基于 Mashup 的文档组合①

黄阳群, 姚志强, 沈薇薇

(福建师范大学 软件学院, 福州 350108)

摘 要: 随着云计算服务的发展, 人们对来自互联网上的海量文档资源的需求日益增大, 如何快速有效地将来自不同来源的文档元素或文本文件组合成为新的文档成为一个研究热点. 基于上述需求, 提出一种基于 Mashup 的文档组合方案, 通过搭建Mashup服务器, 将来自不同域下的文档资源组合成为新的XML文件, 并根据用户需求, 将组合后的文档以 DITA 体系设计编排成集*, 以不同的形式输出, 最后以一个医疗电子健康病历来对基于Mashup 的文档组合方案进行验证.

关键词: 云计算; 揉合; 文档组合; DITA; XML

Document Combination Based on the Mashup

HUANG Yang-Qun, YAO Zhi-Qiang, SHEN Wei-Wei

(School of Software, Fujian Normal University, Fuzhou 350108, China)

Abstract: With the development of cloud computing services, people demand for huge amounts of document resources from the Internet growing. It becomes a research hospot that how to quickly and efficiently merge document elements or context from different sources to a new document. Based on the above requirements, this paper puts forward a kind of document combination scheme based on Mashup. Through constructing the Mashup server, we put the document resources which come from the different domain into the new XML file. And according to users' requirements, the orchestration of the document after the combination with DITA system design into a set of output in the form of different output. Finally, we through a medical electronic health records to verify the document combinationscheme based on Mashup.

Key words: cloude computing; mashup; document combination; DITA; XML

1 引言

随着网络、数字出版等技术的进步, 云计算和互联网的飞速发展, 结构化文档应运而生, 逐步成为互联网信息传播的重要载体. 云计算平台为广大用户提供了存储、通信、分享等服务, 以满足用户对资源的全方位需求, 以及用户与终端、用户与用户之间的协同操作. 然而, 随着在线交互设备的多样化, 传统的单一文档格式已无法满足用户的需求, 结构化文档已发展成为多用户参与、多版本共存的新形态[1]. 面对互联网上海量的文档资源, 人们需要一种能够随时随地获取文档, 并高效地组合成为自己所需的新文档. 例如医疗机构的电子病历, 医务人员在操作时可以随时获取病人的某些信息, 而不需要重新进行编辑. 基于这种新兴的计算模式, 本文对网络化文档组合及重用展开了研究.

文档的组合重用一方面减少文档编辑中不必要的重复工作, 提高重用性; 同时便于梳理文档的主从关系以及相关文档之间的脉络, 增强可读性. 例如微软办公软件2007以后版本的Office Open XML(OOXML)文件格式, 应用"容器"概念来存储、管理和共享文档, 各个部件以 XML 格式分布, 用户只需要改动相应部分的 XML 文件, 亦或是提取某一所需的 XML 文件,

① 基金项目:国家自然科学基金(61370078,61402109);福建省教育厅基金(JB14034)
收稿时间:2015-02-08;收到修改稿时间:2015-04-26

即可重组为新版本的文档. 在云计算环境中, 组合文档具有多用户参与、内容动态、多媒体交互和多安全等级等“活”文档特征[2]; 1)多参与者通过云服务协同创建文档, 如同一病人的病历, 根据所检查的项目不同, 不同科室的医生协同录入病人的健康信息等[3]; 2)用户在不同的时间、地点对文档的访问操作权限是不同的, 结构化文档通过关联、组合、重组应能呈现不同的内容[4]; 3)云服务为用户提供分布于不同域下文档的数据交换和共享, 如同一用户可以在不同的办公环境对要操作的文档进行关联更新等[5].

随着 Web2.0 的兴起与流行, Mashup 作为 Web2.0 的代表技术之一, 体现了聚合、个性化、资源重用等特性, 常见的 Mashup 应用主要有视频图像 Mashup、搜索购物 Mashup、新闻 Mashup、地图 Mashup 等[6]. 然而 Mashup 在医疗、办公、学习等方面还未得到广泛应用, 且聚合后的视频、网页信息等都只能浏览, 而不具备重用的特性. 为满足用户对资源的全方位需求, 实现文档组合的多样性、可重用性和高可用性, 本文提出一种基于 Mashup 的文档组合方案. 该方案在获取本域 XML 文档资源的基础上, 进一步跨域请求, 获取更多符合用户需求的文档; 通过搭建 Mashup 服务器, 处理获取到的文档, 构造种类多样、内容动态的组合文档; 最后使用 DITA 结构化编写方式, 以主题为单位创建、编排, 达到更有效地组织和呈现文档内容.

2 相关知识

2.1 Mashup 技术

Mashup 是一种新型的基于 Web 的数据集成式应用程序, 它将不同数据源提供的数据融合在一起, 提供给用户更新颖、直观、丰富的应用[7]. 它的“查存渠道”(checkpointed channel)的方法可以使得多个合作编辑者协同完成同一文档的编写与存储[8]. Mashup 工具为用户提供了搭建部分自动化的 Mashup 应用, 如 Yahoo! Pipes, IBM Mashup Center, Intel Mash Maker 等, 从而对互联网开放的数据及服务应用进行聚合. 关于 Mashup 的获取方式可概括为以下几点[9]:

(1)集成数据. 在 Mashup 应用中, 有大量以集成数据的方式, 通过 RSS 进行简单的内容联合. 例如新闻和 Weblog 聚集程序, 使用 RSS 和 Atom 等技术订阅个性化的新闻阅读应用, 用户不需要打开各大新闻网站, 一一浏览当天的新闻, 而是以搜索汇聚关键词、提要的形式, 向用户推送个性化的服务, 最大程度满足读者多样化的需求, 实现内容的聚合.

(2)揉合应用程序逻辑. 揉合应用程序逻辑指的是内容提供者发布自己的公共接口 API, 通过 SOAP 或 XML–RPC 协议, 与内容访问者建立连接, 发送请求和响应, 将信息传输给 Mashup 服务器. 目前许多应用如 Amazon、eBay、Flickr、Google、 Microsoft、Yahoo、YouTube 等都发布了自己的公共接口 API, 用户只需要调用相关应用的 API, 即可传递相关的参数信息[10]. 另外, REST(Representational State Transfer)也是一种实现 Web Service 的架构风格. 它是使用 HTTP 和 XML 进行基于 Web 通信的技术, 直接工作在 HTTP 协议之上. 一个 RESTful Web 服务接口的主要特点是: 可访问性, 统一的接口, 连通性和无状态[11]. RESTful 架构的核心部位是一组资源, 网络上的一切事物都可以被抽象为资源: 如文字、图片、音乐等. 这些资源由 URL(比如一个 Uniform Resource Locator URL)和内部表示法(通常为一种自描述形式的数据)来识别, 用户可以使用标准的 HTTP 方法, 如 GET、PUT、POST 和 DELETE 等操作来处理资源.

(3)揉合用户界面. 此外, 用户也可以根据需求, 通过拖拽图形控件订制个性的 Mashup 应用. Yahoo 公司推出的 Yahoo! Pipes 就是利用一个可视编辑器重新混合流行的 Feed 类型来创建数据 Mashup 的工具. 这个可视化的编辑器无需繁琐的编码即可构建完善的用户体验界面, 让非程序员能够轻易使用这个服务.

显然, 上述 Mashup 的获取方式为用户提供了多样化的开发选择, Mashup 的应用也为用户提供了混搭的个性化服务. 然而当前 Mashup 应用大多都是针对特定的 API, 对基于 XML 语言的文档资源的搜索及混合却鲜有提及, 且所提供的工具无法满足用户自由组合文档的需求[12].

基于这些考虑, 本文构建 Mashup 服务器, 对结构化文档进行组合与重用.

2.2 DITA 体系结构

达尔文信息分类体系结构(Darwin Information Typing Architecture, DITA)是一种编写 XML 数据的模型, 用于定义编写、生成和交付内容信息的规则. 它的基本原则是以主题为单位创作文档, 并利用主题图(Map, 集成信息)对主题进行组合, 通过格式转换, 输出多种形式的交付物[13]. DITA 的主题创作可分为三类:

概念主题(concept topic)、任务主题(task topic)和参引主题(reference topic). 概念主题通常解释或定义一些想法, 即“是什么”的问题; 任务主题则描述了事件执行的过程; 参引主题提供前者所需的参引信息集合, 来进一步支持这些任务. 三者的关系可以利用 DITA 图进行组织, 整合为集*, 如图 1 所示.

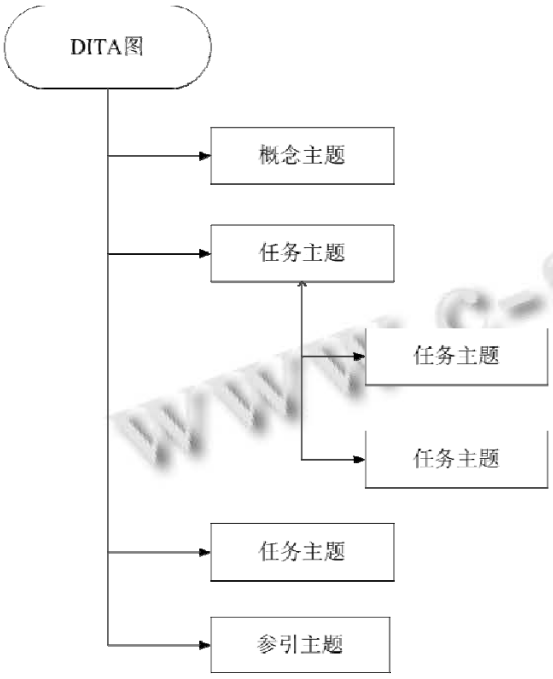

图 1 DITA 图的信息组织过程

3 文档组合方案

传统的 Mashup 系统由三个部分构成: Mashup 内容提供者、Mashup 服务器和 Web 浏览器[14]. 本文以此为基础, 给出方案模型如图 2 所示, 通过手动编码构建 Mashup, 预先确定各个组件, 从而根据需求建立组合文档[15]. 模型包括以下实体: 用户、Web 浏览器、Mashup 服务器、组合文档、域.

用户: 向浏览器发送指令的操作者, 对文档有组合需求的人.

Web 浏览器: 将用户的请求命令发送至 Mashup 服务器, 为组合文档提供可视化的平台.

Mashup 服务器: 接收来自 Web 浏览器的指令, 并获取所需文档, 进行相应的文档组合操作.

组合文档: 获取到的文档经由 Mashup 服务器解析、组合后得到的产物, 这里的文档以 XML 文档为主.

域: 负责存放用户所需的文档数据, 多域为用户提供了多元化的信息环境.

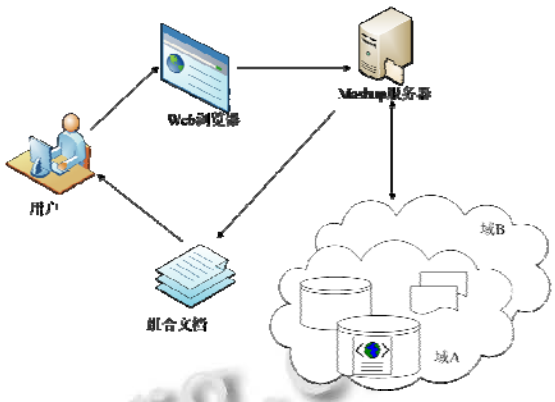

图 2 文档组合方案模型

3.1 文档组合方案的工作原理

如图 3 所示, 在文档组合系统中, 用户通过 Web 浏览器向 Mashup 服务端发送文档请求指令, 服务器根据用户输入的 URL 定向获取所需的文档, 响应后进行一系列组合操作. 本文利用 IIS 搭建两个域 A 和 B, 将 XML 文档存放在不同的域内. 由于所有浏览器中都内置了安全性限制, 如一个 XmlHttpRequest 对象只能从提供此网页的服务器中获取数据, 而不可以从其他站点中使用 XmlHttpRequest 对象, 因此在请求文档过程中, 需要通过 Mashup 服务器作为中间媒介, 而无法直接连接浏览器和合作网站[16].

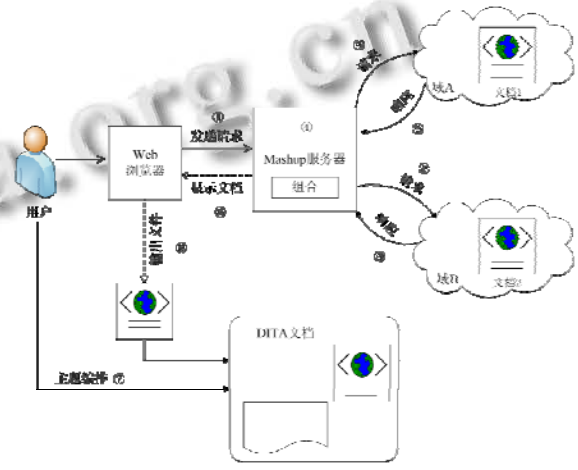

图 3 文档组合方案模型

方案可分为以下三个阶段:

第一阶段为文档请求: ①表示用户通过 Web 浏览器输入指定的 URL, 即文档或网页所在路径. 该请求传达给 Mashup 服务器, 服务器收到来自 Web 浏览器的 HTTP GET 请求后处理该指令. 大部分 Mashup 内容

提供者为客户提供了 REST 和 SOAP 接口, 用户可以任意搜索或指定某一文档或网页内容.

第二阶段为文档组合: 该阶段包含多个步骤, 服务器在接到指令后分别向多个域转发请求, 即步骤②, 同时步骤③表示 Mashup 内容提供者响应到请求并返回至 Mashup 服务器, 此时 Mashup 服务器收到了用户请求的所有指定文档内容, 开始组合工作. 步骤④中, Mashup 服务器首先解析 XML 文档数据, 遍历 2 份文档的节点, 合并节点名称相同的内容, 之后组合成一个新的 XML 文档, 最后将组合结果呈现在 Web 浏览器⑤.

第三阶段为文档编排: DITA 体系结构为文档组合操作者提供了规范化的模型, 例如某学生获取到图书馆的电子资源, 可以按照 DITA 主题编排, 按照中文期刊、外文期刊等方式分类. 组合完的文档以 XML 格式保存在默认路径下, 步骤⑥将合并后的多个独立的 XML 文档采用 DITA 体系设计, 以 DITA 图的形式将文档整合成集*. 步骤⑦表示了用户在 DITA 编排当中, 根据文档的主题可以构造出不同形式的多样化文档, 使得重组后的文档因用户的角色、应用场景等因素具有不同的表现形式[17].

下面从 Mashup 组合框架的三大组成部分, 对本方案系统作详细说明.

3.2 文档组合方案系统描述

3.2.1 Web 浏览器

Web 浏览器使用 Visual Web Developer 2005Express Edition 作为开发工具, 采用 MVC 模式, 通过 Design 视图拖拽相关控件以及 Source 视图手动编码方式相结合创建表单布局, 能够实现所见即所得.

在本文方案中, Web 浏览器作为客户端主要与 Mashup 服务器进行交互. 如下表所示, HttpWebRequest 类提供了对 WebRequest 中定义的属性和方法提供支持, Create()方法为指定的 URL 方案初始化新的实例. 用户输入指定的 URL 后发出 Http Get 请求, 服务器收到指令后进行 Mashup 混合操作. HttpWebResponse 用于创建对象并返回服务器的响应, 从服务器中读取响应的主体通过 GetResponseStream 返回 Stream 对象. 为成功将返回内容显示在用户的浏览器上, StreamReader 对象读取响应流中的主体. 最后, Literal 控件从响应流中读取返回的 XML 文本内容, 并根据返回信息动态更新页面.

```
protected void cmdGo_Click(object sender, EventArgs e)
{
    HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(this.txtURL.Text);
    HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
    Stream myResponseStream = myResponse.GetResponseStream();
    StreamReader myReader = new StreamReader(myResponseStream);
    this.myLiteral.Text = ((StreamReader)myReader).ReadToEnd()
}
```

3.2.2 Mashup 服务器

Mashup 服务器器主要进行以下几个工作: ①接收请求: 用户在 Web 浏览器输入指定 URL 信息, 传输给服务器端. ②响应请求: Mashup 服务器接收到来自浏览器的请求后, 向不同域下的内容提供者发送获取文档的指令. ③文档组合: 将得到的 XML 文档按照指定方式进行混搭[18], 如图 4 所示.

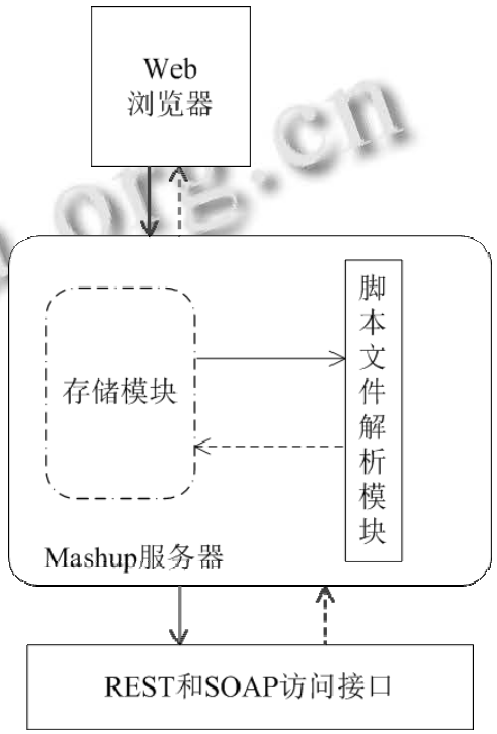

图 4 Mashup 服务器

Mashup 服务器即实现 Mashup 逻辑的地方[19], 包

括了存储模块、脚本文件解析模块.

存储模块: 存储模块主要包括服务脚本文件, 用于存储由客户端发送过来的消息指令, 以及通过 Mashup 服务运行后的文档组合结果, 也保存在该模块.

脚本文件解析模块: 脚本文件解析模块主要是对接收到的脚本文件进行节点读取和组合. 代码如下所示, 首先创建名为 MashupRoot 的根节点, 表示上下文是经由 Mashup 服务器操作的组合文档. 其次使用 ImportNode 方法将文档转化为 XmlNode 对象, 由根节点开始逐个访问子节点, 使用 AppendChild()添加至新创建的 XML 文档中. 当遇到相同节点时, MergeXmlFiles()会将节点下的所有内容合并, 从而将获取到的两份文档生成动态的自由组合的文档.

```
{
    XmlDocument
xmlreader1=new XmlTextReader(myReader1);
    XmlDocument
xmlreader2=new XmlTextReader(myReader2);
XmlDocument    myResults =new XmlDocument();
    XmlNode
myRoot =myResults.CreateElement(“MashupRoot”);
myResults.AppendChild(myRoot);
                                        XmlNode
text1=myResults.ImportNode(xmlreader1.DocumentEle
ment, true);
myResults.DocumentElement.AppendChild(text1);
    XmlNode
text2=myResults.importNode(xmlreader2.DocumentEle
ment, true);
myResults.DocumentElement.AppendChild(text2);
}
```

3.2.3 内容提供者

在文档组合过程中, 本地资源已无法满足用户的需求, 面对云计算环境海量的资源, 他们希望能够从网络上获取到指定的信息为自己所用. 如医生在给患者看病时, 常常需要获取患者的电子健康病历; 而患者移动性强, 可能在不同的医疗服务机构就医. 假设所有的域(对应医疗服务机构)之间相互信任, 医生在获取患者电子健康病历时, 就需要向他域请求文档. Mashup 内容提供者还可以是开放 API 的网站, 他为用户提供了 REST 和 SOAP 两种 Web 服务访问方式, 提高了系统的兼容性和实用性. 用户既可以经由 SOAP 接口提出访问请求, 也可以访问 REST 文档信息. 当服务器向内容提供者转发来自用户的访问请求时, 此时调用 POST 和 GET 的功能, 向服务器推送相应的文档信息.

4 案例研究

4.1 医疗电子病历文档组合

随着科学技术的发展和云计算环境的普及, 纸质病历逐渐被医疗电子健康系统所取代. 本节以此为背景, 实现医疗电子健康病历的文档组合. 在文档组合系统中, 用户提出文档组合需求, 提供要访问的网络资源标识. Mashup 服务器在接收到获取的文档后, 执行混合的命令. 如某位患者可能曾在医院 H1 的其他部门或医院 H2 就诊, 并记录有该患者的电子病历资源, 那么接待他的主治医生则不需要创建新的电子健康病历, 直接从云服务器端或允许访问的其他医疗机构读取、复制或修改病历信息. 如图 5 所示, 域 A 中有一份门诊部的电子诊断书名为门诊部.xml, 化验科也为患者提供化验结果并保存在域 B 上, 命名为化验科.xml 且两份文档均基于电子健康病历的统一标准 CDA 文档为模板而编写. 当其他科室的医生想要了解患者的既往病史以及化验结果, 或其他医院的医生想要获取患者过去的就医情况, 则需要整合两份文档. 因此, 用户输入两份文档的 URL, 点击文档组合按钮, 通过 Mashup 服务器组合成一份新的电子健康病历, 并以 XML 格式保存.

4.2 组合文档编排

在医疗信息服务中, 用户在不同时间和地点可能具有不同的访问权限, 结构化文档也应随之呈现不同的内容; 病人在进行健康检查时, 可能由不同科室的医护人员共同编辑其健康报告. 对于一份完整的电子健康系统, 其内容包括个人信息、门诊记录表、血液生化记录、计算机 CT 扫描影像片和 B 超视频记录等多媒体文档. 为了使组合后的文档更加直观、规范、完整, 用户还可对新生成的 XML 文档进行重新编排与修改.

如图 6 所示为用户利用 DITA 结构重组的文档编排实例. 该文档使用 Oxygen XML editor 编辑工具, 主题的编写遵守 DITA 主题类型结构和相关元素, 保证

了格式的标准性和结构的一致性. 通过 DITA 的结构化编写, 用户可以将组合后的文档以主题形式存放, 如下图左侧为大纲视图, 显示了文档的结构及相关元素, 按照电子健康系统中相关主题如用户个人信息、用户电子健康病历等编排. 通过点击某个主题链接, 可直达右侧显示该主题对应子文档的详细信息. 使用 DITA 图将主题一一映射进行组合排列, 使得用户可在不同的集合中复用 topic, 也可以在不同的 topic 之间复用内容. 文档按照任务主题、概念主题、参引主题等组织成 DITA 图后, 用户只需在创作过程中, 从中挑选相应的主题, 即可建立良构的结构化文档. 最后通过样式模板的 XSLT 转换机制, 根据不同环境需要输出不同格式类型的知识产品(交付物). 输出格式多种多样, 如 PDF、HTML、CHM、RTF 等[20].

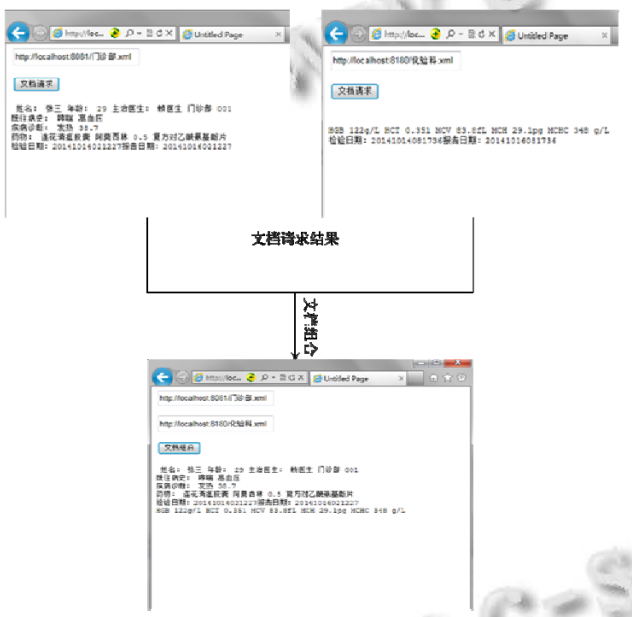

图 5　文档组合实现

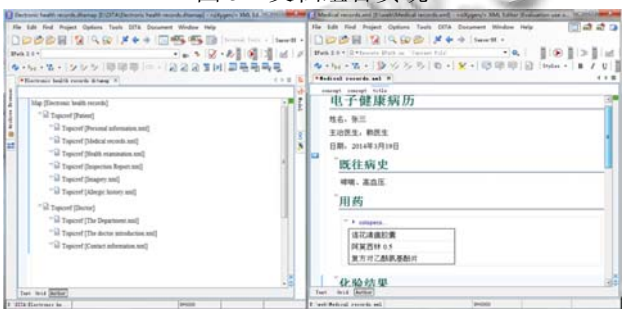

图 6　DITA 电子健康病历

4.3 方案综合分析

目前, Mashup 技术广泛应用于各大领域, 同已有的相关资源整合方案相比, 基于 Mashup 的文档组合方案主要在检索准确率、组合性能和文档重用率等三个方面与其存在差异, 具体对比如表 1 所示.

表 1　基于 Mashup 的文档组合方案与已有方案的比较

方案	检索准确率	组合性能	文档重用率
方案[21]	×	√	×
方案[22]	√	√	×
方案[23]	√	×	×
本文方案	√	√	√

通过方案比对可以发现, 文献[21]提出的 RSS 资源聚合系统, 能够较好地聚合来自网络上的数据信息, 然而 RSS 在资源搜索方面还存在一定局限性, 如关键字搜索不够精确、需要手动添加门户网站等; 文献[22]搭建的 Mashup 客户端获取学习资源能够有效地对网络上的学习资源进行搜索与组合, 然而组合后的资源内容不能高效地为操作者所重用; 文献[23]运用 Mashup 等技术构建的医院专家门诊预约网站系统, 采用 REST 资源搜索和抓取技术, 然而资源的整合并不意味着文档的重组, 其在文档的组合性能方面未有提及. 综上所述, 本文方案在资源的检索、文档组合以及文档的重用方面都具有良好的性能.

5　结语

新兴的计算环境下, 原有的文档模式已无法满足用户采集和运用大量信息, 组合文档在信息交换与传播中起到了至关重要的作用. 为了更有效地对来自不同域下的文档重组, 本文提出一种基于 Mashup 的文档组合方案, 通过搭建 Mashup 服务器来获取、解析、组合 XML 文档, 从而达到建立简短、有效的跨域搜索文档的目的. 同时, 为构建规范化组合文档, 本文还结合当下流行的 DITA 结构体系创作, 提高信息的准确性、规范性, 使得组合后的文档更具阅读性和重用性. 对于协同操作的文档编辑团队, 网络化文档组合保证了数据的统一管理, 并将互联网上海量的文档资源、团队之间不同版本的资料信息组合生成新文档, 提高了工作的便捷性和高效性. 最后, 本文以医疗电子健康系统为背景, 对基于 Mashup 的文档组合方案进行验证, 并和其他相关方案对照, 证明该方案的有效性和实用性.

下一步研究工作将致力于文档组合软件的设计与实现, 并深入建立安全体系机制, 为用户提供更加智

能、安全的文档组合服务.

参考文献

1 Balinsky H, Simske SJ. Secure document engineering. Proc. of the 11th ACM Symposium on Document Engineering. ACM. 2011. 269–272.

2 熊金波,姚志强,马建峰,刘西蒙,马骏.云计算环境中的组合文档模型及其访问控制方案.西安交通大学学报,2014, 02:25–31.

3 熊金波,姚志强,马建峰等.面向网络内容隐私的基于身份加密的安全自毁方案.计算机学报,2014,37(1):139–150.

4 姚志强,熊金波,马建峰等.云计算中一种安全的电子文档自毁方案.计算机研究与发展,2014,51(7):1417–1423.

5 姚志强.普适计算模式下的文档组合与安全研究[博士学位论文].西安:西安电子科技大学,2014.

6 邸杰.基于REST的Mashup开发生成环境的设计与实现[硕士学位论文].北京:北京邮电大学,2012.

7 Ronen B, Palley MA, Lucas Jr HC. Spreadsheet analysis and design. Communications of the ACM, 1989, 32(1): 84–93.

8 Ostrowski K, Birman K. Storing and accessing live mashup content in the cloud. ACM SIGOPS Operating Systems Review, 2010, 44(2): 7–11.

9 Soi S, Daniel F, Casati F. Conceptual development of custom, domain–specific mashup platforms. ACM Trans. on the Web (TWEB), 2014, 8(3): 14.

10 李峰,李春旺.Mashup 关键技术研究.现代图书情报技术, 2009,3(1):44–49.

11 Rauf I, Ruokonen A, Systa T, et al. Modeling a composite RESTful web service with UML. Proc. of the Fourth European Conference on Software Architecture: Companion Volume. ACM. 2010. 253–260.

12 Roy Chowdhury S, Daniel F, Casati F. Recommendation and weaving of reusable mashup model patterns for assisted development. ACM Trans. on Internet Technology (TOIT), 2014, 14(2–3): 21.

13 Bellamy L, Carey M, Schlotfeldt J. 李颖等译.DITA 最佳实践指南——创作、编排和架构的技术路线.北京:科学出版社,2012:69–76.

14 戎修凯.Mashup 研究与应用[硕士学位论文].北京:北京邮电大学, 2013.

15 Kopliku A, Pinel-Sauvagnat K, Boughanem M. Aggregated search: A new information retrieval paradigm. ACM Computing Surveys, 2014, 46: 57–76.

16 Shanahan F.吴宏泉译. Mashups Web 2.0 开发技术——基于 Amazon. com.北京:清华大学出版社,2008:78–96.

17 熊金波,姚志强,金彪.云计算环境中结构化文档形式化建模.计算机应用,2013,33(5):1267–1270.

18 王辉,高成英,刘宁.服务器端Mashup开发平台的设计与实现.计算机工程,2010,36(10):262–264.

19 苏会杰.基于 REST 和 SOAP 的 Mashup 平台 Web 服务研究与实现[硕士学位论文].北京:北京邮电大学, 2012.

20 范炜.达尔文信息类型架构 DITA 研究.情报杂志,2009, 11:172–175.

21 王立峰,郑燕林.JavaFX RIA 框架下学习资源获取 MASHUP 富客户端设计与实现.现代教育技术,2013, 23(7):90–94.

22 林俊生. RSS 资源聚合系统搜索引擎的设计与实现[硕士学位论文].广州:中山大学,2012.

23 邱暐.基于 Rest 和 Mashup 的 SOA 应用框架研究[硕士学位论文].上海:上海交通大学, 2010.