基于 ServiceMix 的 RFID 中间件的研究与实现①

RFID Middleware Based on ServiceMix

邹　伟　王绍源　(湖南大学 电气与信息工程学院 湖南 长沙 410082)

摘　要： 本文介绍了 RFID 技术的基本原理和 RFID 中间件的现状，设计了一种基于 ServiceMix 的 RFID 中间件架构，实现了该架构的边缘层、消息系统层、数据接口层和管理界面层,将该中间件应用于实验器材管理系统中，达到了实验器材的实时跟踪管理。

关键词： RFID 中间件 ServiceMix 规格化消息 WebService

1　引言

无线射频识别[1](Radio-Frequency Identification：RFID)是一种非接触的自动识别技术，基本原理是利用射频信号和空间耦合传输的特征，实现对被标识物理的自动识别。RFID 具有一次处理多个标签、可将处理状态写入标签、不受大小及形状限制、穿透性强、数据的记忆容量大、可重复利用等优点。因此，RFID 在很多领域均具有广阔的应用前景。同时也存在应用 RFID 技术的一些问题：不同设备与应用系统之间的接口问题，大量复杂的 RFID 数据如何处理、利用以及如何实现 RFID 系统与现有信息系统无缝的、快速整合等问题。而解决这些问题的方法就是构建并部署一套 RFID 中间件，RFID 中间件[2]是一种面向消息的中间件，信息以消息的形式，从一个程序传送到另一个或多个程序。目前国内外已经有一些公司进行 RFID 中间件的开发，并有了一些相关方面的产品：如国外的 IBM、Microsoft、BEA、SUN、Sybase 等都已经有或即将发布 RFID 中间件系统，但都有太大的依赖性和较小的扩展性，其适用性和成熟度存在一定的问题。目前国内对 RFID 中间件主要以研究为主，产品开发较少。本文设计了 RFID 中间件的基本架构，并在 Apache 社区开源产品 ServiceMix 基础上实现一个简单的 RFID 中间件。

2　RFID中间件基础架构的设计

RFID 中间件是介于前端 RFID 设备(如 射频识别阅读器、event 伺服器)与后端数据库及应用软件(如 SCM 系统、各种技术背景的 ERP 系统)中间，提供程序管理、资料过滤与汇集、事件管理、安全管理、网络管理等机制，是支持 RFID 应用系统开发和运行的支撑软件。其为 RFID 应用构建了一个标准的平台，一方面屏蔽了芯片、标签、读写器等 RFID 硬件设备以及操作系统、数据库等的差异，提供了高安全、高性能、高扩展性、可管理性等方面的可靠保障；另一方面有效驱动后端的业务应用系统形成统一的协调运作。本文中 RFID 中间件架构是基于 ServiceMix 而设计的，ServiceMix[3]是 Apache 社区的一个完全实现了 JBI（Java Business Integration）标准的开源项目产品，也是一个基于 SOA（Service-Oriented Architecture）架构和事件驱动的 ESB（Enterprise Service Bus）,ServiceMix 已经集成了 Spring 支持，可以成为一个服务总线的提供者，也可以给另一个服务总线提供服务，本质是一个消息传递的系统。基础架构分为：边缘层，消息系统层，数据接口层，管理界面层。如图 1，以下分别叙述各层的功能及其所在体系架构中的位置。

2.1 边缘层

边缘层位于架构的低层，直接与读卡器交互，由边缘服务模块与读卡器代理模块组成。其功能：

1)对射频卡上的数据进行采集，并对其原始数据进行过滤，聚合。

2)对来自不同读卡器的数据进行适配处理，得到

① 收稿时间:2008-09-01

统一的、格式化数据。

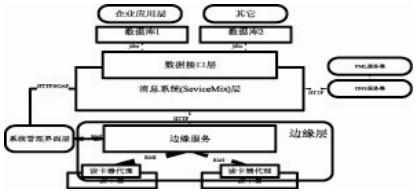

图 1　RFID 中间件体系结构

3)将校验无误的数据按照自定义的协议进行封包，系统采用的是 XML 的形式进行封包，并发送到消息系统中。

4)与管理界面层通信。

以上 1）2）功能在读卡器代理中，3）4)功能在边缘服务中。

2.2 消息系统层

消息系统层是 RFID 中间件架构核心层。一方面各种应用程序以不同的方式频繁地从 RFID 系统中取得数据，另一方面却是有限的带宽，其矛盾使得应设计一套消息系统传递系统的机制，因而采用基于消息传递的系统的 ServiceMix 服务总线是一个合理的设计。边缘层产生的事件，并将事件传递到消息系统中，由消息系统决定如何将事件传递到相关的应用系统。在这种模式下，读卡器不必关心哪个应用系统需要什么数据，应用系统也不需要维护与各个读卡器之间的网络通道，仅需要将需求发送到消息系统中即可。消息系统层具有数据缓存、基于内容路由及数据的分类存储功能。

2.3 数据接口层

数据接口层采用了 WebService[4]技术，易于企业内部与企业之间数据的通信。本系统中数据接口层是嵌在消息系统层中的，但它们却是松耦合的，具体会在实现部分介绍。数据接口层的主要功能：

1)数据的入库。

2)提供数据服务接口。

2.4 管理界面层

管理界面层与边缘服务层与数据接口层通信，其功能：

1)监控边缘层的读卡器代理，停止或启动某个读卡器。

2)查询、修改数据，关闭或启动本地数据库系统。

3)监控一些运行时参数，以防止系统自动崩溃。

3　RFID中间件基础架构的实现

本系统数据协议的定义借鉴了参考文献[5]中的 EPC-64 的思想，自定义了一种新的数据结构格式以满足系统的需求。以下是各层的具体实现。

3.1 边缘层的实现

边缘层由边缘服务模块与读卡器代理模块组成。读卡器代理模块根据硬件厂商的 dll 或 jar 文件编码实现，以实现与读卡器数据的交互。边缘服务模块采用 XML 对读卡器代理模块传来的数据进行封包再发送到消息系统层。边缘服务模块与读卡器代理模块之间运用 MVC(Model-View-Controller)架构、采用 RMI 远程协议进行通信，边缘服务模块对应多个读卡器代理模块，很好的实现分布式处理。读卡器代理模块与边缘服务模块的 UML 图如图 2 所示。

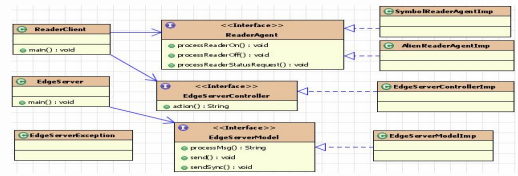

图 2　边缘服务层类框图

EdgeServerModelImpl 边缘服务模块在 RMI 注册表中注册，ReaderAgentImp 读卡器代理模块通过 RMI 传输协议取得边缘服务模块在本地的存根，实现了读卡器代理模块与边缘服务模块之间的数据通信。

3.2 消息系统层的实现

消息系统层的实现采用的是 Apache 社区的 ServiceMix 开源产品,该层也是本系统的核心。其内部主要有 5 大组件(如 图 3)：httpRceiver 绑定组件，数据接口引擎组件，2 个 WebService 绑定组件，路由引擎绑定组件。httpRceiver 绑定组件功能是接收外部 http 请求的服务。数据接口引擎组件属于数据接口层。2 个 WebService 绑定组件是自定义的绑定组件，其功能是请求外部服务(WebService 服务)，在假设外部有两个企业 A 和 B 提供了 WebService 服务基础之上，外部服务可以根据唯一识别码而获得该码对应产品的具体信息。路由引擎很好的实现了根据解析后的唯一识别码的内容路由到相应的组件。其中 httpRceiver 绑定组件、2 个 WebService 绑定组件只需在配置文件（ServiceMix.xml）中配置即可，无需具体编码,而数据接口引擎组件与路由引擎绑定组件除了配置之外还需自己编写一定量的代码。在配置文

件中，除配置好五大服务组件之外，首先还需配置好一个 JBI 的容器。

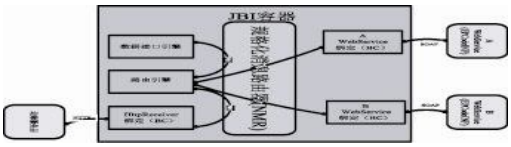

图 3　基于 ServiceMix 消息系统层的流程图

路由引擎服务组件又是该层的核心，核心代码如下：

```
//交换消息转化为 InOut 的消息形式
InOut inout = (InOut) exchange;
// 取出规格化路由消息
Normalized MessageinMessage=inout.getInMessage();
//解析消息得到要路由的目的地
String dest = getProcessedMessage(exchange);
QName qName = null;
//通过解析得到的内容,是 0000,则路由到 A 企业的绑定组件;不是,则路由到 B 企业的绑定组件
if(dest.equalsIgnoreCase("0000"))
qName=new QName("http://servicemix.org/demo", "AEnterprice");
else qName=new QName("http://servicemix.org/demo", "BEnterprice");
InOut inout1 = createInOutExchange(qName, null,null); inout1.setProperty("requestId", exchangid);
//收到请求消息置于新建的 InOut 消息交换中。
inout1.setInMessage(inMessage);
send(inout1);//发送 inout1 到 NMR,由其转发到相应的组件
```

这段代码描述了路由引擎内部路由功能运行机制。

该层的具体业务如下：

外部服务（即图 3 中 2 个企业的 WebService 服务）提拱一个 getProductInf 的服务，这个服务在收到请求后（请求参数为自定义的唯一识别码），将返回此唯一识别码具体的产品信息。得到的信息在数据接口引擎组件中重组，再进行入库操作。

下面是实现该层功能的详细步骤：

1)外部服务请求者(此处为边缘服务层)将送一个 HTTP 的请求到 JBI 容器的专门接收基于 HTTP 协义的请求消息的 httpReceiver 绑定组件；

2)httpReceiver 绑定组件将请求消息转化规格化消息 NM(Normal Message、规格化消息)送到 NMR(Normal Message Router、规格化消息路由器),NMR 再送到路由引擎组件；

3)路由引擎接到 NM 后，根据 NM 中的内容选择不同的服务组件(如选中 A WebService BC)，再将 NM 送到 NMR，NMR 再将 NM 送到 A WebService 绑定组件；

4)A WebService BC 将 NM 转化为 SOAP 消息送到外部服务(此处为 A 企业的 WebService 服务)的 getProductInf 服务；

5)外部服务将结果响应的 SOAP 消息返回给 A WebService BC；

6)A WebService BC 将响应消息转化为 NM 送到 NMR，NMR 将 NM 送到路由引擎；

7)路由引擎接收到 NM 后，再将 NM 送到 NMR，NMR 再将 NM 送到数据接口层引擎；

8)数据接口引擎再将 NM 转化为 POJO 对象，以进行数据入库操作。操作的结果(成功还是失败)将转化为 NM，再由 NMR 将 NM 送到路由引擎；

9)路由引擎将 NM 送到 NMR，NMR 再将其送回到 httpReceiver 绑定组件；

10)httpReceiver 绑定组件将 NM 转化为 HTTP 消息送回到外部边缘服务层。

JBI 容器中的组件是不能直接通信的，不是直接将消息送给服务引擎组件(如:路由引擎组件)或是发送绑定组件(如: A WebServices 绑定组件)。它们需要将和具体的协议解耦后的一起送到 NMR,再由 NMR 将 NM 送到不同的绑定组件和服务引擎组件。这样所有的绑定组件、服务引擎组件处于一种松散耦合的状态,只需稍微配置，便可快速集成新的服务组件。即 ServiceMix 为服务组件提供了一个插拔式的 SOA 运行环境，所有的服务无需做任何修改，只需稍微配置，就可以直将新的后端数据库及应用软件(如 SCM 系统、各种技术背景的 ERP 系统)整合到这个运行环境中去。

3.3 数据接口层的实现

数据接口层本质是一个 ServiceMix 的 JBI 容器中的一个服务引擎组件，由数据移植模块与数据访问接口模块组成。数据移植模块负责数据的入库，即与本地数据库的交互，采用 JDBC 技术将数据移植到各

种流行的数据库中，如 MYSQL、SQLServer、DB2、Oracle 等数据库。数据访问接口模块提供数据访问的 WebService 服务接口，具体实现采用 ServiceMix 提供的 servicmix-jsr181 组件，这是一个 JBI 的服务引擎组件，可直接将 POJO(plain old java object)发布成 JBI 服务总线上的服务，内部机制采用了 Xfire 实现服务和 XML 的序列化工作。

3.4 管理界面层的实现

管理界面层采用 swing 编写 ,数据主要来自与边缘服务层、数据接口层之间的通信。通过查找并获得在 RMI 注册表中注册的边缘服务模块，从而监测并控制读卡器的一些相关信息，(如图 4)左边是一个树状结构，显示数据库、读卡器和本地的一些测试信息。选中左边的 Alien-01 读卡器，右边会出现 Alien-01 读卡器的状态信息，Alien-01 读卡器的类型，并可设置标签事件记录、标签的类型、读取的周期和控制 Alien-01 读卡器开启与关闭。数据结口层采用 ServiceMix 中提供的 servicmix-jsr181 组件服务组件的形式发布的服务接口，所以管理界面层容易与之进行数据通信。

图 4　读卡器管理界面

4　RFID中间件在实验器材管理的应用

在实验器材管理中，本系统能很好的跟踪实验器材的具体流向，故在学校硬件资源共享、优化等方面收到很好的效果。图 5 中，左边树状结构中选中数据库中 SQLserver，进入数据库管理界面，选中管理选项卡,显示数据库的状态等信息，并可终止、更新、初始化数据库连接；选中查询选项卡，输入 SQL 语句，即可查询数据库系统中的数据。

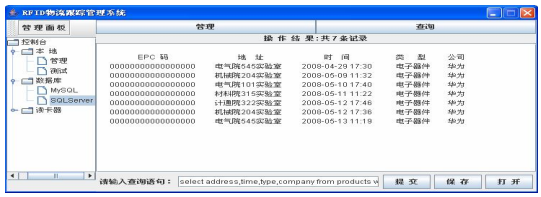

图 5　数据库管理界面

如输入：select epcode，address，time，type，company from products where epcode = '0000000000000000' ,在右边面板中可得到如下信息：

EPC 码	地 址	时 间	类 型	公司
00000000000000000	电气院545实验室	2008-04-29 17:30	电子器件	华为
00000000000000000	机械院204实验室	2008-05-09 11:32	电子器件	华为
00000000000000000	电气院101实验室	2008-05-10 17:40	电子器件	华为
00000000000000000	材料院315实验室	2008-05-11 11:22	电子器件	华为
00000000000000000	计通院322实验室	2008-05-12 17:46	电子器件	华为
00000000000000000	机械院204实验室	2008-05-12 17:36	电子器件	华为
00000000000000000	电气院545实验室	2008-05-13 11:19	电子器件	华为

即可清楚的查看到该实验器材的流向，从而实现了学校院系之间实验器材的共享，最终达到物流跟踪的目的。

5　结论

本文简单的介绍了 RFID 中间件，在采用 Apache 社区的开源产品 ServiceMix 服务总线上，设计出了本文的 RFID 中间件架构，并详细的说明了 RFID 中间件架构的具体实现，在实际运用中获得的很好的效果，达到了物流跟踪的目的。

由于在构建 RFID 中间件采用了服务总线的结构，这就使得整个应用系统变得异常灵活，能很好的与现有的应用系统无缝的、快速的整合，并在一种插拔式可配置的基础上实现了“即插即用”的思想。其次，由于在数据接口层运用了WebService这一核心技术，这也势必会带来中间件软件性能和数据传输安全性方面的问题。

参考文献

1 Srivastava L. Ubiquitous Network Societies：The Case of Radio Frequency Identification. ITU Workshop on Ubiquitous Network Societies, 2005.

2 吴正大,魏俊荣,张继新.RFID 中间件设计技术初探.邮电设计技术,2006(8):39－42.

3 梁爱虎.SOA 思想、技术与系统集成应用详解.北京:电子工业出版社,2007.179－242.

4 赵毅强,曾隽芳.Web Services 在 RFID 系统中的应用综述.计算机应用研究,2006，(12):1－4.

5 Auto-ID Center. EPC tag data specification version1.1,2005.http://www.epcglobalinc.org/standards_technology/specifications.html.

6 游战清,李苏剑.无线射频识别技术理论与应用.北京:电子工业出版社,2004.20－25.

7 邓海生,李军怀.基于 SOA 的 RFID 中间件的研究与实现.计算机技术与应用, 2007,(10):131－134.