DOI:10.19850/j.cnki.2096-4706.2020.18.035

RabbitMQ 消息中间件在 Spring Boot 教学中的应用

唐权，周蓉，张勇

（四川职业技术学院，四川 遂宁 629000）

摘　要：文章对消息中间件的基本工作原理进行了研究，分析了消息中间件的异步处理应用场景，详细研究了消息中间件RabbitMQ工作原理及工作模式，在Spring Boot教学项目中整合RabbitMQ消息中间件技术，实现业务逻辑前后台分离的异步处理方式，提供了消息中件间在Spring Boot项目中的解决方案。

关键词：RabbitMQ；消息中间件；Spring Boot

中图分类号：TP393；TP391　　**文献标识码**：A　　**文章编号**：2096-4706（2020）18-0125-03

Application of RabbitMQ Message Middleware in Spring Boot Teaching

TANG Quan，ZHOU Rong，ZHANG Yong

（Sichuan Vocational and Technical College，Suining 629000，China）

Abstract：This paper studies the basic working principle of message middleware，analyzes the application scenarios of asynchronous processing of message middleware，studies the working principle and working mode of RabbitMQ in detail，integrates RabbitMQ middleware technology in the teaching project of Spring Boot，realizes the asynchronous processing mode of separating business logic from background，and provides a solution for the message middleware in the Spring Boot project.

Keywords：RabbitMQ；message middleware；Spring Boot

0　引　言

JavaEE轻量级应用程序的开发是我校软件技术专业高年级很重要的核心专业课程，该课程综合性很强，经常引入第三方软件整合到框架中解决实际复杂问题。例如：引入Spring MVC框架解决模型层、表示层、控制层业务逻辑分离与控制问题；引入MyBatis框架解决数据库访问层的对象关系映射（ORM）问题等。同样在JavaEE项目中会遇到这样的问题：业务的处理过程涉及不同的业务流程，各个流程之间进行通信和数据交互，如果不同的流程处于不同的系统中还涉及远程调用。采用传统模式，两个流程之间会形成一种同步机制，满足一定的逻辑关系与时间顺序，在执行过程中存在相互制约，影响程序执行效率与响应速度。为了解决这种不同业务流程之间的通信与数据交互，就需要引入第三方消息中间件，实现系统的异步通信、应用解耦、流量削峰、分布式事务管理，以解决不同系统之间的信息的产生与消息消费异步实现机制，大幅提升了程序的运行效率及响应速度。在教学项目中对掌握消息中间件技术，理解消息中间件的工作原理及应用场景，应用消息中间件实现消息通信具有非常重要的作用。

1　消息中间件

1.1　消息中间件工作原理

消息中间件是基于队列与消息传递技术，在网络环境中为应用系统提供同步或异步、可靠的消息传输的支撑性软件系统[1]。消息中间件的由消息代理服务器（Broker）、消息生产者（Producer）、消息消费者（Consumer）、主题（Topic）、消息队列（Queue）、消息体（Message）组成。

消息中间件的工作原理：系统A作为消息生产者把消息发布到消息中间件服务器的消息汇集地，形成不同消息队列，供系统B的消费者使用的过程，其工作原理如图1所示。在业务流程中加入消息服务中间后，系统A与系统B不直接发生联系，形成了系统A→Broker和Broker→B两个独立的过程，完成A→B的异步通信，实现A的业务逻辑与B的业务逻辑解耦。

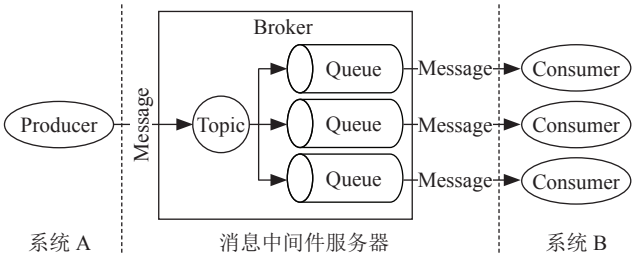

图1　消息中间件工作原理

1.2　消息中间件的应用场景

消息中间件可以在实际开发中应用于异步处理、应用解耦、流量削峰、分布式事务处理场景，提升响应速度，解决相互依赖，实现过载保护、冗余性和可恢复性。下文以异步处理机制来详细说明消息中间件的应用场景。

在客户交易业务逻辑中，完成交易后对数据库进行更新，然后还要把该交易信息通过短信和电子邮件的方式发送

收稿日期：2020-08-10

基金项目：四川省教育厅理科重点项目（16ZA0377）

给客户，处理以上的业务有三种方式可以选择：

（1）串行处理方式：包括三个顺序业务逻辑，交易数据写入数据库→向客户发送交易邮件→向客户发送交易短消息，整个业务响应所消耗的时间是三个业务逻辑所需时间之和，即为 90 ms，如图 2 所示。

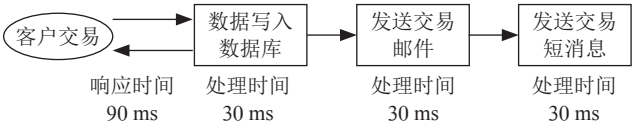

图 2　串行处理

（2）并行处理方式：把向客户发送交易邮件与发送交易短消息两个业务逻辑通过多线程并行处理，其他业务逻辑不变，整个业务响应消耗的时间是数据写入数据库的耗时与后面两个并行业务逻辑耗时最长的一个之和，即为 60 ms，如图 3 所示。

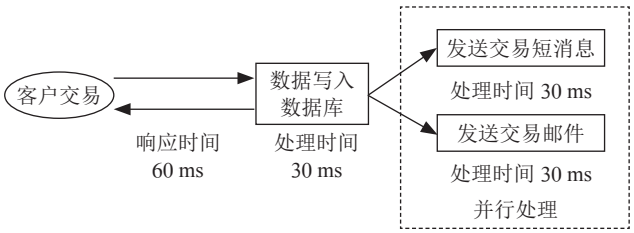

图 3　并行处理

（3）消息中间件处理方式：将交易数据写入数据库，引入消息中间件把消息快速推送到消息服务器的队列中后，结束前台交易逻辑，发送交易邮件与发送交易短消息由后台交易业务配合消息中间件完成，响应业务的时间缩短为 35 ms，工作流程如图 4 所示。

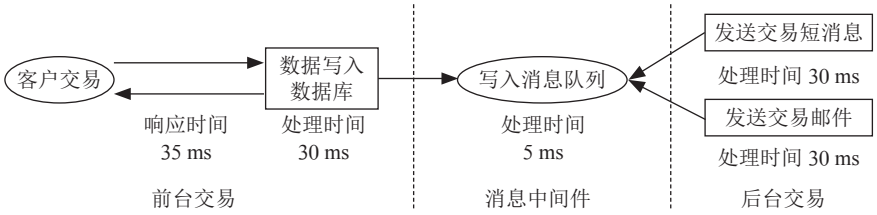

图 4　消息中间件异步处理

通过异步处理应用场景的分析，消息中间件异步处理方式最节省时间，耗时为 35 ms，响应用户的请求最及时。在多用户、大流量的场景中，提高业务的响应速度具有极其重要的作用，直接影响应用程序的性能，可提升用户的使用体验与黏合度。在业务中把即时性要求高的业务流程放在前台处理，把对时间不太敏感的业务，通过引入消息中间件作为后台业务处理，实现业务分离，提升应用程序对客户请求的响应速度，从而提高应用程序的处理效率。常用的开源消息中间有 ActiveMQ、RabbitMQ、Kafka、RocketMQ，在项目的开发中可根据项目的需求特点，选择合适的消息队列中间件。

2　RabbitMQ 消息中间件

2.1　RabbitMQ 消息中间件工作原理

RabbitMQ 是基于 AMQP（Advanced Message Queuing Protocol）的轻量级、可靠、可伸缩和可移植消息代理，在 Spring Boot 项目中通过 AMQP 协议与 RabbitMQ 通信，进行集成化管理 [2]。其工作原理是消息发布者向 RabbitMQ 代理服务器发送消息，代理服务器内部的交换器接收信息，把消息传递到与交换器绑定（Binding）的消息队列中并保存，消息消费者通过与消息代理服务器建立连接（Connection），然后从对应的消息队列中取出消息进行处理的过程，其工作原理如图 5 所示。

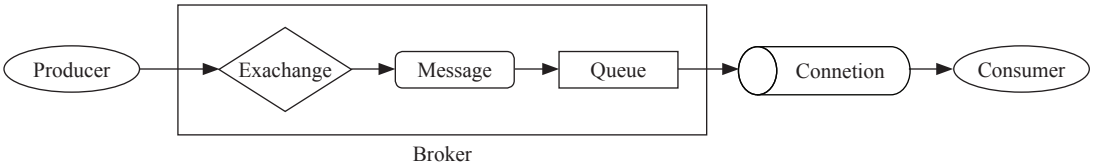

图 5　RabbitMQ 工作原理

2.2　RabbitMQ 消息中间件工作模式

RabbitMQ 消息中间件根据不同的需求，提供了多种模式，主要包括工作队列模式、发布订阅模式、路由模式、通配符模式、RPC 模式、Headers 模式，其中最常用的四种工作模式的特点为：

（1）工作队列模式：在该模式下使用内部默认的交换器把消息发送到唯一的消息队列，该消息可供多个消费者使用，使用后消息被移出。这种模式适合于工作任务相同，但是工作任务较繁重的情况，把任务分给多个消息者轮流处理。

（2）发布订阅模式：在中间件服务器设置一个 fanout 类型交换器，收到的消息不加以分类，全部分配到不同消息队列加以存储，然后由各自消息队列关联的消费者进行消费。这种模式适用于使用相同消息，但是工作任务不同的场景。

（3）路由模式：设置一个 Direct 类型交换器，指定不同的路由键值（Routing Key），根据键值不同对消息进行分类，把不同的消息分配到不同消息队列供消费者使用。这种模式适用于不同任务处理不同消息的场景。

（4）通配符模式：功能路由模式类型基本相同，只是路由键值中包含有通配符，可组成动态路由，具有灵活性。

3　Spring Boot 整合 RabbitMQ

3.1　RabbitMQ 环境配置

RabbitMQ 是第三方消息中间件，具有独立性，在项目中引入消息中间件首先要安装 RabbitMQ 消息中间件到服务

器，使其成为一个独立的服务程序，RabbitMQ 环境设计及相关参数配置为：

（1）安装 Erlang 语言包：RabbitMQ 消息中间件需要 Erlang 语言包的支持，所以在安装 RabbitMQ 前需要安装 Erlang 语言包，根据不同版本选择 32 位或 64 位 Erlnag 语言包进行安装。

（2）安装 RabbitMQ：下载并安装 RabbitMQ 消息中间件，配置 ERLANG_HOME 为 Erlang 语言包的安装路径。

（3）RabbitMQ 端口：RabbitMQ 消息中间件默认有两个端口号 5672 和 15672，其中 5672 作为服务端口，供应用程序访问，另外一个端口是 RabbitMQ 的可视化界面的端口，可以通过 http://localhost:15672 进入可视化界面查看消息中间件服务器各个元素的数据及变化。

（4）用户登录名及密码：通过两个端口访问消息中间件都要通过用户名与密码登录，系统默认的用户名与密码都是 guest，登录系统后可以配置用户及密码。

3.2 基于 Spring Boot 配置类模式整合应用

项目基于 Spring Boot 的配置类模式，就 RabbitMQ 的发布订阅工作模式来阐述 Spring Boot 中对消息中间件 RabbitMQ 整合与应用。其完成的任务就是系统 A 通过 RabbitMQ 发送两条消息，两个不同的任务在系统 B 通过消息中间件使用该消息。

3.2.1 创建 Spring Boot 项目并连接消息中间件

选择 Spring Intializr 创建项目并选择 Messaging → Spring for RabbitMQ 依赖包。在 application.properties 全局文件中配置已经安装好的 RabbitMQ 的连接属性。

```
spring.rabbitmq.host=127.0.0.1
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

3.2.2 创建配置类完成以下任务

（1）定义消息置换器：

```
@Bean[3]
Public MessageConverter msgcnv(){ Return new Jackson2sonMessageConverter(); }
```

（2）定义消息中间件交换器类型为 fanout：

```
@Bean
Public Exchang fanout_x(){
return ExchangeBuilder.fanoutExchange(“fanout_x”).build() ;}
```

（3）定义两个消息队列：

```
@Bean
public Queue fanout_q_em(){return new Queue(“fanout_q_em”);}
@bean
Public Queue fanout_q_ms(){return new Queue(“fanout_q_ms”);}
```

（4）将消息对列与交换器绑定：

```
@Bean
public Binding bindem(){
return BindingBuilder.bind(fanout_q_em()).to(fanout_x()).with("").noargs(); }
@Bean
public Binding bindsm(){
return BindingBuilder.bind(fanout_q_sm()).to(fanout_x()).with("").noargs();}
```

3.2.3 测试类实现 A 系统发送消息到消息中间件

在项目中创建测试类，构建交互信息，通过 rabbit Template 把消息发送到消息服务器：

```
public void sendMQ() {user.setId(1001); user.setName(" 张三 ");
user.setMessage("* 交易成功 "); user.setPhone("13*25897521");
rabbitTemplate.convertAndSend("fanout_x","",user); }
```

3.2.4 服务类实现 B 系统接从消息中间件队列接收信息

构建后台处理类，通过 Rabbit Listener 监听器，监听消息队列是否存在消息，如有就通过 message 对象取出消息进行处理：

```
Public class Rbmqreceivems {
@RabbitListener(queues = "fanout_q_sm")
public void consumersm(Message message){byte[] msbody=message.getBody();
String ms=new String(msbody);System.out.println(" 接收到的短信消息："+ms); }}
```

同样可以定义一个发送电子邮件的队列监听器，监听邮件消息中队列的变化，如有消息产生就触发电子邮件的发送。通过上面的操作，然后通过 RabbitMQ 的可视化界面查询消息的产生与消费的过程。

4 结 论

本文通过对消息中间件的原理分析，帮助学生深刻理解消息中间件在企业级开发中的应用场景。RabbitMQ 是一款非常流行的，基于 AMQP 协议的开源消息中间件，具有较好的数据一致性、稳定性和可靠性，成功地把 RabbitMQ 与 Spring Boot 技术整合在项目中，可非常方便地实现面向消息中间件的应用与开发，并应用于 JavaEE 企业级开发的教学过程中。

参考文献：

[1] 全国信息技术标准化技术委员会（SAC/TC 28）. 信息技术中间件消息中间件技术规范：GB/T 28168—2011 [S]. 北京：中国标准出版社，2011.

[2] 黑马程序员 .SpringBoot 企业级开教程 [M]. 北京：人民邮电出版社，2019.

[3] 汪云飞 .Java EE 开发的颠覆者 Spring Boot 实战 [M]. 北京：电子工业出版社，2010.

作者简介：唐权（1971—），男，汉族，四川遂宁人，副教授，硕士，研究方向：软件技术与理论、中间件、信息安全；周蓉（1969—），女，汉族，四川遂宁人，教授，硕士，研究方向：计算机应用技术、网络技术；张勇（1970—），男，汉族，四川遂宁人，教授，博士，研究方向：数据库技术、网络技术。