第 26 卷第 12 期 电子与信息学报 Vol.26No.12
2004 年 12 月 Journal of Electronics & Information Technology Dec. 2004

MAS 系统中迁移机制的研究与实现 [1]

王汝传 * ** 穆 鸿 * 徐小龙 *

*(南京邮电学院计算机科学与技术系 南京 210003)
**(南京大学计算机软件新技术国家重点实验室 南京 210093)

摘 要: 该文分析了现有移动代理系统的几种代表性的迁移技术，阐述了在作者设计的移动代理系统 (Mobile Agent System, MAS) 中代理迁移机制的具体实现，着重解决了代理的动态路由和与迁移相关的安全要素的改变，并提出了一种扩展的代理迁移机制，最后给出了应用实例分析.

关键词: 移动代理，迁移，远程方法调用
中图分类号: TP31 **文献标识码**: A **文章编号**: 1009-5896(2004)12-2006-07

Research and Implementation of Migration Mechanism in MAS

Wang Ru-chuan* ** Mu Hong* Xu Xiao-long*

*(*Dept of Computer Science and Technology, NJUPT, Nanjing* 210003, *China*)

**(*State Key Laboratory for Novel Software Tech., Nanjing Univ., Nanjing* 210093, *China*)

Abstract This paper surveys some typical approaches to migration models used in Mobile Agent System(MAS) and describes the migration mechanism in MAS which is designed and implemented by authors. Focused on the issues about the agent's dynamic routing and migration-related security changing, an extended migration mechanism of agent is proposed and an application example is given.

Key words Mobile agent, Migration, Remote method invocation

1 引言

随着 Internet/Intranet/Extranet 的迅速发展，网络的开放性、共享性和互联程度不断扩大， WWW 被广泛用作信息发布、电子商务和各种娱乐的业务平台，对应于此，出现了许多新的网络技术，移动代理 [1,2] 技术便是其中的一种.

从广义上讲，移动代理可以是任何代表某个用户的程序段，它可以在计算机网络中漫游，代表用户在不同的节点上进行交互工作。它的应用范畴包括在线购物 [3] 、设备的实时控制以及分布式科学计算等。在具体实现中，用户或代理控制中心向网络发出一个 (或多个) 移动代理，该代理按照特定的路线 (用户预定义，或由它自己确定) 在网络中漫游并与各服务器节点交互，完成任务后携带计算结果返回 (初始用户或代理控制中心) 。

移动代理拥有很多优点，如减少网络流量、增加客户机和服务器的异步性、便于负载均衡和容错、支持移动客户和服务定制等。它在通信网络管理、智能网领域、 Internet 上的智能信息检索以及分布计算等领域有广泛应用前景。

迁移机制是移动代理的核心技术之一，受到学术界和工业界的广泛关注 [4] ，虽然也出现了为数不多的采用不同迁移机制的移动代理系统，但在迁移机制中有自身缺陷，如 Telescript 基于

[1] 2003-07-12 收到， 2004-03-09 改回
国家自然科学基金 (6017303 和 70271050), 江苏省自然科学基金 (BK2003105 和 BK2004218), 江苏省高技术研究计划 (BG2004004) 和江苏省计算机信息处理重点实验室基金 (kjs03061 和 kjs04) 资助课题

进程迁移理论所采用的语句级迁移模型，系统负担重，网络传输量大，尤其是 Java， C++ 等语言并不支持线程状态捕捉，而 IBM 的 Aglets 系统设计采用事件驱动的方式实现语句级对象迁移模型，虽系统负担较低，但却增加了用户的负担，在以上两种移动代理系统中，迁移信息是隐含在代理的代码中的，这在程序的设计调试中都带来了新问题，因为对象在移动过程中，对迁移信息的改变是难以持久化的。事实上，我们可以用一个独立的结构来描述对象迁移信息，这样迁移信息因为独立于任务体所以更易控制，甚至我们可以在对象未实例化之前来控制代理的行为。

在研制的移动代理系统中，我们具体描述了这种独立迁移结构的实现，着重分析了这个机制对动态路由的影响并且考虑了对一些安全措施的影响。更进一步，我们采用任务和控制分离的思想，提出了一种新的代理迁移机制．其主要特点是： (1) 提出了代理访问控制列表 (Agent Access Control List, AACL) 借以控制路由策略和安全； (2) 可持久化保存数据。

2 MAS 中迁移机制的实现

MAS 是我们自行设计的移动代理系统，它采用纯 Java[5] 语言开发，基于 RMI[5,6] (Remote Method Invocation) 通讯机制，采用策略文件、数字签名及对称加密技术等来保障代理应用的安全，在这一部分我们将讨论该系统中的迁移机制实现。

2.1 代理传输机制

在 MAS 中，低层代理传输是通过 Java 语言特有的 RMI 机制实现的。 RMI 特性使得用户能够非常方便地访问位于另一主机上的 Java 对象．本地对象是客户机，远程对象是服务器，同时远程对象也可以是另一个远程服务器对象的客户．客户机可以像调用本地对象一样调用远程对象，这种方式允许 Java 程序利用分布式计算将工作量分散到多个 Java 虚拟机 (Java Virtual Machine, JVM) 上，并且使分布式环境下的 Java 编程不再涉及底层协议及数据的编码解码。

RMI 能提供对象间的双向通信，它能将完整的对象从一个地址空间发送到另一地址空间，完整的类定义可以用序列化和 RMIClassLoader 来传递．对象序行化扩展了 Java 的输入 / 输出类，允许通过网络连接发送 Java 对象．该扩展对 Java 下开发移动代理具有重要意义．对象串行化包 (Object serialization packet) 的功能包括：将 Java 对象所属类 (包含对象可访问类) 打包成串行化数据流，经由网络传送这些数据流，在目标机上重组织成原始对象。因此，对象串行化可以支持持久对象．一个对象可以在执行时被中断，经打包、传输，在目标机上重组织并从程序中断位置恢复执行．对象序列化将数据转换为字节流，然后通过 TCP/IP 发送， RMI 使用的端口为大于 1024 的任意端口。

Java 程序进行远程方法调用时，方法参数发送到服务器中，返回值回送到客户机上。远程方法的参数和返回值可以是基本类型的数值 (比如浮点型)，也可以是对象，但此对象必须实现 Serializable 或 Externalizable 接口。非常重要的一点是， RMI 中包括分布式垃圾收集，它能安全地收集远程对象以释放服务器上的资源。

利用 Java 语言的 RMI 机制我们可以很方便地实现代理移动.在 MAS 中,所有的 MAE(Mobile Agent Environment) 都必须在它底层主机的 RMI 注册表中进行注册。代理主人可以向远程主机请求 MAE 对象，获得对象引用后，通过调用其 receiveAgent 和 returnAgent 方法，就能派出或召回代理。

2.2 代理迁移算法描述

在本算法中，遵循迁移结构分离的原则，我们定义一个移动代理实体 (Entity) 为一个二元组 {MAgent, Itinerary} 。下面所提的代理如不说明是指 MAgent 部分。其中 MAgent 是一个对象，包括了代理的任务体，该对象提供适当方法提供代理的具体任务的运行代码，但它不包括 Itinerary 相关部分。 Itinerary 是矢量类，负责代理迁移控制，其构造形如： “10.10.9.160”, “false” 。

代理迁移的算法描述如下：

(1) 构造 Itinerary 实例.

(2) 找第一个可用的目标地址:

```
i=0
find: try{MAE=Naming.Lookup(itinerary.get(i)/MAE)
}catch(RemoteException ex)
{i++;
 if (i<=itinerary.length) goto find}
if (i>itinerary.length) exit 并给出提示: "Can't Execute Task!"
```

(3) 用私钥签名 MAgent .

(4) 调用 MAE 的 receiveAgent( ) 方法: receiveAgent 方法为: receiveAgent(SignedObject obj, Itinerary itinerary, String lastAccessAddress) .

(5) 验证已签过名的 MAgent : 从 lastAccessAddress 来判断用谁的公钥验证, 我们可以查询当前的证书库找到对应公钥也可以人为指定证书路径.

(6) 运行 MAgent.run( ) 来执行任务, 修改 itinerary 标记本机为访问过, 我们调用方法 setAccessed( ), 我们不给出地址参数防止恶意修改标记访问的值. 伪代码为: itinerary.setAccessed( ){ 操作的是 InetAddress.getHost( ).getHostAddress( )} .

(7) 寻找下一个可访问的主机: NextEnv=Naming.Lookup(itinerary.get(i)/MAE) .

(8) 用私钥签名 MAgent .

(9) 调用可用机器的 receiveAgent("","","") .

如果不存在可用的下一台主机, 可以等待或者停止执行, 在下一台主机上执行 5 及以下.

(10) 终止.

2.3 动态路由安全机制

代理在迁移中的路由分为两种情形: 静态路由和动态路由. 静态路由是初始化时设定的, 它是一系列主机地址顺序列表, 在移动过程中不能动态修改. 这样虽然能降低设计的难度, 但同时也增加了不确定因素. 如在主机访问列表中的某个主机可能暂时无法访问, 如果没有有效的措施则代理将会停止迁移. 而动态路由是指代理在移动过程中可以动态的修改地址列表, 为了安全性的考虑, 我们把对路由表的修改控制在一个允许的范围内, 即根据条件将地址设定为可以访问, 而不允许动态添加或删除主机地址. 两种路由的方式如图 1 所示. 引入动态路由机制虽然带来一些安全问题 [7−10], 但是我们可以通过路由域加以解决.

在 MAS 中我们使用了数字签名 [11] 技术, 在代理发送到目的地之前要对代理进行签名处理, 使用的是自己的私钥, 接受主机在收到代理之后要对代理进行验证, 这时他必须知道使用谁的公钥对代理验证, 我们在发送代理后将地址同时发送过去了, 这样就可以知道如何验证了. 这点在代理迁移算法中已有叙述. 下面着重讨论基于共享密钥的分域概念.

在基于加密套接字协议层 (Security Socket Layer, SSL) 的 RMI 构造中, 我们若完全考虑通信信道的动态性, 即实现完全的端到端安全, 每台主机需要 itinerary.length-1 个 stub , 这显然不切实际, 考虑到我们的目的是建立通道防止非参与主机的攻击, 采用共同密钥系是可行的, 我们称这对密钥系为一个域的密钥系, 这样就可实现基于域的安全了. 我们可以设置一个域边界主机, 若一个代理要出域, 则它首先与两个域的边界主机通信, 再经由该域边界主机将代理传送到目的域. 我们定义一个共享秘钥域是这样一个域, 其中的主机共享同一秘钥系 (公钥、私钥) , 基于域的代理迁移机制, 如图 2 所示.

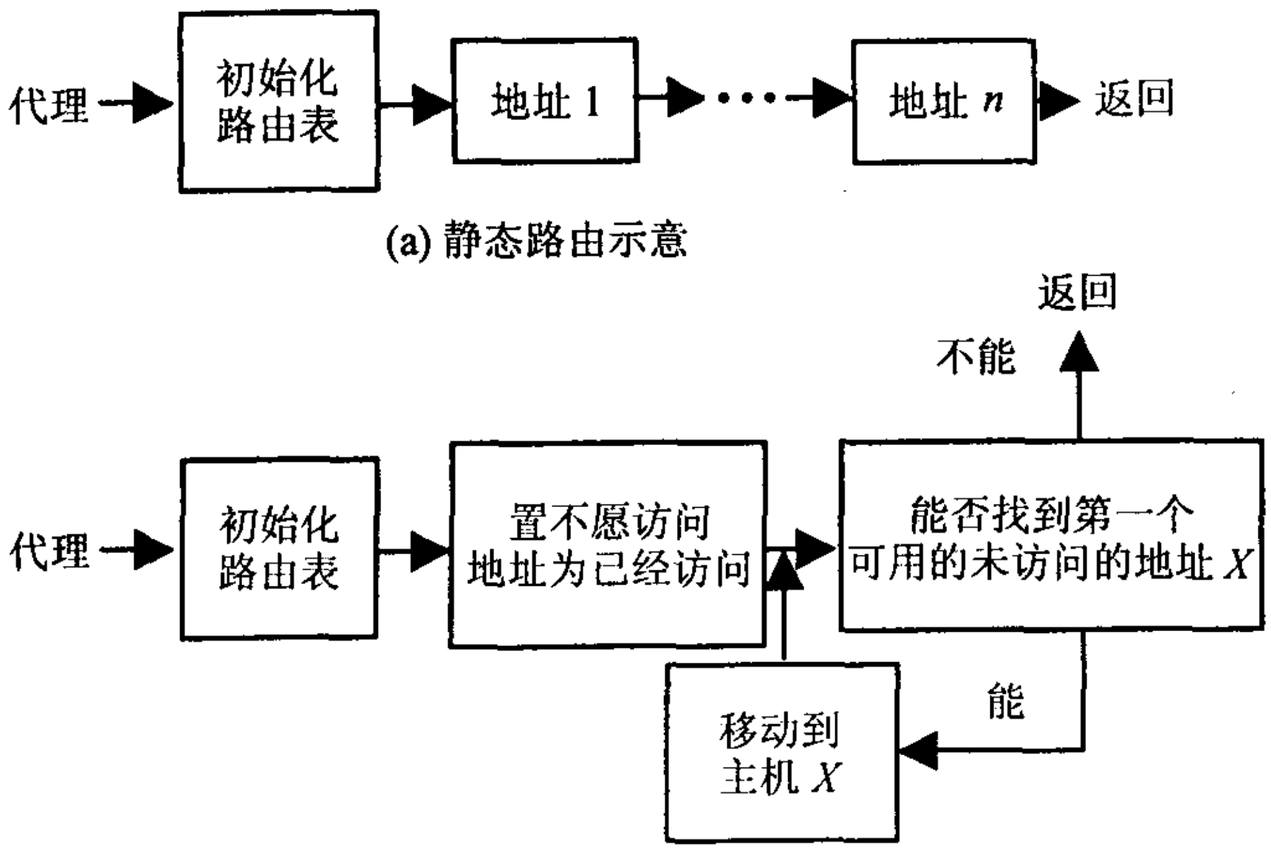

图 1　路由示意图

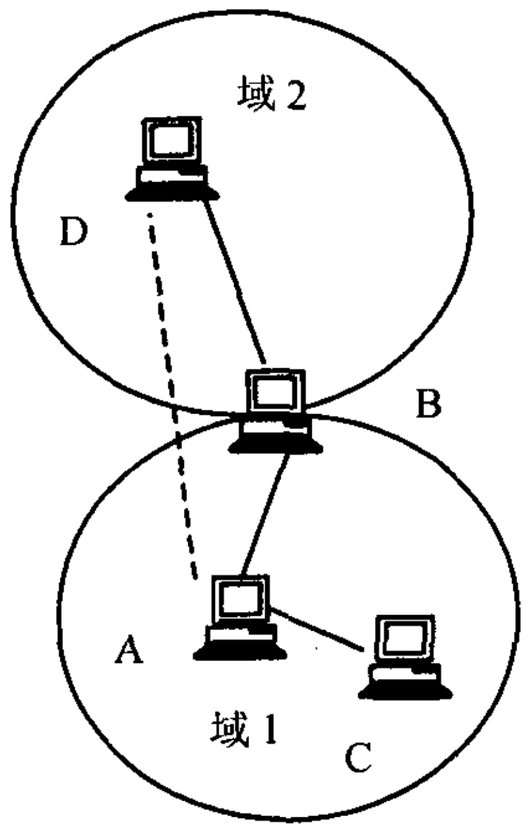

图 2　域路由示意图

如果代理从 A 到 C, 则没问题，通信机制如前所述. 如果代理从 A 到 D，因为 A 和 D 不共享密钥系，所以直接建立端与端的安全是不可以的，如图中虚线所示. 为了从 A 到 D，我们先让代理移动到 B，因为 A 和 B 共享域 1 的密钥系，显然是可以实现的，同时 B 也属于域 2，它和 D 共享域 2 的密钥系，所以代理可以经由 B 到达 D 了。对于 B 这种主机我们称为域边界主机 (Domain border host)。

事实上，我们对迁移算法已做了适当的改变，增加了一个判断修正模块即判断是否要跨域访问，这样就可以实现分域访问功能了。

3 一个扩展的代理迁移机制

我们将进一步讨论扩展的移动代理迁移机制. 我们知道代理的移动大致可以分为三个部分：代码、数据态、执行态. 代码的移动较容易实现，仅有代码的移动被称作弱迁移。数据态的移动是如何保存数据的问题，因为通常数据作为对象的一部分，则当对象消失，数据也就消失了，除非你实现了某种持久化 (如：写到文件或数据库中)。发起方可以采取发送消息的方式，在对象的生命周期中将得到的数据发送回发起方. 我们再看执行态，执行态是最难移动的，一个基于 Java 的实现可能是从虚拟机得到运行时数据，等代理到达目的地再取出参数启动 JVM 执行代理. 基于以上分析我们提出扩展的代理迁移机制。

我们定义一个移动代理实体 (Entity), 它是一个二元组：{TaskPart, CtrlPart} 其中 TaskPart 等同于 MAgent 而 CtrlPart 为代理的控制部分， CtrlPart 控制代理的路由，它的组成如表 1 所示。

对访问控制结构我们定义 AACL 机制来控制代理的路由和安全。AACL 是基于 IP 的访问控制策略，负责赋予代理用户的更多选择和控制路由权利。我们定义如下的访问控制命令，如表 2 所示。

表 1　CtrlPart 的组成部分

Itinerary		路由表及定义在路由表上的操作
AccessControlStructure		访问控制结构用于判断主机是否可以访问等
DataStoreStructure	ResultRelated	结果相关，用于必要时保存任务处理结果
	ExecRelated	执行相关，用于保存执行相关数据以在需要时在另一处继续程序的运行

表 2 AACL 命令及说明

命令类型	语法	用法说明
设置命令	set route seq (顺序) |default (缺省) |priority-based (基于优先权) |method-base (基于方法) set priority < 0 − 255 > set if <<condition>> then <<method>> | exit |else gonext |exit	设置路由方式，顺序方式严格按照路由序列移动代理是最简单的方式，当序列中一台主机不可访问将等待. 缺省方式是可动态路由方式. 基于优先权方式是按照指定的优先权进行路由的方式，缺省情况下所有未明确指定优先权的主机优先权为 5, 命令 set priority 在设定基于优先权的情形下有效. 基于方法的路由控制是更细粒的路由控制体制，它可以查看代理体中某个参数或方法的返回值来判断下一步的行为如是否执行另一方法或者退出.
控制命令	allow|deny network|host << ipaddress>> << mask >>	控制命令将一个或多个地址剔除出路由表
测试命令	debug route	将实际的路由信息记录并在代理迁移结束后返回给代理的发起方，依照需要还可以返回一些额外的路由信息.

引入 AACL 不但增强了代理迁移的控制能力，同时也增加了系统的安全措施，我们无需修改源代码就可以对路由和安全策略进行设置，从而把控制放在配置时而不是设计时，所以系统的弹性是很大的。例如，我们在一个具有稳定网络环境的情形下，可以使用顺序访问的方式以提高访问速度，相对地，在一个不稳定的环境下，顺序访问可能更加耗时，而使用动态方式可以取得较好的效果。使用 AACL 要求移动代理系统引入一个解释模块用来将这些控制命令翻译成一定的方法，但因为这些方法是独立于应用的，所以一个通用的模块是不难实现的。

数据存储结构是一个简单的字符串结构数组，我们统一用字符串来描述结果信息和执行信息，这样可以降低系统的复杂程度。数据存储结构事实上要保存两方面的信息，一个是中间或最后需要的结果数据，另一个是执行状态信息。保存在 ResultRelated 数据结构中的数据将随代理一起传送到下一台主机，这样就可以保存中间和结果数据，当然有必要说明，如果可以通过消息发送的方式将结果信息实时反馈给用户可能没有必要保存中间结果了，所以要视应用而定。执行态的保留是最难实现的，特别是我们很难获得代码执行状态信息，这要求得到相关的 JVM 的支持，所以在实际应用中完全的执行迁移比较难于实现。退而求其次，我们可以实现方法体级的执行中继，即在 ExecRelated 中保存方法执行选择字。在文献 [4] 中对这种方法进行了详细描述. 补充说明的是，字符串编码是基于 UTF-8 ，所以对不同的操作系统要得到正确的数据和参数必须进行编码转换。在分别考查了 CtrlPart 的各个部分的具体内容后，我们对 MAS 进行改进，得到移动代理系统结构如图 3 所示。

MAgent 在传输之前，先构造控制体部分，然后将组装后的代理实体传输到移动代理平台，经过安全模块的检查后， MAgent 部分被交给实例化器，由实例化器来判断类的位置再经由类

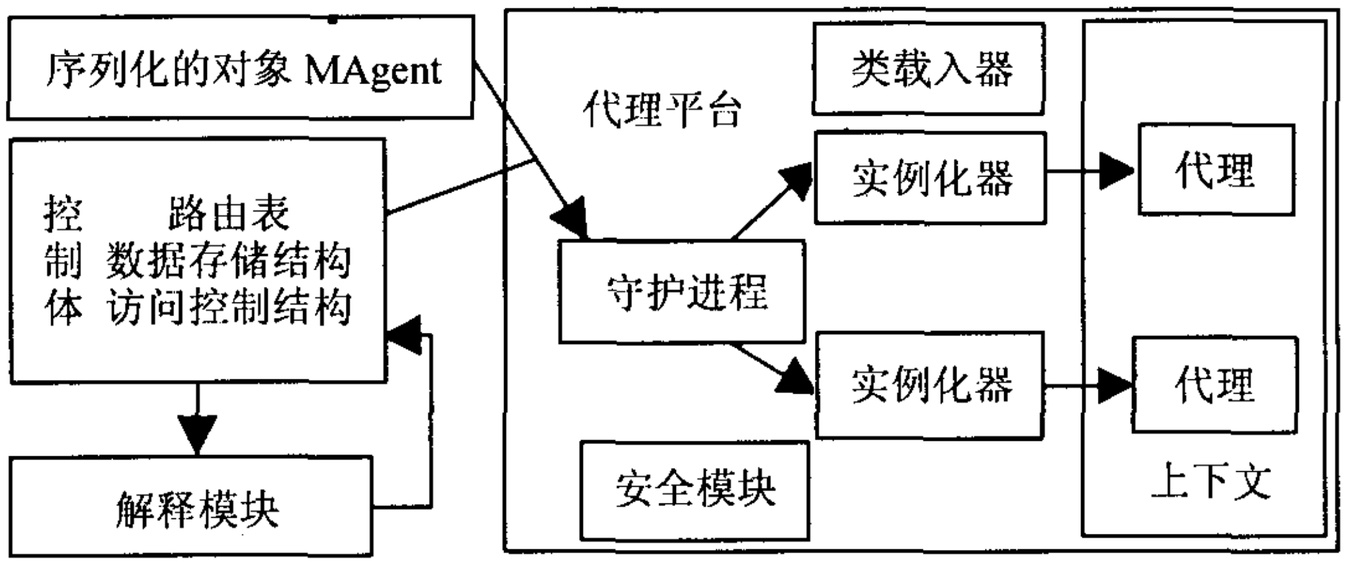

图 3 扩展的代理平台 MAS

载入器从适当的位置载入类代码并实例化，在同一上下文中代理完成任务和实现彼此间的通信，在到下一目标前，依然要重新修改控制体。

4 应用实例分析

我们在 MAS 系统基础上开发了一个个人商务助理应用程序。假定我们需要在几个提供购书服务的站点寻找最佳，例如，找到价格最低的书店。我们在查找代理中输入相关条件，点击发送，这时代理会按照在配置当中指定的路由和剔除条件动态地前往各个供书站点，得到最终结果返回。配置如表 3 所示，我们以局域网环境作为测试条件。

表 3　测试配置列表

角色	配置	备注
客户端	10.10.9.1/MAS	
书店 1	10.10.9.2/MAS	同域主机
书店 2	10.10.9.3/MAS	不同域主机
书店 3	10.10.9.4/MAS	可选测试机
书店 4	10.10.9.5/MAS	作为剔除主机

我们对这个实例做了一些测试，以同一域两台主机、同一域三台主机、不同域两台主机、两台同域一台不同域及三台不同域作为测试用例得到如表 4 所示的结果。通过测试对迁移情形和相关的安全改变有一个感性的认识。

我们发现如果要实现跨域访问实际上增加了一个中介，以时间换取安全在很多应用中是必不可少的，当然具体应用还要视情况而定。

表 4　测试用例及结果图

测试用例 (Test case)	时间 (ms)	
	有安全性	无安全性
同一域两台主机	1400	350
同一域三台主机	2200	570
不同域两台主机	2500	580
两台同域一台不同域	3900	950
三台不同域主机	5100	1200

5 结束语

本文阐述了一种移动代理迁移技术的实现，并对所涉及的动态路由问题和安全相关问题作了阐述。我们进一步提出任务和控制分离的代理迁移机制。在最后，则给出了一个应用实例，对代理传输进行了测试，下一步，我们在实现扩展的代理迁移机制的基础上，研究基于配置的控制即赋予用户更大控制权的同时，只需要修改特定的控制文件而无需修改源代码的迁移机制以进一步实现应用和控制分离的原则。

参 考 文 献

[1] Chess D M. Security issues in mobile code systems. In G Vigna, editor, Mobile Agents and Security, volume 1419 of Lecture Notes in Computer Science, New York, NY: Springer-Verlag, 1998, 1–14.

[2] Karnik N M, Tripathi A R. Design issues in mobile agent programming systems. *IEEE Concurrency Magazine,* 1998, 6(3): 52–61.

[3] Wang X F, Yi X, Lam K Y, Okamoto E. 1998, Secure information gathering agent for internet trading. Proc. of the 4th Australian Workshop on Distributed Artificial Intelligence in joint with the 8th Australian Joint Conference on Artificial Intelligence (DAI 1998), Australia, Lecture Notes in Artificial Intelligence, Springer-Verlag, Vol.1544: 183–193.

[4] 陶先平, 吕建, 张冠群, 李新, 董桓. 一种移动 Agent 结构化迁移机制的设计和实现. 软件学报, 2000, 11(7): 918–923.

[5] Eckel B. Thinking in Java, 2nd Edition, Upper Saddle River, New Jersey: Prentice-Hall, 2000: 973–980.

[6] Schaaf M, Maurer F. Integrating Java and CORBA: A programmer's perspective. *IEEE Internet Computing Magazine*, 2001, 5(1): 72–78.

[7] Wilhelm U G, Staamann S M, Buttyan L. A pessimistic approach to trust in mobile agent platforms. *IEEE Internet Computing Magazine*, 2000, 4(5): 40–48.

[8] Farmer W, Guttmann J, Swarup V. (1996). Security for mobile agents: Authentication and state appraisal. In Proc. of the European Symposium on Research in Computer Security (ESORICS), LNCS, No.1146, Berlin: Springer-Verlag, 1996: 118–130.

[9] 王汝传, 徐小龙. 移动代理安全机制的研究. 计算机学报, 2002, 25(12): 1294–1301.

[10] 王汝传, 赵新宁. 基于网络的移动代理系统安全模型研究和分析. 计算机学报, 2003, 26(4): 477–483.

[11] Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. New York: John Wiley & Sons, 1996, Chapter 19.

王汝传: 男, 1943 年生, 教授, 博士生导师, 主要研究方向是计算机软件理论、计算机网络及信息安全、移动代理技术等.

穆 鸿: 男, 1980 年生, 硕士生, 主要研究方向为计算机网络、移动代理和信息安全技术.

徐小龙: 男, 1977 年生, 博士生, 主要研究方向为计算机软件、计算机网络、信息安全、移动代理等.