From Monolith to Micro services

SATURN 2015 – Einar Landre, Jørn Ølmheim, Harald Wesenberg – Statoil ASA

Statoil

Introduction

- The Case
- Domain Driven Design
- Microservices

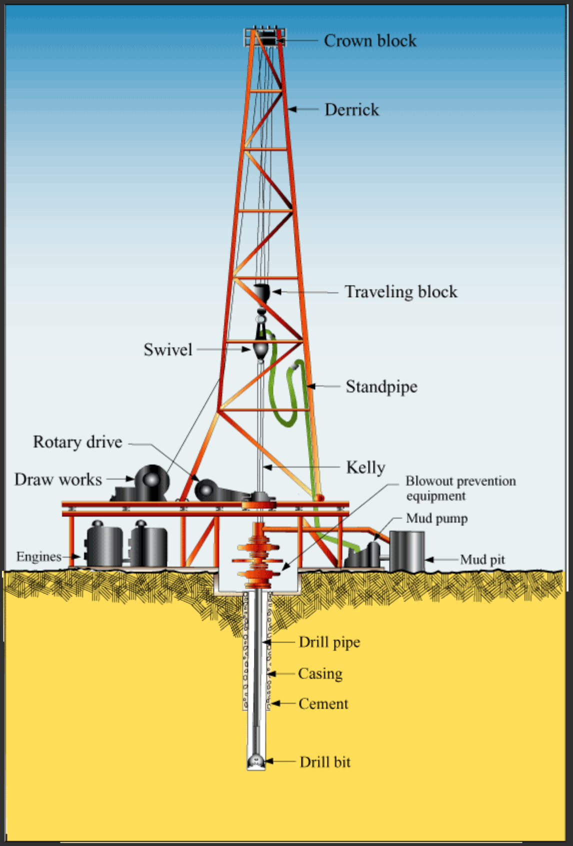
Discussion

- Data vs Domain Driven
- Organization and Team
- Breaking the Monolith

Statoil

The Case

DBR – Drilling Reporting System



Planning and reporting of drilling operations

Began as a simple activity log

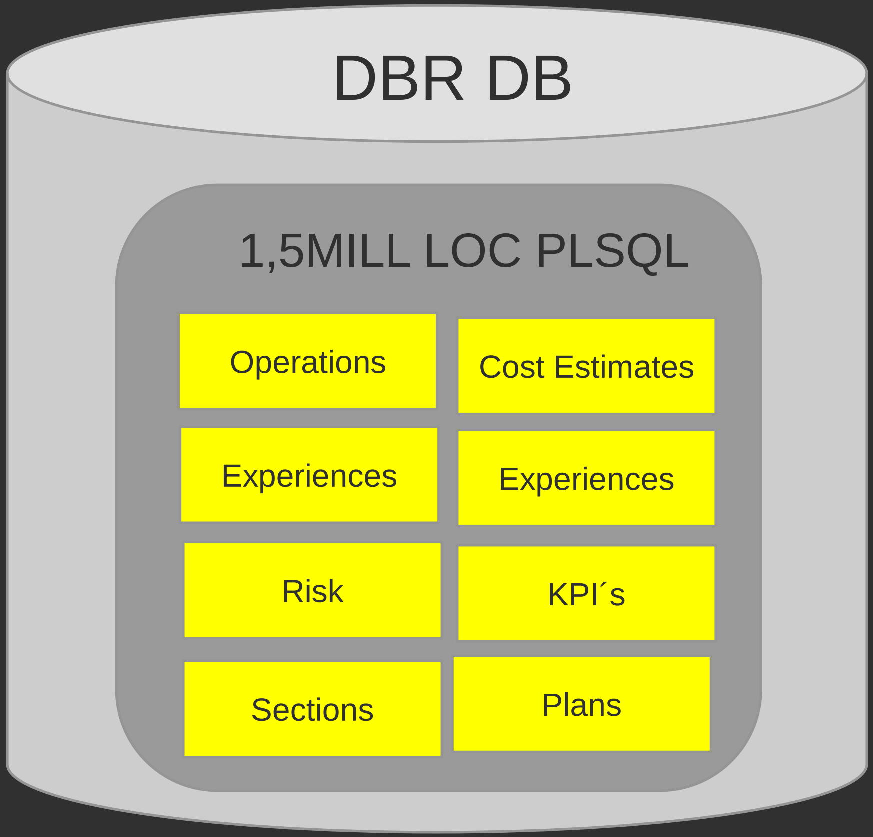
Has evolved into much more over 20 years

Client server application
- 3,5 MLOC
  - 1,5 MLOC PLSQL
  - 1,0 MLOC C#/ APS.NET/Silverlight
  - 1,5 MLOC PowerBuilder

Began as a PowerBuilder / Oracle application
- Extended with Web later

Statoil

Architecture & Technology



Tightly coupled

+10.000LOC procedures

Fat Windows Client / Citrix

Technological fragmented

Scripted business logic

Statoil

The Team

Small (3-5) over very long time
- +15 years
- Now two teams 6+4, two locations

Technology segregated
- Database
- Power Builder
- Web (Microsoft Stack)

Vulnerable
- Dependent on individuals
- Number of years to retirement

Geographically segregated
- Stavanger
- Bergen

Statoil

Painpoints

Long lead times for new functionality

Convoluted database model

Deployment problems (windows client on Citrix)

System level testing

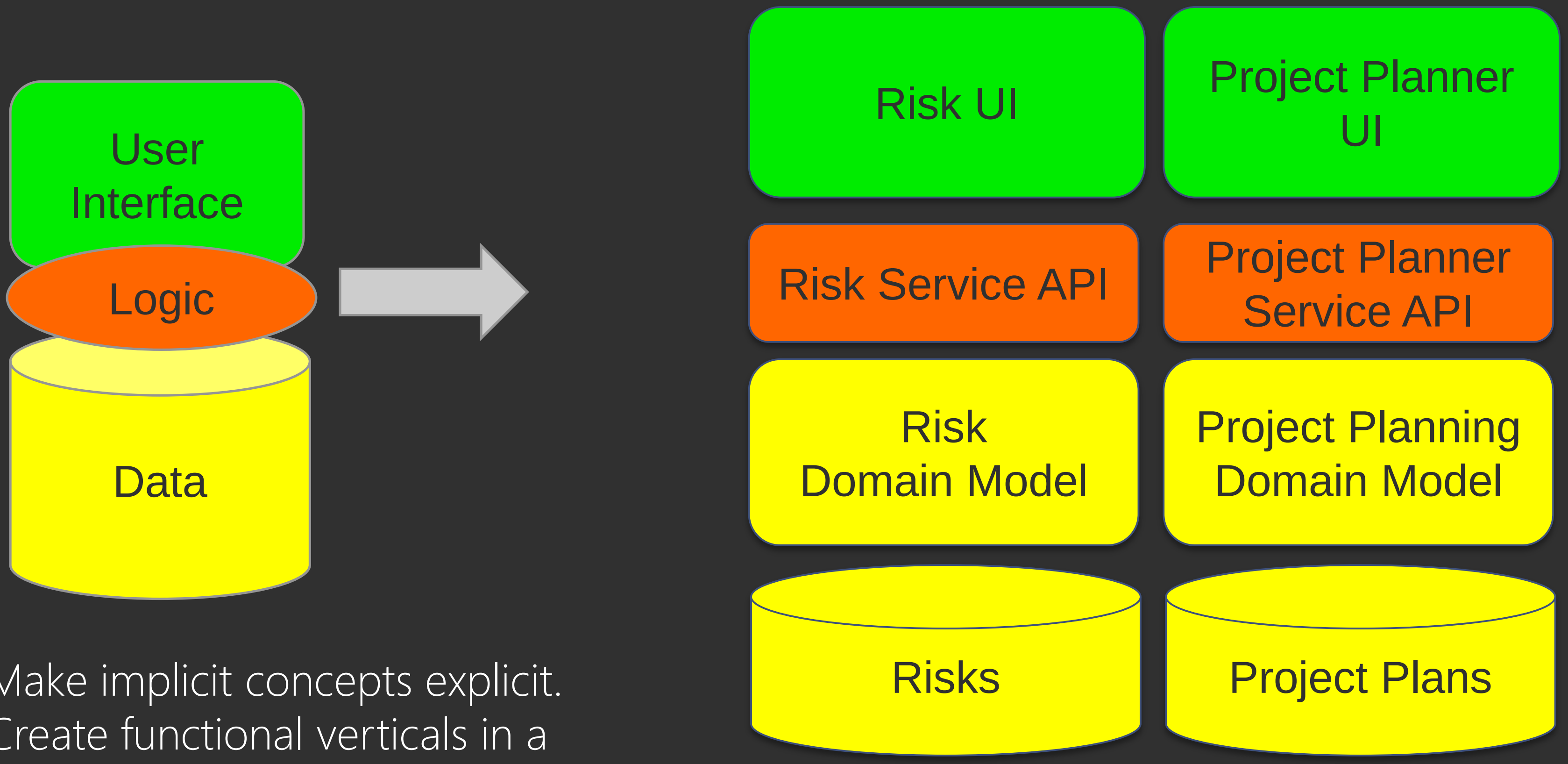
All in one build bundle

Obsolete technology

Short time to retirement

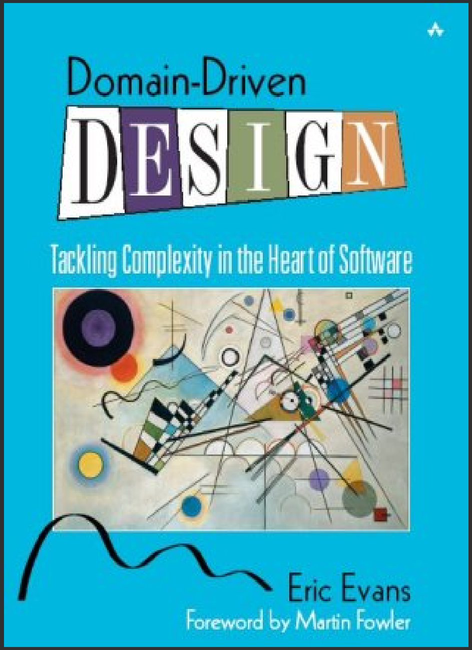
Statoil

Way forward



1. Make implicit concepts explicit.
2. Create functional verticals in a layered architecture.
3. How to split the database?

Domain Driven Design


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

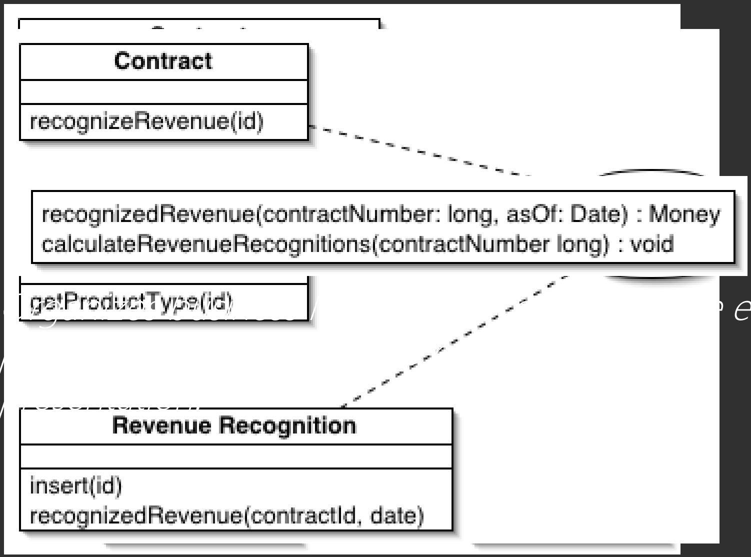
Domain-Driven Design:
Tackling the complexity in the heart of software
Eric Evans, 2003

http://www.domaindrivendesign.org

Domain Logic Patterns

Three main patterns for organizing the domain logic:

- Transaction Script
- Table Module
- Domain Model


Contract
recognizeRevenue(id)
recognizedRevenue(contractNumber: long, asOf: Date) : Money
calculateRevenueRecognitions(contractNumber long) : void
getProductType(id)
Revenue Recognition
insert(id)
recognizedRevenue(contractId, date)

each

A single instance that handles the business logic for all rows in a database table or view.

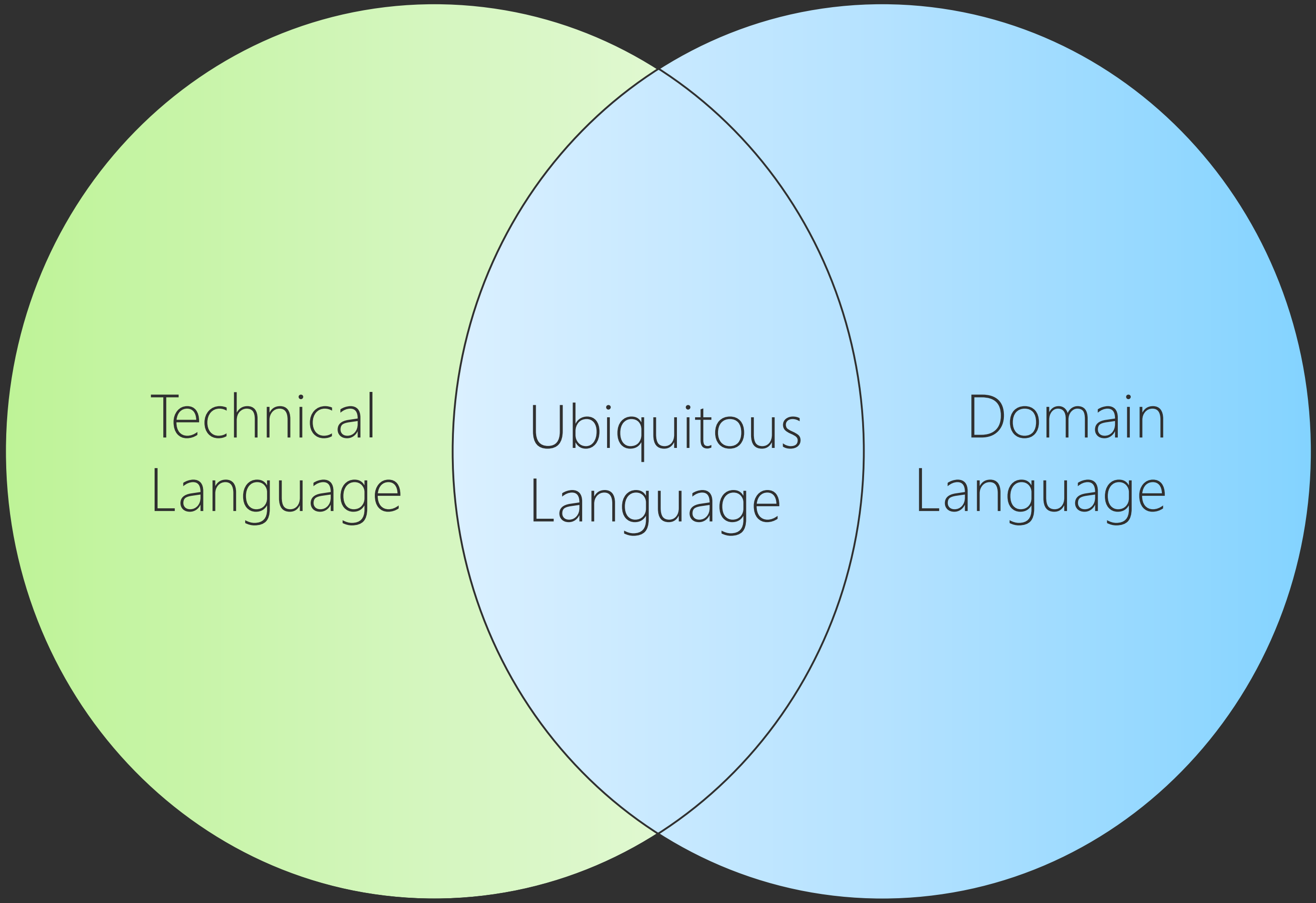
An object model of the domain that incorporates both behavior and data.

Domain Driven Design – distilled

- Ubiquitous (domain based) language
  - A language that is built around the concepts of the business and that permeates every activity in the project.
  - The language used to talk about the domain model in the project
- Patterns for building a domain model
- Strategic design principles and techniques

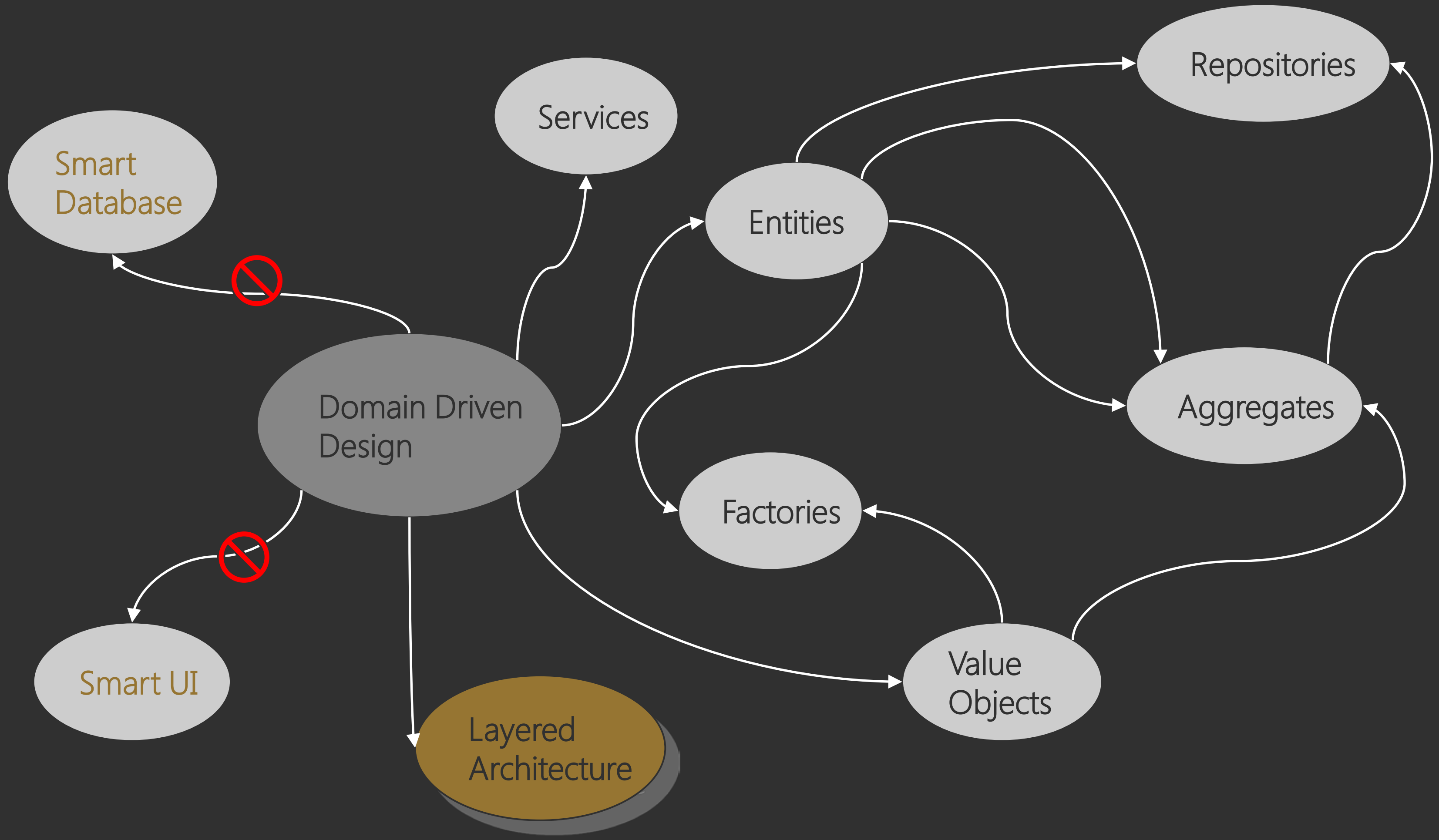
Statoil

Ubiquitous language – A Domain based language



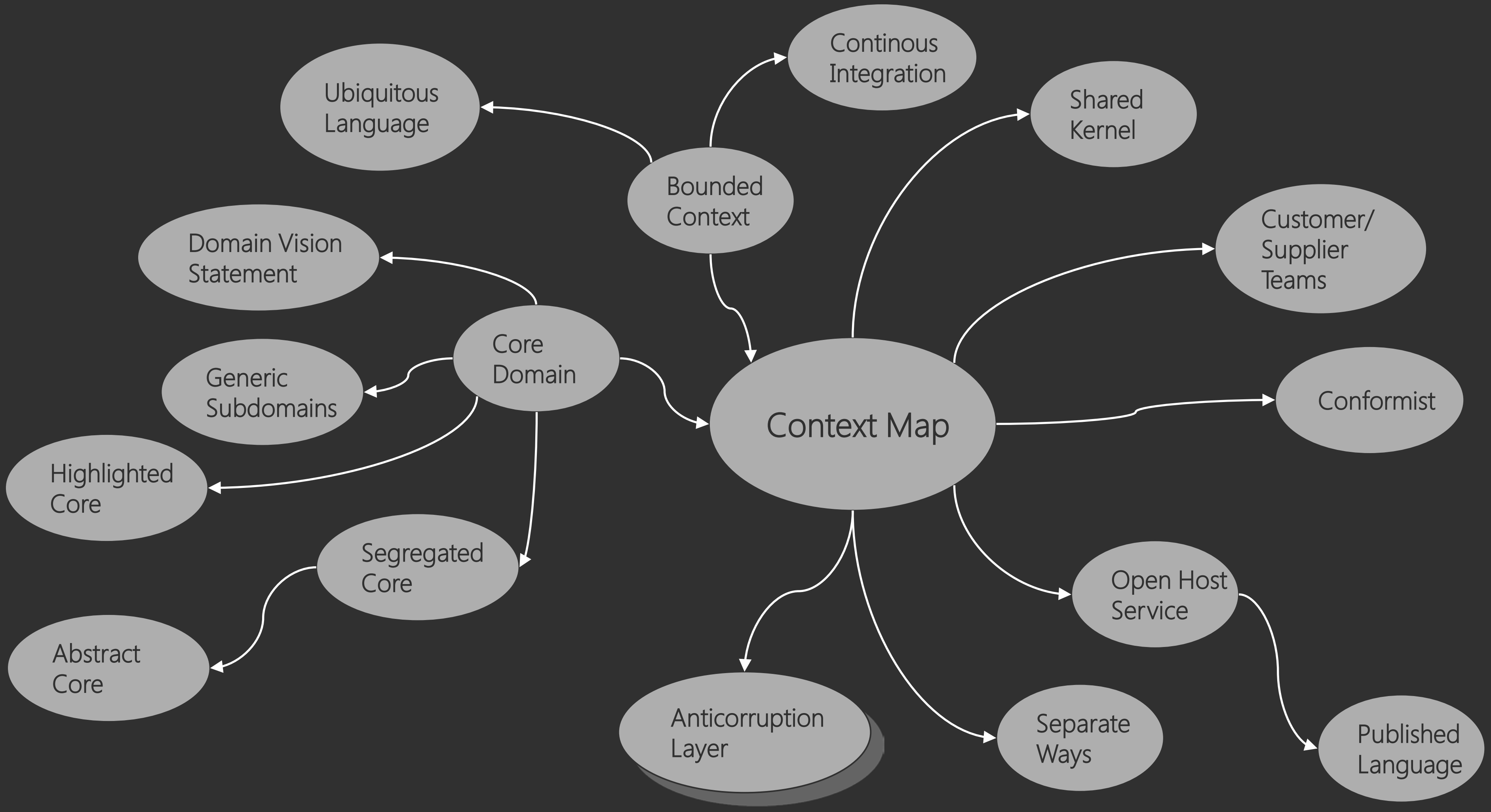
Statoil

Building blocks: Patterns



Statoil

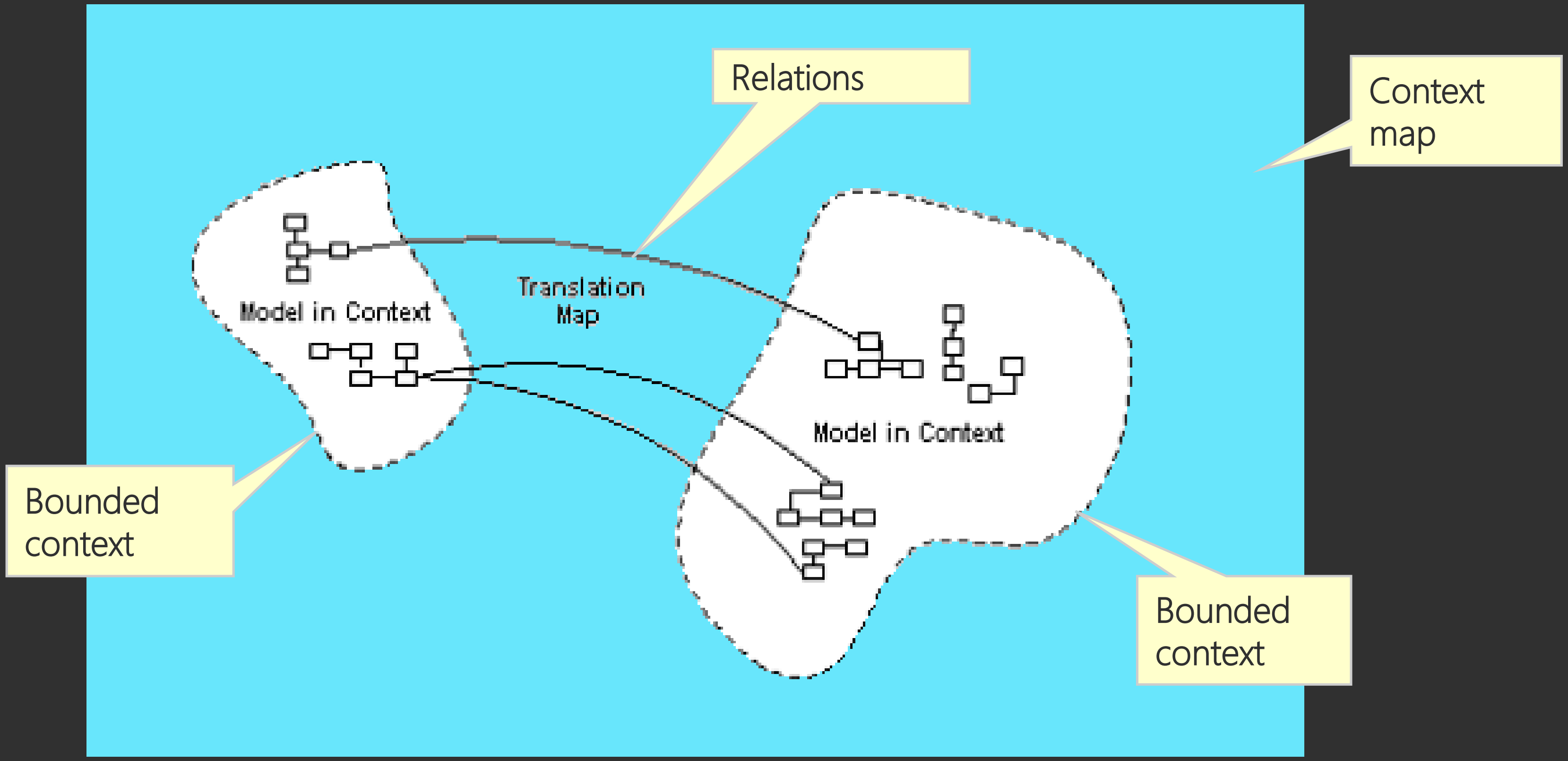
Domain Driven Design – Strategic Design



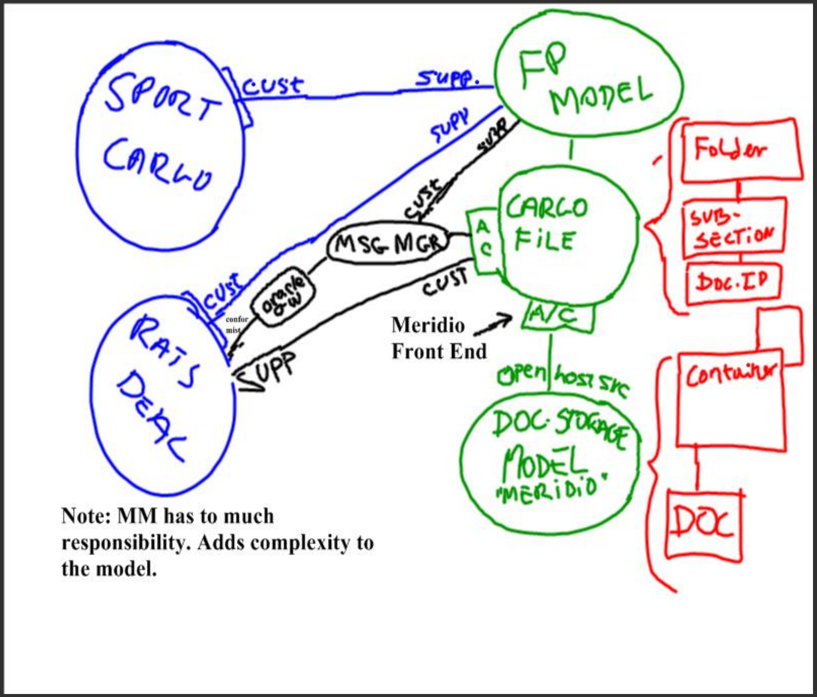
Statoil

Strategic design: Context maps

In large systems (or set of systems), we need a map to give us a picture of the models that are inside.



Strategic design: Integrity across systems



- Bounded context
  - The meaning of a domain concept is bound by the context it is used
- Context map
  - A map that describe the contexts and their relationships
- Relation types:
  - Shared kernel
  - Customer/supplier teams
  - Conformist
  - Anti-corruption layer
  - Separate ways
  - Open host service
  - Published language

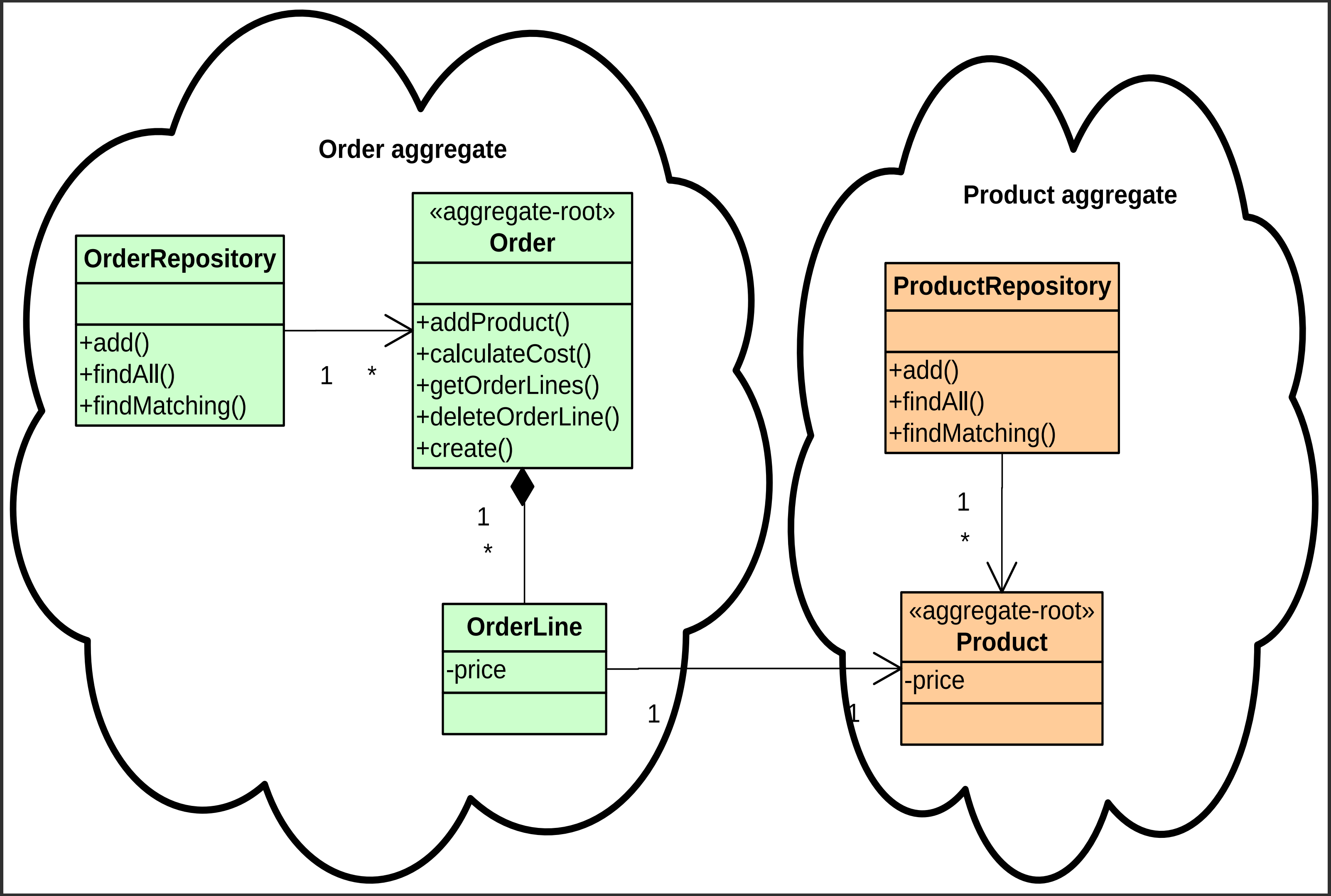
Statoil

Strategic design: Types of relations

Type	Description
Shared kernel	• Overlapping models shared among teams
Customer/supplier development teams	• One bounded context is maintained by one team but used by another
Conformist	• As C/S development teams, but the customer team stricly adheres to the supplier model, without the option to change it.
Anticorruption layer	• Isolation layer between models that take up the differences
Separate ways	• Avoid integration, let the models develop on their own
Open host service	• One system that has an open connection point that can be used by (many) other systems
Published language	• Let the integration be based on a common, well-defined language

Statoil

Aggregates

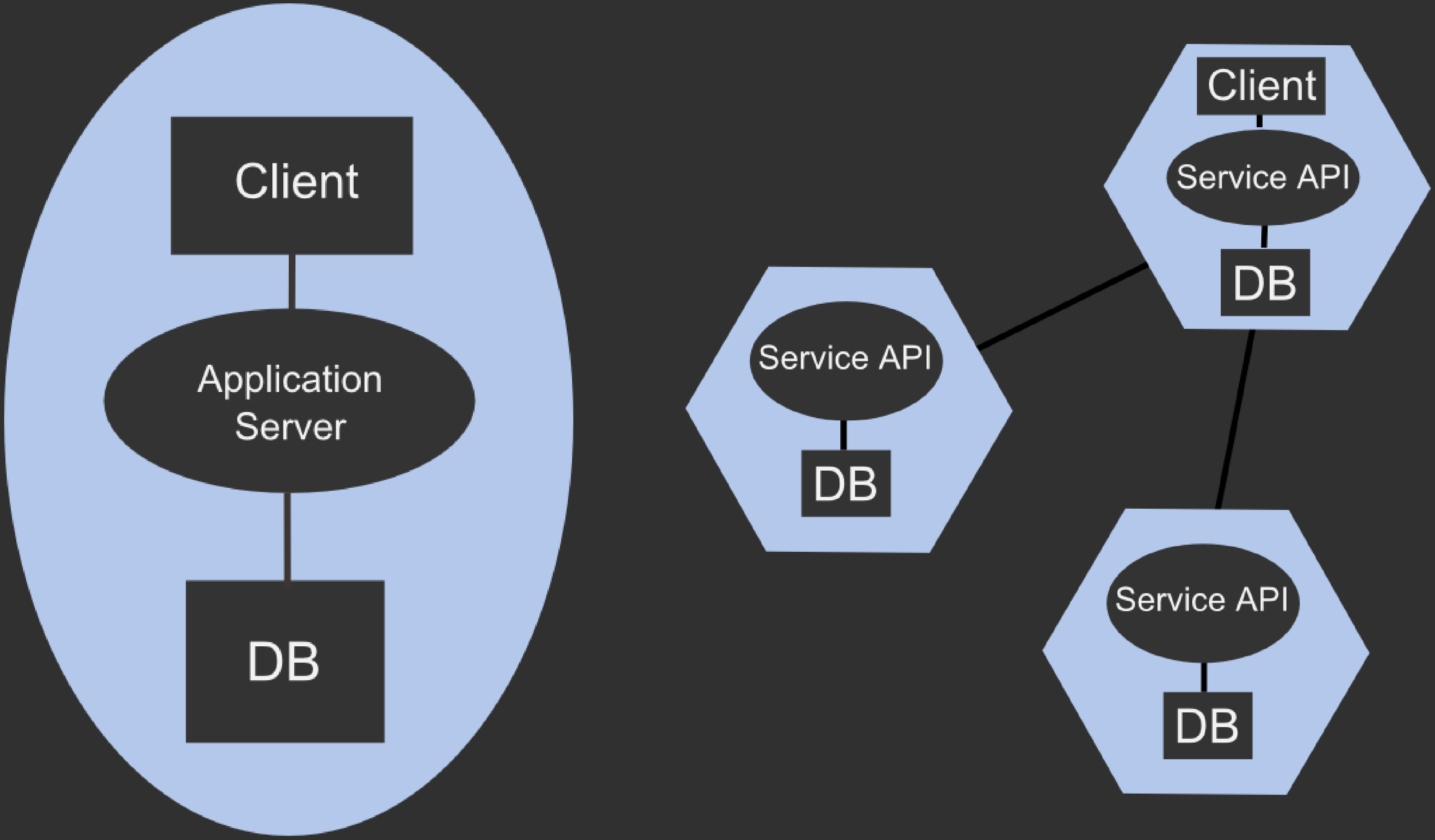


Statoil

Microservices

…is a way
of designing software
as suites of
independently deployable
services

Statoil


Client
Application Server
DB
Client
Service API
DB
Service API
DB
Service API
DB

Statoil

Independent

Team

web services

node.js

Ruby

REST

Mobility

build

Scala

Python

Architecture

governance

javascript

Linux

Skills

deployment

JSON

App Engine

Kanban

Service Management

Docker

Agile

Cloud

data management

Odata

XML

Atom Pub

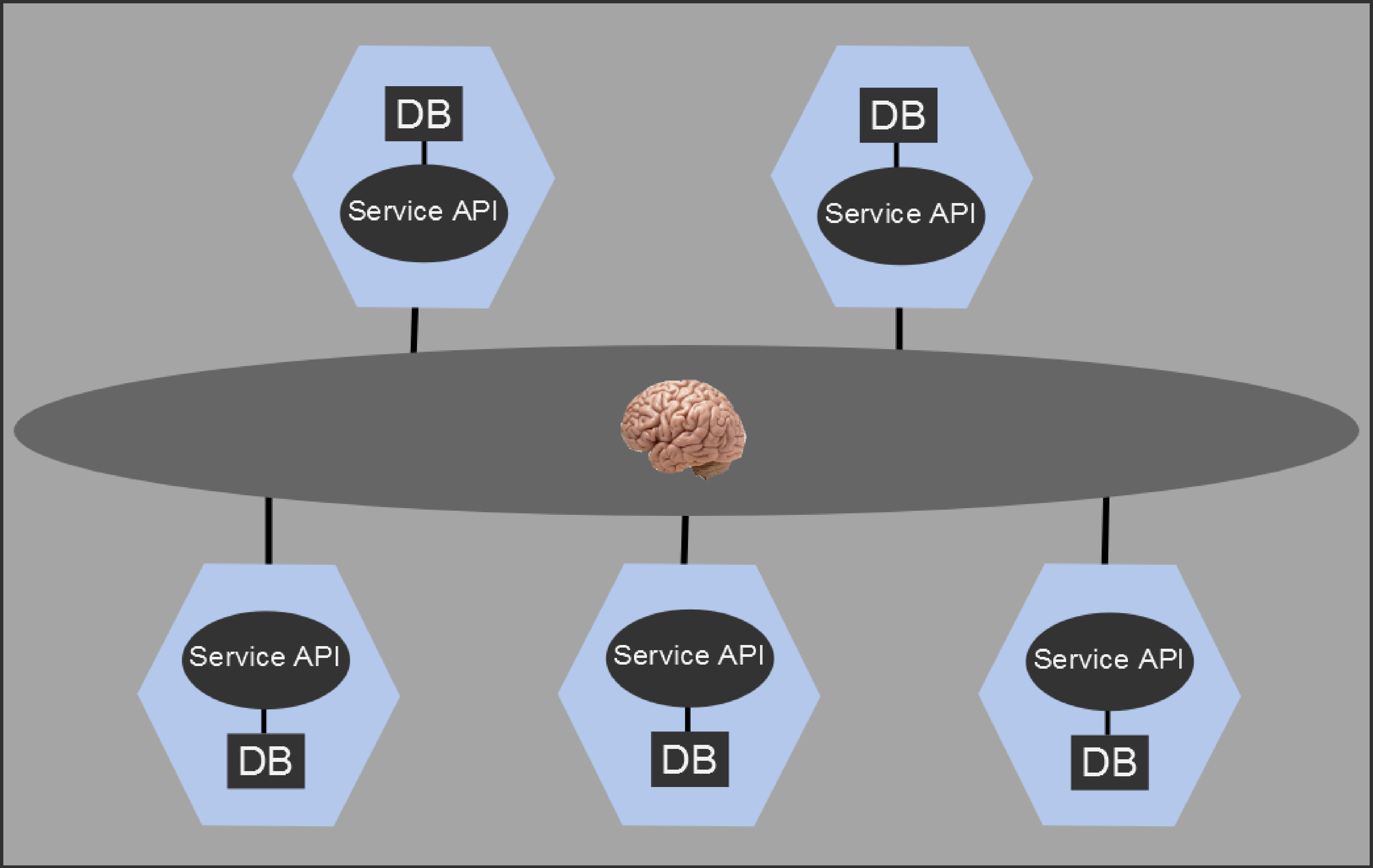
Application Development

technology choice

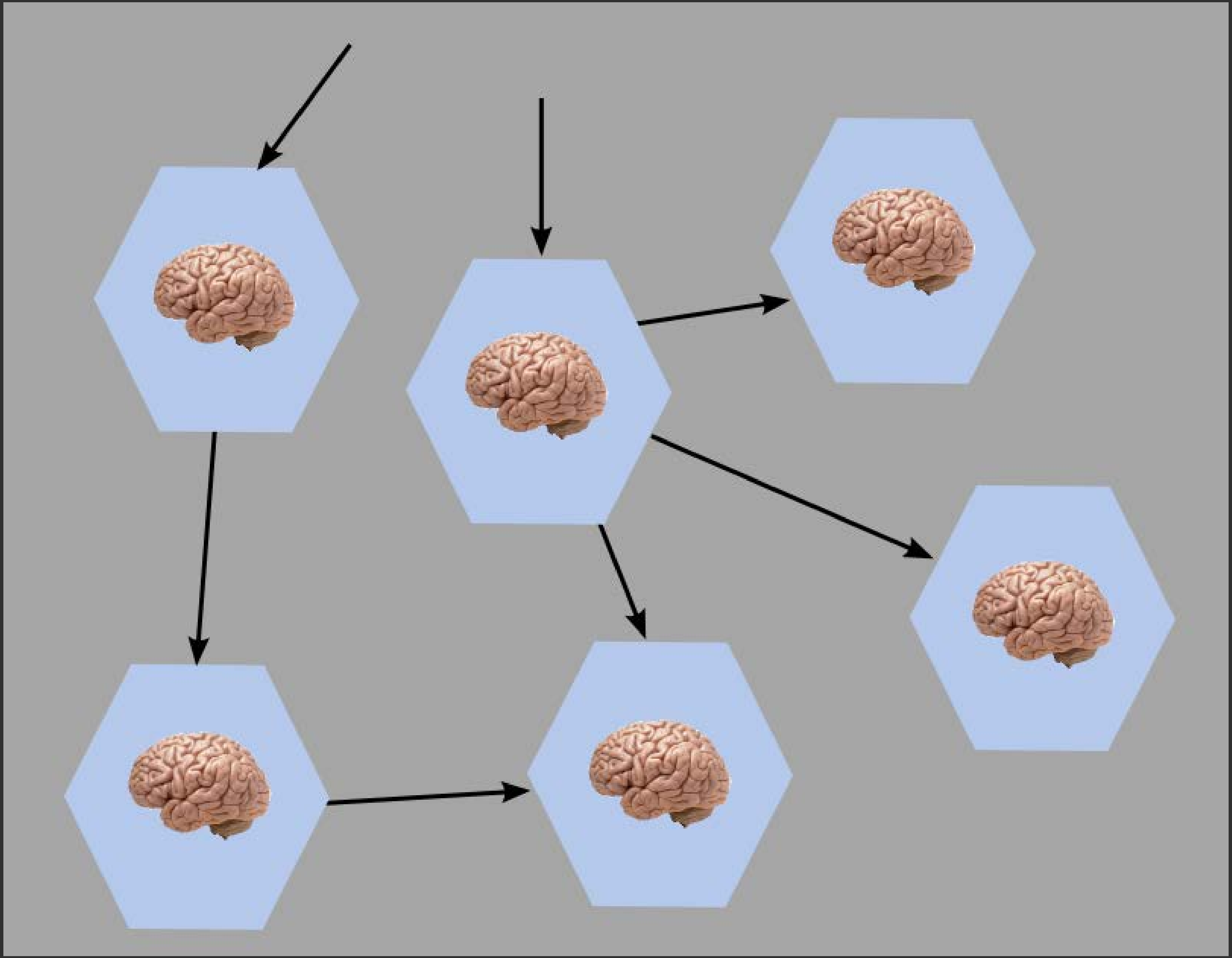
Statoil

Smart Endpoints
Dumb Pipes

Statoil


DB
Service API
DB
Service API
Service API
DB
Service API
DB
Service API
DB

Statoil



Infrastructure Automation

Statoil

Service Interfaces

OData
Open Data Protocol

{json:api}

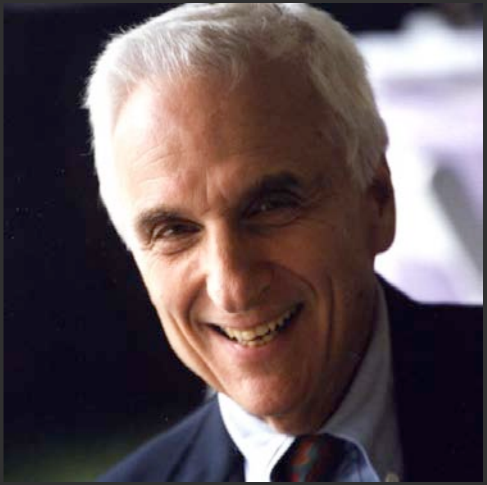
RSS
XML
ATOM

Statoil

Organized around Buisness capabilities

Statoil

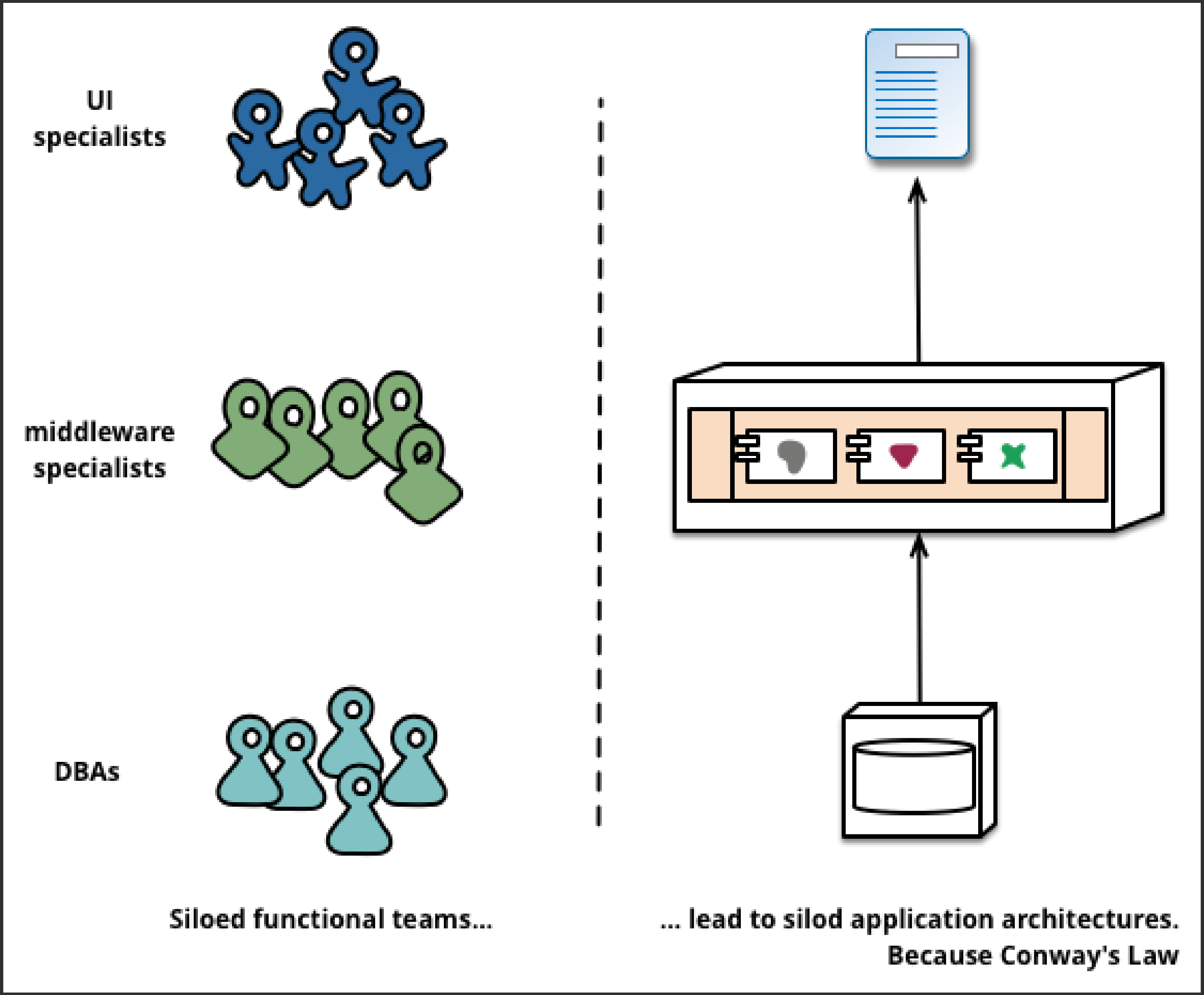
Conway's Law



Organizations which design systems … are constrained to produce designs which are copies of the communication structures of these organizations

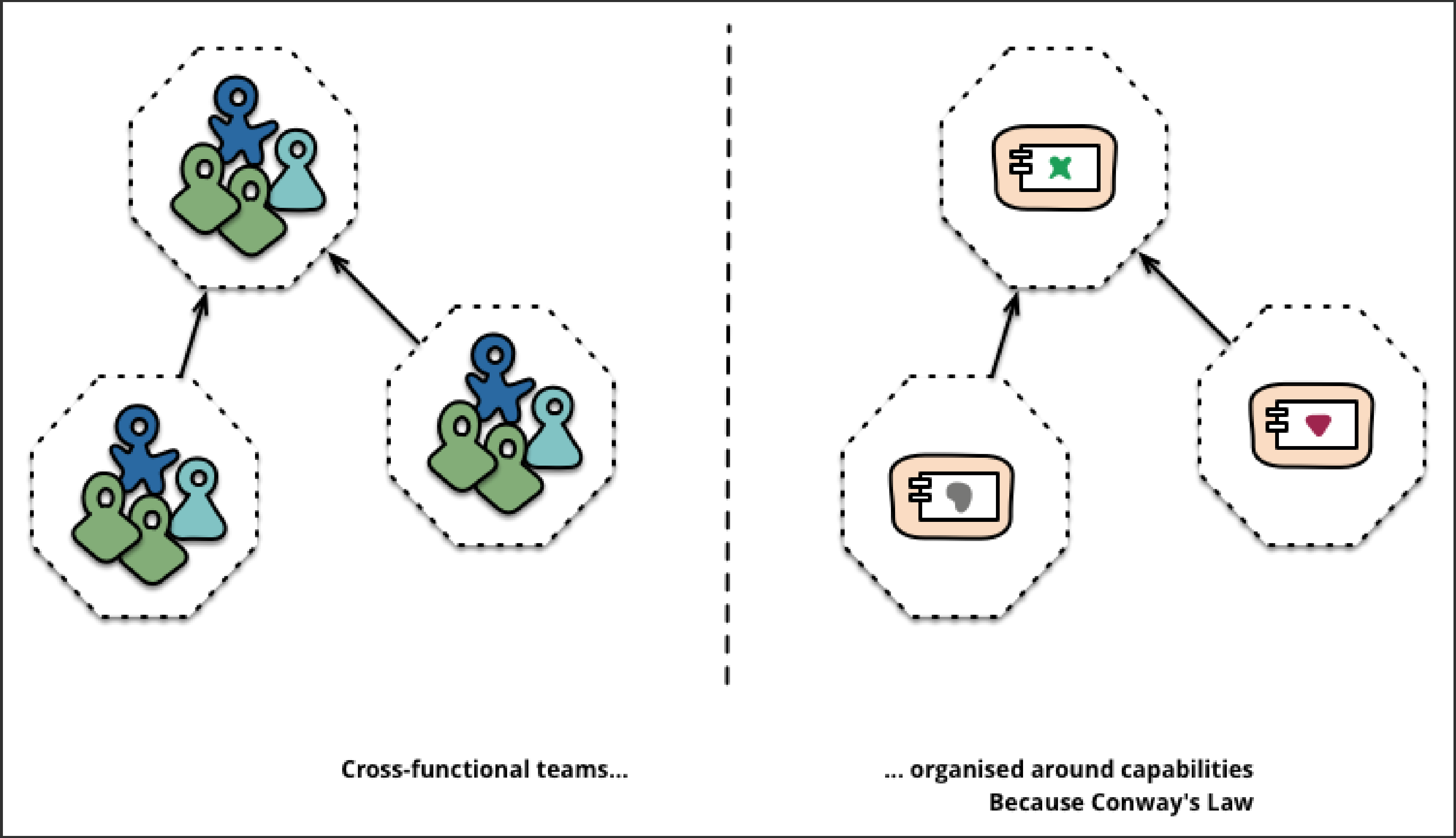
Melvin E. Conway

Statoil


UI specialists
middleware specialists
DBAs
Siloed functional teams...
... lead to silod application architectures.
Because Conway's Law

Borrowed from James Lewis and Martin Fowler's article:
http://martinfowler.com/articles/microservices.html

Inverse Conway maneuver



Borrowed from James Lewis and Martin Fowler’s article:
http://martinfowler.com/articles/microservices.html

Statoil

Important success criteria

-Rapid provisioning
-Basic monitoring
-Rapid Application Development
-DevOps Culture

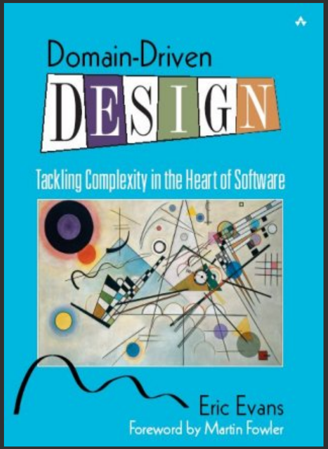
Statoil

Data driven vs Domain driven

Statoil

Discussion

Domain Driven Design is advocated as the best way


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

Still we see that the data driven approach dominate

Object oriented languages used as script languages

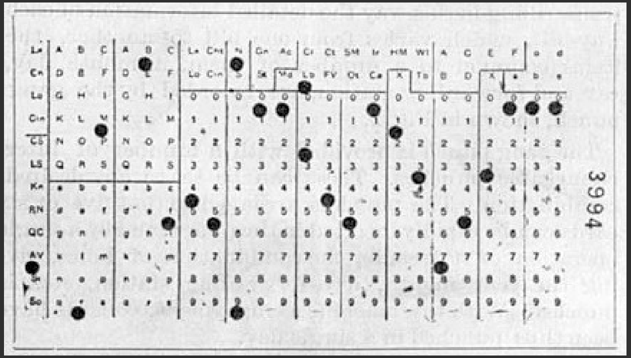
1000 or even 10.000 LOC methods are still written

Why?

Statoil

Data Driven Development

Has its origin in data processing



1890 US Census

Herman Hollerith

Punch cards for data storage

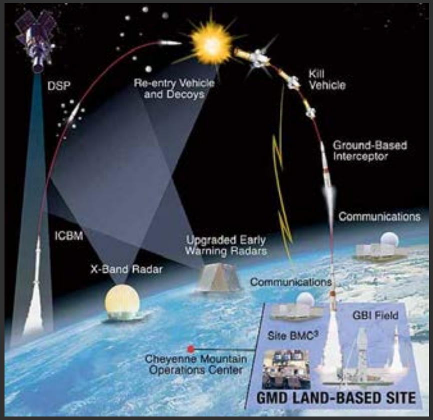
Entry, Validation, Sorting, Summarization, Aggregation, …

Electro-Mechanical machines until the 1950ties…

COBOL programming language since 1959 …

Object Oriented Development

Has its origin in simulation of dynamic systems



Interception of ICBM's

Simula 67 language

Ole Johan Dahl / Kristen Nygaard

Encapsulation of state and behaviour in "classes"

Simplifies the modelling of real-world behaviour

Smalltalk, C++, Java, C#, Scala, ...

Thoughts on DBR and its likes

Began as a data processing systems

Record and report performed operations

- Materials used
- Difficulties encountered
- Failures

With time, more and more dynamic domain's was added

Planning (re-planning)

- Automated planning
- Optimisation
- Monitoring

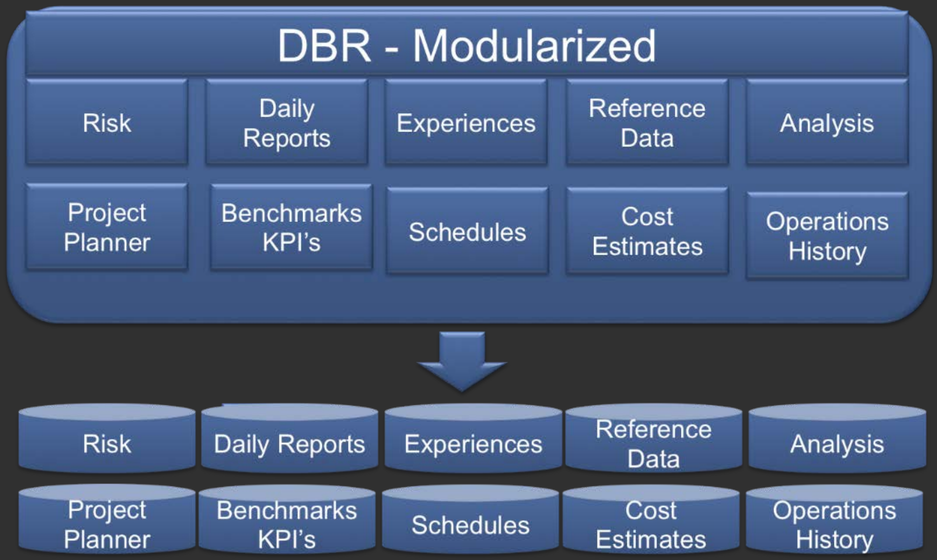
Scheduling

- Cost function
- Automatic re-scheduling
- Optimisation

Dynamic domains are addressed with a data driven approach

Statoil

Micro-services to the rescue?



Planning & Scheduling

- Automated planning
- Multi-agent

Analysis

- R for statistics

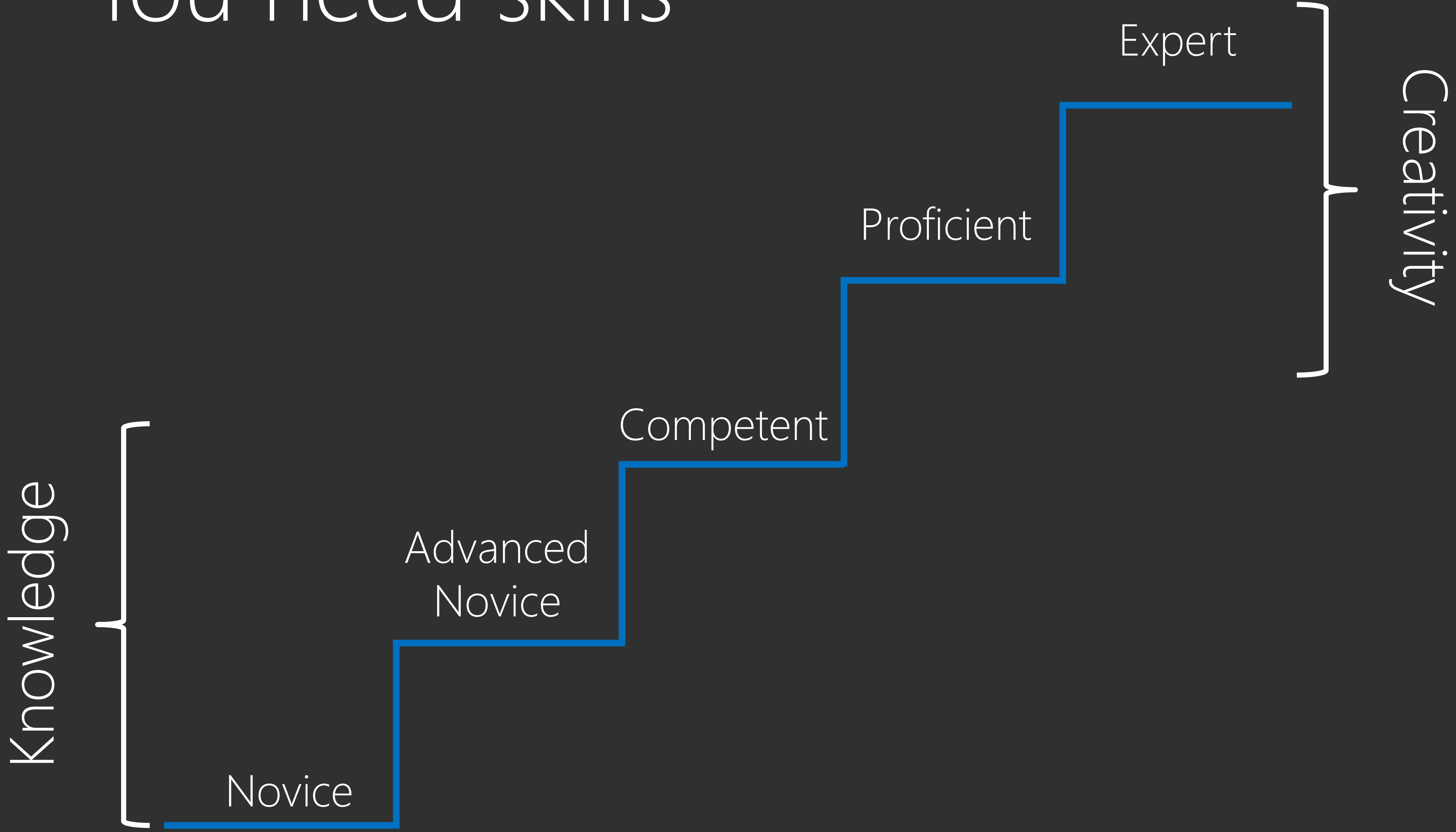
Daily reports

- Data driven

Each service can be implemented with the most suitable technology

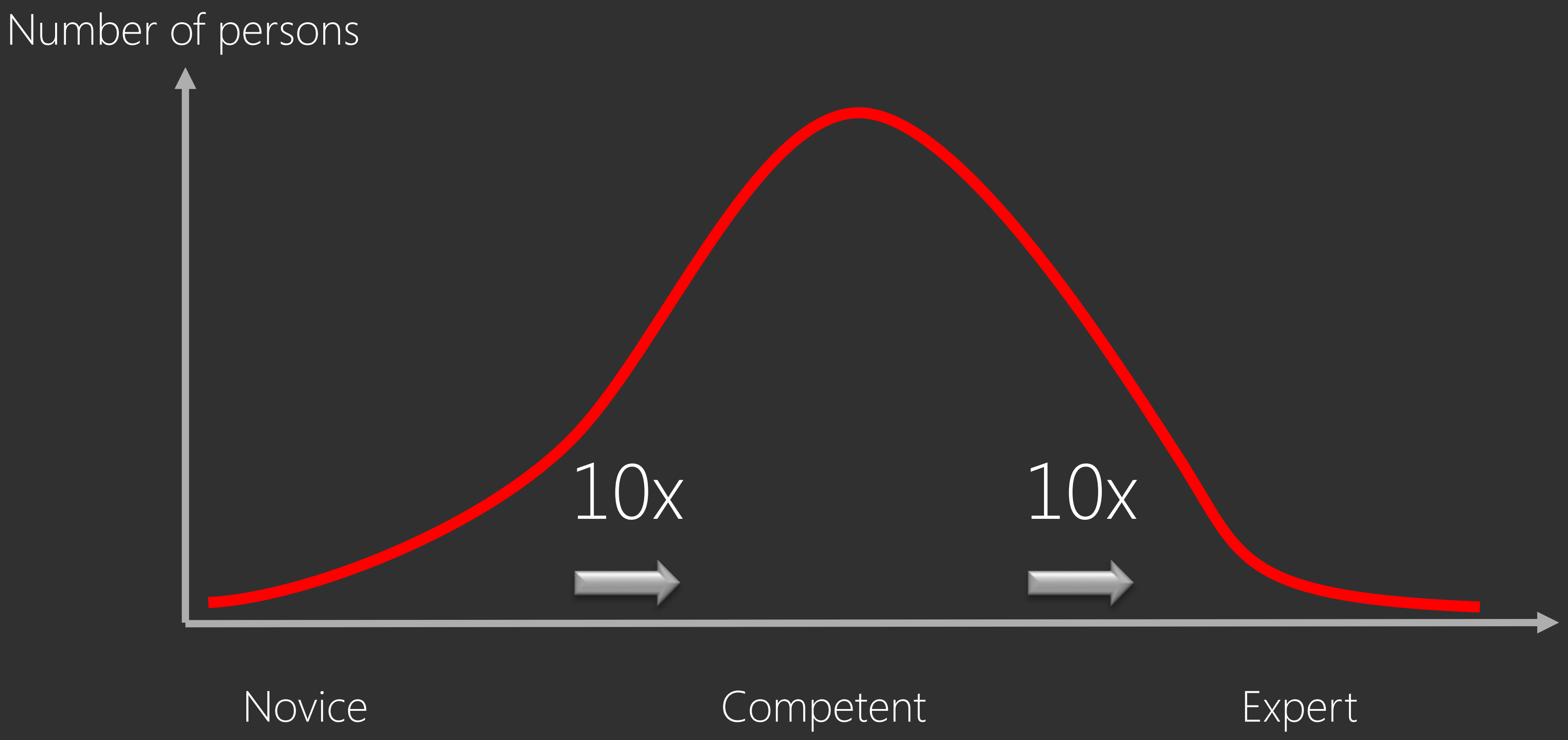
Statoil

You need skills



Statoil

Skills and productivity

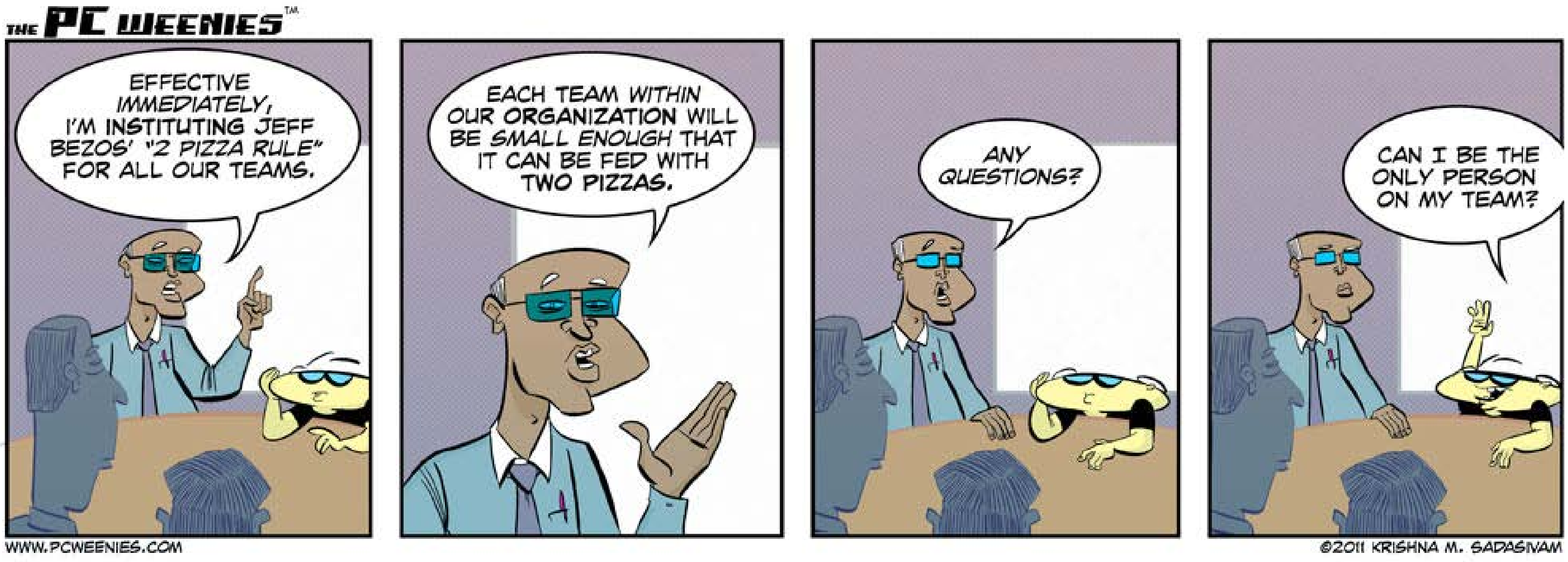


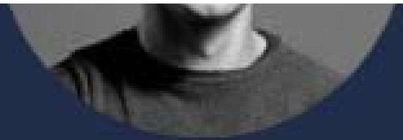
Statoil

Organization

Statoil

Working in


THE PC WEENIES™
EFFECTIVE IMMEDIATELY, I'M INSTITUTING JEFF BEZOS' "2 PIZZA RULE" FOR ALL OUR TEAMS.
EACH TEAM WITHIN OUR ORGANIZATION WILL BE SMALL ENOUGH THAT IT CAN BE FED WITH TWO PIZZAS.
ANY QUESTIONS?
CAN I BE THE ONLY PERSON ON MY TEAM?
WWW.PCWEENIES.COM
©2011 KRISHNA M. SADASIVAM



large.

- Jeff Bezos, CEO, Amazon

Our Team

Small (3-5) over very long time

- +15 years
- Now two teams 6+4, two locations

Technologically segregated

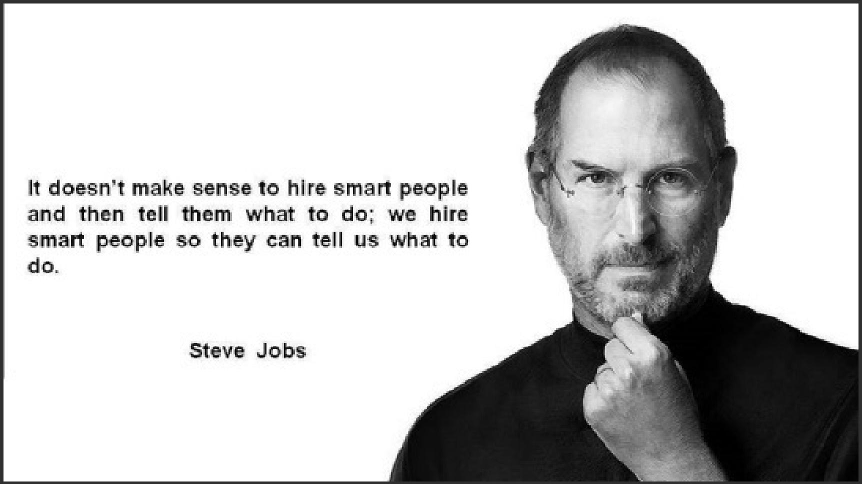
- Database
- Power Builder
- Web (Microsoft Stack)

Statoil

Why have we not succeded?

Statoil

Leadership


It doesn't make sense to hire smart people and then tell them what to do; we hire smart people so they can tell us what to do.

Steve Jobs

Good software leaders are rare

- How to nurture talents?
- How to develop the needed skills?

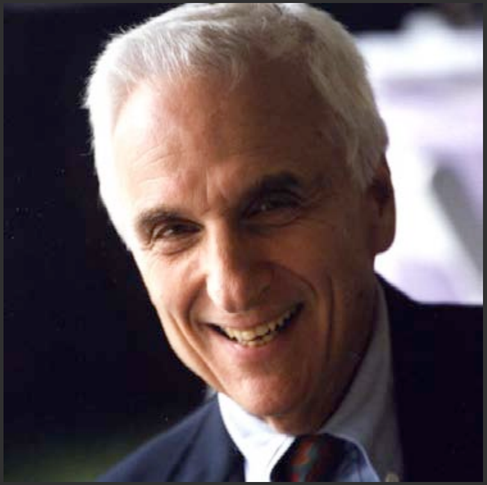
Leading from the front or back?

- How to build trust?

How to ensure individuals pulls as a team?

Statoil

Conway’s Law



Organizations which design systems … are constrained to produce designs which are copies of the communication structures of these organizations

Melvin E. Conway

Our Team

Not cross functional
- Database
- Power Builder
- Web (Microsoft Stack)

Not co-located
- Stavanger
- Bergen

Vulnerable
- Dependent on individuals
- Number of years to retirement

Statoil

Our company

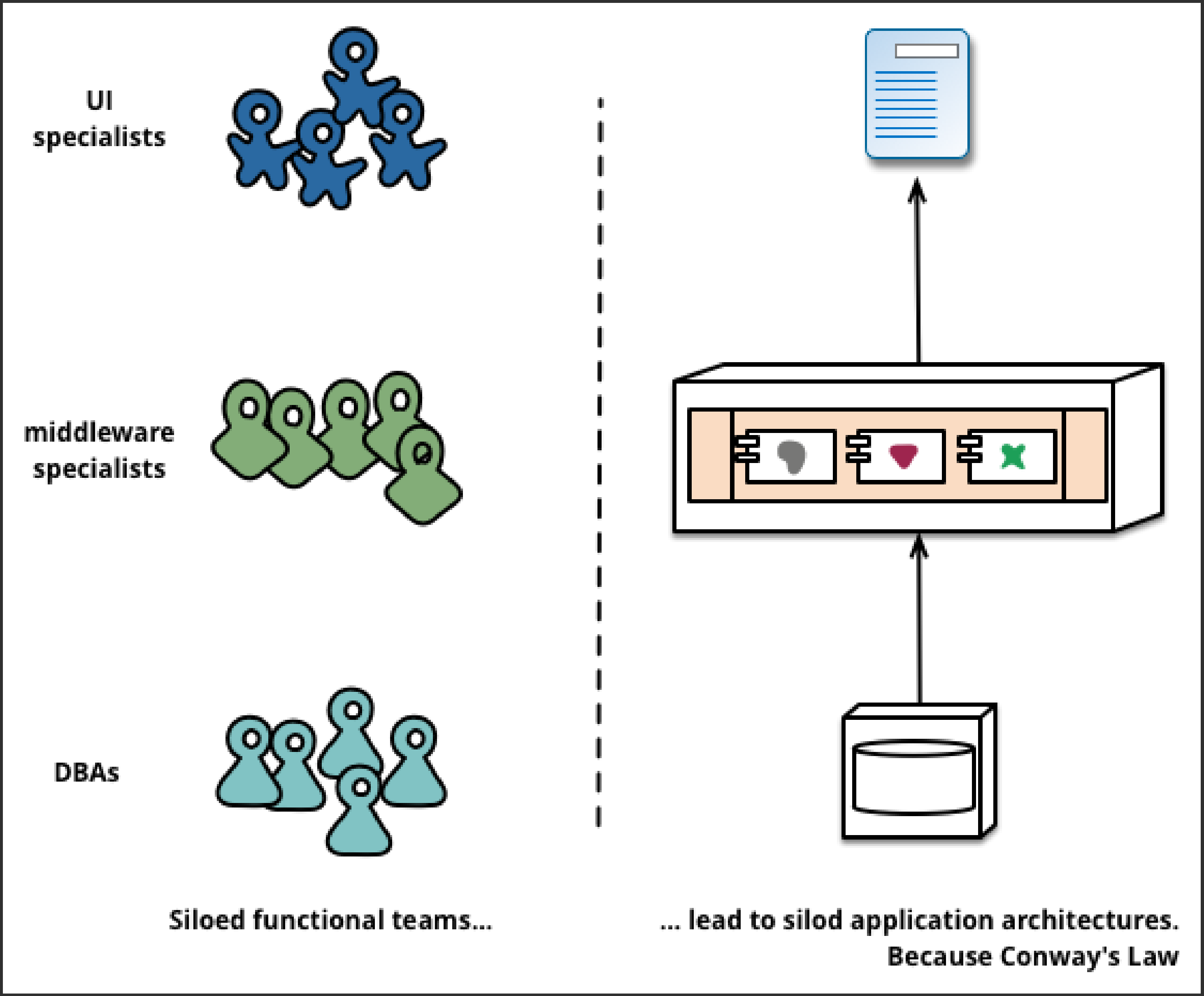
Large enterprise organization

- Divisional structure
- Multinational
- Central IT governance

Lacking

- DevOps culture
- Infrastructure automation

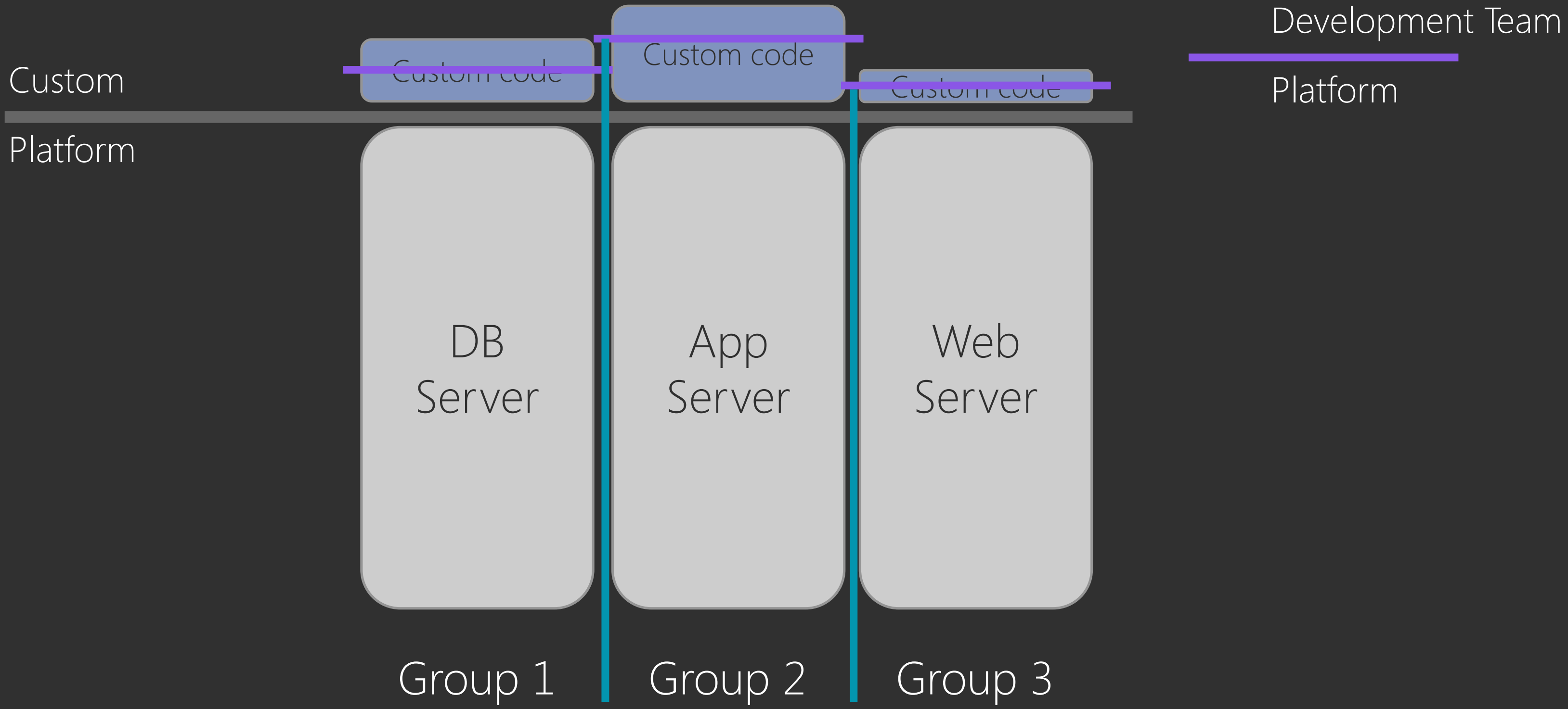
Statoil


UI specialists
middleware specialists
DBAs
Siloed functional teams...
... lead to silod application architectures.
Because Conway's Law

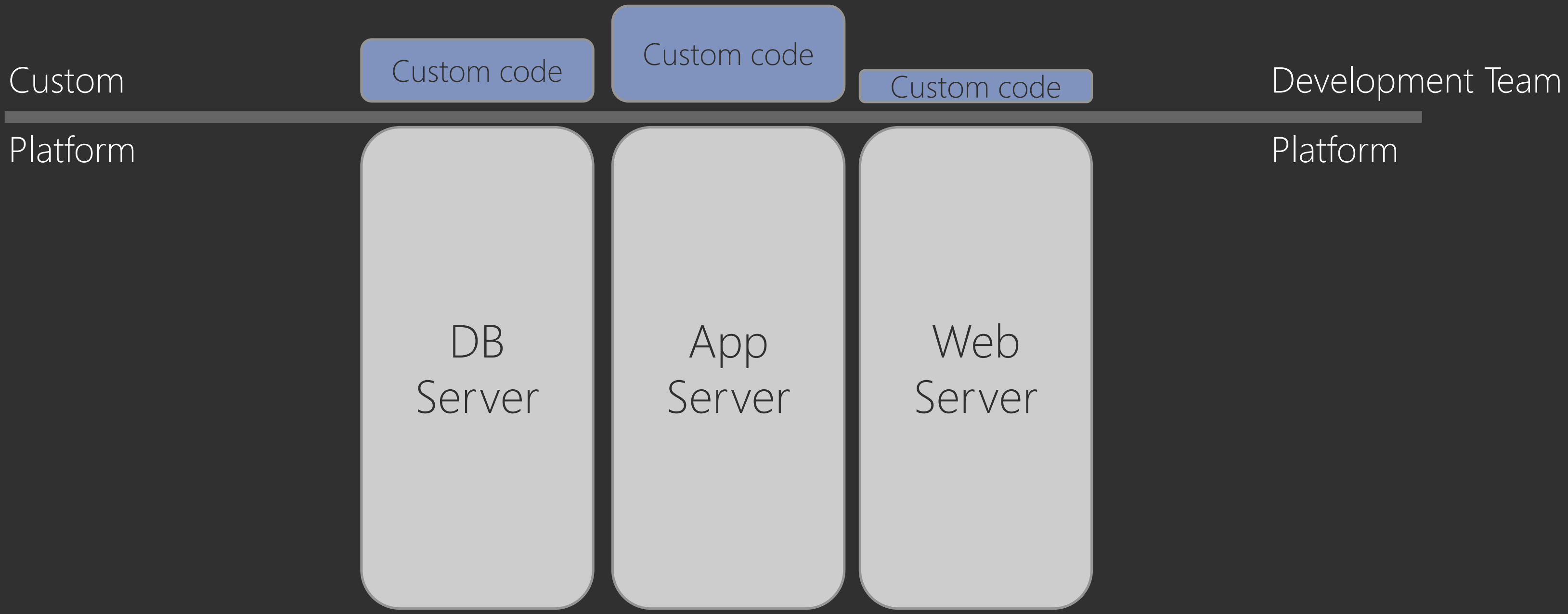
Borrowed from James Lewis and Martin Fowler's article:
http://martinfowler.com/articles/microservices.html

Statoil

Actors = 4


Custom
Platform
Custom code
Custom code
Custom code
DB Server
App Server
Web Server
Group 1
Group 2
Group 3
Development Team
Platform

Actors = 1


Custom
Platform
Custom code
Custom code
Custom code
DB Server
App Server
Web Server
Development Team
Platform

Breaking the Monolith

How do you eat an elephant?

one bite at a time

Statoil

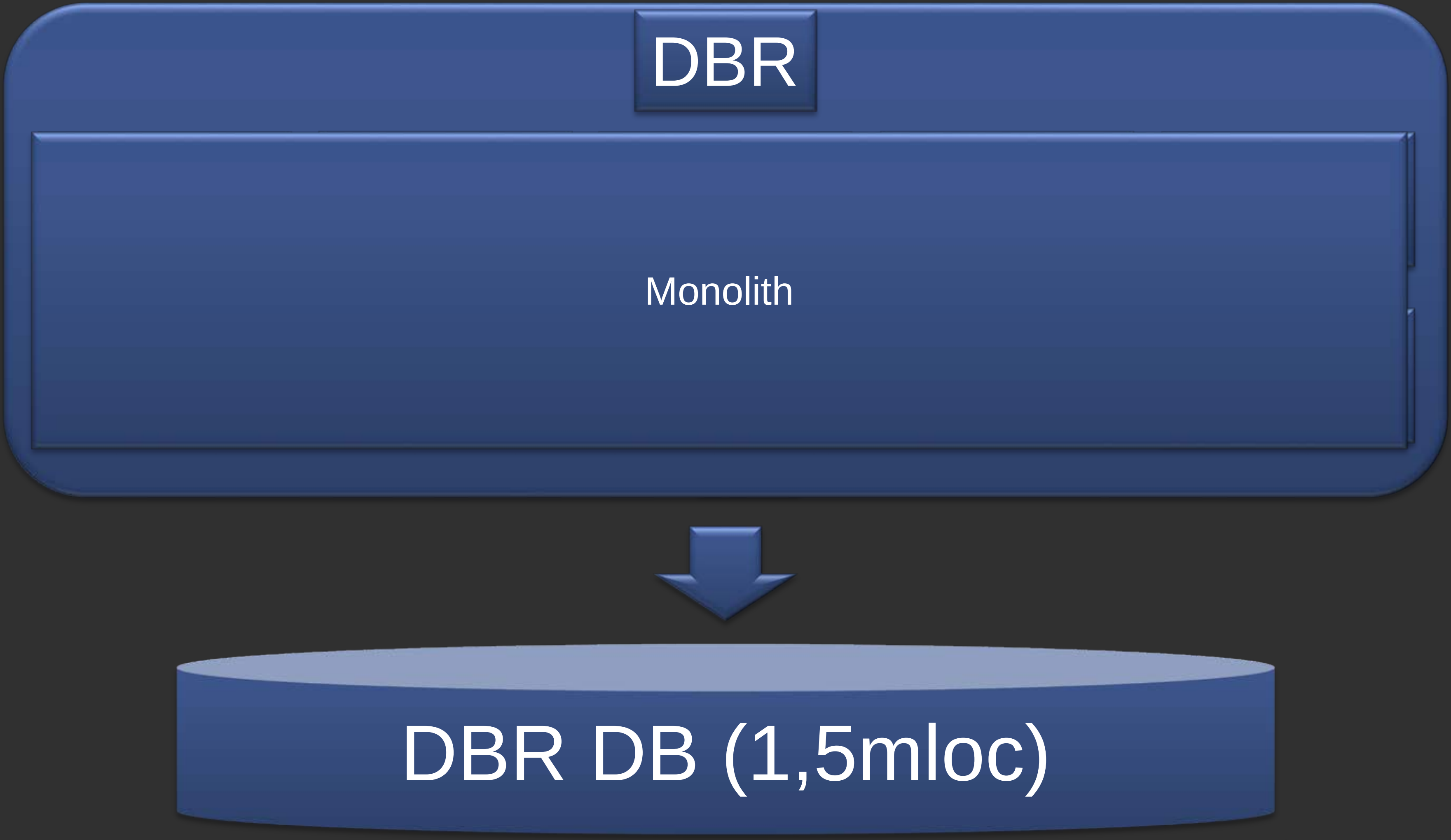
Goals

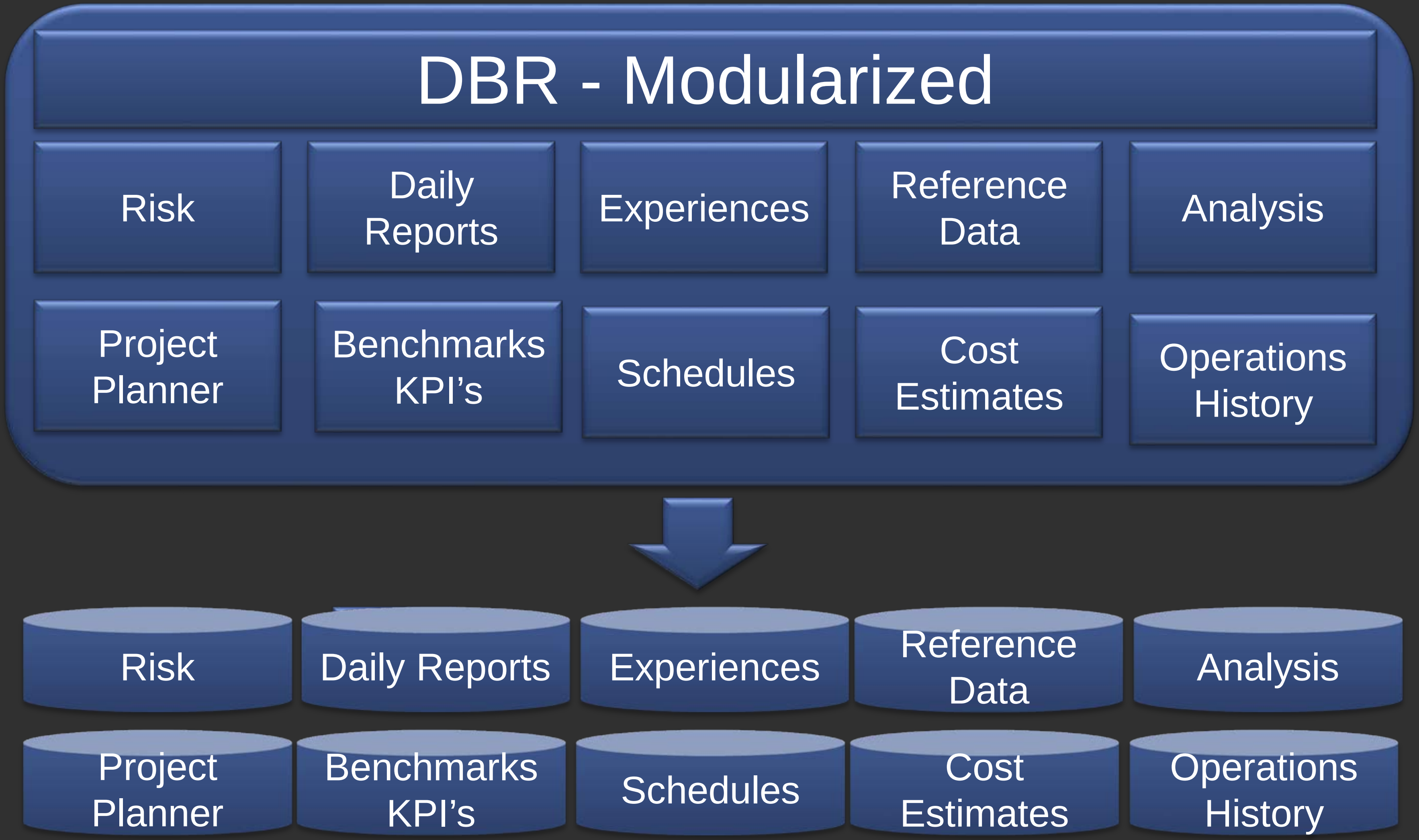
1. Make it easier to implement new features
2. Make stored data more easily available
3. Simplify build and deployment
4. Modernize technology stack

In short: Optimize delivery of new features and availability of data

Statoil

Bounded contexts





Functionality integrated at each module level as services

- Internal bus for DBR functionality
- Services for external data

Statoil

How do we get there?

Statoil

Making changes

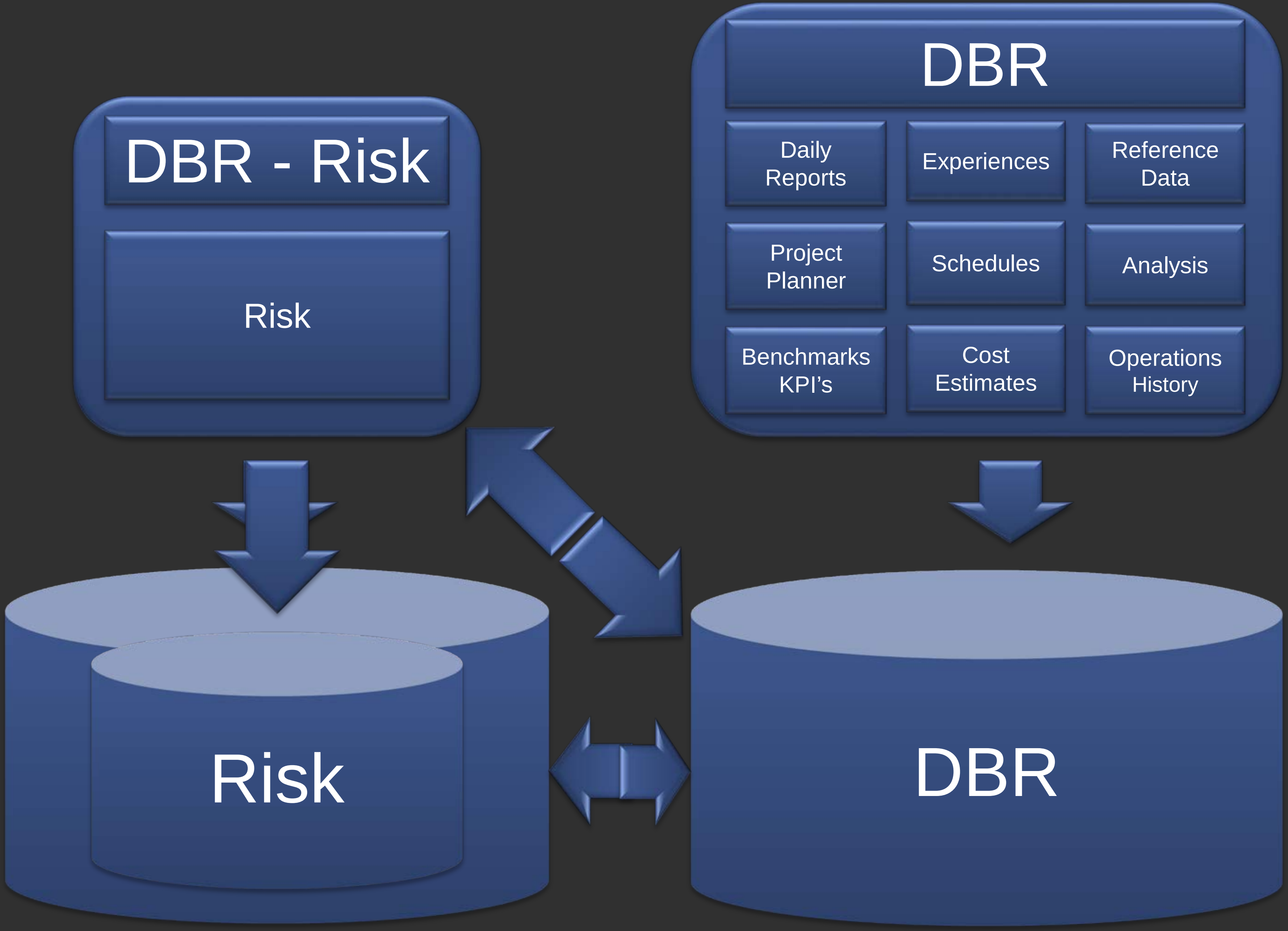
Statoil

Extracting a bounded context

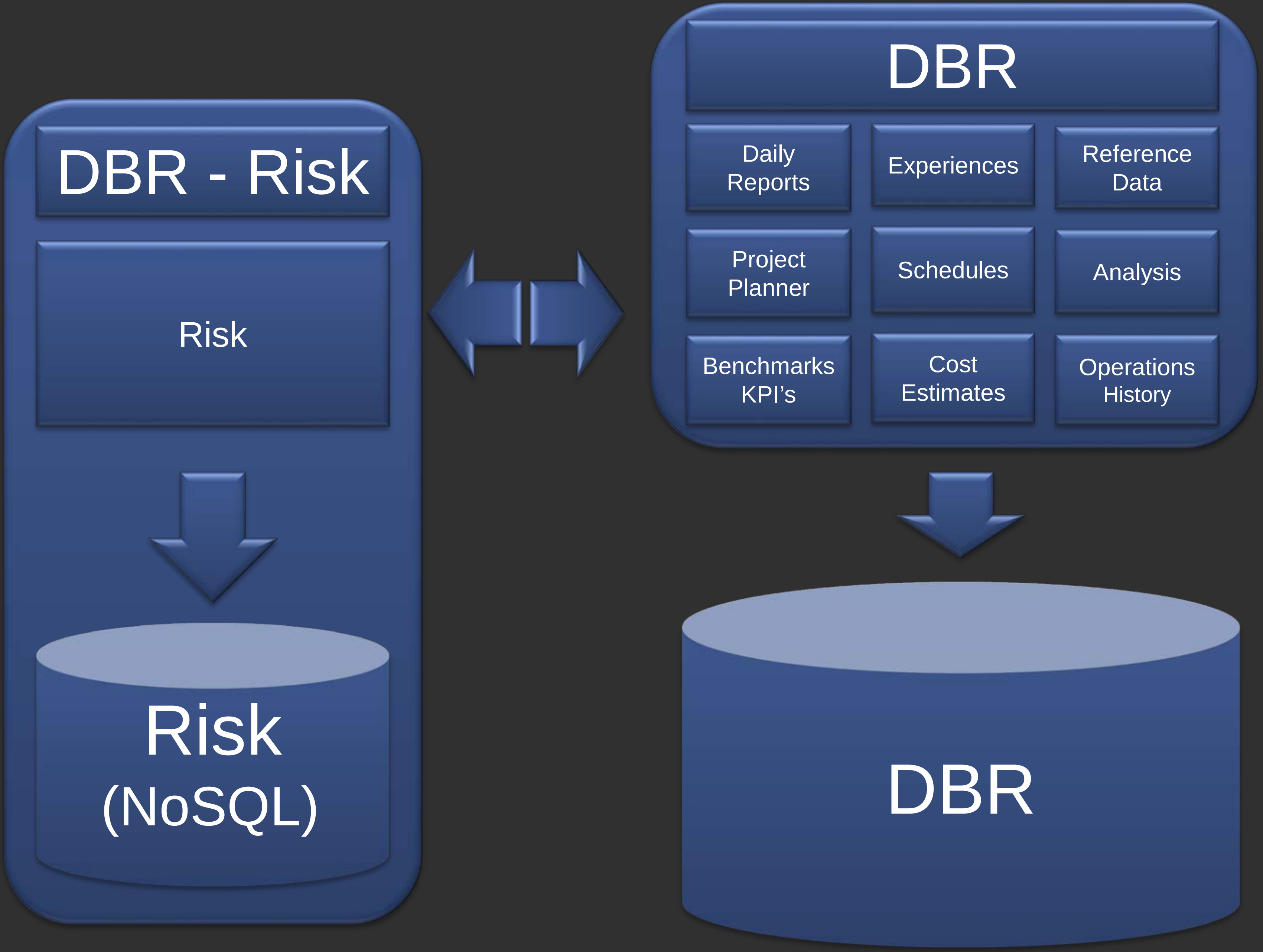
Statoil

Database tactics

1. Duplicate databases, replicate data
2. Duplicate schemas
3. Views

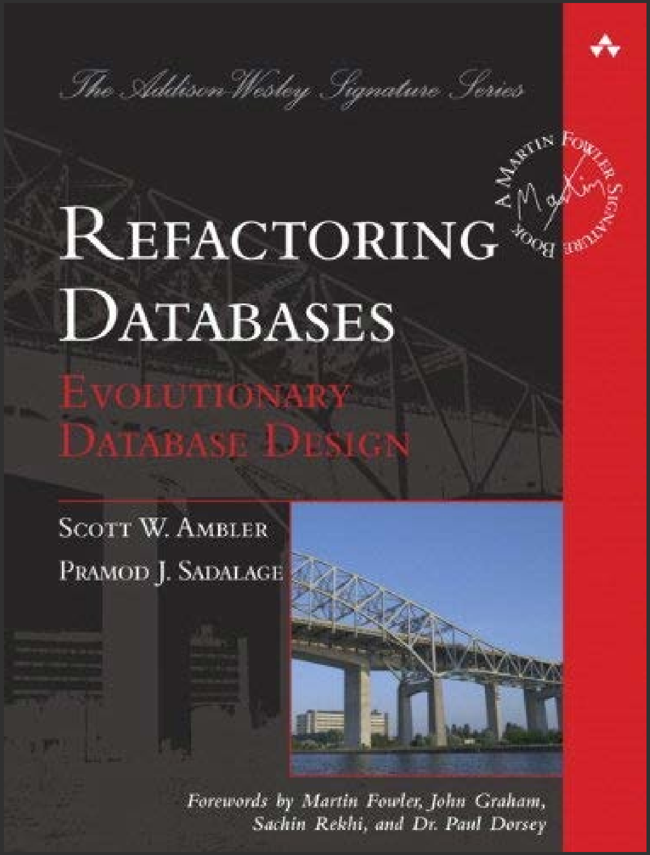

DBR - Risk
Risk
DBR
Daily Reports
Experiences
Reference Data
Project Planner
Schedules
Analysis
Benchmarks KPI's
Cost Estimates
Operations History
Risk
DBR

Statoil


DBR - Risk
Risk
Risk
(NoSQL)
DBR
Daily Reports
Experiences
Reference Data
Project Planner
Schedules
Analysis
Benchmarks KPI's
Cost Estimates
Operations History
DBR

Statoil

Database refactoring



The Addison-Wesley Signature Series

A Martin Fowler Signature Book

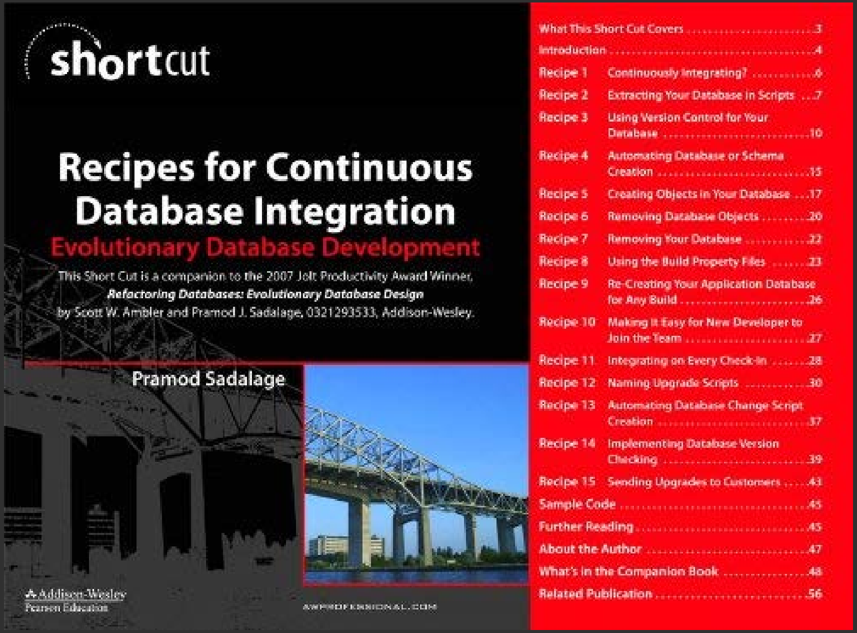
REFACTORING DATABASES

EVOLUTIONARY DATABASE DESIGN

SCOTT W. AMBLER

PRAMOD J. SADALAGE

Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey



shortcut

Recipes for Continuous Database Integration

Evolutionary Database Development

This Short Cut is a companion to the 2007 Jolt Productivity Award Winner, ***Refactoring Databases: Evolutionary Database Design*** by Scott W. Ambler and Pramod J. Sadalage, 0321293533, Addison-Wesley.

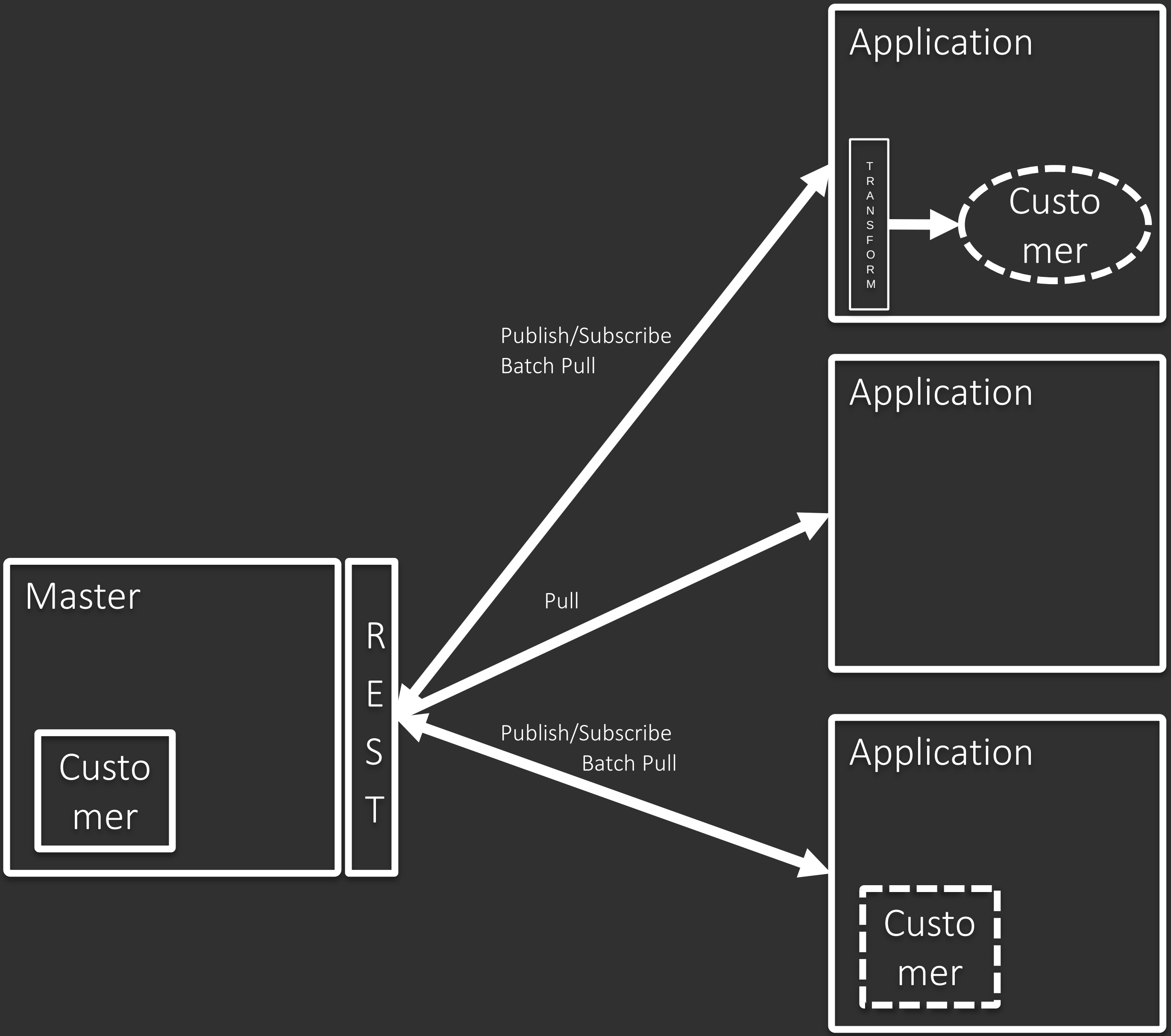
Pramod Sadalage

Addison-Wesley
Pearson Education

AWPROFESSIONAL.COM

Statoil

Master data


Application
TRANSFORM
Customer
Publish/Subscribe
Batch Pull
Application
Pull
Master
REST
Customer
Publish/Subscribe
Batch Pull
Application
Customer

Statoil

Summary

- Data-driven monolithic apps
- Change and adapt your organization
- Bounded contexts as Microservices
- Build domain modelling skills
- Leadership is a critical success factor

Statoil

Thank you

Statoil