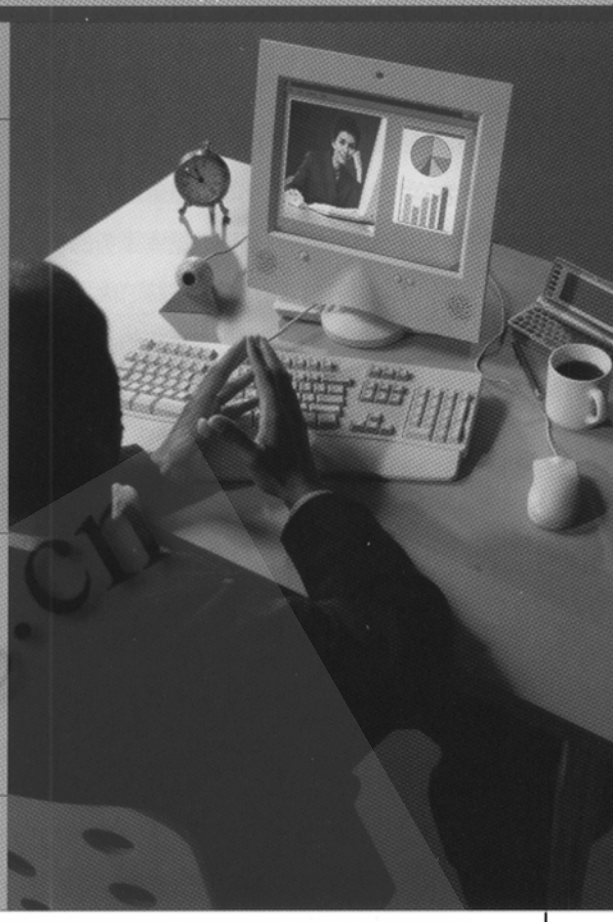
The Integration of Agent System and Legacy Software

Agent数据库系统集成设计与实现



摘要：本文简介Agent软件集成规范，阐述Agent系统与非Agent软件集成的设计与实现技术，介绍基于Grasshopper 平台应用面向Agent的编程技术实现Agent数据库系统集成。

关键词：FIPA 规范　Agent 系统　集成

范宝德　姜远明（烟台大学计算机学院　264005）

1 引言

软件集成可以说是应用程序的集成，软件Agent技术的发展为解决复杂的、动态分布式智能应用提供了一种新的计算手段。Agent系统能够更好地体现人类的社会智慧，具有更高的灵活性、适应性，能更好地满足开放的、动态环境的需要，因而倍受关注。

开放的Agent系统是一个能够兼容MASIF和FIPA规范的多Agent 体系，在大多数重要应用中，Agent系统可能要接受诸如其他Agent系统和传统软件（非Agent）等实体的服务。然而，提供服务的非Agent软件会越来越多，要使所设计的Agent真正实用，必须能够与现存的Agent及非Agent软件系统相连，并能够控制他们，如：数据库系统、WEB浏览器、语音合成软件等。我们把非Agent软件集成到Agent领域称为Agent软件集成。本文将讨论Agent数据库集成。

2 Agent 软件集成规范

FIPA提出Agent软件集成标准规范，但并不强制使用任何一种API或分布式计算技术，而是在Agent通信层（Agent之间信息交换的形势和内容方面）上讨论集成技术。为了支持这一点，FIPA定义两个新的Agent角色：ARB Agent和WRAPPER Agent。如图1 示。

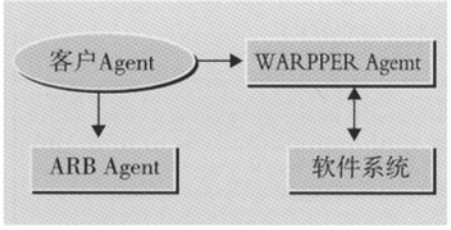


图1 Agetn与非Agent 软件集成结构

（1）这里的ARB和WRAPPER是根据Agent的功能定义的。

（2）ARB Agent：提供一组软件描述，客户通过查询ARB Agent了解可用的非Agent软件服务。

（3）WRAPPER Agent：允许客户Agent与软件描述所唯一定义的非Agent软件系统相连接。客户Agent可以对WRPPERAgent发送消息，并且通过它调用软件系统。WRPPERAgent为Agent与非Agent软件系统提供一种单一通用的方法，FIPA 还为每一个Agent角色定义ACL消息。如图2所示。

3 Agent 数据库集成设计与实现

Agent数据库集成设计与实现是基于Grasshopper1平台之上进行

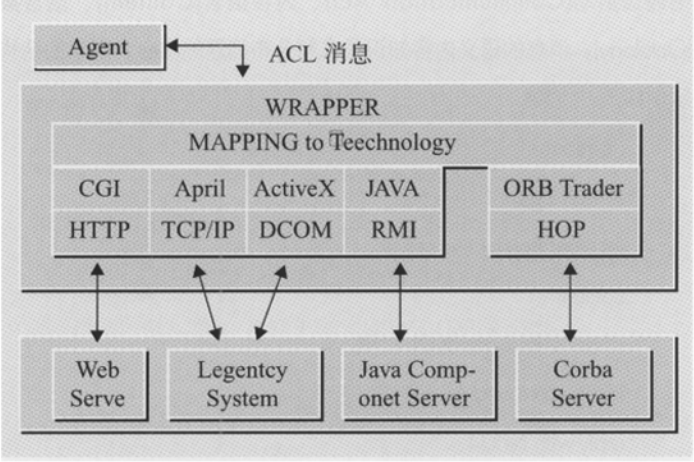


图2 WRAPPER 的参考模型

的，该设计采用面向Agent的设计方法，这里提出了一种实现方案，即：系统的功能由两个Agent角色ARB和WRAPPER协作完成。ARB Agent提供了在查询库中注册并管理数据库服务的功能，即提供一组软件（数据库）服务描述，包括服务名称、类型、语言集等，客户通过ARB Agent了解哪些数据库服务是可用的。WRAPPER Agent提供Agent到数据库的桥接功能。图3描述了Agent数据库系统的集成结构。

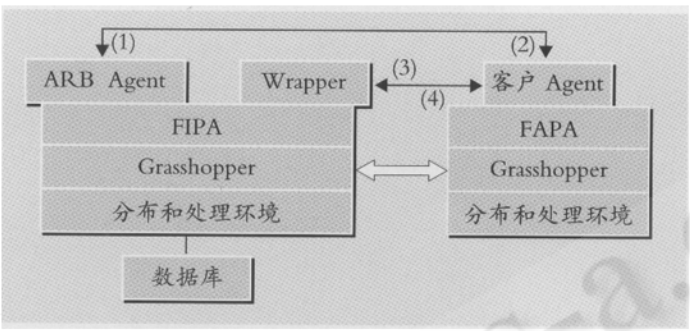


图3 Agent 数据库集成结构图

图中：（1）客户Agent询问是否有满足要求的数据库；（2）ARBAgent做出回答；（3）客户Agent通过WRAPPER Agent请求数据库操作；（4）WRAPPER Agent返回操作结果。

3.1 Agent 数据库集成的通信

使用多Agent系统求解分布式问题，系统中的Agent之间必须能够相互通信协作，通信是协作的基础。本设计采用消息传递的通信方式，这是实现柔性复杂协作策略的基础。使用规定的协议，Agent能够通过消息的交换建立通信协作机制。自由消息内容的格式提供柔性的通信能力。

一个Agent发送特定的消息到另一个Agent，消息在两个Agent之间直接交换，执行中没有缓冲。为了执行协作策略，通信协议必须明确规定通信过程、消息格式并选择通信语言。Agent通信语言(ACL)是以语言行为为基础的，相关的Agent必须知道通信语言的语义。ACL消息包括通信活动(Communication Act)、内容语言(Content)、语言集(Ontology)。消息的语义内容知识是求解分布问题的核心。消息的结构示例如下：

```
(inform
:sender agent-1
:receiver hpl-auction-server
:content
(price (bid good02) 150)
:in-reply-to round-4
:reply-with bid04
:language sl
:ontology hpl-auction
)
```

这是一个来自agent-1的ACL消息，消息内容是告诉货物的价格。在这个消息中，该ACL的speech act 是inform，内容是(price (bid good02) 150)，所采用的ontology由符号hpl-auction 标识，消息的接受者由hpl-auction-server 标识的Agent，告诉使用的语言为s1。:content 的值是内容层，:in- reply- to、:reply-with、:sender和 :receiver的值构成了通信层，speech act的名字、:language 和:ontology 关键字构成了消息层。

在传送格式上，消息被表示为S表达式（s-expressions），消息的第一个要素是一个命令，由它标识正在进行的通信活动，它定义了消息的主要语义，它的后面跟着一个消息参数序列，每个参数都以：参数关键字开始，在“：”与参数关键字之间不留空格。这个结构的消息被串行化为一个字节流，由消息传送服务传送，然后正在接收的Agent负责译码字节流，从语法上分析消息成分并且正确处理它。

Agent数据库集成通信中，两个Agent角色ARB和WRAPPER都有自己的ontology，集成系统必须能解释角色之间的通信消息，使角色了解通信的内容。

下面讨论ARB和WRAPPER的设计与实现。

3.2 ARB Agent 的设计

ARB Agent由消息解析、执行请求并响应 两部分组成。

（1）消息解析：就是对接收到的消息进行解析，以使ARB Agent明白客户Agent要求其完成的任务。例如有消息内容如下：

```
<content>
  <action name='ARBAgent'>
    <register_software>
      <service_name>database-001</service_name>
    </register_software>
  </action>
```

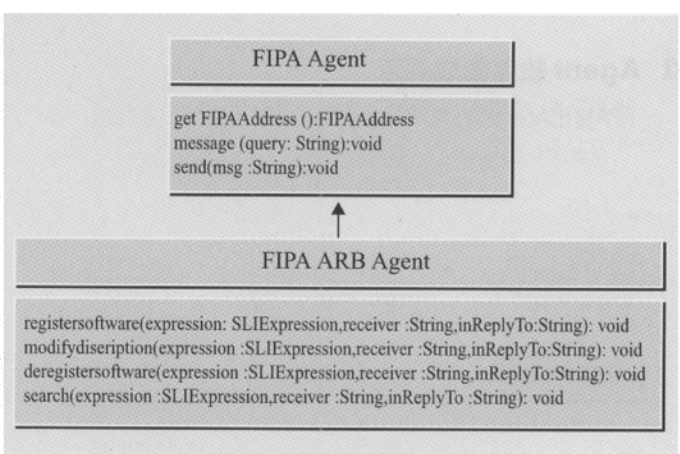


图4 ARB 的类图

</content>

本设计采用IBM XML解析器，得知这条消息是客户Agent向ARBAgent请求注册数据库服务。

（2）执行请求并响应：执行客户请求其完成的任务，任务执行完成之后，会就执行的结果反给客户一个应答。

我们将ARB Agent设计为FIPA Agent类的子类，用UML符号表示如图4。

message()对接收到的消息进行解析，之后根据解析结果执行具体的任务，registersoftware()进行软件注册，modifydiscription()修改注册的软件信息，deregistersoftware()注销已注册的的软件信息，search()查询注册的软件信息。

3.3 WRAPPER Agent的设计

WRAPPERAgent提供了Agent系统到数据库的桥接功能，它的功能实现主要通过三部分来完成：

（1）消息解析：对收到的消息进行解析，他实质上起到映射的作用，也就是将消息映射成对不同数据库的操作，WRAPPERAgent特有的语言集为init、invoke、close、store、restore、software_subscribe、software_unsubscribe、suspend、resume、achieve，其实本Agent的解析主要目的是使本Agent能够明白这些特有的语言集。

（2）执行请求并响应：对消息进行处理并给客户Agent发送响应消息。

（3）连接：实现到数据库的连接，此处采用CORBA技术，用database.idl定义对数据库操作的接口。本质上就是将对不同的数据库操作连接到软件总线上，然后客户方就可以随意调用。这里同样将WRAPPERAgent设计为FIPA Agent类的子类，用UML符号表示类图关系如图5所示。

WRAPPERAgent包括的功能为：

当接收到init请求时，要根据参数service_discription 和content_expression对service_discription所描述的软件进行初始化；当接收到invoke请求时，根据参数operation 对参数service_instance_id指定的软件系统进行操作；当接收到close请求时，关闭与参数service_instance_id指定的软件系统的连接；当接收到store请求时，将参数service_instance_id指定的软件系统的当前状态存储起来，并将当前的状态state-id发给客户Agent；当接收到restore请求时，重新存储参数service_instance_id指定的软件系统的state-id状态；当接收到software_subscribe请求时，确定谓词（subscribed <service-instance-id> <event-name>），如果执行失败，WRAPPERAgent会根据不同情况反回不同的失败消息“ not-valid-service-instance-id”、“service-suspended”、“ unknown-event-name”。如果事件event-name发生，WRAPPERAgent会通知预定该事件的Agent；当接收到software_unsubscribe请求时取消谓词（subscribed <service-instance-id> <event-name>）；当接收到suspend请求时，挂起由service-instance-id指定的软件服务；当接收到resume请求时，重新启动由service-instance-id指定的软件服务；当接收到achieve请求时，设置软件系统service_instance_id与WRAPPERAgent相关的谓词为真。WRAPPERAgent对所执行的操作都会根据不同的情况返回结果信息。

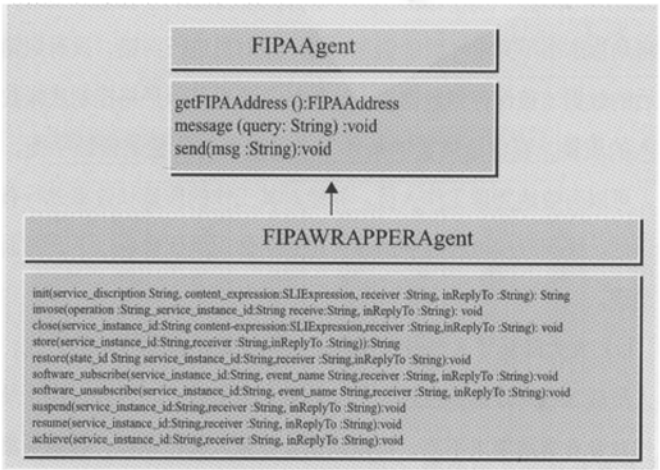


图5 WRAPPER的类图

4 小结

Agent 数据库集成的实现是在Grasshopper平台上的FIPA插件之上，实际上就是在Agent环境中实现的一种应用，也就是Agent软件集成处于应用层。系统的实现采用Java语言，Agent之间的通信采用XML和ACL。系统核心功能的实现是由两个Agent协作完成，即ARB Agent和WRAPPER Agent。ARB Agent的实现类是FIPAARBAgent.java，WRAPPERAgent的实现类是FIPAWRAPPERAgent.java。

参考文献

1 FIPA 97 Specification 1: Agent Management。

2 FIPA 97 Specification 2: Agent Communication Language。

3 FIPA 97 Specification 3: Agent Software Integration。

4 Grasshopper: http://www.ikv.de/products/grasshopper/。

5 徐振宁 等，Ontology建模方法研究 [J]，计算机科学,2002 vol129 no.1。