基于模板和上下文的语义Web服务动态组合

严 娜,黄映辉
(大连海事大学 信息科学技术学院,辽宁 大连 116026)

摘 要:为了适应环境和上下文信息的动态变化并提供适时适地的服务,Web服务组合要能满足处于动态变化环境中的个性化用户需求。文中提出了一个基于模板和上下文的语义Web服务组合框架,该框架使用抽象服务流程进行Web服务组合建模,利用本体来进行上下文信息建模并支持基于JESS的上下文信息推理,在原有基于语义的Web服务匹配的基础上,实时地感知上下文信息来进行Web服务动态绑定。该方法提高了服务组合的成功率和动态适应性,并且满足了用户的个性化需求。

关键词:服务组合;服务匹配;组合模板;上下文;本体
中图分类号:TP393 **文献标识码**:A **文章编号**:1673-629X(2009)12-0089-04

Semantic Web Services Dynamic Composition Based on Templet and Context

YAN Na, HUANG Ying-hui
(College of Information Science and Technology of Dalian Maritime University, Dalian 116026, China)

Abstract: In order to adapt to the dynamic changes of environment and context information and to provide appropriate services, the Web services composition not only has to meet the user's individual requirements, but also adapts to environmental changes. A system framework based on templet and context information is presented, which uses abstract services process model in services composition modeling and uses ontology-based context information modeling which supports JESS-enabled elicitation. At the basis of semantic-based Web service matching, the framework binds the Web service dynamically through real-time context-aware information. This framework improves the success rate and the dynamic adaptability of Web services composition and meets the user's individual requirements.

Key words: services composition; service matching; composition templet; context; ontology

0 引 言

基于语义的Web服务组合依照组合方案的生成方式可将其分为两大类,静态服务组合和动态服务组合。静态服务组合是在业务流程建模时就进行服务绑定,动态服务组合则不在服务定义过程中为活动指定固定的服务提供者,而是将具体的绑定延迟到组合服务执行时动态完成。目前,动态服务组合的方法大体可以分为两类:一是使用人工智能技术来实现Web服务的自动组合,二是使用事先定义好的模板与现有环境中的实际服务相匹配,从而实现Web服务的自动组合。但完全基于模板的服务组合存在自动化程度低、随机应变能力弱、灵活性不够的问题;而完全基于人工智能的自动服务组合存在复杂度高及不容易实现的问题。因此文中提出在全局上采用基于模板的方法进行服务组合的建立,当模板中的抽象服务节点找不到具体服务对应时,采用自动组合方法进行自动合成,提高了服务组合的自动化程度。

动态服务组合已经获得了很多关注,根据不同的方法和技术开发了不同的Web服务动态组合系统。然而现有的动态服务组合系统存在着两个问题:

(1)没有考虑用户的个性化需求。不同的用户即使请求同样功能的服务,用户上下文信息的不同可能会导致选择不同的服务。动态服务组合系统应该根据用户的上下文信息和用户的偏好自动地选择最合适的服务进行组合。

(2)难以适应用户上下文的动态变化。例如,一个用户正在房间A用话筒与朋友通话,若此时他必须去房间B,他就希望能将房间A的话筒换成房间B的话

收稿日期:2009-03-25;**修回日期**:2009-06-11
基金项目:国家自然科学基金资助项目(60672031);辽宁省自然科学基金资助项目(20072142)
作者简介:严 娜(1985-),女,山东淄博人,硕士研究生,CCF学生会员,研究方向为智能信息处理;黄映辉,教授,CCF高级会员,研究方向为智能信息处理。

简继续进行通话。为了适应这种动态的环境,动态的服务组合系统应该能够实时地监控上下文信息的变化自动地组合一个新的服务使用户可以继续获得相同的应用。

为了适应用户的个性化需求和环境的动态变化,文中提出了在原有的基于语义的 Web 服务匹配的基础上根据上下文信息进行 Web 服务选择的方法。由服务组合设计者利用工作流机制来设置服务组合的抽象流程,根据用户请求选择抽象流程,然后对抽象服务组合流程进行具体化,在具体化过程中使用基于上下文感知的服务选择。基于上面的讨论,该文提出了基于模板和上下文的语义 Web 服务组合框架。

1 语义 Web 服务组合

1.1 语义 Web 服务模板

基于模板的服务组合有较高的动态性和灵活性。首先,由服务组合设计者为某些活动设定服务模板即抽象服务流程,但不需要指定具体服务;然后,在执行组合流程时要将抽象的服务组合定义转变成静态的服务组合流程,这就需要进行动态的服务绑定即服务选择。

Web 服务组合模板(Web service composition templet)可以形式化地表示为一个三元组 WST (Profile, WSL, Process)[1]。

Profile 描述模板的基本信息。可以表示为一个三元组(TInPut, TOutPut, TCategory),其中 TInPut 是模板的输入信息,对应一个或多个本体概念;TOutPut 是模板的输出信息,对应一个或多个本体概念;TCategory 表示该组合服务的分类信息。

WSL(Web service list)描述抽象服务的列表。包含一个或者多个抽象服务。每一个抽象服务可以表示为一个三元组,AWS = (AWSInput, AWSOutput, AWSCategory),其中 AWSInput 是抽象服务的输入信息,对应一个或多个本体概念;AWSOutput 是抽象服务的输出信息,对应一个或多个本体概念;AWSCategory 表示抽象服务的分类信息。

Process 描述模板的结构。表示模板中包含的各抽象服务之间的逻辑关系和执行顺序,逻辑关系包括顺序、并行和选择三种。

1.2 语义 Web 服务自动组合算法

利用模板进行 Web 服务组合的过程是将抽象的服务映射成具体的服务的过程,即抽象服务具体化。如果抽象服务组合流程中的抽象服务节点没有具体的服务可以与之映射,那么就会导致服务组合的失败。所以在局部使用自动组合算法,得到对应抽象服务节点的一个服务组合。

WSPR[2]是一个基于 AI planning 的 Web 服务组合算法,利用这个算法进行服务发现时是基于关键字的匹配。文中采用领域本体为 Web 服务进行语义标注,在服务发现时进行基于语义的匹配。

可以把根据用户的需求来自动组合 Web 服务的问题看成是一个状态空间的搜索问题,整个组合过程分两步进行:第一步是一个向前搜索过程,第二步是一个回溯选取过程。

可以把组合过程的状态模型构造为一个五元组 $(W, S, S_0, S_G, \gamma(s))$,其中 W 表示服务集,S 表示状态集,$S_0$ 表示初始状态,S_G 表示要达到的目标状态,$\gamma(s)$表示以 s 为前提条件的服务,其中 $s\in S$,s 表示当前状态集。在向前搜索过程中,S_0 = IN(用户请求的输入参数),S_G 包含 OUT(用户请求的输出参数),而在回溯过程中 S_G = OUT,S_0 包含于 IN。在下面的算法中一致用 IN 和 OUT 来表示。Web 服务可形式化地描述为 W(I, O),其中 W 是服务的名称,I 是服务的输入参数,O 是服务的输出参数。

算法 1:向前搜索算法

```
Compose(W, IN, OUT)
INPUT:IN, OUT
OUTPUT:PE
s=IN;
D=null;
while(OUT⊄s) do
{
  w={m|m∈γ(s), m∉D}
  For each w
  {
    If(similarity(w.I, IN)>r)
    {
      F=F∪w;
      For each o∈w.O
        {PE(o)=w;s=s∪o;}
    }
  }
D=D∪w;
}
```

算法 2:回溯选取算法

```
Backdate(PE, IN, OUT)
INPUT:PE, IN, OUT
OUTPUT:服务组合序列
hsg(w)=|w.O∩goal|;
s=OUT;
Goal=s;
while(goal≠null) do
```

```
{
    ws={PE(o)|o∈goal};
    M=max(hsg(w));
    S=s∪M.out;
    cmp=cmp∪M;
    Goal=(goal∪M.in)-s;
}
S=IN;
while(cmp≠null) do
{
    If∈w∈γ(s)∩w∈cmp)
      {Print w,s=s∪w.O; cmp=cmp-w;}
}
```

2　上下文的感知

2.1　上下文定义

上下文是描述环境状态、资源信息、用户需求等影响服务输入输出的一些属性及属性的取值[3]。在普适计算的环境下,环境信息指服务所在设备的资源情况;用户信息指用户偏好,特指对服务非功能属性的要求;服务本身的信息包括服务需要的资源和能为用户提供的非功能特性。因为在普适计算中服务与用户和设备是相关的,上下文分为设备上下文、用户上下文和服务上下文。

上下文感知计算在普适计算中得到了充分的关注,是指计算系统或网络自动地对上下文及上下文变化进行感知和利用,并据此做出决策从而自动提供相应的响应或服务。上下文感知计算的研究主要包括从环境中提取上下文,对上下文进行建模,进行上下文信息推理以及有效地利用上下文构建支持上下文感知的系统框架等问题。

2.2　上下文信息感知策略

有效地感知环境中的上下文信息是构建上下文感知系统的前提。从环境中感知上下文的过程,就是将环境中与当前应用相关的诸多因素转换成语义明确、格式统一的上下文信息。对上下文的感知是利用上下文传感器来监测和获取原始数据。这些上下文传感器是上下文感知系统与上下文环境的接口,它们可以是物理上的,也可以是逻辑上的。通常对于外界环境状态的感知由物理传感器实现,例如使用 GPS 接收器感知位置信息等。而对设备状态、用户的交互习惯、交互历史等上下文的感知则大多由逻辑传感器完成[4]。

2.3　使用本体表示上下文信息

由于各种上下文的特性不同,所以它们的表示模型也不相同。当前几乎所有的系统都采用自己的方法来建立上下文信息模型,这样不便于系统之间的交互。现在需要的是一种能表示各种上下文信息的通用数据结构。在现实的服务组合中,为了实现不同的目标,服务被分成不同的领域。在不同的领域中,上下文都包含一些共有的概念,因此希望构造一个通用的上下文模型来表示上下文信息以适应动态的环境[5]。

文中使用基于本体的上下文信息模型。上下文模型定义了上下文信息的结构,在支持有效的上下文管理中起到了重要的作用。在 Web 服务领域,上下文一般包括用户端上下文和服务上下文。用户端上下文是面向需求的,与服务的发现和选择密切相关。服务上下文指服务运行环境。文中构建的上下文本体(ContextOntology)包括用户上下文本体(userContextOntology)和服务上下文本体(serviceContextOntology)。用户上下文信息主要包括:用户情况、个人偏好、位置、QoS 和环境等信息。服务上下文本体包括界面风格、服务的个性化业务、QoS 和环境等信息。

不管是用户端上下文还是服务上下文都应该定义服务质量(QoS)和环境描述。QoS 描述是经过一段较长时间统计得到的服务质量有关属性,主要包括:可用性(Availability)、可靠性(Reliability)、安全性(Security)、吞吐量(Throughput)、响应时间(Response Time)、服务费用(Cost)等。环境描述是用来描述计算和通讯环境的,包括网络约束、设备约束等。

2.4　基于 JESS 的上下文推理

根据上下文的知识来源不同,上下文一般分为两类:由传感器系统直接得到的低层上下文,如位置、时间等,是进行上下文推理的素材;利用低层上下文采用某些逻辑规则进行推理得到的启发性知识,属于高层上下文。在建立了上下文本体后,需要根据规则对上下文本体进行推理,文中设计了基于 JESS 的上下文推理系统。

整个基于 JESS 的推理系统分为三个模块,本体编辑模块、上下文感知模块和上下文推理模块。整个推理系统框架如图 1 所示。

本体编辑模块主要是用来获得和组织用户输入的上下文信息,把上下文信息存储在上下文知识库中。

上下文感知模块主要是收集动态上下文信息。例如,利用感知设备来感知用户的周围环境,用户登录时,感知用户使用的设备和网络等环境上下文信息,将其存储到上下文知识库中。在本体编辑模块和上下文感知模块不能获得的用户端的上下文信息,需要通过上下文推理来获得。JESS 是典型的通用本体推理机,它的推理机是开放的,用户只需要提供不同领域的推理规则,JESS 就可以对不同领域进行推理。SWRL(Semantic Web Rule Language)语言根据推理规则能够

实现对本体知识的推理，文中的规则是以 SWRL 语法格式进行描述的。SWRL 规则本身建立在 OWL 本体之上，可以结合已有的 OWL 知识库中的信息来建立 SWRL 规则库[6]。

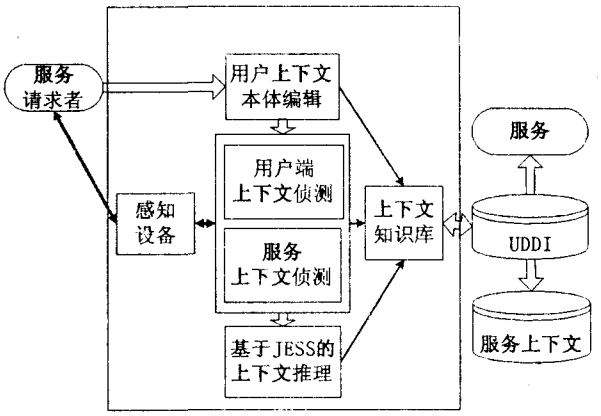

图 1 基于 JESS 的上下文推理系统

整个 SWRL 推理机制包括以下内容：

第一步：把建立的上下文本体和得到的上下文信息存入 OWL 知识库中，结合 OWL 知识库中的本体、实例和属性建立该领域的规则，形成 SWRL 规则库。

第二步：把 OWL 知识库的格式转换为 JESS 推理引擎所要求的格式即事实库；把 SWRL 规则库的格式转换为 JESS 推理引擎所要求的格式。

第三步：利用事实库和规则库在 JESS 推理引擎中进行推理，产生出新的事实。

第四步：将新推理出的事实通过格式转换，存储到原来的 OWL 知识库中，更新或扩充知识库。

例如，制定了一条规则：如果用户的角色(Role)是学生，那么就选择服务费用(Cost)便宜(cheap)的服务，则其 SWRL 规则为：

userContextOntology(? x) ∧ hasPersonal(? x,? y) ∧ hasRole(? y,? z) ∧ hasValue(? z,student) ∧ hasQoS (? x,? a) ∧ hasCost(? a,? b)→hasValue(? b,cheap)

3 基于模板和上下文的语义 Web 服务组合

上下文对组合服务流程的成功执行至关重要，用户需要更好的机制和组合服务运行平台来保证服务质量，服务提供者的失效或网络拥塞等因素会导致某个服务无法访问，从而导致组合服务质量下降或是不可使用，所以文中提出了基于模板和上下文的语义 Web 服务组合框架，如图 2 所示。根据上下文信息，Web 服务组合能够根据用户偏好进行个性化的调整，还能在服务的组合过程中过滤掉不合适的服务。

整个语义 Web 服务组合过程是：首先选择一个抽象服务组合流程，然后把抽象服务映射成具体服务，如果没有与抽象服务相匹配的具体服务，就调用自动组合算法组合一组具体服务来对应抽象服务节点，最后得到一条可执行的服务组合路径。

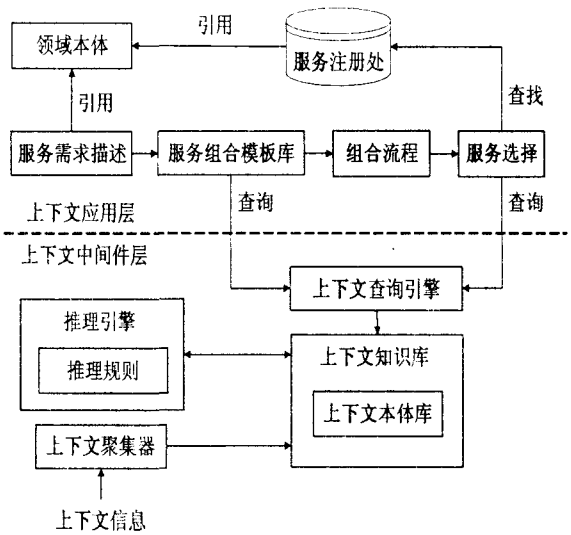

图 2 基于模板和上下文的服务组合框架

上下文信息在 Web 服务组合过程中的用途主要有两个：一是路径选择，当从服务组合模板库中选择出抽象服务组合流程后，可以把上下文信息作为流程中分支节点的路径选择条件；二是服务选择，可以使用上下文作为服务选择的条件。在文中的研究中，只是根据上下文信息进行服务选择。

在服务选择过程中使用了两层匹配：IO 匹配和上下文匹配。在 IO 匹配中，采用的是基于领域本体的语义相似度的计算，利用语义距离计算概念相似度[7]。如果经过 IO 匹配得到的服务数量大于等于两个，就需要通过上下文匹配来决定选择哪个服务。在传统的上下文模型中，上下文信息只有数值描述，上下文信息的匹配即为数值之间的代数比较。在语义 Web 服务中，上下文信息是通过上下文本体来描述的，除了数值信息之外，还具有语义信息。在进行匹配时，首先对上下文参数基于上下文本体进行语义匹配，然后对上下文参数的取值进行数值匹配。

基于上下文的语义 Web 服务组合框架适应了环境的动态变化，该框架在执行请求应用的过程中要时刻监控上下文信息，随时监测上下文的变化，通过语义和上下文的匹配进行服务选择，最后得到一条服务组合路径。

4 结束语

文中提出了一种基于模板和上下文的语义 Web 服务动态组合框架。在基于语义的服务匹配的基础

(下转第 96 页)

2.3 结合读写的预取操作

Linux 预取算法只实现了对读操作的预取。而 Susanne Albers[10]等研究表明，结合读和写操作的预取也能有效地提高系统的性能。

2.4 预取调度器

目前针对预取算法的研究主要集中在单系统环境中，Zhe Zhang[11]等研究发现：在一个多级的存储系统中，可在服务端侧增加一个预取调度器，来根据客户端请求及服务端的资源状况来动态调整预取的大小，从而提高预取的精确度，进一步来提高服务器的服务性能。

3 结束语

文中主要对 Linux2.6.29 - rc2 中相关的预取算法源代码进行详细的分析，分析了预取的监控机制及其对访问模式的判断处理方法，最后提出了对预取算法的改进方法与措施，对于提高 Linux 系统的整体性能具有重要的意义。

参考文献：

[1] 郇丹丹. 高性能存储系统研究[D]. 北京：中国科学院计算技术研究所，2006.

[2] Cao Pei. Implementation and Performance of Integrated Application - Controlled File Caching Prefetching and Disk Scheduling[J]. ACM Transcation on Computer Systems, 1996, 14(4):311 - 343.

[3] Patterson H R, Gibson G A, Ginting E, et al. Informed Prefetching and Caching[C]//15th ACM Symp on Operating System Principles. Copper Mountain, Colorado, United States:[s.n.],1995:79 - 95.

[4] Chang Fay. Using speculative execution to automatically hide I/O latency[D]. [s.l.]:Carmegie Mellon University,2001.

[5] 吴峰光. Linux 内核中的预取算法[D]. 合肥：中国科学技术大学，2008.

[6] Love R. Linux 内核设计与实现[M]. 第 2 版. 陈莉君，等译. 北京：机械工业出版社，2006.

[7] Daniel. 深入理解 Linux 内核[M]. 第 3 版. 陈莉君，等译. 北京：机械工业出版社，2006.

[8] 孙钟秀. 操作系统教程[M]. 北京：高等教育出版社，2003.

[9] Ding Xiaoning, Jiang Song, Chen Feng, et al. DiskSeen: Exploiting Disk Layout and Access History to Enhance I/O Prefetch[C]//2007 USENIX Annual Technical Conference. [s.l.]:USENIX Association,2007.

[10] Albers S, Buttner M. Integrated Prefetching and Caching with Read and Write Requests[C]//Proceedings of the 8th National workshop on algorithms and data structures, 2003, Band 2748 von Lecture Notes in Computer Science. [s.l.]: Springer,2003:162 - 173.

[11] Zhang Zhe, Lee Kyuhyung, Ma Xiaosong. PFC: Transparent Optimization of Existing Prefetching Strategies for Multi - level Storage system[C]//Proceedings of the 28th International Conference on Distributed Computing Systems, DOI 10.1109/ICDCS. Beijing, China:[s.n.],2008.

(上接第 92 页)

上，根据用户和服务的上下文信息进行筛选，将满足上下文信息的 Web 服务优先返回给用户，其结果是对语义 Web 服务查询结果的精化和优化[8]。文中利用本体来描述上下文的信息模型，定义了用户上下文本体和 Web 服务上下文本体两个本体库，并制定了二者之间的推理规则，根据规则选择出优先返回给用户的 Web 服务，可提高用户的满意度。

参考文献：

[1] 宋 驰. 基于用户偏好的启发式 Web 服务组合的研究与实现[D]. 北京：北京邮电大学，2008.

[2] Zheng Xianrong, Yan Yuhong. An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model[C]// Proc. of ICWS. Beijing, China:[s.n.],2008:691 - 699.

[3] 唐 磊，淮晓永，李明树. 一种基于上下文协商的动态服务组合方法[J]. 计算机研究与发展，2008,45(11):1902 - 1910.

[4] 岳玮宁，王 悦，汪国平. 基于上下文感知的智能交互系统模型[J]. 计算机辅助设计与图形学学报，2005,17(1):74 - 79.

[5] Qiu Lirong, Cao Yongcun, Zhao Xiaobing. Promoting adaptation of semantic Web service composition using context information[C]// International Symposium on Computer Science and Computational Technology. Shanghai, China:[s.n.], 2008:652 - 656.

[6] 郭文英. 基于 SWRL 推理的语义关联发现及其在本体映射与集成中的应用[D]. 杭州：浙江大学，2006.

[7] 刘克非，王 红，王卫玲. 基于语义相似度的 Web 服务发现研究[J]. 计算机技术与发展，2007,17(2):16 - 18.

[8] Maamar Z, Benslimane D, Thiran P. Towards a context - based multi - type policy approach for Web services composition[J]. Data & Knowledge Engineering, 2007,62(9):327 - 351.