集成融汇工作流引擎研究*

李春旺[1] 费大羽[1,2] 周 强[1]
[1](中国科学院国家科学图书馆 北京 100190)
[2](中国科学院大学 北京 100049)

【摘要】讨论工作流技术支持下的集成融汇服务系统构建方法,重点分析基于客户端的融汇工作流过程描述规范以及基于客户端的集成融汇引擎系统架构、关键技术,并设计实现一个原型系统:iMashup,以验证相关技术方案的可行性和对相关研究与应用的借鉴性。

【关键词】集成融汇 工作流引擎 富客户端
【分类号】G250.76

Study on Mashup Workflow Engine

Li Chunwang[1] Fei Dayu[1,2] Zhou Qiang[1]
[1](National Science Library, Chinese Academy of Sciences, Beijing 100190, China)
[2](University of Chinese Academy of Sciences, Beijing 100049, China)

【Abstract】This paper discusses methods of the mashup engine based on workflow technologies, mainly analyzes the module of the JSON Process Definition Language, the architecture and key technologies of the mashup workflow engine based on client, and develops a tool: iMashup. The solution from this paper can be used in mashup applications.

【Keywords】Mashup Workflow engine Rich client

1 引 言

随着数字化、网络化技术的发展与应用,基于工作流的 Web 信息集成融汇服务开始受到越来越多的关注。早期的集成融汇服务通常与具体的应用系统绑定,融汇服务的灵活化、可扩展性以及跨平台可集成性受到制约。采用工作流技术,可以将集成融汇服务开发过程转变为脚本定义过程,不但降低技术门槛,提高开发效率,而且提升了融汇服务的开放性、扩展性,科研用户可以按照一定的标准规范自行构建个性化的服务,并支持开放保存、发现以及跨系统的集成调用,实现合作共建、结果共享、个性化服务等目标。同时,基于工作流技术,可以实现融汇过程定义与融汇服务运行相分离,避免了因数据与服务绑定而带来的知识产权等问题,并支持服务内容的动态更新,从而使集成融汇服务具有更广泛的发展前景。

2 相关研究

集成融汇是 Web2.0 核心特征之一。文献[1]和文献[2]分别对集成融汇服务的工作模式、组织机制以及关键技术等进行了比较全面的分析总结。将工作流技术与集成融汇技术相结合是近年来人们关注的热点,主要研究内容包括:工作流过程描述语言规范、控制引擎设计与实现技术等。由于集成融汇服务包括客户端与服务器端

收稿日期:2012-10-08
收修改稿日期:2012-11-26
*本文系国家社会科学基金资助项目"我国数字图书馆集成融汇服务方法研究"(项目编号:10BTQ004)的研究成果之一。

两种实现方式,因此,集成融汇工作流引擎通常也分为基于服务器端引擎与基于浏览器端引擎两种。其中,Cesare[3]提出一个多线程支持下的集成融汇系统 JOpera,支持可视化组件式开发,降低应用开发成本;ICT Romulus 则实现了一个基于浏览器端的集成融汇工具 MyCocktail[4],并构建可视化过程定义器,支持方便快捷的过程定义以及基于客户端的动态融汇运行;JackBe 公司面向企业融汇应用,提出一个基于服务器端的集成融汇平台 Presto[5],支持复杂的企业集成融汇需求;针对数字图书馆领域集成融汇服务的特点[6],中国科学院国家科学图书馆利用开源工具 MyCocktail,经二次开发实现了一个基于客户端的集成融汇工作流引擎系统:iMashup,并在群组知识平台系统 iLibrary 中得到应用。

本文结合 iMashup 研发工作,重点介绍基于客户端的集成融汇过程描述方案及工作流控制引擎实现的关键技术,希望为相关研究与应用提供参考。

3　工作流过程定义规范

规范的过程定义是实现工作流自动化控制的关键。为推动工作流过程定义的规范化建设,工作流管理联盟(Workflow Management Coalition, WfMC)提出通用工作流过程定义语言 XPDL(XML Process Definition Language)[7],它采用 XML 规范,支持开放扩展与解析,在工作流领域得到广泛应用。针对 Web 集成融汇服务的需求及特点,OMA(Open Mashup Alliance)提出支持企业集成融汇描述语言 EMML(Enterprise Mashup Markup Language)[8],Saeed 等[9]提出组件式描述规范 MCDL(Mashup Component Description Language)。XPDL、EMML、MCDL 等语言特别适合服务器端工作流引擎,随着富客户端技术的发展与应用,JavaScript 控制语言与 JSON 描述规范得到广泛应用[10]。为支持基于浏览器端的集成融汇过程定义与运行控制实现,在借鉴 MyCocktail 描述规范基础上,本文提出基于 JSON 格式的数字图书馆集成融汇工作流描述方案 JPDL(JSON Process Definition Language),其结构如图 1 所示。

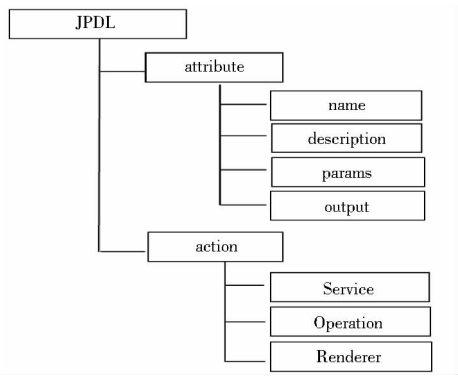

图 1　JPDL 结构图

一个 JPDL 工作流对象主要由两部分组成:

(1)基本属性描述,包括工作流名称(name)、描述信息(description)、参数项(params)、输出对象(output)等。其中,参数项允许在工作流定义阶段定义新的参数,支持融汇引擎在运行时根据参数完成动态操作,包括实现多个工作流融汇服务对象之间的通信与人机交互操作等;输出对象存放工作流过程定义结果脚本。

(2)关于集成融汇行为的定义,即把集成融汇应用系统包含的功能操作划分为一个个服务组件对象,对每个组件对象的基本属性信息、调用接口、输入与输出参数等进行描述,同时定义多个功能组件的调用次序、组合逻辑,实现复杂集成融汇工作流的定义。

一个集成融汇应用系统一般由三层结构组成:资源访问与获取层、信息分析与融汇处理层、结果呈现与交互层[11],因此,JPDL 将集成融汇组件对象划分为三种类型:

(1)资源访问操作对象(Service),定义融汇操作将访问的资源对象,需要指定服务调用接口、输入参数以及结果数据格式等,每个资源对象对应现实中的一个实际资源,一个集成融汇工作流系统中可以定义多个资源访问组件对象,从而支持对多种来源数据的访问、获取;

(2)信息处理操作对象(Operation),定义对获取到的数据进行何种处理,比如:合并、过滤、切分、抽取、替换、分组、排序等,需要定义服务调用接口及输入、输出信息规范,在一个工作流中同样可以定义多个信息处理操作对象,以便根据需要对有关信息进行多种融汇处理;

(3)结果呈现操作对象(Renderer),定义以何种方式对融汇后的综合数据进行呈现,实现与用户的交互。比如:地图、饼图、柱图、表格等呈现方式,也可以定义多个结果呈现操作对象,以便对融汇结果信息进行组合呈现,如分别以饼图、柱图方式呈现有关统计信息,

再以表格方式对饼图、柱图、文字信息进行组合呈现。

所谓集成融汇工作流定义，就是根据需要选择具有相应功能的 Service、Operation、Renderer 服务组件，定义每个组件对象的输入、输出等属性，通过组件名称调用建立组件之间的关联，并按组件对象定义的先后顺序确定工作流执行时序。同时，通过在组件对象定义中插入 JavaScript 控制语句，支持工作流的分支、合并等复杂操作。

以下是一个 JPDL 融汇工作流脚本实例，包括操作行为的三个定义：

```
{"name":", "description":"", "params":[], "output":" ${resultdisplay}",
 "actions":[
    {"type":" Service. Nsl. CrossSearch","name":"cross-search",
      "inputs":{"query":"nano","dbname":"SpringerLink 电子期刊:1"}
    },
    {"type":"Operation. Array. SortBy","name":"sortby",
     "inputs":{"array":" ${crosssearch. results}","property":"title","direction":"ASC"}
    },
   {"type":"Renderer. AdvancedRenders. ResultDisplay",
      "name":"resultdisplay",
      "inputs":
        {"records":" ${sortby}","numberofresult":"10",
          "width":"400","height":"400",
          "labelField":"title","valueField":"url"
        }
   }
 ]
}
```

（1）定义一个资源访问型操作，访问对象是国家科学院图书馆的跨库检索系统“Service. Nsl. CrossSearch”；指定两个输入参数，分别是检索数据库对象“SpringerLink 电子期刊”、默认查询关键词“nano”；指定按默认方式返回查询结果。

（2）定义一个信息处理型操作“Operation. Array. SortBy”，即对跨库查询结果信息进行排序；排序字段是“title”；排序方式是升序“ASC”。

（3）定义一个结果呈现型操作“Renderer. AdvancedRenders. ResultDisplay”，具体指定按表格模式呈现经以上排序处理后的数据信息；显示视图区域宽 400 像素、高 400 像素，表格第一列显示“title”字段内容，第二列显示“url”字段内容，总共显示 10 条信息；指定呈现对象名称“resultdisplay”。

4　运行时控制引擎设计实现

确定工作流过程定义规范后，需要开发计算机控制系统实现对工作流过程定义的解析，并按工作流定义要求自动调用相关服务组件，完成集成融汇处理操作，生成所需要的融汇服务，并与用户交互，这是工作流控制引擎要实现的功能。作为客户端集成融汇引擎系统，iMashup 系统架构如图 2 所示：

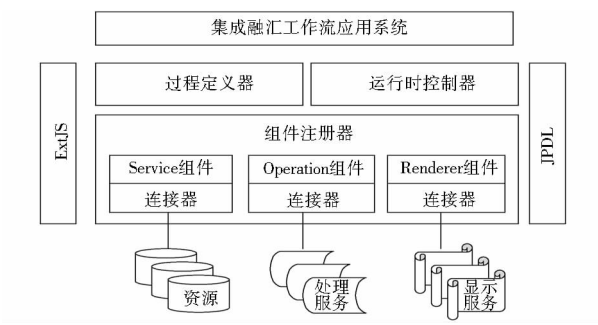

图 2　客户端集成融汇工作流引擎架构

4.1　组件注册器

组件注册器的主要功能是支持各类型融汇组件在集成融汇工作流引擎上的注册。为支持 Service、Operation、Renderer 等多种类型服务组件在工作流引擎中的开放注册、集成调用，需要为融汇组件建立一个连接器，向上定义通用的工作流调用接口，向下建立与本地的、远程的、第三方的服务连接。由于是在浏览器端完成融汇操作，因此，每个组件连接器表现为一个 js 文件，具体包括两部分内容：

（1）定义融汇组件的属性，如组件类型（Service、Operation、Renderer 等）、组件名称、输入/输出参数等；

（2）定义融汇组件要实现的功能，可以利用脚本语言（如 JavaScript）直接编写具有相关功能的程序代码，并将代码嵌入连接器 js 文件中，也可以定义一个服务调用接口，实现对第三方提供的 Web 服务的调用。例如，调用基于 SRU、SQL、RSS、JSONP、Web Service 等接口规范的数据资源访问服务，调用第三方提供的数据分析、过滤、核定等处理服务，调用本地或外部提供的饼图、柱图、地图、表格等可视化显示服务。根据浏览器端集成融汇的特点，第三方服务调用将采取异步

请求模式,返回结果数据要遵循 JSON 规范。

以下是一个关于 Google 搜索服务组件的连接器实例。

```
……
register(new UnitAction({
  name:'GoogleSearch',
  type: 'Service',
  label: 'Google 搜索',
  description: '调用 Google Ajax 搜索服务,获取查询结果',
  inputs: [ { name : 'query', label : '检索词', type : 'String' } ],
  execute : function(request, callback) {
   var url = 'http://www.google.com/uds/GwebSearch? context
       =0&lstkp=0&rsz=large&v=1.0&q=' + encodeURICom-
       ponent(request.params['query']);
   ajax.jsonp.invoke(url, {
     onSuccess : function(ret) {
       if (ret[2] = = '200') {
           callback.onSuccess(ret[1]);
         }
       },
     });
   }
}));
```

为实现对多种不同融汇组件的调用,保证系统简单易用,在连接器开发方面可借鉴跨库检索系统连接器策略[12],即开发通用连接器,实现对接口规范相似的多个融汇组件的调用,提高开发效率并降低培训学习成本;同时,开发专用连接器,实现对特殊接口规范服务的调用,满足个性需求。

4.2　过程定义器

用户参与的集成融汇服务构建是 Web2.0 的主要特征之一。为了降低用户对工作流过程定义规范的学习成本,帮助用户简单快捷完成融汇服务工作流的定义,需要开发融汇过程定义器。过程定义器主要功能包括:

(1)借助 ExtJS 技术实现融汇组件可视化,将组件属性等底层细节信息变成可视化的参数选项,指导用户快速完成相关操作;

(2)通过鼠标拖放、下拉菜单选取等方式,建立服务组件间的关联,ExtJS 开发框架及界面组件库对此有很好的支持;

(3)融汇操作结果所见即所得,根据过程定义要求,系统自动调用相关资源提供处理、呈现等服务,并即时显示融汇操作结果,让用户及时了解融汇效果,以便更好地进行后续子过程定义;

(4)自动生成集成融汇工作流脚本,包括对运行时环境的调用、信息装载等,避免手工编写脚本造成的语法错误、不规范性等问题。

一个工作流过程由若干活动组成,每个活动对应一个服务组件。借助过程定义器的支持,用户通过选择 Service 组件完成资源获取过程定义,通过选择 Operation 组件指定数据处理与融合操作,通过选择 Renderer 组件确定显示交互方式,最终生成一个个性化融汇服务工作流描述脚本。

4.3　运行时控制器

浏览器系统接收到一个集成融汇工作流脚本后,需要装载控制程序,生成运行结果。为支持客户端的集成融汇,需要采用浏览器兼容的脚本语言(如 JavaScript)编写控制程序,并以 js 文件形式保存在工作流引擎服务器上,支持浏览器端的调用。运行时控制器主要功能包括:解析融汇工作流脚本,根据脚本定义,构建活动对象 ActionDef、过程对象 ProcessDef,生成集成融汇工作流对象实例 ProcessInstance;根据工作流定义顺序,依次调用相关服务组件,完成所需的数据获取、分析、融合操作,生成融汇结果,并以可视化方式呈现给用户。由于运行时控制器是在浏览器中被调用、运行的,因此,集成融汇工作流服务支持向任何 Web 应用系统的嵌入。

4.4　ExtJS 框架

ExtJS[13]是一款基于 JavaScript 的客户端 Ajax 的应用开源工具包[14],它参考 JavaSwing 等机制,定义了一系列客户端常用可视化组件,如:文本框和文本域控制、单选框和复选框、表格控件(Grid Control)、树形控件、标签控件(Tabs)、工具条、桌面应用程序菜单、滚动条、Flash 图表以及皮肤布局等。ExtJS 被众多富客户端系统采纳,iMashup 利用 ExtJS 工具包,开发实现了基于客户端的集成融汇应用界面功能。

5　在 iLibrary 中的应用

iLibrary 是国家科学图书馆开发的一款基于表示层服务对象集成融汇的群组知识平台系统[15],目标是支持科研用户自我融合数字图书馆、数字科研、数字教学、数字出版等资源与服务,方便快捷地构建个性化知

识环境。为了满足个性化融汇服务需求,iLibrary 系统对 iMashup 进行集成,具体包括三个层面:

(1)集成 iMashup 的工作流定义器,支持用户以可视化方式定义集成融汇过程;

(2)将 iMashup 生成的融汇工作流脚本作为 iLibrary 表示层对象 Portlet 的内容,支持开放注册、动态发现与集成;

(3)集成 iMashup 工作流引擎,当用户在页面中调用指定 Portlet 实例时,浏览器自动读取 Portlet 内容中的工作流 JPDL 脚本信息,并自行调用部署在 iMashup 服务器上的工作流运行环境文件,解析 JPDL 脚本,按脚本定义完成资源获取、数据处理、显示合成等操作,最后按 Portlet 设定属性显示融汇结果。

借助 iMashup 工具,iLibrary 用户可以根据需要定义个性化的融汇服务工作流,在 iLibrary 平台注册后即可支持开放调用。而且,iLibrary 提供融汇服务对象输出功能,支持融汇工作流服务向第三方系统的嵌入。

6　结　语

在借鉴前人研究成果基础上,本文将集成融汇工作流对象划分为服务、操作、呈现三类,提出一个基于 JSON 规范的、支持客户端解析的融汇工作流描述规范,同时,探讨了基于 ExtJS 架构的集成融汇过程定义器、工作流运行控制器等模块的设计与实现技术,并进行应用分析,研究成果对相关融汇服务系统建设具有一定的借鉴。然而,一个完整的集成融汇工作流引擎涉及多方面的技术,如访问控制与用户认证、个性化数据源连接器自动构建、脚本语言对浏览器的兼容等问题,需要在今后的研究工作中逐步解决。

参考文献:

[1] 李春旺,肖伟. 集成融汇:概念、模式与应用[J]. *现代图书情报技术*,2008(12):22-26. (Li Chunwang, Xiao Wei. Mashup: Concept, Architecture and Application[J]. *New Technology of Library and Information Service*, 2008(12): 22-26.)

[2] 李峰,李春旺. Mashup 关键技术研究[J]. *现代图书情报技术*,2009(1):44-49. (Li Feng, Li Chunwang. Study on Mashup Technology[J]. *New Technology of Library and Information Service*,2009(1): 44-49.)

[3] Cesare P. A Flexible System for Visual Service Composition[D]. Switzerland: Swiss Federal Institute of Technology Zurich, 2004.

[4] MyCocktail Mashup Builder[EB/OL]. [2012-07-18]. http://www.ict-romulus.eu/web/mycocktail/.

[5] Presto Whitepaper[EB/OL]. [2012-07-18]. http://www.presto-eng.com/downloads/wp_pe.pdf.

[6] 李春旺. 图书馆集成融汇服务研究[J]. *现代图书情报技术*,2009(12):1-6. (Li Chunwang. Study on Library Mashups[J]. *New Technology of Library and Information Service*, 2009(12): 1-6.)

[7] Workflow Management Coalition. XML Process Definition Language[EB/OL]. [2012-07-18]. http://www.xpdl.org/standards/xpdl-2.2/XPDL%202.2%20(2012-02-24).pdf.

[8] EMML Changes Everything: Profitability, Predictability & Performance Through Enterprise Mashups[EB/OL]. [2012-06-05]. http://openmashup.org/whitepaper/do-cs/oma_whitepaper_120309.pdf.

[9] Saeed A, Cesare P. The Mashup Component Description Language[C]. In: *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services*, Ho Chi Minh City, Vietnam. New York: ACM, 2011:311-316.

[10] RFC4627:The Application/json Media Type for JavaScript Object Notation(JSON)[EB/OL]. [2012-07-18]. http://tools.ietf.org/html/rfc4627.

[11] López J, Pan A, Bellas F, et al. Towards a Reference Architecture for Enterprise Mashups[C]. In: *Proceedings of the 6th Workshop on Business Solutions for the World of Web*, Gijón, Spain. 2008: 67-76.

[12] 李广建,刘晓娟,黄永文. Cross-Search 系统的设计与实现[J]. *图书馆杂志*, 2006,25(5):46-51. (Li Guangjian, Liu Xiaojuan, Huang Yongwen. Design and Implementation of Cross-Search System[J]. *Library Journal*,2006,25(5):46-51.)

[13] 徐会生, 康爱媛, 何启伟. 深入浅出 ExtJS[M]. 2 版. 北京:人民邮电出版社, 2010. (Xu Huisheng, Kang Aiyuan, He Qiwei. Dissecting Ext JS[M]. 2nd Edition. Beijing: Posts & Telecom Press,2010.)

[14] Web Application Development with Sencha Ext JS Framework[EB/OL]. [2012-09-09]. http://www.sencha.com/products/extjs/.

[15] 王科,纪姗姗,刘芳,等. iLibrary:表示层集成融汇服务及系统实现[J]. *现代图书情报技术*,2010(11):30-36. (Wang Ke, Ji Shanshan, Liu Fang, et al. iLibrary: Presentation Layer Mashup Service and System Implementation[J]. *New Technology of Library and Information Service*, 2010(11): 30-36.)

(作者 E-mail:licw@mail.las.ac.cn)