CADEC 2017 - DDD & MICROSERVICES

STORA FÖRDELAR MED SMÅ TJÄNSTER

ANDREAS TELL

2017-01-25 | CALLISTAENTERPRISE.SE

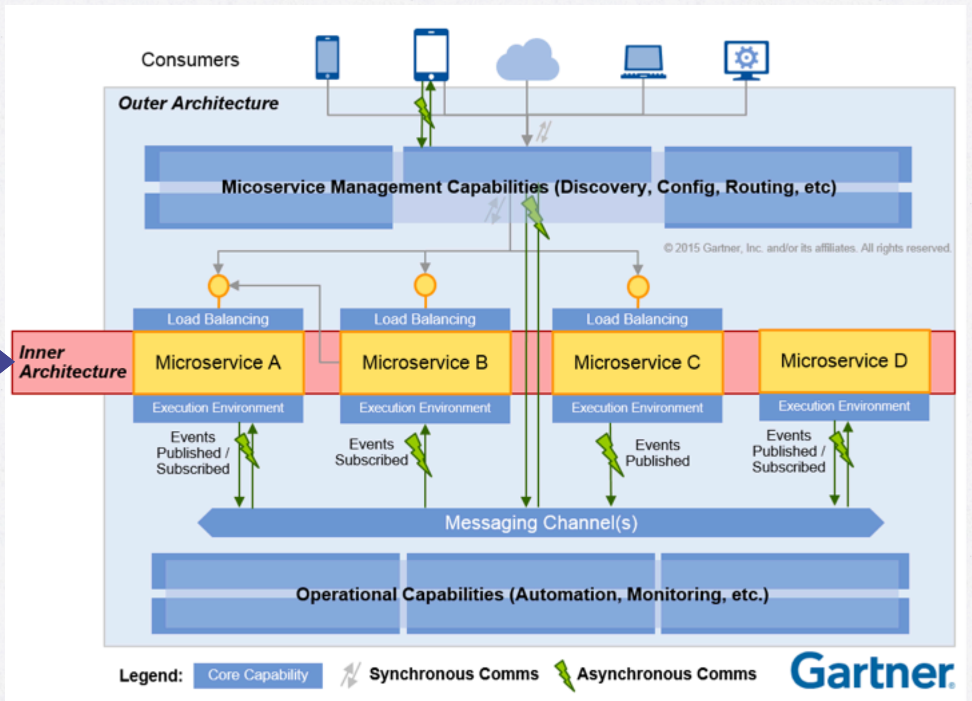
META PRESENTATION

In this talk:

- Brief intro to main concepts
- Rationales
- Useful DDD concepts
- Migration

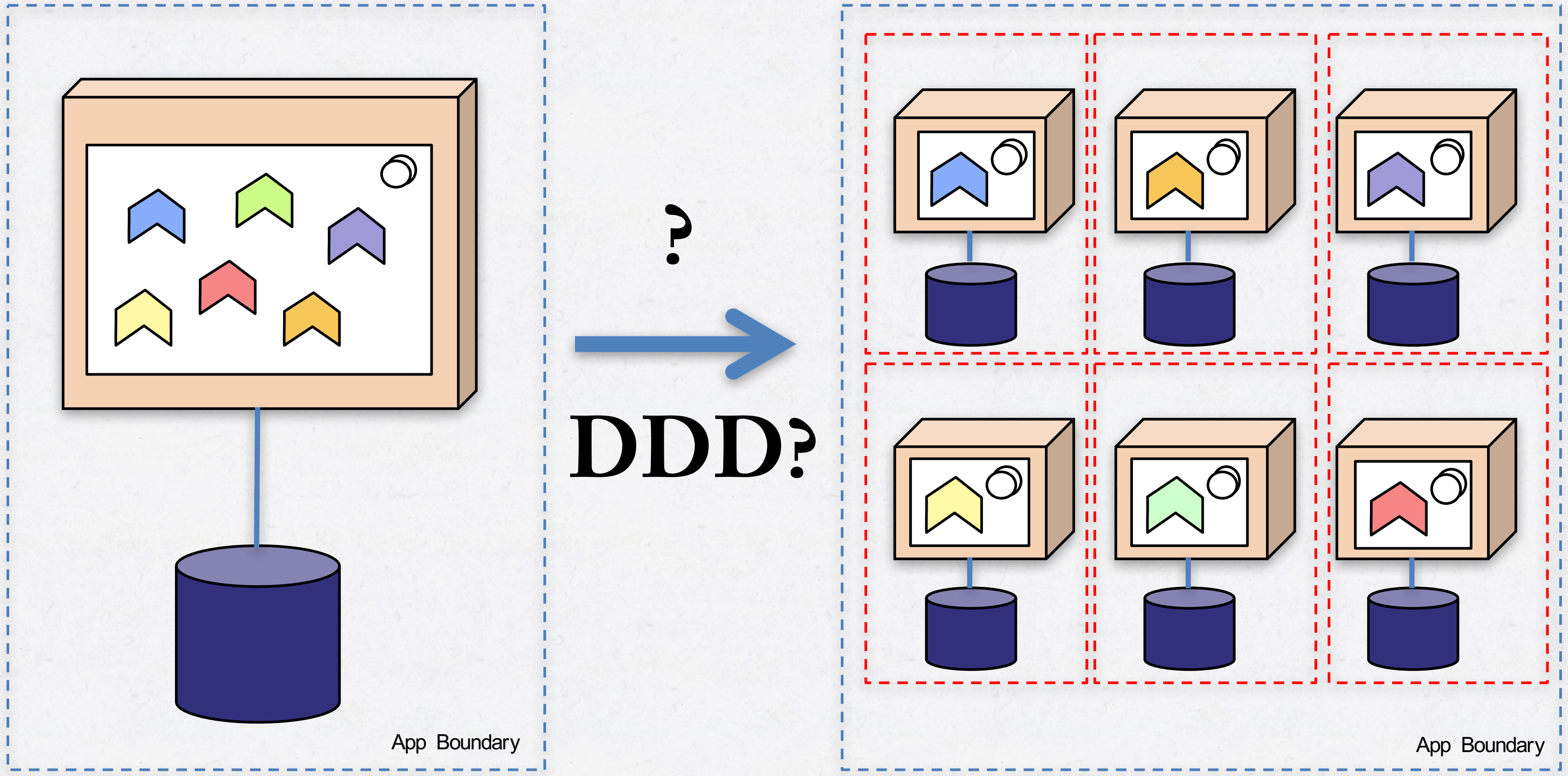
Out of scope:

- Infrastructure
- DDD In-Depth

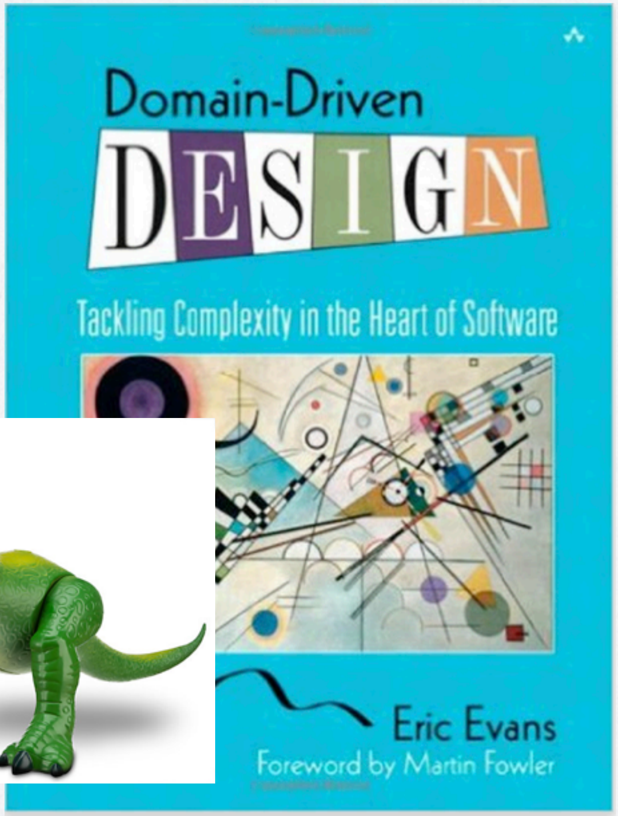


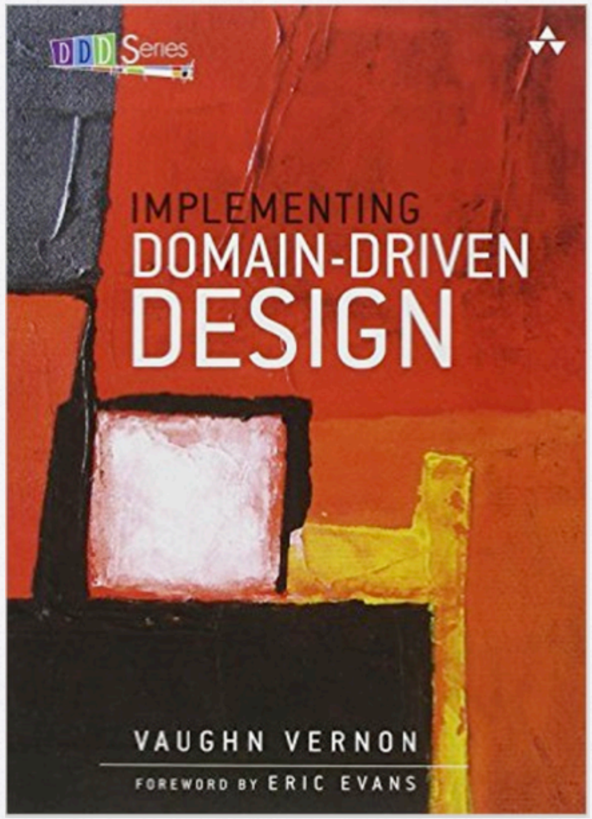
http://blogs.gartner.com/gary-olliffe/2015/01/30/microservices-guts-on-the-outside/

META PRESENTATION



DDD - BLUE OR RED PILL?


Domain-Driven
DESIGN
Tackling Complexity in the Heart of Software
Eric Evans
Foreword by Martin Fowler



(c) Disney

https://www.amazon.com/Domain-Driven-Design-Tackling-Complexity-Software/dp/0321125215
https://www.amazon.com/Implementing-Domain-Driven-Design-Vaughn-Vernon/dp/0321834577

DDD - DEFINITION

Focus on the Domain and the complexity and opportunity within it

”**Domain-Driven Design (DDD)** *is an approach to software development for complex needs by connecting the implementation to an evolving model*”

May not carry it's own weight for trivial problems

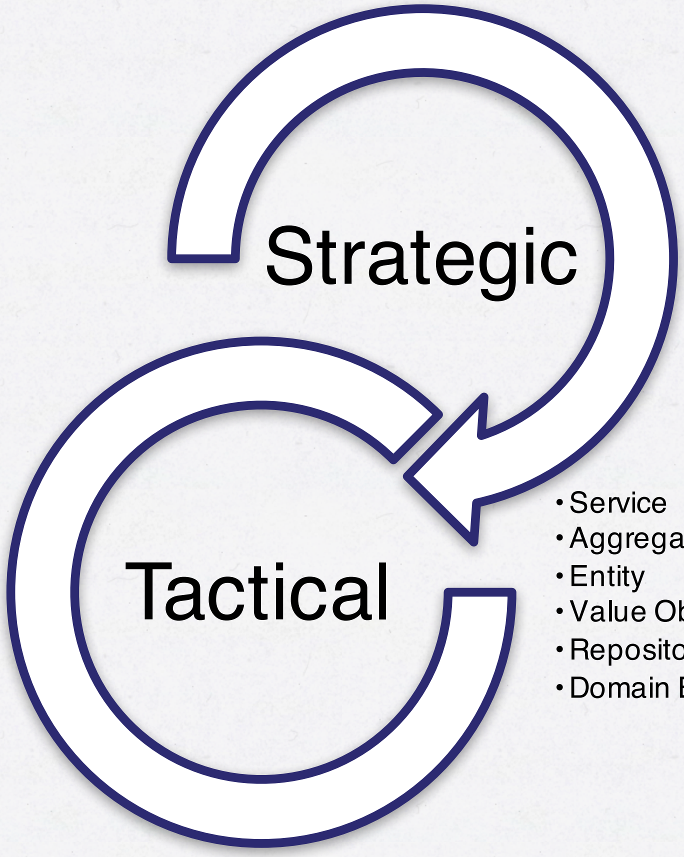
Write software that expresses those models, using a defined terminology and concepts within an explicit boundary

Base complex designs on models… produced by an iterative and close collaboration between Domain Experts and Software Experts

https://en.wikipedia.org/wiki/Domain-driven_design

CALLISTA
– ENTERPRISE –

DDD - TWO PARTS

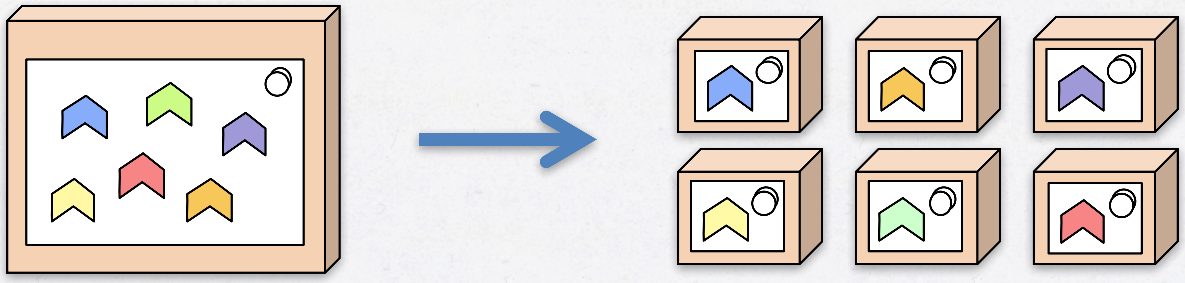


- Bounded Context
- Ubiquitous Language
- Subdomains
- Context Mapping

- Service
- Aggregate
- Entity
- Value Object
- Repository
- Domain Event

MICROSERVICES

Definition



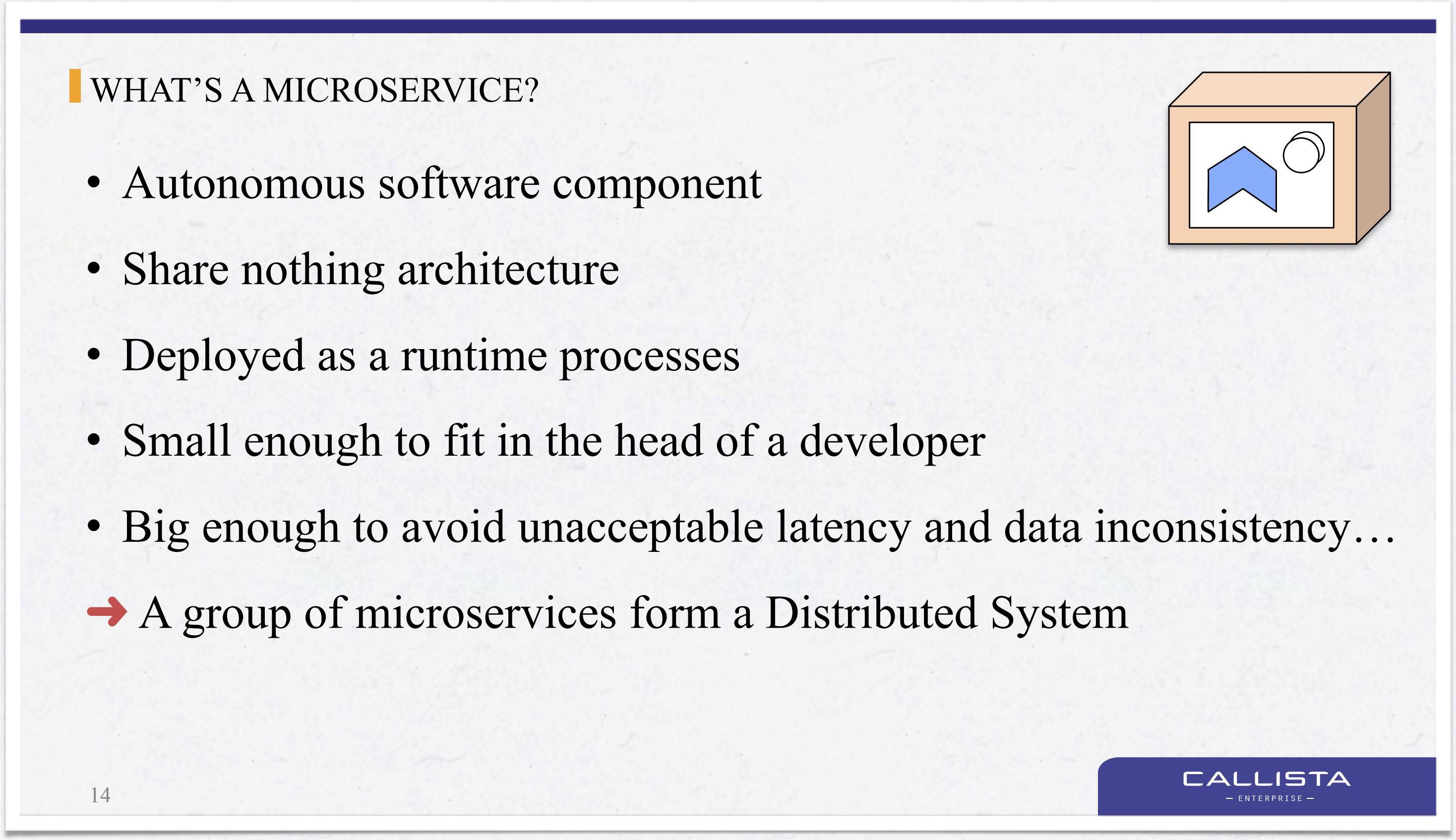
"Small, autonomous services that work together, modelled around a business domain."

Sam Newman, "Building Microservices" O'Reilly Media 2015

MICROSERVICES

Yet a definition

Cadec 2016 – ”Microservices and Docker containers”



WHAT’S A MICROSERVICE?

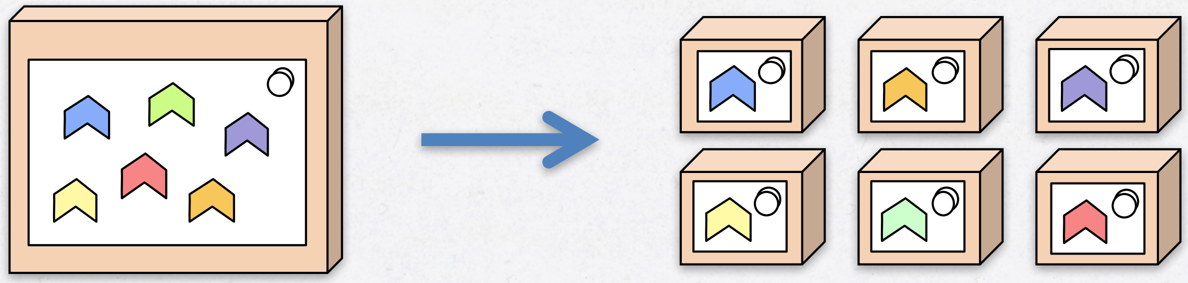
- Autonomous software component
- Share nothing architecture
- Deployed as a runtime processes
- Small enough to fit in the head of a developer
- Big enough to avoid unacceptable latency and data inconsistency…

➜ A group of microservices form a Distributed System

14

MICROSERVICES

Definition

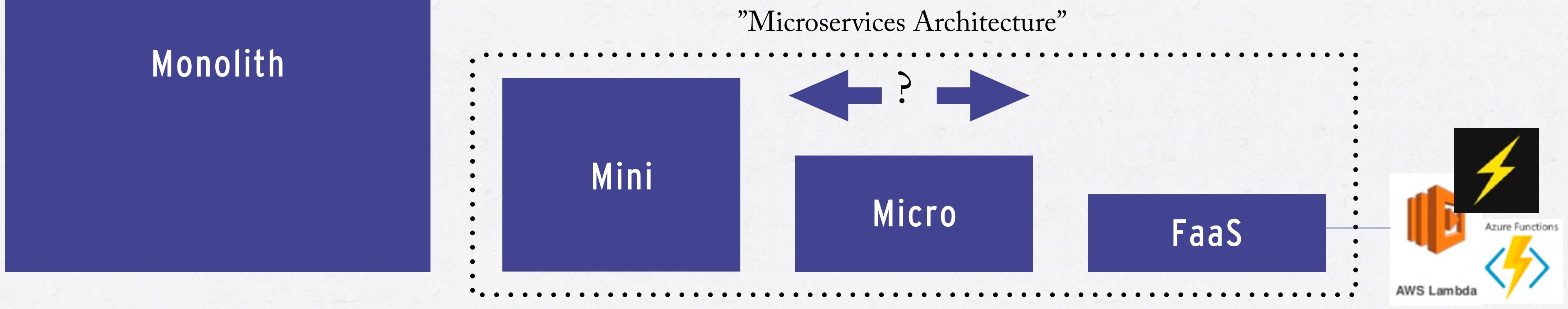


?

”Small, autonomous services that work together, modelled around a business domain.”

Sam Newman, ”Building Microservices” O'Reilly Media 2015

SIZE?



Gartner: By 2017, more than 90% of organizations that try microservices will find the paradigm too disruptive and use miniservices instead.

When starting out with Microservices, aim for coarse grained services.

META PRESENTATION

In this talk:

- Brief intro to main concepts
- **Rationales**
- Useful DDD concepts
- Migration

Out of scope:

- Infrastructure
- DDD In-Depth

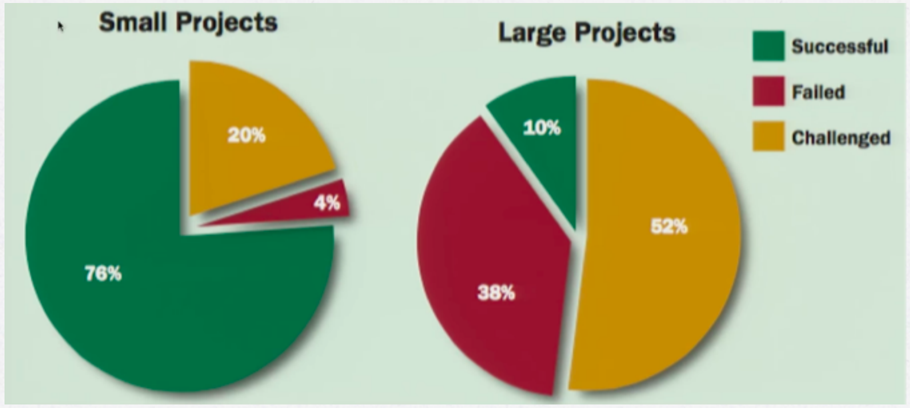
RATIONALE FOR MICROSERVICES

Business	• Time to market • Agility
Runtime	• Scalability (Elasticy, Density, Performance) • Resilience • Deployability
Organization	• Autonomous ”DevOps teams” formed around business capabilities
Maintainability	• Polyglot (across the entire stack) • Replaceability & Composability • Small …

SMALL IS THE NEW BLACK

Benefits of "Small":

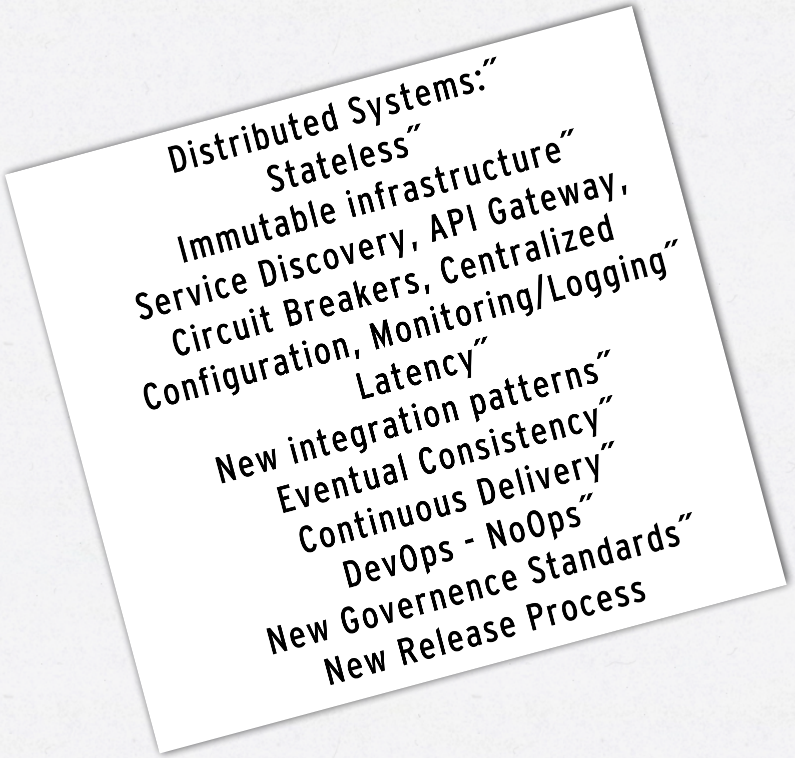
- Easier to understand
- Enables small and efficient teams
- Likelihood of successful project higher (on time and budget)



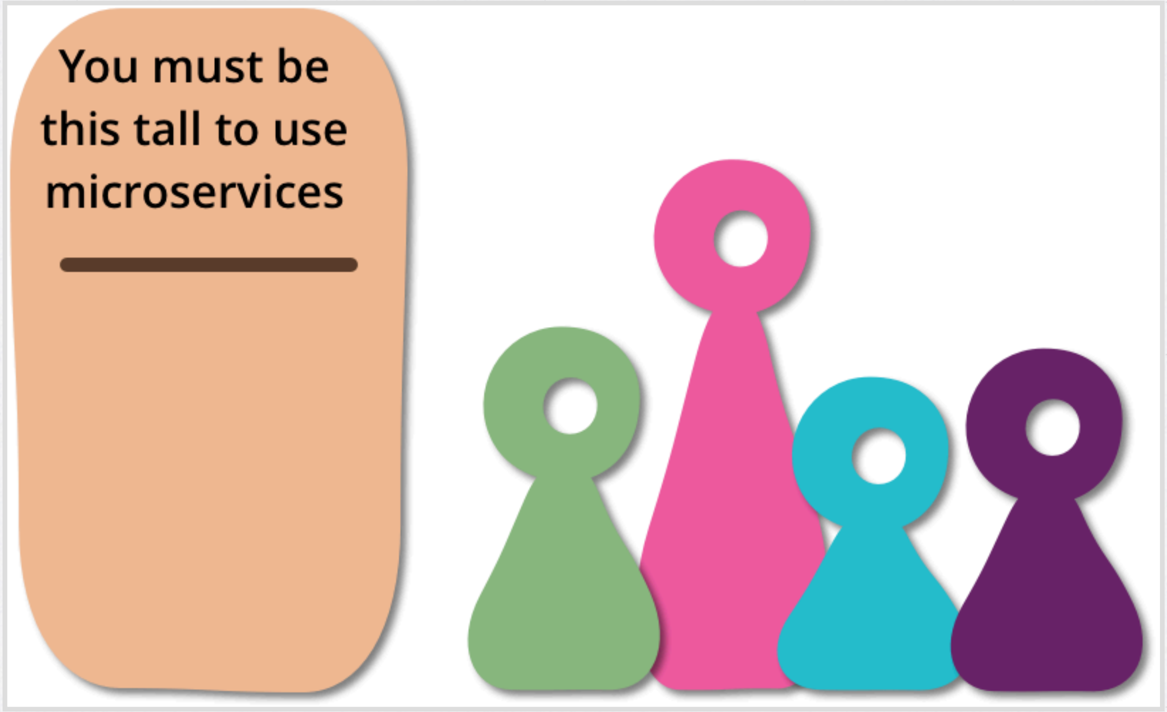
https://www.infoq.com/articles/standish-chaos-2015

… NO SUCH THING AS A FREE LUNCH

- Rapid Provisioning
- Basic Monitoring
- Rapid Application Deployment



"Distributed Systems:"
"Stateless"
"Immutable infrastructure"
"Service Discovery, API Gateway, Circuit Breakers, Centralized Configuration, Monitoring/Logging"
"Latency"
"New integration patterns"
"Eventual Consistency"
"Continuous Delivery"
"DevOps - NoOps"
"New Governence Standards"
"New Release Process



https://martinfowler.com/bliki/MicroservicePrerequisites.html

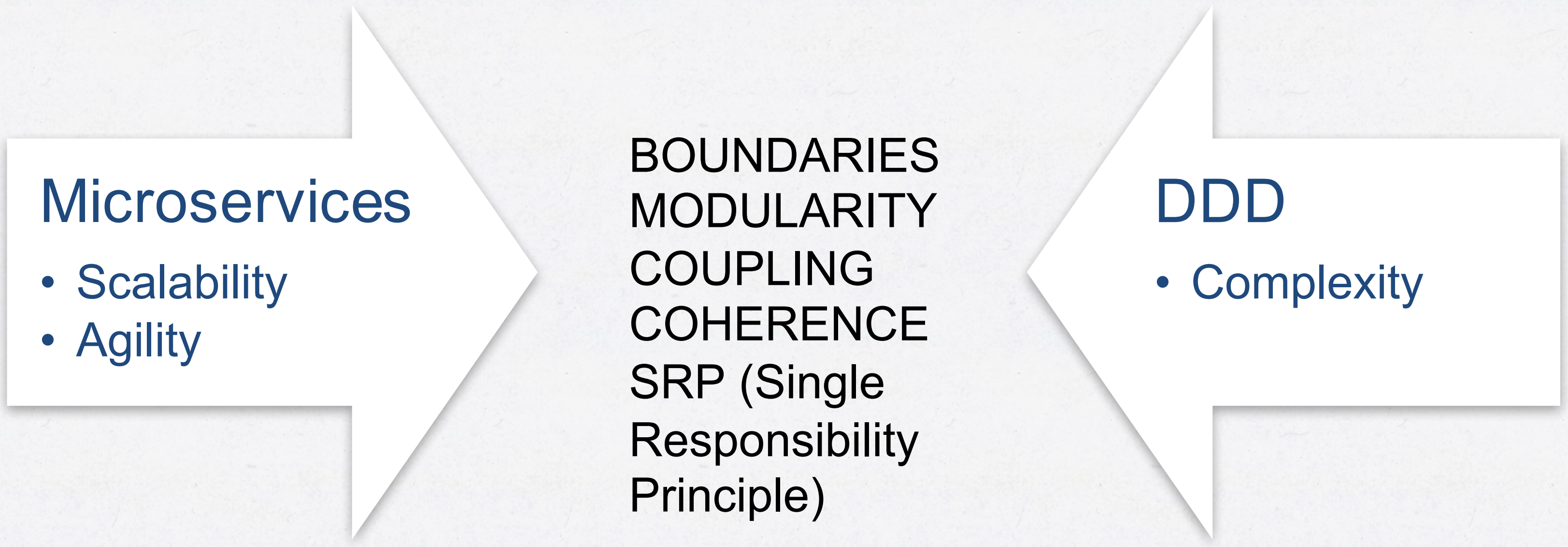
RATIONALE FOR DDD AND MICROSERVICES?

Challenges in the "traditional enterprise"

- Complex business process (and organization)
- A (large) gap between IT and business
- Long lifecycle of software systems
- (Legacy)

https://www.flickr.com

DDD AND MICROSERVICES? HOW DO THEY CONVERGE?



DDD paired with Microservices can amplify the quality attributes of the software solution.

META PRESENTATION

In this talk:

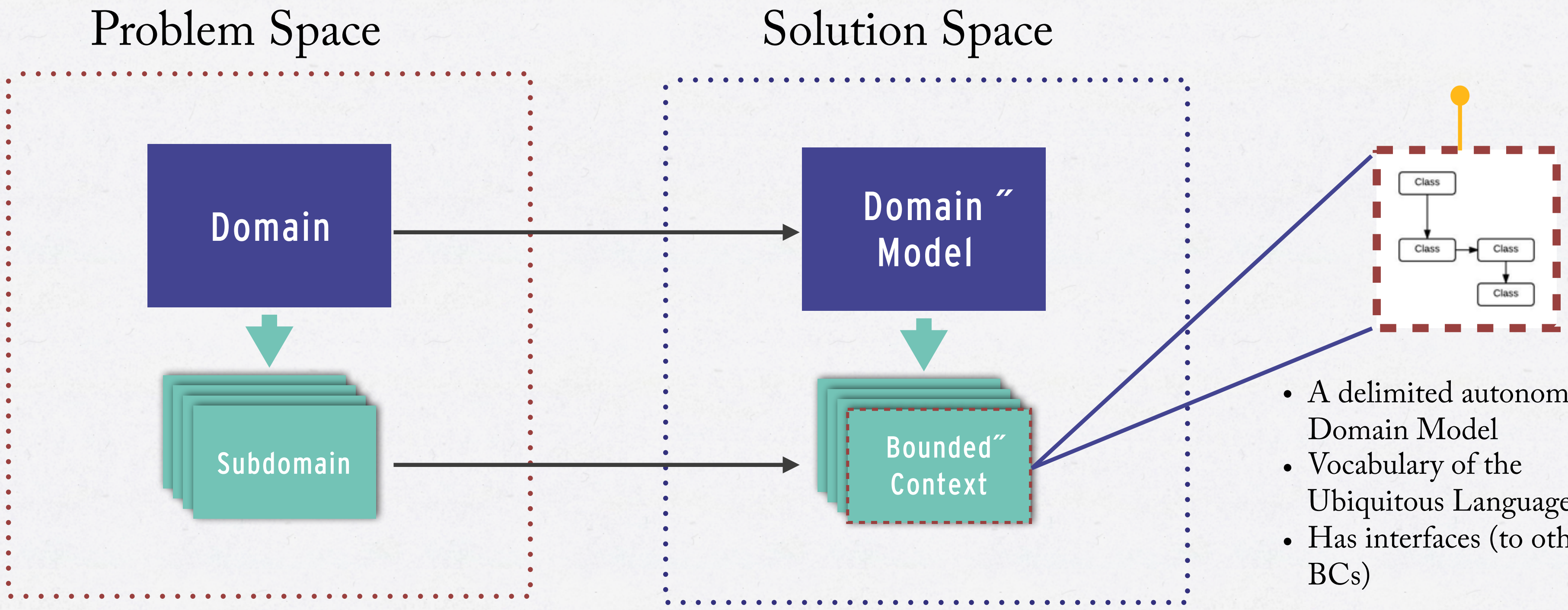
- Brief intro to main concepts
- Rationales
- **Useful DDD concepts**
- Migration

Out of scope:

- Infrastructure
- DDD In-Depth

BOUNDED CONTEXT

"… a boundary (typically a subsystem, or the work of a particular team) within which a particular model is defined and applicable.



- A delimited autonomous, Domain Model
- Vocabulary of the Ubiquitous Language
- Has interfaces (to other BCs)

BOUNDED CONTEXT

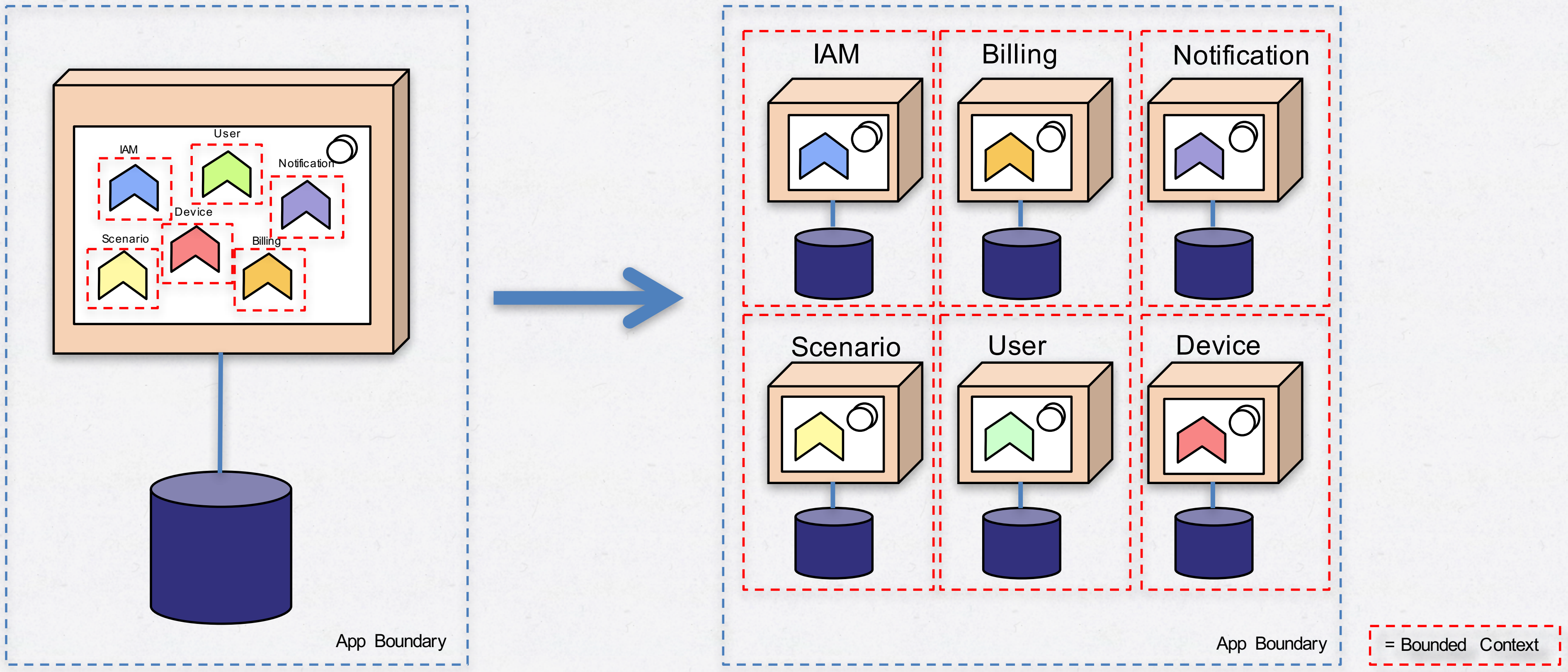
"… a boundary (typically a subsystem, or the work of a particular team) within which a particular model is defined and applicable.

- How do we find them?
- Bounded Context in software:
  - Logical separation-> Weak: Namespaces (JVM: Packages)
  - Binary separation-> Medium: Binary artifacts (JVM: JAR)
  - Process separation -> Strong: Deployment Unit separation

Model your Microservices around business domains, i.e. align Bounded Context with Service Boundary.

BOUNDED CONTEXT

Applied to a fictive domain



CALLISTA
— ENTERPRISE —

CONTEXT MAP

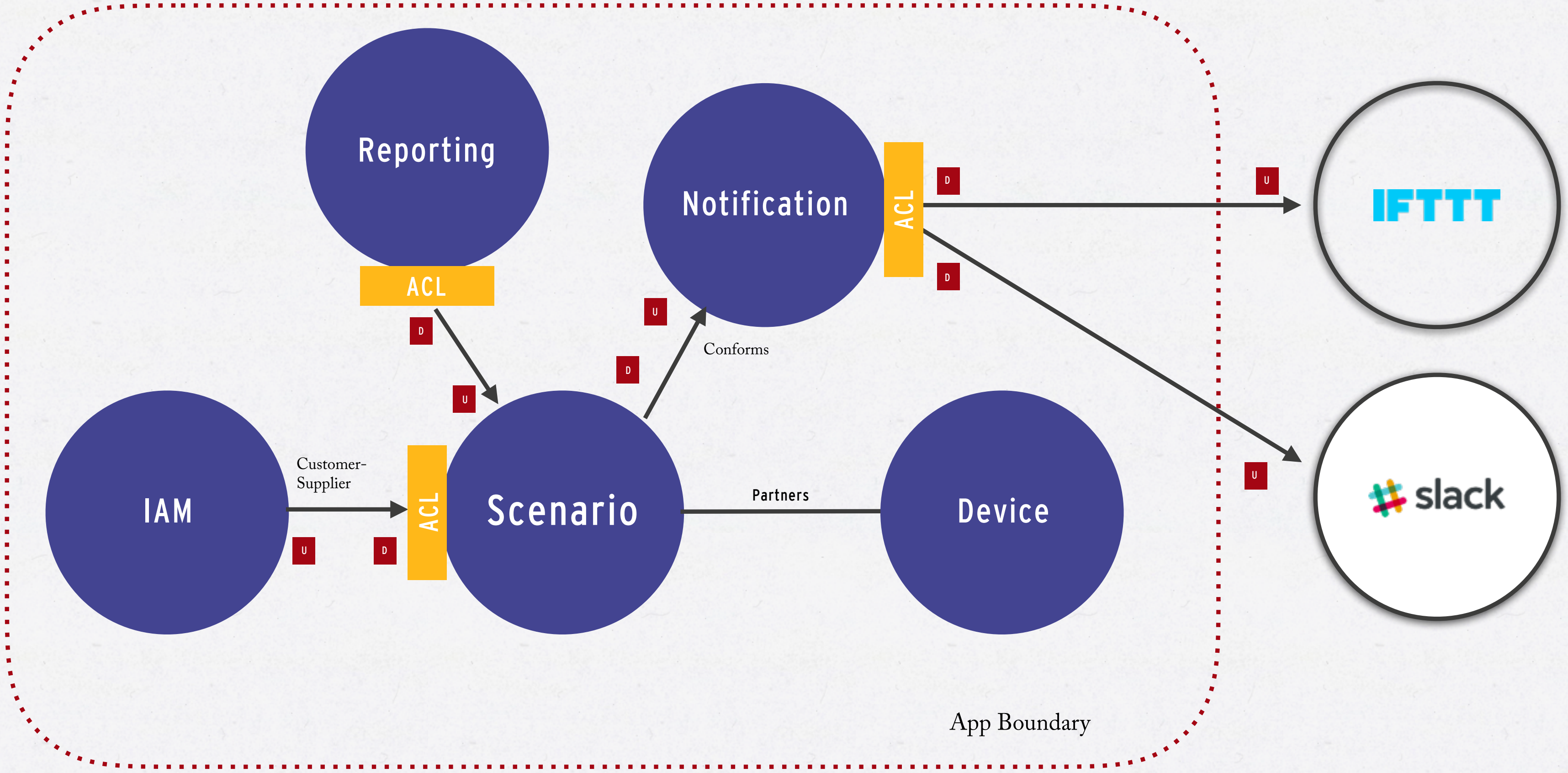
"Identify each model in play on the project and define its Bounded Context"

- A simple diagram that captures the ”existing terrain”
- A catalyst for inter-team communication
- Find relationships with all other projects you depend on
- ”A Context Map is not an Enterprise Architecture or system topology diagram”

CALLISTA
– ENTERPRISE –

CONTEXT MAP

Example



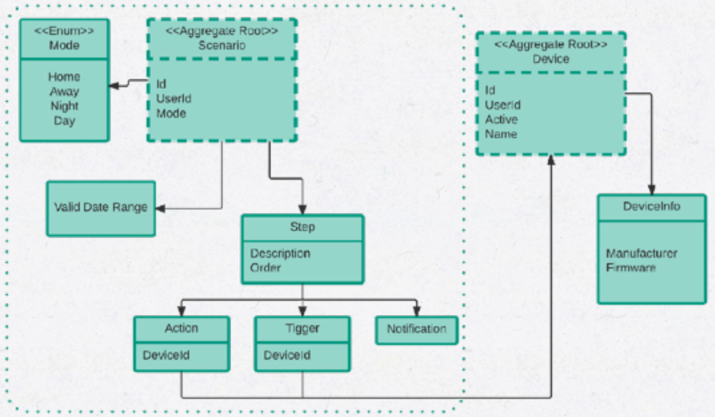
Use Context Maps to get an understanding of how BC's and services depend on each other.

CALLISTA
– ENTERPRISE –

AGGREGATE

"A cluster of associated objects that are treated as a unit for the purpose of data changes"

- Arrange related objects under a common "parent" designated as the *Aggregate Root*
- Reference other Aggregates (Root) by Identity
- A set of consistency rules applies within the aggregate
- Should be kept small (performance, scalability)
- Referenced Aggregates are eventually consistent



Group Domain Objects as Aggregates (may be several in one BC) to identify the "minimum size" of a Microservice.

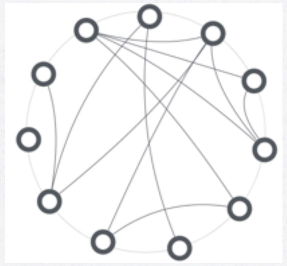
CALLISTA
– ENTERPRISE –

DOMAIN EVENTS

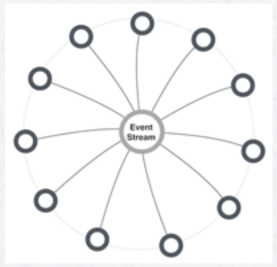
"Something happened that Domain Experts care about"

- Part of the Domain Model expressed in the *Ubiquitous Language*
- Identify Domain Events early to understand cross-service communication needs and find service boundaries
- Event Sourcing and CQRS (Command Query Responsibility Segregation) are common associated patterns…

Point-To-Point Orchestration



EDA Choreography

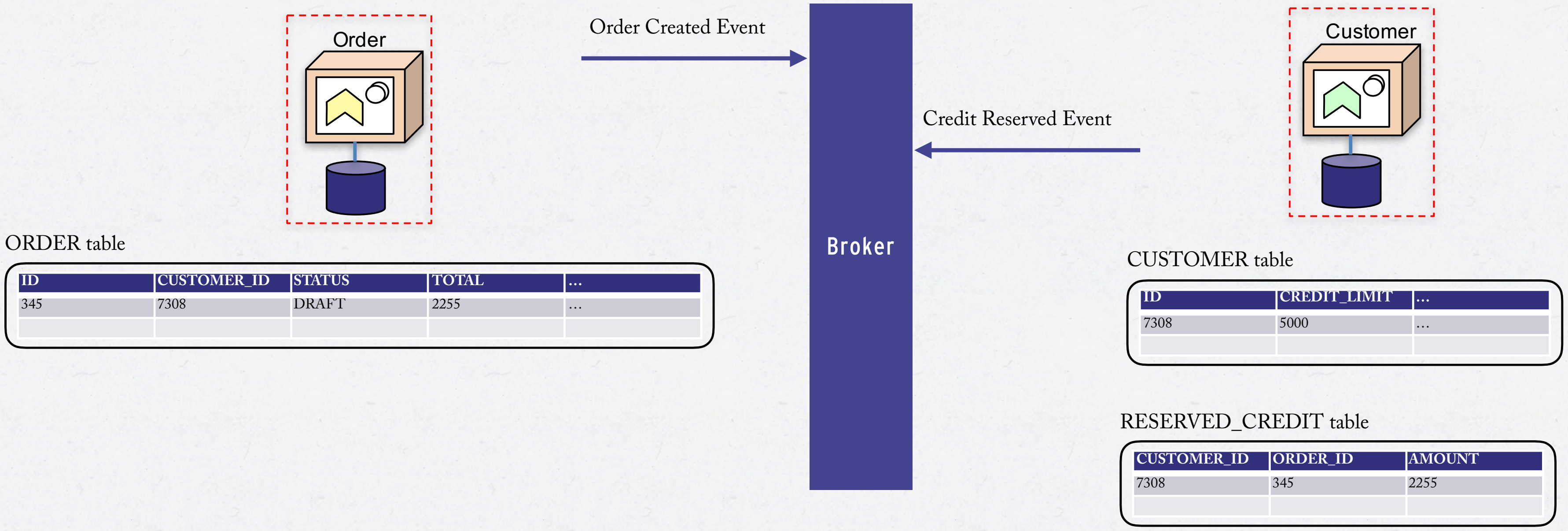


https://www.thoughtworks.com/insights/blog/scaling-microservices-event-stream

DOMAIN EVENTS

Eventual Consistency

"Something happened that domain experts care about"



ORDER table

ID	CUSTOMER_ID	STATUS	TOTAL	...
345	7308	DRAFT	2255	...

CUSTOMER table

ID	CREDIT_LIMIT	...
7308	5000	...

RESERVED_CREDIT table

CUSTOMER_ID	ORDER_ID	AMOUNT
7308	345	2255

Model Domain Events to facilitate eventual consistency across Aggregates and Bounded Contexts - i.e. across services in a Microservice context.

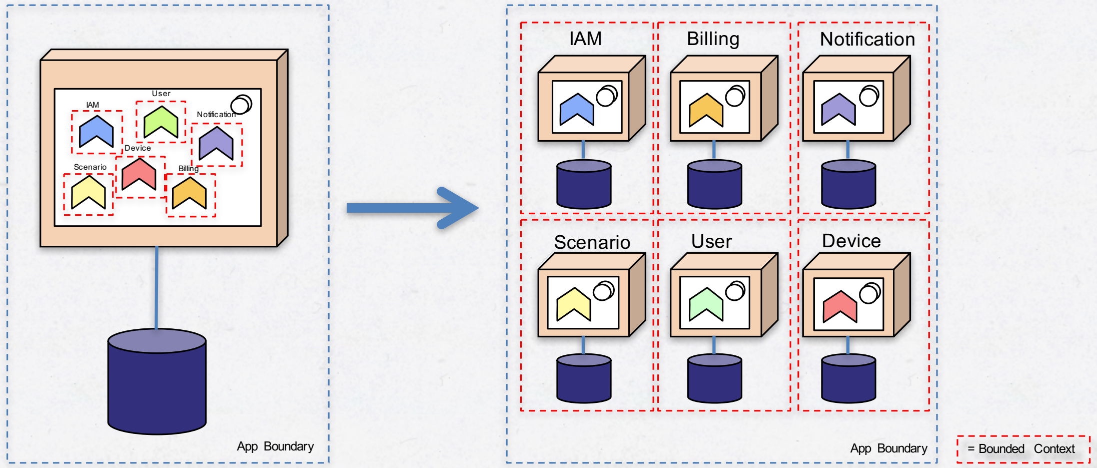
META PRESENTATION

In this talk:

- Brief intro to main concepts
- Rationales
- Useful DDD concepts
- **Migration**

Out of scope:

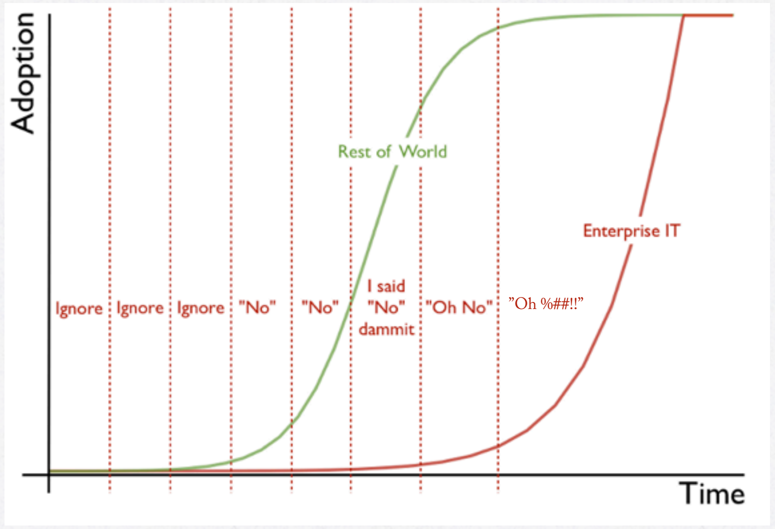
- Infrastructure
- DDD In-Depth



MIGRATION

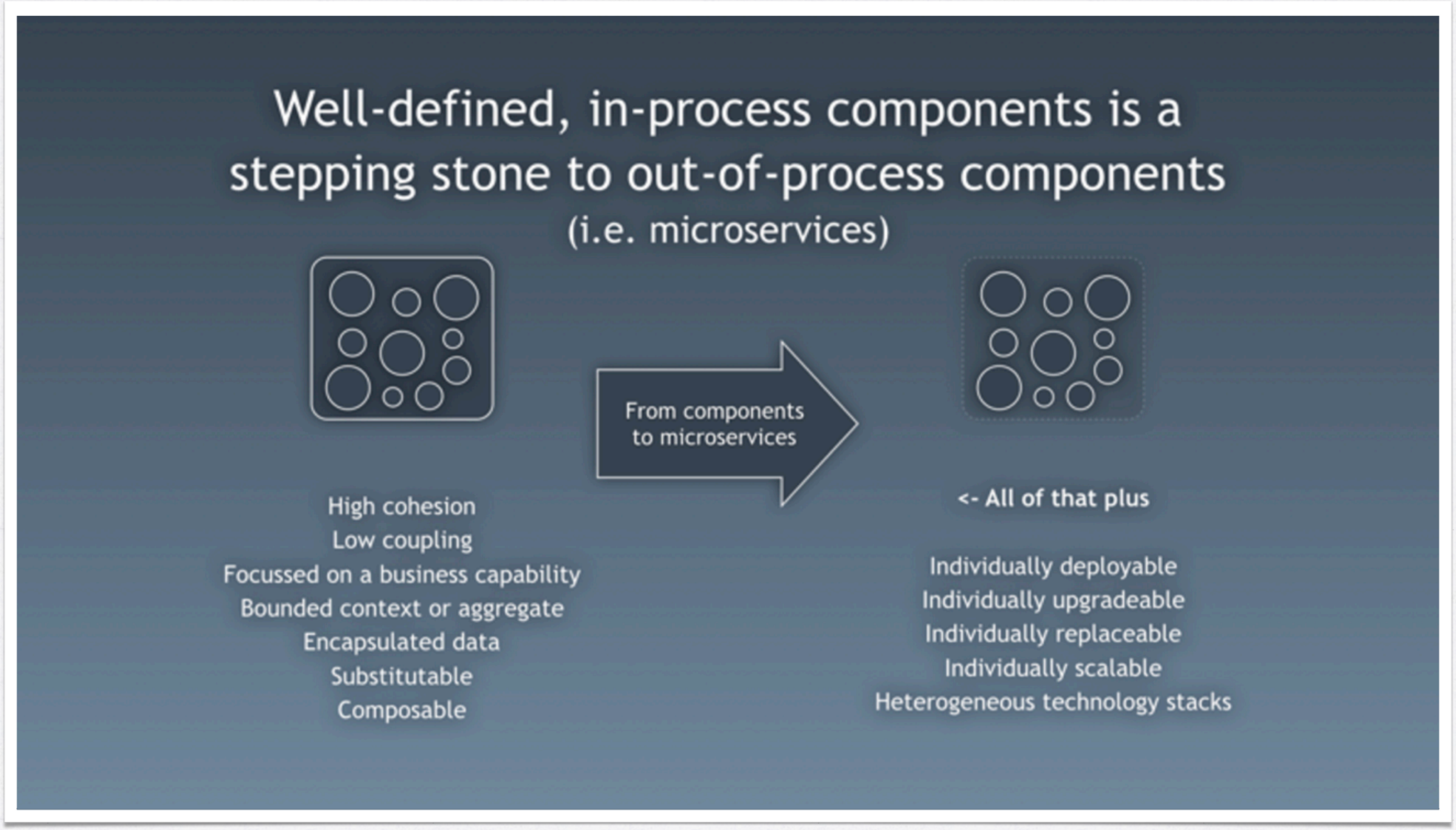
Strategies

- Big Bang : dump and start over from scratch
- Strangler application
  - http://paulhammant.com/2013/07/14/legacy-application-strangulation-case-studies/
- Monolith first…
  - https://martinfowler.com/bliki/MonolithFirst.html
- … or not
  - http://martinfowler.com/articles/dont-start-monolith.html



http://enterpriseitadoption.com/

THE STRUCTURED MONOLITH



Well-defined, in-process components is a stepping stone to out-of-process components (i.e. microservices)

From components to microservices

High cohesion
Low coupling
Focussed on a business capability
Bounded context or aggregate
Encapsulated data
Substitutable
Composable

<- All of that plus

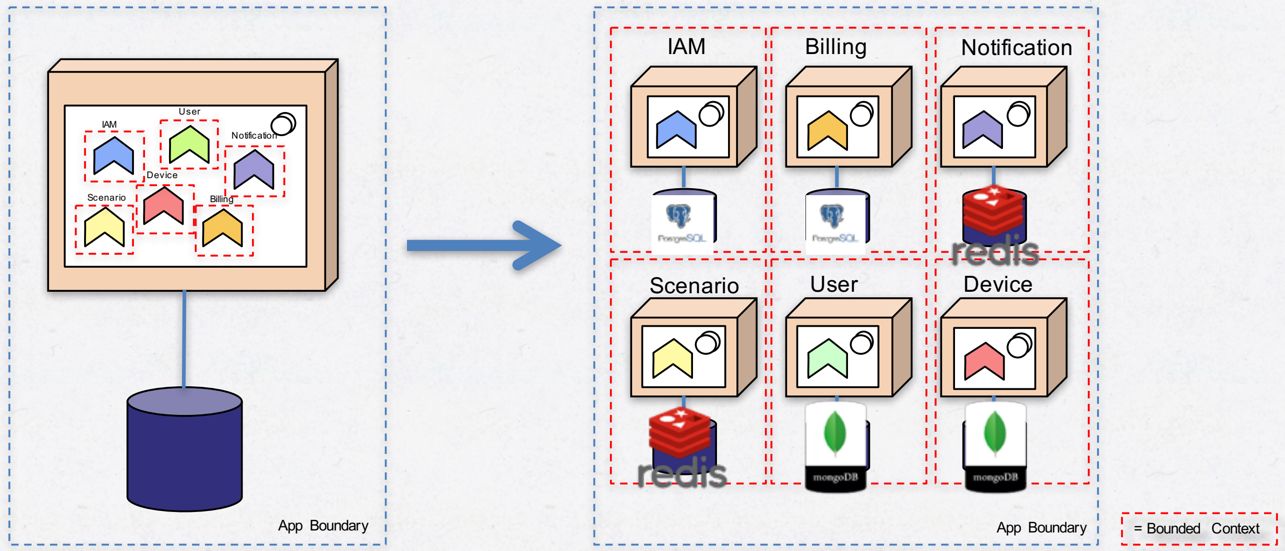
Individually deployable
Individually upgradeable
Individually replaceable
Individually scalable
Heterogeneous technology stacks

Simon Brown http://www.codingthearchitecture.com/presentations/devnexus2016-modular-monoliths

DATA MIGRATION

One DB (schema) to rule them all?

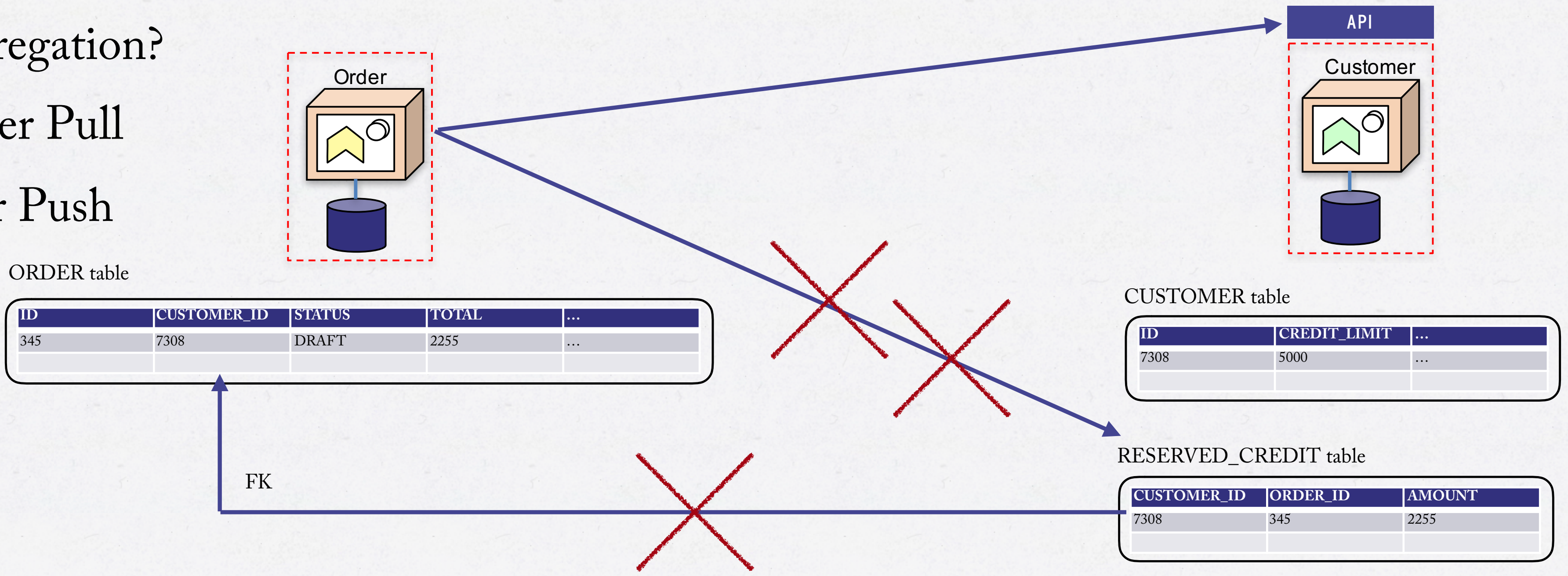
- Independently scalable?
- Low impact schema changes?
- Technology opportunities?



DATA MIGRATION

Consequences and considerations

- ACID to Eventual Consistency
- Orphaned data?
- Data Aggregation?
  - Consumer Pull
  - Producer Push



ORDER table

ID	CUSTOMER_ID	STATUS	TOTAL	…
345	7308	DRAFT	2255	…

CUSTOMER table

ID	CREDIT_LIMIT	…
7308	5000	…

RESERVED_CREDIT table

CUSTOMER_ID	ORDER_ID	AMOUNT
7308	345	2255

TO SUM UP

Microservices

- Most applications will benefit from a Microservices arch:˝
  - Application Longevity - cost and complexity under long term control!˝
  - Not just of about Scalability!˝
- BUT: Does your organization have the capabilities (culture, skills, infra)?

DDD

- DDD is en excellent allied when crafting distributed applications - highly coherent, loosely coupled ar
  in tune with business˝
- Helps us find the Service Boundaries and gives internal structure˝
- Results in a domain model based on crips concepts, with little room for misconceptions.

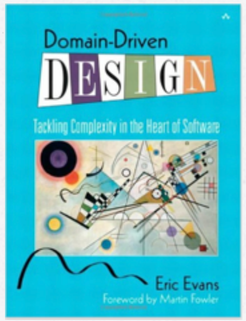
Migration

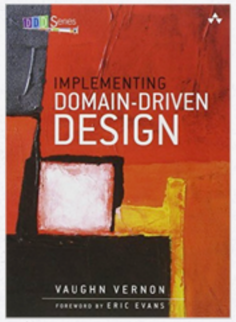
- Stay with a well structured Monolith until you get boundaries right˝
- Partial replacement (Strangler pattern) to play it safe˝
- Start small (i.e. big) and learn as you go...

CALLISTA
– ENTERPRISE –

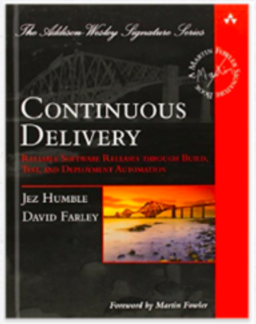
WHERE TO GO FROM HERE

References and Acknowledgments


https://www.amazon.com/Domain-Driven-Design-Tackling-Complexity-Software/dp/0321125215


https://www.amazon.com/Implementing-Domain-Driven-Design-Vaughn-Vernon/dp/0321834577

https://www.amazon.com/Building-Microservices-Designing-Fine-Grained-Systems/dp/1491950358/


https://www.amazon.com/Continuous-Delivery-Deployment-Automation-Addison-Wesley/dp/0321601912/

https://www.amazon.com/Release-Production-Ready-Software-Pragmatic-Programmers/dp/0978739213

Eric Evans - Jan 2016 - ”Tackling Complexity In the Heart of Software”: https://www.youtube.com/watch?v=dnUFEg68ESM

Greg Young, CQRS & Event Sourcing
https://www.infoq.com/news/2016/04/event-sourcing-anti-pattern

http://codebetter.com/gregyoung/2010/02/16/cqrs-task-based-uis-event-sourcing-agh/

Chris Richardson, Developing Transactional Microservices

https://www.infoq.com/articles/microservices-aggregates-events-cqrs-part-1-richardson