新一代网络分布处理技术

移动agent及其应用

张云勇 编著
博嘉科技 审

清华大学出版社
http://www.tup.tsinghua.edu.cn

新一代网络分布处理技术

移动 agent 及其应用

张云勇　编著
博嘉科技　审

清 华 大 学 出 版 社

（京）新登字 158 号

内 容 简 介

移动 agent 的概念是 20 世纪 90 年代初由 General Magic 公司在推出商业系统 TeleScript 时提出的。由于其自身优异的特性，移动 agent 技术已成为继 CORBA、EJB 后新一代分布处理的关键技术，并且在很多新兴领域得到广泛的应用。

本书是国内第一本移动 agent 技术及其应用的著作。作者结合多年研究和教学的经验，从分布处理的角度出发，简单扼要地阐述了移动 agent 技术的基本理论和方法。并根据工科院校的特点，着重介绍了移动 agent 平台 Aglet 及以它为基础的协作电子商务系统，内容全面、叙述清楚，可作为研究生和计算机、网络专业本科高年级的教材，也可供工程技术人员自学参考之用。

本书附带光盘中含有移动 agent 平台 Aglet、KQML 开发包 jKQML 和书中的示例程序。

书　　名：移动 agent 及其应用
作　　者：张云勇
出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）
http://www.tup.tsinghua.edu.cn
印 刷 者：北京鑫丰华彩印有限公司
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：17.75　**字数**：416 千字
版　　次：2002 年 1 月第 1 版　2002 年 1 月第 1 次印刷
书　　号：ISBN 7-900637-37-0
印　　数：0001～3000
定　　价：34.00 元

前　言

移动 agent 的概念是 20 世纪 90 年代初由 General Magic 公司在推出商业系统 TeleScript 时提出的。简单地说，移动 agent 是一个能在异构网络中自主地从一台主机迁移到另一台主机，并可与其他 agent 或资源交互的程序，它实际上是 agent 技术与分布式计算技术的混血儿。移动 agent 技术将服务请求 agent 动态地移到服务器端执行，使此 agent 较少依赖网络传输这一中间环节而直接面对要访问的服务器资源，从而避免了大量数据的网络传送，降低了系统对网络带宽的依赖；移动 agent 不需要统一的调度，由用户创建的 agent 可以异步地在不同节点上运行，待任务完成后再将结果传送给用户；为了完成某项任务，用户可以创建多个 agent，同时在一个或若干个节点上运行，形成并行求解的能力；此外它还具有自治性和智能路由等特性。

移动 agent 技术是 CORBA、EJB 后新一代分布处理的关键技术，鉴于其众多的优点，它已经在很多新兴领域得到广泛的应用。

综观全书，本书有如下特点：

入门要求不高　本书介绍了 agent 最基本的知识及 JDBC、Servlet 编程基础，读者只须具有最基本的 Java 编程基础。

完整性好　本书最后采用移动 agent 技术实现了一个完整的电子商务协作信息系统，包括供货、购货、信息查询、信息发布及系统设置等等。附录 2、附录 3 包括了完整的 Aglet API 包和 jKQML 包，这对移动 agent 系统的开发是非常有帮助的。

概括性高　本书每章的标题就是对该章内容的高度概括，随后对它的解释力求简明扼要、准确贴切。

实用性强　本书紧密结合应用，除了重点阐述一个协作信息电子商务系统外，还给出了多领域的应用开发模式，具有很强的实用性。

本书由浅入深、循序渐进。第 1 章到第 5 章为移动 agent 理论及其基础知识介绍，其中很多结论、体系结构凝聚了作者多年研究的心血；第 6 章为一个移动 agent 开发平台 Aglet 编程开发的详细介绍；第 7 章为基于 Aglet 的一个实际系统——电子商务系统的完整介绍，经过这两章的学习，读者可以实际开发自己的系统；第 8 章和第 9 章为锦上添花，介绍了一个配套产品 jKQML 的开发过程，还重点阐述了移动 agent 在其他领域的开发和应用模式，读者可以实际开发这些系统。

本书在编写过程中，得到博嘉科技资讯有限公司王松先生的热情帮助，在此表示感谢。参与本书编写的还有：成都电子科技大学开放系统研究室的张云松、胡健、郭维娜老师，彭建、李振东、唐勇、郭乐深、杨涛、张显风、王建军、王流一、陈小英、顾艳艳、张向刚和电子科技大学国防抗干扰重点实验室的李宋琛、刘大为、陈军、陈翔、蔡学荣老师，上海中兴公司的陈刚，上海贝尔的高风、周世杰、张枫工程师以及张秀琴、蔡桂芳、许明容女士，蔡万和、张锦林、平春林先生等，在此深表感谢。

由于编者水平有限，加之分布处理技术的飞速发展，书中难免存在某些缺点，恳请广大读者和专家批评指正。

如果读者在学习过程中发现问题，或是有更好的建议，欢迎致电探讨。电话：（028）5404228；E-mail:bojiakeji@163.net。通讯地址：成都四川大学（西区）建筑学院成都博嘉科技资讯有限公司；邮编：610065。

作　者

2001 年 8 月

第1章 软件 agent 技术概述

主 要 内 容

- 软件 agent 起源
- 概念
- 特性
- 分类
- 体系结构
- 应用领域
- 标准化

本章介绍软件 agent 的起源、概念、特性、分类、体系结构、应用领域和标准化等内容。通过本章的学习，读者应该知道软件 agent 的产生背景、框架结构及其应用领域。

1.1 软件 agent 起源

进入20世纪90年代以来，在各种计算机文献和众多公司的技术发展规划中，软件 agent 正日渐成为使用频率最高的词汇之一。很多人想了解究竟什么是软件 agent 技术以及它有哪些用处。这里对软件 agent 技术的发展做一个简要的阐述。

智能软件 agent 技术的诞生和发展是人工智能技术（AI）和网络技术发展的必然结果。从 20 世纪 60 年代起，传统的 AI 技术开始致力于对知识表达、推理、机器学习等技术的研究，其主要成果是专家系统。专家系统把专业领域知识与推理有机地组合在一起，为应用程序的智能化提供了一个低级而实用的解决办法。作为人工智能的一个分支，AI 计划理论的研究成果使应用程序有了初步的面向目标和特征，即应用程序具有了某种意义上的主动性；而人工智能的另一个分支——决策理论和方法则使应用程序具有了自主判断和选择行为的能力。人工智能围绕着知识所进行的广泛研究和应用正逐步形成一门新的学科，这就是知识工程，它涉及知识的获取、存储和管理等许多课题。所有这些技术的发展加快了应用程序智能化的进程。

随着网络技术的发展，多个应用程序间相互作用的模式正从单一的集成式系统向分布式系统演化。一个在物理上和地理上分布的应用程序之间通信与合作的网络底层基础结构正逐渐建立起来。分布式对象技术（如 CORBA 或 DCOM 技术）则进一步使分布且异构的应用程序之间能以一种共同的方式提供和获得服务，实现了在分布式状态下的“软”集成。

智能化和网络化的发展促成了软件 agent 技术的发展，软件 agent 技术正是为解决复杂、动态、分布式智能应用而提供的一种新的计算手段。许多专家信心十足地称：软件 agent 技术将成为 21 世纪软件技术发展的又一次革命。

1.2 概念、特性及分类

1.2.1 软件 agent 的定义

agent 一词直译为“代理”，也有人把它翻译为“智能代理”，广义上它是指具有智能的任何实体，包括人类、智能硬件（如机器人）和智能软件。

agent 思想的诞生可归功于 John McCarthy 在 20 世纪 50 年代末提出的“The Advice Taker”系统，该系统被设想为具有目标性，系统内实体间用人类的术语进行交流，它们从用户利益来考虑从事各种任务。到目前为止，许多研究者提出了各自对 agent 的定义，但至今没有被一个大多数人认可的统一的 agent 定义，不同研究领域的学者考虑各自领域的

技术特点。对 agent 给出的典型定义大致有两种。

- agent 是驻留于环境中的实体，它可以解释从环境中获得的反映环境中所发生事件的数据，并执行对环境产生影响的行为。

这一定义出自 FIPA（Foundation for Intelligent Physical Agent），它是一个致力于 agent 技术标准化的组织。在这个定义中，agent 被看做是一种在环境中“生存”的实体，它既可以是硬件（如机器人），也可以是软件。

而软件 agent 的研究者则对 agent 进行了如下定义：

- 智能软件 agent 是能为用户执行特定的任务、具有一定程度的智能以允许自主执行部分任务并以一种合适的方式与环境相互作用的软件程序。

在更多的应用领域中，研究者把凡是具有智能行为和交互特征的分布式实体都称为 agent。在这种假设下，人类、机器人、智能嵌入式设备、计算机或智能软件程序都可以是 agent。而对其他一些研究者来说，agent 只代表具有自主性的分布式智能软件。

1.2.2　软件 agent 的特性

从 agent 的定义可以知道，agent 首先是智能的，它应对环境有响应性、自主性和主动性等；同时，agent 是具有社会性的。智能软件 agent 的属性如图 1-1 所示。下面对这些属性进行解释。

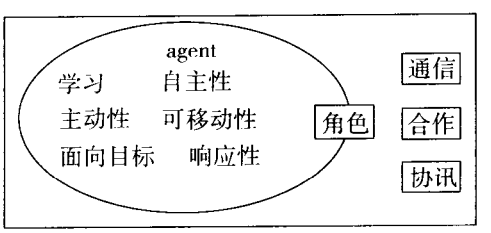

图 1-1　智能 agent 的属性

自主性（Autonomy）：一个 agent 能在没有与环境的相互作用或来自环境的命令的情况下自主执行任务。这是 agent 区别于普通软件程序的基本属性。

响应性（Reactivity）：agent 必须对来自环境的影响和信息做出适当的响应。

主动性/面向目标（Proactivity/Goal oriented）：agent 不仅对环境变化做出反应，而且在特定情况下采取主动行动，这种自身采取主动的能力需要 agent 有严格定义的目标。

推理/学习/自适应能力（Learning/Adaptation）：agent 的智能由三个主要部件来完成，即内部知识库、学习或自适应能力以及基于知识库内容的推理能力。

可移动性（Mobility）：一个 agent 在计算机网络中漫游的能力。

角色（Character）：agent 在社会活动中对安全性、风险、信任、诚实等因素的考虑。

通信/合作/协调（Communication/Cooperation/Coordination）：这是在 agent 群体中应具有的社会属性。

1.2.3 软件 agent 的分类

由于 agent 的定义不统一，各个领域的研究者都把具有某些 agent 属性的研究对象称为某类 agent。因此，为明确这些 agent 的具体含义，必须根据 agent 的不同功能和特性对 agent 进行分类。本节对这些 agent 进行归类总结，并解释它们的具体含义。

agent 可分为人类 agent、硬件 agent 和软件 agent，如图 1-2 所示，本文研究的重点是软件 agent。图 1-3 中列出了在文献中经常提到的一些 agent 类型。

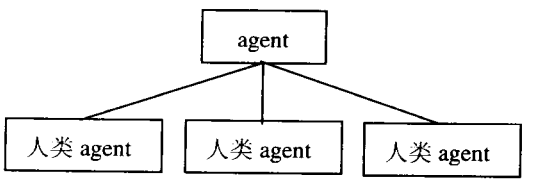

图 1-2 一般 agent 分类

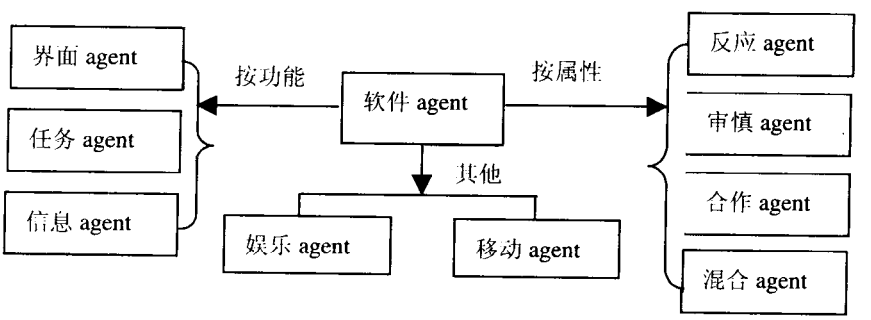

图 1-3 软件 agent 分类

- 按功能划分，有如下几类。

信息/Internet agent（Information/Internet agent）：它支持用户在分布式系统或 Internet 网络中智能搜索信息或智能管理网络资源。

界面 agent（Interface agent）或个人助手：它的主要任务是协助用户完成乏味而重复性的工作。agent 将观察并监督用户怎样执行特定的任务，当这些 agent 能确定用户在特定情况下将如何反应时，它就开始替代或帮助用户完成任务。这些 agent 已针对某一用户进行了个性化处理，适应于特定用户的行为。这些问题与人机接口（HCI）、用户建模和模式匹配密切相关。

任务 agent（Task agent）：它是帮助人类进行复杂决策和其他知识处理的软件 agent。这些 agent 以 AI 领域的机器学习、计划、资源受限的推理、知识表达等为基础在一实用框架中应用。

- 按属性划分，有如下几类。

反应 agent（reactive agent）：具备对当时处境的实时反应能力的 agent。

审慎 agent（deliberative agent）：在目标指导下具备自主行动能力的 agent。

合作 agent（interactive agent）：具备社会合作能力的 agent。

混合 agent（hybrid agent）：具有实时反应、目标指导下自主行动及合作等综合能力的 agent。

- 按行为方式划分，有如下几类。

自主 agent（Autonomous agent）：在复杂动态环境中自主感知和行动。

多重 agent（Multiagent）：一个 agent 能利用关于其他 agent 的知识来协调它与其他 agent 的行动或合作完成目标。

助手 agent (Assistant agent)：只与人类 agent 相互作用。

- 其他一些特殊类型的 agent。

移动 agent（mobile agent）：位于网络中并通过迁移或服务接口能与网络中其他程序进行通信的 agent。它通常是由客户端迁移到服务器端执行的脚本程序。

可信 agent（believable agent）：它是在与人的相互作用（如在一个计算机游戏）中以“令人信任”的特征来执行，它需要处理与人的相互作用中发生的各种情况，而不是局限于把少量事情做得特别好。典型例子有教育、娱乐 agent 等。

1.3　典型软件 agent 的体系结构

软件 agent 的体系结构描述 agent 的功能模块和这些功能模块一起动作的方式。根据软件 agent 的特性，软件 agent 的体系结构包含以下几个基本模块：

- 用户界面用以接受用户信息输入或输出信息给用户。
- 通信接口用来与其他软件 agent 或应用进行通信。
- 感知模块对输入信息进行过滤与分类。
- 推理模块根据 agent 自身知识对信息进行推理。
- 决策模块对推理结果进行评价和决策。
- 计划模块根据决策制订行动计划。
- 执行模块按照计划执行动作。
- 知识库对推理、决策、计划等提供支持。

1.4 软件 agent 的适用领域

在对 agent 的定义、特性和分类进行了探讨之后，必须知道目前哪些研究和应用领域能够在应用程序中采用 agent 技术。下面列举了一些软件 agent 技术的适应领域：

- 与用户有灵活的相互作用，在互相作用中智能地协助用户完成琐碎的工作。
- 在对海量分布式信息搜索中，建立快速智能的搜索机制。
- 在高度动态的环境下，要求应用程序能对多变的环境作出响应或自适应。
- 需要应用程序能自主处理失效或冲突，以进行再调度、再计划或资源再分配。
- 需要应用程序既能进行长期计划驱动的行为，又能从事短期实时响应行为。
- 在复杂的或安全性很重要的应用程序中，保证适宜的反映和应答时间。
- 在地理上或逻辑上分布、自主或异构的节点间提供应用服务或中间件服务。
- 在不完全信息下的复杂或分散的资源分配问题。

1.5 软件 agent 技术的标准化

软件 agent 技术的标准化工作非常重要，因为标准的实现将使采用这项技术的人在对 agent 技术应该具备什么样的功能，怎样提供这些功能等方面达成一致，并且保证异构 agent 的可交互性。FIPA （Foundation for Intelligent Physical Agents）是一个由 IBM、NHK、BT 等公司和政府、学术机构组成的权威的 agent 标准化组织（http:/www.fipa.org），目前该组织正致力于以下三个主要领域的标准制定：

- agent 管理需要认同和发现 agent（白、黄页服务），需要定义它们的各种状态以及哪些角色能与它们相互作用。
- agent 相互作用覆盖最高层 agent 间相互作用的标准，包括 agent 间传递的信息的意义、命令、请求、义务等。
- agent 与软件的接口。

此外，FIPA 还制定了 4 个参考应用领域的标准，包括个人旅行助手、个人助手、声/视娱乐和广播、网络管理等。正在制订的标准包括人类 agent 的相互作用、产品设计与制造 agent、agent 安全管理、支持移动性的 agent 管理、Ontology（共享语汇）服务、agent 消息传送、agent 命名、内容语言库等。尽管 FIPA 标准仍在制订和发展之中，但随着越来越多组织的加入，它必将成为促进软件 agent 应用和发展的主要推动力。

第2章 移动 agent 技术

主 要 内 容

- 移动 agent 起源
- 移动 agent 概念
- 系统结构
- 关键技术
- 安全机制

本章回顾了移动 agent 的历史及背景，从系统结构、关键技术、安全等角度阐述了移动 agent 技术。在介绍了一些移动 agent 组织和规范，评价了典型的移动 agent 开发工具后，对常见移动 agent 应用及基于移动 agent 的系统开发进行了讨论，最后进行了小结和展望。

2.1 移动 agent 简介

agent 的研究起源于人工智能领域。agent 是指模拟人类行为与关系、具有一定智能并能够自主运行和提供相应服务的程序。与现在流行的软件实体（如对象、构件）相比，agent 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，完成某些功能，这就是移动 agent（Mobile agent）的思想。

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。简单地说，移动 agent 是一个能在异构网络中自主地从一台主机迁移到另一台主机，并可与其他 agent 或资源交互的程序，实际上它是 agent 技术与分布式计算技术的混血儿。传统的 RPC 客户和服务器间的交互需要连续的通信支持；而移动 agent 可以迁移到服务器上，与之进行本地高速通信，这种本地通信不再占用网络资源。移动 agent 迁移的内容既包括其代码也包括其运行状态。运行状态可分为执行状态和数据状态：执行状态主要指移动 agent 当前运行时状态，如程序计数器、运行栈内容等；数据状态主要指与移动 agent 运行有关的数据堆的内容。按所迁移的运行状态的内容，移动 agent 的迁移可以分为强迁移和弱迁移。强迁移同时迁移移动 agent 的执行状态和数据状态，但这种迁移的实现较为复杂；弱迁移只迁移移动 agent 的数据状态，其速度较强迁移快，但不能保存移动 agent 的完整运行状态。

移动 agent 不同于远程执行，移动 agent 能够不断地从一个网络位置移动到另一个位置，能够根据自己的选择进行移动。移动 agent 不同于进程迁移，一般来说进程迁移系统不允许进程选择什么时候和迁移到哪里，而移动 agent 带有状态，所以可根据应用的需要在任意时刻移动，可移动到它想去的任何地方。移动 agent 也不同于 Applet，Applet 只能从服务器向客户单方向移动，而移动 agent 可以在客户和服务器之间双向移动。

移动 agent 具有很多优点，移动 agent 技术通过将服务请求 agent 动态地移到服务器端执行，使得此 agent 较少依赖网络传输这一中间环节而直接面对要访问的服务器资源，从而避免了大量数据的网络传送，降低了系统对网络带宽的依赖。移动 agent 不需要统一的调度，由用户创建的 agent 可以异步地在不同节点上运行，待任务完成后再将结果传送给用户。为了完成某项任务，用户可以创建多个 agent，同时在一个或若干个节点上运行，形成并行求解的能力。此外它还具有自治性和智能路由等特性。

2.2　移动 agent 系统结构及其关键技术

2.2.1　移动 agent 系统结构

移动 agent 系统由移动 agent 和移动 agent 服务设施（或称移动 agent 服务器）两部分组成。移动 agent 服务设施基于 agent 传输协议（agent Transfer Protocol）实现 agent 在主机间的转移，并为其分配执行环境和服务接口。agent 在服务设施中执行，通过 agent 通信语言 ACL（agent Communication Language）相互通信并访问服务设施提供的服务。

如图 2-1 所示，移动 agent 体系结构可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。体系结构的最外层为安全代理，它是 agent 与外界环境通信的中介，执行 agent 的安全策略，阻止外界环境对 agent 的非法访问。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 ACL 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调，而通信内容的语义与 ACL 无关。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。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构。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。约束条件是 agent 创建者为保证 agent 的行为和性能而作出的约束，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。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以满足复杂和非确定性任务的求解。

服务设施为移动 agent 提供基本服务（包括创建、传输、执行等），移动 agent 的移动和任务求解能力很大程度上决定于服务设施所提供的服务。一般来讲，服务设施应包括以下基本服务。

- 生命周期服务：实现 agent 的创建、移动、持久化存储和执行环境分配。
- 事件服务：包括 agent 传输协议和 agent 通信协议，实现 agent 间的事件传递。
- 目录服务：提供定位 agent 的信息，形成路由选择。
- 安全服务：提供安全的 agent 执行环境。
- 应用服务：是任务相关的服务，在生命周期服务的基础上提供面向特定任务的服务接口。

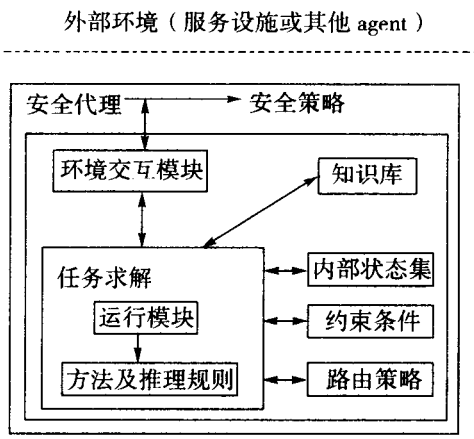

图 2-1　移动 agent 的结构模型

2.2.2　关键技术

移动 agent 利用先进的思想提供智能化的服务和任务规划求解，为实现这个目标，必须解决好几个关键技术。

1. 移动 agent 理论模型

目前一般基于 BDI 系统。BDI 系统也称为意识系统，是把 agent 看做理性主体，通过信念（Belief）、愿望（Desire）或意图（Intention）属性来预测 agent 的行为。把主体看做意识系统的主要好处有：

- 对于设计者和分析者来说，这样是自然的。
- 对于描述复杂系统的行为提供了简洁的表示，有利于理解和解释。
- 不依赖于具体物理实现就可以得到许多主体的规则和模式。
- 可被 agent 自身用来相互推理。

目前大多数学者利用模态逻辑理论来研究 BDI 理论模型，并得出了一些有益的结论。

2. agent 通信语言 ACL

ACL 基于语言——行为理论（speech act），定义了 agent 及服务设施间协商过程的语法和语义。移动 agent 的 ACL 应具有应用的普遍性、简捷一致的语法和语义、通信内容的独立性等。目前常用的 ACL 有 KQML（Knowledge Query and Manipulation Language）和 FIPA（Foundation of Intelligent Physical Agent）ACL，它们的格式非常接近，这里只讨论 KQML。

KQML 被分为三层：内容层、消息层和通信层。内容层包含消息的实际内容，KQML 可以携带任何语言表达的内容，包括表达为 ASCII 码或二进制代码的语言。通信层描述低级的通信参量，如发送者、接收者和与通信有关的惟一标识符。消息层是 KQML 语言的核心，它的主要作用是识别传输消息所采用的网络协议，给出发送者对内容的态度或意图，即语言行为或原语（performatives）。行为原语定义了可作用于 agent 的知识库和目标库（KQML 支持基于 BDI 的 agent 模型）的各种许可的操作，常用的原语有基本操作原语（Tell、Deny）、基于知识数据库的操作原语（Insert、Delete）、基本响应原语（Error、Sorry）、基本查询原语（Evaluate、Reply、Ask-If）、能力宣告原语（Advertise）、网络操作原语（Register、Forward）和协调器操作原语（Broker-One）等，开发者可以自己扩充 KQML 来实现特定的系统。

目前虽然出现了一些 KQML 的开发包（如 KAPI、Jkqml 等），但 KQML 及其类似语言还有一定的局限性，即缺少一个精确的语义系统，通信级上基本无语义。

3. agent 传输协议

agent 传输协议定义了移动 agent 传输的语法和语义，具体实现了移动 agent 在服务设施间的移动机制。IBM 提出的 ATP 框架结构（ATP framework）定义了一组原语性的接口和基础消息集，可以看做是一个 agent 传输协议的最小实现，其基本操作如图 2-2 所示。目前研究的重点是可靠而实时的传输。

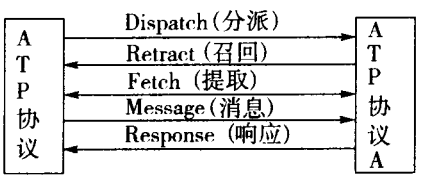

图 2-2　ATP 的示意图

4. 路由策略

移动 agent 的效率很大程度上决定于路由策略的优化。可行的路由策略有两种，分别为固定路由和基于规则及目录服务的动态路由。目前，在路由策略中引入 QoS（Quality of Service）是一个研究重点。

5. 系统性能影响及其测试工具

当前的移动 agent 系统虽然可以减少网络负载和克服网络延迟，但却增加了服务方主机的负载，基于可移植性和安全方面的原因，agent 通常都是采用相对较慢的解释性语言，并且当到达目的地后，必须置入相应的运行环境中才能执行。这些原因使得移动 agent 的执行速度低于普通程序。所幸的是，以 Java 为代表的即时编译（Just-in-time compilation）

取得了很大的进步，使移动代码的执行速度显著提高。

性能测试工具方面的研究目前还很不成熟，基本上没有很好的测试工具。实际评估性能时，一般利用现有的理论如随机 Petri 网（把系统协议用随机 Petri 网形式化描述，然后等价成 Markov 链，利用 Markov 更新方程或者随机 Petri 网工具分析出系统性能）、着色 Petri 网等。

6. 容错策略

移动 agent 系统必须考虑到移动过程中可能存在网络故障、服务设施故障、长时间停机等情况造成的移动 agent 破坏和失败。常见的容错策略有以下几种。

- 创建相同任务的多个备份：它们在网络中独立运行，任务结束后比较结果。
- 集中式容错：特定的服务器保留移动 agent 的原始备份并实施跟踪，通过重发原始备份恢复失效的移动 agent。
- 分布式容错：将容错责任分配到网络中多个非固定的站点进行。

这些策略都离不开检查点（check point）。何时以及如何建立检查点是关键问题，一种简单的方法是把移动 agent 的挂起/继续运行点作为检查点。另外也可利用 Markov 链模型来计算检查点。容错机制是移动 agent 系统服务质量的重要评价标准，也是移动 agent 优势得以体现的重要手段。

7. 互操作性

目前，市场上移动 agent 系统非常多，随着移动 agent 在智能领域中的应用，有必要在 MASIF 规范中体现智能性，实现与符合 FIPA 规范的智能 agent 系统的互操作。MASIF 和 FIPA 的兼容，以及基于语义（底层）的互操作性是将来研究的重点。

8. 控制策略

必须对移动实施有效的控制，避免移动 agent 失控（如不停地复制、迁移等）。另外，为了保证性能，引入负载均衡的机制很有必要。

2.3 移动 agent 的安全性

现有的基于 Java 的移动 agent 系统，基本上都采用了 Java 的沙箱（sand box）安全模型作为其安全机制的实现基础，但是 Java 安全模型本身就存在不完善的地方，而且移动 agent 系统对安全性有着特殊的要求，系统地进行移动 agent 系统安全性研究，有着重要的意义。