第 36 卷 第 4 期 计 算 机 工 程 2010 年 2 月
Vol.36 No.4 Computer Engineering February 2010
·工程应用技术与实现· 文章编号：1000—3428(2010)04—0223—03 文献标识码：A 中图分类号：TS801.8

JMF 消息通信在印刷系统中的应用

罗如柏[1]，周世生[1]，汪炜军[2]，高晓静[1]
(1. 西安理工大学印刷包装工程学院，西安 710048；2. 江西蓝天学院，南昌 330098)

摘 要：为了使印刷制造系统中的设备间能进行即时消息通信，在分析“询问-响应”和“信号”的作业消息格式(JMF)的消息通信基础上，给出基于 B/S 的计算机集成印刷系统。应用结果表明，通过套接字和简单对象访问协议，该系统能实现双向与单向 JMF 消息通信。
关键词：可扩展标识语言；作业消息格式；简单对象访问协议；套接字

Application of JMF Message Communication in Printing System

LUO Ru-bai[1], ZHOU Shi-sheng[1], WANG Wei-jun[2], GAO Xiao-jing[1]
(1. School of Printing and Packing Engineering, Xi'an University of Technology, Xi'an 710048; 2. Jiangxi Blue Sky University, Nanchang 330098)

【Abstract】 In order to realize real-time communications between equipments in printing production system, based on analyzing Job Message Format(JMF) communication modes"Query-Response"and"Signal", this paper gives computer integrated printing system based on B/S. Application results show thar this system can realize bidirectional and unidirectional JMF through socket and Simple Object Access Protocol(SOAP).
【Key words】 eXtensible Markup Language(XML); Job Message Format(JMF); Simple Object Access Protocol(SOAP); Socket

1 概述

在 20 世纪 90 年代，CIMS 概念开始进入印刷工业[1]。由于印刷系统是一个典型的异构系统，因此直到 2000 年作业定义格式(Job Definition Format, JDF)出现后，印刷系统的信息集成才初见曙光。JDF 是 CIP4 组织制定的一个基于 XML、能描述印刷生产全生命周期的数据标准[2]。

JDF 标准包含 JDF 和作业消息格式(Job Message Format, JMF)2 个部分。JDF 是定义印刷作业传票信息的作业格式，JMF 是定义印刷系统内实时通信的消息格式[1]。目前，国内研究 JDF 标准才起步，且多集中在对 JDF 作业格式的研究[3]。印刷系统内的消息通信主要分为无反馈的单向消息通信和有反馈的双向消息通信，它们是设备间高效协作的基础。因此，本文重点探讨基于 Web 的印刷车间的集成方法中 JMF 的双向消息通信和单向消息通信的实现方法，并通过建立原型系统实现 JMF 消息通信。

2 印刷系统中的 JMF 消息通信

2.1 JMF 消息簇

JMF 消息簇中的 JMF 消息可以用于系统设置、设备和作业的状态和错误跟踪、管道控制、设备设置和作业更换、队列处理、队列中作业优先权设置以及设备性能描述。JMF 消息是一个以“JMF”为根节点的 XML 小文档。JMF 根节点可以直接包含“Query”,“Command”,“Response”,“Acknowledge”,“Signal”和“Registration”中的一个或多个元素。根据 JMF 根节点中包含的元素，JMF 消息可以分别定义为询问、命令、响应、确认、信号和注册 6 类 JMF 消息。

询问消息是在不改变控制器(或设备驱动)状态的情况下为获得所需要的消息而从系统中的客户端(控制器)发送给控制器(或设备驱动)的询问，在发出一个询问消息后，会返回一个响应消息。响应消息用于回答询问消息或命令消息，是从控制器(或设备驱动)发送给递交询问消息或命令消息的控制器(或设备驱动)。确认消息是控制器异步响应命令消息(或询问消息)的一种消息，它可在一个长的反应时间后告知命令消息的发送者其命令消息被执行的结果。信号消息是在某些事件发生时发送给其他控制器(或设备驱动)的一种单向消息，它用于自动发布状态的改变。命令消息在语法上相当于询问消息，但它能改变目标设备驱动的状态且须返回一个响应消息。注册消息是 JDF1.3 新定义的一个 JMF 消息家族成员，它能够要求“该注册消息的接受者”发送命令消息到其指定的第三方“命令消息接收者”。根据 JMF 消息簇中各个消息类型的功能与响应特点，它们可形成多种 JMF 消息通信模式，如双向 JMF 消息通信模式有“询问-响应”、“命令-响应”和“命令-响应-确认”等，单向 JMF 消息通信模式有“信号”和“注册”。这些 JMF 消息通信模式将根据印刷系统内所需实现的消息通信功能而分别被使用[4]。目前 JDF1.3 将支持 JMF 消息通信的程度分为 4 个级别，从低到高分别是：不支持 JMF 消息通信，支持“信号”消息通信，支持“询问-响应”消息通信，支持“命令-响应”消息通信和支持通过 MIME 来接收 JDF 信息包[2]。

为了研究 JMF 消息通信的实现，本文重点讨论双向 JMF 消息通信模式“询问-响应”和单向 JMF 消息通信模式“信号”消息通信。

2.2 “询问-响应”和“信号”的 JMF 消息通信模式

在“询问-响应”的 JMF 消息通信模式中，当一个询问(Query)消息发出后，将立即返回一个响应(Response)消息。

基金项目：西安理工大学优秀博士学位论文研究基金资助项目(602-210802)
作者简介：罗如柏(1981－)，男，博士研究生，主研方向：数字化印刷工作流程，印刷复制技术；周世生，教授、博士、博士生导师；汪炜军，工程师；高晓静，硕士
收稿日期：2009-07-11 **E-mail：**luorubai@126.com

如果询问消息中包含了一个订阅(Subscription)元素，那么服务端就会在感兴趣的事件发生时持续地向指定的 URL 地址发送所需要的信号(Signal)消息，除非客户端向控制器发送一个包含“ StopPersistentChanne ”停止持续信道的命令消息。使用“ Subscription ”元素的 JMF 消息交互如图 1 所示。

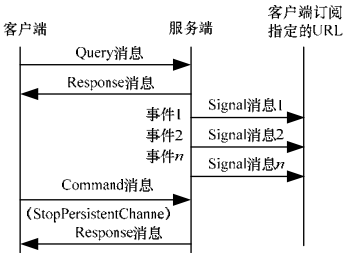

图 1　使用“ Subscription ”元素的 JMF 消息交互

例如，当 ID 为“ XAUT01 ”的控制器(Control)向 ID 为“ BR_002 ”的印刷机设备驱动(Device)询问该印刷机的状态时，首先需要由控制器发送一个 Query 消息给印刷机设备驱动，然后印刷机设备驱动接收该询问后根据实际情况立刻返回一个 Response 消息来告知设备的当前状态。发送的 Query 消息代码如下：

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" SenderID="XAUT01" TimeStamp="2008-11-05T21:19:54+08:00" Ver- sion="1.3">
<Query ID="M081105211954" Type="Status" xsi:type="QueryStatus">
<StatusQuParams DeviceDetails="Details" JobDetails="Brief"/>
</Query>
</JMF>
```

在上述代码中，根节点“ JMF ”直接包含“ Query ”元素，显示其为 Query 消息。在 JMF 元素的属性中定义了该 JMF 遵循的名字空间(“ xmlns ”属性)、消息的发送时间(“ TimeStamp ”属性)、JMF 标准遵循的版本(“ Version ”属性)和该消息的发送者(“ SenderID ”属性)。“ Query ”元素具有属性“ ID ”和“ Type ”。“ ID ”属性是用来唯一标识该 Query 消息，“ Type ”属性定义了询问消息所须询问的内容。当 Type=“ Status ”时，表示该 Query 消息是用于询问下游控制器或设备的情况。“ Query ”元素的子元素“ StatusQuParams ”是定义询问设备或作业的范围，当属性 DeviceDetails=“ Details ”时，要在回复的 Response 消息中包含“ Device ”元素来尽可能详细描述设备的细节信息，当属性 JobDetails=“ Brief ”时，要在回复的 Response 消息中包含“ JobPhase ”元素来详细描述设备正加工的印刷作业信息。因此，印刷机设备驱动在接收到该 Query 消息后会立即依据当前设备的状态信息返回如下 Response 消息：

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="BR_002" TimeStamp="2008-11-05T21:20:03+08:00" Version="1.3">
<Response ID="M081105212003" refID="M081105211954" Type=" Status ">
<DeviceInfo Speed="9000" DeviceStatus="Running" StatusDetails="Good">
<Device DeviceID="BR_002" DeviceType="B300"/>
<JobPhase JobID="MD-1234" Waste="84" Amount="100" Status="InProgress" JobPartID="MD-1234-PRINT-SIG001" StatusDetails="Good" PercentCompleted="1.20"/>
<Employee PersonalID="X001" Roles="Operator" Shift="P1">
<Person FamilyName="Luo" FirstName="Rubai" JobTitle="Master">
<Address Country="China" Region="Shaanxi" City="xi’an" Street="jinhua5" PostBox="801" PostalCode="710048"/>
<ComChannel ChannelType="Email" Locator="mailto: Luorubai@126.com"/>
<ComChannel ChannelType="Phone" ChannelTypeDetails="LandLine" Locator="tel:+81-029-92058800"/>
</Person>
</Employee>
</DeviceInfo>
</Response>
</JMF>
```

在上述代码中，根节点“ JMF ”直接包含“ Response ”元素，显示其为 Response 消息。“ Response ”元素所必需的属性有“ ID ”，“ Type ”和“ refID ”。其中，“ refID ”属性定义了该响应消息是回答哪个 Query 消息，其属性值是被回复的 Query 消息中“ Query ”元素的 ID 属性值。“ Response ”元素内包含了“ DeviceInfo ”元素，它用于详细描述设备的细节信息。其中，该设备正处于正式印刷阶段(DeviceStatus=“ Running ”)，机速为 9 000 张/h(Speed=“ 9 000 ”)。“ DeviceInfo ”元素又包含“ JobPhase ”，“ Device ”和“ Employee ”3 个子元素来分别描述设备的信息、当前正被加工的作业信息和当前的操作人员的信息。在此描述的印刷机正处理“ MD-1234 ”(JobID=“ MD-1234 ”)作业的“ MD-1234-PRINT-SIG001 ”(JobPartID=“ MD-1234-PRINT-SIG001 ”)子作业，并且已经完成正品 100 张(Amount=“ 100 ”)，占总生产计划的 1.20%(PercentCompleted=“ 1.20 ”)，但已产生了 84 张废品(Waste=“ 84 ”)。

“ 信号 ”的 JMF 消息通信可通过 3 种方法激发：(1)如图 1 中通过包含“ Subscription ”元素的询问消息发起初始的询问。(2)在 JDF 文档的 JDF 节点中通过“ NodeInfo ”元素内定义 Query 消息的 JMF 元素发起初始的询问，同样在该 Query 消息中也包含一个“ Subscription ”元素。这 2 种方法都需要一个起始的 Query 消息来订阅(激发)Signal 消息，不同的是传递起始询问消息的路径不同。(3)无需起始的 Query 消息，而是通过硬连线的(hard-wired)方式来建立信道。例如，当一个控制器(或设备驱动)的工作状态发生变化的时刻(如印刷机启动服务、换版开始、换版结束等时刻)就会自动产生一个信号消息，然后根据初始化的控制器的 URL 发送给各个控制器。本文将重点讨论硬连线的“ 信号 ”通信模式。例如当印刷机停止工作的时刻将发出如下 Signal 消息：

```
<JMF xmlns="http://www.CIP4.org/JDFSchema_1_1" SenderID="BR_002" TimeStamp="2008-11-05T21:30:03+08:00" Version="1.3">
<Signal ID=" M081105213003" Type="Status">
<DeviceInfo Speed="0" DeviceStatus="Down" StatusDetails="ShutDown" PowerOnTime="2008-11-05T21:00:01+08:00" CounterUnit="Sheets">
<Device DeviceID="BR_002" DeviceType=" B300"
Manufacturer="Beiren"/>
</DeviceInfo>
</Signal>
</JMF>
```

在上述代码中，根节点“ JMF ”直接包含“ Signal ”元素，显示其为 Signal 消息。它描述了编号为“ BR_002 ”(DeviceID=“ BR_002 ”)的印刷机是在“ 2008-11-05T21:00:01+08:00 ”

(PowerOnTime=“2008-11-05T21:00:01+08:00”)时刻启动的，此刻状态变化为关机状态(StatusDetails=“ShutDown”)。

3 JMF 消息通信的实现

3.1 基于 B/S 的计算机集成印刷系统

在本文中，笔者利用 Java 开发了一个基于 JDF 的用于胶印印刷车间信息集成的原型系统。本系统采用 B/S 结构体系，其示意图如图 2 所示。由于 JDF 文档和 JMF 文档实质是 XML 文档，考虑到 SQL Server 2005 在管理 XML 文件的优越性，系统采用 Microsoft SQL Server 2005 数据库。Web 服务器则选用 Tomcat 5.0，网络编程语言是 JSP。Web 服务器所连“设备”是用 Java 开发的模拟胶印机控制台的 GUI 程序。

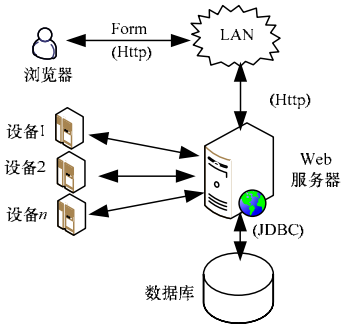

图 2 基于 B/S 的计算机集成印刷系统构架

在本系统中，用户可使用 Web 浏览器访问 Web 页，通过 Web 页上显示的表格与 Web 服务器进行交互操作。用户可以在 Web 页向 Web 服务器所连的“设备”发送询问设备状态的 Query 消息，“设备”会在接收消息后根据状态信息立刻回复一个 Response 消息，并显示在用户的 Web 页上；“设备”在其工作状态发现变化时会发送一个 Signal 消息给 Web 服务器，Web 服务器内的 Servlet 程序将根据 JMF 消息相应地修改数据库中正被加工的印刷作业的 JDF 文档。

3.2 基于 Socket 的双向 JMF 消息通信

为实现用户询问设备状态的双向 JMF 消息通信，分别在 Web 服务器端用套接字程序实现一个“生产监视”模块，在“设备”GUI 程序内开发一个“状态管理”模块。用户首先在 Web 页通过表单数据向“生产监视”模块提交状态询问需求，然后“生产监视”模块采用 Windows Socket 技术向被询问“设备”的“状态管理”模块发送询问设备状态的 Query 消息。“状态管理”模块在接收到 Query 消息后，利用 DOM 解析此 JMF 文档获得询问需求，根据需求立刻生成一个描述设备状态的 Response 消息并反馈给“生产监视”模块。“生产监视”模块在接收到 Response 消息后，立刻用 DOM 解析此 JMF 文档，然后将其中描述设备状态的参数以表单数据形式提交给用户的 Web 页并显示，从而实现一次设备状态的询问与反馈过程。“生产监视”模块和“状态管理”模块间的“询问-响应”双向 JMF 消息通信实现原理如图 3 所示，其中，Q 表示询问设备状态的 Query 消息；R 表示回复设备状态的 Response 消息。Windows Socket 是 TCP/IP 的一个接口，Socket 通过 TCP/IP 协议为“生产监视”模块和“状态管理”模块提供交互通信的可靠连接[5]。

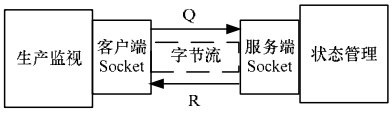

图 3 基于 Socket 的双向 JMF 通信

在 Java 实现时，当“设备”的 GUI 程序启动时，“状态管理”模块创建一个服务端 Socket 对象并开始监听，“生产监视”模块在发送 Query 消息时会创建一个客户端 Socket 对象，并向被询问的服务端 Socket 对象发送连接请求。服务端 Socket 对象接收到连接请求后，用 Socket.accept()方法创建新的 Socket 对象用于和发出请求的客户端 Socket 对象进行通信。即新的 Socket 对象用 Socket.getInputStream()方法接收客户端 Socket 发送的含有 Query 消息的 ObjectInputStream 对象，并将其强制转型为 DOM 中的 Document 对象，解析该对象后获得消息中的询问信息，然后用 Socket. getOutputStream()方法创建一个 ObjectOutputStream 对象，并根据询问信息生成 Response 消息并存放在另一个 Document 对象内，随后用 ObjectOutputStream.writeObject()方法将此 Document 对象写入到 ObjectOutputStream 对象中，最后用 ObjectOutputStream.flush()方法将 ObjectOutputStream 对象发送给客户端的 Socket 对象，随后关闭此 Socket 对象。客户端的 Socket 对象也用 Socket.getInputStream()方法接收含有 Response 消息的 ObjectInputStream 对象，随后关闭客户端的 Socket 对象。

3.3 基于简单对象访问协议的单向 JMF 消息通信

简单对象访问协议(Simple Object Access Protocol, SOAP)是一个 Web 服务标准，使用 Http 协议来实现 RPC 通信，且允许服务提供者和服务客户在 Internet 中经过防火墙进行通信交互。SOAP 对它的有效负荷用 XML 进行编码作为 SOAP 体，并封装在 SOAP 封套中[6]。JMF 消息是 XML 小文档，在实现“信号”消息通信时，可将 Signal 消息作为 SOAP 体封装在 SOAP 封套中，然后通过 SOAP 消息通信实现 JMF 消息通信，其实现原理如图 4 所示。实现的通知“设备”工作状态变化的单向 JMF 消息通信发生在 Web 服务器端的“生产监视”模块和“设备”GUI 程序内的“状态管理”模块间。

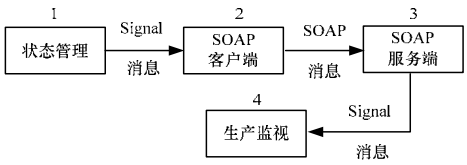

图 4 基于 SOAP 的单向 JMF 通信

其中，过程 1 表示状态变更时生成描述“设备状态”的 Signal 消息；过程 2 表示将 Signal 消息封装在 SOAP 封套中，并生成一个 SOAP 消息；过程 3 接收 SOAP 消息，打开 SOAP 封套，并从 SOAP 体中析取 Signal 消息；过程 4 接收并处理 Signal 消息。

用 Java 在 SOAP 客户端实现生成 SOAP 消息时，首先将“状态管理”模块生成的 Signal 消息用 DOM 解析到 Document 对象中，然后用 2 个 Vector 对象分别定义 SOAP 报头(SOAP-ENV: Envelope 元素)和(SOAP-ENV: Body 元素)，随后在其中一个 Vector 对象中创建 SOAP 报头的内容，Signal 消息的内容则通过 Vector.add(Document.GetDocument Element())方法加载给另一个 Vector 对象而形成 SOAP 体，最后创建一个 Envelope 对象为 SOAP 封套，并利用 envelope.setHeader()方法和 envelope.setBody()方法将分别含有 SOAP 报头和 SOAP 体的 Vector 对象加载到 Envelope 对象中，从而形成如下 SOAP 消息：

(下转第 232 页)