软件, Agent, MSA (4)

计算机科学1998Vol. 25№. 5

62-66

可移动的软件 Agent 研究*)

Research of Mobile Software Agent

TP311.5

陈 翀 麦中凡

(北京航空航天大学计算机科学与工程系 北京100083)

摘 要 In this paper, firstly, we define the concept and appraise the significance of mobile software Agent. Secondly, we introduce the research technology background. After analyzing the main research content, we put forward the research method which we adopt. Finally, we discuss the current research situation.

关键词 Mobile software Agent, Software Agent, Distributed computing

随着网络技术,特别是 Internet 的发展,整个计算环境正在发生深刻的变革,这表现在高度的分布式,环境的异质性和极强的动态性等方面。传统的客户/服务器(C/S)模型因为其灵活性差等特点已不能很好地满足大而复杂的分布式计算要求,我们综合分析了几方面的研究成果(包括软件 Agent,移动代码技术,进程迁移等)认为可移动的软件 Agent (MSA,Mobile Software Agent)可作为 C/S 模型的补充来解决分布式计算中的一些更为复杂、更加灵活的问题,同时它也为下一代网络计算系统的灵活性、主动性、安全性的研究提供基础,例如支持软件动态重组、诊断的软件机器人等。本文首先提出 MSA 的概念和研究意义,在介绍技术背景的前提下,分析了 MSA 的主要研究内容,并提出我们采用的研究方法,最后评述了国内外的研究现状并作全文总结。

1 MSA 的概念及研究意义

简单来说,一个 MSA 是一个应用程序,它可以自主地在异质环境下运行,在多个位置之间迁移;在迁移和运行中,通过和环境交互完成用户指定的任务,MSA 的概念模型如图1所示。其中,异质环境主要指异质的操作系统、异质的网络、异种设备、不同的数据格式等,位置是在网络和操作系统之上的一种抽象概念,是 MSA 运行交互的场所。MSA 迁移的内容既包括 MSA 代码也包括 MSA 的运行状态,运行状态可分为执行状态和数据状态。MSA 的执行状态主要指 MSA 当前运行时状态,例如程序计数器,运行栈内容等。数据状态主要指与 MSA 运行有关的数据堆的内容。按所迁移运行状态的内容,MSA 的迁移可分为强迁移和弱迁移。强迁移同时迁移 MSA 的执行状态和数据状态,但这种迁移实现较为复杂,MSA 状态的捕获、保存和恢复操作费时且昂贵。弱迁移只迁移 MSA 的数据状态,相对于强迁移而言,这种迁移速度较快但不能保存 MSA 完整的运行时信息。这里提到的用户可以是最终用户

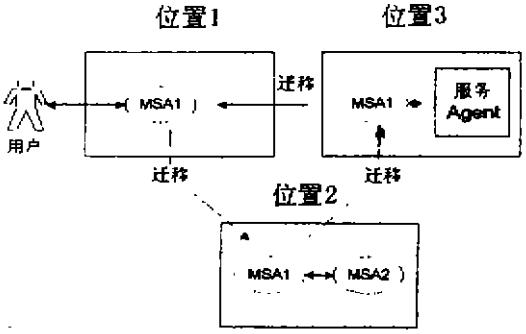


图1 MSA 概念图

*) 本文得到国家自然科学基金项目(69673003)"可移动的软件 Agent 研究"资助。陈 翀 硕士生,主要研究领域:软件工程、软件 Agent。麦中凡 教授,主要研究领域:软件工程、程序设计语言、面向对象方法学、数据库、分布式智能软件系统等。

也可以是其他应用程序(包括MSA),MSA和环境的交互包括MSA同最终用户的交互,MSA同其他MSA的交互,以及MSA同驻留在运行环境中的其他服务Agent的交互。

应该说,MSA的计算方式遵循一种"计算"和"迁移"交替进行的方式,可以形象地称其为"跳跃计算"。同传统的C/S模型相比,MSA由于其自身的移动性和灵活性可以较好地解决下述问题:

·扩充服务器端功能。在传统的C/S模型下,服务方提供一套固定的操作。使用MSA,用户可以编写自己的MSA将其发送到服务器端,在运行时扩充服务器的功能,这样就提高了整个系统的灵活性。

·容易定位和恢复。在大型分布式系统中,传统的C/S模型构成了网状交织的服务和被服务关系,给开发和维护带来困难,当出现部分失败时不易定位。而MSA将分布式计算过程封装在一个应用程序中,通过迁移完成处理,较易跟踪和恢复。

·支持移动式计算。移动式计算作为一种重要的计算模式具有低带宽,高等待时间,线路费用高,时常断连等特点。在数据量很大的情况下,传统的C/S模型也不适应这种计算的需求。采用MSA,用户可以将MSA从个人移动计算设备发到永久连接的网络环境下迁移计算,最终用户再连线取回结果。

·自动软件升级安装。在大型分布式环境下,软件升级和管理是一件十分繁琐费时的工作,需要由系统管理员逐台机器分别操作。在这种情况下,可以利用MSA迁移的特点,编制MSA使其在迁移过程中完成软件的升级安装,减轻系统管理的负担。

此外,由于在大型分布式环境下遇到的问题一般都较为复杂,因此也需要分布式应用具有一定的自主推理、智能决策、与其他应用互操作的能力。而这一能力正是MSA继承了软件Agent特点所拥有的,因此可以满足复杂应用的要求。由上面所述可以看出,MSA的迁移计算模型并不是必须的,所有问题都可以用C/S模型或其他方法解决。但是在某些情况下,特别是适于"跳跃计算"的各种应用领域采用MSA可能是最佳选择[1]。MSA可应用于网络管理、软件发布升级、分布式信息采集、个人数字助理、移动式计算、工作流管理、电子贸易、活动文档、邮件等各个方面。

2 MSA研究的技术背景

可移动的软件Agent是一个新的研究领域,它的提出并非是凭空臆想,有一些相关的研究工作作为其基础。这包括软件Agent、分布式计算模型、分布式对象技术、操作系统中的进程迁移技术等方面。

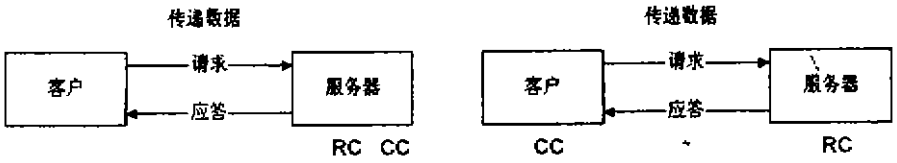
2.1 软件Agent研究

软件Agent的研究可以追溯到70年代分布式人工智能DAI的研究,主要分为两条研究主线[2]:一条从70年代开始一直持续到现在。这条主线围绕经典AI展开,主要研究Agent的拟人行为,多Agent协商模型等。研究方向可分为Agent理论,Agent体系结构,Agent语言,多Agent系统等。一些计算机科学家也称之为智能Agent或"强定义"的Agent[3]。另一条从90年左右开始到现在,这一条以应用研究为主,将经典AI关于Agent的"强定义"弱化,拓宽了Agent应用范围。主要包括界面Agent,基于Agent的软件工程等研究方向。

虽然软件Agent研究工作已经进行了20多年,但对Agent依然没有一个公认的定义。一般认为,软件Agent是具有一定智能的软件,在概念上层次较高;它为完成用户的任务而自主地运行;在执行过程中,它总是处在一定的环境内,并且作为这个环境的一部分感知环境的变化,并通过行动影响环境[4]。一个软件Agent应具备三个基本属性:自主性(Autonomy),社交能力(Social Ability)以及适应能力(Adaptability)。可移动性并不是软件Agent所必须的属性[2,3,4]。

2.2 分布式计算模型

在分布式计算环境中,计算资源可分为资源构件(RC,Resource Component)和计算构件(CC,ComputingComponent),它们分散在各地,要完成



(a) RC和CC在服务器端　　(b) RC在服务器端CC在客户端

图2 C/S模型

计算任务，RC 和 CC 必须处在同一地点，这就需要一种分布式计算模型支持。在 MSA 以前，根据 RC 和 CC 所处的位置、传输的内容以及传输方式可分为三种计算模型：

1)C/S 模型。最常用的是基于消息传递和 RPC 的 C/S 模型，这种模型的特点是：网络上传递的是数据。根据 RC 和 CC 位置的不同又可分为两种情况(如图2所示)：一种是 RC 和 CC 都在服务器端，如数据库；另一种是 CC 在客户端，RC 在服务器端，如文件服务器。

2)远程求值模型(REV，Remote Eval-uation)。其特点是：一方拥有 RC，一方拥有 CC，当需要计算时，将 CC 发到 RC 一方执行。这一模型在网络上传递的是过程代码，并且传递是一种"推"的方式(如图3所示)。REV 比较典型的例子是 SUPRA-PRC，REV 系统等。

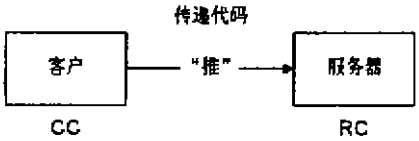


图3 REV 模型

3)代码请求模型

代码请求模型(COD，Code On-Demand)与 REV 模型相似，在网络上传输的是代码，但 COD 模型的传递是通过"拉"的方式，即将 CC 下载到 RC 方执行(如图4所示)。这一模型的例子是 ActiveX 和 Java Applet。

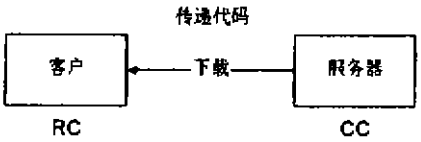


图4 COD 模型

MSA 与这三种模型都不相同：首先，MSA 是在运行时迁移，因此迁移的不仅是代码还有运行状态，而 C/S 模型只传递数据，REV 和 COD 是运行前迁移，仅传递代码。其次，MSA 是自主地迁移，迁移动作已经成为 MSA 的组成部分，而 REV 和 COD 都是由外力驱动迁移。再次，由于 MSA 是运行时迁移，因此迁移可以多次进行。而 REV 和 COD 仅涉及两方，只进行一次迁移。最后，MSA 继承了软件 Agent 特点，在运行迁移中可以和其他 Agent 或应用程序交互，而 REV 和 COD 一般不具有这种能力。比较而言，MSA 与 REV 模型有更多类似之处，可以说，MSA 是对 REV 的扩展。

2.3 分布式对象技术

分布式对象的思想[5]是：在分布式系统中引入一种可分布的、可互操作的对象机制，并且把分布网络上可用的所有资源看作公共可存取的对象集合，使得不同的对象可以集成在一起。此外，一个对象客户能够通过定义在分布对象模型上的接口来访问分布系统的可用对象。

分布式对象技术为对象在分布网络环境中的协作提供了一种模型。它的核心技术包括对象请求代理，对象接口描述语言，对象命名和管理等。目前，主要有两种对象模型即 OMG 定义的 CORBA 和 Microsoft 定义的 DCOM。但是现在分布式对象技术中对象是静态的，不利于动态多变的分布式环境。引入 MSA 可以作为对象模型的扩充，即在对象的静态模型中加入动态机制，使其适用于更复杂的分布式环境。同时，在研究 MSA 交互和管理时，也可以借鉴分布式对象中的一些技术，比如：对象请求代理，对象唯一命名策略等。

2.4 操作系统中的进程迁移技术

在松耦合的分布式系统中，为了达到系统负载平衡和资源平均利用，并且也为了系统应用的健壮性，引入了进程迁移技术。进程迁移[6]就是指操作系统中的一个进程在资源紧张等情况下，由系统监控并调度到其他机器上继续运行。

进程迁移与 MSA 都可以迁移，但两者有不同之处：首先，进程由系统调度，迁移也由系统决定，而 MSA 能够自主地迁移；此外，进程迁移主要为提高系统性能，与进程自身要完成的任务无关，而 MSA 的迁移则带有明确的目的性；再次，进程层次较低，一般只能在同构的系统间迁移，而 MSA 可以在异质环境下运行迁移；最后，进程迁移一般仅限于局域网内，而 MSA 可以在广域网甚至 Internet 上运行。尽管两者存在着不同，但是在研究 MSA 系统模型时，我们仍可以利用进程迁移的某些技术，如：迁移进程的控制，进程状态的保存、迁移和恢复等。

3、主要的研究内容及研究方法

可移动的软件 Agnet 主要的研究内容有以下几个方面：

1)MSA 模型及 MSA 系统的体系结构

如何编制一个 MSA 以及支持 MSA 运行迁移的体系结构是 MSA 系统面临的两个核心问题。由

于 MSA 继承了软件 Agent 的特点，使它与一般应用程序有所不同，因此在构造 MSA 时需要一种模型来支持。设计这一模型不仅应该考虑 MSA 的多种属性（自主地迁移性，社交能力，智能决策性等），使构造的 MSA 能够有充分的表达能力，而且也应该考虑模型的性能，使构造的 MSA 轻便易用符合"跳跃计算"的思想。此外，这一模型也应具有开放性，可以随时扩充一些新的功能。MSA 系统体系结构主要研究构造一个 MSA 系统所必须具备的部件以及各部件之间的关系。一个 MSA 系统至少应该包括一个支持 MSA 运行的运行时环境和一个支持 MSA 迁移的系统传输机制（如图5所示）。另外，还可以包括安全检查部件、MSA 管理部件、本地服务资源管理部件、支持 MSA 与环境交互部件以及其他一些辅助设施。

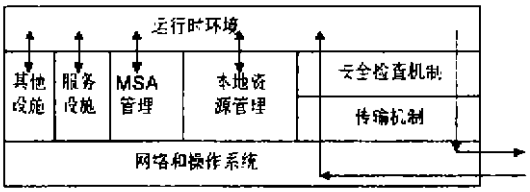


图5 MSA 系统概念图

2）面向 MSA 的程序设计语言

MSA 语言主要研究如何选择和设计一种程序设计语言使其适于 MSA 的开发。MSA 语言在选择设计时应该首先满足 MSA 模型的要求，通过语言可以实现 MSA 的几种特性。在此基础上，还要注意以下几个问题：①语言是选择编译型，编译解释型还是解释型，这既关键到系统的运行效率也关系到系统的安全性等多个方面，在选择时应综合加以考虑。其中，解释型和编译解释型语言在实现 MSA 跨平台运行方面较为方便。②考虑到 MSA 运行时的动态性等特点，MSA 语言应采用何种类型系统（强类型、弱类型和无类型）。③选择何种风格的程序设计语言更适于 MSA 的开发。④是否在 MSA 语言中采纳一些程序设计语言的优秀机制，如 C 语言的指针，SUN 公司开发的 Java 是一种面向对象的、平台无关的、具有一定安全性的、支持分布式计算的语言，因此它是构造 MSA 系统较合适的语言。

3）MSA 与环境的通信

MSA 不可能孤立地运行，为了获得信息它需要同外界进行交流，这包括 MSA 之间通信以及 MSA 同服务提供者通信等方面。要完成 MSA 通信需要考虑两个层次的问题，即通信层和知识层。通信层是信息交流的基础，主要解决 MSA 与外界进行通信的方式和途径等问题，例如，可以采用消息传递和远程过程调用来实现。知识层在通信层之上，主要解决信息的知识表示以及标准化等问题，例如，在知识层可以采用 Agent 通信语言（ACL，Agent Communication Language）[7]，以便让不同领域的 MSA 和服务提供者对传递的信息达成共识，实现它们之间的互操作。ACL 为 Agent 表达其目的、任务、信念和不同领域的知识提供了有力的工具，它主要包括三部分：ACL 词汇表，知识交换格式（KIF，Knowledge Interchange Format）和知识查询与表示语言（KQML，Knowledge Query and Manipulation Language）[8]。

4）MSA 系统的安全性

MSA 系统的安全性包括四个方面：多 MSA 之间的安全性、主机之间的安全性、主机与未授权第三方的安全性以及 MSA 与主机之间的安全性。其中，前三者是分布式系统始终研究的问题，目前多利用加密或认证技术解决，但是最后一种情况是由 MSA 迁移运行的特殊性产生的，值得深入研究。

MSA 与主机之间的安全性是双向的：一方面，主机上的资源要防止恶意的 MSA 入侵；另一方面，MSA 也要防止恶意主机的侵犯，保护信息的私有性。对于前者，首先可以采用身份认证技术作为外围防卫，然后通过运行时检查对 MSA 的行为进行限制作内层防卫，例如：可以将 MSA 限制在一定的空间内运行，并禁止其直接访问非授权的资源构件等。对于后者，还没有较好的方法可以采用，目前，可以通过限制 MSA 迁移的目的地，使其只前往信任的位置运行来初步实现。但是采用这种方法，一方面破坏了系统的开放性，同时也不能保证 MSA 迁移过程中的安全性（被中间主机侵犯），有一定的缺陷。

除以上四个研究内容外，还包括：①MSA 管理，例如，MSA 生命期追踪，MSA 状态保存与恢复，多 MSA 死锁监测，MSA 唯一标识等；②服务资源的标识与发现策略；③MSA 的性能分析模型；④多 MSA 协商模型；⑤MSA 开发工具以及相应的应用开发方法学等多个方面。

从上述分析可以看出，MSA 是一个涉及分布式计算、程序设计语言、人工智能、网络通信，面向对象技术等多个领域的综合研究课题。多种研究内容交织在一起造成一定程度的混乱，增加了 MSA 研究分析的复杂性，因此需要一种适当的研究方法作为

整体指导。针对这种情况，我们提出并采用了一种分层的方法。具体来讲，将 MSA 分为系统层和 Agent 层两个层次进行研究。在系统层以分布式计算为核心，主要研究为 MSA 提供迁移、安全性、底层通信机制以及管理等系统级实现技术。在 Agent 层以软件 Agent、人工智能为主，主要研究 MSA 模型、MSA 语言、MSA 在知识层的通信和 MSA 的协商模型等。此外，在 Agent 层也将研究 MSA 在实际中的应用以及应用开发方法学，即如何利用 MSA 解决实际问题。系统层和 Agent 层并不完全割裂的，它们之间有着内在的联系，例如：MSA 语言决定着运行时环境，MSA 底层通信机制为知识层通信提供了实现途径等。采用这种研究方法一方面可以使概念清晰便于分析问题，另一方面又可以使低层系统实现和具体应用分离，易于提供面向应用的通用开发工具。

4 目前的研究现状

可移动的软件 Agent 是一个新兴的研究领域，与我们同期进行研究的还有一些国外的公司、大学和科研机构，它们主要分布在德国，美国，日本，瑞士等地。其中，Telescript[9]和 AgentTcl[10]是最著名的两个国外系统。

·Telescript 系统。由 Magic 公司于1994年推出，它是第一个商业化的 MSA 系统，主要用于电子商务，一定的网络管理工作。Telescript Agent 采用的是 Telescript 语言，它是一种面向对象的编译解释型语言。使用 Telescript 开发的 MSA 可以实现在任意点迁移，多 MSA 之间可以进行通信。另外，Telescript 引擎也可以对 MSA 实行一定的管理。在安全性方面，Telescript 提供了一些授权和访问控制的机制，可以说，Telescript 是目前最强壮的 MSA 系统。但是，Telescript 的最大缺点是效率较低，没有提供完整的安全模型。此外，MSA 与外界尚不能在知识层进行通信，这限制了 Telescript 的应用范围。

·AgentTcl 系统。是 Dartmouth 大学开发的 MSA 系统。AgentTcl 主要用于 PDA、工作流管理、分布式信息采集等方面。系统的 MSA 采用扩充的 Tcl(Tool Command Language)脚本语言 AgentTcl。和 Telescript 一样，AgentTcl 的 MSA 可以在任意点迁移，Agent 之间可以进行通信。此外，Agent 可以使 Tk 开发用户界面和最终用户进行交互。AgentTcl 的主要缺点是：系统几乎没有安全性，AgentTcl 的 MSA 不具备在异质环境下运行能力，不能在知识层通信等。

参考文献

[1] C Harrison et al., Mobile Agents: Are they a good idea?, IBM Reseach Report, IBM T. J. Watson Research Center, 1995

[2] Hyacinth S. Nwana, Software Agents: An Overview. Knowledge Engineering Review, 11(3)1996

[3] Michael Wooldridge & Nicholas R. Jennings, Intelligent Agents: Theory and Practice, Knowledge Engineering Review, 10(2)1995

[4] S. Franklin & A. Graesser, Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents, http://www.msci.memphis.edu:80/~franklin/Agent-Prog.html,1996

[5] 陈 珅，麦中凡，基于分布式对象的软件构件，计算机科学，24(4)1997

[6] Fred Douglis & John Ousterhout, Transparent Process Migration Design Alternatives and the Sprite Implementation, Software Practice and Experience, 21(8)1991

[7] R. M. Genesereth & S. P. Ketchpel, Software Agents, CACM, 37(7)1994

[8] T. Finin et al., KQML as an Agent Communication Language, In Proc. Third Int. Conf. On Information and Knowledge Management, ACM Press, 1994

[9] J. E. White, Telescript Technology: Scenes from the Electronic Marketplace, General Magic, 1994

[10] Robert Gray, David Kotz, Mobile agents for mobile computing, Technical Report, Department of Computer Science Dartmouth College, 1996