文章编号:1007 - 2985(2013)02 - 0026 - 05

# 基于 Spring Integration 消息流程的文件监控*

丁振凡
(华东交通大学信息工程学院,江西 南昌 330013)

**摘 要**:Spring Integration 提供一种简单、高效的机制实现面向消息的应用集成,并采用声明式配置定义消息通道与消息处理部件的联系,SpringSource Tool Suite 提供可视化工具箱实现消息处理流程的编排.结合某目录下文件安全检查的应用实例,给出 Spring Integration 的具体编程处理方法,通过消息文件存储某目录下所有文件的状态信息.待检测时将目录下文件的信息与消息文件记录的信息进行比较,从而发现文件的变化并作相应处理,通过自动发送邮件通知管理者.

**关键词**:spring integration;消息;消息通道;消息处理部件;文件变化监控
**中图分类号**:TP391 **文献标志码**:A **DOI**:10.3969/j.issn.1007 - 2985.2013.02.006

近年来,面向服务的体系结构(SOA)成为应用集成的主流,而面向消息数据交换是 SOA 应用的主要特征[1].基于消息的数据交换实现发送者和接收者之间的耦合,有利于增量式开发应用.Spring Integration 是 Spring 的一个扩展框架,在继承了 Spring 的 IOC 机制的基础上,提供了一个轻量级的、声明式模型,从而实现面向消息的应用,支持消息驱动和事件驱动机制[2],并通过适配器连接各类消息源(如 File,JMS,JDBC,HTTP 等),实现与外部系统集成.笔者在 Spring Integration 基础知识体系的基础上,结合服务器的文件监控应用提出了具体编程处理方法.

## 1 Spring integration 消息处理

### 1.1 消息的构建

消息(Message)提供了对 Java 对象及元数据的一个通用包装机制,包括 header(消息头)和 payload(消息负载),通过 Message 接口的 getHeaders()方法可得到消息头,通过 getPayload()可得到消息负载.

构建消息有 2 种方式:一种使用消息接口的实现类 GenericMessage〈T〉构建,消息负载可以是任何可持久化的 Java 对象,以下为含消息头参数的构造方法具体格式为 GenericMessage〈T〉(T payload,Map〈String,Object〉 headers);

另一种使用 MessageBuilder 工具类,创建 1 个字符串消息为

Message〈String〉m= MessageBuilder.withPayload("hello").build();

### 1.2 消息通道

消息通道(MessageChannel)是传送消息的部件[3],接口定义了发送消息的 send 方法,MessageChannel 接口有 2 个子接口:一个是 PollableChannel 接口,用来接收消息,可缓存消息,其中含接收消息的 receive()方法;另一个是 SubscribableChannel 接口,用于发布/订阅形式的消息通信,无需缓存,其中含支持消息的订阅/取消的 subscribe 和 unsubscribe 方法.按消息递交处理方式,消息通道可分为点对点通道和

* **收稿日期**:2013 - 01 - 26
**作者简介**:丁振凡(1965 - ),男,江西丰城人,华东交通大学信息工程学院教授,硕士生导师,主要从事云计算与语义 Web 研究.

发布/订阅形式通道 2 大类.

1.2.1 点对点形式通道　点对点通道只有 1 个接收者,包括直接通道(DirectChannel)、队列通道(QueueChannel)、优先级通道(PriorityChannel)等.队列通道支持消息缓冲队列,按 FIFO 规则处理消息,而优先级通道允许指定消息的优先顺序.最简单的直接通道的定义形式为〈int:channel id=" identification "/〉.

1.2.2 发布/订阅形式通道　发布订阅形式通道(PublishSubscribeChannel)用于发布/订阅模式的通信,发布者通过该通道广播消息,所有订阅者将接收到消息.任何订阅者必须是 1 个 MessageHandler 类型的对象,其中含有 handleMessage 方法实现消息处理逻辑.发布订阅形式通道的定义形式为〈int:publish-subscribe-channel id="pubsub-channel" /〉.

### 1.3 消息处理端点

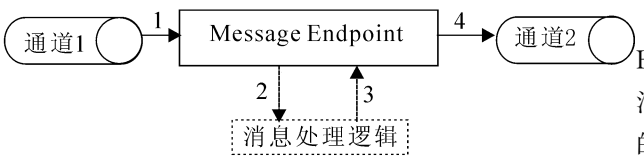


**图 1　典型消息处理部件的处理逻辑**

Spring integration 的消息处理部件(Message Endpoint)实现功能服务与消息框架的连接,用于对消息进行加工处理,图 1 给出了典型消息处理部件的逻辑处理过程.消息处理部件通常包括输入通道、输出通道和业务处理服务逻辑.其处理过程为:(1)从输入通道获取消息;(2)消息处理部件调用业务逻辑,此时消息内容将传递给业务逻辑的方法参数;(3)返回业务逻辑的执行结果给消息处理部件;(4)将结果包装为消息送输出通道.

有的 Endpoint 还可以配置 poller 属性,用来引进事务、触发器、定时器等相关逻辑[4-5].表 1 给出了主要消息处理端点的功能,综合运用这些部件可实现功能强大的应用.

**表 1　主要消息处理端点的功能**

| Message Endpoint 类型 | 使用说明 |
|---|---|
| 消息转换器 | 进行消息内容或结构的变换.如对象到串、对象到 json、对象到 map 等转换器 |
| 过滤器 | 用于发布订阅模式,决定是否消息可传递到相应通道 |
| 路由器 | 根据消息内容选择消息发送的通道 |
| 消息分离器 | 将消息分解为多个消息发送到相应通道 |
| 消息聚合器 | 将多个消息组合为一个消息 |
| 服务激活器 | 是最典型的消息处理端点,用于对消息进行分析加工处理 |
| 通道适配器 | 实现消息源或消息目标与消息系统的连接.根据适配器是提供消息还是接受消息可分为 inbound 和 outbound 这 2 大类.Spring Integration 提供有 File,HTTP,JDBC,JMS,FTP,MAIL 等众多主流适配器 |

## 2　面向消息处理流程的文件监控

检查某目录下文件的安全由 2 步构成:将某目录下所有文件的状态信息通过消息存储保存到消息文件中;在安全检查时将消息文件中的信息读取,与当前文件的状态信息进行比较,从而发现变化的文件.该应用除需要引入 Spring 框架的 jar 包和 spring-integration 的 jar 包外,因任务定时需要,还要引入面向切面编程的依赖包 aopalliance.jar.为实现 Java 对象到 Json 串转换,应用采用了 Google 的 gson 工具,要引入 Google 的 gson-1.7.1.jar 包.

### 2.1 目录下原始文件状况的登记

2.1.1 对文件记录信息的包装　文件登记中记录的不是文件对象,而是文件的部分信息,为此专门定义一个 FileInfo 类实现对文件名称(filename)和最后修改时间(lastmoditime)的封装,某目录下文件的增、删、改都可通过该类的属性检查即可判别.

2.1.2 某目录下文件列表信息的获取　以下代码将获取某目录下的所有文件列表,并保存到消息文件中.由于消息文件中存储的消息负载只能是字符串类型,因此需要将对象变换为串类型,采用 Google 的 Json 工具实现转换[6].

```
public class initProcess {
publicString logTo(String sdir) {// 被检测的服务器的
目录路径
File d=new File(sdir);
File[] flist= d.listFiles();//获取目录下所有文件
FileInfo info[]=new FileInfo[flist.length];//数组保
存文件信息
for (int k=0;k< flist.length;k++){
info[k]=new FileInfo();
info[k].filename= flist[k].getName();
info[k].lastmoditime= flist[k].lastModified();
}
return new Gson().toJson(info);//将数组转换为
JSON 串
}
}
```

Json 转换还有一种方法是用 Spring Integration 提供的对象到 Json 的消息转换器,这时,需要在工程中引入 jackson 的 jar 包.

2.1.3 消息处理的配置文件 Spring Integration 的消息处理部件支持 XML 和注解配置. SpringSource Tool Suite 提供了一组工具箱,用可视化方式进行部件的拖放连接,可通过可视化方式定义部件的属性. 可视化的流程编排对应有 XML 配置源码,图 2 为文件初始登记的处理流程.

点击图 2 中底部的"Source"选项卡可得到 XML 形式的配置源码(config1.xml):

```
<bean id="init" class="initProcess" />
<int:channel id="channel1"></int:channel>
<int:channel id="channel2"></int:channel>
<int:service-activator input-channel="channel1"
ref="init"
output-channel="channel2">
</int:service-activator>
<file:outbound-channel-adapter directory="f:x"
channel="channel2">
</file:outbound-channel-adapter>
```

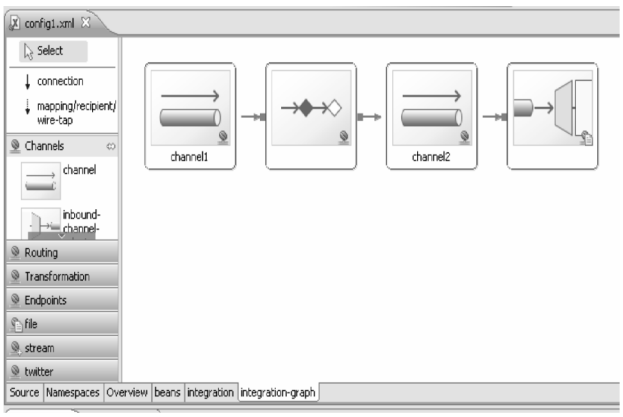


**图 2 文件初始登记的消息流处理流程**

从图 2 的流程和配置代码可看出,应用将从通道"channel1"获取要进行登记检查的目录路径信息,消息经过服务激活器调用标识为"init"的 Bean 的匹配方法(logTo)进行处理,方法返回结果包装为消息通过通道"channel2"送文件出口适配器,将消息以文件形式保存到指定目录(f:x).

2.1.3 在应用程序中装载配置,发送消息激活初始登记 文件初始登记可安排在某个独立的应用程序中,也可有 Web 应用访问触发. 基本工作是装载 XML 配置,将要进行安全检查的具体目录信息包装为消息发送到通道"channel1".

```
ApplicationContext context =new
ClassPathXmlApplicationContext("config1.xml");
MessageChannel input = (MessageChannel) context.
getBean("channel1");
String dirpath="D:SCI";//需要进行文件安全检查的
目录
input.send(new GenericMessage(dirpath));//给通道
channel1 发送消息
```

## 2.2 文件变化检查处理

文件变化检查的处理过程:(1)读取消息文件中登记的信息;(2)登记信息与目录下当前文件信息进行比对,记录变化;(3)将发生变化的文件进行处理,并自动发送邮件通知管理者. 文中将自动发送邮件 Bean 的配置略,利用 Spring 框架的 mail 包的 SimpleMailMessage 实现邮件消息的封装,利用 mail.javamail 包的 JavaMailSenderImpl 实现邮件的发送[7]. 由于 Spring 框架中邮件发送依托 JDK 的 mail 扩展包,所以需要在工程的类路径中加入 javax.mail.jar 包. 利用 Spring Integration 提供的 mail 适配器也可实现邮件发送处理,但要求在消息头中封装邮件的消息头信息.

2.2.1 处理流程的配置文件 图 3 是文件变动检查的消息处理流程. (1)通过入口适配器的 poller 配置触发检查过程,它将每天定时给目录通道"dirpath"发送检查目录;(2)"文件变动检查"服务激活器将当前文件状况与消息文件中记录的信息进行比较,得到变化文件信息;(3)文件变化信息通过通道"channel1"发送给"文件变动处理"的出口适配器进行处理.


**图 3　消息处理流程**

以下为配置对应的 XML 源代码(config2.xml):

```
〈beans……〉
〈bean id="begin" class="BeginWork" /〉
〈bean id="compare" class="CompareFile" /〉
〈bean id="process" class="Process" 〉
〈property name="mailSender" 〉
    〈ref bean="testMailSender"/〉
〈/property〉
〈/bean〉
〈int:channel id="dirpath"〉〈/int:channel〉
〈int:channel id="channel1"〉〈/int:channel〉
〈int:service-activator input-channel="dirpath"
ref="compare"
output-channel="channel1"〉
〈/int:service-activator〉
〈int:inbound-channel-adapter ref="begin" channel=
"dirpath" method="begin"〉
〈int:poller cron="10 50 23 * * *" /〉
〈/int:inbound-channel-adapter〉
〈int:outbound-channel-adapter channel="channel1"
ref="process" method="processChange"〉〈/int:
outbound-channel-adapter〉
〈/beans〉
```

其中“int”为 Integration 的名空间,配置中最关键的是输入通道适配器的任务定时执行,这里采用 Cron 定时,规定每天晚上 11∶50∶10 激活文件检查处理,也可改用间隔延时处理.

2.2.2 入口适配器的定时处理业务逻辑　入口适配器主要实现定时处理,通过配置设置让其定时执行 BeginWork 类中 begin 方法,方法的返回结果将自动包装成消息发送给适配器的输出通道“dirpath”.

```
public class BeginWork {
public String begin(){
return "d:SCI";//被检查目录
}
}
```

2.2.3 文件变动检查　文件变化检查的关键是消息文件中记录信息的读取处理. 有 2 种方法读取消息文件:通过文件入口适配器;通过 FileReadingMessageSource. 笔者采用后一种方法,在程序的 logToXML 方法中借助 FileReadingMessageSource 类读消息文件中的内容[8],由该类的 receive 方法得到消息,进而通过消息对象的 getPayload 方法得到消息负载,它是一个文件对象,用 BufferedReader 流访问该文件即可得到文件内容,再通过 Json 反转换得到原始文件列表的信息.

```
public classCompareFile {
public Map〈String,List〈File〉〉 logToXML(String
dirpath) {
// ① 读取原始登记文件的消息
FileReadingMessageSource src = new FileReading
MessageSource();
src.setDirectory(new File("f:x"));//消息文件的存储
位置
Message〈File〉 msg = src.receive();//从消息源获取
文件消息
File y = msg.getPayload();
BufferedReader x = new BufferedReader(r);
String result = x.readLine();//读取文件中存储的
Json 串
Gson g = new Gson();
FileInfo [] x1 = new FileInfo[10];
x1=g.fromJson(result,x1.getClass());//将 Json 串
反转为 FileInfo 数组
……// ② 获取 dirpath 目录下的文件列表
……// ③ 进行比较,将比较结果存储到 map 映射集
合中
Map〈String,List〈File〉〉 map=new HashMap〈String,
List〈File〉〉();
List〈File〉 newfile=new ArrayList〈File〉();//记录新
增文件
    …… //比较,得到新增文件信息加入 newfile 列表
map.put("新增文件",newfile);//将新增文件加入
map
```

```
    …… // 文件删除、修改进行类似处理                    }
    return map;// ④ 返回反映文件变化的映射集合          }
```

2.2.4 文件变动处理　在流程的出口适配器配置中设置了处理逻辑,它将分析文件变化消息,并进行相应处理,通过自动发送邮件通报给管理者[9].

```
    public classProcess  {                              …… // 从 map 中获取有变化的文件列表,并发送邮
    private TestSenderMail mailSender;//属性注入自动发   件通知
    送邮件程序                                          }
    public void processChange(Map<String,List<File>> f)  }
{
```

对于文件的各类破坏恢复,本系统采用的办法是通过 Web 界面处理,对改动和删除文件进行异地备份重传覆盖. 对新增文件进行删除处理,每次恢复完毕后重新执行初始化登记.

## 3 结语

结合网站目录下文件安全检测应用,提出了 Spring Integration 的应用编程处理方法. 对 Spring Integration 中消息、消息通道的构建及消息处理部件的应用进行了阐述. 该应用可部署到 Web 应用环境中,采用 Spring MVC 编程设计应用操作界面[10-11],根据浏览器的请求执行操作,实现文件监控的远程初始化处理,并启动定时监控服务,实现对服务器文档的远程监控,通过邮件得知文件变化,进而采取相应管理对策.

**参考文献:**


[1] 谢承旺,周　娟. 电力企业信息系统应用集成技术研究 [J]. 华东交通大学学报,2012,29(2):27-30.

[2] DANIEL J BARRETT,LORI A CLARKE,PERI L TARR,et al. A Framework for Event-Based Software Integration [J]. ACM Transactions on Software Engineering and Methodology,1996,5(4):378-421.

[3] 阳王东,祝　青,习胜丰,等. 一种面向频道的消息服务模型 [J]. 计算机工程,2010,36(2):67-69.

[4] MARK FISHER,JONAS PARTNER,MARIUS BOGOEVICI,et al. Spring Integration in Action [M]. USA:Manning Publishing Company,2012.

[5] 孙　垚,廉东本. 一种基于 ESB 的高效可靠的动态路由模型 [J]. 计算机系统应用,2011,20(2):144-148.

[6] 丁振凡. Spring REST 风格 Web 服务的 Json 消息封装及解析研究 [J]. 智能计算机与应用,2012,2(2):9-11.

[7] 钟　珞,刘　玲,夏红霞. 基于 JavaMail API 的 Web 邮件系统开发 [J]. 武汉理工大学学报,2006,28(6):84-86.

[8] DR MARK LUI,MARIO GRAY,ANDY CHAN,et al. Pro Spring Integration [M]. USA:Apress,2011.

[9] 梁富厚. 支持业务协同的集成化多通道消息模型研究及实现 [D]. 济南:山东大学,2011:31-32.

[10] 丁振凡,吴根斌. Spring 3.x MVC 模型的数据校验国际化处理 [J]. 计算机时代,2012(8):26-28.

[11] 丁振凡. 用 Spring MVC 实现数据分页显示处理 [J]. 智能计算机与应用,2012,2(5):20-22.


# File Monitoring Based on Spring Integration Message Flow


DING Zhen-fan

(School of Information Engineering,East China Jiao Tong University,Nanchang 330013,China)



**Abstract**:Spring Integration provides a simple,efficient mechanism to realize message oriented application integration. It adopts declarative configuration definition connection between message channel and message processing unit. Spring Tool Suite provides a visual toolbox to realize message flow process layout. Combined with a directory file security inspection application,this paper gives the specific programming processing method. Through message files all documents state information is stored in a directory. When detecting,the file information in the directory is compared with the recorded information from the message file,in order that the file changes should be found,and a notice mail be automatically sent to the manager.



**Key words**:spring integration;message;message channel;message endpoint;file change monitoring

(责任编辑　陈炳权)