Sistemas Distribuidos I
(75.74)

Message Oriented Middlewares (MOMs)

Definición. ZeroMQ. RabbitMQ.

Docentes

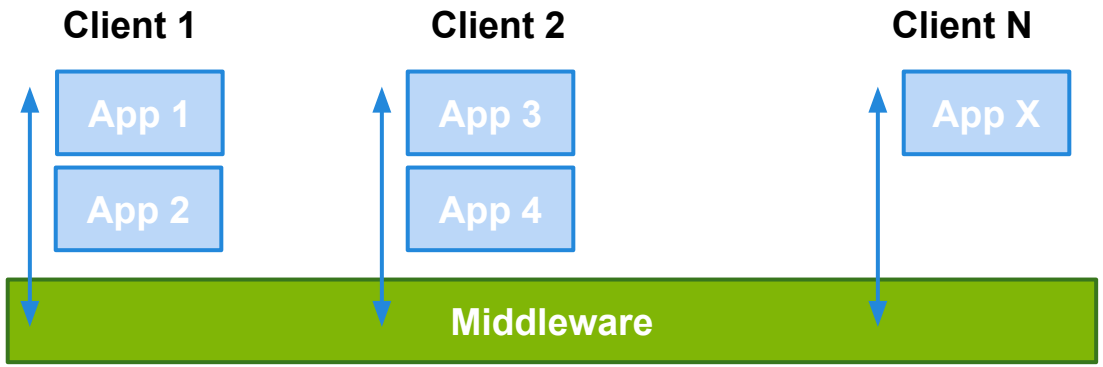
- Pablo D. Roca
- Ezequiel Torres Feyuk
- Ana Czarnitzki
- Cristian Raña

Agenda

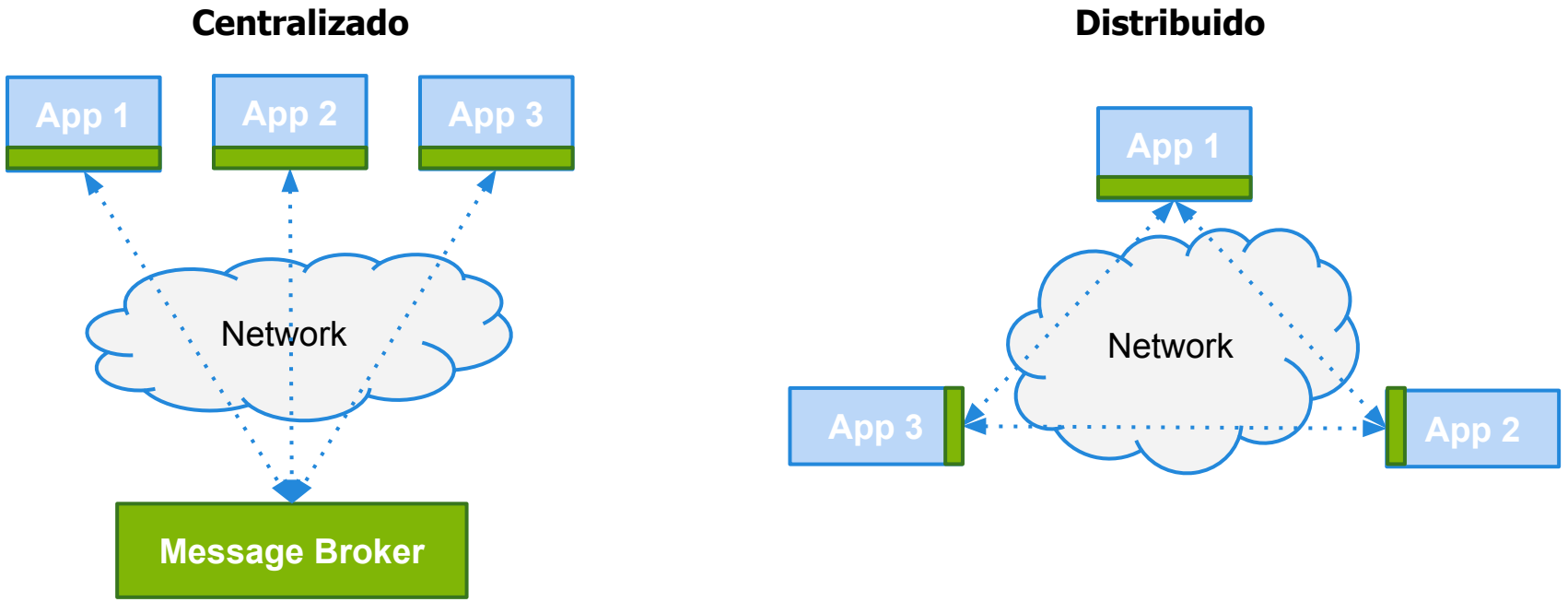
- **MOMs**
- ZeroMQ
- RabbitMQ

MOM | Introducción

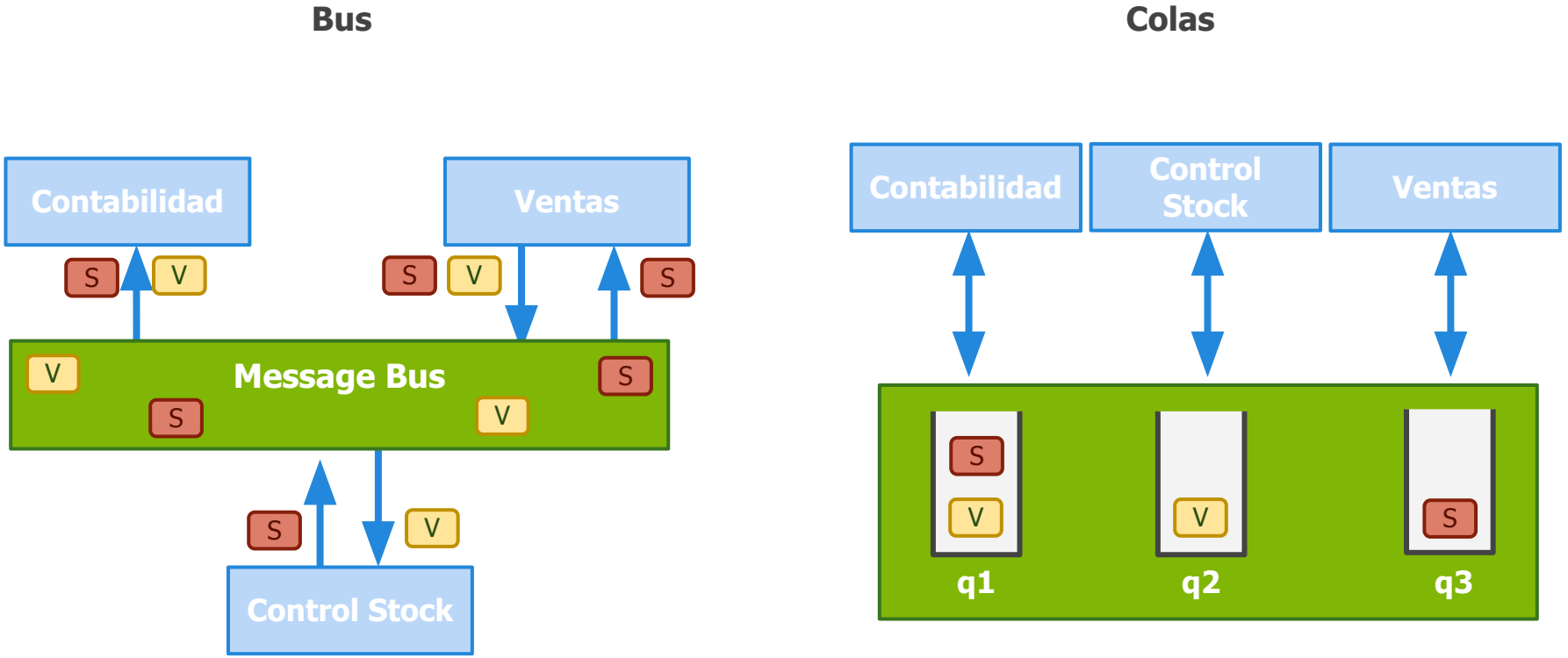
- Implementan la comunicación de grupo de forma transparente a las aplicaciones que la requieren.
- Basan su funcionamiento en el simple concepto de comunicar mensajes entre aplicaciones.
- Resuelve problemas de transparencia respecto de ubicación, fallos, performance y escalabilidad.



MOM | Centralizado vs Distribuido



MOM | Bus de Información vs Colas de Mensajes



Pros

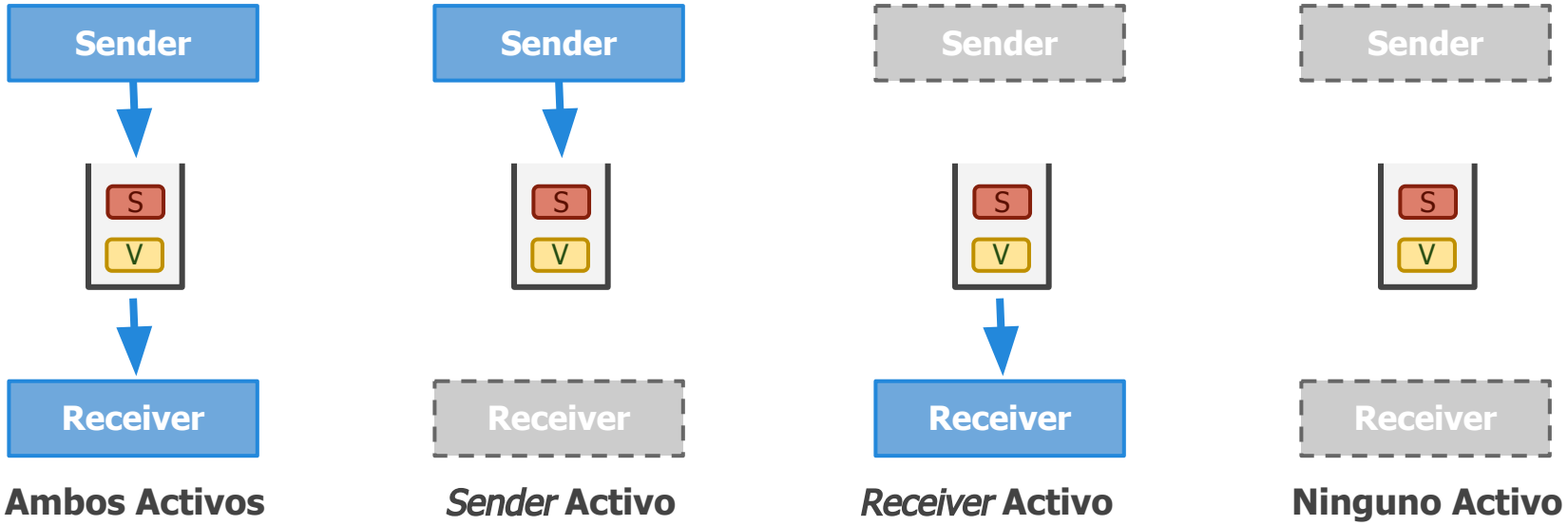
- Se modela como una conexión punto a punto
- Permite obtener respuestas instantáneas a pedidos concretos

Contras

- No permite implementar transparencia frente a errores

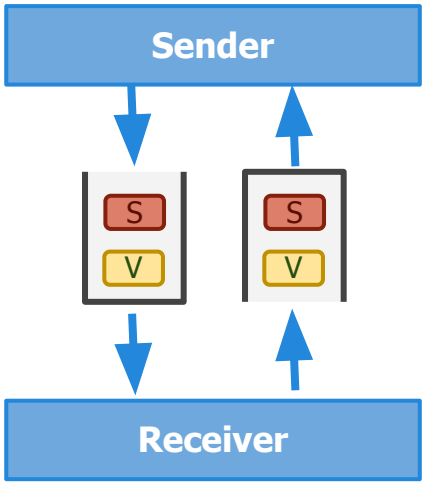
Pros

- Se modela naturalmente con colas
- La arquitectura soporta períodos de discontinuidad del transporte



Contras

- Es complejo recibir respuesta a pedidos realizados (mínimamente es necesario contar con colas para el retorno de info)

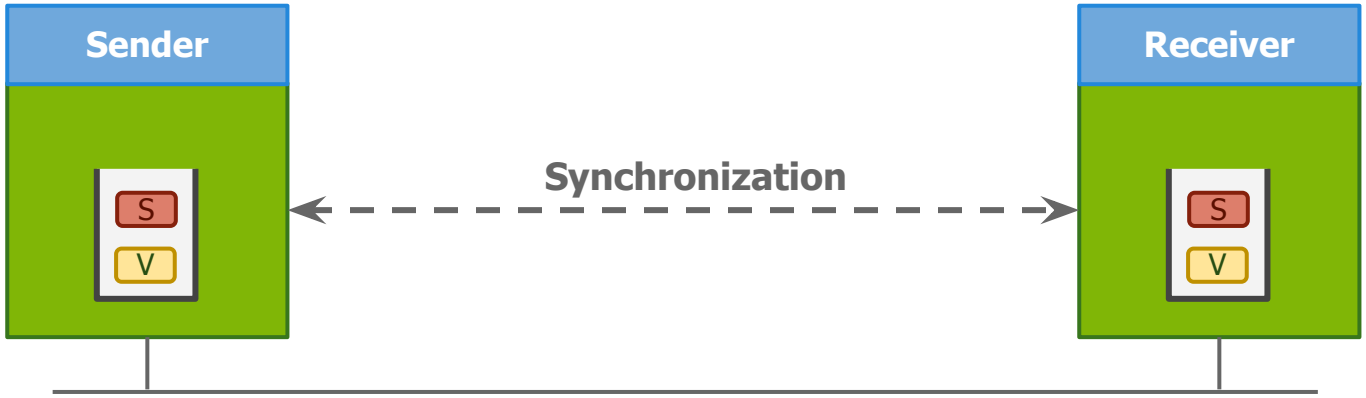


MOM | Operaciones Comunes

- **put**: publicación de un cierto mensaje
- **get**: esperar hasta que un mensaje sea detectado. Luego, eliminarlo de la cola y retornarlo
- **poll**: revisar mensajes pendientes, sin bloquear
- **notify**: asociar un *callback* utilizado por el MOM para ser ejecutado frente a publicación de ciertos mensajes

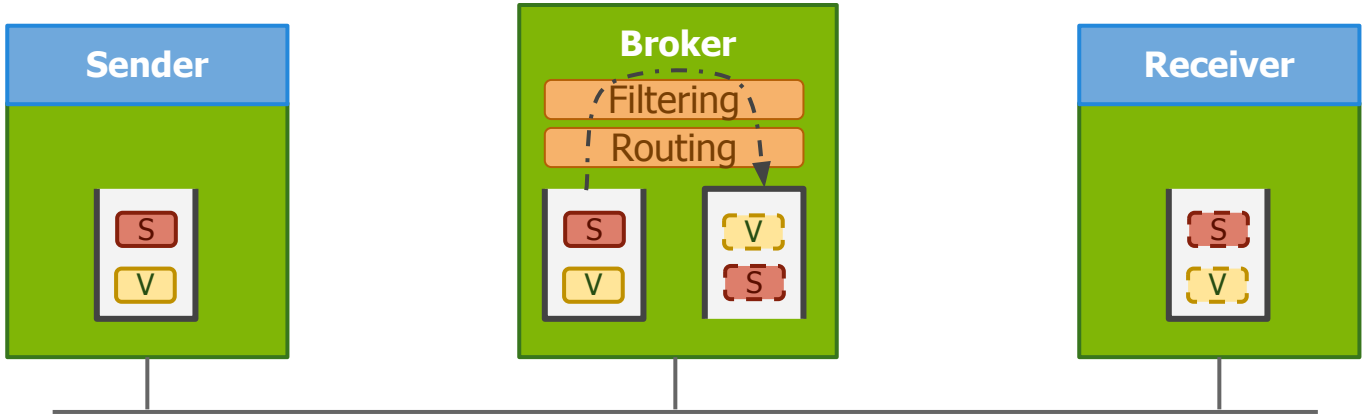
MOM | Colas de Mensajes y Broker

- Pueden existir varias definidas dentro del MOM
- Tienen nombre y longitud definidas
- Los clientes suelen contar con colas privadas intermedias
- Garantía al Emisor de que el mensaje será insertado



MOM | *Brokers*

- Proveen transparencia de localización tanto al Emisor como al Receptor.
- Soportan lógica en el *middleware* para filtrar y modificar mensajes.
- Brindan un punto de control y monitoreo

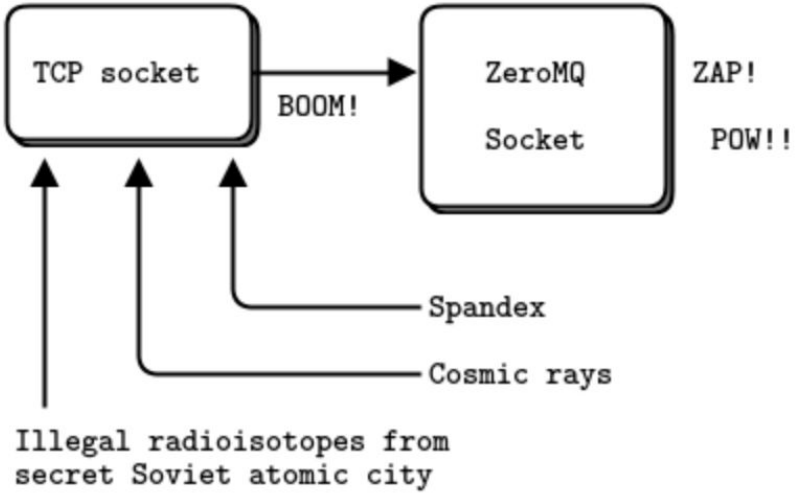


Agenda

- MOMs
- **ZeroMQ**
- RabbitMQ

ZeroMQ | Definición según sus autores

We took a normal TCP socket, injected it with a mix of radioactive isotopes stolen from a secret Soviet atomic research project, bombarded it with 1950-era cosmic rays, and put it into the hands of a drug-addled comic book author with a badly-disguised fetish for bulging muscles clad in spandex. Yes, ZeroMQ sockets are the world-saving superheroes of the networking world.



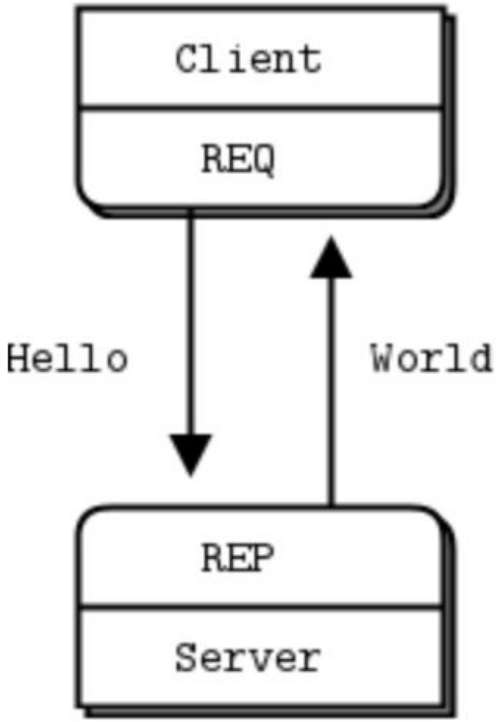
ZeroMQ | Introducción

- Sockets on steroids
- Altamente performante
  - Aunque hay mejores opciones
- Herramienta útil para crear **brokerless middlewares**
- Serialización **a cargo** del usuario
- Soporte para diferentes patrones de mensajería
  - Request-Reply, Publisher-Subscriber, Parallel Pipeline, Patrones avanzados

ZeroMQ | Tipos de conexiones

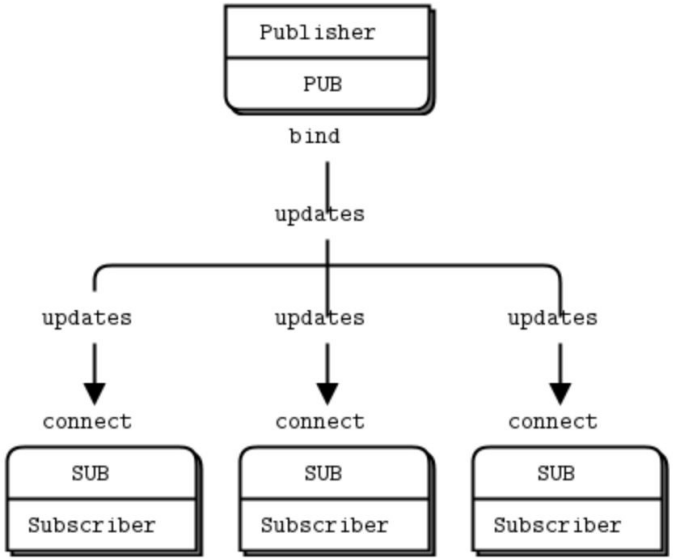
- **TCP**
  - Multicomputing
  - Unicast (Point to Point)
- **IPC**
  - Multiprocessing
  - Comunicación a través de Unix sockets
- **Inproc**
  - Multithreading
  - Queue entre threads
- **Otras**
  - Multicast a través del protocolo PGM

- Cliente-Servidor convencional (aunque no tanto...)
- No posee primitiva ***accept***
- Primita ***bind*** funciona como *bind + accept*
- Primitiva send es **no bloqueante**
- Cliente no necesita esperar a que el servidor esté corriendo para enviar mensajes
- Cómo funciona *under the hood?*
  - I/O threads: 1 thread per GB/s (in or out)
  - Buffering

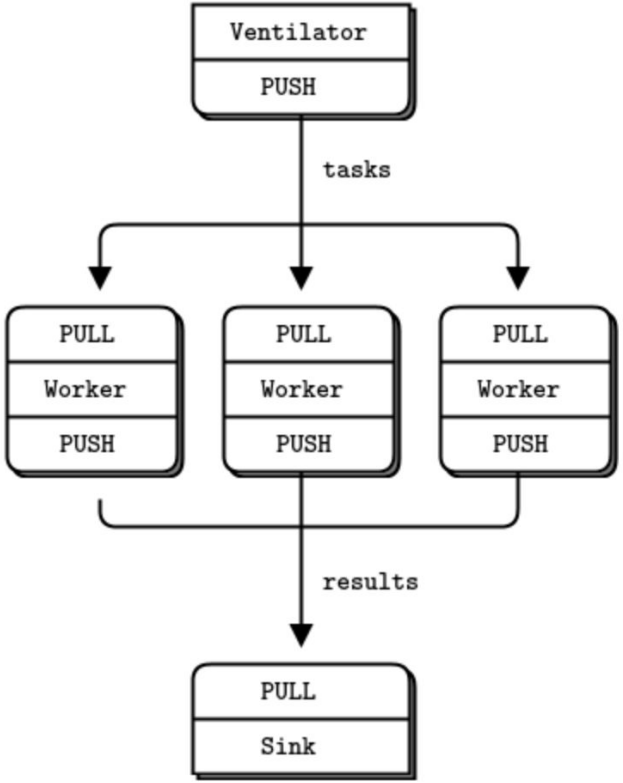


ZeroMQ | Patrones | Publisher-Subscriber

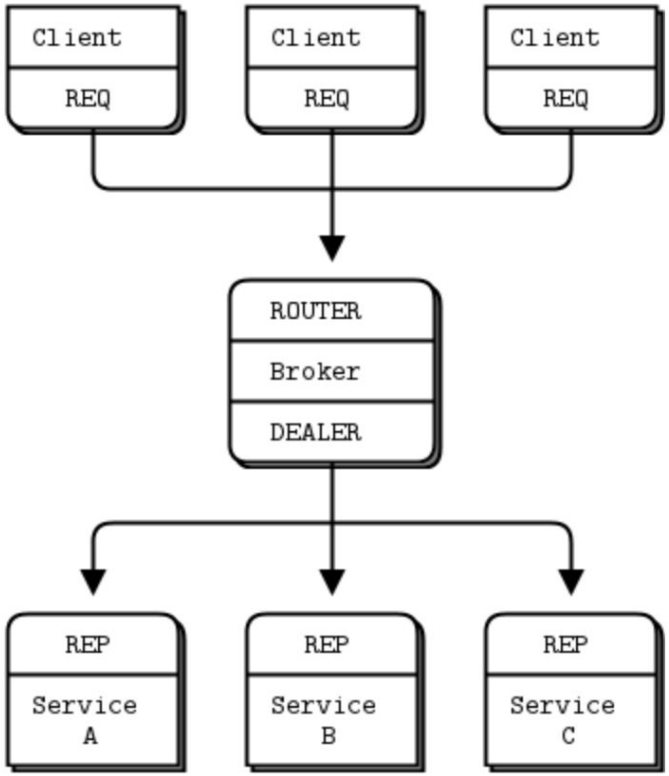
- Un ZMQ PUB socket publica mensajes
- Message pattern: `id field1 field2 ...fieldN`
- N ZMQ SUB sockets se suscriben a los Eventos que desean recibir suscribiendose al ID del evento
- Suscripción puede ser cancelada en cualquier momento
- Mensaje es enviado a todos los sockets suscriptos a un evento determinado
- Múltiples publishers? [XPUB-XSUB pattern]



- Patrón Productor-Consumidor
- Chaining de Productores-Consumidores da como resultado un pipeline
- Mensajes son consumidos de forma Equitativa **(fairness)**
  - Qué lógica utiliza para decidir esto?
- Combinaciones
  - 1 PUSH -> N PULL
  - N PUSH -> 1 PULL



- **ROUTER socket**
  - Agrega al mensaje recibido un ID de destinatario
- **DEALER socket**
  - Rutea los mensajes de forma justa (**fair**)
  - Propaga el ID de origen del mensaje
- Ambos sockets permiten recibir mensajes de múltiples sockets a la vez
- Ambos sockets son asincrónicos
  - **Poll** para recibir mensajes

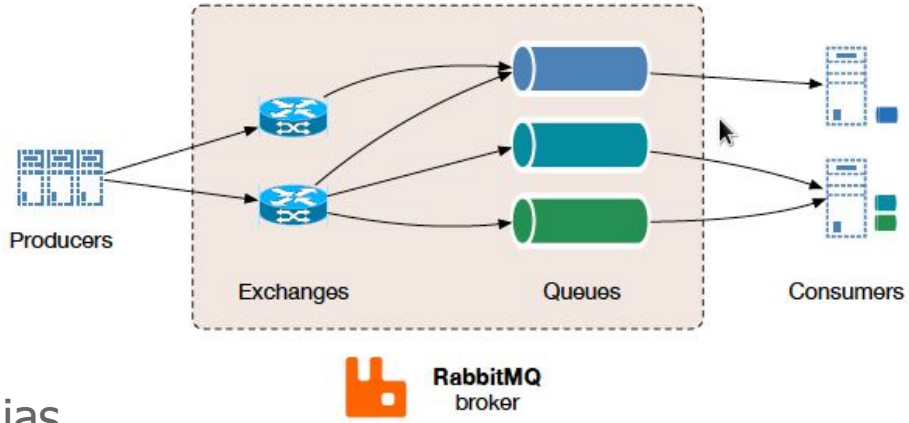


Agenda

- MOMs
- ZeroMQ
- **RabbitMQ**

RabbitMQ | Conceptos

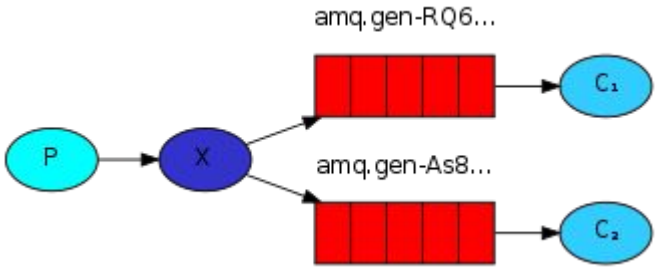
- **Queues**
  - Nombradas vs TaskQueues vs Anónimas
  - Acknowledge: automática por defecto
  - Durabilidad: debe ser definida en la cola y en cada mensaje.
- **Exchanges**
  - Implementan diferentes estrategias para transmitir mensajes
  - Tipos: fanout, direct, topic, headers



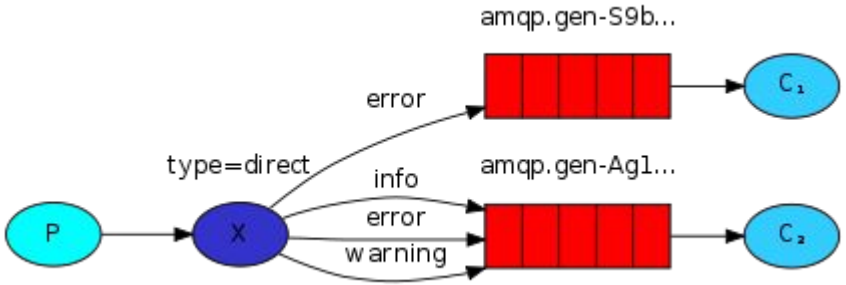
https://content.pivotal.io/blog/understanding-when-to-use-rabbitmq-or-apache-kafka

RabbitMQ | Patrones | Publisher-Subscriber

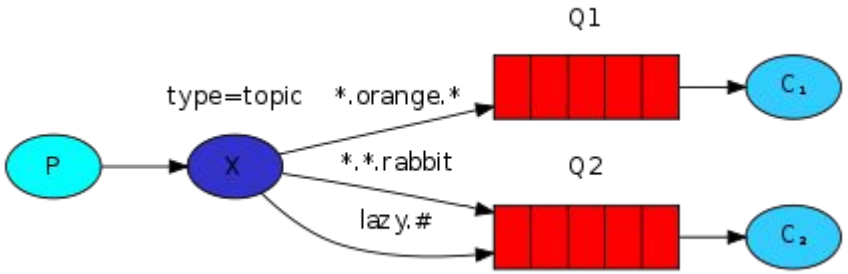
- Productor envía mensajes a un exchange de tipo *fanout*
- Consumidores crean colas anónimas para recibir mensajes del productor
- Colas anonimas son *bindeadas* a exchange del Productor para comenzar a recibir mensajes
- **Exchange Fanout:** Realiza un *broadcast* de de todos los mensajes recibidos a todas las colas conocidas



- **Productor**
  - Envía mensajes a un exchange de tipo *direct*
  - Adosa al mensaje un identificador De routeo (*routing_key*)
- **Consumidor**
  - Realiza binding a exchange *direct* con los *routing_keys* que desea recibir
- **Exchange Direct:** Redirige mensajes con una *routing_key* especifica a aquellas colas que se encuentran *bindeadas* a la misma



- **Productor**
  - Envía mensajes a un exchange de tipo *topic*
  - Adosa al mensaje un identificador de routeo (*routing_key*)
- **Consumidor**
  - Realiza binding a exchange *topic* con los *patrones* que desea recibir
- **Exchange Topic:** Soporta patrones de búsqueda basados en palabras. *routing_key* es un conjunto palabras separadas por punto
  - *: Permite sustituir una palabra
  - #: Permite sustituir una o más palabra



Bibliografía

- M. Van Steen, A. Tanenbaum: Distributed Systems. 3rd Edition. Pearson Education, 2017.
- ZeroMQ: The Guide: http://zguide.zeromq.org/page:all
- RabbitMQ: https://www.rabbitmq.com/getstarted.html
- Ejemplos ZeroMQ: https://github.com/7574-sistemas-distribuidos/zeromq-examples
- Ejemplos RabbitMQ: https://github.com/7574-sistemas-distribuidos/rabbitmq-example