计算机研究与发展
Journal of Computer Research and Development
ISSN 1000-1239/CN 11-1777/TP
43(11):1895~1903, 2006

一种支持最终用户探索式组合服务的方法

韩燕波[1] 王洪翠[1,2] 王建武[1,2] 闫淑英[1,2] 张 程[1,2]
[1](中国科学院计算技术研究所网格与服务计算研究中心 北京 100080)
[2](中国科学院研究生院 北京 100049)
(yhan@ict.ac.cn)

An End-User-Oriented Approach to Exploratory Service Compostion

Han Yanbo[1], Wang Hongcui[1,2], Wang Jianwu[1,2], Yan Shuying[1,2], and Zhang Cheng[1,2]
[1](*Research Center for Grid and Service Computing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100080)
[2](*Graduate University of Chinese Academy of Sciences, Beijing* 100049)

Abstract Service composition is a prevalent means to construct service-oriented, loosely-integrated and agile applications. Current service composition technologies are mostly developed for the situations where business requirements are definite and business processes can be predefined. However, there exist many problem-solving cases in which people are not totally clear in advance how to compose applications, but have to act in a try-and-error manner. A novel approach to dealing with the challenge is proposed. Focuses are on the programming model supporting exploratory service composition as well as a dynamic service recommendation mechanism. The proposed approach is evaluated from the users' perspective.

Key words end-user programming; exploratory service composition; service recommendation; programming model

摘 要 服务组合是构建面向服务、松耦合、高适应性的应用系统的主要途径. 现行服务组合技术大多只适于构造需求明确、业务流程可预先定义的情况. 然而,现实中存在大量需要边执行边探索、"摸着石头过河"式的问题求解形式. 针对这类需求,结合网络化科研协作中的实际问题,提出一种最终用户可探索式服务组合的方法,重点讨论了探索式服务组合编程模型、服务推荐等关键问题. 此外,从使用者角度对所提出的组合方法及其实现机制给予了定性分析和评价.

关键词 最终用户编程;探索式服务组合;服务推荐;编程模型

中图法分类号 TP311.11

经过多年的信息化建设,人们在享受IT技术带来的种种好处的同时,也被形成的"信息孤岛"或"烟囱系统"带来的条块分割、相互封闭等问题所困惑. 如何按需、即时、灵活地集成各类应用,构建集成化的信息系统是IT领域近年来备受关注的热门话题之一. 在此,传统的集成方式面临以下挑战:①缺乏统一的标准和规范,需要定义专有的接口和封装策略,开发效率低;②集成方式过于僵死,难以应变,灵活性差;③需要专业人员帮助集成,难以满足即时性要求.

面向服务计算范型以其标准化、松耦合的集成方式为提高应用系统的互操作能力和敏捷性带来新的曙光. 在面向服务计算范型下,应用系统的构造以服务作为基本单元通过服务组合完成. 由于现行

收稿日期:2006-07-04;修回日期:2006-08-29
基金项目:国家自然科学基金项目(90412005,60573117)

的服务组合方法(如 eFlow[1],SELF-SERV[2]等)和服务组合语言(如 BPEL4WS①,BPML②,WSCI③等)都侧重于软件层面,主要是为软件专业人士而设计,没有充分考虑业务人员的参与(甚至主导),应用系统的建设周期长、开发成本高,应用的即时集成仍旧是一个挑战性问题.

为进一步缩短开发周期,适应快速多变的业务需求,人们开始考虑能否让业务用户按照其业务规范直观、便捷地配置和组织应用资源,自行地构造应用. 我们把这种应用构造方式称为业务端编程. 事实上,从过去的以技术为中心构造应用到以用户为中心构造应用的模式转换正是近年来信息技术领域的一项重大的思维变革.

在科研协作、远程医疗、城市应急等应用领域问题求解的过程中,人们逐渐认识到许多业务流程难以预先定义完备,需要在业务进行过程中实时地根据已完成业务活动的结果和效果、周围环境的信息等确定下一步要做的事. 我们在研发面向科研工作者的动态协同问题求解环境的过程中就遇到了类似的需求及过程不确定性的问题. 在基因工程研究中,一项实验通常要经过多个实验环节,涉及若干分布、异构的计算机应用,需要协同使用本地和远程的信息存储和计算能力,且结果无法完全预测. 当前,网上存在相当数量以 Web 服务形式的计算和信息资源. 为完成上述分析和研究工作,科研工作者需要组织和利用这些 Web 服务.通过引入业务端编程方法,可为科研工作者屏蔽底层细节,特别是 IT 技术相关的细节. 然而,为充分利用这些网上资源,还存在以下挑战:科研工作者做研究时,面对待解问题,往往事先不清楚实验的整个过程,需要根据前一阶段的实验结果来确定下一步的实验内容,根据结果来反复修改实验参数(包括调用不同的服务),这更需要系统提供包括主动推荐在内的辅助智能,最终达到满意的实现效果.

本文针对网络化科研协作项目中的实际问题,探索相应的服务组合方法及其支撑环境,让最终用户在不完全确定组合过程的情况下像使用"文本超链接"一样选择业务层面那些用户可理解的抽象服务,实现"探索式"的应用即时构造,而将相关细节推到技术层面解决.

1 基于 VINCA 业务服务的探索式服务组合基本思路

为支持业务用户主导的网络应用即时构建,我们提出了业务端编程语言 VINCA[3],并实现 VINCA 支撑环境. 业务人员可利用 VINCA 来配置经过了虚拟化处理、具备业务语义的 IT 资源[4]和规划其业务流程. VINCA 采用以业务活动为中心的编程方式,其核心元素是业务活动的一种抽象形式——业务服务[5]. 业务服务一方面体现为业务用户可理解的业务级构件,另一方面也定义了与相关 IT 资源的关联关系,以逐级抽象的方式屏蔽底层计算资源信息. 图 1 示意了我们前期工作定义的业务服务"冰山"模型.

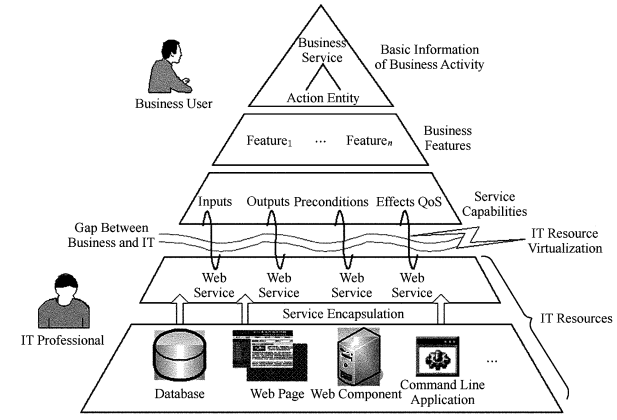


Fig. 1 Iceberg model of business service.

图 1 业务服务冰山模型

本文关注用户主导的问题求解过程的不确定性问题,在 VINCA 方法学的基础上提出了图 2 所示的面向服务计算环境下探索式服务组合方法. 在技术路线上,通过用户自主编程和系统主动推荐相结合,实现用户主导的探索式编程. 文中拓展 VINCA 业务服务,形成适于探索式编程,被称之为业务积木(business lego)的新元素. 图 2 中的各类箭头对应探索式服务组合需要的主要操作,其中空心箭头表示用户的"编程"操作,实心箭头表示编程环境的支撑操作. 首先对可用的 Web 服务进行虚拟化得到最终用户可直接使用的业务积木,加以特定的限定和约束后的某一逻辑领域(逻辑领域是针对问题求解需要对业务领域逻辑上的组合和划分)上的业务积木集合

① Business Process Execution Language for Web Services (BPEL4WS), http://www-106.ibm.com/developerworks/Webservices/library/ws-bpel/

② Business Process Modeling Language(BPML), http://www.bpmi.org/

③ Web Service Choreography Interface (WSCI), http://www.w3.org/TR/wsci/

形成该领域的业务服务社区. 在社区中用户可按需选择并配置积木,系统也会向用户推荐适合的积木,用户可执行并根据执行结果继续构建和执行应用.

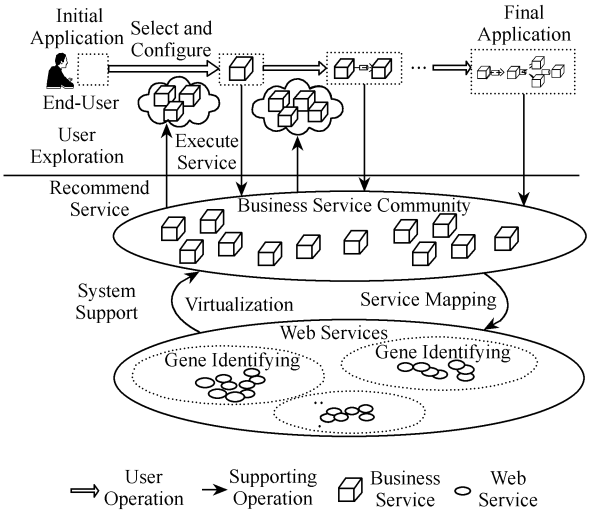


Fig. 2 Principle of exploratory service composition.

图 2 探索式服务组合基本思路

图 2 体现了以下求解问题的新思路:① 呈现最终用户可理解的服务,最终用户是指科研工作者而非专业 IT 人士,因此需要对 Web 服务进行抽象;② 支持用户运行时构建应用,提供"边执行边构造"的应用构造方式;③ 提供辅助编程机制,即时向用户推荐服务,辅助用户快速有效地完成整个应用的开发.

2 编程模型

一个编程模型对应着计算机及其基本支撑系统的一种特定的抽象形式,其具体内容和工作模式取决于求解问题的思路和方式. 一个有效的编程模型总能以某种有效的手段指导资源利用并得出可用的问题求解程序. 针对不同的使用对象和用途,编程模型的构成、抽象程度、表达能力和完备性也各不相同. 本文提出的探索式服务组合方法,旨在让最终用户在系统的帮助下快捷、高效地组装出个性化的、能满足其即时需求的网络化应用,从而降低网络化应用的开发成本,提高开发效率和应变能力. 本节探讨与之相应的一种新型编程模型.

2.1 业务积木与应用超链

如图 3 所示,从用户使用角度,一个业务积木可以被定义成:

定义 1. 业务积木. 业务积木是以业务活动为中心的一种资源抽象表示形式,作为用户直接使用或按需组装的基本构造单位. 它具有业务人员可理解、可配置、可执行、可组合的性质,独立于具体计算资源但可与其进行绑定. 具体可以表示为

$$bizLego = (basicInfoView, bizFeatureView, capabilityView, linkView, implView),$$

其中,$basicInfoView$ 为基本信息视图,描述业务活动基本功能,是业务人员选择服务的基本依据,具体包括业务积木的名称、业务分类、基本动作和客体类型;$bizFeatureView$ 为业务特征视图,描述业务活动的相关业务特征,支持业务人员详细定义服务和个性化配置;$capabilityView$ 为服务能力视图,描述输入输出、前提、效果和服务质量等信息;$linkView$ 为连接视图,描述该业务积木可能的后续连接关系;$implView$ 为实现视图,记录为完成该业务活动可绑定的 IT 资源.

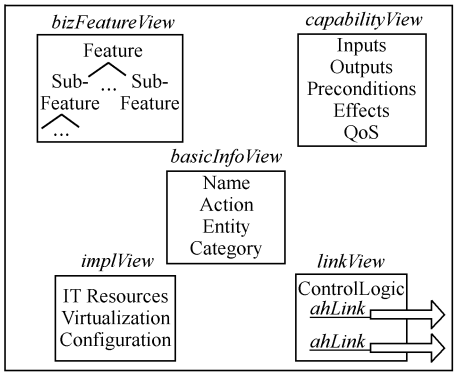


Fig. 3 Views of business lego.

图 3 业务积木视图

针对业务端探索式编程的特殊要求,业务积木具备以下特性:

1) 让业务人员可理解,分级可视

不同用户角色、兴趣、能力不同,对同一业务活动的专注点也不同,因此业务积木以多视图形式描述,使业务人员可从描述最基本信息的视图出发,通过不同视图的有机关联,按需查看各个视图,以了解更详细信息,同时以渐进方式了解和落实业务活动.

2) 是信息自含的独立业务处理单位

积木的基本信息视图以"动作+客体"形式(例如"查询库存")描述单一业务活动功能.

每个积木是一个自包的业务处理单元,包含了刻画、组合和执行一个业务活动的全部信息:

① 业务描述信息和业务操作信息,分别记录在积木的基本信息视图和业务特征视图;

② 落实业务操作的 IT 资源的配备和绑定信息,记录在积木的实现视图;

③ 内置的和其他业务积木的链接关系,记录在

积木的连接视图.

3) 业务人员可自行配置

业务特征视图通过业务术语详细描述该业务活动的相关业务信息,用做业务人员实现个性化配置.

4) 业务人员可执行

业务积木作为连接业务人员和 IT 资源(如 Web 服务)的桥梁,独立于具体 IT 资源但可与其进行绑定. 业务积木的实现视图记录了可完成该业务活动的 IT 资源信息,用户根据所要完成的业务活动选择相应积木并作配置后即可直接执行.

5) 业务人员可组合

业务人员可根据完成某一任务所要完成的活动步骤选取并组织相应积木得到业务流程.

探索式组合的特点是程序的构造过程是一个不断探索的过程. 我们借鉴网页链接的思想,使能用户以类似网页间跳转的方式构造应用. 业务积木连接视图 $linkView$ 负责定义用户探索式构造应用时该业务积木与其下步可能执行积木间的连接关系. 然而,与网页超链记录下一步网页地址及其数据关联不同,积木间还需表达顺序、分支、合并等控制逻辑. 连接视图由标记下一步执行积木的应用超链集合和超链间控制关系的逻辑表达式集合组成,可定义为

定义 2. 应用超链. 连接视图定义业务积木预置的后续连接关系,可以表示为一个二元组:

$$linkView = (AppHyperLink, ControlLogic),$$

其中,$AppHyperLink$ 为应用超链的集合 $\{ahLink_1, ahLink_2, \cdots, ahLink_n\}$;$ControlLogic$ 为超链间控制关系的逻辑表达式集合. 应用超链可表示为

$$ahLink = (linkID, name, sourceLegoID, targetLegoID, DataLink),$$

其中,$linkID$ 表示该应用超链的惟一标识;$name$ 为超链的名称;$sourceLegoID$ 为该超链所在积木(前置积木)的标识;$targetLegoID$ 为目标积木(后置积木)的标识;$DataLink$ 为前置积木与后置积木之间的数据关联关系的集合,集合中的每个元素都是一个二元组,由前置积木的某个输出参数和后置积木的某个输入参数组成.

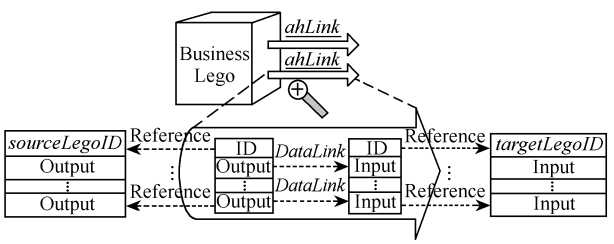


Fig. 4 Illustration of application hyperlink.

图 4 应用超链示意图

如图 4 所示,每个应用超链内部封装了下步可能执行的业务积木的标识及其与超链所在积木间的数据关联关系. 在技术实现上,业务积木的输入输出参数可通过领域本体进行语义标注,系统可基于本体的精确匹配自动建立超链的前置积木的输出参数和后置积木的输入参数间的数据关联并向用户推荐.

应用超链是全时段可定义的:

1) 在业务积木建模时,领域专家可根据积木执行完毕后通常执行的积木预定义超链,供最终用户在应用构建时进行选择,以降低最终用户应用构建的难度,提高应用构建效率.

2) 在应用构造过程中,如果当前业务积木的应用超链中没有用户期望的连接,他可以自定义应用超链. 自定义超链需要配置超链的基本信息、要连接的业务积木、数据映射关系及连接子类型和条件,第 3 节所述的积木推荐机制可辅助用户自定义超链.

应用超链只能表示业务积木间的简单控制逻辑.为了表示复杂的流程,有必要将一系列后置业务积木组织起来. 逻辑表达式集合 $ControlLogic$ 定义连接之间的逻辑关系:

$$f: 2^{AppHyperLink} \rightarrow ControlLogic.$$

每个业务积木中的连接视图只能表示从一个积木出发到多个目标积木间的控制逻辑关系. 为了表示不同业务积木到同一目标积木间的控制逻辑关系,我们定义了一种特殊的业务积木——Any 元素,其特征视图和实现视图为空. 通过两种连接约束,Any 可以辅助建立积木间控制逻辑,其输入是它的所有前置业务积木的输出参数的集合,其输出参数是所有后置积木的输入参数的集合.

通过应用超链集合和连接控制集合间的映射关系,可以表示应用超链间的不同控制逻辑. 由于篇幅所限,此处不展开论述.

当用户选择某一积木到应用中时,产生相应业务积木实例.

定义 3. 业务积木实例. 业务积木实例是对业务积木具体化并配置的结果. 实例化过程表示为

$$g: BizLego \xrightarrow{op} BLI,$$

其中 $BizLego$ 是服务社区中所有积木组成的集合,BLI 为积木实例集合,op 通过以下操作完成:

1) $initiateLegoInstance()$,系统为积木实例分配惟一标识,并对积木实例的状态进行初始化.

2) $inheritLego(bizLego)$,继承业务积木基本信息和实现视图,得到积木实例相应视图信息.

3) $configLego(bizLego)$,对业务积木特征、能

力和连接视图进行按需配置,得到相应的积木实例视图信息.

为支持根据业务积木已有执行结果再选择下一步要执行的积木,积木实例允许在执行完毕后仍然可配置其连接视图.

2.2 探索式服务组合过程

为了实现前面所述的边构造边执行的特点,我们基于如下思想定义探索式服务组合过程:在探索式服务组合的过程中,用户每执行一个操作就会在当前应用的基础上产生一个新的应用.其形式化定义如下:

定义 4. 链接关系.业务积木实例集合 BLI 上的链接关系 R,对于 $a,b\in BLI$,如果存在从 a 到 b 的超链,则 $(a,b)\in R$.

定义 5. VINCA 应用的探索式组合过程.一个 VINCA 应用的探索式组合过程是一个三元组 (VP,OP,δ),其中:

1) $VP=\{va_i\mid i=1,\cdots,n\}$ 是一个有限集,由此次构造过程中所产生的所有应用构成.每个应用 va 是一个二元组 $(AppBLI,appState)$,其中 $AppBLI$ 是非空业务积木实例的集合,并且该集合中的元素满足:$\forall a\in AppBLI,\exists b\in AppBLI$,使得 $(a,b)\in R$ 或 $(b,a)\in R$;每个应用有确定的状态,以 $appState$ 表示,$appState\in\{\text{ready},\text{executed},\text{terminative}\}$.

2) OP 是操作的集合,共有 3 个元素 $OP=\{\text{add},\text{execute},\text{terminate}\}$,分别代表向应用中增加积木、执行应用和中止构造过程.

3) δ 是一个转换法则:$VP\times OP\rightarrow VP$.$\delta(va_i,op)=va_k$ 意味着操作 op 作用于应用 va_i,产生了新的应用 va_k.其中的转换体现在应用的状态和应用包含的元素两个方面.不同操作的映射法则如下面表 1 和表 2 所示.

Table 1 Transition Rule of Application States Affected by Operations

表 1 各种操作对应用状态的转换法则

Application State	Operation		
	add	execute	terminate
ready	ready	executed	∅
executed	ready	∅	terminative
terminative	∅	∅	∅

Table 2 Change Rule of Business Lego Instances Set Affected by Operations

表 2 各种操作对应用中积木实例集合的改变规则

Instance Set	Operation		
	add	execute	terminate
$AppBLI$	$AppBLI\cup\{newBLI\}$	$AppBLI$	$AppBLI$

其中 $newBLI\in BLI$,∅表示对应操作不能作用于相应状态.

性质 1. add 和 execute 操作循环作用于 VINCA 应用,产生结果仍然是 VINCA 应用.

与普通的应用构造方式不同,上述探索式服务组合过程通过几个基本操作,按转换法则 δ 不断作用于已产生的应用,即可实现 VINCA 应用的边构造边执行.当用户执行 terminate 操作后,此次组合过程终止.该组合过程描述可以类推到基于其他编程元素的应用构造过程,判断其是否具有探索式性质.

以上述定义作为基础,我们用顺序图对探索式构造 VINCA 应用的实现过程进行描述,如图 5 所示.其中服务社区、执行区、推荐区、监控台分别

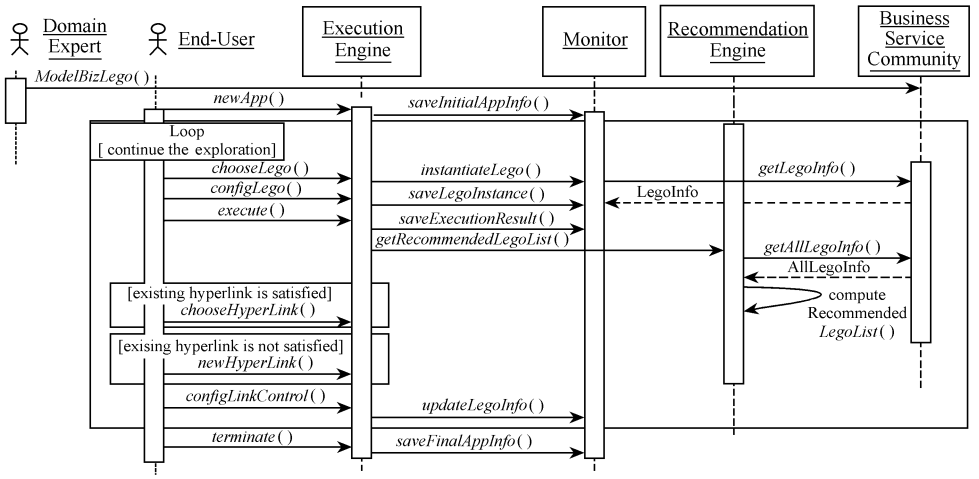


Fig. 5 Process of exploratory application construction.

图 5 探索式应用构造过程

用于业务积木管理、执行、推荐和记录积木实例的执行历史. 在探索式应用构造前需要领域专家对业务积木建模,然后最终用户通过 Loop 循环中的操作原语(其中 *chooseLego*()操作内部包含了 add 操作)实现 VINCA 应用的边构造边执行. 同时,为方便最终用户使用,我们给每个操作原语提供了可视化的表现形式.

3 业务积木的即时推荐机制

如前面所述,使用探索式服务组合方法即时构造应用的过程具有如下两个特点:①需求并非完全事先确定,用户往往根据前一步的执行结果即时地决定下一步做什么;②用户对业务积木的需求以及积木资源集合都会发生变化. 这两个特点增加了用户即时得到所需业务积木的难度. 探索式编程需要用户和系统互动来实现,我们在此从系统支撑角度提出了一种服务即时推荐机制以辅助用户探索. 在构造应用的每一个步骤,自动推断及捕获用户变化的隐式业务需求,并据此对积木集合进行过滤,生成一个满足当前需求的个性化积木空间呈现给用户.

本文以用户的使用习惯作为推荐依据,当应用的构造状态发生变化时,动态确定用户的个性化积木空间.

3.1 个性化的业务积木空间

对于用户已经构造的应用,按流程的结构被调用服务之间在功能上存在着一种转移关系,这种转移关系体现了用户对服务使用顺序的一种习惯. 本文以这种转移关系以作为推荐的依据,根据正在执行的业务积木对候选积木进行过滤,形成当前的个性化业务积木空间. 首先给出如下定义:

定义 6. 路径. 给定一个应用 $va=(AppBLI, appState)$,设 $a,b\in AppBLI$,若 a 和 b 之间存在一个通过超链实例链接形成的积木实例的序列 P,则称 a,b 之间存在一个路径.

定义 7. 路径长度. P 是业务应用 va 中两个积木实例 a 和 b 之间的路径,则路径的长度 $d(a,b)$ 为 P 中超链实例的数目. 若 a 和 b 之间不存在路径,则 $d(a,b)$为无穷大.

定义 8. 转移概率. 转移概率用于量化的表示两个业务积木之间可转移性的大小.

对于任意的 $a,b\in va.\ AppBLI$,s_i 与 s_j 分别是 a,b 所对应的业务积木,则 s_i 与 s_j 的转移概率 $D(s_i,s_j)$按下式计算:$D(s_i,s_j)=\max(1/p)$,p 是 a 与 b 之间的最短路径长度.

定义 9. 用户习惯. 在用户已有的调用过程中两个服务的平均转移概率称为用户习惯. 本文中采用如下方法计算:

$$H(s_1,s_2)=\sum_{i=1}^{n}D_i(s_1,s_2)/n.$$

定义 10. 转移概率矩阵. 转移概率矩阵 $\boldsymbol{T}$ 是一个 $n\times n$ 的矩阵,它的每一行和每一列都代表一个业务积木,$T[i,j]=H(s_i,s_j)$.

以上面定义为基础,个性化的业务积木空间的生成步骤如下:

1) 从用户已构造的应用中计算业务积木之间的转移概率,生成用户习惯,并以转移概率矩阵储存;

2) 捕获当前正在构造的应用中已使用的业务积木;

3) 以步骤 2)所得到业务积木作为条件,在转移概率矩阵中找出具有转移关系的业务积木,形成当前的个性化空间.

由于用户习惯随着使用次数的增多而改变,需要及时调整,这里给出相应的调整算法.

3.2 用户习惯的增量调整

当用户构造的应用个数不断增加时,按定义 9 的方法计算用户习惯将会带来如下问题:所需的计算时间和存储空间不断增加. 为此本文给出了相应的调整算法,采用 Java 语言描述如下:

算法 1. *AdjustDegree*(***DSet***,*Pn*)

输入:***DSet*** 是当前转移概率矩阵,*Pn* 是新的业务应用;

输出:调整后的转移概率矩阵;

步骤:

```
1) NSPSet = getNewSP(Pn); /* 从 Pn 中获取可转移积木对 */
2) OSPSet = DSet.getOldSP(); /* 取得已有的可转移积木对 */
3) for(int i=0; i<NSPSet.size(); i++) {
     /* 将积木对放在变量 tempSP 中 */
     tempSP = NSPSet.get(i);
   /* 如果积木对在交集中 */
4) if (OSPSet.isExist(tempSP)) {
     /* 取出原转移概率 */
     oldValue = DSet.getValue(tempSP);
     /* 取出 Pn 中的转移概率 */
     tempV = Pn.getValue(tempSP);
```

```
        /* 计算新的转移概率 */
        newValue = (oldValue * (n - 1) +
        tempV)/n;
        /* 更新转移概率集合 */
        DSet.update(tempSP,newValue);
        /* 将该积木对从 A 中删除 */
        OSPSet.delete(tempSP);}
      /* 如果在 M - I 中 */
5) else {
        /* 计算 Pn 中的转移概率 */
        tempV = Pn.getValue(tempSP);
        /* 更新转移概率集合中的值 */
        DSet.update(tempSP, tempV/n);
        } } /* 如果 A 中还有剩余元素 */
6) if(!OSPSet.isEmpty()) {
      for(int i = 0;i < OSPSet.size();i ++) {
        /* 取出剩余服务对 */
        tempSP = OSPSet.get(i);
        tempV = DSet.getValue(tempSP);
        /* 更新这些服务对的转移概率 */
        DSet.update(tempSP,tempV * (n -
        1)/n);} }
7) return DSet.
```

其中 M 表示新应用中产生的转移概率不为 0 的积木对的集合,以 A 表示已经存在的具有可转移性的积木对的集合,集合 $I = M \cap A$.

按上面算法可以按增量的方法调整积木对的转移概率,降低了时间和空间复杂度,定量分析不在本文探讨范围内.

4 应用与评价

文中给出的探索式服务组合编程方法和相应支撑环境已经在网络化科研协作项目中得到初步的实践应用. 在文章开始我们已经对该项目的需求特征进行了分析,归纳起来,主要是科研工作者对实验流程存在不确定性以及实验本身所具有的结果非确定性,存在着对‘边执行边构造’探索地构造应用的强烈需求. 下面,结合华大基因科研人员在网络化科研协作项目中的一个具体应用需求为例说明探索式服务组合能够满足该项目的特定需求,并在后面给予定性分析.

4.1 应用实例

下面我们以华大基因研究人员小张完成‘水稻基因重测序’的实验为例,着重以该应用的其中一步探索为例说明科研工作者是如何使用系统进行探索.由于是新员工,小张并不熟悉该实验及各种实验相关应用程序. 通过探索式服务组合平台,小张可通过领域专家事先定义好的业务积木了解实验可用的服务信息. 小张在进行水稻基因组重测序时,他首先设计并确定引物,之后进行基因测序,当用户利用测序仪得到水稻基因的原始峰图文件后,他试着利用 PhredPhrap 服务进行碱基的识别和错误率的估计,然后对该结果进行 SNP 筛选,我们选定这一步(如图 6 所示)示例探索过程.

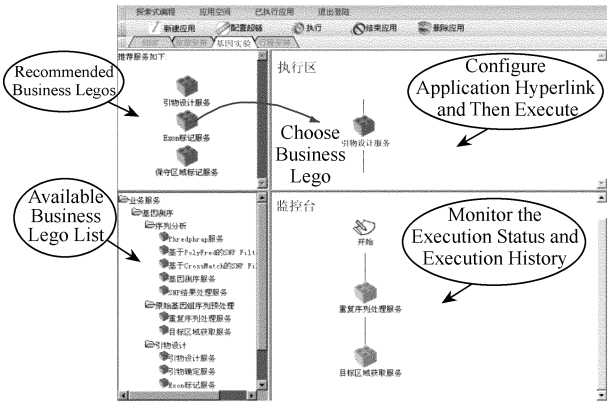


Fig. 6 Snapshot of exploratory service composition tool.

图 6 探索式服务组合探索示例

以上应用实例中的 SNP 筛选过程中存在过程非确定性的问题,在该步,可进行 SNP 筛选的服务有很多,用户需要尝试着选用一个服务并根据结果和自己的专业知识,决定是否需要继续选用其他积木进行尝试,直到得出合适结果. 当“PhredPhrap 服务”执行结束后,小张根据应用的结果选择“PhredPhrap 服务”中领域专家事先定义的应用超链 PhredPhrap2CrossMatchLink 和 PhredPhrap2PolyFredLink,分别和‘基于 CrossMatch 的 SNP 筛选服务’、‘基于 PolyFred 的 SNP 筛选服务’建立关联关系. 当这两个服务执行完毕后,小张发现结果并不理想,则须自定义超链. 前文提到的推荐方法会对业务积木进行标价,会按照满意度由高到低推荐给用户. 因此小张选取推荐指数最高的服务——‘基于 ClustalW 的 SNP 筛选服务’到执行区,此时系统自动建立初始应用超链 PhredPhrap2ClustalWLink,并在‘PhredPhrap’服务的输出参数和‘ClustalW’服务的输入参数间根据语义建立缺省数据关联,用户可以根据需求修改缺省的应用超链的内容. 用户执行完毕‘ClustalW’服务后观察服务结果,判断 SNP 筛选结果是否满意,

如果不满意可以选用推荐区的其他服务重新进行SNP筛选.每一步探索结束后,最终用户都可以继续下一步探索,或者结束应用(此时用户认为已得到期望的结果).

4.2 相关工作分析

本节从第1节所述探索式服务组合方法试图解决的3个问题角度对文中提出的方法与相关工作进行比较,以论述本文方法如何支持最终用户进行探索式服务组合.

1)由IT人员主导应用构造转变为业务用户主导应用构造

以往应用构造主要面向IT专业人员,IT人员对业务需求进行分析,并在此基础上进行编码开发.IT人员对业务需求的理解、分析会花费较长的时间,并且需要与业务人员多次交流,使应用构造的总时间较长."面向最终用户"的服务组合这种提法已经初露端倪:Akkiraju等人[6]提出从业务专家的角度定义抽象业务过程,但业务专家仍然需要具备领域本体构建的知识.在科研领域,为让用户从底层编码中解脱,更专注于过程的设计,提出了SCUFL语言[7]和Triana系统[8].但它们都要求使用者在开发工作流程前自己查找Web服务并理解其含义,对使用者的要求较高.

本文提出的方法可以使业务用户进行应用构建时不需要了解每个Web服务的具体信息,仅需要操作Web服务集合虚拟化后的实体-业务积木,并且通过与编程环境的交互可以以半自动的方式确定不同业务积木间的关系,这种方式有效简化了用户定义流程的复杂度.

2)提供需求不明确的情况下构建业务流程的方法

目前无论是在工作流领域还是在面向服务计算领域,大多数的应用系统只适用于构造需求明确、业务流程可预先定义的情况.在工作流领域,许多主流工作流系统都是先构建流程然后执行.流程执行过程中,一旦出现问题就需要重新构建再执行,这种方式效率很低.为了改善这种状况,出现了一些支持运行时修改的工作流系统.例如,动态工作流支持用户运行时根据需求变化动态修改业务流程[9],一般用于处理异常或局部需求变化的情况.在面向服务组合领域,Sirin等人[10]提出一种半自动化的交互式组合方法.该方法基于语义根据用户给定的约束信息及问题域进行自动筛选服务.它与探索式服务组合方法的相同之处是都是交互式组合方式,系统辅助用户选取服务但用户对最终服务的选取有决定权.不同之处在于该方法的出发点是基于目标来组合服务,需求必须确定.而在许多情况下需求并不明确,无法事先定义业务流程.

探索式服务组合方法为解决这类问题提供了一种解决方法.从第2节对探索式编程模型的分析可知,我们的探索式服务组合方法支持应用中当前积木执行完毕后继续选择业务积木.这种"边构造边执行"的应用构建方法与动态工作流相关工作的不同在于用户可以在当前应用执行完毕后仍然继续对应用进行修改,只有当用户完成应用探索选择结束应用时才进入应用结束状态.

3)服务即时推荐方法

在传统的基于内容的推荐方法中,推荐过程实际是计算候选条目相对于用户偏爱的符合程度.在语境敏感的服务选取和推送方面,Christos等人提出了一种基于语境信息的树形的服务组织模型[11];Maamar等人通过Agent和服务语境降低发现和组合服务的复杂性[12]等.

与上述工作相比,本文所提出的即时推荐方法具有如下特点:面对的操作对象是业务积木,并针对个人用户的使用习惯推荐,在开放的服务网格环境中来考虑如何推荐服务.

5 结束语

针对面向服务计算环境下用户构造应用的需求不确定性问题,本文提出了一种探索式服务组合编程方法.这种方法不同于传统编程中事先确定逻辑的模式,而是通过一种类似文本超链接的形式组合服务,支持最终用户以一种"边执行边构造"的方式构造应用.编程模型中只有一种基本元素——业务积木,业务积木之间的关系可通过引入的应用超链进行有效链接.从系统支撑角度,本文提出了一种服务即时推荐机制方便用户快速有效地找到自己所需要的服务.在生物信息领域的实践表明,文中倡导的理念和方法有相当的实用价值,有望推动一种新的面向服务的应用构造模式的发展.未来的工作中将在更大的范围内应用,同时将结合实际应用进一步研究编程模型的表达能力和易用性、智能流程挖掘和组合结果的正确性保障等,不断完善这一探索式服务组合的方法.

参 考 文 献

[1] F Casati, *et al*. Adaptive and dynamic service composition in eFlow [C]. The 12th Int'l Conf on Advanced Information Systems Engineering, Stockholm, Sweden, 2000

[2] B Benatallah, *et al*. The self-service environment for Web services composition [J]. IEEE Internet Computing, 2003, 7(1): 40-48

[3] Y Han, *et al*. VINCA—A visual and personalized business-level composition language for chaining Web-based services [C]. The 1st Int'l Conf on Service-Oriented Computing, Berlin, 2003

[4] Fang Jun, *et al*. A service virtualization mechanism for business end programming [J]. Chinese Journal of Computers, 2005, 28(4): 549-557 (in Chinese)
(房俊, 等. 一种支持业务端编程的服务虚拟化机制 VINCA-VM [J]. 计算机学报, 2005, 28(4): 549-557)

[5] Wang Jianwu, *et al*. On domain-specific modeling of business-level services [R]. Research Center for Grid and Service Computing, Institute of Computing Technology, Chinese Academy of Sciences, Tech Rep: VINCA-06-TR-06, 2006 (in Chinese)
(王建武, 等. 面向领域的业务服务建模方法研究[R]. 中国科学院计算技术研究所网格与服务计算研究中心, 技术报告: VINCA-06-TR-06, 2006)

[6] R Akkiraju, *et al*. Executing abstract Web process flows [C]. The 14th Int'l Conf on Automated Planning and Scheduling, Whistler, British Columbia, Canada, 2004

[7] T Oinn, *et al*. Taverna: A tool for the composition and enactment of bioinformatics workflows [J]. Bioinformatics, 2004, 20(17): 3045-3054

[8] M Burnett, *et al*. FAR: An end-user language to support cottage E-services [C]. The 1st IEEE Symp on Human-Centric Computing Languages and Environments, Stresa, Italy, 2001

[9] M Reichert, P Dadam. A framework for dynamic changes in workflow management systems [C]. The 8th Int'l Workshop on Database and Expert Systems Applications, Toulouse, France, 1997

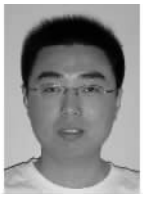
[10] E Sirin, *et al*. Filtering and selecting semantic Web services with interactive composition techniques [J]. IEEE Intelligent Systems, 2004, 19(4): 42-49

[11] D Christos, V Michalis. Querying and updating a context-aware service directory in mobile environments [C]. The 2004 IEEE/WIC/ACM Int'l Conf on Web Intelligence (WI '04), Beijing, 2004

[12] Z Maamar, *et al*. Toward an agent-based and context-oriented approach for Web services composition [J]. IEEE Trans on Knowledge and Data Engineering, 2005, 17(5): 686-697

Han Yanbo, born in 1962. Professor and Ph. D. supervisor, senior member of China Computer Federation. His main research interests include software integration and service grid.
韩燕波, 1962 年生, 研究员, 博士生导师, 中国计算机学会高级会员, 主要研究方向为软件集成与服务网格.

Wang Hongcui, born in 1980. Master. Her main research interests include software integration and service grid.
王洪翠, 1980 年生, 硕士, 主要研究方向为软件集成与服务网格.



Wang Jianwu, born in 1980. Ph. D. candidate. His main research interests include software integration and service grid.
王建武, 1980 年生, 博士研究生, 主要研究方向为软件集成与服务网格.

Yan Shuying, born in 1979. Ph. D. candidate. Her main research interests include software integration and service grid.
闫淑英, 1979 年生, 博士研究生, 主要研究方向为软件集成与服务网格.

Zhang Cheng, born in 1977. Ph. D. His main research interests include dynamic services composition and context-aware computing.
张程, 1977 年生, 博士, 主要研究方向为动态服务组合、语境敏感计算技术.

Research Background

This reported research work investigates appropriate ways for scientists to assemble available and best-suited networked services in their scientific explorations. Current service composition technologies are mostly developed for software professionals and for the situations where business requirements are definite and business processes can be predefined. We hereby explore end-user-doable and exploratory ways in composing business-level services, propose a novel approach to enabling on-the-fly composition, and try out the approach in real settings.