一种基于移动 Agent 的主动网络体系结构

张君雁 闵 帆 杨国纬
(电子科技大学计算机学院 成都610054)

An Active Networks Architecture Based on Mobile Agent

ZHANG Jun-Yan MIN Fan YANG Guo-Wei
(Computer Science Academy, University of Electronic Science and Technology, Chengdu610054)

Abstract This paper proposes an active networks architecture which supports both integrated and discrete operation model based on mobile Agent technology. Mobile Agent is the component for customized function transferring, active node provides software application layer, and Agent server processes mobile Agent specific customization. For development and deployment of specific application protocol, this paper also proposes an abstract protocol framework and a protocol loading mechanism to enhance network performance.

Keywords Active networks, Mobile Agent, Agent server, Architecture, ATP protocol

1. 引言

随着 Internet 的发展,网络应用和网络用户的行为更加多样化.流行的 TCP/IP 网络体系结构已经不能适应网络通信技术发展和 Internet 服务拓展的需要.然而现有的网络体系不可能被完全替代,一种可行的方法是对其升级并保持向下兼容,主动网络[1]是一种较好的途径.通过对互操作层次特性做根本性变化,主动网络能提供一个软件可编程的范例,允许中间网络节点完成动态网络定制,避开冗长的标准化过程,实现网络技术的快速发展和服务应用的快速引进,并改善网络计算能力和功能。

创建主动网络有两种主要方法[1]:离散法和集成法.此外,也有人将两种方法混合使用.但这些方法只适用于传递压缩的程序,不够灵活[4].本文使用移动 Agent 建立主动网络体系结构.移动 Agent 技术支持封装机制,代码传递机制,程序嵌入和执行,这些特性使得移动 Agent 能异步自主执行,动态适应,减少网络通信量,并能提高网络健壮性和容错能力,因而适用于构建主动网络[5]。

本文介绍的主动网络体系结构非常灵活,能提供快速的服务引入.在特定应用协议的开发和配置方面,结合了离散法和集成法的传送模型,使得移动 Agent 可以将网络应用和服务快速、有效地引入.移动 Agent 可以是一个完全的 Agent,能携带特定的主代码对象和附属部分一起跨越主动网络;也可以是一个简洁的 Agent,和某些服务 Agent 联系在一起间接地完成定制.原型使用了具有可信赖的认证的"第三方"作为服务提供者以促进服务的配置,简化了服务传播机制。

2. 技术背景及与相关技术的比较

2.1 移动 Agent 技术和主动网络之间的关系

当前许多主动网络体系结构使用了和移动软件 Agent 技术非常相近的代码移动范例[5].而主动网络在协议封装,以及服务定制、配置和维护方面更为通用化.二者的一个本质区别是:主动网络使用了网络分层处理的概念;而移动 Agent 系统是作为应用程序运行的.一般的移动 Agent 系统都设计成能支持"Agent 迁移计算"以构建一个普遍存在于 Internet 中的计算环境.相反,主动网络的最初目的是使网络通信更便捷,最终提供一个灵活、可动态定制、可编程的网络体系结构.移动 Agent 范例提出:将网络看成是多 Agent 友好环境,而将 Agent 看作是从一处迁移到另一处并执行特定用户任务的可编程实体.而主动网络将网络概念化成一个能执行任何计算的主动节点的集合和一个携带程序的主动包的集合.从这个观点上看,移动 Agent 可以被看作是一类特殊的主动包,传统网络中和移动 Agent 兼容的节点可看作是一种特殊类型的主动节点[6]。

2.2 主动网络研究

目前,一些研究者为主动网络定义了一个通用体系结构[10],其中包括使多个组件能互操作的执行环境.使用主动网络主干(Active Network Backbone,ABONE)测试执行环境并实施应用.在实施这一通用结构中的主要问题是在可用性,灵活性,安全性和性能方面之间的均衡。

由于篇幅限制,本文下面仅列出部分相关工作:MIT ANTS[20]的特色在于定义了一种由 JAVA 编程的封装,它能传送专门的定制和有效数据.具有 JAVA 虚拟机(JVM)的 ANTS 节点为封装的解释和执行提供 API.ANTS 中的封装能安装声明,并调用预载的类.Pennsylvania 大学的 SwitchWare[4]在分层 SANE 之上,使用 PLAN[9]脚本语言,实现了强大的资源管理和安全保障功能.在 SwitchWare 中,不允许在网络节点中安装状态.哥伦比亚大学 NetScript[10]为中间网络节点的编程功能建立了基于 Agent 的中间件,其中使用了两种主动网络模型,一种是引入新的服务作为控制而不是用作数据传送;另一种是由网络管理 Agent 而不是由所有用户引入服务.BBN 的 SmartPacket[3]主要通过 Sprocket 和 Spanner 语言设计技巧,将主动网络运用于解决网管问题.将 Sprocket 程序编译成 Spanner 代码能获得非常小的目标程序.Georgia Institute of Technology 的 CANES[11]和 Liane[12]力图运用精简的可编程模型构建动态的、可信赖的服务,并通过包头中的控制信息给出一个可靠的、预定义服务库的集合,其优点在于设计者可以对需求的安全性进行分析和限制.与 CANES 类似,Washington 大学的 DAN[13]设计了一种 DAN 包,其中包含了有关功能和参数标识符的有限序列.DAN 的

目标是建立大量代码服务器的拓扑结构，以简化代码分布。

本文的工作是对其他几个主动网络的补充，利用基于JAVA的移动Agent作为本文主动网络的构件，在具有一般主动网络优点的同时，更为灵活、安全。

3. 基于移动Agent的体系结构

3.1 体系结构概述

如图1所示，主动网络体系结构包括一组能通过点到点或共享传输介质方式跨越LAN或WAN互连的主动节点。基于程序编码和执行环境的协定，Agent服务器用于处理传送特殊定制的移动Agent，并为它们提供严格限制的、透明的执行环境。这样，特定的服务就能依照移动Agent的传输，从一个主动节点到另一个主动节点地在网络中展开。此外，还使用动态代码加载机制，在需要的时候，远程加载涉及到协议代码对象。如果有的网络元素不支持该协定，它就使用默认的协议处理这些消息（即移动Agent），就可以实现主动网络节点和传统网络兼容。

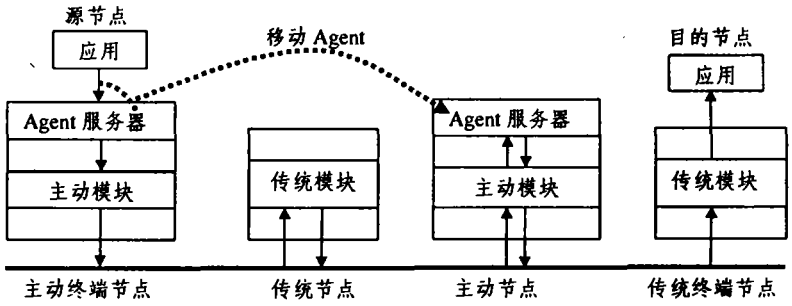

图1 基于移动Agent的主动网络

3.2 移动Agent逻辑格式

在原型中，主动网络中的移动Agent包含专门的程序和有效数据。移动Agent内部的数据采用了可定制的结构，移动Agent的计算状态是可移植的。如图3顶部所示：移动Agent必须包括一个Agent头，以将他们和其他网络消息区分开来。Agent信息域包括移动Agent属性，以及用于该Agent在主动网络的主动节点上执行的协定或信息。Agent特殊定制域包含了在主动节点上执行的特定代码对象。Agent有效数据域包含了传送的数据，或用于迁移计算的计算结果。

3.3 Agent服务器

如图2所示，主动节点的运行环境在逻辑上可以分如下两层：低层是物理网络节点，高层是Agent服务器。Agent服务器提供的Agent与节点的接口是移动Agent与网络定制之间联系的真正实体。通过这种一致的接口，移动Agent能间接地访问本地资源，实现计算和通信。

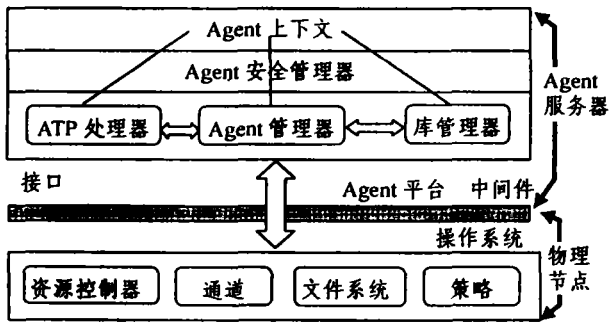

图2 主动节点运行环境

Agent上下文 是Agent的临时执行环境。限定移动Agent只能通过该接口使用Agent服务器的原语来获取本地资源。

Agent安全管理器 用于检测任何Agent上下文以外的操作，滤除恶意的Agent行为和有害交互。本文对所有移动Agent上的系统资源进行统一控制。

ATP处理器 从输入流中区分出主动消息，并将其传递给评估过程。同样，ATP(Agent Transmission Protocol，Agent传送协议)处理器也传递按ATP形式的主动消息封装好的移动Agent。

Agent管理器 记录本节点中所有的移动Agent和已初始化的服务Agent，并管理本节点的协议配置。

库管理器 存储特定协议的"软状态"和用于Agent之间协调和协作的信息。

3.4 应用/服务配置方案

抽象协议结构围绕协议标识符，服务Agent和移动Agent信息协议定义，自变量协议定义，代码对象协议定义等进行构建。任何新协议的开发都必须扩展协议的抽象，以使得所有协议有统一结构。因此，在本文的主动网络环境中，主动节点能操作、管理、维护(OA&M)那些已配置的协议。

本文原型利用"第三方"——服务提供者来辅助协议配置。主动节点禁止特定应用代码的随意注入。为确保安全，仅将定制权授予少数认证用户。第三方即特殊应用代码库，能对"协议代码请求"做出响应。

4. 原型设计

4.1 移动Agent封装

图3描述了Agent封装的语法结构。Agent头用于确定ATP通信协议。Agent数据域对网络设计者开放，使他们可以自己定义。Agent信息域存放用于管理和控制的信息。Agent特殊定制域存储引导类和所有必需的其他类。Agent有效数据域存储字符类型的中间/最终计算结果。

4.2 Agent服务器的设计

本文的原型提供了运行在主动节点上的作为软件应用层的移动Agent服务器，以代表移动Agent支撑特殊定制。

Agent服务器功能 首先，Agent服务器能解析收到的主动消息，从而产生Agent，之后为之提供临时执行环境。其次，Agent服务器能将该移动Agent代码对象、自变量、附属的和中间结果封装成一个主动消息，并将其分派到下一个主动节点。第三，Agent服务器可以将移动Agent传递给相应的服务Agent。最后，Agent服务器能依照来自管理服务器的

ATP 控制消息应用控制操作。

Agent 服务器示例 当给定主动网络的示例后，主动节点的 Agent 服务器首先初始化存储在本地文件系统中的原型属性，并按照本地已经注册的协议将服务加载到特定协议的移动 Agent 中。最后，Agent 服务器触发 ATP 处理器处理接收到的 ATP 消息。

移动 Agent 在 Agent 服务器上的执行 出于安全性的考虑，本文规定：仅允许移动 Agent 通过 Agent 上下文接口间接地使用 Agent 服务器提供的服务。这样，特定 Agent 定制被限制在临时 Agent 上下文内，禁止在该范围内进行任何直接操作。

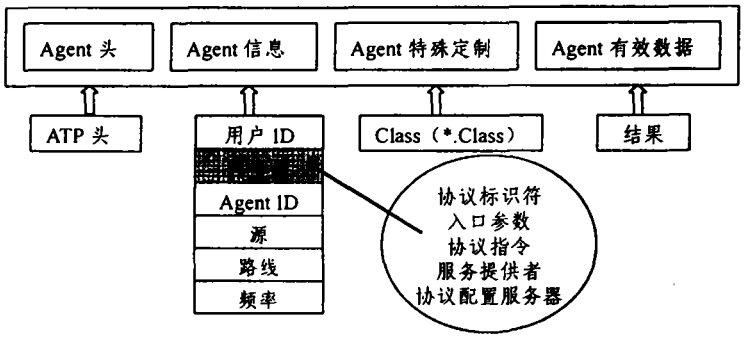

图3 移动 Agent 封装

4.3 协议配置方法

本文的原型设计了一种数据结构，能保持协议的抽象信息和从属信息，如图4。

```
public class Protocol{
    String User_id;//用户名
    String Protocol_ID;//协议标识
    String protocol_Status;//协议所处的状态
    String Class_Cluster;//类簇，记录所有与协议相关，用于动态配置协议的类名
    String Agent_Cluster;//Agent 簇，记录所有特定协议的移动 Agent 名
    String Service_Agent_Name;//服务 Agent 名，记录该协议的服务 Agent 名
    Vector Arguments;//向量，用作存储所有协议相关的自变量或参数的临时库
}
```

图4 协议信息数据结构

为了能最终实现动态协议配置的目标，原型使用了可信赖的服务提供者，使得协议配置更容易。服务提供者的位置是默认指定的，也可以灵活地在 Agent 数据域的自变量向量中指定位置。

本文的设计能支持在一个 Agent 服务器中同时存在多种协议。此外，具有不同专门定制的移动 Agent 在一个主动节点中可以配置不同的协议。为了有效地维护已配置的协议，Agent 服务器使用了协议表(如图5)，其中记录了在一个主动节点中，为了管理和控制协议的所有与配置协议相关的信息。

用户 ID	协议 ID	状态	服务 Agent	配置时间	更新时间

图5 协议表

当 Agent 服务器初始化时，首先将已配置的信息存储到协议表中。当该主动节点有新的协议配置时，相关信息也被存储在该表中。而且，给可信赖的第三方授权，让其代表移动 Agent 适应已经配置在主动节点上的协议。

结论 本文基于移动 Agent 技术提出了一种主动网络体系结构，其主要特点在于：这种体系结构能同时支持集成法和离散法，并在移动 Agent 和特定的服务 Agent 之间实现互操作，还引入了动态协议配置机制以推动网络的发展。文中设计了 Agent 封装形式，主动网络的体系结构，Agent 服务器中包含 Agent 的定制处理，以及结构原型，这些都说明了移动 Agent 是非常适合构建主动网络的技术。

参 考 文 献

1 Tennenhouse D L, Wetherall D J. Towards an Active Network Architecture. In: Proc. of Multimedia Computing and Networking, Jan. 1996

2 Wetherall D J, Guttag J V, Tennenhouse D L. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. In: Proc. of IEEE OPENARCH'98, April 1998

3 Schwartz B, et al. Smart Packets for Active Networks. In: Proc. of IEEE OPENARCH'99, March 1999

4 Alexander D S, et al. The SwitchWare Active Network Architecture. IEEE Network, May/June 1998. 29～36

5 Fuggetta A, Picco G P, Vigna G. Understanding Code Mobility. IEEE Transaction on Software Engineering, 1998, 24(5)

6 Psounis K. Active Networks: Applications, Security, Safety, and Architecture. IEEE Communications Surveys, First Quarter 1999

7 Wetherall D J, Legedza U, Guttag J. Introducing New Internet Services: Why and How. IEEE Network, May/June 1998. 12～19

8 Lange D B, Aridor Y. Agent Transfer Protocol - ATP/0.1 draft number: 4. In IBM Tokyo Research Laboratory. http://www.trl.ibm.co.jp/aglets/atp/atp.htm, March 1997

9 Hicks M, et al. PLAN: A Packet Language for Active Networks. In: Proc. of the Intl. Conf. on Functional Programming (ICFP'98), 1998

10 da Silva S, Florissi D, Yemini Y. Composing Active Services in Netscript. In: Proc. of DARPA Active Networks Workshop, March 1998

11 Calvert K, Zegura E, Sterbenz J. CANEs: A Modest Approach to Active Networking. In: Proc. of IEEE Computer Communications Workshop, Sep. 1997

12 Bhattacharjee S, Calvert K, Zegura E. LIANE-Composition for Active Networks. In: Proc. of IEEE Computer Communications Workshop, Sep. 1998

13 Decasper D, Plattner B. DAN: Distributed Code Caching for Active Networks. In: Proc. of IEEE INFOCOM'98, March 1998