第 33 卷　第 20 期　　计　算　机　工　程　　2007 年 10 月
Vol.33　No.20　　Computer Engineering　　October 2007
·博士论文·　　文章编号：1000—3428(2007)20—0046—03　　文献标识码：A　　中图分类号：TP391

基于虚拟组织模型与 Web 服务的 MAS 平台

张树东[1]，廖乐健[2]，吕慧颖[3]
(1. 中国科学院软件所，北京 100080；2. 北京理工大学计算机学院，北京 100081；3. 首都师范大学信息工程学院，北京 100037)

摘　要：虚拟组织是电子商务研究中所提出的智能体协同模型，Web 服务则提供了实现互联网上异构智能体间互操作与协同的技术基础。该文提出了一个基于虚拟组织模型与 Web 服务技术的 MAS 平台系统，介绍了系统的体系结构及主要模块的实现。该平台实现了基于 UDDI 的 Agent 注册管理和基于 SOAP 的消息传递机制，提供了系统管理和监控等功能。

关键词：Web 服务；多智能体；虚拟组织；语义 Web

Framework for MAS Based on Virtual Organization Model and Web Services Technology

ZHANG Shu-dong[1], LIAO Le-jian[2], LV Hui-ying[3]
(1. Institute of Software, Chinese Academy of Sciences, Beijing 100080; 2. College of Computer, Beijing Institute of Technology, Beijing 100081; 3. College of Information Engineering, Capital Normal University, Beijing 100037)

【Abstract】Virtual organization model is an agent cooperation model presented in the research of e-business. Web services technology provides a mechanism to support cooperation and collaboration of multi agents. A framework for multi-Agent system management and communication is presented, called In-Agent, which is based on virtual organization model and Web services technology. In-Agent uses UDDI as its register center to manage agent register information, and uses SOAP as its communication tool to transfer information between agents. In-Agent also provides toolkits for system management and monitoring. With In-Agent, developing and deploying a multi-Agent system is more convenient.

【Key words】Web services; multi-Agent; virtual organization; semantic Web

Agent 是一个基于硬件和软件的计算机系统，具有自主性、社交性、反应性、能动性等性质，是分布式人工智能的一个重要研究领域。虚拟组织是随着电子商务与互联网技术的发展而提出来的一种面向跨组织协作的分布式管理与建模的抽象概念模型。

Web Services 是 W3C 提出的一种软件体系结构，其目标是在现有的各种异构平台的基础上构筑一个通用的与平台无关的、语言无关的技术层，各种不同平台之上的应用依靠这个技术层来实施彼此的连接与集成。

本文提出并实现了基于虚拟组织模型和 Web Services 技术的 Agent 管理通信平台——In-Agent 系统。该平台在 UDDI 的基础上扩展实现了 Agent 的注册管理；提供 Agent 的动态部署、监控等基本管理功能；支持透明的通信服务，提供对 ACL 和 KQML 两种 Agent 通信语言的支持。

1　系统体系结构

In-Agent 系统体系结构如图 1 所示，In-Agent 系统由分布式异构执行环境、基础平台、注册中心、基础设施、Agent 和 Agency 组成。

系统运行在分布式异构执行环境中，具体的操作系统可以是 Linux、Windows 以及各种 Unix 操作系统。

基础平台的作用是屏蔽运行环境的异构性和分布性，为 Agent提供一体化的运行环境。Agent运行在Agent容器中，并通过Agent容器进行信息交互。Agent容器之间通过SOAP进行通信[1]。

注册中心负责 Agent 容器、Agent 的注册管理，并为系统提供名字服务和 Agent 能力服务。注册中心遵循 UDDI 规范，在 JUDDI 基础上进行扩展，并提供一个图形化的注册管理工具。

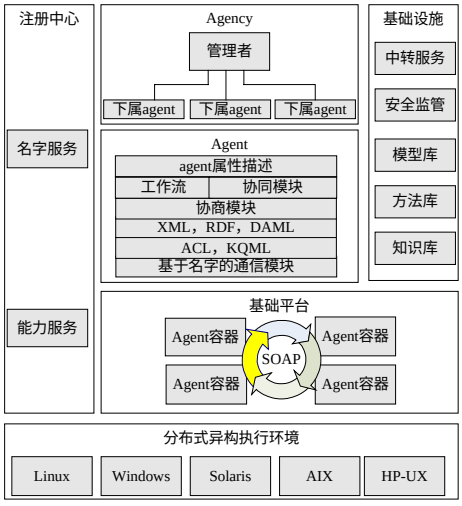

图 1　系统体系结构

基础设施提供 Agent 系统一些必需的服务，包括中转服

基金项目：国家自然科学基金资助项目(60373057)
作者简介：张树东(1969－)，男，博士后、高级工程师，主研方向：人工智能，网格技术，分布式计算；廖乐健，博士、教授、博士生导师；吕慧颖，在职博士研究生、讲师
收稿日期：2006-12-05　**E-mail：**shudong@ios.cn

务、安全监管服务、模型库、方法库和知识库。

一个 Agent 可以是独立的决策个体，也可隶属于代表一个组织或虚拟组织的 Agency。一个 Agent 在外部形态上包括若干属性以及用工作流模型表示的粗框协同行为。这些属性与粗框协同行为的不同部分发布在注册中心中。

Agency是一个具有一定组织结构、一定职能及若干资源的Agent群体。一个Agency可包含若干角色，每个角色被指派为一个或多个Agent或子Agency。每一个Agency都有一个相应的Agent负责管理与协调其运作，并代表该Agency与外界交互。该Agent定义为一个基本的角色——管理者[2~3]。

2 智能体行为描述语言 ABDL

智能体行为描述语言ABDL用于描述粗框的智能体行为，尤其是与外部的协同行为，而隐藏细节的问题求解代码。由于工作流模型在表达协同行为的作用，该语言本质上是基于工作流的。但考虑到智能体的反应能力特征与规划能力特征，在传统工作流模型的基础上加入了事件与异常的监控及目标生成机制[4~5]。系统支持两种表示形式：Lisp 表示(简约表示)及XML表示，并提供两种表示间的转换。依简约表示的语法定义，一个工作流活动递归地呈下列形式(其中：W_1 $W_2\cdots W_n$为活动)

(operation $A_1\cdots A_n$): 一个基本的操作，每个参数A_i可以是输入型，形式为表达式；或输出型，形式为一变量。

(sequence W_1 $W_2\cdots W_n$)：顺序执行活动W_1 $W_2\cdots W_n$。若某个W_i阻塞，则该活动被阻塞。若某个W_i失败，则该活动失败。

(concurrent W_1 $W_2\cdots W_n$)：按任意次序执行执行活动W_1 $W_2\cdots W_n$ 。若某个W_i失败，则该活动失败。若其中某些活动被阻塞，而其余活动均已成功执行，则该活动阻塞。

(later W $W_1\cdots W_n$)：目标生成。在活动$W_1...W_n$执行完以后的某个时刻执行W。

(conditional C W_1 [W_2]): 条件语句，W_2为可选部分。C成立则执行W_1，否则执行W_2。

(case (C_1 W_1)$\cdots$(C_n W_n) [W_{n+1}]): 情形语句，W_{n+1}为可选部分。若条件C_i成立则执行W_i。缺省时执行W_{n+1}。

(select (R_1 W_1)$\cdots$(R_n W_n)):选择执行语句，选择执行R_i 值最大的W_i。

(trigger V C W)：触发语句: 当变量(集)V 为已知，且条件 C 为真时，执行活动 W；否则继续等待。

(monitor W (E_1 W_1)$\cdots$(E_n W_n)): 监控语句：执行W，其间若发生事件E_i，$i\in[1,n]$，则执行W_i。

(for X E_0 C E_X B): 循环语句，变量X的取值由循环初值E_0依迭代表达式E_X迭代，直到循环条件C不满足，每次迭代执行循环体B。

3 基于语义 Web 的智能体互操作技术

语义Web[6]国际互联网组织W3C制定的关于未来Web的一个蓝图，主旨是通过对信息源内容用事先定义的语义知识(ontology)进行语义标注，从而在XML提供的结构互操作性基础上进一步实现语义互操作性。语义Web的核心技术层次划分如图 2 所示。其中，XML、UNICODE、URI提供信息源的基本数据表示；RDF(resource description framework)表示通用资源的基本语义模型。Ontology 则提供用于定义领域概念与术语，目前的主流语言是DAML(DARPA Agent markup language)。Ontology之上则包括领域领应用逻辑及其推理。Web 服务技术则包括由国际W3C推出的一系列技术规范，包括XML、简单对象访问协议 (SOAP)、WSDL(Web Services 描述语言)及UDDI(统一描述、发现和集成)等。系统通过采用面向服务的DAML ontology-DAML-S，来实现语义Web技术对基本Web服务架构描述与查询能力的提升[7~8]。

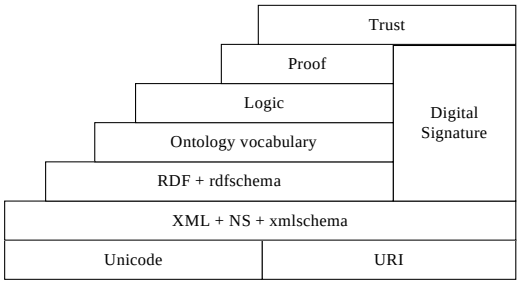

图 2 语义 Web 核心技术结构

系统的能力服务器的实现由外至内由如下层次的模块组成：

(1)语义 Web 信息解析模块：依次对 XML、RDF 及 DAML 及 DAML-S 各层进行语法解析。

(2)语义信息推理模块：实现基于描述逻辑的概念演绎及基于 DAML-S 服务 ontology 的语义匹配。

(3)语义 Web 查询服务器：通过语义 Web 查询语言实现对信息源语义内容的查询。

(4)服务资源信息库：管理与维护有关智能体能力描述的数据库，其中每一个记录对应一个智能体的 DAML-S 描述。

(5)服务交互：提供智能体能力与服务的注册、更新与查询。

4 智能体协商

IN-Agent 系统定义了两个协商协议:面向任务与资源调配的拍卖协议与面向分布式决策问题求解的分布式约束满足协议。

4.1 拍卖协议

拍卖过程实现如下：

(1)初始化(initialization)：系统中生成的每一个任务执行 Agent 都必须向有关的任务管理 Agent 汇报自身的位置、状态、能力等信息。

(2)任务拍卖(task auction)：当一个待分配的任务到达任务管理 Agent 后，任务管理 Agent 把它分解为子任务，同时生成这些子任务的投标截止期,并立即向相关任务执行 Agent (由初始化的信息得到的执行 Agent 的能力与子任务的要求相匹配)发送任务拍卖信息。

(3)投标(bidding)：每个接到任务信息的执行 Agent 根据自身状态和知识库的信息及相关的拍卖协商协议对收到的任务拍卖信息进行评估，若评价结果“满意”，则向任务管理 Agent 发送投标消息。

(4)中标宣布(result announcing)：当任务的投标截止期一到，任务管理 Agent 根据所有任务执行 Agent 的投标情况采用拍卖最优化算法挑选出最好的标值，并向所有中标的任务执行 Agent 发出 Success 消息，向其余竞标失败的任务执行 Agent 发出 Failure 消息。

(5)任务执行(executing)：接到 Success 消息的任务执行 Agent 把此项任务加入任务队列中，在适当的时间执行此项任务。

4.2 分布式约束满足协议

分布式约束满足协议的机制可描述为如下问题的求解: 给定Agents: organizer, participant1,…, participant$_n$。给定一组变量V，且有关变量的一组显性约束，求解满足该组约束及为participant1,..., participant所接受的该组变量的赋值[9]。

分布式约束满足协议的多智能体系统执行过程如图 3 所示，其中：O 为 organizer；P 为 participant-1,…, participant-n。

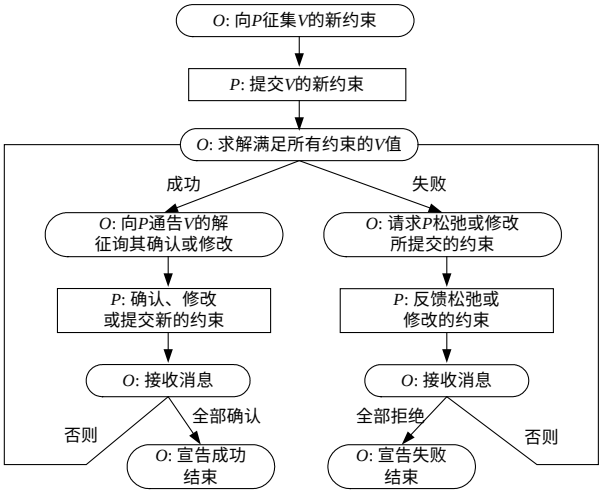

图 3　多议题约束满足协议模型

5　消息传递

In-Agent 平台提供一个透明的消息机制，流程如下：

(1)发送方 Agent，用相应的消息内容语言对消息内容进行描述。系统支持两类内容格式：DAML 和可序列化的 Java 对象。

(2)用相应的 Agent 通信语言进行消息封装。系统支持 ACL 和 KQML 两类 Agent 通信语言。

(3)调用平台提供的 send 方法发送消息。平台首先通过查询 UDDI 中心，验证发送方和接收方的合法性，然后调用 SOAP 操作将消息发送给接收方 Agent 所属的 Agent 容器。

(4)接收方首先验证消息的合法性和完整性。如果验证通过，向接收方返回“ ok ”，否则返回“ false ”。然后将消息进行 base64 解码，放入相应 Agent 的接收队列中。

(5)Agent 从接收队列中接收消息，并进行相应的消息处理。

(6)如果 Agent 尚未登陆，则将消息发往中转服务中心，进行消息的缓存；中转服务中心定时查询注册中心中 Agent 的状态，如果 Agent 已登陆，则将消息转发给 Agent。

6　结论

In-Agent 提供了一个多 Agent 运行平台。虚拟组织可视为传统组织建模的延拓，其实体以互联网作为媒介，相互间通过共同的任务与活动而形成“ 临时的组织形式 ”，并在该组织形式中担当不同的角色。系统采用 Web 服务作为基础技术框架，主要体现在基于 WSDL 的 Agent 属性描述，基于 SOAP 的 Agent 通信，以及基于 UDDI 的注册服务。系统还提供了一些应用工具，使应用开发能够提高开发效率，缩短开发时间，提高系统的稳定性，减少系统的测试开销。

参考文献

1 张树东. 基于 Web Services 的 MAS 管理与通信平台[J]. 计算机应用与软件, 2005, 22(10):137-138.

2 Nwana H S, Ndumu D T, Lee L C. ZEUS: An Advanced Tool-kit for Engineering Distributed Multi-Agent Systems[C]//Proc. of PAAM'98. 1998: 377-391.

3 Lam M S, Rinard M C. Coarse-grain Parallel Programming in Jade[C]//Proceedings of the 3rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 1991.

4 Chalmers S, Gray P, Preece A. Supporting Virtual Organisations Using BDI Agents and Constraints[C]//Proc. of the 6th Int'l Workshop on Cooperative Information Agents. 2002.

5 Hendler J, McGuinness D L. Darpa Agent Markup Language[J]. IEEE Intelligent Systems, 2001, 15(6): 72-73.

6 McIlraith S, Son T C, Zeng H. Semantic Web Service[J]. IEEE Intelligent Systems, 2001, 16(2): 46-53.

7 Sycara K, Klusch M, Widoff S, et al. Dynamic Service Matchmaking Among Agents in Open Information Environments[J]. ACM SIGMOD Record, 1999, 28(1): 47-53.

8 Wang Huaiqing, Liao Lejian. A Framework of Constraint-based Modeling for Cooperative Decision Systems[J]. Knowledge-based Systems, 1997, 10(2): 111-120.

9 Wang Huaiqing, Liao Lejian. Modeling Constraint-based Negotiating Agents[J]. Decision Support Systems, 2002, 33(2): 201-217.

(上接第 45 页)

综上所述，注册仓模式适用于以下情况：(1)系统大部分模块需要以某种方式共享同一数据；(2)系统需要强调数据的发布者对使用者是透明的；(3)系统需要独立管理使用者对数据的访问可见性；(4)在多线程环境中系统大部分模块需要在线程内共享数据。

本模式建议以只读方式共享数据，如果用户修改线程间共享的享元，会产生数据的不一致性，虽然文中给出了解决该问题的办法，但并不够灵活，如何灵活有效地解决这一问题需要进一步的工作。

参考文献

1 Pree W, Sikora H. Design Patterns——Essentials, Experience, Java Case Study[C]//Proc. of Asia-Pacific Software Engineering Conference. 1997: 534-535.

2 Al-Otaiby T N, Bond W P, AlSherif M. Software Modularization Using Requirements Attributes[C]//Proc. of ACM Southeast Regional Conference. 2004: 104-109.

3 Silva A R, Sousa P, Antunes M. Design Pattern and Framework[C]//Proc. of International Computer Software and Applications Conference. 1998: 316-323.

4 邹　娟, 田玉敏. 软件设计模式的选择与实现[J]. 计算机工程, 2004, 30(10).

5 Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-Oriented Software[M]. Boston: Addison-Wesley, 1995.