



SaaS 软件服务基于大规模定制的客户端层框架研究 *

吴清烈, 罗小利

(东南大学管理工程研究所 南京 211189)

摘要

大规模定制是 SaaS 软件服务成功实施的关键。在客户端层,为满足租户对系统界面的差异化需求且通过各种终端访问应用软件,本文提出了 SaaS 软件服务基于大规模定制的客户端层框架,用 XML 文档详细描述了客户对应用系统界面的需求,通过调用界面配置算法生成满足客户需要的数据,并将其与终端集成组件进行交互,最终生成客户需要的客户界面。最后通过案例证明了此框架的可行性。

关键词 SaaS;大规模定制;界面;配置;终端

1 引言

随着互联网应用的不断发展以及应用软件的日益成熟,作为一种新颖的软件提供方式,软件即服务(software as a service, SaaS)模式受到人们的普遍关注。SaaS 把应用软件以服务的形式通过互联网提供给客户,客户只需交纳一定的租赁费用就可以通过浏览器随时随地使用软件的服务功能^[1]。

在 SaaS 模式中,多个租户共同使用一个应用实例^[2]。软件服务商为了降低成本以快速地响应客户的需求,总希望提供标准化的界面、统一的客户端接口。而对于每个租户来说,总是希望在系统界面符合自己需求的基础上用方便的客户端快速地访问应用软件,且应用软件能与已有系统相集成。因此,在竞争激烈的市场环境下,SaaS 服务商在提供满足租户个性化需求界面、面向多租户终端软件服务的基础上也要降低成本、提高效率,以提高市场竞争力

位^[3]。大规模定制是根据客户的需求,向客户提供个性化的产品或服务来提高客户满意度,同时又通过适度标准化生产以降低产品或服务的成本,提高效率。大规模定制的对象既可以是实物产品也可以是软件或服务。据此,本文试图按照大规模定制的理念设计 SaaS 软件服务客户端层架构,使同一应用软件实例集成了各种客户端及已有应用软件,并支持客户个性化定制系统界面,从而满足 SaaS 软件服务模式的多租户性和可配置性特征。

本文首先分析 SaaS 软件服务在客户端层应用大规模定制的必要性,提出基于大规模定制设计 SaaS 软件服务客户端层框架的思路,解决 SaaS 软件服务集成各种客户端和定制客户界面的问题;接着,对此框架进行分析,详细描述客户对系统界面需求的 XML 文档和界面配置算法;最后用案例说明 SaaS 软件服务大规模定制客户端层框架的可行性。

2 SaaS 软件服务基于大规模定制的客户端层框架设计思路

SaaS 软件服务的租户是通过客户端界面来体验 SaaS

* 教育部人文社会科学研究规划基金资助项目(No.09YJA630022)



软件服务的,所以如何构建既能满足租户功能需求,又能提升租户使用体验的客户端,是SaaS模式成功的关键因素。在SaaS模式下,不同行业、不同文化背景、不同管理模式的企业对应用程序的界面外观要求是不一样的,一般都希望拥有自己风格的界面,以体现其独特性,符合企业文化的特征;即使是同一企业,在不同时期对客户界面的要求也会发生变化。这就要求SaaS软件服务商要通过定制化设计满足租户提出的客户界面个性化需求。在SaaS模式中,尽管每个租户对应用程序界面的要求是不一样的,但是在内容上却大体相同^[4]。这为按照采用模块化和部件标准化实现低成本的大规模定制思想设计SaaS软件服务客户端层提供了可能性。

根据以上两点,笔者基于大规模定制思想思考SaaS软件服务客户端层框架的设计问题^[5,6]。通过分析,将客户对应用界面的需求分为两类:客户对界面内容的需求和客户对界面外观的需求。同一领域的企业对同一应用程序的界面内容要求大体相同,把界面内容抽象成界面功能模块,存储在界面功能构件库中。界面管理模块把客户需求通过表示层翻译成计算机能够识别的语言,向业务逻辑层发送服务请求。在业务逻辑层,界面功能配置引擎根据客户的内容需求在界面功能构件库中检索相应的功能模块,若完全与客户的需求相符合,则直接输出;若只需修改相关参数就能满足客户的需求,则修改后输出;若检索不到相应的功能模块或者需要做大幅度的修改,则重新开发相应的功能模块,并把它标准化再输出。界面外观定制通过CSS和SKIN等技术设置满足客户特定需求的页面风格。业务逻辑层把处理结果经过表示层翻译成高级语言,与终端集成组件进行交互,最终通过客户指定的终端类型显示

客户界面。

3 SaaS 软件服务基于大规模定制的客户层框架设计

根据上述思路,提出如图1所示的SaaS软件服务基于大规模定制的客户层整体结构,包括界面管理模块、终端集成组件、客户界面。客户界面的生成流程分为4个阶段:客户调查阶段;客户需求分析阶段;应用系统界面配置阶段;面向各种终端的界面展示阶段。

3.1 客户调查阶段

应用系统界面是租户每天必然接触到的,也是在体验SaaS软件服务时评价SaaS模式最重要的指标之一,所以SaaS软件服务提供商必须满足租户对应用程序界面的需求,以抓住现有客户并增加新客户。客户需求调查的宗旨是忠实地反映客户的真实需求。客户需求调查主要包括:企业环境调查;客户端界面外观调查;客户端界面呈现内容调查。调查租户通过不同终端对应用程序界面呈现的内容需求,租户可以自由地决定其界面上有哪些内容。

3.2 客户需求分析阶段

通过调查客户对应用系统界面的需求,建立客户需求文档。根据定制的多少,客户需求文档分成两个部分,即界面外观需求分析和内容需求分析。不同企业对应用系统的外观要求基本不一样,他们希望能够拥有自己的页面风格,色彩、字体、业务数据的表现形式等均能符合自身企业文化的特征。但是同一领域的企业对同一应用系统界面的内容要求大体相同。

3.3 应用系统界面配置阶段

界面管理模块通过对界面需求文档的解析,经过表示

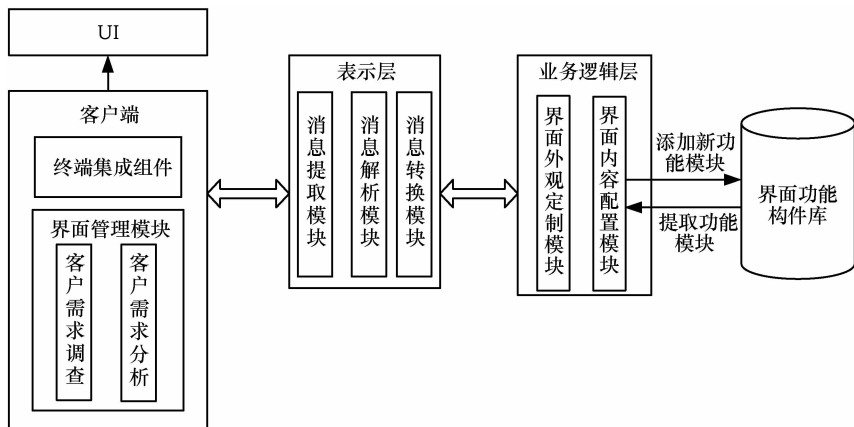


图1 SaaS 软件服务基于大规模定制的客户层整体结构

层的提取、解析、转换,生成计算机能够识别的语言,向业务逻辑层请求服务。界面功能配置引擎负责在界面功能构件库中检索响应的功能模块,若完全与需求文档相符合,则直接提出;若绝大部分相符合,则修改相应的参数,使之满足客户的需要;若检索不到相应的功能模块,则创建满足客户需要的新功能模块,并把它标准化,存储在界面功能构件库中。调用级联样式表(cascading style sheet)设计满足租户要求的页面风格。

界面功能构件库分为领域共性功能模块和个性功能模块,领域共性功能模块是对同一应用系统在特定领域进行分析,剥离出一些个性的需要做大量更新的功能,从而产生在一定领域共同具有的模块。个性功能模块是特定租户对应用系统界面内容的特殊要求。

3.4 面向各种终端的界面展示阶段

业务逻辑层集成组装各种界面功能模块,调用 CSS 等页面风格函数生成界面文档,经过表示层生成高级语言,与终端集成组件交互,最终提供面向各种终端界面的布局,向租户提供差异性页面风格的人—机交互界面。终端集成组件是应用系统在客户端的代理组件,其功能是实现与应用系统交互及向租户展示人—机交互接口。该组件接收客户的操作请求,通过表示层向业务逻辑层请求服务以及接收表示层发送的响应数据,向租户显示处理结果。

图1所示的客户端层整体结构,应用大规模定制设计理念,在满足客户对应用系统界面的个性化设置需求下,充分利用资源、尽力降低成本,快速响应客户服务请求,具有以下4方面的优势。

(1)充分利用资源,降低成本,实现客户界面的快速生成。根据定制的范围大小,客户界面的需求分为界面功能配置和界面外观定制。界面功能配置引擎检索功能模块时,对于领域共性功能模块可以直接重用或者修改后重用,这大大提高了界面生成的效率,降低了成本,有利于 SaaS 软件服务提供商在激烈的市场竞争环境下求得生存和发展。

(2)同一应用软件面向多类客户终端,满足 SaaS 模式“单实例多租户”特征。

SaaS 软件服务提供商与租户之间是一对多的关系,不同租户可能使用不同的终端来访问同一应用软件,即使同一租户也会要求使用不同的终端类型,如电脑终端、移动终端。这就要求 SaaS 软件服务提供商提供的应用软件要面向多客户终端,以扩展软件服务的应用范围。租户通过

终端集成组件向业务逻辑层发送请求以及呈现业务逻辑层处理的结果,实现了同一应用软件绑定多种终端组件^[7]。租户可以根据自身喜好选择恰当的终端组件,终端组件实质上是客户终端与应用软件服务的中介,对应用软件屏蔽了不同终端类型的差异性。所以 SaaS 软件服务基于大规模定制的客户层框架对每种应用系统均可以开发出多种终端类型,不需要对该系统进行修改就可以实现该应用系统面向多种终端类型。

(3)客户界面具有柔性,能够适应企业不断变化的需求。

客户可以改变其页面内容或风格,且不会对其他租户的界面产生影响。界面的柔性主要表现在界面外观的变化和界面内容的变化两方面。

- 界面外观的变化:租户对界面外观的需求发生变化时,如调整字体大小,可以通过调用级联样式表来改变字体字号样式,满足客户的需求。
- 界面内容的变化:当租户对页面上的内容需求发生改变时,如提出新内容模块,界面功能配置引擎快速地在界面功能构件库中检索相应的模块并将其输出,从而使其映射的界面满足客户的要求。

(4)SaaS 模式客户端层具有松耦合性。

终端集成组件通过表示层与应用系统相交互,从而实现了终端组件与应用软件的松耦合,即应用系统发生变化时不会对租户产生附带的影响。

4 SaaS 软件服务基于大规模定制客户端层框架实现的关键技术

SaaS 软件服务商要满足客户对应用系统界面的需求,并降低成本、提高效率,首先细分客户的界面需求,根据界面内容需求文档在界面功能构件库中检索相应的功能模块,根据外观需求文档调用 CSS、SKIN 等函数设计界面风格,业务逻辑层将处理的结果与表示层进行交互,最终通过终端集成组件显示客户界面。

4.1 基于 XML 的界面需求文档

下面是 XML 文档对界面需求的具体描述:

```
<?xml version="1.0" encoding="UTF-8"?>
<client-app xmlns:xsi = "http://www.w3.org/2001/
XMLSchema-instance" xsi:noNamespaceSchemaLocation = "C:\
\client-app.xsd">
  <service-info>
    <name>服务名称</name>
```



```

<description>客户的服务请求</description>
<source-port>SRC-PORT-1</source-port>
<destination-port>DEST-PORT-1</destination-port>
<message>
  <message-method>REQUEST/RESPONSE</message-
method>
  <terminal-type >WEB/MOBILE/DESKTOP/HOME-
TERMINAL</terminal-type>
  <default-config>TRUE/FALSE</default-config>
  <content-params>
    <param>
      <name>transmission-security</name>
      <value>LEVEL-1/LEVEL-2/LEVEL-3</value>
    </param>
    <param>
      <name>verification</name>
      <value>STANDARD/ADVANCED</value>
    </param>
  </content-params>
  <interface-params>
    <param>
      <name>style</name>
      <value>TRADITIONAL/CLASSICAL/POPULAR<
/value>
    </param>
    <param>
      <name>skin</name>
      <value>SNOW-SKIN/GRASS-SKIN/BLUESKY-SKIN<
/value>
    </param>
    <param>
      <name>layout</name>
      <value >VERTICAL/HORIZONTAL/MIXING </
value>
    </param>
  </interface-params>
  <extend-params>
    </extend-params>
</message>
</service-info>

```

```
</client-app>
```

从客户界面需求的 XML 描述可以看出, 客户可以通过移动、桌面、Web 等终端与表示层进行交互。SaaS 软件服务提供商按照客户对界面的要求开发相应的内容, 并通过终端集成组件显示业务逻辑层处理的结果, 最终生成客户界面。

4.2 界面配置算法

基于大规模定制的 SaaS 软件服务在客户端层设计满足租户需求的界面时, 经过分析把客户的界面需求划分为界面内容需求和界面外观需求两类。把界面内容板块抽象成功能模块存储在界面功能构件库中。配置客户界面时, 界面功能引擎首先解析客户内容需求文档, 并检索界面功能构件库。根据关键字的匹配程度相应地对功能模块进程进行处理, 然后输出。界面外观定制通过解析客户外观需求文档, 调用 CSS、SKIN 等函数处理界面风格。具体的算法如下:

```

recv(MESSAGE mes) //接收消息
{
  ComponentList list;
  MESSAGE mes_content,mes_interface;
  mes_content=parseForContent (mes); //解析消息并生成与“内容”相关的配置文档
  mes_interface=parseForInterface (mes); //解析消息并生成与“外观”相关的配置文档
  submit(mes_content,list); //提交消息
  submit(mes_interface,list);
}
parseForContent(MESSAGE mes)
{
  params=getContentParams(mes); //提取与“内容”相关的字段
  mes_content=createConfigDoc (params); //根据字段生成配置文档
}
parseForInterface(MESSAGE mes)
{
  params=getInterfaceParams(mes); //提取与“外观”相关的字段
  mes_content=createConfigDoc(params);
}
submit(MESSAGE mes2,ComponentList list)
{
  componentInfoSet=init(mes2); //初始化
  foreach(componentInfo in componentInfoSet)

```

```

{ matching=queryLib(componentInfo); //查询模型库
  if(matching equals "exact_matching") //“完全匹
    配”的情况
    { component=getComponent (componentInfo); //
      直接获取组件
      list.add(component); //添加
    }
    else if(matching equals "partial_matching") //“部
      分匹配”的情况
    { component=getComponent(componentInfo);
      modify(component,componentInfo); //修改组
      件参数
      list.add(component);
    }
    else //未匹配到
    { component =createComponent (componentInfo);
      //创建新组件
      list.add(component);
    }
  }
}

```

界面配置算法处理了客户界面的需求信息,生产满足客户所需界面要求的数据。

5 SaaS 软件服务基于大规模定制的客户 端层框架举例

假定某一 SaaS 服务商提供考试系统租赁服务,要求考试系统的登录界面要满足其个性化需求且可以通过电脑和移动终端来访问。下面说明如何通过 SaaS 软件服务基于大规模定制的客户端层框架满足考试系统登录界面的个性化需求。

(1) 调查并分析客户对登录界面的需求

通过调查分析客户对登录界面的需求,需求信息可划分为登录界面内容需求和登录界面外观需求。登录界面内容需求包括学校 Logo、考试系统名称、登录对话框等板块,登录界面外观需求包括界面背景颜色、字体大小及格式等。设置 XML 文档相应的参数,如终端类型设置为 Web 终端和移动终端。不同的终端类型对应不同的界面风格函数,如 Web 终端类型对应 CSS,通过 CSS 调整登录界面的肤色、字体等界面风格。

(2) 配置登录界面

界面管理模块把客户的登录界面需求文档经过表示层的解析、翻译,向业务逻辑层发送请求,界面配置算法根据翻译后的界面内容文档,在界面功能构件库中检索相应的功能模块。有些功能模块可以直接复用,像登录对话框;有些功能模块则需要修改相应的参数,像考试系统名称模块,需要修改成该学校的名称;有些功能模块则需要重新开发,像该学校的 Logo。

(3) 显示登录界面

业务逻辑层把处理结果传递给表示层,经过表示层的解析、翻译与终端集成组件进行交互。图 2 和图 3 分别是通过电脑终端和移动终端访问的考试系统登录界面。



图 2 通过电脑终端访问考试系统的登录界面



图 3 通过移动终端访问考试系统的登录界面

从图 2 和图 3 可以看出,电脑终端和移动终端显示的是同一考试系统的登录界面,而考试系统端口不需要做任何修改,只需要定义终端集成组件,就可以实现同一应用系统面向多种终端的功能。



6 结束语

本文提出了 SaaS 软件服务基于大规模定制的客户层面框架,将客户对应用系统界面的需求划分为界面内容需求和界面外观需求,把界面内容板块抽象成界面功能模块,存放在界面功能构件库中,客户需求信息经过表示层解析翻译,向业务逻辑层请求服务。界面配置算法根据界面内容需求文档,在界面功能构件库中检索相应的功能模块,生成满足客户需要的内容数据;界面外观定制根据界面外观需求文档按照不同的终端类型调用不同的界面风格函数,以满足客户通过各种终端访问应用软件的外观需求。业务逻辑层把处理的数据经过表示层的转换,与终端集成组件进行交互,最终生成满足客户需要的客户界面。在该框架下,客户可以自定义界面外观和界面内容,且可以通过不同的终端类型访问软件服务。

参考文献

1 Gold N, Knight C, Mohan A, *et al.* Understanding service

oriented software. *IEEE Software*, 2004, 21(2): 71~77

- 2 Ralph Mietzner, Andreas Metzger, Frank Leyman, *et al.* Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications. *Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems*, 2009
- 3 Xin Mingdi, Levina N. Software-as-a service model: elaborating client-side adoption factors. *Proceedings of the 29th International Conference on Information Systems*, Paris, France, 2008
- 4 Prasun Dewan, Rajiv Choudhary. A high-level and flexible framework for implementing multiuser user interfaces. *ACM Transactions on Information Systems*, 1992, 10(4): 345~380
- 5 Eleni M, Ang J, Arsanjani A, *et al.* Patterns: service-oriented architecture and web services. <http://www.Chinagrid.Net/grid/paperppt/Patterns-Services.pdf>
- 6 Jaejeong Lee, Jinhan Kim, Byungjeong Lee. Semantic and dynamic web service of SOA based smart robots using Web 2.0 OpenAPI. *Proceedings of the 2008 Sixth International Conference on Software Engineering Research, Management and Applications*, 2008
- 7 陈小兵,武泽旭.支持多类终端与服务定制的 SaaS 软件服务架构. *计算机应用*, 2010, 30(10): 2754~2762

Research of Client Level Framework for SaaS Software Service Based on Mass Customization

Wu Qinglie, Luo Xiaoli

(Graduate Institute of Management Engineering, Southeast University, Nanjing 211189, China)

Abstract Mass customization is the key to the successful implementation of SaaS software service. In this paper, in order to meet diverse requirements of different tenants in the system interface, and to visit application software through all kinds of terminal, a client level framework for SaaS software service based on mass customization is proposed. In which, XML document describes the detailed requirements for the application system interface. According to this, the interface configuration algorithm is designed to produce data, which interacts with the terminal components to create user interface for tenants. Finally an example demonstrates the feasibility of this framework.

Key words SaaS, mass customization, interface, configuration, terminal

(收稿日期: 2012-08-02)