RabbitMQ-

Serviço de mensagem ou broker que trafega mensagens de um serviço para o outro.

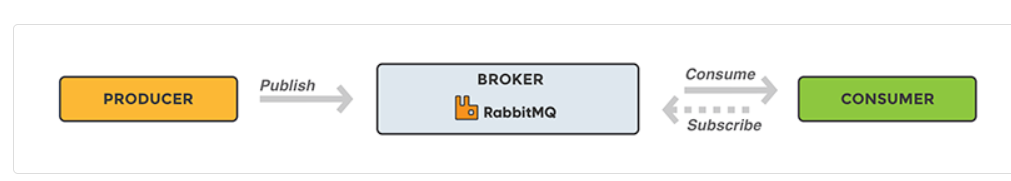
Pode ser qualquer coisa, uma mensagem para começar uma demanda ou somente um log.

Eles podem ser usados ​​para reduzir as cargas e os tempos de entrega dos servidores de aplicativos da web, delegando tarefas que normalmente levariam muito tempo ou recursos a terceiros que não têm outro trabalho.

A arquitetura simplificada de um evento de mensageria:

Temos um serviço qualquer que envia uma mensagem a um broker (no caso rabbitMQ), ao serviço que envia a mensagem chamamos de PRODUCER e ao fato de enviar uma mensagem ao broker chamamos de PUBLICAR (publish).

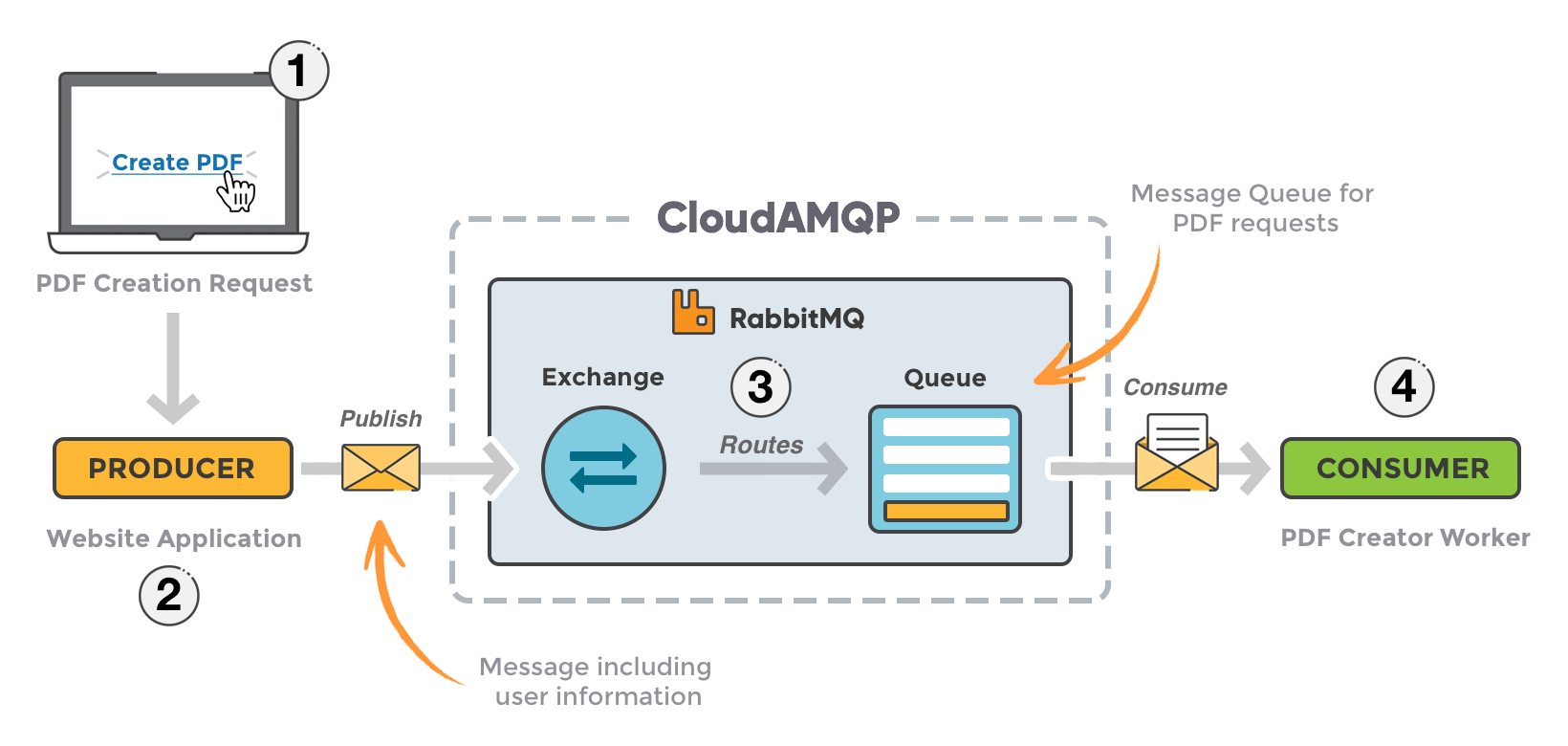
Broker é onde as mensagens são enviadas ela fica lá até alguém receber ela. O serviço que fala para o broker que ele quer receber aquela mensagem chamamos de COSUMIDOR (consumer) e o fato dele avisar ao broker que ele quer receber aquela mensagem ele se INSCREVE (subscribe) no broken para aquele tipo de mensagem.



Quando e por que usar:

Mensageria permite respostas rápidas ao invés de ficar esperando (síncrono) algum serviço executar.

Mensageria também permite que você escale com facilidade os consumidores, distribuindo-os entre consumidores (workers).



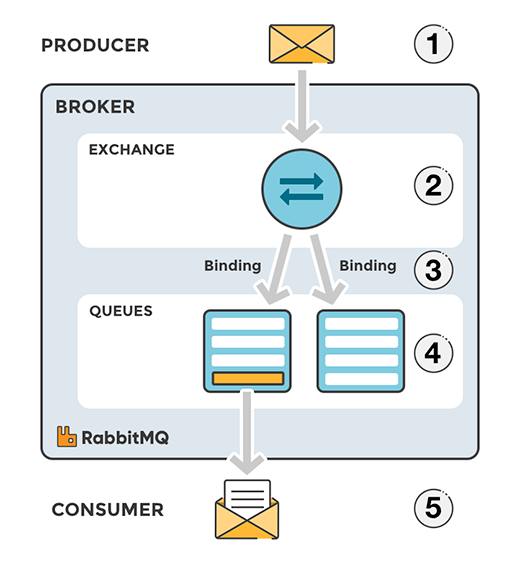
O consumidor retira uma mensagem da fila e começa a processar o PDF. Ao mesmo tempo, o produtor está enfileirando novas mensagens. O consumidor pode estar em um servidor totalmente diferente do produtor ou em um mesmo servidor. A solicitação pode ser criada em uma linguagem de programação e tratada em outra linguagem de programação. O ponto é que os dois aplicativos só se comunicam através das mensagens que estão enviando um ao outro, o que significa que o remetente e o destinatário têm pouco acoplamento.

**EXCHANGES (TROCAS)**

As mensagens não são publicadas diretamente na fila (queue), em vez disso o produtor envia a mensagem para uma Exchange.

Uma Exchange é responsável por guiar (rotear) as mensagens para diferentes filas com a ajuda dos bindings (ligações) e routing Keys (chaves de roteamento).

Um BINDING é um link de ligação entre a fila e o exchange



1 – O produtor publica uma mensagem na exchange. Quando cria uma Exchange precisamos especificar o tipo, existem 4, mas vamos falar disso mais tarde.

2 – A Exchange recebe a mensagem e agora é responsável por rotear ela (entregar para a fila). A exchange leva em consideração vários atributos da mensagem, como a routing key dependendo do tipo de exchange.

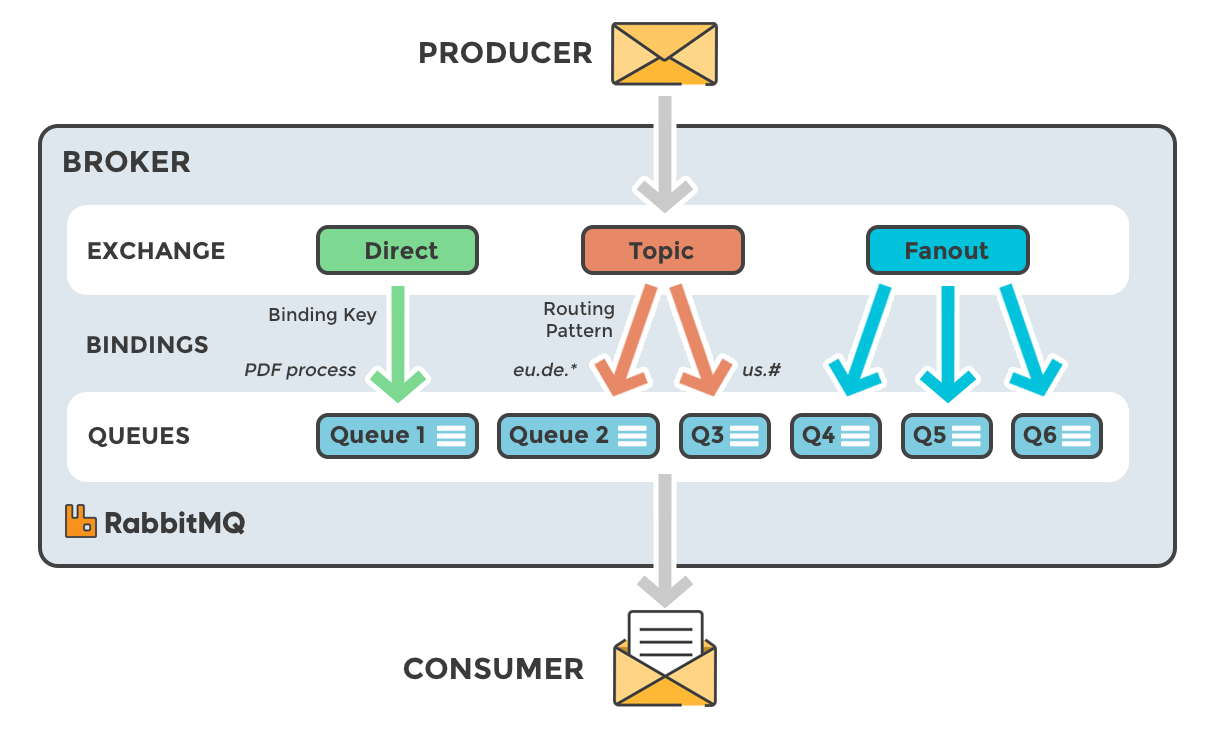
3 – Bindings (ligações) precisam ser criados da exchange para as queues (filas). A exchange encaminha a mensagem para a fila dependendo dos atributos da mensagem.

4 – A mensagem fica na fila até algum consumer (consumidor) tratar com ela.

5 – O consumidor recebe a mensagem e faz o que quer com ela.

Tipos de Exchanges

Explicação rápida



Direct – A mensagem é encaminhada para a fila que tem a binding key exatamente igual a routing key da mensagem.

Ex – Se a fila estiver vinculada à exchange com a binding key (enviarEmail) a mensagem que chegar com a routing key (enviarEmail) vai ser encaminhada para essa fila.

Fanout – Todas as filas vinculadas ao exchange recebem a mensagem.

Topic – Ele encaminha a mensagem para a fila a partir de um “curinga”, uma expressão colocada na binding key

Ex com \* : binding key está como “america.brazil.\*”, todas as mensagens que vierem com com a primeira palavra américa e a segunda brazil “america.brazil.” vao ser encaminhadas para essas filas, como “america.brazil.minas” ou “america.brazil.saopaulo”.

Ex com #: é usada para comparar zero ou mais palavras por exemplo como “america.brazil.#”, vai para qualquer fila uqe tenha américa ou brazil (REVER).

Headers – Usa os atributos do cabeçalho (header) para fazer o roteamento.

Sopa de letrinhas RabbitMQ

Produtor – aplicação que envia a mensagem

Consumidor – aplicação que recebe a mensagem

Queue – buffer (área de armazenamento temporária) que guarda as mensagens

Message: informação que é enviada do produtor para o consumidor através do rabbitmq.

Connection: a conexão TCP entre sua aplicação e o broker RabbitMQ

Channel: uma conexão virtual dentro de uma conexão. Quando é publicado e consumido uma mensagem de uma fila, tudo é feito pelo channel.

Exchange: recebe a mensagem do produtor e encaminha para uma fila.

Binding: é uma ligação entre uma fila e uma exchange.

Routing key: é uma chave que a exchange olha dentro da mensagem para encaminhar para uma fila. Pense como se fosse o endereço para onde a mensagem vai.

AMQP: Advanced Message Queuing Protocol é o protocolo usado pelo RabbitMQ para mensageria.

USERS: É o usuário para se conectar (logar) no rabbitMQ. Da para colocar permissões específicas para os vhos específicos.

**Vhost, virtual host:** Fornece uma maneira de segregar aplicativos usando a mesma instância do RabbitMQ. Usuários diferentes podem ter permissões diferentes para diferentes vhost e filas e trocas podem ser criadas, portanto, elas existem apenas em um vhost.

Subir container

docker run -d --hostname rabbit-local --name rabbit-local -p 5672:5672 -p 15672:15672 -e RABBITMQ\_DEFAULT\_USER=admin -e RABBITMQ\_DEFAULT\_PASS=Tadmin rabbitmq:3-management

Ref

<https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html>