www.centricconsulting.com

Event-Driven Architecture, Micro-Services, and the Cloud

@ShawnWallace

SAFETY FIRST

BE

SAFETY FIRST

CAREFUL

THIS MACHINE
HAS NO BRAIN
USE YOUR OWN

WARNING

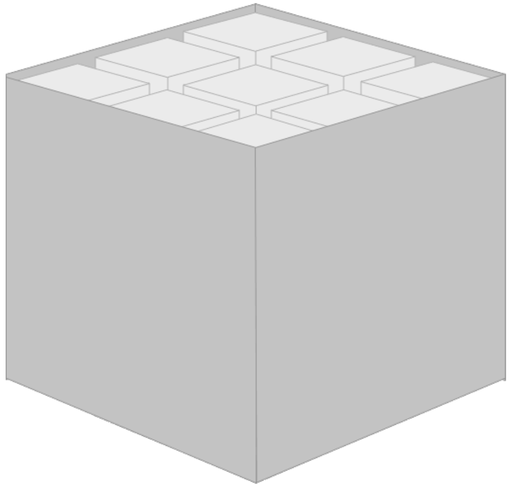
Heavy Object.
Can Cause
Back Injury.
Do not lift
with back.

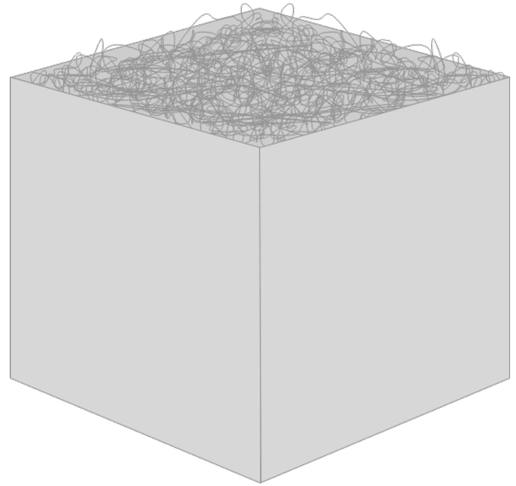
A word about ‘productizing’ patterns and practices

((CENTRIC))

The Great Debate…

http://martinfowler.com/articles/dont-start-monolith.htm

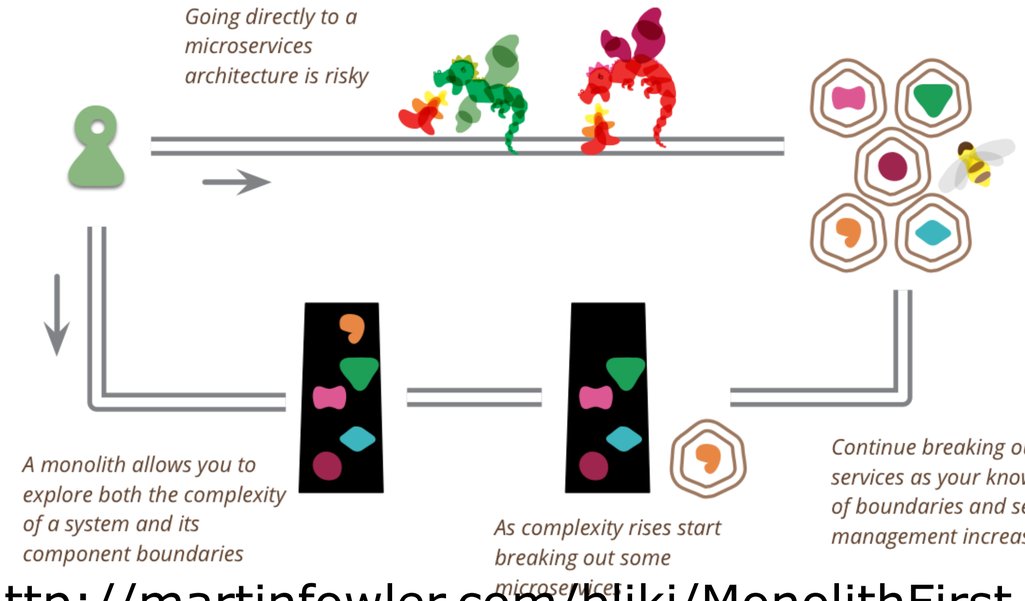




Hope vs. Reality

Micro services First?

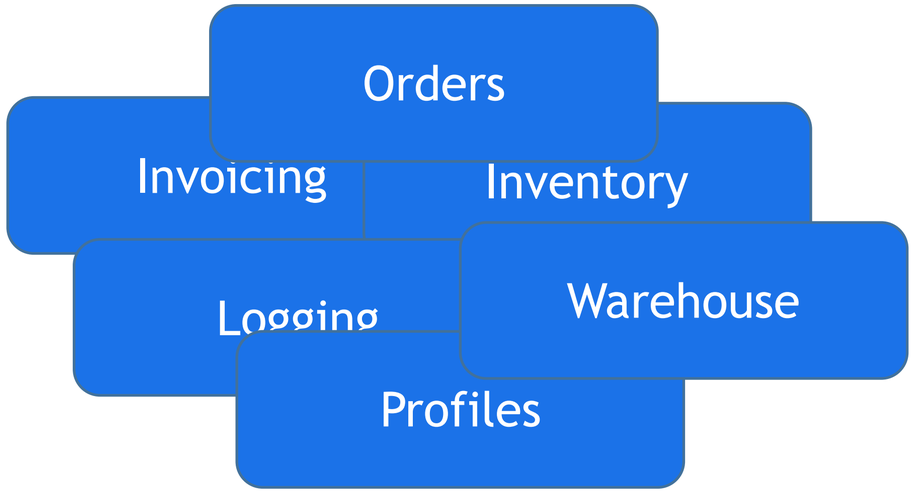
Monolith First?



http://martinfowler.com/bliki/MonolithFirst.html

((CENTRIC))

Monolithic



((CENTRIC))

A problem with ‘Monolith First’

A product of our agile approach to our projects…smaller features finished earlier.

The goal is to deploy more often…this is hard.

One solution is to have smaller applications.

((CENTRIC))

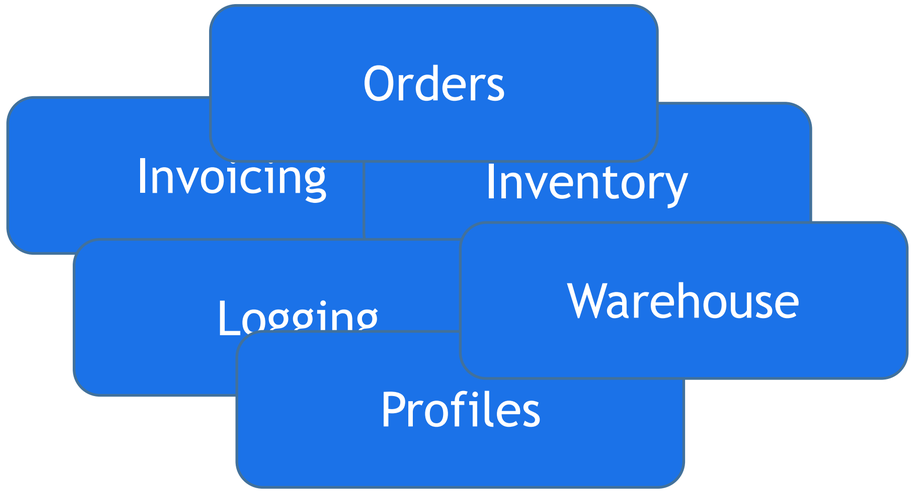
A problem with ‘Monolith First’

A product of our agile approach to our projects…smaller features finished earlier.

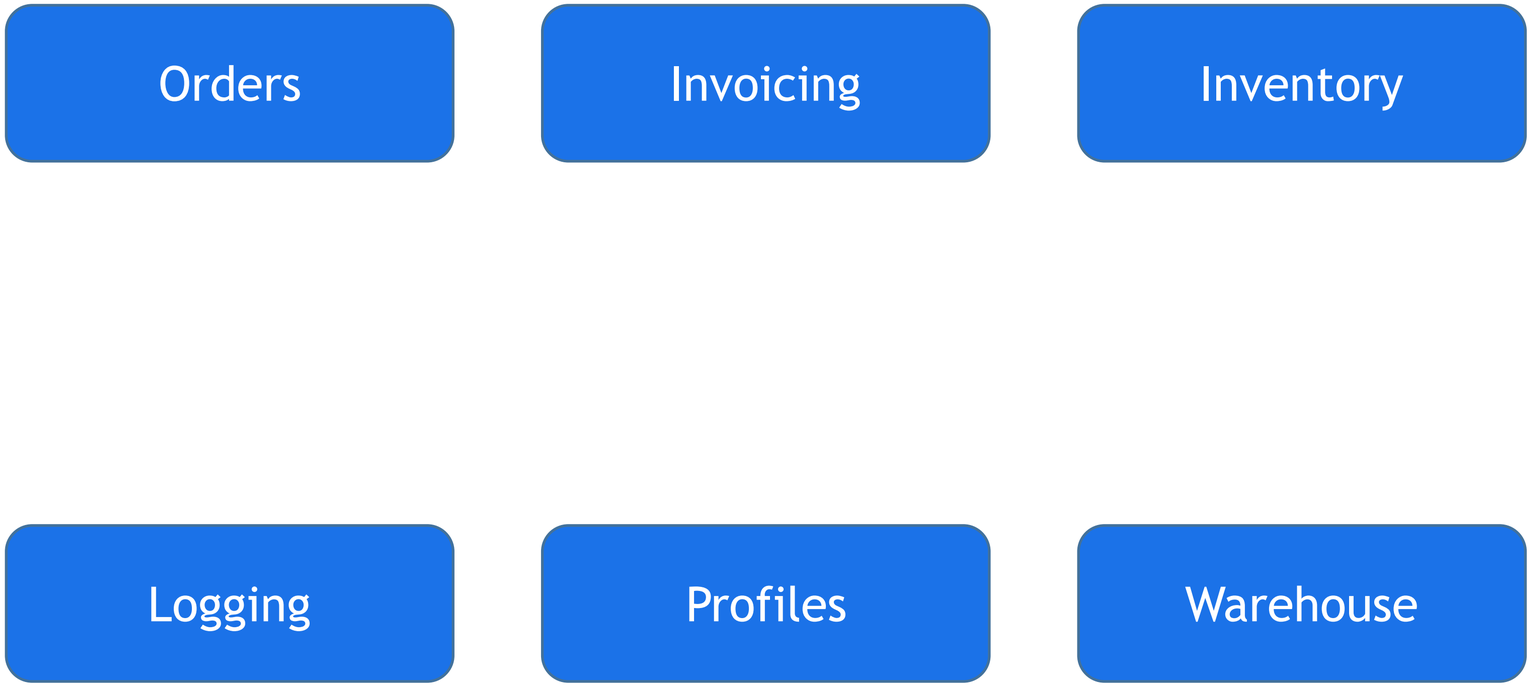
The goal is to deploy more often…this is hard.

One solution is to have smaller applications.

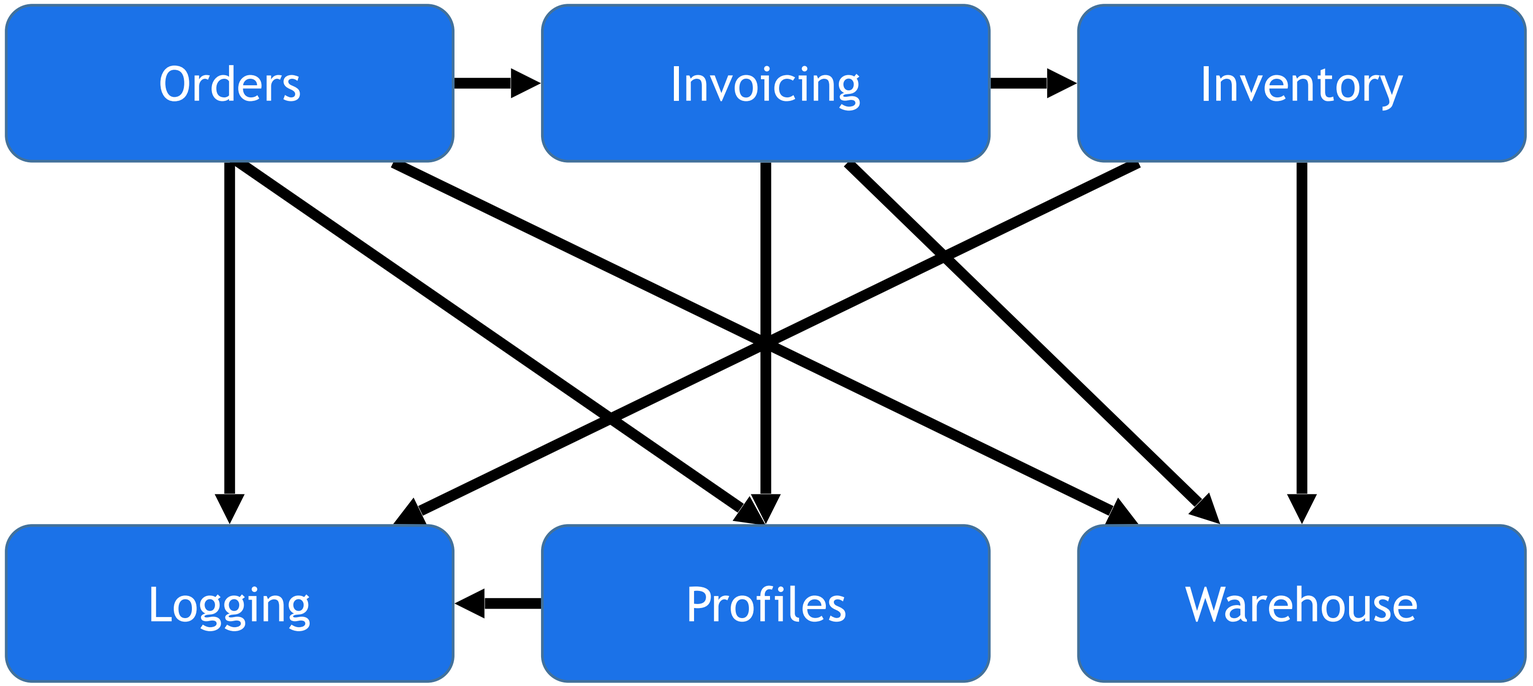
Monolithic



SOA



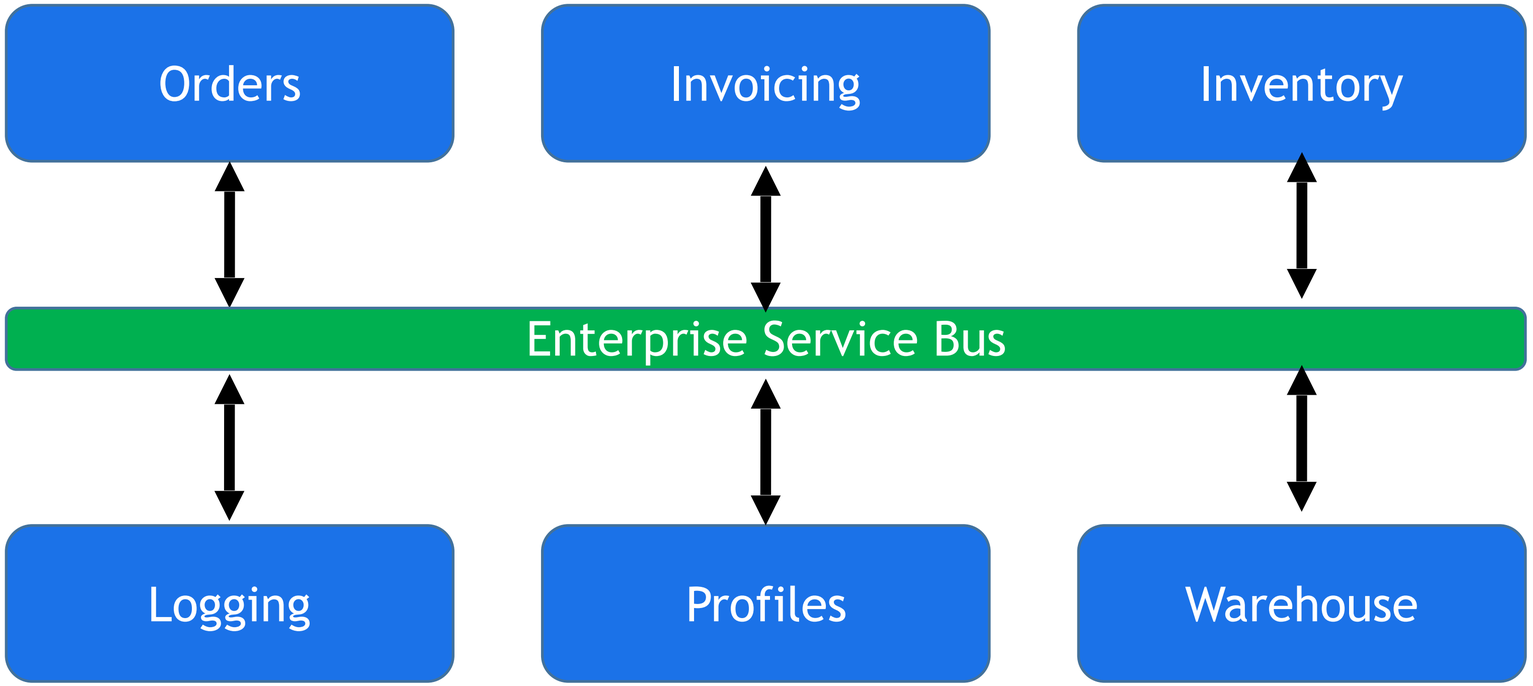
SOA



Tightly Coupled

((CENTRIC))

SOA with Enterprise Service Bus

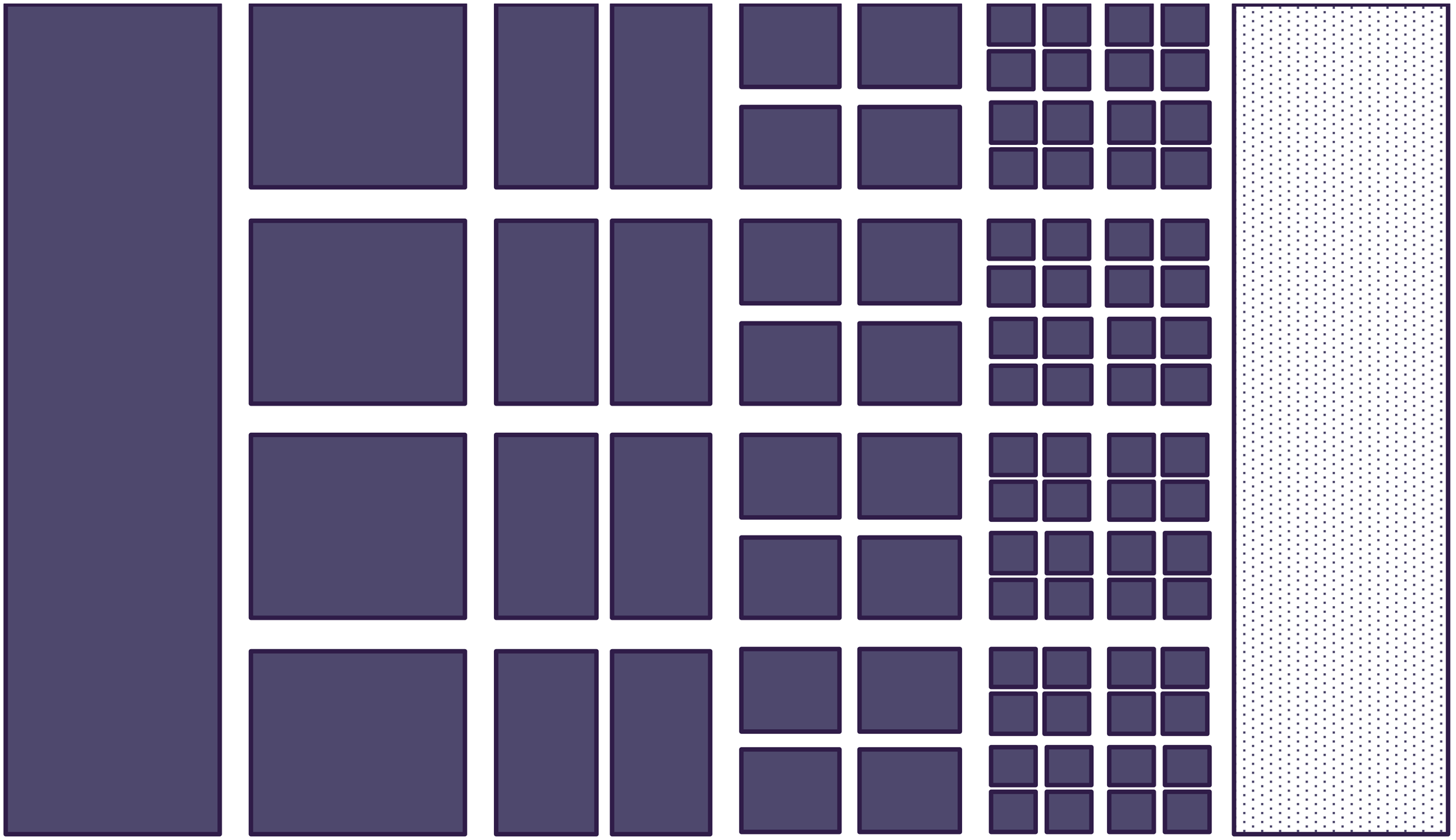


STILL Tightly Coupled

((CENTRIC))

“The Role of the ESB still has its place — now in the form of a modern scalable message queue”

Source: Reactive Micoservices Architecture (Design Principles for Distributed Systems p. 34
Jonas Bonér


MONOLITH
CLIENT SERVER
WEB & INTERNET
CLOUD & WEB SERVICES
MICRO SERVICES
2020 - 2030

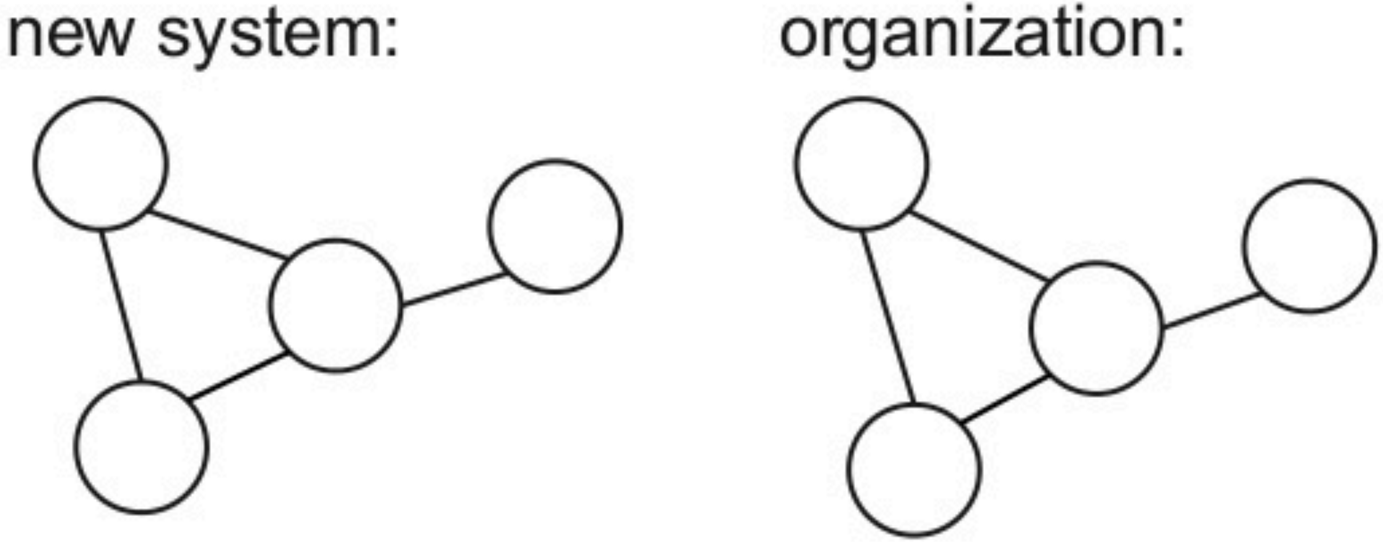
What are Microservices

- Independently testable/deployable
- Operate in isolation (loosely coupled)
- Maintains own state
- Asynchronous external interactions that favor message passing

Important Early Decisions

- Embrace Asynchronous Interactions
- Communication Pattern
- Logging Infrastructure

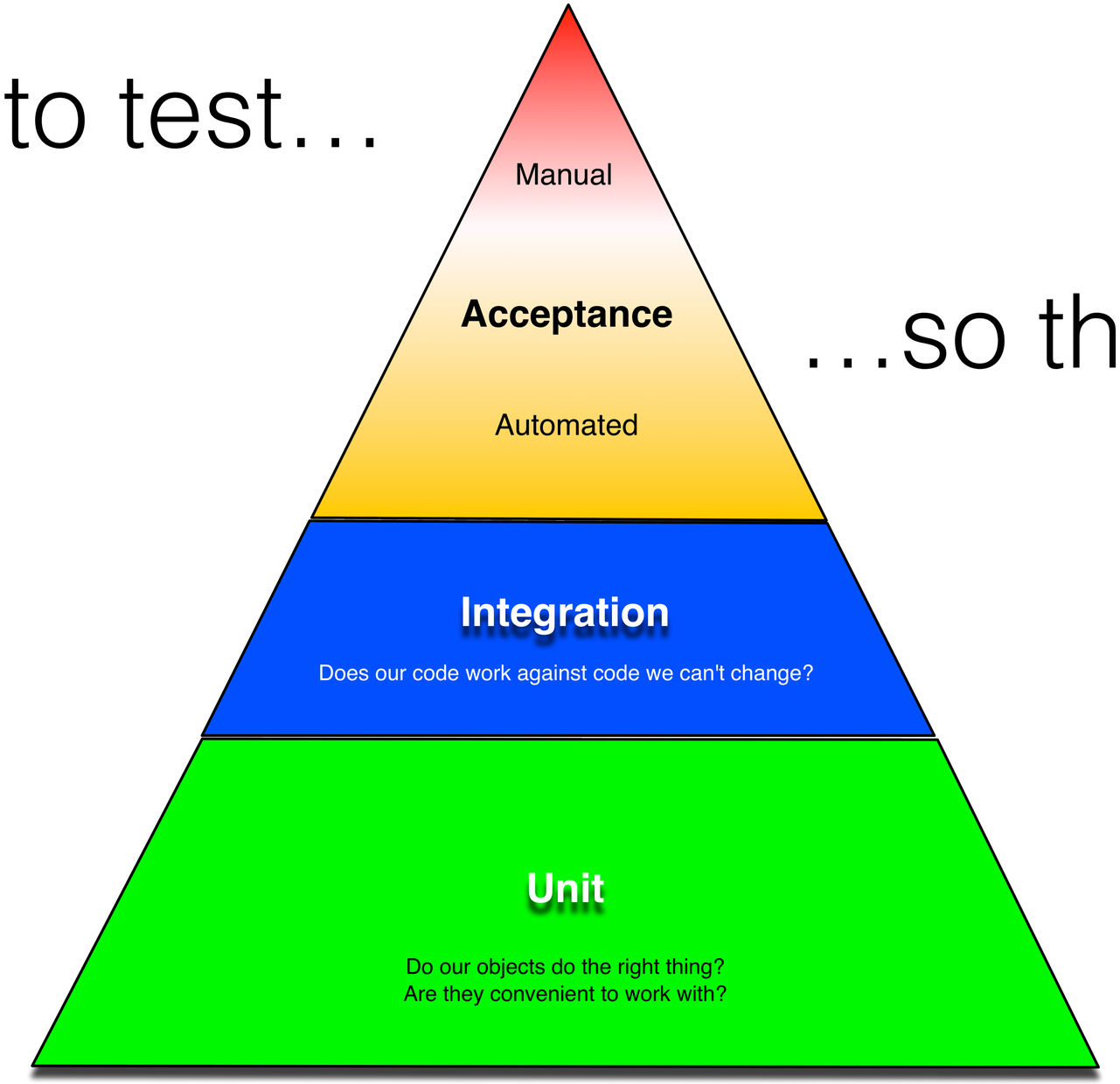
conway's law Embrace it



The basic thesis of this article is that organizations which design systems (in the broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.

--- Melvin Conway

Be easy to test…



…so that you do.

((CENTRIC))

ReST is NOT your friend

Code Reuse is Overrated

Don’t I need Transactions?

Should Client Communication Still be Asynchronous?

((CENTRIC))

We have to think about our data differently

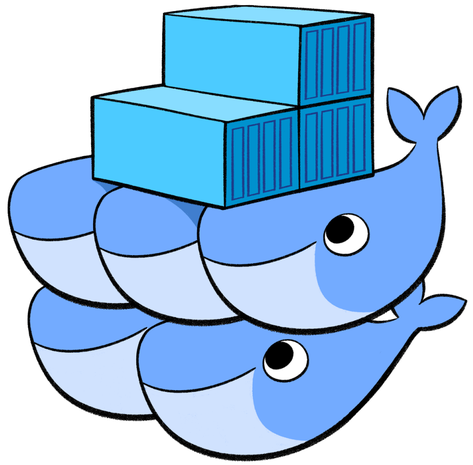
((CENTRIC))

We have to think about our data differently

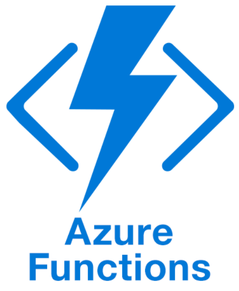
- Taking data from being OFFLINE to ONLINE
- Moved from “data at rest” to “data in motion”

Scaling Model

((CENTRIC))



Scaling Model



Azure
Container Service

((CENTRIC))

Monitoring/Logging/ Troubleshooting

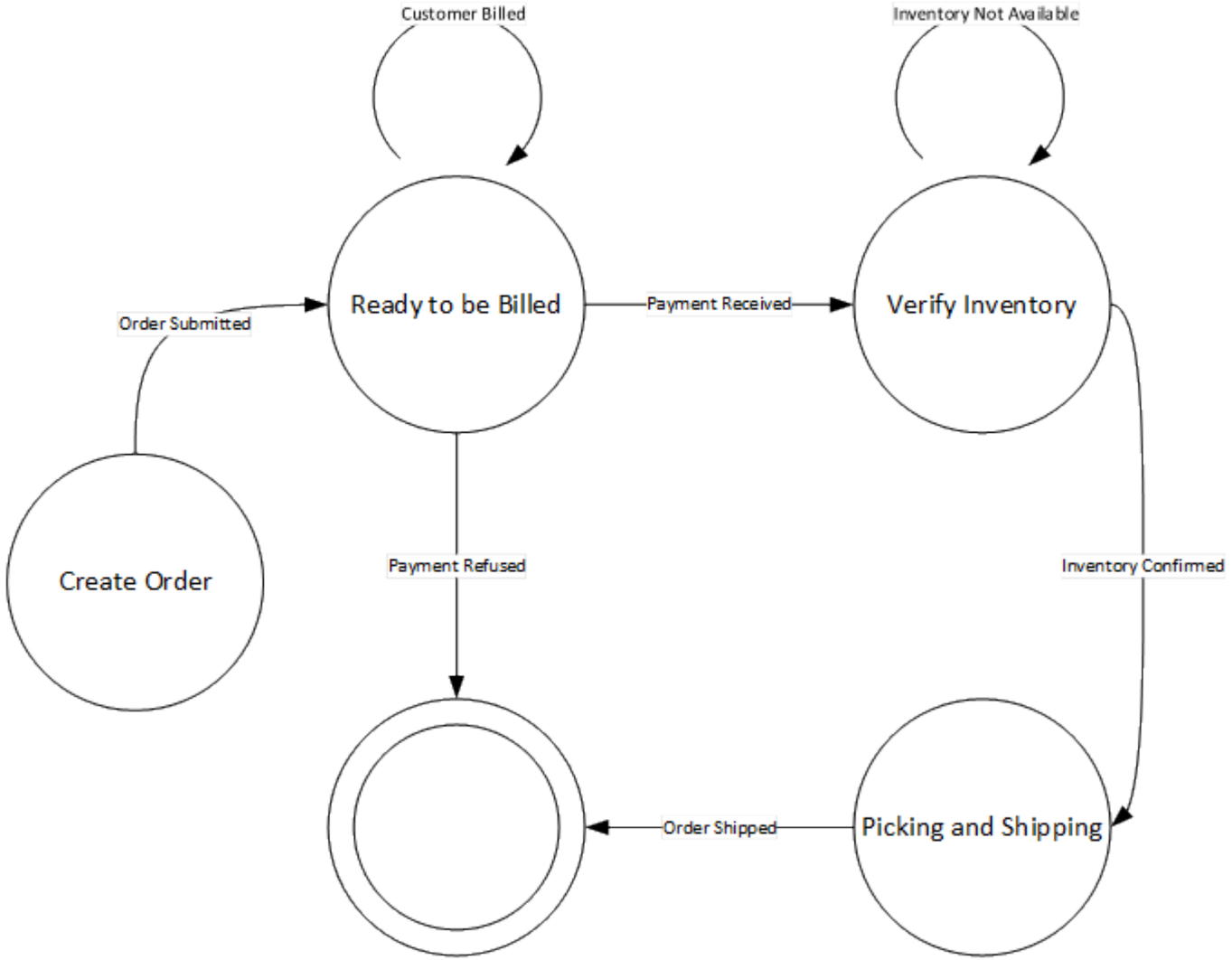
Event Driven Architecture

- An Event is represents something that happens in a domain
  - *Customer Submits Order*
  - *Customer Billed*
  - *Payment Received*
  - *Order Ready for Shipment*
  - *Order Shipped*
- While Events and their payload are designed at the enterprise system level, their implementations are left to the specific subsystems.

Event Driven Architecture

- No point-to-point integrations
- Loosely coupled, highly scalable systems
- Loosely coupled, TEAMS
- Easier to test
- Easier to change
- Technology agnostic

Easily Modeled as a finite state diagram

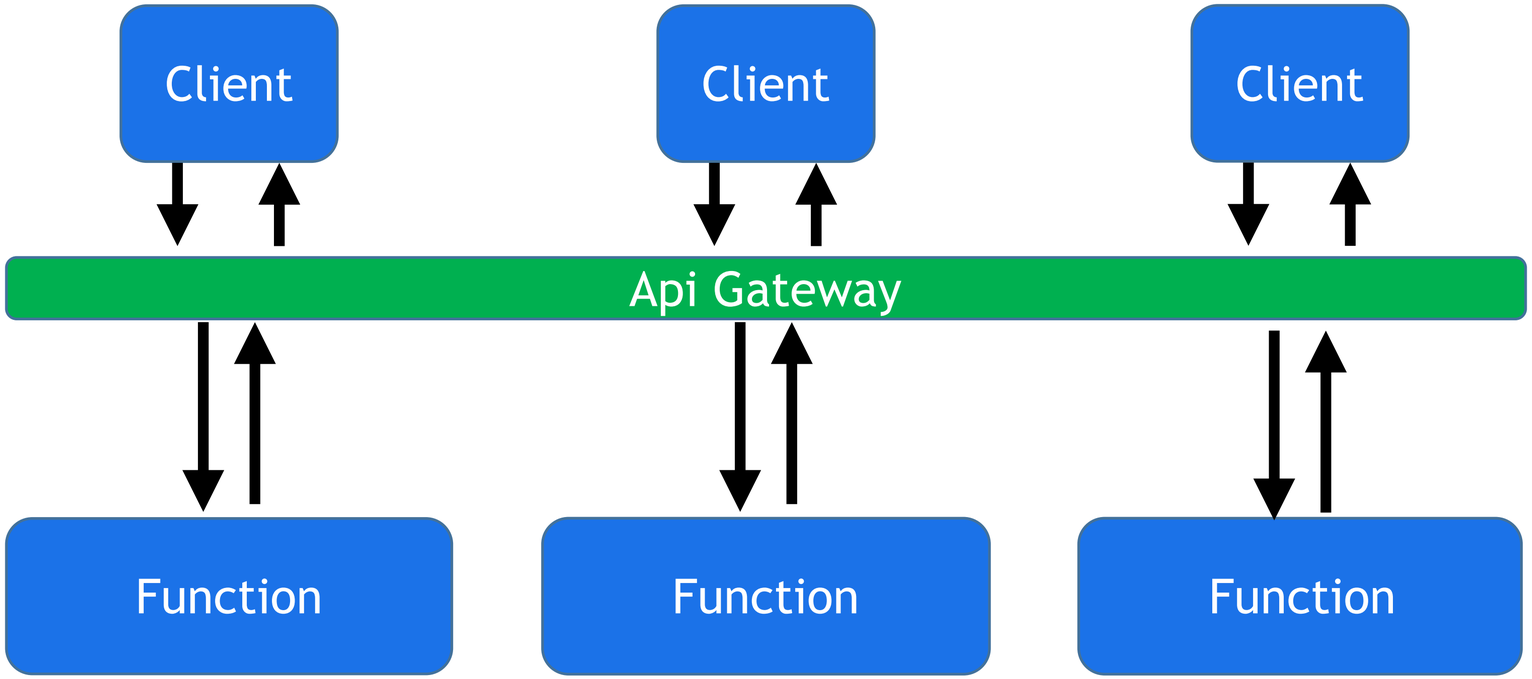


Composition

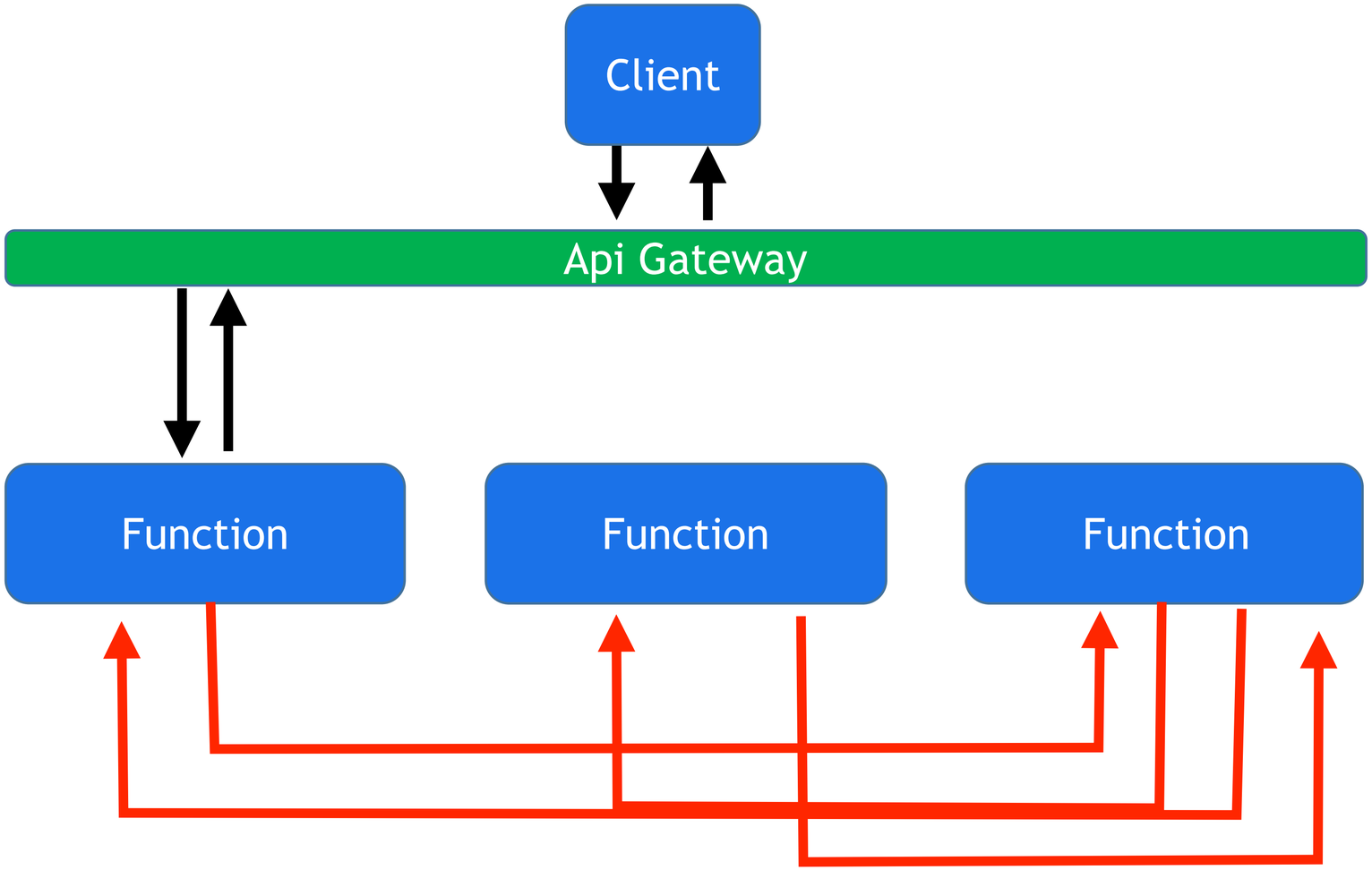
API Gateway

((CENTRIC))

Function as a Service

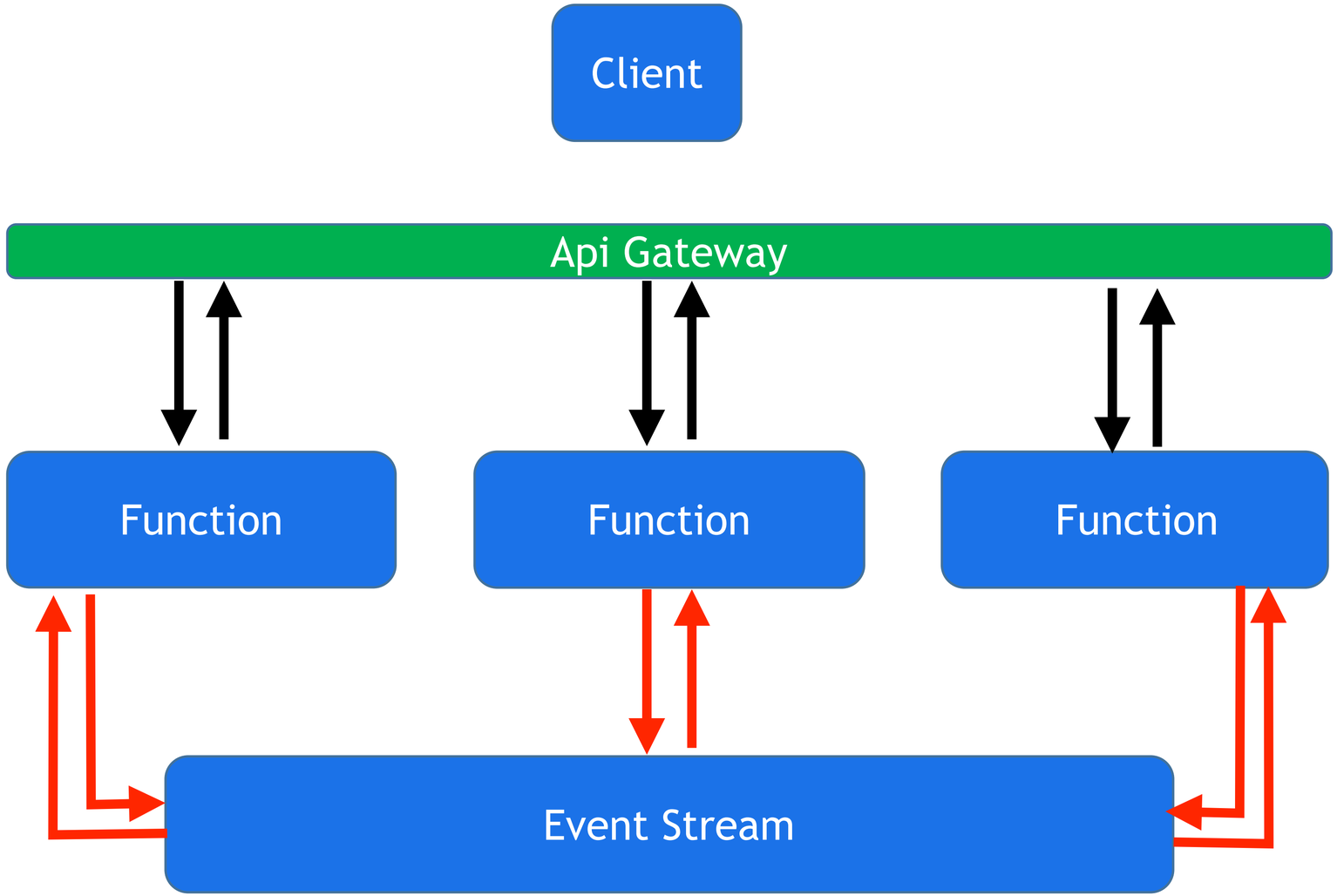


Function as a Service



((CENTRIC))

Function as a Service

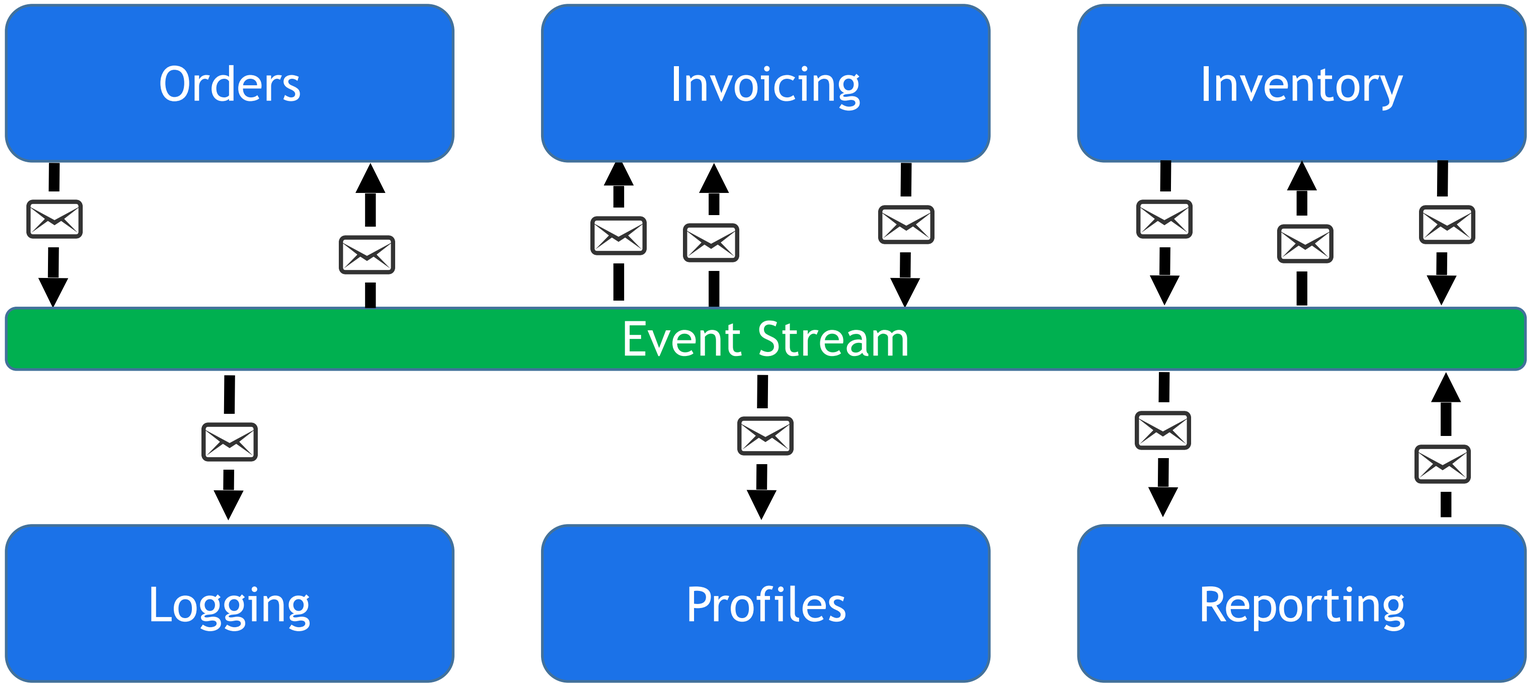


((CENTRIC))

Pub/Sub Event Stream

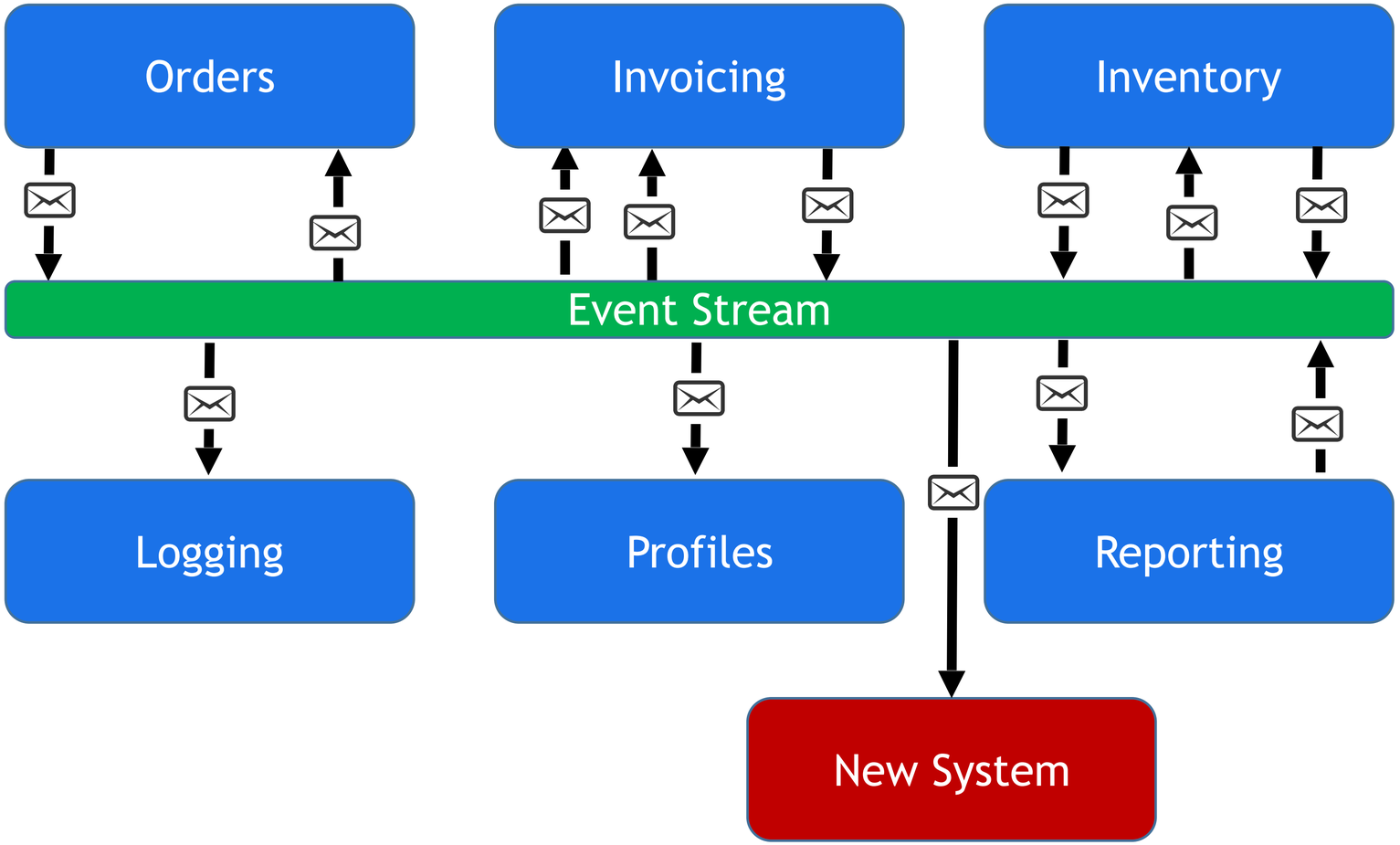
((CENTRIC))

Loosely Coupled



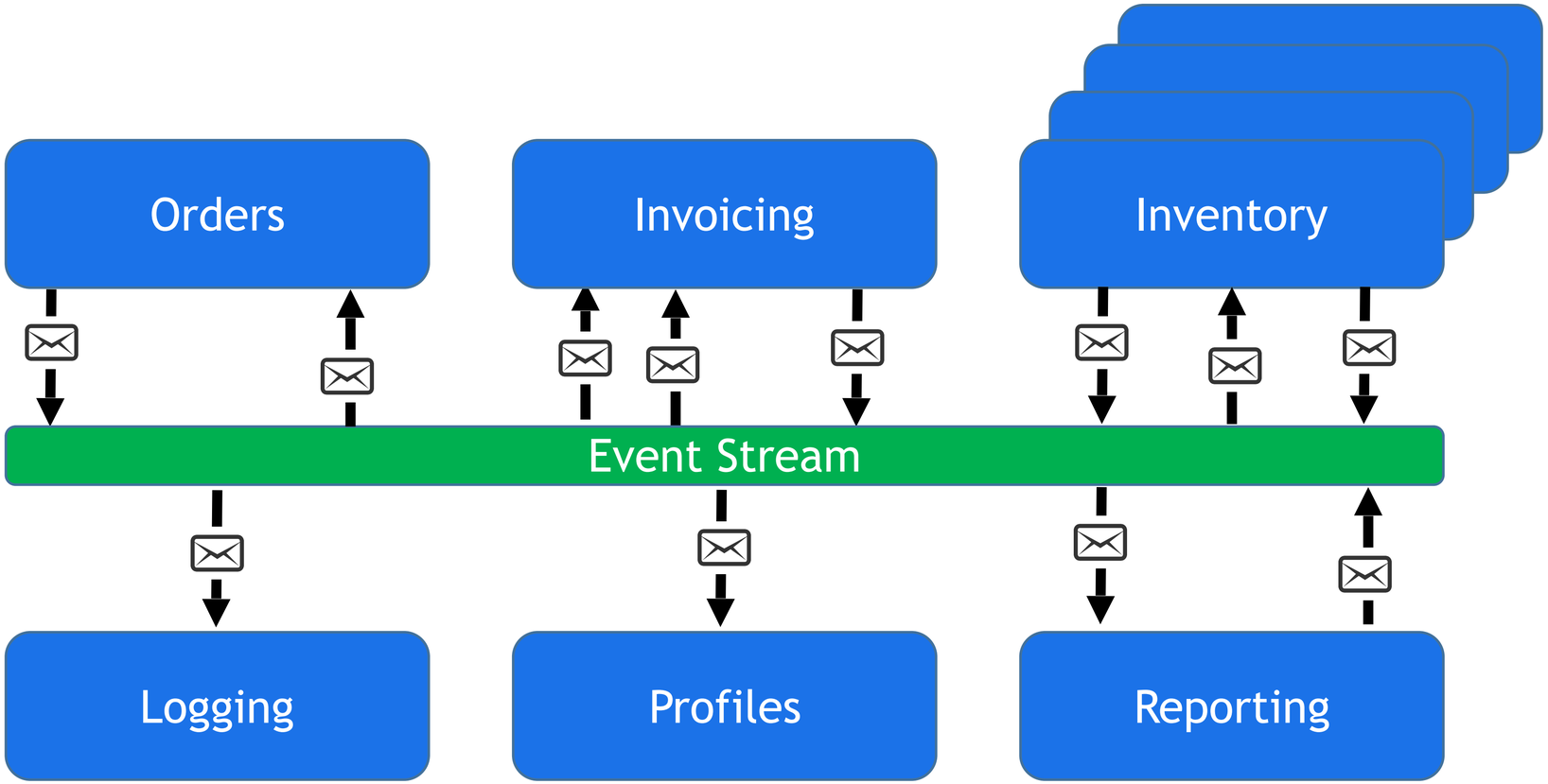
((CENTRIC))

Easy to Integrate New Systems



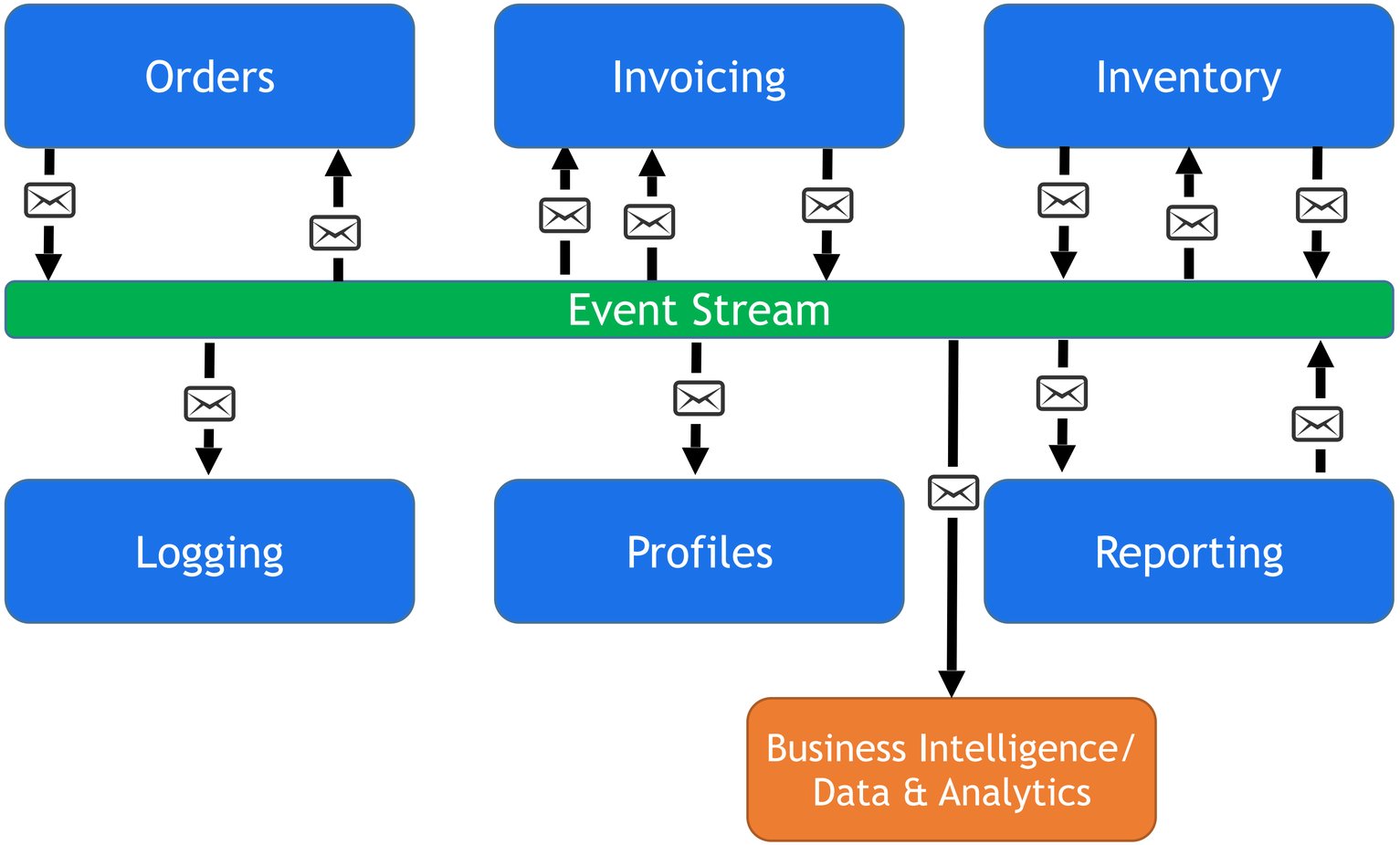
((CENTRIC))

Scales



((CENTRIC))

Addresses “Data in Motion”

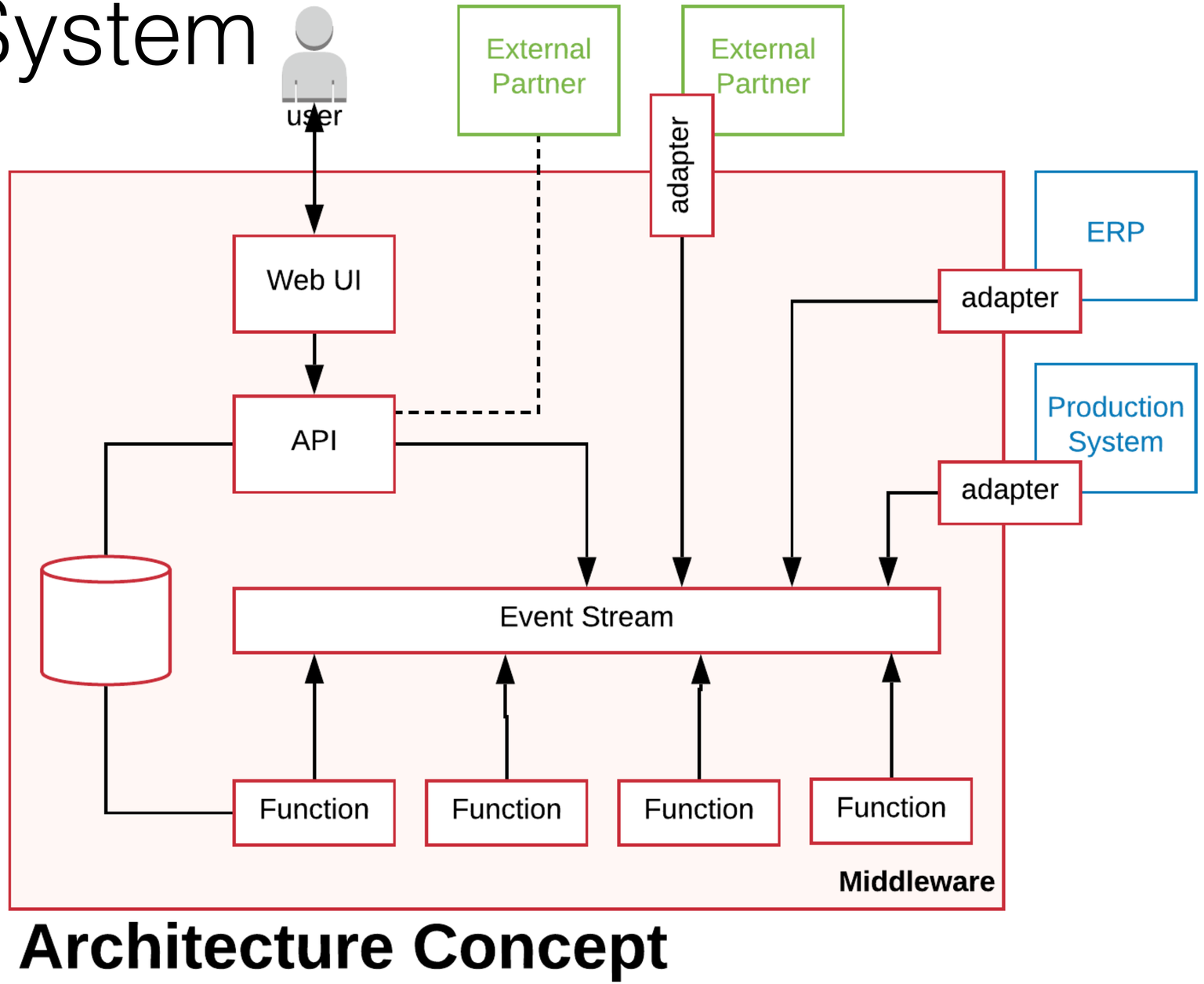


((CENTRIC))

Other benefits

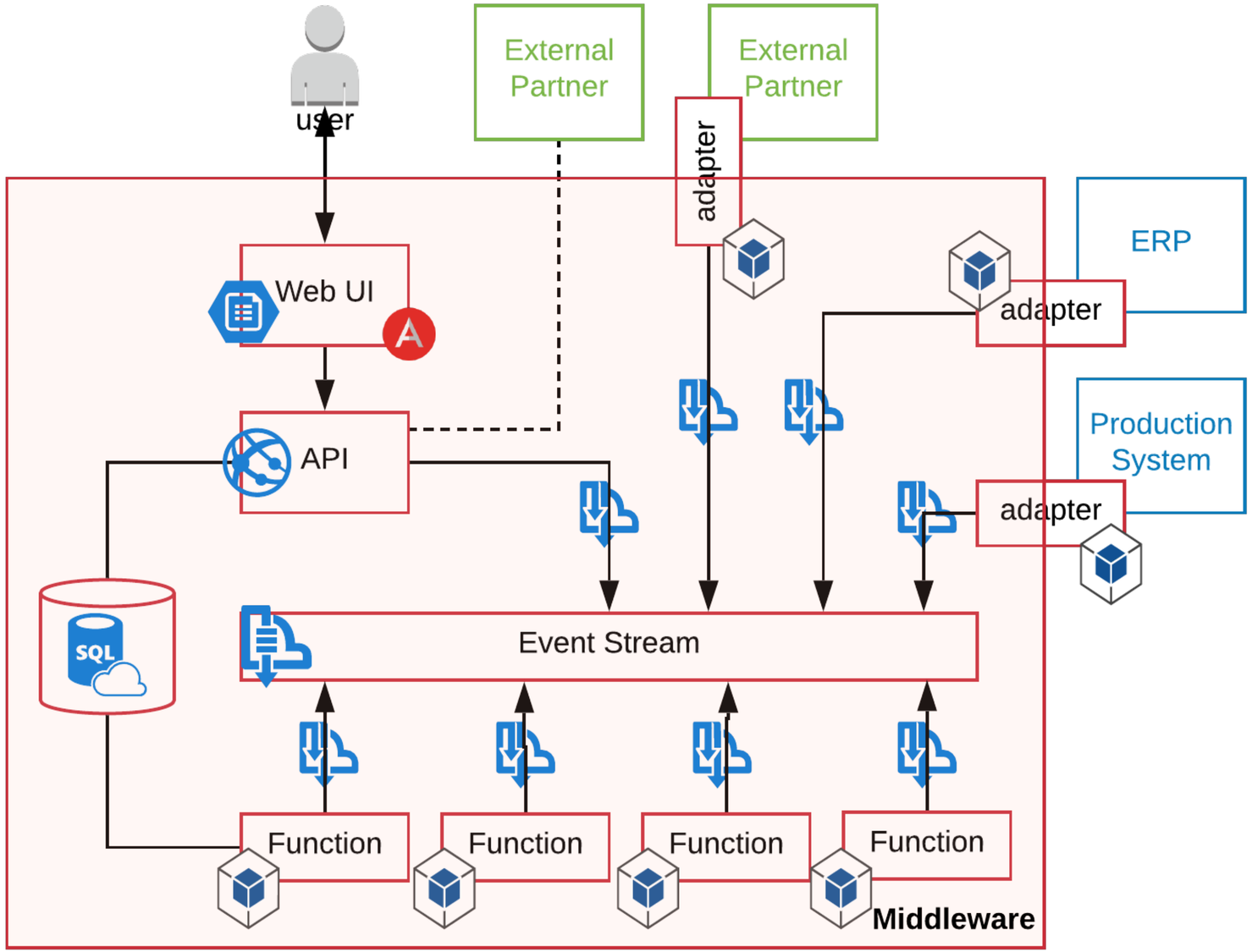
- Ease of 3rd party integration
- Existing systems can be ‘wrapped’
- Can be deliberate about scaling
- Fault tolerant
- Event messages can be logged and ‘replayed’
- Can test subsystems in isolation

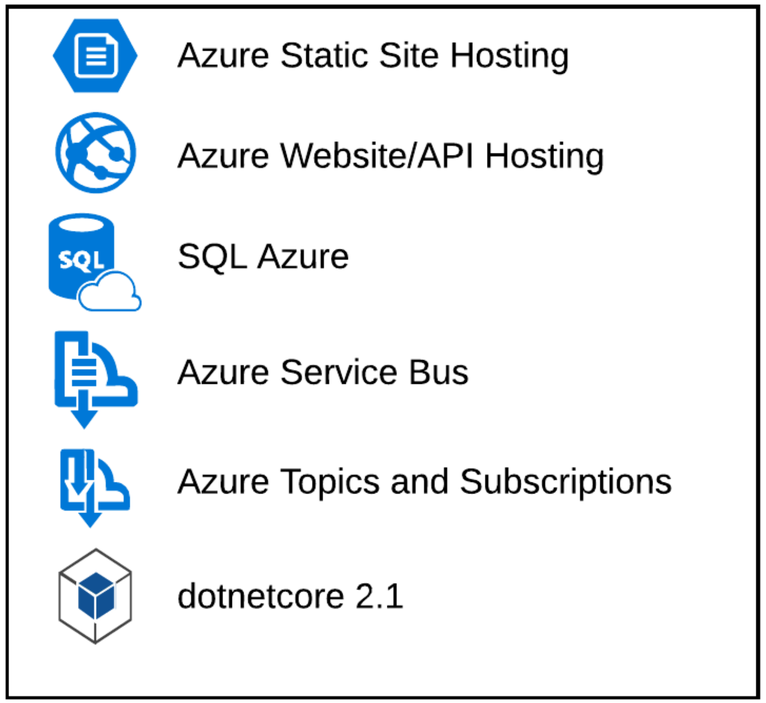
A Retail System



Architecture Concept

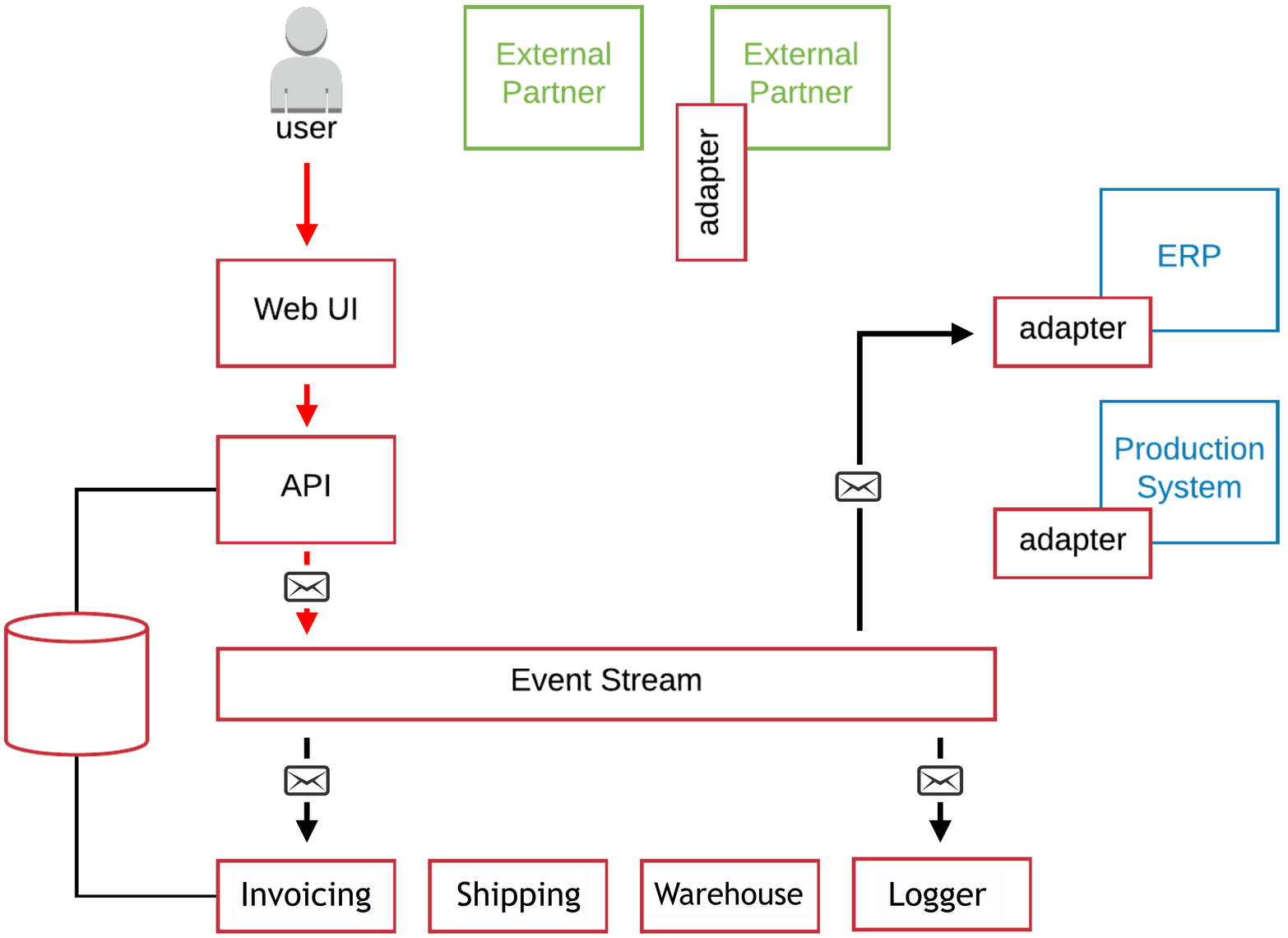
A Retail System



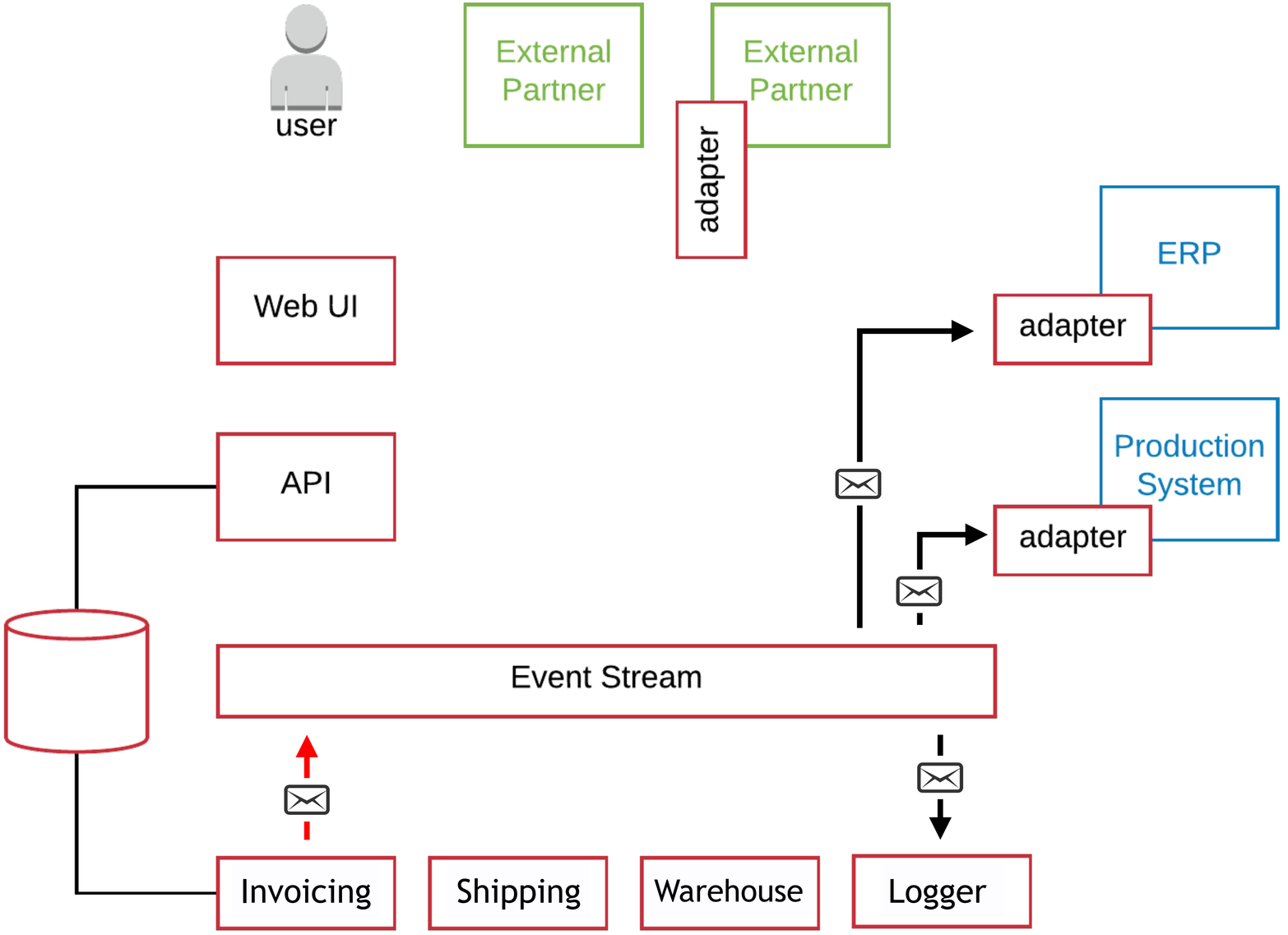


Architecture Concept

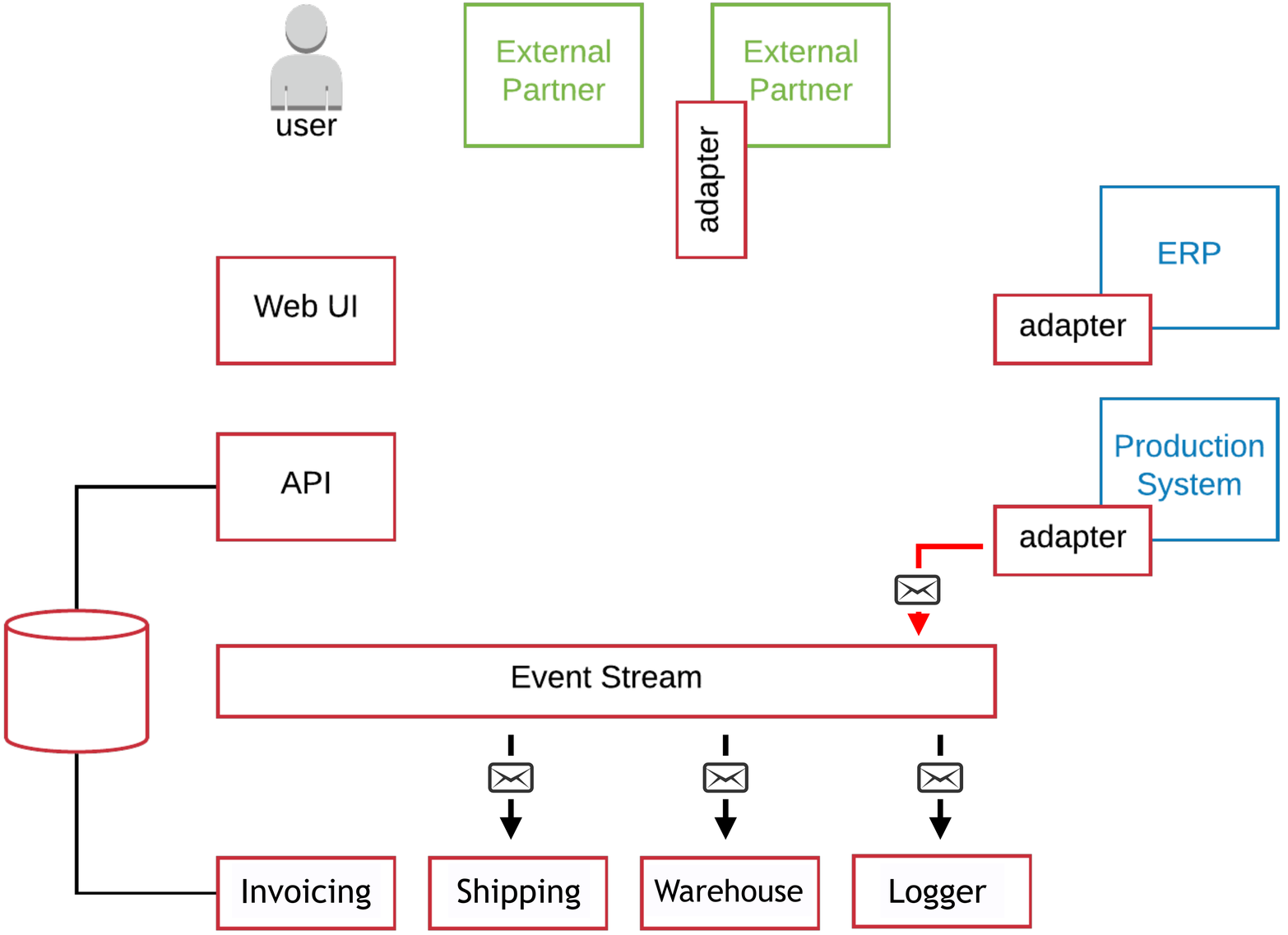
Order Created Event is generated by a User



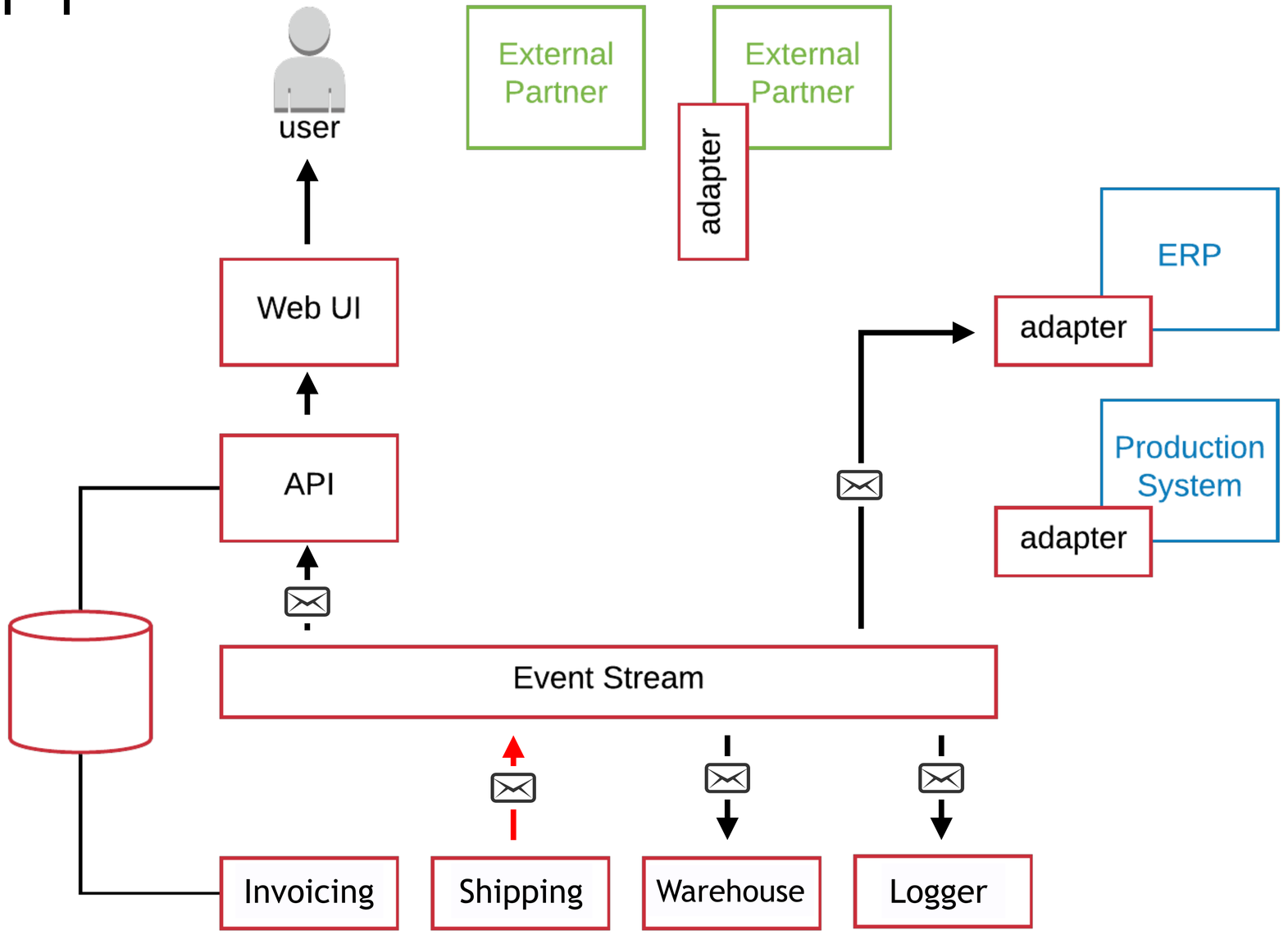
Customer Billed Event



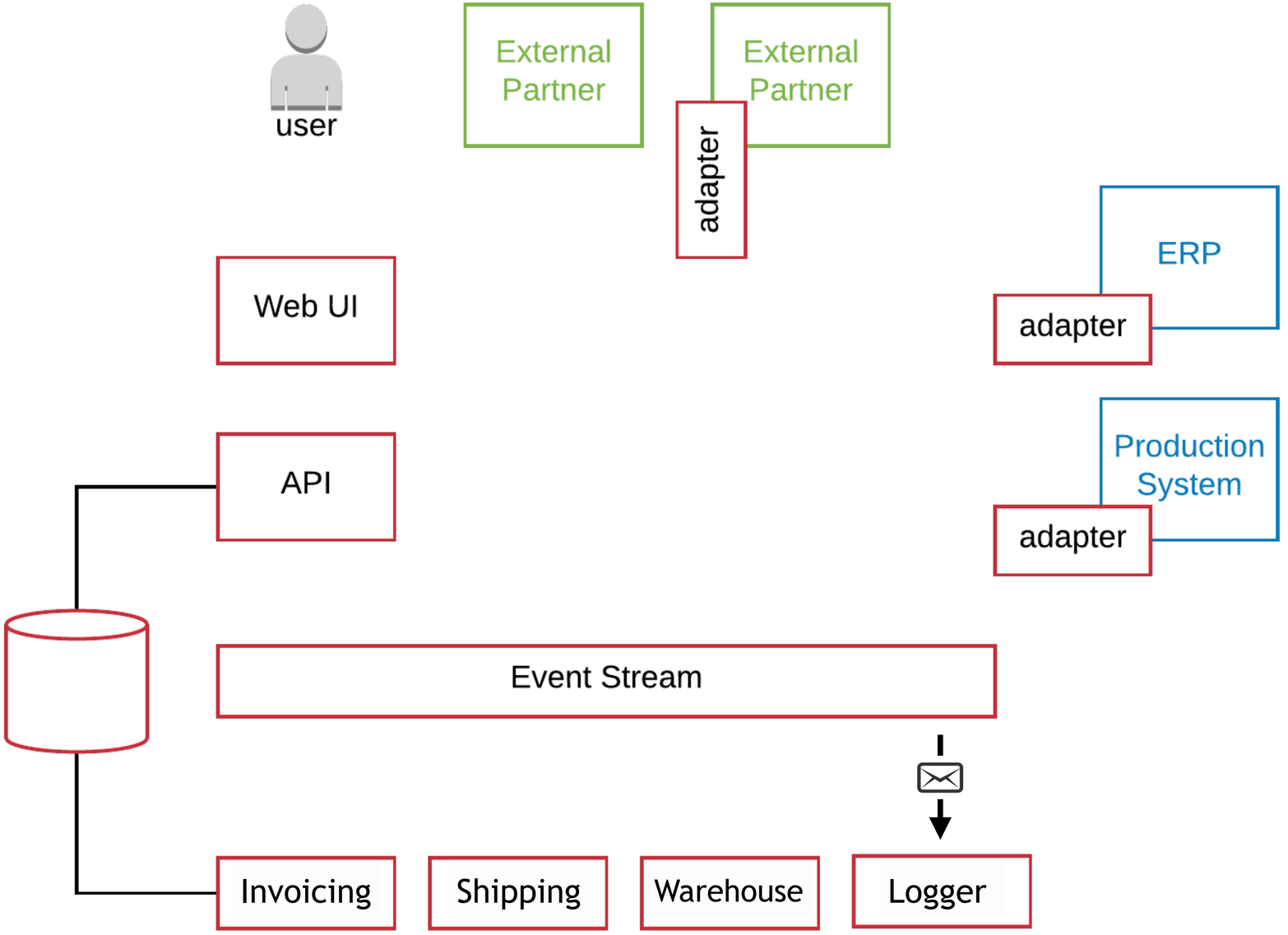
Item Produced Event



Order Shipped Event



The Logger consumes **EVERY** event



Event Sourcing

((CENTRIC))

Event Sourcing

- Events are immutable
- Events describe the action
- Eventually consistent data
- Embraces CQRS

Event Sourcing

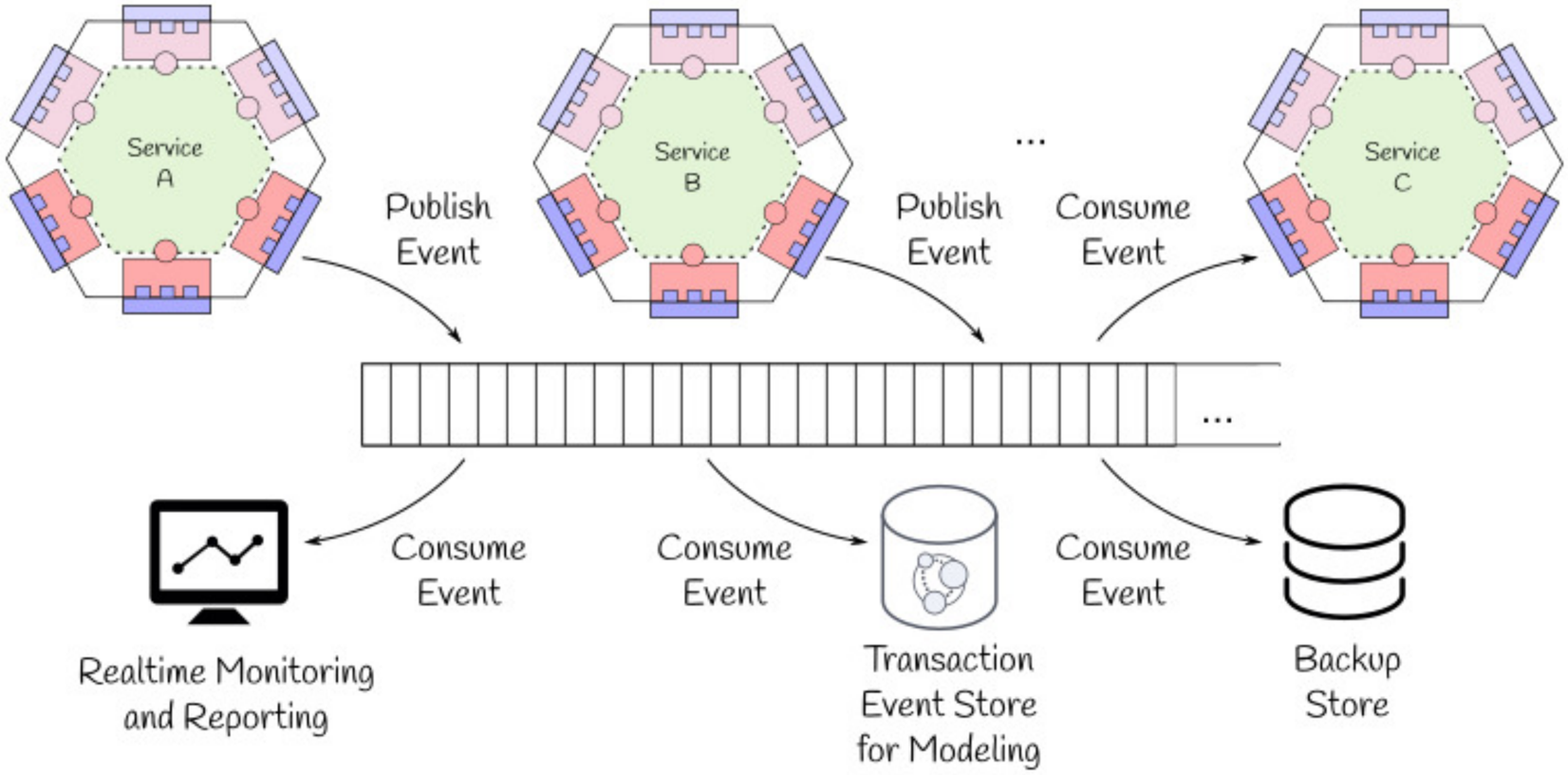


Image via: [Microservices with Clojure]

((CENTRIC))

For Further Reading

- THIS presentation (and code) on GitHub

https://github.com/shawnewallace/eda-rabbitmq

- **Stephan Norberg EDA**

http://www.infoq.com/presentations/Domain-Event-Driven-Architecture

- **“Programming Without a Call Stack”**

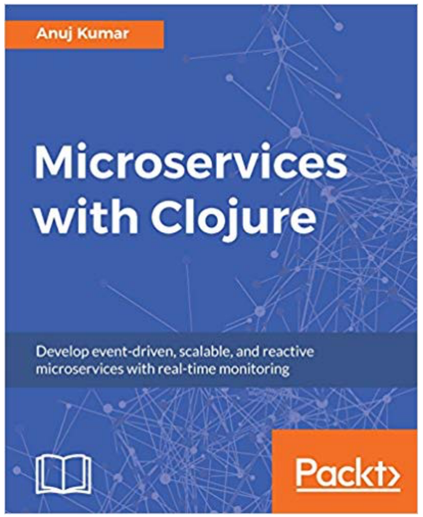
http://www.enterpriseintegrationpatterns.com/docs/EDA.pdf

- **“Event-Driven Architecture Overview”**

http://www.omg.org/soa/Uploaded%20Docs/EDA/bda2-2-06cc.pdf

- **“…Event Sourcing is Hard”**

https://chriskiehl.com/article/event-sourcing-is-hard



((CENTRIC))

Shawn Wallace

((CENTRIC))

www.centricconsulting.com