



基于语义的 NGN 融合业务合成算法研究^{*}

彭荣群¹, 张连俊¹, 糜正琨²

(1. 山东理工大学计算机学院 淄博 255049; 2. 南京邮电大学通信与信息工程学院 南京 210003)

摘要: 分析了当前两种基于语义的业务合成算法在应用于电信域和网络域业务融合时的不足, 提出了一种新的基于语义的双向链自动业务合成算法, 并对该算法的性能进行了详细分析, 证明了其在业务的平均合成时间和合成成功率上明显优于已有的相关算法。最后给出了一个融合业务的合成实例, 以验证算法的正确性。

关键词: 语义; NGN 融合业务; 业务合成算法; 双向链合成算法

doi: 10.3969/j.issn.1000-0801.2013.01.012

Research on Semantic Based NGN Converged Service Composition Algorithm

Peng Rongqun¹, Zhang Lianjun¹, Mi Zhengkun²

(1. College of Computer Science and Technology, Shandong University of Technology, Zibo 255049, China;

2. School of Telecommunication and Information Engineering,

Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: Analyzing the disadvantage of the two existing propylene-based service composition algorithms used in the converged process between telecommunication services and web services, a novel semantic-based double-chaining automatic composition algorithm was presented. Also, a distinct virtue in less time cost and higher composition success rate of the proposed algorithm than that of these existing related algorithms was proved by discussing its performance in detail. Additionally, a composing scenario of converged service was given to support the proposed DCA algorithm.

Key words: semantic, NGN converged service, service composition algorithm, double-chaining composition algorithm

1 引言

NGN(next generation network, 下一代网络)是业务驱动的网络, 能否快速高效地提供用户所需要的各种业务决定着网络未来的发展。在过去的 20 多年中, 计算机网络和电信网络都在不断地演进, 但二者提供业务的机理不同^[1]: 电信网络是一个封闭的垂直网络, 提供保证收费质量的业务; 而计算机网络是一个开放的平面网络, 提供免费的尽力而为的服务, 因此它们在很长一段时间内都各自独立地

提供基于自身网络的业务。但随着社会的进步以及电信网向全 IP 网的演进, 用户越来越不满足于只使用由计算机网络或电信网络提供的单纯业务, 而是希望能够享用由这两种网络提供的融合服务。根据客户请求自动发现、匹配以及动态合成新业务, 特别是由两种网络业务合成的融合业务, 成为未来网络发展的关键技术。

NGN 以前的电信网络是一个封闭的网络, 只有电信工程师才能理解电信业务的提供机制, 导致了两种业务合成过程实现的困难。Web Service^[2]技术的出现使业务的合成

^{*} 山东理工大学博士基金资助项目(No.412011)

变得简单并易于实现。Web Service 是近年来计算机界提出的一种新的分布技术,具有平台独立性、松耦合性、自包含性、基于可编程性等诸多优点,在计算机网络领域得到了广泛的应用。随着 Web Service 技术的成熟,将该技术应用于电信网络中成为现在研究的一个新热点^[3]。由 ETSI、OSA Parlay Group 以及 3GPP 定义的 Parlay X Web Services API^[4]和由欧洲计算机制造商协会(ECMA)制定的一系列计算机支持的电信应用(CSTA)^[5]标准,都是 Web 业务技术在电信领域的成功应用。

传统的电信业务可以被封装成一系列具有标准接口的 Web Service,具有丰富网络业务开发经验的非电信业务开发人员可以像调用普通网络业务那样调用电信业务,在传统的网络业务中融入呼叫、即时消息等电信业务功能,实现网络业务和电信业务的无缝集成,缩短了融合业务的开发周期,给电信业带来了新的发展机遇。

Parlay X Web Service 和 CSTA 业务与 Web Service 类似,都基于 WSDL 描述业务,它们没有提供业务的语义信息,不支持基于语义的操作,不利于业务的自动发现、调用与合成。这就需要采用机器能够理解的基于语义的 Web 业务描述语言。

语义 Web 业务描述语言采用一种具有明显语义的表示方式来表示和交换信息,以利于 Web 业务的自动处理。基于语义支持的自动业务合成算法主要采用 OWL-S^[6]、WSMO^[7]两种语义建模语言来描述业务,它主要包含两种具体算法^[8],即前向链合成算法(forward-chaining composition algorithm, FCA)^[9]和无回溯的反向链合成算法(non-backtrace backward-chaining composition algorithm, NBCA)^[10]。这两种算法虽然在网络域得到了广泛的应用,但对于由网络域和电信域业务合成的融合业务来说,它们存在着明显的不足。

1.1 前向链合成算法

参考文献[9]给出了一种前向链合成算法,该算法的基本思想是根据用户的输入来推导各种 Web Service 输出,把推导中用到的 Web Service 和它们产生的输出分别加到 Web Service 输入集合和 Web Service 输出集合中,同时要把每次 Web Service 产生的输出加到 Web Service 的输入集合中,这样一直迭代,直到用户所请求的结果完全包含在 Web Service 输出集合中。

但 FCA 存在几个很显著的问题,介绍如下。

- 该算法结束的条件是用户请求的输出都已经获得。

对于电信业务来说,这个结束条件并不充分,因为某些电信业务只是执行一个动作,并没有明显的输出。如果没有用户要求的输出参数,即 $U_0=\emptyset$,那么 FCA 就不执行。以 MakeCall 业务为例,该业务的主要功能是建立通信双方的连接,而不是产生明确的输出结果。

- 算法中没有考虑业务的前提(P)和效果(E)参数,而这两个参数对电信域业务来说是非常重要的。
- 没有考虑对于合成中需要的某些业务,用户未提供它的某些输入或某些前提不满足时的情况。

当算法遇到上述情况时就会返回,导致合成失败结果,使合成成功率降低。

1.2 无回溯反向链合成算法

无回溯反向链合成算法的基本思想是:从用户所想要得到的查询结果开始,试图找出能得到该结果所需要的所有生成式。如果存在多个与当前目标匹配的生成式,则先去缓冲区查找有无对该输入用户的输入闭包;若没有,则先要计算该用户的输入闭包,并放入缓冲区中,然后选用生成式体部项全部包含在输入闭包中的生成式进行合成。输入闭包计算只需要进行一次,只要用户的输入和模型描述库不变,以后的合成都可以从缓冲区中取出使用。

与 FCA 相比, NBCA 由于在算法中引入了输入闭包,克服了前向链合成算法中多次回溯的缺点,因此在合成效率方面有了显著的提高。但对于电信域业务来说,仍不能直接使用,原因如下。

- NBCA 的启动条件是用户需要的输出参数集合非空,但由于很多电信业务只是要求完成一个动作,并没有明确需求的输出,这样 NBCA 就没法启动。
- 算法中也没有考虑业务的前提(P)和效果(E)参数,而这两个参数对电信域业务来说是非常重要的。

针对以上分析,本文提出了一种双向链合成算法(double-chaining composition algorithm, DCA),以满足网络域和电信域业务合成的需求。

2 双向链合成算法的设计

DCA 的基本思想是:首先针对 FCA 和 NBCA 存在的问题,对它们分别进行改进;然后把两种业务合成算法结合起来使用,并充分利用两者各自的优点。当用户请求的输出集合为空($U_0=\emptyset$)时,由于 FCA 用 U_0 作为算法的结



束条件,因此对它进行简单的改进后就可以顺利完成合成过程,但对 NBCA 来说,就很难做到这一点,因为 U_0 是该算法的启动条件。因此在 $U_0=\emptyset$ 的条件下,可以首先采用改进后的 FCA,如果能成功完成合成,那么就输出合成结果;对于合成不成功的情况,如果此时的 U_0 已经非空 ($U_0\neq\emptyset$),那么还可以启动 NBCA 进行合成,这样可以进一步提高业务合成的成功率。如果一开始 U_0 就非空 ($U_0\neq\emptyset$),那么就可以直接启动改进后的 NBCA,这样仍可以保证较好的合成成功率。DCA 的基本思想如图 1 所示。

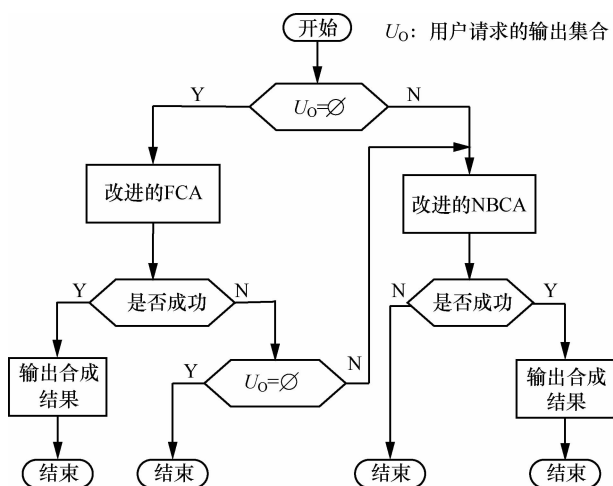


图 1 DCA 的基本思想

2.1 对 FCA 的改进

对 FCA 的改进介绍如下。

- 该算法结束的条件根据用户请求的输出都已经得到。如果没有用户要求的输出参数,即 $U_0=\emptyset$,那么 FCA 就不会执行。因此,为了适应电信业务的特殊性,在前期工作中^[11]对该算法进行了改进,在结束条件中加入了功能说明项,使算法只有在所有输出都得到且所有要执行的功能都完成后才能结束。
- 在算法中增加了业务的前提(P)和效果(E)参数的匹配。
- 考虑了在合成中需要的某些业务,在用户未提供某些输入或某些前提不满足的情况。当合成需要的业务的某些输入参数不在用户输入集合中时,就把这些没有的参数加入用户需求的输出集合中,再去查找能生成它们的业务,使合成得以完成,由此提高合成的成功率。

改进的 FCA(improved FCA,I-FCA)算法参见参考文献[11]。

2.2 对 NBCA 的改进

对 NBCA 的改进介绍如下。

- 与对 FCA 的改进类似,在算法中增加了对业务功能(F)的匹配。也就是说,算法结束的条件是除了要得到用户所要求的所有输出参数之外,还要完成用户要求的所有功能。
- 算法中增加了业务的前提(P)和效果(E)参数的匹配。

改进后的 NBCA(improved NBCA,I-NBCA)的计算流程如图 2 所示。

2.3 双向链合成算法

综合上述 I-FCA 和 I-NBCA,本文提出了支持网络域和电信域业务合成的双向链合成算法,计算流程如图 3 所示。

DCA 采用 Java 语言编程开发。程序中利用美国马里兰大学开发的 Mindswap OWL-S API 调用 Pellet 推理机来获取两概念间的语义相似度。另外,在算法中把 I 和 P、O 和 E 同等对待,因为只有在业务执行时才需要验证前提(P)是否满足,效果(E)是否能得到;而在业务合成中,业务只需要 P 和 E 参数项,不用验证,因此它们与 I、O 地位相同,可以放在一起处理。

3 算法性能

3.1 性能指标

对于业务合成性能,可以从以下两方面来测试。

- 平均合成时间:指成功合成业务所用时间的平均值,表示如下:

$$\text{平均合成时间} = \frac{\text{成功合成业务所用的总时间}}{\text{成功合成的业务总数}} \quad (1)$$

- 合成成功率,表示如下:

$$\text{合成成功率} = \frac{\text{成功合成的业务个数}}{\text{用户提出的合成请求总数}} \quad (2)$$

3.2 性能测试的设计

性能测试的步骤设计如下。

(1)测试用业务集合包括扩展后的 CSTA 电信域业务(共 154 个)和网络域业务(OWLS-TC_v2.2_revision_3_(profile_taxonomy)^[12]测试集的 1 007 个业务)。

(2)合成请求由系统随机生成,I、P 和 O、E 的个数从 0 到 10 逐渐变大,I、O、P、E 参数均来自于从测试集业务中提取的 I、O、P、E 参数集合。

(3)在具体测试中采用 2.6 GHz 双核 Intel 处理器、内存 2 GB 的个人计算机实现对算法性能的评估,每组 100 个业务请求,分别做 100 组求均值。

```

UserInputs: 用户的输入;
UserPreconditions: 用户提供的前提;
UserOutputs: 用户请求的输出;
UserEffects: 用户请求的效果;
UserFunctions: 用户请求的功能;
RulesSet: 产生式(模型)集合;
InputClosure: 输入闭包;
ServicesUsed: 合成中用到的 Service 集合
AvailableOutputs: 可用的 Service 输出集合
RulesUsed: 合成中用到的产生式;
NonBacktraceBackwardChaining (UserInputs, UserOutputs,
UserPreconditions, UserEffects, UserFunctions, RuleSet)
{
    AvailableOutputs=UserOutputs $\cup$ UserEffects, RulesUsed= $\emptyset$ ,
    ServicesUsed= $\emptyset$ ;
    repeat
    {
        For each subgoal AvailableOutputs,
        if(subgoal UserInputs) then
        {
            AvailableOutputs=AvailableOutputs-subgoal;
            UserFunctions=UserFunctions-Function(subgoal)
        }
    }
    elseif({subgoal}  $q(p_1, p_2, \dots, p_n, q)$  RulesSet)
    {
        Failed; break;
    }
    elseif((!subgoal)  $q=1$ ) then
    {
        If( $r$  RulesUsed)
        RuleUsed=RuleUsed +  $r$ ;
        if (WS( $r$ ) ServicesUsed)
        ServicesUsed= ServicesUsed+WS( $r$ );
        if( $p_1, \dots, p_n$  AvailableOutputs)
        AvailableOutputs=AvailableOutputs+( $p_1, \dots, p_n$ );
        AvailableOutputs=AvailableOutputs-subgoal;
        UserFunctions=UserFunctions-Function(subgoal)
    }
    else
    {
        {
            If((Cache InputClosure (UserInputs))= $\emptyset$ ) then
            {ComputingInputClosure (UserInputs, UserPreconditions,
            RulesSet)}
            for each  $r: p_1, \dots, p_n, q \in$  InputClosure,
            if(subgoal= $q$ ) do
            {
                If( $r$  RulesUsed)
                RuleUsed=RuleUsed +  $r$ ;
                if (WS( $r$ ) ServicesUsed)
                ServicesUsed= ServicesUsed+WS( $r$ );
                if( $p_1, \dots, p_n$  AvailableOutputs)
                AvailableOutputs=AvailableOutputs+( $p_1, \dots, p_n$ );
                AvailableOutputs=AvailableOutputs-subgoal;
                UserFunctions=UserFunctions-Function(subgoal)
            }
        }
        } Until( AvailableOutputs= $\emptyset$ &&(UserFunctions= $\emptyset$ ))
    }
    return ServicesUsed, RulesUsed;
}
    
```

图 2 I-NBCA 计算流程

3.3 测试结果

FCA、NBCA、DCA 3 种算法的合成时间与请求的输出个数、参与的业务数量以及用户输入个数之间的关系曲线

```

PublishedServices: 已发布的 Service 集合
UserInputs: 用户的输入;
UserPreconditions: 用户提供的前提;
UserOutputs: 用户请求的输出;
UserEffects: 用户请求的效果;
UserFunctions: 用户请求的功能;
ServicesUsed: 合成中用到的 Service 集合
AvailableInputs: 可用的 Service 输入集合
AvailableOutputs: 可用的 Service 输出集合
AvailableFunctions: 可用的 Service 功能集合
CompletedFlag: 合成的成功标志
RulesSet: 产生式(模型)集合;
InputClosure: 输入闭包;
DoubleChaining (UserInputs, UserOutputs, UserPreconditions,
UserEffects, UserFunctions, RuleSet)
{If (UserOutputs= $\emptyset$ ) then
    { call I-FCA;
      If (! CompletedFlag)&&(UserOutputs??)
        call I-NBCA;
    }
  Else
    call I-NBCA;
}
    
```

图 3 DCA 计算流程

分别如图 4~图 6 所示。

由图 4~图 6 可以看出, 3 种算法的平均合成时间均随着用户提供的输入参数个数、用户请求的输出参数个数和参与的业务数量的增加呈现上升趋势。其中, FCA 的平均合成时间最长, NBCA 所用时间最少, 而 DCA 略高于 NBCA, 这是因为 FCA 是由输入推导输出, 算法中存在回溯, 而 NBCA 中消除了回溯, 因此所用时间最少, 合成效率最高。而对于 DCA, 当用户需求的输出参数个数集不为空集时, 采用的是改进型的 NBCA, 因此合成效率与 NBCA 类似, 但由于在算法中增加了 P/E 和 F 参数的匹配, 因此所用时间略高于 NBCA。

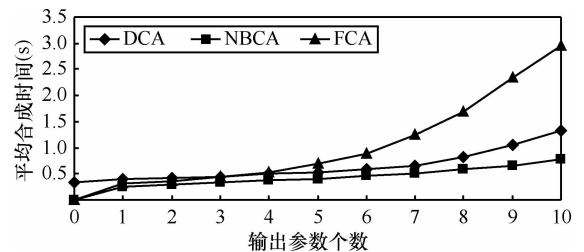


图 4 平均合成时间与用户请求的输出参数个数之间的关系

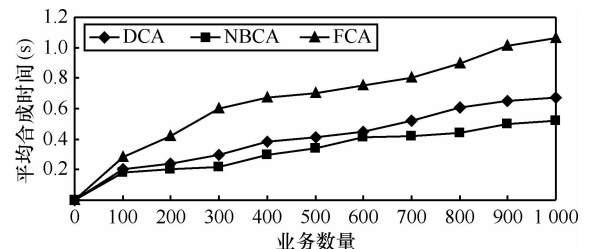


图 5 平均合成时间与参与的业务数量之间的关系

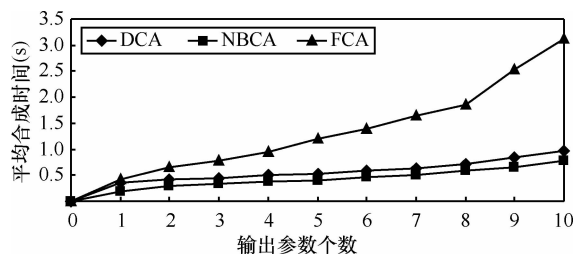


图6 平均合成时间与用户提供的输入参数个数之间的关系

单就合成效率来看,DCA不是最好的,但从合成的成功率来看,DCA却是最好的,如图7~图9所示。特别是当用户请求的输出参数集为空时,FCA和NBCA根本就不能启动,因此合成成功率为0;而DCA就可以启动并完成合成,如图9所示。这一点特别适合应用于电信业务和网络业务融合的场所。

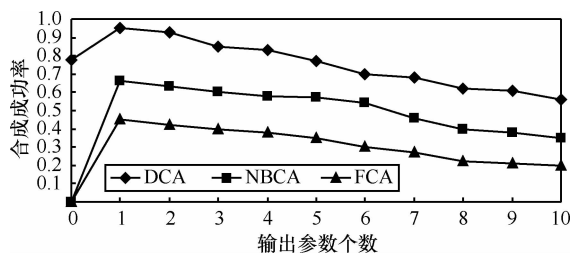


图7 合成成功率与用户请求的输出参数个数之间的关系

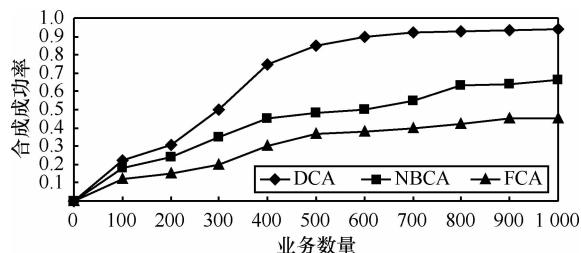


图8 合成成功率与参与的业务数量之间的关系

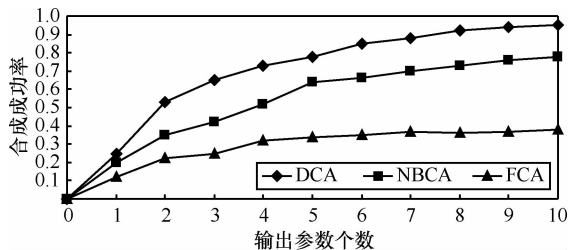


图9 合成成功率与用户提供的输入参数个数之间的关系

4 业务合成实例

假设某“股票信息业务”的需求是:从某个日期开始,每隔一段时间自动查询在某个城市股票交易所的某支股票的价格信息,如果价格高于或低于某个数值,就自动接通用户和股票经纪人的电话,以便商讨如何处理股票。

该用户请求的业务可以表示为:

WSautoStockQuote (I:stockname,city,datetostart,interval,toplimit,bottomlimit,usertelnum,stockbroker telnum;O: Ø;P: Ø;E: establish a call between user and broker; F: timing stock quote, data comparison, make call)

利用上面提出的DCA可以找到9个原子业务,其中6个是电信域业务,3个是网络域业务,如图10所示。输出的合成业务如图11所示。

WS₁: Timing Stock Quote Service
I: stockname,city,datetostart,interval
O:stockprice

WS₂: Data Comparison Service
I: toplimit,bottomlimit,inputdata
O:overlimit

WS₃: Bridge Service
I: Ø
O: applicationInfo,resquestedProtocolVersions(RPV)

WS₄: Start Application Session Service
I: applicationInfo,resquestedProtocolVersions(RPV)
O: sessionID,actualProtocolVersion(APV), actual SessionDuration(ASD)

WS₅: Make Call Service
I: callingDevice(CD),calledDirectoryNumber(CDN)
O: callingDevice
E: established
P: sessionID

WS₆: Monitor Start Service
I:monitorObject(=CallingDevice)
O:crossRefIdentifier(CRI)
E:connection cleared
P:sessionID

WS₇: Monitor Stop Service
I: crossRefIdentifier
P: call cleared

WS₈: Clear Call Service
I:callToBeCleared(=CallingDevice)
E:call cleared
P:connection cleared

WS₉: Stop Application Session Service
I:sessionID
P:call cleared

图10 DCA中找到的原子业务

5 结束语

前向链合成算法和无回溯反向链合成算法是目前常用的两种基于语义支持的自动业务合成算法,但由于它们是基于网络域开发的,因此不太适用于网络域和电信域融合业务的合成。本文通过分析两种算法的不足,提出了一种新的基于语义的双向链自动业务合成算法,克服了现有算法的不足,并在缩短平均合成时间和提高合成成功率方面获得了明显的改进。并且利用该算法时,用户只需要提

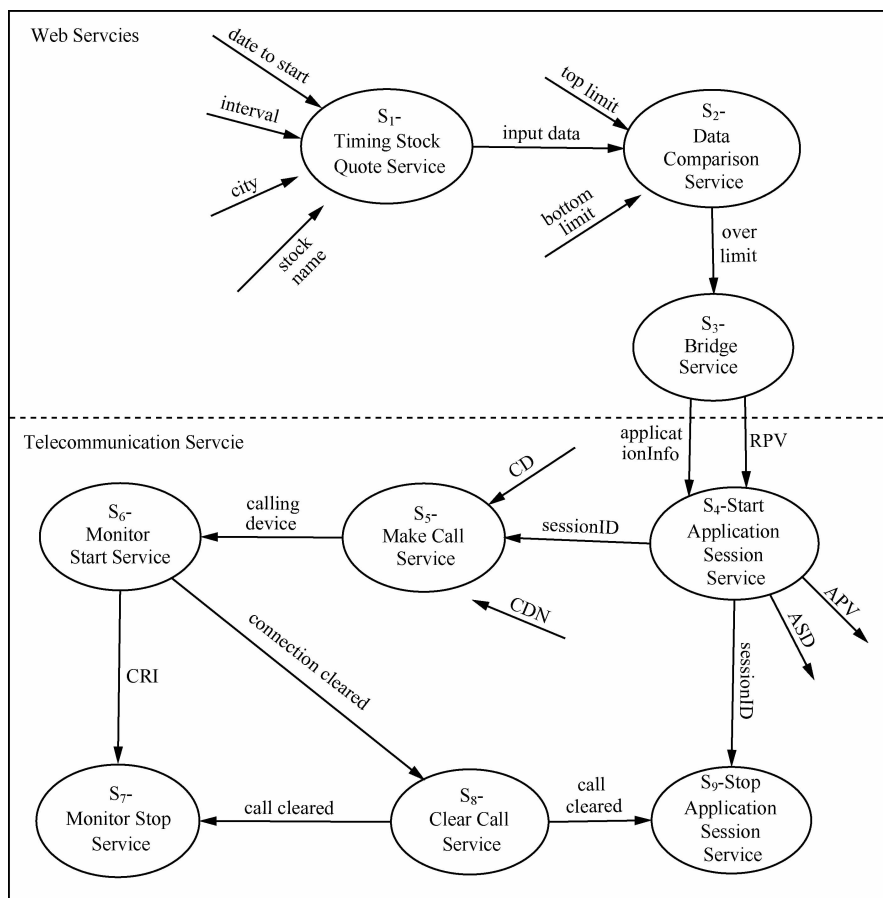


图 11 自动股票查询系统的合成结果

供必要的输入输出信息、部分处理过程,在合成过程中需要的其他部分可以由系统自动生成,这对不熟悉电信业务的 IT 开发人员是非常有帮助的,既节省了开发成本,又缩短了开发周期,提高了开发效率。

参考文献

- 1 乔秀全, 李晓峰. 融合业务网络的发展和演进技术研究. 中国新通信, 2008(7): 77~84
- 2 Booth D, Haas H, McCabe F, *et al.* NOTE-ws-arch-20040211. Web Services Architecture. W3C, Massachusetts, USA, 2004
- 3 Griffin D, Pesch D. A survey on web service in telecommunications. IEEE Communications Magazine, 2007(7): 28~35
- 4 Parlay X Working Group. ES 202 504. Parlay X 3.0 Web Service. ETSI, Nice France, 2007
- 5 ECMA-269. Services for Computer Supported Telecommunications Applications (CSTA) Phase III, 6th Edition. <http://www.ecma-international.org/publications/standards/Ecma-269.htm>, 2004
- 6 Martin D, Burstein M, Hobbs J, *et al.* Semantic Markup for Web Services. W3C, Massachusetts, USA, 2004
- 7 Bruijn J, Bussler C, Domingue J, *et al.* SUBM- WSMO- 20050603. Web Service Modelling Ontology. W3C, Massachusetts, USA, 2005
- 8 顾宁, 刘家茂, 柴晓路等. Web Services 原理与研发实践. 北京: 机械工业出版社, 2006
- 9 Thakkar S, Knoblock C A, Ambite J L. Dynamically composing web service from on-line sources. AAA Workshop on Intelligent Service Intergration, Edmonton, Alberta, Canada, 2002: 54~80
- 10 Liu J M, Gu N, Shi B L. Non-backtrace backward chaining dynamic composition of web service based on mediator. Journal of Computer Research and Development, 2005, 42(7): 1153~1158
- 11 彭荣群, 糜正琨. 基于 CSTA 扩展接口的下一代网络融合业务合成研究. 南京邮电大学学报(自然科学版), 2009, 29(1): 23~28
- 12 Meditskos G, Bassiliades N. OWLS-SLR: an OWLS service profile matchmaker. <http://lpis.csd.auth.gr/systems/OWLS-SLR/datasets.html>, 2009

【作者简介】 彭荣群,男,博士,山东理工大学副教授,主要研究方向为下一代网络的理论与技术。张连俊,男,山东理工大学教授,主要研究方向为现代通信网理论与技术。糜正琨,男,南京邮电大学教授、博士生导师,ITU-T 中国专家组成员,主要研究方向为宽带交换和通信网技术。

(收稿日期: 2012-09-29)