Agenda

- ✗ Problema
- ✗ Contexto
- ✗ JMS
- ✗ Implementação JMS
- ✗ Spring
- ✗ Implementação Spring

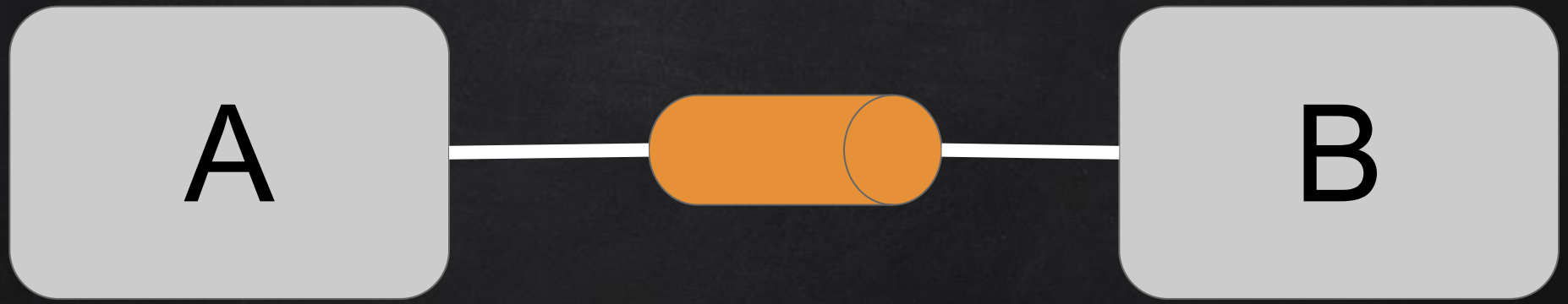
Paula Santana

- Desde 2008 em Tecnologia
- Des. Java 2014
- Arquitetura de Soluções
- Engenharia de Software
- Praia > SP
- Devs JavaGirl
- SouJava

Por quê?

- Assunto mal compreendido em TI
- Facilita a compreensão de abordagens como EDA
- Mostrar novas implementações através do Spring

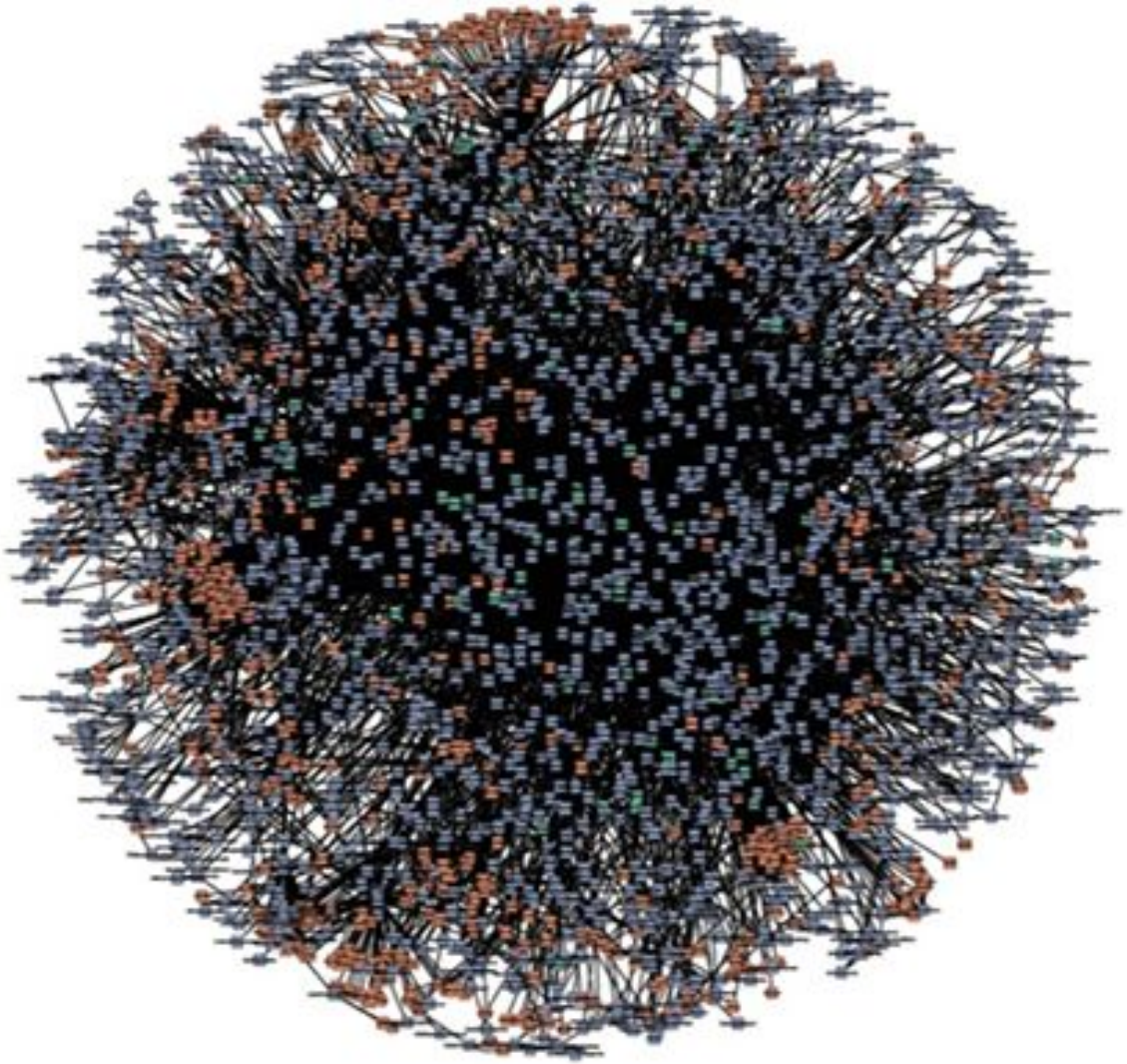
Comunicação



Resolve

Mensageria

Problema



amazon.com

NETFLIX

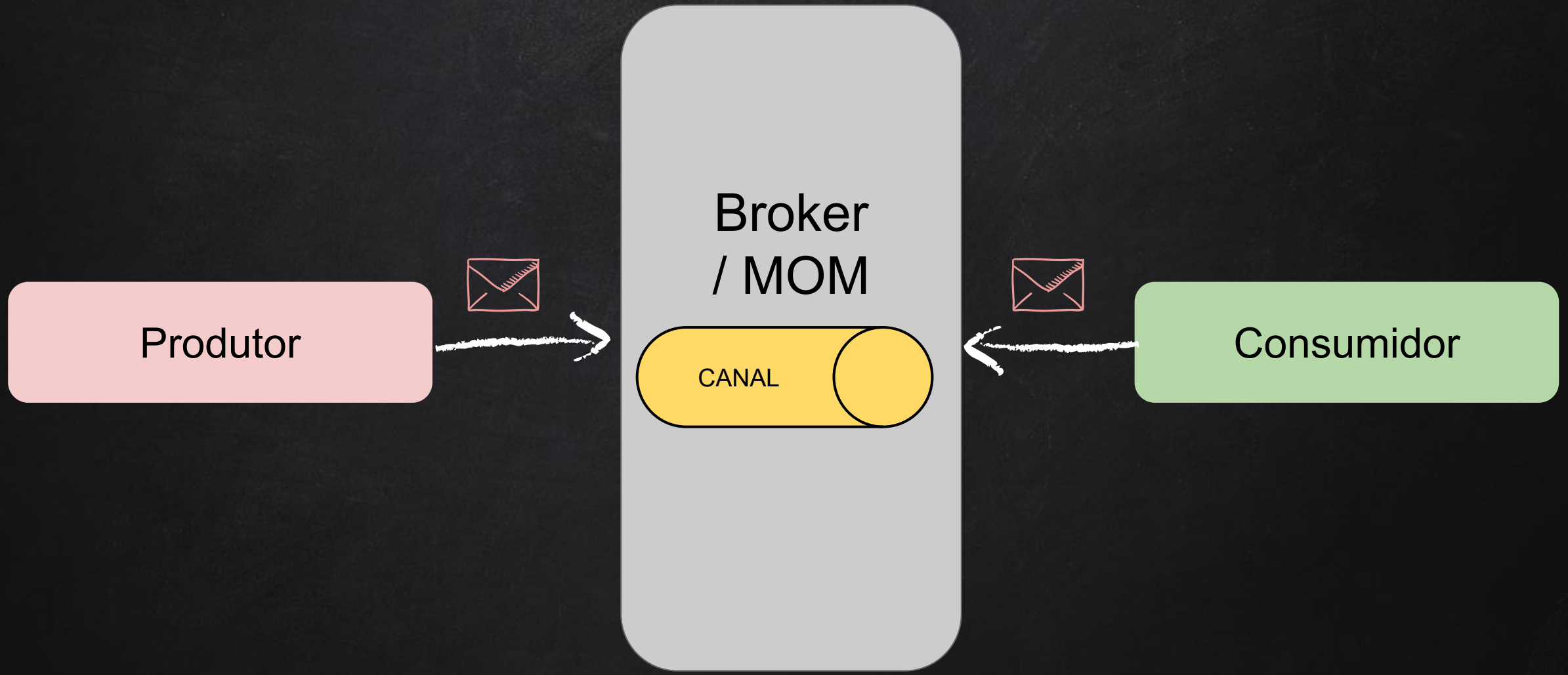
Fonte: https://www.appcentrica.com/wp-content/uploads/2016/11/Microservices-Architecture-1.png

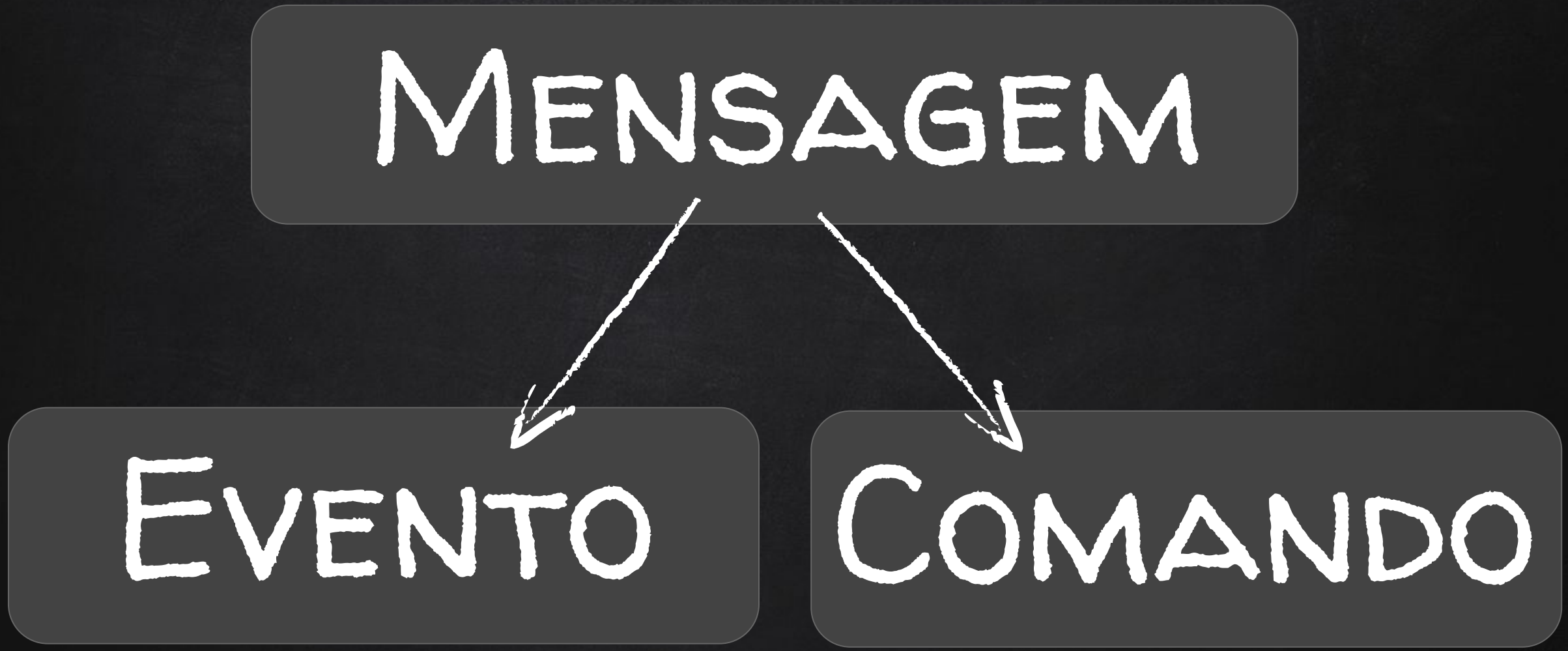

O QUEEEEEEEEEEE?

Mensageria

- ✗ Permite desacoplar aplicações
- ✗ Flexibilidade e agilidade
- ✗ Facilidade em acrescentar novos consumidores
- ✗ Operações Assíncronas
- ✗ Arquitetura com escalabilidade
- ✗ Sistemas distribuídos

ESTRUTURA




Mensagem
Evento
Comando

Enterprise Integration Patterns

Canais de mensagem

Tubos e filtros

Transformação de mensagem

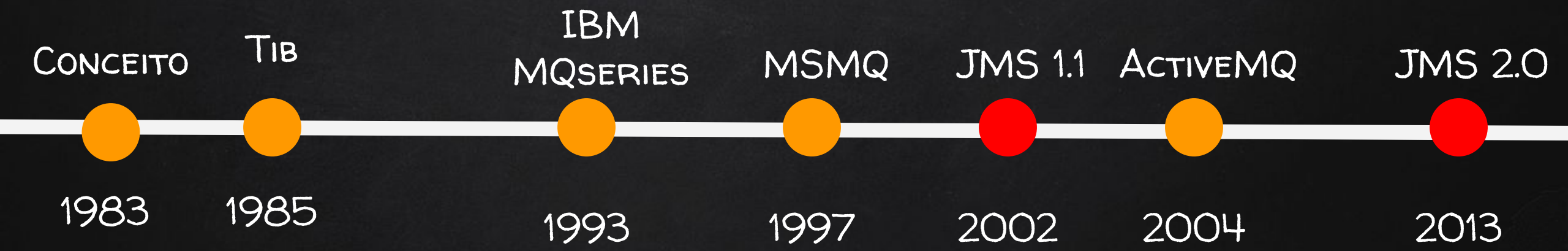
Construção de Mensagem

Roteamento de Mensagem

Endpoint de Mensagens

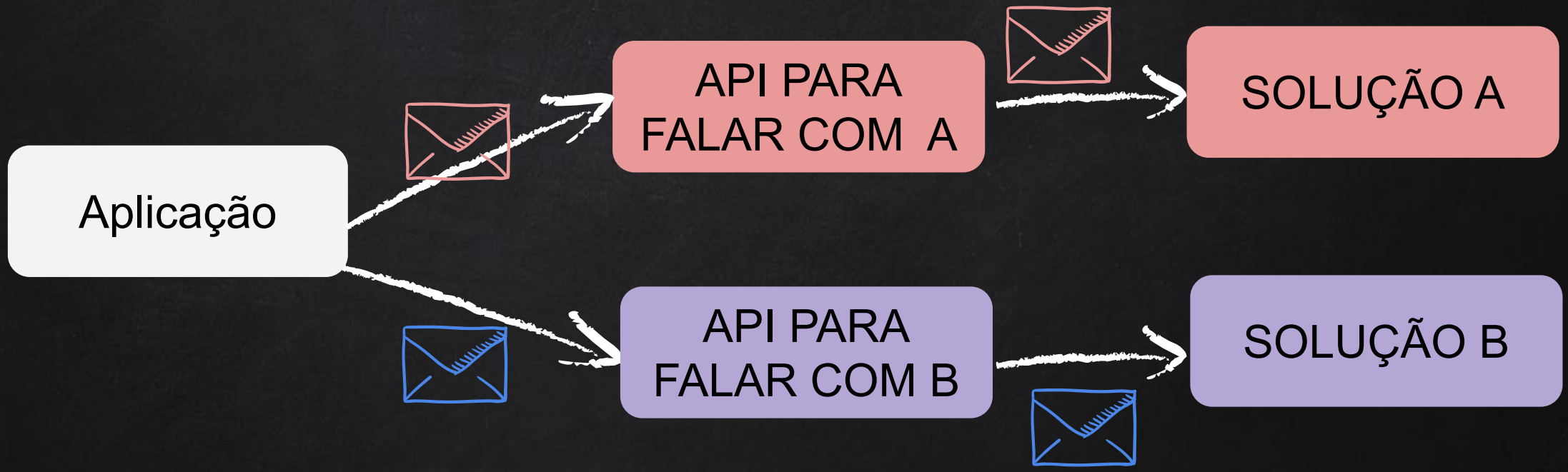
História

Java Message Service



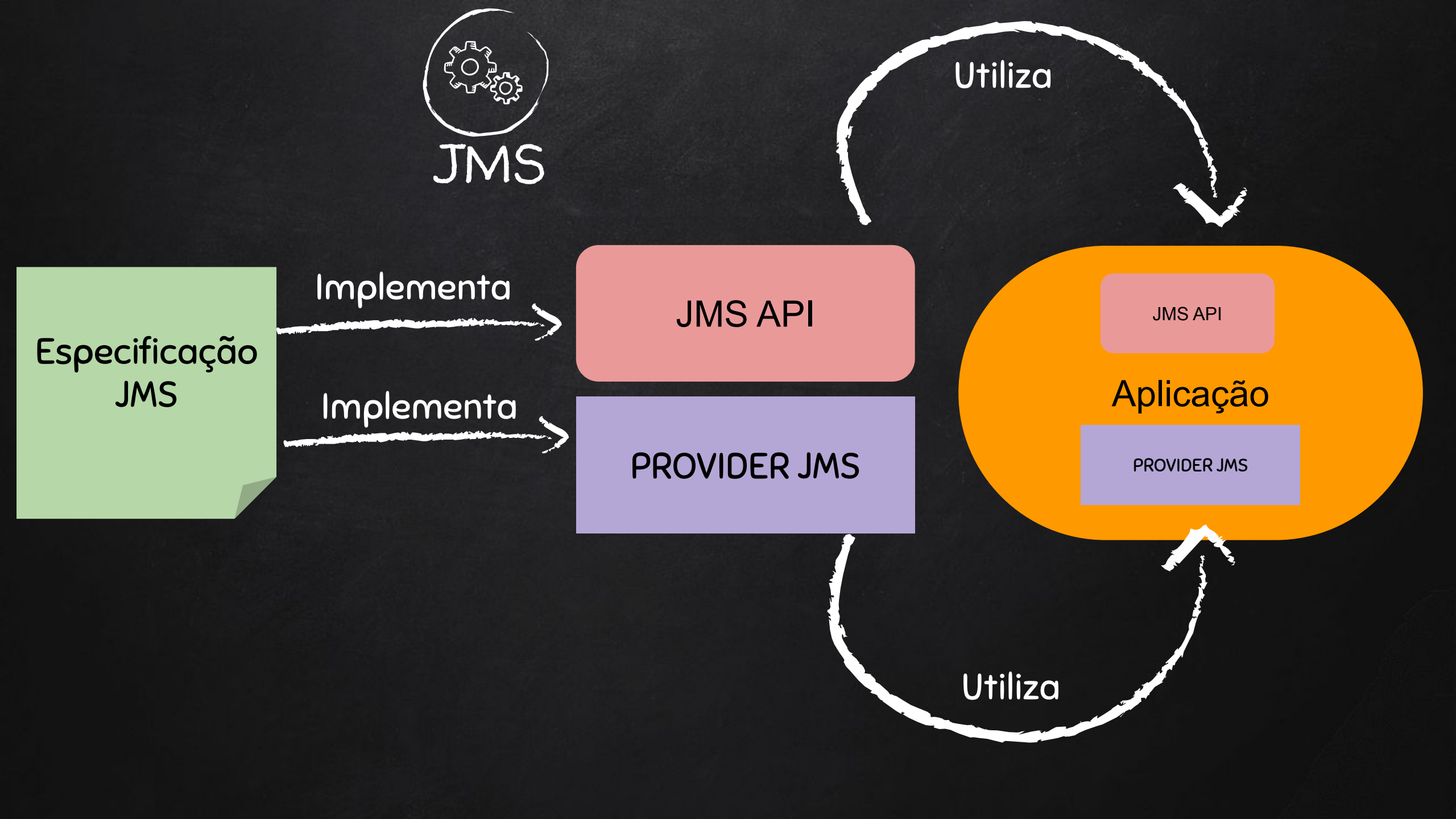
Eu...toda vez que muda o broker de mensagem

PROBLEMA QUE O JMS RESOLVEU



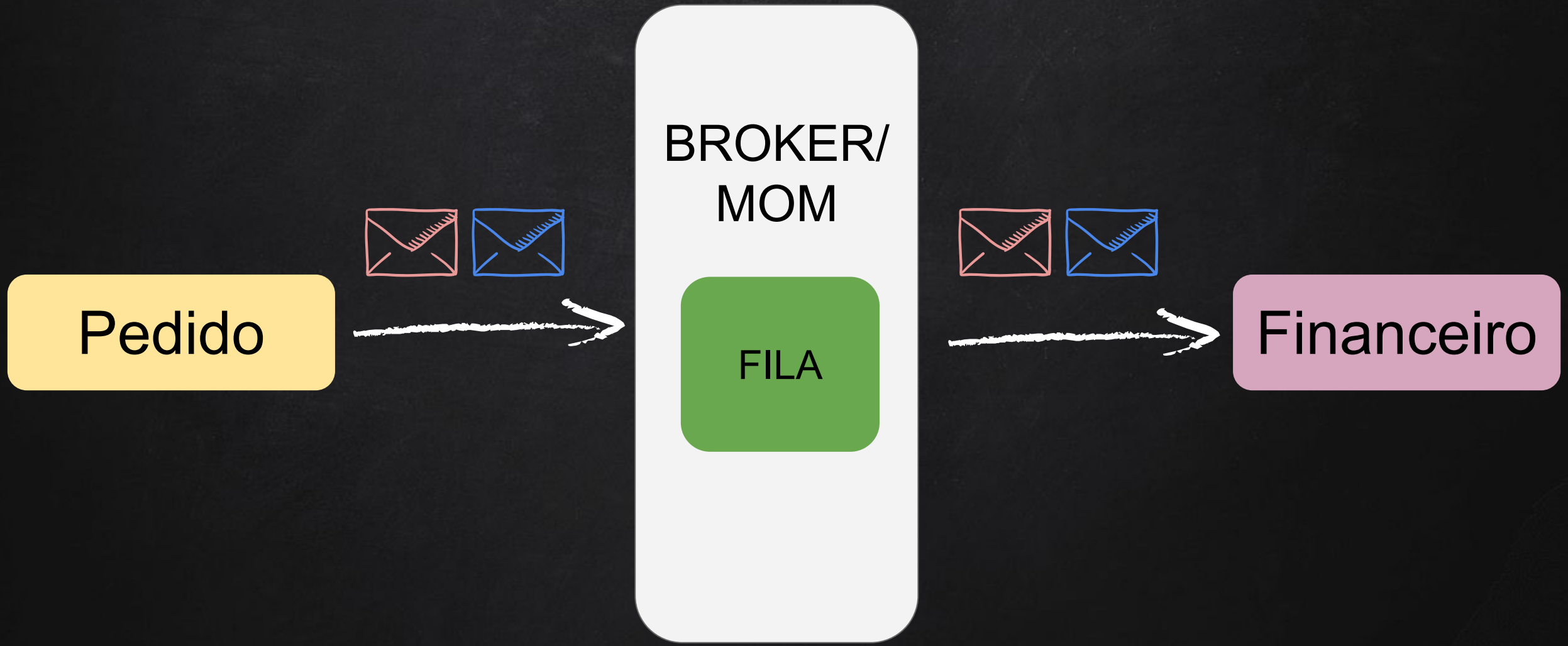
A especificação Java Message Service (JMS) foi criada justamente para definir um conjunto de funcionalidades comuns à maioria dos produtos de mensageria e uma API padronizada que permitisse a aplicações Java utilizarem os serviços de *middlewares* orientados a mensagens, compatíveis com a API, de uma maneira uniforme, para a criação, envio e recebimento de mensagens.

JMS é um protocolo?

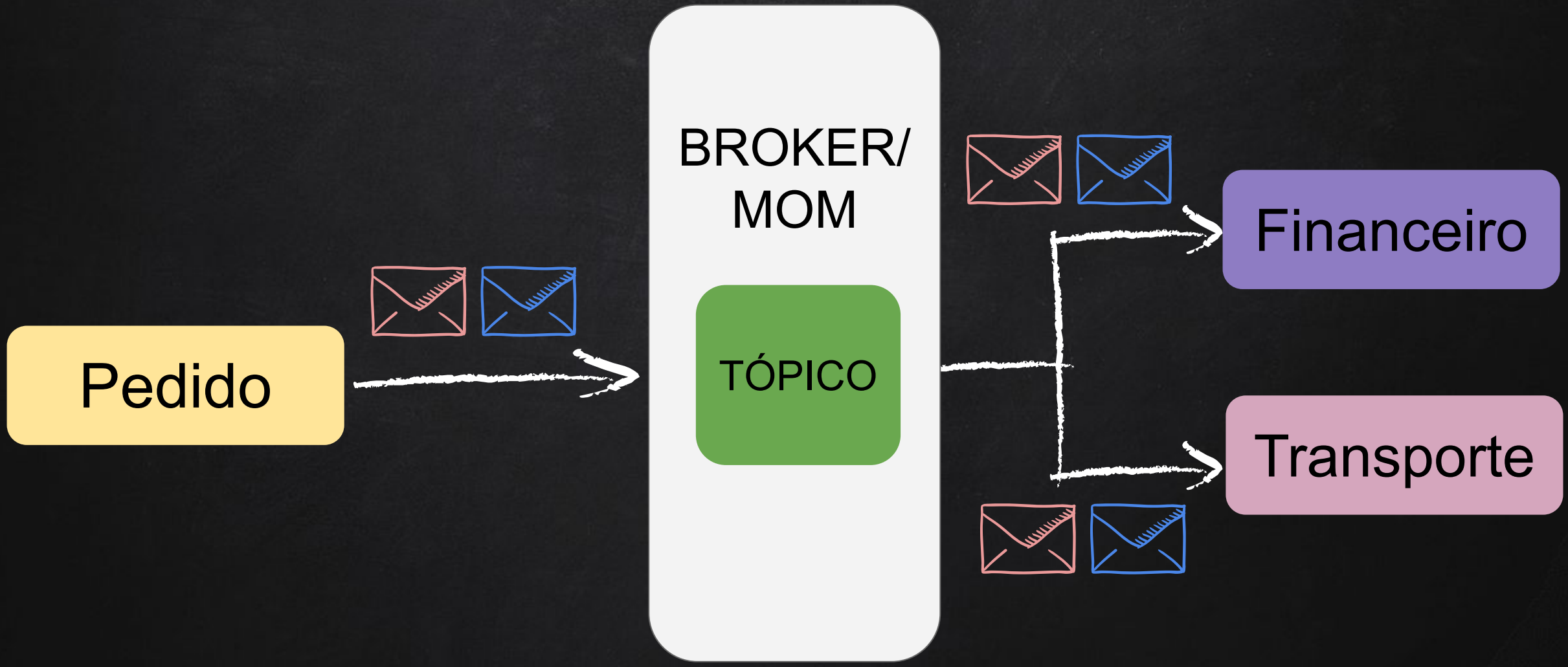

JMS
Utiliza
Especificação JMS
Implementa
Implementa
JMS API
PROVIDER JMS
JMS API
Aplicação
PROVIDER JMS
Utiliza

Modelos de Entrega

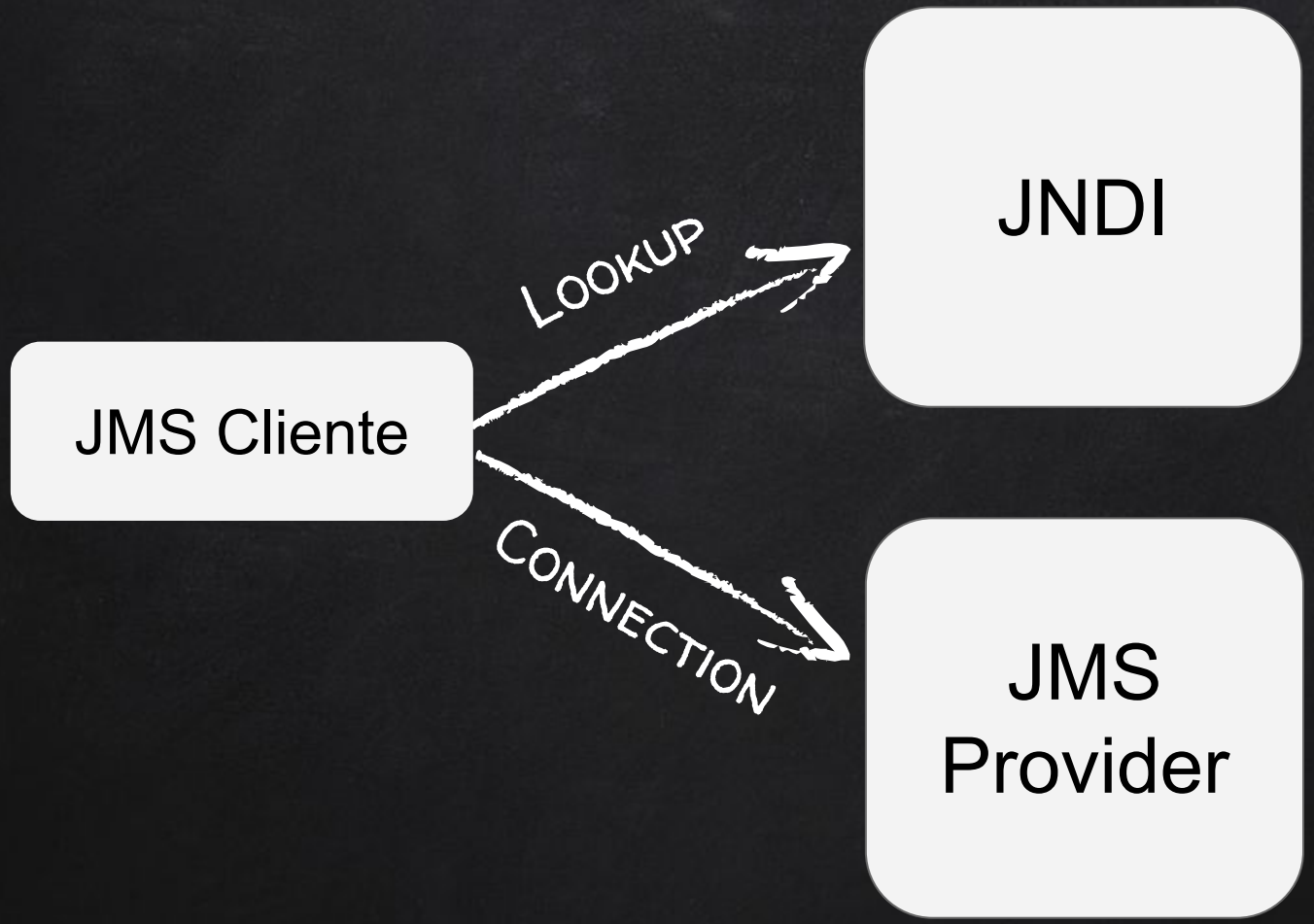
Modelos de Entrega – Fila (ponto a ponto)



Modelos de Entrega - Tópico (publish/subscribe)



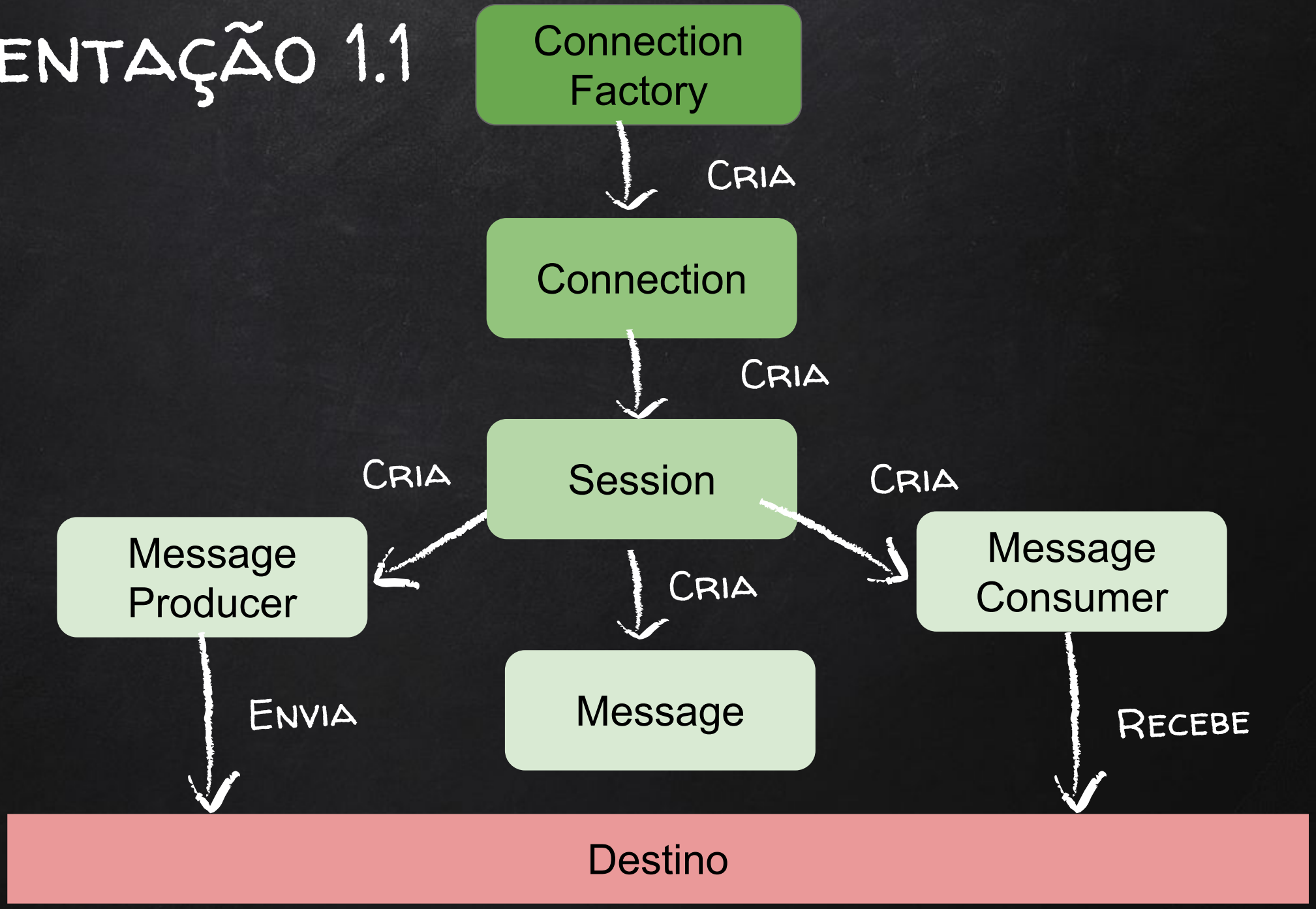
Arquitetura JMS



Os produtos compatíveis com a API JMS são chamados de provedores JMS (ou JMS Providers), como WebSphere MQ, SonicMQ, FioranoMQ , ActiveMQ, HornetQ, Oracle AQ, EMS , OpenJMS e RabbitMQ

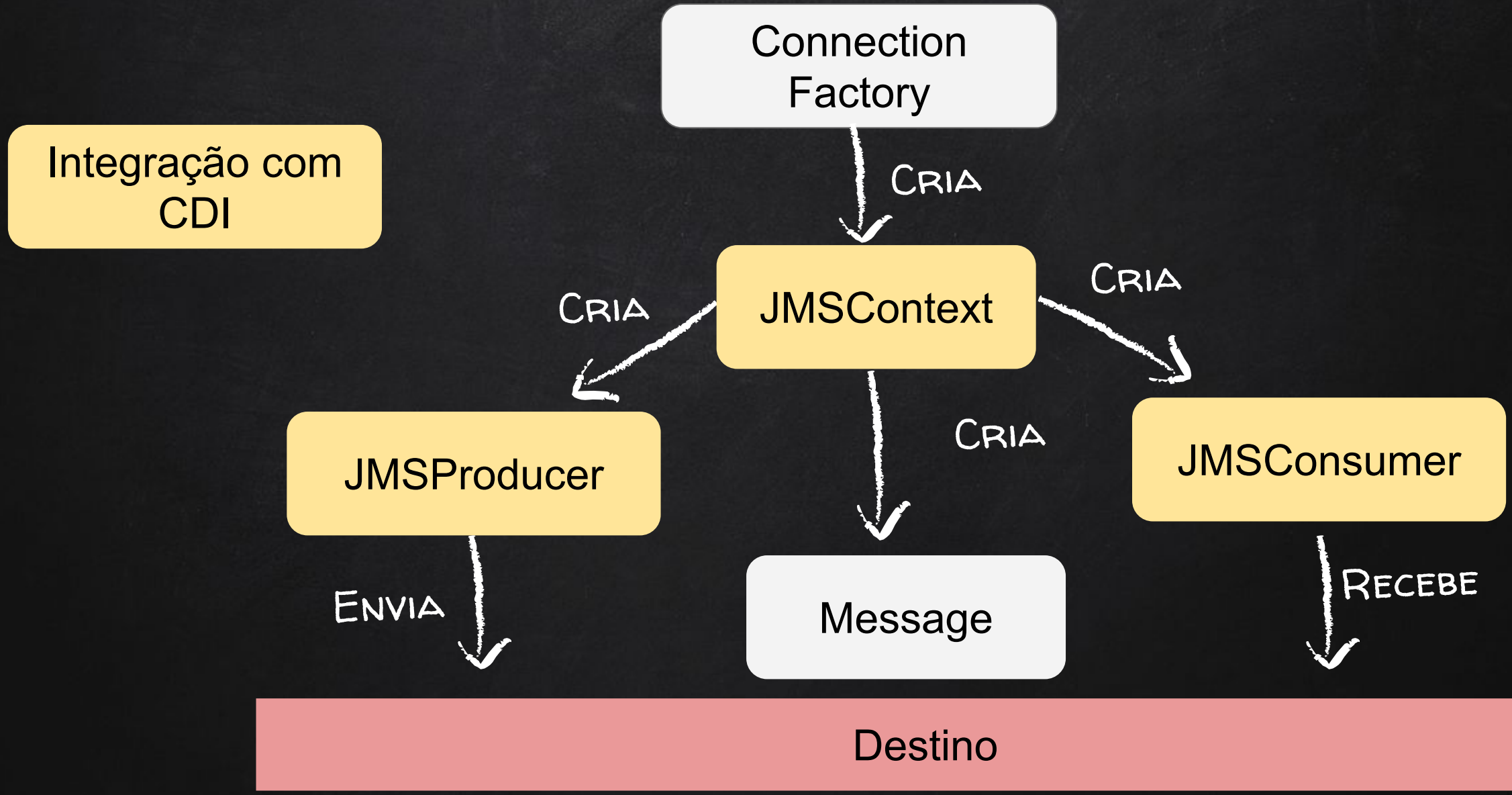
JNDI é uma API utilizada em aplicações que acessam recursos externos, ela permite obter esses recursos através do nome. Ela especifica a interface de serviço SPI e esse mecanismo permite que o suporte de vários serviços de diretório, como : LDAP, DNS, NIS, RMI, CORBA, entre outros.

Implementação 1.1



```
public class ShowMeCode {

    public static void main(String[] args) {
        System.out.println("Show me the code");
    }

}
```

Implementação 2.0



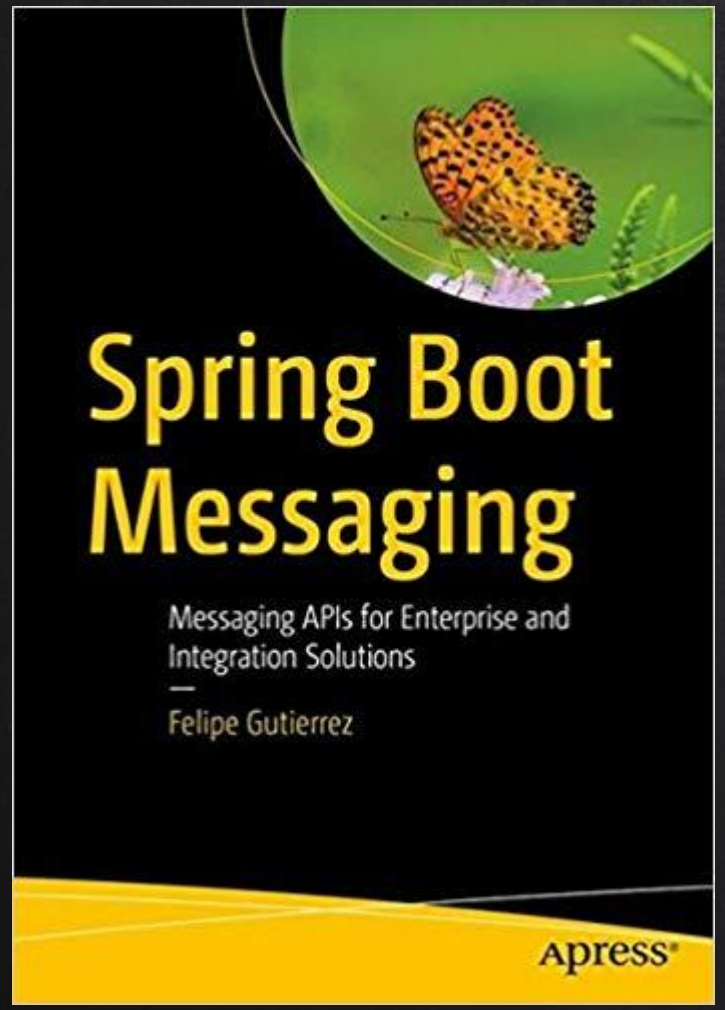
```java
public class ProdutorFila {

    @Resource(lookup = "jms/queue")
    private Queue queue;

    @Inject
    @JMSConnectionFactory("jms/connectionFactory")
    private JMSContext jmsContext;

    public void sendMessage(String message) {
        jmsContext.createProducer().send(queue, message);
    }

}
```


Spring Cloud Stream
Spring for ActiveMq
Spring for RabbitMq
Spring Integration
Spring for ActiveMq
Spring Integration

O Spring Framework permitiu uma maneira simples e fácil de enviar mensagens, implementando um padrão de design de modelo que pode ser usado com qualquer sistema de mensagens. Ele suporta a API JMS com o JmsTemplate, o AMQP com o RabbitTemplate, o STOMP e o sistema interno de mensagens com eventos e ouvintes.

Implementação Spring

Referência



Alura

ActiveMq

Conteúdo legal

Projetos

Nutella

Raiz

Obrigada!

Dúvidas?

@psanrosa13

paula-macedo-santana-dev

@paulasantana