO Huai-Kou LIU Huan-Zhou

(School of Computer Engineering and Science of Shanghai University, Shanghai 200072)

Abstract This paper presents an approach to modeling Web applications based on integration testing. By analyzing the architecture of Web application, we partition it by using layered Finite State Machines and model the behaviors of a Web application, and then direct testers to perform web application integration testing by adopting navigation model based on Finite State Machines. By extending UML this paper proposes a component dependency model to model the interaction of web components, and then provides corresponding criteria for generating test case.

Keywords Web modeling, UML extension, Test model, Finite state machines

1 引言

Web 应用的巨大成功和不断发展,使其渗透到商业领域和个人生活的各个方面。在市场需求和 Internet 技术进步的不断推动下,Web 应用日益增加,软件规模不断扩大,复杂性增加,因此软件质量越来越成为人们关注的问题。作为保证 Web 软件质量和可靠性的重要手段,软件测试成为 Web 软件开发过程中的重要环节。

Web 软件的主要特征是封装性、松散耦合性、跨组织和跨平台性。这些特征使得传统的软件测试方法和技术很难实现对 Web 应用系统的测试。传统软件测试方法对于 Web 应用软件的多层次特点、基于超链接的程序结构以及事件驱动等特点缺乏有效的解决方法。目前针对 Web 软件的测试模型、测试策略、测试层次和测试过程方面的研究工作刚刚起步,许多问题仍需进一步研究和探讨。例如 Rica 和 Tonella[5]建议为 Web 应用提供分析与测试策略的模型,这些策略主要是建立在对静态页面的分析以及一些初步的动态分析。随着越来越多的信息通过 Web 应用来进行动态分析,它们的重点应该转到 Web 应用的动态行为方面。Anneliese Andrews, Jeff Offutt 和 Roger Alexandert[10]等人分析构成 Web 应用的网页和软件构件之间 8 种连接关系,提出了一种基于有限状态机(FSM)的 Web 应用建模和测试用例生成方法。该方法通过对 Web 应用进行功能簇和逻辑网页的划分并用带约束的分层 FSM 来表示逻辑网页及逻辑页面的导航关系。这种方法没有进一步考虑 Web 应用中软件构件的交互和合成的测试问题。Conallen 在文[6]中建议使用扩展 UML 进行复杂建模,并建立了标准建模结构与 Web 应用制品之间的映射关系,然而在他的研究工作中并没有考虑到测试工作。Kung[7]提出了一种描述 Web 站点的图形模型,并在这些图形模型基础上提出了一些 Web 测试的基础定义,但没有考虑到服务器端 Web 的动态行为。

总的说来,现有的 Web 软件的测试研究只是分别考虑 Web 软件检验和测试的一个或几个方面,没有关注面向 Web 软件整体的建模以及基于 Web 软件的模型产生测试用例的方法。这方面的研究还是比较初步的。

本文从 Web 应用的行为和功能建模入手,通过对 Web 应用的功能簇的划分,用分层的有限状态机导航模型来对 Web 应用的各组成部分进行行为建模;在簇的划分层面上,对 Web 应用中组件的交互建模,建立基于 UML 的组件依赖模型,并提供相应的测试用例生成规则来提取测试用例。本文把对 Web 应用的行为和功能进行测试而建立的模型称为测试模型,把建立模型这个过程称为测试建模。测试建模方法通过一个常见的新闻发布网站中的后台管理子系统实例进行阐述。

2 Web 应用概述

2.1 Web 应用体系结构

对于基于 Web 的应用系统,用户直接面对的是客户端浏览器。用户在使用系统时,请求之后的事务逻辑处理和数据的逻辑运算由服务器与数据库系统共同完成,对用户而言是完全透明的。运算后得到的结果再通过浏览器的方式返回给用户。这个过程可分成一些子步骤,每一个子步骤的完成可理解为通过一个单独的应用服务器来处理,这些应用服务器在最终得到用户所需的结论之前,相互之间还会进行一定的

*)本文的研究工作受国家自然科学基金(60373072,60673115)和上海市教委科技发展基金资助。胡 蓉 硕士,主要研究方向:软件工程、软件测试;缪淮扣 教授,博导,主要研究方向:软件工程、软件形式方法、自动推理;刘焕洲 硕士,软件测试。

维普资讯 http://www.cqvip.com

数据交流和传递。图1是Web的应用体系结构简图。

Web应用是用来完成某个事物的一系列相关Web页面和其他资料的集合，页面之间的联系通过链接和组件实现。从面向对象方法学的角度看，Web应用中的每个服务器页面对应于一个类，每个客户端页面可以看作是一个对象。客户端页面对象是页面属性和方法的组成体，其属性一般由会话变量表示，方法则是通过服务器页面或组件中的函数来实现，对象的状态则由属性决定。值得注意的是，一个服务器页面可以根据不同的请求生成多个具有不同动态行为的客户端页面实体，所以这些客户端页面可视为不同于其对应的服务器页面的独立活动对象。

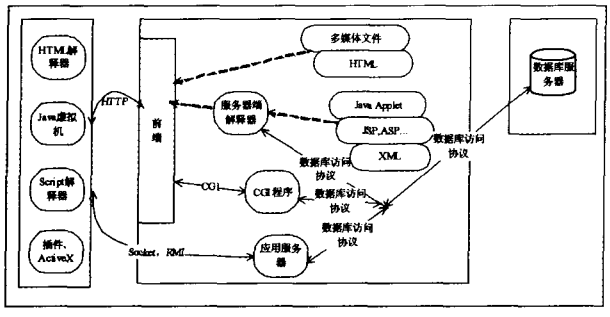

图1 Web体系结构图

2.2 Web应用的特点

Web页面是一个可以在单个浏览器窗口查看的信息体，可以作为静态HTML文件存储，也可以由软件动态生成，如JSP或Java Servlet。一个Web站点是Web页面和连接的组件的集合，这些组件通过链接和其它控制机制与Web页面相关联。Web应用是在一个或者多个Web服务器上运行的程序，通过Web站点，Web应用可以被用户使用。

Web应用的特点体现在它的动态性和异构性，这种特点是影响Web应用一个最重要的因素。本文采用Jeff Offutt[10]等人给出的Web应用中不同成分的连接方式来分类，主要针对静态链接(HTML－〉HTML)、动态链接(HTML－〉software)、动态建立的HTML(software－〉HTML)、软件连接(connections)、站外软件构件连接(connections)和动态连接(connections)几个分类方式来分析Web应用。

3 Web测试建模

3.1 基本概念

本节将从测试建模的角度定义建模过程中的一些基本概念。

Web页面：从Web系统的表现形式看，任何一个Web应用都是由一定数量的Web页面组成。Web页面包括静态页面和动态页面，页面之间存在导航关系。从细粒度层次看，Web页面包含很多内部组件，比如文本、图片、表单、文本框、多媒体对象、脚本及applets等。其中脚本和applets表示页面的活动组件，因为它们执行一些跟Web应用行为相关的动作。在这个抽象层次上，服务器页面关联了很多Web组件，通过这些Web组件可以访问后台的DBMS或其它系统。

页面导航：把一个页面跳转到另外一个页面定义为页面导航。页面导航包括4种方式：超链接(Link)，它只发生在客户端页面；页面请求(Request)，即客户端页面对服务器端页面发起http请求；页面响应(Response)，即服务器端页面对客户端页面所发Http请求的响应；重定向方式(Redirect)。

簇：Web应用的异构性和动态性使对Web应用的行为和结构建模非常困难。在此采用功能簇对大型的Web应用进行功能上的划分，从而可以用有限状态机对每个分层上的簇建立有限状态机导航模型。本文把一个Web应用分为若干个功能模块，在这里我们使用一个通用的术语“簇”，来代表实现Web应用的逻辑功能模块的集合。

Web应用划分的第一步是把Web应用分解成功能独立的簇。从高层面来讲，簇是抽象的、可以实现用户定义的功能模块。从下一个层面来讲，簇是相关联的软件构件和Web页面，一起共同实现一些用户层功能。从底层来讲，簇可以是单独的Web页面和Web构件，它们代表自己单个主要的功能。簇可以从站点导航设计，并从联合组件的设计信息中识别确定出来。

逻辑网页：许多Web页面包含不止一个HTML表单，每一个表单动态链接不同的后台软件构件。为了便于测试这些组件，Web页面被定义为若干逻辑页面。逻辑页面可以是整个物理页面，也可以是Web页面的一部分，它们通过HTML表单从用户那里接受数据，然后把数据传送给特定的组件。逻辑网页可以被自动提取，因为在HTML Form标签中定义了在Web页面上每个独立的表单。处理表单数据的后台构件在Form Action属性里面声明，而客户端软件(如JavaScripts)是指在单个Form的输入域。通过提取逻辑网页，就可以得到与之相连的后台组件，这样可以便于对后台组件之间的关系建模。

Web组件：目前大多数的Web应用都是采用基于组件方式的开发方法。一个基于组件的Web应用通常包含了一系列独立的、可复用、松散耦合的Web组件，每个Web组件可以通过统一的接口调用实现一个或多个功能，它们可以以即插即用的方式进行集成。在一个应用中，Web组件可以采用不同的编程语言编写，并运行于不同的平台，可以是新开发的组件、旧版本的组件或是第三方提供的商用构件。Web组件代码通常是无法获得的。

接口：组件之间的交互可以是直接的，也可以是间接的。直接交互包括组件接口的互相调用，间接的交互可以通过一系列的事件。当对基于组件的Web软件系统进行测试时，应当假设每个单独的组件都经过了充分的单元测试。因此，软件系统可靠性的成功关键是确保组件之间交互的准确性。而这种交互往往是通过接口来实现的。接口是激活组件最常用的方法[1]，因此有必要在系统集成测试中对每个接口进行至少一次的测试。

事件：测试接口确保每个接口在运行过程中至少被调用过一次。这种情况与传统测试规则是一致的，即每个系统功能和过程都必须至少被测试一次。然而，一个接口在不同的背景下被不同的组件调用，产生的结果不同。因此需要考察每个接口在系统运行时可能的行为，每个接口的调用至少需要被测试一次。另外，一些事件并非通过接口而触发，但对组件也有影响，这些事件同样需要测试。

3.2 Web应用的测试建模方法

本文采用基于有限状态机和UML扩展的模型对Web应用进行测试建模。Web应用的UML扩展引进了许多版型(比如接口、事件、Web页面、实体等)，标注值和约束。本文通过Web应用的划分导出了两种测试模型：有限状态机导航模型、组件依赖模型。其中有限状态机导航模型主要用于Web应用的系统行为测试，包括前台页面测试，以及指导测

维普资讯 http://www.cqvip.com

试人员进行后台服务器端测试。而组件依赖模型是用来生成测试用例的核心模型，包括系统领域中与不同逻辑网页相连的不同组件之间的背景依赖关系模型和内容依赖关系模型。

测试模型的获取从 Web 应用的划分开始。从站点导航设计，可以对 Web 应用进行功能簇的划分，用分层的有限状态机导航模型来对 Web 应用的各组成部分进行行为建模。在 Web 应用分解成簇之后，模型从抽象的功能簇具体到底层的 Web 页面和组件，这样可以在分层上提取组件，建立分层的组件依赖模型。Web 应用的测试建模过程如图 2 所示。

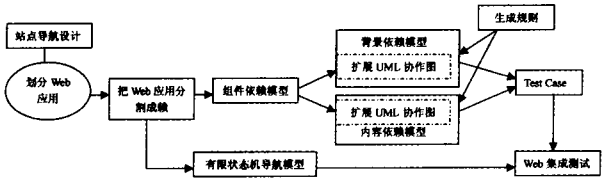

图 2　Web 应用的测试建模过程

以下将通过一个新闻发布网站中的后台管理子系统来阐述 Web 应用的测试模型。

3.3　基于有限状态机的导航模型

导航行为作为一种特性被 Web 应用引进。通过页面之间的连接，用户可以在 Web 应用页面间冲浪浏览。然而，页面之间连接的不确定性和不可达的页面经常会困扰用户，导致用户对 Web 应用有不确信的感觉。而且，由于 Web 页面之间连接的复杂性，用户有时无法了解系统的运行流程。为了验证导航行为是否正确恰当，并指导测试人员进行 Web 集成测试，引入导航模型。本文用有限状态机对页面动态的导航行为建模，研究 Web 应用的交互特性。通过 Web 应用的划分，可以把整个 Web 应用分割成簇，在分层的同时，为每个簇建立有限状态机。

本文的一个实例，新闻发布网站中的后台管理子系统集合了 Web 应用特点中的几个常见的连接方式。通过高层簇的划分，可以看到管理员登录后有 3 个功能簇：用户管理、新闻发布、BBS 管理。图 3 是由站点导航设计提取出的高层有限状态机页面导航模型。图 4 是由上层簇中的结点 User Manage 分解得到的。

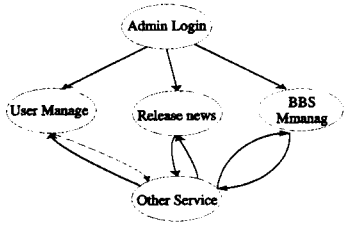

图 3　高层的有限状态机导航模型

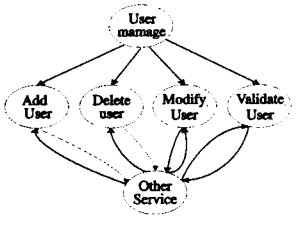

图 4　底层的有限状态机导航模型

采用有限状态机可以先对 Web 应用的各组成部分进行行为建模，然后对整个 Web 应用的行为建模。首先，有限状态机是从底层簇中产生的，这些簇只包含组件和 Web 页面（即不包含簇）。接下来，聚集下层的有限状态机集合，组成高层的簇。在这些簇中，下一级簇群得到的每一个有限状态机用一个单结点来表示状态。最后，有限状态机集合可以聚集到一个应用级状态有限状态机，它可以用来定义整个 Web 应用的有限状态导航模型。

3.4　基于 UML 的组件依赖模型

通过有限状态机导航模型，可以指导测试人员测试 Web 页面导航行为。许多 Web 页面包含不止一个 HTML 表单，每一个表单（逻辑网页）动态链接不同的后台软件构件，也就是组件。Web 软件测试建模的核心是通过模型来获得测试用例。如何对这些组件的行为以及组件之间的交互关系建模，是模型实现面临的一个难题，特别是当组件的源代码（如包装的构件）无法获得的时候。如何有效获取测试元素来进行测试也是需要关注的一个问题。没有源代码，但可以获得关于接口和事件的规格说明，然后设法获得关于接口的背景依赖关系和内容依赖关系所需要的信息。

本文采用 UML 来获得一个划分层面上组件之间的关系模型，以精确获得背景依赖和内容依赖测试模型。

3.4.1　背景依赖模型

背景依赖关系：接口和事件测试确保组件之间所有的交互都执行过。然而，执行一个 Web 应用系统包含了一组组件的交互。不同的事件触发序列将产生不同的结果。为了获得事件之间的内部关系，定义一个背景依赖关系，它和传统软件中的控制流依赖关系类似。如果存在一条执行路径，当触发 e1 将直接或间接触发事件 e2，则事件 e2 具有一个背景依赖关系。对于一个事件 e，有必要测试与 e 存在背景依赖关系的所有事件。它可以让测试者了解事件 e 历史执行记录对事件 e 执行结果可能产生的影响。

背景依赖关系不仅包括直接的交互，也包括接口之间或接口与事件之间间接的协作关系。因此，测试背景依赖关系可以用来检查组件之间由于不适当的交互关系而产生的互操作错误。

当对组件进行集成时，程序员一般关注如何定义组件的接口和事件。但是对于这些组件接口、事件如何交互，以及对于集成系统可能产生的风险往往很少考虑。可以通过引入事件依赖关系来扩展 UML 协作图，用它来获得一个分层上与不同逻辑网页相连的不同组件之间的依赖模型。如图 5 所示，通过 UML 协作图描述图 4 中与不同逻辑网页相连的不同组件之间的背景依赖关系。

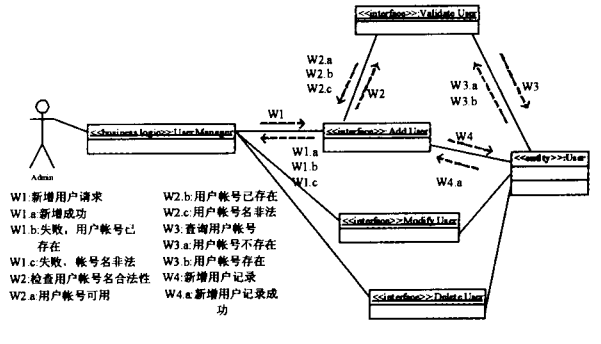

图 5　用户管理模块的背景依赖关系图

根据图 5 的依赖模型，并分析所有事件依赖关系，得到 3 个存在依赖关系的事件链(a,b,c)。因此，可以建立如下组件之间的事件依赖关系矩阵：

维普资讯 http://www.cqvip.com

	W1	W2	W3	W4
a	W1. a	W2. a	W3. a	W4. a
b	W1. b	W2. b	W3. b	
c	W1. c	W2. c		

其中每一行代表存在依赖关系的事件，既包括直接依赖，也包括间接依赖。例如对于依赖关系链 a，W2. a(用户帐号可用)依赖于 W3. a(当前用户不存在该用户帐号)。

从依赖矩阵中，可以获得多条执行路径，而每条路径可以把它称作一个 Web 系统的执行情景。如路径：W1-〉W2-〉W3-〉W3. a-〉W2. a-〉W4-〉W4. a-〉W1. a 表示新增用户成功的一个情景。而路径：W1-〉W2-〉W2. c-〉W1. c 表示由于帐号名称非法导致新增用户失败这样一个情景。

3.4.2 内容依赖模型

内容依赖关系：一个组件接口的调用将导致对该组件所实现的某个功能的调用。因此，当在接口 v1 定义的一个功能与另外一个接口 v2 定义的功能存在内容依赖关系时，则 v1 和 v2 的调用顺序将影响执行结果。如果这两个接口存在数据依赖关系，那么一个内容依赖关系就存在于两个接口之间。一个接口封装了一个或更多的信号，每个信号表示接口所实现的一个功能。当一个接口被调用，会根据被请求的服务执行一个或更多的功能。因此，接口依赖关系可以从继承功能依赖关系得到。更精确地说，功能 f2 依赖功能 f1 当且仅当在 f1 中定义的变量值也在 f2 中使用。内容依赖关系可以定义为：当且仅当接口 v1 包含功能 f1 的信号，接口 v2 包含功能 f2 的信号，并且 f2 依赖 f1，则接口 v2 对接口 v1 存在一个内容依赖关系。

背景依赖关系反映的是组件中对象根据角色和组件之间的单一交互得到的控制序列。然而，在不同组件交互中的内容依赖关系可能不能完全由控制流信息得到。例如，图 7 所示的是本文实例新闻发布网站中的用户管理服务组件。该组件包括两个接口：新增用户和管理用户。背景依赖关系描述各个组件之间的交互，但是组件之间的内容依赖关系则无法获得。例如，删除用户接口依赖于新增用户接口，因为删除用户事务将修改 Uesr 实体，而修改事务将使用该实体来验证该用户帐号是否存在。不幸的是，内容依赖关系既不能在程序中也不能在 UML 模型中直接得到。同样，可以通过用事件依赖关系扩展的 UML 协作图来描述 Web 组件之间的内容依赖关系。

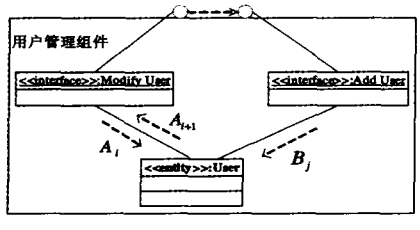

图 6 用户管理模块的内容依赖关系图

UML 协作图和序列图描述组件中对象之间的交互关系。当交互关系包括了实体类时，协作图可以描述两个交互之间的依赖关系。例如，图 6 描述了消息 B_j 流转到实体类 User，而没有信息流出 User。一般来说，消息 B_j 将更新 Uesr 对象的信息，这个过程定义为 update 消息。另外一方面，消息 A_i 和消息 A_{i+1} 流入流出 User，表示 User 的信息检索，该过程称之为 retrieve 消息。根据对内容依赖关系的定义可知，包含消息 A_i 和 A_{i+1} 将依赖于包含 B_j 消息的序列，即存在路径：B_j-〉A_i-〉A_{i+1}。一般来说，接口 I 依赖接口 I' 当且仅当 I 所包含的一个消息序列包括一个 update 消息，而 I' 包含的另外一个序列则包括了一个 retrieve 消息。

从测试角度来看，如果执行 Modify User 接口，必须先执行 Add User 接口。

4 测试用例的生成

对 Web 应用进行测试建模的主要目的就是获得可用的测试用例。通过前面的背景依赖模型，可以识别出一个系统事件所包含的测试情景，这种测试情景可用于产生测试用例。针对背景依赖模型和内容依赖模型，本文提出了以下几条用于产生测试用例的规则。

(1)在背景依赖关系模型中，每个事件链至少需要测试一次，即可生成一个测试用例。

(2)在内容依赖模型中，存在内容依赖关系的事件链要按依赖顺序，正反各测试一次。

如果对背景依赖模型中每个情景的前置条件和事件流进行分析，则很容易标识输入变量和约束将导致系统情景的不同状态(即后置条件)。依据规则 1 采用矩阵格式可以清晰地描述每个事件依赖链产生的测试用例。

表 1 由背景依赖图生成的测试用例

测试用例 ID	情景	帐号	密码	帐号格式非法	已存在该帐号	预期结果
TC1	成功新增用户	User1	Pass1	False	False	新增成功
TC2	帐号无效	234_w34	Pass1	True	A/N	新增用户失败，帐号无效
TC3	帐号重复	User2	Pass2	False	True	新增用户失败，账号已存在

同样，根据规则 2 可以得到两个测试用例。

表 2 由内容依赖图生成的测试用例

测试用例 ID	情景	帐号	用户资料	该用户帐号已添加	预期结果
TC1	修改用户成功	User1	{Pass1，name，Address，…}	True	新增成功
TC2	帐号不存在	User1	{Pass1，name，Address，…}	false	修改用户失败，该帐号不存在

从逐步求精的测试来看，通过 Web 应用的划分，测试用例可以随着测试模型分层的细化而细化，最终可以产生具体的测试数据。所以，根据测试的具体要求，对于以上的这些测试用例可以通过模型的细化进一步精化，来达到测试的最终要求。

总结 本文通过对 Web 应用的划分，提出了一种基于 Web 软件集成测试的建模方法和相关测试方法。该方法通过用分层的有限状态机对 Web 应用的各组成部分进行行为建模，得到有限状态机导航模型；并通过扩展 UML 的协作图来描述 Web 应用中不同组件之间的背景依赖关系和内容依赖关系，得到组件依赖模型。在组件依赖模型的基础上获得测试情景，最后根据相应的生成规则由测试模型生成一个分层上的 Web 组件的测试用例。同时，本文用到了基于有限状态机的导航模型，用以帮助测试人员理解系统功能和页面之

维普资讯 http://www.cqvip.com

间的导航关系，为系统集成测试提供指导。

参 考 文 献

1 Wu Ye, Chen Mei-Hwa, Offutt J. UML-based Integration Testing for Component-based Software. In: The 2nd International Conference on COTS-based Software Systems (ICCBSS), Ottawa, Canada, February 2003
2 Nilawar M. A UML-based Approach for Testing Web Applications, 2003
3 Lee S, Offutt J. Generating Test Cases for XML-based Web Applications. In: Proceedings of 12th International Symposium on Software Reliability Engineering, 2001. 200～209
4 Wu Y, Offutt J. Modeling and Testing Web-based Applications. GMU ISE Technical ISE- TR- 02-08, 2002
5 Ricca F, Tonella P. Analysis and Testing of Web Applications. In: Proceedings of the 23rd International Conference on Software Engineering (ICSE 2001), 2001. 25～34
6 Conallen J. Building Web Applications with UML. Second Edition. Addison Wesley, 2002
7 Kung D, Liu C, Hsia P. An Object-oriented Web Test Model for Testing Web Applications. In: Proceedings of IEEE 12th Annual International Computer Software and Application Conference, 2000. 3～5
8 Conallen J. Modeling Web Application Architectures with UML. Comm of the ACM, 1999, 42(10): 63～70
9 Ricca F, Tonella P. Testing Processes of Web Appplications. Annals of Software Engineering, 2002, 14(1-4): 93～114
10 Andrews A, Offut J, Alexander R. Testing Web Applications by Modeling with FSMs. Software Systems and Modeling, August 2005, 4(3)
11 Booch G, Rumbaugh J, Jacobson I, The Unified Modeling Language User Guide. Addison-Wesley, 1998
12 许蕾，徐宝文，陈振强. Web 测试综述[J]. 计算机科学，2003
13 颜炯，王朝，陈火旺. 基于模型的软件测试综述[J]. 计算机科学，2004, 31(2): 184～187

(上接第 252 页)

任务(Task)对象来执行。若实时执行体的调度算法是基于抢占式的调度，则开发者只需静态设置构件服务的优先级数值即可。在接口描述中设计 PD 优先级参数供设计者静态设置优先级，以方便 RTMOS 根据优先级对构件服务动态调度，可增强配置的灵活性。

当然此种调度算法并不适合所有情形，对于选择其它合适调度策略的 RTMOS(如时间轮询，事件驱动等调度)[7]，显式指定应用构件的时间特性是必要的，以便为 RTMOS 调度器的调度提供基础，并且这些时间特性在实时应用软件的仿真和验证阶段也要用到。

3.2 资源控制接口 RCI

资源控制接口 RCI 在构件模型的图形符号中由终端器表示。代表实时系统控制的外部设备，是同外部硬件环境通讯的接口。定义 RCI 是个二元组的集合：RCI=｛⟨DN, OP⟩｝。其中：

DN(Device Name)：表示实时系统控制的外部设备名。

OP(Operation)：表示作用于该设备操作的集合。

所监控的设备可以是数据采集部件传感器、机器人等。作用在这些设备上的操作有读数据、写数据和设置设备状态等。设备资源由详细设计层的中断或资源对象建模。其它的操作有通过远程服务调用请求服务，发送信号给异步事件，对设备资源中断进行管理，异常情况处理等。

其它构件通过服务接口的服务原语访问 RCI 提供的外部设备服务(操作)。

3.3 合成接口 CIF

合成接口 CIF 定义构件的聚合关系及服务间的同步互斥关系等。定义 CIF 是个三元组：CIF=⟨ADT, SDT, MDT⟩。其中：

ADT(Aggregation Descriptor)：聚合描述子，表示构件的聚合关系。

SDT(Sequence Descriptor)：执行先后顺序描述子，表示构件服务的同步关系。

MDT(Mutex Descriptor)：互斥描述子，表示构件服务的互斥关系。

定义聚合描述子 ADT 为各个子构件的集合：ADT=｛AO_1, AO_2, …, AO_n｝。从构件层次化分解设计的角度看，构件可由下一层子构件复合而成，以形成更大粒度的可重用构件。其上层构件的接口应指定说明下一层组成的子构件，此构造称为聚合概念。表示了构件的内部构造。子构件又可由下一层子构件合成，因而构件的聚合关系定义是递归的。被聚合的子构件提供的服务接口通过其上层构件接口向外界展示。

定义序列描述子 SDT 是一个二元关系先后执行顺序序列组的集合：SDT=｛Sequence⟨sn_i, sn_j⟩｝。其中，sn_i, sn_j 是各子构件或交互构件的服务，Sequence⟨sn_i, sn_j⟩表示执行先后顺序的二元关系，sn_i 在 sn_j 之前执行，表明 sn_i 与 sn_j 是同步的。构件服务的执行可能会调用其聚合的子构件服务或同层其它 AO 构件的服务。为方便实现层的任务构件对服务的执行时序的调度以及实时任务同步的要求，有必要指明同聚合的子构件服务或调用的同层其它 AO 构件服务之间的先后执行顺序，通过定义先后顺序关系及互斥关系来达到此目的。

定义构件服务之间的互斥关系描述子 MDT 是一个二元关系互斥序列组的集合：MDT=｛Mutex⟨sn_i, sn_j⟩｝，Mutex⟨sn_i, sn_j⟩表示 sn_i 与 sn_j 有互斥关系。构件服务之间的互斥关系是对称的，如果 sn_i 与 sn_j 是互斥的，则意味着 sn_j 与 sn_i 互斥。根据具体应用，MDT 可以为空。

结束语 本文主要论述 DRSCDE 构件的非功能性接口模型，诸如时间性、合成性及同步互斥等。给出分布式 C/S 关系实时多任务应用系统图形化设计软件的实时构件的接口定义说明。针对实时特性，提出实时构件的非功能性接口在时间性、调度性、合成性、同步、互斥以及资源设备控制方面的语义规约。构件提供的服务接口、与客户构件的交互协议及状态接口已在另文中论述。该实时构件接口模型独立于基础设施，如操作系统、中间件等。将要做的研究工作是如何在 RT-CORBA 中间件上发布 DRSCDE 构件。

参 考 文 献

1 Atkinson C, BayerJ, et al. 顾剑，等译. 基于构件的产品线工程 UML 方法. 机械工业出版社，2005
2 OMG. UML Profile for Scheduling, Performance, and Time - Request for Proposal. OMG, Inc, OMG document ad/99, April 1999
3 刘晓燕，张云生，等. 复杂实时系统软构件对象设计. 计算机工程与应用，2003, 39(31): 119～121
4 刘晓燕，张云生，等. 基于构件的分布式实时系统架构图形化建模. 计算机应用研究，2005(增刊): 52～53
5 Schwarz J-J, Maranzana M, Skubich J J, Martinez Y. Applicative Objects Interconnection in a Graphical Design for Real-Time Applications [C]. In: 20th IFAC Workshop on Real Time Programming (WRTP'95), Florida, USA, November 1995
6 Skubich J, Schwarz J J, Maranzana M, Szmuc T. Time Requirement Specification in Graphical Real-Time Design. IFAC Control Engineering Practice, 1996, 4(2): 207～215
7 张云生. 实时控制系统软件设计原理及应用. 国防工业出版社，1998
8 Pasetti, Pree W. The component software challenge for real-time systems [C]. In: Proceedings of the First International Workshop on Real-Time Mission-Critical Systems, Scottsdale, AZ, Nov. /Dec. 1999
9 Wang Shengquan, Rho S, et al. Real-Time Component-based Systems [C]. In: IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2005), San Francisco, California, March 2005