面向服务软件的网络拓扑表示方法

孟 晖[1] 周剑明[2] 肖俐平[1] 李德毅[3]
(解放军理工大学指挥自动化学院 南京 210007)[1] (解放军通信指挥学院 武汉 430010)[2]
(中国电子工程系统研究所 北京 100039)[3]

摘 要 主要研究了面向服务软件的网络拓扑知识表示问题。首先通过分析OWL-s中服务轮廓和服务模型的定义,研究了Web服务的属性及Web服务之间的主要逻辑关系;而后,提出了面向服务软件的拓扑知识表示方法,运用属性关系图(ARG)来描述面向服务软件的结构;最后,用一个实例说明了该方法的可行性。通过该实例可以看出,该拓扑表示方法可以准确描述服务间存在的多种关系类型及服务本身的各种属性,为进一步研究面向服务软件的结构特性打下基础。

关键词 拓扑映射,Web服务,软件网络

Topological Representation Approach of Service Oriented Software

MENG Hui[1] ZHOU Jian-ming[2] XIAO Li-ping[1] LI De-yi[3]
(Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007, China)[1]
(Commanding Communications Academy, Wuhan 430010, China)[2]
(China Institute of Electronic System Engineering, Beijing 100039, China)[3]

Abstract This paper mainly discussed the issue of topological representation approach of service-oriented software. The attributes and logical relations between services were firstly studied by analyzing the definition of service profile and service model in OWL-s. Then a topological representation approach, which uses attributed relational graph to describe the structure of service-oriented software, was put forward. In the end, we used this approach to represent a simple traveling system to demonstrate its feasibility. We can see that this approach could correctly describe the various relation types between services and attributes of services. This approach will help us to study the structure of service-oriented software.

Keywords Topological mapping, Web service, Software network

1 引言

Internet的出现和面向服务计算SOC(Service-oriented Computing)导致软件应用系统的主要形态、生产方式、运行方式和使用方式发生了巨大的改变。软件的开发和应用越来越依赖日益丰富的网络资源,软件也逐渐从最初的单体软件转变为群体软件的交互与协同。在网络化和服务化的背景下,软件生产的主要目标是实现满足个性化与多元化的大众需求的规模化定制。为了应对这一历史性变革,研究人员围绕服务这一核心,提出了面向服务的软件工程,将服务视为组成软件的基本单元,软件由服务组合而成,甚至有学者提出了"软件就是服务(Software as a Service)"的思想。

目前,用复杂网络理论研究面向对象软件结构的工作已经取得了一些成果,发现面向对象软件的内部结构呈现出小世界和无标度特性。这些研究成果对软件的开发具有一定的指导意义。在当今的环境下,面向服务软件已经日益成为主流。因此,有必要研究面向服务软件的结构,挖掘其结构特性和结构的演化规律,从而更好地指导面向服务软件的开发。为实现这一目标,首先要研究面向服务软件的拓扑表示方法,将面向服务软件抽象成为网络拓扑;而后,运用网络化数据挖掘方法挖掘其结构特性,并通过观察软件在其演化过程中拓扑结构的变化来挖掘其演化规律。

本文主要研究面向服务的软件的拓扑知识表示问题。第2节简要介绍复杂网络理论与软件工程相结合的研究所取得的一些成果,分析其不足之处,并指出本文的研究意义;第3节研究服务的属性和服务之间的关系类型,并提出了一种将面向服务软件映射为网络的拓扑映射方法;第4节用该方法对一个面向服务软件系统进行拓扑映射,说明了该方法的可行性,最后对全文进行总结,并对今后的工作进行展望。

2 相关工作及研究意义

复杂系统和复杂性科学已经被认为是21世纪科学研究

到稿日期:2008-04-10 本文受国家973重点基础研究发展规划项目(2007CB310804),国家自然科学基金面上项目(60675032)资助。
孟 晖 博士研究生,主要研究方向为复杂网络、数据挖掘,E-mail:xiaoxiao51341@sina.com;周剑明 博士研究生,副教授,主要研究方向为Web服务、作战仿真;肖俐平 博士研究生,主要研究方向为复杂网络、数据挖掘;李德毅 博士生导师,中国工程院院士,主要研究方向为人工智能、数据挖掘、指挥自动化、智能控制。

的前沿。随着小世界特性和无标度特性的发现，复杂网络的研究取得了巨大进展，为系统地、全局地研究系统的结构动力学特性提供了有力的工具。近年来，软件工程开始与复杂网络相结合，关于软件系统拓扑结构的分析已经取得了一些初步成果。

2002 年，Valverde[1] 等人首次运用复杂网络理论来研究软件系统。他们试图研究软件工程中的类图，使用无向图来表示软件系统，将类映射为网络节点，将类之间的继承关系和关联关系映射为边。通过两个软件系统——JDK 1.2 和 Ubi Soft Prorally 2002 的研究，他们发现了由类图抽象得到的软件网络呈现出小世界特性和无标度特性。次年，Myers 等人[2] 指出类之间的协作(collaboration)和调用(calling)反映了系统中的控制流。因此，在对类图进行抽象时，边的方向性是不能忽略的。基于该思想，他们用有向网络来描述软件系统，研究了 6 个开源软件系统。除了发现软件有向网络同样具有小世界和无标度特性之外，他们还发现软件网络的入度与出度的幂律分布指数有明显的区别，出度的指数大于入度，并指出这种现象可能是由于软件开发中鼓励重用所导致的，如图 1 所示。

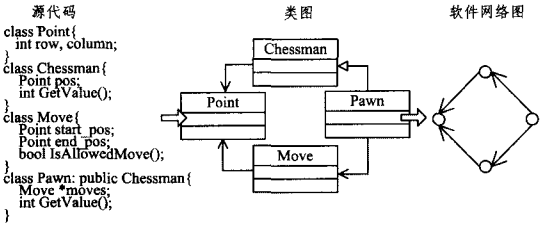

图 1　软件类图的拓扑映射

除了对软件类图的研究，还有研究人员在不同粒度上对软件结构进行抽象。LaBelle 等人[3] 在包级(将包视为网络中的结点)对一些软件的依赖网络(inter-package dependency network)进行分析，Wood 等人[4] 对包、类、方法层次的协作网络进行分析，同样发现这些网络具有小世界和无标度特性。此外，Potanin 等人[5] 还发现不仅由源代码得到的静态软件结构具有无尺度特性，在运行时获取的软件对象网络同样也具有无尺度特性。

在上述对软件网络的研究中，都是将软件网络映射为图结构 $G(V,E)$，V 是节点集合，E 是边的集合。这种映射方法所得到的网络拓扑无法体现软件模块(类，函数等)间关系的多样性，从而造成信息的丢失。因此，Li 等人[7] 采用属性关系图 ARG(Attributed Relational Graph)[8] 来表示面向对象软件的类图，并在此基础上研究了基于图匹配的设计模式挖掘算法。

随着近年来 Web 服务的快速发展，面向服务的思想已经被许多软件工程师所接受。因此有必要分析面向服务软件的结构，以指导面向服务软件的开发。借鉴以往软件网络研究的经验，我们同样可以运用复杂网络理论来研究面向服务软件，将其结构映射为网络拓扑，而后运用网络化数据挖掘方法，发现网络中的关键节点和关键边，这些节点和边分别对应着软件中的重要 Web 服务和服务间的重要关系。此外，还可以通过研究软件演化前后拓扑结构的变化，发现面向服务软件结构的演化特性。因此，本文试图通过分析面向服务软件中服务间的关系，实现软件到网络拓扑的映射，为下一步用复杂网络理论分析软件结构打下基础。

3　面向服务软件的网络拓扑知识表示

通过自然语言或符号语言(如 OWL-s)这种一维的知识表示方式来描述服务，无法从全局的角度呈现面向服务软件的复杂结构。而网络拓扑作为一种二维知识表示方法，能够弥补这一不足。因此，我们需要分析面向服务软件中服务的属性和服务间关系，用网络拓扑这种知识表示方法来表示面向服务软件，为今后运用复杂网络理论研究面向服务软件的结构打下基础。

3.1　Web 服务的属性

OWL-s[6] 采用服务轮廓(Service Profile)来描述服务。在服务轮廓中，定义了若干 Web 服务的属性，如图 2 所示。这些属性大致可以分为以下几类。

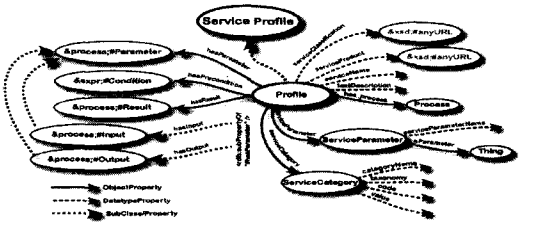

图 2　Service Profile 中的类和属性[6]

描述服务基本信息的属性

描述服务基本信息的属性包括服务名称(serviceName)、文字描述(textDescription) 和联系信息(contactInformation)。这些属性的取值通常都是由文本表示的，目的是为了给使用该服务的人提供必要的信息。

描述服务功能的属性

OWL-s 用以下 4 个属性来描述服务的功能，即输入(Input)、输出(Output)、前件(Precondition)和结果(Effect)。

描述服务分类的属性

这些属性描述了服务的分类情况，包括服务的类别名称(categoryName)，所采用的分类法(taxonomy)，类别名称在分类法中对应的值(value)，以及每种服务类型在分类法中对应的编码(code)。

3.2　Web 服务间的逻辑关系

服务之间的关系可大致分为逻辑关系和非逻辑关系。面向服务软件的功能通常是由原子服务(atomic service)通过一定的编排方式组合而成的，这种编排所体现出的服务间的关系就是逻辑关系。除逻辑关系外，服务之间存在非逻辑关系，比如服务间相通提供者的关系、可替换关系等等。由于本文的主要目的是研究面向服务软件的结构，因此在对软件进行拓扑映射时，仅考虑服务之间的逻辑关系。

OWL-s 的服务模型中，定义了 9 种过程的控制结构(control construct)，即 Sequence，Split，Split＋Join，Any-Order，Choice，If-Then-Else，Iterate，Repeat-While，Repeat-Until。由于原子过程与原子服务之间存在对应关系，因此原子过程之间的逻辑关系与原子服务之间的逻辑关系是等价的。下面，本文将对这 9 种控制结构进行分析，找出服务之间的逻辑关系。

Sequence

Sequence 结构是指该结构内的过程的执行顺序必须满足

某一特定的序列，如一个服务的执行需要另一个服务的输出，如图 3(a)所示。该结构表明过程之间存在序列关系。

Split 和 Split+Join

Split 结构是指该结构所包含的过程可以并行执行，从而提高执行效率；Split+Join 与 Split 略有不同，该结构所包含的过程可以并行执行，但必须要有障碍同步点(barrier synchronization)。在 Split 结构和 Split+Join 结构中，过程之间存在并行关系。

Any-Order

Any-Order 结构是指该结构所包含的过程不需要按照特定的执行顺序加以执行，但不能并行执行。显然，该结构中的过程之间存在互斥关系。

Choice 和 If-Then-Else

Choice 结构与程序设计语言中的 switch 语句功能类似，即从该结构所包含的过程中选取出一个进行执行；而 If-Then-Else 结构则与 if 语句类似，即“Test If-condition; if True do Then, if False do Else”。因此，Choice 结构和 If-Then-Else 结构所包含的过程之间存在选择关系，即在这些过程之中只有一个过程被选择执行。

Iterate(Repeat-While 和 Repeat-Until)

Iterate 结构包括 Repeat-While 结构和 Repeat-Until 结构。Repeat-While 结构类似于程序设计语言中的 While 语句，即判断条件是否成立，如果条件成立则执行循环体，否则结束循环；而 Repeat-Until 结构则类似于 Do-while 语句，先执行循环体，然后判断条件是否成立，如果成立则退出循环，否则继续执行循环体。在 Repeat-While 结构中，循环体中的过程有可能不被执行，而在 Repeat-Until 过程中，循环体中的过程至少被执行一次。在 Iterate 结构中，过程之间仅仅存在序列关系。

此外，由于组合服务都是由若干子服务通过一定的控制结构编排而成，因此组合服务与其调用的各服务之间，还存在调用关系。

综上所述，服务之间的逻辑关系可以被归纳为 5 种，即序列关系、并行关系、互斥关系、选择关系和调用关系。

3.3 面向服务软件的拓扑表示方法

根据 3.1，3.2 节的分析，服务本身具有若干属性，服务与服务之间也存在多种关系类型。而如果用传统的网络拓扑 $G(V,E)$ 来表示面向服务软件的结构，则无法表示服务间的多种关系类型，造成信息丢失。为了避免出现这种情况，我们在对面向服务软件进行拓扑映射时可以采用文献[7]中的方法，将面向服务的软件映射为属性关系图 ARG。下面我们首先对属性关系图进行简要介绍。

ARG 由一个 6 元组 $G_s=(N_s,R_s,A_s,E_s,\mu_s,\upsilon_s)$ 组成。其中 $N_s:\{n_1,n_2,\cdots,n_n\}$ 是带属性节点集合，$R_s:\{r_1,r_2,\cdots,r_n\}$ 是带属性边的集合，A_s 是节点属性(如节点编号、节点类型)和节点属性取值的子母表，E_s 是边属性和边属性取值的字母表，$\mu_s:N_s\rightarrow(A_s\times A_s)_p$ 是节点属性查询函数，用于返回“节点属性、节点属性值”对，p 是节点的属性个数。$\upsilon_s:R_s\rightarrow(E_s\times E_s)_q$ 是边属性查询函数，用于返回“边属性、边属性值”对，q 是边的属性个数。

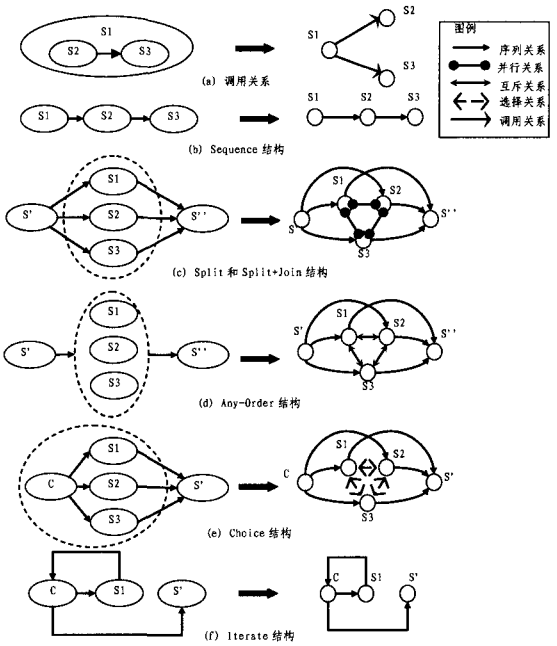

图 3 Web 服务控制结构的映射方法

根据上文的分析，将面向服务的软件映射为属性关系图时，可将 Web 服务映射为节点，将服务间的关系映射为边。对于节点而言，其属性应该包括 3.1 节中列出的所有 Web 服务属性；而对于边而言，仅存在边类型这一属性，其取值为3.2 节中列出的 5 种关系类型中的一种。对于服务间的调用关系，可用图 3(a)所示的方式来映射。而对于软件中出现的 OWL-s 中定义的控制结构，分别按照如下方式对节点间的关系进行拓扑映射：

Sequence

对于 Sequence 结构，只需将 Web 服务映射为节点，将服务之间的序列关系映射为有向边，同时指定边所代表的关系类型，图 3(b)所示。

Split 和 Split+Join

对于 Split 和 Split+Join 结构，假设控制结构之前的服务为 S′，之后的服务为 S″，则 S′与控制结构内的服务之间均存在序列关系，控制结构内的服务与 S″间也均存在序列关系，控制结构内服务之间存在并行关系。在进行映射时，将服务映射为节点，将序列关系映射为有向边，而并行关系映射为无向边，同时指定边所代表的关系类型。具体映射方式如图 3(c)所示。

Any-Order

Any-Order 结构的映射方式与 Split 和 Split+Join 结构类似，只是结构内部服务之间的关系类型为互斥关系，同时指定边所代表的关系类型。具体映射方式如图 3(d)所示。

Choice 和 If-Then-Else

假设结构中条件为 C，控制结构之后的服务为 S′，则条件 C 与控制结构中服务之间存在序列关系，控制结构中的服务与 S′之间也存在序列关系，控制结构内部之间存在选择关系。在进行映射时，将服务和条件均映射为节点，将序列关系映射为有向边，而选择关系映射为无向边，同时指定边所代表的关系类型。具体映射方式如图 3(e)所示。

Iterate(Repeat-While 和 Repeat-Until)

假设 Iterate 结构中的条件为 C，结构的后续服务为 S′。在进行映射时，将服务和条件均映射为节点，将序列关系映射为有向边，同时指定边所代表的关系类型，图 3(f)给出了 Repeat-While 结构的映射方式。

4 实例

本节以一个用户出行系统为例，进一步说明面向服务软件的拓扑映射方法。由于篇幅限制，本节仅对该系统提供的 3 个功能进行分析：即私家车出行导航、用户资料修改和酒店房间预订。

该用户出行系统的结构图如图 4 所示。在该系统中，这 3 个功能是分别由 3 个组合服务提供的。用户在登录系统后，只能选择一个功能。因此，这 3 个组合服务之间存在选择关系。

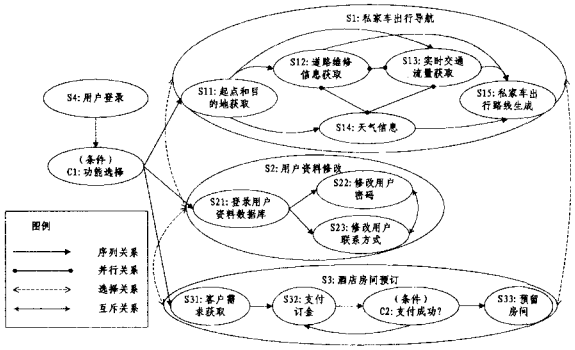

图 4 某用户出行系统的结构图

对于私家车出行导航服务而言，首先需要获取用户的起点和目的地。考虑到用户输入方式的多样性，比如文本输入、图形输入等多种方式，因此需要调用起点和目的地获取服务；而后，需要查询起点和目的地之间可能经过区域的天气信息、道路维修信息及实时交通流量等信息。获取这 3 类信息的服务是可以并行执行的，因此它们之间存在并行关系；最后，根据获取得到的信息，调用私家车路线生成服务生成最佳的行车路线，返回给用户。

对于用户资料修改服务，首先需要登录用户资料数据库。而后，用户也许需要修改其用户密码和联系方式。对于这二者而言，其执行顺序的不同不会导致执行结果的变化，但由于这二者都是对用户数据库的写操作，不能并行执行。因此，修改用户密码的服务和修改用户联系方式的服务之间存在互斥关系。

对于酒店预订服务，首先需要调用客户需求获取服务来获取客户的预订需求；而后调用订金支付服务；接下来执行判断，看订金是否支付成功，如果成功则调用预留房间服务为客户预订房间，否则继续调用订金支付服务，直到订金交付成功为止。

根据以上的分析，运用本文中的拓扑映射方法，可以将该软件系统映射为如图 5 所示的 ARG 图。该图中一共有 17 个节点和 35 条边，其中表示调用关系边有 16 条，序列关系边有 12 条，并行关系边和选择关系边各 3 条，互斥关系边 1 条。通过这个例子可以看出，本文提出的映射方法得到的 ARG 图能够有效展示服务间的多种逻辑关系，较为完整地保留了原软件系统的结构。

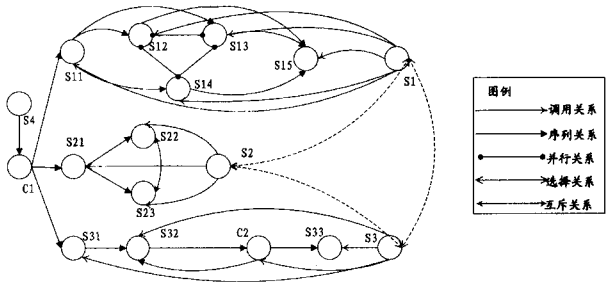

图 5 出行系统的拓扑图

结束语 本文主要研究了面向服务软件的网络拓扑知识表示问题。首先通过分析 OWL-s 中定义的服务轮廓(Service Profile)和 9 种控制构造子，研究了服务的属性和服务之间的主要逻辑关系；随后，提出了面向服务软件的拓扑知识表示方法，运用属性关系图(ARG)来描述面向服务软件的结构；在文章的最后，用该方法表示一个用户出行系统的结构。该方法可以有效表示服务间存在的多种关系类型和面向服务软件系统的结构。

在下一步工作中，可以运用复杂网络理论分析抽象得到的网络拓扑，获取该网络拓扑的统计特征，发现网络中的频繁子图。这些频繁子图可能对应着 SOA 软件中的设计模式，这些设计模式对于 SOA 软件的开发有着重要指导意义。其次，可运用网络化数据挖掘方法，挖掘网络中的关键节点。这些关键节点往往对应着组成软件系统的关键服务。通过改善这些服务的服务质量，可以有效提高整个软件系统的服务质量。再次，通过分析软件不同版本对应的网络拓扑间的差异，可以发现该软件系统的演化特性，从而更好地指导软件系统的维护。

参 考 文 献

[1] Valverds S, Cancho R F, Sole R V. Scale-free networks from optimal design [J]. Europhysics Letters, 2002, 60: 512-517

[2] Myers C R. Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs[J]. Physical Review E, 2003, 68: 1-15

[3] Labelle N, Wallingford E. Inter-package dependency networks in open-source software[J/OL]. eprint arXiv: cs/0411096, 2004, 11

[4] Hyland-Wood D, Carrington D, Kaplan S. Scale-Free nature of Java software package, class and method collaboration graphs. Technical Report of MIND Laboratory (No. TR-MS1286) [R]. Park: University of Maryland College Park, 2006

[5] Potanin A, Noble J, Frean M, et al. Scale-free geometry in OO programs[J]. Communications of the ACM, 2005, 48(5): 99-103

[6] Martin D, Burstein M, Hobbs J, et al. OWL-S: Semantic Markup for Web Services [EB/OL]. Cambridge: W3C. 2004(2004. 11. 22)[2008. 3. 28]. http://www.w3.org/Submission/OWL-S/

[7] LI Qing-hua, ZHANG Zhi-xiang, BEN Ke-rong. Design Pattern Mining Using Graph Matching[J]. Wuhan University Journal of Natural Sciences, 2004, 9(4): 444-448

[8] Eshera M A, FU K S. A graph distance measure for image analysis[J]. IEEE Trans. Systems, Man and Cybernetics, 1984, 14(3): 398-408