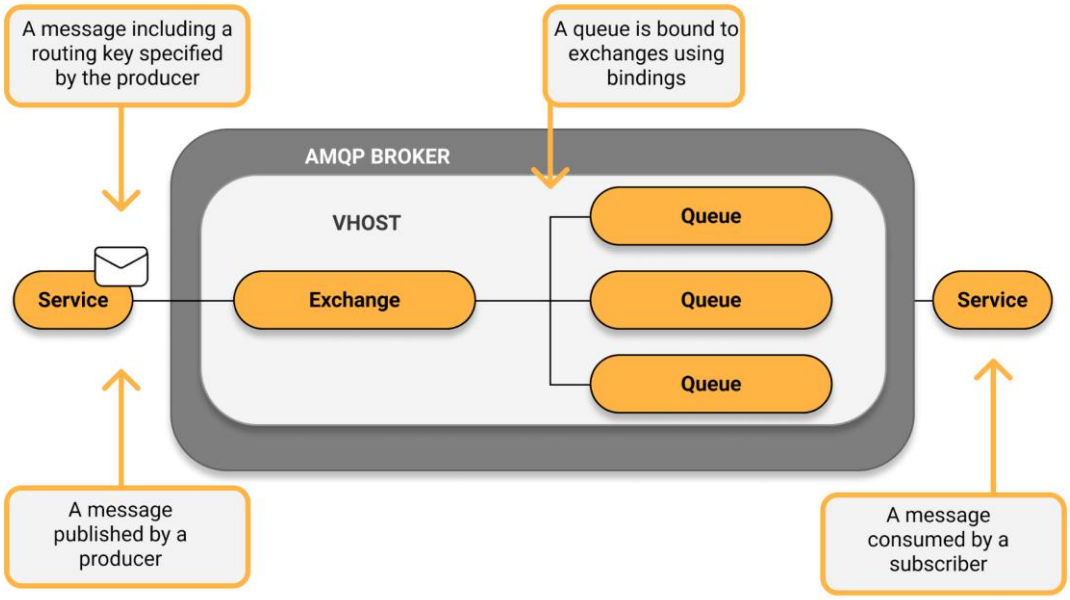
# Introducción a AMQP

AMQP es un protocolo de mensajería avanzado que permite la comunicación entre servicios y aplicaciones. Lo que vas a ver en este documento son los siguientes temas:

- Qué es el protocolo AMQP.
- La historia asociada al desarrollo del protocolo.
- Las entidades principales que juegan un rol clave para intercambiar información.
- Qué son los exchanges, y las descripciones generales de cada tipo.
- Las colas y los mensajes que consumen las aplicaciones.
- Las vinculaciones entre los exchanges y las colas, para poder transmitir un mensaje de origen a destino.
- Las versiones del protocolo y su evolución a lo largo del tiempo.
- Las implementaciones de AMQP para diferentes entornos y lenguajes de programación.

## ¿Qué es AMQP?

AMQP (Advanced Message Queuing Protocol) es un protocolo abierto que funciona a nivel de capa de aplicación. Sus características definen la creación de mensajes, encolamiento, enrutamiento de los mensajes producidos y exactitud para entregar los mensajes a los consumidores.
Se compone de un broker de mensajería que internamente posee exchanges, donde se conectan los productores de mensajes, y las colas, que se vinculan a los exchanges a través de diferentes criterios. Los consumidores de los datos se conectan a las colas para extraer los mensajes que producen los publicadores. En esta figura podemos ver un diagrama de la arquitectura usando este protocolo.




El protocolo establece el comportamiento tanto del servidor de mensajería como de los clientes que se conectan al broker, de manera tal que las implementaciones de diferentes proveedores son interoperables. Con AMQP se pueden implementar aplicaciones multiplataforma utilizando agentes, bibliotecas y frameworks heterogéneos, todos independientes del proveedor. Esto se debe a que el formato del mensaje AMQP proporciona la unidad de trabajo necesaria para intercambiar información.
Es un protocolo orientado a crear un "cable" entre las aplicaciones conectadas. Incluye funcionalidad para entregar mensajes de manera fiable, representar los datos a través de diferentes formatos, flexibilidad para definir los datos, preparado para escalabilidad, y capacidad de definir varias topologías en un mismo sistema.

> Estas características la vuelven un buen protocolo para IoT, ya que se puede implementar en dispositivos edge, cloud, en infraestructuras on-premise y que puede ser ejecutado a través de contenedores de software.

# Historia

El protocolo AMQP fue definido entre 2004-2006 por el banco JP Morgan Chase e iMatix Corporation, quienes también desarrollaron implementaciones en C/C++ y Java. Luego de crear la primera versión, la documentaron y la entregaron a un grupo de trabajo que incluía a varias empresas como Red Hat, Cisco e iMatix. Con el paso del tiempo se sumaron distintas entidades importantes que le dieron mayor seriedad aún al protocolo. En esta imagen podemos ver los logos de algunas empresas.

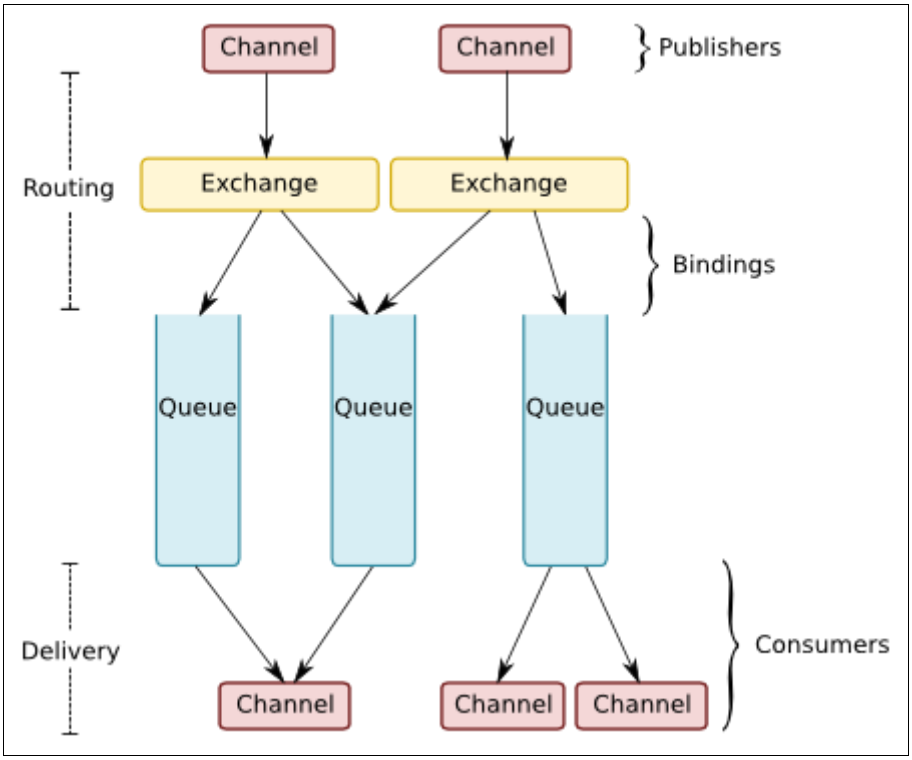

Un objetivo importante del diseño de AMQP era conseguir la creación de un protocolo que permita interoperar mediante mensajes, tanto dentro de la misma organización como entre organizaciones, mediante la creación de un mensaje con estándar abierto para realizar transacciones de negocio.
Si bien AMQP tiene sus orígenes en la industria financiera, es aplicable a una amplia gama de problemas de middleware, y tiene gran relevancia a la hora de diseñar aplicaciones multiplataforma y multiservicio.

# Entidades AMQP

Para realizar la comunicación AMQP define las siguientes entidades:

- **Broker:** un servidor de mensajería al que los clientes se conectan usando el protocolo AMQP y que se encarga de distribuir los mensajes.
- **Usuario:** un usuario es una entidad que puede conectarse a un broker.
- **Conexión:** una conexión física sobre algún protocolo de transporte como TCP/IP o SCTP que está ligada a un usuario.
- **Canal:** una conexión lógica que está ligada a una conexión.

Las entidades utilizadas para la transferencia de mensajes entre aplicaciones son declaradas dentro de un canal, que garantiza la creación lógica de los elementos necesarios para la comunicación, como la creación de un exchange, una cola a la cual enviar mensajes y la vinculación entre las entidades. En esta figura podés ver un diagrama de comunicación de las entidades.




Veamos en detalle las entidades que definen el protocolo para la creación y consumo de mensajes mediante AMQP.

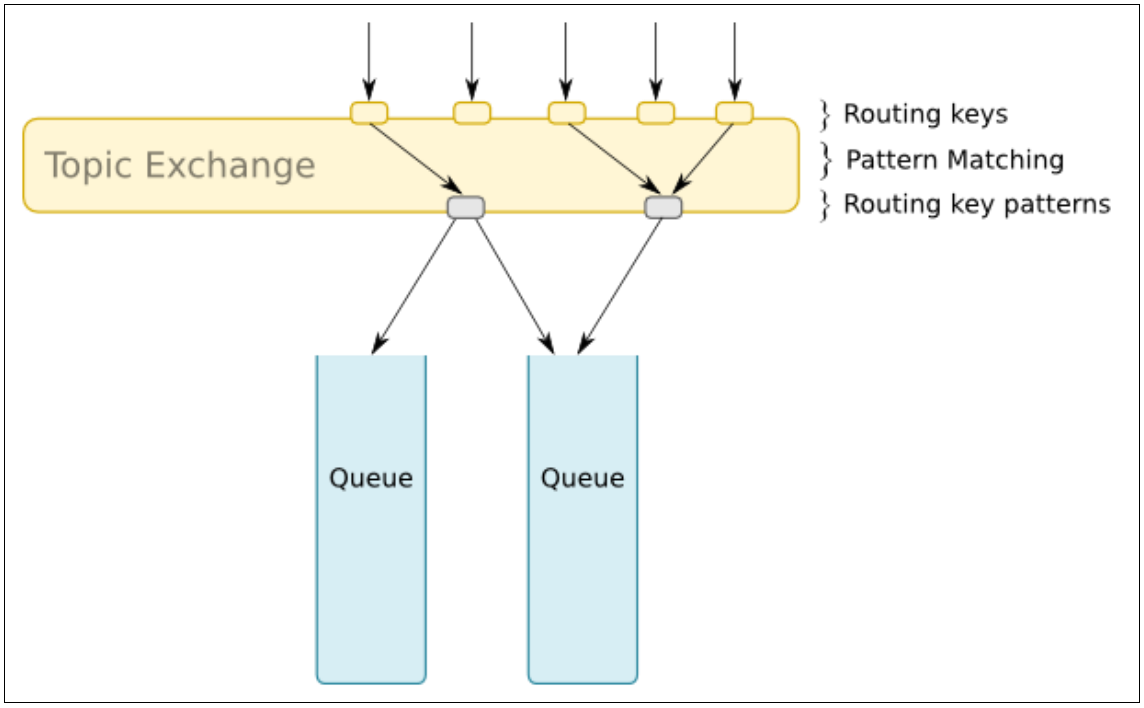
---

# Exchanges

Los exchanges son las entidades a las que los productores envían los mensajes, y desde donde se envían los datos a las diferentes colas. Los mensajes se envían a una o más colas según el exchange y la clave de enrutamiento (routing_key) con que se publican. Para establecer una analogía con el envío de mails podemos definir que un mensaje se publica en una dirección como routing_key@exchange. Veamos los tipos de exchange que existen.

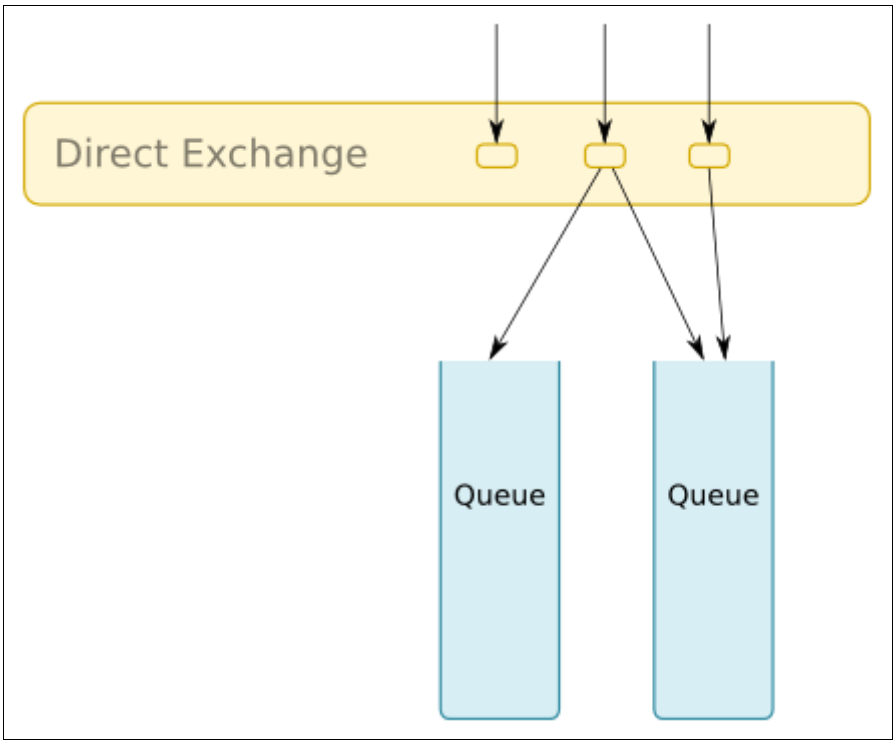
**Topic exchange**

El topic exchange es el más general. Cuando se publica un mensaje, el exchange determina qué colas lo recibirán comparando la routing_key del mensaje con el patrón de enrutamiento (binding) que vinculan la cola con el exchange.



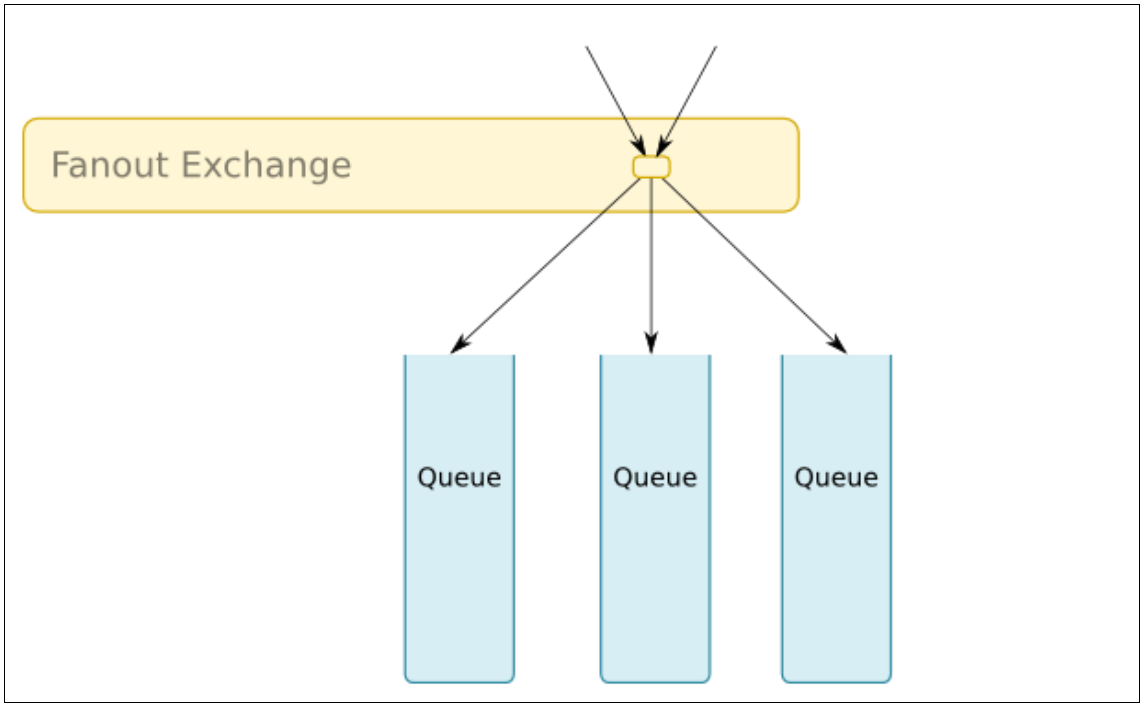

**Direct exchange**

Un direct exchange envía mensajes a un receptor concreto y para ello los productores publican los mensajes en los exchanges con una routing_key específica. Por otro lado, los consumidores crean colas que se vinculan con el exchange mediante una vinculación (binding) que debe contener la misma routing_key. Si la routing_key con la que se publica el mensaje y la routing_key del binding son iguales, el mensaje se envía a la cola y luego es recibido por el consumidor. Es posible que una cola tenga varios routing_keys, y por lo tanto, conectarse a diferentes exchanges. Así mismo, varias colas pueden compartir una routing_key. En esos casos, el exchange multiplica el mensaje y lo envía a varios destinatarios.




**Fanout exchange**

El fanout exchange es muy simple. Las claves de enrutamiento no se tienen en cuenta, por lo que solo hay un punto final al que se publican todos los mensajes y se pueden vincular las colas. Todos los mensajes publicados en el exchange se transmitirán a todas las colas vinculadas al mismo.




**Header exchange**

El header exchange el header de un mensaje para enrutar mensajes hacia diferentes colas. A diferencia de los exchanges anteriores, con el header exchange se pueden lograr combinaciones de enrutamiento más complejas que solo utilizando una única routing_key.

# Colas

Cada cola tiene su propio nombre, que la identifica entre las demás entidades. El nombre puede ser definido por un cliente o por el broker de manera automática. Una cola es un fragmento de memoria que puede estar asociada con almacenamiento no volátil o en la memoria RAM del sistema. La versión no volátil asegura que, incluso después de reiniciar el broker, la cola persiste. Sin embargo, no garantiza que los mensajes se guarden de forma permanente, ya que depende de la configuración de los mensajes cómo serán tratados.

El mecanismo intrínseco de AMQP permite enviar mensajes a las colas y garantizar su procesamiento, ya que tiene un mecanismo de ACK - acuse de recibo - con el que es posible confirmar la recepción y procesamiento adecuado del mismo. Si un mensaje que tiene asociado un ACK no es confirmado por un consumidor - ya sea porque se perdió la conexión, porque se produjo un fallo de la aplicación al procesar, o cualquier otro problema - el broker vuelve a encolar el mensaje nuevamente, de manera tal que pueda ser consumido y procesado nuevamente.

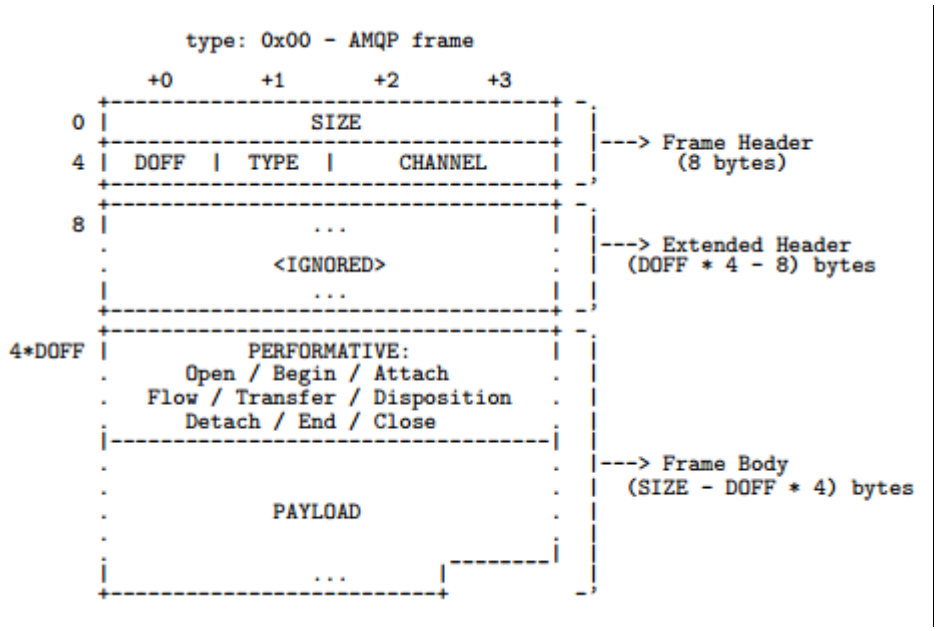
También es posible rechazar la aceptación de un mensaje - conocida como NACK - conscientemente. Esto puede ser útil cuando el procesamiento del mensaje no funciona. La respuesta del consumidor implica que el broker pueda borrar el mensaje o bien enviarlo a otras colas.

# Mensajes

Los mensajes son publicados a un exchange y consisten en un header y un cuerpo o payload. El cuerpo del mensaje no es tomado en cuenta por el broker, ya que contiene la información que intercambian las aplicaciones. El broker sólo toma los datos del header, que contiene estas propiedades:

- **routing-key:** este campo se usa de diferentes formas dependiendo del tipo de exchange.
- **modo de entrega:** indica que un mensaje puede ser perdurable, y en estos casos, el broker hace un intento de mejor esfuerzo para impedir la pérdida del mensaje antes de que sea consumido. Los modos de entrega no persistentes no muestran este tipo de comportamiento.
- **prioridad:** un indicador en el rango entre 0 y 9 para indicar la prioridad del mensaje.
- **vencimiento (expiration):** la duración en milisegundos antes de que el broker pueda descartar el mensaje si no fue consumido.

En esta imagen podemos ver un formato genérico de mensaje AMQP donde se ven sus datos asociados.




# Bindings

Una vinculación (binding) es una relación entre una cola y un exchange, que especifica cómo fluyen los mensajes. Para ello, una vez que están creados un exchange y una cola, es necesario indicarle al broker que vincule estas dos entidades mediante un binding. Dentro de las propiedades del binding se debe especificar una routing_key, que, según el tipo de exchange, realizará el enrutamiento adecuado a las colas.
Un exchange entregará como máximo una copia de un mensaje a una cola si hay correspondencia entre las propiedades del mensaje y las propiedades con la que se realizó el binding. Para cada exchange hay interpretaciones específicas que vemos a continuación.

- Un direct exchange considera que hay correspondencia cuando la routing_key del mensaje y la routing_key del binding son idénticas.
- Un fanout exchange siempre considera que hay correspondencia, ya que no se toma en cuenta la routing_key del mensaje.
- Un topic exchange considera que hay correspondencia si la routing_key de un mensaje coincide con las palabras claves de la routing_key de un binding. Estas palabras son cadenas de caracteres separadas por puntos. En este tipo de correspondencia se establecen el caracter asterisco (*) que reemplaza una palabra, y el caracter numeral (#), que reemplaza cero o más palabras.
- Un header exchange considera que hay correspondencia cuando coinciden los pares clave-valor de un mensaje con los pares clave-valor de un binding.
- También es posible implementar un exchange propio no definido por la especificación y generar otros criterios de correspondencia.

El concepto de vincular colas con exchanges tiene propiedades poderosas, ya que mantiene las aplicaciones que producen aisladas de las aplicaciones que consumen. Luego, se pueden realizar las configuraciones dentro del broker que permiten enrutar los mensajes adecuadamente sin necesidad de cambiar las aplicaciones productoras o consumidoras.
Es posible realizar configuraciones para vincular una única cola a múltiples exchanges, o varias colas a un mismo exchange. Esto en la práctica resulta muy flexible y permite implementarlo a nivel de protocolo y no de aplicación, liberando así de parte de desarrollo de código.

# Versiones

Como vimos, el protocolo fue definido entre 2004-2006 por un conjunto de compañías asociadas al sector financiero, y que luego fue sumando diferentes entidades. A continuación, podemos ver la cronología de las versiones publicadas hasta el momento.:

- La versión 0-8 fue publicada en junio de 2006.
- La versión 0-9 fue publicada en diciembre de 2006.
- Para la versión 0-10 no se tiene una fecha exacta aunque fue entre 2007 y 2008.
- En noviembre de 2008 se publicó la versión 0-9-1.
- En mayo de 2010 presentaron un draft de la versión 1.0.
- En octubre de 2011 presentaron la versión final 1.0.

**AMQP 1.0 vs. 0-9-1**

En la actualidad existen dos versiones completamente independientes de AMQP. El grupo OASIS fue el encargado de desarrollar y definir la versión 1.0 más reciente del protocolo. Así mismo, RabbitMQ - un broker AMQP muy completo y utilizado - implementa la versión 0-9-1. Ambas versiones no son compatibles entre sí, por lo que es necesario tener en cuenta este detalle antes de implementar alguna de las dos versiones del protocolo.

# Implementaciones

Para el protocolo AMQP existen diferentes implementaciones que están adaptadas a versiones específicas, como así también a entornos específicos broker o clientes. A continuación, podés ver las implementaciones públicas de AMQP.

- **Broker y cliente**
  - OpenAMQP, la implementación original en código abierto de AMQP, escrita en C por iMatix. Funciona bajo Linux, AIX, Solaris, Windows y OpenVMS. Incluye un broker, APIs para C/C++ y Java JMS, una consola de administración remota, secuencias de comandos, federación, conmutación por error y AMQP-sobre-HTTP mediante el protocolo RestMS.
  - AMQP Infrastructure, un paquete instalable mediante yum que cumple con AMQP 0-10 y mantenido para las últimas versiones de Fedora. Incluye el broker, herramientas de administración, agentes y clientes.

    - Apache Qpid, un proyecto de la Apache Software Foundation. Posee vinculaciones para muchos lenguajes evitando el uso de librerías dinámicas.
    - Red Hat Enterprise MRG implementa la versión 0-10 de AMQP cuya gama de características incluye federación, distribución en modo activo-activo usando Qpid como upstream, una consola web y otras características orientadas a empresas.

- **Solo broker**

    - RabbitMQ, una implementación independiente de código abierto. El servidor está escrito en Erlang y también incluye un cliente de referencia escrito en Java.
    - ZeroMQ, una plataforma de mensajería que es capaz de tratar a los brokers de AMQP como nodos de una red de mensajería distribuida. Tiene vinculaciones a su implementación subyacente en C, así como a otros lenguajes como Python, C++, Lisp, Ruby y otros.
    - Zyre, un corredor que implementa RestMS y AMQP para permitir acceso HTTP basado en REST a redes AMQP.

- **Solo cliente**

    - DE.SETF.AMQP, una librería cliente de AMQP para Common Lisp.
    - amqplib, una librería cliente de AMQP 0-8 para Python, probada con RabbitMQ.

# Conclusiones

AMQP es un protocolo muy completo, y resulta muy conveniente su utilización en proyectos con arquitecturas distribuidas, para microservicios, y para intercambiar información entre distintas aplicaciones - tanto internas como externas - a través de un lenguaje común que es un mensaje AMQP. Para hacer un resumen, en este artículo vimos los siguientes temas.

- Qué es el protocolo AMQP.
- La historia asociada al desarrollo del protocolo.
- Las entidades principales que juegan un rol clave para intercambiar información.
- Qué son los exchanges, y las descripciones generales de cada tipo.
- Las colas y los mensajes que consumen las aplicaciones.

- Las vinculaciones entre los exchanges y las colas, para poder transmitir un mensaje de origen a destino.
- Las versiones del protocolo y su evolución a lo largo del tiempo.
- Las implementaciones de AMQP para diferentes entornos y lenguajes de programación.

# Bibliografía


- Wikipedia, AMQP - [*Disponible: Julio 2021*]
- IONOS, AMQP - [*Disponible: Julio 2021*]
- Python Hosted, AMQP Protocol - [*Disponible: Julio 2021*]
- Ably, Intro to AMQP 1.0 - [*Disponible: Julio 2021*]