

# Web 遗留系统的服务包装器环境设计

周家晶<sup>1</sup>, 邹翔<sup>2</sup>, 沈备军<sup>1</sup>, 胡善学<sup>1</sup>

(1. 上海交通大学软件学院, 上海 200240;

2. 信息网络安全公安部重点实验室, 上海 200031)

**摘要:** Web 遗留系统的白盒服务迁移需要侵入原有系统修改代码, 因此会产生运行风险。针对该问题, 设计并实现一种黑盒包装器环境。通过为遗留系统配置包装器, 忽略原系统内部的复杂性, 在不改变系统部署和运行状态的情况下, 将遗留的 Web 应用转化为 Web 服务。应用结果表明, 基于外部包装和规则配置的黑盒方法更具可控性。

**关键词:** 遗留系统; 服务包装器; Web 服务; 黑盒; 包装策略

## Design of Service Wrapper Environment for Web Legacy System

ZHOU Jia-jing<sup>1</sup>, ZOU Xiang<sup>2</sup>, SHEN Bei-jun<sup>1</sup>, HU Shan-xue<sup>1</sup>

(1. School of Software, Shanghai Jiaotong University, Shanghai 200240, China;

2. Key Laboratory for Information and Network Security, Ministry of Public Security, Shanghai 200031, China)

**【Abstract】** White-box method for Web legacy system migration need to modify the application source codes and invade the original system, which has great operational risks. To solve the problem, this paper designs a black-box wrapper environment for Web service. Through the wrapper, the complexity of the internal structure of the original system is ignored. The Web application can be migrated into Web services without changing the deployment and operational status of the original system. Application result proves that the method has better controllability.

**【Key words】** legacy system; service wrapper; Web service; black-box; wrapping strategy

DOI: 10.3969/j.issn.1000-3428.2011.19.023

### 1 概述

遗留系统是企业已经完成开发并部署运行的系统, 特别是指那些长期运行并且已经难以维护或进一步演化的复杂信息系统<sup>[1]</sup>。遗留系统包含企业运营知识, 在信息系统集成中需要慎重对待, 由于 Web 服务是目前主流的 Web 应用集成方法, 提供了业界普遍认可的服务发布与交互标准, 因此将遗留系统改造为 Web 服务可以延长其生命周期, 便于构建集成系统。国内对信息发布类的 Web 应用集成已经有较充分的研究, 特别是针对 HTML 信息抽取的研究, 成果丰硕<sup>[2]</sup>, 但无法实现交互性的应用集成, 因此, 出现了不少遗留系统的 Web 服务白盒迁移研究成果, 并提出了遗留系统的服务形式化封装<sup>[3]</sup>, 但白盒法需要侵入遗留系统, 可能导致不稳定性。针对上述问题, 本文提出了一种针对 Web 遗留系统的黑盒服务包装方法。

### 2 模型概念和相关研究

黑盒法是指通过包装器的方式继承遗留系统。通过包装器, 可以很好地解决遗留系统应用程序接口与集成遗留系统所需提供的接口不匹配的问题。

IBM 研究人员提出了一个概念体系结构<sup>[4]</sup>用于支持对遗留系统的 Web 服务包装。文献[5]分析了基于 Form 表单的遗留系统的特点, 提出了一个可自动将基于 Form 的遗留系统封装成为 Web 服务的包装器, 文献[6]进一步提出了一个页面交互模型。

图 1 是 Web 遗留应用黑盒服务重构的概念模型, 将包装器部署在 Web 服务运行平台上, 即可实现对遗留系统的封装, 并将其发布为 Web 服务。

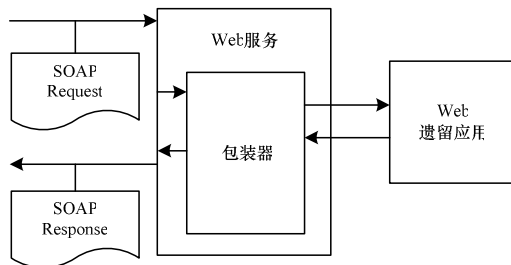


图 1 Web 遗留系统服务黑盒包装模型

### 3 包装器环境定义与实现

得到 Web 遗留系统的服务包装器需要经过 3 个步骤: 首先获取遗留系统的 HTTP 会话数据; 然后分析数据, 获取功能点, 并制定相应的包装策略; 最后根据包装策略生成包装器逻辑代码。

#### 3.1 包装器环境的形式化定义

在类似 a: A 的描述中, 符号“:”表示 a 是 A 的一个实例。本文将包装器环境做如下形式化定义:

遗留系统样本元数据为  $Element = \langle Request, Response \rangle$ , 它是一次请求和应答对。

来自遗留应用 legacySystem: LegacySystem 的样本数据集  $q: Q = \{ele: Element | ele: Element \text{ from } legacySystem\}$ 。用于

**基金项目:** 国家“863”计划基金资助项目(2008AA01Z412)

**作者简介:** 周家晶(1987—), 男, 硕士研究生, 主研方向: Web 语义, 信息安全; 邹翔, 副研究员、博士; 沈备军, 副教授、博士; 胡善学, 硕士研究生

**收稿日期:** 2011-03-27

**E-mail:** spider.zhou@gmail.com

对样本数据集进行分类的聚类策略记为 ClusterStrategy。使用策略 cS: ClusterStrategy 对 q 做聚类分析后得到聚类族:

$$q^c = \{p_1, p_2, \dots, p_n\}, p_i \in q$$

用于包装会话的包装策略由请求封装策略和应答抽取策略两部分组成, 记为:

Strategy=<CasingRequest, ExtractResponse>

包装后的数据元由包装后输入和包装后输出组成, 记为:

WarpDataElement=<WarpIn, WarpOut>

WarpDataSet=P×Strategy 表示聚类采用策略分析后得到包装数据集。对一个聚类 p 使用策略 strategy 分析后得到的包装数据集为:

wDS: WarpDataSet=p × strategy={wDE: WarpDataElement| warpDataElement=e. check(strategy: Strategy), e ∈ p}

包装后的数据符合一定的数据结构:

ObjectInf=<RequestObj, ResponseObj>=  
ObjectCheck(WarpDataSet)

包装器包含如下 2 类: 环境服务项 Envservice, 包装服务项 Warpservice。前者用于维护独立的包装运行环境(如建立和销毁独立的 HTTP 会话), 后者用于对遗留系统的功能进行包装。

包装服务项主要由包装数据结构和包装策略组成:

Warpservice=<ObjectInf, Strategy>

其形式化逻辑如下:

Define wS: Warpservice  
in rqO: RequestObj, env: Env  
execute(strategy: Strategy)  
out rpO: ResponseObj

包装器环境服务项包含如下 2 项: initialEnv: Envservice, destoryEnv: Envservice, 分别用于建立和销毁独立的运行环境。

用 WarpserviceSet 表示 Warpservice 项的集, 则包装器可记为:

sWM: serviceWrapMachine={initialEnv, destoryEnv} ∪ wSS: WarpserviceSet

### 3.2 样本数据获取

HTTP 的 Cache 机制可以降低网络负担并提高应用性能, 但会对客户端与应用的交互性分析产生干扰, 因此, 本文设计了一个代理服务器, 用于移除双方 HTTP 会话中的 Cache 头以关闭客户端与服务器的 Cache。

为方便分析 HTTP 报文, 以 XML 格式记录 HTTP 报文(XML-HTTP), 需要对原始数据统一编码, 并将 HTML 数据 XML 标准化。一份样本数据由请求和应答两部分组成, 如下所示:

```
<request type="post" version="1.1">
  <uri host="bbs.sjtu.edu.cn">
    <segment>login.do</segment>
    <queryStr>
      <param name="type">1</param>
    </queryStr>
    <uri/>
    <entity type="application/x-www-form-urlencoded"
      showType="nameValuePair">
      <param name="name">admin</param>
      <param name="pd">admin123</param>
    </entity>
  </request>
```

### 3.3 包装制定与数据结构代码生成

使用相似性分析作为聚类算法对样本数据进行聚类分

析, 得到的聚类就是遗留系统的功能点, 而遗留系统的每项功能都是潜在的服务。之后对每个聚类制定包装策略, 策略的核心是制定参数格式并说明如何将参数封装为 HTTP 请求, 以及如何从 HTTP 应答中抽取数据并包装为指定格式数据, 分别称为请求包装策略和应答抽取策略。

本文为请求包装策略设计了 Argument and HTTP RMR (Request Mapping Rule) 映射规则作为请求包装规则。下面给出一个 RMR 片段:

```
<param name="name">
  <ex: rmr value="login\@name"/>
</param>
```

以该片段为例, RMR 是在 XML-HTTP 请求报文的基础上通过<ex value="login\@name"/>将参数包装为 HTTP 报文请求。同时采用 XSLT 作为应答抽取策略, 文献[7]对此有较为充分的研究。

为了支持带参数传入型的 XSLT, 同样设计了一个 XPMR (argument and XSLT Paraments Mapping Rule) 规则用于说明如何为 XSLT 传值。示例如下:

```
<schemaArgs>
  <arg name="selectArea"
    value="login\args\selectArea">
  <example>一区</example>
  <example>二区</example>
</arg>
</schemaArgs>
```

在该例中, 从包装器参数的 login\args\selectArea 位置取值为 XSLT 参数 selectArea 传值, 并给出可能的值, XPMR 中的参数项也是包装器请求数据的一部分。

为聚类配置 PMR、XSLT 和 XPMR 规则后, 用规则匹配和抽取对应聚类的样本数据可以得到包装请求和包装输出的 XML 数据, 系统分析这些数据后可以得到匹配这些数据的 XML schema, 建模可以得到数据结构的 Java 代码, 这样就完成了包装器请求和输出的数据结构信息, 从而得到 3.1 节中的 ObjectInfo。这就得到了每个功能聚类的数据包装结构, 并制定了完整的包装策略。

### 3.4 服务包装器生成

交互性的 Web 遗留系统与客户端在一个会话环境下进行交互, 会话环境主要指 Cookie 和 SSL/TSL 连接(在安全连接下), 每个客户端应该有独立的会话环境, 为此, 包装器需要生成 2 个环境服务项目, 分别用于建立和销毁独立的会话环境, 其中:

(1) initialEnv 建立会话环境后返回会话环境编号, 包装器自主管理会话环境, 对于基于双向 SSL/TSL 连接的 Web 遗留系统, 需要将用户的 X.509Certification 作为参数传入。

(2) destoryEnv 销毁指定编号的环境, 参数为调用 initialEnv 得到的环境编号。

在建立会话环境后, 就可以调用服务包装器项。服务包装器项的输入包含封装请求和环境编号, 输出为输出对象, 其中, 封装请求和输出为可选。

图 2 给出了服务包装器的程序逻辑, 其中, 虚线表示可选的对象或数据流, 包装策略是包装器项的运行策略, 其根据包装数据结构信息分析请求信息, 通过 RMR 规则得到 HTTP 请求, 得到应答信息后, 用 XSLT 和 XPMR 规则抽取应答数据, 并封装为制定包装器数据结构信息返回。

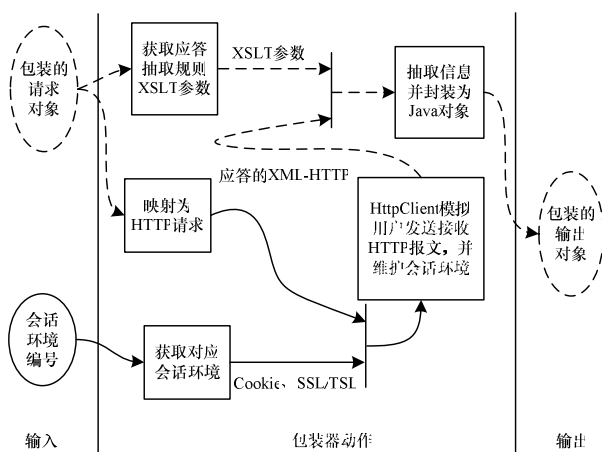


图2 包装器服务项的程序逻辑

#### 4 应用实例

对上海交通大学校园论坛(bbs.sjtu.edu.cn)应用本包装环境, 4 名使用经验丰富的用户通过包装器环境提供的代理服务访问该站点, 在 3 天内获得了 1.2 万余对数据记录, 通过聚类分析得到 28 个聚类, 最终选择对其中较有特点的 10 个聚类进行服务包装策略配置, 如图 3 所示, 将生成的包装器配置到支持 JAX-WS2.0 的 GlashFish 2.1 服务环境之下, 图 4 是通过 SOAP UI(www.soapui.org)工具检测到的 Web 服务项, 由此证明了本文方法的可行性。服务包装过程与白盒法相比, 更多地利用了原系统使用者的经验和行为数据, 更强调系统的业务逻辑, 不依赖代码分析, 同时, 基于外部包装和规则配置的黑盒方法更具可控性。

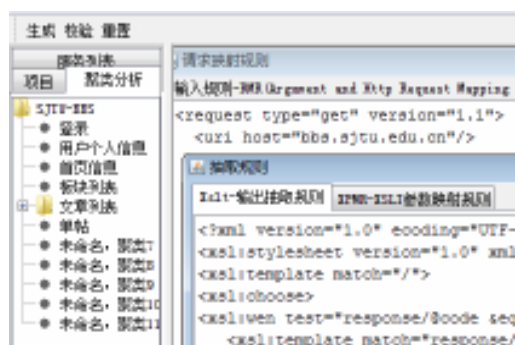


图3 包装器规则制定

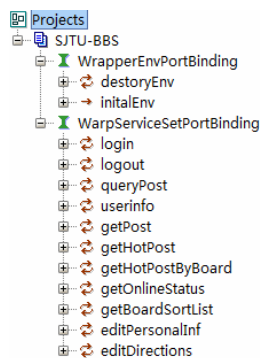


图4 用 SOAP UI 检测得到的 Web 服务列表

#### 5 结束语

本文提出了一种基于封装器的 Web 遗留应用服务封装方法, 开发了原型系统, 并通过实验验证了该方法的可行性。其包装器环境可用于集成遗留系统的服务开发, 延长遗留系统的生命周期, 降低企业信息化开发的代价。

#### 参考文献

- [1] Brodie M L, Stonebraker M. Migrating Legacy Systems: Gateways, Interfaces, and the Incremental Approach[M]. San Francisco, USA: Morgan-Kaufman Publishers, 1995.
- [2] 蒋宏潮, 王大亮, 班晓娟, 等. 基于 XML 的 Web 数据半自动采集[J]. 计算机工程, 2009, 35(21): 51-53.
- [3] 姜峰, 范玉顺. 服务环境下基于数据约束的遗留系统形式化封装[J]. 高技术通讯, 2008, 10(18): 1059-1065.
- [4] Kuebler D, Eibach W. Adapting Legacy Applications as Web Services[EB/OL]. [2010-11-20]. <http://www.ibm.com/developerworks/library/ws-legacy/index.html>.
- [5] Draheim D, Weber G. Modelling Form-based Interfaces with Bipartite State Machines[J]. Interacting with Computers, 2005, 17(2): 207-228.
- [6] Dilenzo G, Fasolino A R, Melcarne L. Turning Web Applications into Web Services by Wrapping Techniques[C]//Proceedings of the 14th Working Conference on Reverse Engineering. [S. l.]: IEEE Press, 2007.
- [7] 廖灵睿, 肖田元. 基于 XSLT 的 Web 包装器环境[J]. 计算机工程与科学, 2006, 28(9): 15-17.

编辑 张帆

(上接第 72 页)

#### 6 结束语

本文提出了一种用于全局数据整体规划的数据库聚类算法, 先计算数据库包含属性信息的相似度, 再根据相似度进行数据库聚类和聚类效果评价。该算法具有简易性及良好的通用性, 适用于数据集成环境复杂、数据规划人员经验缺乏情况下的初步数据规划。算法没有考虑数据-现实模型和实体-活动联系对数据规划和数据应用的影响, 因此, 在本文研究的基础上对输出聚类进行细化、调整和修正, 是进一步研究的方向。

#### 参考文献

- [1] 王玉书, 董不明. 主题数据库规划合理性估计的数学公式[J]. 软件学报, 1997, 8(2): 93-98.
- [2] 周炎涛, 谢东, 吴正国. 主题数据库技术的亲和性分析方法

研究[J]. 计算机工程与应用, 2005, 41(12): 168-170.

- [3] 刘文远, 徐丽娜, 王宝文, 等. IRP 中基于广义聚合度的主题数据库划分技术[J]. 情报杂志, 2007, (10): 70-74.
- [4] 刘文远, 于家新, 徐丽娜, 等. 基于依赖关系的大规模主题数据库的分解模式[J]. 计算机科学, 2008, 35(5): 223-225.
- [5] 刘文远, 陈灿, 陈国鹰, 等. 基于实体亲和度的主题数据库规划[J]. 计算机工程, 2008, 34(15): 42-43.
- [6] 刘文远, 杨丹丹, 王宝文. IRP 中基于聚类分析的主题数据库划分研究[J]. 情报杂志, 2009, (1): 16-18.
- [7] Zhang Chengqi, Zhang Shichao. Database Clustering for Mining Multi-databases[C]//Proc. of 2002 International Conference on Fuzzy Systems. Honolulu, USA: IEEE Press, 2002.

编辑 张帆



