基于J2EE平台的Java构件库的研究和实现*)

曾 一[1] 郭永林[1] 曾 勇[2] 袁 纲[1]

(重庆大学计算机学院 重庆 400044)[1] (重庆宏声新思维信息产业有限责任公司 重庆 400000)[2]

摘 要 构件库是构件复用的重要部分。以项目为背景,提出了一种Java构件库系统的设计与实现方案。介绍了基于MVC(Model View Control)的构件分类树结构,描述了该结构下的构件表示模型,阐述了分类树深度优先遍历的构件检索方法和结合构件规约和可控词汇表的规范函数匹配方法。该构件库系统提供了一种语义和语法相结合的经验模型。

关键词 J2EE,构件库,MVC,深度优先算法,构件规约

Research and Realizeation of a System of Java Component Library Based on J2EE Platform

ZENG Yi[1] GUO Yong-Lin[1] ZENG Yong[2] YUAN Gang[1]

(College of Computer Science, Chongqing University, Chongqing 400044)[1]

(Chongqing HS-Sysway Information Industry Group Co., Ltd, Chongqing 400000)[2]

Abstract This paper introduces a framework about a java component library system that is important to software reuse. Then describes a classified construction of component based on MVC(Model View Control) and a relevant method of component representation. In the end, it writes up how to retrieve component by depth-first search algorithm and how to adapt it by norm function. This component library system is a kind of incorporative model between specification and syntax.

Keywords J2EE, Component library, MVC, Depth-first search algorithm, Component signature

1 引言

软件构件技术作为软件复用的核心技术,已得到了广泛的研究和应用。基于构件的软件开发(CBSD)也日益成熟,到目前为止,Sun公司的EJB、微软的COM以及OMG的CORBA作为CBSD的三大主流技术都提出了自己的构件复用模型。然而在CBSD中,支持构件复用的构件库系统是另一个至关重要的部分,它可以减小构件的复用代价,是实施软件复用的必要设施。构件库系统的主要功能是对构件进行描述、管理、存储和检索,以满足基于"构件—构架"复用的软件开发过程的需要。构件库系统应该有效地组织和管理大量可复用构件,并提供相应的工具支持构件使用者在软件开发过程中方便地查询、理解和选取构件,从而保障基于构件复用的软件开发成功地进行。对于它的研究,国内外已经有相应的标准和成果,如美国军方和政府资助的CARDS、SSET、DSRS等构件库系统,国内有著名的北大青鸟构件库系统和上海构件库系统等[1]。目前的构件库依然存在下面的问题:基于规约的构件库系统,是从构件接口的语法方面检索匹配的,但忽略了构件要解决的问题域和功能域。由于语义网络表示的困难,基于语义网络的构件库,目前还没有成熟的模型。本文提出的J2EE平台下Java构件库系统HSCL(HS-Sysway Component Library)系统是结合语义分类检索和语法匹配的经验模型。

2 J2EE构件分类及特点

J2EE是由Sun公司联合IBM、Oracle、BEA等大型企业应用系统开发商于1998年共同制订的一个基于Java组件技术的企业应用系统开发规范,该规范定义了一个多层企业信息系统的标准平台,旨在简化和规范企业应用系统的开发和部署。

J2EE将一个完整企业级应用的不同部分纳入不同的容器中,每个容器包含各自相应的构件,具体包括以下四种:Applet、Application客户构件、Web构件和EJB。其中Applet和Application客户构件在客户端运行,Web构件和EJB在服务端运行。Web构件又分为JSP和Servlet两种。这几种构件中,除了JSP是将Java和HTML语言相结合以外,其余构件都符合Java的语法规则,从这一点来看它们是相同的,然而在J2EE体系架构中,它们所起的作用却不同。本文描述的HSCL系统利用它们遵循Java语法结构的特点和它们在J2EE规范中所处的地位,对它们进行统一的表示、分类和组织。所以说,HSCL系统是基于J2EE平台的Java构件库系统。

3 基于J2EE的HSCL系统框架

HSCL系统框架借鉴了国内外先进的构件库系统[1,3],从构件的制作组装、提取分类、查询、适应性修改、应用软件代码自动生成等多个方面进行了流程化的设计。目标是使开发商

*)基金项目:重庆市信息产业发展资金资助(编号:200301002)。曾 一 副教授,硕士生导师,主要研究方向:软件过程、软件集成环境、软件开发环境、软件工程。郭永林 硕士研究生,主要研究方向:软件工程、软件复用、构件技术。曾 勇 硕士研究生,主要研究方向:软件工程、企业信息化。袁 纲 硕士研究生,主要研究方向:软件工程、软件复用。

在尽可能短的时间内，利用已有的构件搭建新的应用程序。整个框架的简化模型如图 1 所示。

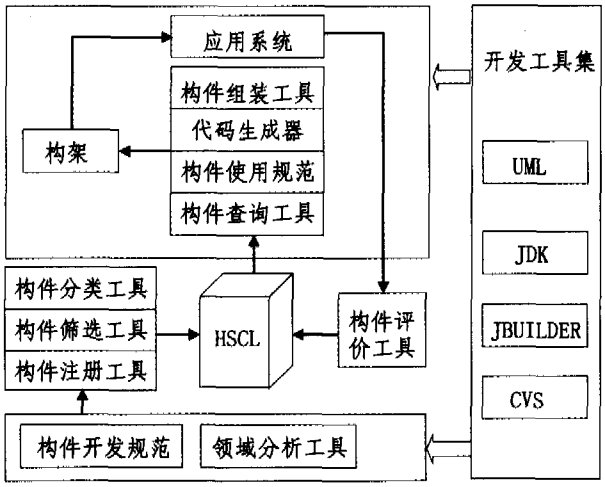

图 1　HSCL 系统体系框架

整个框架分为三个部分：构件开发、构件管理和构件使用。构件开发是指利用开发工具集，借助于领域分析工具，按照开发规范进行构件生产的过程。构件管理是指构件生产者通过注册、申请构件、然后由构件库管理系统自动将其筛选、分类和入库的过程。构件使用是指构件使用者利用查询工具提取构件，并进行组装和适应性修改，最终形成企业应用系统的过程。在构件库管理系统中，构件的分类、表示、检索、匹配是四个重要的环节。它们决定着构件的查全率、查准率等复用效率。下面分别对它们进行讨论。

4　HSCL 构件的分类和表示

在 CBSD 中，构件的分类和表示方法决定构件的存储方式和检索机制。目前已经有多种分类方法，例如：枚举分类、关键字分类、刻面分类和超文本分类等。然而，由于自动化程度和检索效率的要求不断提高，对于它的研究还存在很大的空间[2]。上述几种分类方法都是基于构件本身分类的，所有的构件处于平行分类。本文将介绍一种基于层次结构的构件分类方法。

4.1　HSCL 构件的分类方法

本系统的构件分类采用了 J2EE 软件设计的典型结构 MVC(Model View Control)思想。在 MVC 体系结构下，一个应用被分为三个部分：Model、View 和 Controller，每个部分负责不同的功能。Model 是指对业务，数据/信息的处理模块，包括对业务数据的存取、加工、综合等；View 是指用户界面，也就是面向用户的数据表示；Controller 则负责 View 和 Model 之间的流程控制。

在构架层，MVC 应用于功能级的划分，把所有的构件分为表现层、通讯层和业务层三大类。表现层主要包括 Swing 基本控件和可视化的扩展组件。通讯层主要包括前后台代理和数据库连接等服务类组件，业务层则包括 Java Bean 和 EJB 等业务组件。

在构件层，MVC 又应用于 Swing 控件的实现上。它封装了三个在大多数图形应用程序中都存在的通用抽象：模型、视图和控制器。其中，模型负责维护数据、视图负责提供模型部分数据的可视图，控制器为视图处理事件。这样使得一个复合构件可以更加灵活地构造。

在此基础上，结合构件粒度大小不同的特点，可以形成一棵构件分类树，如图 2 所示。其中，所有的非叶子节点对应于 Java 中的包体。叶子节点包括类文件、接口和 JSP 页面等。

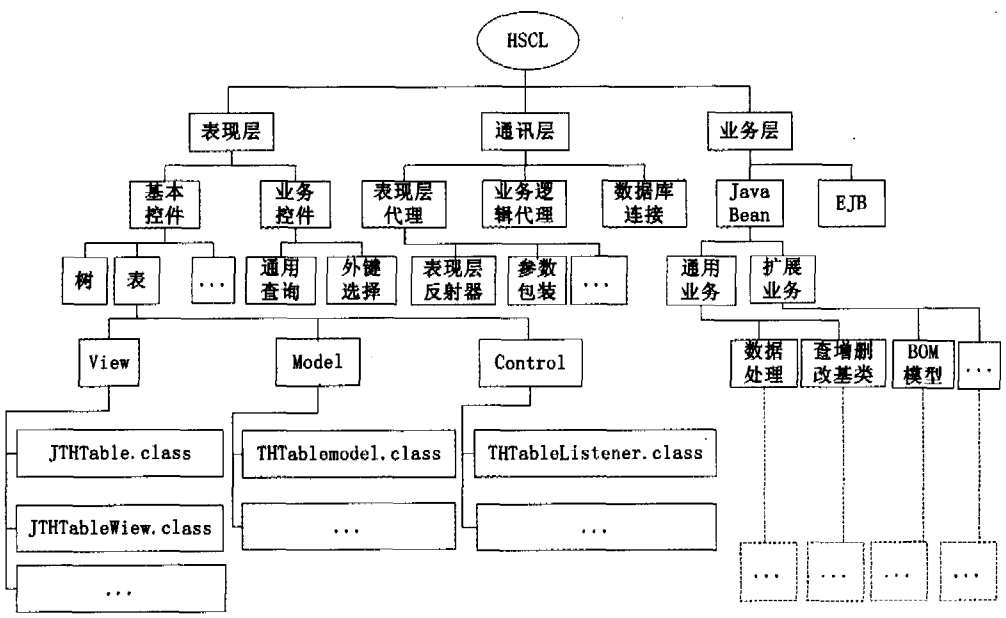

图 2　HSCL 分类结构图

4.2　HSCL 构件的表示方法

构件是软件系统的一个物理单元，是封装了服务的源代码实体。由单个类实现的构件称为原子构件，由多个类相互作用实现的构件称为复合构件[4]。在复合构件中，有些成员构件作为复合构件的核心部分，提供复合构件的外部接口，称为核心构件；另外一些构件作为支持，提供内部服务，称为支持构件。

构件规约是构件表示的主体，包括接口和结构两部分。接口描述了构件的服务，结构描述了构件的复合特征。

服务是封装了构件 Provides 和 Requires 两部分函数的集合。Provides 是该服务提供的功能集合。Requires 是该服务必须实现的功能集。

下面以表组件 DBTableView 来举例说明本系统构件的表示。它由 JTHTable、THTablemodel 和 THTablelistener 三个原子构件组成。JTHTable 必须实现 JTableInterface 定义的服务，并增加排序等其他服务。

原子构件 JTHTable 的表示方法：

```
Component HS. Swing. Table. View. JTHTable{
    Properties{
    Row, Column, RowbgColor, ColumnbgColor…
    }
    Requires{
      JTableInterface( )
    }
    Provides{
      Function Sort(String in Condition, List out DataList)
    }
}
```

其中：HS. Swing. Table. View——构件的存储路径。

JTHTable——构件名称。

Properties——原子构件的属性集。

Requires——表示该构件必须实现的函数集，这部分定义为 Java 接口类型。

Function——规范函数，是函数实体的规范化描述形式。具体将在构件检索匹配中描述。

复合构件 DBTableView 的表示方法：

```
Component HS. Swing. Table. DBTableView{
    Component HS. Swing. Table. View. JTHTable
    Component HS. Swing. Table. Model. THTablemodel
    Component HS. Swing. Table. Control. THTablelistener
    Relation ( Component JTHTable, Component THTablemodel,
      TYPE reference)
    Relation ( Component JTHTable, Component THTablelistener,
      TYPE reference)
}
```

其中：HS. Swing. Table——复合构件的表示路径，也就是离所有成员构件最近的共同的祖辈节点路径。

Relation——描述成员构件间的关系。

可以看出，复合构件是由一个核心构件 JTHTable 和两个支持构件 THTablemodel，THTablelistener 组装来实现的。为了把核心组件和支持构件区分开来，构件命名约定所有的核心构件以“JTH”开头。

构件的表示是整个系统的核心部分。一方面它包含了构件的分类存储路径，为构造检索算法提供数据结构；另一方面，它采用规范函数的定义方法，为构件匹配提供了接口。

5 HSCL 构件的检索和匹配

对于构件的使用者来讲，构件复用依然是以问题域（功能域）驱动的，所以构件的检索和匹配分为两个步骤。第一，检索过程，寻找满足功能要求的构件；第二，匹配过程，检查这些构件接口是否与系统接口相匹配[3]。

5.1 HSCL 构件检索机制

构件的检索依赖于构件的分类和表示方法，分类表示方法不同，所选用的检索机制也不同。本系统构件分类采用图 2 所示的树型结构，所有的构件按照 MVC 分类并按功能域的粒度大小从上到下组织成一棵树。父节点和子节点的关系为功能域的包含关系。若一个问题属于某个功能域 A，则该问题必定也属于 A 的父节点域。所以，按照本系统的分类方法，检索出来的应该是一棵匹配树。本系统中采用了树的深度优先搜索策略来遍历并标识，得到如图 3 描述的树结构，节点 L 表示该构件所属的分类路径。节点 Y 表示目前路径下有满足功能的构件（可能是原子构件，也可能是复合构件）。图 4 表示某问题的最大匹配树。

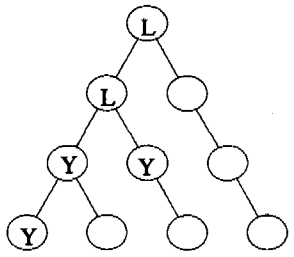

图 3

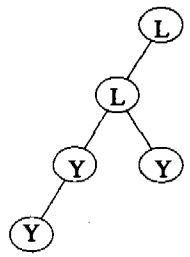

图 4

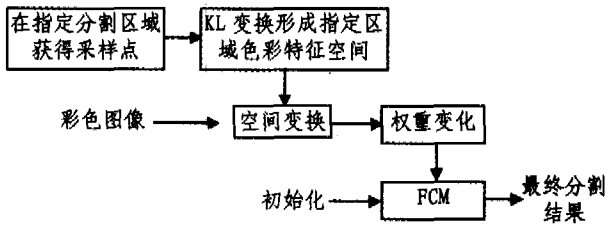

图 4

5.2 HSCL 构件匹配机制

最大匹配树只是从功能级检索到可能满足需要的构件，是从问题域出发的大粒度的检索。然而判断这些构件哪些可用，还需要进行构件匹配。

判断构件是否匹配应该考虑两方面的因素。第一，匹配构件与被匹配构件是否具有一致的接口；第二，对应的接口提供的服务是否一致。

从语法的角度来看，function 中功能名和参数名是可以替换的，并不反映接口语法的本质，而参数类型和返回值类型则是不可替换的，代表着接口语法的本质特征，因此参数类型和返回值类型是语法中与检索相关的信息。

定义 1 n 个字符串 $s1,s2,\cdots,sn$ 按字母顺序从小到大排序后依次连接而成的字符串称为 $s1,s2,\cdots,sn$ 的规范连接，记为 $Norm(s1,s2,\cdots,sn)$[5]。在文[5]中，作者已经证明对于相同的 n 个字符串，不管它们最初的排列次序如何，总可以得到相同的规范连接。

定义 2 任何函数接口 $F(s1,s2,\cdots,sn)$ 通过将参数类型列表转换为它的规范连接并且将函数名规范化后形成的新函数 $F_N(s1,s2,\cdots,sn)$ 称为规范函数，它们之间是多对一的映射关系。其中 F_N 表示系统可识别的规范函数名。例如，数字相加、字符串连接、数据集合并的规范函数名都为“PLUS”。所有规范函数名存储在一个可控词汇表中，形成了构件检索的语义知识库。

从定义 2 可以看出，规范函数中，函数名是对提供服务的规范化表示；函数类型是对参数类型的规范化表示。

构件 A 和构件 B 精确匹配的必要条件是 $A\{F_{N1},F_{N2},\cdots,F_{Nm}\}=B\{F_{N1},F_{N2},\cdots,F_{Nm}\}$。其中：$A\{F_{N1},F_{N2},\cdots,F_{Nm}\}$ 表示构件 A 的所有规范函数的集合。

可见，规范函数匹配，是在构件规约匹配的基础上进行的改进。它利用可控词汇表定义函数名，在接口匹配的同时，考虑接口本身的具体含义，提高了匹配的准确性。

结论 文章从 HSCL 系统的体系结构出发，分别介绍了该构件库系统的分类、表示、检索和匹配方法。按照 MVC 结合构件粒度分类，符合人们认识事物的一般规律，有效地解决了构件理解的困难，并为构件检索提供了一种数据结构。构

（下转第 280 页）

用户功能的必须是管理员角色等。

4 细粒度扩充后的 RBAC 模型的应用实例

某大型企业的综合业务管理系统采用 B/S 结构，采用细化后的 RBAC 模型实现其安全访问控制体系，取得了良好效果。系统主要数据表及处理过程如下：

4.1 基础表

单位表、功能菜单表、操作级别表、数据周期表、数据对象表。这些表主要是存储系统的各项基础数据。

4.2 角色类表

功能角色表(功能角色 ID，功能角色名称，所属单位 ID，父功能角色 ID，互斥角色 ID，必备角色 ID，基本限制数)

功能角色权限表(功能角色 ID，功能菜单 ID，操作级别 ID，数据周期 ID，数据对象 ID)

单位角色表(单位角色 ID，单位角色名称，所属单位 ID，父单位角色 ID，互斥角色 ID，必备角色 ID，基本限制数)

单位角色权限表(单位角色 ID，允许访问单位 ID)

角色表(角色 ID，角色名称，所属单位 ID，父角色 ID，单位角色 ID，功能角色 ID，互斥角色 ID，必备角色 ID，基本限制数)

管理员在系统中分别定义单位角色(存入单位角色表)和功能角色(存入功能角色表)，然后给这些角色设置相应单位权限和功能权限(包括功能菜单 ID、操作级别 ID、数据周期 ID、数据对象 ID 等)，分别存入单位角色权限表和功能角色权限表。

管理员定义角色基本信息，然后给该角色指定单位角色和功能角色，存入角色表。

管理员还可以分别给角色、单位角色和功能角色设置该角色的互斥角色、必备角色和基本限制数，用来对其进行相应的约束。另外角色、单位角色和功能角色支持角色继承，设置的相应父角色 ID 分别存入角色表，单位角色表和功能角色表的父角色 ID 列。

4.3 用户类表：

用户表(用户 ID，用户姓名，所属部门 ID，用户级别 ID)

用户角色表(用户 ID，角色 ID)

用户权限表(用户 ID，允许访问单位 ID，功能菜单 ID，操作级别 ID，数据周期 ID，数据对象 ID)

管理员首先录入用户基本信息(存入用户表)，然后给用户指定一个或多个角色(这些信息存入用户角色表)，系统会根据角色表中的互斥角色和必备角色约束来判断这些指定是否正确；如果正确，那么系统根据角色 ID 及父角色 ID 在角色表中找出对应的单位角色 ID 和功能角色 ID，然后根据单位角色 ID 和功能角色 ID 分别在单位角色权限表和功能角色权限表找出相应的明细权限，组合后存储到用户权限表中。这样就完成了每个用户的明细权限设置。

用户在登录系统后，系统自动在页面上根据用户权限表中的用户 ID 和功能 ID 列出该用户有权访问的功能菜单；用户点击某个功能菜单后，系统根据部门 ID(从单位角度)、数据周期 ID(从时间周期角度)、数据对象 ID 来决定该用户能够访问的哪些单位哪些时间的表、文档、项目等等。最后，系统会根据操作级别 ID 来决定用户在具体功能页面上能够对这些数据进行怎样的操作(增删、修改、查询等等)。

结论 通过对 RBAC 模型进行细粒度的扩充，在单位、功能、数据等维度对模型进行了细化，实现了大型信息系统的多级管理员体系，使得系统的功能及数据的控制更加灵活，用户和角色的管理更加方便。该模型在某省级电信运营商的综合信息系统中进行了实践应用并取得了很好的效果，受到用户广泛的好评。

参 考 文 献

1 Sandhu R S, Samarati P. Access control: principles and practice [J]. IEEE communications, 1994, 32(9): 40～48

2 Sandhu R S, Coyne E J, Feinstein H, et al. Role-Based Access Control Models. IEEE Computer, 1996, 29(2): 38～47

3 Sandhu R, Bhamidipati V, Coyne E, et al. The ARBAC97 model for role-based administration of roles: Preliminary description and outline [A]. In: Proceedings of the Second ACM Workshop on Role-Based Access Control [C], Fairfax, Virginia, USA: ACM, 1997. 41～50

4 Sandhu R S, Ferraiolo D F, Kuhn D R. The NIST model for role based access control: Towards a unified standard. In: Proc. of the 5th ACM Workshop on Role Based Access Control, New York, NY: ACM Press, 2000. 47～63

5 Ferraiolo D F, Sandhu R S, Gavrile S, et al. Proposed NIST standard for role-based access control. ACM Transactions on Information and Systems Security, 2001, 4(3): 224～274

(上接第 276 页)

件表示采用的规范函数，在原有规约匹配的相关理论基础上，对函数名做了一定的规范化处理，大大提高了构件的查准率。以它为基础，已经搭建了包括 ERP、OA 等多个企业应用系统。实践表明，该构件库的检索效率令人满意。今后将在两个方面继续进行优化，一方面减少构件的入库代价，逐步实现 Java 构件的自动化分类和表示，另一方面提供构架构件，通过用户修改构架配置，系统自动检索匹配满足构架的构件，实现企业应用系统的快速成型。

参 考 文 献

1 杨芙清，梅宏，李克勤. 软件复用与软件构件技术[J]. 电子学报，1999(2)：68～75

2 Mili R, Mili A, Mittermeir R T. A Survey of SoftwareStorage and Retrieval [J], Ann. Software Eng., 1998, 5(2): 349～414

3 Morel B, Alexander P, Member S. SPARTACAS: Automating Component Reuse and Adaptation [J]. IEEE Trans. Software Eng., 2004, 30(9): 587～600

4 张世琨，张文娟，常欣，等. 基于软件体系结构的可复用构件制作和组装[J]. 软件学报，2001，12(9)：1352～1359

5 马亮，等. 基于规约匹配的构件检索[J]. 小型微型计算机系统，2002，23(17)：1153～1157