文章编号:1001 - 9081(2008)04 - 1055 - 03

RFID 中间件的结构设计

成修治,李宇成
(北方工业大学 机电工程学院,北京 100041)
(chengxz@ redflag2000. cn)

摘 要:RFID 中间件是介于前端读写器硬件模块与后端数据库和应用软件之间的重要环节,它是 RFID 应用部署运作的中枢。针对目前相关企业的需求,设计了一种面向服务体系(SOA)的实时系统的 RFID 中间件结构,并给出了设备管理及询问器相应的 UML 类图结构,对 RFID 中间件中各组成部分的作用和构成作了详细地说明。通过对一个基于餐饮系统应用的测试表明,中间件可提高系统的可移植性,增强了系统的可维护性和可靠性。

关键词:射频识别;中间件;面向服务体系架构;应用层事件
中图分类号: TP311.52 **文献标志码**:A

Design of RFID middleware architecture

CHENG Xiu-zhi, LI Yu-cheng
(*College of Mechatronic Engineering, North China University of Technology, Beijing* 100041, *China*)

Abstract: Radio Frequency IDentification (RFID) middleware is the key software between tag readers and database and enterprise applications. It is the center of RFID application deployment operation. A Service-Oriented Architecture (SOA) real-time RFID middleware infrastructure was designed according to the requirement of related enterprises, and the corresponding UML class diagrams for the device management and integration components were given. The roles of RFID middleware components were described. The test of catering system shows that this middleware can improve the portability, maintainability and reliability of the system.

Key words: Radio Frequency IDentification(RFID); middleware; Service-Oriented Architecture (SOA); Application Layer Event (ALE)

0 引言

射频识别(RFID)技术,是一种利用射频通信实现的非接触式自动识别技术,由于具有高速移动物体识别、多目标识别和非接触识别等特点,RFID 技术显示出巨大的发展潜力与应用空间,已经应用到仓储物流管理、资产跟踪、生产过程控制、身份认证、智能交通等领域,并且仍在不断扩大。看到目前各式各样 RFID 的应用,企业最想问的第一个问题是:“我要如何将我现有的系统与这些新的 RFID Reader 连接?”这个问题的本质是企业应用系统与硬件接口的问题。通透性是整个应用的关键,正确抓取数据,确保数据读取的可靠性,以及有效地将数据传送到后端系统都是必须考虑的问题。传统应用程序与应用程序之间数据通透是通过中间件架构解决的,并发展出各种 Application Server 应用软件。

RFID 中间件是介于前端读写器硬件模块与后端数据库和应用软件(如 ERP 、CRM、WMS)之间,是 RFID 应用部署的重要环节。它是 RFID 运作的中枢,扮演 RFID 标签和应用程序之间的中介角色,从应用程序端使用中介软件提供的一组通用的 API,即能连到 RFID 读写器,读取 RFID 标签数据。如此一来,即使储存 RFID 标签情报的数据库软件或后端应用程序增加或改由其他软件取代,或者读写 RFID 读写器种类增加等情况发生时,应用端不需修改也能处理,省去多对多连接的复杂维护。否则,因为 RFID 读写器提供的接口数量增加,应用程序的数量也持续增加,撰写和维护点对点接口将成为一项艰难的任务。企业中复杂的情况如 50 个应用程序与 10 种 RFID 读写器连接,点对点的连接数量最多会增加至 500 个。中间件使得 RFID 应用系统的开发变得更容易,提高了软件的可移植性,增强了系统的可维护性和可靠性,所以它的架构设计解决方案是 RFID 应用的一项极为重要的核心技术。

1 中间件结构

要设计一个中间件,首先要对中间件建模,我们主要关心两个方面的问题:第一个是对接口的格式和协议各不相同的设备如何处理,第二个是如何保证中间件的实时性。针对第一个问题我们决定使用代理和对象技术,给每一个设备(RFID 读写器、传感器等)做一个代理。尽管设备接口在格式和协议方面不同,但所有的代理通过 UDP/IP 上的 XML 信息通信,很好地保证了以后的可扩展性。代理作为它们相关的封装或翻译器使用。针对第二个问题,我们采用面向服务的 SOA 结构。

1.1 带有中间件的 RFID 应用软件的结构

典型的 RFID 应用系统的结构如图 1 所示。RFID 读写器读取 RFID 标签的数据位于较低的硬件层,读取的原始数据被传送到中间件处理。在中间件的设备和数据管理层,具有识别多次读取 RFID 标签的过滤机制以及过滤冗余数据的机制,数据被过滤后,只有和上层相关的数据才被传送到事件管理层。实时产生的 RFID 信息经过事件管理层处理供上层的交易过程和解决方案使用。

收稿日期:2007 - 10 - 19。

作者简介:成修治(1973 -),男,河南孟州人,硕士研究生,主要研究方向:智能感知与识别处理、中间件、大型软件架构; 李宇成(1958 -),男,北京人,教授,博士,主要研究方向:模糊控制、智能感知与识别处理、嵌入式。

中间件还有 API 的接口或基于开放标准的 XML Web service,属于这个层的其他服务包括智能交易、分析报告和事件通知。中间件为上层 RFID 应用程序提供服务接口,一般安装在数据中心。中间件作为商务应用的桥在后台存在,这种结构简化了 RFID 系统的开发、安装、操作和维护。

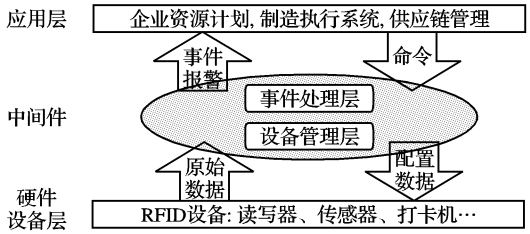

图 1 RFID 应用软件的结构

1.2 中间件中面向服务的结构

面向服务(Service Oriented Architecture,SOA)的中间件结构如图 2 所示。SOA 的目的是为用户提供容易维护的灵活的服务,采用 SOA 的系统具有可扩展性高,可维护性好的特点。在 SOA 环境中,应用系统的通信操作用服务接口来实现,而不是采用双向的接口,这意味着对单个软件组件的改变会非常柔性,不会影响到其他系统,也不必每次进行大量的集成测试。

SOA 的这种特点正符合 RFID 中件间的要求,可以解决本文引言中厂家提出的问题。SOA 可以使中间件一方面对多种多样的读写器类型做到很好的扩展,另一方面可以通过简单配置完成对多种应用系统的支持。实现 SOA 的常用的工具是 XML 语言,用来形成和它所表述的媒介无关的、统一的结构化信息。

图 2 面向服务的中间件

2 RFID 中间件的详细结构设计

中间件设计包括 RFID 设备管理组件和事件过程管理组件。RFID 设备管理组件是分布式的代理,它负责第一级的事件过滤。设备管理包括设备询问器,对每一个读写器和传感器设备,代理必须互相作用,过程管理组件通过 RFID 事件下一级的过滤,把事件放置到交易环境中,然后发布应用层事件(ALE),其结构如图 3 所示。

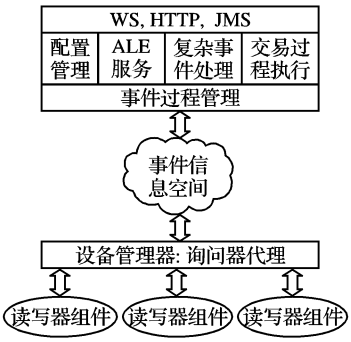

图 3 RFID 中间件的结构设计

2.1 设备管理

设备管理器提供远端设备的配置接口,管理一个或多个询问器到读写器和其他传感器,管理被指定的 ALE 事件。设备和询问器之间是一对多的关系。每一个询问器被分配一个物理设备概要表,每一个物理设备概要表可能有多个传感器(如多个天线)。当一个设备管理初始化自身的时候,会确定它负责哪一个物理设备表及其配置信息,然后安全地下载和初始化合适的询问器,最后注册事件。

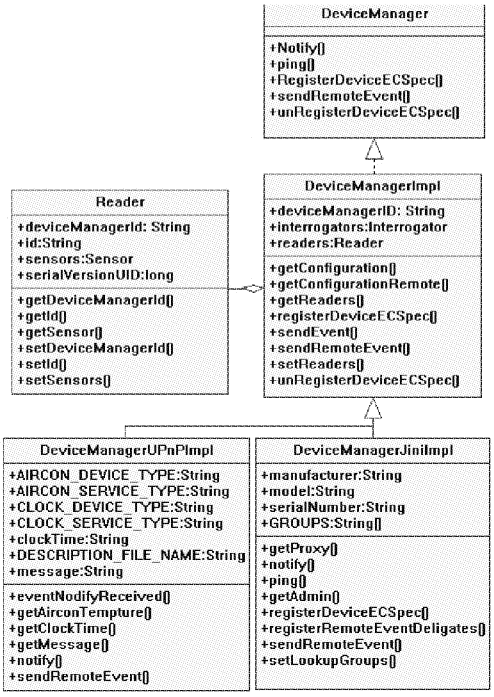

图 4 设备管理器的类图

设备管理器接听来自询问器的传感器事件,这些传感器事件和它们指定的读写器事件等同。来自多个询问器的传感器事件必须合并创建读写器事件,这些读写器事件和指定的由设备管理器发送到 ALE 服务的逻辑读写器事件相同。逻辑读写器可由任何数目的其他读写器构成,由一个或多个设备管理器的实例管理。设备管理器对参与到一个逻辑读写器中的其他读写器是什么没有反应的。同样一个读写器被限定到一个设备管理器的实例所控制的范围内。

由于每个 ALE 读写器事件流可能来自多个物理设备配置表,设备管理器为每个设备表创建一个询问器,并通知询问器什么样的传感器被绑定到指定的读写器上。询问器发送传感器事件流到设备管理器,设备管理器将一个或多个传感器事件流构造成读写器事件,这是因为读写器事件流可能来自不同的传感器事件流。设备管理器把初步处理的读写器事件发送到 ALE 服务器。设备管理器必须设定的唯一的物理识别符如下:设备管理 ID,是唯一的识别号;设备概要 ID,物理设备表的唯一的识别符;读写器 ID,每个传感器事件来源的唯一识别符,一个物理设备概要表可能有多个读写器;传感器 ID,唯一的物理设备(传感器)标志。

1)询问器代理

一个设备管理器的配置由它管理的设备和它要咨询的询问器组成,然后和它对应的设备管理器交互。每一个设备概要表由物理设备属性和询问器配置组成。物理设备属性是被命名过的传感器(例如天线和一个金属传感器)。

询问器代理是一个对应于特定读写器或其他设备的适配器实例。每个物理设备概要表有一个询问器代理,每个读写器模块有一个特定类型的代理。尽管一个询问器代理可以服务于多个物理设备概要表,但每一个物理读写器只能有一个询问器。每一个被指定设备的询问器可以由多个传感器,例

如多个天线或多个单线的设备。每个传感器有一个 SensorID。

设备管理把带有物理设备概要表的配置信息进一步组合。询问器代理把来自每个传感器的信息流发送到设备管理器,进而构造读写器事件。主要 UML 类图结构如图 5 所示。

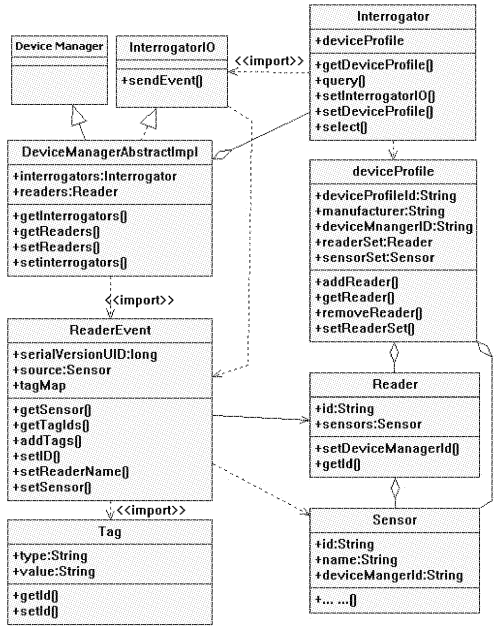

图 5　询问器代理的相关类图结构

2)事件信息空间

事件信息空间类似一个公共的容错事件信息经纪人。它支持异步接收来自设备管理器的事件,ALE 事件以及其他来自事件过程管理的配置需求。事件信息空间同时提供一个存储转发机制,确保重要的事件在中断的网络或其他组件失效的情况下不丢失。

2.2　事件过程管理

事件过程管理(Event Process Managment, EPM)由 ALE 服务、配置管理、复杂事件过程以及交易规则执行组成,对 EVP 的访问能通过 HTTP、JMS 以及网络服务接口来实现。EPM 登记/订阅它感兴趣的事件,这样当在信息空间中有事件时,它就会被通知,其结构如图 6 所示。

一旦接收到这些事件,随后会应用复杂事件处理(过滤器),结合交易规则对这些事件进行处理,或者在另一种情况下,外部的客户端(如 EPC-IS)已经注册接收 ALE,这些过滤后的事件会被发送到 ALE 客户端指定的位置。这种交易规则意味着:来自设备管理器的信号需要和其他企业信息系统的具体要求结合,才能达到交易过程的要求。

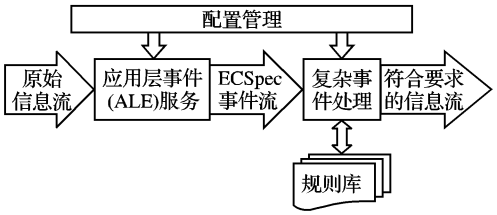

图 6　事件过程管理结构

1)配置管理

物理设备的物理信息例如 MAC 地址、IP、公钥、逻辑读写器任务等,被放在这个组件中管理。这些信息可以存放在本地,也可以放在通过密钥访问外部地址服务或者其他企业信息管理仓库中。

它包括下列项及其和其他部分关系的定义:传感器(天线、单一的事件)、物理设备概要表(可以有一个或多个传感器)、读写器(来自一个或多个物理设备概要表的传感器事件)、逻辑读写器。

2)应用层事件服务

这些服务是一个对 EPCglobal ALE 规范的实现。它会提供一个网络服务客户端接口以及和 ALE API 的 JMS 接口。ALE 规范中允许含有逻辑的读写器、被创建的逻辑读写循环、一个或多个读写器,以及一个更多的来自那些能组成一个逻辑事件的读写器的读写器事件。ALE 也规定了那些必须发送的逻辑事件。ALE 服务精心编制了设备管理器的参与者:逻辑读写器以及相应的逻辑事件, ALE 服务将这些事件通知到合适的接受器。

3)复杂事件处理

它是一种规则引擎,是一种过滤器,它把事件变为有意义的可用交易规则的交易事件。ALE 服务将维持复杂事件规则,随后产生有用的事件。这些事件被用来启动交易规则。

4)交易过程执行

因为中间件期望在各种商业环境被应用,所以具体的功能不能硬性规定必须链接到中间件中。交易过程一般可定义为:通过接口接听事件,通过使用规则引擎启动各种交易过程。ALE 提供了一个丰富的事件接口,但是这些事件的翻译以及随后必须的处理过程,只有在具体的环境才能最后明确。交易过程可以是一个可被关闭的门或启动的警告,等等。

当交易过程在一个较高的层次上时,需要和企业系统如仓储管理系统集成,所以在中间件层中的交易规则引擎应允许有较好的定制性,以及与其他底层设备相结合的可扩展性。

3　RFID 中间件的实现及测试

上述结构的中间件图形中间件界面利用 Eclipse Rich Client Platform 作为 GUI 开发的基本框架。在基本实现之后,对原来的餐饮管理系统软件进行了改造,并进行了初步的功能性测试,。实验环境的搭建如图 7 所示,现有的售饭终端是通过 485 组网的,台式机上安装相应的 485 网卡,也可通过外接 485/232 转换器接入计算机的串口。

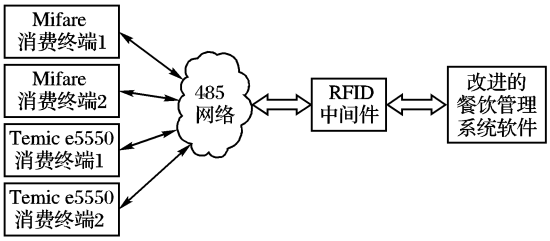

图 7　RFID 中间件的实验环境

实验表明,原来的餐饮系统仅仅支持一种固定卡型的读写器,如 Philiphs 的 Mifare 或 Mifare Ligh,或者是 Timic 公司的 e5550 系列,几年前的系统还有接触卡的读写器,由于随着时间的推移,卡片在不断进步,卡型也在不断变化,如果要更新整个硬件系统,会造成极大的浪费。现在采用了 RFID 中间件以后,可以在一个售饭系统中采用各种卡型的读写器,而上层程序不需要再进行修改,增加了系统的可扩展性和易维护性,节约了时间和成本,系统稳定性也有大的提高,有效地解决了引言中企业关心的问题。

(下转第 1060 页)

需求的模板,约束是需求的参数,元规格和约束的结合使抽象需求成为具体需求。

在模型中,元规格是文法推导的起点。规则库中的规则是上下文有关的规则,约束就是推导业务逻辑时的外部上下文。用户的约束作为筛选依据从规则库中筛选相应的规则进行推导,规则引擎(Rule engine)按照文法指导实现求精,就是元规格逐步求精的过程。约束是在推导过程中,对不同的路径进行选择的依据。同一个元规格,如果没有约束,可能产生不同的规则集合和业务逻辑结果。在推导过程中出现分支情况时,通过对用户需求约束和规则上下文进行匹配(match)处理,选中符合用户偏好的路径,从而保证最终的业务逻辑结果是用户所偏好的业务逻辑结果。

对元规格求精最终可以得到表现为树或森林的层次状结构模型。树或森林的叶子节点(如构件 a,b1,b2,b3)通过反射映射(mapping)到构件库中的物理构件,就能形成一个完整的反射体系。这样的反射体系能实现复杂的业务功能,并对外表现为构件的形式。

3 实例应用

读者向销售商发出包含书籍信息与数量的订购请求,销售商查询仓库向读者做出包含是否同意定购和书籍货款信息的回应;若读者和销售商双方同意,读者支付货款并向销售商发出已支付信息,销售商确认支付并发货,交易完成。如图2所示。

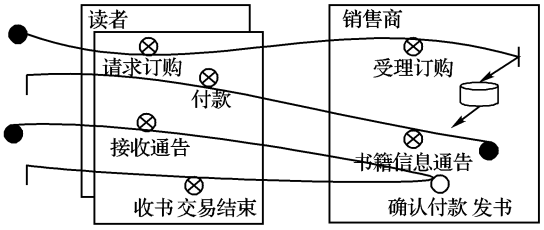

图2 图书订购过程

这是一个典型的观察者模式的应用。主要的求精过程属于行为求精。例如,在收到读者的订购请求后,当订购数量大于一定数值,销售商应向具有不同供应能力的供货商发出查询请求,供货商通过对自己的存储服务器查询,通知销售商查询结果,销售商根据查询结果做出决策。对销售商的求精过程如图3所示。

该求精过程是一个行为求精过程,可以用上面定义的上下文相关文法表示。假设复合销售商构件 CSA、CSB 都能完成销售商(S)功能,订购数量(BN)是选择求精结果的唯一依据。相应的上下文相关求精文法如下:

T = {CSA, CSB}
N = {S}
R = {(S, CSA, (BN, ≤, 1000)),
(S, CSB, (BN, >, 1000))}
S = {S, (BN, =, 500)}
C = {(BN, ≤, 1000), (BN, >, 1000)}

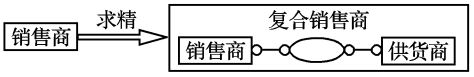

图3 销售商求精

4 结语

本文利用上下文有关的求精方法对构件进行求精,为构件的使用和复用提供了新的方式,且能保证构件在"有法可依"的原则下进行求精。也可以将构件看作广义的体系结构,将上下文有关的求精方法应用到更高的抽象层次上,指导体系结构级的求精。如果求精后的构件能够动态演化就能使构件具有更强的适应性。此外,如何保证构件求精前后的一致性(如状态、行为等),都将是下一步要研究的问题。

参考文献:

[1] 杨芙清,梅宏,李克勤. 软件复用与软件构件技术[J]. 电子学报,1999,27(2):68-75.

[2] 李长云. 基于体系结构的软件动态演化研究[D]. 杭州:浙江大学, 2005.

[3] 吕映芝,张素琴, 蒋维杜. 编译原理[M]. 北京:清华大学出版社,1998.

[4] CHOMSKY N. Three models for the description of language [J]. IEEE Transactions on Information Theory,1956, 2(3):113-124.

[5] 蒋宗礼,姜守旭. 形式语言与自动机理论[M]. 北京:清华大学出版社, 2002.

[6] McDANIEL P. On context in authorization policy[C]// Proceedings of the 8th ACM Symposium on Access Control Models and Technologies. New York: ACM Press, 2003:80-89.

[7] 许文,方海,林惠民. π演算互模拟判定算法的优化和实现[J]. 软件学报, 2001,12(2):159-166.

(上接第1057页)

4 结语

本文中我们提出了一种面向服务 SOA 的中间件的结构设计,并详细说明了中间件各部分的含义和作用。这种中间件结构能很好地屏蔽低端各种物理设备的信息,经过自身的处理将其上传给需要的应用程序。这种结构通过使用基础的服务适用于不同的硬件、操作系统和通信系统。这些服务能被嵌入到微内核的平台中,因此这种中间件能用到 PC 和低端的具有较少内存和 CPU 资源的嵌入式系统中。由于采取了模块化的结构,首先可以根据需要进行裁减,在需要的时候再加入相应的模块,如可根据需要是否添加认证和安全模块。下一步的工作是对中间件在嵌入式设备中的移植以及对中间件的性能测试。

参考文献:

[1] THOMPSON C. Smart devices and soft controllers[J]. IEEE Internet Computing, 2005,9(1): 82-85.

[2] THOMPSON C, PAZANDAK P, TENNANT H. Talk to your semantic Web[J]. IEEE Internet Computing, 2005, 9(6):75-78.

[3] SONG J, KIM H. The RFID middleware system supporting context-aware access control service[C]// The 8th International Conference on Advanced Communication Technology. [S.l.]: IEEE, 2006,1:20-22.

[4] FENG BO, LI JIN-TAO, ZHANG PING, *et al*. Study of RFID middleware for distributed large-scale systems[C]// Information and Communication Technologies 2006. Washington DC: IEEE Computer Society, 2006:2754-2759.

[5] HOAG J E, THOMPSON C W. Architecting RFID middleware[J]. IEEE Internet Computing, 2006,10(5): 88-92.

[6] MOON M, KIM Y, YEOM K. Contextual events framework in RFID system[C]// Third International Conference on Information Technology: New Generations. Washington, DC, USA: IEEE Computer Society, 2006:586-587.

[7] The Application Level Events (ALE) Specification, Version 1.0 [Z]. EPCGlobal Inc, 2004.