基于 Mobile Agent 的服务移动性实现

于玉海,张 平
(北京邮电大学无线新技术研究室,北京 100876)

摘 要: 本文提出了一种基于 Mobile Agent 的服务移动性系统(MASM,Mobile Agent Based Service Mobility),并在模拟环境中加以实现.MASM 系统充分利用了 Mobile Agent 和 Java 技术的优势,提出了用"打点"和模块化等技术设计 Mobile Agent 的新方法,从而有效地减少了网络传输负载,灵活地实现了跨不同网络的服务移动.本文着重在 MASM 系统和 Mobile Agent 结构设计上以及具体服务移动的实现上.

关键词: Mobile Agent; Java; MASM 系统; 移动通信网络; 服务移动性
中图分类号: TN393.08 **文献标识码:** A **文章编号:** 0372-2112 (2002) 12A-2061-05

Mobile Agent Based Realization of Service Mobility

YU Yu-hai, ZHANG Ping
(*Wireless Technology Innovation Labs, Beijing University of Posts and Telecommunications, Beijing* 100876, *China*)

Abstract: This paper presents a service mobility system based on Mobile Agent: MASM (Mobile Agent Based Service Mobility) and realizes it in simulation environments. MASM takes full advantages of Mobile Agent and Java, brings forward a new design method of Mobile Agent based on "Dotting" and modularization technologies, and accordingly it reduces the network transfer load effectively and makes the realization of service mobility across different networks neatly. This paper mainly focuses on the design of both MASM system and Mobile Agent inner architecture and the implementation of concrete service mobility.

Key words: Mobile Agent; Java; MASM system; mobile communication network; service mobility

1 引言

随着移动通信的飞速发展,通信基础设施的日渐完善,服务质量和种类的提供成为第三代移动通信的焦点.第三代移动通信体系结构的代表 UMTS(Universal Mobile Telecommunication System)的目标之一,就是将目前二代系统的各种异质网络结构统一起来,使得用户在任何时间、任何地点都可以获得所需的服务[1].也就是说,当用户漫游到不同网络(该网络中不存在用户正在使用或申请的服务)的时候,要求当前网络服务提供者与用户本地网络服务提供者或第三方服务提供者进行联系,使所需的服务可以移动到当前的网络中继续为用户服务,并保证正在使用的服务不间断,这就是服务移动性的实现.

从 90 年代初期以来,人们开始致力于 Mobile Agent 技术的研究.Mobile Agent 作为对分布式技术的发展,在继承了分布式技术优点的同时又拥有自身的特点.Mobile Agent 具有自治性、移动性等特点,将其应用在移动通信领域中可以有效地减少网络传输负载、解决连接不稳定等问题[2].近年来 Sun 公司推出的 Java 技术获得了普遍的应用,由于 Java 具有平台独立性、动态类下载、多线程编程和对象串行化(Serialization)等特点,有利于 Mobile Agent 平台和 Mobile Agent 技术的实现.上述 Mobile Agent 和 Java 技术的特点非常适合服务移动性,因此可以将基于 Java 的 Mobile Agent 技术应用于服务移动性的实现中.

目前,在移动通信领域中存在许许多多的网络运营商和服务提供商,他们之间的服务往往互不兼容,因此如何保证用户在跨不同网络漫游时服务不间断将成为十分迫切的问题.解决这一问题的办法之一就是实现服务移动.本文从设计简单易行的服务移动性系统出发,提出了 MASM 系统和全新的设计 Mobile Agent 的方法,从而有效地实现了用户跨不同网络漫游时服务不间断,减少了网络传输负载,克服了连接不稳定问题.

2 MASM 系统结构

关于 Mobile Agent 目前没有一个统一的定义,由于应用领域的不同,Mobile Agent 将有不同的特性.MASM 系统中 Mobile Agent 的定义是:i 软件程序;ii 可以在自身的控制下从一个环境移动到另一个环境,并代表用户或应用自治地完成一项任务(即具有移动性和自治性);iii 可以在一些特定的位置点将程序挂起,并在移动完成后,在相应的位置点继续执行[3].

收稿日期:2002-05-28;修回日期:2002-07-20
基金项目:国家 863 项目(No.2002AA123045)

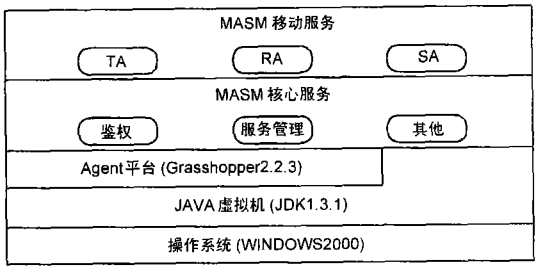

图 1 MASM 系统结构

MASM 系统结构(图 1)中包括两类服务:核心服务和移动服务.核心服务是采用 Static Agent(与 Mobile Agent 相比缺少移动性但增加了一些智能性)来实现的,包括鉴权 Agent(AUTA)、服务管理 Agent(SMA)和其他 Agent;移动服务是采用 Mobile Agent 来实现的,包括终端 Agent(TA)、漫游 Agent(RA)和服务 Agent(SA).下面简单说明各种 Agent 的功能:

• 终端 Agent(TA):是一个从当前网络的 SMA 处得到的 Agent,为用户提供一个界面来完成用户鉴权,建立、更新和完成服务连接,浏览当前网络中可获得的服务,申请和开始服务.

• 漫游 Agent(RA):由 SMA 创建,带有用户当前正在使用的服务信息和所有可获得的服务信息,跟随用户一起漫游到当前网络中,在当前 SMA 处进行注册,根据相应的服务信息定位所需 SA.

• 服务 Agent(SA):代表用户所需的各种各样的服务,每个用户使用的 SA 是系统中 SA 的一个拷贝(拷贝工作由 SMA 来完成),每个 SA 在漫游时携带其相应的程序和运行数据.

• 鉴权 Agent(AUTA):与 TA 连接,完成用户的鉴权,并将鉴权信息提供给 SMA.

• 服务管理 Agent(SMA):提供一个 TA 到用户终端;得到 AUTA 鉴权成功的数据后,查询用户的服务信息;为用户在网络中创建一个 RA,并将相应的服务信息传给 RA;为用户创建一个所需的 SA 的拷贝;代表 RA 与其它的 SMA 进行通信.

• 其它 Agent:利用 Java 的 API 直接编写,完善 Grasshopper 平台的功能,该部分留到下一步实现.

3 MASM 服务移动性

目前针对服务移动性的研究还处于初级阶段,主要集中在计算机和通信两个领域.在计算机领域中比较典型的是 Follow-Me 应用[4],其核心思想是:如果用户在完成某项任务的过程中离开了自己的计算机,那么他可以在另一地点的计算机上继续这项任务.Follow-Me 与 VNC(Virtual Network Computing)有所不同,后者只是不断地传送远端计算机的画面,而 Follow-Me 则是传送相应的工作环境.由于网络环境的差异,例如带宽资源等,使得通信领域的服务移动性的实现方式与计算机领域又有所不同.移动通信领域的服务移动性主要强调用户在跨网络漫游时,正在使用的语音和数据服务不间断.针对语音服务可以采用切换技术来保证服务不间断,而针对数据服务主要采用服务跟随用户漫游的方式来实现服务不间断.ITU-T F.852 对个人移动性和业务移动性进行了规范,主要是指 UPT(Universal Personal Telecommunication)用户通过个人号码能够在任意终端上经由多个网络与任意用户进行通话,其中涉及了服务移动性的思想.3GPP TS 22.121 中给出了 VHE(Virtual Home Environment)的概念,规范了用户的 PSE(Personal Service Environment)的便携性,即用户无论处于哪个网络、使用什么样的终端都将被呈现相同的个性化特性、接口特性和服务,VHE 中也包含了服务移动性的思想.在以上两个规范中只是涉及了服务移动性的思想,但并没有进行专门描述.本文根据 ITU-T F.852 和 3GPP TS 22.121 的描述进一步明确了移动通信领域中的服务移动性思想,即用户可以从一个网络漫游到另一个网络,并保证正在使用的服务不间断,同时提出并设计了一个基于 Mobile Agent 技术的服务移动性系统 MASM,有效地实现了移动通信网络中的服务漫游.图 2 描述了 MASM 系统解决服务移动性的流程.

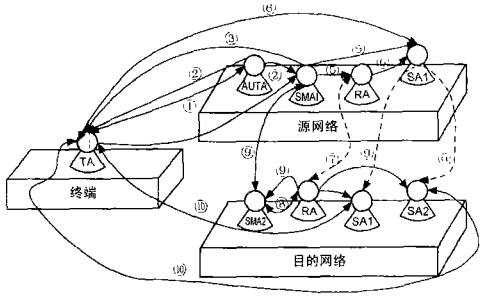

图 2 服务移动流程

服务移动性流程:

①用户使用 TA 向 AUTA 发送鉴权数据(用户名和口令)

②AUTA 向 TA 发送鉴权结果,若鉴权通过则将权限级别发送给 SMA1

③SMA1 搜集用户可获得的服务信息,并将服务信息发送给 TA

④用户通过 TA 选择所需的服务,并发送服务请求给 SMA1

⑤SMA1 根据用户的服务请求,定位所需的 SA,获得一个备份 SA1(如 SA 在本地则由 SMA1 负责创建备份,否则由远端 SMA 负责创建备份并移动到本地),同时为用户创建一个 RA,并将服务请求和可获得的服务信息传给 RA

⑥RA 指示 SA1 与用户建立服务连接,并开始服务

⑦当用户漫游到另一个网络时,他的 RA 也随着漫游到目的网络

⑧RA 与 SMA2 联系,查询所需的服务是否存在,并将查询结果返回给 RA

⑨根据查询结果,若当地网络存在所需的服务 SA2,RA 通过 SMA2 与 SMA1 进行联系,指示 SMA1 只将 SA1 的运行数据传递给 SA2,否则需将整个 SA1(包括程序和运行数据)移动到目的网络

⑩RA 指示 SA1 或 SA2 继续为用户服务

可以看出,基于 Mobile Agent 的服务移动性使终端用户及相应服务始终位于同一网络中,实现了用户服务的本地化.在传统的基于 RPC(Remote Procedure Call)的服务使用方式中,用

户应用与服务主要是通过 Client-Server 方式远程相互作用,而基于 Mobile Agent 方式的服务使用则使服务更接近用户,实现了服务与用户的本地相互作用.因此,基于 Mobile Agent 方式的服务使用与基于 RPC 方式相比,可以减少远程网络传输负载[5],见图 3.但这也需要一定的前提,即要设计一个合理的服务 Agent 结构及使用方式,使服务 Agent 的尺寸小于 RPC 方式中相互作用的数据量,否则将无法达到减少网络传输负载的目的.另外,由于服务 Agent 一次性地传输到目的网络,并在目的网络中直接与用户相互作用,所以可以克服服务使用过程中由于远程网络连接不稳定而带来的服务间断问题,从而保证了服务的可靠性.

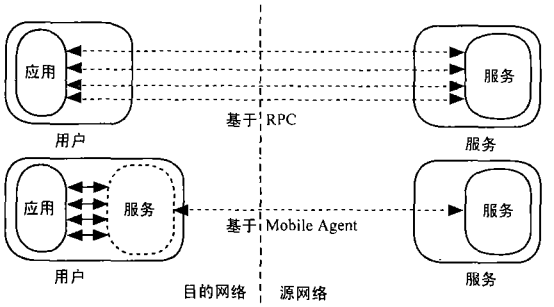

图 3　Mobile Agent 与 RPC 方式的比较

4　MASM 系统的实现

MASM 系统基于 IKV + + Grasshopper 平台[6],Grasshopper 平台符合 OMG MASIF(Mobile Agent System Interoperability Facility)标准[7],可以和其他遵循这一标准的平台进行交互.目前的模拟环境是使用局域网中的四台 PC 机,分别模拟移动终端、本地网络环境、当前网络环境和用户服务注册中心.所使用的软件及其版本为:操作系统 Windows 2000、Java 虚拟机 JDK1.3.1,Mobile Agent 平台 Grasshopper2.2.3,开发和编译工具 JBuilder 6.0 企业版.

Mobile Agent 是一个以自治的方式在网络中传递的程序(包括代码和运行数据),因此为了使所设计 Mobile Agent 的结构和性能更好地适应服务移动性的环境,MASM 系统在设计时主要考虑了如下因素.移动性是要考虑的首要问题,而且为了使移动的数据最少,程序要模块化;其次,由于 Mobile Agent 要控制自身的运行,因此需要一定的自治性,主要是设计一个合理的 Mobile Agent 生命周期;而且,服务移动性是由不同种类的 Agent 之间交互作用共同完成的,一个良好的通信机制就成了这一切的基础.最后,需要考虑由于程序在网络中移动而带来的服务定位问题.

4.1　移动性

Mobile Agent 的移动性是其核心功能[8][9],主要有两种:弱移动和强移动.弱移动包括移动代码和有关的数据,而强移动还要包括移动相应的执行状态.Java 是一种基于解释型的语言,用它来接触执行状态是相当困难的.目前的标准 Java 虚拟机不支持强移动,Grasshopper 平台是基于标准的 Java 虚拟机,因此也不支持强移动.但是为了在服务移动中保证服务不间断,又必须有类似强移动机制的支持.为此,MASM 提出并设计了一种用现有的弱移动模拟强移动的机制.

具体实现方法:首先,将 Mobile Agent 程序进行"打点",即在程序中每隔一定的间隔做一个标记(这里用变量来标记),使得 Mobile Agent 程序在移动之前必须执行完当前"打点"间隔内的语句.移动前先将程序挂起并保存运行数据(包括标记变量值),到达目的地后根据运行数据恢复运行环境,并在相应的标记点处继续运行.实际的实现方式采用的是 *switch*(标记变量)...case...语句,在程序的挂起和继续执行时分别使用的是 Java 的串行化(serialization)和反串性化(deserialization)机制.

实验证明,用"打点"的方法可以有效地标识程序执行点,保证服务不间断,但"打点"的时候要遵循两个原则:i"打点"的密度不要太大,否则会过分增加程序的代码量,增加传输负载;ii"打点"的密度又不能太小,这样将增加服务延迟时间.可以看出,这两个原则之间是个矛盾,在具体开发服务的时候,应根据每个服务具体的性能要求,通过对不同"打点"方式进行实验,找到一种折衷的方案.

4.2　模块化

Mobile Agent 技术的一个主要优势是可以减少网络传输负载,因此在设计 Agent 的时候应最小化它的尺寸.MASM 系统提出了一种模块化的 Agent 设计方式,即将 Agent 分成两部分:核心部分和辅助部分.核心部分是一个 Agent 运行所需的最小类,是一个核心类,辅助部分实现具体的服务内容,由一系列辅助类组成.这种设计方式充分利用了 Java 的动态类加载机制.当 Agent 移动的时候,只需将其核心类、运行数据及辅助类的 URL 移动到目的网络.在 Agent 运行的过程中,如果需要使用辅助类,则可以将其从相应的 URL 的 CODE BASE 处动态下载到当前网络,并动态加载到正在执行的 Agent 中,从而减少了不必要的辅助类的传输.这种动态类下载机制称为 COD(Code On Demand)[10].

4.3　生命周期

Mobile Agent 的生命周期是指一个 Agent 在运行过程中所有可能经历的状态.也就是说,在 MASM 系统中各种 Mobile Agents 无论运行在哪个网络中,它必然处于下面各个状态中的一个.其中在状态跃迁前,有时可能需要一些预处理,这些预处理是通过调用相应的预处理函数来实现的[11,12].

MASM 系统设计了一个完善的 Mobile Agent 生命周期(图 4),增强了 Agent 运行时的自治能力.Mobile Agent 创建起来之后,要执行相应的初始化函数 *Init*( )来完成 Mobile Agent 的初始化,之后进入 Agent 执行阶段.若在执行过程中需要移动,

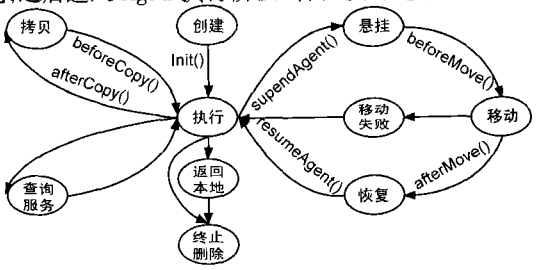

图 4　Mobile Agent 生命周期

则要先将 Agent 悬挂,保存相应的程序和运行数据,移动完成后在目的地恢复运行数据,并继续执行,如果移动失败,先进行失败处理,再继续执行.在 Agent 执行过程中还可以进行拷贝和服务查询工作.所有服务完成之后,根据需要决定是否将所携带的信息返回本地后再终止并删除 Agent,还是直接在当地网络处终止并删除 Agent.

4.4 通信

两个 Agent 环境(即 Agency)之间的相互作用需要特定协议的支持[13],MASM 系统中目前支持的通信协议有(见图 5):IIOP(Internet Inter-ORB Protocol),Java 的 RMI(Remote Method Invocation),plain socket 连接,以及被 SSL(Secure Socket Layer)保护的 RMI 和 plain socket.目前,建立在局域网上的 MASM 试验环境,主要采用两种协议:RMI 和 plain socket.分两种情况:

- 访问注册中心(Region)
  协议名称://主机名:端口号/Region 名称
- 访问 Agent
  协议名称://主机名:端口号/Agency 名称/Place 名称

其中协议名称为 RMI 或 Socket,主机名为 IP 地址,使用 RMI 时端口号为 7050,使用 Socket 时端口号为 7000 - 7020,Region 名称是指整个局域网所属的 Region 的名称,Agency 名称包括终端、源网络、目的网络的 Agency 名称,Place 名称是 Agent 所在的 Place 名称[6].

此外该系统也可以根据需要动态增加新的协议.

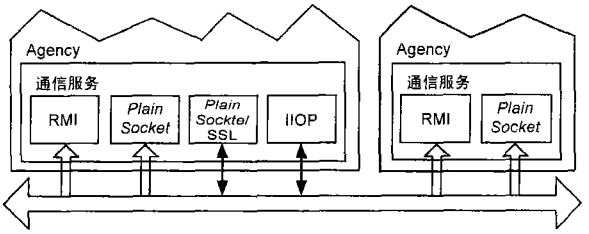

图 5 多协议的通信支持

4.5 服务定位

当用户在使用服务过程中发生漫游的时候,应尽量使用当前网络中可能提供的服务以节省传输负载,因而要求 MASM 系统中提供服务查询和定位机制.具体实现方式为,在 MASM 系统中建立一个用户及服务信息的注册中心,该中心记载了每个网络中存在的服务及用户信息.当用户到达新的网络时,目的网络的 SMA 将应 RA 的要求与注册中心进行联系,查找所需的服务信息,然后由 RA 决定如何使用服务.服务使用原则为:

- 尽量在当前网络中找到一个和原服务相同或相似的服务.对于相同的服务可以直接采用,对于相似的服务经 RA 分析后决定是否采用.在使用当前网络服务的情况下只需移动相应服务的运行数据.
- 若当前网络中不存在所需的服务,将源网络中的服务程序和运行数据一起移动过来.

5 服务移动性的应用

基于 MASM 系统我们开发了一系列支持服务移动性的应用,下面介绍一个跨网络漫游时连续下载电话号码簿的应用(见图 6).通过鉴权后,在用户的终端界面上将呈现出可使用的服务信息.用户根据自身要求选择所需的服务,然后向网络发送定制信息,服务定制成功后,便可开始使用.在服务使用过程中,用户可以跨网络漫游,但下载工作会继续进行,并会在服务使用状态栏中得知下载进度,详细流程见图 7.实验证明,利用 MASM 系统可以有效地实现服务移动.

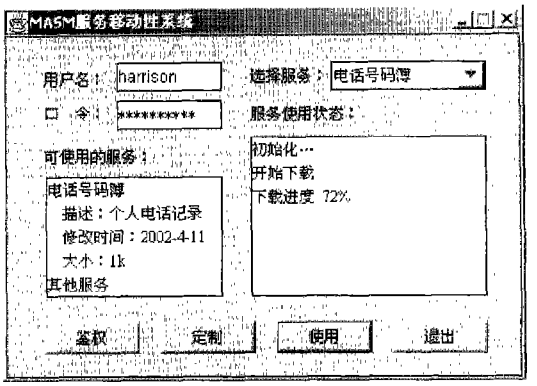

图 6 服务移动性的应用

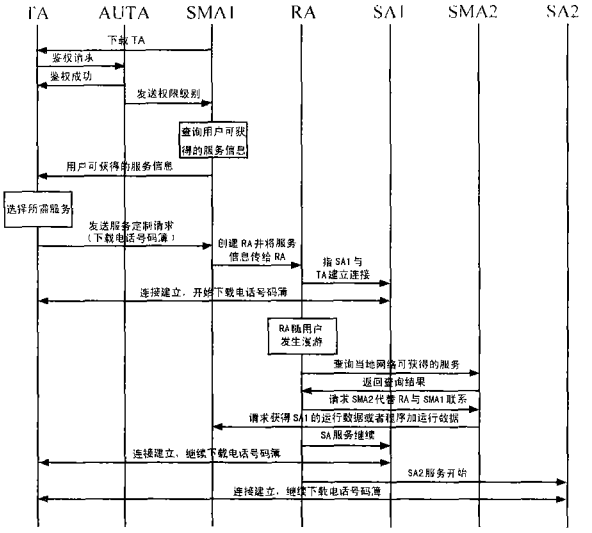

图 7 电话号码簿下载

在设计基于 MASM 的服务时,应使终端侧的功能尽量简单,并将服务所需的大部分功能放在网络侧,从而合理使用终端及网络侧的资源.MASM 系统中只有 TA 位于终端侧,它代表用户与网络侧进行交互.因此,TA 的界面和内部程序设计要尽量简单,使其更好地适应移动终端的运行环境.

目前,移动通信领域针对服务移动性的研究还处于初级阶段,还没有可商用化的产品.虽然一些项目,例如 CAMELEON(Communication Agents for Mobility Enhancements in a Logical Environment of Open Networks)、VESPER(Virtual Home Environment for Service Personalization and Roaming users)等,已经用 Mobile Agent 技术实现了 VHE 的概念,但其实现的内容和方式过于复杂,很难直接应用于实际系统中.因此,我们主要针对其中的服务移动性进行研究,并努力改进 MASM 系统的性能,

尽量使其实用化.

6 结论

MASM 系统有效地解决了移动通信网络中的服务移动问题,为异质网络的互通提供了基础. Mobile Agent 方法的使用提高了服务移动的灵活性,它结合"打点"和模块化方法,实现了跨网络漫游时服务的连续性,减少了网络传输负载,克服了远程连接不稳定等问题. MASM 系统的初步实现使我们在将 Mobile Agent 技术应用到移动通信领域方面有了一个良好的开始,同时也为今后的工作提供了开发和研究的基础.

参考文献:

[1] Uskela Sami, et al. Service portability across mobile networks [A]. ACTS Mobile Communications Summit'99 [C]. Sorrento. ACTS, 1999. 8 - 11.

[2] Green, S, et al. Software agents: A review [R]. IAG Report, Trinity College, Dublin, 1997.27 May.

[3] Finnset W, Grav J, Rogde G W, Zoric J, Aarbø J. Intelligent mobile agents-status and four promising application domains [R]. FoU N 24/99.

[4] Kazunori Takashio, Gakuya Soeda, Hideyuki Tokuda. A mobile agent framework for follow-me applications in ubiquitous computing environment [A]. ICDCSW'01 [C]. Mesa, Ariyona USA, April 2001.

[5] C G Harrison, D M Chess, A Kershenbaum. Mobile agents: Are they a good idea [R]. IBM Research Report, T. J. Watson Research Center, NY, 1995.

[6] Grasshopper platform [DB]. Available from http://www.grasshopper.de/.

[7] D Milojicic, et al. MASIF: the OMG mobile agent system interoperability facility [A]. Proc. Mobile Agents Int'l Workshop [C]. Springer-Verlag, Berlin, 1998. 50 - 67.

[8] Paolo Bellavista, Antonio Corradi, Cesare Stefanelli. Mobile agent middleware for mobile computing [J]. IEEE Computer, 2001, 34(3): 73 - 80.

[9] A Fuggetta, G P Picco, G Vigna. Understanding code mobility [J]. IEEE *Transaction on Software Engineering*, May 1998.

[10] Karnik N Tripathi. Design issues in mobile-Agent programming systems [J]. IEEE Concurrency, July-September, 1998.

[11] Kothari, N. AgentOS-a java based mobile agent system [R]. ICS Honors Project Final Report, 1997. available from http://netresearch.ics.uci.edu/agentos/doc/nikhil-final-report.pdf.

[12] Laurent Magnin, et al. Our guest agents are welcome to your agent platforms [DB/OL]. 2002. http://www.crim.ca/~lmagnin/pub/publis/guest_aims2002_A4.pdf.

[13] J Baumann, et al. Communication concepts for mobile agent systems [A]. In: Mobile Agents, Proc. 1st Int. Workshop, MA'97 [C]. Springer, 1997.

作者简介:

于玉海　男,1972 年生于黑龙江省齐齐哈尔市,现在北京邮电大学攻读博士学位,主要研究方向是移动代理技术在移动通信系统中的应用.

张　平　男,1959 年生于陕西省汉中市,现任北京邮电大学教授,博士生导师,主要研究方向是移动通信系统.