计算机科学1999Vol. 26№. 3

Internet 环境中的软件 Agent*)

Software Agents in Internet Environment

周立柱 赵洪彪

(清华大学计算机系 北京 100084)

Abstract This paper discusses the application requirements from future Internet, analyses the concept of agent, introduces Message-typed Agent, mobile Agent, and their roles in Internet applications, and presents related study issues

Keywords Internet, Message-typed Agent, Mobile Agent

1 引 言

Internet 已经成为最重要的信息来源，由于 Internet 中的信息没有中心的控制点，多为半结构化和无结构的信息，信息格式可以互不相同，信息服务形形色色，用户的需求也不象数据库的数据操作那样明确和直接对应于某种操作语言，Internet 环境中的信息处理比起传统的数据库技术来说要复杂得多，需要新的软件技术提供支持。目前 Agent (Agent)技术在 Internet 上的应用迅速增加，在增加服务质量，提高易用性，信息发现和过滤，信息用户化和任务自动执行方面表现出很大的应用潜力，

我们相信在不远的时间内，Agent 软件将成为一种重要的计算模式，许多的应用系统将会不同程度地采用 Agent 技术，这种观点是基于下面一些理由：

(1)桌面应用系统正变得多种多样，以至于一般用户往往只能掌握其中一小部分功能，随着 Internet 的发展，工具软件大量出现，但是新的软件在易用性方面仍然不能让人满意，致使不少老用户对一些新的工具都望而却步，同时出现了越来越多的未经训练的计算机用户，随着手提计算机和交互式电视的普及，这类用户的数量还会增加。这种形势导致开发者和用户都在寻求更有利的用户帮助手段，Agent 技术可以在此扮演一个重要的角色，它可以代表用户完成工作，帮助人记忆，分析复杂的数据，通过不断的学习改善自身的能力，了解用户的兴趣，习惯和爱好，提供更有效率的服务。Agent 可以屏蔽软件使用的复杂性，帮助用户完成工作。

(2)信息资源及其内容正在急剧增加，Agent 可以帮助进行数据挖掘和确定有价值的资源。

(3)越来越大的网络带宽意味着数据正以更快的速度出现在用户面前，但是用户的工作时间仍然是一个常数，Agent 可以帮助管理这种流动、只将最需要的信息发送给用户。

(4)Internet 的迅速发展正在使它成为一个非常复杂的计算环境，这意味着正在从简单的网络连接方式(比如，终端对主机，或者客户对服务器)变成一个复杂的多服务器和服务互相连接的高速网络，可用的服务很多、但是它们时刻都在变动。"信息高速公路"呈现的将是基于计算机任务和服务的组合爆炸，新的环境的复杂性需要计算机成为一个具有智能的、主动的、人格化的合作者、提供更自然的手段表达用户的意图，能够理解和转换用户请求，寻找和利用已有的服务满足用户的需求。

(5)计算机正在进入人们的日常生活，比如新闻和信息搜索与过滤，电子邮件处理、会议日程安排、社会交往、图书电影音乐等的选择等都越来越多地依赖计算机。计算机提供的服务方式还落后于需求、当前的交互模式要求用户明确设置所有的任务，然后监控所有的事件，用户需要承担比较重的工作负担，这种状况亟待改变，Agent 是采用人工智能技术对计算机用户提供更灵活，更人性化的服务、允许人给计算机分配工作，然后由计算机自动执行、它会大

*)国家自然科学基金资助项目69773027、

大改善计算机的使用方式和作用。

(6)关于 Agent 已经不只限于理论研究。在 Internet 上出现了大量的实验系统，有众多的厂商提供基于 Agent 的服务，几乎覆盖了 Internet 的各种应用领域，为 Agent 技术的应用奠定了基础。Agent 技术在管理、协作、商业、桌面应用系统、信息访问与管理、消息传递、移动访问管理、网络管理和用户界面等应用领域也会有广阔的应用前景。

我们认为，在这样的背景下深入研究 Internet 环境中 Agent 的概念、结构、技术与应用对于我国的信息产业有着特别重要的意义。

2 关于 Agent 的概念

关于 Agent 的定义有很多的讨论和争论。有的 Agent 的定义过于广泛，几乎包含了所有计算机硬件和软件的属性，这样显然不利于 Agent 的研究。由于在 Agent 的必要特征方面尚未达成共识，以至于 Agent 一词在不同的环境中常常表示不同的含义。对于 Agent 的深入研究又需要我们将 Agent 软件与其它软件区分开来，这对于研究 Agent 的技术实现是有价值的。在许多的文献中认为 Agent 一词意味着自主性(autonomous)和智能性(intelligent)，Franklin 和 Graesser 在分析了各种 Agent 的特征之后，避免了对智能性一词的纠缠，给出了这样一个描述[1]："自动 Agent 是一个处于一个环境之中并且作为这个环境一部分的一个系统，它随时可以感测这个环境并且执行相应的动作，同时逐渐建立自己的活动规划以应付未来可能感测到的环境变化。" Wooldridge 和 Jennings[2]则将"自主"作为它的必要特征，他们关于自主性的定义具有更多的操作语义："具有周期性的动作，自发执行和主动性，Agent 能够采取抢先的和独立的最终有利于用户的行动"。

但是，利用"智能"一词定义 Agent 有三个方面的问题。首先，"智能"与"自主"的含义是什么？目前它们仍然是一些主观的标签，依赖于观察者与 Agent 的交互过程中的判断，目前还没有关于智能与否的测试。其次，这些主观的标签对于设计没有太大的实际意义，"智能"无法作为设计的目标，而只能体现设计的一些属性。第三，存在各种"智能"的定义。所以这个标签不足以将 Agent 从其它声称具有智能的软件区分开来。

自主性似乎比智能性对于区别 Agent 与其它软件更为重要。有时候又用自主性来定义智能性。自主性意味着能够自己创建与实现目标有关的计划，许多 Agent 研究小组都将自主性作为 Agent 概念的核心，但是我们在设计 Agent 系统时需要的是可操作的和客观的自主性定义，关心是否存在对自主性的图灵测试，是否存在离开智能的自发性，它在软件设计上会带来什么不同，这些对于工程研究者更为重要。

用户有时也会利用自主性来描述需求，但是当用户使用自主性和智能性这样的词汇时，其含义可能不止是指某个服务器，某种可移动的代码，有时是指社会和技术方面或者人的概括的特征。而我们则更关心建立独立于领域的一种智能性的结构和通讯机制。

Wooldridge 和 Jennings[2]给出了弱 Agent 和强 Agent 的概念，也可以看成是基本和非基本的 Agent 特征。Agent 的基本特征包括：(1)所有的 Agent 都是自治的(autonomous)，即 Agent 可以控制它自己的活动，当用户将任务委派给 Agent 之后，Agent 无论遇到什么情况，都应该设法独立完成用户的请求；(2)Agent 是目标驱动的，Agent 具有一个目的，所有的行动都应该符合这个目标；(3)Agent 具有反应性(reactive)，Agent 能够感知环境的变化，并且及时地对这些变化做出反应；(4)连续执行性，为了完成用户的委派，Agents 应该能够连续执行，即使当用户离开时也能正常执行。主要的非基本特征包括：(1)Agent 具有社会性(social)，即 Agent 之间存在相互作用和通讯，这种通讯可以是以专有的方式，或者是以标准的方式，比如利用知识查询和操纵语言(KQML)，某些多 Agent 系统，甚至主要由通讯和协作 Agent 构成；(2)Agent 是定制(customized)或者具有某种适应性，可以根据以前的经验改变自己的行为；(3)Agent 是可以移动的，可以从一台计算机移动到另外一台计算机，通过移动 Agent 可以距离它们需要处理的数据更近；(4)Agent 具有某些方面的情绪和人格，但是采用诸如目的(intention)和信念(belief)之类的主观词汇，容易造成 Agent 概念的争议，给 Agent 的设计与实现造成一定的困难。

目前在商业领域，往往把通过 Internet 访问的智能软件叫做 Agent[3]。它们有具体的应用目标，但是显然有不同的侧重点，比如 BargainFinder 和 Cyber Yenta 为用户执行搜索，CompassWare 可以对自然语言进行分析然后执行新闻过滤。MetaCrawler 是一个多线程的 Web 搜索服务，AlphaConnect 则将搜索结果翻译成各种格式。它们基本上是一些搜

索服务，其搜索手段是按照不同的关键词收集元信息，然后存放在一个大数据库中，利用搜索机查找，这里的智能还只是以串匹配为支持手段，主要采取一问一答的使用方式，所具有的自动更新等功能也不过是按照用户规定的模式进行定时更新，这些软件似乎与以往的软件概念没有质的区别，如果只是将以往的服务器称做 Agent 则会造成概念的混淆。

我们认为有两类 Agent 将会对 Internet 环境中的应用发挥重要作用，这就是类型 Agent 和移动 Agent。

3 类型消息 Agent

Genesereth 和 Ketchpel 从系统工程的角度给出了 Agent 一个较为客观的描述[2]，"软件 Agent 利用共享的外部语言、内部语言和 ontology 进行通讯"。接近 Wooldridge 和 Jennings 的弱 Agent 概念，这种观点倾向于将 Agent 作为一种集成技术，按照 Genesereth 的定义，为了完成任务，Agent 之间需要使用共享的消息协议交换消息，比如 KIF、KQML 和 ontology，这些消息协议的消息语义独立于应用程序，因此被称为类型消息(Message-typed)Agent。单个 Agent 可以采用自己的概念结构，如果概念结构不同，它们之间就难以交互，因此提供理解和联系不同概念的手段就非常重要。ontology 就可以用来提供这种支持，利用公共的 ontology 能够支持各个协作 Agent 使用各自不同的术语，从而强化消息交换。

工程领域的 Agent 研究已经有比较长的历史[1]，比如 COSMOS 系统、MACE 应用 Agent、CIFE Agent 项目、STRAND 的 Agent 分析系统和 Next-Link Agent 构架等。这些都是类型消息 Agent 或其变种，它们不同程度地利用 KQML，KIF 和 ontology，支持不同领域的工程师之间以及工程师与计算机之间的协作。这类工程 Agent 集中在特定的工程项目中，不使用 Web 界面，其中一个重要的原因是它们需要对等的协议，同时 Java 出现较晚，未能为这些项目所用。

一些协作设计试验[4]，比如 MADEFAST 取得了成功，其中使用了基于 Internet 的协作工具，并且证明 WWW 是最有效的公共技术，他们在6个月中建立了一个导弹搜寻原型系统[4]，但是没有使用 Agent 技术，问题来自信息结构，因为基于 HTML 的标记描述的只是格式，而 Agent 需要的是基于任务的语义和任务级的计算结构，Agent 不能够直接读取和产生 HTML 页，对于结构问题有两条可行的解决途径，一是转换 Web 页到关系数据库，另外是扩展 HTML 标记记录语义信息或者增加指向 ontology 的标记，以支持 Agents 对 Web 页的访问。

为了集成 web 与 Agent，Greg Olsen[4]曾经编写 CGI 程序实现 Web 页与 KQML 消息之间的转换，但是这种简单的转换有一个缺陷，就是一个请求一个答复，来自 Agent 的多个消息被丢失，在 HTML 3.0中增加的 server push 和 client pull 有效地将 HTTP 变成对等协议，对 Agent 通讯很有意义，Rob Frost 的 Java Agent Template(JAT)支持编写可以发送 KQML 消息的 Java Agent，可以允许人通过浏览器与 Agent 交互，但是在 Agent 和浏览器之间需要对等的连接，而且一个 applet 只能打开与产生其服务器的连接，为了给网上的其它 Agent 发送消息，必须编写一个 Agent 路由器，并且一直处于开放状态。

一些非工程性的 Agent 实现了与 web 的连接。比如 MIT 的"Firefly"，是智能化的音乐推荐服务，Webdoggie 则是一个人性化的文档过滤系统，它们在后台自动交换消息，但是对于用户它们仅仅是一个服务器，这里 Agent 成为 web 服务器的低层技术，用户看不到 Agent 的行为和 Agent。

4 移动 Agent

通过在客户和服务器之间传送可执行程序，使程序远程执行的想法久已有之，70年代叫做"远程批作业处理"，在80年代称为"函数发送"(function shipping)和"远程求值"(remote evaluation)，基本的出发点是解决本地运算能力的不足，共享资源和实现分布式系统负载平衡。这些概念都是在比较专用和封闭的环境中提出的。而 Agent 则是一个开放的概念，可以看作对远程调用的扩展，除了代码与数据，每个 Agent 都有明确的执行状态，允许程序在一个节点在某个状态暂停，迁移到另外的节点后继续执行，此外移动 Agent 能够创建子 Agent，有效地分布处理任务，这样 Agent 可以在不同的节点以协调方式执行特定的任务。

在执行过程中，活动 Agent 可以从一个节点转移到另外一个节点，逐渐地完成它的任务，即 Agent 能够暂停执行，然后将自己的执行代码、数据和运行状态传送到网络的其它节点，然后从暂停点继续执行。在它转移过程中，可以启动新的 Agent 来传送需要的信息给客户或者某个子任务。这种方式与以

往的RPC调用有显著的不同。

移动Agent适用于提供复杂服务，比如复杂的Internet过滤和搜索程序，智能消息传递和智能通讯和管理。这类Agent的开发手段主要是通过脚本语言象Save-Tcl，Java和General Magic的Telescript。Safe-Tcl和Java仅允许Agent在Internet上的远程执行，而Telescript Agent可以在活动状态下迁移。

Agent的移动性是很有使用价值的属性，将会影响传统的通讯和服务实现方式，但是它的安全性是最大的问题。Agent程序必须在一个开放和保证安全的环境中运行，由于具有很好的可移动性和安全性，脚本语言引起了大家的重视。移动Agent的性能取决于移动模式和移动Agent的大小。

移动Agent技术带来了Internet环境中新的应用机会[5]：(1)异步与协作处理，移动Agent通过向一个或者多个节点分配任务可以实现动态和并行的计算，这种计算允许连接中断和弱客户计算。(2)服务用户化和可配置化，通过对现有服务的再配置和定制，将Agent作为服务适配器，大大简化了安装。(3)即时服务和电子化交易，利用Agent缓解集中式网络管理的压力，移动Agent可以按时间或者按空间分布管理活动；Agent通过配置用户的通讯环境，信息格式转换和网间互联为更高级的通讯提供支持。

移动Agent可以利用脚本语言实现，在脚本中预先定义它的动作和目标；Agent也可以是一个由目标驱动的过程性程序；Agent也可以由规则驱动，这是更一般的定义Agent目标的方法；此外一些更复杂的嵌入Agent目标的方法包括计划方法，或者由Agent自己随时间改变目标。

移动Agent的关键是允许进程在其它计算机上执行的一种软件结构，这是一种新的软件技术，对它的深入研究有着巨大的现实意义和广阔的应用前景。

5 *Internet*环境中的Agent应用

从使用角度看，Agent具有可以接受委派任务，自动执行和给委托者以帮助的特征。有些Agent对用户是不可见的，这时Agent或者是一个二级委派，或者是作为系统构造时自动执行的一部分。Agent技术在Internet环境中的应用集中在下面一些方面。

信息查找——在查找信息或者网络浏览时，用户可能不知道需要的信息在哪里，Agent可以通过发现和过滤信息帮助解决这个问题。

信息组织——随着信息的增加，整理和组织信息成为一件繁重的任务，借助于Agent可以节约许多整理与组织的时间。

预定事件响应——在现实世界中事件发生以后，需要及时采取行动。比如新软件发布和服务登记等，Agent可以代替人执行一些动作，以便及时响应。

客户帮助——客户帮助系统的任务往往是回答客户在电话中提出的问题，最初这种工作需要训练有素的专业人员通过手工查找手册、客户数据库或者与其它人员协商来完成，现在CD光盘代替了手册，出现了Internet搜索工具。Agent会很好地解决这一类问题，Agent在得知客户的问题后，在本地、光盘或者通过Internet自动搜索相关的数据库，然后回送最合适的信息。

Web浏览器智能——在今天的Internet上，借助于搜索工具也很难找到所需要的信息，搜索者象是在走迷宫，难免进行无谓的重复。借助于搜索辅助Agent记录搜索过程，记住可能忘记的事情，建立搜索者的视图，按照所访问的最频和最新网点排序，它可以根据用户提供的关键词返回到相应的网点，可以通知用户某些网页的更新，也可以脱机下载。在下载之前可以指示相对的速度，如果发现你按照某种特定的模式搜索，它可以直接连接到搜索者要去的地方，不需要回溯。

个人帮助Agent——对于Internet购物者或服务接受者，选择适当的商品目录和排列顺序显得非常重要，个人购物帮助Agent可以帮助购买者比较商品，提醒购买者在某个时间比如生日、周年纪念日去进行购买活动，或者某些感兴趣的甩卖，或者向用户推荐图书、音乐等等。也可以帮助店主发掘某个购买者群体的购买模式，发现商品之间的购买关联关系。

知识工具——支持信息检索和管理，允许用户将各种信息集成在一起，形成有意义的模式，帮助用户检索适当的信息。同时将用户已有的私有数据与Internet数据结合在一起，形成知识库。

知识挖掘——利用表达和模型技术对大量数据进行模式分析，突出感兴趣的模式和特征，在此基础上进行分类，建立关联等等。将多种技术象模糊规则系统，数据过滤和翻译嵌入应用系统，并且按照需要修改模型和数据过滤的标准，对用户完成某些任务

提供导航帮助。

Internet 常规任务——帮助执行或者自动执行 Internet 环境中的一些常规任务，比如文档过滤，发电子邮件，寻找某种标题的文献，进展跟踪，定时提醒等。

Agent 可以是私有 Agent，只供某个用户使用，也可以是共享 Agent，可以供各种用户用来访问应用程序或者数据库，可以由用户手动运行，或者当某种事件发生时自动执行，比如邮件到达，或者在预定的时间间隔内执行。

6 相关的研究问题

除了技术问题，与 Agent 有关的还有法律问题和文化差异问题，比如：(1)认证，如何判断 Agent 真的代表所要代表的人。(2)机密性，如何保证 Agent 维护用户的私有性，如何保证一个 Agent 不为其它用户所用。(3)私有性，当 Agent 为用户服务时，如何保证它按用户所期望的程度维护其私有性。

Internet 环境是一种大众化的服务，因此文化差异、教育水平差异、行业差异、民族差异、个体差异对应用效果会有重大的影响。如何表达和体现种种差异，更是需要仔细研究的问题。因为在不同的背景下，相同的信息会有不同的含义，因此在公共的基本服务和网络软件构架之上，如何对不同背景的用户提供适当的支持，需要各个层面上的通盘考虑。

Internet 系统与已有系统的集成，和其它系统的改造是另外一个重要研究课题，如何将局部服务网络化和利用网络服务扩充已有系统的功能，从数据、处理和用户界面等不同的方面的集成则是必然的趋势和很重要的研究任务。

结束语 如果我们不能区分软件 Agent 与其它软件技术，包括其它的专家系统和基于知识的系统，使用 Agent 这一概念就是无意义的。有人将消息交换作为判别特征，面向对象的技术也使用消息传递来协作完成任务，Agent 与它的区别是使用独立于应用程序的类型消息协议，而面向对象技术并不采用这样的协议，同样这一特征也将服务器和 Agent 区分开来。

关于 Agent 的一个期望是它们应该提供料想不到的有帮助的信息，具有隐蔽性和记忆能力，而不是简单的一次反应。Agent 还应该能够保存要解决的任务状态，对状态的改变进行推理，采取有助于整个任务解决的步骤。这是一个非常现实的期待，但是 Internet 环境中的实现机制还需要研究。无疑 Agent 与知识的利用紧密相连，但是建立一个包罗万象的大规模知识库是不容易实现的任务，于是建立一个通过语言 ontology 在不同的系统之间共享和复用知识的构架是更为现实的途径。

Internet 环境中的 Agent 引起越来越多的关心，许多商业产品目前实际上仍然属于服务器范畴，一方面关于 Agent 特征的含义仍然存在许多争论，另一方面这些特征对于 Internet 的含义还需要进一步研究。如果给软件 Agent 一个过泛的定义对于工程是无益的。我们认为，研究 Agent 的应用和实现更为重要，Agent 技术在 Internet 环境中应该发挥更大的作用。

参 考 文 献

1 Franklin S, Graesser A. Is It an Agent, or just a Program?: A Taxonomy for Autonomous Agents. In: Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag, 1996

2 Wooldridge M, Jennings N. Intelligent Agents: Theory and Practice. Knowledge Engineering Review, 1995, 10 (2)

3 Gensereth M R, Ketchpel S P. Software Agents Comm. of the ACM, 1994, 37(7)

4 Petrie C J. Agent-Based Engineering, the Web, and Intelligence IEEE Expert. December, 1996

5 Chess D, et al. Itinerant Agents for Mobile Computing. IEEE Personal Communications, 1995