数字图书馆中基于 Java 的用户界面组件库研究与开发*)

来 强[1] 邢春晓[2]

(清华大学图书馆 北京100084)[1] (清华大学计算机科学与技术系软件所 北京100084)[2]

Research and Development of Java Based User Interface Component Library in Digital Library

LAI Qiang[1] XING Chun-Xiao[2]

(Library of Tsinghua University, Beijing 100084)[1]

(Software Institute, Department of Computer Science and Technology, Tsinghua University, Beijing 100084)[2]

laiqiang@lib. tsinghua. edu. cn

Abstract It is one of the most important task in digital library to develop a universal and customized user interface for search and retrieval in heterogeneous, distributed environment. In this paper, we briefly introduce working principles and workflow of SiteSearch system developed by OCLC. Based on analyzing the functions of this system, we study and design composing elements and implementation mechanism of the customized interface. By analyzing and illustrating the examples, we develop Java based component library for dynamically generating the customized interface in Chinese information platform of SiteSearch system. Finally, we make some remarks on the difference of development of this system with other similar systems.

Keywords User interface, SiteSearch system, Z39. 50 Protocol, Java based component library

随着互联网上数字化信息资源建设的发展，不同类型、不同载体的数据库不断增加，虽然用户获取信息的途径增多了，但是异构、分布数字图书馆的检索软件和用户界面也为用户检索和获取信息带来了巨大的困难，用户不得不使用多种检索工具或检索软件才能获得较为全面的检索结果。因此，人们迫切地期盼着能够在统一界面下同时检索各种数据库的检索系统。我们在分析和改进 SiteSearch 系统的基础上，设计和实现了为不同用户灵活定制适合个性化需求的用户界面 Java 组件库，为解决上述问题提供了一种有效的途径。

1 SiteSearch 系统简介

SiteSearch 系统是国际组织"在线计算机图书馆中心(OCLC)"开发的一个主要用于图书馆领域的信息检索和管理软件，它的主要特点是在一定程度上很好地解决了数字图书馆建设中的互操作问题，支持 Z39. 50协议，可以实现跨库查询和跨馆查询。它完全采用基于 Java 语言的软件开发平台，可以把所有的信息集成在一个按自身需求定制的电子资源环境中，能使各个图书馆不仅具有提供本地服务的功能，还能促进本馆与其他成员馆之间的读者服务协作。

目前，能够通过 SiteSearch 检索的大量信息来自各个支持 Z39. 50协议的数据库。不同的数据库服务器拥有不同的 IP 地址和端口号，以及许多各自的特征，需要有不同的连接方法和不同的用户认证与访问授权，SiteSearch 带有一套十分优秀的可以同时访问所有数据库的通用界面，并能够将跨数据库检索到的结果集进行合并、排序和剔重。而且，通用界面的内容还会根据不同的用户资料动态显示，这些动态呈现界面显示的功能都是通过构造 Java 应用来实现的。通过修改相应的配置文件和 Java 类还可以对一个用户群体的通用界面进行修改和定制。

2 系统的工作原理

当今 Internet/Intranet 技术已经十分普及，信息通过 WWW 发布是一种被人们普遍接受的方案，Web 也成为因特网的标准用户界面。采用浏览器/服务器(Browser/Server)模式，可以充分利用网络的分布式特点，建立一个超大规模、开放性、分布式的数字信息资源网络体系结构。通过它能实现复杂信息的加工存取和海量信息存贮的功能。同时 WWW 网络系统又具有良好的兼容性、互操作性、可扩充性和跨平台性能，结合数据库技术可以很方便地实现检索功能。

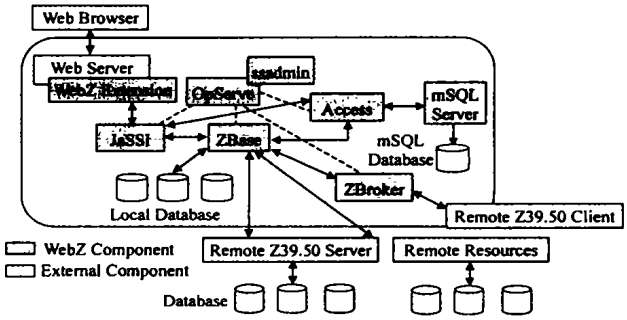

图1 SiteSearch 系统的体系结构

SiteSearch 系统利用 Web 的这种优势，使用目前普遍采用的动态 Web 方案——由客户端浏览器、Web 服务器、应用服务器、后台数据库构成的四层 Web 服务架构。SiteSearch 的一大创新是改变了以往用户访问 Z39. 50服务器必须使用 Z39. 50客户端的方式，采取了通用的 Web 浏览器客户端的方式，因此在服务器端增加了一个 Z39. 50网关，即 WebZ Ex-

*)本项目得到国家重点基础研究发展规划资助项目(G1999032704)资助和清华基础研究基金资助。邢春晓 博士，副教授，主要研究方向为数字图书馆、分布式多媒体数据库等。

tension 组件，实现输入信息从 http 协议到 Z39.50协议的转换。

用户浏览器输入的信息进入 SiteSearch 系统，首先需要经过 Z39.50网关，然后传入 WebZ 模块进行处理。WebZ 模块作为 Web 服务器的扩展，主要包括 Access，JaSSI，ZBase，OpServe 等几大组件。它实现系统的前台人机交互界面，接收用户的查询请求，将请求发送给本地和远程的数据库，并将返回结果呈现给用户。用户查询信息时，在 WebZ 提供的 WWW 界面中输入检索词，然后经过 JaSSI 和 Zbase 组件的处理和传递，最后把信息提交给后台数据库检索，并把数据库返回的符合条件的数据进行一系列的格式转换，最终以浏览器可识别的格式显示出来。

SiteSearch 系统中各个组件之间的联系如图1所示。当用户执行一个标准化的检索时，WebZ 对输入输出信息处理的过程大致可以分为如下6个阶段：

·信息和命令通过 URL 传送到 WebZ Extension 网关；

·WebZ Extension 定义会话，继续把信息和命令向下传送；

·JaSSI 组件解析收到的 URL，并动态调用相应的 JAVA 包，如，使用 VERBS（一类特殊的 JAVA 包）专门处理与 Z39.50通信有关的事务，使用 Widgets（名称/数值对）把系统执行过程中需要的参数提交给系统；

·JaSSI 组件针对本次检索建立一个新线程，生成 Z39.50 查询表达式；

·Zbase 组件接收 Z39.50查询表达式，将其向下传递给相关的数据库，并等待结果；

·在得到 Z39.50协议支持的检索结果后，Zbase 组件把结果传递给 JaSSI 组件；

检索结果被格式化显示在输出页面上，呈现给用户。

3 用户界面 JAVA 组件库的设计

3.1 实现机制与设计原则

SiteSearch 系统最大的特色是实现了用户对分布在全世界的遵守 Z39.50协议的信息的跨库检索功能，包括对检索结果集的合并、排序和剔重等。由于系统的用户众多，每个用户的检索兴趣都不尽相同，系统需要为不同的用户提供与之检索兴趣相适应的独特界面，因此，WebZ 模块的用户界面采用了由 Java 程序处理动态界面的方式。

ORG.oclc.gadget 包实现了 WebZ 模块的动态界面显示功能。这个包就好象一个工具箱，其中包含了很多构建 HTML 页面的小工具 gadget，实际上，这些 gadget 共同构成系统的用户界面 Java 组件库。HTML 的页面元素几乎全部通过这个组件库动态构造。gadget 在构造页面的时候，不仅需要用户输入信息，还要读入存储在系统预置的.ini 配置文件中的数据。

作为整个系统开发 API 的重要组成部分之一，gadget 充分利用了 Java 语言面向对象的特点，利用对象封装成员变量和方法，实现数据和操作的封装及信息的隐藏。同时它还通过类的继承机制和动态的接口模型，子类可以使用父类所定义的方法，实现了代码的多次复用。用户输入信息和存储在.ini 配置文件中的信息由相关 JaSSI 包中的 Java 类读入，经过处理传送给数据库服务器，取回的数据则经过各种具有不同功能的 gadget 处理后通过 HTML 页面显示给用户。Gadget 的本质是构成用户界面的 Java 组件，开发新的 gadget 即是对用户界面 Java 组件库的扩充。下面，以用户登录系统的过程为例，详细阐释动态界面的产生原理和设计原则。

通常情况下，当用户用 Web 浏览器登录访问 SiteSearch 系统时，WebZ Extension 网关通过 URL 传递的命令和参数传递给 WebZ，WebZ 的 OpServ 组件会为用户建立一个新的进程，指定一个进程号和与之对应的一个用户状态对象（UserStateObject），接下来 JaSSI 组件用 Action 类和 RequestManager 类解析并处理用户请求，同时为用户授权（authorization）。如果授权成功，系统会为用户分配与其用户身份相对应的数据库资源列表。经过授权，WebZ 的 Access 组件还要对用户进行资格审查（authentication），校核用户所输入的用户名和密码是否合法，然后负责处理包含在 URL 中的 next 命令的 RequestManager 类向浏览器传送用户请求的第一个页面——homeframe.HTML。这个页面带有框架，包含 toolbar.HTML 和 dblist.HTML 两个需要动态生成页面，浏览器再次向 WebZ 请求这两个页面。其中 dblist.HTML 页面中含有很多只有通过 JaSSI 组件解释才能生成的 HTML 源代码。这些源代码中包含了 gadget，要运行这些 gadget 并使用户看到它们构造的界面，需要以下几种十分重要的 Java 元素：

1）JavaPage 接口　要在 HTML 文件中嵌入可执行的 Java 类，首先需要在 HTML 文件中的第一行使用〈oclc-app〉〈/oclc-app〉元标记，并在其中指定指令类别（type）属性为 JavaPage 接口，例如，显示 dblish.HTML 页面的第一行是：

〈oclc-app type="JavaPage"
name="dblistscreen"〉〈/oclc-app〉

还要为 name 属性赋值"dblistscreen"，使得 dblist.HTML 页面与 dblistscreen 类相关联，这样"JavaPage"就能使 dblist.HTML 页面中加入的用于产生动态显示特性的 Java 类生效了。

2）UserStateObject 类　众所周知，一般的 Client/Server 模式中都是无状态的 HTTP 连接，通常情况下，服务器端无法记录一个会话中的用户状态数据。然而，SiteSearch 中每个用户都拥有一些用户自定义的信息，如用户自定义的数据库组和查询结果记录等，这些信息需要有状态的连接来维护。因此，SiteSearch 在用户登录系统的时候会生成一个用以存储会话中相关的用户信息和历史状态的用户状态对象（UserStateObject），需要动态生成的用户界面中的很多数据都保存在这个对象中。

从数据结构上看，UserStateObject 对象是一个扩展哈希表，它的基本结构包括一个键（key）和与之相对应的键值（value）。在 SiteSearch 中，UserStateObject 对象用作记忆对象，用户进程中的数据通过它进行存储。每生成一个新的用户进程，就会同时生成一个相应的 UserStateObject 对象来存储该进程中的数据。在整个用户进程中，UserStateObject 对象是存储用户数据库列表、用户界面风格、检索结果等的对象。每一个用户进程都有其自己的 UserStateObject 对象。UserStateObject 对象使得 WebZ 可以维持一个持续的进程（stateful session），看起来好像用户在使用一个从浏览器到 WebZ 系统的专用连接。如果没有 UserStateObject 对象，用户进程就无法存在。

3）实体（Entities）　其本身不是 Java 类，而是一种 HTML 元素。在普通的 HTML 文件中，实体只是某些特殊符号的占位符，它以"&"开头，以";"结尾，如"¢"。在浏览

器中，这个实体显示为符号“@”。

在SiteSearch系统中，实体的作用得到了很大的扩展。它的内容可以包含Java类名，相当于一个容器，作为大段的HTML代码的占位符，从而生成动态界面。实体主要分为两类，一类是界面风格实体(StyleTable entities)，另一类是格式显示实体(Format Display entities)。

界面风格实体相当于特定用户界面显示风格的值的占位符。界面风格实体的语法构成包括三个部分，第一部分是“&StyleTable.”，第二部分是所引用的界面风格配置文件中的节(section)的名称，第三部分则是该节中某个具体字段的名字，例如，出现在homeframe.html页面中的实体“&StyleTable.pages.toolbar;”代表界面风格配置文件pagesFrame.ini(或者pagesNoFrame.ini)的pages节中toolbar字段的取值。如果用户的工具条页面的名称改变了，则无需一一改动其他页面中各个指向工具条页面的链接，只要修改界面风格配置文件中的toolbar值即可。当需要作大量的用户页面的更新时，只需修改存储在配置文件中的某几个字段的值，这是非常方便的。

格式显示实体指向frameDisplayGadgets.ini(或者noframeDisplayGadgets.ini)文件中的对应值——某个具体的gadget。这种实体解析出的结果通常是一大段由gadget动态生成的HTML代码。在SiteSearch中，最典型的gadget的应用是用户数据库列表和检索结果列表的显示。与界面风格实体类似，格式显示实体的语法也包括三部分，它以“&FmtDisplay.”开头，接下来的两部分分别指向frameDisplayGadgets.ini(或者noframeDisplayGadgets.ini)配置文件中相应的节和字段。例如，“&FmtDisplay.FullRecord.gadget-;”是用以显示检索结果单条记录全部字段内容的full.html页面中的实体，它所指向的gadget可以在配置文件frameDisplayGadgets.ini中的FullRecord一节中的gadget字段查找到，即ORG.oclc.gadgets.FormatRecordsWithDuplicates，这个类根据在相关的记录格式显示.ini配置文件中已经设定好的参数显示返回结果记录的全部内容。

在SiteSearch中，实体的应用非常灵活，它既可以直接引用包含在系统配置文件中的参数，也可以引用JAVA组件，这就使开发人员可以比较轻松地为有特定兴趣的用户定制各不相同的用户界面。

4) RequestManager类　RequestManager类在JaSSI组件中用于处理HTTP请求并作出响应。当用户浏览器向装有SiteSearch系统的服务器请求页面时，JaSSI组件会调用RequestManager类来处理解析传入的URL，经过分析URL中所包含的数据之后，RequestManager类首先定位页面中的实体，然后在用户状态对象的实例中(即哈希表中)寻找与实体名相对应的键名，用键名对应的键值替换实体。键名通常对应某个.ini文件中某一节的某个字段的取值，这个取值可能是一个Java类也可能是一个HTML页面的相对URL。当页面中的所有实体都被替换完毕后，RequestManager就将页面发送给用户浏览器，于是用户看到了经过实体替换动态生成的网页。

Java组件是在以上介绍的4种元素的基础上发挥作用的。我们可以凭借系统的API定制新的JAVA组件库，即对原有Java组件进行修改或者创新开发，从而为系统的用户界面增添更多功能或风格样式。要创建新的JAVA组件，需要明确以下几点：

·新组件将要用在什么样的位置，包括它的功能定位和具体应用的HTML页面；

·应该非常具体地列出它所能完成的各项功能；

·设计之前要确认目前没有和将要设计的组件具有相同或相似功能的组件；

·新组件需要在多大的程度上介入系统；

·新组件需要的参数都有哪些；

·一定不要让一个组件执行过多的功能；

·为了使新组件尽量通用，需要对它的功能进行抽象，从而抽取出它的界面风格信息。

在设计的过程中，可以首先写出这个Java组件将要生成的HTML源代码，然后根据代码进行字符串的连接等工作。凡是需要重复完成的代码，都尽量使用循环来解决。此外，还要注意的是，完成后的新组件应尽可能简洁、功能单一，并符合面向对象程序开发的一般原则。

3.2　基于JAVA组件库的动态用户界面

SiteSearch系统中的所有页面几乎都使用Java类来实现动态页面生成，JAVA组件库是构造页面组成元素的最直接和最主要的Java程序，完成了大多数动态页面功能，为用户的跨库检索提供了极大的方便。

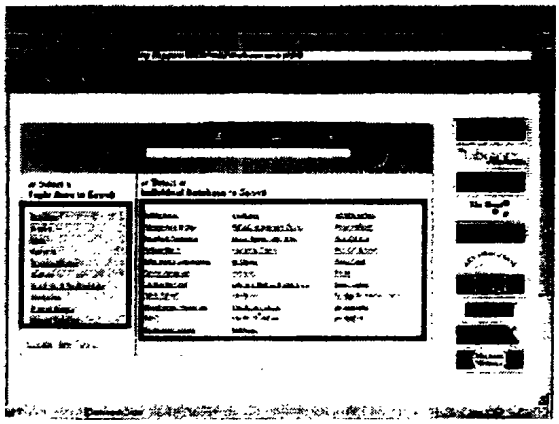

图2　dblist.html页面中小器件类功能

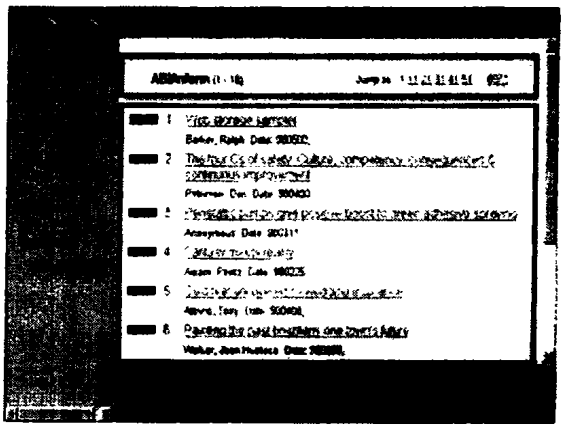

图3　resultsframe.html页面中JAVA组件库功能

由JAVA组件库生成的两个典型的页面分别是数据库列表页面dblist.html和记录显示页面resultsframe.html。dblist.html页面的源代码中包含了“&StyleTable.DatabaseList.gadget;”，这个实体调用DatabaseList类生成如图2所示的页面中央(底色为黄色)的数据库名称列表，这个列表的html源代码仅仅是一个由二十几个字符组成的实体

名，JAVA 组件库通过获取存储在 UserStateObject 对象中的用户信息和.ini 系统配置文件中的信息，实时构建了这个列表，显示给用户，用户点击列表中的数据库名，就可以进入到该数据库的检索页面。图2所示的另外两个带红框的区域也是由 JAVA 组件库动态生成的。

值得说明的是，在数据库如此丰富的今天，每个学科都有相当多的数据库可供选择，一个数据库会有不同的用户检索系统提供服务，数据库用户界面的使用便利性会对用户的选择使用产生极大的影响。用户界面的使用便利性就是对于不同使用背景的用户，数据库都能够提供恰当的检索途径、丰富的检索策略编制工具、充分的在线帮助信息和灵活的检索结果的输出方法。SiteSearch 系统提供了一系列的功能帮助用户在系统所集成的各个数据库中查找信息，这些功能在 Web 页面上的体现全都是通过 JAVA 组件库完成。

图3显示了检索结果页面中由 JAVA 组件库生成的检索记录列表和检索记录浏览导航链接。中央的大块（带红框的）区域是检索记录列表，列出了馆藏记录中的名称和作者，类似的，生成这个区域的小器件 FormatRecordsWithDuplicates 类读取相关的记录格式显示配置文件，将记录格式化后显示给用户。通过修改格式显示配置文件，用户可以看到不同显示信息和显示效果。例如，可以把记录显示变成右对其，并为每条馆藏记录增加显示诸如出版单位、合作著者，著者单位等字段的信息。中央大块区域上方的横条（带蓝框的）是记录结果集的浏览导航链接，使用户方便地在各个记录之间进行浏览。

结束语 在数字图书馆的建设中，包括可定制界面在内的一系列用户个性化功能是其中一个十分重要的组成部分。SiteSearch 系统在以 Java 语言为主要软件开发平台的基础上，通过 API 中包含的 Java 组件库实现了用户界面的可定制功能。

基于众多因素的考虑，这个系统没有采用主流的商用开发模式，如 ASP、JSP 等动态网站制作技术，而是独辟蹊径，采用了 JaSSI 和 ZBase 组件与包含在 HTML 页面中的各种 Java 元素相结合、并辅以众多的系统配置文件的方式。这主要是因为，系统所连接的数据库都是针对图书馆界采用 Z39.50 协议标准的数据库，其协议的独特性在一定程度上决定了开发的独特性。

与“Web 服务器＋ASP/JSP＋数据库”的模式相比，SiteSearch 系统的开发模式较为复杂，需要 JavaPage 接口，实体和系统配置文件等多种机制的共同配合，但是它既能够满足针对 Z39.50数据库的跨库检索，又能实现统一的可定制用户界面的功能。而 ASP/JSP 作为较通用的技术，虽然也能完成动态用户界面，但只能访问常见的商用数据库，难以实现对图书馆界遵守 Z39.50协议的数据库的访问。

致谢 我们衷心感谢美国 OCLC 技术人员在该国际合作项目中给我们的大力支持和帮助。同时感谢该项目组的其他所有成员。

参 考 文 献

1 Oclc SiteSearch Online Help Documents. http://cypress.dev.oclc.org:7301/help/
2 Umlauft M. A Java Component User Interface for the Minstrel Push System. 2000
3 Peng Chengyuan, Vuorimaa P. Development of JAVA User Interface for Digital Television. Advanced Visual Interfaces, 2000. 276～279
4 Kwong Bor Ng. The applicability of universal pragmatics in information retrieval interaction: a pilot study, Information Processing and Management. Information Processing and Management, 2002, 38(2): 237～248
5 张德运，李德锴. 采用 JAVA 语言的通用型用户界面设计. 微型机与应用，1998
6 桂君，郭依群. 网络数据库用户界面的使用便利性研究. 情报理论与实践，2001
7 衡中青. Z39.50和 World Wide Web 的比较. 情报科学，2000

（上接第174页）

为.L 文件作为 FLEX++程序的输入，生成词法分析器；根据 DODL 语言的语法要求按照 YACC 语法写成后缀为.Y 文件作为 BISON++程序的输入，生成语法分析器。ONONET 语言编译器的生成过程和上述过程类似。关于 FLEX++程序、BISON++程序、.L 文件和.Y 文件的具体内容见文[8]。

结束语 领域工程是软件重用技术的一个重要分支和研究热点。领域工程的第一阶段——领域分析主要着重于解决领域建模问题。本文讨论了面向本体的 MIS 领域分析（ONODA）方法是指在一个特定的领域——管理信息领域中，定义一系列的领域本体类和对象类以构建领域的通用模型，为系统建模提供了前提条件；本文还讨论了支持该方法的原型系统的实现过程。在 ONODA 方法中，领域分析阶段被认为是一个知识获取的过程，因此，引入人工智能思想，采用本体作为知识表示方法，是 ONONDA 方法在理论上与一般领域分析方法的最大区别。虽然 ONODA 方法和 Promis 4.0系统在实现领域分析部分自动化工作中已经取得了一定的成绩，但是这部分工作还有需要继续完善的地方，如自动地获取领域知识、实现领域模型的修改。进一步地，我们希望 Promis 4.0 系统可以实现从领域分析阶段到后继的领域设计以及领域实现阶段的转换，将这三个阶段无缝地联接在一起，系统最终产生一个可运行的具体的管理信息系统。

参 考 文 献

1 Lu R, Jin Z. Domain Modeling-Based Software Engineering. Kluwer Academic Publishers, 2000
2 Sodhi J, Sodhi P. Software Reuse: Domain Analysis and Design Processes. McGraw-Hill Companies, Inc. 1998
3 Jacobson I, Griss M, Josson P. Software Reuse: Architecture Process and Organization for Business Success. 北京：世界图书出版公司北京公司，1998
4 Tracz W, Coglianese L, Young P. A Domain-Specific Software Architecture Engineering Process Outline. IBM Corporation Federal Systems Company
5 Gruber T. What is an Ontology? 1995
6 Uschold M, Gruninger M. ONTOLOGIES: Principles, Methods and Applications. Knowledge Engineering Review, 1996, 11(2)
7 Farquhar A, Fikes R, Rice J. The Ontolingua Server: a Tool for Collaborative Ontology Construction: [Technical Report KSL-96-26]. 1996
8 天鹰工作组：[天鹰项目技术报告]. 2000, 6