支持互操作的构件库管理系统①

An Interoperable Component Library Management System

范 菁　白 杰　熊丽荣　（浙江工业大学 软件学院 浙江 杭州 310023）

摘　要：本文针对构件库互操作问题提出了基于软件适配器的构件库互操作模型，并应用此模型设计和实现了相应的构件库管理系统。文章介绍了构件库管理系统的组成及功能，重点阐述了互操作模型的设计思想，并通过实例对模型如何支持互操作的实现方法进行了说明。

关键词：构件库 构件库管理系统 互操作 刻面分类 软件适配器

软件复用为避免软件开发过程中的重复劳动提供了解决方案，可以提高软件开发的效率和软件的质量，而软件构件技术是实现软件复用的关键技术[1]。企业在项目实践中提取积累的构件、商业 COTS（Commercial Off The Shelf）构件、其它软件组织开发的构件以及中间件等面向非具体应用的软件都是软件构件的存在形式，这些软件构件为推动软件开发面向大规模复用方向前进提供了基础和条件。构件库是支持软件构件化开发的一个重要基础设施，它提供对软件构件进行描述、分类、存储和检索等功能[2]。

随着构件库技术的发展，REBOOT 构件库[3]、青鸟构件库系统[4]、ComponentSource、上海构件库等众多的大型构件库系统相继出现，在这些构件库中存储了大量的构件，如何在不同构件库间共享资源和无缝互操作成为人们越来越关注的问题[2]。

RIG 的一个技术委员会提出的数据模型 BIDM[5]、OMG 组织提出的 RAS 规范[6]等都旨在建立一个构件库所共同遵循的基本数据模型来解决这一问题。文献[7,8]在这方面都做出了非常好的尝试。但由于数据模型是在系统建模阶段就确定下来的，在系统完成后不能够轻易改变也不容易扩展。如何能保证在不改变现有模型的基础上实现互操作是一个亟待解决的问题。本文设计并实现了一个构件库管理系统，并通过软件适配器来实现构件库间的互操作。

1　构件库管理系统的组成及功能

本文设计的构件库管理系统主要分为 3 部分，即构件管理子系统、用户管理子系统和系统管理子系统。各子系统结构及相互关系如图 1 所示。

（1）构件管理子系统 主要实现构件入库、构件维护、构件提取和构件反馈等功能。

构件入库支持构件提供者在线提交构件的各种信息和实体，以及根据这些信息生成构件描述文档等。用户在提交构件时，要为构件填写基本属性，关联一组关键词，在各个刻面关联一组术语，并指定构件之间的关系，分别为属性值、关键词、刻面等分类与检索以及构件间关联导航提供支持。构件维护包括在构件库中检索、修改和删除构件，支持基本属性检索、关键词检索、刻面分类检索和构件间关联导航等检索方式。系统共定义了构件类型、功能、应用领域和使用环境四个刻面，对刻面术语进行层次编码，采用树匹配方式进行刻面检索。构件提取是将构件所有相关信息以构件包的形式导出构件库。构件反馈是以自然语言的方式对构件进行评分、评价以及对反馈进行统计分析等。

（2）用户管理子系统 主要实现用户信息的注册、修改和注销，用户登录与登出，用户权限管理，组织信息的注册、修改和删除等功能。采用基于角色的用户访问控制机制（RBAC）管理用户权限，为用户提供了灵活可靠的操作机制。

（3）系统管理子系统 包括术语空间管理、适配器

① 基金项目 浙江省科技计划面上项目《基于 Web Service 封装的信用构件库》（2007C21011）

管理和系统配置。术语空间管理是对刻面分类中的术语、同义词及术语和同义词之间的关系进行管理。系统配置包括设定目录、系统邮件和 FTP 等信息。

适配器是系统支持构件库互操作的一个功能模块，是与其它构件库共享构件资源的转换接口。文章第二节将详细介绍适配器的设计和实现。适配器管理主要是通过增加和删除适配器种类，从而支持与不同的构件库进行互操作。

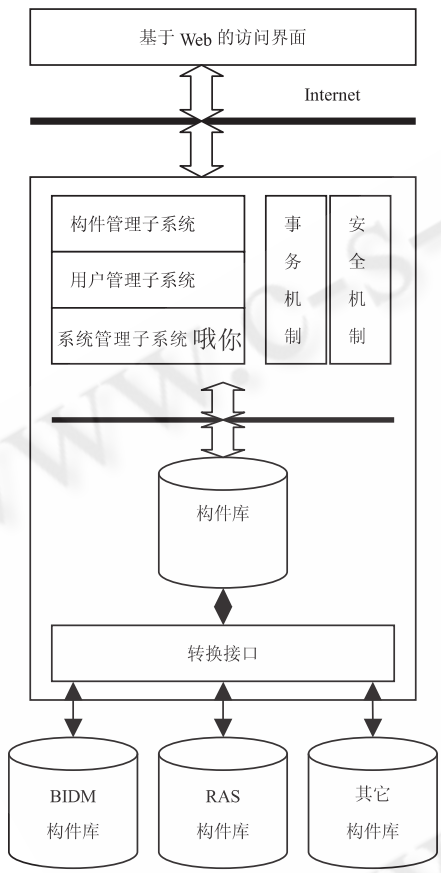

图 1　构件库管理系统体系结构

2　构件库管理系统互操作

2.1　互操作模型

互操作模型主要对支持构件库间互操作的软件模块进行描述。由于在构件库管理系统中增加了转换接口，可以通过管理实现了转换接口功能的适配器来完成与其它构件库之间的互操作。

图 2 给出了基于软件适配器的互操作模型。

模型中支持互操作的构件库系统 CL－I 在构件库管理系统里增加了转换接口即适配器服务层，主要提供两个功能：一是定义适配器接口，接口中包括适配器必须具备的导入导出构件包等功能；二是对实现接口的各适配器进行管理，包括载入和卸载适配器等。CL－I 通过具体的适配器与其它构件库系统进行构件共享等互操作。

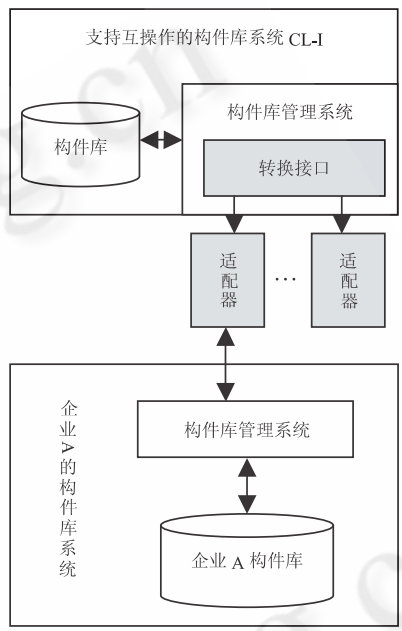

图 2　构件库管理系统互操作模型

此模型假定构件库系统中存在以下条件：

（1）构件库系统支持以构件描述文件的形式对构件进行描述，并发布了构件描述文件的定义文档。构件描述文件是记录构件属性值的文档，一般由构件基本信息、构件接口、构件分类、构件间关联等信息组成。

（2）构件库系统定义了针对自身构件库数据模型的构件包格式。构件包应至少包含构件描述文件。

在以上假定的基础上，CL－I 和构件库 A 的互操作可以按以下步骤进行。

（1）CL－I 获取构件库 A 的构件描述定义文档，根据构件库 A 的构件包格式编写适配器 A，适配器 A 实现了 CL－I 的适配器服务层定义的接口，此程序的导出接口将 CL－I 中的构件相关信息导出成构件库 A 定义的构件包格式，导入接口将构件库 A 定义的构件包格式转换成 CL－I 接收的格式。

（2）CL－I 将适配器 A 部署在构件库管理系统中，适配器服务层负责将它发布成外界可访问的接口。

（3）构件库 A 与适配器 A 通信，交换构件，实现互操作。

2.2 实例说明

下面就以本文构件库管理系统中具体例子的实现来说明此模型。

系统定义的构件库核心数据结构如图 3。

这个数据模型是围绕着 Component（构件）组织起来的，Resource（资源）是组成构件的实体文件，Term（术语）是描述构件的刻面分类术语，它与 Synonym（同义词）一起组成了 Facet（刻面）中各个刻面的术语空间。User（用户）和 Organization（用户所属组织）则是构件的提交者或复用者。

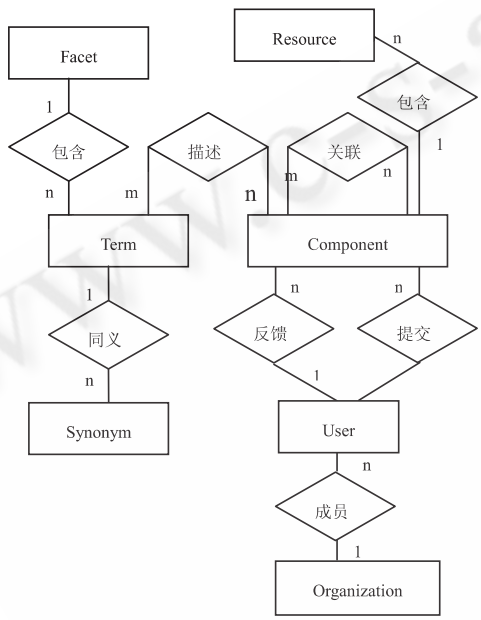

图 3　构件库核心 E－R 图

系统定义的适配器接口如下。

```
Interface ComponentAdapter {
    Object marshal(String compID);
    Object unmarshal(Object comp);
}
```

marshal 接口将构件 ID 对应的构件导出为目标构件库的构件包文件，返回值可以是二进制输出流或文件地址；unmarshal 接收目标构件库的构件包文件，经过解析后导入构件库中。

系统现与一个基于 BIDM 数据模型的构件库进行互操作，已知其发布的构件描述文档结构如图 4，定义的构件包格式如图 5。

图 4 中 asset、element、library、organization 等元素分别对应 BIDM 模型中的 Asset、Element、Library、Organization 等类，具体内容可参见文献[5]。

图 5 中的 <构件 ID>.xml 是构件描述文档。

实现适配器程序时遵循了以下原则：

（1）映射含义相同的实体。若目标实体在系统中不存在，则硬编码在程序中或从外部配置文件读取；若源实体在目标实体中不存在，则忽略。

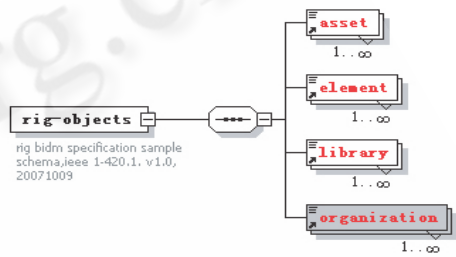

图 4　rig－objects. xsd

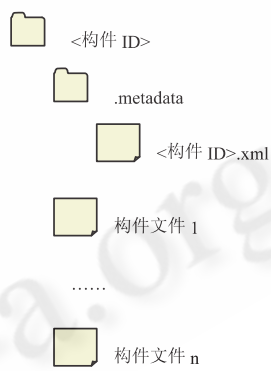

图 5　构件包格式

（2）含义相同的实体中映射含义相同的属性。若源实体不存在目标实体某属性，则根据目标实体定义文档源取默认值或置空值；若目标实体不存在源实体某属性，则忽略。

（3）为弥补映射过程中的信息丢失情况，分别将原构件描述文档与生成的构件描述文档一起导出（导出接口）、保存（导入接口）。

针对以上信息，编写适配器程序 BIDMComponentAdapter，它实现了 ComponentAdapter 接口。

（1）marshal 方法中利用构件库管理系统提供的 API 和 Java JAXP API 根据构件 ID 查询得到 Component 对象及与其相关的构件资源对象、提交者组织对象，按

照图 4 定义的格式生成构件描述文档，将这些对象分别映射成 asset、element 和 organization 等节点。由于系统是单库结构，没有 library 节点对应的实体，根据原则(1)，程序从配置文件读取相关信息。把生成的构件描述文档、原构件描述文档和构件实体文件一起按图 4 中的结构打包，原构件描述文档与生成的文档在同一目录下，文件名格式为 <构件 ID - CLI >. xml。返回生成的构件包的输入流对象。

(2) unmarshal 方法正好相反，它接收并解析传入的构件包，利用上述 API 解析构件描述文档并生成 Component 对象及相关对象，映射规则与 marshal 相同。保存构件包中的描述文档和构件实体文件。

系统使用 Spring 作为适配器服务层的部署管理工具，并将上述适配器发布成 Web Service。其配置代码如下：

```
<bean id = " bidmAdapter"
class = " ccl. component. spi. BIDMComponentAdapter" / >
<bean id = " bidmAdapterService"
class = " org. codehaus. xfire. spring. remoting. XFireExporter" >
<property name = " serviceFactory"
ref = " xfire. serviceFactory" / >
<property name = " xfire" ref = " xfire" / >
<property name = " serviceBean " ref = " bidmAdapter" / >
<property name = " serviceClass"
value = " ccl. component. spi. ComponentAdapter" / >
</bean >
```

当构件库系统部署在本地时，通过 http://localhost:8080/ccl/component/BidmAdapterService?wsdl 获得 WSDL 文档，如图 6 所示。

在获取 WSDL 文档后，就可以生成客户端程序调用适配器提供的互操作服务。

2.3 优势与不足

基于软件适配器的互操作模型有以下的优势：

(1) 此模型提供部署时级别的互操作扩展与支持，不必对构件库模型作任何更改。

(2) 此模型可以通过增加适配器与更多的构件库进行互操作，具有良好的可扩展性。

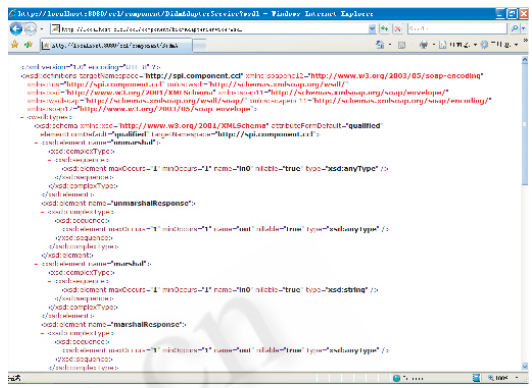

图 6 获取适配器服务的 WSDL 文档

(3) 此模型倡导构件库之间采用 XML 文件存储构件描述信息，并提出构件库应有独自的构件包结构，为构件库之间进行互操作提供了一种实现思路。

但模型也存在以下的不足：

(1) 对构件库系统设定条件过于复杂，转换接口不容易统一，实现困难。

(2) 适配器编写复杂，自动化程度不高。而且容易出现信息无法转换的情况。这种情况下就要根据构件语义以置空或替代的形式硬编码在适配器中。

3 总结

构件库是支持软件复用的重要基础设施，它负责管理构件从发布到提取组装再到反馈维护的整合过程。文中提出的构件库管理系统支持对构件进行发布、提取、分类与检索、存储、反馈与评价等，并采用了基于软件适配器的互操作模型来对支持构件库互操作，可以在不改变已有构件库数据模型的基础上完成数据共享和互操作。

参考文献

1 杨芙清，梅宏. 软件复用与软件构件技术. 电子学报，1999，27(2)：68 - 75，51.

2 潘颖，赵俊峰，谢冰. 构件库技术的研究与发展. 计算机科学，2003，30(5)：90 - 93，156.

3 Morel J M，Faget J. The REBOOT Environment，BULL S. A. Rue Jean JAURES，F - 78340 LESCLAYES - SOUS - BIOS，France.

(下转第 5 页)

(上接第 21 页)

4 Yang FQ, Mei H, Li KQ, Yuan WH, Wu Q. An Introduction to JB3 System Supporting Component Reuse. Computer Science, 1999, 26(5): 50 - 55 (in Chinese with English abstract).

5 Reuse Library Interoperability Group. RIG Basic Interoperability DataModel (BIDM): [RPS - 0001], 1993.

6 Object Management Group. Reuse Asset Specification. 2005. http://www.omg.org/docs/formal/05 - 11 - 02.pdf.

7 李琰, 邹艳珍, 潘颖, 谢冰, 孙家□. 一种可扩展的构件库数据模型. 计算机科学, 2006, 33(5): 282 - 286, 封四.

8 潘颖, 刘杨, 谢冰, 杨芙清. 支持管理在线构件的基本构件描述模型. 电子学报, 2003, 31(12A): 2110 - 2114.