基于 xml 技术的软件构件表示与检索

Xml – based technology Software Component Representation and Retrieval

孙念民 李万龙 郑山红 赵 辉 (长春工业大学 长春 130021)

摘要:软件构件库作为构件复用的基础物质仓库,扮演了极为重要的角色,而可复用构件的表示、分类、检索是构件库的关键。本文在构件库系统的构件分类与描述方案的选择设计中,分析了构件的刻面分类方法,提出采用XML技术实现构件的刻面描述方案,并从技术实现、发展演化等方面对该方案作了较详细的分析,之后参照国内外相关学术成果,结合XML技术特点,给出了具体的刻面设计及描述实现,从而建立起软件构件库系统的构件数据模型。

关键词:可复用构件 构件库 分类 刻面

1 引言

提高软件的生产效率和质量,软件复用是一条切实可行的途径。近年来,由于得到了面向对象等新技术的支持,软件构件复用技术已成为当前软件复用研究的焦点。基于构件的软件开发(CBSD)将软件开发的重点由原来的软件编程转移为利用已有构件组合集成新的软件系统。构件库理论研究的重点是构件的分类与检索,即研究构件分类策略、组织模式、检索手段和构件相似性分析。

在可复用构件库中存储、查询、获取构件是复用技术的关键技术之一。尽管可复用对象复用的前景十分乐观,但不得不面对分类、存储、获取可复用构件的问题。只有合适的分类方法才能更好地满足检索的需要,而检索是分类的目的。可复用构件的分类应该满足下面的标准[1]:

(1) 可以容纳不断扩大的构件集合,这是大部分软件组织的特点之一;

(2) 支持寻找相似构件的能力,而不是精确匹配;

(3) 支持跨领域寻找功能等同构件的能力;

(4) 准确并且有较强的描述能力;

(5) 易于维护,即增加、删除、更新分类的结构和词汇不需要对构件重新分类;

(6) 对于管理人员和用户来说都易于使用;

(7) 服从自动化的过程。

2 构件的刻面分类方法

刻面分类方法将关键词(术语)置于特定的语境中,从而避免了关键词的杂乱无章,而且它通过从特定的反映构件本质特性的视角(刻面)来观察要分类的项,从而出现了更加精确和准确的分类。多数专为构件的检索而设计的工具都采用了这种方法,北大西洋公约组织(NATO)标准推荐在构件库中采用刻面分类模式(Faceted Classification scheme),对构件的分类使用一组{刻面,刻面术语}对,也称为描述符。每个刻面中有一组术语,术语间由于有一般特殊关系和同义词关系而形成结构化的术语空间。构件的描述术语仅限在给定的刻面之中选取,在术语空间中游历可以帮助复用者理解相关领域。

3 xml 技术在软件构件的分类与检索中的应用

3.1 xml 技术简介

XML 是 W3C[2](World Wide Web Consortium)推荐的一种可置标语言,它的设计动机是在网络上规范化文档传输,它在本质上与程序设计语言一样,是具有严格语法定义的形式语言,但较一般的形式语言简捷和通用。与 HTML 一样,XML 也源自 SGML(Standard Generalize Markup Language),它保留了 SGML 80% 的功能,使复杂程度降低了 20% [3],尽管如此,

XML 却有着 HTML 语言所欠缺的巨大伸缩性与灵活性。XML 不再像 HTML 一样有着一成不变的格式。XML 实际上是一种定义语言，即使用者可以定义无穷无尽的标记来描述文件中的任何数据元素，从而突破了 HTML 固定标记集合的约束，使文件的内容更丰富更复杂并组成一个完整的信息体系。XML 主要的优点有：良好的可扩展性；严格的语法要求；内容与形式分离；不同系统间方便的信息传输（通用数据交换格式）；较好的保质性。

3.2 应用 xml 技术对构件的表示

参照 NATO 关于可复用构件的开发、可复用构件库的管理指导标准，以及 REBOOT 环境，结合国内青鸟系统的构件刻面分类描述方法，考虑到构件技术向多层次、简化安装的方向发展以及系统需求向工 Internet、动态演变、管理自动化等方面的发展趋势，结合 XML 语言自身的特点，系统设计了如下刻面与子刻面。

（1）构件形态：表示形式，种类，在开发中的抽象层次；

（2）应用环境：硬件环境，软件环境；

（3）应用领域应用领域，备注；

（4）实现功能：功能分类，功能描述，关键词族，接口族；

（5）文件属性：文件名，文件大小，制作日期，版本，作者。

在这些刻面中：

“构件形态”刻面是指构件的形式和状态等集合。其中，“表示形式”元素是用来描述构件内容的语言形式或媒体，构件库中任何构件都以一定的形式存在，因而必然有其外在的表现形式。根据实际应用情况，将其术语空间指定为枚举值：DLL. EXE，源码、图形、其它；“种类”元素是指定构件成为最终产品时的应用种类，根据实际应用情况，指定为枚举值：COM/COM +，ActiveX（OCX）、. NET，VCL/CLX，JavaBean，CORBA，ClassLibrary，Others；“在开发中的抽象层次”元素是构件相对于软件开发过程阶段的抽象层次。根据实际应用情况，指定枚举值为：分析、设计、编码、测试其他。

“应用环境”刻面是使用（包括理解/组装/修改）该构件时必须提供的硬件和软件平台。如所需的特定的硬件环境、操作系统、数据库平台、网络环境和编译系统等。构件库中任何构件都必须依赖于一定的使用环境才能得到复用，同时作为扩展，在软件子刻面中增设“Other”元素，用于记录附带信息，如构件依赖关系等等。

“应用领域”刻面是该构件可能被使用到的应用领域（及其子领域）的名称。这里的领域，指共享某种功能性的系统或应用程序的集合。构件库中任何构件都有一个适用的领域，一般的通用构件适用领域为通用，根据实际应用情况，指定为枚举值：通用、商业、科学、人文、政府、财政、制造、航天、保健、法律、传媒、银行、办公、运输、安全、其它等等；“备注”元素可以说明一些附带情况。

“实现功能”刻面是该构件在原有或可能的软件系统中所提供的软件功能集。构件库中任何构件都必须提供一种或多种功能。根据具体情况，其“功能分类”元素指定为枚举值：图形图像、网络通信、数学、数据库、用户界面、系统、驱动程序、适配器、容器、业务逻辑、协调任务、代理程序、协作、字处理、文件操作、其他等等；“功能描述”元素则提供文字性的描述，也为将来语言的智能分析提供一种扩展；“关键词族”、“接口族”子刻面是功能描述信息中重要的部分，根据实际情况，允许多个关键词和接口。

“文件属性”刻面是构件的实际提交载体属性，可以将需要的文件（如：构件实体、使用说明、帮助文件、IDL 文件等）打包提交，因此其文件名不依赖构件名；对作者信息描述便于构件信息的反馈、更新及使用者与作者的交流等，故对作者信息有较详细的描述。

以上五个刻面彼此间相互隔离，而且比较充分地体现了构件与复用相关的特性，能较好的适应构件库今后的发展。四个刻面与其子刻面，分别对应 XML 文件的元素和子元素。同时，设计中尽可能多的对元素术语指定枚举值（并在具体实现的系统界面中指定下拉列表选择），在一定程度上减少了软件术语差异产生的影响。

3.3 基于 XML 的构件描述文档的树形表示

DOM（Document Object Model，文档对象模型）是一种处理 XML 文档的好方法，它是通过在内存中构造 XML 文档的树形表示模型来处理 XML 文档的。在此模型中，每个独立的数据项被视为一个节点（node），所有的子元素或其中的文本被称为子节点。XML 文档中的元素、元素属性、文本等都是此模型的一个节点。

XML 文件数据结构的定义与校验采用 XML Schema。我们可以用以下带文档类型定义的 XML 文档(Component.xml)简要地来定义构件的结构:

```
<? xml version = "1.0" encoding = "gb2312" ? >
< Document xmlns: xsi = " http://www. w3. org/2001XMLSchema - - instance"
xsi:noNamespaceSchemaLocation = " F: \My Programs \XML\Component. xsd" > <Component >
    <! - - 构件的名称 - - >
     <Name > Text - to - Speech </Name >
     <! - - 制作或提供该构件的单位或个人的名称,以及联络地址等相关信息 - - >
      <Author >
       <Name > Rodney White </Name >
       <Email > rdwhiteLmsn. com </Email >
      </Author >
     <! - - 构件的制作日期 - - >
     <BuildDate > 2006 - 10 - 05 </BuildDate >
    <! - - 构件的入库日期 - - >
<AdoptedDate > 2007 - 02 - 12 </AdoptedDate >
    <! - - 构件的版本号 - - >
    <Version > 1.0 </Version >
<! - - 使用(包括理解/组装/修改)构件时必须提供的硬件和软件平台 - - >
<ApplicationEnvironment >
 <HardwarePlatform > Intel X86
 </HardwarePlatform >
 <SoftwarePlatform > Windows 2000
 </SoftwarePlatform >
 <SoftwarePlatform > Windows XP
 </SoftwarePlatform >
</ApplicationEnvironment >
<! - - 构件原来或可能被使用到的应用领域(及其子领域)的名称 - - >
<ApplicationDomain > CAI </ApplicationDomain >
 <! - - 构件在原有或可能的软件系统中所提供的软件功能集 - - >
    <Functionality >
     Transform simple text to speech
    </Functionality >
<! - - 构件相对于软件开发过程阶段的抽象层次 - - >
 <LevelOfAbstraction > Coding
</LevelOfAbstraction >
 <! - - 用来描述构件内容的语言形式或媒体 - - >
 <Representation > Visual C + + . NET using WTL
 </Representation >
    <Size unit = " megabytes" > 3.2 </Size >
 <DevelopmentTool > Microsoft Visual C + + . NET
 </DevelopmentTool >
 <FileProperty >
  <FileName > Grant </FileName >
  <FileSize unit = " MB" > 20 </FileSize >
 </FileProperty >
</Component >
</Document >
```

3.4 .NET 中实现基于 XML 的构件检索

XPath (XML Path Language, XML 路径查询语言)提供了简单的语法来方便地选取 XML 文档中的节点。利用 XPath,我们可以指定类似目录的路径,以及依照路径设定条件来获取内容节点。目前 XSLT. XML DOM 和 XPointer 都支持 XPath 规范。XPath 查询是以位置路径(Location Path)表达式来指定的。利用位置路径我们可以选择一组与内容节点相关的节点。

大致流程如下:

(1) 载入 XML 构件信息文档

为了便于利用 XPath 来查询构件信息文档,我们可以利用 XPathDocument 来载入 XML 文档,示例代码如下:

```
Dim Doc As New
XpathDocument ( Server. MapPath ( " Compo - nent. xml" ) )
```

(2) 验证 XML 构件信息文档

载入 XML 构件信息文档之后,我们可以对其进行合法性验证,这一步是利用 XMLUalidatingReade 来提供的合法性检查功能,合法性检查通过则执行第 3 步。

(3) 检索 XML 构件信息文档

XML 构件信息文档验证通过之后,我们就可以开始检索构件信息了,检索构件信息主要是利用 XPath-

Navigator, XpathNodeIterator 这两个类,检索时我们可以利用 XPath 查询表达式提供的强大查询功能,示例代码如下:

```
Dim myNav As XpathNavigator
myNav = Doc.CreateNavigator()
Dim myIter As XpathNodeIterator
myIter = myNav.Select("/Component/Name")
While (myIter.MoveNext())
1stName1.Items.Add(myIter.Current.Value)
End While
```

(4) 将检索得到的构件信息定制输出

我们还可以根据需要将检索得到的信息定制输出,要实现构件检索信息的定制输出我们主要是利用 XslTransform 类和 XmlTextWriter 这两个类。利用 XmlTextWriter 类我们可以将构件检索信息保存为 XML 文档,接着还必须根据定制输出的要求编写相应的 XSL 文档,最后就可以使用 XslTransform 类将 XML 构件检索信息文档转换成预想的格式。若要实现 XML 构件信息文档与 SQL Server 数据库的数据互换,我们可以利用 ADO.NET 提供的 DataSet 数据集对象,DataSet 对象透过 SQL 数据适配器对 SQL Server 数据库进行操作。

4 结束语

软件复用技术将促进软件产业的变革,使软件产业真正走上工程化、工业化的发展轨道。软件复用将造成软件产业的合理分工,专业化的构件生产将成为独立的产业而存在,软件系统的开发将由软件系统集成商通过购买商用构件,集成组装而成。

参考文献

1 Prieto – Diaz . Implementing Faceted Classification for Software Reuse [J]. Communication, ACM, 1991: 88 ~ 97.

2 W3C. http://www.w3.org/

3 XML 中国论坛. XML 实用进阶教程[M],北京:清华大学出版社. 2001:21 – 25.

4 杨芙清,软件复用及相关技术[J],计算机科学,1999, 26 (5) :1 – 4.