一种基于服务的自适应网络应用框架

罗 森,许 斌,孙科武

(清华大学 计算机科学与技术系,北京 100084)
E-mail: luos@keg.cs.tsinghua.edu.cn

摘 要: 提出一个基于服务的自适应网络应用框架—iWeb.框架建立了上下文信息模型,用以描述收集上下文信息;框架收集了数以千计的服务,并对这些服务依据其功能性进行了分组.在服务质量信息,上下文信息以及服务功能性分组的基础上,提出了一种创新的服务选择方法,该方法可以根据服务质量信息和上下文信息选择最佳的服务.框架使用具有相同功能,由统一接口封装的服务组作为基本功能模块,服务的选择,服务的调用以及服务之间的调度由框架内的服务引擎负责;框架还提供了应用编辑器,来加速应用界面和流程的开发.使用 iWeb 框架,可以在更短的时间里开发出具有基于网络服务的自适应性的应用.最后,通过可用性分析,验证了 iWeb 是一个实用,高效的网络应用开发框架.

关 键 词: Web 服务;服务质量;上下文;Web 服务选择;Web 服务组装
中图分类号: TP311 **文献标识码:** A **文 章 编 号:** 1000-1220(2013)01-0016-07

A Service-oriented Adaptive Web Application Framework

LUO Sen,XU Bin,SUN Ke-wu
(Department of Computer Science and Technology,Tsinghua University, Beijing 100084, China)

Abstract: The paper proposes a service-oriented web application framework named iWeb, which enables web application adaptive to both context and QoS. In iWeb, a context model is established and context information is collected systematically according to the context model. A service selection approach based on context and QoS is proposed in the framework as the leverage of implementation of application adaption. This approach can select the best service(s) respect to QoS and context. For the usability of iWeb, thousands of available web services are collected and grouped by functionality in order to make service selection practicable. By providing development tools, service-oriented web application can be developed easily, and can fulfill its functionality using the selected service(s). The framework makes service-oriented web application more adaptive and more flexible. In the end, through analysis of practical relevance and experiments, this paper validate that iWeb is a practical and efficient web application framework.

Key words: Web service; QoS; context; Web service selection; Web service composition

1 引 言

Web 服务由于其可互操作性,可重用性和全球化,近年来得到了广泛的关注和使用.随着网络中的提供各种各样功能 Web 服务越来越多,使用 Web 服务来构建应用,为各式各样的问题提供解决方案成为了可能.基于 Web 服务的应用因为其能增加重用性,降低开发成本,成为了 Web 服务研究中的一个热点问题.

Web 服务以及基于 Web 服务的应用的繁荣,在带来了机遇的同时也带来了挑战.随着网络中 Web 服务的规模越来越大,具有相同功能的服务开始出现,如何在功能相同的服务中寻找最佳服务是一个富有挑战的问题.图 1 中的轻松聚会应用中,用户使用休闲搜索服务进行搜索后,应用能在电子地图中显示搜索结果的位置,并通过短信或者邮件的方式将聚会的信息发送给用户想要邀请的人员.轻松聚会使用 4 个 Web 服务,休闲搜索服务,地图服务,邮件服务和短信服务,这 4 种服务在网络中存在多种选择,从功能相同的服务中挑选出最佳的服务,能极大的提高应用的可用性以及用户的体验.

传统的 Web 服务选择方法侧重于服务的质量,致力于寻找服务质量最优的服务.这类服务质量驱动的 Web 服务选择方法只能在一定程度上提高应用的可用性和用户的体验.但

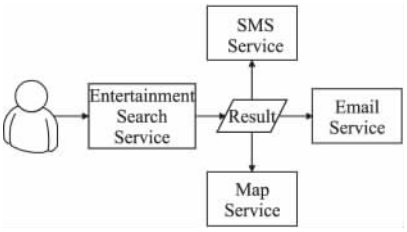

图 1 轻松聚会
Fig. 1 Easy party

是因为他们忽略了复杂的不断变化的服务环境,他们不能给不同的用户提供不同的解决方案,也不能在不同的环境里给出不同的解决方案.比如服务质量最优的邮件服务是一个英

收稿日期:2011-05-02 收修改稿日期:2011-06-10 基金项目:国家高技术研究发展计划项目(2007AA010306)资助;国家自然科学基金重大项目(61035004)资助;国家自然科学基金项目(61170212)资助. 作者简介:罗 森,男,1986 年生,硕士研究生,研究方向为 Web 服务选择、Web 服务组装;许 斌,男,1973 年生,博士,副教授,研究方向为 Web 服务、语义 Web、感知网络;孙科武,女,1986 年生,硕士研究生,研究方向为 Web 服务组装.

文的服务,服务质量驱动的 Web 服务选择方法的结果会使得只懂中文的用户使用起来就会很吃力; 又比如服务质量最优的地图服务是谷歌地图,这样的选择结果可能使得一个习惯于使用百度地图的用户使用起来很不顺手. 为此本文提出了一种创新性的服务选择方法并将它运用到 iWeb 框架之中,该方法不仅使用服务质量信息,还使用上下文信息来进行服务选择,以进一步提高应用的可用性和用户的体验.

框架的贡献主要包括以下几点.

· 实现了上下文信息收集系统. 通过上下文信息特征的分析,建立了上下文信息模型. 并且根据模型实现了上下文信息收集系统. 系统可以作为上下文信息研究领域的一个基础设施.

· 建立了服务索引. 收集了网络中数以千计的服务,并对这些服务使用聚类的方法进行功能性分类. 服务索引为 Web 服务研究领域的相关研究提供了重要的基础.

· 提出并实现了一种创新性的服务选择方法. 这种方法极大提高了应用的可用性和用户的体验.

· 实现了功能强大的服务引擎. 服务引擎负责服务的选择,服务的调度以及服务的调用,使得应用具有了动态的自适应性.

· 提高了应用的开发效率. 使用 iWeb 开发的应用具有普通框架开发的应用所没有的优点—应用具有自适应性(应用根据服务质量和上下文信息动态调用服务实现其功能) ,而在使用 iWeb 开发应用的过程中,较普通框架相比,开发人员不需要投入额外的时间和精力.

2　框架整体介绍

如图 2 所示,框架可以分为 3 层.

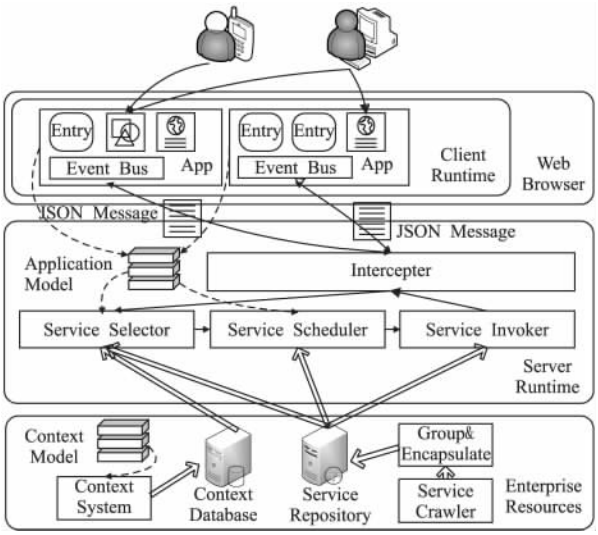

图 2　框架概述

Fig. 2　Framework overview

上层负责与用户的交互,用户界面是这一层应用的重要组成部分,而服务调用不是上层的工作. 上层会将调用任务转移到中间层,并将中间层返回的结果呈现给用户,因此在上层应用只需要服务调用入口. 上层通过 JSON 格式的信息与中间层进行沟通. 事件总线机制用来将服务调用入口和用户界面组织成一个统一的应用.

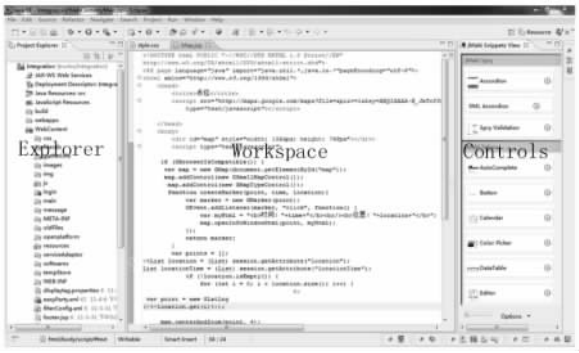

图 3　应用编辑器

Fig. 3　Web application editor

底层是框架的基础资源,主要包括上下文数据库和服务信息库. 上下文数据库存储上下文信息,这些信息由上下文信息收集系统根据上下文模型进行收集. 服务爬虫收集 Web 服务,这些服务按照功能进行分组和封装,在一个组内的服务具有相同的功能,并且由一个统一的接口进行封装. 服务信息库存储分组和封装的结果.

图 4　应用模型编辑器

Fig. 4　Application model editor

中间层是框架的核心. 服务引擎运行在这一层,包括拦截器,服务选择器,服务调度器和服务调用器. 拦截器用于捕获来自上层的服务调用请求,这个请求可以是请求调用单个服务,也可以是请求调用通过流程组合起来的复合服务. 捕获调用请求后,服务选择器会根据服务质量和上下文选择最佳服务. 如果请求是调用复合服务,调度器会调度这个复合请求. 最后,调用器会调用实际的服务,并将结果返回给拦截器. 拦截器通过 JSON 格式的信息将结果传递到上层.

上层的应用在中间层有一个对应的应用模型,这个模型为服务引擎提供关于这个应用的信息. 使用 iWeb 框架进行应用开发,开发者只需要开发用户界面和应用模型. 框架还提供了用户界面开发工具和应用模型开发工具,来进一步减少开

发者的负担.

3 框架工具

在文章上一节我们已经提到,使用 iWeb 框架进行网络应用开发,开发人员只需要开发框架中的最上层和其对应的应用模型. 框架提供了应用程序编辑器和应用模型编辑器,来进一步减少开发人员的负担.

应用编辑器是一个 Eclipse 插件,他提供了项目管理,将应用程序按照标准的发布格式进行管理. 此外通过提供通用的用户控件和一些有用的机制,比如事件发布订阅机制,定时器机制等,来加速开发人员进行开发.

应用模型编辑器也是一个可视化编辑器,用以提高应用模型开发的效率. 应用模型编辑器由概述编辑器和流程编辑器组成.

应用编辑器和应用模型编辑器提高了框架的可用性,降低了开发难度,缩短了开发周期.

4 框架实现

4.1 网络应用

使用 iWeb 框架开发的网络应用由 HTML 代码,CSS 代码和 JavaScript 代码组成. 网络应用通过浏览器呈现给用户,实现了平台独立性. 一个网络应用通常包括用户界面,服务调用入口以及他们之间的逻辑.

用户界面负责用户与应用之间的交互,框架提供了常用的预定义布局和用户界面控件,来方便开发人员进行用户界面的开发.

实际的服务调用是由服务引擎完成的,所以在网络应用

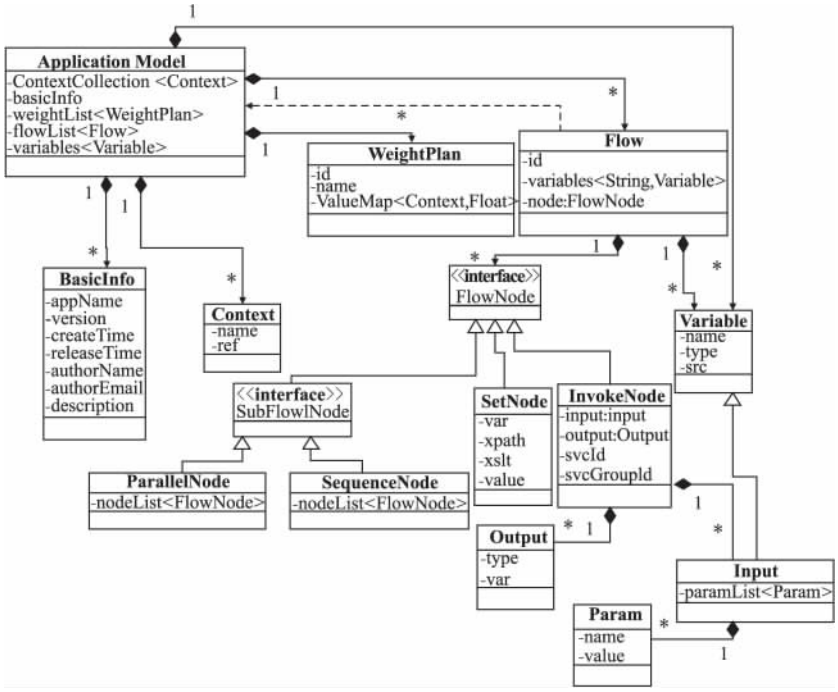

图 5 应用模型

Fig. 5 Application model

中只有服务调用的入口. 当某个功能需要完成或者某些信息需要获取时,应用会找到相应的服务调用入口,将调用请求传递到服务引擎进行实际调用,然后将结果返回界面呈现给用户.

用户界面和服务调用入口之间的逻辑通过事件总线的方式实现. 应用中所有的事件都会发布到事件总线上,通过事件的订阅和发布,用户界面和服务调用入口之间的逻辑就能得到建立. 除了事件订阅发布机制以外,框架还提供了注入器机制,定时机制等,来进一步减轻开发人员的负担.

每一个网络应用都有一个对应的应用模型,如图 5 所示,它形式化的对网络应用进行描述,帮助服务引擎进行服务选择和调度. 应用描述模型包含网络应用的基本信息,用于服务选择的权重方案,流程,在应用中使用到的上下文信息和流程间的中间变量.

4.2 基础资源

4.2.1 上下文数据库

iWeb 框架的一个重要特点就在于上下文信息的使用,上下文信息能在网络应用中方便的使用,而服务选择也是基于服务质量和上下文信息进行的.

要做到这些,上下文信息的收集是必要的. 环境中任何相关的信息都是上下文信息,上下文信息的种类很多,因此需要一个上下文模型来对上下文信息进行分类描述. 而要想建立一个完美的上下文信息模型基本上是一个不可能的任务,环境是在不停变化的,事物可能在环境中出现或者消失,因此上下文信息模型需要扩展性.

针对上下文信息的特点,我们使用 OWL [23] 来建立上下文模型. OWL 语言是形式化描述语言,具有良好的可扩展性. 由于上下文信息在不同领域差异很大,在语义上将上下文模型上分为两个部分,如下页图 6 所示. 第一部分是上层本体,提供了代表一般概念的基本词汇. context 类是声明上下文信

息模型的入口. 它被 4 个子类继承,计算上下文,用户上下文,物理上下文和时间上下文. 这意味着在模型中上下文信息被分为了 4 大类. 这 4 大类也被更具体的子类继承. 上下文模型的另外一个部分就是特定领域本体,它是底层本体,通过属性值来对通用概念的细节进行描述. 上下两部分一起构成了上下文模型.

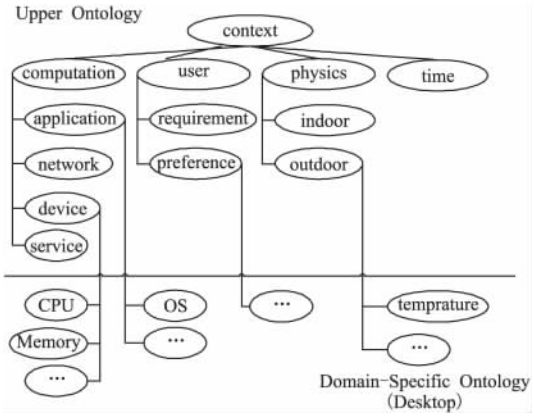

图 6　上下文模型

Fig. 6　Context model

在上下文信息模型的基础上,我们实现了上下文信息系统,系统可以从不同的用户不同的平台收集上下文信息. 除了上下文信息的收集,上下文信息系统对这些不同来源的上下文信息进行管理,并提供统一的接口以供查询使用.

4.2.2　服务信息库

网络应用的实际功能是通过调用服务完成的. 服务是框架的基本要素,因此我们使用一个爬虫程序从网络上收集数以千计的.

随着服务的迅速发展和普及,网络上还存在具有相同功能的服务,我们将服务进行功能性分组,使得用服务选择来提高网络应用的灵活性成为了可能.

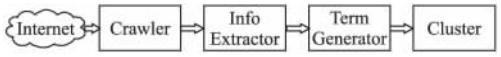

图 7　服务分组

Fig. 7　Service grouping

服务分组的流程如图 7 所示,当爬虫程序完成服务的收集以后,信息提取器会抽取收集到的服务的信息,比如类型,消息,端口等. 然后项生成器会用这些信息生成项,包括 4 个子步骤,词法分析,标记移除,分割词移除和向量生成. 最后,聚类器会使用 K-MEAN [31] 算法对服务进行功能性分组. 自动分组并不能保证完全准确,我们还会手动的进修校正,让分到一个组的服务具有相同的功能.

当服务分组完成后,虽然一个组内的服务具有相同的功能,但是他们可能有结构不同的输入输出. 我们使用一个统一的接口对同一组内的服务进行封装,以方便这些服务的使用. 封装一个服务需要 3 个要素要素,一个 Java [24] 文件,一个 XML [25] 配置文件和一个 XLST [26] 转换文件.

Java 文件需要实现下面所示的 Java 接口.

XLST 转换文件用于将服务返回的结果转换为一个统一的 JSON 格式的字符串. 为一个组内的服务都编写相应的 XLST 转换文件,服务调用者就能用相同的输入参数调用一个组内的所有服务,并得到结构相同的输出结果.

表 1　Java 接口

Table 1　Java interface

```
public interface ServiceAdapter{
    //The protocol to be used for calling the service, could be get, post, SOAP1.1, SOAP1.2, etc.
    public abstract Protocol getProtocol();
    //The URL of the service.
    public abstract String getServiceURL();
    //The URL of the SOAPAction, return null if the protocol is not SOAP1.1 or SOAP1.2.
    public abstract String getSOAPActionURL();
    //The map structure params represents the uniform input parameter for the group, this interface transforms the uniform input parameter to the parameter the service needs.
    public abstract String getStructuredParams(Map<String, String> params);
    //Get the XLST file.
    public abstract InputStream getXSLT();
    //Return the result in string using unified format.
public abstract String postResult();
}
```

XML 配置文件记录封装的相关信息. 有了这 3 个文件,同一组的服务调用就不会有任何区别. 在此基础上,框架就能直接使用服务组进行开发,而不是使用单个服务进行开发. 我们已经收集了几千个服务,将服务分为了上百个组并进行了相应的封装,储存在服务信息库中,作为框架的基础资源.

4.3　服务选择

服务选择是 iWeb 框架的重要特点,它使得使用 iWeb 开发的网络应用具有了自适应性.

随着网络上具有相同功能但表现出不同的非功能性的服务的出现,服务选择变得可行. 服务选择是从一组功能相同的服务中选择出最佳的服务. 基于服务质量的服务选择从一组服务中选择出服务质量最优的服务,近年来已经得到了广泛的研究. 表 2 列出了一些常用的服务质量.

表 2　常用服务质量

Table 2　Common QoS

Cost	使用服务需要支付的费用
Reliability	服务的可靠程度
Reputation	基于用户反馈的服务信誉
Throughput	服务的吞吐量
Response Time	服务的响应时间

基于服务质量的服务选择使用效用函数来评估服务. 效用函数最普遍的计算方法是使用权重加权的方式,如公式(1).

$$F_i = \sum_{j=1}^{n} w_j \cdot f_{ij}, \text{where } \sum_{j=1}^{n} w_j = 1 \text{ and } 0 \leqslant w_j \leqslant 1 \quad (1)$$

其中对象函数 f_{ij} 表示单个服务质量的评价得分,由于服务质量的多样性,加总为 1 的权重系数 w_j 被使用来反映不同

服务质量之间的权衡. 虽然基于服务质量的服务选择方法能在一定程度上提高基于服务的应用的灵活性,但是他们有着局限性. 他们不能在不同的环境中提供不同的解决方案,因为他们完全忽略了复杂的,高速变化的环境.

针对这些缺陷,我们提出了一种改进的服务选择方法,这种方法既考虑服务质量,也兼顾环境信息. 服务选择是从服务的非功能性方面进行选择,在这个方法中,服务的非功能属性不仅包括服务质量属性,还包括上下文属性,如图 8 所示.

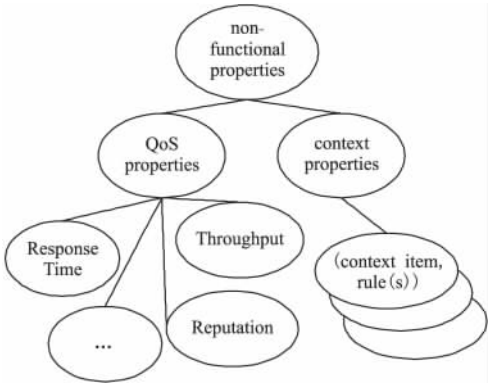

图 8 非功能属性

Fig. 8 Non-functional properties

服务质量属性通常是一些数值,而上下文属性应该告诉服务如何响应环境. 使用前面提出的上下文信息模型,环境信息会通过上下文信息项集合的方式呈现. 我们将上下文属性设置为键值对,其中的键是上下文属性所关联的上下文信息项,而值通过规则的形式表示服务如何对关联的上下文信息项进行响应.

在值中,规则类似于一阶谓词逻辑,由常量,变量,谓词和逻辑符号等基本元素组成. 基本元素的细节信息在表 3 中给出.

使用这些元素,我们可以通过递归的方式来定义 AssertClause.

1) 如果 A 是作用在变量上的谓词,那么 A 是 AssertClause;

2) 如果 A 是 AssertClause,那么 ~A 是 AssertClause;

3) 如果 A 和 B 都是 AssertClause,那么 $A \wedge B$ 是 AssertClause;

4) 如果 A 和 B 都是 AssertClause,那么 $A \vee B$ 是 AssertClause.

表 3 规则元素

Table 3 Rule elements

Constants	Individual values, sets and ranges
Predicate Symbols	LessThan, NotEqual, Equal, LessThanOrEqual GreaterThan, etc which compare individual value with another; ContainedIn, NotContainedIn which judge relation for individual value and set; Within, NotWithin which judge relation for individual value and range
Variables	attributes for context items which can be individual values, sets and range
Logical Operators	~, $\wedge$, $\vee$, $\rightarrow$

$$\text{AssertClause} \rightarrow \text{Score}(n) \tag{2}$$

规则具有公式(2)的形式,表示如果 AssertClause 是真的,那么服务会得到一个分数 n,n 在 0 到 1 之间,否则会得到 0.5 分(表示对上下文信息项响应不大).

对象函数表示单个服务质量属性或者单个上下文属性的得分. 假设 G 是一个包含服务 $S_1,\cdots,S_n$ 的服务组,对于其中的服务 S_i,其服务质量属性为 $q_{i1},\cdots,q_{im}$,上下文属性为 $q_{i(m+1)},\cdots,q_{io}$. 服务质量属性的目标函数计算方式如公式(3)(4). 公式(3)表示该服务质量属性越大越好,而公式(4)表示该服务质量属性越小越好. 上下文属性的对象函数计算方式如公式(5),即使用规则计算.

$$f_{ij} = \frac{q_{ij}}{\max(q_{kj})\ (k=1,\cdots,n)} \tag{3}$$

$$f_{ij} = \frac{\min(q_{kj})\ (k=1,\cdots,n)}{q_{ij}} \tag{4}$$

$$f_{ij} = r_{ij}() \tag{5}$$

则服务 S_i 的效用可以通过公式(6)计算,这里假设前 x 个服务质量属性越大越好.

$$F_i = \sum_{j=1}^{x} w_j \frac{q_{ij}}{\max(q_{kj})\ (k=1,\cdots,n)} + \sum_{j=x+1}^{m} w_j \frac{\min(q_{kj})\ (k=1,\cdots,n)}{q_{ij}} + \sum_{j=m+1}^{o} w_j r_{ij}() \tag{6}$$

如公式(6)所示,服务的效用由 3 部分组成,其中前两部分计算服务质量属性的效用,一部分计算上下文属性的效用. 使用这样改进的效用计算方法,服务质量和上下文信息都得到了考虑,使得服务选择不仅能选出服务质量最优的服务,更能选出最符合环境的服务,让基于服务的应用更加灵活.

4.4 服务引擎

实现网络应用实际功能的服务是通过服务引擎进行调用的,服务引擎是框架的核心部分.

服务引擎在服务运行时中运行,由拦截器,服务选择器,服务调度器和服务调用器组成.

拦截器负责和网络应用进行通信,并且在服务调用前和服务调用后都提供了回调函数,以提供更高的可操作性.

通过使用上节的服务选择方法,服务选择器根据服务质量和上下文信息选择最佳的服务.

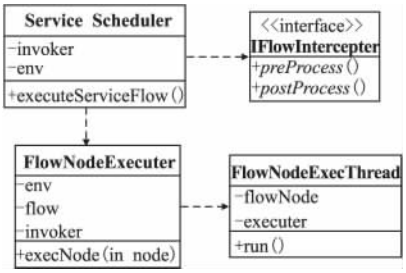

图 9 服务调度器

Fig. 9 Service scheduler

服务调度器主要功能可以分为 5 个部分,拦截信息解析,串行服务调用,并行服务调用,单一服务调用以及参数的调配. 在应用模型中分别对应 IntercepterInfo,SequenceNode,ParallelNode,InvokeNode 和 SetNode. 服务调度器的 UML 图如图 9 所示. 服务调用器会根据服务选择的结果,通过服务封装后的统一接口调用具体服务,并将响应根据 XLST 转换成统一的结构. 服务调度器则负责调用器调用服务的逻辑顺序.

服务引擎通过调用服务完成网络应用的功能,其间利用服务选择方法以及基础资源来实现提高网络应用的灵活性.

5　框架可用性分析

本文的服务选择方法是在经典的服务选择模型上提出的. 即从功能相同的一组服务中选择出最优的服务. 经典服务选择模型在实际中是否可用,即实际网络中是否存在功能完全相同的服务,是否存在提供各种各样功能的功能组,近年来也得到了广泛的研究和讨论. 在本节中,我们通过对网络上实际服务的分析,验证了经典服务选择模型在实际中是可用的,使用这种模型的 iWeb 框架也是实际可用的.

按照 4.2.2 小节提出的服务分组流程,首先从网络上搜集了 13,568 个实际可用的服务,然后将这些服务分为 300 个组. 通过人工确认,发现确实存在数百个提供各种各样功能的网络服务. 比如航班服务提供航班查询功能,而我们收集到了 14 个这样的服务; 邮件服务提供邮件发送接受功能,这样的服务我们发现了 34 个; 短信服务提供短信的发送功能,这样的服务我们收集到了 165 个之多.

通过分析,实际网络中确实存在提供各种功能的服务组,这些服务组又包含若干提供相同功能的服务. iWeb 框架确实是实际可用的.

使用 iWeb 开发的基于服务的网络应用,还具有提高网络应用质量,增加网络应用自适应性,提高开发人员效率等优点.

框架中的服务选择能从功能性相同的服务组内挑选出服务质量最好的服务,以提高该服务乃至整个网络应用的质量. 并且当组内服务质量差异越大时,服务质量提高越明显.

框架提供了为网络应用添加上下文规则的机制,上下文规则能将不同的环境,不同的用户进行了区分,使得网络应用能针对不同环境,不同用户来选择服务,提高了网络应用的自适应性. 并且上下文规则越多,网络应用的自适应性越强,呈现指数级增长的趋势.

框架对具有相同功能的服务组进行了封装,组内服务的选择替换不需要开发人员的干预,节省了大量的开发时间.

6　相关工作

Web Mashups 是整合网络上分散资源的一种应用[1]. 研究人员已经创造了各种各样的 Web Mashups 工具和平台,比如 Yahoo Pipes[27],Google Mashup Editor[28],Microsoft´s Popfly[29] 和 IBMs QEDWiki[30]. 然后,在 Web Mashups 中网络编程接口和 RESTful 服务是主流. 对于 Web 服务的整合并不是一个简单的工作,跟不要说创造具有灵活性的基于 Web 服务的应用.

服务选择能够用来强化基于服务的应用的用户体验. 现在已经有很多服务选择方法被提出. Zeng[2,3] 针对服务组装中的服务质量的服务选择提出了一种全局规划的方法. Tao[4] 提出了一种代理架构,并在架构上实现了多种混合方法来寻找最佳服务. Alrifai[5] 提出结合全局优化和局部优化的方法,来寻找最佳服务. 然而,这些方法都集中在服务质量上,而没有考虑环境信息,他们只能在一定程度上提高基于服务的应用的灵活性.

上下文常在服务发现和服务推荐中用到. Chen[6] 设计了一个事件驱动规则系统,来根据上下文信息的变化推荐服务. Yang[7] 实现了一个系统,系统能找到根据服务请求寻找上下文匹配的服务. Hua[8] 提出根据上下文之间的关联来推测用户可能需要的服务. 在这些方法或系统中,上下文被用来进行用户意图的推测. 然而在 iWeb 框架中,基于服务的应用的功能性对上下文并不敏感,现存的上下文信息研究并不能被直接利用.

7　结束语

本文介绍了基于服务的网络应用开发框架,iWeb. 框架使用服务质量信息和上下文信息进行服务选择以提高应用的可用性和用户的体验.

框架实现了上下文信息系统和服务信息索引. 上下文信息系统在上下文模型的基础上系统的收集上下文信息,该系统可以作为上下文信息研究领域的基础设施; 服务信息索引收集了数以千计的服务,并将服务按功能进行分组和封装,让服务选择变得切实可行.

本文提出了一种创新性的服务选择方法,这个方法可以根据服务质量信息和上下文信息选择最佳的服务. 框架实现的服务引擎负责使用这种创新性的方法来进行服务选择,还负责服务的调度和调用,是实现自适应性的核心所在.

框架还提供了应用程序编辑器和应用模型编辑器等开发工具以加速应用的开发. 使用 iWeb 框架,可以在不增加开发成本的基础上,创造出具有自适应性的基于服务的应用.

References:

[1] Jin Yu, Boualern Benatallah, Fabio Casati, et al. Understanding mashup development [J]. IEEE Internet Computing, 2008, 12(5): 44-52.

[2] Zeng L, Benatallah B. QoS-aware middleware for Web service composition [J]. IEEE Transactions on Software Engineering, 2004, 30(5): 311-327.

[3] Zeng L, Benatallah B, Dumas M, et al. Quality driven Web services composition [C]. In Proceedings of International Conference on World Wide Web(WWW), Budapest, Hungary, ACM Press, May, 2003.

[4] Yu T, Zhang Y, Lin K J. Efficient algorithms for Web services selection with end-to-end QoS constraints [C]. In ACM Transactions on the Web, Vol. 1, No. 1, Article 6, May, 2007.

[5] Alrifai M, Risse T. Combining global optimization with local selection for efficient QoS-aware service composition [C]. In Proceedings of International Conference on World Wide Web(WWW), April, 2009.

[6] Chen I Y L, Yang S J H, Jiang J. Ubiquitous provision of context aware web services [C]. IEEE International Conference on Services Computing (SCC), Chicago, USA, 2006: 60-68.

[7] Yang S J H, Zhang J, Chen I Y L. A JESS-enabled context elicitation system for providing context-aware Web services [J]. Export Systems with Applications, 2008, 34(4): 2254-2266.

[8] Hua Xiao, Ying Zou, Joanna Ng, et al. An approach for context-aware service discovery and recommendation [C]. In Proceedings of International Conference on Web Service (ICWS), August, 2010.

[9] Khalid Elgazzar, Ahmed E Hassan, Patrick Martin. Clustering WS-

DL documents to bootstrap the discovery of Web services [C]. In Proceedings of International Conference on Web Service(ICWS), August,2010.

[10] Qi Zhao,Liu Xuan-zhe,Sun Da-wei,et al. Mashing-up rich user interfaces for human-interaction in WS-BPEL [C]. In Proceedings of International Conference on Web Service (ICWS),August,2010.

[11] Braga D,Ceri S, Daniel F,et al. Mashing up search services [J]. IEEE Internet Computing,2008,12(5):16-23.

[12] Kyriakos Kritikos, Dimitris Plexousakis. Mixed-integer programming for QoS-based web service matchmaking [J]. IEEE Transactions on Services,2009,2(2):122-139.

[13] Al-Masri E,Mahmoud Q H. Discovering the best web service [C]. In WWW,2007:1257-1258.

[14] Xia Wang, Tomas Vitvar, Mick Kerrigan,et al. A QoS-aware selection model for semantic Web services [C]. International Conference on Service Oriented Computing,2006:390-401.

[15] Luo Sen,Xu Bin,Sun Ke-wu. Compose real Web service with context [C]. 2010 IEEE International Conference on Web Services (ICWS 2010),Miami, Florida, USA,2010:5-10.

[16] Zhao Xiang-peng,Qiu Zong-yan,Cai Chao,et al. A formal model of human workflow [C]. Proceedings of 2008 IEEE International Conference on Web Services (ICWS 2008),2008:195-202.

[17] Kwei-Jay Lin,Zhang Jing,Zhai Yan-long,et al. The design and implementation of service process reconfiguration with end-to-end QoS constraints in SOA [J]. Journal of Service Oriented Computing and Applications (SOCA),2010,4(3):157-168.

[18] He Q, Yan J, Jin H,et al. Adaptation of web service composition based on workflow patterns [C]. In: Proceedings of International Conference on Service-oriented Computing (ICSOC), 2008.

[19] Jurca R,Faltings B,Binder W. Reliable QoS monitoring based on client feedback [C]. 16th International Conference on World Wide Web (WWW2007), ACM, 2007:1003-1012.

[20] Daniel A Menascé,Emiliano Casalicchio,Vinod Dubey. On optimal service selection in service oriented architectures [M]. Performance Evaluation Journal,North-Holland,Elsevier Science,In Press,July, 2009.

[21] Liu Xuan-zhe,Sun Wei,Hui Yi,et al. Investigating service composition based on mashup [C]. Service Congress,2007:332-339.

[22] Web ontology language [EB/OL]. http://www.w3.org/2004/OWL/,2009.

[23] Web service definition language [EB/OL]. http://www.w3.org/TR/wsdl/,2009.

[24] JAVA [EB/OL]. http://http://www.oracle.com/technetwork/java/index.html/,2010.

[25] Extensible makeup language [EB/OL]. http://www.w3.org/XML/,2009.

[26] Extensible stylesheet language transformations [EB/OL]. http://www.w3.org/TR/xslt/,2009.

[27] Yahoo pipes [EB/OL]. http://pipes.yahoo.com/,2010.

[28] Google mashup editor [EB/OL]. http://code.google.com/gme/,2010.

[29] Microsoft's popfly [EB/OL]. http://www.popfly.com/,2010.

[30] IBM's QEDWiki [EB/OL]. http://services.alphaworks.ibm.com/qedwiki/,2010.

[31] Jain AK, Dubes RC. Algorithms for clustering data [M]. Prentice-Hall, Englewood Cliffs, 1988.

[32] Hu Jian-qiang,Li Juan-zi,Liao Gui-ping. A multi-Qos based local optimal model of service selection [J]. Chinese Journal of Computer,2010,33(3):526-534.

[33] Dong Yuan-yuan,Ni Hong,Deng Hao-jiang,et al. Service selection strategy offering global optimal quality of service [J]. Journal of Chinese Computer Systems,2011,32(3):455-459.

[34] Li Jin-zhong,Xia Jie-wu,Tang Wei-dong,et al. Survey on Web services selection algorithms based on Qos [J]. Application Research of Computers,2010,27(10):3622-3628.

[35] Li Man,Li Jian-hua. Study on Web service discovey based on service grouping [J]. Engineering and Application,2010,46(21):162-164.

[36] Hu Jia,Feng Zhi-yong. Semantic Web service composition framework based on abstract template [J]. Computer Applications,2009, 29(11):3150-3154.

[37] Li Yan,Zhou Ming-hui,Li Rui-chao,et al. Service selection approach considering the trustworthiness of QoS data [J]. Journal of Software,2008,19(10):2620-2627.

附中文参考文献:

[32] 胡建强,李涓子,廖桂平. 一种基于多维服务质量的局部最优服务选择模型 [J]. 计算机学报,2010,33(3):526-534.

[33] 董元元,倪 宏,邓浩江,等. QoS 全局最优的服务选择策略 [J]. 小型微型计算机系统,2011,32(3):455-459.

[34] 李金忠,夏洁武,唐卫东,等. 基于 QoS 的 Web 服务选择算法综述 [J]. 计算机应用研究,2010,27(10):3622-3628.

[35] 李 曼,李建华. 基于服务分组的语义 Web 服务发现研究 [J]. 计算机工程与应用,2010,46(21):162-164.

[36] 胡 佳,冯志勇. 一种基于抽象模板的语义 Web 服务组合框架 [J]. 计算机应用,2009,29(11):3150-3154.

[37] 李 研,周明辉,李瑞超,等. 一种考虑 QoS 可信性的服务选择方法 [J]. 软件学报,2008,19(10):2620-2627.