维普资讯 http://www.cqvip.com

基于 Jini 的多 Agent 系统研究

A Study of Jini-Based Mulit-Agent Systems

肖正光　王振杰　徐良贤

（上海交通大学计算机系　上海200030）

Abstract　Agent-oriented techniques represent an exciting new means of analysing, designing and building complex software systems. In this article, we introduce the main properties and communication approaches of Multi-agent systems, and we present a Jini-based multi-agent architecture. At the same time, we discuss the advantages of the architecture to build the open, dynamic, heterogonous multi-agent systems.

Keywords　Agent, Multi-agent systems, Jini, Distributed systems

1　引言

随着 Internet 的飞速发展，我们已经进入了一个世界范围的网络计算时代。由于 Internet 固有的开放性、动态性、分布性、异构性使得 Internet 软件变得越来越复杂，用传统的面向过程和面向对象方法开发复杂的 Internet 软件已非常困难，于是便出现了面向 Agent 的软件，面向 Agent 的技术是分析、设计和构造复杂的分布式软件的一种新技术，它代表了软件开发未来发展的方向[1]。

面向 Agent 的软件系统致力于解决传统方法不能很好处理的问题有：

1. 开放性。在 Internet 这样高度开放的环境中，用户预先不能确定所有与其交互的各种资源，这就要求替用户完成某种任务（如在网上查找信息或确定用户的旅游行程）的软件具有适应性。Agent 具有这种适应性，它能够对环境的变化做出反应，在适当的时候采取面向目标的行动，以及能从其自身的经历、外界的环境和其它 Agent 的交互过程中学习。

2. 复杂性。问题领域的复杂性和不可预测性使得通用的软件系统的开发方法不可行，唯一的解决途径是开发一些能够解决特定领域问题的专用模块化组件。一个大的复杂的问题求解可以分解成许多小的较简单的子问题，这样子问题子系统易于开发维护，分解策略使得每个 Agent 可以采用最合适的方法去解决特定领域问题，这样系统的效率也就很高。面向 Agent 系统与基于组件的面向对象系统相比，其优点在于：Agent 能以灵活的、与环境有关的方式（它具有自治性、反应性、主动性和社会性），而不是通过一些预先严格定义的接口函数来与外界进行交互作用。此外，环境的不可预测性要求 Agent 能对环境变化做出反应，又能根据目标主动规划自己的行为。

面向 Agent 的软件系统具有开放性、动态性、分布性、异构性，所以用来实现软件 Agent 的框架必须能很好地提供这些方面的支持，而 Jini 技术是构建简单、灵活、开放的分布式系统的一种有前途的技术，因此我们提出了基于 Jini 的多 Agent 系统，它可以用来方便地实现多 Agent 系统。本文对多 Agent 系统特性和 Jini 关键技术作了简要介绍，提出了基于 Jini 的多 Agent 系统，并进行简要分析，最后是结论。

2　多 Agent 系统介绍

单个 Agent 的能力因其所拥有知识、资源和视野的限制而不适合于面向 Internet 的分布式软件，在 Internet 这种开放的分布式环境中信息资源、通信连接和完成某项任务的 Agent 其出现和消失是不可预期的，单 Agent 系统显然不适合这种开放的、动态的分布式环境。目前，在 Internet 上的 Agent 大多从事信息获取、信息收集和电子商务等工作，这要求多个 Agent 通过交互进行协调，而这些都需要用多 Agent 系统来实现。

多 Agent 系统是由多个 Agent 组成的集合，在多 Agent 系统中 Agent 之间以及 Agent 与环境之间通过通信、协商和协作来共同完成单个 Agent 所不能解决的问题。多 Agent 系统具有如下特性[1]：(1)每个 Agent 具有解决问题的不完整的信息或能力；(2)没有系统全局控制；(3)数据是分散的；(4)计算是异步的。这些特点使得多 Agent 系统非常适合用于分布式系统中。

在开放、动态的多 Agent 环境下，具有不同目标的多个 Agent 必须对其目标、资源的使用进行协调。例如，当单个 Agent 无法独立完成某项任务，需要其它 Agent 的帮助时，就需要进行协作。另一方面，在多个 Agent 共享某种资源发生冲突时，若没有很好的协商和冲突消解机制，就有可能出现死锁。这一切都需要 Agent 之间进行交互通信，通信是 Agent 之间协商和协作的基础。多 Agent 间的通信方法有基于黑板的通信和消息传送[3]：

(1)基于黑板的通信：在多 Agent 系统中黑板提供公共工作区，Agent 可以在此交换信息、数据和知识。一个 Agent 在黑板上写入信息项，然后该信息项可为系统中其它的 Agent 使用。Agent 可以在任何时候访问黑板，查看有没有新信息到来。Agent 并不需要阅读所有信息，可以采用过滤器抽取它感兴趣的信息。在黑板系统中 Agent 之间不进行直接通信，每个 Agent 独立地完成它所处理的子问题。黑板可用在任务共享和结果共享的系统中。

(2)消息传送：采用消息来通信是实现灵活复杂的协调策略的基础。使用规定的协议，Agent 彼此交换的消息可以用来建立通信和协作机制。为了支持协作策略，通信协议必须明确

肖正光　博士生，研究方向为分布式技术、多 Agent 技术、Agent 协商。王振杰　博士生，研究方向为多 Agent 学习、数字城市。徐良贤　教授，博士生导师，研究方向为系统软件、软件工程、Agent 技术。

维普资讯 http://www.cqvip.com

规定通信过程、消息格式和选择通信语言。另外 Agent 间的通信是知识级的通信，参与通信的 Agent 必须知道通信语言的语义。有两种方式来实现 Agent 间的消息传送：

* 直接通信：在直接通信的方式中，每个 Agent 必须知道消息应该在什么时候发送去什么地方，系统中有哪些 Agent 是可以合作的，这些 Agent 各具备什么样的能力等。这要求系统中的每个 Agent 都拥有关于其它 Agent 的大量信息，但在开放的分布式系统中这往往是做不到的。

* 通过中介 Agent 的通信：在基于中介的消息传送中，若干相距较近的 Agent 通过一个中介 Agent(Mediator 或 Facilitator)来进行交互和消息发送，而远程 Agent 间的交互是由局部 Agent 群体中的中介 Agent 协作完成的。

3 基于 Jini 的多 Agent 系统

Jini 是由 Sun Microsystems 于1999年推出的，它将 Java 技术发明所建立起来的以网络为中心的计算模式向前推进，摧毁了传统的网络壁垒，使用户可从任何地点将任何消费类电子产品和企业设备简单地相连。Jini 为分布式应用系统之间实现信息互访、知识共享和协同工作提供了有力的支持。采用 Jini 技术建立的多 Agent 系统，使在网络上分布的 Agent 具有"即插即用"的特点，便于 Agent 之间灵活地组织成共同完成某项任务的联盟。它为各 Agent 之间实现数据和知识的交换、协调和协作提供了良好的支撑。

3.1 Jini

Jini 是以 Java 技术为中心，在充分利用 Java 代码的可移动性的基础上，以"服务"的观念构建简单、灵活、开放的分布式系统的一种技术。Jini 的五个基本概念[4,5]如下：

(1)发现(Discovery)：一个 Jini 实体(服务或应用)在使用其他 Jini 服务之前，必须先找到一个或多个 Jini 群体，途径是寻找跟踪该群体中共享资源的查找服务。这个查找可用的服务过程称为发现。Jini 发现协议是 Jini 程序寻找 Jini 群体的途径，一旦找到某个群体，Jini 实体就会按照加入协议的一系列规范加入到该群体中，并在该群体中公布服务。

(2)查找(Lookup)：查找是控制提供某个特定服务所需要的程序如何把自己提供给使用服务的参与者的方式。查找在每个 Jini 群体中的作用相当于目录服务，它提供用于寻找一个群体中已知服务的功能。但查找服务的功能实际上要比名字服务器复杂得多，名字服务器只是把字符串映射到对象上，而 Jini 的查找服务可在对象类型的基础上进行，在搜寻过程中甚至还可以通过查找存储对象的超类和超接口进行搜寻。

(3)租借(Leasing)：租借技术使 Jini 具有自修复能力，它保证了一个群体在某些关键服务失败的情况下，一段时间之后可以恢复。租借还保证了长时间运行的服务(如查找服务)不会"积累"其群体的信息，若没有租借，长时间运行的服务会无限制地增长。

(4)远程事件(Remote Event)：远程事件是 Jini 服务彼此通报状态的变化所采用的范例。在一个群体可用的服务发生变化时，它可以利用远程事件方式通知相关的参与者。

(5)事务(Transaction)：事务是 Jini 使包含多个服务的计算到达一个"安全"状态的机制，即调用者可以得到保证，计算要么全部完成，要么都没有进行，无论哪种情况，系统都到达一个确知的状态。Jini 的事务模型可以消除分布式系统中部分失败所带来的危害，极大改善了服务的健壮性，使服务对网络故障有更大弹性。

Jini 群体的成员可以共享服务。服务是一个能被一个用户、一个程序或另一个服务使用的实体。一个服务可以是一个硬件设备、一个软件 Agent、一个通信信道或一个用户。一个 Jini 群体可以被看成是能够完成特定任务的服务组成的一组组件和程序，或用户和程序，或一组客户和服务器。一个服务可以使用其它服务，并且一个服务的客户本身可以是带有客户的服务。Jini 群体的动态性允许服务根据需要在任何时候加入或退出群体。

Jini 提供了在分布式系统中服务的建立、查找、通信和使用的机制。服务在 Jini 中相互通信是使用各自的通信服务协议，即一组 Java 接口，协议集是公共的。基本的 Jini 系统定义了一些处理关键事物的协议。

服务间的相互通信是通过 RMI(远程方法调用)来完成的。RMI 提供了发现、激活和垃圾收集对象集合的机制。RMI 还提供了广播、复制的机制以及基本安全和保密的机制。RMI 不仅允许网络上对象间的数据传送，而且还可以传送整个对象，包括代码。它是通过将代码封装成对象在网络上传送的。这个特性使得它非常适合于开发能在网上迁移的移动 Agent 系统。

3.2 基于 Jini 的多 Agent 系统体系结构

Jini 提供了建立分布式系统的简单和健壮的基础平台，但是构造高度适应性、交互性、互操作性和自治性的分布式系统只有 Jini 体系结构是远远不够的。由于 Agent 具有自治性、社会性、反应性、自适应性、推理能力、可移动性等，因此将 Agent 技术和 Jini 结合起来构造多 Agent 系统则可以得到应用于 Internet 上的开放、动态的多 Agent 系统。

在基于 Jini 的多 Agent 系统中，Agent 有通用 Agent 属性和领域专用 Agent 属性，它们是与 Jini 的查找服务相关的 Jini 服务属性。通用 Agent 属性定义了 Agent 与具体领域无关的属性，领域专用 Agent 属性则定义了与一个 Agent 所在问题域的有关的属性。通用 Agent 属性用来以一种与 Agent 的能力和特性无关的方式将一个 Agent 与另一个 Agent 区别开来，常用的通用 Agent 属性有：Agent ID 号，Agent 角色等。领域专用属性描述该 Agent 的能力。

一个 Jini 群体可以是一个 Agent 联盟，在一个 Agent 群体中的 Agent 代表服务的提供者或消费者。一个 Agent 联盟有多个 Agent，一个 Agent 可以同时属于多个 Agent 联盟。两个 Agent 之间若要交互，则它们必须是处在同一个 Agent 联盟中。当一个 Agent 加入一个 Agent 联盟时，它带有一组说明它的身份的通用属性及描述其能力的领域专用属性，其它的 Agent 为完成某个目标或某项任务需要其它 Agent 的协作时就检查这些属性以确定能提供这种服务的 Agent。若某个服务能满足该 Agent 的需要，该 Agent 就与提供此服务的 Agent 协商，以获取该服务的使用。其交互过程如图1。

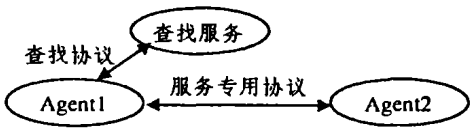

图1 联盟内 Agent 间的交互

在新的 Agent 准备加入一个 Agent 联盟时，它使用组播协议来寻找附近所有在运行的查找服务。当在监听的查找服务接收到一个组播发现(Discovery)请求时，它直接连到发出

维普资讯 http://www.cqvip.com

请求的 Agent 并发送一个点到点消息进行应答，这个消息包含了查找服务的代理对象，Agent 可通过它来访问查找服务。具体实现如图2。

图2 Agent 加入联盟及与其它 Agent 的交互

提供服务的 Agent2通过发现协议找到查找服务(Lookup Service)，再由加入协议将所提供的服务注册到查找服务中，要注册的信息包括服务标识号、服务实体(包含服务类型的信息)和服务属性。使用服务的 Agent1也通过发现协议找到查找服务，再由查找协议按所需服务的类型和属性查询服务，然后 Agent1以特定的服务协议与 Agent2通信以使用 Agent2提供的服务。

对于查找服务，没必要与各 Agent 联盟一对一地映像，网络上的每个查找服务都可以为一个或多个 Agent 联盟提供服务，每个 Agent 联盟也可以有一个或多个查找服务支持。这样就有效地消除了单点故障(Single Point Failure)。

基于 Jini 的多 Agent 系统中，Agent 模型如图3所示。

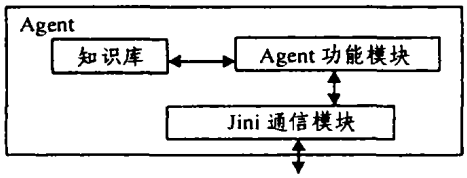

图3 基于 Jini 的 Agent 模型

Agent 的知识库包括 Agent 对自身的描述、Agent 对世界状态的描述及对其它 Agent 状态的描述。功能模块主要是根据 Agent 自身的目标完成规划、调度和动作，功能模块是 Agent 的核心，因为它反映了 Agent 的真正功能和决定了 Agent 的结构，根据 Agent 功能模块的不同，Agent 可为慎思型 Agent、反应型 Agent 和混合型 Agent[6]，以适应不同的需要。Jini 通信模块用来与其它 Agent 进行通信，通信的内容用 Agent 通信语言(ACL)来编码，如可以使用 Agent 通信语言事实上的标准语言 KQML 或 FIPA ACL。

4 分析

Jini 提供了可用来实现动态 Agent 联盟的机制。通过使用租借，一个 Agent 可以表明它加入一个 Agent 联盟的时间段。在租借结束时，Agent 可以考虑重新租借或退出 Agent 联盟。这样一个 Agent 可动态地加入和退出 Agent 联盟，可以很容易地实现动态、开放的多 Agent 系统。Jini 技术的另一个对实现多 Agent 系统有用的特性是一个要加入 Agent 联盟的 Agent 不一定要必须运行在 Java 虚拟机(JVM)上，这个 Agent 可以使用一个代理(proxy)来作为与其它 Agent 及查找服务交互的中介。通过代理的使用，一个 Agent 可以透明地和另一个 Agent 进行通信：只需预先知道另一个 Agent 有什么功能，而不必了解该 Agent 是如何去做的，包括实现该 Agent 的语言、软硬件平台、物理位置和使用的通信协议。这样 Agent 就可以不用 Java 实现，甚至能用代理来封装遗产软件(legacy software)，实现异构的多 Agent 系统。

另外，Jini 还提供了可以用来方便地实现基于黑板通信的多 Agent 系统的服务—JavaSpaces。JavaSpaces 是建立在基本 Jini 框架之上的特殊服务，它为 Java 对象提供存储功能。利用 JavaSpaces，Jini 服务和客户可以创建并控制 Java 对象的"空间"，它们可以为 Java 对象得到租借的存储区，可以在存储空间内搜寻存储的对象或删除存储对象。这样一组 Agent 可以使用 JavaSpaces 作为"共享黑板"来存储和取出对象。对"空间"的所有修改操作都以事务安全的方式进行，多个 Agent 可以使用 Jini 的事务机制来实现多个操作或多"空间"更新的原子性。在空间中存放的对象可以表示要做的工作(因此可用于创建生产者/消费者模式的多 Agent 系统)，或只表示数据信息(可用于创建有分布式投票功能或报价的多 Agent 系统)。

结论 理想的多 Agent 系统构造平台对于实现开放、动态、异构的多 Agent 系统的重要性是不言而喻的。Jini 提供了构建简单、灵活、开放的分布式系统的基础，而 Agent 技术是构建复杂、"智能"软件最有潜力的技术，基于 Jini 的多 Agent 系统框架能够结合 Agent 技术和 Jini 技术的优点，使构建面向 Internet 的开放、复杂的软件变得更为高效。

参 考 文 献

1 Jennings N R, Wooldridge M. Agent-Oriented Software Engineering. in Handbook of Agent Technology (ed. J. Bradshaw) AAAI/MIT Press, 2000(to appear)
2 Sycara K. MultiAgent Systems. AI Magazine, 1998, 19(2): 79～92
3 Weiss G. MultiAgent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, 1999
4 Edwards W K, Jini C. Prentice Hall PTR 1999
5 Technology J. Available at http://www.sum.com/Jini
6 Wooldridge M, Jennings N R. Intelligent Agents: Theory and Practice. Knowledge Engineering Review, 1995, 10(2)

(上接第91页)

3.3 编写对 XML 文档的查询语句

Transact-SQL 关键字 OPENXML 为内存中的 XML 文档提供了数据行。OPENXML 与表或视图是比较相似的，提供了对 XML 数据的访问途径。在数据行提供者(表、视图或 OPENROWSET)可以作为数据源出现的地方，都可以在 SELECT 或 SELECT INTO 语句中使用 OPENXML 关键字。要使用 OPENXML 编写对文档的查询语句，必须首先调用 sp_xml_preparedocument 系统过程分解 XML 文档，并得到对分解后文档的处理。使用系统存储过程可以释放内存中的 XML 文档。

结束语 面向 Internet 的数据挖掘是一项复杂的技术，而 XML 是直接面对 Web 数据的，不仅可以很好地兼容原有的 Web 应用，而且可以更好地实现 Web 中的信息共享与交换。XML 可看作一种半结构化的数据模型，可以很容易地将 XML 的文档描述与关系数据库中的属性对应起来，实施精确的查询与模型抽取。微软的 SQL Server 2000支持完全集成的 XML 环境，使面向 Internet 的数据挖掘成为可能。

参 考 文 献

1 罗运模. SQL Server 2000数据仓库应用与开发. 北京：人民邮电出版社，2001，7
2 长城工作室数据组. SQL Server 2000高级应用. 北京：人民邮电出版社，2001，8
3 Inmon, W. H. 著，王志海等译. 数据仓库. 北京：机械工业出版社，2000，5
4 Lou Agosta 著，潇湘工作室译. 数据仓库技术指南. 北京：人民邮电出版社，2000，11