知件可插拔机制的设计与实现

潘明钢,张楚才,钟 维
(湖南师范大学 数学与计算机科学学院,湖南 长沙 410081)

摘 要:2005年知件的概念被提出,随后如何构建知件成为了研究热点。运用知件的形式,可以把软件中含有知识的部分分离出来,使软件和知件成为两个不同的研究主题和两个不同的产品,但在知件的使用中,如果更换了知件,导致客户端也要重新编写,这样增加了客户端的工作量。针对此问题,提出了知件的静态和动态可插拔模型,从知件的接口语言来研究知件更换的同时不引起客户端的重新编写,从而实现知件的即插即用的效果。最后在Eclipse平台进行了仿真实验。

关键词:知件;知件IDL;IDL解释器;知件动态模型
中图分类号:TP181 **文献标识码**:A **文章编号**:1673-629X(2015)05-0091-04
doi:10.3969/j.issn.1673-629X.2015.05.022

Design and Implementation of Knowware for Pluggale Mechanism

PAN Ming-gang,ZHANG Chu-cai,ZHONG Wei
(College of Mathematics and Computer Science,Hunan Normal University,
Changsha 410081,China)

Abstract:Since the concept of Knowware was put forward in 2005,and the study of how to construct the Knowware has become the research hot spot. Use the form of Knowware,could separate the software with knowledge,so software and Konwware become different research's themes and products. But in the use of Knowware,if replacing the Knowware,can result in client being rewritten,so it will increase clients workload. In order to solve this problem,propose the model of dynamically and statically pluggable Knowware,and study the replacement of Knowware,at the same time will not cause client rewritten,achieving Knowware's plug and play. Finally,the simulation experiment is carried out on the platform of Eclipse.

Key words:Knowware;Knowware IDL;IDL interpreter;Knowware dynamic model

0 引 言

2005年,中科院院士,计算机科学家陆汝钤参加“计算机科学前沿高端学术论坛”,作了题为“知件和知件工程”的学术报告,提出了知件的概念。知件就是独立的、计算机可操作的、商品化的、有完备文档的、可被某一类软件调用的知识模块[1-4];2007年,澳门科技大学的丁利亚教授发表了《知件系统的设计和发展》一文[5-7]。提出了知件系统由五部分组成,包括编辑界面、测试者、知识描述语言、基于知识生产的仓库和安装者;2008年,陆院士发表了《从基于知识的软件工程到基于知件的软件工程》一文,简述知件工程的3个生命周期模型,分别为熔炉模型、结晶模型、螺旋模型[1]。他们只是研究了如何去构建一个知件,并没有提到如何去使用这些构建出来的知件,文中在此基础上,提出了一个知件的动态模型,来满足用户的即插即用。

1 静态模型

1.1 静态模型结构

从图1中,可以清楚地知道静态模型由IDL文件、IDL编译器、接口存根、ORB(对象请求代理)、接口框架、知件和用户组成,因为关键技术是CORBA技术,所以主要在此基础上进行二次开发应用。基于文中的字数有限,每一个部分不做详细说明。

1.2 接口存根

IDL编译器[8-10]把IDL文件编译成两个文件,其中一个就是接口存根,接口存根是连接用户与ORB的桥梁。接口存根主要是对外的接口,用户通过接口存

收稿日期:2014-06-11 **修回日期**:2014-09-16 **网络出版时间**:2015-02-23
基金项目:湖南省教育科研项目(10C0941,11C0812);湖南师范大学青年基金项目(60902);湖南省研究生科研创新项目(CX2014B220)
作者简介:潘明钢(1989-),男,硕士,CCF会员,研究方向为知件、知件接口;张楚才,讲师,硕士,研究方向为语义网、情境感知。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150223.1239.035.html

根可以知道知件中拥有哪些服务供用户使用,因为接口存根相当于接口,只有一个函数的说明,具体的服务被封装在哪里,不是很清楚,这样为服务的安全性提供了保证,很好地实现了说明和代码的分离。

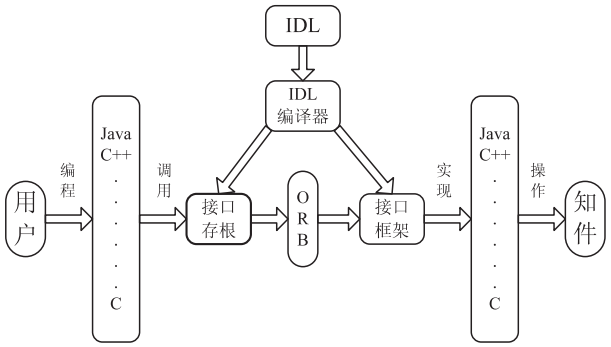


图 1 知件静态模型

1.3 用 户

用户就是服务的使用者,服务中是用很多对象来实现的,用户可以访问这些对象的引用或者调用它们的操作来达成自己的目的。用户只需通过接口存根了解这些对象的逻辑结构和对象拥有的方法。虽然一般把用户作为初始化的对象请求或任务,但是一定要意识到,对于某些对象,也可以被看作是实现用户功能的一部分。例如,实现一个对象可以看作其他对象中的用户。

在对象和接口存根[11-12]中,需要用户从语言映射角度来对待,把用户提升到程序员级别上来。用户应该拥有最好的可移植性,并且不需要去修改任何源代码,就可以在任意 ORB 上运行,而且 ORB 具有实现需求的接口对象实例和支持需求的语言映射。用户根本不要去了解对象是用什么语言实现的,也不需要知道对象使用什么对象适配器来实现或者不同适配器之间的访问机制。

1.4 接口框架

接口框架也是 IDL 编译器编译得到的另一文件,它是一个对内接口框架,相当于在服务提供端使用的接口,在对象实现中需要用到接口框架,对象实现就是实现接口框架的内容,而接口框架中只是一个接口定义的说明,并没有具体实现的方法。所以,每个对象实现中的操作都要与接口中的方法连接起来使用。而接口框架可以看作是“对象实现”编程中编写操作的“目录”,通过里面的内容,ORB 就能够查询到有关的操作编程,激发或调用对象的操作。接口框架在 ORB 中主要完成以下功能:

(1)查询请求相关的具体方法在对象实现中的代码实现;

(2)把请求相关的操作和方法中的参数进行分组,通过客服端把它们传送到服务端转换成对象实现语言所支持的格式;

(3)把响应中的格式转换成客户端代码的编程语言所能支持的形式。

通过 IDL 编译器,可以从 OMG IDL 定义接口中获得接口框架。这个接口框架中的每个操作和方法都会与相应的对象实现中对应的实现连接起来。实际上,可以认为:接口框架中明确说明了程序员需要完成哪些函数的实现。这是 CORBA 为保证所有接口操作都可以被激发而强迫给软件开发人员的编程义务。但是,CORBA 并没有规定,这个接口中的操作只能由上述对象实现中的方法完成。

1.5 对象请求代理

对象请求代理(ORB)[13-15]是连接接口存根和接口框架的桥梁,接口存根和接口框架的通信就是由它来完成,对象请求代理具有对象位置透明性,也就是客户端根本不需要你调用的服务在什么地方,你只需知道有这么一个服务在就行,剩下的对象请求代理会帮你找到。还有,ORB 能够把应用程序、各类对象、应用服务和应用工程集有序地分割开来就可以有效实现 CORBA 分布式软件集成和即插即用的功能。

1.6 静态模型的工作步骤

一个用户需要调用知件中的某个服务的步骤如下:

(1)用户通过某种方法查找到需要调用服务的引用。查找的方法有 ORB 中提供的方法 resolve_initial_reference();或者使用 CORBA 中的对象命名服务(Naming Service)。

(2)用户向知件服务对象实现发出请求。可以向相应的接口框架中发出请求,因为接口框架就是对于对象实现的。

(3)请求会传到 ORB。由 ORB 去查找知件对象实现的具体位置。

(4)知件对象实现的响应请求。知件对象实现会通过 ORB 把响应传回给用户。

(5)用户得到返回结果。在服务端得到的结果也会通过 ORB 把结果返回给用户。

一个完整的调用过程就是这样的,在用户和服务端之间的通信,不需要去管,一切交由 ORB 去处理,ORB 拥有不同的通信协议,为开发人员减轻了负担。

2 动态模型

2.1 动态模型结构

知件动态模型主要由用户、知件 IDL 注册管理中心、IDL 解释器、知件系统和 Java 模块对象组成,如图 2 所示。知件系统的开发商会提供知件 IDL 和 Java 模块对象,供用户来自动生成自己所需的知件调用界面。

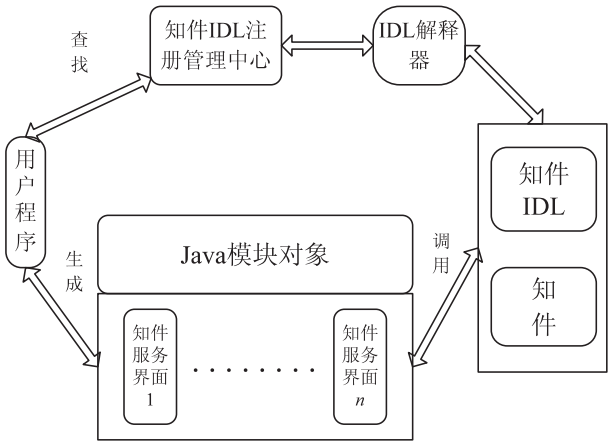


图 2　知件动态模型

2.2　知件接口描述语言

把知件看成一个对象,那么它的服务特性就是知件对外使用的接口。所有的知件和知件的服务功能都可以看作是一个模块,每个模块都可以通过接口和其他模块通讯,这刚好符合 IDL 语言模块化的特性[8-10]。

知件接口描述语言的 BNF

<知件接口>::=<接口头><接口体>

<接口头>::=<知件名称>“is”“<”<知件类型>“>”

<接口体>::=“{”<接口描述>*“}”

<接口描述>::=<applet 类型声明><属性>*<操作>+<关联说明>*

<applet 类型声明>::=“appletType”<applet 类型值>“;”

<属性>::=[“attribute:”“‘”属性名称“’”]<属性值>“;”

<操作>::=[“function:”“‘”操作名称说明“’”]<操作返回值类型><名称>(<参数说明>)“;”

<参数说明>::=<空>|<参数>

<参数>::=<参数方向><参数定义>

<参数方向>::=in | out | inout

<参数定义>::=<参数类型><参数名>

<关联说明>::=“usedBy”<知识中间件类型>“;”

<知件名称>::=字符串

<属性名称>::=字符串

<applet 类型值>::=字符串

<知识中间件类型>::=字符串

<返回值类型>::=void | boolean | URI | RDF | Ontology

<参数类型>::=string | URI | RDF | <枚举类型名称>

其中,=表示定义为;|表示可选;()表示包含了参数说明;<>表示变量;“”表示规定书写的内容;+表示可重复 1 次或多次;*表示可重复 0 次或多次;{}表示一个独立的接口;[]表示对后面内容的说明;如果[]内包含的关键字是 attribute,则说明后面表示的是一个属性;如果[]内包含的关键字是 function,则说明后面表示的是一个操作。

2.3　IDL 解释器

知件 IDL 对于知件的描述主要分为属性、操作和关联关系三个部分,所以,IDL 解释器也就是针对这三个部分进行解释的。除了这些,由于知件有它隶属的类别,为了操作方便,也为了关联知件时的准确,还需要知件的类别生成一个知件列表。

(1)知件列表。

根据知件类型和知件名称可以组成一层次结构,相当于父类和子类的关系。

(2)属性。

提取知件 IDL 中的属性名称和属性值作为一个记录。

(3)操作。

由于每个知件的操作都可能涉及到知件间相互调用的问题,因此,具有 provided 的操作除了作为接口库中的记录外,还要放入事件通道中去,以供关联知件的调用。

(4)知件关联。

根据 usedBy 语法,提取关联的知件类型,组成一个关联类型的列表。该表中的知件 ID 号与信息通道中的知件 ID 号相对应。

2.4　文件注册过程

当知件将 IDL 发送到 IDL 解析器中,解释器对其解释并把解释之后的数据注册到知件 IDL 注册管理中心,注册中心中包含了知件 IDL 所有声明的属性和接口信息,用户根据自己的需求去查找注册中心的信息,生成自己需要的服务界面。

下面以一个实例来讲述文件的注册过程。

```
Test 知件 is <领域知件>{
[attribute:'知件 ID']int 1001 ID;
[function:'网址查询']String URLSearch(in String str);
};
```

IDL 解释器对 Test 知件进行逐行扫描:

(1)第一行,根据知件名称—Test 知件,在知件列表中添加一个记录。然后根据 Test 知件的知件类型—领域知件,在知件表中查看该知件类型是否存在,如果不存在,则新增一个知件类型的记录,然后将其值加入新添的知件记录中去;如果存在,则直接将该知件类型加入新添的知件记录中去。知件表中的知件号从属性表中的特殊属性中获得。

(2)依次扫描知件 IDL 中的属性,提取属性名称、属性类型和属性值,然后在属性列表中分别为每个属性新增一个记录,完成属性信息的注册。这里要注意的一点,就是这个属性后面带有 ID 这个标准,这是一个特殊的属性,可以从这里知道知件的知件号,前面和后面提到的知件号就是从这里获取的。

(3)顺序扫描知件 IDL 中的操作,获得操作名称说明、返回值类型、操作名称、参数方向、参数类型和参

数类型说明等内容,然后在接口表中分别为每个操作新增一个记录,完成操作信息的注册。接口表中各接口所属知件号从前面说过的特殊属性中获得。

(4)IDL 解释器在扫描知件声明时,对于具有 provided 声明的方法,获取操作名称存入到事件通道表中,事件通道表中的知件号从属性表中的特殊属性中获得。形成一个独立的事件通道表方便知件之间的互相调用。状态表的信息注册是来自知件服务端的服务启动,知件服务启动时会在状态表中注册自己的信息,以便用户可以查询服务端是否启动,如果启动了用户可以生成知件服务界面,没有启动就会返回没有此服务或者没有启动。

(5)依次扫描知件 IDL 中的关联操作,提取关联知件类型名称,然后在知件关联表中新增一个记录,将上述内容分别注册到知件关联表中。知件关联表中的知件号从属性表中的特殊属性获得。这个知件关联表与事件通道表在逻辑上是一个整体,知件互调用的时候需要同时查询这两个表中的内容。

通过以上过程,IDL 解释器完成了对知件 IDL 的分析扫描,并把得到的信息都注册到知件 IDL 注册管理中心。

2.5 Java 模块对象

Java 模块对象是知件系统开发商提供给用户自动调用的知件服务界面,这里不做详细介绍。

3 实 验

动态模型的实验仿真是在 Eclipse 平台上进行,搭建好 CORBA 在 Eclipse 中的环境配置,就可以进行实验。

3.1 静态模型实验

先把服务器启动后,再启动客服端,会显示一个窗口,在窗口中输入要查询的信息,就会在显示框中看到查询结果的网址,此处由于实验结果截图不清晰,就不展示结果图了。

3.2 动态模型实验

用户程序查找知件 IDL 注册管理中心,根据查找到的知件表的信息生成如图 3 左侧所示的树形结构,点击树中 Test 知件,如果 Test 知件服务已经启动,就会看到图 4,生成可以调用知件的服务界面,如果 Test 知件服务没有启动就会看到跟图 3 一样没有变化。知件服务启动会在注册中心的状态表中注册相关信息,在查找到服务生成界面时也会去读状态表,看此服务有没有启动,启动就会生成服务界面,没有启动就没有反应。

图 4 中的右边界面,是知件系统开发商提供的一个模块,以便于插拔过程中的自动生成。

图 3 动态知件服务界面

图 4 Test 知件界面

4 结束语

文中针对如何使用开发出来的知件系统,提出一个知件的动态和静态可插拔模型,在知件更换中不影响用户的使用,来达到即插即用的目的。在实现中,毕竟仿真实验与真实实验还有一些区别,因此存在一些不足之处,希望在今后的实际应用中促进知件系统的开发和使用。

参考文献:

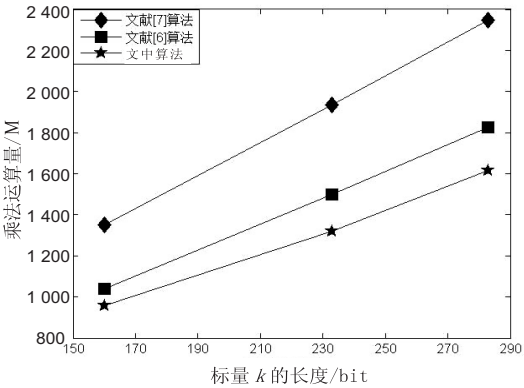
[1] 陆汝钤,金 芝. 从基于知识的软件工程到基于知件的软件工程[J]. 中国科学:E 辑,2008,36(6):843-863.

[2] Lu Ruqian, Jin Zhi. From knowledge based software engineering to knowware based software engineering[J]. Science in China Series F: Information Sciences, 2008, 51(6): 638-660.

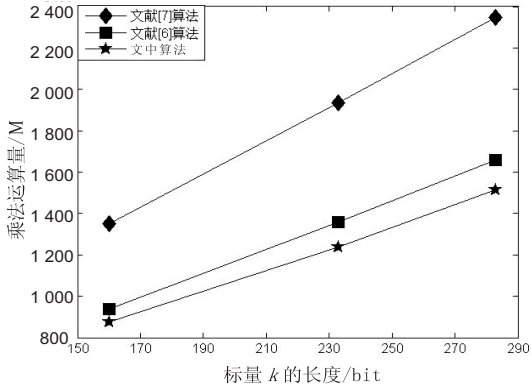
[3] Lu Ruqian, Jin Zhi. Beyond knowledge engineering[J]. Journal of Computer Science and Technology, 2006, 21(5): 790-799.

[4] Lu Ruqian. From hardware to software to knowware: IT's third liberation? [J]. IEEE Intelligent System, 2005, 20(2): 82-85.

(下转第 98 页)



N=2 时的运算量曲线



N=5 时的运算量曲线

图 1 相应算法运算量比较

参考文献:

[1] Miller V S. Use of elliptic curves in cryptography[C]//Proc of CRYPTO'85. [s. l.]:Springer-Verlag,1986:417-426.

[2] Koblitz N. Elliptic curve cryptosystems[J]. Mathematics of Computation,1987,48(177):203-209.

[3] 陈厚友,马传贵. 椭圆曲线密码中一种多标量乘算法[J]. 软件学报,2011,22(4):782-788.

[4] Dimitrov V,Imbert L,Mishra P K. Efficient and secure elliptic curve point multiplication using double base chain[C]//Proc of Cryptology-ASIACRYPT'05. [s. l.]: Springer-Verlag, 2005.

[5] Mishra P K,Dimitrov V S. Efficient quintuple formulas for elliptic curves and efficient scalar multiplication using multibase number representation[C]//Proc of ISC'07. Valparaiso: Springer-Verlag,2007.

[6] 洪银芳,桂 丰,丁 勇. 基于半点和多基表示的标量乘法扩展算法[J]. 计算机工程,2011,37(4):163-164.

[7] Purohit G N,Rawat S A,Kumar M. Elliptic curve point multiplication using MBNR and point halving[J]. International Journal of Advanced Networking and Applications, 2012, 3(5):1329-1337.

[8] Doche C, Imbert L. Extended double-base number system with applications to elliptic curve cryptography[C]//Proceedings of the 7th international conference on cryptology. Berlin: Springer-Verlag,2006:335-348.

[9] 蒲 冰,牛荣健. 扩展的 DBNS 椭圆曲线标量乘算法[J]. 计算机工程与应用,2011,47(26):98-102.

[10] Knudsen E W. Elliptic scalar multiplication using point halving[C]//Proc of ASIACRYPT'99. [s. l.]: Springer-Verlag, 1999.

[11] 郝艳华,李 磊,王育民. 利用多基链计算椭圆曲线标量乘的高效算法[J]. 电子科技大学学报,2008,37(6):868-871.

[12] 陈 辉,鲍皖苏. 基于半点运算与多基表示的椭圆曲线标量乘法[J]. 计算机工程,2008,34(15):153-155.

[13] Liu D G,Ning P. Establishing pairwise keys in distributed sensor networks[C]//Proceedings of the 10th ACM conference on computer and communication security. New York: ACM Press,2003:52-61.

[14] 殷新春,赵 荣,侯红祥,等. 基于折半运算的快速双基数标量乘算法[J]. 计算机应用,2009,29(5):1285-1288.

[15] 赖忠喜,张占军,陶东娅. 椭圆曲线中直接计算 7P 的方法及其应用[J]. 计算机应用,2013,33(7):1870-1874.

(上接第 94 页)

[5] Ding Liya. Design and development of knowwares system [C]//Proc of the 2nd international conference on innovative computing, information and control. Kumamoto, Japan: [s. n.],2007:152-158.

[6] Ding Liya, Lo Sio-Long. Inference in knowware system[C]// Proc of international conference on machine learning and cybernetics. Baoding:IEEE,2009.

[7] Ding Liya. A model of hierarchical knowledge representation toward knowware for intelligent systems[J]. Journal of Advanced Computational Intelligence and Intelligent Informatics, 2007,11(10):1232-1237.

[8] 廖瑞华. 基于 CORBA 的智能信息家电的可插拔的模型研究[D]. 长沙:湖南师范大学,2004.

[9] 阳俐君. 信息家电接口描述语言及其编译器的研究与设计[D]. 长沙:湖南师范大学,2007.

[10] 黄慧华. 基于信息家电接口定义语言的远程监控系统的设计与实现[D]. 长沙:湖南师范大学,2005.

[11] 王鹏杰. CORBA 核心服务的研究与实现[D]. 长春:吉林大学,2003.

[12] 朱其亮,郑 斌. CORBA 原理及应用[M]. 北京:北京邮电大学出版社,2001.

[13] 潘慧芳,周兴社,於志文. CORBA 构件模型综述[J]. 计算机应用研究,2005,22(5):14-15.

[14] 周丽娟,姚丽娜. Java RMI 技术的研究与应用[J]. 株洲工学院学报,2006,20(2):42-44.

[15] 熊志斌. 基于 CORBA 的智能小区网络模型的研究与实现[D]. 长沙:湖南师范大学,2005.

知件可插拔机制的设计与实现

作者： 潘明钢， 张楚才， 钟维， PAN Ming-gang， ZHANG Chu-cai， ZHONG Wei
作者单位： 湖南师范大学 数学与计算机科学学院,湖南 长沙,410081
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2015(5)