# 基于MSMQ消息中间件实现企业应用集成

曹 鹏,李振坤,龚志新,覃 英
(广东工业大学 计算机学院,广东 广州 510090)

**摘 要**:介绍通过基于MSMQ的消息中间件技术,统一卫生局内各个独立分布的信息管理系统中的关键异构数据,实现一个松散耦合的企业应用数据集成,为从宏观上把握各卫生系统情况和对决策的制定提供数据支持。

**关键词**:MSMQ;消息中间件;企业应用集成
**中图分类号**:TP393.09 **文献标识码**:A **文章编号**:1673-629X(2007)06-0232-03

# Realization of Enterprise Application Integration Based on MSMQ Message Oriented Middleware

CAO Peng, LI Zhen-kun, GONG Zhi-xin, QIN Ying
(Faculty of Computer, Guangdong University of Technology, Guangzhou 510090, China)

**Abstract**: Introduce message oriented middleware based on MSMQ which consolidate the isomerous data in distributed information systems into one database. And realize a loose-coupling enterprise application integeration-data integration. It's very helpful for people to master macroscopical situation and to make a decision for board of health.

**Key words**: MSMQ; MOM; EAI

## 0 引 言

卫生局为了管理辖区内的各个医疗机构情况和及时跟踪市民的卫生健康状况,配置了大量的各类系统,几乎每个系统都由不同的软件开发公司开发,分别负责统计不同方面的数据。一段时间以来一直很好地为卫生部门的各项工作开展提供高效保障,但随着时间推移,一个最重要的问题也暴露出来:由于各信息系统的强行划分,造成各个系统数据孤立,反映的信息单调而缺乏联系,有时甚至需要人工进行各系统中相关联数据的汇总,工作量大、耗时长,使得工作由于缺乏实时统计数据而陷于被动局面,不利于卫生部门各项工作的开展。根据这种情况,提出了用基于MSMQ消息中间件技术来实现各应用系统的集成,提供一定层次上的数据分析与挖掘,提高了工作效率。

## 1 企业应用集成(EAI)

企业为了保护投资、节省成本,在不同的时期,根据不同的需要,投入人力、物力和财力构建了一系列适合本单位使用的计算机应用系统,不仅方便了各部门进行内外业务管理和处理日常工作,而且大大地提高了工作效率及对信息资源的管理、利用和开发。但这些应用系统由于是在不同的时期、针对具体问题、采取不同的技术自行开发或者购买的。它们之间共享性差、兼容性不好,每个系统都是一个独立的"信息孤岛"。为了解决应用系统之间的信息交互和共享问题,人们提出了企业应用集成EAI(Enterprise application integration)——将进程、软件标准和硬件联合起来,在两个或更多的企业应用系统之间实现无缝集成,使它们像整体一样进行业务处理和信息共享[1]。

解决企业应用集成方案通常有3种,即数据集成、业务流程集成和函数或方法集成。

## 2 消息中间件(MOM)

中间件是一种独立的系统软件或服务程序,分布式应用软件借助这种软件在不同的技术之间共享资源。中间件现在一般分成以下几类:消息中间件、数据访问中间件、远程过程调用中间件、交易中间件、对象中间件等[2]。

目前对消息中间件(MOM)的定义还未形成统一的行业标准,我国也正加快对消息中间件技术的标准化研究工作。一般认为,消息中间件是一种由消息传

**收稿日期**:2006-08-04
**作者简介**:曹 鹏(1979-),男,湖南衡阳人,硕士研究生,研究方向为计算机网络与分布式计算;李振坤,教授,硕士生导师,主要研究方向为计算机网络与分布式计算。

送机制或消息队列模式组成的中间件技术,利用高效可靠的消息传递机制进行平台无关的数据交流,并基于数据通信来进行分布式系统的集成。与其它中间件技术不同(例如 ORB 和 RPC),一般来说,消息中间件并不要求系统具备一个可靠的底部传输层,而是通过以消息的形式收发应用程序数据来连接运行于不同系统上的应用程序。信息可以同步传送,也支持异步传送。在异步方式下,应用程序并不需要消息即时即刻传送到对方,只是由 MOM 确保把信息以消息的方式传送到适当的目的地,并且只传一次[3]。

消息中间件是众多中间件中的一种,也是最为重要的一种。消息中间件和邮局的概念有些类似,消息被放到消息中间件中,消息中间件负责将消息发送到目标系统。消息中间件是利用高效可靠的消息传递机制进行平台无关的数据交流,并基于数据通信来进行分布式系统的集成。主要的消息传输机制有队列(Queue)(如图 1 所示)和发布订阅(Pub/Sub)(如图 2 所示)两种[4,5]。

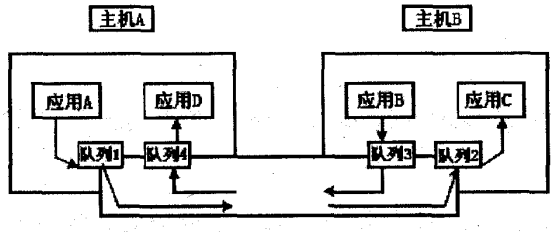


图 1　队列模式

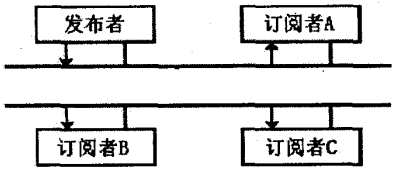


图 2　发布订阅模式

主要特点包括以下 6 个方面:(1)异步传送;(2)防御通信;(3)并发执行;(4)日志通信;(5)多种通信方式;(6)应用程序与网络复杂性相隔离。

典型的商用产品有 IBM MQseries、微软的 MSMQ、BEA 的 MessageQ 以及开源的 JORAM,OSMQ 等。

## 3　MSMQ (Microsoft Message Queuing)

MSMQ((微软消息队列))是微软公司的成熟产品,最新版本为 MSMQ3.0。

MSMQ 集运行系统、管理工具集和开发系统三者于一身,既为上层应用系统提供了可靠、高效的数据通信服务,又为网络系统提供了实时管理和监控的工具,同时还为编程人员提供了简单、易用、功能强大的应用开发接口,由于它对异构网络的透明性,并且保证消息的一次性可靠传输,对事务的支持,有效的路由机制等特性,使得 MSMQ 特别适合开发分布的,松散耦合的 Windows 应用程序。

MSMQ 技术是一种利用队列机制实现部件间或者是应用程序间通信的技术,它允许以异步、实时的方式相互传递信息。消息队列是一种灵活而可靠的通信机制,并且适合于各种程序,开发人员并不需要了解许多的细节,MSMQ 具有以下特性:

(1)异步通信,MSMQ 可以分布式组件并以一种异步方式进行通信,而不必担心占用宝贵的网络资源。

(2)消息路由可靠,MSMQ 保证消息只传递一次,这样使两个接收者不会意外地得到相同的消息,消息也不会丢失。

(3)事务集成,MSMQ 可以自动地调用事务服务来保证数据的完整性。

(4)自动消息日志,MSMQ 日志保存发送和接收所有的消息,并且进行自动的审计跟踪,发生错误时易于恢复。

(5)安全性,MSMQ 具有保密性、数字签名等特性,能够对网络上传送的消息进行数字签名及加密传输,而且在 MSMQ 服务端可以过滤掉未经授权的消息,从而保证了安全性。

(6)优先级,MSMQ 支持消息队列优先级机制,它将有选择地传送优先级较高的消息,让程序能够优先处理重要事件。

(7)协议无关性,MSMQ 消息的发送与网络协议无关,只需提供消息队列的名字即可实现消息的发送与接收。

## 4　系统设计与实现

鉴于卫生局各信息系统都不是自主开发,不可能对各个系统进行修改来实现各分布系统集成,而只能通过对各个系统的数据库进行分析以实现数据层面上的集成,并最终达到应用集成的目标。系统改造后的基本框架结构图如图 3 所示。

所有的机器必须在一个域的范围内,并且必须有一台是安装了主域控制器(Active Directory)。可以看出,通过在各个分布的系统上安装一个 Windows 服务程序(其中安装在主机 1 到 4 上的是消息程序的发送部分,安装在主机 5 上的是消息程序的接受部分)在系统启动后自动开始监视各自系统的数据库文件,一旦数据库文件发生修改后自动触发 MSMQ 向主机 5 发送消息,主机 5 则从消息队列中接受消息来对数据库进行相应修改,实现数据同步。

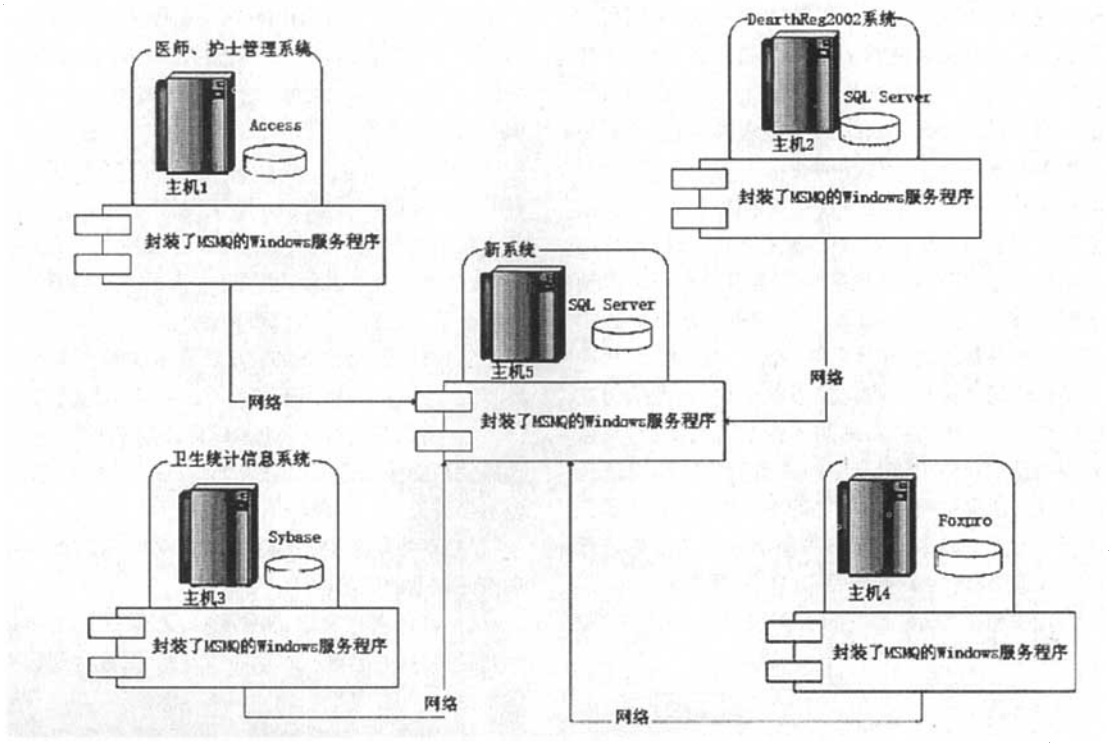


图 3 系统结构图

每一个消息系统的内部结构都如图 4 所示。

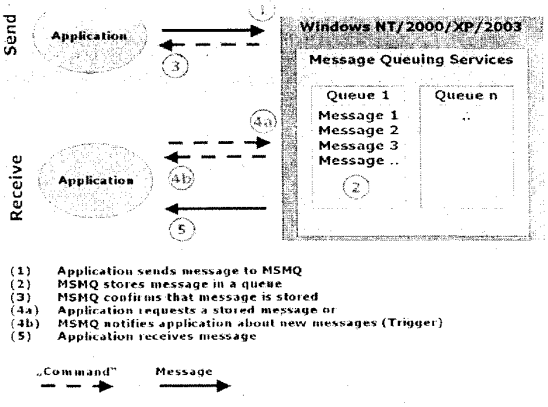


图 4 一个消息系统的内部结构

下面详细介绍程序的工作过程与一些重要技术的实现细节，系统在 Microsoft .NET Framework 2.0 采用 C# 实现。

我们希望在用户启动机器后，甚至在用户还未登录的情况下就开始运行各个系统的监视程序(主要监视数据库文件)，所以采用将监视程序写成了 Windows 的服务程序，监视程序被注册成 Windows 的服务程序后，开机后自动开始监视活动并适时向队列发出消息，对用户来说，这些活动完全透明，用户对系统的操作也完全不受影响。

MSMQ 中的单个最大消息传输容量为 4MB，同时也为减少网络数据传输数据量，对原来各系统的数据库进行分析，发现每个数据库会发生改变的表很少，我们在监视程序的配置文件中将这些表名标出，监视程序每次发送的消息只从配置文件中标示的表中提取数据，并存储在 DataSet 对象中，DataSet 对象将序列化成 XML 格式形式进行发送，一个表对应一个 DataSet，所以一般情况下一个消息的大小也都控制在了 200kB 左右。发送代码实例：

```
System.Messaging.MessageQueue myQueue = new System.Messaging.MessageQueue("FormatName:PUBLIC = 84B54000 - 441A - 4711 - 912A - 614E158BDE44");
myQueue.Send(dataset);
```

默认情况下 DataSet 对象以 XML 格式形式序列化对象并发送，在消息接收端需要再将消息体数据转换成 DataSet 对象，并最后将数据更新至主机 5 的 SQL Server 的数据库中。接收代码示例：

```
myQueue.Formatter = new XmlMessageFormatter(new Type[]{typeof(System.Data.DataSet)});
System.Messaging.Message myMessage = myQueue.Receive();
System.Data.DataSet myDataSet = (System.Data.DataSet)myMessage.Body;
```

(下转第 238 页)

```
property = propertys.substring(fromIndex);
alPropertys.add(property);
for(int i=0;i<alProperiys.size();i++){
property = (String)alProperiys.get(i);
String getPropert = "get" + property.substring(0,1).toUpperCase() + property.substring(1);
String setProperty = "set" + property.substring(0,1).toUpperCase() + property.substring(1);
```

②从 actionForm 中获得指定的属性值：

```
Class cl = actionForm.getClass();
Method method1 = c1.getMethod(getPropety,null);
Object value = method1.invoke(actionForm,null);
```

③对实体类对象的各个指定的属性赋值：

```
Class c2 = valueObject.getClass();
Object[] argsl = {value};
Class[] classl = {value.getClass()};
Method method2 = c2.getMethod(setProperty,classl)
method2.invoke(valueObject,argsl);
}
```

④)调用相应的方法以实现对库表的操作：

```
DeviceClient client = new DeviceClient();
Class c4 = client.getClass();
String methodName = aMapping.getMethodName();
Class[] class2 = {Clsss.forName(ClassName)};
Method method4 = c4.getmethod(methodName,class2)
Object[] args2 = {valueObject};
method4.invoke(client,args2);
...
}
```

通过以上对 Action 组件的黑盒化设计，不需要再对每个模块都编写代码类似的 Action，也不用关心 Action 的具体调用逻辑，只需要建立一个公用的 MyAction类，并且对每一模块按照新的方法重新配置 Structs－config.xml 就可以完全公用实现增、删、改操作，这使整个系统流程更加简单清晰。但是此公用 Action 只适合于增、删、改操作，并不适合于查询操作，这是因为查询操作往往有检索条件、返回值，并且要涉及翻页等情况，相对而言逻辑比较复杂，从而不容易被同时黑盒化。

## 4 结束语

MVC 是非常优秀的 B/S 设计模式，它已经越来越多地运用于企业平台之上，许多大型网站成功地应用该设计方法使项目分工明确，大大缩短了项目周期。文中在对 MVC 设计模式探讨和研究的同时，通过某单位综合信息系统详细描述了 MVC 设计模式的实现流程。最后，对原本基于白盒设计的 Action 组件提出黑盒化改进方案，这对今后更好地使用 MVC 进行 B/S 开发提供了参考。

**参考文献：**


[1] 姜明霞. MVC 设计模式及 Struts 框架的研究与应用[D]. 大连：大连海事大学，2005.

[2] Goodwill J. Mastering Jakarta Struts[M]. [s.l.]:Wiley Publishing, Inc.,2002.

[3] Husted T, Dumoulin C, Franciscus G, et al. Struts in Action Building web applicatinos with the leading java framework [M]. [s.l.]:Manning Publications Co.,2003.

[4] Barish G. J2EE WEB 应用高级编程[M]. 林 琪，英 宇译. 北京：清华大学出版社，2002.

[5] Nilsso D R, Mauget L E. J2EE 应用与 IBM WebSphere[M]. 马竹青，鞠文飞 译. 北京：电子工业出版社，2004.


(上接第 234 页)

主机 5 的消息接受程序使用 MSMQ 的 MSMQ Trigger(触发器)，该机制能通过用户自定义的相应规则自动判别进入队列的消息，而做出具体响应。

## 5 结束语

文中以 MSMQ 为例，分析 Windows 系统环境下的数据集成问题，详尽地介绍了在使用 MSMQ 进行企业应用集成的具体应用，并提出了一种基于 MSMQ 的数据集成方法。利用该方法，可以直接从分散的数据库中提取需要数据，并在接收端汇总数据，而不需要修改任何原来的程序，保存了固有的投资，使企业不断变化的需求在最低投入上得到满足。根据具体企业应用情况不同，该模式可被大量应用到其它需要进行类似企业应用集成的部门。

**参考文献：**


[1] 蓝 烨，翁惠玉. 利用 MSMQ 实现的服务引擎[J]. 计算机工程，2004(增刊)：121－123.

[2] 袁 琳，许林英. 中间件集成企业应用[J]. 计算机工程，2005(7)：91－92.

[3] 高 勇，吴 健. 基于 MSMQ 的分布式应用架构[J]. 计算机应用研究，2004(5)：56－57.

[4] 高建斌，董传良，谢 莉，等. 数据集成中消息中间件的设计[J]. 计算机工程，2003(10)：95－97.

[5] 刘保罗，石 磊. 电子商品交易系统中间件的设计及应用[J]. 计算机应用与软件，2006(1)：23－24.