摘 要

如任何技术创新一样，Agent 的产生和发展具有时代背景，同时受到需求牵引和技术推动的作用。90 年代先进计算机应用系统，已经从基于知识的系统演进到各计算单元为了同一解题目标、协同工作的智能性复杂系统。其组织结构及运行方式如人类社会一样，称社会式组织的系统。由 Agent 单元组成的多 Agent 系统就是这样一种多元协作系统，系统中各 Agent，如人类社会成员一样各尽其责，为了同一目标协同工作。在形式化体系上，Agent 有实时处理多维信息、相互作用和通信能力，从而适应了应用软件开发中实时、并发处理的需要。通过网上合作，充分利用空间分布的智力、信息和技术资源以合作方式解决问题，这种结构体系和运作方式已经形成新的多智能体系统规范。

20 世纪 90 年代后期，随着互联网迅速向人类社会各层面扩展其应用和影响。对于网上资源共享和高端计算能力直接向用户提供的需求更显迫切，一批网络计算高新技术的出现犹如雨后春笋。其中移动智能体系统的出现，更是众所瞩目。移动智能体除了具备智能体，多智能体系统已有的技术特点外，它可以按照任务执行的需要，移动到网络节点上去工作。其主要技术特征是实现计算执行程序的异地执行，将“处理”机制移向被处理的“数据”，大幅度节约网络带宽资源。特别在分布式超级计算、高吞吐量计算、数据密集型计算以及合作计算中发挥了巨大的作用。尽管这一新兴技术在多种应用中显示出出人意料的功效。但是，也因为它还在蓬勃发展中，该项技术在理论基础、结构体系、公共设施、系统标准、运行安全等方面还有许多问题，有待研究。作者的论文是作者试图追随这一技术前沿的努力。

面向应用需求，结合具体解决方案，除了研究和开发了面向问题求解的基于移动智能体技术的应用原型系统外，作者在研究实践基础上进行了诸如移动 Agent 系统的基准模型、Agent 的迁移机制、Agent 之间的通信和协作、Agent 及其宿主机的安全等问题的研究，论文总结和报告了以下各项内容：

（1）提出了移动 Agent 系统的基准模型。在系统分析了移动 Agent 技术、环境和现有的多个移动 Agent 系统的基础上，该基准模型给出了移动 Agent 技术的公共概念集，并从实体、服务和接口三方面阐述了移动 Agent 系统的基本成份、各成份的基本功能以及各成份之间的关系。为移动 Agent 技术的研究和探索、系统的设计和实现提供了一个规范的基准模型。

（2）提出了一种新的结构化迁移机制。该机制的结构化主要体现在以下三个方面：①Agent 的旅行计划和功能体完全分离，便于移动 Agent 的开发、理解、复用和维护；②旅行计划具有严格定义的结构，便于旅行计划的开发和使用；③提供了两种灵活的迁移模式，增强了迁移机制的表达能力。在上述工作的基础上，本文还对该机制使用的结构化迁移信息描述语言的操作语义进行了探讨，初步刻画出它的结构化操作语义框架。与已有的同类工作相比，这种结构化的迁移机制具有描述能力较强、复用性较好、开发简便和用户友善性强的特点。

（3）提出了一种移动智能体基于 KQML 的分层多模式通信框架。该框架中的 KQML 模式支持基于知识交换的高层协作。其间接访问模式以消息传递为通信手段，直接访问模式允许 Agent 直接进行基于网络协议的数据传输。在间接访问模式下，针对移动 Agent 系统特有的通信失效现象，提出了一个较为完整的解决方案。

（4）设计了一个基于互联网的移动智能体系统 IBMAS。在上述工作的基础上，设计了 IBMAS，从而对移动 Agent 计算进行了系统支持。它提供了移动机制、通信机制和安全机制，从而便于用户在此基础上进行应用程序的开发和实际的应用。

（5）在 IBMAS 的基础上，开展了一系列应用研究。详细阐述了移动智能体在入侵检测系统、漏洞扫描系统和信息检索系统中的应用。

关键词：迁移机制，通信框架，入侵检测，漏洞扫描，信息检索

Abstract

Like any other technology innovation, the appearance and development of agent have their era background, and at the same time, they are affected by the demand traction and technology motivation. In the 1990s, the advanced computer application systems have evolved from knowledge based systems to intelligent complex systems in which each computing unit is for the same goal and cooperates with each other. The systems, whose framework and running manner are like human society, are called socially organized systems. The multi-agent system which is made up of agent unit is such a many-member cooperation system. All the agents in the system cooperate with each other for the same goal, just as members of human society do their best according to their abilities. On the formalization system, agent has the abilities to real time deal with several-dimension information, affect and communication with each other, thus, it adapts the requirements of real time, concurrence disposal in the design of application softwares. The framework system and running manner, making good use of distributed brains, information and technology resources and solving problems by the cooperation in the network, have formed new multi-agent system criterion.

At the end of 1990s, with the internet rapidly spreading applications and effects to society, the demand of resource sharing in the internet and offering powerful computing ability to consumers seems to be more anxious, a series of network computing technologies appeared like bamboo shoots after spring rain. Thereinto, the appearance of mobile agent system catches people's eyes. Mobile agent system has the character of agent, multi-agent system, what's more it can work by moving to network node according to the execution of mission. Its main technology character is that executive programs can run in another place. Moving management mechanism to disposed data, reduces the used network bandwidth greatly. It takes an important role especially in distributed super computing, complex computing, large data computing and cooperation computing. Although this advanced technology has showed its great efficacy in many applications, there is still many problems such as theory foundation, framework system, commonality establishment, system criterion, running security, which need to be solved. The author tries to follow the leading edge of this technology and makes great efforts in this thesis.

Facing the application demand and combining with the detail blue print, the author makes research such as norm model of mobile agent systems, agent migration mechanism, communication and cooperation between agents, security of agent and agent hosts except the investigation and design of the application prototype systems which adopt mobile agent technology. The thesis summarizes and reports as the following:

(1) A norm model for mobile agent systems is given. It is established after analyzing the current mobile agent systems and the related technologies. This model presents a glossary of mobile agent technology and describes the basic elements of mobile agent systems, the functionality of each element, the relationship between these elements from the perspective of entity, service and interface. This model provides a good foundation for the further research and system design of mobile agent.

(2)A new structured agent migration mechanism is provided. The main features of this new structured migration mechanism are: ①full separation of the mobile agent's itinerary from its function body; ②strict and explicit definition of the structure of the itinerary; ③Two kinds of flexible and powerful migration modes. Based on the work above, this paper also discusses the operational semantics of a structured migration information description language used in IBMAS migration mechanism and presents a primary framework of its structural operational semantics. With its structural mechanism, the migration requirement can be easily described, the complexity of mobile agent can be managed and the reusability is improved.

(3)A hierarchical multi-mode communication framework of mobile agent is proposed. There are three modes in this framework: the KQML mode which supports knowledge based high level cooperation, the indirect mode which mainly employs inter-agent message passing and the direct mode which allows agents interchange data directly with sockets APIs. To avoid the communication failure caused by the autonomous mobility of mobile agents and the asynchronous inter-agent message passing, a new message delivery algorithm is proposed.

(4)The IBMAS is designed and completed. Based on the above work, the paper implements a mobile agent system. That is IBMAS, which provides a systematic support for the mobile agent based computing in three important aspects, i.e., the agent migration mechanism, the inter-agent communication, and the security protection. IBMAS also supports mobile agent systems developing and applying.

(5)Based on the IBMAS, a series of typical applications are given. The applications of mobile agent technology in intrusion-detection systems, vulnerability scanning systems, and information retrieval systems are described in detail.

Keywords: Migration Mechanism Communication Framework Intrusion Detection Vulnerability Scanning Information Retrieval

1 绪论

1.1 智能体研究概况

Agent 的原意是“代理”，即一个人代表另一个人或（另）一个组织去完成某件（些）事情。在计算机领域，可以认为 Agent 是被授权的“个人软件助理”（Personal Software Assistants），是一种在分布式系统或协作系统中能持续、自主地发挥作用的计算实体，常常简称为智能体[1]。

Agent 的概念出现于 20 世纪 70 年代的人工智能（Artificial Intelligence，AI）中，80 年代后期才成长起来。由于分布并行处理技术、面向对象技术、多媒体技术、计算机网络技术，特别是 Internet 和 WWW 技术的发展，Agent 不仅成为人工智能和计算机领域最活跃的研究内容之一，而且引起了科学界、教育界、工业界甚至娱乐界的广泛关注[1]。

Agent 技术，特别是多 Agent 技术，为分布式开放系统的分析、设计和实现提供了一种崭新的方法，被誉为是“软件开发的又一重大突破”[2-3]。目前，对 Agent 的研究大致分为如下三个相互关联的方面：①智能 Agent；②多 Agent 系统（Multi-Agent System，MAS）；③面向 Agent 的程序设计（Agent Oriented Programming，AOP）。智能 Agent 也称软件 Agent，是多 Agent 系统研究的基础，我们也可以将智能 Agent 的研究统一在多 Agent 系统的研究之下，这样，智能 Agent 被看成是多 Agent 系统研究中的微观层次，主要研究 Agent 的理论和结构，包括 Agent 的概念、特性、分类，Agent 的形式化表示和推理等；而有关 Agent 之间的关系的研究则构成了多 Agent 系统研究的宏观层次，它主要研究多个 Agent 组成的系统中 Agent 的组织以及 Agent 之间的通信、规划、协同、协作、协商与冲突消解、自组织、自学习等问题。智能 Agent 和多 Agent 系统的成功应用与开发要借助于面向 Agent 编程的方法论，即 AOP 以及 AOP 开发工具或平台[2-3]。

移动Agent（Mobile Agent，MA）的思想起源于Agent，受到互联网技术的推动和分布计算需求的牵引，移动Agent技术成为20世纪90年代最受瞩目的新技术。它是Agent研究领域的一个重要分支[1]，也是最具发展潜力的分支。

1.1.1 智能体技术的发展

1.1.1.1 智能体的产生

- 互联网的发展

智能体技术的发展与互联网的发展是密不可分的。首先，互联网的发展是网络技术的进步，这种进步使得人们思考和探索这样的问题：计算机怎样互联起来以更好地利用资源，高效地完成各种信息处理？许多研究工作就是围绕这个目的而展开的，如美国国防部DARPA的基于智能体系统的控制，及可分配任务的智能体软件工具等项目的研究。它们的共同特点是在保持单机或单个应用灵活性的前提下，提供简便和广泛的网络资源共享和异构信息集成。

其次，网络上的信息资源也越来越庞大，人们在享受它方便和快捷的同时，也为它所包含的庞大芜杂的信息所淹没，为了找到自己需要的信息往往要花费大量的时间和精力。如何能够更有效、更准确地找到自己感兴趣的信息，滤除与自己的需求无关的信息，已成为人们非常关心的问题。目前网上的Yahoo、Google和Sohu等搜索引擎，一般以信息查询技术为基础，帮助用户获得所需信息，但是这种信息查询方式存在着明显的不足：

①用户必须以关键词的形式归纳出自己的信息需求，不同的关键词和及其组合方式对搜索结果的有效性影响很大，因此，在有些情况下，这种方式很难获得好的效果；

②由于网络信息不断更新，用户为获得最新信息需要用相同的关键词进行反复查询；

③查询方式的结果与用户兴趣无关，不能实现个性化的信息服务。人们迫切需要能根据自己的兴趣来过滤和获取信息的各种类型的信息系统。

而且，人们的日常生活也越来越依赖于网络。随着电子商务的发展，人们从网络上购买自己所需的商品，企业之间依靠网络完成商品的交易，政府部门通过网络提供各种不同类型的服务。

- 面向对象方法学的进展

面向对象的方法是20世纪80年代初期提出的一种新兴的程序设计方法，它彻底克服了过去数据流等方式的缺点，采用直接对问题域进行自然抽象的方法，并逐渐发展成包括面向对象分析、设计、编程、测试、维护等一套完整的内容体系。

面向对象的方法论所体现的对客观世界的描述比较符合人们对客观世界的认识，从而使软件设计在数字世界中描述客观存在与现实世界人们的认识统一起来，并保持整个软件开发过程中概念的一致性，便于各个阶段的管理与控制，是人们在软件开发过程中认识的一次飞跃，其基本思想是用对象、类、继承、封装等基本概念来进行程序设计。其主要特点有：继承性(Inheritance)，封装性(Encapsulation)，多态性(Polymorphism)。

随着软件规模和复杂性的不断增长，人们对软件开发过程认识进一步深入，于是面向对象的开发方法得到了改造和提升，出现了一些具有“智能”的对象，具体表现为：

①消息机制的引入为对象间进行通信提供了手段，同时也为更有效地实现不同类对象间的协作提供了保障；

②把知识表示方法和行为模型与面向对象技术相结合，使得对象具有能根据外界环境的变化而自主地对自己的行为和状态进行调整的能力，从而适应外界环境的改变；

③对象的复杂性不断增长，重用的粒度和级别发生变化。组件式程序设计使得重用粒度变得更灵活，语言无关的重用性进一步增强。

1.1.1.2 智能体基本概念

许多研究人员依据不同的研究内容和目标对术语“智能体”赋予了不同的含义。英国Wooldridge和Jennings教授对不同定义进行了总结，他们认为，“智能体”按其用法可分为两种，也就是智能体的两种定义：弱定义与强定义。智能体的弱定义是将其定义为具有如下特性的计算机软件或硬件系统[4-6]：

① 自主性(Autonomy)　智能体能自行控制其状态和行为，能在没有人或其它程序介入时操作和运行；

② 社会性(Social Ability)　智能体能通过智能体通讯语言与其它智能体或人类进行交互；

③ 反应性(Reactivity)　智能体能及时地感知和响应其所处环境的变化；

④ 能动性(Pro-Actively)　智能体主动表现出目标驱动的行为，能自行选择合适时机采取适宜动作。

而符合强定义的智能体，它们除了具有弱定义中的特性以外，通常还具有以下一种或多种特性：

① 可移动性(Mobility)　指具有在计算机网络上进行移动的能力，且在此过程中保持状态一致；

② 理性(Rationality)　智能体的行动有助于达到其目标，不能接受冲突的目标；

③ 可靠性和诚实性（Benevolence，Veracity）　智能体采取的动作及产生的结果应是可靠和符合用户利益的；

④ 适应性(Adaptively)　应能进行自我调整，具有适应其用户的工作方法、方式及偏好的能力；

⑤ 协作性(Collaboration) 智能体应能在多智能体环境中协同工作和消解冲突，以执行和完成一些互相受益且自身无法独立求解的复杂任务。

1.1.1.3 智能体和多智能体系统

智能体系统是当前人工智能研究的一个前沿课题，几乎涉及人工智能的所有深层次问题。对智能体的研究可以从很多角度展开，基于不同研究所做出的假设是不同的，因而对智能体的描述也有很多。不论什么样的基于智能体的系统，必然涉及两个方面的问题。

① 单个智能体的行为刻划（即单智能体规范）。不论什么样的多智能体系统，其系统的行为必然要归约为单个智能体的行为，我们在研究多智能体系统时也必须对单个智能体的行为进行刻划，而在实现的时候还必须考虑单个智能体的结构（它们应具有什么样的功能）。

② 每个智能体如何与环境及其它智能体进行交互。在协作情况下，如何相互帮助达到各自目标。

关于智能体系统的研究大体上可以分为两大类型。

第一类研究往往基于 BDI（Belief Desire Intention）模型对单个智能体的心智状态进行刻划，采用的工具是模态逻辑和分支时序逻辑[6-10]。同时也采用 BDI 模型对多智能体系统中的智能体行为进行规范和描述，即刻划智能体在多智能体系统中所应具有的高级认知状态；这种研究描述的智能体通常是协作型的，而研究的目的在于给出多智能体协作的形式化模型[11-15]。

另外一类研究往往运用面向对象[16-18]、并行计算[19-20]或基于知识[21-22]的方法来描述和构建智能体，在多智能体系统中则采用博弈论、信息经济学、社会学等领域的方法对智能体的行为进行描述与控制，并采用这些方法来控制智能体之间的协商与协调，或控制智能体在系统中的行为。这类研究主要用于刻划自私智能体(只考虑自身利益的智能体)在一个多智能体系统中的行为，或者研究如何让自私的智能体对其所处的系统(或社会)作贡献，这些研究已取得了很多成果[23-30]。

1.1.2 移动智能体的产生

Agent与现在流行的软件实体（如对象、构件）相比，它的粒度更大，智能化程度更高。随着网络技术的发展，可以让Agent在网络中移动并执行和完成某些功能，这就是移动Agent的思想[4]。

20世纪90年代初由General Magic公司在推出商业系统Telescript时提出了移动Agent的概念。移动Agent是指能够迁移的Agent，即具有跨地址空间持续运行

的Agent，简单地说，是一个正在执行的程序，能够在异构网络环境下从一台主机迁移到另一台主机上并且持续运行[31-33]。移动Agent在分布式环境下有以下特点：

①移动性：移动Agent不依赖于操作系统和平台，可以从一台机器移动到另一台机器。Agent通过移动到达需要处理的信息源，激活本地资源，减少中间数据在网络上传输，节省带宽和延迟。即使网络连接中断，Agent仍可以继续执行；

②自主性：移动Agent可以控制自身的行为。移动Agent具有独立的局部于自身的知识与知识处理方法，对遇到的事件自主采取行动；

③反应性：移动Agent可以对环境变化做出感知和应变。它可以根据将要执行的任务和当前网络状态采取行动，即发送多少Agent，发送到哪儿，这些Agent是否将迁移或保持静态，它随网络的变化而变化；

④异步操作性：移动Agent可以独立于用户和其它Agent，执行自己的操作。它可以从某一站点被发送出去，到达另一站点后激活，异步自主地操作；

⑤协作性：移动Agent具有合作求解和管理通讯的能力。它可以通过一系列移动Agent或发送子Agent到其它机器上，与当地静态Agent和远程资源连接，实现分布任务动态并行计算；

⑥学习能力：移动Agent具有利用获得的关于环境的信息，调整和修改自己行为的能力；

⑦安全性：指对移动Agent本身及它的运行环境的安全性保障，体现为移动Agent及Agent运行环境抵御恶意攻击和无意破坏的能力。

1.1.3 移动智能体系统

移动Agent系统是由在网络平台上合作完成某个任务的多个移动Agent以及网络环境里涉及到该任务的其它软、硬件资源共同构成的。在移动Agent系统中，向一个移动Agent系统提交一个任务，形式上表现为用户将任务的处理逻辑和Agent迁移逻辑封装在一个或多个Agent中并派遣出去，在网络上漫游，自主寻找合适的资源，代表用户完成任务，任务完成之后，再通过回收或者接受携带了结果的Agent得到任务结果。因此，任务的完成过程其实就是移动Agent的派遣，在网络上通过漫游、计算、协同而不断产生状态变化并最终回送任务提交者的过程，如图1-1所示。这个完成任务的过程中，有二项关键技术。其一是支持移动Agent与外界，包括各任务相关节点上的操作执行环境，用户和其它Agent等的通信机制，其次是移动Agent在网络节点间迁移的机制。分别阐述如下。

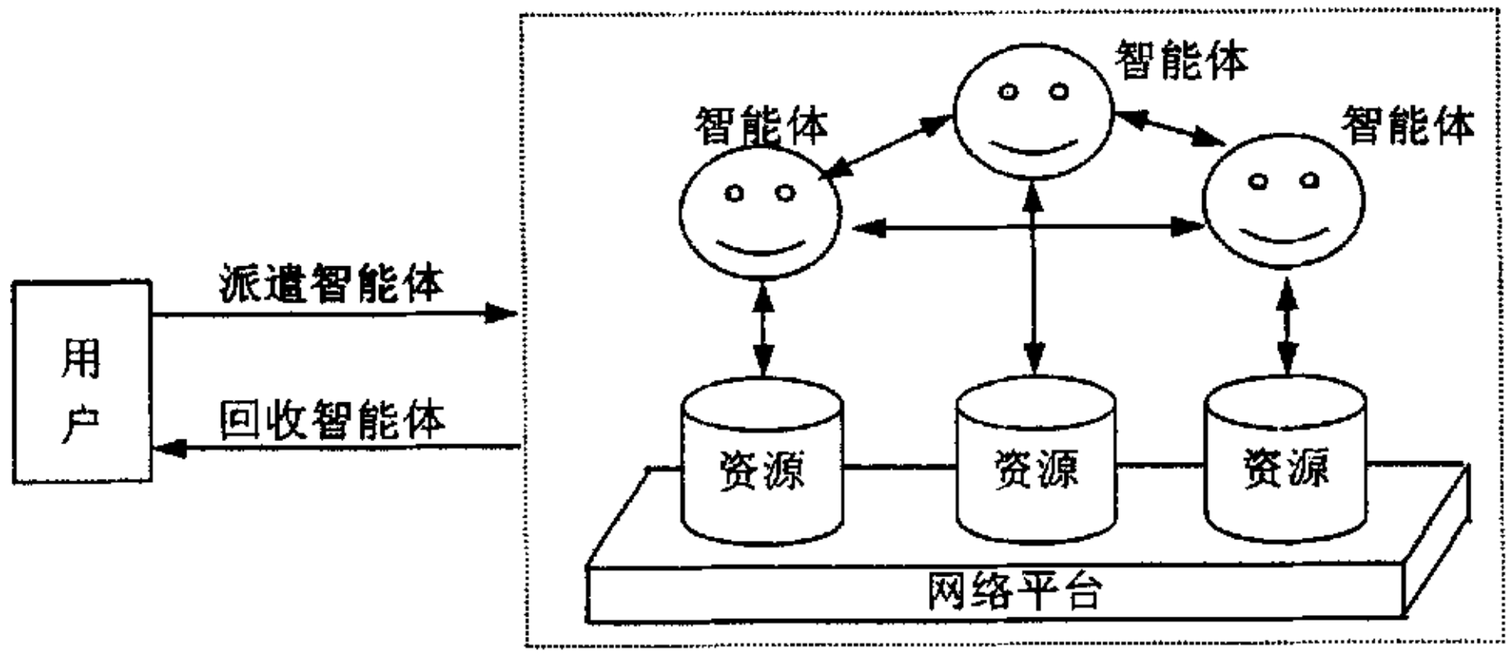


图1-1 移动智能体系统示意图

① 移动Agent的通信

移动Agent的通信是Agent之间进行协作的基础方式。移动Agent系统中，通信对象主要是移动Agent和环境资源，通信方式主要有局部通信和跨节点远程通信两种，一个良好的移动Agent系统中，大部分的通信应为局部通信。获得局部通信的主要手段是进行Agent的迁移，移动Agent通过迁移可以将大量远程交互变为局部交互，如图1-2所示。

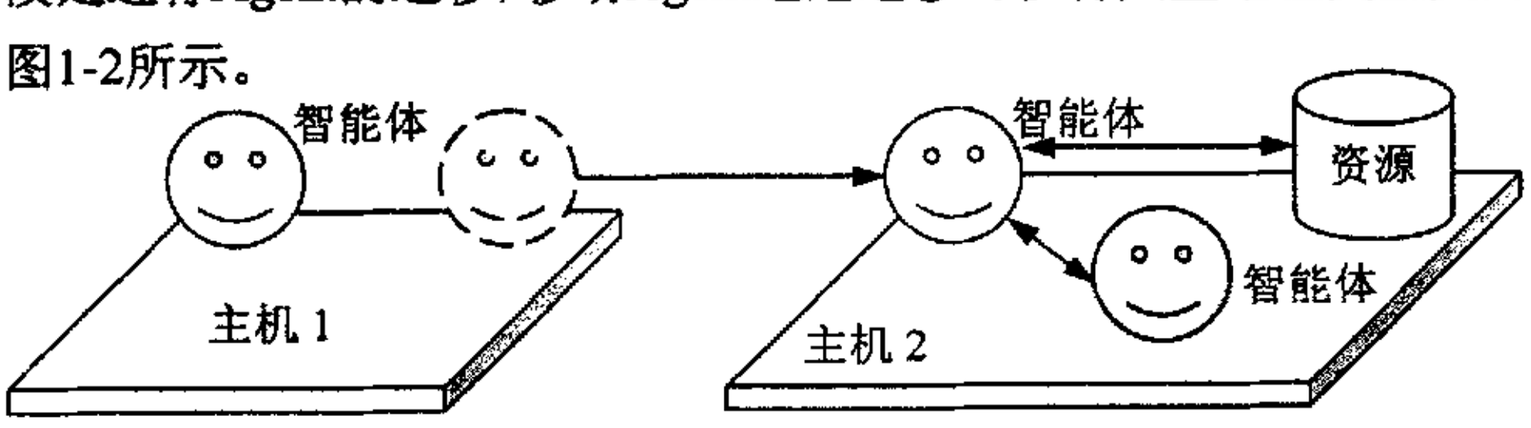


图1-2 移动智能体系统中的局部通信

② 移动Agent的迁移

移动Agent的迁移特性赋予了移动Agent系统两方面的优越性：由于移动Agent可以在网络上自主寻找合适的网络资源，因此移动Agent系统的效率更高；移动Agent可以感知网络环境的变化，这使得移动Agent系统具有良好的网络适应性，可以动态地根据网络环境的变化调整移动Agent系统的配置。

移动Agent的迁移不同于单纯、简单的代码迁移，它是移动Agent在异构环境中的计算状态的迁移和恢复执行，它实现的是计算迁移。在一个移动Agent的计算迁移过程中，必须确保Agent实体位置的变迁并不影响整个移动Agent系统的计算完整性。这其中涉及到大量的状态捕获、封装、迁移和恢复工作。其迁移模型如图1-3所示。

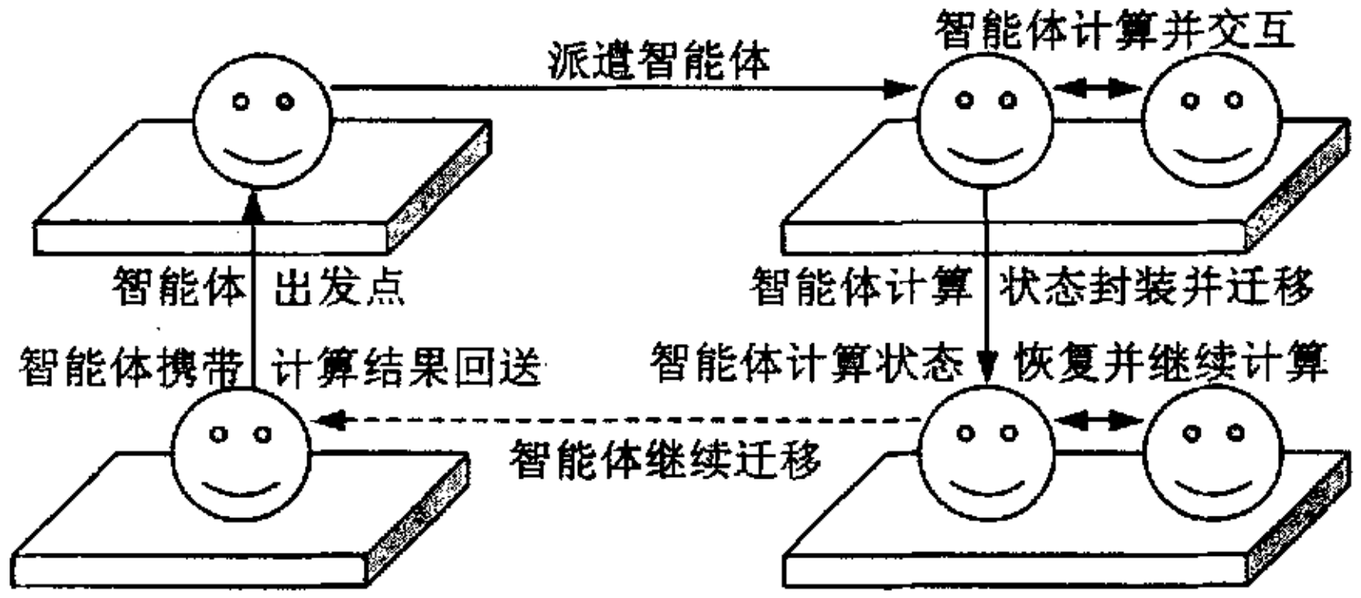


图1-3 移动智能体的迁移

1.2 移动智能体技术研究综述

1.2.1 移动智能体的研究概况

- 移动智能体代表性系统[3-4]

移动Agent系统的研制和开发引起了广泛兴趣，迄今已经商化的平台和应用系统不下数十种，以下只列举若干有代表性的系统，可见其技术水平和各自应用的普适性。

Telescript[31,34]和**Odyssey**[32]：1995年General Magic公司首次提出了移动Agent概念并申请了专利。Telescript是其第一个商用的移动Agent系统。它为移动Agent提供了指令级的移动机制，这是一个功能很强的移动机制，它使得移动Agent可在任意点上中断执行，移动到目标站点，然后再恢复运行。因为Telescript基于专有的语言和复杂的支持机制，难以推广。随着Internet的普及和Java语言的出现，General Magic公司在Odyssey系统中用JAVA语言重新实现了Telescript中的概念，建立了一个JAVA类库，开发者可在此基础上建立自己的移动Agent应用。该系统支持IIOP（Internet Inter Orb Protocol）和分布式组件对象模型（Distributed Component Object Model，DCOM），提供审计和跟踪机制支持简单的Agent调试，采用“开会”和“散会”方式支持Agent的通信协同。

Aglets[35-36]：Aglets（Agent和Applet的缩略语） Workbench是IBM公司开发的基于Java的移动Agent系统，是目前最流行的移动Agent系统之一。其主要特点是实用性，易于安装，界面友善，便于使用。Aglets提供了较为灵活的移动机制，并采用事件机制来增强其移动机制的能力。在通信机制方面，Aglets提供了简单但相对完整的命名和通信机制，命名机制保证了Agent的命名唯一性，通信机制提供了丰

富的同步和异步通信机制，在移动Agent的安全性方面，Aglets制定了一套安全策略，并主要借助事件机制中的触发回调功能来保证其安全性。

Aglets核心概念有三个：智能体Agent、代理机Proxy和内容Context。在Aglets中，一个Agent由Agent核和Agent Proxy构成，Agent核是Agent的数据和处理逻辑，决定了Agent的能力。Agent Proxy是Agent核的一层封装，任何对Agent核的访问都必须通过Agent Proxy的中转才能完成。同时它还提供寻址和安全保护方面的支持。Agent Context是Agent存在和运行的环境，提供对所有的底层操作系统、网络资源以及外部软件系统资源访问的中介，Aglets生命周期的管理也通过Context来完成。

Concordia[37]：日本三菱公司Mitsubishi的Concordia是一个可进行移动Agent应用的开发和管理的架构，它可延伸到任何支持JAVA的系统上。Concordia由多个部件组成，这些部件用JAVA书写，它们组合起来构成一个完整的分布应用的计算环境。一个简单的Concordia系统包含一个标准JAVA虚拟机（Virtual Machine，VM），一个Server和一组Agents。在Concordia中，移动Agent的移动由一个相对独立的实体，即旅行计划来描述，从而可在某种意义上提高移动Agent的易理解性。

Agent Tcl[38]和**D'Agent**[39]：Agent Tcl是Dartmouth学院研制的一个基于Tcl的移动Agent系统。该系统由导航和通信服务、安全保障机制以及调试和跟踪工具组成。Agent Tcl的主要部分是一个运行于各站点上的Server，该Server支持一个Agent的完整的执行状态（如局部变量、指令指针等）的迁移。当一个Agent需迁移到另一机器上时，它要调用一个“Agent—Jump”方法，该方法自动捕获此Agent的状态并将该状态发送给迁移目标Server。此后，目标Server将启动一个Tcl，载入该状态信息，从Agent的断点处恢复执行。D'Agent是在Agent Tcl基础上的一个扩充，它提供了一个简单的、开放的层次式体系结构，该结构分四层：TCP/IP等基础传输协议、Agent Server层、支持各种语言的解释器层以及应用Agent层。该结构可以同时支持以Tcl语言、Java语言和Scheme语言为编程语言的Agent的运行以及多种传输协议，并且可以方便地增加新的语言和协议。系统中驻留的支持Agent提供了应用Agent的移动支持、通信支持和安全支持。D'Agent采用Tcl解释器支持基于Tcl的强迁移，采用扩充的JVM支持基于Java的移动Agent的强迁移。

Mole[40]：Mole是德国斯图加特大学研制的第一个基于Java的移动Agent系统，于1996年完成。其最新版本为Mole 3.0，该版本的鲜明特色是其“Exactly Execute Once”。Mole提出了Agent的执行模型，Agent的执行是一个Stage的序列，Agent进入某一站点即开始了一个Stage，离开该站点则结束上述Stage。每个Stage包含一个Worker节点和若干ObServer节点。Worker节点是该Stage的主要工作节点，ObServer

节点负责对Worker节点的监测，是Worker节点的备用节点。此外，Mole还设计了消息队列缓冲机制，相邻两个Stage之间设有缓冲，当一个Agent按旅行计划准备进入下一Stage时，Agent本身（代码及状态）被送入上述消息队列中进行缓冲。下一Stage的Worker首先从该队列中取出Agent并恢复执行。与此同时，所有ObServer监视Agent在Worker上的执行，发现Worker的故障后，ObServer将通过“投票”“选举”出Worker的替身，从消息队列中取出Agent并恢复执行。在迁移模型方面，Mole 3.0的旅行计划由Itinerary Entry构成，简单Entry是一个二元组（Node，Method），表示在Node上进行Method操作。复合Entry有三种：Sequence，Set和Alternative。Sequence是[e_1，e_2，…，e_n]，表示按序执行Entry e_1，e_2，…，e_n；Set是｛e_1，e_2，…，e_n｝，表示以任意次序执行所有的e_i，Alternative（e_1，e_2，…，e_n）指在e_1，e_2，…，e_n中任选一个e_i（i＝1，2，……，n）执行。

Jumping Beans[41]：Jumping Beans是一个由美国Ad Astra Engineering Inc.设计的、支持可移动Java应用程序构建和运行的系统平台。该平台由一个中心Server和若干Host Clients组成，Jumping Beans可在各个Clients节点上移动，完成应用需求，但每次移动必须从中心Server上经过。这种体系结构简化了Jumping Beans的迁移支持、通信支持，易于管理，易于和已有的Java系统进行无缝集成，其安全性也能得到较好的保证。

ROMA[42]：ROMA是美国麻省理工学院MIT Robust Open Multi－Agent Systems Research Group的缩略语，这个小组的工作主要集中于如何在一个开放环境中提高移动Agent系统面对各种不安全因素时的健壮性，比如基于知识的异常处理等容错技术。目前，该小组的工作方向主要有：Agent系统的设计方法学、Agent协商算法、Agent之间兴趣匹配的中介技术以及处理Agent系统异常的服务设施等。

- 标准化工作[4]

（1）**MAF**

从上述几个代表性移动Agent系统来看，它们在体系结构和系统实现上都有较大差异，这种现状极大地妨碍了移动Agent的互操作及技术的推广应用。因此，Crystaliz公司、General Magic公司、GMD公司、IBM公司以及Open Group等单位共同提出了移动Agent的标准化草案（Mobile Agent System Interoperability Facility，MASIF），并引起了OMG的重视[43]。

MAF定义了Agent系统之间的接口，没有涉及Agent应用、Agent系统自身的标准化。此外，对移动Agent而言，语言互操作的难度太大，故MAF没有语言互操作

的内容，它只限于不同开发者用相同语言实现的移动Agent系统之间的互操作。因此，MAF实际上是一个在Agent系统级上，而非Agent本身级别上的接口标准化。

MAF制定了以下四个方面的标准：

①**Agent管理**：从事移动Agent的工作人员非常希望对Agent管理进行标准化。很明显，管理不同类型的移动Agent系统的管理员肯定期望这些系统能在同一标准下运行。该标准下，可以创建一个Agent，给定一个名字，然后在一个标准的模式下挂起、恢复或终止这个Agent的执行；

②**Agent转移**：Agent应用系统创建的许多Agent在不同类型的Agent系统中自由移动的能力是标准化的主要目标之一，它规定了一个公共的基础架构；

③**Agent和Agent系统的命名**：Agent系统的互操作除了要求Agent操作的标准化，其参数的语法和语义也必须标准化，特别是Agent和Agent系统命名也要标准化，它使得应用能够识别Agent和Agent系统，也使得Agent和Agent系统能互相识别；

④**Agent系统类型和定位**：只有能被某一目标Agent系统所支持，Agent才能向该Agent系统迁移，因此，Agent迁移前必须要能从异地获取目标系统的类型信息，故Agent系统的定位语法必须标准化。从Agent系统彼此定位的角度来讲，定位语法也需标准化。

（2）FIPA模型

FIPA（Foundation for Intelligent Physical Agenta）是研究智能Agent的最大的国际化组织，自1997年起，陆续推出了众多的智能Agent标准化草案，其中影响较大的如FIPA Abstract Architecture Specification，FIPA Agent Management Specification，FIPA Agent Software Integration Specification，FIPA Agent Message Transport Service Specification等，上述FIPA的Agent模型标准化规约主要围绕智能Agent之间的通信而开展的，以标准的智能Agent之间的语义互通为出发点，给出了通信技术、通信语言、通信安全等方面的基准模型，形成了智能Agent研究和系统构建的事实标准[44]。

随着移动Agent研究热潮的掀起，FIPA也意识到有必要在其基准模型中增加关于移动Agent的若干热点技术和框架的讨论，因此在2000年公布了FIPA Agent Management Support for Mobility Specification。在该文件中，FIPA给出了Agent有关移动问题的一个Ontology，定义了一些基本概念，如移动Agent，移动Agent系统，移动Agent语言等，对基本的移动功能进行了描述，讨论了移动Agent的移动生命周期模型，给出了支持Agent移动的系统最小需求和技术考虑，比如移动协议、Clone和Invocation等技术。

1.2.2 移动智能体的目前水平[45]

自从1995年General Magic公司推出Telescript系统，并系统阐述了移动Agent技术之后，移动Agent技术在世界范围内引起了学术界和企业界的广泛关注，掀起了移动Agent的研究和应用开发热潮，例如，General Magic公司的Odyssey、IBM公司的Aglets[35-36]、ObjectSpace公司的Voyager[46]、Mitsubishi公司的Concordia[37]、DartMouth学院的D'Agent[39]、MIT的ROMA、德国斯图加特大学的Mole[40]、我们自行设计和开发的IBMAS，以及国内有关研究单位在智能Agent技术方面的研究工作等。此外，有关Agent技术的标准化工作也在进行之中，例如由若干大公司向OMG提出的MAF（Mobile Agent Facility）。软件Agent技术的应用，特别是在电子商务中的应用方兴未艾，如美国Washington大学的Jango、MIT的Kasbah Marketplace[47]、Minnesoda大学的MAGMA等。

在已有进程迁移、消息传递、RPC等各项技术的基础上，移动Agent的关键技术取得了较大的发展，应用领域不断扩大，应用系统也时有报道，归结起来，可以认为，移动Agent技术及实验性系统的研究为建立基于Internet的移动Agent系统奠定了良好的基础，标准化工作的开展为Agent技术的广泛应用架起了桥梁，Agent技术的初步应用已经展示出广阔的前景。然而，移动Agent技术的研究正在进行之中，一些关键性技术尚需进一步研究，如移动Agent系统的基准模型、Agent的迁移机制、Agent之间的通信和协作、Agent及其宿主机的安全等问题。另外，标准化工作正在起步，尚缺乏较大规模的、有说服力的标志性应用。

1.2.3 移动智能体存在的问题

作为一种正在蓬勃发展中的新技术，移动Agent技术还存在不少的问题，主要包括：

- 移动Agent系统的基准模型[48]

关于移动Agent技术的研究方兴未艾，各种移动Agent系统的原型也层出不穷。然而，这些系统大多数是基于各自的系统构建者对移动Agent技术和环境的理解而建立的，缺乏共识，对构建移动Agent系统缺乏系统性的阐述，从而使得这些系统一方面自身的功能不够全面，另一方面系统之间的互操作性较差，很难臻于实用。虽然MAF规约给出了提高移动Agent系统互操作的原则性阐述，但是该规约并不全面，它主要侧重于系统的外部接口和移动Agent管理，缺乏对系统内部构件之间的接口规约，缺乏对构建一个实用的移动Agent系统的指导和建议。

- 迁移机制[49-56]

在移动Agent技术中，Agent的迁移机制是其核心技术之一。最早提出移动Agent概念的General Magic公司，在Telescript中的基于进程的迁移理论采用了语句级线程迁移模型。该模型提供了Go指令，Agent可根据需要在线程执行的任意点使用Go指令改变自己的位置。该方法中系统应承担Agent线程状态的捕获、封装、发送和恢复工作，显然负担较重，网络传输量大，尤其是绝大部分语言如JAVA、C++等并不支持线程状态的捕获。因此IBM在采用JAVA实现其Aglets系统时，设计了事件驱动方法实现语句级对象迁移模型，用户可根据需要使用“派遣”或“回收”指令进行Aglet的迁移[35-36]。迁移发生时，Aglets一方面利用JAVA的对象序列化机制，另一方面还需要用户在迁移前根据需要进行必要的线程状态收集工作，以帮助迁移后的线程恢复执行。因此，Aglets提供了若干帮助完成上述工作的方法（如：OnDispatching()，OnArrival()，OnReverting()），并允许用户进行灵活修改。和Telescript相比，该机制虽然采用对象迁移取代了Telescript中的线程迁移，降低了系统负担，但其将实现语句级迁移所必须完成的线程状态的迁移工作转加给了用户，增加了用户负担。此外，从用户使用的角度看，以上述模型为代表的语句级迁移模型还存在不足：Agent迁移条件和动作都是隐含在Agent的代码之中，所以，Agent功能体和Agent的迁移是混合在一起的。对一般程序员来讲，若不对Agent本身及Agent移动条件有深入的了解，则Agent功能及移动的设计、测试和调试都有一定的难度，尤其是在Internet环境中，更是难以管理和控制。如果移动Agent的迁移信息能从Agent功能体中分离出来，用一种有足够能力的结构进行描述，即可在不降低迁移描述和处理能力的同时，方便地实现过程级对象迁移，减轻系统和用户负担，方便使用。在这一点上，Concordia系统的Itinerary模型提出了旅行计划的概念，该旅行计划是一个独立于Agent本身的数据结构，储存并维护Agent的移动信息。它由多个目的地的描述组成，每个目的地描述了Agent的移动目标位置及Agent恢复时所需执行的动作。Concordia虽然可实现迁移信息和Agent功能体的分离，但其旅行计划的描述信息较少，也未提供对旅行计划进行动态变动的手段，因此，其能力和灵活性都不够，不能有效地表达各种迁移现象。

- 通信机制[57-62]

现有通信机制存在以下不足：

①对移动 **Agent** 的通信支持不够全面。一方面，知识交换以及消息传递的工作实际上是分离的，没有同时支持 KQML 和消息传递两种协作方式的通信机制；另一方面，在实际应用中，移动 Agent 之间的信息交流是多级别、多层次的，因此，我们认为一个系统的移动 Agent 通信机制应支持 Agent 之间开展不同层次的协

作通信。

②对通信可靠性的支持不够。网络环境的多变性，Agent本身也在不断运动，在一个多变的环境中，运动着的 Agent 之间很容易产生各种通信异常，如安全问题、容错问题、通信失效问题、通信因果顺序问题等。这些问题如不解决，移动Agent之间的协作便不可能顺利进行。

- 安全性问题[63-75]

在移动 Agent 提出后不久，人们就意识到安全问题，开始着手研究，借鉴了现代分布式系统中的许多技术和方法，根据移动 Agent 的特点，进行了改进和扩充，取得了一定进展，提出了许多提高移动 Agent 系统安全性的方法，其中有些已经应用到实验系统中去了。其代表性的工作主要有：基于软件的错误隔离、数字签名技术、携带证明的代码（Proof Carrying Code）、用加密的函数进行计算、混淆代码（Obfuscated Code）等工作。但是，这些方法大多仍处于研究状态，其实用性不强，而且有些问题用目前的技术还难以解决，需要探索新技术。

- 典型的应用研究[56,76-84]

移动Agent技术是一种新兴的网络平台上的分布计算技术。它在网络安全、分布式信息检索、电子商务、个人助理、并行处理等方面具有广阔的应用前景，需要开展典型的应用研究。

1.3 本文研究工作的动机

1.3.1 研究工作的动机

除了跟踪国际技术发展前沿，试图掌握有关智能体的理论和方法的求知渴望外，国内相关的技术研发需求也促使作者为此付出努力。实际需求背景可举：2000年以来作者所在的研究室面临的B2B，B2C乃至C2C电子商务应用需求，以及企业和政府机构解决网络与信息安全、主动防御系统设计等需求。作者参加了多项预研和实用原型系统的研究与开发。另外，本文是在实验室多年来在继面向对象方法OOM，人工智能AI嵌入OOM的研发平台AIOOM和继而形成的面向智能体平台AOPP基础上基于Agent技术的新进展，也是继唐超博士论文面向应用需求的软件智能体系统研究与开发的基础上完成的进展性研发工作，重点在移动智能体系统的理论方法和应用。具体有下述各项工作：

① 入侵检测系统（Intrusion Detection System，IDS）是近年来出现的新型网络安全技术，它可以弥补防火墙的不足，为网络安全提供实时的入侵检测及采取相应的防护手段。但是，传统的入侵检测系统存在如下问题：中心主机是一个单

一失效点；检测系统的可伸展性受到很大限制；检测系统缺乏灵活性和扩展性。移动Agent技术实现的分布式IDS可以克服传统IDS的缺憾。

② 漏洞扫描系统能发现并评估系统存在的弱点和漏洞，据以估计安全风险，建议补救措施。然而，当网络规模较大，结构复杂时，其远程执行有挤占带宽、效率不高且不能有效执行的缺点。基于移动Agent技术实现的移动扫描系统，可以实现有免疫作用的主动漏洞扫描。

③ 智能信息检索和个性化智能信息检索系统是帮助人们快速获取信息的有效手段。然而，现有系统仍然存在如下一些缺陷或不足：非个性化检索方式适应用户兴趣变化的能力较差；没有综合利用个性化检索和集中浏览的优点；用户与检索系统的交互方式比较单调；缺少分布式智能信息检索和适应信息源信息变化的能力。移动Agent可充当人类的网络知识获取助手。

④ Web技术作为当今电子商务框架的主流技术，推动着电子商务的发展。然而，Web技术并非完美无缺，其最大的不足在于Browser/Server结构的静态性。其存在以下问题：市场结构是封闭和集中式的，市场之间难以提供信息和服务，难以适应急剧变化的交易数量和种类繁多的交易规则；就客户使用而言，面对全球日增1%的Web主页，想找到理想的商务主页将如大海捞针，无从下手。因此，从宏观上讲，并不太易于使用；就商业机构而言，为满足客户对自身主页的随机访问和及时的商务处理将付出较大的代价去维护网络及站点的可用性。网站的健壮性难以得到保证；静态的Web信息缺乏主动性和交互性。Web本身并未提供更先进的“电子”手段来主动开拓市场。这样的Web可能很快就被淹没，其网络市场开拓能力是受限的。

为了有效解决上述问题，我们采用了移动Agent技术[85-108]。它具有性能优越和结构灵活等优势，该模式下开发的网络应用有以下优点：可有效降低网络负载、可支持移动设备的网络断开操作、支持分布并行计算、具有自然异构性、具有较高的环境应变能力、具有较高的健壮性和容错能力。现有的移动Agent技术的应用领域和范例虽然都可以使用传统的技术加以实现，但是越来越多的移动Agent技术研究人员认为：移动Agent技术提供了一种灵活的基于Internet的分布式计算模式和分布式程序设计范式，目标不在于完全替代现有的技术，而在于借助其高性能和高灵活性，充分发挥Internet的潜能。本文试图在移动Agent的系统模型、关键技术和典型应用等三方面开展研究、设计和应用。

1.3.2 研究工作的意义

（1）提出了一个移动Agent系统的基准模型：在系统分析了移动Agent技术、环境和现有的多个移动Agent系统的基础上，该基准模型给出了移动Agent技术的公共概念集，从实体、服务和接口三方面阐述了移动Agent系统的基本成份、各成份的基本功能以及各成份之间的关系。为开展移动Agent技术的研究和探索、系统的设计和实现提供了一个良好的基准模型。

（2）提出了一种新的移动Agent系统的结构化迁移机制：在理解和分析已有移动Agent迁移机制的基础上，设计了一种新的结构化迁移机制，该机制的结构化主要体现在以下三个方面：①Agent的旅行计划和功能体完全分离，便于Agent的开发、理解、复用和维护；②旅行计划本身也具有严格定义的结构，便于旅行计划的开发、使用；③提供了两种灵活有力的迁移模式，大大增强了迁移机制的表达能力。在此基础上开展了结构化迁移信息描述语言的形式化研究，从语法范畴、语法规则和动态语义三方面，给出了基于互联网的移动智能体系统（Internet Based Mobile Agent System，IBMAS）迁移逻辑描述语言SMDLan的结构化操作语义框架。上述特点使得这种结构化的迁移机制具有较强的描述能力、复用性和用户友善性，为以后开展的后续研究奠定了良好基础。

（3）提出了一种分层多模式通信框架：该框架中的KQML模式提供高层的基于知识交换的KQML协作，间接访问模式以消息传递为通信手段，直接访问模式允许Agent在直接进行基于网络协议的数据传输。在间接访问模式下，就移动Agent特有的通信失效现象，提出了完整的解决方案。总之，上述工作使得分层多模式通信框架具有易于使用、效率较高、可靠性较好等优点。

（4）提出了基于信任传递的授权模型：该模型和IBMAS中的访问控制、安全通道、密写操作等功能，共同构成了一套较为完善的移动Agent系统的安全体系。

（5）设计并实现了IBMAS：结合上述工作，设计并实现了一个IBMAS，它能够对移动Agent系统进行全面支持。IBMAS不仅提供了移动机制、通信机制和安全机制，而且还建立了Agent的开发、运行和监控环境，从而便于用户在此基础上进行应用程序的开发和实际的应用。

（6）开展了一系列应用研究：在IBMAS的基础上，研究了移动智能体技术在网络安全和互联网上信息检索系统中的应用。

1.4. 本文的内容安排

本文内容分8章：

第一章绪论，对智能体、多智能体和移动智能体的发展历程进行综述。介绍了移动Agent的研究现状和研究需求。阐明作者的工作动机和思路，并对论文的内容组织作了简要说明。

第二章是移动智能体系统的基准模型，首先介绍了软件智能体的定义，并提出了软件智能体的理论模型。其次，介绍了基准模型的定义，并在分析移动Agent技术、环境和现有的多个移动Agent系统的基础上，给出了一个基于Internet的移动Agent系统的基准模型，该基准模型给出了移动Agent技术的公共概念集，并从实体、服务和接口三方面阐述了移动Agent系统的基本成份、各成份的基本功能以及各成份之间的关系。

第三章是互联网上移动智能体系统的设计，主要介绍基于Internet的移动Agent系统IBMAS的设计工作，并重点阐述了一种新的移动Agent结构化迁移机制。

第四章是移动智能体系统关键技术通信机制和安全机制的设计，提出了一种移动Agent的分层多模式通信框架及通信失效解决方案。并简单介绍了IBMAS中的安全体系及基于信任传递的授权模型。

第五章在详细讨论传统入侵检测系统的基础上，提出了一种基于移动Agent的入侵检测系统MABIDS，并对其实现的关键技术进行了讨论和总结。

第六章主要介绍了移动扫描智能体系统的实施策略、技术方案和应用实例。

第七章在评述了现有信息服务系统的缺陷和解决方案基础上，提出了一种基于移动Agent的信息检索系统IRS-Agent，并与相关信息检索系统进行了比较，同时给出了一个实例。

最后，在本文的第八章中，给出了全文的总结以及以后的工作考虑。

2 移动智能体系统的基准模型

移动智能体技术对于未来的分布式系统的设计、实现和维护具有重要意义，它可有效地降低网络拥挤、适应网络的动态变化，其异步性与自主运行特性提高了分布式系统的健壮性和容错性[109-116]。移动 Agent 技术将成为未来主流的分布式计算模式。关于移动 Agent 技术的研究方兴未艾，各种移动 Agent 系统的原型也层出不穷[31,34-40,46]。然而，这些系统大多数是基于各自的系统构建者对移动 Agent 技术和环境的理解而建立的，缺乏共识，对构建移动 Agent 系统缺乏系统性的阐述，从而使得这些系统一方面自身的功能不够全面，另一方面系统之间的互操作性较差，很难臻于实用。虽然 MAF 规约给出了提高移动 Agent 系统互操作的原则性阐述，但是该规约并不全面，它主要侧重于系统的外部接口和移动 Agent 管理，缺乏对系统内部构件之间的接口规约，缺乏对构建一个实用的移动 Agent 系统的指导和建议[4]。因此，在以上工作的基础上，结合我们对移动 Agent 技术和系统的研究，给出了一个移动 Agent 系统的基准模型（Norm Model of Mobile Agent System，NM_MAS），该模型阐述了基于 Internet 的移动 Agent 系统、系统中涉及的公共术语及其解释、通用的体系结构、标准的服务和接口。

本章共分 6 节：第 1 节介绍了软件智能体的定义，并提出了软件智能体的理论模型。第 2 节介绍了一个基准模型的定义，并列举了具有代表性的基准模型；第 3 节给出了移动 Agent 系统基准模型中的基本概念、移动 Agent 模型和移动 Agent 服务环境模型；第 4 节详细阐述了移动 Agent 系统的基准模型 NM_MAS；第 5 节将本文工作和相关工作进行了比较；第 6 节进行了小结。

2.1 软件智能体的理论模型

本节内容基于唐超博士的工作，再加整理。我的研究工作着重在移动智能体系统的理论方法和应用。

2.1.1 软件智能体的定义[117]

软件智能体的本质是一段具有智能性、自主性、社会性、反应性、能动性的计算机程序，它的根本目标是接受另外一个实体（可以是用户、计算机程序、系统或机器等）的委托并为之提供帮助和服务，能够在该目标的驱动下主动采取包括通讯、学习或推理等手段在内的各种智能行为，以感知、适应外界环境并对动态环境的变化进行适当反应。因此，智能体在具体软件实现上可以是对象（即使非常简单），可以是组件，还可以是一个计算机程序，关键是在一个特定的应用环

境中通过其外在行为表现，实现接受委托并提供服务。

在这个定义中，主要强调了以下三点：①接受外界实体的委托并为其提供帮助和服务；②感知、适应外界环境的变化；③在目标的驱动下，主动采取包括通讯、学习或推理等手段在内的各种智能的行为。

2.1.2 智能体基础模型[117]

- 抽象智能体构建问题

智能体能够感知、适应外界环境并对动态环境的变化作出适当反应，从感知和适应的角度上看，它是一种拟人化的软件实体，具有认识世界和在一定目标驱动下改造世界的能力。就认识世界的过程而言，即使相同的世界在不同人看来也是存在差异的。因此，在建立智能体模型之前。考虑并区分两种方式的外部世界模型。

① 全知世界观测模型：观测者对外部世界的特征和变化规律具有深刻的理解，这种理解建立在对外部世界的良好知识的基础上。

② 智能体世界观测模型：智能体由于计算资源和计算能力的限制，对外部世界的特征和变化规律只能依据不完备的信息和知识，因此对外部世界的观测可能是局部的、片面的，甚至是不完全正确的。

我们在确定智能体的实现目标时，采用的是第一种世界观测模型，因为设计者的目标是按照设计者的要求改变世界的状态；在设计智能体的感知和对外界环境产生的行为时，智能体采用的是第二种世界观测模型。因此，在智能体的模型中定义了两种概念：第一个概念是智能体外部表现行为（简称表现行为），是一种从智能体的外部行为表现来看的按照设计者要求改变世界状态的行为。第二个概念是智能体规划行为，是一种从智能体的内部考虑的策略选择和执行行为。

外部世界由一系列对象构成 $W=\{w_1, w_2, \cdots\}$。每一个对象具有一些属性，这些属性是时间的函数：$f_{ij}(t)$。因此，世界的状态可以表示为这一系列对象的属性集合，世界状态也是时间的函数。一个事件 α 定义为对象的属性在$[t_{start}, t_{end}]$时间段内的变化，由于事件的出现，世界不断演进。如果能够清楚地确定事件的原因，那么称这种事件为行为。例如：我们能清楚地确定某种事件是由智能体引起的，实质上是指由智能体的行为引起的。对于其它我们所不能确定原因的对象世界属性的改变，称为事件。本文假设行为或事件都是瞬时的，即 $t_{start} = t_{end}$；

因此某时刻 k 的事件可以表示为：$\alpha_k = \Delta_k f_{i1}, \Delta_k f_{i2}, \cdots$；其中 $\Delta_k f_{xy} = f'_{xy} - f_{xy}$；$f_{xy}$表示对象 w_x在事件发生前属性 y 的值，f'_{xy}表示对象 w_x在事件发生后属性 y 的值。

我们把智能体看成是对象世界的一个对象 A∈W，从它的外部行为表现上看它可以产生离散行为 α 。所以，智能体行为函数（Agent Behavior Function，ABF）可以表示为：

$ABF: (W^* \times A^*) \rightarrow \alpha^*$

其中：W^*表示 W 的幂集，A^*表示 A 的幂集

智能体只能产生有限的行为 C（A），称为智能体的能力：

$C(A) = \{\alpha \mid \exists\ W \subset W^*,\ \ ABF(W, A) = \alpha\}$

定义 1：如果对象世界的子集使得布尔函数 B：W→ {True, False} 为真，那么称这个子集为布尔函数 B 的目标世界。

如果在一个世界中，所有的事件都是智能体的行为的结果，我们称这个世界是一个控制的世界；如果存在有不是由于智能体行为引起的事件，我们称这个世界是一个弱非控制的世界；如果试图产生的行为 α 与世界变化的事件 α' 不同，即 $\alpha' \neq \alpha$，我们称这个世界是一个强非控制的世界。

定义 2：如果智能体具有相同的智能体行为函数，那么称智能体是等同的。

定义 3：抽象智能体构建问题是寻找一个智能体行为函数 ABF（W，A），在有限时间内从任意状态 W 到达另外一个状态 W_{goal}，这种状态使得 B（W_{goal}）=True。

- 智能体基础模型

我们所说的抽象智能体是建立在对外部世界全知的基础上的，相对于智能体，设计者是外部世界全知的（设计者只有对外部世界的特征和变化规律具有深刻的理解，才能用足够精确和有限的模型来描述外部世界并设计智能体）。下面我们定义智能体的内部行为函数是建立在智能体所能获取的世界观测模型基础上的，同时智能体所面对的是强非控制的世界。

设智能体以知识表达的形式反映外部对象世界，M＝ {m_1，m_2，…}，其中每个对象都近似地表现对应的外部对象世界的对象 $m_{wi} \approx w_i$；称 M 是外部世界的一个面向对象模型，下称世界模型（Model of World）。

定义 4：称函数 K：（W^*）→M 为知识表达，如果函数把外部对象世界的对象集映射为面向对象模型中的对象。

由于智能体观察外部世界受到世界模型的限制，并且只能获取到有限的信息，所以我们不能采用定义 1 中的建立在外部世界上的判断函数 B，因此我们在世界模型的基础上重新定义布尔函数 G：

$G: M^* \rightarrow \{True, False\}$

G 是定义在世界模型上的目标判断函数。类似的，我们重新定义智能体行为

函数为智能体策略函数（Agent Strategy Function）：

$S:(M^* \times G^*) \rightarrow \alpha^*$

定义 5：智能体基础模型 AM_0 是一个三元组（G，S，M），其中 G 是目标判断函数，S 是智能体策略函数，M 是世界模型。如图 2-1 所示：

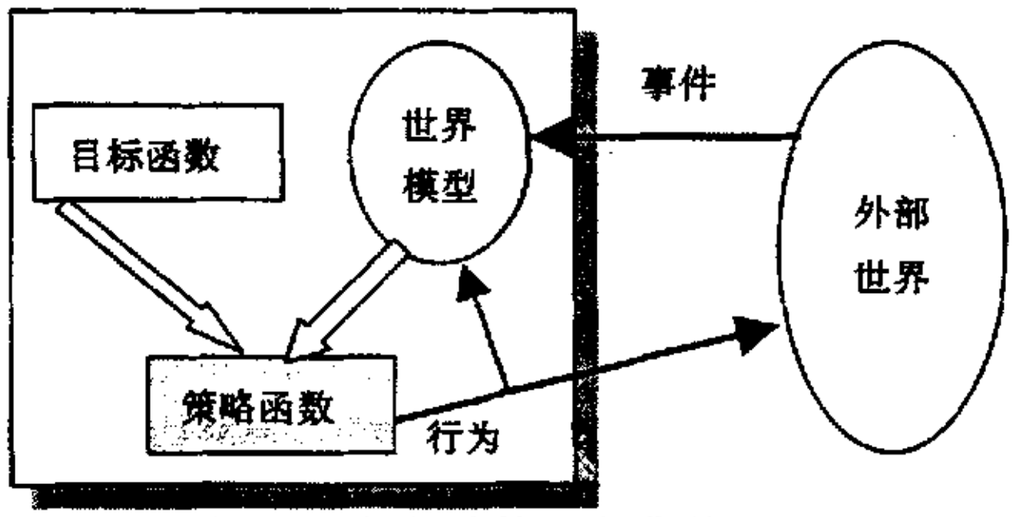


图 2-1 智能体的基本模型 AM_0

定义 6：智能体构建问题是建立智能体的世界模型和在世界模型基础上的目标判断函数，并寻找一个智能体策略函数，在有限时间内使得目标判断函数为真。

定义 7：对于智能体基本模型 AM＝（G，S，M），智能体策略函数 S'是对智能体策略函数 S 的替换，如果：S（M，G）＝S'（M，G）。

2.1.3 智能体理论模型 AM_n—考虑外部世界状态[117]

从实现的角度上看，AM_0 模型中的智能体策略函数适用于应用问题较简单的情况。一般情况下，在智能体的一个生命周期内，由一个函数处理所有的感知到的事件，并产生所有的行为是比较困难的。为了应付这种情况，我们可以根据外部世界模型变量的可能取值分解为不同情况进行处理，以降低实现智能体策略函数的复杂性。

定义 8：智能体的完全状态空间由世界模型中变量的全集决定。假定模型变量的值是有限和离散的，对智能体世界模型变量 $M=\{m_1, m_2, \cdots, m_n\cdots\}$ 而言，称 $m_1, m_2, \cdots, m_n$ 决定了智能体的 n 阶状态。

由此定义，$m_1, m_2, \cdots, m_n$ 的具体数值 $m_1^x, m_2^x, \cdots, m_n^x$ 决定了智能体的 n 阶状态 X；

定义 9：智能体的 n 阶策略函数定义为：

$S_x^{(n)}(\{m_{n+1}, m_{n+2}\cdots\}, G) = S^{(0)}(\{m_1^x, m_2^x, \cdots, m_n^x, m_{n+1}, m_{n+2}\cdots\}, G)$

n 阶策略函数是零阶策略函数在 $m_1, m_2, \cdots, m_n$ 为固定值的情况下的投影，或者说零阶策略函数由一组 n 阶策略函数构成，这组函数的个数由 $m_1, m_2, \cdots, m_n$ 离散值空间的组合决定。

定义 10：智能体模型 AM_1 是一个四元组（G，$S^{(1)}$，M，F），其中 G 是目标判断

函数，M 是世界模型 M= $\{m_1, m_2, \cdots\}$，其中 $m_1 \in \{m_{11}, m_{12}, \cdots, m_{1n}\}$；$S^{(1)}$是一组一阶智能体策略函数，$S^{(1)} = \{S_1^{(1)}, S_2^{(1)}, \cdots, S_n^{(1)}\}$。对于模型变量的每一个取值 m_{0i}，决定了智能体的一个一阶智能体策略函数 $S_i^{(1)}$。有限状态机转移函数 $F^{(1)}$：$(M^* \times G^*) \rightarrow m_1$ 决定模型变量 m_1 变化的事件。

在智能体模型 AM_1 中模型变量 m_1 决定了智能体的一个一阶智能体策略函数，有限状态机转移函数 $F^{(1)}$ 描述根据模型变量 m_1 变化的事件实现智能体策略函数的变化。从实现的角度看，希望有限状态机的状态尽可能地越少越好，同时状态的转换在逻辑上具有连续性。

定理 1：智能体基本模型 AM_0 的世界模型中有一模型变量 m_1，$m_1 \in \{m_{11}, m_{12}, \cdots, m_{1k}\}$，那么存在一个 S' 是对 AM_0 中智能体策略函数 S 的替换。

证明：构造映射 ϖ：$(M^* \times G^*) \rightarrow m_1$

$$S(M, G) = \begin{cases} S_1'(M, G) & \text{if } \varpi(M, G) = m_{11} \\ S_2'(M, G) & \text{if } \varpi(M, G) = m_{12} \\ \cdots & \\ S_k'(M, G) & \text{if } \varpi(M, G) = m_{1k} \end{cases}$$

其中：$S_i' = (\{m_2, m_3 \cdots\}, G) = S(\{m_{1i}, m_2, \cdots\}, G)$；$i \in \{1, 2, \cdots, k\}$；$m_1 \in \{m_{11}, m_{12}, \cdots, m_{1k}\}$，由智能体 n 阶策略函数的定义可知，$S_i'$ 构成一阶策略函数 S'，S' 即为智能体策略函数 S 的替换。证毕。

定义 11：两个智能体如果具有相同的世界模型，目标函数和策略函数，就认为这两个智能体是相同的。

如果把智能体考虑为对象聚合而成，每一个元组对应一个对象，AM_1 模型比 AM_0 模型将增加有限状态机对象，以完成策略函数的转换。

定义 12：采用递归的方式来定义 n 阶智能体模型 AM_n，对于 AM_{n-1} 模型中的任一(n-1)阶策略函数 $S^{(n-1)}$ 和模型变量 m_n 的一组取值，在模型变量 m_1，m_2，m_3，…，m_{n-1} 为固定值的条件下，产生一组 n 阶策略函数 $S^{(n)}$。如图 2-2 所示：

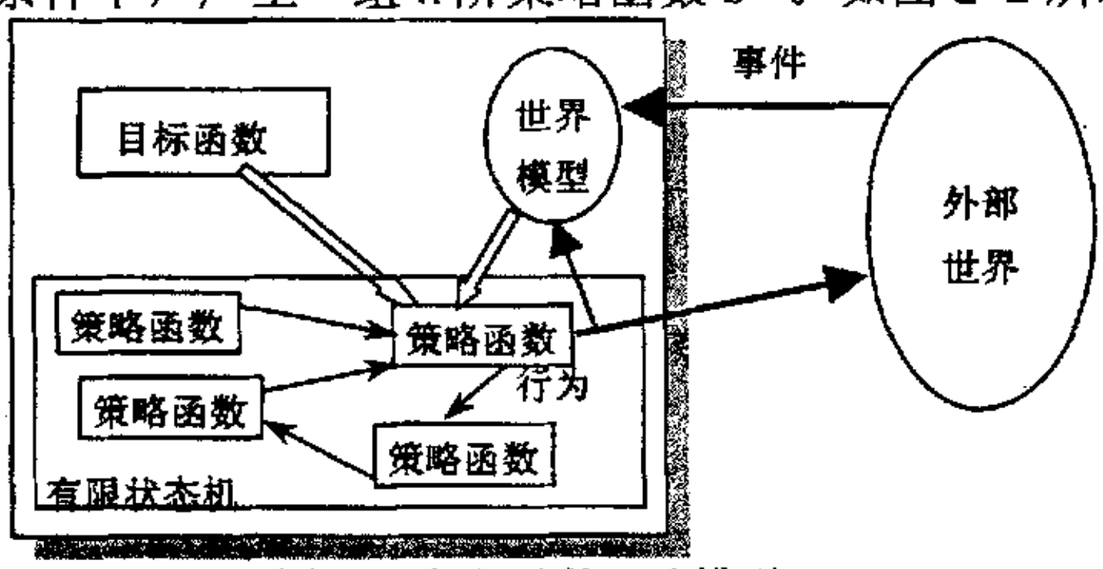


图 2-2 智能体的理论模型 AM_n

推论 1：智能体基本模型 AM_0 的世界模型中有 n 个模型变量 m_1，m_2，…m_n，$m_i \in \{m_{i1}, m_{i2}, \cdots, m_{ik}\}$，$i \in \{1, 2, \cdots n\}$。那么存在一个 S' 是对 AM_0 中智能体策略函数 S 的替换。

讨论：智能体理论模型 AM_n 反映了工程实践中人们解决问题的办法，对于一些复杂的系统，人们根据不同系统状态，采用不同的应对措施。比如对于一些非线性系统，可以采用分段线性化的方法，根据系统运行的状态用不同的线性函数来处理非线性系统，AM_n 能够反映这样的处理复杂问题的方法。即把 n 个智能体模型变量的值域空间进行分割，由一组有限的 n 阶策略函数分别进行处理。

2.1.4 智能体理论模型 AM_n^{mp}—考虑任务分解和执行时序[117]

对于复杂的系统，除了根据系统所处的不同状态，采用不同的策略函数实现问题求解外，还可以把系统分解为子问题，串行或并行的求解。对于 AM_n 模型来说，它是把 AM_0 模型中的唯一策略函数分解为多个在某一时刻只能执行其中之一的策略函数。实际上在某一时刻可以并行地执行子任务以完成子问题。

定义 13：称 σ 为 mp 维调度函数，mp 是一有限正整数，

$$\sigma : (M^* \times G^* \times t) \rightarrow \{0, 1, \cdots, mp-1\}$$

对于 $\forall k \in \{0, 1, \cdots, mp-1\}$；$\forall M$；$\forall G$，$\forall t$，$\exists \Delta t < t_{scheduled}$ 使得：

$$\sigma (M, G, t+\Delta t) = k$$

其中 $t_{scheduled}$ 是调度时间间隔。调度时间间隔表达了解决子问题的具体时间段，多维调度函数表达了被分解的子问题个数。

定义 14：对于多维调度函数决定的子问题的个数，称 $S^{(n)mp}$ 为 n 阶多维策略函数：

$$S^{mp}(M, G) = \begin{cases} S^0(M, G) & \text{if } \sigma(M, G, t) = 0 \\ S^1(M, G) & \text{if } \sigma(M, G, t) = 1 \\ \Lambda & \\ S^{mp-1}(M, G) & \text{if } \sigma(M, G, t) = mp-1 \end{cases}$$

其中 S^0，S^1，…，S^{mp-1} 是 n 阶策略函数。

如果一个复杂任务的求解可以分解为若干个子任务，那么就可以采用多维调度函数来解决问题。

定义 15：智能体模型 AM_n^{mp} 是一个五元组（G，$S^{(n)mp}$，M，F^{mp}，σ），其中 G 是目标判断函数，M 是外部世界模型 $M = \{m_1, m_2, \cdots\}$，$S^{(n)mp}$ 是 n 阶多维智能体策略函数，有限状态机转移函数 F^{mp} 决定模型变量变化的事件。对于多维调度函数 σ 决定

的子问题的个数 mp，有与之对应的 n 阶多维智能体策略函数 $S^{(n)mp}$ 和 n 阶策略函数 S^0，S^1，…，S^{mp-1} 以及由 m 个有限状态机转移函数 F^0，F^1，…，F^{mp-1} 构成的 F^{mp}。如图 2-3 所示。

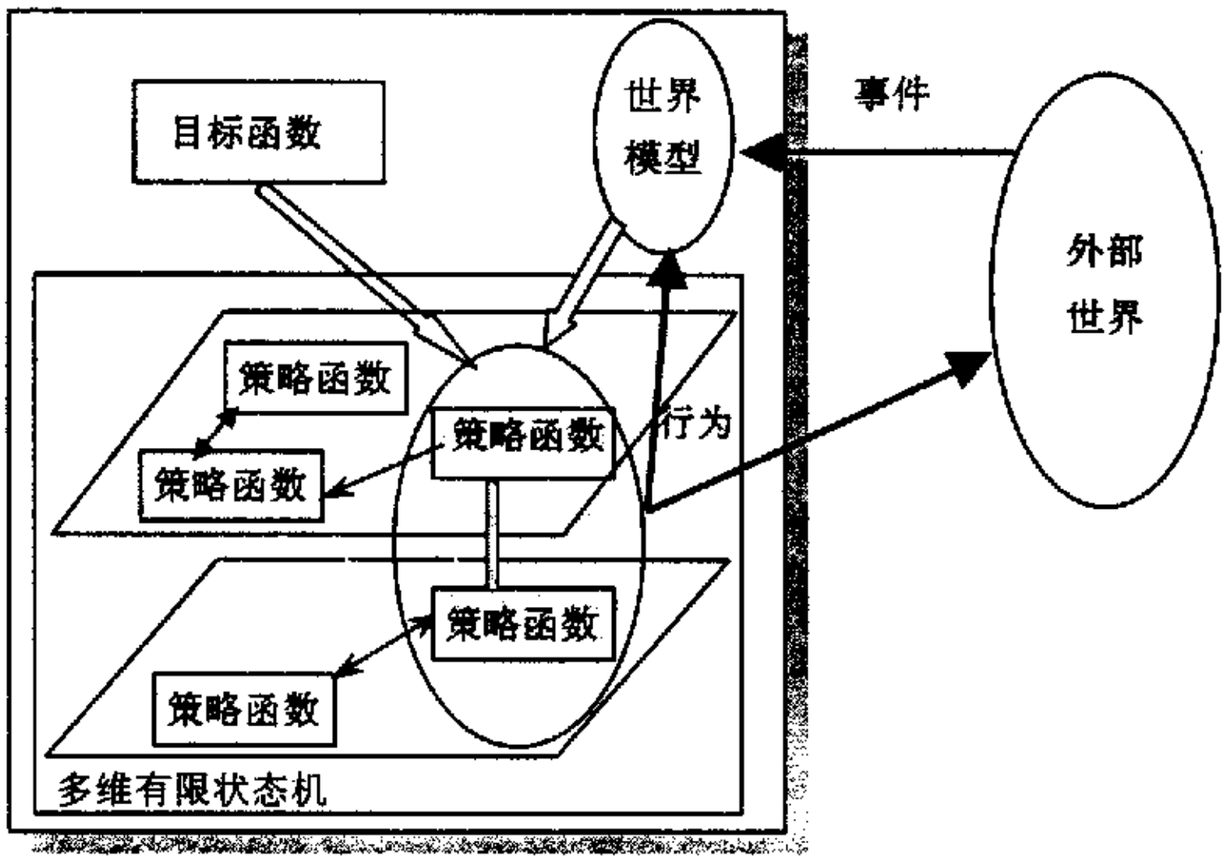


图 2-3 智能体的理论模型 AM_n^{mp}

定理 2：智能体基本模型 AM_0，如果存在 mp 维调度函数 σ，那么存在一个 S' 是对 AM_0 中智能体策略函数 S 的替换。

证明：由 mp 维调度函数 σ 的定义，对于 ∀ k∈ {0, 1，…，mp-1}；∀ M；∀ G，∀ t，∃ Δt< $t_{scheduled}$ 使得： σ（M，G，t＋Δt）＝k 。

$$S(M, G) = \begin{cases} S^0(M, G) & \text{if } \sigma(M, G, t) = 0 \\ S^1(M, G) & \text{if } \sigma(M, G, t) = 1 \\ \Lambda & \\ S^{mp-1}(M, G) & \text{if } \sigma(M, G, t) = mp-1 \end{cases}$$

其中 S^k，k∈ {0, 1，…，mp-1} 是对第 k 个子任务的策略函数；S' ＝ {S^k|k ∈ {0, 1，…，mp-1}} 即为智能体策略函数 S 的替换。证毕。

2.1.5 智能体理论模型 AM_n^{mp} —考虑网络拓扑结构[117]

智能体的发展与网络计算和分布式智能密不可分，计算由软件执行，而软件本身又可以移动。移动智能体在平衡网络负载，增强系统效率方面具有较强优势。因此，我们考虑智能体在网络空间中的不同物理地点实现问题求解。

定义 16：设 X＝ {x_1，x_2，…，x_{mb}} 是非空有限地点集合，τ ⊂ X^* 是 X 的子集族，称（X, τ）为移动空间，若 τ 满足下列条件：

① ∅ ∈ τ，X ∈ τ；

② τ中任意多个成员的并属于τ；

③ τ中任意多个成员的交属于τ。

在智能体的移动过程中，X 可以是统一资源定位符（Uniform Resource Locator)。在智能体的外部世界模型中，可以用图的形式表示智能体的移动空间。称这个移动空间（X, τ）为 m_{space}（$m_{space} \in M$），称Γ为移动函数：

$$\Gamma: (M^* \times G^*) \rightarrow X$$

定义 17：对于非空有限地点集合 $X=\{x_1, x_2, \cdots, x_{mb}\}$，称 $S_{mb}^{(n)mp}(M, G)$ 为移动 n 阶多维策略函数：

$$S_{mb}^{(n)mp}(M, G) = \begin{cases} S_1^{mp}(M, G) & \text{if } \Gamma(M, G) = x_1 \\ S_2^{mp}(M, G) & \text{if } \Gamma(M, G) = x_2 \\ \Lambda & \\ S_{mb}^{mp}(M, G) & \text{if } \Gamma(M, G) = x_{mb} \end{cases}$$

其中 S_1^{mp}，S_2^{mp}，…，S_{mb}^{mp} 是在任一地点的 n 阶多维策略函数。

定义 18：智能体模型 AM_n^{mp} 是一个六元组（G，$S_{mb}^{(n)mp}$，M，$F_{mb}^{(n)mp}$，σ，Γ），其中 G 是目标判断函数，M 是外部世界模型 $M=\{m_1, m_2, \cdots\}$，存在移动空间 $m_{space}=(X, \tau)$，且 $m_{space} \in M$，$S_{mb}^{(n)mp}(M, G)$ 为移动 n 阶多维策略函数。$F_{mb}^{(n)mp}$ 为一组 mp×mb 个有限状态机转移函数 $F_{mb,}^{mp,}$（i=0，1，…，mp-1，$j=x_1, x_2, \cdots, x_{mb}$）。对于非空有限地点集合的个数 q，称 $S_{mb}^{(n)mp}(M, G)$ 为移动 n 阶多维策略函数，模型中还包括 n 阶多维策略函数 S_1^{mp}，S_2^{mp}，…，S_{mb}^{mp} 和移动函数Γ。如图 2-4 所示。

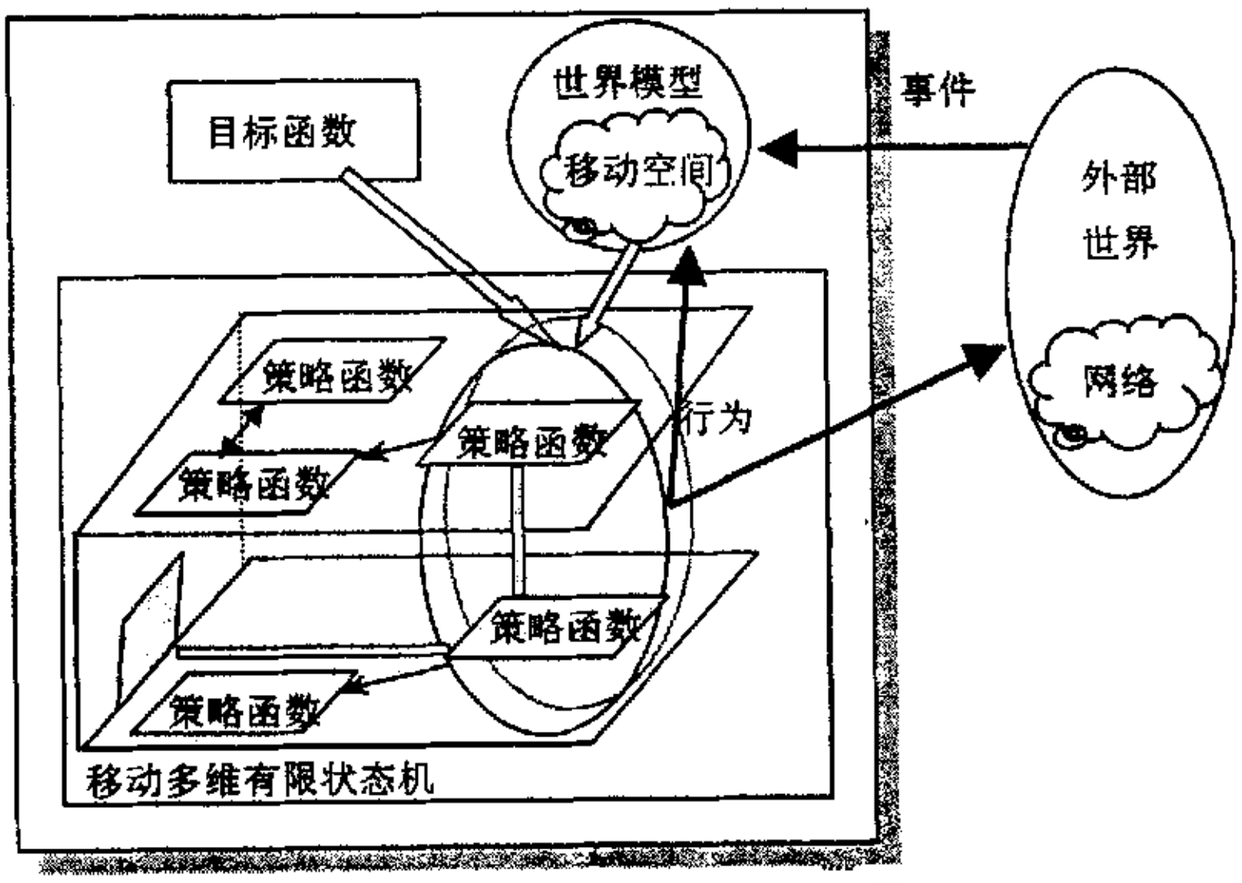


图 2-4 智能体的理论模型 AM_n^{mp}

定理 3：智能体基本模型 AM_0，如果存在移动空间（X, τ）中的移动函数Γ，那么存在一个 S' 是对 AM_0 中智能体策略函数 S 的替换。

证明：由移动空间（X, τ）中的移动函数Γ的定义，Γ:（$M^* \times G^*$）→ X；其中 X＝｛x_1，x_2，…，x_{mb}｝，

$$S(M, G) = \begin{cases} \overset{1}{S}(M, G) & \text{if } \Gamma(M, G) = x_1 \\ \overset{2}{S}(M, G) & \text{if } \Gamma(M, G) = x_2 \\ \Lambda & \\ \overset{mb}{S}(M, G) & \text{if } \Gamma(M, G) = x_{mb} \end{cases}$$

其中 $\overset{k}{S}$，k∈｛1, 2，…，mb｝是在第 k 个地点执行的策略函数，$S' = \{\overset{k}{S} \mid k \in \{1, 2, \cdots, mb\}\}$ 即为智能体策略函数 S 的替换。证毕。

2.2 基准模型

2.2.1 基准模型的定义[118]

基准模型是在特定的研究范围内一类系统模型的抽象，它描述了所有可能的软件构件、构件提供的服务以及构件之间的关系。其中，构件之间的关系表示构件之间的组合关系和交互关系，以构件之间的接口为表示形式。

一个基准模型可以表示为一个四元组：

NM＝<Range，Entities，Services，Interfaces>

从相似系统和模型中抽取得到的基准模型主要包括研究领域抽象、实体抽象、服务抽象和接口抽象四方面的内容：

①领域抽象：研究领域界定了该基准模型所涵盖的范围，主要包括其产生的主要背景、所依赖的领域环境、适用人员等内容的阐述；

②实体抽象：实体抽象表现为公共的概念、统一的解释、服务和接口的抽象；

③服务抽象：服务抽象是对已有系统和模型中提供的类似服务进行抽象；

④接口抽象：接口在基准模型中起着二方面的作用：一是作为服务的使用方式的规范描述；二是系统内部软件构件之间的关联关系及其互操作的阐述。基准模型中的接口通常可划分为两类：水平接口和垂直接口，前者主要是模型中同层次实体之间相互提供服务的使用接口；后者主要是模型中相邻两层实体之间相互提供服务的使用接口。

2.2.2 代表性基准模型

IEEE POSIX OCE 模型：IEEE POSIX（Portable Operating System Interface）标准及基准模型是在 1986 年颁布的、发展至今的、庞大的标准和基准模型家族。图 2-5 是 IEEE 1995 年公布的 POSIX OCE 开放系统环境的基准模型[119]。

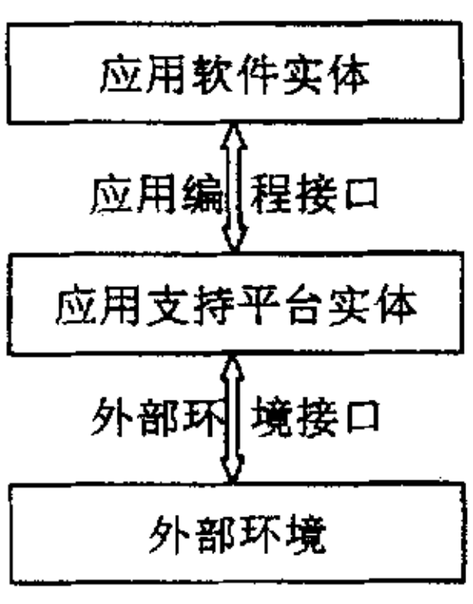


图 2-5 POSIX OCE 基准模型

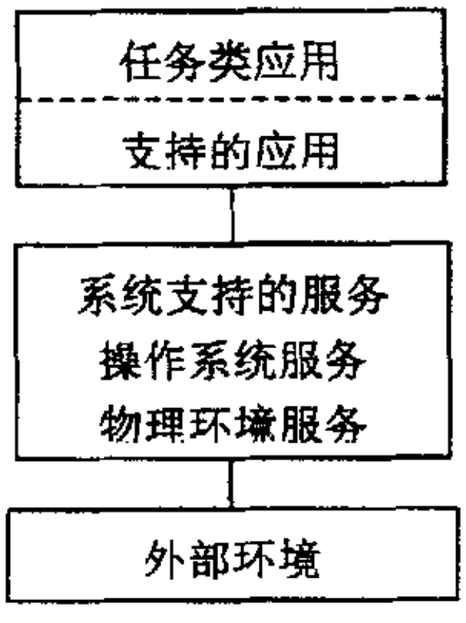


图 2-6 TAFIM 模型

POSIX OCE 模型主要包括三类实体和两层接口：

①应用软件实体；

②应用编程接口；

③应用支持平台实体；

④外部环境接口；

⑤外部环境。

TAFIM 模型：TAFIM（Technical Architecture Framework for Information Management）基准模型是美国国防部（DOD USA）为提高其所辖信息管理系统的集成度而发布的一个基准模型，该模型提供了信息系统的基础架构、定义了公共的概念框架、服务及其接口、采纳的标准、软件构件及其配置管理策略[119]。

TAFIM TRM 主要侧重于分析和规约系统的服务，其概要图示如图 2-6。

GOA 模型：GOA（Generic Open Architecture）框架是 SAE（Society of Automotive Engineers）于 1996 年发布的一个框架模型[120]。该模型从规约开放系统的接口标准入手，建立了一个为软/硬件设计和实现服务的结构模型。该模型提出了水平接口和垂直接口的概念，分别对应于同层次构件之间的交互接口标准和相邻层次之间构件的交互接口标准。其简化模型如图 2-7。

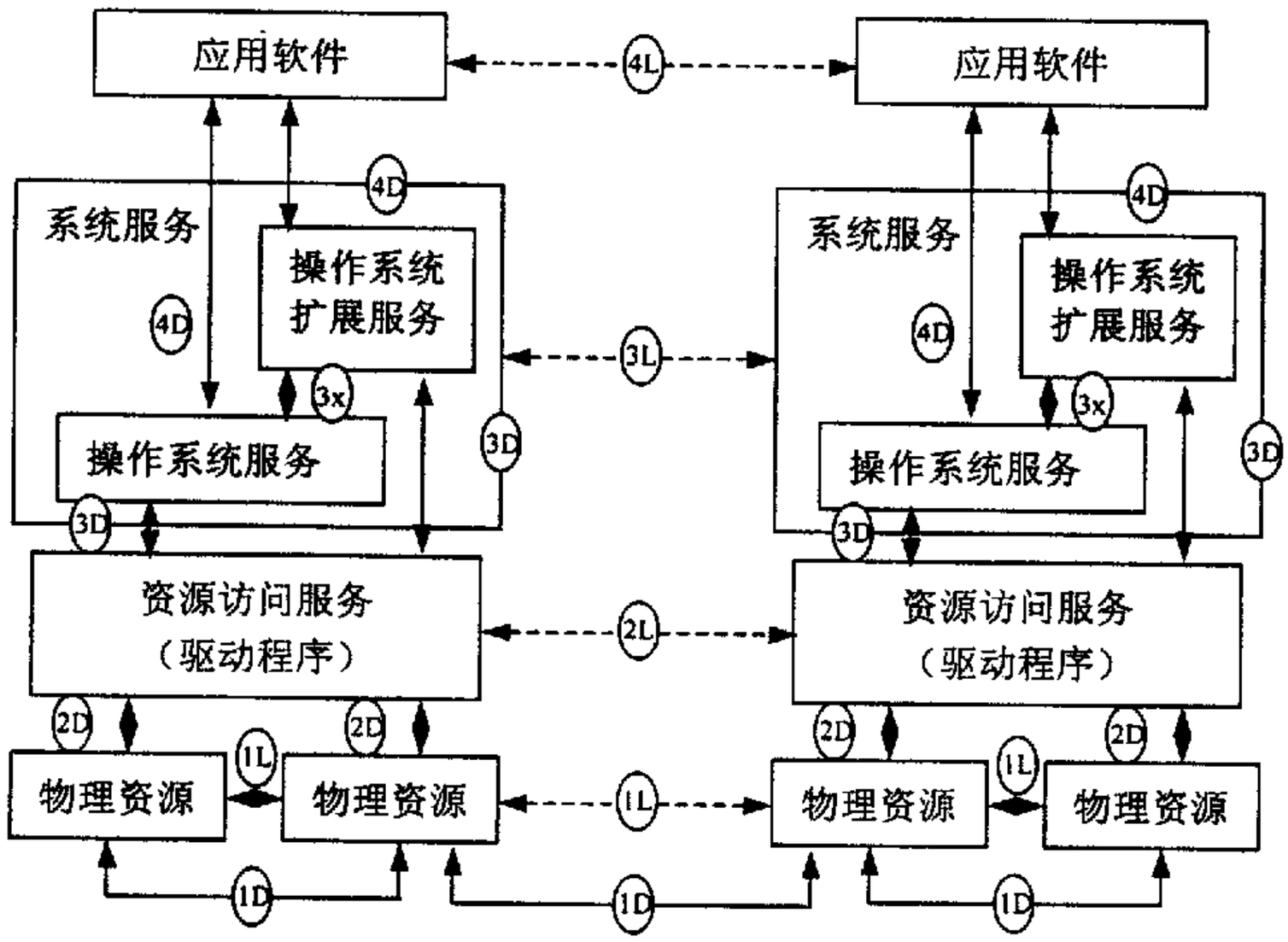


图 2-7 GOA 模型

其中虚线表示水平接口，实线表示垂直接口。

2.3 NM_MAS 概述

2.3.1 NM_MAS 的领域需求和条件限定

NM_MAS 是基于 Internet 的移动 Agent 系统的基准模型，因此，该模型的研究领域主要界定为 Internet 环境、移动 Agent 支持系统、移动 Agent 应用系统以及移动 Agent 实体自身等几个层面。我们限定如下：

①研究内容不涉及智能 Agent；

②研究内容中的通信协议不涉及 IP 层以下内容；

③操作系统、TCP/IP 协议和 JVM 是研究环境中每个机器的默认软件配置。

同时，我们不在以下方面进行限定：

①不限定网络可用性。研究环境中，允许网络环境发生关于可用性的变化，如网络物理线路故障、网络节点崩溃等；

②不限定网络是安全的。研究环境中，允许产生各种安全隐患，允许发生所有类型的不安全事件，如窃听、伪装、重放等事件；

③不限定网络通信是可靠的。研究环境中，不限定网络数据传输和网络通信是无延时、无丢失、保序的。

2.3.2 NM_MAS 基本概念

NM_MAS 中的基本概念有：

（1）软件 **Agent**[115]

软件 Agent 是一个封装良好的计算实体，代表用户完成一定的任务，生存于一个执行环境中并拥有以下基本特性：①封装良好；②代理能力；③自主性；④协作性。

（2）移动 Agent

移动 Agent 是运行于开放和动态的网络环境中的封装良好的计算实体，它代表用户自主地在网络上移动，完成指定的任务。通常，移动 Agent 由数据、操作和行为规则封装而成。它具有自主性、移动性、协作性和安全性等特性。

移动 Agent 可以在各主机之间自由移动。在某个执行环境中建立后，移动 Agent 可携带自身状态和代码在网络中转移到另一环境中去，并可在该环境中恢复执行。其中“状态”是指 Agent 在异地目标环境中恢复执行时所需的属性值。而“代码”是 Agent 执行的必要条件。

（3）静止 Agent

并非 Agent 都必须移动，Agent 可以是静止的，以通常方式和周围环境交互，如各种形式的 RPC 及消息传递，这种不需要或不能移动的 Agent 称静止 Agent。

（4）Agent 身份

Agent 身份是其所代表的用户的标识。它可以是一个独立个人、团体或机构。Agent 身份一般用于 Agent 运行中的身份认证和授权。通常由一个用户标识符以及相关资质验证属性如密码组成。

（5）Agent 名

每个 Agent 必须有一个标识自身的名字，主要用于 Agent 的管理及协作，这个名字在 Agent 运行的分布环境中必须是唯一的，且在 Internet 范围内也应是唯一的。

（6）移动 Agent 环境

移动 Agent 环境是架构在异构网络环境上的、一个或多个同构或异构的分布式软件系统，移动 Agent 在这些系统组成的环境中得到创建、消亡、执行、迁移、协作、安全保障及资源访问等运行需求的支持。

（7）移动 Agent 运行空间

当 Agent 在上述网络环境中运行时，这个 Agent 所能到达并能正常工作的网络节点所构成的网络环境为这个 Agent 的运行空间，即 Space。

（8）移动 Agent 服务器

移动 Agent 网络环境由若干物理节点构成，每个物理节点上都应配置一个移动 Agent 服务器，该服务器应能完成本物理节点内 Agent 的创建、消亡、执行、迁移、协作、安全保障及资源访问等运行需求的支持，并能和相关节点的 Agent 服务器合作，完成 Agent 的跨节点运行需求。

（9）移动 Agent 运行地点

移动 Agent 运行空间中，安装了上述移动 Agent 服务器的物理节点上的各类软、硬件资源和服务称为这个 Agent 的运行地点，即 Place。

（10）基于移动 Agent 的应用系统

基于移动 Agent 的应用系统是一种运行在移动 Agent 环境中的分布式应用系统，它由一系列静止或移动的 Agent 构成。

2.3.3 移动智能体模型[4]

我们将移动 Agent 的体系结构定义为以下相互关联的模块：安全代理、环境交互模块、任务求解模块、知识库、内部状态集、约束条件和路由策略。如图 2-8 所示。

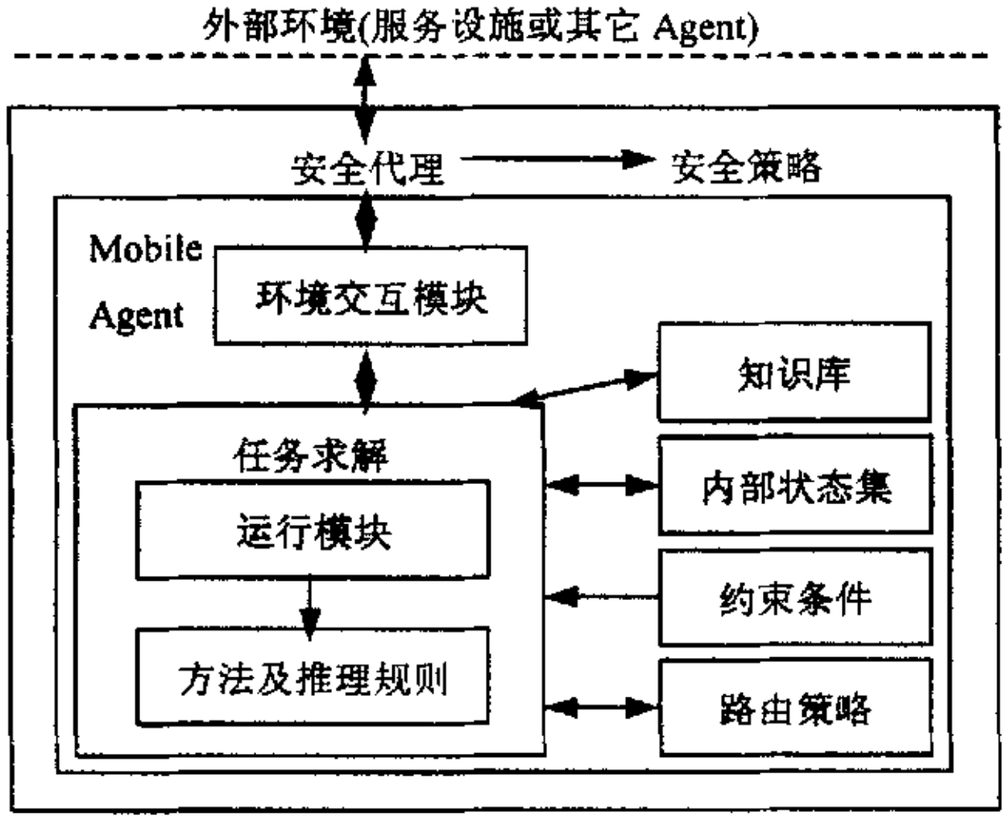


图 2-8 移动 Agent 的结构模型

① 体系结构的最外层为安全代理(Agent Proxy)，它是 Agent 与外界环境通讯的中介，执行 Agent 的安全策略，阻止外界环境对 Agent 的非法访问。

② Agent 通过环境交互模块感知外部环境并作用于外部环境。环境交互模块实现智能体通信语言（Agent Communication Language，ACL）的语义，保证使用相同 ACL 的 Agent 和服务设施之间的正确通信和协商，而通信内容的语义与 ACL 无关。

③ Agent 的任务求解模块包括 Agent 的运行模块及 Agent 任务相关的推理方法和规则。运行模块包括 Agent 的初始化程序和事件处理程序，前者在初始或移动到另一节点后启动事件处理线程，后者持续自主运行，感知外部环境的请求，并依据内部的规则和状态产生动作。Agent 运行模块可以设计为任务独立的模块，任务相关性由不同的推理方法和规则集来实现。

④ 知识库是 Agent 所感知的世界和自身模型，并保存在移动过程中获取的知识和任务求解结果。

⑤ 内部状态集是 Agent 执行过程中的当前状态，它影响 Agent 的任务求解过程，同时 Agent 的任务求解又作用于内部状态。内部状态必须支持 Agent 跨平台的持续执行。

⑥ 约束条件是 Agent 创建者为保证 Agent 的行为和性能而做出的约束，如返回时间、站点停留时间及任务完成程度等，一般只有创建者拥有对约束条件的修改权限。

⑦ 路由策略决定 Agent 的移动路径。路由策略可能是静态的服务设施列表(适用于简单、明确的任务求解过程)，或者是基于规则的动态路由满足复杂和非确定性任务的求解。

2.3.4 移动智能体服务环境[4]

移动智能体服务环境的结构如图 2-9 所示。

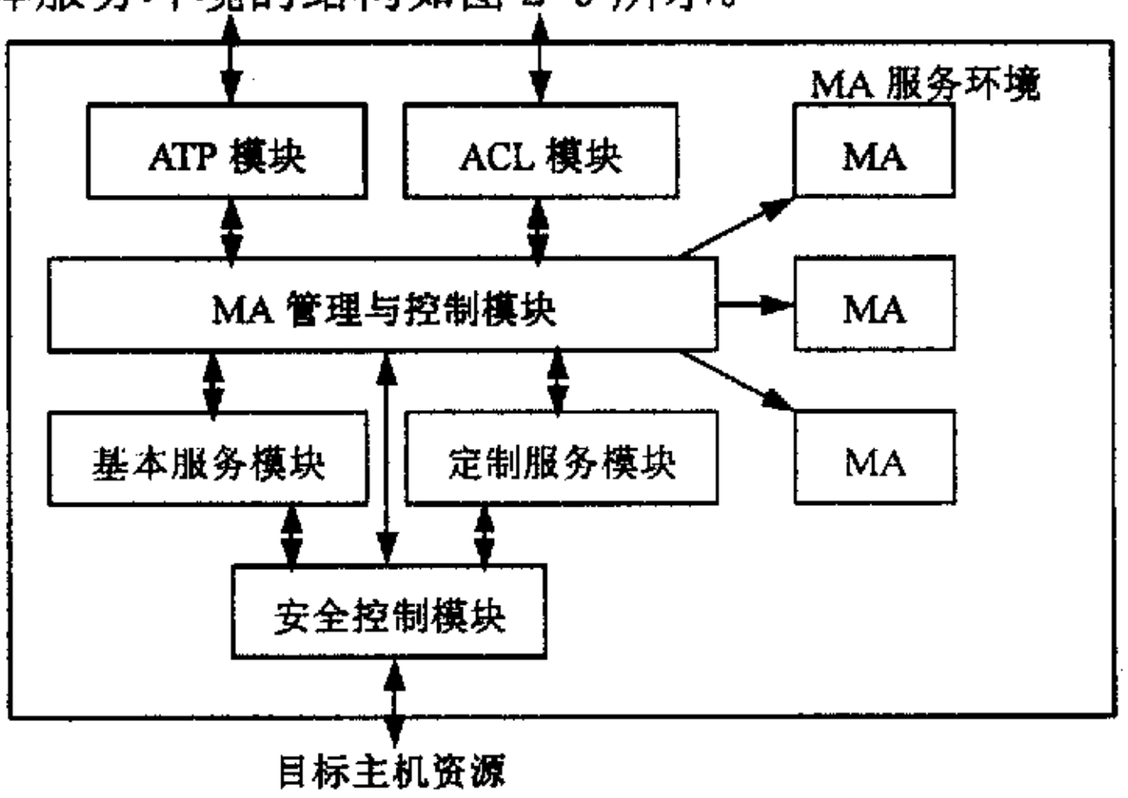


图 2-9 移动智能体服务环境结构

ATP 模块：智能体传输协议（Agent Transfer Protocol，ATP）定义了移动智能体在各主机之间移动的语法和语义，ATP 模块具体实现 Agent 在服务环境间的移动机制，包括移出和移入。

ACL 模块：采用 ACL 完成除移动智能体移动外的其它通信任务，主要用于移

动智能体之间的通信。

MA 管理与控制模块：它是移动智能体服务环境的中心部件，将有关移动智能体正常运行所需的各项服务正确分配给相应的模块，如将有关移动智能体执行环境的建立、启动的服务交给基本服务模块，将移动智能体身份认证的服务交给安全控制模块，将移动智能体要完成的特殊任务交给定制服务模块，将重新移动的任务交给 ATP 模块，将移动智能体之间通信的任务交给 ACL 模块等，并协调各个模块的正常运行。该模块的另一个重要功能是实施移动智能体的约束机制，并根据约束条件控制各个模块的运行。

基本服务模块：提供基本的移动智能体服务，如移动智能体生命周期服务(包括移动智能体初始参数的设置、安全管理器的设置、移动智能体的启动、保证移动智能体的正常运行等)、完成与其它模块的交互(如调用安全控制模块进行移动智能体的可靠性验证、通过 MA 管理、控制模块与其它移动智能体通信)等。

定制服务模块：为移动智能体提供领域相关的任务求解服务，以组件的形式出现，以便充分利用第三方的产品。

安全控制模块：主要用于实现主机的安全性策略。如进行数字签名验证、移动智能体代码的解密/解压缩等工作，还控制移动智能体对本地资源的安全访问，进行付费检查等。

MA 服务环境的分配有两种策略：一种是为每一个移动智能体分配单独的服务环境，另一种是为所有的移动智能体分配同一个执行环境。显然，第一种分配策略具有更强的安全性，而且对于异构的移动智能体必须得分配在不同的执行环境中，但这种方法会占用更多的资源。

2.3.4.1 智能体传输协议 ATP[4]

ATP 定义了移动智能体传输的语法和语义，具体实现了移动智能体在支持 ATP 的各服务设施（平台）间的移动机制。我们参考 IBM 公司在 Aglet 产品中提出的 ATP 框架结构，自定义一组原语产生的接口和基础消息集，可以说这是 ATP 的最小实现。其基本操作示意图见图 2-10。

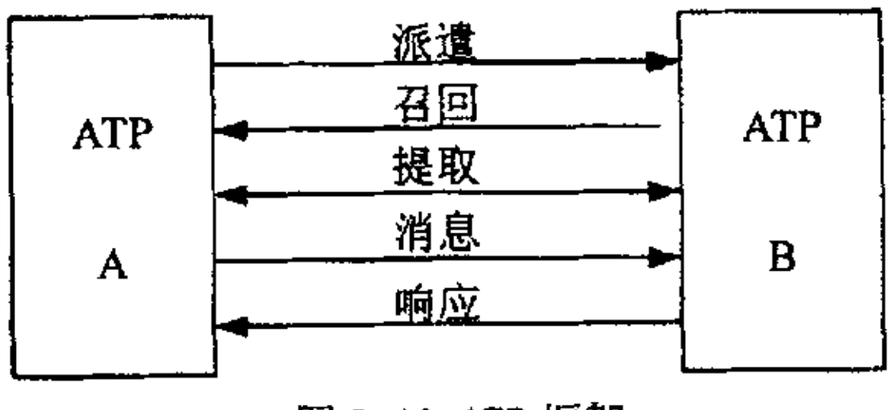


图 2-10 ATP 框架

2.3.4.2 移动智能体生命周期模型

- 生命周期视图[4]

移动智能体从创建、发送、执行任务到完成任务后返回原宿主机待命的整个生命周期，可以用六个状态描述。一个移动 Agent 在其生命周期里一定处于某一个状态，并且在某类特定事件发生时，不断发生状态的转换，最终转为结束状态，完成用户指派的任务。各状态间的过程联系如图 2-11 所示：

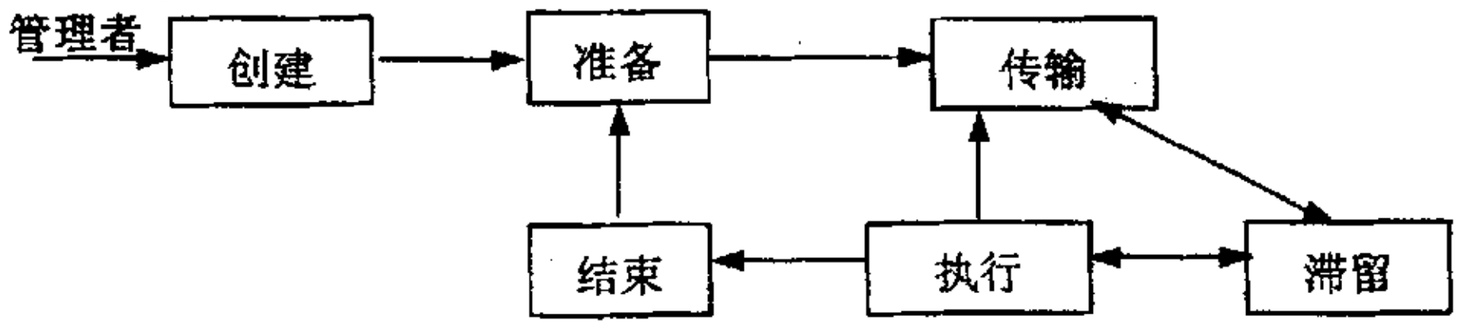


图 2-11 移动智能体的生命周期

- 状态分析

① 创建状态：移动 Agent 应用系统主要由若干静止或移动的 Agent 构成。移动 Agent 首先应被创建。创建工作主要包括：给定移动 Agent 名及身份、确定功能语义和迁移语义。创建工作需要移动 Agent 环境提供一个良好的用户使用环境和一套丰富的移动 Agent 应用程序编程接口，帮助完成上述工作。

②准备状态：当 Agent 创建完成或被置于发送队列时称准备状态。

③执行状态：执行状态是移动 Agent 的核心状态，Agent 在执行期间代替用户使用各类资源，进行协作，最终完成任务。移动 Agent 进入执行态有两种情况：首先是创建后的 Agent 被激活后，转入执行状态；其次是 Agent 经过迁移到达一个新的站点时，恢复原来的执行状态。

④滞留状态：移动 Agent 需要按照一定的规则在网络上进行移动。这种移动是一个计算状态和计算上下文的移动，因此，Agent 移动前必须暂时中止 Agent 的计算，进行运行状态和运行上下文的封装，此时，移动 Agent 的状态将转换为滞留状态。因此，Agent 需要运行环境能够提供上述功能从而完成 Agent 执行的滞留。此外，处于滞留状态的 Agent 无法对外界环境做出反应，此时如果有协作方的协作要求，Agent 将无法及时做出反应。因此，运行环境应对所有处于滞留状态的 Agent 进行上述协作信息的暂存和缓冲，等该 Agent 迁移到新节点并恢复执行时，将上述协作要求再转发至该 Agent。

⑤传输状态：移动 Agent 的滞留是为迁移进行准备，迁移状态是一个移动 Agent 按照迁移信息进行物理转移过程中所处的状态。迁移的完成必须依赖运行环

境，借助网络和操作系统的数据传输能力才能完成。和滞留状态一样，处于传输状态的 Agent 也会“丢失”此时到达的协作信息，运行环境需要对上述协作信息进行暂存、缓冲和转发。此外，迁移是完成一个移动 Agent 在物理上两个节点之间的数据和代码转移，因此，处于迁移状态的移动 Agent 还要求环境进行必要的容错处理。否则将造成协作的失败和应用系统的破坏。

⑥结束状态：移动 Agent 应用程序完成用户赋予的工作后，该程序所创建的所有 Agent 应被撤销。通常，撤销有两种方式：第一种是按照 Agent 的功能语义和迁移语义，由系统主动撤销；另一种是用户在线发布撤销指令，由运行环境强行完成 Agent 的撤销。

2.3.4.3 移动智能体管理

一个移动 Agent 在其生命周期中，总是处于上述六种状态，并且不断发生位置变迁和状态变换。这给移动 Agent 的协作和管理带来了不便。因此，运行环境还必须提供一个简便、易用的移动 Agent 管理设施，通过提供状态登记、注册、注销等服务记录移动 Agent 的运行状态信息和物理位置信息，满足 Agent 的需要。

2.4 NM_MAS 的详细分析

2.4.1 NM_MAS 的层次式多视角综合模型结构

本章第二节分别给出了 IEEE POSIX OCE 模型、DOD TAFIM 模型以及 SAE 的 GOA 模型，可以发现，POSIX OCE 模型描述了环境构成的三类实体以及两层接口，该模型具有较强的通用性，可以适合通用或专用等不同目的的系统和环境的描述和规约。因此，我们采用了 POSIX OCE 的模型结构，建立了 NM_MAS 的层次结构。DOD TAFIM 模型以环境构件提供的服务为中心，描述和规范了 DOD 下属的信息系统的互操作，而以互操作构件之间的接口标准化、规范化为出发点的 GOA 模型则从设计和实现的角度给出了开放系统的互操作需求，这三类模型虽然出发点不同，表现形式也不同，但却存在着共性和互补性。NM_MAS 在模型的层次结构基础上，以实体、实体服务、服务接口为该模型的三要素，方便、准确地刻画了移动 Agent 系统基准模型：

①基于 Internet 的移动 Agent 系统同样属于开放软件系统环境，NM_MAS 采用 POSIX OCE 模型结构进行描述是一种最自然、最简洁的方式；

②移动 Agent 环境的根本任务就是为移动 Agent 的运行进行支持，提供 Agent 所需的各种服务。因此，运用以服务为中心的 TAFIM 模型的基本思想和方法，给

出环境的各类服务抽象，也是NM_MAS的重要内容之一；

③建立移动 Agent 系统基准模型的一个主要目的就是给出一个规范的、完整的框架模型，便于研究人员在理解、扩充、设计甚至实现一个移动 Agent 系统时能有所参考、借鉴和裁剪。因此，给出NM_MAS的接口规约也是必须的。

NM_MAS刻画了一个基于Internet的移动Agent系统的体系结构、成份、相互关联以及相应的接口。除了采用了上述多视角结构外，NM_MAS还采用了层次式结构，分别从不同的抽象层次上对NM_MAS进行了阐述。

2.4.2 移动智能体系统模型

基于Internet的移动Agent系统是架构在异构网络环境上的一个或多个同构或异构的分布式软件系统。移动 Agent 应用是运行于该环境中的一个分布式应用系统，一个移动 Agent 应用是由若干静止或移动 Agent 构成的。因此，移动 Agent 环境及应用是一种 Space_Place_Agent 结构，在该结构下，NM_MAS 的最高层抽象模型可以表述如图 2-12。

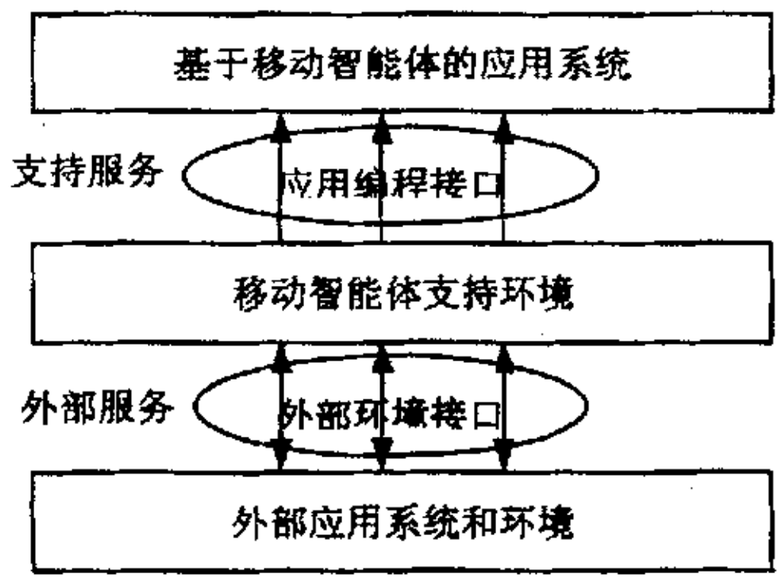


图 2-12 NM_MAS 最高层抽象模型

该模型中的实体有基于移动 Agent 的应用系统、移动 Agent 支持环境、外部应用系统和环境三类。其中，基于移动 Agent 的应用系统主要由若干静止或移动 Agent 组成，它使用移动 Agent 支持环境提供的 API 获得环境的各类支持服务，并且能够通过移动 Agent 支持环境，和外部非基于移动 Agent 的应用环境进行交互及合作。

上述模型是移动 Agent 环境的最高层抽象模型，其中基于移动 Agent 的应用系统和移动 Agent 支持环境两个实体构成了移动 Agent 环境的空间模型。图 2-13 是以两个移动 Agent 环境中的节点为例，给出的空间模型示意图：

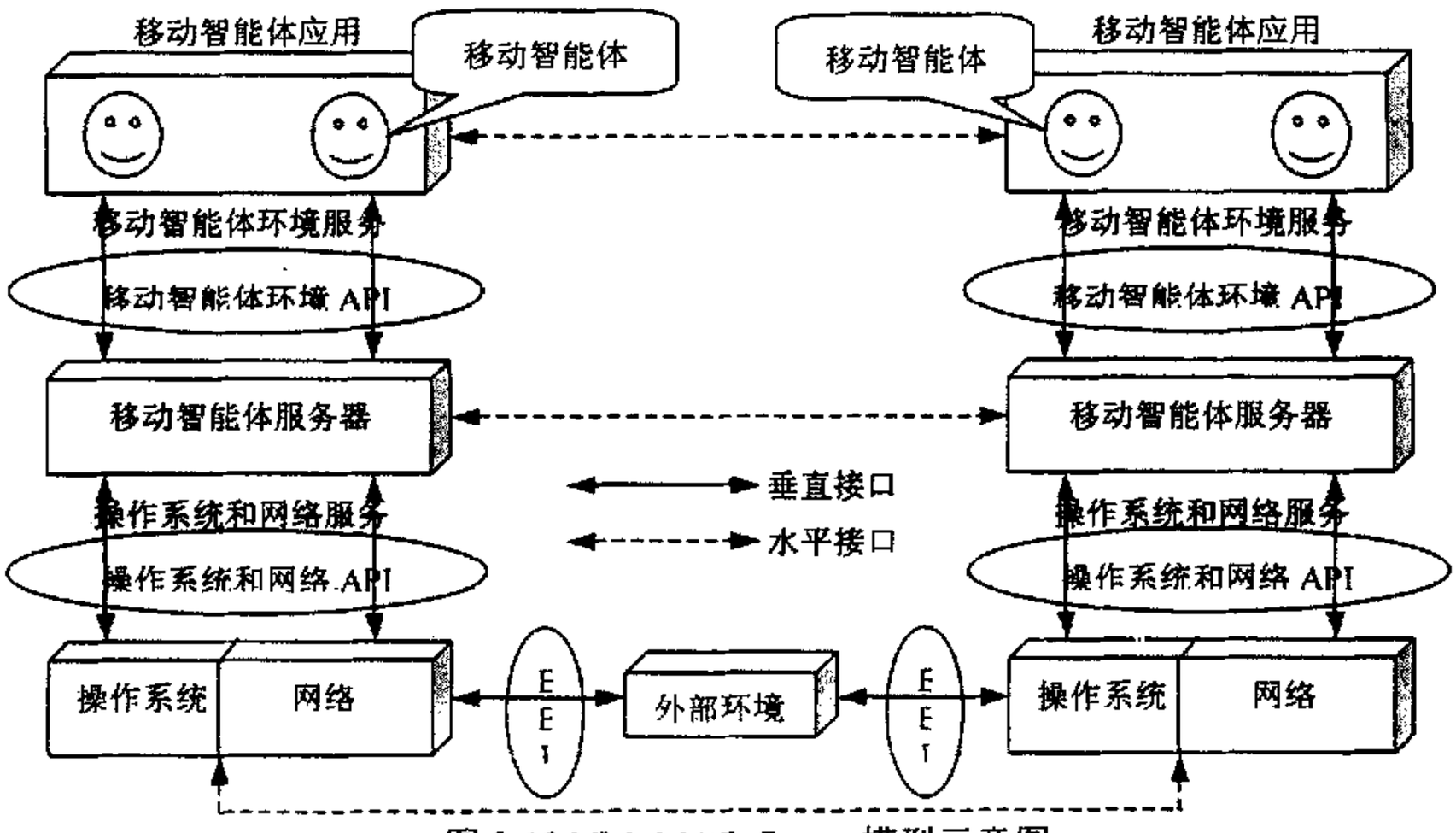


图 2-13 NM_MAS_Space 模型示意图

在上图所示的模型中，移动智能体应用系统、移动智能体服务器、网络和操作系统构成了一个节点上的 Place 模型。Place 和 Place 之间通过外部应用环境交互，或者直接关联。Place 上运行的移动 Agent 可以在 Place 之间迁移，可以通过移动智能体服务器的辅助，使用网络或者操作系统接口访问本地资源，也可以通过移动智能体服务器的辅助和另一个 Place 上的移动 Agent 进行通信、协作，共同完成指派的任务。

上述模型中的实体主要有：

① 基于移动 Agent 的应用系统：具体完成用户任务的应用系统，由多个分布在一个或多个 Place 上的静止或移动 Agent 构成。

② 移动 Agent 服务器：分布在 Space 中的每一个 Place 节点上，负责提供移动 Agent 正常运行所需的服务，如工程服务、迁移服务、通信服务、安全服务等。由提供上述服务的构件共同组成。

③ 网络及操作系统：是一个 Place 中最底层的软件成份，主要提供各种网络协议服务、数据传输服务和本地局部资源的访问服务。

④ 外部环境：外部非移动 Agent 应用系统。

相应的服务主要有：移动 Agent 提供的环境支持服务、网络和操作系统提供的网络和操作系统服务、外部环境提供的服务等。

上述模型中的垂直接口主要有：移动 Agent 环境提供的应用程序编程接口、网络协议接口、操作系统调用接口、外部环境接口。水平接口主要有：基于移动 Agent 的应用程序之间的协同及互操作接口、移动 Agent 服务器之间的接口、跨平

台的网络和操作系统之间的接口。

2.5 相关工作比较

移动 Agent 技术始于 1995 年的 Telescript 系统，迄今为止，发展十分迅速，许多科学工作者在此方面进行了研究。移动 Agent 系统基准模型的研究工作也时有报道，其中较有影响的是 MAF 和 FIPA 两个模型[43]。

MAF、FIPA Mobility 和 NM_MAS 都是在移动 Agent 研究领域里针对 Agent 的移动特性、移动支持、系统环境而开展的工作，其相似之处是不言而喻的：

①都给出了一个移动 Agent 系统的概念模型；

②都讨论了移动 Agent 的移动特性、移动支持和相应的系统环境；

③三个模型都是一个抽象模型，与实现细节无关。

然而，MAF、FIPA Mobility 和 NM_MAS 虽然都是对移动 Agent 研究的讨论，但是三者之间无论是从出发点还是从内容看，都存在较大的区别：

①MAF 侧重于移动 Agent 环境之间的互操作问题，考虑的是在已有 Agent 系统的基础上，如何借助如 CORBA 服务或添加最少的服务，从而提高其互操作性，并没有给出一个新系统的构建指南；

②FIPA Mobility 标准则是沿用智能 Agent 的思路，采用给出移动 Agent 的迁移 Ontology 的方法来描述和定义一个能够支持 Agent 移动的最小功能集合，试图从语言、通信协议等角度规范一个移动 Agent 系统的设计和实现；

③NM_MAS 从实体、服务和接口三个方面总结和抽象出了一个良好的、实用的移动 Agent 系统的基准模型，相比而言，内容更丰富、更系统，也更具有实用性。

其实，上述三个模型是从不同的角度、不同的层次上分别刻画了一个移动 Agent 系统的标准化问题，具有较强的互补性。

2.6 本章小结

针对目前移动 Agent 的研究中缺乏统一的移动 Agent 系统基准模型的现状，本章首先给出了软件智能体的定义和理论模型。其次给出了一个基于 Internet 的移动 Agent 系统的基准模型 NM_MAS，该模型从研究领域分析、移动 Agent 系统的实体模型、移动 Agent 的环境服务以及移动 Agent 环境的接口等四个方面，系统阐述了基于 Internet 的移动 Agent 技术及其系统中涉及的公共术语及其解释、通用的

体系结构、规范的服务及接口。和 MAF 模型和 FIPA Mobility 模型相比，NM_MAS 的内容更丰富、更系统，也更具有实用性。

3 互联网上移动智能体系统的设计

IBMAS 以 Internet 为基础网络环境，采用了 Space_Place_Agent 体系结构。Space 由分布于 Internet 上的若干 Place 组成。每个 Place 上驻留一个负责移动 Agent 管理的服务器。Agent 移动并运行于各个 Place 中。

移动 Agent 服务器支持移动 Agent 的创建、执行、传输和终止，并为移动 Agent 提供通信和安全服务，满足其工作需求。其结构设计图如图 3-1 所示。它提供了移动 Agent 运行时的全面支持环境：迁移支持子系统、通信支持子系统、安全支持子系统和开发监控子系统。其中：迁移子系统负责管理移动 Agent 的迁移机制，必要时加以干预；通信子系统负责实现移动 Agent 之间、移动 Agent 和主机之间以及移动 Agent 和用户（程序）之间的通信，并解决与通信相关的问题；安全子系统负责本地移动 Agent 环境的安全，以及在本地产生的移动 Agent 的安全；开发监控子系统负责创建、终止移动 Agent，以及对移动 Agent 的运行进行监控和调度。

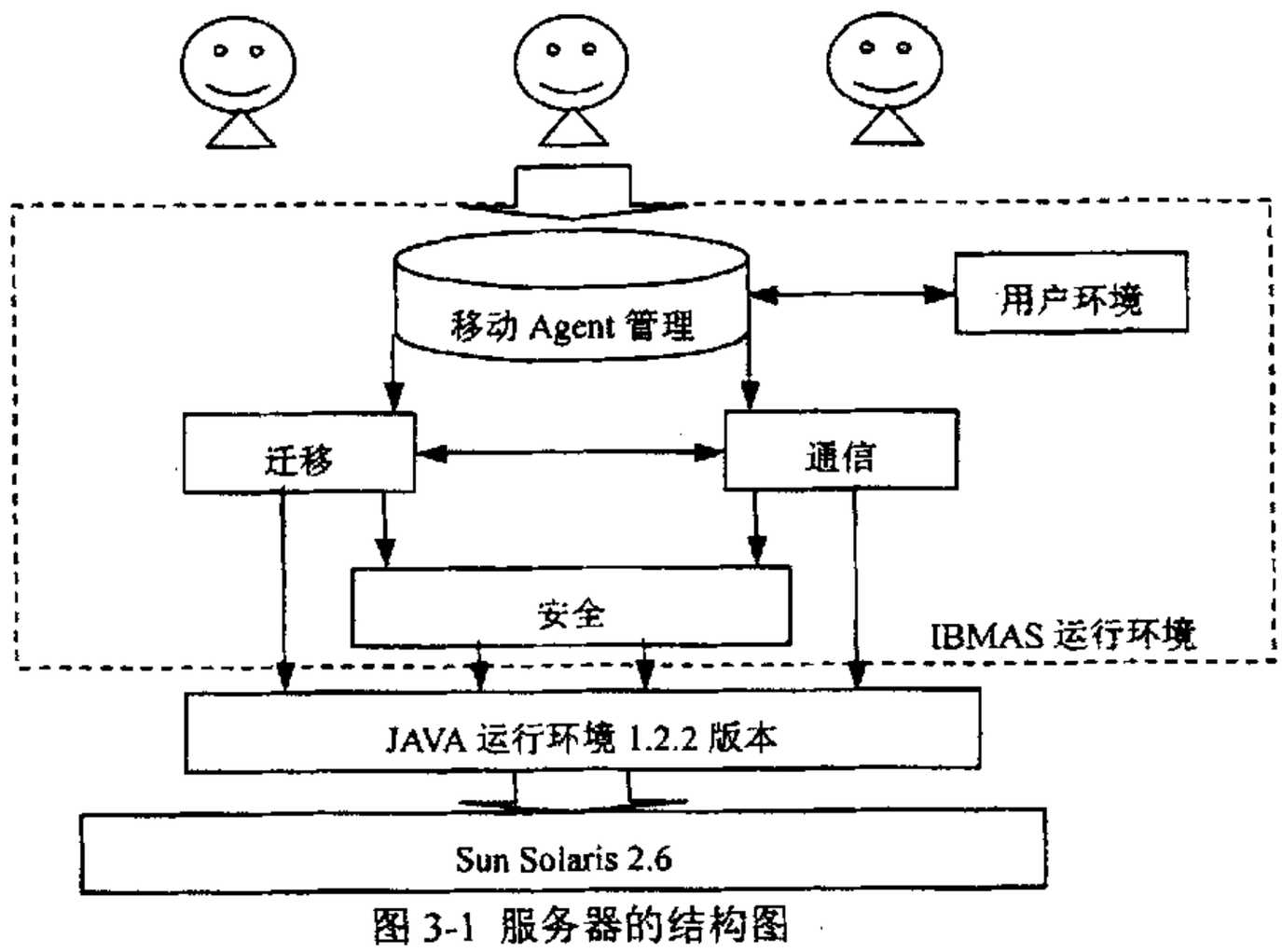


图 3-1 服务器的结构图

本章首先给出了系统的设计目标和原则，然后介绍了移动 Agent 的设计、移动 Agent 结构化迁移机制的设计及其与相关工作的比较。

3.1 设计目标和原则

IBMAS 的设计目标有：

① 技术探索：对移动 Agent 技术开展细致而深入的研究十分必要，建立一个

完整的支持环境是最好的技术探索方式；

② 建立实验平台：移动 Agent 技术是一门新兴的网络技术，其应用十分广泛，建立一个完整的系统将为以后的研究工作提供一个很好的实验平台；

③ 进行示范应用：移动 Agent 技术具有广阔的应用前景，在 IBMAS 上我们进行了网络安全、信息查询、电子商务等不同领域内的应用探索。

在 IBMAS 的设计和实现过程中，始终遵循了以下原则：

① 开放性：开放性主要体现在标准的采纳上。IBMAS 的设计尽可能遵循现有的标准，使得系统具有较好的互操作和可扩展性；

② 实用性：IBMAS 是移动 Agent 的运行支持系统，实用性要求很高，在系统设计和实现中，系统的容错能力、性能、可靠性和健壮性是重要的考虑因素；

③ 友善性：友善性体现在两个方面，一是采用良好的图形用户界面，简化用户使用；二是尽量使用现有的工具。

3.2 移动智能体的设计

3.2.1 移动智能体名的设计

移动 Agent 名是移动 Agent 的标识，在移动 Agent 的移动、通信及对其监控中的地位十分重要，可动态移动 Agent 的名应满足以下要求：

① 在移动 Agent 运行的分布环境中必须是唯一的，即在整个 Internet 范围内应是唯一的；

② 为便于移动 Agent 的寻址，该名应动态反映移动 Agent 位置的变化，标识或映射某时刻移动 Agent 所处位置的物理地址；

③ 易于用户理解，方使用户使用。

基于以上要求，我们设计了双层名（包括逻辑名和物理名）的命名机制[51,121-122]。

在 IBMAS 的双层名中，一个移动 Agent 名由如图 3-2 所示的两部分构成：

创建节点主机 IP（域名）：符号名
当前节点主机 IP（域名）：ID 号

图 3-2 移动 Agent 的双层名

第一层名标识称为逻辑名，由两部分构成：创建移动 Agent 的主机的 IP 或域名以及该移动 Agent 被用户赋予的一个符号名，显然，只要保证在一台主机上生成的尚未消亡的移动 Agent 被赋予了不同的移动 Agent 名，命名的全局唯一性便可以实现。而用户对名的使用如迁移或者通信等工作，则可直接使用移动 Agent 逻

辑名；但是，单纯的逻辑名不能动态反映移动 Agent 位置的变化，无法帮助寻址工作，因此，我们定义一个移动 Agent 的同时还分配了一个物理名，即第二层名标识，该层名标识也由两部分构成：移动 Agent 迁移之后的当前主机的 IP 或域名以及该主机上移动 Agent 服务器为该移动 Agent 赋予的一个 AID，该 AID 类似 UNIX 系统中的进程号 PID，由移动 Agent 服务程序统一管理，保证移动 Agent 在这台主机上的 AID 唯一性。因此，在移动 Agent 的整个生命周期中，其逻辑名保持不变，用户通过逻辑名来标识和使用一个移动 Agent；移动 Agent 的物理名可以反映其在任一时刻的位置，移动 Agent 每迁移到一台主机，物理名都会改变，物理名是系统对移动 Agent 的标识，对用户是透明的。

在这种命名规则下，假设一个用户在 192.168.1.1 上创建了一个移动 Agent，用户赋予该移动 Agent 一个符号名为 Client，其第一层名就是 192.168.1.1：Client，且该名在这个移动 Agent 的生命周期内不会发生变化。当该移动 Agent 被派遣出去，到达一个新的节点 192.168.2.26 之后，该节点上的移动 Agent 服务器将为这个移动 Agent 分配一个唯一的标识 1974，此时，该移动 Agent 的完整名将如图 3-3 所示：

192.168.1.1：Client
192.168.2.26：1974

图 3-3 Client 的双层名字

显然，这种双层名的命名机制可以保证移动 Agent 的命名唯一性要求，同时也动态反映了移动 Agent 的位置变化，记录了当前的物理位置和标识，为移动 Agent 的管理和通信奠定了良好的基础。

3.2.2 身份设计

移动 Agent 在计算、协作过程中，代表一个特定的用户完成移动 Agent 功能语义所刻画的任务，不同用户在网络环境和具体网络节点中具有不同的权限，因此，移动 Agent 必须含有所代表用户的信息，通常采用移动 Agent 身份的方式来表达。移动 Agent 身份通常由一个用户标识符以及相关资质验证属性如密码组成。

3.2.3 功能语义和迁移语义的分离

移动 Agent 代表用户完成特定的任务。在移动 Agent 的成份中，移动 Agent 功能语义部分就是上述具体工作的描述。移动 Agent 的一个特色就是可以按照一定的规则，在中断移动 Agent 的计算工作后，自主迁移到另一个节点上并恢复工作。而这种迁移并非单步迁移，可以进行多次的自主迁移。因此，一个移动 Agent

必须携带反映或指导这些迁移的具体算法与信息，这部分信息的描述称为移动 Agent 的迁移语义。

移动 Agent 应携带足够的功能语义和迁移语义。移动 Agent 的功能语义部分（功能体）和迁移语义部分（旅行计划）是一个移动 Agent 的两个重要成份。

移动 Agent 的功能体由完成用户需求的若干方法构成。

移动 Agent 的旅行计划部分由用户制定的移动 Agent 旅行计划以及若干迁移控制处理方法构成，而每个旅行计划由若干旅行步构成。

在 IBMAS 中，移动 Agent 的功能语义和迁移语义是分离的。

3.3 结构化迁移机制的设计[49-56,123-124]

移动 Agent 是按照一定规则在网络上漫游的 Agent。移动能力是其基本需求。因此，支持 Agent 移动的迁移机制是移动 Agent 的核心技术，它支持移动 Agent 在网络上自主移动，能在合适节点上停留下来，继续计算，然后继续移动和执行，直到完成用户的任务，最后把结果回送给用户。移动 Agent 的移动不仅包括代码的移动，还包括计算的移动，迁移机制涉及许多关键技术，下面将从移动的目的、移动支持的技术需求、各个关键技术的设计等方面阐述 IBMAS 中迁移机制的设计工作。

3.3.1 迁移的目的

移动 Agent 迁移的目的可以从以下三个方面来描述：

① 在应用程序的体系结构上，增强应用系统的网络适应性，提高灵活性和健壮性；

② 在应用程序的性能指标上，降低对网络资源可用性的要求，降低通信开销，提高系统效率；

③ 在应用领域的开拓方面，移动 Agent 模式不仅支持传统 C/S 模式的应用系统的构建，而且在移动设备支持、嵌入式设备支持、数据敏感应用以及软件协同等方面具有传统模式较难或不能支持的应用。

3.3.2 迁移的技术需求

迁移支持完成移动 Agent 从迁移请求开始直到在下一节点上恢复运行为止的所有过程，图 3-4 是移动 Agent 移动过程的一个示意图：

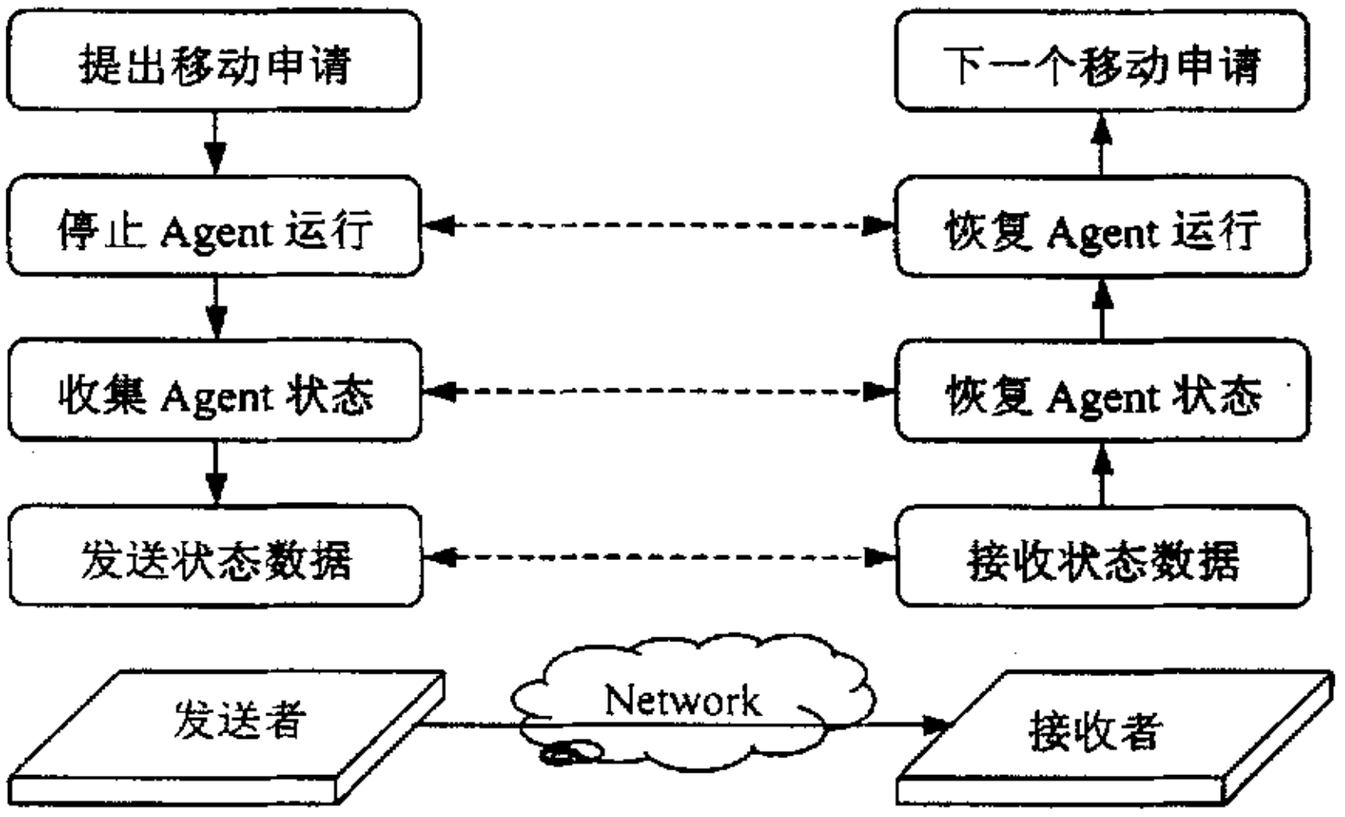


图 3-4 移动 Agent 的迁移过程示意图

从上图可以看出，IBMAS 中的迁移支持主要应包括以下一些服务：

①迁移语义的解析服务：移动 Agent 在运行过程中需要进行位置迁移时，通过迁移请求要求运行环境满足其请求。此时运行环境的 Migration 构件应提供迁移语义的解析服务，确定该移动 Agent 的下一个运行节点的位置。

②运行状态的封装服务：Migration 构件完成迁移请求的解析后，应暂时中止该移动 Agent 的运行，收集现场信息，进行各类状态的封装，做好移动 Agent 网络迁移的准备。上述这些工作，应由 Migration 构件提供服务来完成。

③移动 **Agent** 网络传输服务：移动 Agent 网络传输服务主要借助于网络和操作系统的服务，参照迁移语义解析结果，完成一个移动 Agent 运行实体的网络传输。网络传输服务应由两个物理节点上的 Migration 构件配合完成。

④迁移容错处理服务：移动 Agent 的迁移可能发生在物理上相互独立的两个机器节点之间，主要通过网络传输完成，因此，很难保证传输过程顺利进行，很可能会发生传输故障，也就是说，单纯地依赖网络协议进行移动 Agent 迁移，很难保证其正确性。因此，Migration 构件必须提供迁移容错处理服务，以确保移动 Agent 的完整性。

⑤恢复移动 **Agent** 运行：移动 Agent 通过迁移，到达某个节点后，可以借助 Migration 构件提供的该服务恢复在上一个节点的运行状态，以便继续工作。

深入分析以上服务可知，迁移语义的解析是迁移机制的关键，而迁移语义解析的核心问题是迁移语义的表达和组织形式，不同的迁移请求表达方式会导致迁移表达能力的不同，不同的迁移语义组织形式也会导致不同的迁移处理方法以及移动 Agent 应用程序的不同开发方式。IBMAS 中迁移机制的设计主要是在分析迁

移的表达方式和组织形式的基础上，设计了一个结构化的迁移表达和处理机制。

3.3.3 结构化迁移机制设计思想[49-56]

移动 Agent 迁移机制的研究受到了来自学术界和工业界的广泛关注，代表性的工作有：Dartmouth 学院的 Agent_Tcl、General Magic 公司的 Telescript、IBM 公司的 Aglets[35-36]、Mitsubishi 的 Concordia 等。

将移动 Agent 的迁移信息从移动 Agent 功能体中分离出来，用一种有足够能力的结构进行描述，并提供灵活的迁移信息处理手段，将静态表达和动态修改有机结合，不仅可以克服现有移动机制存在的移动与功能体难以分离、结构化程度不高、难以理解和复用等问题，而且可以在这种结构化的、通用的移动机制基础上研究其形式语义和移动模式复用等问题，从而为移动 Agent 的移动提供有效的支持[54]。

针对以上目标，我们提出了一种结构化的迁移机制，其结构思想体现在以下方面：首先是移动 Agent 本身的结构化，迁移信息的描述和处理在结构上完全独立于移动 Agent 功能体，因此，无论移动 Agent 的功能体还是移动 Agent 的迁移都可单独开发和使用，而且支持两者的动态装配；其次，迁移信息的描述和处理也是结构化的，完整地刻画了从迁移请求到恢复执行的全部信息，有足够的能力和灵活性表达多种迁移现象；旅行计划由旅行模式和若干旅行步组成，一个旅行步说明了一个旅行要求，旅行模式是对若干旅行步的解释方法。我们采用结构化的方式来描述一个旅行步，同时对旅行步的解释也按照固定的规则进行，此外，该机制还提供了两种典型的迁移模式和灵活的修改手段，使得该机制可以实现完全的语句级迁移。

3.3.3.1 功能和迁移信息的结构分离

在移动 Agent 结构化迁移机制中，我们采用结构化的旅行计划模型把移动 Agent 的移动信息从功能体中分离出来，用户可以分别对它们进行分析和设计，在设计功能体时不需要考虑网络的实际情况，在设计移动信息时也不需要考虑功能体的具体实现，这样有效地降低了设计和编写移动 Agent 的复杂性，也便于用户对移动 Agent 的调试和管理。在这种设计思想下，一个移动 Agent 的结构可以如图 3-5 所示：

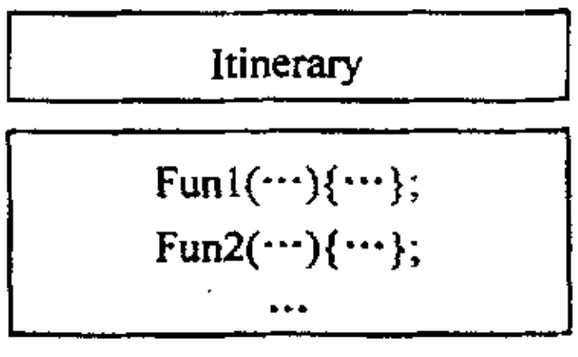


图 3-5 移动 Agent 结构示意图

如上图所示，移动 Agent 的迁移语义专门使用一个结构化的旅行计划表来进行刻画，而移动 Agent 的功能语义部分集中在功能体部分描述。上述两个部分完全独立，结构分离。这简化了移动 Agent 开发时的用户负担，用户可以单独进行旅行计划表和功能体的开发，开发过程中不需要考虑对方的细节问题，大大提高了移动 Agent 的可理解和可维护性。

旅行计划和功能体的结构分离可以实现旅行计划和功能体的动态装配。用户可以事先完成对移动 Agent 功能体的开发，然后在运行时根据当前的网络情况制定一个旅行计划，或者是从已有的旅行计划中选择一个，旅行计划和功能体动态装配好后才开始移动；反之亦然。动态装配较大地提高了功能体和旅行计划的复用程度，用户可以对已有功能体和旅行计划进行调整来产生一个新的移动 Agent，这充分展示了移动 Agent 对网络环境的适应性，如图 3-6。

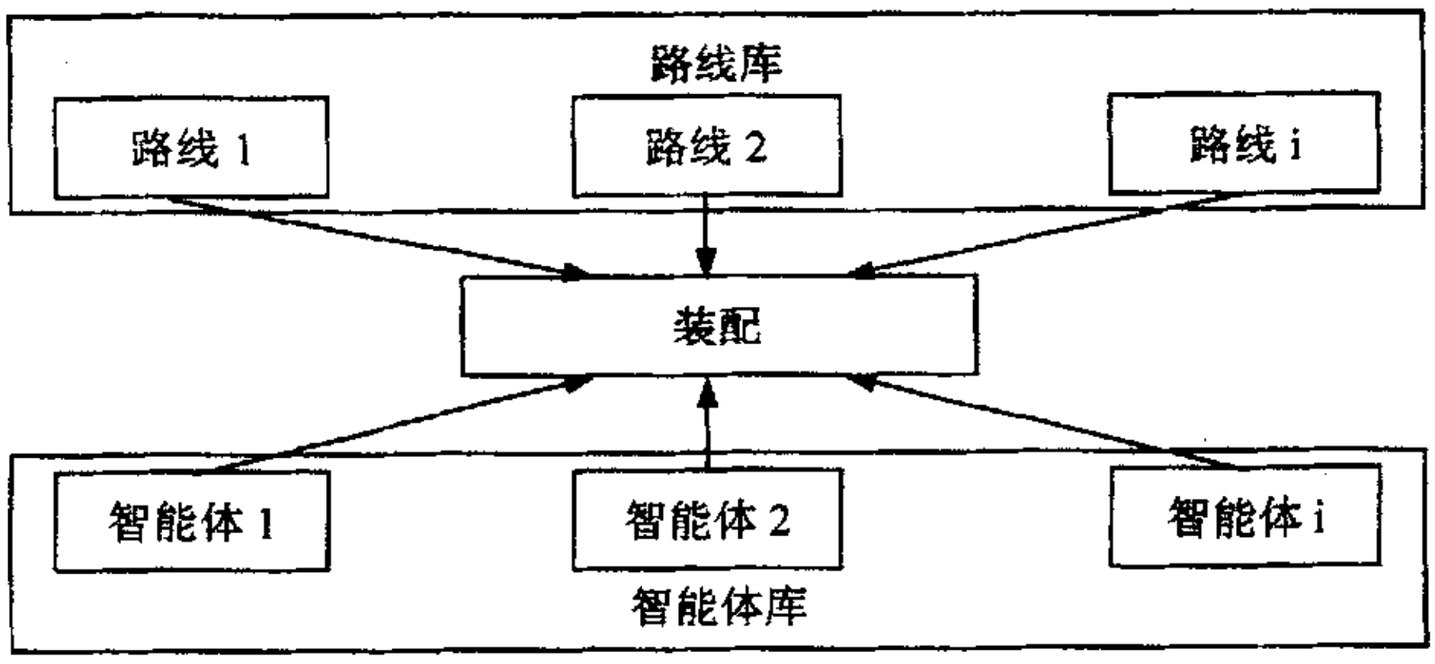


图 3-6 移动 Agent 装配示意图

3.3.3.2 旅行计划结构

IBMAS 支持用户采用一种叫旅行计划表的数据结构来描述移动 Agent 的迁移信息。该旅行计划表由若干旅行计划步组成，在不同方式下，这些旅行计划步分别解释执行。一个旅行步包含了移动 Agent 一次移动所需的所有信息，包括目标节点地址、在目标节点上应完成的任务等。我们用图 3-7 所示的结构来描述旅行步信息：

旅行步控制	目标节点	入口方法		旅行计划更新方法	名字说明
P	H	G	M	T	Name

图 3-7 旅行步结构图

图 3-7 中各个成分的含义如下：

① P 是实施该步旅行的控制。对 P 的验证在移动 Agent 迁移之前进行，P 回送为真表示该旅行步在移动 Agent 的这次运行过程中有效，移动 Agent 下一步将移动到相应的主机上去；如果 P 回送为假则该旅行步在移动 Agent 的这次运行过程中被略过。P 近似反映了移动 Agent 运行时刻对应的网络环境的[illegible]性。

② H 通常是一个 IP 地址或者域名，表示这一步旅行的目标节点地址，说明移动 Agent 应移动到哪台主机上去执行任务。

③ G 是布尔表达式或者是一个布尔方法，是入口方法 M 的前置控制，G 方法主要是对环境以及移动 Agent 本身状态进行判断，只有当 G 方法回送为真时，才会执行 M 方法。提出 G 方法的目的是把用户任务的实现和对环境的判断分离开，对同一个用户任务可以动态选择不同的 G 方法以改变对环境的要求，提高旅行计划和功能体之间装配的灵活性。

④ M 为入口方法，描述了用户任务的入口。移动 Agent 支持多入口的移动，用户可以指定不同的入口方法。

⑤ T 说明了旅行计划的更新方法，只有在整个旅行计划都解释完后才会被执行。移动 Agent 提供两种更新方式：BACK 方式，移动 Agent 最后回到创建节点，向用户报告旅行和任务的完成情况；UPDATE 方式，移动 Agent 在旅行的最后节点上根据本身和环境状态来调整旅行计划，然后按调整后的旅行计划继续旅行。UPDATE 可以是移动 Agent 的自主行为，也可以采用获得用户指令的方式。

⑥ Name 表示移动 Agent 的名字，它和创建节点地址构成全局唯一的移动 Agent 标识。

结构化模型要求在旅行过程中保证旅行计划结构不受破坏，不允许移动 Agent 在运行过程中修改旅行计划。但是不能动态修改带来的问题大大降低了旅行计划的表达能力，要求用户在移动 Agent 旅行前就制定一个完善的旅行计划也不太可能，用户无法事先对动态变化的网络环境作预测，而且灵活性也受到影响，移动 Agent 不能根据当前的网络环境做出最佳选择，只能按照预先的计划进行。为了增强旅行计划的表达能力和灵活性，同时又不影响整个旅行计划的结构化模型，我们在旅行计划中增加了限制条件的旅行计划调整方法：在旅行步中提供旅行步控制 P 和计划更新方法 T，移动 Agent 可以通过它们对旅行计划进行动态调整。由于

对 P 和 T 的解释执行是按照解释规则在特定时刻进行的，所以不会破坏旅行计划的结构。P 方法类似于一个条件控制，用户在其中提供了一些条件规则，只有满足了这些规则，移动 Agent 才会到相应的主机上去。对 P 的判断是在按照旅行步移动之前，通过 P 对环境以及本身状态的判断可以使移动 Agent 略过某些旅行步的执行，选择到当前最佳的主机上去，间接达到调整旅行计划的效果。T 是唯一可以直接修改旅行计划的方法，但它只能在旅行计划被全部解释后执行。用户如果想继续旅行，可以事先说明 T 为 UPDATE 方式，并在 T 方法中根据旅行记录以及收集到的各种信息修改和调整旅行计划，移动 Agent 将按新的旅行计划开始新的旅行。

3.3.3.3 两种基本旅行模式

旅行计划表只是提供旅行计划各个旅行步骤信息的描述形式，而移动 Agent 在网络上的漫游有许多方式，例如：顺序完成若干节点的漫游、有选择地完成若干节点的漫游、在网络上复制自己并发地漫游等。为了更好地支持上述各种漫游方式，提高移动 Agent 迁移信息的表达能力，我们设计了两种灵活、有力的旅行模式：顺序旅行和并行旅行。通过分析和试验，我们认为它们基本上可以表达用户所有的移动要求，表达能力较强。

在解释旅行模式之前，先介绍当前旅行步的概念：移动 Agent 按旅行步移动到目标节点并执行，则在下一次移动之前，这个旅行步称为移动 Agent 的当前旅行步。

（1）SEQ 模式

SEQ 模式和人们常见的旅行方式类似，移动 Agent 按照制定的旅行计划顺序移动，直至做完所有的旅行步为止，执行最后旅行步的 T 方法，对上一次旅行进行事后处理，同时也可以根据当前知道的信息修改旅行计划，进行下一步旅行。顺序旅行方式适用于前后旅行顺序相关的场合，上一步旅行的结果可能对下一步旅行有影响，例如信息检索，上一步的结果可以作为下一次的检索条件；或者顺序的计算过程，例如在石油勘探领域，首先要进行计算，然后数据成像，再进行数据过滤，最后把结果显示出来，整个计算过程的内在顺序性要求只能顺序执行。

SEQ 模式下，移动 Agent 以统一的名字旅行，在旅行过程中不能修改名字，因此 Name 域无效，旅行计划的结构如图 3-8：

SEQ	P_0	H_0	G_0	M_0	T_0
	P_1	H_1	G_1	M_1	T_1
	.....				
	P_{n-1}	H_{n-1}	G_{n-1}	M_{n-1}	T_{n-1}

图 3-8 顺序模式旅行计划

在 SEQ 模式下，移动 Agent 按照旅行计划中的次序，顺序判断并执行各条旅行步。首先顺序判断各条旅行步，直至得到一个有效旅行步，然后移动 Agent 根据这条旅行步移动，当移动 Agent 到达某个 Hi 时，调用或判断执行条件 Gi，并在目标节点执行 Mi 或越过在这个节点上的运行，执行完后在当前旅行步节点上继续判断下一条旅行步。这样不断重复判断、移动、执行，直至解释完所有旅行步，最后执行当前旅行步的处理方法 T，执行完 T 后如果有新的旅行计划则继续旅行，否则结束移动 Agent 生命周期。

如下例中，P1 和 P3 将回送 True，而 P2 则回送 False，这样移动 Agent 会先到 H1 上执行（G1，M1），然后略过 H2，直接到 H3 上执行（G3，M3），最后执行 T3，具体执行路径如图 3-9 所示。

SEQ	True	H1	G1	M1	T1
	False	H2	G2	M2	T2
	True	H3	G3	M3	T3

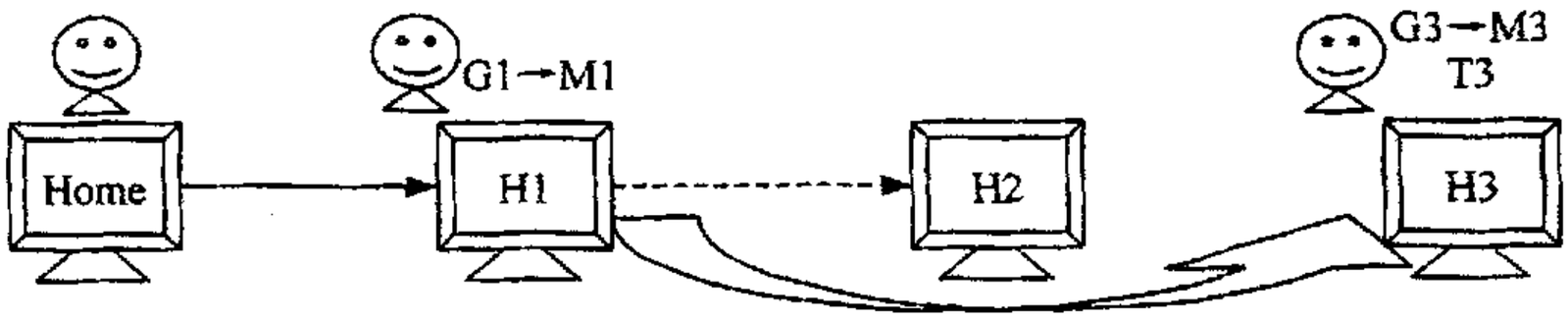


图 3-9 顺序模式旅行计划示例

（2）ALL 模式

ALL 模式实现了一种并行移动的模式，以满足对并行性的要求。旅行计划结构如图 3-10：

ALL	P_0	H_0	G_0	M_0	T_0	$Name_0$
	P_1	H_1	G_1	M_1	T_1	$Name_1$
	.....					
	P_{n-1}	H_{n-1}	G_{n-1}	M_{n-1}	T_{n-1}	$Name_{n-1}$

图 3-10 并行模式旅行计划

ALL	True	H1	G1	M1	T1	"aa"
	False	H2	G2	M2	T2	"bb"
	True	H3	G3	M3	T3	"cc"

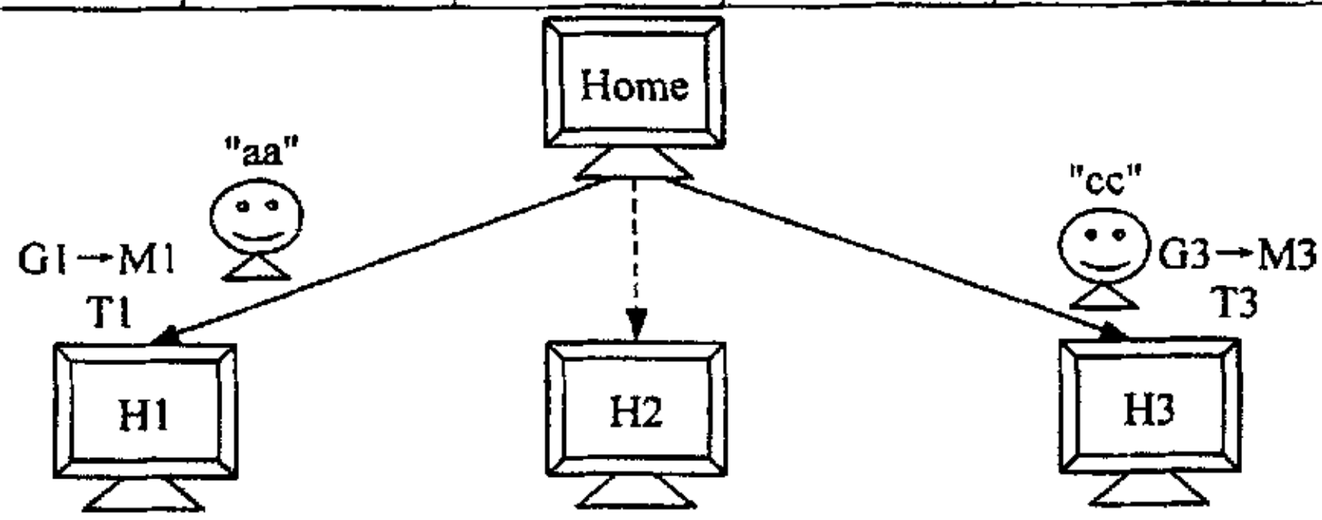


图 3-11 ALL 模式旅行计划示例

对 All 模式旅行计划的解释方法是，在移动 Agent 的创建节点上，首先判断出所有有效旅行步，然后根据每条有效旅行步克隆出一个新的移动 Agent，并启动它们的运行。新移动 Agent 的旅行计划只有一条相应的旅行步，并以 SEQ 模式进行迁移。新移动 Agent 在旅行过程中的标识由创建节点地址和相应旅行步的 Name 域组成。示例如图 3-11。

分析上述两种模式可以发现，它们和结构化程序设计方法学中的顺序结构、分支结构及循环结构有一定的相似之处：SEQ 方式对应顺序结构和分支结构、ALL 方式对应循环结构。SEQ 方式下，所有 P 值为真时，移动 Agent 依次顺序访问所有目标地址，相当于执行高级语言程序的顺序结构中的每一条指令；而 SEQ 模式下移动 Agent 只进入 P 为真的旅行步所确定的节点，如同典型的 Switch 语句中进入入口条件为真的 Case 分支一样；再看 ALL 方式，ALL 方式下“克隆”出的 n 个移动 Agent 在 n 个地址上的行为相当于在不同的数据集上将移动 Agent 重复执行了 n 次，进行了一个次数为 n 的循环。值得提出的是，ALL 方式在移动 Agent 技术的并行任务处理方面具有潜在的优势。

3.3.4 结构化迁移的形式语义研究[123]

移动 Agent 的结构化迁移机制，将移动 Agent 迁移部分的语义信息和功能部分的语义信息进行了分离，具有良好的结构性能，一方面提高了移动 Agent 的网络适应性、易理解性、易维护性和可复用性；另一方面也便于对移动 Agent 特有的迁移语义描述开展理论和形式化研究，刻画其语义，探讨其在 Internet 环境中的本质。

我们的主要工作就是在上述结构化迁移机制的基础上，开展移动 Agent 迁移

描述的形式化研究，首先给出了一个完整的移动 Agent 迁移语言 SMDLan 的语法描述，然后采用 Plotkin 于 1981 年提出的结构化操作语义（Structured Operational Semantics，SOS），给出了上述迁移描述语言的结构化操作语义的框架。为开展基于 Internet 的移动软件构件的迁移程序设计方法学的研究奠定了良好基础[123]。

3.3.4.1 SMDLan 语法

以下是 SMDLan 语法的 BNF 描述：

<旅行计划>::=<顺序旅行计划> | <并行旅行计划>

<顺序旅行计划>::=SEQ [<顺序旅行步串>] ENDSEQ

<并行旅行计划>::=ALL [<并行旅行步串>] ENDALL

<顺序旅行步串>::=Null |<顺序旅行步串>；<顺序旅行步>

<并行旅行步串>::=Null |<并行旅行步串>；<并行旅行步>

<顺序旅行步>::=MIG<迁移卫士函数名> => <迁移节点位置>RES<入口卫士函数名> => <入口方法名>UPD<旅行步更新函数名>

<并行旅行步>::=MIG<迁移卫士函数名> => <迁移节点位置>RES<入口卫士函数名> => <入口方法名>UPD<旅行步更新函数名>NEWN<新生代理名>

<迁移卫士函数名>::=<标识符>

<迁移节点位置>::=<标识符>

<入口卫士函数名>::=<标识符>

<入口方法名>::=<标识符>

<新生代理名>::=<标识符>

<旅行步更新函数名>::=<标识符>

其中，迁移卫士函数、入口卫士函数、入口方法、新生代理名等都是移动 Agent 中的函数或方法。

3.3.4.2 SMDLan 的结构化操作语义

语义表示语言的意义，一般来说，语义学研究语言和它所指对象之间的关系。SMDLan 的语义研究意义重要且影响深远：一方面，有助于我们理解移动 Agent 的迁移本质；另一方面有助于严格界定 SMDLan 的语义，保证移动 Agent 迁移程序的正确性。通常，语义的研究有操作语义、公理语义、代数语义和指称语义几种途径。我们采用了操作语义的途径，给出了 SMDLan 的形式操作语义框架。

操作语义的基本思想是用抽象方法描述语言中每一成份的执行效果，以免所描述的语义依赖于该语言实现时所用的具体计算机。通常是设计一台抽象机，定

义一组抽象状态，把语言的语法表示成抽象的形式，然后指明抽象机每加工一个语言成份时对状态作何种修改[123]。

我们采用结构化操作语义来定义操作语义框架，为了给出一个程序设计语言的 SOS 描述，应同时列出其语法范畴，语法规则（含静态语义）及动态语义。其中语法范畴是指该语言使用的所有语法符号，语法规则（含静态语义）给出所有合法的语句结构，其中基本规则以公理形式给出，辅助规则以推理规则给出，动态语义以转换三元组的形式给出。

为了方便表示，先引入条件语句和条件表达式结构：

定义 1（条件语句）：条件语句的形式为 P→E，当且仅当条件 P 为真时才回送相应值 E，否则相当于执行空语句。

定义 2（条件表达式）：条件表达式的一般形式是：（P_1→E_1，P_2→E_2，…，P_n→E_n），其中要求至少有一个 P_i（谓词）为真，此时整个表达式的值即是与第一个取真值的 P_i 相对应的 E_i。为了使该条件表达式一定有值，可以扩充为下列形式：

（P_1→E_1，P_2→E_2，…，P_n→E_n，E_{n+1}）

下面从语法范畴，语法规则和动态语义三方面来描述 SMDLan 的结构化操作语义框架。

（1）语法范畴

①移动 Agent 状态是移动 Agent 在某个时刻所有变量值的集合，以 ms 表示。移动 Agent 生命周期中所有移动 Agent 状态的集合，称为 MS。

②移动 Agent 迁移目标地址是指移动 Agent 下一次迁移的目的地，以 h 表示。移动 Agent 生命周期中的所有迁移目标地址的集合，称为 MH。

③移动 Agent 环境状态是指移动 Agent 在某个时刻可以存取的网络环境变量的值的集合，其中应有 IP 形式的主机地址。环境状态以 host 表示。移动 Agent 生命周期中所有移动 Agent 环境状态的集合，称为 HA。

④一组旅行计划步，称为 IS，成员以 s 表示。s 可以是顺序旅行步，也可以是并行旅行步。

⑤旅行计划 t。t 由多条相同类型的旅行计划步构成。T 表示旅行计划集合。

⑥一组迁移卫士函数，称为 MG，成员以 mg 表示。迁移卫士函数是一个布尔函数或者布尔表达式，它根据当前有关的网络状态进行判断，回送一个布尔值，并不修改移动 Agent 状态。

⑦一组入口卫士函数集，称为 EG，成员以 eg 表示。入口卫士函数是一个布尔方法或者布尔表达式，回送真或者假，不修改移动 Agent 状态。

⑧一组入口方法，称为 RF，成员以 e 表示。入口方法是移动 Agent 功能体中实现用户任务逻辑的方法，修改移动 Agent 状态。

⑨一组旅行步更新函数，称为 IUF，成员以 u 表示。旅行步更新函数可以动态修改或创建一个移动 Agent 旅行计划，让移动 Agent 根据新的旅行计划继续旅行。

（2）语法规则

以公理形式给出 SMDLan 的语法规则，⊢s 表示 s 是合法的顺序旅行步，⊩s 表示 s 是合法的并行旅行步，⊢t 表示 t 是合法的旅行计划。

① ⊢Null

② ⊩Null

③ ⊢MIG mg=>h RES eg=>e UPD t

④ ⊢MIG mg=>h RES eg=>e UPD t NEWN n

⑤ $\dfrac{\vdash s_1 \,\&\, \vdash s_2}{\vdash s_1; s_2}$ 其中 s_1，s_2 为任意顺序旅行步

⑥ $\dfrac{\Vdash s_1 \,\&\, \Vdash s_2}{\Vdash s_1; s_2}$ 其中 sl，s2 为任意并行旅行步

⑦ ⊢SEQ[Null]ENDSEQ

⑧ ⊢ALL[Null]ENDALL

⑨ $\dfrac{\Vdash SEQ[s_1; s_2; \cdots s_n]ENDSEQ \,\&\, \vdash s}{\Vdash SEQ[s_1; s_2; \cdots s_n; s]ENDSEQ}$ 其中 s，s_i 为任意顺序旅行步

⑩ $\dfrac{\Vdash ALL[s_1; s_2; \cdots s_n]ENDALL \,\&\, \vdash s}{\Vdash ALL[s_1; s_2; \cdots s_n; s]ENDALL}$ 其中 s，s_i 为任意并行旅行步

⑪ $\dfrac{\Vdash SEQ[s_1; \cdots s_{i-1}; s_i; s_{i+1}; \cdots s_n]ENDSEQ}{\Vdash SEQ[s_1; \cdots s_{i-1}; s_i; s_{i+1}; \cdots s_n; s]ENDSEQ}$ 其中 s_i 为任意顺序旅行步

⑫ $\dfrac{\Vdash ALL[s_1; \cdots s_{i-1}; s_i; s_{i+1}; \cdots s_n]ENDALL}{\Vdash ALL[s_1; \cdots s_{i-1}; s_i; s_{i+1}; \cdots s_n; s]ENDALL}$ 其中 s_i 为任意并行旅行步

（3）动态语义

首先，给出若干定义和约定：

定义 3（顺序旅行步）：mg 和 eg 表示任意布尔函数或表达式，h 表示一个网络地址，e 表示一个移动 Agent 方法，u 表示一个旅行计划的更新方法，五元组<mg，h，eg，e，u>称为一个顺序旅行步，表示当 mg 为真时，移动 Agent 将迁移至 h 节点，在 h 节点上，当 eg 为真时，将调用 e 方法，当 e 执行结束时，将执行 u 方法。

定义 4（并行旅行步）：mg 和 eg 表示任意布尔函数或表达式，h 表示一个网

络地址，e 表示一个移动 Agent 方法，u 表示一个旅行计划的更新方法，n 表示一个新生移动 Agent 的名字，六元组<mg，h，eg，e，u，n>称为一个并行旅行步，表示将“克隆”出一个新的移动 Agent，以 n 为名，以顺序旅行步<mg，h，eg，e，u>为这个新生的移动 Agent 旅行计划的全部旅行步，进行旅行。

在此基础上，为便于表达 SMDLan 的操作语义，给出下列辅助函数定义：

定义 **5**（**State** 函数）：State：MH→HA，表示获得 MH 中元素 h 节点上的网络状态。

定义 **6**（**Guard** 函数）：Guard：HA×MS×MG→Bool，表示在网络环境 host 中，在移动 Agent 的状态 s 下计算函数 g 所得到的布尔值，为真或者为假。

定义 **7**（**Exec** 函数）：Exec：HA×MS×RF→MS，表示在网络环境 host 中，在移动 Agent 的状态 s 下进行方法 m 的运行。该方法的运行，将改变移动 Agent 的当前计算状态，我们标记 Exec 函数回送的新状态为 s'。

定义 **8**（**More** 函数）：More：T×MH×MS→Bool，以 t∈T 表示一个顺序旅行计划，其结构形如 SEQ[S_1，S_2，…，S_n]ENDSEQ，n≥0。其中的任一 s_i 都是一个<mg，h，eg，e，t）的顺序旅行步五元组。s∈MS，h∈MH。采用条件表达式方式，定义 More 如下：

$$\text{More}(t,\ h,\ s)=\bigvee_{i=0}^{n} Guard(State(h),s,mg_i)$$

该函数判断在顺序旅行计划 t 中，是否有剩余的有效旅行步。

定义 **9**（**ItiCreate** 函数）：ItiCreate::A×MS×IUF→T。表示在当前网络环境 host 中，在移动 Agent 的状态 s 下进行方法 u 的运行。运行结果会产生一个新的旅行计划，标记为 Iti'。

定义 **10**（**Update** 函数）：Update：MH×MS×IUF×T→T。h∈MH，表示当前节点位置，s∈MS，表示当前状态，u∈IUF，表示旅行计划更新方法，t∈T，表示一个顺序旅行计划，其结构形如 SEQ[s_1，s_2，…，s_n]ENDSEQ，n≥0。采用条件表达式方式定义函数如下：

Update（h，s，u，t）＝（More（t，h，s）→t；ItiCreate（State（h），s，u））

Update 函数将回送一个新的旅行计划，标记为 t'。

定义 **11**（组态）：host 表示移动 Agent 当前所处的网络环境状态；cs 表示移动 Agent 当前的计算状态（可以理解为移动 Agent 在完成计算任务过程中的全体（变量，值）构成的集合）；it 表示移动 Agent 当前尚未完成的旅行计划，三元组<host，cs，it>称为一个组态，表示在当前网络环境 host 中，移动 Agent 状态为 cs 时，尚

需完成 it 表达的旅行任务。此外，<host，cs>以及<host>也称为一个组态，表示当前旅行任务已经全部完成，不再进行任何移动行为。

定义 11（终极组态）：终极组态是指形式上表现为<host，cs>、<host>的组态，或者是这样的组态<host，cs，it>，对它们不存在另一个组态<host'，cs'，it'>，使得<host，cs，it>→<host'，cs'，it'>成立，并且 it≠it'。

定义 12（组态转换）：<C，T，R>称为一个转换三元组，其中 C 是全部可能的组态的集合，T 是全部终极组态的集合，R 是转换关系集，是全部公理和推理的集合，满足 R≦C×C，其中每个关系可写为<r，r>，其直观形式是：r →r

定义 13（规则）：

（1）State（h）=host 是一个规则，其中 host 为网络状态；

（2）Guard（host，s，g）=b 是一个规则，其中 b 为布尔值；

（3）Exec（host，s，m）=s'是一个规则，其中 s'是移动 Agent 状态；

（4）Update（h，s，u，t）=t'是一个规则，其中 t'是旅行计划；

（5）<host，cs，it>→<host'，cs'，it'>是一个规则，表示从组态<host，cs，it>出发，经过若干旅行步的旅行，到达新的组态<host'，cs'，it'>。当 it'为 Null 时，上式右边变为<host'，cs'>；

（6）有限个规则的并和交仍是规则；

（7）若 r_1 和 r_2 是规则，则 $\frac{r_1}{r_2}$ 也是规则，表示若 r_1 成立，则 r_2 也成立。

下面给出 SMDLan 的结构化操作语义的动态语义描述：

（1）<host，cs，SEQ[Null]ENDSEQ>→<host，cs>

（2）<host，cs，ALL[Null]ENDALL>→<host，cs>

（3）顺序旅行语句的动态语义：

规则 1：

$$\frac{Guard(host,cs,mg)=False}{<host,cs,SEQ[<mg,h,eg,e,u>;S_2;...s_n]ENDSEQ>\rightarrow<host,cs,SEQ[s_2;s_3;...s_n]ENDSEQ>}$$

规则 2：

$$\frac{Guard(host,cs,mg)=True\ \&\ Guard(State(h),cs,eg)=False}{<host,cs,SEQ[mg,h,eg,e,u>;s_2;...s_n]ENDSEQ>\rightarrow<State(h),cs,update(h,cs,u,SEQ[s_2;...s_n]ENDSEQ)>}$$

规则 3：

$$\frac{Guard(host,cs,mg)=True\ \&\ Guard(State(h),cs,eg)=True}{<host,cs,SEQ[<mg,h,eg,e,u>;s_2;...s_n]ENDSEQ>\rightarrow<State(h),Exec(State(h),cs,e),update(h,cs',u,SEQ[s_2;...s_n]ENDSEQ)>}$$

（4）并行旅行计划的动态语义

一个并行旅行计划的执行将克隆出多个移动 Agent，每个移动 Agent 将以顺序旅行方式进行相应的迁移，其顺序旅行计划由对应序号的单条旅行步构成。因此，我们将并行旅行计划的动态语义描述如下：

<host，cs，ALL[S_1；S_2；…S_n]ENDALL>→

[<host，cs，SEQ[S_1]ENDSEQ>，

<host，cs，SEQ[S_2]ENDSEQ>，

……

<host，cs，SEQ[S_n]ENDSEQ>]

3.3.5 相关工作比较和结构化迁移机制的特点

从系统实现上看，移动 Agent 的迁移分线程迁移和对象迁移，线程迁移又称强迁移，移动 Agent 迁移时线程尚未执行结束，系统必须捕获所有执行线程的状态，封装移动 Agent 执行环境并完成上述内容的迁移，因此系统工作量大，迁移时通信量也大。另外，因标准 JVM 不支持线程运行状态的捕获，采用 JAVA 语言实现线程迁移时，要么修改 JVM，要么系统自行完成该任务，所以除 General Magic 公司的 Telescript 和 Dartmouth 学院的 Agent Tcl[38]采用自行设计的语言实现线程迁移外，大部分使用 JAVA 的移动 Agent 系统都采用了对象迁移。对象迁移又称弱迁移，是指迁移时系统只考虑 Agent 对象的迁移，不考虑其执行状态的迁移，一般以 Java 为实现语言的系统皆以 JVM 提供的序列化机制为基础来实现。如 Aglets，Concordia 和 IBMAS。相对而言，对象迁移的系统实现简单。

从用户的角度看，迁移又分语句级和过程级迁移。语句级迁移指迁移信息和动作命令在程序体中描述，迁移行为在线程结束前发生。如 Telescript、Agent TCL、Aglets 等。语句级迁移涉及线程状态的处理，如 Telescript 和 Agent TCL 均采用了线程迁移，而 Aglets 则在对象迁移的基础上将线程状态的处理工作交给了用户，用户负担较重。而过程级迁移将迁移信息和 Agent 功能体完全分离，没有显式的迁移指令，迁移动作不发生在线程执行过程中，Agent 在某个 host 上执行的所有线程运行完毕即进行下一次迁移，Agent 系统根据旅行计划自动完成该迁移动作。显然，过程级迁移可用对象迁移技术进行实现。如 Concordia 和 IBMAS 就属过程级对象迁移。

综合系统实现和用户使用两方面的因素，我们认为，以 Aglets 为代表的语句级对象迁移和以 IBMAS 为代表的过程级对象迁移较为合理，其中后者更为理想，原因如下：

① 语句级迁移的迁移指令可在程序中书写，其迁移前置条件及后继处理等迁移信息是隐含在程序的上下文中的，因此其描述功能较强，能够表达较复杂的迁移情况，但是，如将迁移的描述、表达和程序设计相比较，Telescript 中的 Go 方法、Aglets 中的迁移机制等语句级迁移相当于程序设计中的 Goto 机制。相比之下，IBMAS 的迁移机制不仅能完整地刻画迁移行为中的必要信息，它还提供了三种类似于结构化程序设计中的顺序、分支和循环的移动方式及多种旅行计划的修改、更新手段，因此，在极端情况下，IBMAS 迁移机制可以模拟实现全动态的语句级迁移。例如可以使用单步指令顺序移动方式及旅行计划动态调整来实现语句级迁移。因此 IBMAS 虽然是过程级迁移，但也有足够的能力和灵活性处理各种迁移现象。

② 以 Aglets 为代表的语句级对象迁移虽然能有效减轻系统的负担，但它是以增加用户负担为代价的。用户必须在迁移前收集足够的线程状态信息，在恢复时根据这些信息自行寻找恢复点。而 IBMAS 的用户完全没有这方面的负担。因此，过程级迁移和对象迁移实现是一种更自然的组合。

③ 语句级迁移中迁移信息和 Agent 代码是混合在一起的，一方面增加了用户的编程负担，不能集中于 Agent 功能体的设计和实现；另一方面，降低了 Agent 移动的灵活性，不能很好地适应网络环境的多变性及在不同网络环境中 Agent 的复用。而过程级迁移中旅行计划和功能体的分离则减轻了用户的编程负担，用户可进行脱离具体移动环境的功能体的单独开发，也可进行纯移动功能的开发。旅行计划的动态装配增加了网络环境中 Agent 移动的灵活性，可较好地适应网络环境的多变性。此外，在不同网络环境中 Agent 的复用度也可以得到保证。因此，用户更乐意使用过程级迁移。

另外，就 Concordia 和 IBMAS 而言，两者的旅行计划和功能体都是分离的，但是 Concordia 旅行计划的描述信息较少，也未提供对旅行计划进行动态调整的手段，因此，其能力和灵活性都不够，不能有效地表达各种迁移现象，只是 IBMAS 的特例，IBMAS 具有更强的迁移灵活性和描述能力；其次，Concordia 要求用户必须额外设计一个完成旅行计划制订和装配的类，而 IBMAS 提供了 Agent 类库和旅行计划库，提供了一个图形化的开发工具，用户可以方便地使用该工具分别完成上述装载、装配和运行工作；最后，IBMAS 旅行计划在 IBMAS 运行过程中可以根据环境动态调整，较好地体现了 Agent 特有的自适应能力，这是 IBMAS 的特点。

系统	迁移表达	迁移信息模型	入口方式	网络适应性	复用性
Aglet	命令方式	无	固定入口	较差	差
Grasshopper	命令方式	无	固定入口	较差	差
Concordia	计划方式	目的地列表	多入口	较差	一般
IBMAS	旅行计划	结构化旅行计划模型	多入口	较好	较好

表 3-12 IBMAS 结构化迁移机制同其它代表性系统迁移机制的比较

总之，IBMAS 中的结构化迁移机制具有以下特点：

①结构化程度较高，易于使用。IBMAS 中旅行计划和功能体的完全分离，使其可独立开发，装配使用，易于 Agent 的理解、开发使用和复用。

②功能较强。IBMAS 中的迁移机制有足够的能力和灵活性表达多种迁移现象。Concordia 模型可视为 IBMAS 中的特例。

③动态方式和静态方式有机结合，具有较强的灵活性。在极端情况下，该机制可以表达完全动态的迁移。

④便于对迁移机制进行深入研究。旅行计划具有完全的独立性和良好的结构，便于定义其上的操作，抽取其设计模式，刻画其语义，研究其本质。

⑤系统实现采用过程级的对象迁移，简单高效且用户负担较小。

3.4 本章小结

基于 Internet 的移动 Agent 技术可以实现数据、代码和计算的同时移动，表现出良好的网络适应性，提供了一种全新的、统一的网络计算模式，引起了学术界和工业界的广泛关注。然而，在许多移动 Agent 系统中，还存在以下问题：缺乏全面、系统地对移动 Agent 计算的支持系统；在若干关键技术上，如 Agent 的迁移机制、Agent 的通信机制、Agent 的安全机制、Agent 的开发和运行支持等方面，尚有不足之处。

在移动 Agent 迁移机制方面，本章提出了将迁移机制和功能体分离的思想，并在此基础上提出了能力较强的结构化迁移机制，不仅简化了移动 Agent 的理解、开发、使用和复用，而且也较好地体现了移动 Agent 的灵活性和网络适应性。

4 通信机制和安全机制的设计

4.1 本章概述

如何实现远程 Agent 通信的位置透明性、保证消息不会因为目标 Agent 迁移而丢失。一直是移动 Agent 通信所面临的难题，在现有的很多移动 Agent 系统中都没有得到解决。我们在 IBMAS 中提出的通信算法初步实现了通信的位置透明性和可靠的消息传输。另外，移动 Agent 系统中的安全问题是阻碍其广泛应用的原因之一。通过对 Agent 的安全问题及其研究现状的分析，提出了解决一些安全问题的方法，对防止和解决移动 Agent 系统中的安全问题有一定的作用。

本章主要阐述移动 Agent 的分层多模式通信框架、安全体系的设计、及上述内容和相关工作的比较。

4.2 分层多模式通信机制的设计[42,57-62,125-126]

移动 Agent 的本质是代表用户在网上寻找合作伙伴，进行交互并最终完成用户指派的任务。因此，协作性是移动 Agent 的必然要求，移动 Agent 环境必须提供移动 Agent 对周围合作伙伴的感知需求以及基本的通信需求。下面将在分析移动 Agent 协作需求的基础上，介绍一种分层多模式通信框架，详细分析该框架中间接通信层的基本成份以及提高移动 Agent 通信可靠性的一个通信失效解决算法。最后和相关工作进行了比较[125]。

4.2.1 协作的目的

协作性是移动 Agent 的必然要求，它包括以下几个方面：

（1）合作　每个 Agent 的知识是有限的，因而能力也是有限的，然而用户的需求可能是复杂的，可能需要多个完成特定功能的 Agent 合作才能圆满完成任务，因此，这些 Agent 之间有必要进行合作。

（2）管理　在网络上漫游的 Agent，没有一个 Agent 能够具有全局的控制能力，每个 Agent 都在寻找合适的合作伙伴，这些 Agent 会导致网络上 Agent 的混乱局面，因此，必须提供一种协调能力，对 Agent 进行有序的管理，降低网络上移动 Agent 的复杂程度。

（3）交流　移动 Agent 在网络上漫游，其合作伙伴可能是陌生的，它必须具有对外界环境和陌生 Agent 的感知能力，了解和交换各自的能力，从而开展合作，因此，交流的需求也是移动 Agent 协作的一个本质。

（4）协同　协作 Agent 之间的关系不是相互独立的，必定会存在一些内在的相互依赖关系，这使得 Agent 之间的协作可能会出现各种方式，比如：同步、异步、组协同等。

4.2.2 协作的技术需求

Agent 协作体现为若干移动 Agent 在网络中相互协作并合作完成某一任务，主要包括功能互通、协作联盟/模式和通信机制三个层次：

①功能互通：功能互通主要解决陌生 Agent 之间如何相互“认识”并深入了解对方的功能，以便开展有效协作，功能互通不可避免地涉及到语义层次，目前除了在理论研究方面进行一些语义理解的工作以外，其它方面代表性的工作有 XML 处理机制和最新的“语义网络”（Semantic Web）的工作[50,127]；

②协作联盟/模式：对于多 Agent 系统的构建具有重要意义，其主要目的在于提供 Agent 之间开展协作的方法，主要的工作有基于集中管理的组织模式，如 Master/Slave 和 Client/Server 模式，以及基于分布协同的合同网模式；

③通信机制：功能互通和协作模式两方面的研究在智能 Agent 领域比较普遍，而在移动 Agent 系统中，对软件 Agent 协同性的支持主要集中于通信机制的研究。代表性的研究工作有两类，一是基于知识交换的 KQML 等工作，二是基于消息传递的 Aglets 系统等。

IBMAS 中移动 Agent 协同技术的研究主要集中在通信机制上，我们仔细分析了现有通信机制，发现存在两方面的不足：

①对移动 **Agent** 的通信支持不够全面。一方面，上述知识交换以及消息传递的工作实际上是分离的，没有同时支持 KQML 和消息传递两种协作方式的通信机制；另一方面，在实际应用中，移动 Agent 之间的信息交流是多级别、多层次的，因此，我们认为一个系统的移动 Agent 通信机制应支持 Agent 之间开展不同层次的协作通信。

②对通信可靠性的支持不够。网络环境的多变性，Agent 本身也在不断运动，在一个多变的环境中，运动着的 Agent 之间很容易产生各种通信异常，如安全问题、容错问题、通信失效问题、通信因果顺序问题等。这些问题如不解决，移动 Agent 之间的协作便不可能顺利进行。

基于上述分析，在 IBMAS 通信机制的设计中，我们提出了一个分层多模式通信框架，支持多层次、多模式的 Agent 信息交流。

4.2.3 分层多模式通信框架[57-62]

移动 Agent 之间的信息交流是多方面的，包括以下三个层次：

①移动 Agent 之间可能需要传输大量数据，这种层次的信息交流往往只是完成数据的转移，数据的处理逻辑并不重要。这种情况下，交流的主要需求是性能上的要求，此时系统应提供或允许用户使用高效的传输通道；

②数据交换的通信手段要求用户对网络传输比较了解，很多的底层细节需要用户自行处理，显然不能用于所有的信息交换，因此，移动 Agent 往往采用消息传递模型来进行不涉及大量数据的、简单的处理信息和数据信息的传递。消息传递机制具有较好的灵活性和较强的处理能力，可以帮助用户完成这种信息交流；

③消息传递机制仍然是一种将消息传递的技术细节留给用户的做法，用户无法利用这种机制进行复杂的通信协作模式和协议的建立和实施。因此，在 Agent 协作领域出现了基于 XML 或 KQML 的高层的基于知识交换的通信模式，支持用户进行知识交换和自动处理。

因此，我们在通信机制的设计过程中，提出了一种分层多模式的通信框架，其体系结构见图 4-1。

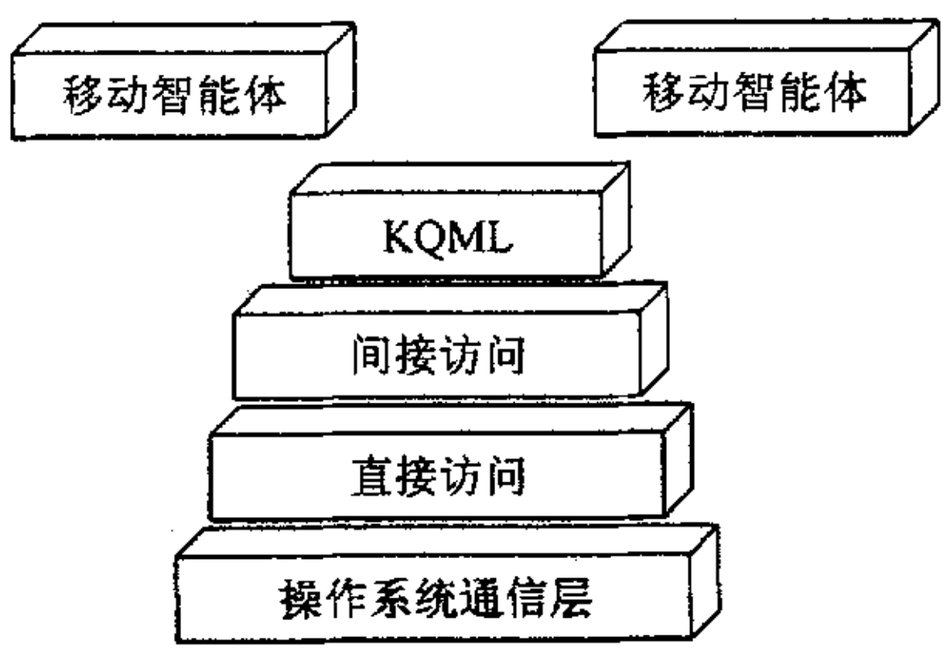


图 4-1 分层多模式通信框架

其中 KQML 层支持移动 Agent 之间进行高层的基于知识交换的协作，间接访问模式以消息传递为通信手段，直接访问模式允许 Agent 直接进行基于网络协议的数据传输。移动 Agent 可以使用其中的任意一种模式进行通信协作。

从技术支持上看，我们采用自底向上的方法构建整个通信机制，每一底层都向紧邻上层提供实现支持。目前的工作主要集中于间接访问层的构建上，在 JVM 的基础上，遵循 MPI 标准，设计和实现 IBMAS 的通信机制。

4.2.4 通信基础平台

通信基础平台中设置了双层名命名机制，为 Agent 通信提供了透明寻址，采用基于消息传递的通信方式，仿照 MPI，提供了一组常用的、完整的同步和异步

通信原语，支持 IBMAS 进行点对点、广播和组通信。其基础架构采用一种基于 Home-Communicator 的通信模型。每个 Agent Server 中拥有一对上述构件。其中 Home 构件负责记录该 Host 上创建的 Agent 的动态信息，和 Communicator 合作完成透明寻址和通信失效解决工作。Communicator 构件记录该 Host 上所有 Agent 的信息，负责 Agent 之间通信的目标寻址、信件转发和通信失效解决等工作。既为处于同一台主机的 Agent 提供了高效的本地通信方式，又支持不同主机上 Agent 之间的异地通信，为 Agent 用户提供了较大的灵活性。如图 4-2 所示。

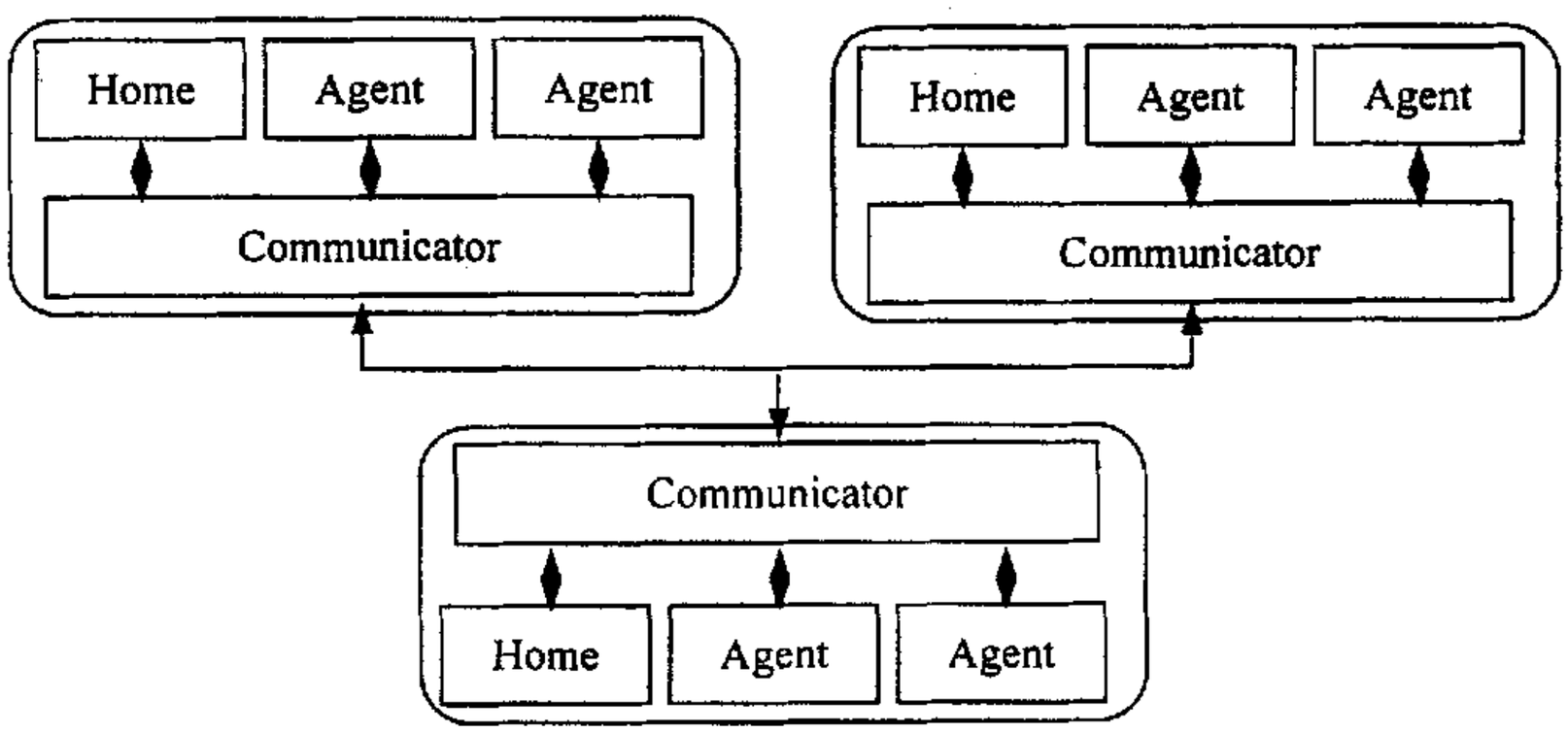


图 4-2 间接访问模式通信结构

4.2.5 Agent 的寻址和通信

要实现通信，必须明确消息接收者所处的地址，而移动 Agent 位置不断变更为目标 Agent 的寻址带来了困难。因此，一个实用的移动 Agent 系统应允许用户通信时只需指定通信目标 Agent 名称，系统自动完成从 Agent 名到其物理位置的映射，并完成信件传输。即实现 Agent 的按名使用，透明寻址。

Agent Server 中的 Home-Communicator 构件可以合作完成 Agent 通信的寻址和信件转发工作。其中，Home 构件负责记录该 Host 创建的 Agent 的动态信息，Agent 每次迁移到一个新的 Host，必须及时向创建该 Agent 的 Host 上的 Home 登记其包含当前地址信息的物理名。而 Communicator 构件记录该 Host 上所有 Agent 的信息，负责 Agent 之间通信的目标寻址、信件转发和通信失效解决等工作。

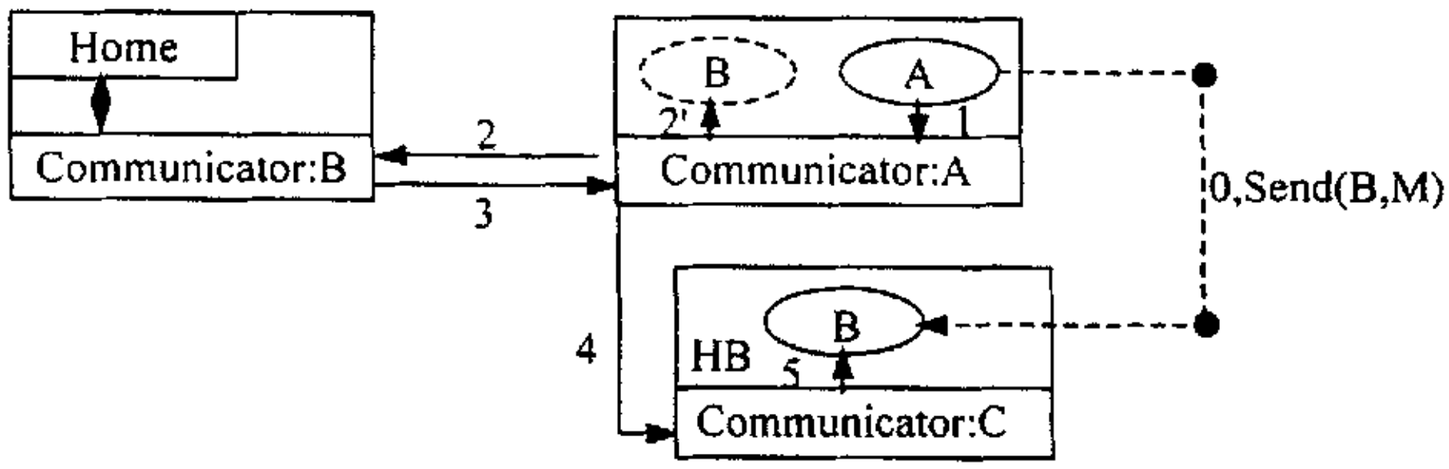


图 4-3 Agent 通信中的寻址和信件发送

如图 4-3 所示，Agent 要求与其它 Agent 通信时，可以使用目标 Agent 的名通过通信原语直接通信。通信请求被本地 Communicator 截获（如 1），Communicator 首先检查目标 Agent 是否也在本地。若是，则直接将消息放入其信箱（如 2'），从而实现高效的本地通信；否则，根据目标 Agent 名查出其逻辑名，向它的 Home 询问其物理名（如 2，3），按照物理名中包含的当前地址信息，将消息发往该地址（如 4）。消息被目标主机的 Communicator 截获，放入目标 Agent 的信箱中（如 5），从而完成一次通信。从以上通信过程中可以看出，目标 Agent 的寻址对程序员完全透明，只要在程序中指出目标 Agent 名，便可完成通信。

采用这种基于 Communicator 的辅助通信方式，由 Communicator 集中实现寻址和消息路由，而不是由 Agent 获得目标地址后直接建立网络连接发送消息，有以下优点：①便于使用；②灵活性较强；③Communicator 功能易扩充。

4.2.6 通信可靠性研究

4.2.6.1 通信失效定义和现象分析

在这种基础通信架构中，移动 Agent 依靠通信来完成其合作。在假设无故障的理想状况下，网络通信协议如 TCP/IP 可以保证无网络位置变迁的“静止”对象或软件之间的通信是“有效”的，即软件 A 向 B 发出的一封信在有限时间内一定会被 B 收到。但是在移动 Agent 的通信环境中，因为通信中的主体 Agent 在任意时刻都可能发生网络位置的变化，故单纯的网络通信协议已不能保证其正常的通信，例如：Agent A 向 Agent B 发出一封信，当该信在途中时，Agent B 移动到了另一个地址，B 将不能及时收到该信。这种在移动 Agent 环境中因通信主体物理位置变化而造成不能正常通信的现象称为移动通信失效。通信失效是移动 Agent 协作的致命缺陷。它使得协作中的 Agent 不能及时甚至无法正确得到协同信息，从而导致协作的失败甚至是错误的协作。由于问题复杂，大多试验性系统在这方面的处理均不理想，如德国斯图加特大学的 Mole 系统，在提及该问题时，给出了三

种方案：①将失效信件抛弃；②抛弃的同时回送错误信息；③将失效信件就地保存，待被通信 Agent 回送时交付，显然，这几种方法难以令人满意，并未真正解决问题。

针对移动 Agent 的通信失效现象，我们详细分析了移动通信失效的现象和根本原因，在此基础上，提出了一个能有效解决通信失效问题的架构模型，并给出一个示例。

上述模型和路由通信算法能有效地支持大部分情况下的移动 Agent 之间的本地和异地通信协作，但它不能排除通信失效现象的出现，在以下几种情况下可能会出现本地或异地的通信失效：

① 本地通信失效：如图 4-4，当 A 和 B 同处一地时，C1 在截获通信请求分析出 B 在本地后，如果 B 发生了位置移动，此时 C1 再执行 2，向 B 的信箱送信就会出现异常。

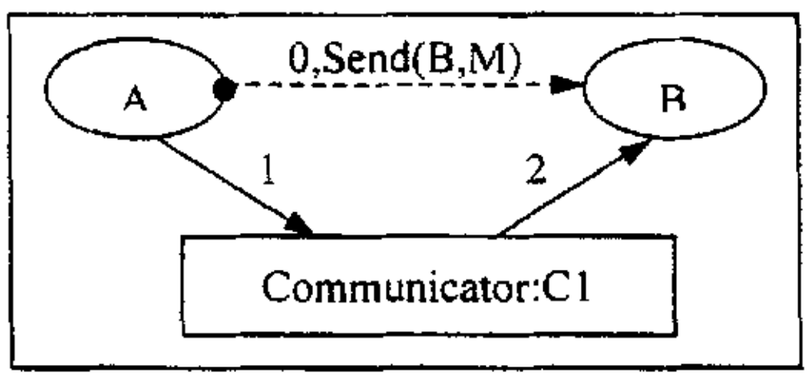


图 4-4 本地通信失效

② 异地通信失效：如图 4-5，在第 3 步或第 4 步过程中如果 B 发生了位置变化，迁移到了 H3，C2 所执行的第 5 步将会出现异常。

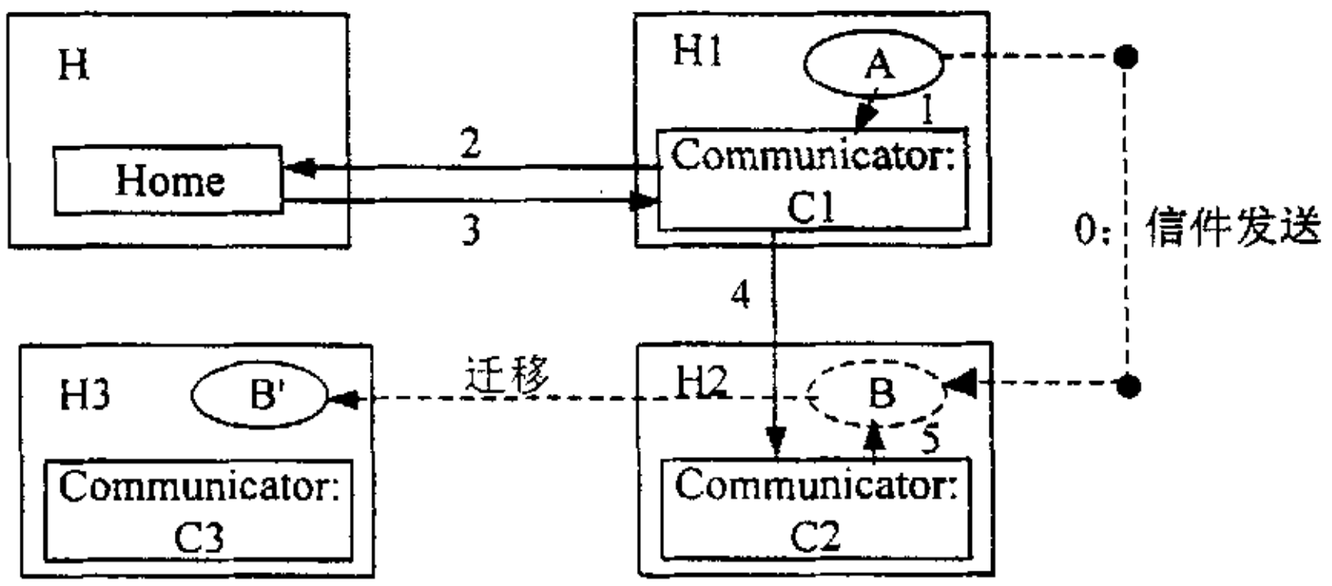


图 4-5 异地通信失效

仔细分析上述两种现象可知，虽然它们的表现形式不同，但其本质是一样的：

①无论本地通信还是异地通信，其通信模型是一致的。Communicator 获得目标 Agent 名和相关信件，通过路由机制获得目标 Agent 的地址，然后向该 Agent 发送信件。其中本地通信只是该模型的简化。

②通信失效本质上都是因为在路由信件和实际信件传输过程中，目标 Agent

发生了物理位置的变化，而这种变化是随机的，不可预知的。

③从操作系统的角度来讲，通信和移动所共享的“位置”信息未进行同步控制是造成通信失效的根本原因。当两者竞争使用该资源时，移动优先于通信的做法将产生通信失效，通信优先于移动的做法将束缚移动的完成，造成安全和协作上的隐患。只有以移动请求和通信请求发生的时间为使用权的衡量标准才能合理、有效地解决通信失效问题。

实际上，通信失效现象并不是 IBMAS 特有的现象。它是因 Agent 移动带来的可能会出现在任意一个移动 Agent 系统中的普遍现象。

4.2.6.2 通信失效问题解决方案基本思路

通信失效是由信件接收者的随机移动造成的，主要问题在于通信和移动所共享的“位置”信息的同步使用问题，也就是 Agent 位置的改变。因此，我们从区分 Agent 的移动状态和非移动状态的角度出发，将一个移动 Agent 的生命周期划分如下图 4-6：

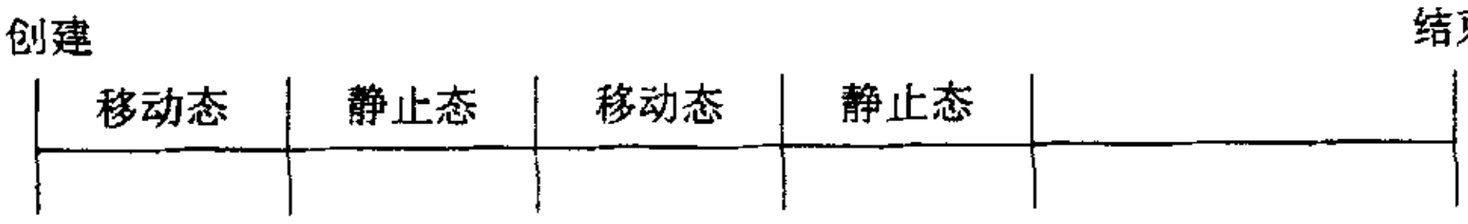


图 4-6 Agent 状态划分

其中 Agent 处于“静止态”指 Agent 从到达某个站点被恢复执行开始，到该 Agent 开始向另一个站点迁移为止的这一段时间内所表现出的物理位置的无变化状态。“静止”不是指 Agent 不执行，而是指不发生位置的变化。Agent 处于“移动态”指 Agent 从开始迁移直到在某一站点被恢复执行为止这一段时间内所表现出的物理位置的不确定状态。显然，处于移动态的 Agent 的“移动”拥有“位置”资源的使用权，只有当它处于“静止态”时，才能接受信件，“通信”才能使用“位置”资源，反之，在通信过程中，“移动”将被限制，Agent 不允许发生状态的变化，否则，同样会出现通信失效。具体来讲，如果向一个处于“移动态”的 Agent 发送信件，必定会出现通信失效现象。此外，因为网络传输的时间延时，信件传输过程中，如果通信对象 Agent 发生了从“静止态”到“移动态”的状态变化，也必定会造成通信失效。因此，接收者的“状态”在通信失效问题中具有决定性的意义，当通信和移动相矛盾时，该“状态”就成为了一个必须互斥使用的“资源”。解决了这个互斥问题就能解决通信失效问题。所以，我们认为，在一个能够避免通信失效的移动 Agent 系统中，必须做到以下三条：

①准确记录 Agent 的状态信息。

②只能向一个处于“静止态”的 Agent 发送信件。

③信件发送过程中必须限制接收者从“静止态”向“移动态”的状态转换。

基于以上分析，我们在 IBMAS 中原有通信机制的基础上，增加了相应的设施，修改了相关路由通信和移动管理算法。新的通信机制一方面有效地解决了系统中的通信失效问题，另一方面合理地处理了 Agent 的移动和通信的矛盾。

4.2.6.3 通信基础架构的改进

（1）Home 结构的改进

首先，我们增加了 Agent 状态描述信息，Agent 状态分为静止态和移动态，该值由 Agent 在原创建节点中的 home 结构维护，其状态的设定和修改遵循以下规则：

①当一个 Agent 在某个节点上执行时，其状态值定义为静止态。

②当一个 Agent 按旅行计划准备向另一个节点迁移时，当前节点上的 Host Server 必须通知该 Agent 的 Home，Home 接受到通知后应将状态值改为移动态。

③当一个 Agent 到达某个新节点时，新节点上的 Host Server 必须向 Agent 的 Home 通知该 Agent 的到达，Home 收到通知后将状态值改为静止态。

实际上，上述三条可以“准确记录 Agent 的状态信息”。

其次，在 Home 中增加了“寻址信件回复等待队列”，负责管理移动态 Agent 对应的 Home 收到寻址信件后的回复工作。

（2）Communicator 的改进

在 Communicator 中，为每个 Agent 设置了“在途信件数”：Mail_Num_0nTrip。该值记录了当前时刻以该 Agent 为通信对象的在途信件数。“在途”指 Home 回复了寻址信件后开始，直到这次通信的信件到达 B 为止的这段时间。以图 4-7 为例，“在途”涵盖了 3，4，5 三段时间。显然，Home 在回复了寻址信件后应及时向 H2 提交“增加在途信件”通知。H2 在完成 5 后应及时将该值减 1。

实际上，“在途信件数”非 0 反映了目前“通信”功能已经在使用“位置”资源，所以，当一个 Agent 准备离开某个节点时，它所对应的“在途信件数”必须为 0，否则，移动要求将被暂缓。

（3）路由算法的改进

以图 4-7 为例，当 C1 截获 A 的通信请求后，C1 向 H 中的 Home 结构发去一封寻址信件（2），Home 收到该信后，分析 B 的状态，当状态为“静止”时，Home 回复 C1，告知 B 的地址是 H2（3），然后向 H2 的 Communicator 发去一封“增加一封在途信件”的通知（3’）。如果 B 的状态是“移动态”，Home 将该寻址信件放入“寻址信件回复暂缓队列”中，直到未来某时刻 B 的状态变为”静止态”，再一

次回复所有“暂缓信件”。

这种路由算法实际上保证了“不向移动态的 Agent 发送信件”。

（4）移动管理的改进

在新的通信机制中，Agent 的移动管理在两方面进行了改进以满足移动和通信协调的需要（以图 4-7 为例）：

①B 准备向 H3 移动时，B 先向 H2 的 Host Server“申请移动”，Host Server 通过 C2 向 H 的 Home 提交“移动登记”（图中Ⅰ）。Home 在收到“移动登记”信件后，先将 B 的状态改为”移动态”，然后回复 H2“登记成功”（图中Ⅱ）。当 H2 收到“登记成功”后，检查 B 对应的“在途信件数”是否为 0。如果非 0，B 的“申请移动”将被堵塞直到“在途信件数”减为 0。这个方法能有效地在通信尚未完成时暂缓 Agent 的移动。

②当 B 成功地移动到 H3 时，B 应通过 C3 向 H 的 Home 发送“到达登记”（图中Ⅳ），Home 应修改 B 的地址和状态并处理可能的“寻址回复暂缓队列”。

（5）信号量设置

系统为每个 Agent 引入了 2 个信号量：迁移状态和在途信件数。

①迁移状态：保存在 Agent 对应的 Home 结构中。它记录 Agent 是否处于迁移过程中，其值可能为“迁移”态或“静止”态。当一个 Agent 在某个 Host 上执行时，其状态值定义为静止态，当一个 Agent 按旅行计划准备向另一个 Host 迁移时，当前 Agent Server 必须通知该 Agent 的 Home 将状态值改为移动态；当一个 Agent 到达某个新 Host 时，新 Host 上的 Agent Server 必须向 Agent 的 Home 通知该 Agent 的到达并将状态值改为静止态。

②在途信件数：保存在 Agent 当前所处的主机上。该值记录了当前时刻以该 Agent 为通信对象的在途信件数。显然，Home 在回复了寻址信件后应及时向目标 Host 提交“增加在途信件”通知，目标 Host 在完成信件入箱后应及时将该值减 1。实际上，“在途信件数”非 0 反映了目前“通信”功能已经在使用“位置”资源，所以，当一个 Agent 准备离开某个节点时，它所对应的“在途信件数”必须为 0，否则，移动要求将被暂缓。

4.2.6.4 通信失效的解决示例

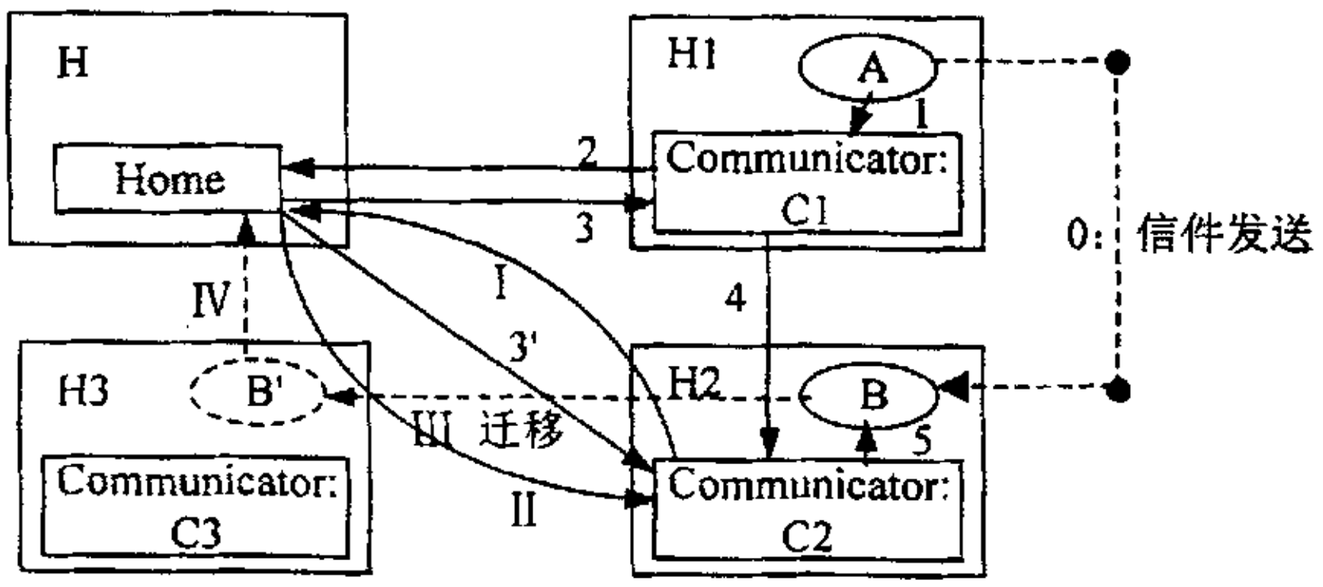


图 4-7 通信失效解决示例

上述改进的基于 MP 的移动 Agent 通信机制能够有效地全面支持移动 Agent 之间的通信，避免移动通信失效现象的出现。下例（图 4-7）是一个在异地通信的同时发生 Agent 请求移动的处理流程。假设 B 创建于 H，目前在 H2，下一站是 H3，位于 H1 的 A 向 B 发出一封信 M。假设 M 在传输过程中，B 要求移动到 H3。其处理如下：

迁移状态初始化为“静止”态，在途信件数为 0。

（1）远程通信

①A 向 B 发出信件。

②H1 上的 Communicator 截获通信要求，检查 B 是否在 H1 上。

③向目标 Agent 的 Home 发送寻址消息。

④H 上的 Home 收到寻址消息后，检查该 Agent 的迁移状态。若状态为“静止”态，向 H1 上 C1 回送目标 Agent 的当前地址 H2，同时通知 B 所在主机的 Communicator 将其在途信件数加 1（如图中 3’）。若状态为“迁移”态，表明目标 Agent B 正在迁移过程中，不回送物理地址，将寻址信件放入通信阻塞队列中，直至 B 到达新的 Host 并向 Home 登记后，才将寻址信件从阻塞队列中释放，并回送更新后的地址。

⑤C1 获得 B 的地址 H2 后，将信件发往 C2。

⑥C2 将消息直接放入 B 信箱，然后将 B 的在途信件数减 1。

（2）迁移请求

①B 通过 Communicator 向其 Home 提出迁移请求。

②Home 收到请求后，修改 B 的迁移状态置为“迁移”，并回送确认信息。

③Agent 收到确认信息后，检查在途信件数是否为 0。若为 0，表明没有信件在发送途中，可以立即迁移；若不为 0，则等待至所有途中的信件到达，即在途信件数减为 0 为止。Agent B 进入实际迁移状态。

（3）地址注册

Agent B 到达新的 H3 后，向 Home 注册其当前物理地址。Home 收到注册信息后，更新其保存的 B 的物理名，将 B 的迁移状态置为“静止”。释放通信阻塞队列中所有等待与 B 通信的其它寻址信件，并一一通知 H3 的 Communicator B 的“在途信件数加 1”。

4.2.7 相关工作比较

移动 Agent 之间的协同是一个具有挑战性的关键技术，其中的通信机制在移动 Agent 研究领域受到了广泛关注，代表性的工作有基于知识交换的 KQML 通信语言以及 IBM 公司研制的 Aglets 系统。它们分别从 Agent 通信语言研究和 Agent 通信的消息传递的系统支持角度展开深入研究。然而，我们尚未见到同时支持 KQML 和消息传递两种协作方式的 Agent 通信机制。而在实际的应用中，信息的交流是多级别、多层次的，因此，我们提出了一种分层多模式的移动 Agent 通信机制，并介绍了关键的间接访问模式的设计工作。

IBMAS 中的通信机制充分考虑了在移动 Agent 环境下基于消息传递的 Agent 移动通信所面临的困难，有效地解决了移动 Agent 系统中特有的 Agent 标识唯一性问题、Agent 通信的透明寻址问题和保证 Agent 通信可靠性等问题，一方面支持多种形式的 Agent 协作通信，另一方面保证了 Agent 通信的可靠性和高效性。总之，该通信机制具有结构灵活、使用方便、效率较高、通信的可靠性较高等优点，为进行各种后续研究奠定了良好基础。

4.3 安全机制的设计[63-75,128]

4.3.1 移动 Agent 系统中的安全分析

4.3.1.1 移动 Agent 系统中的安全特性

相比传统的安全技术，移动 Agent 系统具有自己的特色，使得移动 Agent 安全问题显得更加突出和困难：①难以追踪 Agent 来源；②匿名用户拥有使用权力；③大量跨网域 Agent 的执行；④操作系统的安全保障能力不足。

总之，网络的发展是开放、分布和互联的，而这正和安全的内在要求，即封闭、独立和集中相矛盾，尤其是在移动 Agent 技术中表现更为突出。

4.3.1.2 移动 Agent 系统中安全问题的分类

Robert S.Gray 把移动 Agent 系统中的安全问题分为四类[39]：

①保护主机：主机在支持 Agent 移动和运行的同时，必须确保自身不受恶意的或者有错 Agent 的破坏。这类安全问题主要是 Agent 利用主机上安全设施的不足或缺陷发起针对主机的攻击，主要有伪装、拒绝服务和未授权访问。

②保护其它移动 **Agent**：恶意 Agent 不仅可能攻击主机，还会攻击在同一主机上运行的其它 Agent，甚至能够攻击其它主机上的 Agent。这类问题是指 Agent 可能会利用系统的缺陷对在主机上运行的其它 Agent 进行攻击。

③保护移动 **Agent** 自身：Internet 上，不仅 Agent 不可信任，主机本身也不可信任。恶意的主机可以获得 Agent 所有的代码和数据，进行仔细分析，从而发起有效的攻击。这是所有问题中最难，也是移动 Agent 带来的新的安全问题。正因为如此，人们进行了大量研究。德国斯图加特大学的 Fritz Hohl 仔细区分出主机攻击 Agent 的 12 种不同的途径，主要包括窥视、篡改、非正确运行、伪装、DOS 等。

④保护主机构成的网络：有时，单个 Agent 的行为是正常的，但从一组主机构成的网络的角度来看却是有危害的，Agent 可以在一个站点上只消耗少量资源，其行为也完全符合站点的安全策略，但它却暗中以隐蔽的方式破坏网络的可用性，进而会使一些主机瘫痪。这类攻击的防范也比较困难，从单台主机着手显然不能解决问题，必须从网络整体考虑。

4.3.1.3 研究现状

在移动 Agent 提出后不久，人们就意识到安全问题，开始着手研究，借鉴了现代分布式系统中的许多技术和方法，根据移动 Agent 的特点，进行了改进和扩充，取得了一定进展，提出了许多提高移动 Agent 系统安全性的方法，其中有些方法已经应用到实验系统中去了。

其代表性的工作主要有：基于软件的错误隔离、数字签名技术、携带证明的代码（Proof Carrying Code）、用加密的函数进行计算、混淆代码（Obfuscated Code）等工作。但是，这些方法大多仍处于研究状态，其实用性不强，而且有些问题用目前的技术还难以解决，需要探索新技术[64,67,69,129]。

4.3.2 安全子系统的设计目标

安全子系统的设计目标如下：

①系统性：安全体系的建立是一个系统工程，任何一个极小的安全漏洞都可能导致整个安全体系的崩溃，因此要求安全保护工作具有系统性，从而对移动 Agent 系统的安全特性和安全需求进行全面支持。

②保密性：保密性是安全体系的首要任务，要求保证受保护的信息具有隐私

性，无法被窃听或破译。这主要包括：移动 Agent 之间的通信必须保密，移动 Agent 本身在移动时应保密，站点和 Agent 上的数据应保密。

③完整性：信息保密的同时也需要保证完整性，完整性要求受保护数据具有完整性，不能被破坏或者被修改，一旦发生破坏完整性的行为，应及时通知相关对象。

④可用性：可用性是指保证受保护的对象具有可以自由使用的特性。这主要是针对 DOS 攻击而提出的一种安全要求。

4.3.3 IBMAS 安全体系的建立

针对以上四个目标，结合我们对移动 Agent 系统中安全特性和安全需求的分析，在充分考虑了实现可行性的情况下，在不考虑进行移动 Agent 自身保护的前提下，我们在 IBMAS 中建立了一个较为完善的移动 Agent 安全体系，该安全子系统由六部分组成，另外还有一些辅助设施，总体结构如图 4-8 所示。

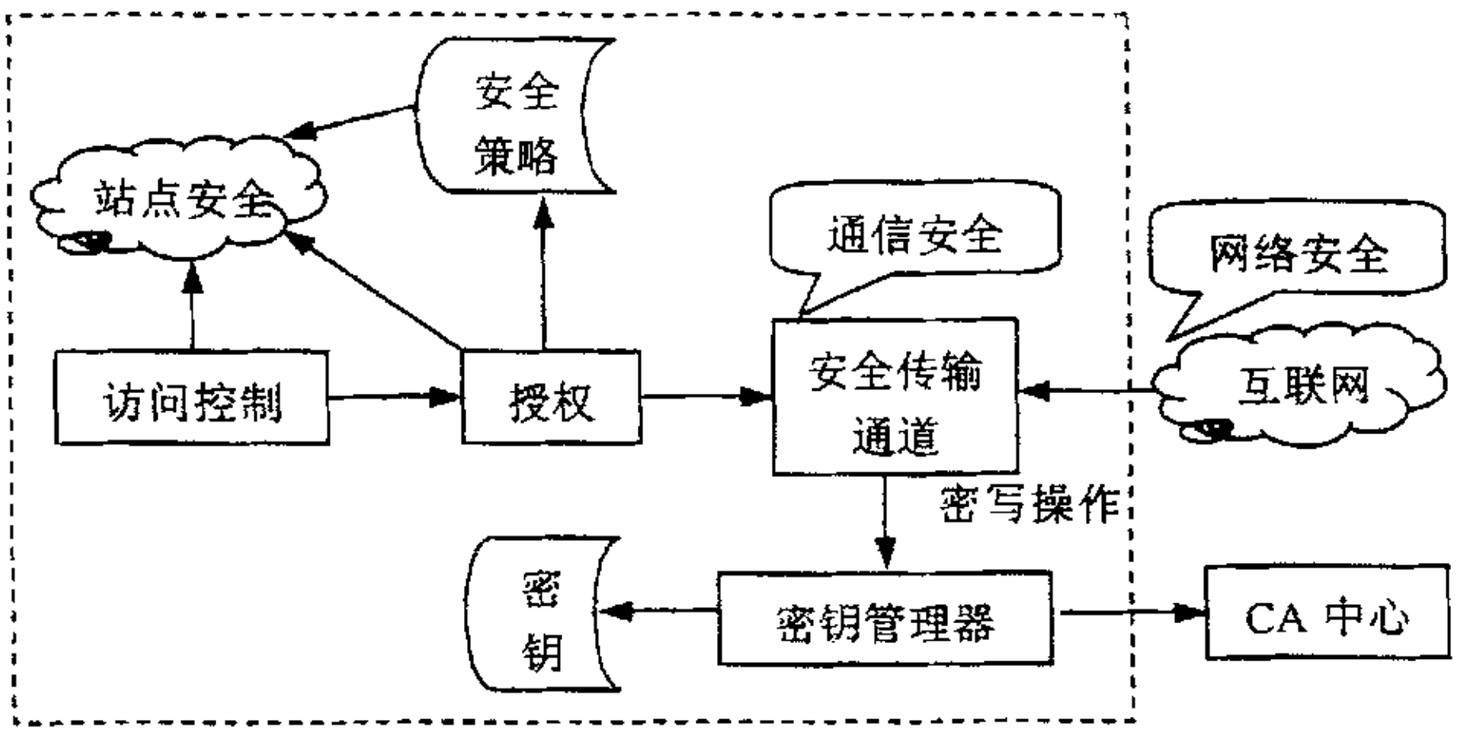


图 4-8 IBMAS 安全子系统结构示意图

上述体系结构中，CA 中心和密钥管理器构成了 IBMAS 中安全体系的基础环境，我们在其上建立了 IBMAS 的安全支持：安全传输通道、授权和访问控制。

①安全传输通道：分析 IBMAS 中的安全需求，发现通信是移动 Agent 系统中的基本行为，也是最容易遭受攻击的行为，大量的攻击行为都发生在通信过程中，因此，保障通信的安全，将是移动 Agent 系统中最基本的需求。因此，我们参照 Netscape 的 SSL 技术，在 IBMAS 中提供了一个安全传输通道，所有通信行为都可以在这个安全的传输通道中进行。安全的传输通道是 IBMAS 安全体系的核心。无论是保护节点、保护其它 Agent 还是保护通信方面都起到了关键作用。

②授权和访问控制：授权和访问控制是操作系统中最普遍采用的对系统资源的保护方式。在 IBMAS 安全体系中，对站点的安全保护也采用了授权和访问控制

的方式，然而，传统的授权和访问控制方式在移动 Agent 系统中存在缺陷，不能有效进行站点的保护，因此，我们提出了一种基于信任传递的授权模型，很好地解决了移动 Agent 系统中站点保护问题。

IBMAS 未提供移动 Agent 本身的安全保护支持，其主要原因是技术难度较大、现有技术的不成熟以及实现开销等方面的考虑。然而，这一部分内容并不影响其它方面的安全保护。下面我们将简述安全传输通道和基于信任传递的安全模型的设计。

4.3.4 安全传输通道

安全传输通道基于 Netscape 公司推出的安全套接字 SSL 技术，综合运用现代密码学技术，在可能不安全的网络传输层服务之上建立一个可靠的安全通道，安全传输通道的结构如图 4-9 所示[130]。

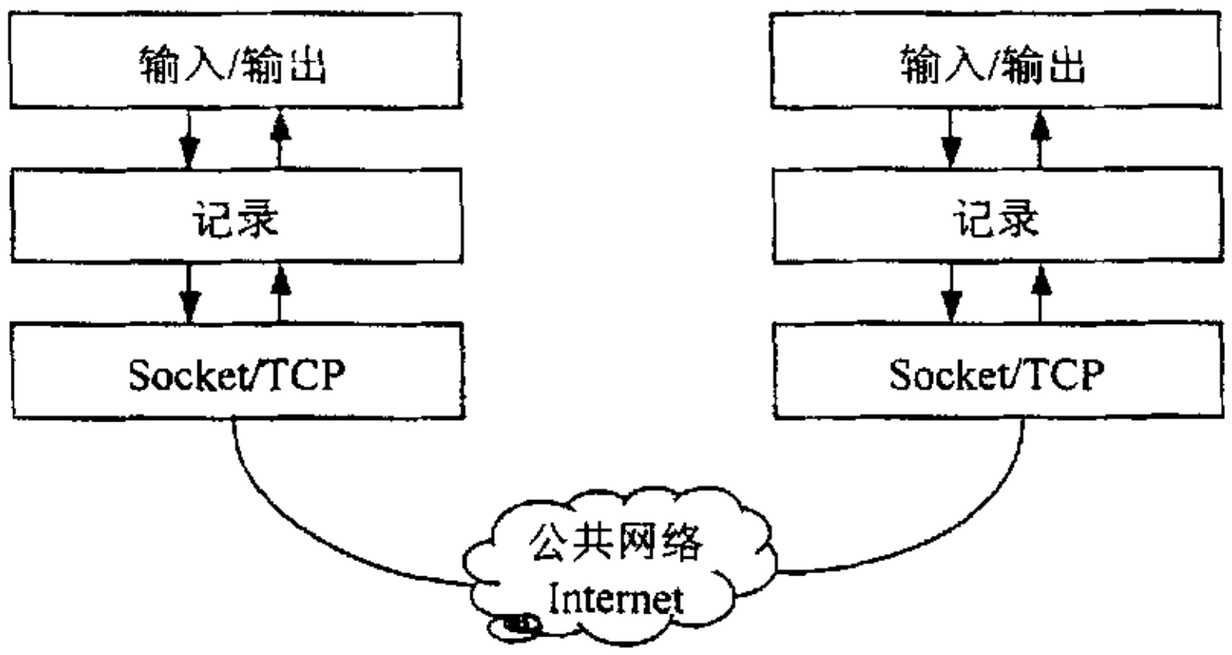


图 4-9 安全传输通道结构示意图

整个安全通道分为三层：输入/输出流层，记录层和套接字层。

4.3.5 基于信任传递的授权模型

授权是给一个用户的 Agent 设置可访问的资源，在移动 Agent 系统中，其依据实际上是 Agent 平台的管理者和 Agent 用户之间的信任关系。如果 Agent 平台管理者对 Agent 用户比较信任，他认为 Agent 不会有恶意的行为，就会分配较大的权力；反之，如果 Agent 用户是一个不可信的甚至是匿名的，那么 Agent 平台管理者只能分配很少的权力给 Agent，以确保自身的安全。显然，在上述过程中，授权是关键，它决定了站点安全的程度。

然而，传统的授权方式忽视了 Agent 的流动性，尚有不足。Agent 的一次旅行可能会经历多台主机，从 Agent 服务器管理者的角度来讲实际上涉及到三类用户，一是 Agent 的发送者，二是管理者自身，三是 Agent 旅行中其它服务器上的用户。

上述模型仅考虑了前两类用户，忽视了第三类。例如在图 4-10 所示的一个网络环境中，A 和 B 是相对友善的节点，B 给予 A 上用户的权限较高。但是网络中存在不友好的 X 和丫节点。当某个 Agent 开始旅行时，它从最初用户的 A 计算机出发，经历了多台中间的 Agent 服务器 X 和 Y，最后到达服务器 B，那么服务器 B 上的管理者在给 Agent 授权时不仅要考虑自己和 A 用户之间的信任关系，实际上还要考虑他和中间所有 Agent 服务器 X、丫上用户的信任关系。由于目前安全技术尚有许多不足，特别是针对恶意的主机还没有保护 Agent 的有效方法，所以即使是一个信任的用户发出的 Agent，在经过途中一个恶意站点时仍有可能被篡改，变成一个恶意的 Agent，所以在授权时必须考虑所有的三类用户。

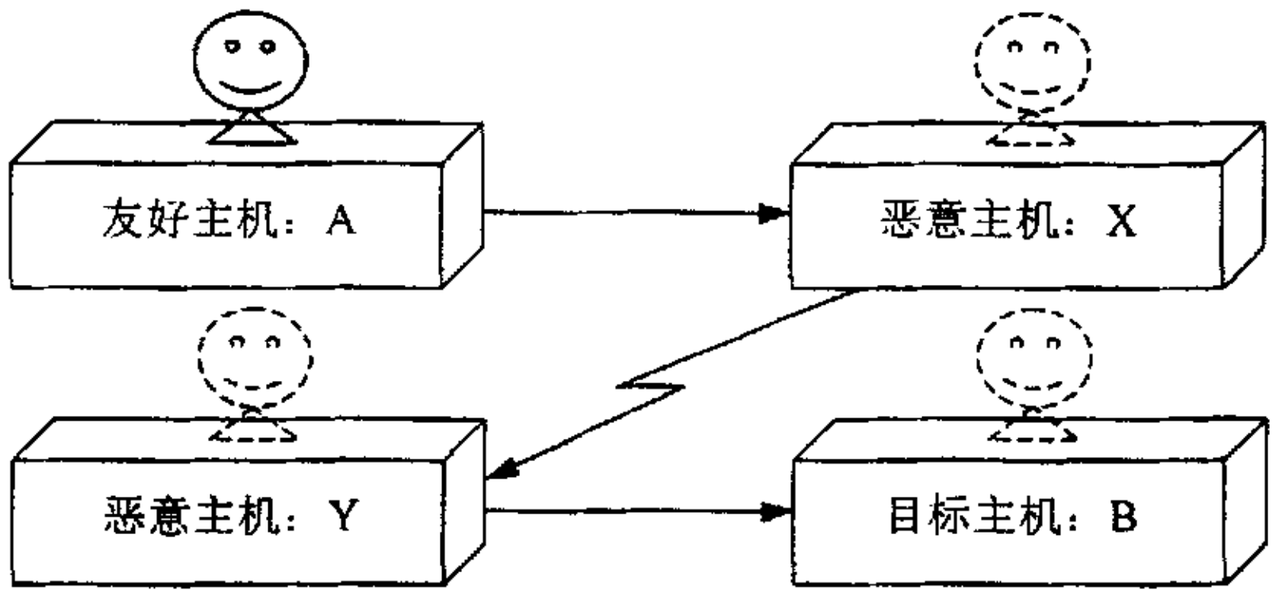


图 4-10 移动环境中的授权失败

因此，服务器管理者在配置安全策略时不仅要设置每个用户的 Agent 可以访问哪些资源，而且要对每一项资源设置可访问的信任主机，即 Agent 访问过的服务器必须全部属于这些列出的信任主机时才能访问该资源，否则不可以。这种新的基于用户信任关系和访问历史的授权模型将给 Agent 服务器管理者更加细致的安全控制，如图 4-11 所示。

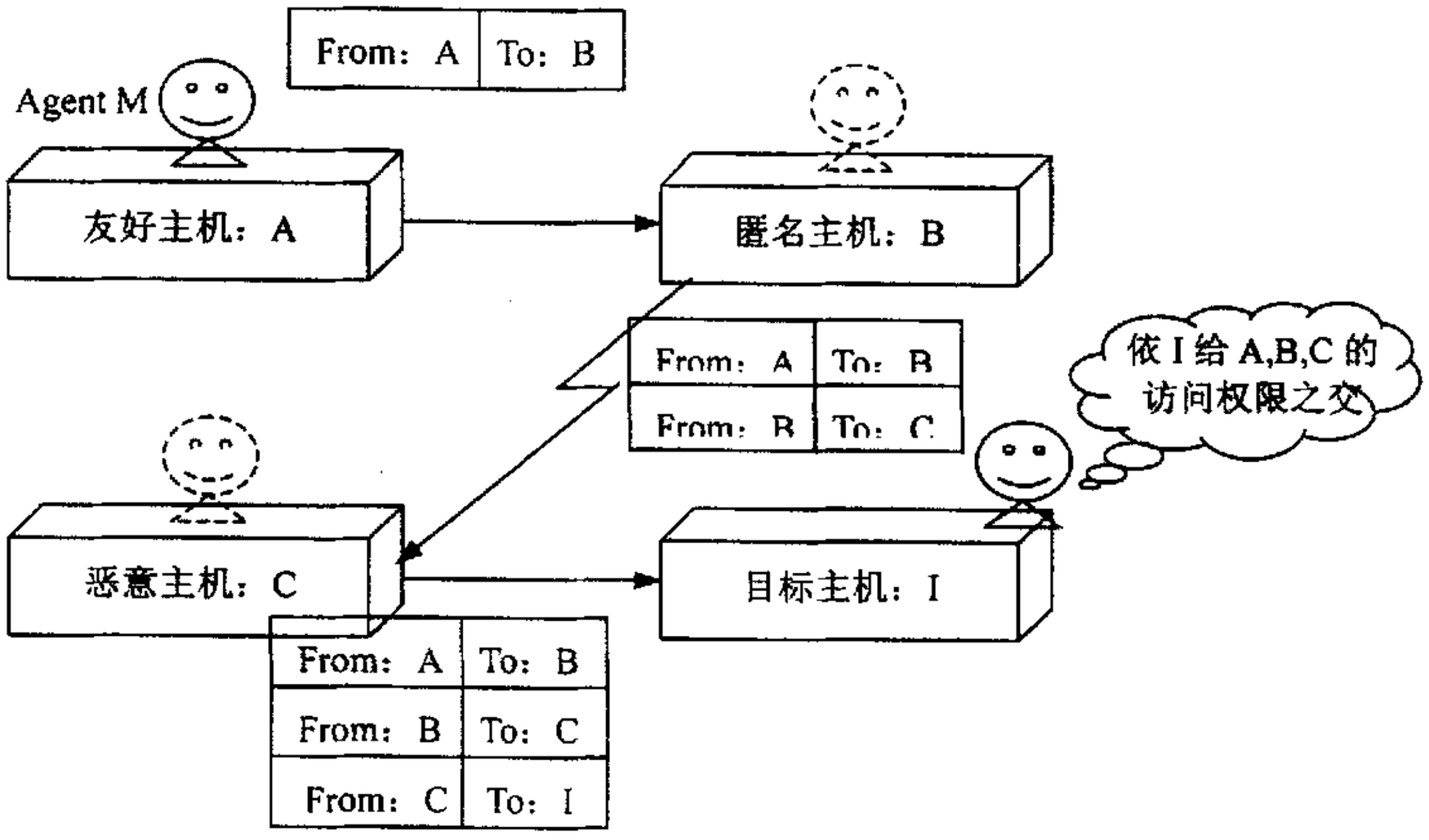


图 4-11 基于信任传递的授权模型

在上图所示的情景中，m 经历了 B 和 C 两个节点，到达了 I 节点，迁移过程中，安全支持环境采用数字签名和联锁机制将每次迁移的路径信息进行记录，当 m 到达 I 节点时，节点 I 可以得到 m 的所有迁移节点信息，其授权采用所有经过的节点的访问权限集合的交集。显然这种基于信任传递的安全授权方法，对 Agent 的限制最强，另一方面，对节点的安全保护也最严格。

4.3.6 相关工作比较和 IBMAS 安全机制的特点

相比其它系统，IBMAS 的安全设施在安全体系的建立及基于信任传递的授权模型方面有一定特色，如表 4-12 所示。

系统	认证	授权	传输通道	代码数字签名
Aglets	Domain Authentication	GUI_Customizable	Symmetric Algorithms	No
Concordia	Identity Encrypted	Preference File	SSL protocol	No
Grasshopper	X.509 Certificates	GUI_Customizable Security Preference	SSL protocol	Yes
Odyssey	No	No	No	No
Voyager	No	Voyager Security Manager Class	No	No
IBMAS	X.509 Certificates	GUI_Customizable Travel History info Based	SSL-Like protocol	Yes

表 4-12 IBMAS 安全设施和其它系统安全设施的比较

4.4 本章小结

在移动 Agent 通信机制方面，本章提出了分层多模式通信模型，并解决了移动 Agent 特有的通信失效问题。一方面支持多种形式的 Agent 协作通信，另一方面保证了 Agent 通信的可靠性和高效率。

在移动 Agent 安全保障方面，本章提出了基于信任传递的授权模型，它和 IBMAS 中的访问控制、安全通道、密写操作等功能，共同构成了一套较为完善的 Agent 系统安全体系。

5 移动智能体在入侵检测系统中的应用

5.1 本章概述[4,84,95-102,107,131-135]

网络和电子商务已经成为企业制胜的必由之路，越来越多的企业将自己的关键业务置于网络之上，并取得了卓越的成绩。然而，高度发达的网络也带来了高风险，网络安全已经成为一个非常重要的问题。防范网络攻击最常用的方法就是防火墙，利用防火墙技术，经过仔细的配置，通常能够在内外网之间提供安全的网络保护，降低网络的安全风险。但防火墙并不是万能的，因为入侵者可以寻找防火墙背后可能敞开的后门；其次防火墙完全不能阻止内部袭击，对于企业内部心怀不满的员工来说防火墙形同虚设；第三，由于性能的限制，防火墙通常不能提供实时的入侵检测能力。入侵检测系统是近年来出现的新型网络安全技术，它可以弥补防火墙的不足，为网络安全提供实时的入侵检测及采取相应的防护手段，如记录证据用于跟踪和恢复、断开网络连接等。

本章在详细讨论传统入侵检测系统的基础上，提出了一种基于移动 Agent 的入侵检测系统（Mobile Agent Based Intrusion Detection System，MABIDS），并对其实现的关键技术进行了讨论和总结。

5.2 问题的提出[4,84,95-102,107,131-135]

随着计算机技术的迅速发展，由计算机处理的事务从单机系统向网络迁移。网络技术的普及、攻击工具和技术的公开，使更多的人知道了网络，但也给一些人创造了破坏的可能性。黑客攻击的案例已经屡见不鲜。2000 年初，美国包括 Yahoo、AOL 在内的多家企业遭到黑客攻击，国内也有很多类似的事件发生。

目前要完全预防和保护网络不受攻击是不可能的。但是，我们可以通过检测已经发生的入侵或入侵企图，来预防进一步的入侵和修复损失。入侵检测定义为“检查并确认那些未经授权而使用计算机系统，以及可以合法访问系统但是滥用了权利的用户”。而入侵检测系统就是用来检测入侵任务的计算机程序。入侵检测技术是近 20 年来出现的主动保护网络免受黑客攻击的一种新型网络安全技术。入侵检测技术不但可以帮助系统对付网络攻击，而且扩展了系统管理员的安全管理能力（包括安全审计、监视、进攻识别和响应），提高了信息安全基础结构的完整性。它从计算机网络系统中的若干关键点收集信息，并分析这些信息，检查网络中是否有违反安全策略的行为和遭到袭击的迹象。

入侵检测系统是防火墙之后的第二道安全闸门，它能在不影响网络性能的情况下对网络进行监视，从而提供对内部攻击、外部攻击和误操作的实时保护。此外，它还可以弥补防火墙的不足，为网络安全提供实时的入侵检测及采取相应的防护手段。它以监测与控制为技术本质，起着主动防御的作用，是网络安全中极其重要的部分。

5.3 传统的入侵检测系统

5.3.1 入侵检测系统分类[4,84,95-102,107,131-135]

入侵检测系统可分为两类：基于主机的入侵检测系统和基于网络的入侵检测系统。前者用于保护关键应用的服务器，实时监视可疑的连接、系统日志检查、非法访问的闯入等，提供对典型应用的监视，如 Web 服务器应用监视。后者则主要用于实时监控网络传输的数据信息。两种入侵检测系统都具有自己的优点和不足，互相补充。前者是对网络中各种系统和设备上的与网络安全相关的活动进行监控，可以精确地判断针对本主机的入侵事件，并能立即对入侵事件做出反应，但它会占用主机宝贵的资源。而后者只能监视经过本网段的活动，仅仅监控在连接媒介上通过的数据包，并且精确度较差，在交换网络环境难于配置，防范入侵欺骗的能力也比较差，但是它可以提供实时的细粒度的网络监视。因此，在实际应用中，二者通常是配合在一起使用的，以提供跨平台的入侵检测解决方案。

5.3.2 传统的入侵检测系统

传统的入侵检测系统是基于 Denning 的入侵检测模型，在这一模型中，将审计记录、网络包以及任何其它可观察到的活动作为检测系统中非正常操作的根据，并且使用跟踪和与已知攻击方式进行比较的方法来进行检测。具体实现时，这种模型可以分为异常入侵检测（Anomaly Detection）与误用入侵检测（Misuse Detection）两类。

①异常入侵检测记录用户在系统上的活动，并且根据这些记录创建活动的统计报告。如果报告表明它与正常用户的使用有明显的不同，那么检测系统就会将这样的活动视为入侵。

②误用入侵检测是事先对已知的入侵方式进行定义，并且将这些方式写进系统中，将网络上检测到的攻击与系统定义的已知入侵方式进行对比，如果两者相同，则认为发生了入侵。

5.3.3 传统入侵检测系统的缺点[4,84,95-102,107,131-135]

基于主机或网络的传统入侵检测系统大都采用单一的体系结构。在这些系统中，数据由一个主机收集，并由单一的模块来分析。有些检测系统使用位于网络节点上的模块实现分布式数据收集，但是所收集的数据被传输到一个中心主机上用单一的引擎进行处理和分析。这些系统具有以下一些问题：

①中心主机是一个单一失效点。如果入侵者能够防止其工作，或者系统不能正常工作时，整个网络就失去了保护。

②检测系统的可伸展性受到很大限制。在单一的主机上处理所有的数据和信息限制了所能检测的网络规模。

③检测系统缺乏灵活性和扩展性。当系统需要加入新的模块和功能时，整个系统就需要修改和重新安装。

为了有效解决上述问题，建立一个健壮、灵活和具有良好伸展性的入侵检测系统，我们提出了基于移动智能体技术的入侵检测系统。

5.4 基于移动智能体的入侵检测系统[4,84,95-102,107,131-135]

5.4.1 移动智能体的应用优势

将移动智能体技术应用到入侵检测系统中可以有以下优势：

① 智能体通过迁移到所需资源的地方后，不需要通过网络即可与中间数据进行交互，显著地降低应用对带宽的要求；

② 移动智能体允许传统的客户机和服务器根据机器的性能和当前的负载，在客户机和服务器之间分配任务；

③ 智能体允许将自己的组件动态地配置到任意的网络中，并根据网络条件的变化重新配置这些组件；

④ 移动智能体能够按顺序通过一系列机器进行迁移，派遣一批子智能体的方式来并行访问多台机器，仅用一些代码就可实现程序的复用性、高效性和健壮性。

移动智能体分布式应用的特点正好弥补了传统的入侵检测系统分布于独立的网段中、彼此协调性差的缺点，使在不同网段中的多个系统能够联合起来，协同工作，及时发现安全问题，更好地保护企业内部重要信息资源的安全。

5.4.2 基于移动智能体的入侵检测系统[4,84]

各种检测方法和技术手段都有其局限性，没有比较通用的检测方法。一个解决方法就是对不同的检测环境进行分类，对不同的环境采用不同的检测方法和技术手段。这就是分布式入侵检测的思想，它采用多个检测部件，各个检测部件选

用不同的检测方法，协同合作，完成检测任务。从而取各种检测方法之长，以便大幅度地提高检测的效率和准确性。

Agent 是能独立运行的实体，它能从系统中加入或移出，各个 Agent 能在运行时动态配置而无须重新启动系统。多个 Agent 可以组成一个组，通过各自简单的功能来共同完成一项复杂的任务。系统中各个 Agent 运行和停止是相对独立的，一个 Agent 如果需要增加或者更新功能，只要它的外部接口保持不变，系统的其余部分将不受影响，甚至不用重新启动整个系统。系统通过 Agent 收集本机的网络信息，可以检测到底层网络攻击。Agent 可以单独编程和运行，从而使得它能从系统审计信息中获得数据，捕获网络数据包，或者从别的地方获得数据。所以，用 Agent 组成的入侵检测系统跨越了传统的基于主机和基于网络入侵检测系统的界线。

基于移动 Agent 的入侵检测系统是基于网络和基于主机的入侵检测的结合，并使用了移动 Agent 技术和智能代理的概念，其系统结构如图 5-1 所示，主要由以下几种元件组成：网络检测点、主机检测点和检测控制中心。在每一个元件上，可分为移动 Agent 平台和入侵检测智能体（Intrusion Detection Agent，IDA）。IDA 可以在完成任务后返回控制中心，如图中实线表示；也可以在完成任务后继续移动到下一个检测点，如图中虚线所示。

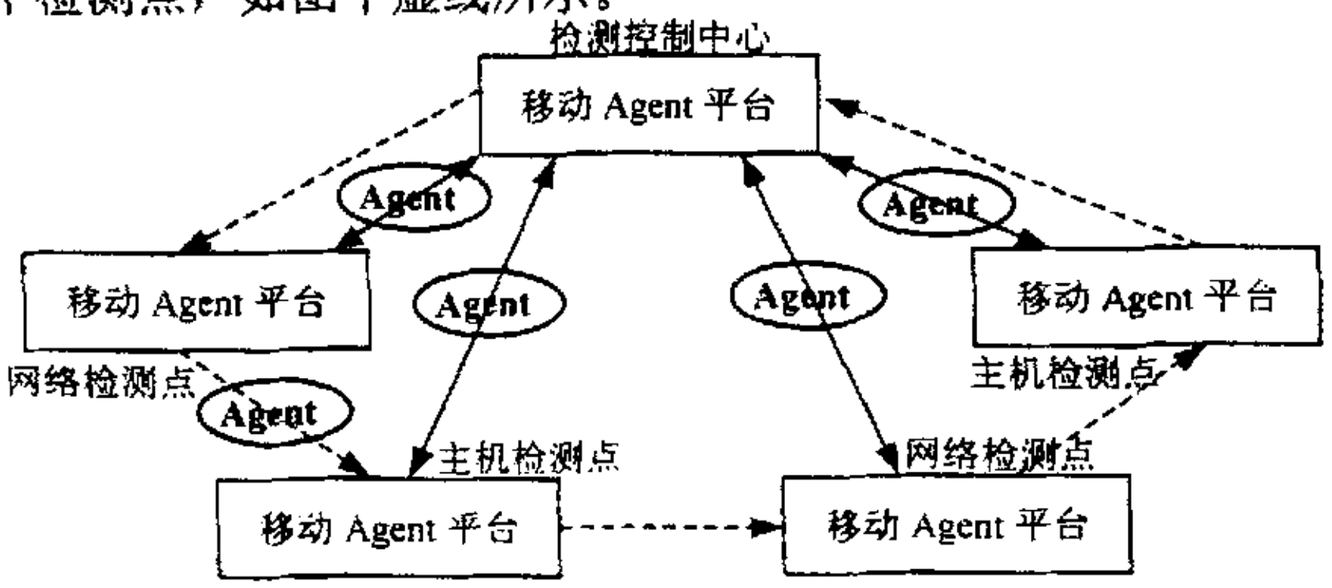


图 5-1 基于移动 Agent 的入侵检测系统示意图

系统的检测任务主要由检测点上的检测程序执行并向检测控制中心报告。系统中，多个检测点同时存在，网络检测点运行网络检测程序，主机检测点运行主机检测程序，这些检测程序都由移动 Agent 负责安装、维护和升级。这些检测程序都可以看作静态 Agent，一旦到达目标节点就不再移动，而是寄宿在检测点上。系统中同时有很多的移动 Agent 在网络中巡游，并将检测结果或过滤后的数据传输给检测控制中心，也可以在检测点进行一些操作。在每一个监视点上必须运行移动 Agent 平台，移动 Agent 平台为 Agent 的迁移和执行提供运行环境。检测控制

中心是系统中的最高级别的实体，在一个系统中必须至少有一个检测控制中心。在检测大型网络时，也可以有多个组织成层次结构的检测控制中心。检测控制中心对移动 Agent 有绝对的控制权，可以创建、派遣、回收 Agent。同时，检测控制中心接收 Agent 发回的报告或经 Agent 过滤后的数据，并进行分析和处理。检测控制中心收集所有 Agent 发回的信息，所以它们可以检测涉及多个主机或整个网络的入侵，以及采用多种手段和技术的高级入侵。另外，检测控制中心还应该为用户界面提供信息和命令接口。

① 主机检测点

系统中的一些重要服务器，需要重点检测，都需要设置为主机检测点。主机检测点运行主机检测程序，检查主机日志、流量、系统资源等审计信息，并进行进一步入侵分析。主机检测程序需要和控制中心保持长时间稳定的链接，运行后向控制中心注册，并建立其 TCP 连接。主机检测程序的运行需要额外消耗系统资源，对于服务繁多的服务器就不太适合。

② 网络检测点

网络检测点安装在网络中附加的一台主机上，它的运行不影响其它机器的正常工作。网络检测程序收集网络底层的数据通讯，对于通讯量比较大的网段，通常有很多数据包，检测程序还要对这些数据进行相应的处理和过滤，最后把结果和有用的数据发给控制中心。对于采取交换机的情况，网络检测点一定要连接在 SPANNING 端口上，才能检测到整个网段的数据包。

③ 检测控制中心

检测控制中心的任务可以分为两大类：

一类是管理入侵检测 Agent。检测控制中心创建所有的入侵检测 Agent、记录这些入侵检测 Agent 的状态并对这些 Agent 进行管理。控制中心对入侵检测 Agent 拥有完全的控制权。检测控制中心可以在需要的时候向被监控的网络和主机分发入侵检测 Agent，例如当检测控制中心接收到入侵检测 Agent 发回的某些特定的警告或者事件时。同时当 Agent 完成检测任务时，或在网络中有过多的 Agent 并对网络设备的正常运行造成影响时，检测控制中心负责将不需要的 Agent 收回或使之消亡。另外，根据需要检测控制中心可以要求移动 Agent 平台挂起、恢复、传输检测 Agent。

检测控制中心的另一类功能则是数据管理和检测入侵。控制中心负责收集、存储 Agent 发回的信息和数据，并进行处理和分析以确定是否有入侵发生。检测控制中心所得到的信息是由多个移动 Agent 收集的。这些信息为被监控网络的状

态和知识提供了一个全局、高级的视图。检测控制中心可以方便地检测到涉及多个主机或整个网络的高级、复杂的入侵活动。

④ 入侵检测 Agent

入侵检测 Agent 是一些独立运行的软件实体，它们代表监视器或用户在网络中巡游并执行入侵检测任务。基于移动 Agent 技术，这些检测 Agent 可以在异质的网络节点间迁移并收集从主机检测点和网络检测点中得到的信息。检测 Agent 在检测到入侵时，可以简单地向控制中心发回一个表述入侵的警告。当 Agent 不能确定某个入侵时，就将相关的数据过滤后传回控制中心作进一步的调查和分析。Agent 之间也可以相互合作以确定某个入侵是否发生。

系统中对检测 Agent 的功能没有任何特别的要求和限制，检测 Agent 可以采用任意的检测技术。检测 Agent 可以是一个简单的用来检测某个已知的系统漏洞是否被系统管理员修复的程序，也可以是使用多种分类算法或相关规则组的复杂系统。检测 Agent 是一个相对独立的实体，也可以采用基于人工智能的技术。当需要完成较复杂的功能时，可以派出多个入侵检测 Agent 共同合作。

⑤ 移动 Agent 平台

移动 Agent 平台是整个系统的基础。在每个检测点上必须有一个移动 Agent 平台。移动 Agent 平台为检测 Agent 的传输和执行提供了必要的机制和运行环境。移动 Agent 平台主要提供以下功能：

- 代表检测控制中心管理位于同一节点上的检测 Agent。移动 Agent 平台可以启动、挂起或停止检测 Agent。
- 在网络节点间传输检测 Agent。相关的任务包括初始化检测 Agent 的传输、接收和分发检测 Agent。
- 当准备迁移时，Agent 首先要确定迁移的终点。当目标节点确定后，检测 Agent 向源节点的移动 Agent 平台请求将自己传输到目标节点。当目标节点收到源节点的传输请求后，在源节点上的移动 Agent 平台将执行下列初始化过程：

挂起检测 Agent；序列化检测 Agent 的 Java 类和状态的实例；选择传输协议对序列化后的检测 Agent 进行编码；认证目标系统；传输检测 Agent。

在检测 Agent 被目标节点接收前，目标节点的守护进程必须确定其是否能解释这个检测 Agent。如果确定则可以接收，进而目标节点的移动 Agent 平台就执行下列接收过程：

认证对方；用相同的协议对检测 Agent 解码；反序列化检测 Agent 的类和状态；实例化检测 Agent；恢复检测 Agent 的状态；恢复检测 Agent 的执行。

● 移动 Agent 平台要为检测 Agent 和检测点提供必要的安全保护。因为检测 Agent 能够在网络中迁移，一个恶意的 Agent 会对主机及检测系统造成危害。移动 Agent 平台必须负责对检测 Agent 进行身份认证。同时，移动 Agent 平台要保证 Agent 只在允许的范围内访问主机资源，如文件系统、CPU、内存等。

基于移动 Agent 的入侵检测系统将大大减少系统的通信负载，并且能够检测到新的或未知形式的攻击。另外，为了解决入侵检测系统互操作问题，目前提出了一些互操作标准，如 CIDF、CHECKPOINT 的 OPSEC、CCI 以及 ISS 的 ANSA 等，对于基于移动 Agent 的入侵检测系统，MASIF（Mobile Agent System Interoperability Facility）规范可以很好地解决互操作问题。

5.4.3 入侵检测智能体的工作流程

IDA 是本系统的基本检测单元，它们分布在主机和网络各处，每个 IDA 独立承担一定的检测任务，检测系统或网络安全的一个方面。在系统中，各 IDA 有独立的数据源、运行模式和响应方式，各 IDA 之间进行相互协作，对系统和网络用户的异常或可疑行为进行检测。不同的 IDA 按照检测环境的不同，采用不同的检测方法和技术。

在本系统中，各 IDA 的行为模式是很相似的，一般都经过以下几个步骤：

(1) 截获系统或网络信息，作日志；

(2) 进行审计分析(模式匹配或异常越界检查，并对相应事件进行响应)；

(3) 进行数据处理，确定可疑度；

(4) 与其它 IDA 通信，对可疑级别达到一定程度的事件进行可疑广播；

(5) 如果需要将数据传送给其它 IDA，就通过 Agent 平台将数据送出；

(6) 记录用户信息。

5.4.3.1 入侵检测智能体检测数据的获取

入侵检测首先需要对系统作日志和审计，而日志和审计涉及到登记和分析系统状态，计算机系统的特点是通过实体集 E 和活动集 A 来表现的。E 是系统的组成部分和各自的状态，A 是引起系统状态变化的事件集合。对所有的 $e \in E$，VAL(e) 是与实体 e 相关的状态集合，$VAL(E)=\{VAL(e) \mid e \in E\}$。字符串集合 N_E，N_V，N_A 用于命名集合 E，VAL(e) 和 A 中的实体，h_E：$E \rightarrow N_E$，h_V：$VAL(E) \rightarrow N_V$ 和 h_A：$A \rightarrow N_A$ 是 3 个命名函数。为了描述方便，我们指定集合 A 中包含空事件。

定义 1：系统状态 s 是一个一元组(E)。所有可能状态的集合 S 组成状态空间。系统相关状态 $\sigma \subseteq s$ 是 s 中要考虑的部分。Σ 是相关状态空间，$\Sigma=\{\sigma \mid \sigma \subseteq s \wedge s$

$\in S)\}$。

定义 2：系统是一个四元组(A，S，s_0，T)，其中 s_0 是初始状态，T：$A\times S\rightarrow S$ 是系统迁移。T 是反映状态改变的映射函数。在系统的生命期内，一系列映射函数 T 将被执行，系统状态将随之而变化。

定义 3：设 N 是自然数集，系统历史函数 Π：$N\rightarrow A\times S$，其中 $\Pi(0)=(a_0, s_0)$，s_0 是系统的初始状态。

$$\forall n\in N\ [\Pi(n)=(a, s)\wedge\Pi(n+1)=(a^*, s^*)]\rightarrow s^*\in T(a, s)$$

相关状态历史是函数 Π：$N\rightarrow A_\Sigma\times\Sigma$，因此

$$\forall n\in N\ [\Pi(n)=(a, s)\rightarrow\Pi(n)=(a, \sigma)]$$

如果 $\Pi(i)=(a, s)$，$\Pi(i+1)=(a^*, s^*)$，我们把使系统状态从 s 变为 s^*的转换函数 T 的成员记为 T_i。也就是说，T_i：$S\rightarrow S$ 和 T：$a\times S\rightarrow S$ 是相同的，其中 a 是 $\Pi(i+1)=(a, s)$的第 1 个元素。

相关状态转换映射函数 τ：$A_\Sigma\times\Sigma\rightarrow\Sigma$。

定义 4：如果条件 $\forall i\{T_i(s_{i-1})=s_i\rightarrow\exists\tau_i[\tau_i(\sigma_{i-1})=\sigma_i]\}$ 成立，则我们说相关状态空间 Σ 是蕴含的。

也就是说，如果 σ 的元素包含足够的系统状态信息，使得 σ_{i-1} 在 T_i 的一些转换关系下映射为 σ_i，那么在转换关系 T_i 下，下一状态的相关部分只通过查看 σ_i 就可以决定，这样，只有 $\sigma_i\subseteq S_i$ 的那些部分是相关的。如果假定系统状态的相关部分是蕴含的，我们只需处理相关部分就可以了。

日志函数分析和提炼系统状态的相关部分，并把它们转化为输出。

定义 5：λ_{state}：$\Sigma\rightarrow N_E\times N_V$，$\lambda_{state}$ 是状态日志函数。

定义 6：λ_{change}：$A_\Sigma\times\Sigma\rightarrow N_A\times N_E\times N_V$，$\lambda_{change}$ 状态转换日志函数。

合起来，$\lambda=\lambda_{change}\cup\lambda_{state}$ 是日志函数。

λ：$A\times\Sigma\rightarrow O$，$O=N_E\times N_V\cup N_A\times N_E\times N_V$，O 是日志函数的输出集合。

直观上来说，状态日志函数记录相关部件的系统状态，状态变化日志函数记录特殊的活动，这些活动会使系统状态的相关部分发生变化。日志函数的输出是记录系统状态和变化的数据。在一个日志系统中，两类日志函数需要同时发挥作用。

定义 7：如果存在唯一的事件 a，实体 e 和实体的状态 VAL(e) 使得 $h_A(a)=n_a$，$h_E(e)=n_e$ 和 $h_V(VAL(e))=n_v$。我们就说 $o=(n_a, n_e, n_v)$(或 $o=(n_e, n_v)$)是可逆的。

直观上说，输出是可逆的则意味着实体的状态可从日志信息中得出。

定义 8：系统日志就是一系列输出 o_0，o_1，…，其中 $\forall i\ \exists j[\lambda(\Pi(j))=o_i]$，

如果每个 o_i 都是可逆的，那么日志是唯一的。如果日志是唯一的，并且产生输出 $o_i(i\geq 0)$ 的相关状态序列就是系统的相关状态历史，那么日志是完备的。

命题　假定 σ_0，σ_1，…是相关状态历史记录，日志 $L=(o_0, o_1, \cdots, o_m)$ 是完备的，当且仅当下列条件成立：

(1) 每个 o_i 是可逆的；

(2) $o_0=\lambda_{state}(\sigma_0)$；

(3) 对所有 $i\geq 1$，$o_i=\lambda(\Pi(i))$。

这表示只有变化日志是不够完备的，相对于系统初始状态，还必须进行状态日志。要正确地追踪系统，状态日志部分必须是蕴含的。转换执行后，引起转换的活动和新的状态必须记录。

5.4.3.2 审计分析

审计包括日志的求精、结果分析和用户或程序的结论。函数 r：$0\to 0$ 求精从 λ 输出的日志信息；函数 a：$0\to\Omega$ 分析求精后的日志，输出审计信息序列 $\omega\in\Omega$；函数 n：$\Omega\to\Sigma\times\Omega$ 给出结果的恰当分类，可能引起系统修改相关状态部分。为方便起见，我们将它们合并成一个函数 α：$0\to(\Sigma\times\Omega)$。设 $0_i=\{o_k|k\leq i\}$。如果 $\alpha(0_i)=(\sigma_i, \omega)$，那么审计函数不改变系统的状态，也就是说只提供信息。如果 $\alpha(0_i)=(\sigma_j, \omega_j)$，其中 $j\neq i$，审计函数反馈一个信息给系统，修改系统的状态，这是反馈性的。

5.4.3.3 入侵检测智能体之间的协作

在我们的系统模型中，一个关键思想是，同时存在许多 IDA，但每个 IDA 都是独立的检测分析单元，没有控制核心，每个 IDA 监控主机或网络安全的一个方面。有时单个 IDA 所收集的信息可能不能判定可疑行为，需要数个 IDA 一起才能涵盖入侵的所有方面。为了正确检测到可疑行为，这些 IDA 必须相互协作来完成检测任务。

（1）数据收集

在分布式环境下，一些 IDA 的检测数据是来自多台主机的审计数据，这时就需要相应主机上有 IDA 来完成数据收集任务。在我们的模型中，数据收集是通过 IDA 间的通信实现的，当一个 IDA 收集到的数据能满足对方的要求或部分要求时，则会将自己所收集的数据中的相应部分发送给对方。

设有 N 个 IDA 为 IDA B 收集数据，如果 A_j 的审计信息输出 ω_{jk} 是和 B 相关的，

B 的日志函数输出为 $o_k=\sum_{j=0}^{N}\omega_{jk}$，$\omega_{jk}$是 A_j的审计信息序列。IDA B 的日志信息序列集合：$O_i=\{o_k|k\leqslant i\}$，如果$\forall k\,\alpha_k(O_i)=(\sigma_{ki},\ \omega_{ki})$，那么审计函数不改变系统的状态，也就是说只提供信息。如果$\exists k\,\alpha_k(O_i)=(\sigma_{kj},\ \omega_{ki})$，其中 $j\neq i$，审计函数反馈一个信息给 A_k，修改 A_k的相关系统状态，这是反馈性的。

（2）可疑广播

在我们的模型下，有一种警告广播包。当一个 IDA 接收到这种广播包时，就表示有 IDA 相信可能有与可疑相关的活动发生，希望提醒相关 IDA 注意。该 IDA 将提升自己的可疑度，当该 IDA 进行后续分析时，也可以进行可疑广播。最终，某个 IDA 的可疑度将超过预先设定的阈值，它就可以向操作员发出警告信息，请管理员检查是否发生了入侵事件。

设有 M 个 IDA 相互协作，如果 A_i的可疑度广播值是 v_{ik}，A_j的可疑度为 $\beta_{jk+1}=\beta_{jk}+\sum_{i=0}^{M}\lambda_i V_{ik}\bullet\beta_{j0}=0$，$\lambda_i$是加权因子 $0\leqslant\lambda_i\leqslant 1$。

另外，我们引入了一个定时器函数：θ：T→N，可以让 IDA 的可疑度随时间而下降，直到降到一个规定的下限为止。如果 IDA 接收到一个可疑广播，将提升自己的可疑度。如果没有其他广播来提升，它就会逐渐恢复到正常操作状态，并继续监控。

5.4.4 基于移动智能体的入侵检测系统的特点[4,84,95-102,107,131-135]

基于移动智能体的入侵检测系统具有以下特点：

① 独立性　入侵检测智能体是单独运行的程序实体，可以独立开发，在放入具体运行环境前，可以进行独立测试；

② 灵活性　入侵检测智能体可以单独地启动和停止，也可以进行动态配置，而不影响其它入侵检测智能体的正常运行。要收集新的数据或检测新类型的入侵，可以通过对原入侵检测智能体进行重新配置或设计新的入侵检测智能体来实现；

③ 可扩展性好　无论是新增检测主机，还是在主机中增加入侵检测智能体都简单、方便；

④ 错误扩散小　如果某个入侵检测智能体出现问题或受到破坏，那么，仅仅与该入侵检测智能体相关的检测部分失效，其父节点会很快检测到它的状态，进行相应的处理，使危害限制在最小的范围内；

⑤ 数据源不受限制　不同的入侵检测智能体可选用不同的数据源。由于入侵检测智能体是单独实现的，所以数据源可以采用多种形式：如审计数据、检查系

统配置、捕获网络包或其它合适的来源；

⑥ 兼容性　该模型可以既包含基于主机的入侵检测智能体，又包含基于网络的入侵检测智能体，超越了传统入侵检测模型的界限；

⑦ 跨平台　由于移动智能体本身的移动性，开发出来的入侵检测应用自然可以在不同平台上运行；

⑧ 协作性　每个入侵检测智能体检测主机或网络安全的一个方面，它们之间进行信息共享，便于综合分析。

5.4.5 基于移动智能体的入侵检测系统的优点[4,84,95-102,107,131-135]

MABIDS 是一个具有高度可扩展性和灵活性的系统。入侵检测智能体是MABIDS 中的可扩展元件。向 MABIDS 中加入新的模块和功能就变得很容易实现，只需要加入新的检测智能体即可。同时检测智能体的设计和实现与入侵检测系统分离开来，新的入侵检测智能体也很容易实现。而对现有的检测智能体的升级也可以通过简单的更换来完成。

检测系统所造成的网络流量将大大减少。在 MABIDS 中，很大一部分检测工作是由监控器上的检测智能体来完成的。由于智能体具有可移动性，智能体可以被传输到数据所在的网络节点上执行检测任务，而不是将所有的数据都通过网络传输到一个中心节点来处理。而对智能体不能确定或完成的检测，智能体可以根据一定的知识规则对数据进行过滤，只将过滤后的数据传输给检测控制中心，这样通过网络的数据流量将显著减少。同时，由于检测智能体所需要处理的数据量的减少，在监视大型网络时，检测系统的可扩展性也得到很大改善。在 MABIDS 中，随着被监视网络节点的增多，并没有较大地增加检测控制中心主机的负荷，所以 MABIDS 可以实现对大型网络的监控。

与传统的入侵检测系统相比，MABIDS 具有更好的稳定性和健壮性。在 MABIDS 中，由于检测入侵的任务是由多个检测智能体分散实现的，在 MABIDS 中不再有单一失效点。监控器不能正常工作时，检测智能体仍然可以独立地执行入侵检测的任务。

5.5 MABIDS 实现的关键技术

①安全的移动 **Agent** 技术　由于无法保证网络和主机系统的安全性，并且允许 Agent 运行的 Agent 服务器有可能将恶意代码植入 Agent 中，因此必须保证派遣 Agent 的系统、允许 Agent 运行的系统和 Agent 本身的安全。

②派遣 Agent 的策略　使用移动 Agent 的目的是既要减少网络流量，又要提高系统的性能。所以，如何派遣 Agent 和分配 Agent 的任务需要一些好算法。

③Agent 的控制问题　Agent 具有高度的自主性，因此必须要有好的控制策略，对 Agent 进行有效的管理，使它不会失效，不会对其它系统造成危害。

④Agent 在网络中的路由　好的路由策略可以使 Agent 更高效地完成任务，不会造成额外资源的耗费。

⑤实时数据截取技术　监控器的核心是数据截取部分。实际实现时通过直接访问数据链路层，使应用层程序实时捕获底层数据包。

⑥多个入侵检测系统间的互操作与协作　为了提高入侵检测的效率，多个入侵检测系统之间必须相互协作。

⑦实时、安全数据库　知识库和信息库都需要一个安全数据库的物理支持，可以采用安全性较高的多平台数据库。

⑧系统的支撑平台　它必须具有很高的安全性和高效性。

5.6 应用实例

MABIDS 采用集中管理的分布式网络安全监测模式，安全控制中心完成整个分布式安全监测预警系统的管理与配置。监控器监测其所在网段上的数据流，实时进行攻击自动识别和响应。

监控器安装在开放的外部网络与被保护的内部网络之间或内部网络中敏感主机的网段上，对流经的数据进行实时检测。当一个企业内部网络有多个出口或多个敏感部位的时候，可以在每个需要监测的位置安放一个监控器。每个监控器就是一个哨位，而哨位可以按照用户系统的安全要求分布在每个需要它的地方。当监控器发现安全违规事件的时候，将有关的报警信息和日志信息发送到安全控制中心。

一个企业的安全管理应该是由安全管理部门统一负责的。虽然监控器的设置是分布式的，但是对于它的管理和控制却应该是集中的。安全控制中心是整个企业网络的安全控制中心，它通过移动 Agent 来启动监控器、向监控器加载安全规则、处理各个监控器发来的实时报警和日志信息、定义和生成各个监控器应执行的安全规则、提供对报警和日志信息的查询接口和生成系统安全审计报告等任务。

例如一个企业的网络结构如图 5-2 所示，那么我们就需要安装 5 个监控器和 1 个控制中心。

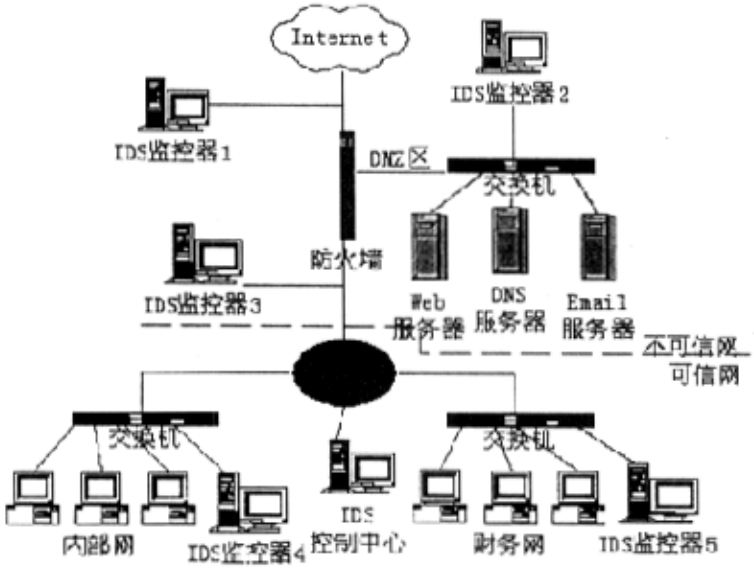


图 5-2 基于移动智能体的入侵检测系统的应用实例

监控器 1：安装在防火墙的非信任边的监控器可以侦测入侵企图。它同时会产生安全管理员不感兴趣的警报。记录可能的威胁将有助于决定是否使用 IDS，但是对正确配置的防火墙，这将不是严重的问题。只有到达子网的攻击才可能导致破坏。

监控器 2：很多站点选择将需要外部访问的设备安装在独立的子网上，通常成为非管制区（Demilitarized Zone，DMZ）。在那里放置一个监控器很重要，因为那里提供的很多服务是常被攻击的对象。

监控器 3：这是一个企业放置 IDS 监控器最关键的地方。扫描那些通过防护的网络流量信息，找出可疑行为，产生代表潜在严重安全威胁的警告。

监控器 4 和 5：用于观测内部活动。大型企业内部网有很多可以获取的访问点，如：Modem 接入、危险的帐号、不忠的雇员及有组织的进攻等，都是入侵者可以越过防火墙的例子。

IDS 控制中心：它通过派遣移动 Agent 来启动监控器、向监控器加载安全规则，并处理各个监控器发来的实时报警和日志信息、定义和生成各个监控器应执行的安全规则、提供对报警和日志信息的查询接口和生成系统安全审计报告等任务。系统界面如图 5-3 所示。

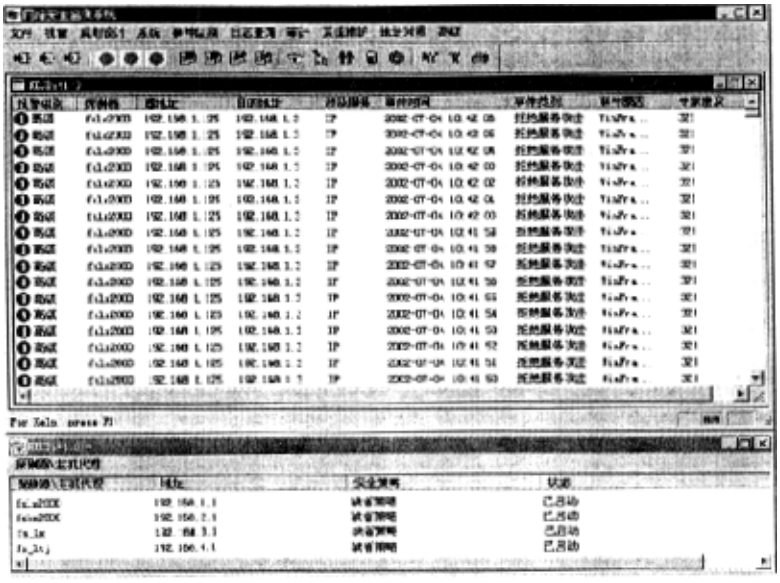


图 5-3 IDS 控制中心

5.7 本章小结

网络技术在不断地发展，安全防范技术也在不停地经受考验，本章阐述的基于移动 Agent 的入侵检测系统 MABIDS 是一种安全性强、移植性好、效率高的入侵检测系统。检测控制中心可以派遣检测 Agent 自动迁移到监控器上，收集只与入侵有关的信息，不再需要将系统日志传输给检测控制中心，而且亦可与其它基于移动 Agent 的入侵检测系统有效协作，大大提高了入侵检测能力。

但是，IDS 和防火墙是无法互相替代的，作为一个完整的网络安全体系，二者缺一不可。防火墙属于第一级的防护措施，而 IDS 属于第二级，前者是主动的防护行为，针对外部的入侵行为可以采取截断的行为，但是通常只依据简单的规则（包过滤型）并且可能会影响网络的传输。而后者 IDS 是被动的监控行为，首先不会影响网络的传输或者说系统负荷几乎为零，其次不仅对于外部的攻击而且对于来自内部的攻击都可以检测到。因此，为了保护网络，不能一概而论地采取某种技术和方法，而应该从各个层面来提高整体的安全性。

6 移动智能体在网络漏洞扫描系统中的应用

6.1 本章概述[136-139]

1994 年以来，由于互联网在全球迅猛发展。在信息化极大地促进了经济发展的同时，网络与信息安全也随之成为众所瞩目的问题。在我国，利用计算机网络进行的各类违法行为每年以 30%递增。由于利用系统缺陷或后门进行攻击，以及内部用户的窃密、泄密和破坏行为，在非法入侵案件中占到 50%以上的比例，所以各部门计算机系统管理员应加强对系统的安全检测和防护能力，及时检测安全漏洞和配置错误，对发现的系统弱点和漏洞及时采取措施进行修补改造、升级，做到防微杜渐，防范未来。

支持这一防范策略的软件，称漏洞扫描系统即网络安全性分析系统。它应能对 Internet/Intranet 中所有部件如 Web 站点、防火墙、路由器、TCP/IP 及相关协议、服务等进行攻击性扫描、分析和评估。发现并评估系统存在的弱点和漏洞，据以估计安全风险，建议补救措施。然而，当网络规模较大，结构复杂时，其远程执行有挤占带宽、效率不高且不能有效执行的缺点。为了有效解决此问题，我们试图将网络安全性分析系统的“功能体系”包装成移动智能体的智能行为机制，将网络上纳入扫描范围的各个设备状态作为外部世界描述，当执行扫描的移动智能体向被扫描对象发出攻击探测信号时，被扫描的对象产生的反应信号就被分析处理从而得到网络是否存在安全漏洞、何种漏洞、如何补救等信息。因此，网络安全管理人员可根据网络责任范围及有关入侵知识制定适当的扫描策略和相应修补措施，使系统在遭到实际进攻时有备无患或使黑客无隙可入，达到主动防御的效果。这种模拟遭受入侵达到增强网络系统抗入侵能力的移动智能体系统称为免疫扫描智能体系统。本章将阐述其技术基础、系统实现和应用实例。

6.2 问题的提出[136-139]

入侵检测系统面临着如下问题：

①传统的入侵检测系统是一种“被动”的系统。不论是基于主机的入侵检测系统还是基于网络的入侵检测系统，都是在发现入侵的行为后采取相应措施。如果能在入侵者行动之前采取适当措施，使得系统对于一些已知的入侵行为（如利用系统漏洞的入侵方式）具有“免疫”能力，从而使整个系统能够在事前具有抵抗入侵的能力：

②网络系统的拓扑结构和功能在不断变化，传统的漏洞扫描系统不能及时地适应这种变化。同时，由于没有经过适当定制的扫描系统可能使网络系统或主机的运行受到影响，不能够根据已有的扫描结果自主确定再次扫描的优先次序，只能运行在指定的软件平台上，如果通过广域网远程扫描系统，还可能耗费网络带宽资源。

对于以上二个问题，我们可以利用具有模拟攻击能力的智能体来发现系统的漏洞，并针对发现的问题由智能体采取相应的策略予以解决。同时控制这种攻击带来的影响，使得尽可能地在攻击真正到来之前对某种攻击行为具有免疫能力。这种智能体能够在网络主机空闲时运行，有选择性地扫描以避免损害主机的运行，一旦发现漏洞，则加以记忆，在扫描下一个网络主机或节点时或再次扫描该网络时利用，这样使得智能体具有自我更新的能力。由于模拟攻击行为，特别是拒绝服务、穷举密码等攻击行为消耗网络带宽资源较大，如果通过广域网远程扫描系统，可能浪费网络带宽资源。如果智能体具有移动能力和跨平台运行能力，则能较好地解决这些问题。因此免疫扫描智能体是一种较好的技术选择，它是一种具有网络拓扑结构感应能力和行为自主性的软件智能体。

6.3 基于移动智能体的网络漏洞扫描系统[136-139]

基于上述移动智能体系统的优越性能，使移动智能体携带网络漏洞扫描任务。将选定的扫描执行组件嵌入创建中的移动扫描智能体作为其任务求解模块的内核。在网络节点上移动，并在相应节点上完成指派的扫描任务，并将扫描结果向其控制中心报告。首先，在智能体服务设施或技术平台包装器 Wrapper（一种 Enabler）支持下，创建漏洞扫描智能体。具体作法是将高性能漏洞扫描机制装备给移动网络漏洞扫描智能体 MA，该智能体接受智能体控制中心给定的跨网段扫描任务，按派遣规定在本地或远程执行特定进攻性扫描过程，并将分析结果报告控制中心，为网络安全性分析提供依据。为便于说明，图 6-1 给出由广域网顺序相连的两个局域网上各主机和网络设备被扫描的情况，图中的序号表明过程。

6.3.1 免疫扫描系统的实施策略

随着网络配置不断更改，黑客 Hacker 进攻手段不断更新，网络中的最薄弱环节也就随之变化，因此，最有效的安全防御策略应立足于及时发现和修正网络存在的弱点和漏洞，不仅要把维系网络安全的检查、分析和评估当成经常性的工作，还要根据 Hacker 采用的最新手段更新系统知识库（攻击模式库），即网络系统安

全性体现为一个动态过程。通过各类别免疫扫描过程不断增强网络自身的抗入侵能力。

6.3.2 移动智能体技术方案

网络漏洞扫描系统的工作原理实际上是模拟黑客进攻方式，根据网络上各个受攻击的主机、路由器、交换机、防火墙等的反应作出判断的。

设已知漏洞 v_j，那么漏洞系列是一个向量组 $V=\{v_1, v_2, \ldots v_n\}$；一个确定的网络系统，存在可能受攻击有限主机 $H=\{h_1, h_2, \ldots h_m\}$；每一个漏洞对于确定的主机而言的风险程度不同，与时间和主机的软硬件环境有关。那么存在安全状态矩阵 Ψ:

$$\Psi=\begin{bmatrix} f_{11}(v_1,e_1,t) & f_{12}(v_1,e_2,t) & \cdots & f_{1n}(v_1,e_n,t) \\ f_{21}(v_2,e_1,t) & f_{22}(v_2,e_2,t) & & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ f_{m1}(v_m,e_1,t) & f_{m2}(v_m,e_2,t) & \cdots & f_{mn}(v_m,e_n,t) \end{bmatrix}$$

其中 $f_{ij}(v_i,e_j,t)$ 为主机 h_j 由于漏洞 v_i 产生的风险，风险的大小与时间 t 和软硬件环境 e_j 相关，其中 $i\in\{1, 2\ldots n\}, j\in\{1, 2\ldots m\}$。$\Psi$ 反应了系统在已知漏洞攻击方式下的安全程度。称系统在已知漏洞攻击的安全程度 Ψ 不劣于 Ψ，如果 $f_{ij}'(v_i,e_j,t)\leqslant f_{ij}(v_i,e_j,t)$；称系统在已知漏洞攻击的安全程度 Ψ' 优于 Ψ，如果 $f_{ij}'(v_i,e_j,t)\leqslant f_{ij}(v_i,e_j,t)$，且至少存在一个 i,j，使得 $f_{ij}'(v_i,e_j,t)<f_{ij}(v_i,e_j,t)$。免疫扫描智能体首先通过可以控制危害程度的模拟攻击，获取安全状态矩阵 Ψ，然后针对具体的漏洞采取措施使得新的基于已知漏洞攻击的安全程度 Ψ' 优于原有的 Ψ。

如果把一个网络考虑为若干子网，子网内部带宽资源丰富，而子网之间带宽资源紧张，那么没有必要在对每一子网扫描全部漏洞，而是有选择性的扫描。对于每一个漏洞的扫描可能性为 p_i，移动智能体在某子网的漏洞扫描为三元组 $SM^{hj}=(V, P^{h_j}, \omega)$，其中 $V=\{v_1, v_2, \ldots v_n\}$；$P^{h_j}=\{P_1^{h_j}, P_2^{h_j}, \ldots P_n^{h_j}\}$ 为在主机 h_j 处漏洞扫描的可能性向量；ω 为扫描阈值，若 $P_i^{h_j}>\omega$ 则进行扫描，否则不进行扫描。

若 $P^{h_j}=\{P_1^{h_j}, P_2^{h_j}, \ldots P_n^{h_j}\}$，那么 $P^{h_j+1}=\{P_1^{h_j+1}, P_2^{h_j+1}, \ldots P_n^{h_j+1}\}$，$P_0$，$P_1$，$P_2>0$，

$P_i^{h_j+1}=P_i^{h_j}+P_0$ 若在 h_j 处发现漏洞 v_i；

$P_i^{h_j+1}=P_i^{h_j}-P_1$ 若在 h_j 处没有发现漏洞 v_i；

$P_i^{h_j+1}=P_i^{h_j}+P_2$ 若在 h_j 处没有扫描漏洞 v_i。

但是，经常使用网络漏洞扫描系统，特别是广域网上远程操作会占用相当昂贵的广域网带宽资源，影响网络正常运作，导致经常性检查的成本过高，更有甚者导致系统自身的安全问题。为此，我们提出的技术方案是，将网络漏洞扫描系统的机制和功能作为面向任务智能体的任务模型执行程序，装入移动智能体外壳。在移动智能体内部结构中，扫描执行程序就是作为组件包装入任务求解模块的。当该智能体处于执行态时，网络扫描过程即被执行。远程作业时，网上移动的是智能体而并非原网络漏洞扫描系统产品的应用程序和/或响应数据。大大缓解了带宽占用矛盾。进而利用智能体的记忆能力从黑客和网络配置两方面在不同条件下的攻防力量对比建立较有完备的资料作到防患于未来，而且由于资料齐备，还可以设置陷阱，主动完全变被动防守策略，诱敌。

6.4 应用实例

图 6-1 表明由移动智能体 MA 和控制台 A 组成的网络扫描智能体系统，由广域网左端局域网跨过广域网到达右端局域网执行扫描的过程，现将扫描执行过程简述如下：

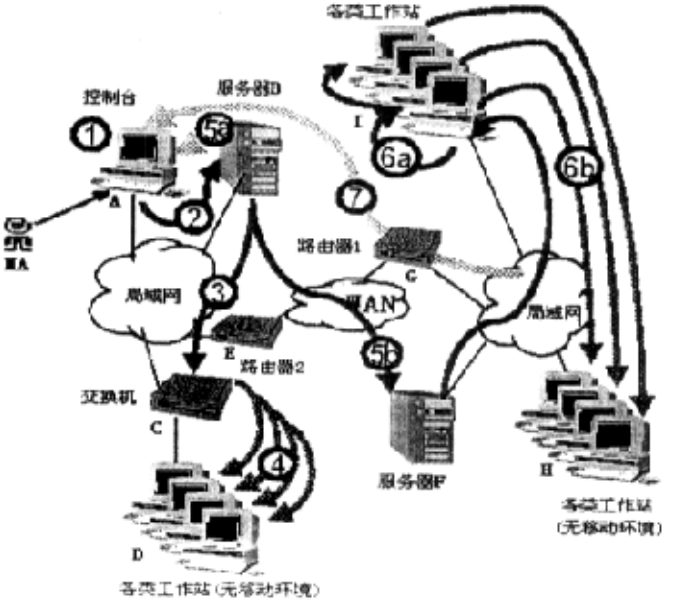


图 6-1 网络扫描智能体系统

① 启动系统，检查所创建的移动智能体 MA，由用户确定扫描任务涉及的网络设备和主机（IP 地址），MA 处于准备态。网络管理员操作控制台 A 时，界面如图 6-2

所示。

② 按 ATP，MA 由准备态转入传输态，被遣送到装备了智能体平台的服务器 B，在服务器 B 支持下，执行左端局域网上的第一波扫描，此时 MA 处于执行态。在服务器 B 显示器的界面上，看到 MA 已到达，正在执行扫描任务，尚无扫描结果，见图 6-3。

③ 扫描左端局域网上备有扫描智能体服务设施的设备，如交换机 C、路由器 E。

④ 在 C 的环境支持下对 D 组无 MA 执行环境的工作站进行扫描。

⑤ (5a)对 D 组工作站的扫描结束后，服务器 B 将扫描结果送到控制中心 A。在服务器B显示器界面上可见扫描结果，此结果同时在控制台界面上显示，见图6-4。(5b)由服务器 B 支持下实现跨广域网传输，到达右端局域网的服务器 F。

⑥ 在服务器 F 支持下，执行右端局域网上各设备和主机的扫描，路径(6a)完成对有智能体服务设施的各主机 I 组的扫描，路径(6b)则表现在 I 组各主机支持下对 H 组无智能体服务设施的各主机的扫描(图略)。

⑦ 移动扫描智能体 MA 返回控制台，提交全部的扫描结果。完成全部任务进入结束态，等待新的任务指派，见图 6-5。

⑧ 控制台按收集到的各 IP 地址上的安全状况，评估弱点、漏洞，通过人一机交互，制定修正和调制措施，为下一轮系统安全评估作准备。

图 6-2 控制台

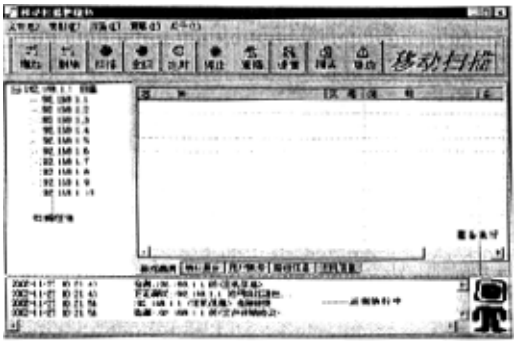


图 6-3 移动智能体到达服务器 B

图 6-4 服务器 B 上 D 组的扫描结果

图 6-5 控制台上所有的扫描结果

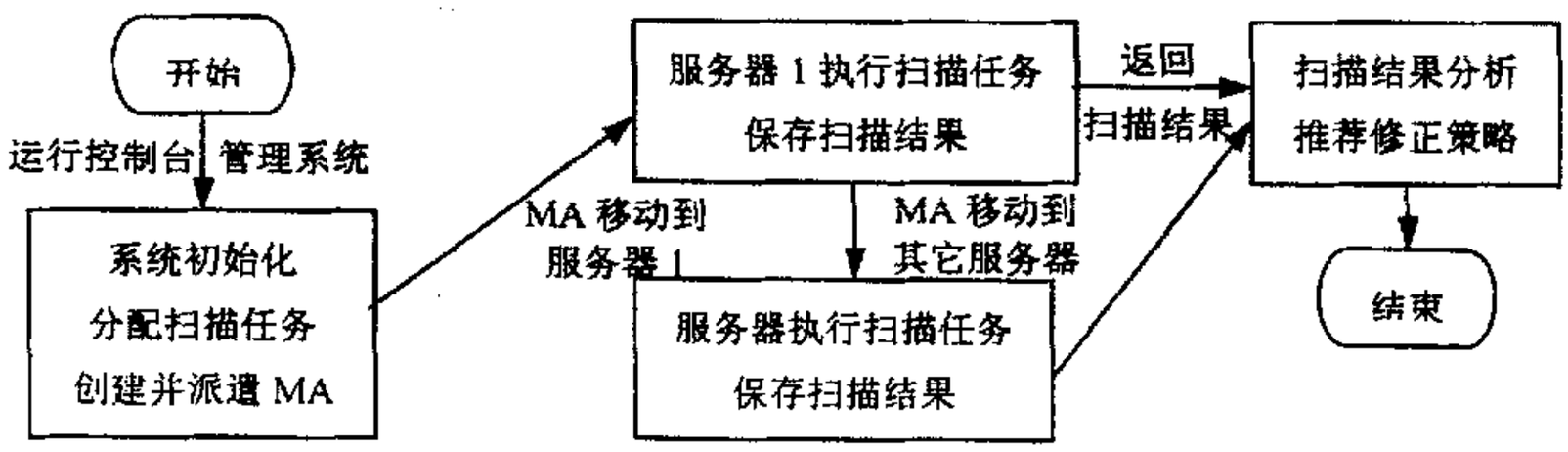


图 6-6 系统的工作流程

免疫漏洞扫描系统实现采用 Delphi 语言实现，MA 采用面向对象方法实现，即面向任务的移动智能体，而控制台在整个生命周期活动中提供运行环境和使能工具的支持，图 6-2 至图 6-5 表明控制台人一机交互界面所见到的 MA 任务执行过程（其中，移动智能体到达服务器 F，并在服务器 F 上对 H 组、I 组主机扫描的界面与图 6-3、图 6-4 类似）。系统的工作流程见图 6-6。

6.5 程序解释

6.5.1 移动扫描智能体

移动扫描智能体负责建立自己的运行状态数据（agentState.ini）和运行环境数据（vulns.txt）。agentState.ini 是一个典型的 WINDOWS 格式的配置文件，程序通过类 TIniFile 来读取和配置。在程序中的使用如下：

```
TotalNodes:=IniFile.ReadInteger('移动站点序列','总站点数目',1);
CurrentNode:=IniFile.ReadInteger('运行状态','当前所在站点',1);
CurrentNodeIP:=IniFile.ReadString('移动站点序列','移动站点
'+Inttostr(CurrentNode),
'127.0.0.1');
IniFile.WriteInteger('运行状态','当前所在站点',CurrentNode+1);
```

vulns.txt 是一个以 TAB 为分隔符的文本文件，代表了智能体的漏洞知识库。每一行即表明了一个漏洞的名称，风险级别和详细描述。对于这个知识库的解释函数为 Split，见 cStrings.pas，其函数原型为。

```
Function Split(const S:String;const Delimiter:String):StringArray;
```

移动扫描智能体的移动是依靠向下一站的 UDP 端口 5999 来实现的，移动过程中每一机器必须安装 ATP server。向 UDP 端口 5999 发送数据过程如下：

```
WSocket1.Proto:='udp';
WSocket1.Addr:=IniFile.ReadString('移动站点序列','移动站点
'+Inttostr(CurrentNode+1),
```

```
'127.0.0.1');
WSocket1.Port:='5999';
WSocket1.LocalPort:='0';
WSocket1.Connect;
WSocket1.SendStr(ftpserver);
WSocket1.Close;
```

Uagent.pas 中是移动扫描 agent 的程序实现。

6.5.2 智能体传输协议

标准的 ATP 协议是建立在 ATP Server 之间的请求/响应基础上，所实现的智能体借助于本地 ATP Server 与远端 ATP Server 来完成智能体的移动。智能体传输协议简单实现了智能体可执行程序的传输。智能体传输协议实现为一个监听 UDP 端口 5999 的程序。如下所示：

```
WSocket.Proto:='udp';
WSocket.Addr:='0.0.0.0';
WSocket.Port:='5999';
WSocket.Listen;
```

下面实现了扫描智能体可执行程序的传输。

```
NMFTP1.Host:=ftpserver;
NMFTP1.Port:=21;
NMFTP1.Timeout:= 5000;
NMFTP1.UserID:='UserID';
NMFTP1.Password:='Password';
try
    NMFTP1.Connect;
    NMFTP1.ChangeDir('C:\FTP\');
Except
    On E:Exception do
       writeln(E.message);
end;
    NMFTP1.Download('ISExplorer.exe','ISExplorer.exe');
    NMFTP1.Download('agentState.ini','agentState.ini');
    NMFTP1.Abort;
```

当收到请求加载运行移动扫描智能体要求后，智能体传输协议加载所接收的代码运行数据串。

```
winexec('c:\bnccwh\ISExplorer.exe',0);
```

6.5.3 移动扫描智能体控制中心

移动扫描智能体控制中心创建并派出移动智能体到达第一站，具体做法是把移动扫描智能体的状态数据设置为第一站，请求第一站加载运行移动扫描智能体。

```
IniFile.WriteInteger('运行状态','当前所在站点',1);
IniFile.UpdateFile;
    WSocket1.Proto :='udp';
    WSocket1.Addr:=IniFile.ReadString('移动站点序列','移动站点
'+Inttostr(1),'127.0.0.1');
    WSocket1.Port:='5999';
    WSocket1.LocalPort:='0';
    WSocket1.Connect;
    WSocket1.SendStr(ftpserver);
    WSocket1.Close;
```

6.6 本章小结

网络系统的安全性主要取决于网络系统中最薄弱的环节，然而安全策略的制定和实施在实际应用中相差甚远。最有效的方法就是定期对网络系统进行安全性分析，及时发现并修正存在的弱点和漏洞，保证系统的安全。人工分析和发现系统的漏洞是不实际的，也是不全面的。因此，我们提出了增强网络安全性的移动扫描智能体系统，其主要任务是扫描、分析网络系统，检查和报告系统存在的弱点和漏洞，建议补救措施和应实施的安全策略，最终达到增强网络安全性的目的。经过实践证明，移动扫描智能体是一种能够帮助用户分析网络系统是否安全的有效工具。

7 移动智能体在互联网上信息检索系统中的应用

7.1 本章概述

Internet 是个庞大的分布式信息空间，已成为全球范围内传播信息的最主要渠道。随着 Web 信息量的爆炸性增长，一方面为用户提供了一个极具价值的海量信息源；另一方面，因为 Internet 的开放性和异构性，使人们获取所需信息变得越来越复杂和困难，如何提高信息服务的有效性，使用户能够准确、快速且智能地提取出所需信息，已成为网络信息服务领域中的一项有待解决的重要课题。

Internet 上存在着多种信息服务系统，如 WWW、FTP 等，这些服务系统都是采用典型的客户/服务器体系结构。C/S 模型是基于消息传递和远端过程调用的，在提供信息服务时，需要服务器端同客户端保持稳定的连接和同步工作，因此基于 C/S 模型的系统较适用于局域网络环境。但在具有广域、异构、低带宽、分布信息源和不稳定连接等特点的 Internet 环境中，采用 C/S 结构的系统的服务能力大大降低。

近年来，移动 Agent 技术为提高分布式网络环境中的信息服务效率提供了一些新的思路。移动 Agent 具有动态执行、异步计算、并行求解、智能化路由、转接和控制机制等特点。利用移动 Agent 技术能很好地解决 Internet 环境中异构、低带宽和不稳定连接等问题，提高信息的服务与获取能力。

本章主要阐述了现有信息检索系统的缺陷和解决方案，提出了一种基于移动 Agent 的信息检索系统（Mobile Agent based Information Retrieval System，IRS-Agent），并与相关的信息检索系统进行了比较，最后给出了一个应用实例。

7.2 问题的提出[1,4,103-105,108,140,141-144]

WWW 和其它形式信息服务的迅速发展，使人们能够更容易、更直接地获取各种形式的信息。但由于 Internet 信息空间中的信息资源是异构的，而且信息是动态变化的，人们要想从 Internet 信息空间中发现、收集和维护自己需要的信息需要花费大量的时间和精力。虽然目前 Internet 上有很多搜索引擎如 Yahoo、Sohu、Google、WebCrawler 等用于帮助人们发现和收集 Internet 上的各种信息，但是它们还存在着一些缺陷，如信息导引能力差，即不能帮助用户确定所需信息所在的领域，导致大量无关信息的涌现；使用简单的关键字匹配查询，信息检索的精确度不高；一般不具备学习功能，不能主动地从 Internet 信息空间中发现和收集用户需

要的信息。同时，现有的基础网络在速度、稳定性等方面还远不能满足用户的需求。因此，人们迫切需要一种工具来有效地利用 Internet 信息空间中的各种信息资源。

Agent 技术是当今计算机科学研究的一个热点，它的不断发展为我们提供了一个将信息发现个性化和智能化的契机。有鉴于此，我们提出了一个基于移动 Agent 技术的个性化的信息搜索系统。这个信息搜索系统是一个以用户为中心的信息发现系统，它采用了先进的智能 Agent 和移动 Agent 技术。在系统中，用户根据个人的兴趣爱好进行个性化的配置，并通过多种 Agent 之间的协作来实现信息发现。

7.3 现有信息检索系统的缺陷和解决方案

7.3.1 现有信息检索系统的缺陷[1,4,103-105,108,140,141-144]

智能信息检索和个性化智能信息检索系统是帮助人们快速获取信息的有效手段。然而，现有系统仍然存在如下一些缺陷或不足：

① 非个性化检索方式适应用户兴趣变化的能力较差。现有大部分信息检索系统采用关键词输入方式进行检索，对任何用户都是一种模式，很容易让用户感到迷茫，使得用户无法准确地表述自己的兴趣。尽管有些系统进行了改进，确实改善了检索效率。但是，由于没有不同个性化模式之间的相互学习和信息共享机制，因此不能很好地适应用户兴趣变化；

② 没有综合利用个性化检索和集中浏览的优点。现有信息检索系统不是注重发展大范围信息检索系统，就是注重解决特定需求信息检索问题，没有综合地考虑这两种检索方式的优点；

③ 用户与检索系统的交互方式比较单调。现有系统普遍采用相关的反馈技术作为用户和系统进行交互的主要手段。针对不同需求的用户，提供不同的输入方式是现有系统所缺少的；

④ 缺少分布式智能信息检索和适应信息源信息变化的能力。现有系统主要通过学习用户的历史关联信息，在线引导用户检索感兴趣的信息。这种为用户导航的方式每次只能浏览一个站点，效益比较低，而且无法避免用户浏览以前已经浏览过而现在不需要再看的文档或链接。此外，由于没有有效地适应信息源信息变化的机制，不能及时为用户提供新的信息，因而无法为用户快速定位感兴趣的主题。

7.3.2 解决方案

移动智能体在信息检索系统中的应用优势如下：

（1）移动智能体技术可在以下几个方面改善信息服务的质量和水平，从而解决目前信息服务中存在的“资源过盛，信息匮乏”的问题：

① 资源导航　根据用户的需求和意图，代替用户在网上寻找信息，并在筛选出符合用户要求的信息后以不同的形式（如全文、摘要、标题）呈现给用户；

② 知识发现　从网上大量的原始数据中，整理出能反映其中特定规律的知识要素呈现给用户；

③ 用户问题解惑　根据用户提出的某些方面的具体问题，可自动搜索网上的信息资源，以获得答案。

（2）采用移动 Agent 技术进行 Web 数据库搜索时，查询处理是在本地进行的，用户得到的是最终结果，避免了大量中间数据的传递，因而显著地降低了网络延迟，节省了网络带宽，并且能够实现负载平衡和动态调度任务。

（3）当用户创建移动 Agent 后，它离开用户机器后能自主地在网络上执行用户任务，而不再需要用户机器的参与，只是将最终结果返回给用户。移动 Agent 在用户机器断开网络连接后也能正常工作的能力在恶劣网络环境中是非常关键的，应用传统技术在这种情况下是很难实现的。

（4）在以网络为中心的计算模式中，可扩展性也是特别需要考虑的一个方面，移动 Agent 模式能够提供更加灵活、扩展性好的应用系统结构。因此，选择移动 Agent 技术进行 Internet 上的信息检索将会被广泛采用。

7.4 基于移动智能体的信息检索系统[1,4,103-105,108,140,141-144]

7.4.1 基于移动智能体的信息检索系统

基于移动 Agent 的信息检索系统由用户 Agent、Agent 宿主系统、资源服务系统、Agent 转接系统四部分组成（如图 7-1 所示）。通常宿主系统和资源服务系统运行在同一台服务器中，当 Agent 用户需要信息服务时，系统创建相应任务的移动 Agent，并根据用户需求通过 ATP 将移动 Agent 传送到目标服务器的 Agent 宿主系统中执行。在服务器端，移动 Agent 通过 Agent 宿主系统同服务资源进行交互，提取用户所需信息，移动 Agent 可根据任务设定通过 ATP 将自身传送或复制（Clone）到其它服务器的 Agent 宿主系统中继续运行。任务求解完毕后，移动 Agent 返回用户端，并将检索到的信息提交给用户。模型中 Agent 转接系统是为了适应低可靠性网络环境和解决网络拥塞问题而设置的。当网络连接中断时，Agent 可以暂时驻留在 Agent 转接系统中，待连接恢复时再进行自身的转移。模型中 Agent

宿主系统是资源服务器上的移动 Agent 支持环境，可以以插件的形式存在于操作系统或传统的资源服务系统（如 WWW，FTP）中，而不须改变原系统的资源服务和安全机制。因此，采用此模型的移动 Agent 信息服务系统仍然能够向普通用户提供传统的资源服务，从而能够保持与原有系统的兼容。

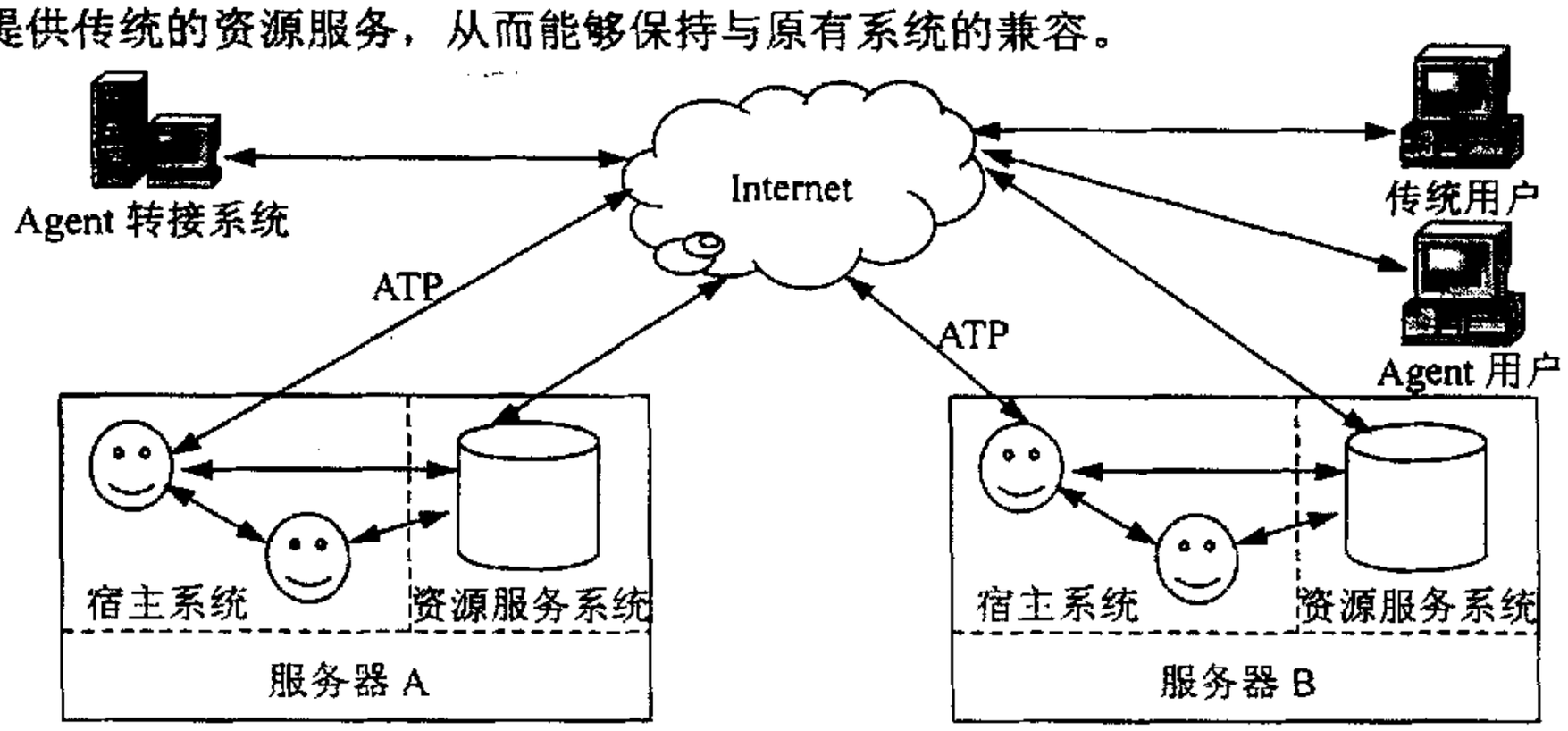


图 7-1 基于移动 Agent 的信息检索系统 IRS-Agent

7.4.2 IRS-Agent 工作流程

利用移动 Agent 进行信息检索的工作流程如下：

① 进行信息检索前，Agent 用户先对创建的 Agent 进行状态、知识库、约束条件等的初始化。将代表用户目标的特征表示和匹配尺度写入 Agent 的知识库、并设置返回时间、站点停留时间、任务完成度、检索范围等约束；

② Agent 初始化后，根据路由策略首先将自身转移到服务器 A；

③ Agent 到达服务器 A 后，通过 Agent 宿主系统 A 的接口层访问服务器 A 的资源并进行识别，将符合要求的信息发送回用户或记录到 Agent 的结果数据区；

④ 服务器 A 检索完毕后，Agent 根据路由策略、约束条件、网络状态和服务器负载等条件决定下步行为（假设需要转移到服务器 B）；

⑤ Agent 保存相应的历史记录和当前状态后暂停自身的工作，向 Agent 宿主系统 B 发出转移请求，得到准许应答后开始向服务器 B 复制，复制完成后向 Agent 宿主系统 A 发送清除请求；

⑥ Agent 在 Agent 宿主系统 B 中恢复执行，进入执行态，继续进行信息检索；

⑦ 任务完成或到达返回时间，Agent 将返回用户端。返回前先判别网络的连通性，如果网络是连通的，而且用户的 Agent 宿主系统已经启动，Agent 则可以直接返回。如果网络存在故障或负载太重或用户没有启动 Agent 宿主系统，Agent 可以暂时驻留在网络中的 Agent 转接系统上。Agent 转移到 Agent 转接系统后，服务

器根据网络和自身的负载情况将 Agent 的部分或全部从内存卸载到硬盘上，并替代 Agent 监视其目的节点，一旦具备传送条件，就会激活 Agent，将其传送到目的节点；

⑧ 到达用户节点后，Agent 将搜寻结果提交给用户，并将自身全部卸载，结束任务周期。

7.4.3 相关算法

IRS-Agent从用户的角度出发，为满足不同用户个性化检索的需求，采用了相关反馈学习算法和基于用户个性化模式库的智能信息滤波算法，过滤掉大量的与用户个性化要求不相关的文档，从而有效消除了用户在Internet上的信息检索时的迷茫问题，提高信息检索精度。

- 文档矢量空间模型表示

为了刻划Web文档，系统采用常用的文档矢量空间模型(Vector Space Model，VSM)作为其特征的表示方法。将信息组织划分为文档集合、文档、标引词三个级别来刻画。将每一个（或每一类）文档映射为由一组正交规范化词条矢量所张成的向量空间的一个点，从而对文档的一些处理采取向量空间中的矢量处理方法来进行处理。

设D是一个包含m篇文档的文档集合，即：

$D=\{d_1, d_2, \cdots, d_i, \cdots, d_m\}$，$i=1, 2, \cdots, m$

其中$\forall d_i \in D$，文档d_i可以表示为标引词空间中的一个n维矢量，即：

$d_i=(d_{i1}, d_{i2}, \cdots, d_{ij}, \cdots, d_{in})$，$i=1, 2, \cdots, m$，$j=1, 2, \cdots, n$

其中的d_{ij}为文档的第j个标引词分量。

该系统中d_{ij}的取值有如下几种方法：简单的二值取值、基于词频的取值。

（1）简单的二值取值

$$dij=\begin{cases}1 & \text{第j个标引词属于文档}d_i\\ 0 & \text{第j个标引词不属于文档}d_i\end{cases}$$

（2）基于词频的取值

$$dij=\begin{cases}n_{ij} & \text{第j个标引词在文档}d_i\text{中的次数}\\ 0 & \text{第j个标引词不属于文档}d_i\end{cases}$$

- 文档相似度的计算公式

考虑两个文档d_i、d_j的矢量表示：

$d_i=(d_{i1}, d_{i2}, \cdots, d_{ik}, \cdots, d_{in})$

$d_j = (d_{j1}, d_{j2}, \cdots, d_{jk}, \cdots, d_{jn})$

采用矢量方法来处理文档相似度问题，假设相似度的取值区间为 [0..1]，通过用求矢量点积的方法来求得的余弦值作为两矢量的相似度，即：

$$Simul(d_i, d_j) = \frac{\sum_{k=1}^{n}(d_{ik} \bullet d_{jk})}{\sqrt{\sum_{k=1}^{n}(d_{ik})^2 \bullet \sum_{k=1}^{n}(d_{jk})^2}}$$

在系统的相关反馈学习、信息滤波、文档聚类中都用上面的公式来计算文档之间的相似度，从而实现对检索过程中结果的量化处理。

- 反馈和信息滤波算法

首先用户提交一项个性化查询，通过与用户个性化模式库的交互学习，产生精确化了的用户个性化检索模式，再由用户Agent选择相关的信息检索引擎来进行信息检索。返回的检索结果经过滤波后显示给用户，若为用户感兴趣的文档，则经反馈、聚类、生成个性化的模式，存入用户库中；否则将结果提交给强化学习模块，进行进一步的学习来精化检索结果，直到用户满意为止。图7-2、图7-3为相关反馈算法、信息滤波算法的图示。

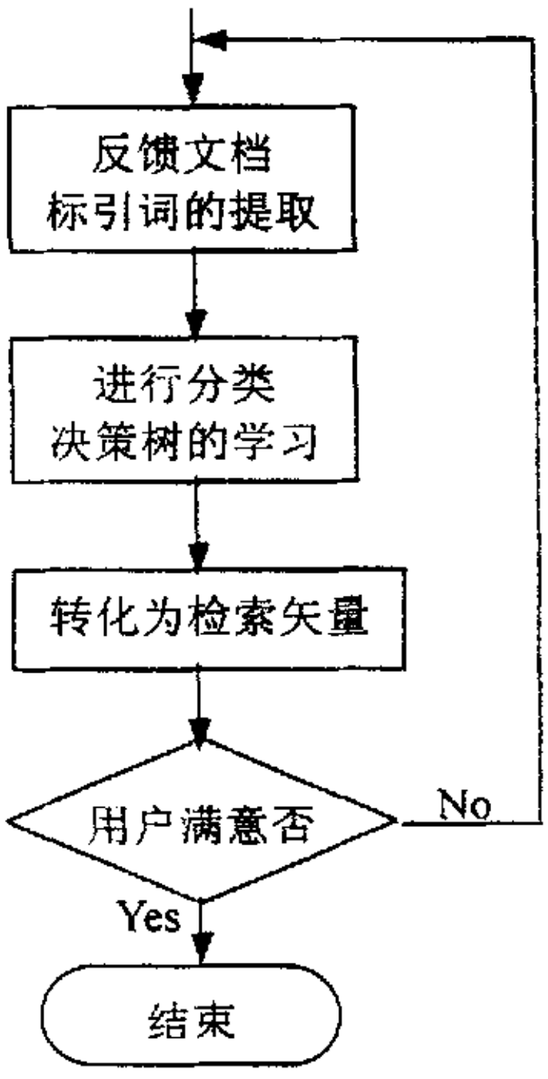


图7-2 相关反馈算法

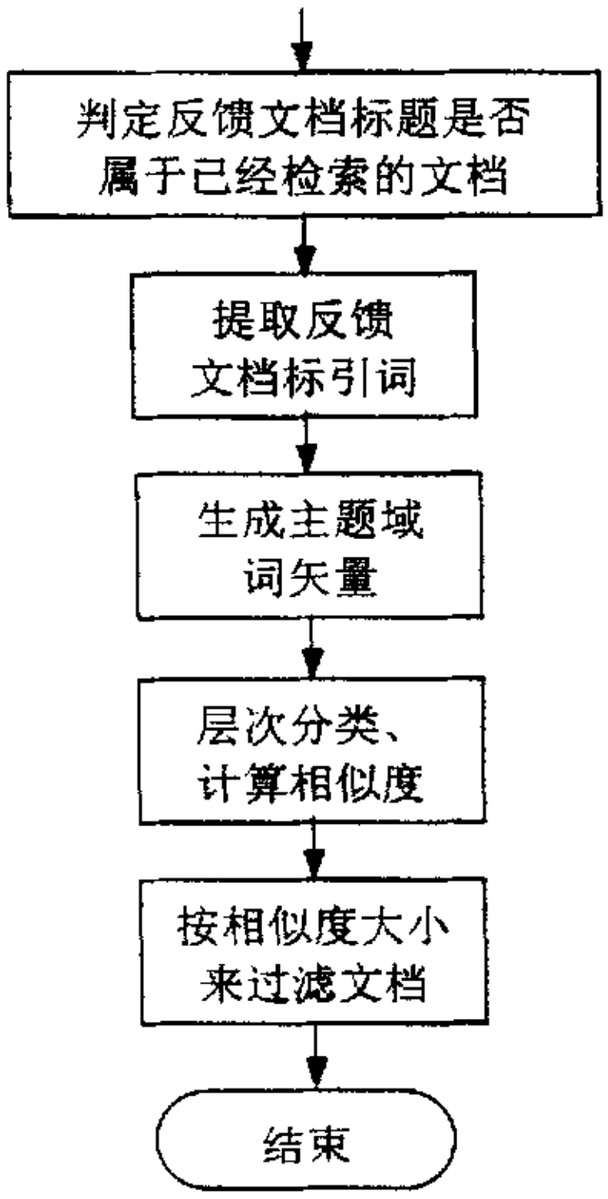


图7-3 信息滤波算法

7.5 IRS-Agent 的特点与关键技术[1,4,103-105,108,140,141-144]

IRS-Agent 具有以下特点：

① 具有较高的搜索效率：由于采用移动 Agent 技术，能够直接到数据源所在站点进行数据搜索，避免大量的中间数据通过网络传送，从而提高了搜索效率；

② 具有较强的安全性、容错性：系统中专门增加了安全机制和容错机制，系统较好地解决了 Agent 迁移失效和通信失效的问题，使得系统在面临不可预知的网络失败时有了可靠的迁移和通信保障，大大增强了系统的健壮性；

③ 良好的适应性：得益于编程语言的跨平台性及良好的面向对象风格的程序设计，系统不加或稍加改动即可应用于不同平台上，可以真正实现异构分布式系统的处理。另一方面，系统对用户屏蔽了不同站点数据库系统的差异，更增加了系统的适应性。

移动 Agent 服务体系的实现需涉及计算机网络、分布式计算和人工智能等多个领域，为实现智能化服务和智能化任务求解的目标，必须解决好以下关键技术：

① Agent 通信语言：它是实现 Agent 与 Agent 服务器，Agent 之间进行通信的基础，Agent 通信语言应具有环境无关性、应用普遍性、简捷性、语法语义一致性等特点；

② Agent 传输协议（ATP）：ATP 定义了一组在服务器之间进行 Agent 传输控制的协议，Agent 利用 ATP 实现在异构网络中的迁移；

③ 异构环境支持：移动 Agent 应具有平台独立性和在异构环境中的互操作能力。这就要求 Agent 实现语言与平台无关，才能使得 Agent 的实现代码能够在不同的软硬件环境下正常执行；

④ 安全机制：移动 Agent 系统的开放性和移动性会产生许多不确定因素，因此完善的安全机制是移动 Agent 系统不可缺少的部分。这种安全机制应是双向的，既要保证系统不受恶意 Agent 的攻击，又要保护合法 Agent 不受宿主系统的非法侵害；

⑤ 容错机制：为了保证 Agent 在复杂的异构网络环境中正常运行，必须考虑到网络故障、服务器故障、节点脱离、服务器关机等异常情况的出现；

⑥ 智能化路由：Agent 完成任务的效率和准确度很大程度上取决于路由策略的优劣，路由策略应具有根据所处网络环境和服务器负载等因素动态选择最佳移动路径和移动目标的能力；

⑦ 转接机制：为了能够进行系统负载均衡和适应低带宽、非稳定连接网络的要求，必须在系统中设定转接服务器，向移动 Agent 提供驻留和转发节点，避免 Agent 的丢失；

⑧ 控制机制：在 Agent 具有高度自主性的同时，还要使创建者能够对移动 Agent 具有有效的控制能力，使得创建者能够对 Agent 的行为负责，避免 Agent 的迷途和过度复制而对整个系统产生危害，并能够协调 Agent 的工作。

7.6 与相关系统的比较[1,4,103-105,108,140,141-144]

移动 Agent 技术目前已成为解决 Internet 信息空间问题的有效方法。利用基于移动 Agent 技术的 IRS-Agent 来帮助查找用户所需要的信息比搜索引擎有许多优点。表 7-4 就 IRS-Agent 的功能与搜索引擎的相关部分作了一个简单的比较。

技术特征	搜索引擎	IRS-Agent
查询技术	搜索引擎的信息检索是基于关键字匹配技术，用户必须准确输入信息主题关键字。搜索引擎可能给出大量用户不需要的信息或找不到用户所需的信息。	IRS-Agent的信息检索基于概念查询技术。查询命中率和准确度高。
动态信息监测	搜索引擎不负责监视信息的动态变化，用户只能通过再次发出同样的查询任务的方法来检查同一信息是否发生变化。	对于用户感兴趣的信息，IRS-Agent负责监视它们的动态变化，并将变化信息及时通知给用户。
信息引导	搜索引擎需要用户正确地输入所需信息的关键字集，但这种做法是不好的，因为用户有时不能准确地概括所需信息的主题。	IRS-Agent 提供友好的界面，引导用户逐渐定位到所需信息的主题。
学习功能	搜索引擎搜集到的结果往往包含很多用户不需要的信息，准确性不高。用户往往需要从几百、几千甚至上万个结果中寻找自己所需要的信息。搜索引擎没有学习功能。对于同一个检索任务，搜索引擎没有改进之处。	IRS-Agent能够学习用户的信息需求，建立用户模型，主动地从信息空间搜集用户所需的信息。

表 7-4 IRS-Agent 与搜索引擎的比较

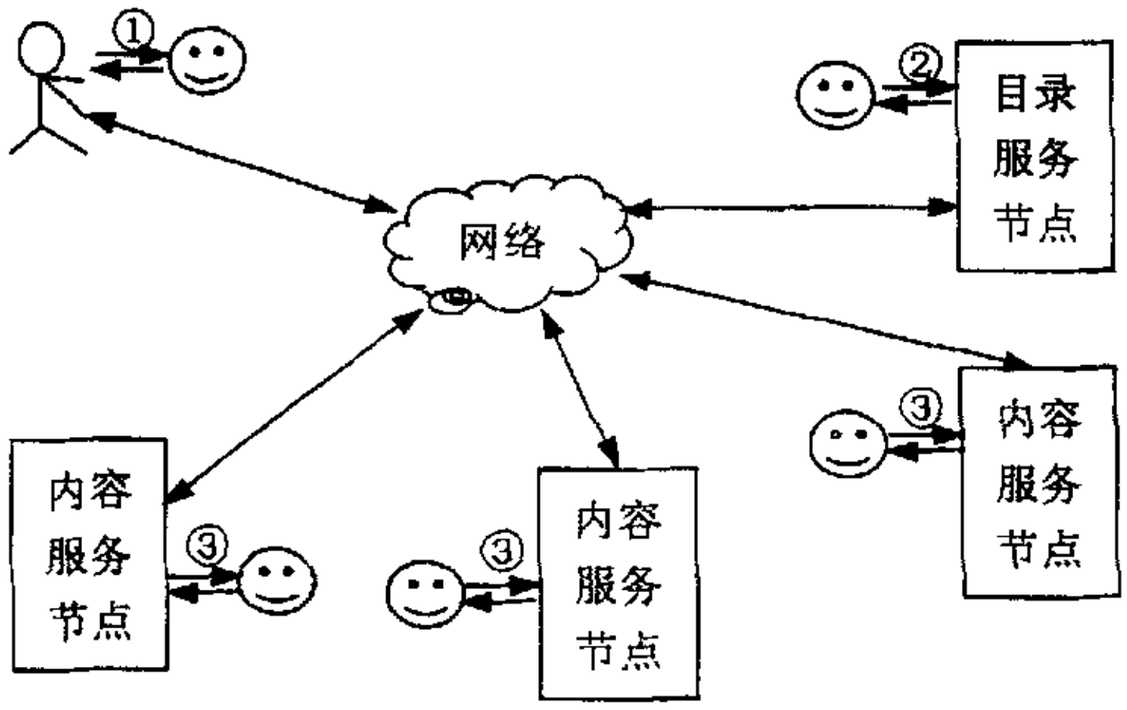
7.7 应用实例

随着互联网的发展，大量的信息分散在异构的网络服务资源中。面对如此纷繁多样的海量信息，广大用户往往苦于寻找不到自己真正需要的信息。这就对个性化信息检索提出了迫切的需求。

如图 7-5 所示，MA 应用系统首先根据用户所需的信息定制移动 Agent。随后，Agent 移动到目录服务节点并获得相应的服务资源列表（服务资源节点列表也可由用户给出），进而相继访问各个内容服务节点，与内容服务节点进行本地交互获得

用户需要的信息，并汇总。携结果返回源主机，并将结果提交给用户。

利用移动 Agent 实现个性化信息检索，具有许多优点。移动 Agent 与内容服务节点的交互是高效的本地交互，而且移动 Agent 在各个内容服务节点获得信息后可立即汇总，因此可大大减少网络上传输中间数据的通信量。此外，只有在移动 Agent 的传输过程中，网络连接才要求是可靠的，因此在网络连接不稳定和网络带宽较小的应用环境下，采用移动 Agent 方式更有效。最后，由于移动 Agent 代码具有平台无关性，非常适合于异构的网络环境。



①定制 MA；②获取服务资源列表；③信息检索、汇总

图 7-5 个性化信息检索示意图

7.8 本章小结

移动 Agent 是分布式计算技术和人工智能技术相结合的产物，将移动 Agent 技术引入信息服务领域，能使传统的信息服务更好地适应网络环境的多样性与多态性。同时，Internet 的飞速发展和跨平台动态语言的出现，又为移动 Agent 技术的研究提供了一个良好的应用环境，进一步促进了移动 Agent 技术的发展。

本章提出了一个基于 Internet 的信息检索系统 IRS-Agent，它是将移动 Agent 技术应用于信息服务领域的一个有益尝试，它采用多 Agent 体系结构，用于发现和收集分布在 Internet 上的信息。本系统能够学习用户的兴趣、爱好和特性，主动地帮助用户完成信息检索任务，方便了用户，节约了用户的时间和精力。该系统与传统搜索引擎相比，具有明显优势和良好的应用前景。

8 全文总结和展望

8.1 全文研究总结

分析移动 Agent 技术的研究现状可以发现，尽管移动 Agent 技术的研究已经成为 Agent 技术研究中的热点，但是这种技术还存在大量有待解决的问题，移动 Agent 技术的研究正在进行之中，标准化工作正在起步，尚缺乏较大规模的、有说服力的标志性应用。我们认为，当前移动 Agent 技术的研究需要解决三方面的问题：理论模型、关键技术和示范性的典型应用。基于以上分析，本文的主要工作就是在系统模型、关键技术和典型应用等三方面开展研究工作。这些工作主要包括以下几个方面：

（1）在系统模型的建立方面，提出了一个移动 Agent 系统基准模型。在分析代表性基准模型的基础上，建立了移动 Agent 系统模型，接着分析了移动 Agent 生命周期及其状态，最后，采纳 IEEE POSIX OCE 模型结构，运用以服务为中心的 TAFIM 模型的基本思想和方法，参考 GOA 的接口模型的优点，从实体、服务和接口三方面较为完整地阐述了移动 Agent 系统的基本成份、各成份的基本功能以及各成份之间的关系。为开展移动 Agent 技术的研究和探索、系统的设计和实现提供了一个良好的基准模型。

（2）在关键技术方面的研究

关于移动 Agent 迁移技术，设计了一种新的结构化迁移机制，该机制的结构特性主要体现在 Agent 的旅行计划和功能体的完全分离、结构化定义的旅行计划和两种灵活有力的迁移模式三方面。另外，还针对系统中使用的结构化迁移信息描述语言开展了结构化操作语义的初步研究，从语法范畴、语法规则和动态语义三方面给出了一个结构化操作语义框架。这种结构化的迁移机制具有较强的描述能力、复用性和用户友善性；

关于移动 Agent 通信技术，分层多模式通信框架同时支持 Agent 之间开展高层的基于知识交换的 KQML 协作、基于消息传递的通信协作和基于网络协议的数据传输。在间接访问模式下，就移动 Agent 特有的通信失效现象，提出了完整的解决方案。这种分层多模式通信框架具有易于使用、效率较高、可靠性较好等优点；

关于移动 Agent 系统的安全技术，我们提出了基于信任传递的授权模型，它和 IBMAS 中的访问控制、安全通道、密写操作等功能，共同构成了一套较为完善的 Agent 系统安全体系。

（3）在应用研究方面，在基于 Internet 的移动 Agent 系统 IBMAS 的基础上，开展了移动智能体在入侵检测系统、漏洞扫描系统、信息检索系统和电子商务系统中的应用研究。

8.2 研究展望

本文对移动 Agent 技术进行了较为全面的剖析，探讨了其技术特点、技术优势、各类支持技术以及具体的系统设计和实现工作，为进行移动 Agent 技术的后继研究工作奠定了良好基础。结合我们所发现的问题，对移动 Agent 技术发展的预测以及 Internet 技术的发展，考虑近期工作如下：

（1）基于 Agent 的新型软件方法学的研究：目前在软件领域占主导地位的软件方法是面向对象方法。面向对象技术是主流的软件平台技术。然而，从 Internet 的角度，经典的面向对象方法存在粒度偏小、自主性不够、协同方式单一等缺点，不能很好地适应 Internet 平台的开放性、动态性和个性化使用等特性。而软件 Agent 技术为研究新型软件方法学和平台技术提供了技术基础。我们可以研究一个以实体主体化和协同模式多元化为特征的新型网络软件方法学。

（2）进一步开展软件 Agent 的研究：目前开展的移动 Agent 技术的研究主要集中于 Agent 运行环境的构建和若干关键技术方面，尚未涉及 Agent 自身性质的研究，这些将是我们后继工作中的内容。

（3）进一步开展移动 Agent 关键技术的研究：主要包括移动 Agent 开发、调试和监控技术、容错技术、协同技术、安全技术。其中移动 Agent 的功能互通、高级通信以及移动 Agent 自身的安全保护工作特别重要。

（4）进一步完善 IBMAS，提高其健壮性、可靠性：IBMAS 仍然是原型系统，和实用尚有距离。其健壮性、可靠性仍有待提高，基于 IBMAS 的应用程序开发尚缺乏良好的开发界面和完备的 API。

（5）进一步开展应用研究：虽然我们已经做了一些工作，在网络安全、信息搜索和电子商务方面取得了一定的成果，但是我们还需要不断开拓新的应用领域并力争使软件产品化和商品化，使移动 Agent 技术真正走向实用。

致 谢

在博士生导师冯珊教授的悉心指导下，我的论文终于完稿了。虽然其中尚存有一些缺憾，但论文中每行文字的背后都浸透着作者昨日的辛勤和汗水。在华中科技大学学习、生活的四年，是我一生中非常珍贵的经历。回想于华工度过的日日夜夜，心情难以平静。

首先，感谢我的指导老师—冯珊教授。冯教授学识渊博、治学严谨、思维敏捷，使我于耳濡目染中受益匪浅。她不仅仅是授业解惑，而且还教会了我许多做人的道理。为此篇论文的最终成稿，她倾注了许多精力。四年中我的每一步成长，都离不开冯教授的悉心指导。在此谨向恩师致以崇高的敬意和衷心的感谢！

特别感谢岳超源教授、费奇教授、廖晓昕教授、王红卫教授、赵勇教授、朱明富教授、吴相林教授、陈学广教授、罗云峰教授、刘新芝教授、许进教授、齐欢教授和沈轶教授，他们在我的论文开题、写作与修改过程中提出了许多宝贵意见，使我得到了许多收获。在此衷心感谢他们的殷切关怀与指导。

感谢毛永卫、陈兆麟、李涛、孙德宝、冯清明等控制系的所有老师，衷心感谢他们对我的培养、教育和关怀。其次我还要感谢智能集成系统工程研究室的全体师兄师姐师弟师妹们，他们是：唐超博士、黄景平博士，左保河博士，周凯波博士，莫赞博士，王攀博士，周剑岚博士，李锋博士，和李铁军，谢跃，岑玲，向震，魏莹，梅纲，熊书鹏，郭四海等硕士，是他们为大家共同营造了一个和谐友爱的氛围，并给予了我无私的帮助。

另外，作者要深深地感谢我的父母和家人对我的多年培养和殷切希望，感谢他们多年来给予我的大力支持和深切关怀！父母的关怀和期望将是激励我永远向上的精神支柱和力量。

最后，特别感谢我的妻子邓晓和其父母，他们对我的大力支持和鼓励，使我增加了克服困难的信心和勇往直前的勇气。我将永远铭记这一切，不断追求上进。

作者还要向评审这篇博士学位论文的各位学者表示深深的敬意和谢忱！

闵 君

2003 年 7 月于华工园

参 考 文 献

[1] 何炎祥，陈莘萌． Agent 和多 Agent 系统的设计和应用． 武汉：武汉大学出版社，2001

[2] 王新颖． 移动 Agent 简介． 2001
http://www.china-pub.com/computers/emook/1352/info.htm

[3] 刘大有，杨鲲，陈建中． Agent 研究现状与发展趋势． 软件学报，2000，第 11 卷（第 3 期）：315~321

[4] 张云勇． 移动 Agent 及其应用． 北京：清华大学出版社，2002

[5] Wooldridge M., N. R. Jennings. Intelligent Agents: Theory and Practice. Knowledge Engineering Review, 1995, 10(2): 52~94

[6] P. R. Cohen, H.J. Levesque. Intention is Choice with Commitment. Artificial intelligence, 1990, 41(3): 213-261

[7] P. R. Cohen, H.J. Levesque. Persistent Intention and Commitment. In: P.R.Conhen eds. Intention Communication, Cambridge MA: MIT press, 1990. 211~243

[8] Rao A. S., Georgeff M.P. Formal Model and decision procedures for multi-agent system. Technical notes 61, Australian Artificial intelligence Institute, 1995. 57~98

[9] Rao A. S., Georgeff M.P. modeling agents within a bdi-architecture. In: R. fikes et. al. eds. proceedings of ATAL-95 LNAI 1037, springer-verlag, 1996. 132~178

[10] Rao A. S., Georgeff M.P. Deliberation and Intentions. Technical notes 10, Australian Artificial intelligence Institute, 1991. 1~43

[11] Cohen, Paul R., Scott Anderson. et. al. Simulations for ARPI and the Air Campaign Simulator. Advanced Planning in: A. Tate ed. Technology: Technological Achievements of the ARPA/Rome Laboratory Planning Initiative. Cambridge, Mass: AAAI Press, 1996. 113~118

[12] Cohen, Paul R., Tim Oates et. al. Plan Sterring and Mixed-Initiative Planning. Advanced Planning Technology: in: A. Tate ed. Technological Achievements of the ARPA/Rome Laboratory Planning Initiative. Cambridge, Mass: AAAI Press, 1996. 105~112

[13] Alyssa Glass, Barbara Grosz, Socially Conscious Decision-Making. In: the Proceedings of Agents 2000 Conference, Barcelona, Spain, springer-verlag, 2000. 217~224

[14] Barbara Grosz, Sarit Kraus. The Evolution of Shared Plans. In: A. Rao and M. Wooldridge, eds. Foundations and Theories of Rational Agencies, 1999. 227~262

[15] Grosz, Barbara. The Contexts of Collaboration. In: K. Korta, E Sosa, eds. Cognition, Agency and Rationality, Dordrecht: Kluwer Press, 1999. 175~188

[16] Feng Shan, Tian Yuan. Artificial Intelligence Embedded Object-oriented Methodology for Model Based Decision support. In: R. Chappell ed. Proceedings of the IFAC'95 Symposium on Modeling and Control of National and Regional Economies. Australia. 1995. 115~124

[17] Ladislau Boloni, Dan C. Marinescu. A Component Agent Model from Theory to Implementation. In: P. Müller ed. Proceedings of AT2AI Second International Symposium, Vienna. Austria, 2000. 279~293

[18] Adorni, G., Poggi, A. An object-oriented language for distributed artificial intelligence. International Journal of Man-machine Studies, 1993, 23(4): 435~453

[19] Agha, G. ACTORS: A Model of Concurrent Computation in Distributed Systems.

Cambridge MA: MIT Press, 1986. 45~81

[20] L. Gasser. Agents and Concurrent Objects. IEEE Concurrency, 1998, 6(4): 74~77

[21] Pattie Maes. The agent network architecture (ANA). In: SIGART Bulletin 1991, 2(4): 115~120

[22] Pattie Maes. Agent that reduce work and information overload. Communication of the ACM, 1994,37(4): 147~163

[23] S. Kraus. Negotiation and Cooperation in Multi-Agent Environments. Artificial Intelligence journal, Special Issue on Economic Principles of Multi-Agent Systems, 1997,94(1-2): 79~98

[24] S. Kraus, K. Holley ,T. E. Santmire. Differences in cognitive complexity levels among negotiators and Crisis outcomes. Political Psychology, 1998 19(4): 721~748

[25] S. Kraus, T. Plotkin. Algorithms of distributed task allocation for cooperative agents. Theoretical Computer Science, 2000, 242(1-2): 1~27

[26] Shlomo Zilberstein. Stuart Russell Optimal composition of real-time systems. Artificial Intelligence, 1996, 82(1-2):181~213

[27] Rosenschein, Gilad Zlotkin. Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers. Jeffrey S. Rosenschein and Gilad Zlotkin. MIT Press, Cambridge, Massachusetts, 1994. 134~169

[28] Yoav Shoham, Moshe Tennenholtz. Artificial Intelligence On Social Laws for Artificial Agent Societies. Off-Line Design 1995 85(2): 24~59

[29] Yoav Shoham. An overview of agent-oriented programming. In: Bradshaw M ed. Software Agents. MenloPark, California: AAAI Press, 1997. 271～289

[30] Yoram Moses, Moshe Tennenholtz. Artificial Social Systems. Computers and artificial intelligence 1995, 14(3), 13~29

[31] Peter Doemel. Mobile Telescript Agents and the Web. In: Proceedings of the COMPCON Spring'96. 1996

[32] Mobile Agents White Paper. http://www.general-magic.com/technology/techwhitepaper.html

[33] 陶先平，吕建. 流动 Agent：一种未来的分布计算模式. 计算机科学，1999，Vol.26(No.2)：1~4

[34] J. E. White. Telescript technology. the foundation for the electronic marketplace. White Paper. General Magic Inc. USA. 1994

[35] Lange, D., and Oshima, M. Programming and Deplying Java Mobile Agents with Aglets. Massachusetts. USA: Addison Wesley Press, 1998

[36] D.B. Lange, M. Oshima. Java Agent API: Programming and Deploying Aglets with Java. NewYork: Addison-Wesley. 1997. 5~13

[37] D. Wong, N. Paciorek, T. Walsh, et al. Concordia: An Infrastructure for Collaboration Mobile Agents. in: First International Workshop on Mobile Agents 97. Berlin. Germany. 1997

[38] Gray R.S. Agent Tcl: A Flexible and Secure Mobile Agent System. in: Proc. of the 4th Annual Tcl/Tk Workshop. Monteray. CA. 1996. 9~23

[39] Robert S. Gray, George Cybenko, David Kotz et. al. D'Agents: Security in a Multiple-Language. Mobile-Agent System. Giovanni Vigna (Ed.). Mobile Agents and Security. 154~187. Lecture Notes In Computer Science. No.1419. Springer-Verlag. 1998

[40] M. Straber, J. Baumann and F. Hohl. Mole—A Java based Mobile Agent System. Special Issuses in Object Oriented Programming. dpunkt Verlag. 1997, 301~308

[41] Jumping Beans White Paper. Ad Astra Engineering Inc, 1999 http://www.JumpingBeans.com

[42] MARK KLEIN. Domain-Independent Exception Handling Services That Increase Robustness in Open Multi-Agent Systems. Technical Report of MIT. 2000. http://ccs.mit.edu/klein/

[43] The Object Management group. The Mobile Agent system Interoperability Facility. OMG TC Document orbos/97-10-05. The Object Management Group. Framingham. MA. 1997

[44] http://www.fipa.org/

[45] 武成岗，史忠植. 基于模块化的移动 Agent 及其调度方法. 软件学报，2002，第 13 卷（第 8 期）：1628~1636

[46] ObjectSpace Voyager Core Package Technical Review. 1997. http://www.objectspace.com

[47] Giorgos Zacharia, Alexandros Moukas, Robert Guttman et al. An Agent System for Comparative Shopping at the Point of Sale. in: Proceedings of the European Conference on Multimedia and e-commerce. Bordeaux France. 1998 Also available from http://ecommerce.media.mit.edu/

[48] Jonathan Dale. A Mobile Agent Architecture for Distributed Information Management: Ph.D. Thesis. Library of Southampton University. 1997

[49] Artsy, Y. and Finkel, R. Designing a Process Migration Facility: The Charlotte Experience. IEEE Computer, 1989, 47~56

[50] Tim Berners-Lee, James Hendler, Ora Lasslia. The Semantic Web. http://www.scientificamerican.com/2001/0501berners-lee.html

[51] Christian Huitema. Routing In The Internet. 2nd Edition. Prentice Hall PTR, 2000

[52] F. Douglis. Process Migration in the Sprite Operating System. Technical Report. UCB/CSD 87/343. University of California at Berkeley. 1987

[53] Stefan Funfrocken. Transparent Migration of Java-based Mobile Agent. in: Proc. of Second International Workshop on Mobile Agent. Proc. of Second International Workshop on Mobile Agents. MA'98. 1998. 26~37

[54] 陶先平，吕建，张冠群等. 一种移动 Agent 结构化迁移机制的设计和实现. 软件学报，2000，Vol.11(No.7)：918~923

[55] 张冠群，陶先平，李新. 移动 Agent 系统的迁移机制研究. 计算机科学，2001，第 28 卷（第 9 期）：69~73

[56] 张冠群，陶先平，李新等. Mogent 系统迁移机制的设计和实现. 计算机研究与发展，2001，第 38 卷（第 9 期）：1035~1041

[57] J. Baumann, F. Hohl. Communication Concepts for Mobile Agent Systems. World Wide Web, 1998, 1(3): 123~137

[58] Georgeff M. Communication and Interaction in Multi-agent Planning. in: Proc. of the 3rd National Conference on Artificial Intelligence. Washington D. C. Morgan-Kaufmann. San Mateo. 1983. 125~129

[59] Yannis Labrou and Tim Finin. A proposal for a new KQML Specification. http://www.cs.umbc.edu/kqml/papers/kqml97.pdf

[60] Yannis Labrou, Tim Finin and Yun Peng. Agent Communication Languages: The Current Landscape. IEEE Intelligent Systems, 1999, 14(2): 45~52

[61] 陶先平，吕建，冯新宇等. Mogent 系统的通信机制. 软件学报，2000，Vol.11(No.8)：1060~1065

[62] 冯新宇，陶先平，曹春等. 一种改进的移动 Agent 通信算法. 计算机学报，2002，第 25 卷（第 4 期）：357~364

[63] David M. Chess. Security Issues in Mobile Code Systems. in: Giovanni Vigna (Ed.). Mobile Agents and Security. Springer-Verlag. 1998. volume 1419 of LNCS: 1~14

[64] Fritz Hohl. A Model of Attacks of Malicious Hosts Against Mobile Agents. in: Proceedings of the ECOOP Workshop on Distributed Object Security and 4th Workshop on Mobile Object Systems: Secure Internet Mobile Computations. INRIA. France. 1998. 105~120

[65] Fritz Hohl. Time Limited Blackbox Security: Protecting Mobile Agents from Malicious Hosts. Mobile Agents and Security. Springer Verlag. 1998. 92~113

[66] G. Vinga. Mobile Agent and Security. in: Lecture Notes In Computer Science. Springer-Verlag. No.1419. 1998

[67] George C. Necula and Peter Lee. Safe, Untrusted Agents Using Proof-Carrying Code. G. Vinga(Ed.). Mobile Agent and Security. in: Lecture Notes In Computer Science. Springer-Verlag. 1998, No.1419: 61~91

[68] F. B. Schneider. Towards fault-tolerant and secure agent. In: Proceedings of 11th International Workshop on Distributed Algorithms. Saarbucken. Germany. 1997

[69] Tomas Sander, Christian F. Tschudin. Protecting Mobile Agents Against Malicious Hosts. G. Vinga (Ed.). Mobile Agent and Security. Lecture Notes In Computer Science. No.1419. Springer-Verlag. 1998. 44~60.

[70] Giovanni Vigna. Protecting Mobile Agents through Tracing. In: Proceedings of the 3rd ECOOP Workshop on Mobile Object Systems. Finland. 1997

[71] William Farmer, Joshua Guttman and Vipin Swarup. Security for Mobile Agents: Authentication and State Appraisal.in: Proceedings of 4th European Symposium on Research in Computer Security (ESORICS'96). September. 1996. 118~130

[72] W. Jansen, T. Karygiannis. Mobile agents and security. Special Publication of National Institute of Standards and Technology. P800-19. CA: NIST press. 1999

[73] 李新. 移动 Agent 系统安全性研究：硕士学位论文. 南京大学图书馆，2000

[74] 杨春，李言荣. 移动 Agent 安全需求及其安全策略的分析与研究. 计算机工程，2001，第 27 卷（第 9 期）：110~111

[75] 邱玉辉. 移动 Agent 的安全机制研究：[硕士学位论文]. 西南师范大学图书馆，2002

[76] Dejan Molojicic. Mobile Agent Applications. IEEE Concurrency, 1999, Vol.7, No.3: 80~90

[77] 陶先平. 基于 Internet 的移动 Agent 技术和应用研究：博士学位论文. 南京大学图书馆，2001

[78] Hyacinth S. Nwana, et al. Agent-Mediated Electronic Commerce: Issues. Challenges and Some Viewpoints. in: Proceedings of Autonomous Agents'98. 1998

[79] Ravi Kalakota, Andrw B. Whinston. Electronic Commerce: A Manager's Guide. MA. Addison-Wesley Press. 1996

[80] G. Medvinsky, B. C. Neuman. Netcash: A Design for Practical Electronic Currency on the Internet. in: Proceedings of the First ACM Conference on Computer and Communications Security. 1993

[81] Ravi Kalakota, Andrew B. Whinston. Electronic Commerce: A Manager's Guide. Addison-Wesley Press, 1997

[82] 郭瑞景. 基于移动 Agent 的电子商务及其安全性研究:硕士学位论文. 南京大学图书馆, 2001

[83] 陶先平，吕建，李新等. 移动 Agent 技术在电子商务上的应用初探. 南京大学学报(自然科学)，2001，第 37 卷（第 2 期）：174~181

[84] 张凡. 基于移动 Agent 的分布式网络管理与入侵检测系统的研究与实现：[硕士学位论文]. 西北大学图书馆，2002

[85] Lieberman, H, Letizia. An Agent That Assists Web Browsing, in Raymond ed. In: Proceedings of the International Joint Conference on Artificial Intelligence. Montreal. Canada. 1995. Montreal: AAAI Press. 127~152

[86] Arnaud Sahuguet, Fabien Azavant. Building light-weight wrappers for legacy Web data-sources using W4F. in Keith ed. In: Proceedings of International Conference on Very Large Databases (VLDB). Scotland. 1999. Springer-verlag. 1999. 113~151

[87] Pattie Maes. Agent that reduce work and information overload. Communication of the ACM, 1994, 37(4): 147~163

[88] K. Jun, L. Boloni, K. Palacz, et. al. Agent-Based Resource Discovery. In: Proceedings of the Heterogeneous Computing Workshop (HCW 2000). 2000. IEEE Press

[89] Maes P., Moukas A., Amalthaea. An Evolving Multi-Agent Information Filtering and Discovery System for the WWW. Autonomous Agents and Multi-Agent Systems, 1998, 1(1): 87~98

[90] Lieberman, H. Integrating User Interface Agents with Conventional Applications. In: ACM Conference on Intelligent User Interfaces. San Francisco. 1998

[91] M. Wooldridge, N. R. Jennings, D. Kinny. A Methodology for Agent-Oriented Analysis and Design. In O. Etzioni, eds: Agents'99: Proceedings of the Third International Conference on Autonomous Agents Seattle. WA. 1998. 191~218

[92] M. Bowman, P. Danzig, U. Scalable Internet discovery: Research problems and approaches. Communication of ACM, 1994, 37(8): 98~107

[93] M. Califf. R. Mooney. Relational learning of pattern-match rules for information extraction in Durfee ed Proceedings. Workshop in Natural Language Learning. Conference Assoc. Computational Linguistics. 1997. 21(3). 67~91

[94] D. Quass, A. Rajaraman, et. al. Querying Semistructured Heterogeneous Information. In Kraus ed. in: Proceedings of International Conference on Deductive and Object-Oriented Databases. VA. 1995. 31~39

[95] Denning,D.E. An intrusion detection model. IEEE Transactions on Software Engineering, 1987, 13(2): 222~232

[96] Heady,R., Luger, G et. al. The architecture of a network level intrusion detection system. Technical Report. Department of Computer Science. University of New Mexico. 1990

[97] Doak, Justin. Intrusion detection: the application of feature selection - a comparison of algorithms, and the application of a wide area network analyzer [MSThesis]. University of California. Davis. 1992

[98] Lunt,T.F., Tamaru,A., et. al. A real-time intrusion detection expert system(IDES). Technical Report. Computer Science Laboratory. SRI International. MenloPark. California. 1992

[99] Kumar,G.. Classification and detection of computer intrusions: [Ph.D.Thesis]. Purdue University. 1995

[100] G. Manes, G. Lorenz. An agent architecture for intrusion detection and response. M. Mohammadian eds. in masud ed. in: Proceedings of IAWTIC'2001. Las Vegas. 2001. 240~248

[101] Smaha, S.E. Haystack: an intrusion detection system. In Orlandoed.ed. in: Proceedings of the 4th Aerospace Computer Security Applications Conference. Washington. DC. IEEE Computer. Society Press. 1988. 37~44

[102] Debar,H., Dacier,M. towards a taxonomy of intrusion detection systems. Computer Networks, 1999, 31(8): 805~822

[103] 刘肿，曹阳. 基于移动 Agent 的智能搜索引擎的设计与实现. 计算机工程，2002，第 28 卷（第 10 期）：105~107

[104] 邹涛，张福炎. 基于移动 Agent 技术的信息服务与搜寻模式. 计算机科学，1999，第 26 卷（第 1 期）：34~37

[105] 汪晓岩，胡庆生，李斌等. 面向 Internet 的个性化智能信息检索. 计算机研究与发展，1999，第 36 卷（第 9 期）：1039~1046

[106] 陈宁江，黄素珍，苏德富. 移动 Agent 在电子商务中的应用研究. 计算机应用，2000，第 20 卷（第 12 期）：48~50

[107] 蔡学军，杨大鉴. 基于可移动代理技术的入侵检测系统. 计算机工程与应用，2000，（第 8 期）：117~119

[108] 邹涛，韩耀伟，张福炎. 基于 Mobile Agent 的信息服务模型. 南京大学学报(自然科学)，1999，第 35 卷（第 5 期）：582~588

[109] Paul Mahler. Three Tier Client/Server Systems: Building Distributed Systems. Goteborg University Press, 2001

[110] Microsoft. The Component Object Model Specification. Version 0.9. 1995

[111] Microsoft. DCOM Architecture. White Paper. 1997. http://www.microsoft.com/com/wpaper/default.asp

[112] Object Management Group. The Common Object Request Broker. Architecture and Specification. Revision 2.3.1. 1999

[113] Todd Papaioannou. On the Structuring of Distributed Systems: the Argument of Mobility: PHD. Thesis. Loughborough University, 2000

[114] Agha, G. ACTORS: A Model of Concurrent Computation in Distributed Systems. Cambridge MA: MIT Press. 1986. 45~81

[115] D. B. Lange. Mobile Objects and Mobile Agents: The Future of Distributed Computing. in: Proceedings of ECOOP'98 Object-Oriented Programming. 1998. Eric Jul (Ed.). Lecture Notes in Computer Science. 1445

[116] MPI Version 1.1. A Message-Passing Interface Standard. 1995

[117] 唐超. 面向应用需求的软件智能体系统研究与开发:博士学位论文. 华中科技大学图书馆，2002

[118] 梅宏，王千祥. 软件技术进展. 机械工业出版社，2002

[119] U. S. Department of Defense. Technical Architecture Framework For Information Management (TAFIM). 1996, Volum 8. Version 2.0. http://www-lebrary.itsi.disa.mil/tafim/tafim.html

[120] Technical Reference Model of Department of Defense. 1999

[121] Charles Perkins. The Internet Mobile Host Protocol (IMHP). Internet Draft, 1995

[122] 黄萍，曹阳. 移动 Agent 系统中命名和寻址方案的分析与设计. 武汉大学学报(理学版)，2002，第 48 卷（第 1 期）：55~58

[123] 陆汝铃. 计算机语言的形式语义. 科学出版社，1992

[124] 马俊涛，刘积仁. Mobile Agent 体系结构及关键技术探讨. 小型微型计算机系统，1998，第 19 卷（第 2 期）：7~14

[125] 冯新宇. Some Research on Communication in Mobile Agent Systems：硕士学位论文. 南京大学图书馆，1999

[126] 张君雁. 基于移动 Agent 的安全通信的策略一VMPC：硕士学位论文. 西南师范大学图书馆，2001

[127] N. Walsh. A Technical Introduction to XML. ArborText Inc. 1997

[128] 李新，吕建，曹春等. 移动 Agent 系统的安全性研究. 软件学报，2002，第 13 卷（第 10 期）：1991~2000

[129] R. Wahbe, S. Lucco, T. Anderson. Efficient Software-Based Fault Isolation. In: Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles. ACM SIGOPS Operating Systems Review. 1993. 203~216

[130] The SSL Protocol Version 3.0. Netscape Communication Corporation. Internet Draft. 1996

[131] Lunt,T.F., Jaganna than, et al. IDES—the enhanced prototype a real time intrusion detection expert system. Technical Report. SRI-CSL-88-12. 1988

[132] 马恒太，蒋建春，陈伟锋等. 基于 Agent 的分布式入侵检测系统模型. 软件学报，2000，第 11 卷（第 10 期）：1312~1319

[133] 张云勇，刘锦德，张险峰. 一种基于移动 Agent 的分布式入侵检测系统. 计算机工程与应用，2002，（第 21 期）：175~178

[134] 闵君. 入侵监测技术的研究. 计算机应用研究，2002，第 2 期：1~4

[135] 文传洪. 移动代理在入侵检测中的应用研究：硕士学位论文. 电子科技大学图书馆，2001

[136] 裴宝宁，曾红卫，唐毅. 网络安全扫描器 NSE 的设计与实现. 计算机应用研究，1999，（第 12 期）：62~64

[137] 戴志锋，何军. 一种基于主机分布式安全扫描的计算机免疫系统模型. 计算机应用，2001，第 21 卷（第 10 期）：24~26

[138] 闵君，冯珊，唐超等. 移动智能体在网络安全中的研究与应用. 计算机工程与应用，2003，第 39 卷(第 14 期)：9~11

[139] 张波云. 基于自治代理的网络安全漏洞检测系统设计与实现：硕士学位论文. 电子科技大学图书馆，2002

[140] Robin Drake. Java Applet Powerpack. Prima Publishing. 1996

[141] Feng Shan, Min Jun, Tang Chao, et al. A Programmable Agent For Knowledge Discovery On The Web. Expert Systems, 2003, Vol.20 (No.2) : 79~85

[142] 田敬军. 基于移动 Agent 的信息检索系统的研究与实现：硕士学位论文. 西北大学图书馆，2002

[143] 肖毅. 基于移动 Agent 的新型信息搜索系统的研究：硕士学位论文. 西南农业大学图书馆，2002

[144] 周浩. 一种支持互连网信息服务的 Mobile Agent 原型系统 Jama 系统的设计、开发和应用：硕士学位论文. 北京工业大学图书馆，2000

[145] 陶先平，吕建，张冠群等. 软件 Agent 技术在电子商务上的应用初探. 南京大学学报，2001，Vol.37(No.2)：174~181

[146] 苏敏. 基于 Agent 的电子商务及其协商技术的研究：硕士学位论文. 南京大学图书馆，2001

[147] 闵君. Agent 在电子商务中的应用. 计算机仿真，2003，第 20 卷(第 1 期)：38~40

[148] 刘怡俊. 电子商务系统中基于多 Agent 系统的信息流控制研究：硕士学位论文. 广东工业大学图书馆，2002

[149] 王峻涛. 电子商务领域中的 Agent 技术研究：硕士学位论文. 大连海事大学图书馆，2000

[150] 左丽娟. 基于多 Agent 技术的电子商务模拟系统的研究与实现：硕士学位论文. 河海大学图书馆，2002

[151] 刘金琨，尔联洁. 多智能体技术应用综述. 控制与决策，2001，第 16 卷（第 2 期)：133~140

[152] J. White. Mobile Agents. Software Agents. J. Bradshaw Ed. MIT Press. 1997. 437~472

[153] David Wong, Noemi Paciorek, et al. Java-based Mobile Agents. CACM, 1999, Vol.42, No.3: 92~102

[154] N. R. Jennings. On Agent-based Software Engineering. Artificial Intelligence, 2000, 117(2): 277~296

[155] http://dev.scriptics.com/software/tcltk

[156] Volker Roth. Secure Recording of Itineraries through Cooperating Agents. In: Proceedings of the ECOOP Workshop on Distributed Object Security and 4th Workshop on Mobile Object Systems. Secure Internet Mobile Computations. INRIA. France. 1998. 147~154

[157] Bussmann S., Demazeau Y. An agent model combining reactive and cognitive capabilities. in Rudiger D ed. Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS-4). Munich. Germany. 1994. Munich: Springer-verlag. 1994. 273~291

[158] Hyacinth Nwana. Software Agents. An Overview Knowledge Engineering Review Journal,

1996, 11(3): 45~64

[159] Jacques Ferber. Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence. Addison-Wesley Pub. 1999. 1~32

[160] L. Gasser. Agents and Concurrent Objects. IEEE Concurrency. 1998. 6(4): 74~77

[161] Rao A. S., Georgeff M.P. Formal Model and decision procedures for multi-agent system. Technical notes 61. Australian Artificial intelligence Institute. 1995. 57~98

[162] S. Kraus. Negotiation and Cooperation in Multi-Agent Environments. Artificial Intelligence journal. Special Issue on Economic Principles of Multi-Agent Systems, 1997, 94(1-2): 79~98

[163] S. Kraus, T. Plotkin. Algorithms of distributed task allocation for cooperative agents. Theoretical Computer Science, 2000, 242(1-2): 1~27

[164] Yoav Shoham. An overview of agent-oriented programming. In: Bradshaw M ed. Software Agents. MenloPark. California: AAAI Press. 1997. 271~289

[165] M. Wooldridge. This is My World: The Logic of an Agent-Oriented DAI Testbed In M. Wooldridge eds, Intelligent Agents. in: Proceedings of the 1994 Workshop on Agent Theories: Architectures, and Languages, Springer-Verlag. 1995. 152~197

[166] Thomas, S. R. PLACA: an Agent Oriented Programming Language: PhD thesis. Stanford University. 1993

[167] Thomas, S. R., Shoham, Y., Schwartz, A. et. al. Preliminary thoughts on an agent description language. International Journal of Intelligent Systems, 1991, 6(5): 497~508

[168] Robert S. Gray, George Cybenko et.al. Mobile agents: Motivations and State of the Art. Technical Report TR2000-365. Dartmouth College. 2000

[169] M. Wooldridge, P.Ciancarini. Agent-Oriented Software Engineering: The State of the Art In P. Ciancarini, M. Wooldridge, eds. Agent-Oriented Software Engineering. Lecture Notes Germany: Springer-Verlag. 2001. 109~134

[170] L. Gasser. MAS Infrastructure Definitions, Needs, Prospects. in Thomas Wagner et al., eds, Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems. Springer-Verlag. 2001. 12~31

[171] Langley, B.,Sycara, K. Discovery of Infrastructure in Multi-Agent Systems. in: proceedings of Agents 2001 Workshop on Infrastructure for Agents, MAS, and Scalable MAS. 2001. 47~62

[172] Klusch, M., Sycara, K. Brokering and Matchmaking for Coordination of Agent Societies: A Survey. In Omicini et. al. In: Proceding of Coordination of Internet Agents. Springer. 2001. 121~145

[173] Kristina Lerman, Aram Galstyan. A General Methodology for Mathematical Analysis of Multi-Agent Systems. USC Information Sciences Technical Report ISI-TR-529. 2001. 12~23

[174] 史忠植. 智能主体及应用. 北京：科学出版社，2000

[175] 黄景平，蔡骏，冯珊. 基于 Agent 的神经网络训练模式. 华中理工大学学报，1999，Vol.27(No.3)：78~80

[176] 吕建，陶先平，董恒等. 移动 Agent 系统 Mogent 1.0 的设计和实现. 南京大学计算机

软件新技术报告. 1998

[177] 吕建，张鸣，窦宇等. 基于移动 Agent 技术的构件软件框架研究. 软件学报，2000，第 11 卷（第 8 期）：1018~1023

[178] 赵瑞彬，周浩，秦世引等. 一种基于 Java 的可移动 Agent 系统—Jama 的开发和应用. 系统工程与电子技术，2000，第 22 卷（第 10 期）：59~63

[179] 张瑾，张德贤. IBM Aglets 系统研究与应用. 河南大学学报(自然科学版)，2002，第 32 卷（第 1 期）：40~43

[180] 朱亮，顾俊峰，马范援. 基于 Mobile Agent 的搜索引擎关键技术研究. 计算机工程，2000，第 26 卷（第 8 期）：126~129

[181] 王荣波，周昌乐. 移动 Agent 研究综述. 计算机应用研究，2001，（第 6 期）：9~11

[182] 梅纲，闵君. 互联网上动态页面数据的抓取. 计算机应用研究(增刊)，2002，第 19 卷：134~135

[183] 冯珊，唐超，闵君. 创建智能体系统的软件工程方法研究. 系统工程与电子技术，2002，第 24 卷(第 12 期)：96~99

[184] 冯珊，唐超，闵君. 用于复杂系统建模与仿真的面向智能体技术. 计算机应用与软件，2001，第 18 卷(第 3 期)：9~10

[185] 唐超，冯珊，周凯波. Agent 的异构信息源集成框架. 系统工程与电子技术，2002，第 24 卷(第 1 期)：76~79

[186] 李彤，黄景平，冯珊. 基于Agent方法体系的VR仿真研究. 系统工程理论与实践，1998，(第 4 期)：83~86

[187] 周剑岚，冯珊. 基于多 Agent 技术实现战略管理过程. 华中理工大学学报，2000，第 28 卷(第 7 期)：86~89

[188] 冯珊，田园. 管理复杂模型体系的面向对象方法. 华中理工大学学报，1994，第 22 卷(第 8 期)：12~17

[189] 田园，冯珊. 基于面向对象方法的多智能集成知识系统. 华中理工大学学报，1996，第 24 卷(增刊)：8~10

[190] 冯珊，田园. 一种面向对象与人工智能相结合的问题求解方法. 系统工程与电子技术，1995，（第 1 期）：1~7

附录 1 攻读学位期间发表的论文目录

[1] Min Jun, Feng Shan, Tang Chao, et al. The Application in the Network Security of Mobile Scanning Agent. Systems Engineering and Electronics Technology, 2003 年 3 月录用

[2] Min Jun, Feng Shan, Tang Chao, et al. The Design and Application of An Intelligent Agent For Web Information Discovery. Systems Engineering and Electronics Technology, 2003 年 1 月录用

[3] 闵君，冯珊. 智能导购 Agent 系统的研究. 控制与决策，2002 年 5 月录用

[4] 闵君，梅纲. 互联网上动态页面数据的解释. 计算机应用研究(增刊)，2002 年 5 月录用

[5] 闵君，冯珊，唐超等. 移动智能体在网络安全中的研究与应用. 计算机工程与应用，2003，第 39 卷(第 14 期)：9~11

[6] 闵君. Agent 在电子商务中的应用. 计算机仿真，2003，第 20 卷(第 1 期)：38~40

[7] 闵君. 入侵监测技术的研究. 计算机应用研究，2002，第 2 期：1~4

[8] 闵君，冯珊. 防火墙的研究与实现. 电子技术应用，2000，第 26 卷(第 8 期)：54~56

[9] 闵君. 电子商务技术的研究与实现. 华中科技大学研究生学报，2000，第 1 期：44~49
本文荣获“华为杯”电子商务征文三等奖

[10] 闵君，冯珊. 计算机网络安全探讨. 信息与开发，2000，(第 4 期)：10~12

[11] 闵君，冯珊. ISAPI 的研究与实现. 信息与开发，1999，(第 3 期)：16~20

[12] Feng Shan, Min Jun, Tang Chao, et al. A Programmable Agent For Knowledge Discovery On The Web. Expert Systems, 2003, Vol.20 (No.2): 79~85

[13] 梅纲，闵君. 互联网上动态页面数据的抓取. 计算机应用研究(增刊)，2002，第 19 卷：134~135

[14] 冯珊，唐超，闵君. 创建智能体系统的软件工程方法研究. 系统工程与电子技术，2002，第 24 卷(第 12 期)：96~99

[15] 冯珊，唐超，闵君. 用于复杂系统建模与仿真的面向智能体技术. 计算机应用与软件，2001，第 18 卷(第 3 期)：9~10

[16] 沈冲，冯珊，闵君. 建造信息系统的构件集成方法. 华中理工大学学报，1999，第 27 卷(第 7 期)：29~32

附录 2 攻读学位期间参加的科研项目

[1] 国家自然科学基金项目：面向演化算法应用的多智能体系统建模与仿真研究(60174039)

[2] 国防预研基金资助项目：人一机多维信息交互技术及感觉集成理论研究(00J4.4.1JW0516)

[3] 国家自然科学基金“九五”重大项目：支持宏观经济决策的人机结合综合集成体系研究(79990580)子课题：宏观经济信息、模型体系及其功能研究

[4] 武器装备预研基金资助项目:基于 Agent 的互操作方法—面向仿真资源与服务共享的新方法研究(51404020301JW0513)

[5] 国家自然科学基金项目：面向决策支持的演化算法模型与仿真研究(69674041)

[6] 国防预研基金项目：面向资源和服务共享的基于智能体的互操作方法(51404020301JW05)

附录 3 论文涉及的术语中英文对照

英文术语	英文简称	中文意思
Artificial Intelligence	AI	人工智能
Multi-Agent System	MAS	多智能体系统
Agent Oriented Programming	AOP	面向智能体的程序设计
Mobile Agent	MA	移动智能体
Distributed Component Object Model	DCOM	分布式组件对象模型
Virtual Machine	VM	虚拟机
Mobile Agent System Interoperability Facility	MASIF	移动智能体的标准化草案
Intrusion Detection System	IDS	入侵检测系统
Internet Based Mobile Agent System	IBMAS	基于互联网的移动智能体系统
Norm Model of Mobile Agent System	NM_MAS	移动智能体系统的基准模型
Agent Behavior Function	ABF	智能体行为函数
Agent Communication Language	ACL	智能体通信语言
Agent Transfer Protocol	ATP	智能体传输协议
Structured Operational Semantics	SOS	结构化操作语义
Mobile Agent Based Intrusion Detection System	MABIDS	基于移动智能体的入侵检测系统
Intrusion Detection Agent	IDA	入侵检测智能体
Demilitarized Zone	DMZ	非管制区
Mobile Agent based Information Retrieval System	IRS-Agent	基于移动 Agent 的信息检索系统
Vector Space Model	VSM	矢量空间模型