ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, Supplement, November 2006, pp.211−218

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

普适计算环境下一种目标驱动的服务组合方法*

张抗抗[1,2+], 李庆忠[1]

[1](山东大学 计算机科学与技术学院,山东 济南 250061)

[2](山东财政学院 计算机信息工程学院,山东 济南 250014)

A Goal-Driven Approach of Service Composition for Pervasive Computing

ZHANG Kang-Kang[1,2+], LI Qing-Zhong[1]

[1](School of Computer Science and Technology, Shandong University, Ji'nan 250061, China)

[2](School of Computer and Information Engineering, Shandong University of Finance, Ji'nan 250014, China)

+ Corresponding author: Phn: +86-531-88395947, E-mail: lqz@sdu.edu.cn, http://www.sdu.edu.cn

Zhang KK, Li QZ. A goal-driven approach of service composition for pervasive computing. *Journal of Software*, 2006,17(Suppl.):211−218. http://www.jos.org.cn/1000-9825/17/s211.htm

Abstract: The pervasive computing environment is transparent for users. In order to simplify the interaction between users and the environment, this paper presents a goal-driven approach of service composition. A task-oriented semantic representation model of Web services is built and based on this model goal-driven service composition is performed dynamically to achieve user's goal. A simulated application scenario is described and the process of task definition and service composition according to the scenario are discussed in this paper. This approach simplifies the interaction between users and pervasive computing environment and improves the flexibility of service cooperation and composition in the environment.

Key words: pervasive computing; service composition; goal-driven; ontology

摘 要: 普适计算环境的服务对用户来说是透明的,为了简化用户与环境的交互,提出了一种目标驱动的服务组合方法,建立面向任务目标的服务语义表示模型,并在此模型下进行动态的目标驱动的服务组合,实现用户目标.描述了一个模拟的应用场景,并结合应用场景说明了任务定义和服务组合的过程.该方法简化了用户与环境的交互,提高了普适环境下服务组合和协同的灵活性.

关键词: 普适计算;服务组合;目标驱动;本体

普适计算[1]环境是以人为中心的计算和通信环境,包含了众多的设备(如传感器、计算设备、人机接口设备等)和服务(如 Web Service),以满足人们多样化的应用目标任务.从普适计算的角度来讲,环境中的设备和服务对用户来说是透明的,用户不需要关心哪些服务为其提供服务以及如何提供,因此,需要提供一种简单而人性化的交互方式,使得普适计算环境能够通过上下文感知获取用户的意图,并进而协调各种服务完成用户目标.

由于面向服务计算(SOC)[2]的概念在普适计算中越来越体现出其重要性,因此目前很多研究[3−5]开始关注如何组织和协调服务特别是 Web Service 来满足用户的各种需求,形成一种面向服务的普适计算环境[6].文献[3]提出了一种本体支持的面向服务的普适计算系统结构 OSOA,其中也提到了使用目标驱动的方式实现服务的

* Received 2006-03-30; Accepted 2006-10-08

组合;文献[4]使用业务过程和工作流建立人与普适计算环境的交互模型,并以此协同多个服务,实现用户目标;文献[5]提出了一种基于策略的实现普适计算环境的自适应服务体系结构,将服务增加了语义描述、状态视图、行为模型和对外调用模型后形成一个自适应服务元件(adaptive service element,简称 ASE),该元件由于具有服务、状态及行为等相当丰富的语义信息,因而具有一定的行为自主性.

本文提出了一种目标驱动的服务组合方法,首先建立面向任务目标的服务语义表示模型,在较高的抽象层次上对应用场景和任务目标进行定义,并在此模型下进行动态的服务发现和匹配,实行目标驱动的服务组合,实现用户目标.该方法简化了用户与环境的交互,提高了普适环境下服务组合和协同的灵活性.

本文首先描述一个模拟的应用场景,然后建立面向任务目标的服务语义表示模型,并在该模型下探讨目标驱动的服务组合方法,并结合应用场景分析该方法的执行效果,最后给出本文的结论.

1 应用场景

为了更好地说明本文所讨论的方法,首先我们来描述一个模拟的应用场景.

假定一个用户进入一个具有普适计算环境的音像图书零售商店,该环境中运行着若干服务分别执行特定的功能,如登陆服务可以使环境记录进入商店的客人身份,布局服务可以为客人提供路径指引,不同的商品服务可以为客人提供不同种类商品的详细信息,购物车服务可以记录客人选择的商品,付款服务使客人可以为其选择的商品付款,并可以选择付款方式,等等.

假设用户进入这样一个普适计算环境,他随身携带着一部手持设备,该手持设备上也运行着若干服务(比如记录用户个人信息的服务以及购物意愿的服务等),并且能够与环境服务进行便捷的通信.一旦用户进入商店,商店普适环境就会通过他的手持设备获取该用户的信息,登陆该用户,同时在手持设备上显示一个页面,在此,用户可以指明其在该商店的任务目标.假设用户想要购买一本有关普适计算的书,当他通过手持设备确认该任务请求后,手持设备上会呈现出为了实现该任务目标所需要执行的一系列子任务.这些子任务可能如下所示:

(1) 寻找到陈列有计算机类别图书的区域;
(2) 浏览书籍,选择适合自己需要的;
(3) 选中书籍,并将其加入购物车;
(4) 选择一个空闲的收款台;
(5) 选择相应方式付款;
(6) 指定图书包装和递送服务选项.

以上列表是从任务本质出发对任务进行的分解.当用户在手持设备上按照逻辑顺序一步步执行时,普适计算环境就会动态地发现和协调各种必要的服务,为用户提供路径指引、购物车以及付款等一系列服务,直至达成用户买书的最终目标.

2 服务组合的基础

为了实现服务的动态匹配和组合,简化用户与普适计算环境的交互方式,我们建立面向任务目标的服务表示模型,引入丰富的语义,对环境资源、服务以及任务目标进行描述和定义.这种定义的目标是首先实现在抽象层次上的服务组合描述,然后在此基础上通过动态的选择和绑定服务产生一个可执行的服务流程.

本节首先对一般的服务组合过程中一些基本问题进行简单的分析,然后探讨本文想要建立的面向目标的服务表示模型.

2.1 一般服务组合过程要点分析

服务组合过程的生命周期可以分为 5 个阶段:定义阶段、编排阶段、构造阶段、执行阶段和演化阶段[7],其中涉及服务组合和流程生成的是前 4 个阶段.另外,文献[7]中还在这样一个阶段理论下探讨了当前流行的 Web Service 流程语言所包含的组合元素,并分析了业务规则在服务组合过程中的控制和指导作用.但文献[7]提

出的方案中,服务组合过程开始后还需要过多的人为参与,自动化程度受到影响.另外,仅使用业务规则也很难为服务组合过程提供足够的语义支持.

因此,本文将编排阶段的部分有关业务执行模式方面的工作提前到定义阶段进行,其余工作则放到构造阶段,这样,本文中,服务组合的生命周期将考虑定义、构造和执行 3 个阶段.定义阶段是对任务目标进行抽象定义,获得一个任务过程的执行框架,它与具体的服务组合的区别在于,在抽象定义中,实际的服务及其提供者和执行控制信息不进行具体指定;构造阶段则根据该框架,分解任务,匹配和绑定服务,生成一个可执行的服务流程;执行阶段实现对实际服务的调用,执行流程,完成用户目标任务.

在这样一个框架下,我们结合本体的语义表达能力和业务规则的流程控制和约束表示能力,形成一种更高层次上的对任务目标的抽象定义,进而形成一个层次结构的面向任务的服务表示模型[8].下面一节我们再对模型作一个简单的介绍,关于模型的详细信息可以参考文献[8].

2.2 面向任务目标的服务表示模型

面向任务目标的服务表示模型包括 5 个层次,并通过一组客户调用接口对外发布.这 5 个层次是:任务表示层、服务表示层、服务接口层、消息层和传输层,其中上面 3 个层次为语义层,通过一个应用领域的资源语义模型提供语义支持.如图 1 所示.

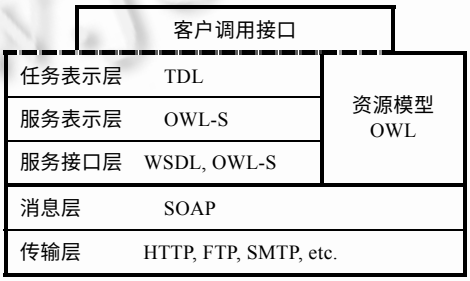

Fig.1 Task goal-oriented service representation model protocol stack

图 1 面向任务目标的服务表示模型协议栈

对于用户任务目标的抽象定义在任务表示层进行.任务表示层由任务定义语言 TDL 组成,TDL 语言结合本体和业务规则的优势,在比较高的抽象层次上定义了完成某项任务的标准流程,描述了完成该任务所需活动的计划草案.其中,使用本体提供了一套统一的描述术语,用于描述抽象的概念名称和任务目标计划草案,各种不同类型的业务规则用来提供对流程控制、数据和状态约束、资源以及异常处理等相关信息的描述.

目标任务定义采用如下格式:

〈*Task*〉::=〈*TaskId*〉〈*TaskName*〉〈*TaskInfo*〉〈*TaskStruct*〉〈*TaskRule*〉〈*TaskNFP*〉,

其中:

(1) 〈*TaskId*〉和〈*TaskName*〉分别为任务标识符和任务名称;

(2) 〈*TaskInfo*〉是任务的描述信息,包括任务功能描述以及输入、输出,〈*TaskInfo*〉::=〈*FuncDes*〉〈*IN*〉〈*OUT*〉;

(3) 〈*TaskStruct*〉是任务结构,指定任务中包含的活动以及它们的逻辑关系,〈*TaskStruct*〉::=〈*Construct*〉〈*ActSet*〉,构建符有 4 种:*Sequence*,*Unordered*,*Choice* 和 *Iteration*,原则上,所有的过程都可以用这 4 种逻辑表示[9],〈*ActSet*〉是参与任务的活动的集合,每一个活动可以是基本的,也可以是一个已经定义的任务;

(4) 〈*TaskRule*〉是任务规则集,指定任务相关的数据、状态约束、资源及异常处理等,〈*TaskRule*〉::={〈*RuleType*〉〈*RuleSet*〉},〈*RuleType*〉为规则类型,其取值为以下 4 种类型之一:{*Data*,*Constraint*,*Resource*,*Exception*},〈*RuleSet*〉为该类型下规则的集合,规则形式采用 ECA 形式,规则的具体描述可以采用现有的规则语言,如 SWRL[10]来描述;

(5) 〈*TaskNFP*〉为任务的其他非功能描述信息.

3 目标驱动的服务组合

本节将结合第 1 节描述的场景讨论服务组合过程,分别从上文所说的服务组合生命周期的定义、构造和执行 3 个阶段详细论述.图 2 所示为组合过程的系统结构,为了便于说明和理解,图中仍然以上文所述的场景为例.

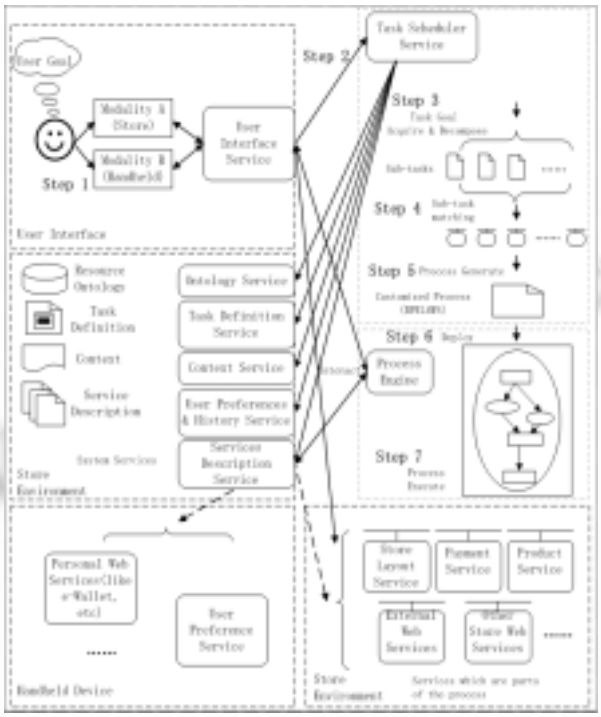

Fig.2 System structure of composition process
图 2 组合过程系统结构

3.1 定义阶段

从服务组合生命周期考虑,定义阶段主要体现在服务表示模型的最上层,即任务表示层.通过使用 TDL 语言对任务目标进行抽象定义.但是,对于一个完整的系统来说,定义阶段所包括的内容远不止这些,还包括环境资源本体的建立、环境服务的注册和语义描述等.除了环境内的资源和服务以外,普适计算面临的另外一个问题是用户携带的手持设备上也运行着一些服务,这些服务也要参与用户与环境的交互过程,并且还有可能成为最后产生的业务流程的一部分,例如电子钱包服务就是一个非常典型的例子.一种解决的方法[4]是,在环境中建立一组代理服务模块,作为环境系统服务的一部分,普适环境生成流程时,如果需要用户手持设备服务,则将使用这些代理服务进行绑定,代理服务在执行时再跟手持设备服务进行绑定和通信.

由于本文重点讨论服务组合,因此,在此也着重描述任务目标的定义.上文场景中描述的任务目标经过抽象定义后可以形成以下任务模板(因篇幅所限,没有进行完整定义,仅列出了一部分定义,见表 1).

表中定义了一个目标任务 BuyBook,它由 *FindBookShelf*,*SelectBook*,*FindCashier*,*Pay*,*Pack&Deliver* 几个活动组成,每个子活动都分别进行了相应的定义.对基本活动则不再进行进一步的定义,它们都可以在环境中找到相应的服务提供其功能,如表中的 *Locate*,*Route*,*Pay* 等.

为了使服务表示模型的协议栈向下兼容,TDL 语言也使用了 XML 语法表示,因此,表 1 所示的任务定义在实际系统中即可以转化为 XML 格式表示.

Table 1 Example of task definition

表 1 任务定义示例

TaskId: *Task*01; *TaskName*: *BuyBook*; *TaskInfo*: *FuncDes*: Look for and buy a book on Pervasive Computing and make it delivered to his place; *IN*: (*Book Classification*); *Out*: (*Confirmation of Success*); *TaskStruct*: *Construct*: *Sequence*; *ActSet*: (*FindBookShelf*, *SelectBook*, *FindCashier*, *Pay*, *Pack&Deliver*); *TaskRule*: ; *TaskNFP*: ;
TaskId: *Task*02; *TaskName*: *FindBookShelf*; *TaskInfo*: *FuncDes*: Find the book shelf where the books that the user wanted are displayed and guide the user there; *IN*: (*Book Classification*); *Out*: ( ); *TaskStruct*: *Construct*: *Sequence*; *ActSet*: (*Locate*, *Route*); *TaskRule*: ; *TaskNFP*: ;
.....
TaskId: *Task*04; *TaskName*: *FindCashier*; *TaskInfo*: *FuncDes*: Find an available cashier and guide the user there; *IN*: ( ); *Out*: ( ); *TaskStruct*: *Construct*: *Sequence*; *ActSet*: ((*Construct*: *Unordered*; *ActSet*(*Locate*, *FindAvailableCashier*),) *Route*); *TaskRule*: ; *TaskNFP*: ;
TaskId: *Task*05; *TaskName*: *Pay*; *TaskInfo*: *FuncDes*: Select the way of payment and pay for the book; *IN*: ( ); *Out*: (*Confirmation of payment success*); *TaskStruct*: *Construct*: *Sequence*; *ActSet*: (*SelectPaymentWay*, *Pay*); *TaskRule*: (*RuleType*: *Exception*; *RuleSet*: (if *paymentway*="*CreditCard*" & *state*="*CreditCardUnavailableException*" then *paymentway*="*Cash*")); *TaskNFP*: ;
.....

3.2 构造阶段

经过资源建模、服务描述注册以及任务定义后,环境系统即可以进入工作状态,接受用户的服务请求.用户与环境的接口是多样化的,既可以通过用户手持设备输入请求,也可以通过环境的上下文感知获取用户请求(Step 1).用户请求首先被提交给任务协调器(task scheduler)(Step 2),该模块是任务执行流程构造的中心协调模块.任务协调器获取用户请求后的首要工作是推理用户意图,明确任务目标,系统环境中的相应服务会为其提供资源本体访问、任务定义、上下文以及用户偏好和历史等服务,这个过程也是将任务目标分解为基本子任务的过程(Step 3).

下一步的工作是将分解后的子任务目标进行服务匹配(Step 4),根据子任务定义的功能描述、输入、输出等语义信息进行服务的语义匹配[11],发现相应的能够完成各子任务目标的服务,实现任务目标与具体的服务的绑定.通过 Step 4,我们获得了完成各子任务目标所需服务的 OWL-S 描述,构造阶段的最后一项工作是将发现的具体服务按照任务定义的流程模板组合起来,这样就可以构造出完成整个任务目标的复合服务的 OWL-S 描述,

然后将该复合服务使用某种流程表示语言(如 BPEL4WS)来表示,形成一个可以执行的流程(Step 5).

由此可以看出,整个流程构造阶段实际上形成了一个从 TDL 到 OWL-S 再到 BPEL4WS 的映射,这种映射的基本核心是其复杂的控制逻辑的映射.表 2 列出了 TDL Construct,OWL-S Construct 以及 BPEL4WS 结构化活动之间的映射关系.

Table 2 Mapping among TDL, OWL-S constructs and BPEL4WS structured activities

表 2 TDL Construct,OWL-S Construct 以及 BPEL4WS 结构化活动映射关系

WS-BDL construct	OWL-S construct	BPEL4WS structured activity	
Sequence	Sequence	Sequence	
Unordered	Split	Flow	
	Split+join		
	Unordered	Sequence	
Choice	Choice	Pick	Switch
	If-Then-Else		
Iteration	Iterate	While	
	Repeat-While		
	Repeat-Until		

为了保证映射的唯一性,我们制定如表 3 所示的映射规则.

Table 3 Mapping rules

表 3 映射规则

From TDL to OWL-S	From OWL-S to BPEL4WS
1) WS-BDL 中的 Sequence 映射到 OWL-S 中的 Sequence;	1) OWL-S 中的 Sequence 映射到 BPEL4WS 中的 Sequence;
2) WS-BDL 中的 Unordered 映射到 OWL-S 中的 Split+join;	2) OWL-S 中的 Split 和 Split+join 映射到 BPEL4WS 中的 Flow;
3) WS-BDL 中的 Choice,如果包含一个 φ?活动(条件测试活动),则映射到 If-then-else;否则,映射到 Choice;	3) OWL-S 中的 Unordered 映射到 BPEL4WS 中的 Sequence;
4) WS-BDL 中的 Iteration,如果包含一个 φ?活动,则映射到 Repeat-while;否则,映射到 Iterate.	4) OWL-S 中的 Choice(1.0 及以前版本)映射到 BPEL4WS 中的 Pick;OWL-S 中的 Choice(1.1 版本)和 If-Then-Else 映射到 BPEL4WS 中的 Switch;*
	5) OWL-S 中的 Iterate,Repeat-While 和 Repeat-Until 映射到 BPEL4WS 中的 While.

3.3 执行阶段

获得了定制的可执行流程之后,任务协调器将部署该流程脚本(Step 6),并通过一个流程引擎解释执行(Step 7),通过服务绑定阶段所获得的服务调用信息实现对具体服务的顺序调用.

4 模拟实验验证

为了验证本文所提出方法的效率,我们利用上文所说的应用场景对本文方法进行了模拟验证.因受实验条件限制,并且实验的目的主要是检验方法的可行性,因此,我们在实验中使用两个应用程序 App1 和 App2 分别模拟普适环境和手持设备,App1 中设置了一个 Store Layout Service、一个 Product Service、多个 Payment Service 以及一个模拟的书店布局图,App2 中设置了一些参数调用来表示不同类型图书的位置.从多次模拟实验的效果来看,App1 总是可以根据 App2 的调用参数的不同,选择距离当前位置最合理的 Payment Service 完成整个购书过程.

* Choice 构建符在较早版本的 OWL-S 规范中可以选择多个分支执行,考虑到实际应用中无须如此复杂,在最新的 OWL-S 1.1 版本中进行了简化,仅选择了其中一个分支.

5 讨论和相关工作比较

由于 Web Service 的语法一致性以及松散耦合的特性,面向服务的计算(SOA)已经在多个应用领域显示了其优越性.但是由于语义信息的缺失,服务之间的交互效率受到了影响,Web Service 的松散耦合的特性也在一定程度上受到了损害.本文提出的目标驱动的服务组合方法结合了我们提出的面向业务的语义服务表示模型,将业务本身的定义抽象出来,独立于具体的服务,因而增强了业务执行的灵活性.同时,语义信息的丰富提高了系统交互的效率,使得 SOA 体系的优越性得以更加充分的发挥.

目前,基于 SOA 体系的普适计算研究也得到了很多研究者的关注.文献[3]提出了一种本体支持的面向服务的普适计算系统结构 OSOA,其中也提到了使用目标驱动的方式实现服务的即时(ad-hoc)组合,但该文仅仅提出了一个基本框架,没有对目标定义和服务组合方法进行深入的探讨,还没有形成比较系统而成熟的方法;文献[4]中使用业务过程和工作流建立人与普适计算环境的交互模型,使用 BPEL4WS 语言描述工作流,协同多个服务,实现用户目标.该方案通过工作流定义将复杂任务进行了分解,具有一定的灵活性和可伸缩性,但是,所有的服务都在工作流定义阶段进行了静态的绑定,极大地降低了服务选择的灵活性和业务处理过程的动态性;文献[5]的方案借鉴了许多 Agent 的思想,使得每个服务结点在增加了对服务、状态及行为的语义描述后成为一个自适应服务元件 ASE,具有一定的行为自主性,但是,在这种模式下,ASE 要进行相当复杂的设计,并且该方案过分强调了服务结点的自主动作,因而缺乏针对任务目标的全局的协调和控制.

6 结论与展望

普适计算环境中服务对用户来说是透明的,用户与环境的交互应该以一种更加简单和人性化的方式进行.为了实现这个目标,本文提出了一种目标驱动的服务组合方法,以帮助用户完成他们的任务目标.我们建立了一个语义服务表示模型,对用户的任务目标进行抽象定义,完成任务相关的具体服务在执行阶段进行动态地绑定,形成一个可执行的业务流程.环境的灵活性得到了较大的提高,对用户的透明性也得到了增强.

本文对普适环境下的服务组合问题的探讨还仅仅是初步的,构造阶段的许多问题还需要得到继续的关注,如事务问题、服务组合过程中的数据转换问题等.为了简单起见,本文中的应用场景没有考虑服务组合过程中的事务问题.但是,对于一个实用的系统来说,事务问题是无法回避的.另外,服务组合以后产生的业务流程要想正常地执行,也需要考虑数据流的畅通,其中必然要对服务之间的数据传递实现从概念级别上的一致性到数据格式的一致性的转化.今后,我们将继续关注从任务抽象定义产生可执行流程过程中的相关问题.

References:

[1] Weiser M. The computer of the twenty-first century. Scientific American, 1991,94−100.

[2] Huhns MN, Sing MP. Service-Oriented computing: Key concepts and principles. IEEE Internet Computing, 2005,9(1):75−81.

[3] Ni Q, Sloman M. An ontology-enabled service oriented architecture for pervasive computing. In: Proc. of the Int'l Conf. on Information Technology: Coding and Computing (ITCC 2005). 2005. 797−798.

[4] Ranganathan A, McFaddin S. Using workflows to coordinate Web services in pervasive computing environments. In: Proc. of the IEEE Int'l Conf. on Web Services. 2004. 288−295.

[5] Lewis D, O'Donnell T, Feeney K, Brady A, Wade V. Managing user-centric adaptive services for pervasive computing. In: Proc. of the Int'l Conf. on Autonomic Computing. 2004. 248−255.

[6] Bellur U, Narendra NC. Towards service orientation in pervasive computing systems. In: Proc. of the Int'l Conf. on Information Technology: Coding and Computing (ITCC 2005). 2005. 289−295.

[7] Orriēns B, Yang J, Papazoglou MP. A framework for business rule driven service composition. In: Technologies for E-Services. LNCS 2819, 2003. 14−27.

[8] Zhang KK, Li QZ. A task-oriented semantic representation model of Web services process integration. In: Proc. of the 10th Int'l Conf. on Computer Supported Cooperative Work in Design (IEEE Cat. No. 06EX1292). 2006. 724−727.

[9] Aalst V, Hee K. Workflow Management Models, Methods, and Systems. Cambridge, London: MIT Press, 2002.

[10] Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M. SWRL: A semantic web rule language combining owl and RuleML. 2003. http://www.daml.org/2003/11/swrl/

[11] Paolucc M, Kawamura T, Payne TR, Sycara K. Semantic matching of web services capabilities. In: Proc. of the 1st Int'l Semantic Web Conf. (2002). LNCS 2342, 2002. 333−347.

张抗抗(1974－),男,山东滕州人,博士生,主要研究领域为企业应用集成与信息集成.

李庆忠(1965－),男,博士,教授,博士生导师,主要研究领域为应用集成与信息集成,信息网格,数据仓库,数据库技术.