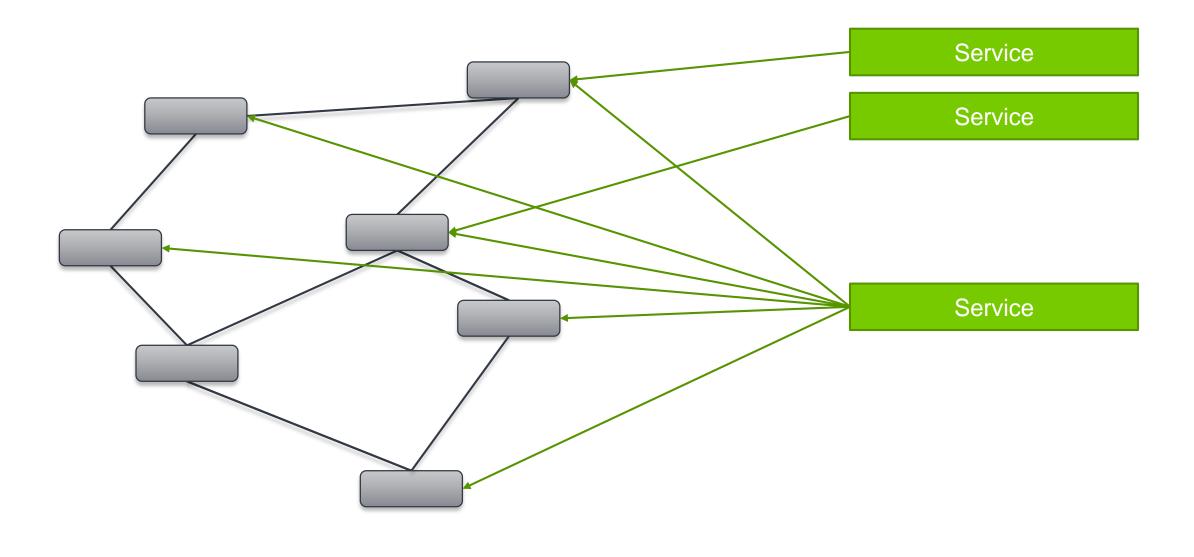
Pragmatic Event-Driven Microservices

Allard Buijze @allardbz - allard@axoniq.io







'Normal' SQL QUERY

CREATE ALGORITHM = UNDEFINED DEFINER = 'dbat'@'%' SQL SECURITY DEFINERVIEW 'BUSINESS FROCESS VIEW DISABLED' AS SELECT 'tpo'. NAME' AS 'TRADING PARTNER NAME', 'bt'. NAME' AS 'TRADING PARTNER NAME', 'tb'. NAME', 'tb'. NAME', 'tb'. NAME', 'tb'. NAME' AS 'TRADING PARTNER NAME', 'tb'. NAME', 'tb'. NAME' AS 'TRADING PARTNER NAME', 'tb'. NAME', 'tb 'rtedid'. 'VALUE' AS 'TRADING PARTNER ROOT EID', 'tpot'. 'ORG SUB TYPE NAME' AS 'TRADING PARTNER CUSTOMER TYPE', 'tpot'. 'STATUS' AS 'TRADING PARTNER TYPE', 'tpot'. 'NAME' AS 'TRADING TYPE', 'Tpot'. 'NAME' AS 'TRADING TYPE', 'Tpot'. 'NAME' AS 'TRADING TYPE', 'Tpot'. 'NAME' AS 'TRAD 'TRADING PARTHER_DUMS', 'bglm'.'VALUE' AS 'TRADING_PARTHER GLN', 'bp'.'BUSINESS_PROCESS' AS 'BUSINESS_PROCESS', (CASE CONCAT(('myo'.'ORGANIZATION_ID' = 'rel'.'PARTY1_ID'), ('myo'.'ORGANIZATION_ID' = 'rel'.'PARTY2_ID'), 'bp'.'PARTY1_DIRECTION') WHEN '10TO' THEN 'TO WHEN '10FROM' THEN 'FROM' WHEN '11TO' THEN 'TO' WHEN '01TO' THEN 'FROM' WHEN '01FROM' THEN 'TO' END) AS 'DOCUMENT DIRECTION', 'bp'. STATUS', 'idt'. 'DOC_TYPE' AS 'INBOUND_TYPE', 'idt'. 'DOC_SUB_TYPE' AS 'INBOUND_SUB_TYPE', AS 'INBOUND_SUB_TYPE 'idt'.'DOC_VERSION' AS 'INBOUND_VERSION', 'iodt'.'DOC_DIRECTION' AS 'INBOUND_DIRECTION', 'odt'.'DOC_DIRECTION', 'odt'.'DOC_DIRECTION' AS 'OUTBOUND_TYPE' AS 'OUTBOUND_SUB_TYPE', 'odt'.'DOC_VERSION' AS 'OUTBOUND_VERSION', 'iodt'.'DOC_DIRECTION' AS 'OUTBOUND DIRECTION', 'bp'.'GO LIVE DT' AS 'GO LIVE DATE', 'bp'.'BUSINESS FROCESS ID' AS 'ID', 'myo'.'ORGANIZATION_ID' AS 'ORG_ID', 'myo'.'NAME' AS 'ORG NAME', 'meid'.'VALUE' AS 'ORG_EID', 'ro'.'ORGANIZATION_ID' AS 'ROOT_ORG_ID', 'ro'.'NAME' AS 'ORG_EID', 'ro'.'NAME' AS 'ORG_EID', 'ro'.'ORGANIZATION_ID' AS 'NOT_ORG_ID', 'ro'.'NAME' AS 'ORG_EID', 'ro'.'NAME' AS 'ORG_EID', 'ro'.'ORGANIZATION_ID' AS 'NOT_ORG_ID', 'ro'.'NAME' AS 'ORG_EID', 'ro'.'NAME' AS 'ORG_ 'ROOT_ORG_NAME', 'xeid'.'VALUE' AS 'ROOT_ORG_EID', 'idt'.'DOC_TYPE_ID' AS 'INBOUND_DOC_TYPE_ID', 'odt'.'DOC_TYPE_ID', 'xell'.'RELATIONSHIP_ID' AS 'RELATIONSHIP_ID', 'oop'.'ISAOS' AS 'OB_ENV_SNDR_QUAL', 'oop'.'ISAOS' AS 'OB_ENV_SNDR_ID', 'oop'.'ISA07' AS 'OB_ENV_RCVR_QUAL', 'oop'.'ISA08' AS 'OB_ENV_RCVR_ID', 'oop'.'GS02' AS 'OB_IN_RCVR_ID', 'oop'.'ISA05' AS 'IB_ENV_SNDR_QUAL', 'iop'.'ISA06' AS 'IB_ENV_SNDR_ID', 'iop'.'ISA07' AS 'IB ENV RCVR QUAL', 'iop'.'ISAO8' AS 'IB ENV RCVR ID', 'iop'.'GSO2' AS 'IB IN SNDR ID', 'iop'.'GSO3' AS 'IB IN RCVR ID', 'bpop'.'IB OVR ENV RCVR QUAL', 'bpop'.'IB OVR ENV RCVR QUAL', 'bpop'.'IB OVR ENV RCVR ID', 'bpop'.'IB OVR ENV RCVR QUAL', 'bpop'.'IB OVR ENV RCVR ID', 'bpop'.'IB OVR ENV RCVR QUAL', 'bpop'.'IB OVR ENV RCVR ID', 'bpop'.'IB OVR ENV RCVR ID', 'bpop'.'IB OVR ENV RCVR QUAL', 'bpop'.'IB OVR ENV RCVR ID', ' 'bpop'.'IB_OVR_ENV_SNDR_QUAL' AS 'IB_OVR_ENV_SNDR_QUAL', 'bpop'.'IB_OVR_ENV_SNDR_ID' AS 'IB_OVR_ENV_SNDR_ID' AS 'IB_OVR_ENV_SN 'bpop'.'OB_OVR_ENV_RCVR_QUAL' AS 'OB_OVR_ENV_RCVR_QUAL', 'bpop'.'OB_OVR_ENV_RCVR_ID', 'bpop'.'OB_OVR_ENV_SNDR_QUAL', 'bpop'. AS 'OB_OVR_INNER_RCVR_ID', 'bpop'.'OB_OVR_INNER_E B_TRX_ENV_SNDR_QUAL' AS 'IB_TRX_ENV_SNDR_QUAL', 'bpop'.'IB_TRX_ENV_SNDR_ID' AS 'IB_TRX_ENV_SNDR_ "IB TRX INNER E OB TRK ENV RCVR OUAL', 'bpop'.'OB_TRX_ENV_RCVR_ID' AS 'OB_TRX_ENV_RCVR TRX ENV SNDR I INNER RCVR ID', 'bpop'.'OB_TRX_INNER_ENV_SNDR_ID' AS 'OB_TRX_IN LATIONSHIP' 'rel' ON (('rel'.'RELATIONSHIP_ID' = 'bp', 'RELATIONSHIP ID'))) LEFT JOIN 'ORGANIZA' ORGANIZATION I ID' = 'ro'. 'ORGANIZATION ID'))) LEFT JOIN NIZATION ID'))) LEFT JOIN 'BOARDING TYPE' 'bt' 22 JOINS 6 SUBQUERIES ORGANIZATION ID 'rto'.'ORGANIZATION_ID') AND ON (('bta'.'BOARDING_TYPE_ID' = 'bt'.'BOARDING ('rteid'.'PRIMARY FLAG' = 1) AND ('rteid'.'ORG TYPE ID' FROM 'D)))) LEFT JOIN 'ORG TYPE ASSOC' 'tpoa' ON (('tpoa'.'ORGANIZATION_ID' = 'tpo'.'ORGANIZATIO oa'.'ORG_TYPE_ID myo'.'ORGANIZATION_ID') AND

"ET THE". 'NORG IDENTIFIER 'Ye' IN AND ('tag'. 'ORGANIZATION ID' = 'rel'. 'PARTY1 D') AND ('tag'. 'tag'. 'tag'

LE ID, ELOW , DIW

)) LEFT JOIN 'ORG IDENTIFIER' 'teid' ON

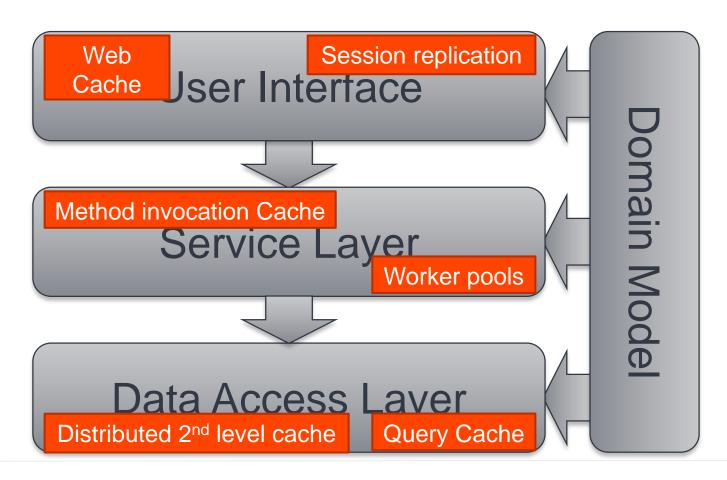
R_TYPE' WHERE ('DIM_ORG_IDENTIFIER_TYPE'.'NAME'



('meid'.'PRIMARY_FLAG' = 1) AND ('meid'.'ORG_ID

((('teid'.'ORGANIZATION_ID' = 'tpo'.'ORGANIZATION_ID' = 'tpo'.'ORGANIZATION'ID' = 'tpo'.'ORGANIZ

Layered architecture















Source: http://www.sabisabi.com/images/DungBeetle-on-dung.JPG

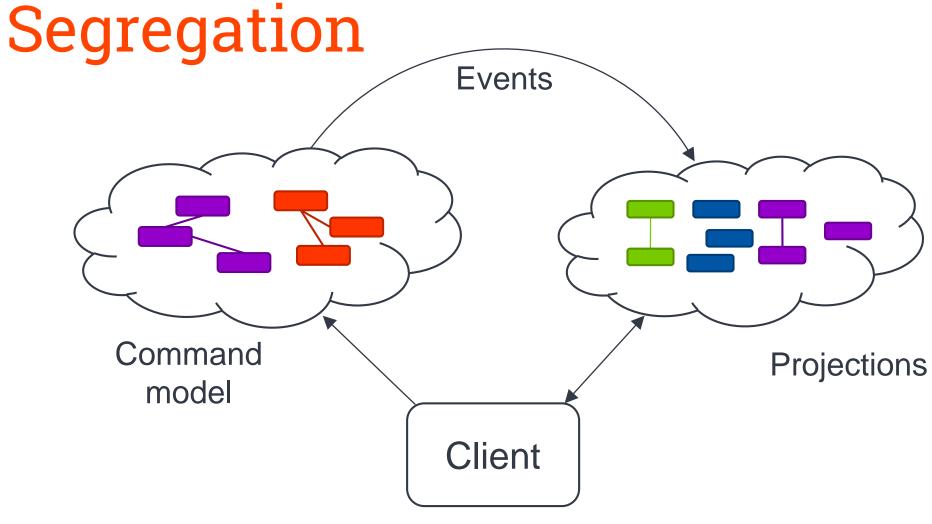








Command Query Responsibility





Monoliths







St Breock Downs Monolith - www.cornwalls.co.uk



Microservices vs Monoliths

Monoliths

Almost all the successful microservice stories have started with a monolith that got too big and was broken up

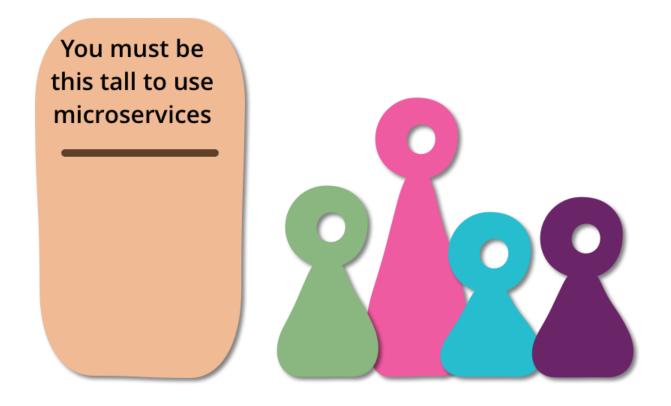
Microservices system

Almost all the cases where I've heard of a system that was built as a microservice system from scratch, it has ended up in serious trouble.

Martin Fowler



Are you tall enough?





Noun? -> Service!



OrderService



CustomerService



ProductService

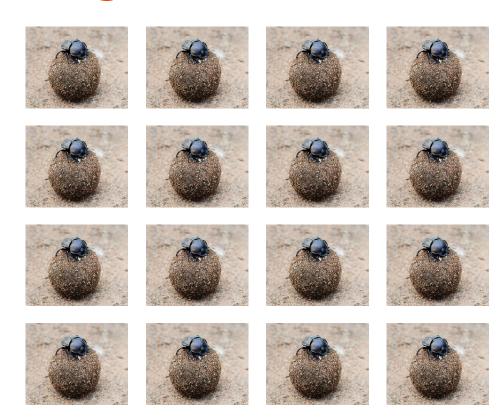


InventoryService

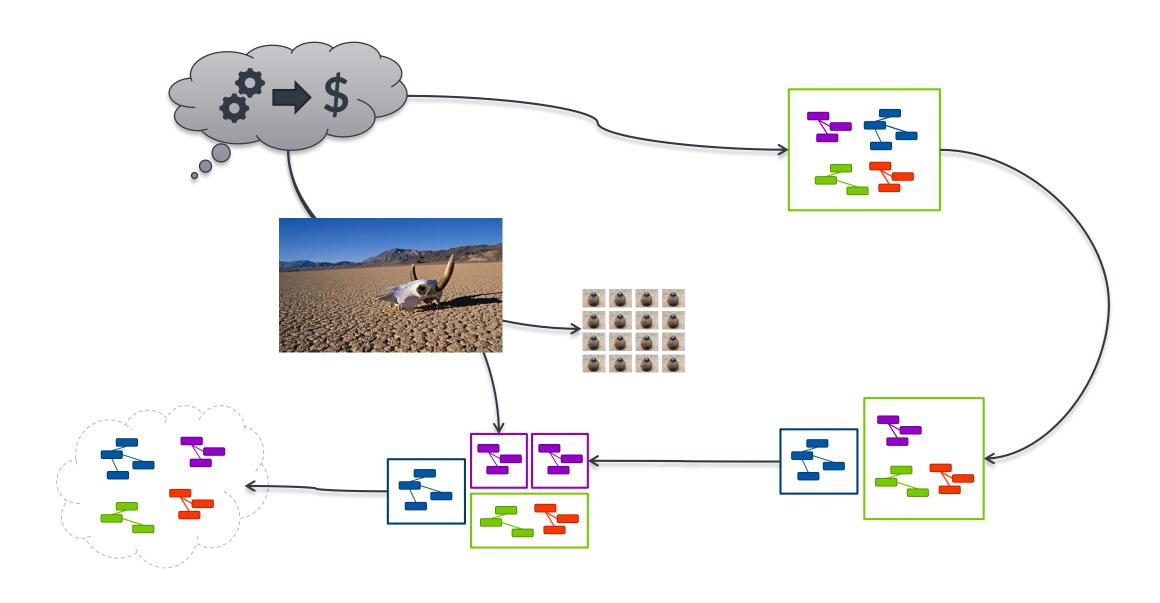






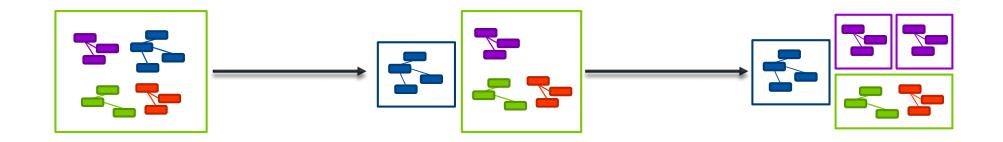








Location transparency



A component should neither be aware of nor make any assumptions about the location of components it interacts with.

Location transparency starts with good API design (but doesn't end there)

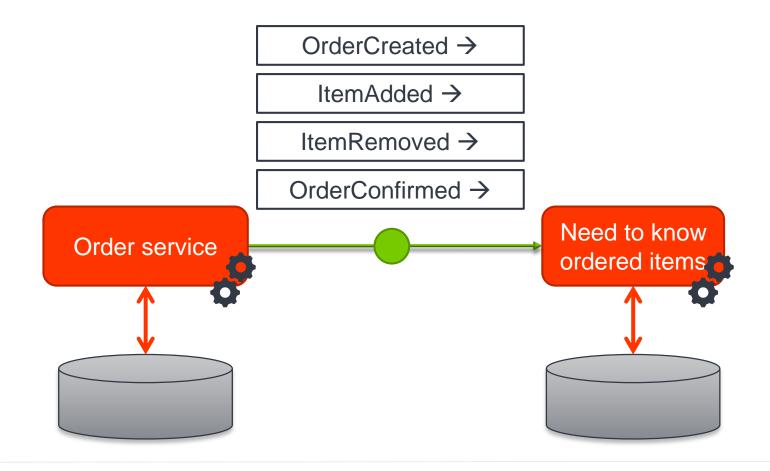


'Event-Driven' Microservices

Event Something has happened

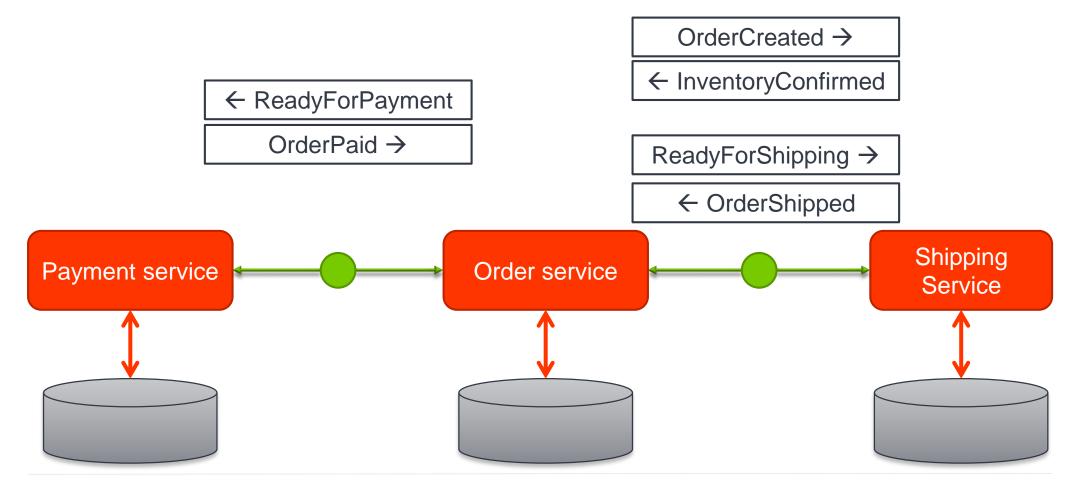


'Event-Driven' Microservices





Or worse...





Reasons to send a message

Event Something has happened

Command I want the system to do something

Query I want to know something

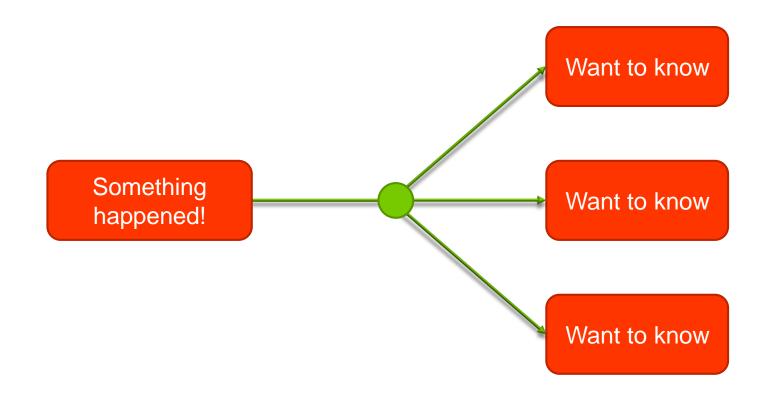


Something has happened - Event

- Data change
- Deadline passing
- Or anything else that's relevant in the domain

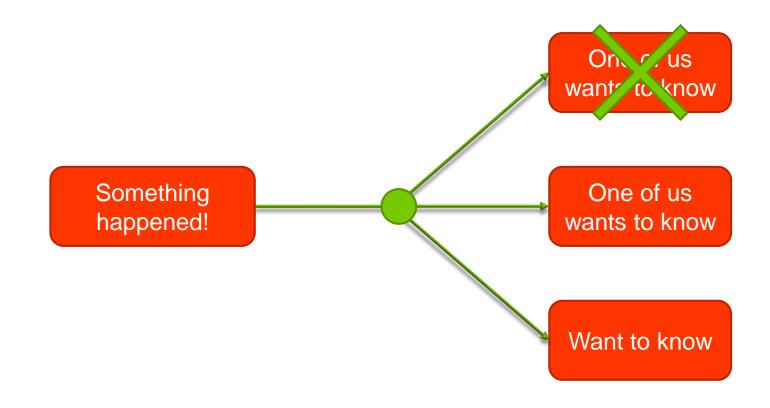


Publish-subscribe



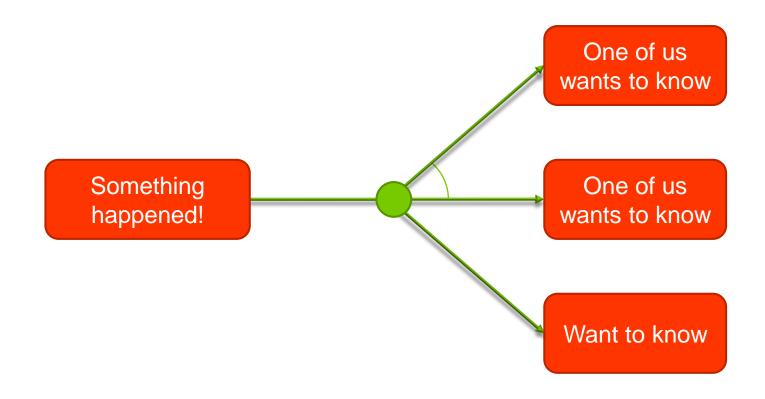


Exclusive consumers



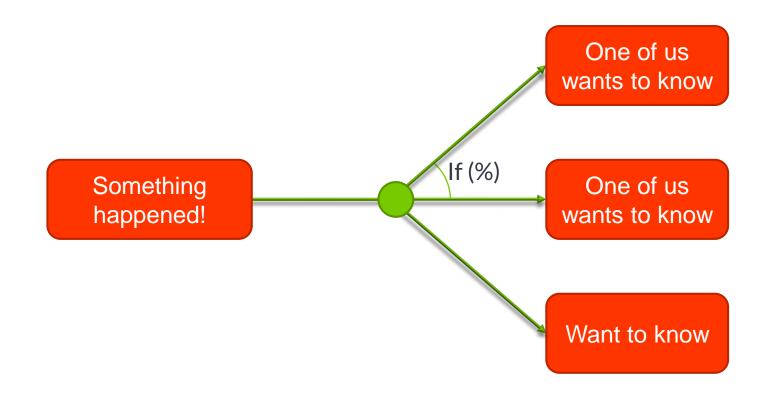


Competing consumers





Balanced consumers



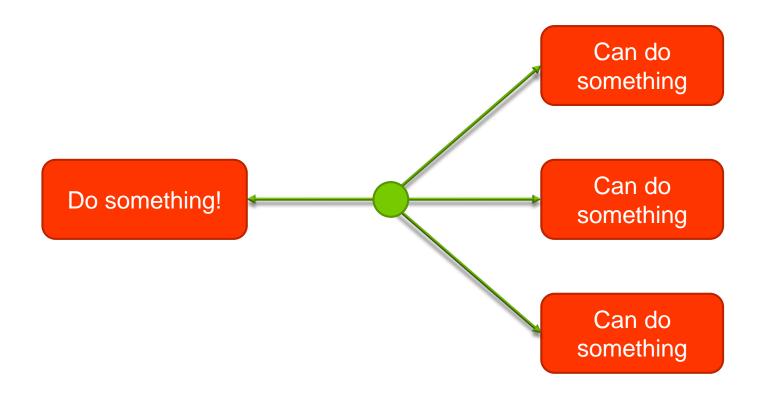


I want something done - Command

- Request-a-side-effect
 - Change data / application state
 - Send email
- Exactly 1 destination
- OK / NOK reply
 - Maybe some data

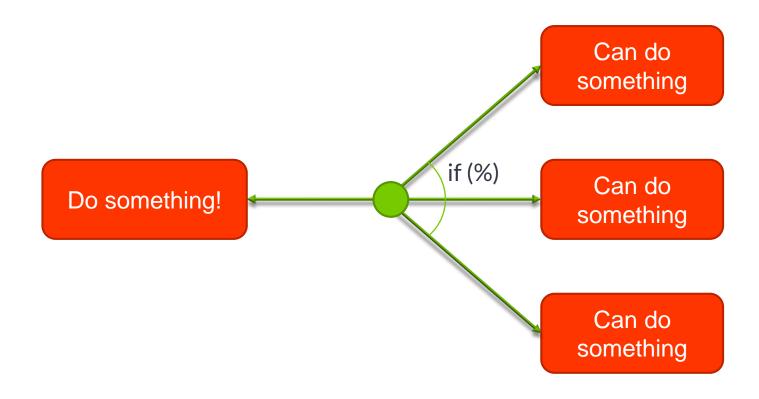


Command Routing





Command Routing



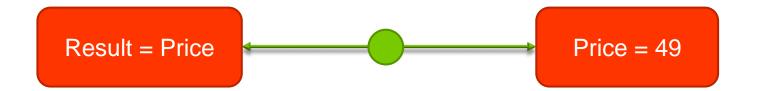


I want to know something -Query

- Desire for information
 - The response has more value than the question
 - (Usually) side-effect free
- Different messaging patterns
 - Single destination
 - Scatter gather query
 - Subscription

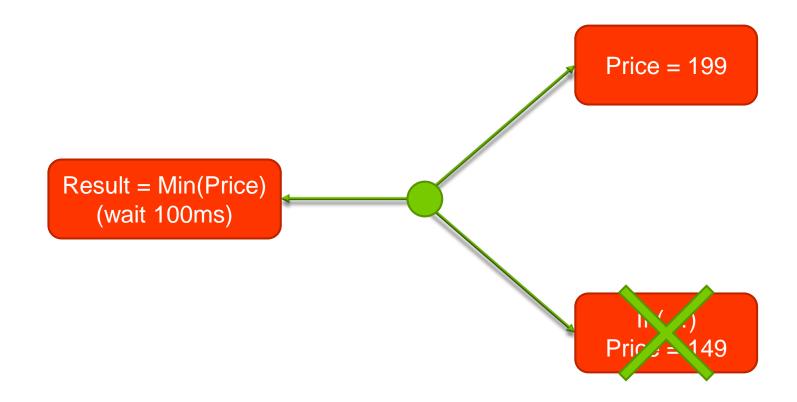


Query – point to point



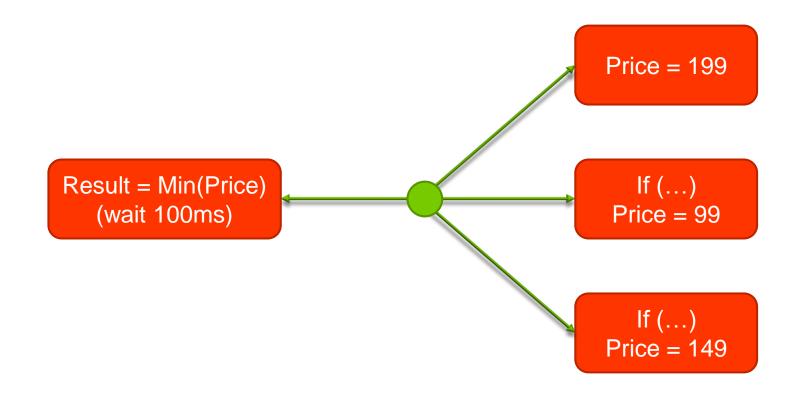


Query – scatter-gather



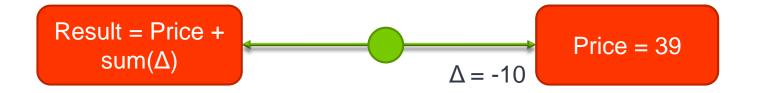


Query – scatter-gather



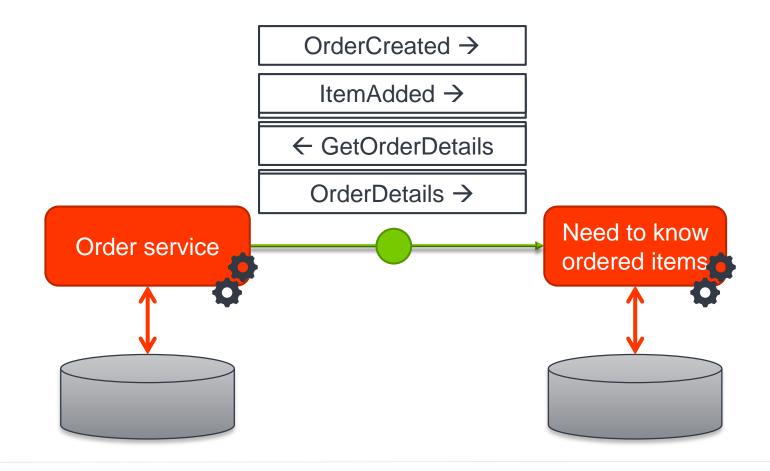


Query – subscription





'Event-Driven' Microservices





Events retain value

Event Sourcing is an Architectural pattern in which Events are considered the "source of truth", based on which components (re)build their internal state.

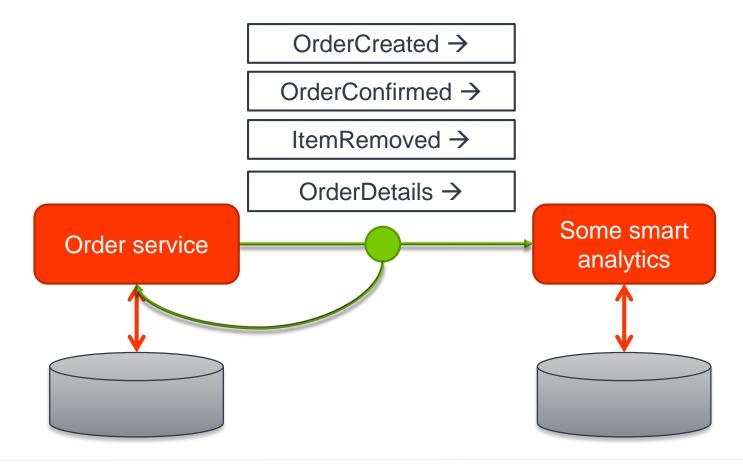


Event Store

An Event Store stores the published events to be retrieved both by consumers as well as the publishing component itself.



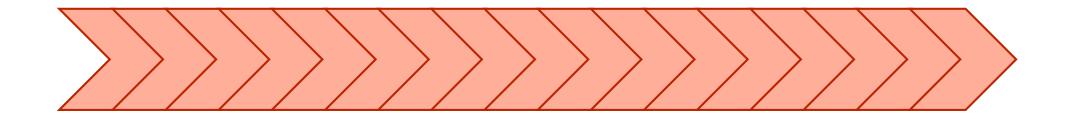
Event Sourcing





Event Store operations

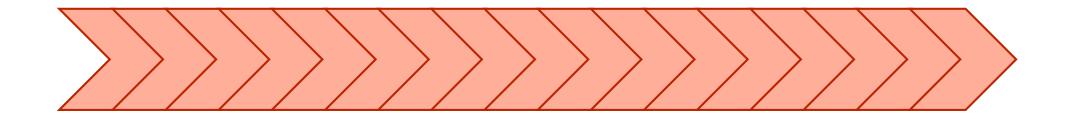
- Append
- Validate 'sequence'





Event Store operations

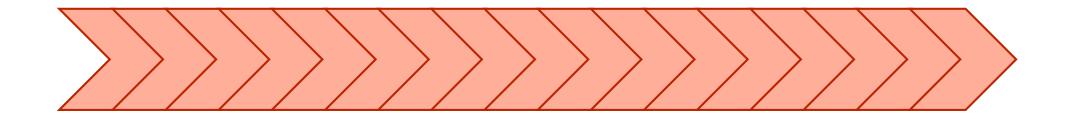
Full sequential read





Event Store operations

Read aggregate's events





1. Define which routing patterns to apply

2. Choose technology/protocol accordingly

















HTTP/2



At scale, different rules apply



How do you route all these events to all components?

How will this scale?



You Don't!

It Won't!



Unmanageable mess

Order Created
Item Added to Order
Shipping Address Added
Billing Address Added
Order Confirmed



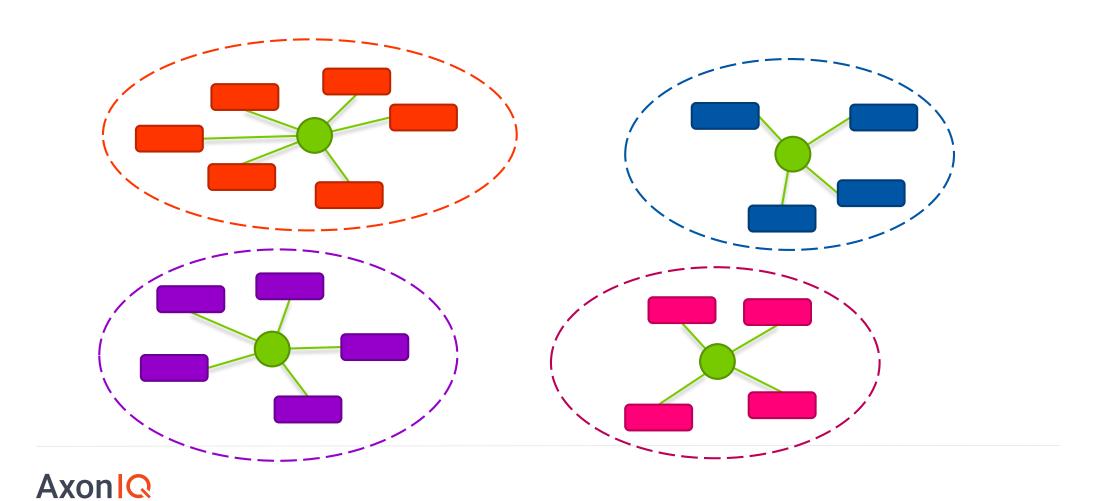


Bounded context

Explicitly define the context within which a model applies. Explicitly set boundaries in terms of <u>team organization</u>, usage within specific parts of the application, and <u>physical</u> <u>manifestations</u> such as code bases and database schemas. Keep the <u>model strictly consistent</u> within these bounds, but don't be <u>distracted or confused by issues outside</u>.



Within a context, share 'everything'



Between contexts, share 'consciously' As shipping module, I want to know when an order is placed Order Created + Item Added + Order Confirmed → Order Placed **Order Created** Item Added to Order Shipping Address Added Billing Address Added Order Confirmed



Between contexts, share 'consciously' As shipping module, I want to know when an order is placed Order Confirmed + Get Order Details → Order Placed **Order Created** Item Added to Order Shipping Address Added Billing Address Added Order Confirmed

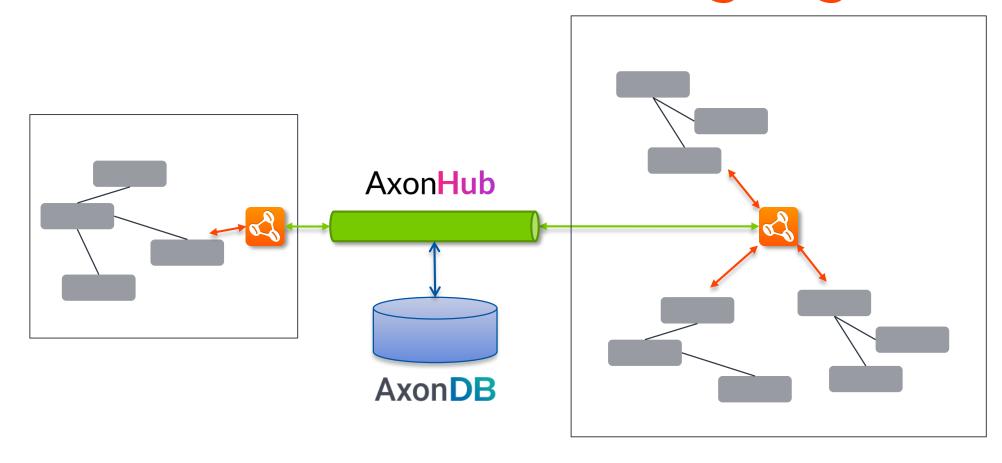


Where does AxonFramework fit?

- Inside each component in bounded context
- Axon provides the Java APIs towards platform
 - EventBus, CommandBus, QueryBus
- Separation of business logic and infrastructure logic



Microservices Messaging





"Just enough" intelligence

AxonHub

Message Broker

Sends messages. Main value add is reliability.

Understands difference between Commands, Events, Queries and their routing patterns. Does not care about the content of these messages.

Enterprise Service Bus

Understands message content. Hard to configure and maintain.

dumb

smart



Our mission

Provide the APIs and implementations necessary for eventdriven microservices to cooperate harmoniously, allowing each of them to focus on the business logic.

