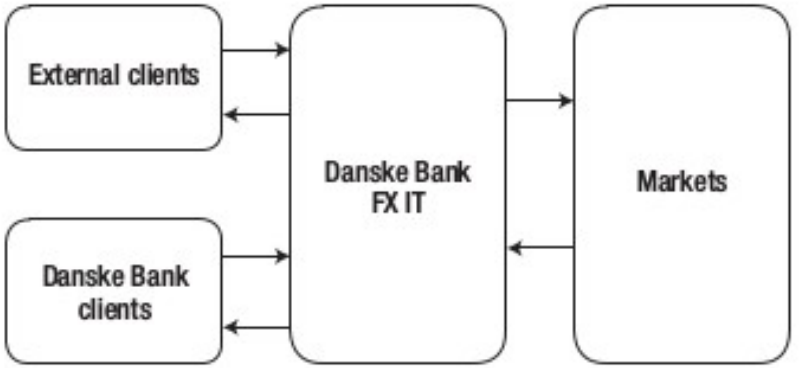
FX IT System

Conway’s Law:

"organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations."

— M. Conway

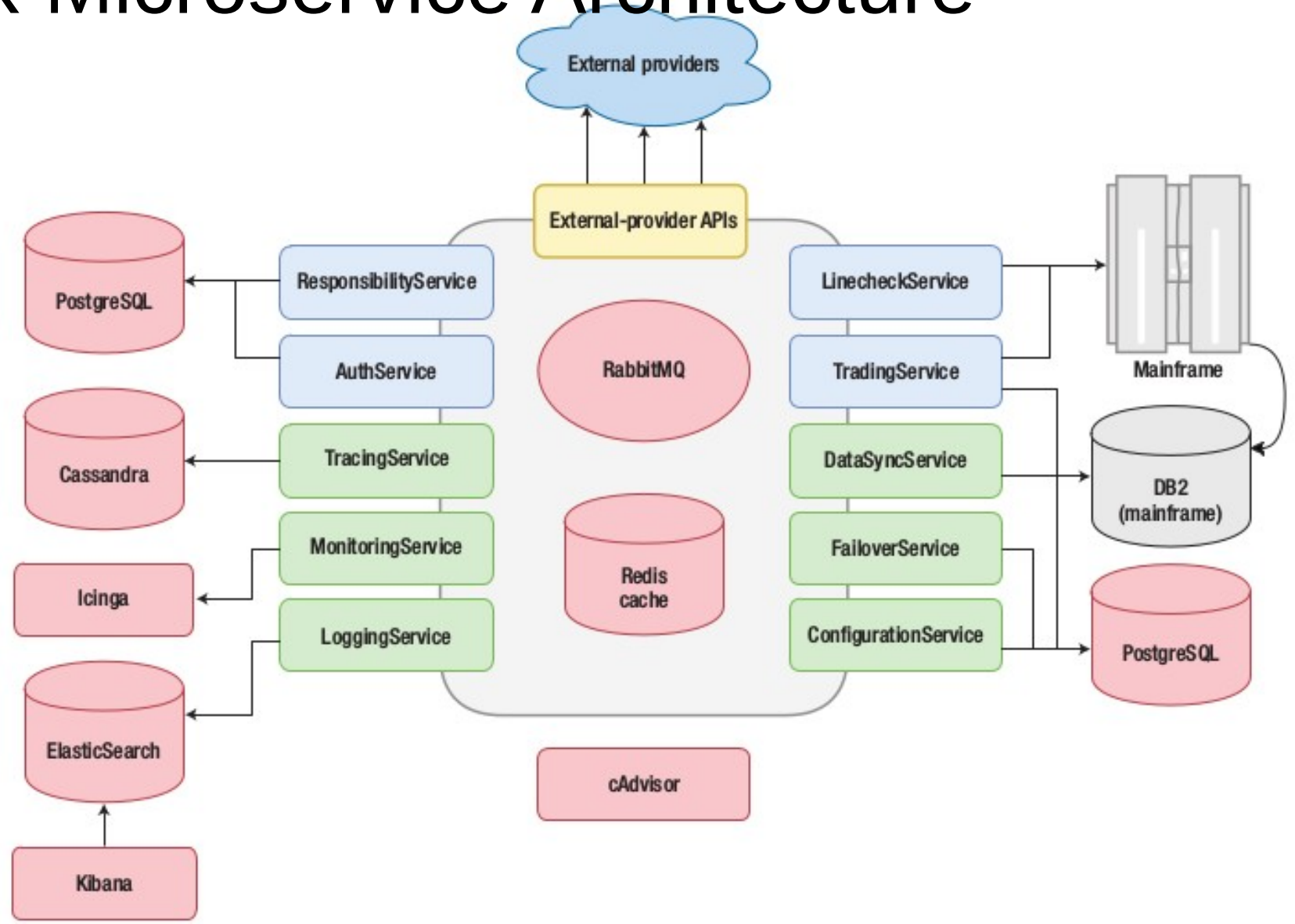
Conway, Melvin E. (April 1968), "How do Committees Invent?"



Bucchiarone, Dragoni, Dustdar, Larsen, Mazzara (2018). From Monolithic to Microservices, An Experience Report from the Banking Domain. IEEE Software, May/June 2018. pp. 51

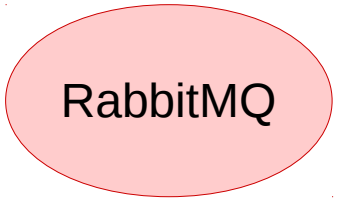
Based on this quote being featured in the article, perhaps we can assume that the architecture of the FX Monolithic Architecture was similar to its communication structures.

FX Microservice Architecture



Bucchiarone, Dragoni, Dustdar, Larsen, Mazzara (2018). From Monolithic to Microservices, An Experience Report from the Banking Domain. IEEE Software, May/June 2018. pp. 52

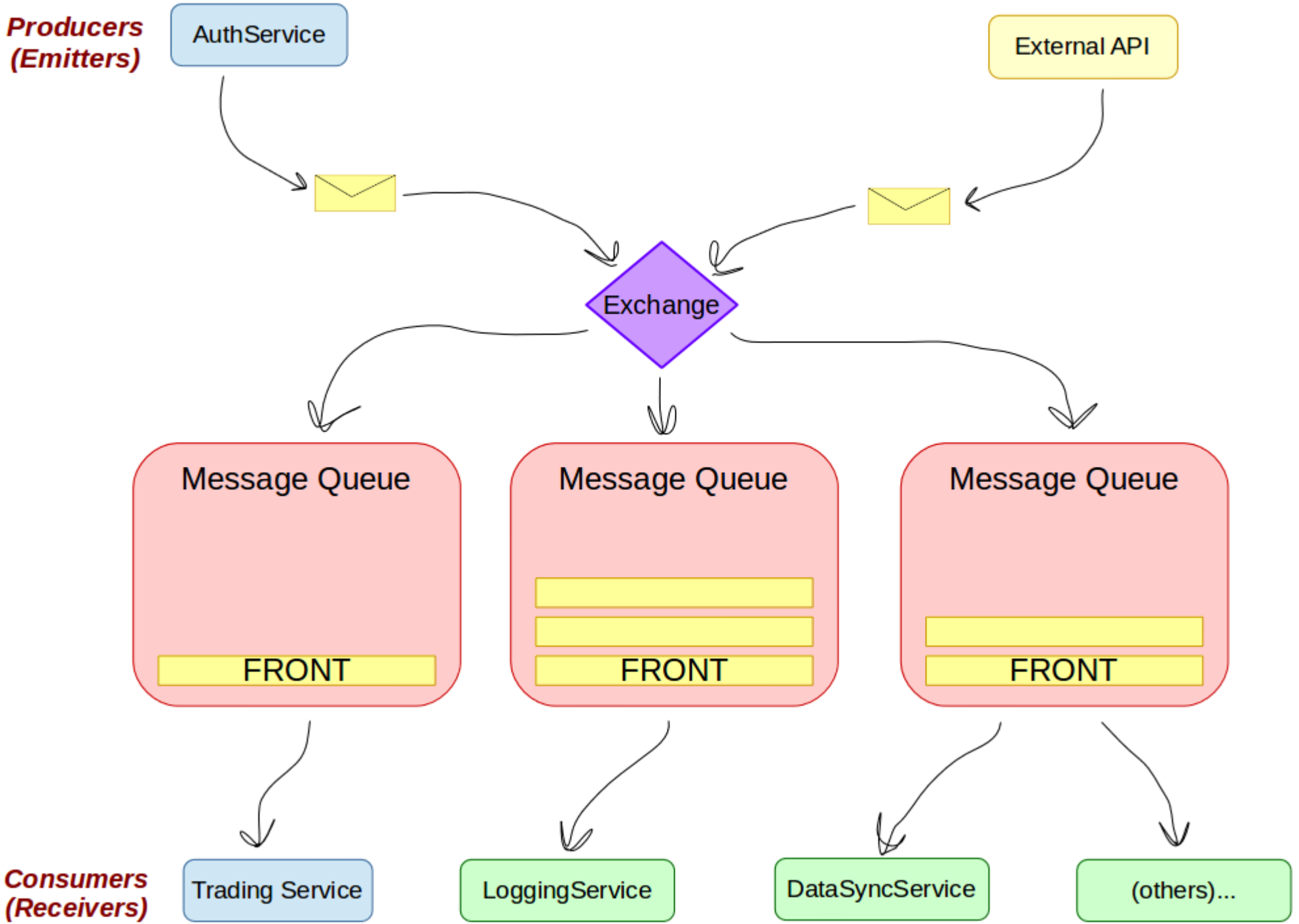
FX Microservice Architecture



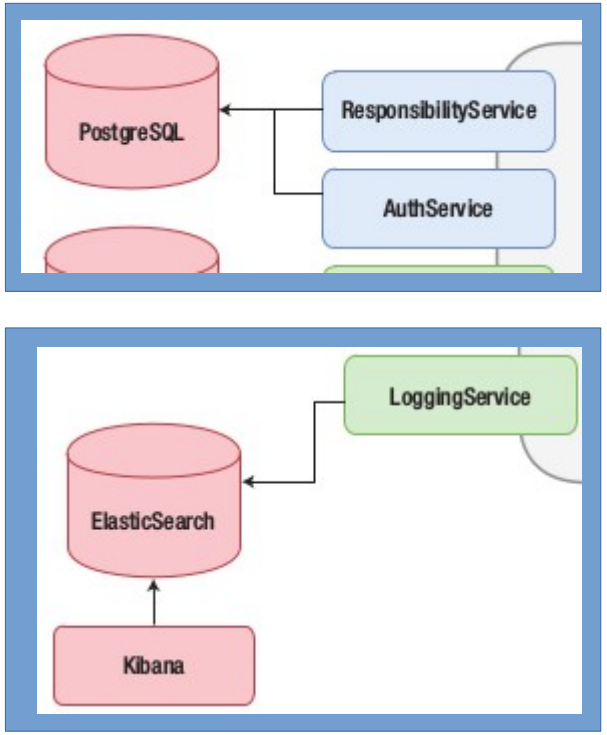
RabbitMQ is a message broker that works to pass messages to/from services.

The message queue receives messages from the internal **services**, as well as from the **external APIs**. These messages carry requests.

The Message Queue stores messages and sends them to the recipient **service**.


Producers
(Emitters)
AuthService
External API
Exchange
Message Queue
FRONT
Message Queue
FRONT
Message Queue
FRONT
Consumers
(Receivers)
Trading Service
LoggingService
DataSyncService
(others)...

FX Microservice Architecture



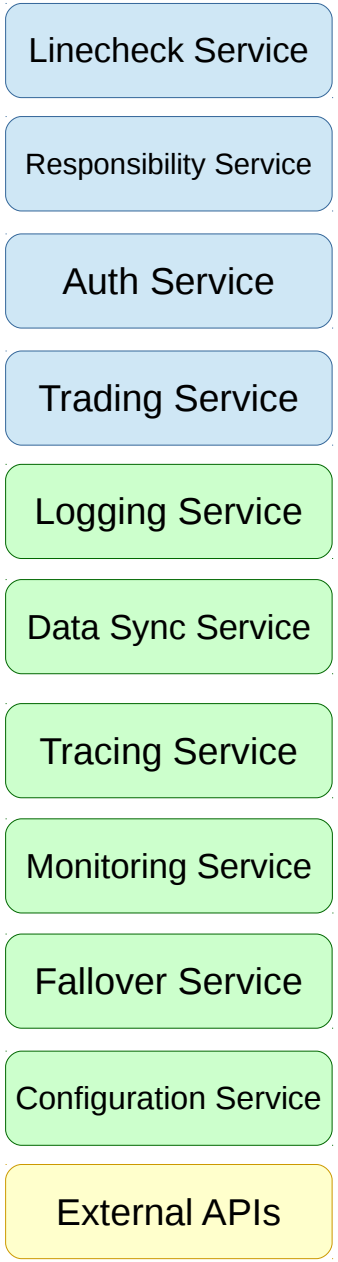
Each service has its own responsibility, and has its own system setup.

Different services may use different databases may be implemented in different programming languages.

And by being independent, services don’t have to wait on shared resources to get their work done.

Bucchiarone, Dragoni, Dustdar, Larsen, Mazzara (2018). From Monolithic to Microservices, An Experience Report from the Banking Domain. IEEE Software, May/June 2018. pp. 52

FX Microservice Architecture



For an example, the Logging service is probably in heavy use – all services probaby log their activity.

By using RabbitMQ, which guarantees delivery, a service can send off a “logging” message and continue on with its work.

In a Monolithic architecture, the logging process would take up resources, and if the system weren’t multithreaded, all processing might halt until logs were written each time.