第42卷 第6期 西 安 交 通 大 学 学 报 Vol.42 №6
2008年6月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Jun. 2008

可用性语义Web服务的通用发现机制

朱正东[1]，缪相林[1]，胡亚红[2]，李增智[1]
(1.西安交通大学计算机科学与技术系，710049，西安；2.浙江工业大学信息工程学院，310032，杭州)

摘要：针对复杂组合Web服务的选择问题，提出了一种基于可用性的语义Web服务发现方法.使用动态自适应模板来寻找可用的组合Web服务，结合语义匹配度量化Web服务的功能、非功能属性，在形式化定义组合Web服务的可用性等规则的情况下，基于可用性的量化值进行排序，从而得到最合适的Web服务.将以建议规则实现的匹配引擎运行于原型系统，结果表明所提方法可以实时选择复杂组合Web服务，适用范围也比较大，并在查全率的基础上能够获得最优解.

关键词：语义Web服务；复杂组合；服务发现
中图分类号：TP311 **文献标志码**：A **文章编号**：0253-987X(2008)06-0659-05

Usability-Based General Semantic Web Services Discovery Mechanism

ZHU Zhengdong[1]，MIAO Xianglin[1]，HU Yahong[2]，LI Zengzhi[1]
(1. Department of Computer Science and Technology，Xi′an Jiaotong University，Xi′an 710049，China；2. College of Information Engineering，Zhejiang University of Technology，Hangzhou 310032，China)

Abstract：A discovery method for usability-based semantic Web services is presented to deal with the service selection in complex composite Web services. The useful composite Web services are discovered using a dynamic adaptive template under the considerations of semantic matching degree，and rules such as the usability of composite Web services that are formally defined through the quantification of the functional property and non-functional property of Web services. The most suitable Web service is obtained according to the ordered values of the usability quantification for all the services. A matching engine using the proposed rules has been applied to a prototype system. Experiment results show that the proposed discovery method can realtimely fulfill the selection of complex composite Web services. Moreover，the optimal service can be obtained based on the premise of finding more available Web services.

Keywords：semantic Web service；complex composite；service discovery

近年来，在语义Web服务中，基于功能属性的匹配选择规则一般用来描述实例与客户需求之间的输入、输出、前提、效果(IOPE)的语义相似性匹配[1-3]，而在普适机器人伙伴的语义Web服务发现(URCSD)[4]中，将服务模板作为需求与具有相兼容特性的Web服务进行比较，并按多个关键词进行多次排序，以寻找最合适的组合Web服务.基于非功能属性的匹配选择规则，多数是基于简单组合Web服务的QoS属性计算的[5-7]，可以很好地满足用户要求，但却未将功能属性和非功能属性同时作为不可缺的因素来考虑，也未考虑语义匹配程度的影响，所以它不适用于具有并发服务的复杂组合Web服务.

针对复杂组合Web服务的选择问题，本文提出了一种可用性语义Web服务的通用发现机制.

1 组合Web服务发现

根据用户需求可以获取本体知识库中的动态自适应抽象模板，该模板是语义Web服务经自动组合后保存于本体知识库的解，其由几个描述明确的Web服务块组成.根据块可以找到所有与块完全匹

收稿日期：2007-09-01. 作者简介：朱正东(1963—)，男，讲师. 基金项目：国家自然科学基金资助项目(60673170).

配、或不完全匹配但功能相同的代用的可用 Web 服务，可用的 Web 服务根据模板进行自动组合，从而形成可用的组合 Web 服务.

一种以名称订购商品的二级抽象模板如图 1 所示，它是最简单的复杂组合 Web 服务，存在 2 个并发服务(或称任务)，即购物信息服务、网上银行服务. 订购商品中可用的 Web 服务如表 1 所示，这些服务可以是代用的可用 Web 服务，必须被 Web 服务的发现结果所包括. 在设计匹配引擎时，应考虑到代用的 Web 服务.

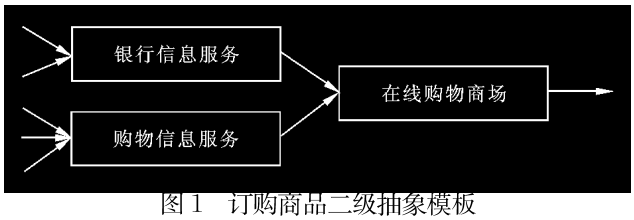

图 1 订购商品二级抽象模板

表 1 订购商品中可用的 Web 服务

服务类型	输入	输出
WS11* 银行信息服务	账号，密码	付款状态
WS12 银行信息服务	账号，密码，姓名，地址	付款状态，地址
WS13 银行信息服务	账号，密码，身份证号	付款状态
WS14 银行信息服务	账号，密码，姓名，身份证号	付款状态
WS21* 购物信息服务	商品名称，规格，品牌	商品号
WS22 购物信息服务	商品名称，规格，公司名	商品号，送货方式
WS31* 在线购物商场	商品号，付款状态	送货信息
WS32 在线购物商场	商品号，送货方式，付款状态	送货信息，地址

注：* 表示与模板完全匹配的服务，其他表示不完全匹配但能为任务所使用的服务.

2 组合 Web 服务可用性

可用性是根据 Web 服务与需求的不同语义匹配度、用户对 Web 服务的评价、Web 服务执行情况，以属性重要程度定制的权重和减少无意义结果原则定义的 Web 服务的可用程度.

定义 1 组合 Web 服务可用性是指，满足用户需求的组合 Web 服务的可用程度，表达式为

$$U_{WS}=\omega U_{IOPE}+(1-\omega)U_{QoS} \tag{1}$$

式中：U_{IOPE}为功能属性可用性；U_{QoS}为非功能属性可用性；ω 为反映功能属性在组合 Web 服务中的重要程度的权重. U_{WS}可以反映组合 Web 服务满足于用户的程度，对该值进行排序可以选择最合适的 Web 服务.

2.1 功能属性可用性

定义 2 功能属性可用性是指，在需求与服务名称符合“精确”匹配的前提下，输入(前提)、输出(结果)的可用程度，表达式为

$$U_{IOPE}=U_{Input}+U_{Output} \tag{2}$$

式中：U_{Input} 为输入可用性，用输入匹配度量化；U_{Output}为输出可用性，用输出匹配度量化.

定义 3 语义匹配度有 3 种：①如果 A 与 B 概念相等，则匹配度为“精确”；②如果 A 包含 B，则匹配度为“可替代”；③如果 B 包含 A，则匹配度为“包含”；④如果 A 与 B 没有关系，则匹配度为“失败”. 本文使用 Massimo Paolucci 定义的 2 个概念匹配度[8]来计算 IOPE 的语义匹配度.

定义 4 输出匹配度是指，假设需求的输出参数集合为 R_O，Web 服务的输出参数集合为 A_O，当 $A_O\supseteq R_O$，表示多个服务输出满足需求，否则匹配失败. 该输出匹配度计算如图 2 所示.

前提：按Output搜索所有单个$A_O\cap R_O\neq$null的Web服务
$U_{Output}=0$
if多个或单个服务$A_O\supseteq R_O$
$U_{Output}=((R_{O\alpha}/R_O)\alpha_O+(R_{O\beta}/R_O)\beta_O+(R_{O\gamma}/R_O)\gamma_O$
else
$U_{Output}=0$ ；匹配失败

$R_{O\alpha}$、$R_{O\beta}$、$R_{O\gamma}$分别表示“精确”、“可替换”、“包含”匹配的输出集合；α_O、β_O、γ_O 分别为输出的“精确”、“可替代”、“包含”匹配的权重

图 2 输出匹配度的计算

定义 5 输入匹配度是指，假设需求的输入参数集合为 R_I，Web 服务的输入参数集合为 A_I，当 $A_I\supseteq R_I$，可以认为虽然没有提供足够的输入参数，但可以补充输入且满足需求；当 $R_I\supseteq A_I$，可以认为提供了多余的输入，匹配度受到了影响. 该输入匹配度计算如图 3 所示.

前提：多个或单个Web服务$A_O\supseteq R_O$；// 输出满足需求 //
$U_{Input}=0$
if多个或单个服务$A_I\supseteq R_I$
$U_{Input}=(R_{I\alpha}/R_I)x_I+(R_{I\beta}/R_I)\beta_I+(R_{I\gamma}/R_I)\gamma_I$
else
if $R_I\supseteq A_I$
$U_{Input}=(R_{I\alpha}/R_I)\alpha_I+(R_{I\beta}/R_I)\beta_I+(R_{I\gamma}/R_I)\gamma_I-(R_I-A_I)(1/R_I)\theta_I$
else
$U_{Input}=0$;匹配失败

$R_{I\alpha}$、$R_{I\beta}$、$R_{I\gamma}$分别为 “精确”、“可替换”、“包含”匹配的输入集合；α_I、β_I、γ_I、θ_I 分别为输入的“精确”、“可替代”、“包含”、“失败”匹配的权重

图 3 输入匹配度的计算

用户可以定制权重参数，但要求满足 $0\leqslant\alpha_O+\beta_O+\gamma_O+\alpha_I+\beta_I+\gamma_I+\theta_I\leqslant 1$，同时还应该考虑输出权重大于输入权重.

2.2 非功能属性可用性

定义 6 Web 服务非功能属性可用性为[5]

$$U_{QoS}=\sum Q_k\Phi_k \quad 0\leqslant\Phi_k\leqslant 1,\sum\Phi_k=1 \tag{3}$$

式中：Q_k，$k\in[1\cdots N]$为第 k 个非功能属性统一量化值；Φ_k 是属性权重，表示第 k 个度量的重视程度. 属性数量不受限制，k 可取任意整数.

组合 Web 服务非功能属性有价格、响应时间、信誉度、成功率和可靠率等，对于复杂组合 Web 服务，有些 QoS 的概念及其量化定义与简单组合服务有所不同.

定义 7 组合 Web 服务响应时间是指，Web 服务从接受需求到返回结果之间的延迟时间，等于在最长路径时间上所有子 Web 服务的延迟时间之和，即

$$Q_e=\sum_{k=1}^{n}Q_{ETk} \tag{4}$$

式中：Q_{ETk}是最长路径时间上第 k 个子服务的延迟时间；n 表示该路径上所含子服务数.

为了统一量化 QoS，可将 QoS 分为 2 类：价格、响应时间等为消极度量，其值越大，性能越差；信誉度、成功率和可靠率等为积极度量，其值越大，性能越好. 消极度量、积极度量分别为

$$Q_N=\begin{cases}\dfrac{\max Q-Q}{\max Q-\min Q} & \max Q-\min Q\neq 0\\ 1 & \max Q-\min Q=0\end{cases} \tag{5}$$

$$Q_A=\begin{cases}\dfrac{Q-\max Q}{\max Q-\min Q} & \max Q-\min Q\neq 0\\ 1 & \max Q-\min Q=0\end{cases} \tag{6}$$

式中：$\max Q$、$\min Q$ 是基于模板的可用组合 Web 服务非功能属性的最大值和最小值；Q 是非功能属性的平均值. 这些值将保存在 QoS 数据库中.

综上所述，用户对组合 Web 服务可提出全局可用性需求与限制，目标函数 U_{WS} 值越大且满足全局 QoS、输出和输入匹配度限制的个体，其竞争力水平越高.

3 发现、选择算法

3.1 Web 服务发现算法

在接收到用户需求之后，用户需求名称应与知识库中模板的名称进行概念语义精确匹配，如果无精确匹配的模板，系统将使用动态自适应组合方法建立该模板，否则根据模板进行 Web 服务发现. 在 Web 服务发现算法中，QoS 的消极度量值必须小于模板的非功能属性最大值之和，而 QoS 的积极度量值必须大于非功能属性最小值之和，否则终止算法. Web 服务发现算法如图 4 所示.

```
输入：需求名称，需求性能参数
    （成功率Req.Qs，响应时间Req.Qe）//Req.Qs为积极度量//
输出：所有的可用组合Web服务
LWCn=();          //设置n个队列为空,用于存放组合服务//
GATHER=();        //可用Web服务集合//
Input(Reg.Name);  //输入需求名称//
Input(min.Reg.Qs, max.Reg.Qe);  //输入需求性能参数//
Exact_Search(Mod_Lib, Reg. Name); //在知识库精确查找//
if found() then GetMod. Para(); // 得到模板所有参数//
   else call Auto_comoine;     // 调用自动组合算法//
if min.Reg.Qs≥min.Mod.Qs and max.Reg.Qe≤max.Mod.Qe
else goto end;
for(i=1; i≤Mod. WS. Number; i++);    //Mod. WS. Number
                                     是模板中的块数//
  {j=1; Do While Exact_Search(WS_Lib, Mod. WSi. Name);
  //精确查找第i个Web服务//
      if found() and (minWSij. Qs≥minMod.WSi. Qs) and
      (maxWSij. Qe≤maxMod.WSi. Qe)
       then Addin(WSij, GATHER); mi=j; j=j+1;
   enddo; }
for (i=1, i≤Mod. WS. Number; i++); {if mi=1 then goto end}
      //对于某个块,如果没有相应的可用Web服务，结束//
for (n=1; n≤100; n++);    //n的最大值可以是m1m2m3…mi,
                          i是模板中的块数//
{for(i=1; i≤Mod. WS. Nmber; i++);
        {for(j=1; j≤mi; j++); {if not WSij∈ LWCn then
        LWCn=Addin(WSij); }
        }
   }
```

图 4　组合 Web 服务发现算法

3.2 Web 服务选择算法

首先，根据用户经验定制权值，并将模板功能块作为需求，在可用的组合 Web 服务中对各级 IOPE 进行概念语义匹配，计算各级输出、输入匹配度，计算各级功能属性的可用性 U^*_{IOPE}，并取其中最小值作为各组合 Web 服务的 U_{IOPE}. 其次，计算组合 Web 服务的非功能属性平均值、最大值和最小值，再计算统一量化值，得到 U_{QoS}. 最后，计算 U_{WS} 并进行排序，得到最合适的组合 Web 服务. Web 服务选择算法如图 5 所示.

4 原型系统及测试实验

4.1 原型系统结构

将定义的可用性规则用于发现机制，并运行于自主开发的原型系统（XJTU-WSDCS），结果如图 6 所示.

系统的“搜索引擎”根据发现、选择算法得到最优解，“执行引擎”基于模板将最优解转换为

BPEL4WS的具体工作流程，管理、执行与用户任务相应的 Web 服务.

“QoS 数据库”包括所有 QoS 的矩阵定义，存储每个(组合)Web 服务的相应值，支持那些原本不支持 QoS 属性存储的系统(如 UDDI). 知识库本体除了应有的功能，还能存放 Web 服务模板，该模板可由发布者定制，也可在第一次使用 Web 服务时由系统自动产生.

“Web 服务 QoS 发布”是为提供者输入服务 QoS 矩阵值的一个应用窗口，如服务的价格等. “Web 服务 QoS 反馈”是为用户使用 Web 服务进行评估而设置的应用接口.

4.2 测试用例及结果

将表 1 中的 4 个银行信息服务分别安装在 4 台语义 Web 服务器上，购物信息服务 WS21 与 WS22 安装于一台服务器，在线购物网站服务 WS31 与 WS32 安装于另一台服务器，用户终端为安装了 Windows XP 的工作站. 为了说明问题，在实验环境中只考虑不受用户主观影响的非功能属性，如成功率、可靠率和响应时间等，并认为成功率与可靠率为 100%，权重定义相同，即本文仅考虑响应时间.

```
输入：所有可用的组合Web服务，权重参数
输出：最优解
Input(LWC_n, α_O, β_O, γ_O, α_I, β_I, γ_I, θ_1, φ_1, φ_2, ω)；//参见发现算法//
 for (n=1,n≤100, n++)；          //n的最大数是组合服务数//
{for (k=1, k≤Mod. level), k++)；//Mod. level是组合服务级数//
      {get(WS_k, LWC_n);      //从队列中取同一级的Web服务WS_k//
      calculate(WS_k.U_Output);   //计算某级输出度//
      calculate(WS_k.U_Input);    //计算某级输入度//
      calculate(WS_k.U_IOPE);      //根据模板计算某级功能属性
                                   可用性//
      LWC_n.U_IOPE =min(WS_k.U_IOPE,k=1,..., Mod. level)
      calculate(minLWC_n.Q_s, minLWC_n.Q_e);
      calculate(maxLWC_n.Q_s, maxLWC_n.Q_e);
      calculate(avgLWC_n.Q_s, avgLWC_n.Q_e); //根据量化式计算组合
                                             服务的非功能属性//
      calculate(LWC_n.Q_s, LWC_n.Q_e);
      LWC_n.U_QoS ();   //计算组合服务的非功能属性的可用性//
      LWC_n.U_WS ();    //计算组合服务的可用性//
      }
}
sort(LWC_i, i=1,..., n) //排序，获得最优解//
```

图 5 组合 Web 服务选择算法

假设使用图 1 中的模板，模板的非功能属性值满足用户需求，非功能属性权重 Φ_k 均设为 0.2，上述 8 个 Web 服务是按模板发现的可用 Web 服务，可用 Web 服务的测试响应时间如表 2 所示，那么基于模板组合得到的可用组合 Web 服务有 16 个.

另外，假设随机数 $\omega=0.5$，功能属性权重 α_O、β_O、γ_O、α_I、β_I、γ_I、θ_1分别定制为0.3、0.1、0.1、0.2、0.075、0.075、0.05. 在上述事例中，概念“公司”包含“品牌”，分别计算两级 U^*_{IOPE}，取二值中较小的作为 Web 服务的 U_{IOPE}，再计算 U_{WS} 并排序，结果如表 3 所示(取前 10 个). 从表中可知，最合适的服务为 (WS11∧WS21)→WS31，其结果与使用的模板最相符，也是预期的结果. 如果结果不同或模板不能使用，则匹配求其次，该结果将作为新的模板.

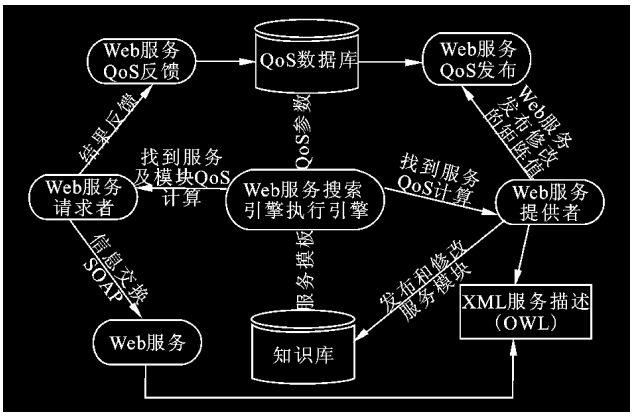

图 6 XJTU-WSDCS 系统结构

表 2 运行得到的响应时间的平均值、最小值和最大值

服务	WS11	WS12	WS13	WS14	WS21	WS22	WS31	WS32
avgQ_e/ms	510	500	510	490	890	860	1 130	990
minQ_e/ms	450	470	480	450	810	800	940	900
maxQ_e/ms	570	590	580	530	980	890	1 310	1 130

4.3 结果分析

与 URCSD 算法[4]相比，本文算法更具有通用性，因为采用的匹配选择规则不仅适用于复杂组合服务，而且适用于简单组合服务，这使得系统能够找到更多的 Web 服务. 与文献[1-3]方法相比，本文算法综合考虑了不同语义匹配度和 QoS 对匹配的影响，并以语义匹配度的量化值代替了相似性匹配规则，所以能全面、准确反映 Web 服务的可用程度. 另外，基于模板发现的可用 Web 服务，经过自动组合得到组合 Web 服务的方法，降低了自动组合的复杂度和不确定性，但算法的计算复杂度将有所提高.

5 结 论

本文提出了一种基于可用性的语义 Web 服务通用发现机制，在没有完全匹配或有效匹配的 Web 服务的情况下，能给用户提供最合适的 Web 服务. 该匹配选择规则不仅适用于复杂组合服务，而且适

表 3　组合 Web 服务及其 QoS 可用性、IOPE 可用性和 WS 可用性

可用复合 Web 服务	avgQ_e/ms	minQ_e/ms	maxQ_e/ms	U_{QoS}	U^*_{IOPE}		U_{IOPE}	U_{WS}
					第一级	第二级		
WS11∧WS21→WS01	2 020	1 750	2 290	0.203	0.500	0.500	0.500	0.352
WS13∧WS21→WS31	2 020	1 750	2 290	0.203	0.467	0.500	0.467	0.335
WS14∧WS21→WS31	2 020	1 750	2 290	0.203	0.443	0.500	0.443	0.323
WS12∧WS21→WS31	2 020	1 750	2 290	0.203	0.343	0.500	0.343	0.273
WS11∧WS22→WS31	1 990	1 740	2 200	0.164	0.375	0.500	0.375	0.270
WS11∧WS21→WS32	1 880	1 710	2 110	0.228	0.500	0.283	0.283	0.256
WS14∧WS21→WS32	1 880	1 710	2 110	0.228	0.443	0.283	0.283	0.256
WS13∧WS21→WS32	1 880	1 710	2 110	0.228	0.467	0.283	0.283	0.256
WS12∧WS21→WS32	1 880	1 710	2 110	0.228	0.343	0.283	0.283	0.256
WS13∧WS22→WS32	1 880	1 710	2 200	0.164	0.346	0.500	0.346	0.255

注：∧为依赖关系；→为次序关系；U^*_{IOPE}为分级下的 IOPE 可用性.

用于简单组合服务. 通过量化的语义匹配度、功能属性与非功能属性，设置相应权重的可用性来寻找最合适的组合 Web 服务是可行的. 今后的工作将以动态的方式为基于环境和条件的工作流构建智能、自适应的抽象模板，进一步研究 Web 服务的调度与执行算法，利用遗传算法进行非线性多目标规划以获取自适应权重，进一步降低计算复杂度等.

参考文献：

[1] MARTIN D, BURSTEIN M, HOBBS J, et al. OWL-S: semantic markup for web services [OB/EL]. [2007-05-09]. http://www.w3.org/Submission/OWL-S/.

[2] LEI Li, HORROCKS I. A software framework for matchmaking based on semantic Web technology [J]. International Journal of Electronic Commerce, 2004, 8(4):39-60.

[3] BINDER C W, FALTINGS B. Efficient matchmaking and directory services [C]// The 2003 IEEE/WIC International Conference on Web Intelligent. Piscataway, NJ, USA: IEEE, 2003:75-81.

[4] HYUN N G, CHUNG M Y, KIM K I, et al. Effective semantic Web services discovery using usability [C]// Proceedings of 8th International Conference on Advanced Communication Technology. Piscataway, NJ, USA: IEEE, 2006:2199-2203.

[5] 张成文，苏森，陈俊亮. 基于遗传算法的 QoS 感知的 Web 服务选择 [J]. 计算机学报，2006，29(7)：1029-1037.
ZHNAG Chengwen, SU Sen, CHEN Junliang. Genetic algorithm on Web services selection supporting QoS [J]. Chinese Journal of Computers, 2006, 29(7):1029-1037.

[6] MAKRIPOULIAS Y, MAKRIS C, PANAGIS Y, et al. Web service discovery based on quality of service [C]// Proceedings of 2006 IEEE International Conference on Computer Systems and Applications. Piscataway, NJ, USA: IEEE, 2006:196-199.

[7] 郭得科，任燕，陈洪辉，等. 一种 QoS 有保障的 Web 服务分布式发现模型 [J]. 软件学报，2006，17(11)：2324-2334.
GUO Deke, REN Yan, CHEN Honghui, et al. A QoS-guaranteed and distributed model for web service discovery [J]. Chinese Journal of Software, 2006, 17(11):2324-2334.

[8] PAOLUCCI M, KAWAMURA T, PAYNE R, et al. Semantic matching of Web services capabilities [C]// Proceedings of 1st International Web Conference on the Semantic Web. Berlin, Germany: Springer-Verlag, 2002:333-347.

（编辑　苗凌）