

基于体系结构的网构软件演化关键技术研究

陈志雨,刘俊强,叶子文,胡 明

(长春工业大学 计算机科学与信息工程学院,吉林 长春 130012)

摘 要:随着 Internet 的发展,网构软件的研究越来越受到人们的重视。研究者在网构软件的自适应演化方面虽然做了许多研究,取得了很多成果,但也有不足之处。针对网构软件的演化,文中提出了一种以主动连接器、规则库、构件选择推理机、构件描述库为核心的体系结构,在构件实体方面采用 Web 服务技术。在软件结构的基础上研究了规则库的建立过程、研究了推理机的推理算法和构件选择策略,这使得软件运行时能根据上下文环境动态选择调用 Web 服务构件。

关键词:网构软件;演化;体系结构;规则库;推理机

中图分类号:TP31

文献标识码:A

文章编号:1673-629X(2014)05-0036-04

doi:10.3969/j.issn.1673-629X.2014.05.009

Research on Key Technology of Internet-ware Evolution Based on Architecture

CHEN Zhi-yu, LIU Jun-qiang, YE Zi-wen, HU Ming

(Dept. of Computer Science and Information Engineering, Changchun University of
Technology, Changchun 130012, China)

Abstract: Along with the development of the Internet, more and more people start to emphasize the research of the internet-ware. The researchers have done a lot of work on the aspect of the evolution of the internet-ware, though they also have obtained some achievements, there are also have some deficiency. For the evolution of the software, put forward a architecture with the active connector, the rule-base and the component description library as the core, using the Web service technology in entity component. Based on software architecture the rule-base processing of establishment is researched, and a inference algorithm is also put forward. All of these research make the software can choose the component dynamically when it is on running.

Key words: internet-ware; evolution; architecture; rule-base; inference

0 引 言

随着 Internet 的快速发展与普及,如何在开放、动态、难控的网络环境下实现各类资源的共享和集成已经成为计算机软件技术面临的重要挑战之一。以构件等技术支持的软件实体以开放、自主的方式存在于 Internet 的各个节点上,任何一个软件实体可在开放的环境下通过某种方式加以发布,并以各种协同方式与其他软件实体进行跨网络的互连、互通协作和联盟,从而形成一种与当前 WWW 类似的 Software Web, Web 不再仅仅是信息的提供者,而是各种服务的提供者,这种新的软件形态被称为网构软件^[1]。

网构软件的一个重要特征是具有演化性,它的演

化性是指软件可以根据程序运行环境上下文的变化或人们需求的变化进行自我调整的能力。

国内外的研究人员对软件演化的理论和技术作了广泛的研究,代表性的成就主要有基于反射式中间件的软件演化研究^[2]、基于构件运算的软件演化研究^[3-5]和基于软件体系结构的演化研究^[6-8]。

反射式中间件^[9]是通过引入反射原理,中间件运行时的内部状态和行为可通过受限的方式进行操作和访问。反射系统包括一个基层和一个或多个元层,在基层中的实体执行正常的业务功能,元层中的实体则负责维护系统自我描述。北京大学的杨芙清、黄罡研究了基于反射式中间件的 PKUAS 系统,实现了 EJB

收稿日期:2013-07-05

修回日期:2013-10-13

网络出版时间:2014-02-11

基金项目:吉林省自然科学基金项目(20101523)

作者简介:陈志雨(1974-),男,吉林长春人,副教授,博士,研究方向为数据库与软件工程;刘俊强(1986-),男,硕士研究生,研究方向为软件工程。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20140211.1450.012.html>

构件的演化,但没有解决分布式环境下的演化问题。在构件运算方面有很多成就,张友生等人基于代数理论,研究了构件的调用运算、协作运算和条件运算三种关系,利用这三种关系对体系结构进行抽象描述,并且证明了构件运算关系对体系结构描述是完整的。王小平等提出一种基于构件的软件演化模型^[10-11],以构件为基本单位,基于软件体系结构部署和实施演化并且给出了构件添加、删除和替换算法。吕建等人提出面向动态软件体系结构的在线演化方法^[12],设计并实现了一种运行时刻的软件体系结构模型,基于该方法开发了可视化支撑平台 Artemis-ARC。体系结构将开发人员的关注点从代码转移到了粗粒度的构件和构件之间互连的结构上,使软件开发人员从更高的角度来关注系统结构:构件之间的互连、构件的调度。软件体系结构的显著特征就是使用连接件(也称连接器)连接构件^[13-14],这样就将构件的计算功能与构件的通信功能分离,最大程度上减少了构件之间的相互依赖关系,方便对系统的理解、分析和演化,构件之间的交互依赖于环境上下文。

当前的研究都是基于某种特定的构件模型(例如三种经典的构件模型 EJB、COM 和 CORBA),由于构件模型的实现方式不同,所以不同模型的构件间在通信交互上会存在问题。

文中针对网构软件的演化,提出了一种由主动连接器、规则库、构件选择推理机和构件描述库等组成的软件体系结构,在该结构中,采用 Web 服务来表示提供服务的实体,Web 服务是基于标准的,所以它们之间不会存在由于模型不同而不能交互的问题。

1 Web 服务

1.1 Web 服务的定义

简单来说,Web 服务就是一个应用程序,它向外界暴露出一个能够通过 Web 进行调用的 API。Web 服务是一种新的 Web 应用程序分支,它们是自包含、自描述、模块化的应用,可以在网络中被描述、发布、查找以及通过 Web 来调用。

1.2 Web 服务的特点

1) 高度可集成能力。由于 Web 服务采取简单的、易理解的标准 Web 协议作为组件界面描述和协同描述规范,完全屏蔽了不同软件平台的差异,无论是 CORBA、DCOM 还是 EJB(它们是三种典型的构件模型)都可以通过这一种标准的协议进行互操作,实现了在当前环境下最高的可集成性。

2) 使用标准协议和协约规范。作为 Web 服务,其所有公共的协约完全需要开发的标准协议进行描述、传输和交换。

2 演化体系结构

演化结构包括主动连接器、规则库、构件选择推理机、构件描述库和数据总线五个部分,演化结构如图 1 所示。

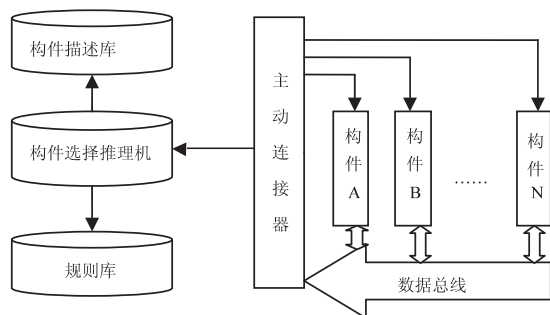


图 1 演化结构

2.1 演化结构部分介绍

构件描述库的作用是描述 Web 服务的信息,这些信息主要包括 Web 服务的 ID、Web 服务的功能、Web 服务的地址、Web 服务的方法等功能信息;其次还包括 Web 服务价格、服务质量等非功能信息,当有多个 Web 服务满足功能要求时可以根据这两个指标从中选择较好的 Web 服务。

规则库的作用是存放软件运行时的规则文件,规则文件采用产生式的表现形式($A \rightarrow B$)。在文中定义如下:产生式的前件表示当前 Web 服务构件的功能,产生式的后件表示下一步要调用的 Web 服务构件的功能。文中也采用数据库表来存放规则。在规则数据库表中,还有区分规则是否是软件运行结束的标识(用 true 或 false 表示,true 表示该规则执行完后软件运行结束,false 表示该规则执行完后软件运行还没结束,还要继续运行)。

构件选择推理机在软件运行时根据规则库中的规则、上下文(主要指 Web 服务返回的结果)推理出下一步要调用的构件应具备的功能(可能满足功能要求的服务不止一个,这些服务将形成一个集合),再结合 Web 服务信息描述表里的信息如价格、服务质量根据一定选择策略从候选的 Web 服务集合里选择一个较好的 Web 服务构件。构件选择推理机选择出服务构件后就可以通知主动连接器进行构件调用。

数据总线传递 Web 服务构件间的上下文(主要指 Web 服务构件计算返回的中间结果)。

2.2 规则库的建立

该体系结构中规则库的建立是借鉴了程序结构化设计思想。在结构化程序设计思想中,有三种最基本的程序控制结构:顺序、分支和循环,任何程序都可以由这三种结构构造。

● 顺序结构。

顺序结构表示程序中的各操作是按照它们出现的

先后顺序执行,其表现形式为: $P_1 \longrightarrow P_2$,其中 P_1, P_2 是两个处理操作过程。

● 选择结构。

选择结构表示程序的处理过程中出现了分支,它需根据某一条件选定其中一个分支执行,选择结构有单选择、双选择和多选择三种形式,分别如下:

单选择: $\text{if}(\text{condition}) \{P\}$ 。其中 condition 是条件, P 是当满足条件要执行的操作。

双选择: $\text{if}(\text{condition}) \{P_1\} \text{ else } \{P_2\}$ 。其中 condition 是条件, P_1 和 P_2 是两种处理操作。当满足条件时执行 P_1 , 否则执行 P_2 。

多选择: $\text{if}(\text{condition}_1) \{P_1\}$

.....

$\text{if}(\text{condition}_N) \{P_N\}$

其中 $\text{condition}_1, \text{condition}_2, \dots, \text{condition}_N$ 是条件, P_1, P_2, \dots, P_N 是满足相应条件时要执行的操作。

● 循环结构。

循环结构是表示程序反复执行某种操作,直到某条件为假时才终止循环,循环结构的流程如图 2 所示。

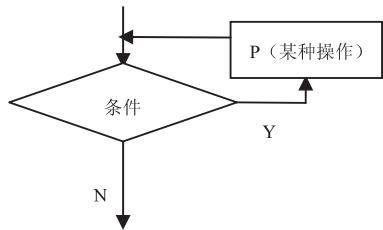


图 2 选择结构

无论软件系统的流程多么复杂,归根结底都可以用上述三种控制结构和它们之间的组合来表示。通过这三种结构可以建立 Web 服务构件之间的关系。在建立关系的时候,要注意两个 Web 服务间参数的个数与类型要一致(即前一服务返回的结果应该满足后一服务的输入要求)。

通过对系统进行分析,可以将系统逐步分解成单独的模块,直至每个模块可以由单独的 Web 服务构件实现,根据模块之间的关系结合上述的三种基本结构就可以制定出 Web 服务构件间的调用规则。当建立好规则后,有可能会出现规则冲突,即有多条规则的前件都一样,在文中建立规则的时候在规则的后面添加一个标志字段 nextId 来指示下一步调用哪条规则。

2.3 构件描述库

文中用数据库表存放 Web 服务构件的信息。表中的信息包括 Web 服务编号(Id)、服务功能、服务地址、服务的方法、服务的价格和服务的质量信息。

2.4 构件选择推理机

构件选择推理机的推理算法如下:

(1) 根据初始化条件,从规则库中选择满足条件的规则。

(2) 得到规则后,先根据规则表中的标志字段(文中是 IsOver)判断规则是否是终结规则,如果是则推理机执行完规则前件指示的当前构件后转第五步,否则推理机根据规则得到下一步需要的构件的功能后,查询构件描述表,根据表得到满足要求的 Web 服务构件,如果有多个构件的话形成一个集合,跳转到第三步,若只有一个转第四步,若没有符合要求的构件,则转第六步。

(3) 推理机根据一定的策略把满足要求的构件按满意度从高到低进行排序,推理机优先选择满意度较高的服务构件。

(4) 主动连接器执行推理机得出的 Web 服务构件。构件执行完成后,推理机根据结果和规则表重新推理选择规则(如果有规则冲突,用 nextId 解决冲突),转第二步。在执行过程中如果 Web 服务构件失效(Web 服务不能被访问了),若第二步查询结果是构件集合的话,那么根据第三步的排列结果从中依次选择构件。若所有的 Web 服务构件都不能被调用,则转第六步。

(5) 软件正常结束,给出结果。

(6) 软件运行过程中发生错误,终止软件运行,给出提示。

2.5 主动连接器

主动连接器的作用是调用 Web 服务构件。文中的主动连接器是一个 WebServiceHelper 类,该类的 InvokeWebService 方法能动态地调用 Web 服务构件,在调用时,需要传入 Web 服务地址 wsurl ,服务方法名 method 和具体参数 args 。

3 实验模拟

实验环境是 Visual Studio2008,数据库是 SQL Server2005。

以一个简单的数字处理业务说明软件演化过程,在该流程中,要求用户输入一个整数,系统对整数加五操作,然后判断结果是否大于 10,如果大于 10,则对数值再进行乘 2 和加 2 的操作然后输出结果,否则进行乘 3 和加 3 的操作;当进行完加 3 操作后判断值与 26 的关系,如果结果大于 26,则直接输出,否则对结果循环加 1 直到结果大于 26。

经过业务需求分析,得出业务流程图(见图 3),根据该流程可以建立软件演化规则库(见图 4)。要说明的是,该实验中规则库中的规则没有发生冲突,所以规则中的 nextId 为 Null。

实验中 Web 服务构件的价格区间是 $[10, 15]$,服

务质量区间是 $[0.7,0.95]$,实验中 Web 服务构件满意度函数定义为: $-1/20 * price+Qos$ ($price$ 是构件的价格, Qos 是服务质量)。假如一个构件的价格是 10,服务质量是 0.8,根据满意度函数计算出该构件的满意度是 0.3。实验中设定 Web 服务的超时时间为 5 s,若超过 5 s,推理机将从候选服务构件里选一个来替代。实验中用到的部分 Web 服务构件如图 5 所示,其中 Id 为 5 和 6 的 Web 服务构件运行的端口号是 4571,其他服务构件运行的端口号是 3625。

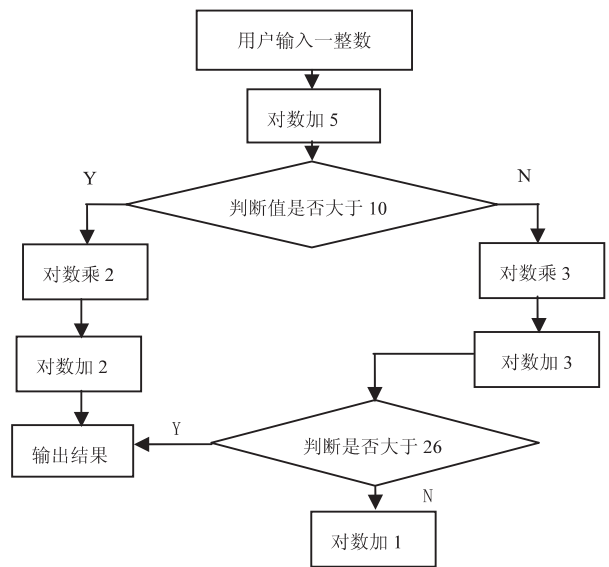


图 3 业务流程图

Id	currentwsfun	middlert	nextwsfun	IsOver	nextId
1	接收用户输入	Null	对数加 5	false	Null
2	对数加 5	Null	判断值是否大于 10	false	Null
3	判断值是否大于 10	Y	对数乘 2	false	Null
4	判断值是否大于 10	N	对数乘 3	false	Null
5	对数乘 2	Null	对数加 2	false	Null
6	对数乘 3	Null	对数加 3	false	Null
7	输出结果	Null	Null	true	Null
8	对数加 2	Null	输出结果	false	Null
9	对数加 3	Null	判断值是否大于 26	false	Null

图 4 演化规则库

id	wsfun	wsurl	method	price	Qos
1	接收用户输入	localhost:3625/s	getInput	10	0.8
2	对数加 5	localhost:3625/s	AddFive	15	0.9
3	判断值是否大于 10	localhost:3625/s	IsBig	12	0.7
4	对数加 10	localhost:3625/s	AddTen	10	0.8
5	对数乘 2	localhost:4571/s	MutiTwo	11	0.8
6	对数乘 3	localhost:4571/s	MutiThree	13	0.75

图 5 Web 服务构件

实验结果:
当用户输入不同的值如 4、2、7 时,软件会根据规则库与上下文调用相应的 Web 服务构件运行,调用 Web 服务构件的顺序如图 6 所示。

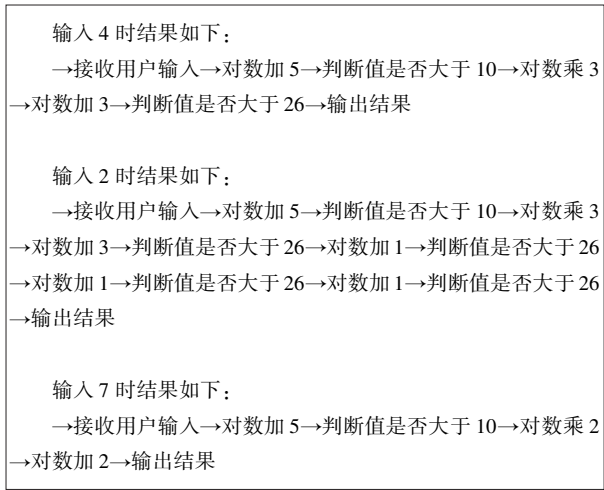


图 6 实验结果

4 结束语

文中针对当前在网构软件演化方面由于构件模型不同而构件之间不能交互的问题提出一种支持网构软件演化的软件结构,该结构的功能单元是独立的 Web 服务构件,给出了建立规则库的方法和推理算法。对多个满足要求的 Web 服务选择时,可以根据具体情况构造合适的满意度函数。

参考文献:

[1] 吕 健,马晓星,陶先平,等. 网构软件的研究与进展[J]. 信息科学,2006,36(10):1037-1080.
[2] 黄 昱,梅 宏,杨芙清. 基于反射式中间件的运行时软件体系结构[J]. 中国科学 E 辑:技术科学,2004,34(2):121-138.
[3] 张友生,李 雄. 基于构件运算的软件体系结构设计方法[J]. 计算机工程,2008,34(9):48-49.
[4] Bonnes U, Buzzard J, Dzelajlija Z, et al. Winedeveloper's guide[EB/OL]. 2006-12-22. <http://www.winehq.com/site/docs/winedev-guide/index>.
[5] Duffy J. CLRinside out: Know the costs[EB/OL]. 2006-09-10. <http://msdn.microsoft.com/msdnmag/issues/06/09/clrinsideout/decs/winedev-guide/index>.
[6] 杨芙清,吕 建,梅 宏. 网构软件技术体系:一种以体系结构为中心的途径[J]. 中国科学 E 辑:信息科学,2008,38(6):818-828.
[7] Oreizy P, Gorlick M M, Taylor R N, et al. An architecture-based approach to self-adaptive software[J]. IEEE Intelligent System, 1999, 14(3):54-62.
[8] Garlan D, Schemerl B. Using architectural models at runtime;


```

11.  $f[i] := \max + 1$ 
// 以下是输出数组  $a$  和  $b$  的最长公共上升子序列
长度  $\max$  值
12.  $\max := 0$  // 初始化
13. for  $i = 1$  to  $m$  do
14. if ( $\max < f[i]$ ) then
15.  $\max := f[i]$ 
16. print  $\max$ 
// 以下是输出 LCIS 序列
17. TYPE  $s = \text{RECORD}$  // 定义栈结构
    elem: ARRAY[1..1 000] of elemtp;
    top: 0..1 000
18.  $k := m$ 
19. while ( $\max < > 0$ ) do
20. if (( $f[k] = \max$ ) and(  $a[k] < (s.\text{elem}[s.\text{top}])$ )) then //  $f[k] = \max$  且  $a[k]$  小于栈顶元素
21. push_stack( $s, a[k]$ ) // 将  $a[k]$  入栈
22.  $k := k - 1, \max := \max - 1$ 
23. for  $i = 1$  to  $m$  do
24. LCIS[ $i$ ] := pop_stack( $s$ ) // 当  $\max = 0$  时, 栈
中所有元素出栈, 并依次存放在数组 LCIS 中

```

由于算法3将两个数组 a 和 b 合二为一考虑, 将问题转化为从数组 a 中找出和数组 b 元素相同且满足上升条件的子集, 使用一维数组存储长度及 LCIS, 从而实现空间状态压缩一倍, 空间复杂度降低至 $O(m)$ ($m \geq n$)。

算法3的时间复杂度与算法2相比, 并没有改变, 仍是 $O(mn)$ 。

2 三种算法的时间与空间复杂度分析

LCIS 的算法1和算法2相比, 考虑到 LCIS 属于经典的动态规划问题, 算法2充分利用了动态规划算法具有的最优子结构性性质, 将算法1的时间复杂度从 $O(m^2n^2)$ 降至 $O(mn)$ 。而算法3又在此基础上, 使用状态压缩对算法2存储空间进行了压缩, 构造状态转移方程, 将空间复杂度从 $O(mn)$ 降至 $O(m)$ ($m \geq n$) 或 $O(n)$ ($n \geq m$)。

(上接第39页)

- Research challenges[C]//Proc of the 1st European workshop on software architecture. Andrews:Spring-Verlag, 2004:200-205.
- [9] 黄 罡, 王千祥, 梅 宏, 等. 基于软件体系结构的反射式中间件研究[J]. 软件学报, 2003, 14(11):1819-1826.
- [10] 朱 庆, 王小平, 薛小平, 等. 基于构件的网构软件系统动态演化[J]. 计算机工程, 2010, 36(1):55-57.

3 结束语

文中探讨了求解最长公共上升子序列(LCIS)算法问题。对于 LCIS 问题, 通过对上述三种算法的时间和空间复杂度进行比较, 最终提出使用经过状态压缩的改进动态规划算法进行分析设计, 可将求解 LCIS 问题的算法时间和空间复杂度控制在二次方范围内, 从而有效降低了求解此类问题的时间和空间复杂度, 为后续的应用推广奠定了基础。

参考文献:

- [1] Alsuwaiyel M H. Algorithms design techniques and analysis [M]. Beijing: Publishing House of Electronics Industry, 2003.
- [2] Cooper R. Dynamic programming: An overview[EB/OL]. 2001. <http://econ. bu. edu/faculty/cooper/Dynprog/introlect. pdf>.
- [3] 王晓东. 计算机算法设计与分析[M]. 第2版. 北京: 电子工业出版社, 2005.
- [4] 王映龙, 杨炳儒, 宋泽锋, 等. 基因序列相似程度的 LCS 算法研究[J]. 计算机工程与应用, 2007, 43(31):45-47.
- [5] 张晓敏, 陈 昊, 明 仲. 寻找序列的变化内容[J]. 计算机工程与应用, 2011, 47(31):49-52.
- [6] 胡 婕, 业 宁, 罗晓波, 等. 多序列的近似 LCS 改进算法[J]. 计算机工程, 2011, 37(2):166-168.
- [7] 冷强奎, 秦玉平, 王春立. 基于句子相似度的论文抄袭检测模型研究[J]. 计算机工程与应用, 2011, 47(24):199-201.
- [8] 金 博, 史彦军, 腾弘飞. 基于篇章结构相似度的复制检测算法[J]. 大连理工大学学报, 2007, 47(1):125-130.
- [9] 林贤明, 李堂秋, 陈毅东. 句子相似度的动态规划求解及改进[J]. 计算机工程与应用, 2004, 40(35):64-65.
- [10] 王 品, 黄广君. 信息检索中的句子相似度计算[J]. 计算机工程, 2011, 37(12):38-40.
- [11] 严华平, 李 刚, 张建宏. 生物信息挖掘中 LIS 算法研究[J]. 计算机应用研究, 2009, 6(1):62-63.
- [12] Yang I-Hsuan, Huang Chien-Pin, Chao Kun-Mao. A fast algorithm for computing a longest common increasing subsequence[J]. Information Processing Letters, 2005, 93:249-253.
- [11] 吴 波. 面向构件的软件系统动态配置技术的研究[D]. 北京: 北京工业大学, 2010.
- [12] 余 萍, 马晓星, 吕 建, 等. 一种面向动态软件体系结构的在线演化方法[J]. 软件学报, 2006, 17(6):1360-1371.
- [13] 李玉龙, 李长云. 软件动态演化技术[J]. 计算机技术与发展, 2008, 18(9):83-86.
- [14] 管贤春. 基于构件化软件的动态演化研究与应用[D]. 广州: 广东工业大学, 2010.

基于体系结构的网构软件演化关键技术研究

作者：

陈志雨，刘俊强，叶子文，胡明，[CHEN Zhi-yu](#)，[LIU Jun-qiang](#)，[YE Zi-wen](#)，[HU Ming](#)

作者单位：

[长春工业大学 计算机科学与信息工程学院, 吉林 长春, 130012](#)

刊名：

[计算机技术与发展](#)

英文刊名：

[Computer Technology and Development](#)

年，卷(期)：

2014(5)

本文链接：http://d.wanfangdata.com.cn/Periodical_wjfz201405009.aspx