

JavaEE平台技术 消息服务器-RocketMQ

邱明 博士

厦门大学信息学院

mingqiu@xmu.edu.cn

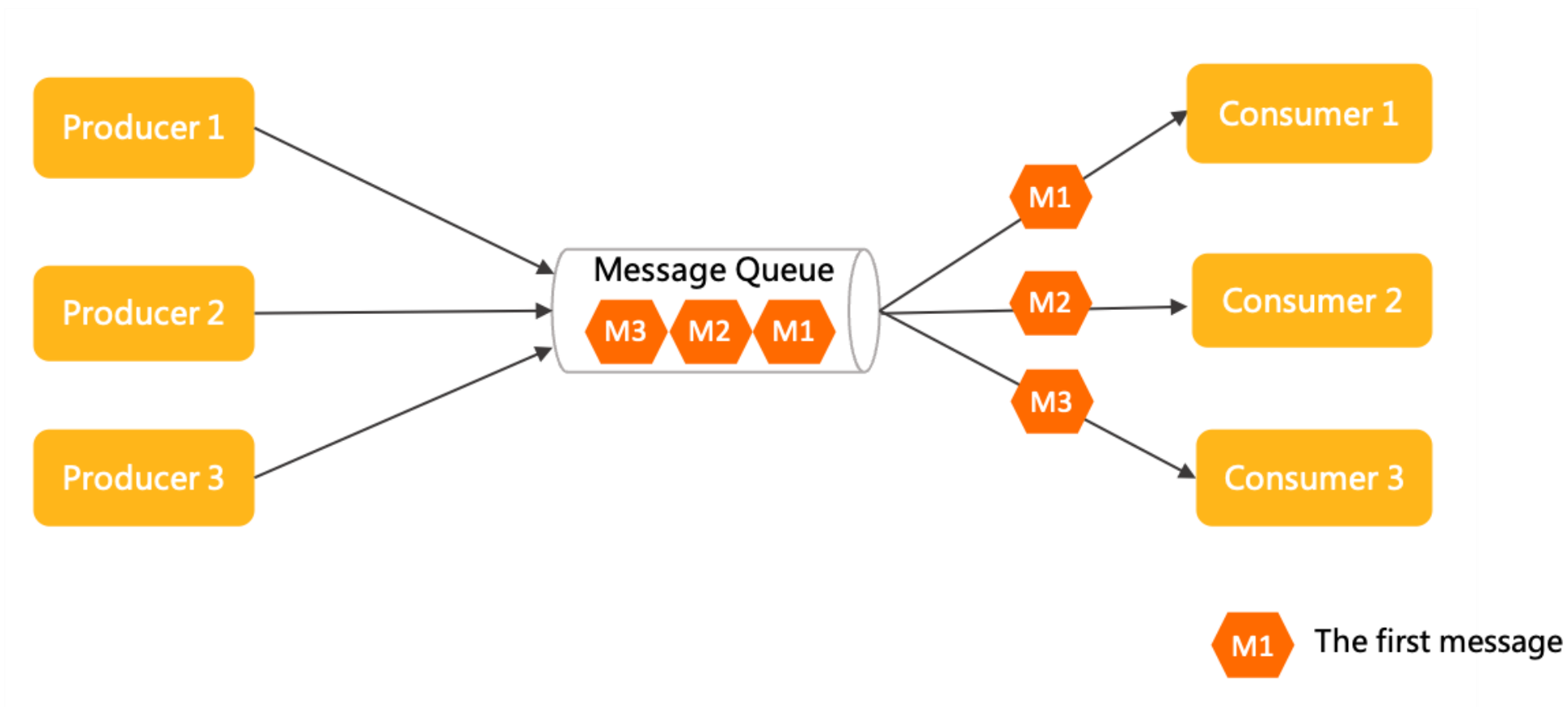
1 基本概念

- 消息服务是一个分布式异步通讯模型
- 基于发布/订阅的消息模型



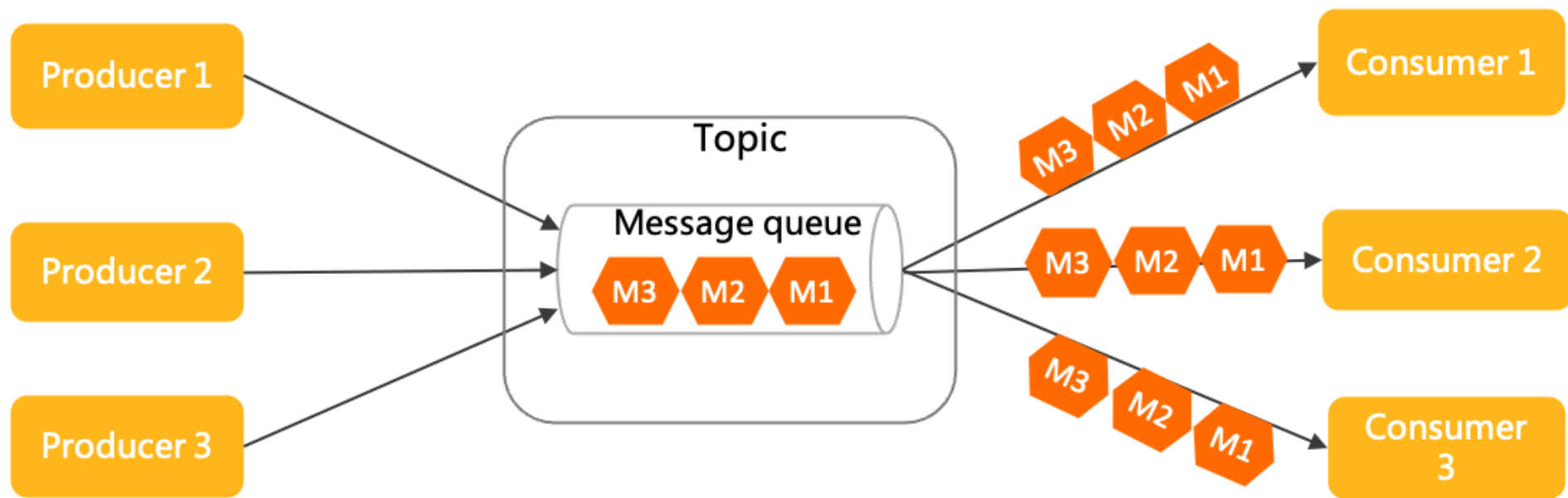
1 基本概念


- 点对点通讯模式



1 基本概念

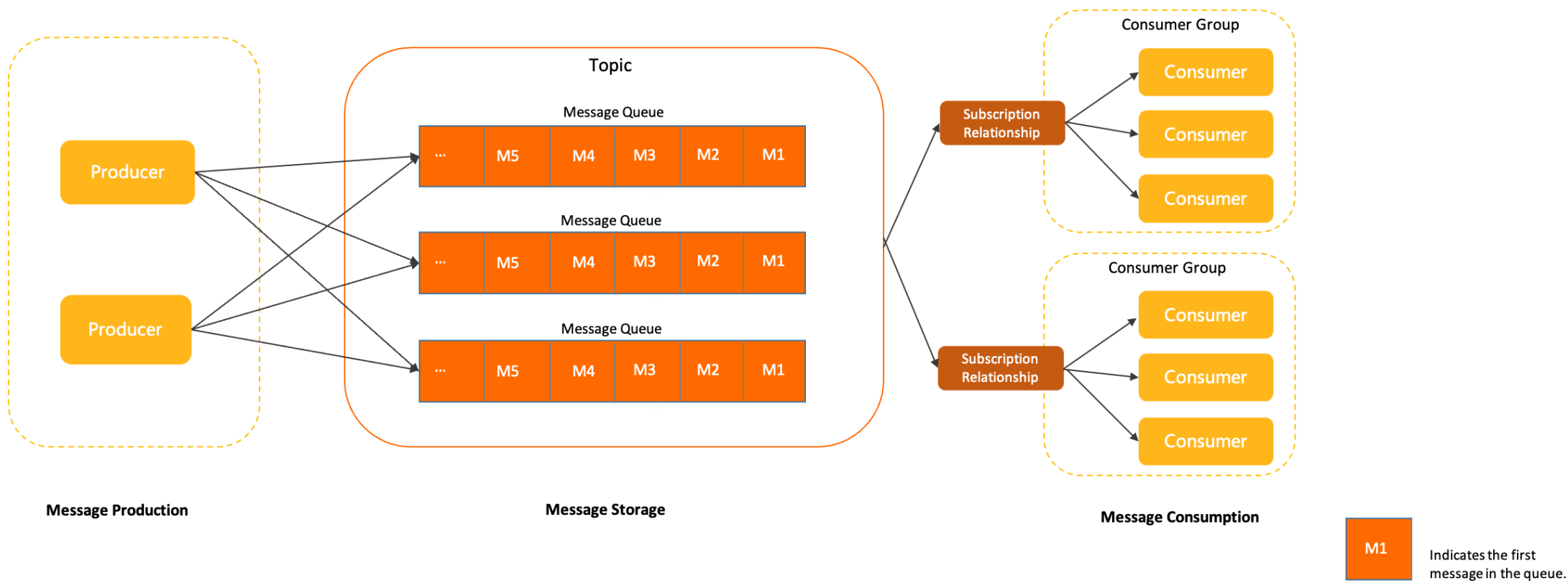
- 订阅和发布通讯模式



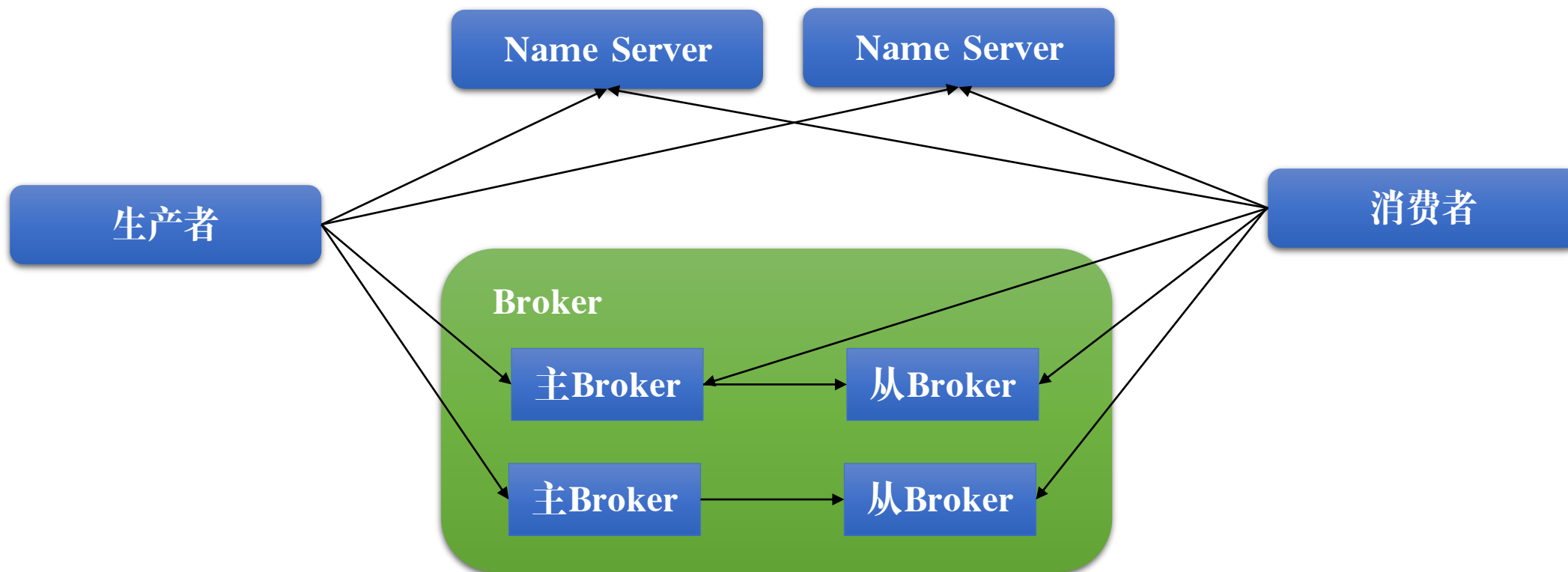
 The first message

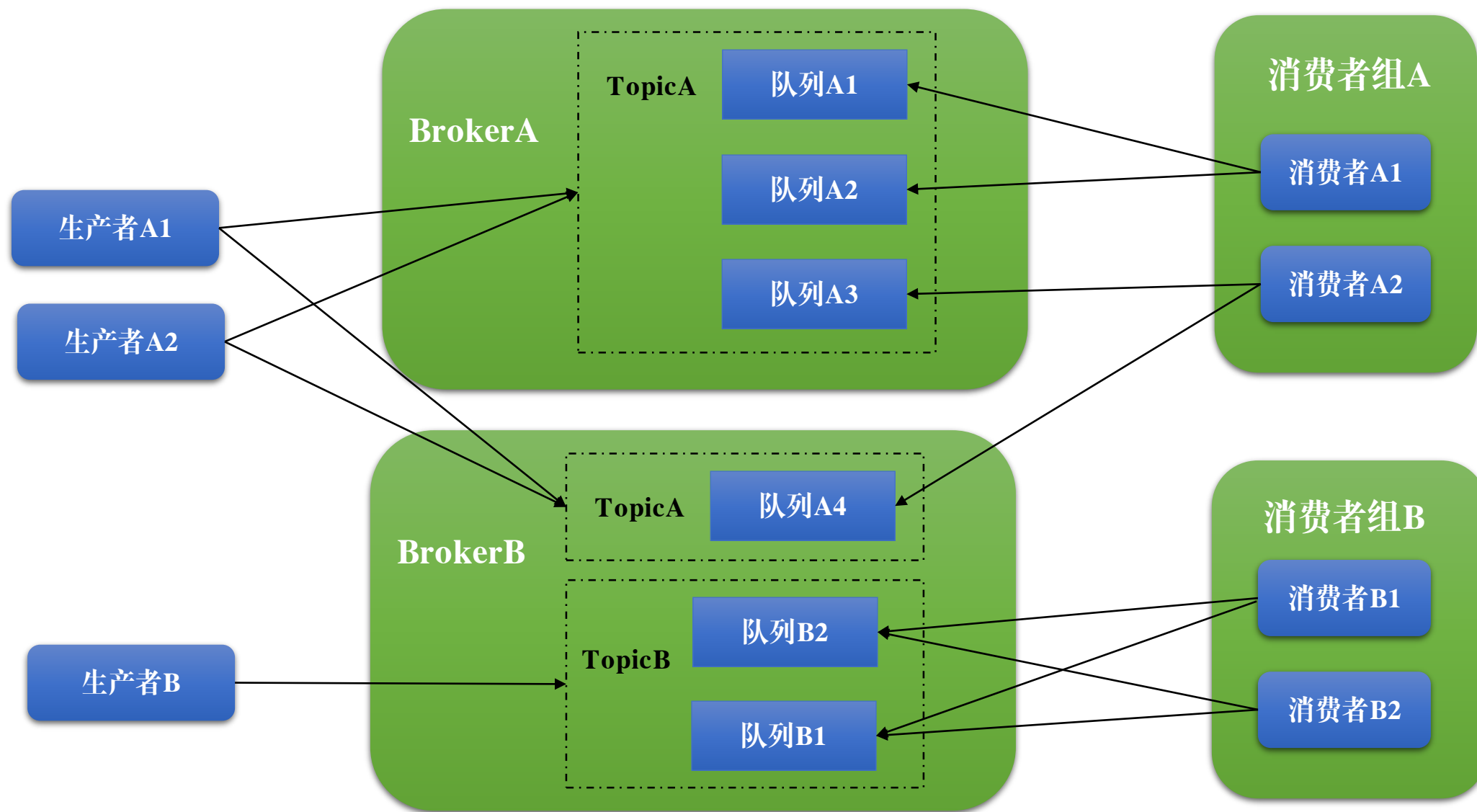
1 基本概念

- Topic-队列的集合



2 RocketMQ的结构





3. 消息的类型

- 普通消息



```
public void sendLogMessage(Log log){  
    String json = JacksonUtil.toJson(log);  
    Message message = MessageBuilder.withPayload(json).build();  
    logger.info("sendLogMessage: message = " + message);  
    rocketMQTemplate.sendOneWay("log-topic:1", message);  
}
```


3. 消息的类型

- 延时消息



固定的延时等级 "1s 5s 10s 30s 1m 2m 3m 4m 5m 6m 7m 8m 9m 10m 20m 30m 1h 2h"

3. 消息的类型

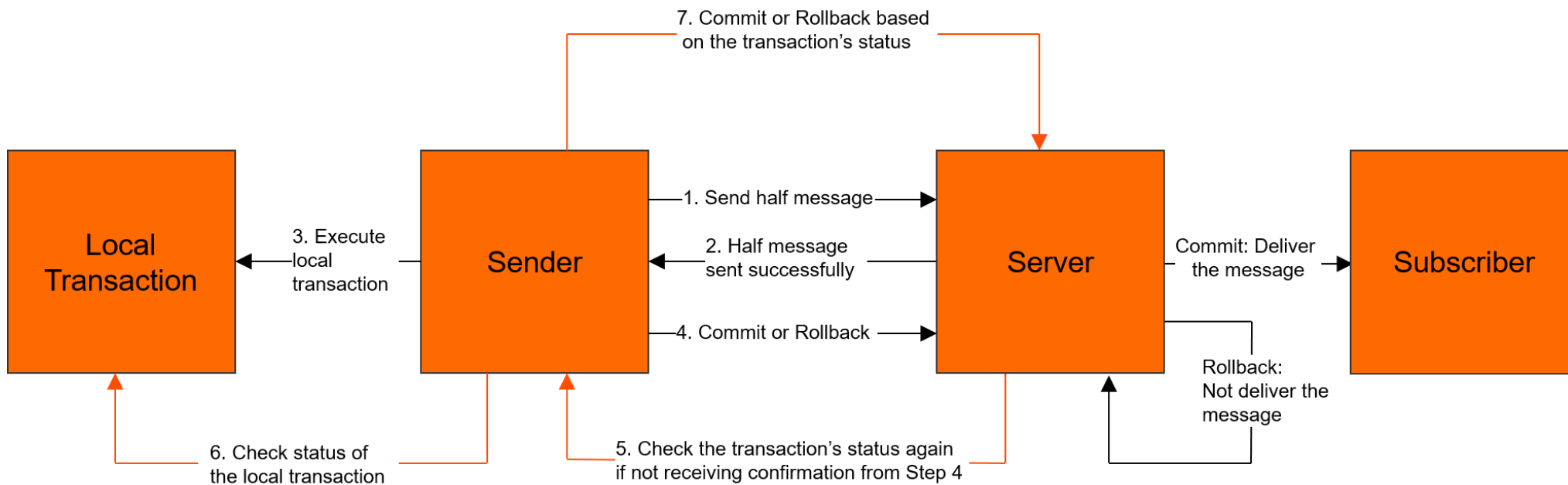
- 延时消息示例

```
public void sendOrderPayMessage(Long orderId){
    logger.info("sendOrderPayMessage: send message orderId = "+orderId+" delay ="+delayLevel+" time =" +LocalDateTime.now());
    rocketMQTemplate.asyncSend("order-pay-topic", MessageBuilder.withPayload(orderId.toString()).build(), new SendCallback() {
        @Override
        public void onSuccess(SendResult sendResult) {
            logger.info("sendOrderPayMessage: onSuccess result =" + sendResult+" time =" +LocalDateTime.now());
        }

        @Override
        public void onException(Throwable throwable) {
            logger.info("sendOrderPayMessage: onException e =" + throwable.getMessage()+" time =" +LocalDateTime.now());
        }
    }, timeout * 1000, delayLevel);
}
```

3. 消息的类型

- 事务消息



3. 消息的类型

- 事务消息



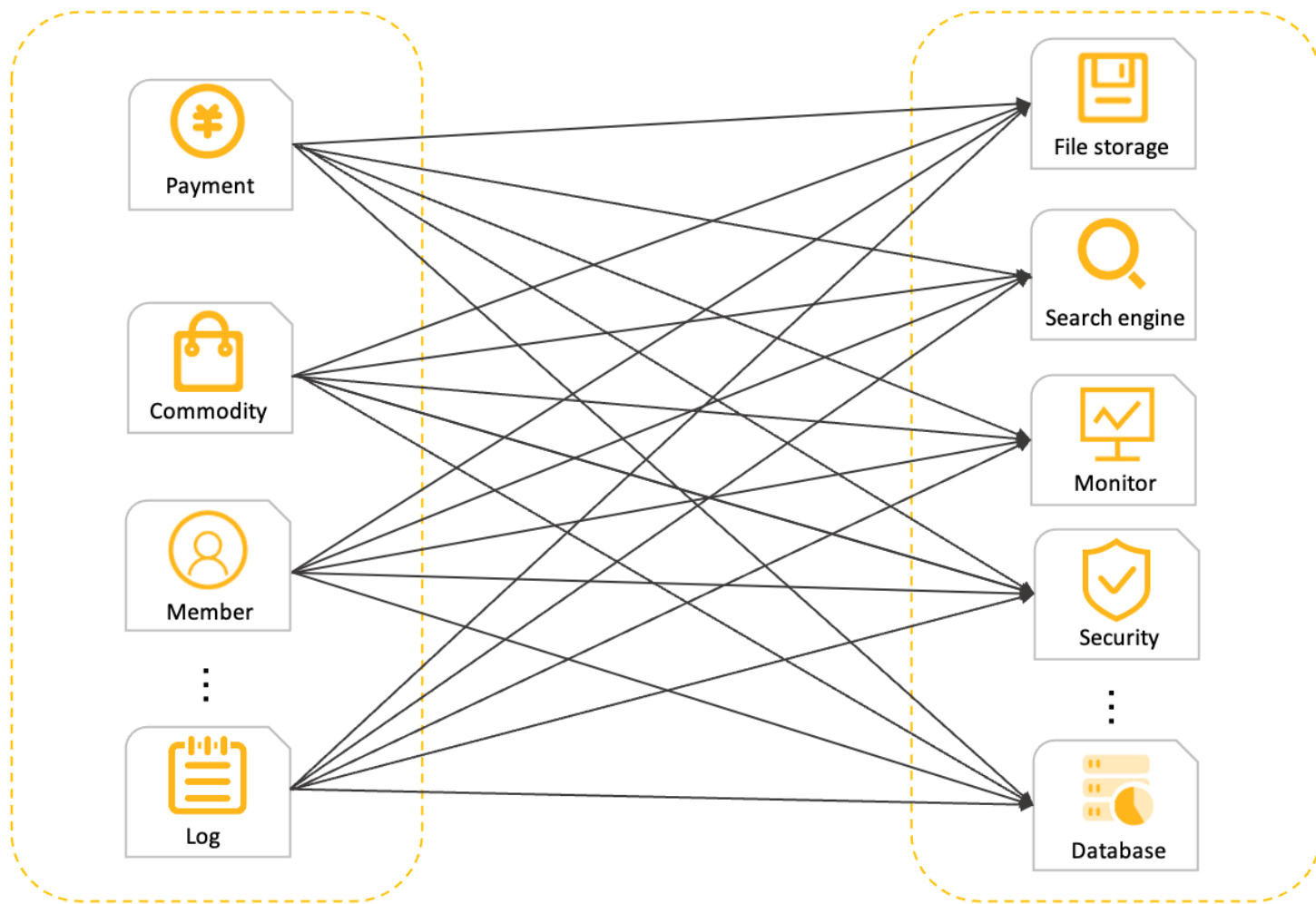
4. 消息的消费者

```
@Service
@RocketMQMessageListener(topic = "log-topic", selectorExpression = "1", consumeMode = ConsumeMode.CONCURRENTLY, consumeThreadMax = 10, consumerGroup = "log-group")
public class LogConsumerListener implements RocketMQListener<String>, RocketMQPushConsumerLifecycleListener {
    private static final Logger logger = LoggerFactory.getLogger(LogConsumerListener.class);
    @Override
    public void onMessage(String message) {
        Log log = JacksonUtil.toObj(message, Log.class);
        logger.info("onMessage: got message log =" + log);
    }

    @Override
    public void prepareStart(DefaultMQPushConsumer defaultMQPushConsumer) {
        // defaultMQPushConsumer.setConsumeFromWhere(ConsumeFromWhere.CONSUME_FROM_TIMESTAMP);
        // defaultMQPushConsumer.setConsumeTimestamp(UtilAll.timeMillisToHumanString3(System.currentTimeMillis()));
        logger.info("prepareStart: consumerGroup =" + defaultMQPushConsumer.getConsumerGroup());
    }
}
```

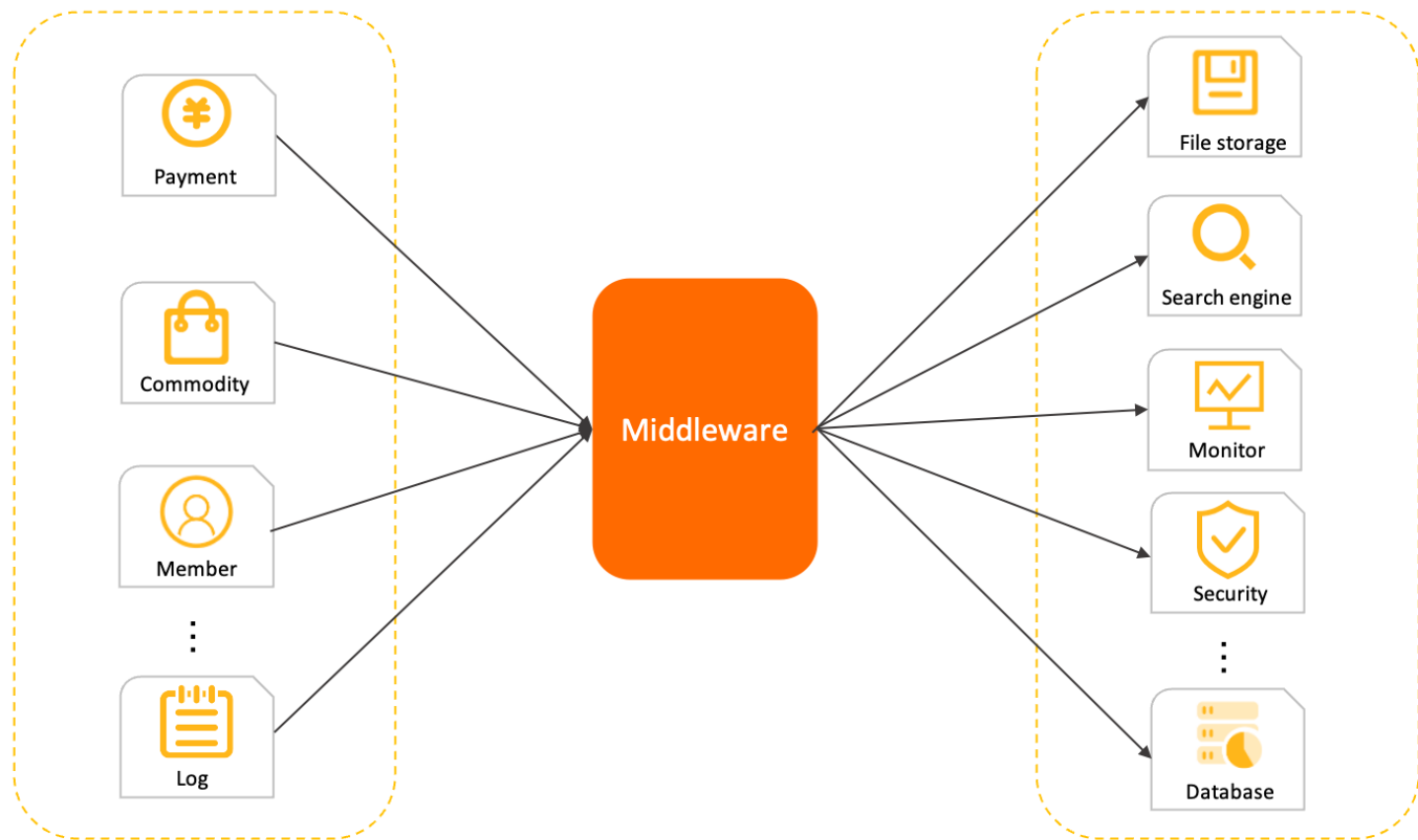
4 消息服务器的作用

- 系统解耦



4 消息服务器的作用

- 系统解耦



4 消息服务器的作用

- 系统解耦

