



GOTO Berlin 2019

"Good Enough" Architecture

Stefan Tilkov
stefan.tilkov@innoq.com
@stilkov

INNOQ

(Software) Architecture Definitions

Fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution (ISO 42010)

Whatever the architect considers important enough to merit their attention

Architecture represents the significant design decisions that shape a system, where significant is measured by cost of change (Grady Booch)

Architecture is not an upfront activity performed by somebody in charge of telling everyone else what to do

Architecture is a property of a system, not a description of its intended design

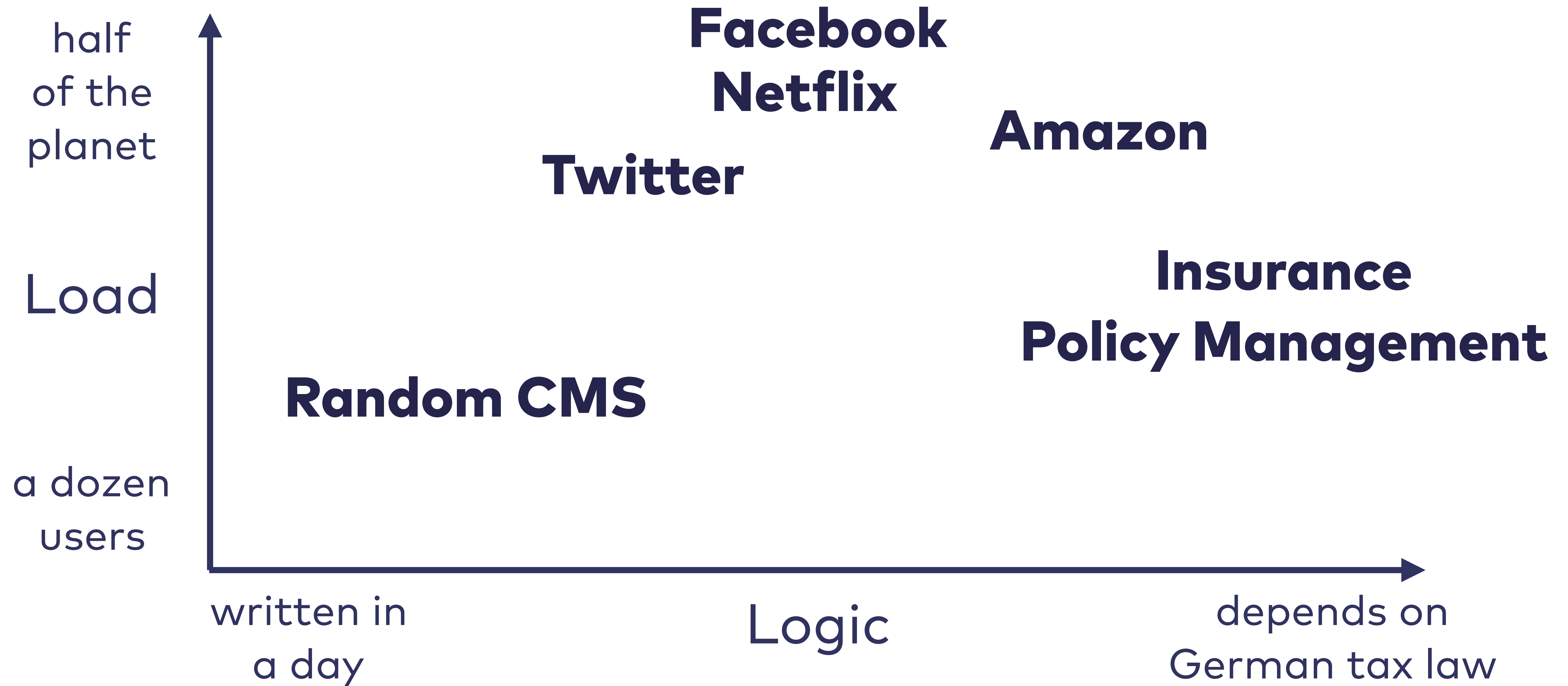
Pick the best car:



Quality



Scaling Dimensions



There is no "good" or "bad" architecture without context; architecture needs to take specific quality attributes into account

Cases

Context:

- ...

Observation(s):

- ...

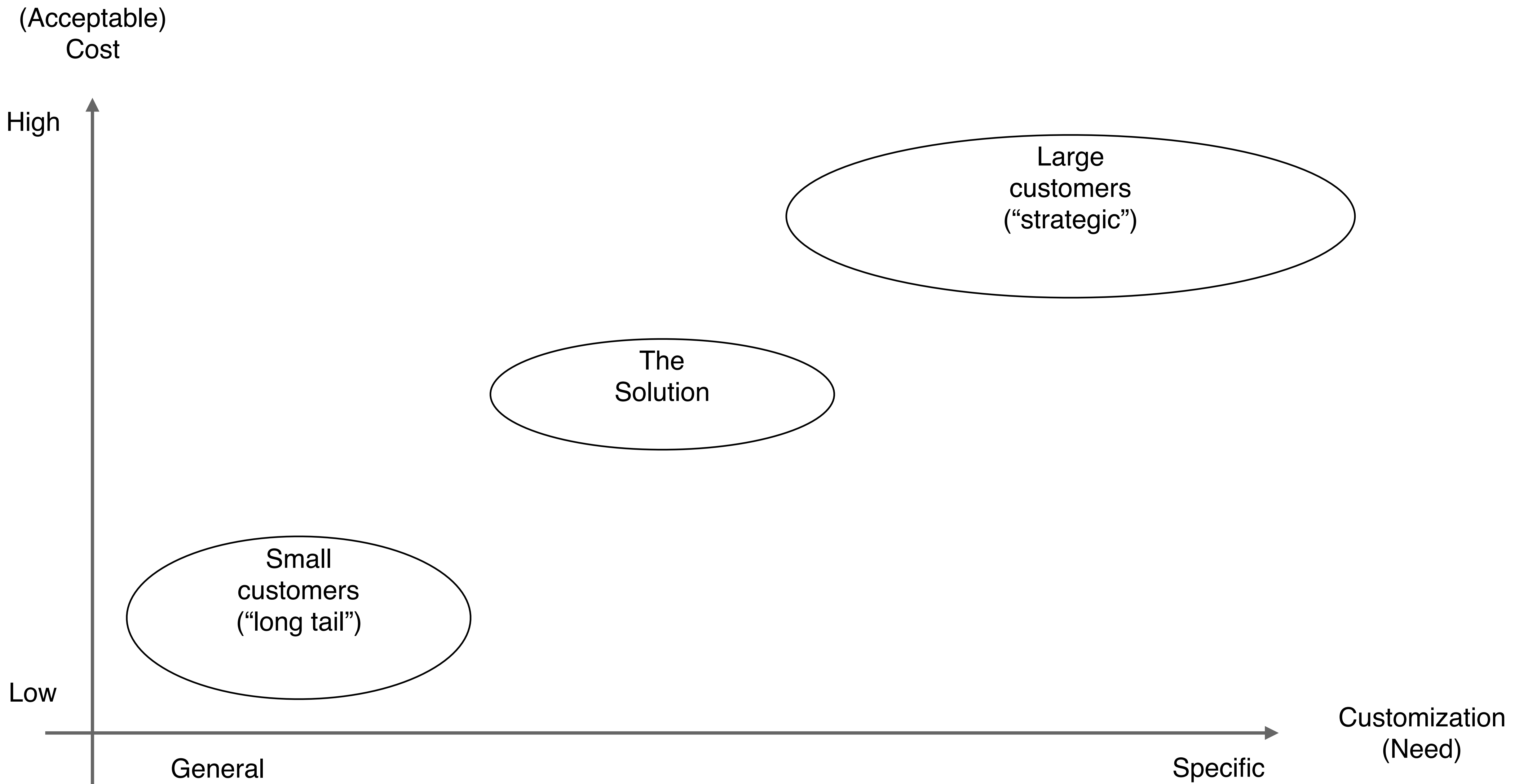
Lesson(s) learned:

- ...

#1: Non-extensible Extensibility

Context

- **E-Commerce (retail) provider**
- **Global customer base**
- **Catalog/CMS/Shop/Fulfillment**
- **Multi-tenant**
- **Highly customizable**



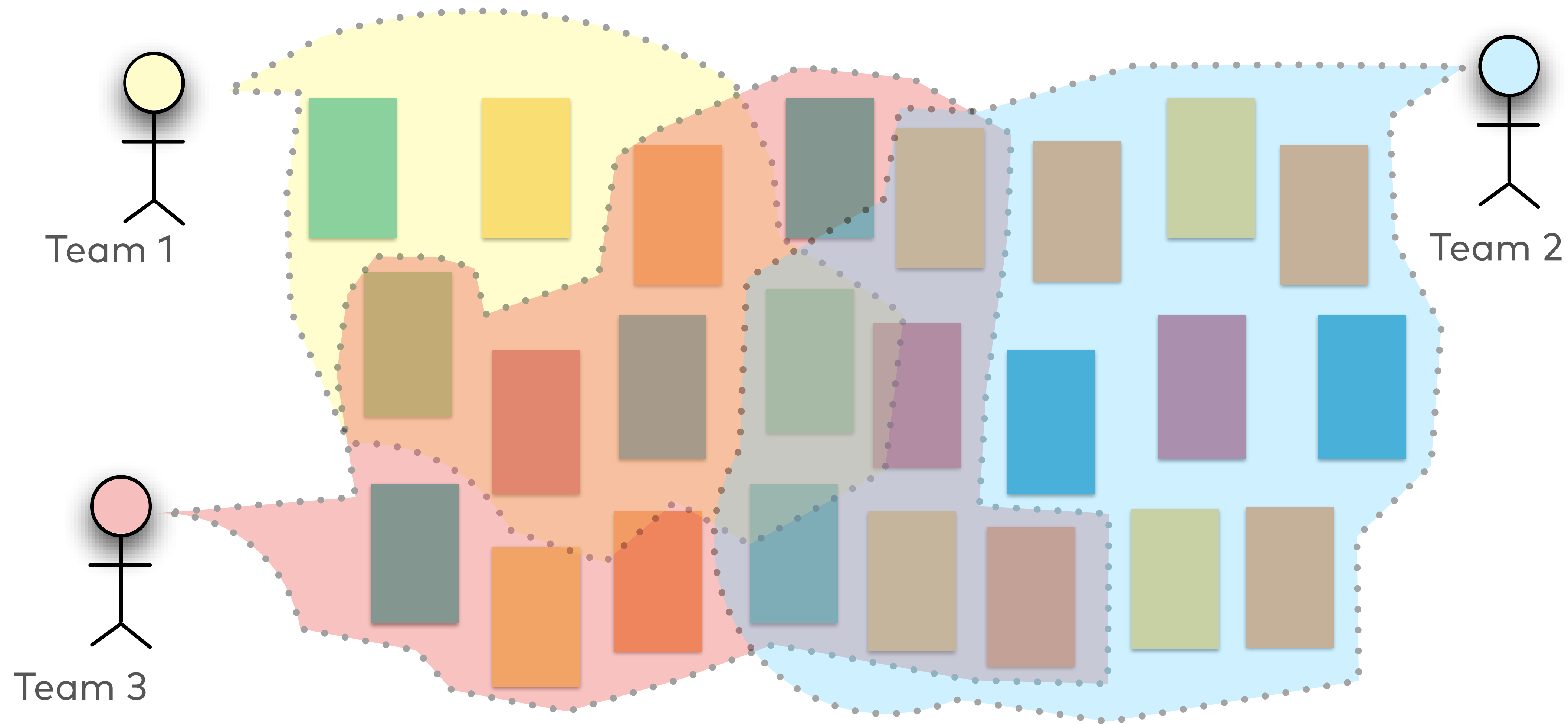
If your design attempts to satisfy everyone, you'll likely end up satisfying no one

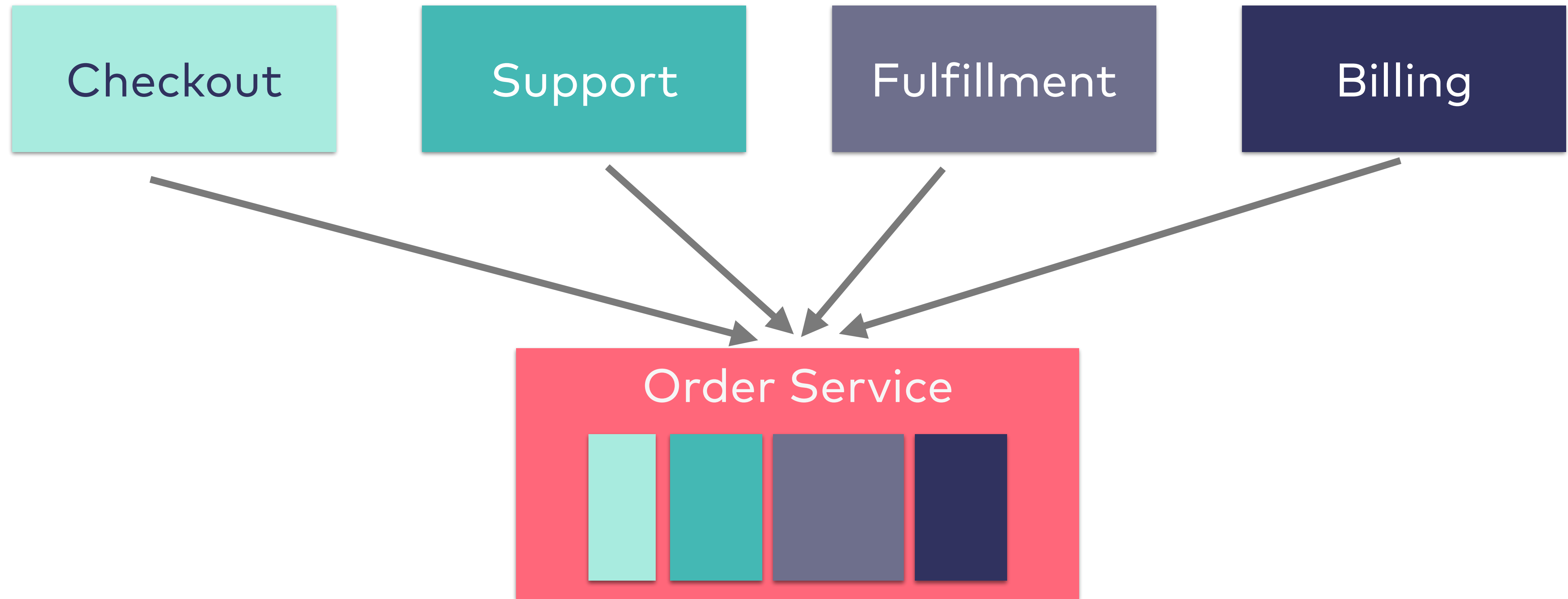
Highly specific code is often preferable to sophisticated configuration


#2: Perilously fine-grained

Context

- **Large-scale B2B food retailer**
- **New company-wide shop and logistics system**
- **>200 developers**

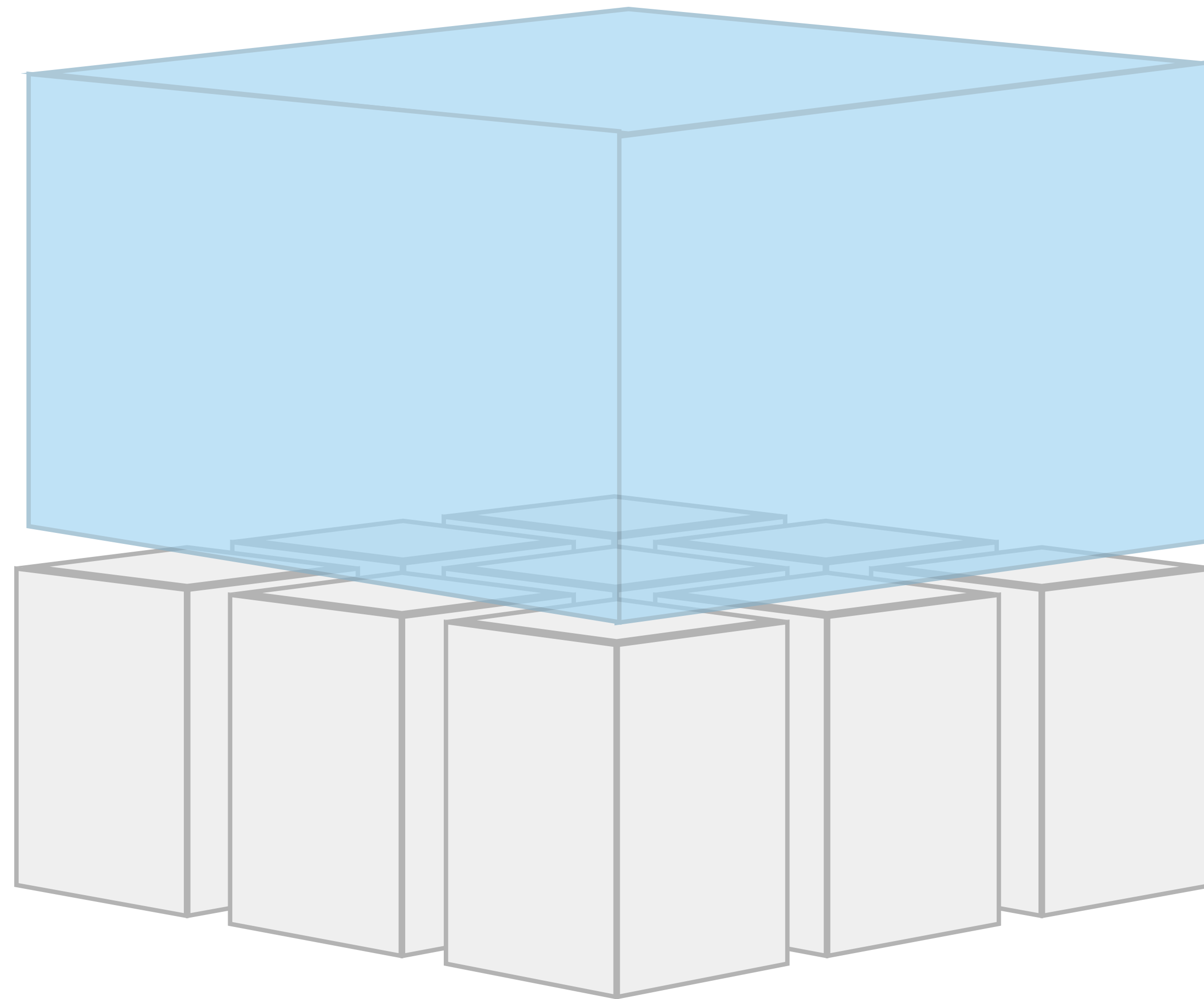


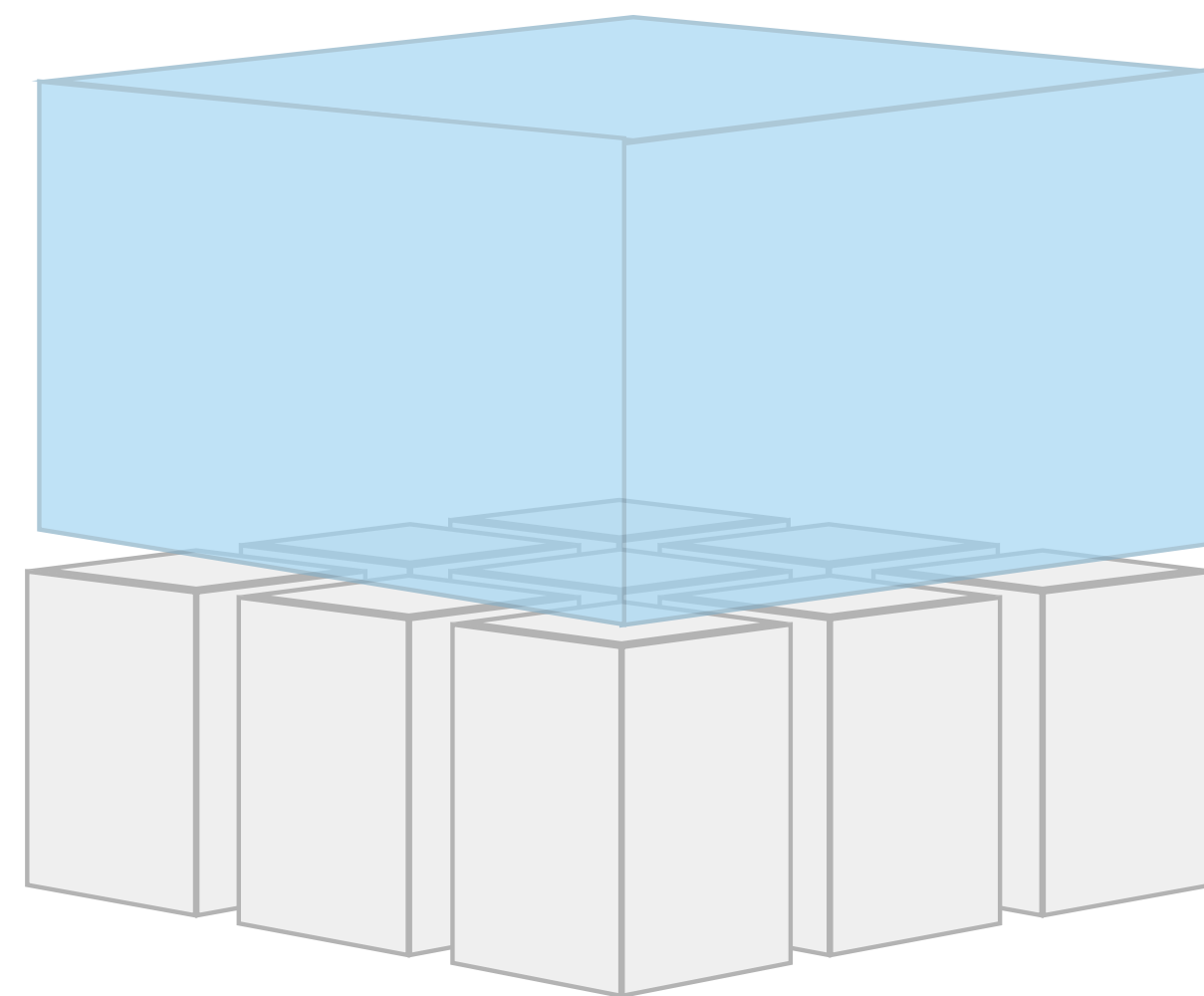
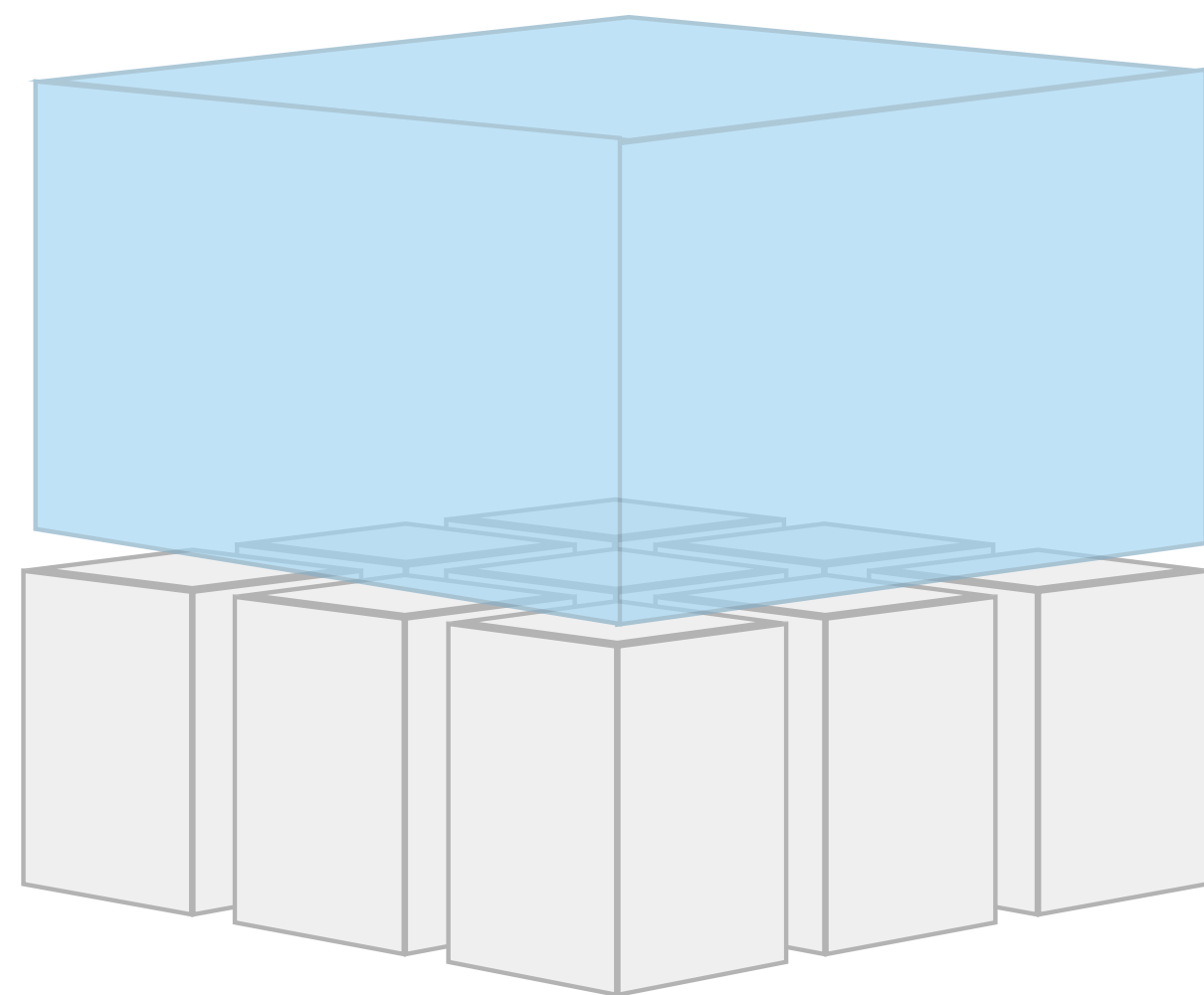




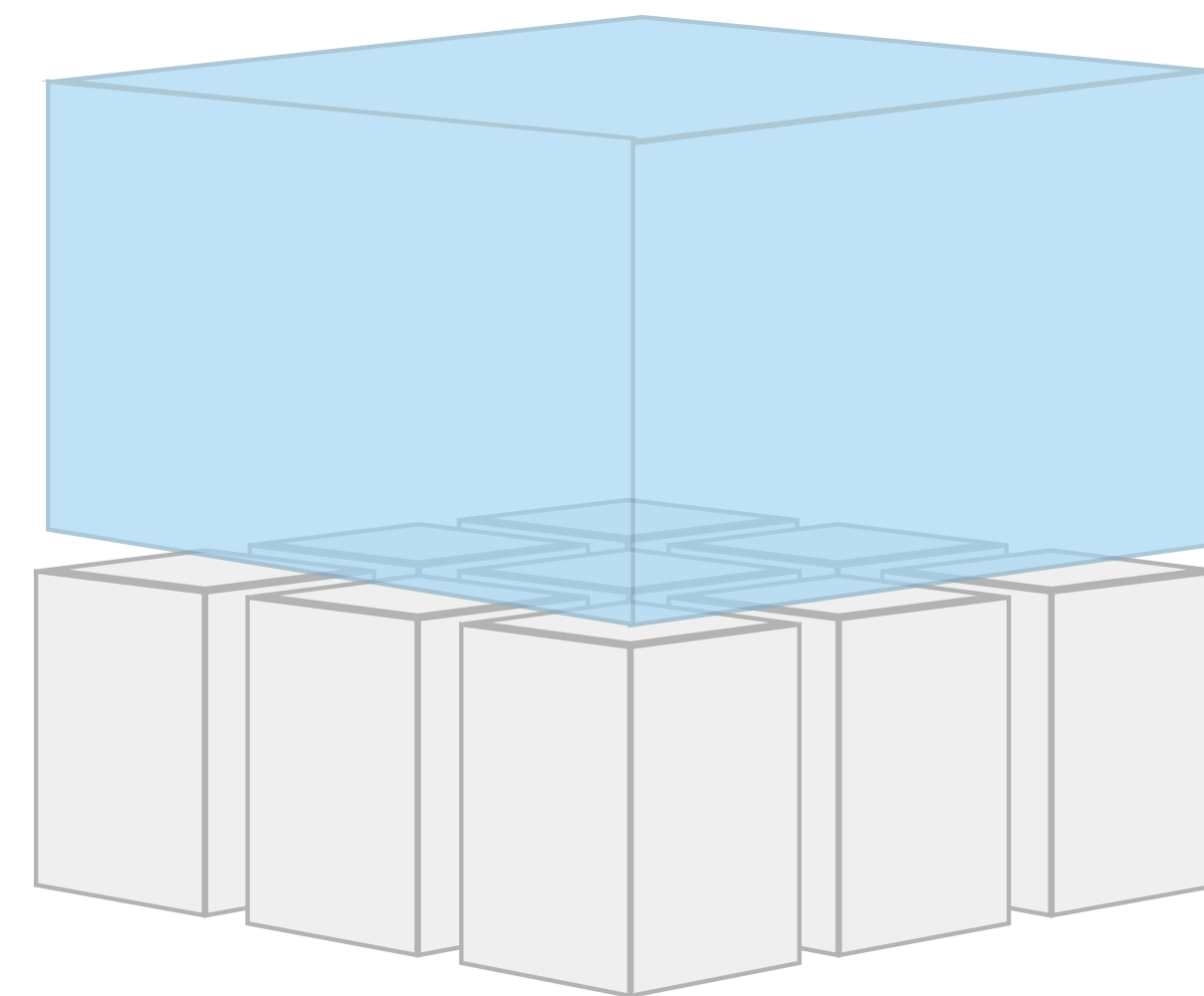
**Why would you cut up
your *system* into tiny,
distributed, hard-to-
manage fragments?**

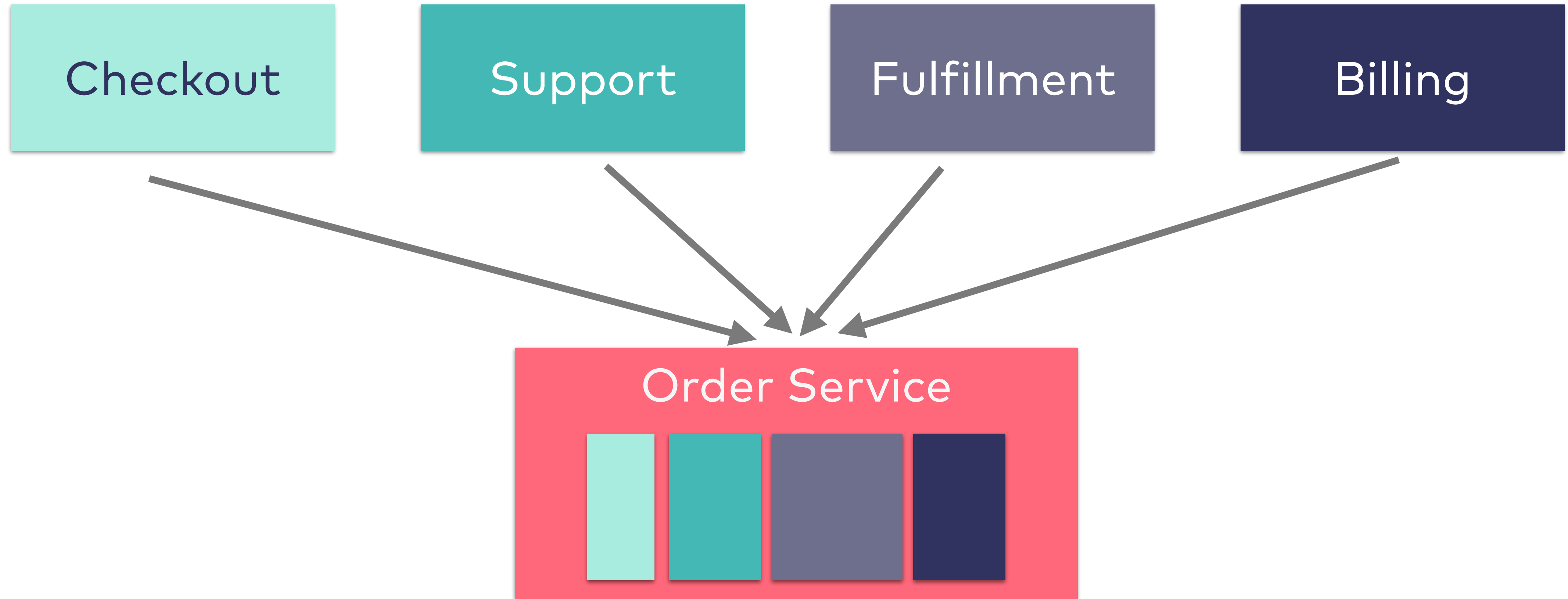
Everybody wants to be Netflix, but nobody is





...





Checkout

Support

Fulfillment

Billing

Order Service

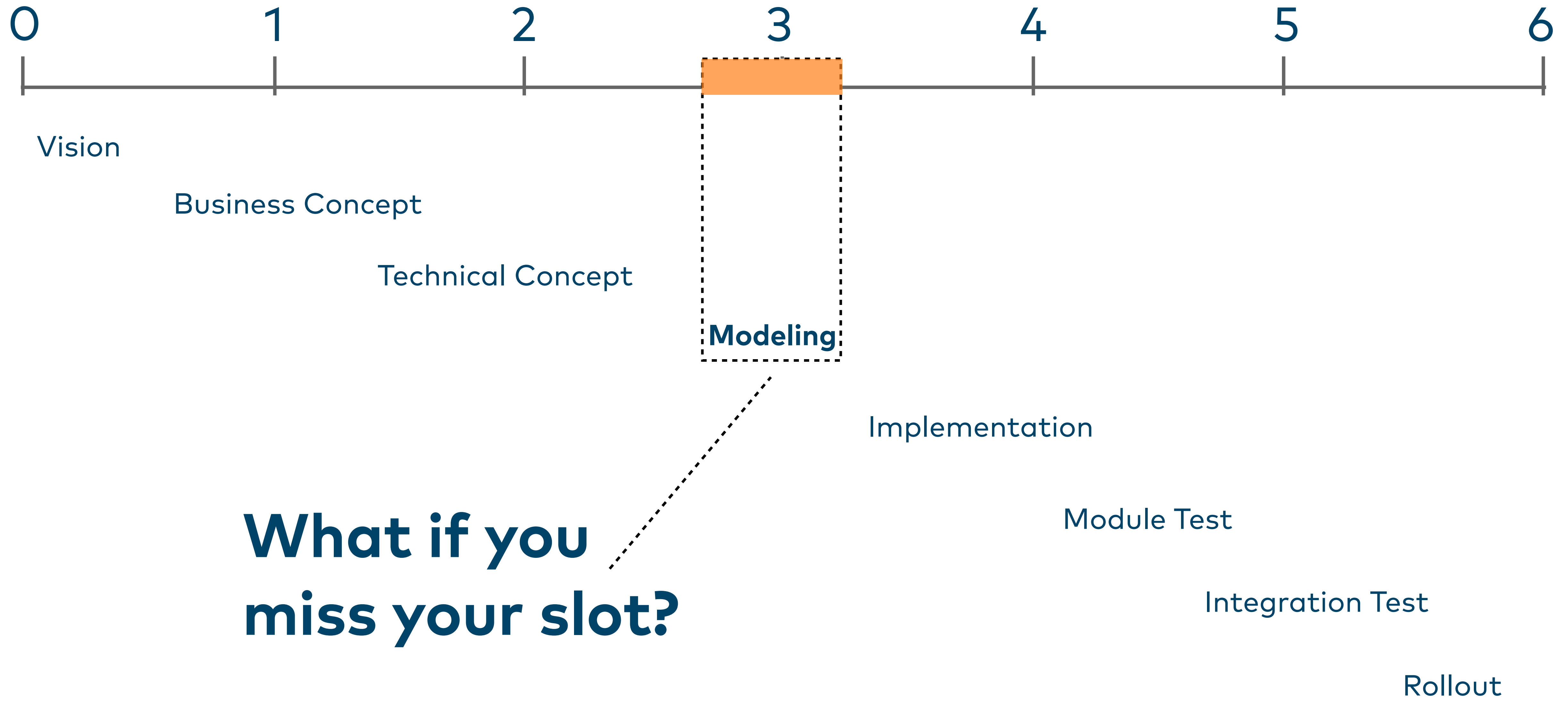
Lessons learned

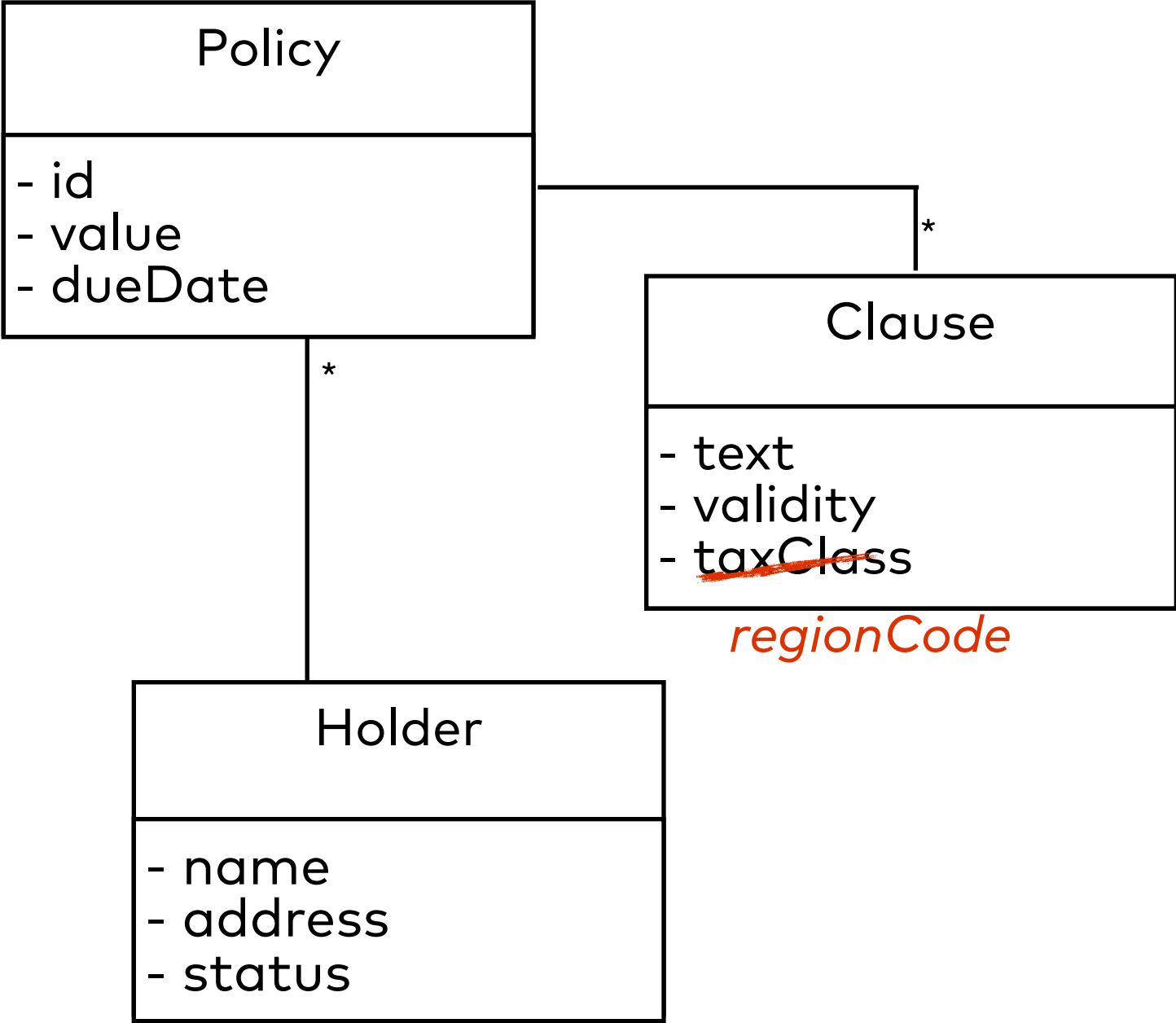
- **Small is not always beautiful**
- **Refactoring within team boundaries much easier than globally**
- **Ignore organizational parameters at your own risk**

#3: Your *system* *WILL* be dynamic

Context

- **Large-scale insurance system**
- **Model-driven development**
- **> 100 developers**
- **2 Releases/year**





Model Name	New Name (Meaning)	Description	Release Introduced
taxClass	regionCode	...	10.3
...			

Lessons learned

- **Centralized responsibility hurts**
- **Faced with too much rigidity, a way around the rules will be found**
- **Just because you're used to it doesn't mean it's acceptable**

#4: Free-style architecture

Context

- **E-Commerce/Online shop (Retail)**
- **100-120 developers**
- **~10 self-contained teams**

strength of
decoupling



systems

μservices

components

modules

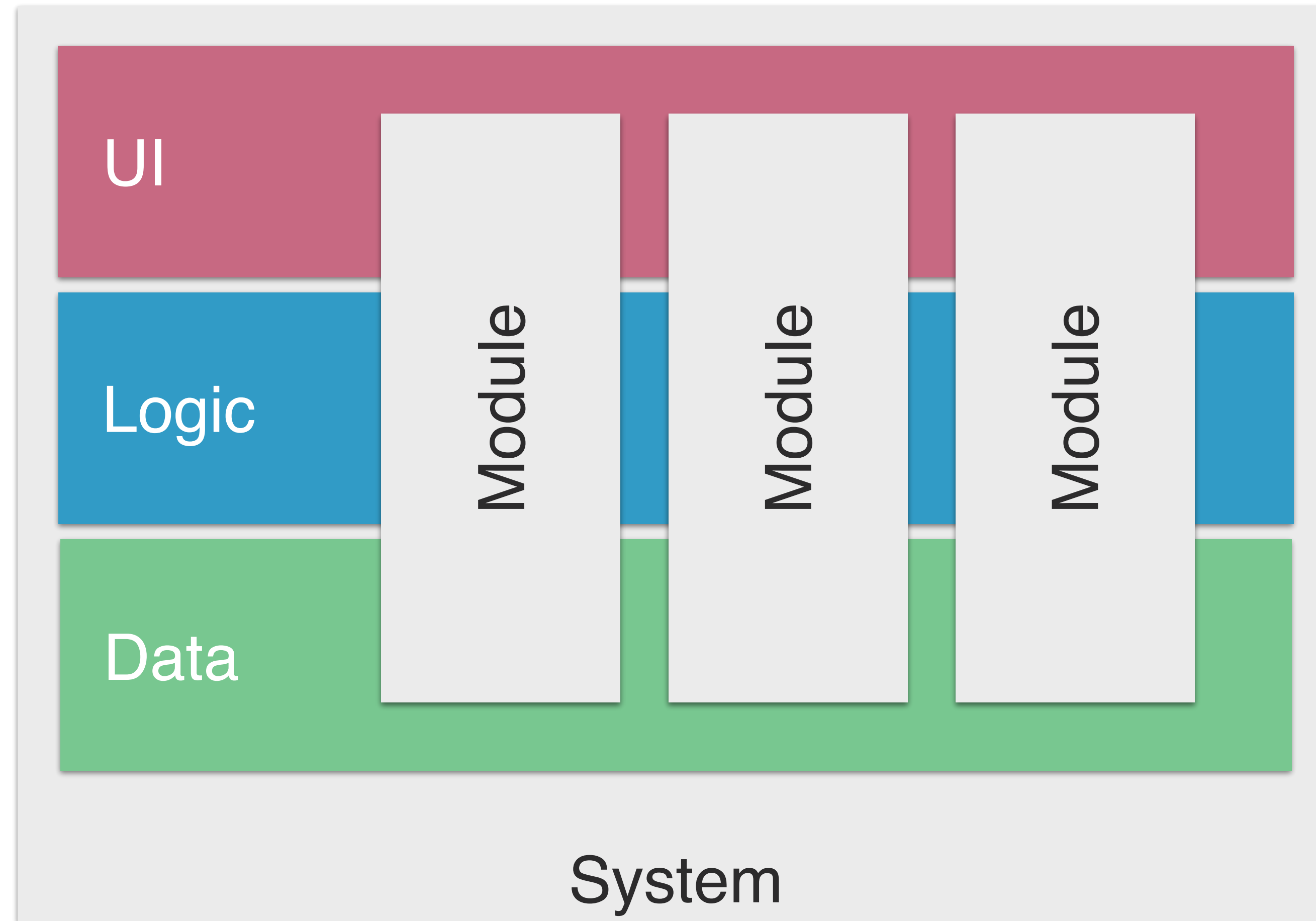
methods

number of
developers

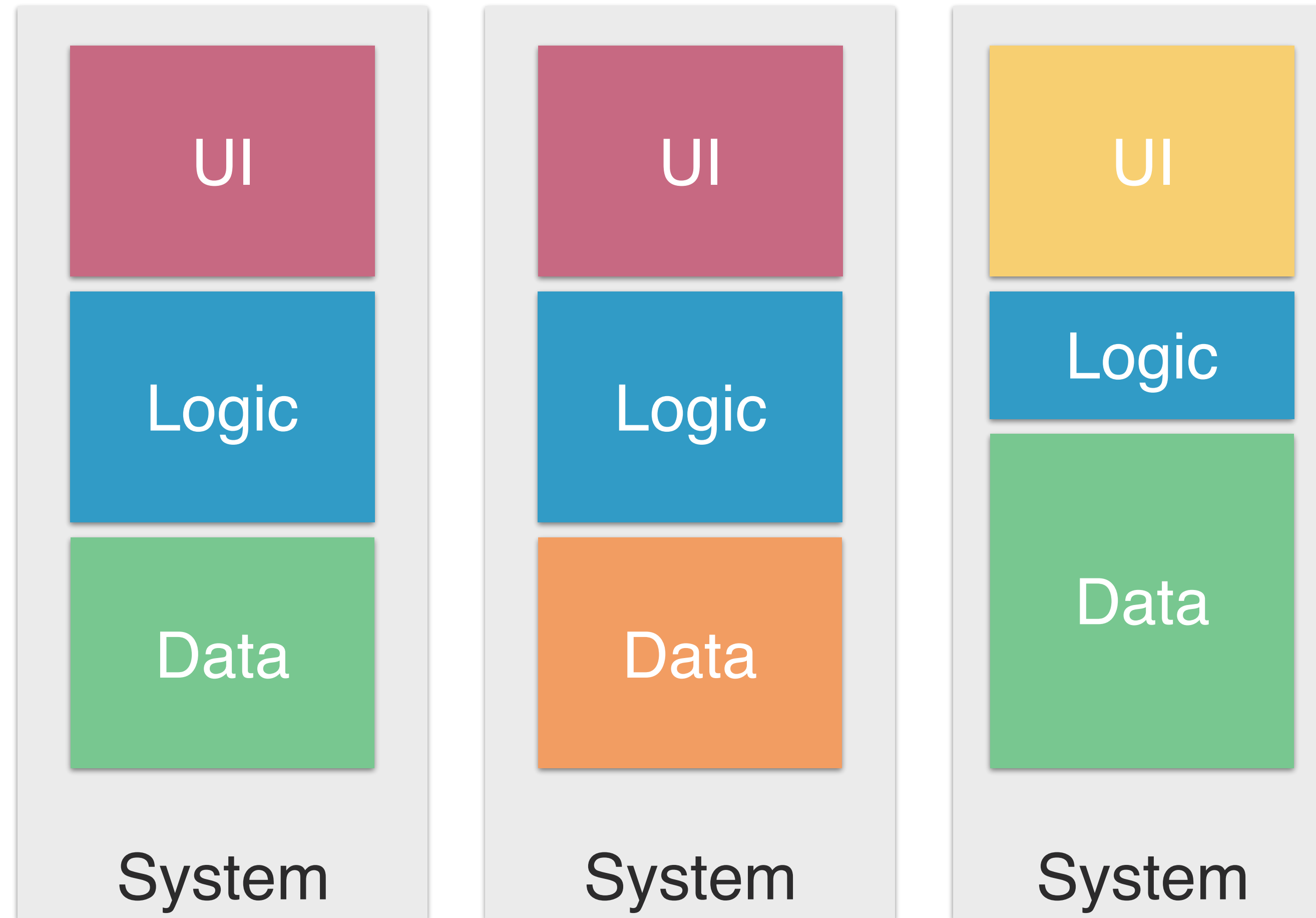


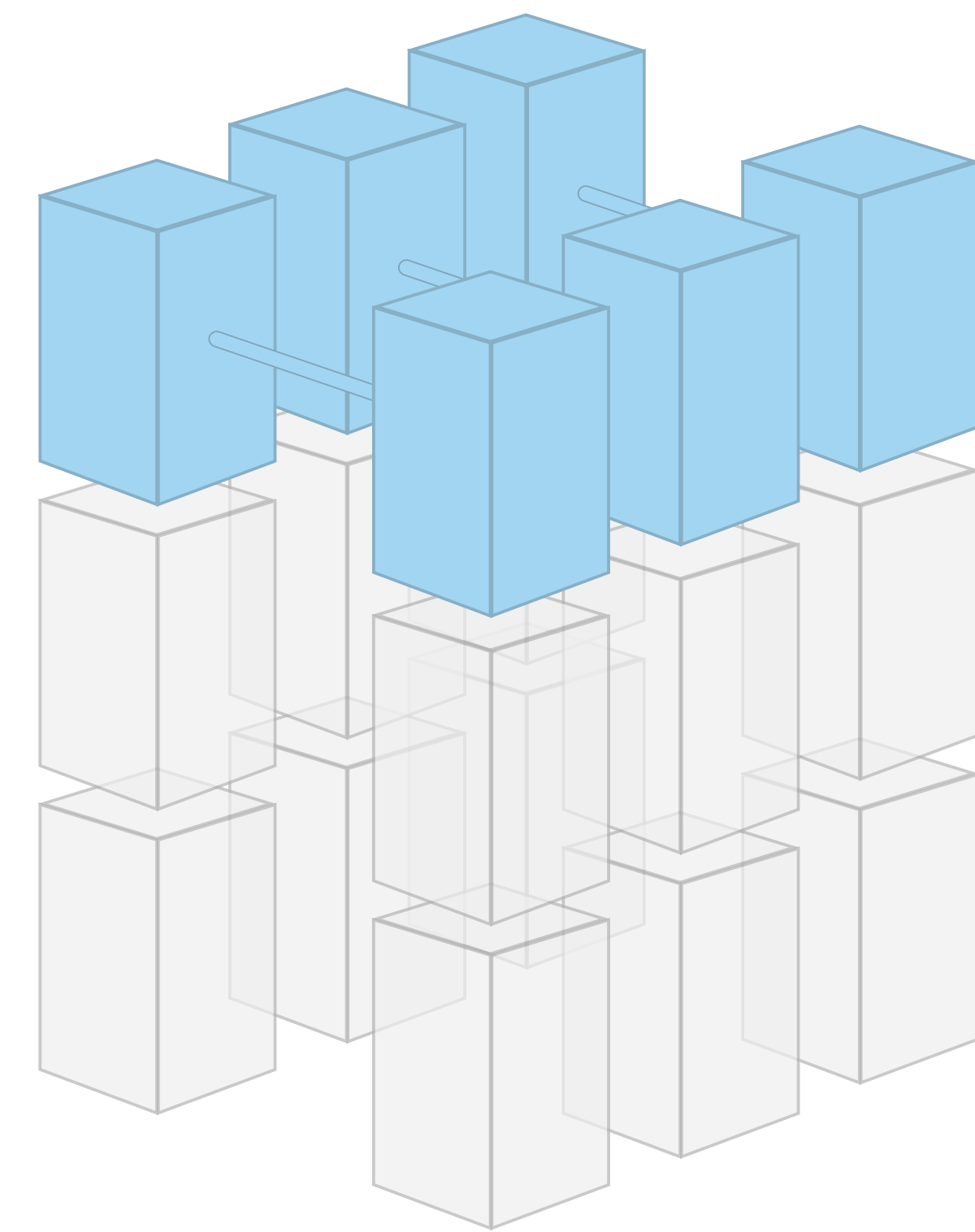
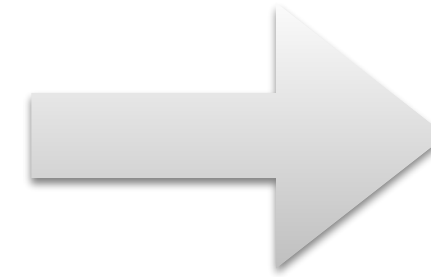
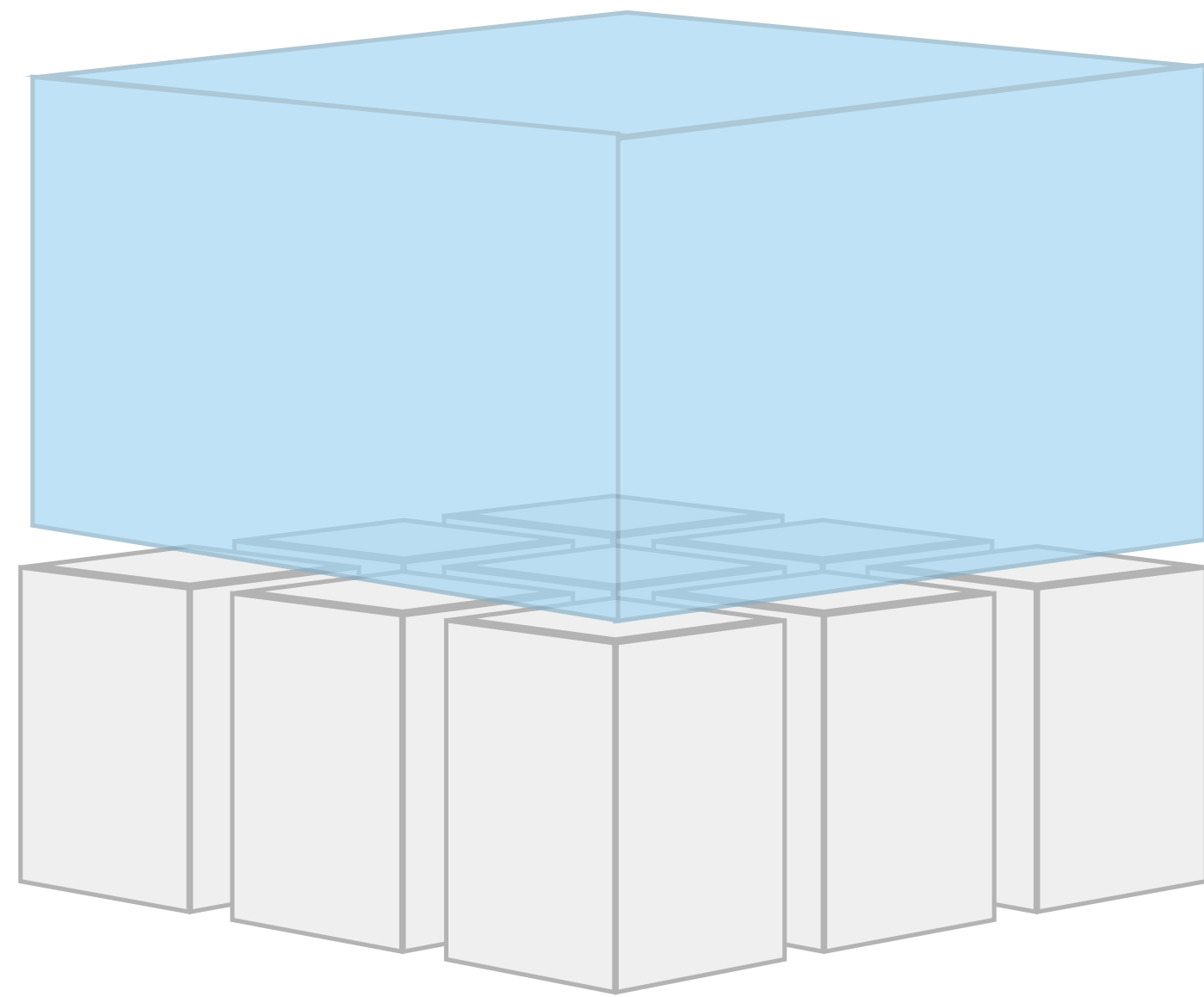
@stilkov

From a layered system ...



... to a *system of systems*





In-page
JavaScript method calls
Shared abstractions & frameworks
Common language runtime
HTML 5 JS platform

Cross-page
Links & redirects
Micro-architecture
HTTP
Standard Browser

But ...

- **Lack of standardization led to inefficient UI integration at runtime**
- **Vast differences in API style, formats, documentation created needless extra work**
- **Despite no centralised frontend, a central frontend team created a new bottle neck**

**You cannot decide to not have an architecture;
if you don't actively create it, be prepared to
deal with the one that emerges**

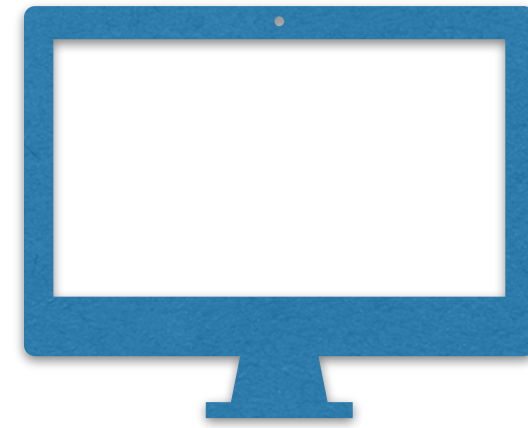
There's a fine line between diversity (that adds value) and chaos (that doesn't)

Extremely loose coupling requires very few rules, but they need to be enforced strictly

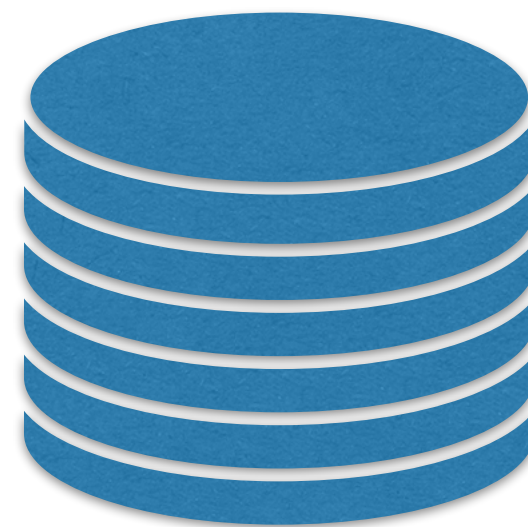
#5: Cancerous Growth

Context

- **Financial services provider with independent brokers as clients**
- **~30 developers**
- **20 years of company history**



Oracle Forms App

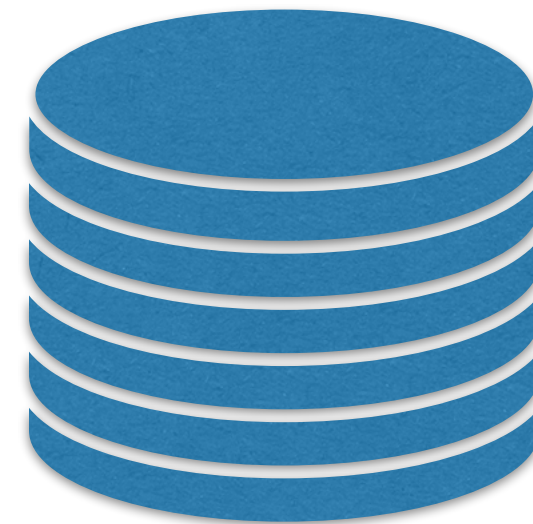


Oracle DB

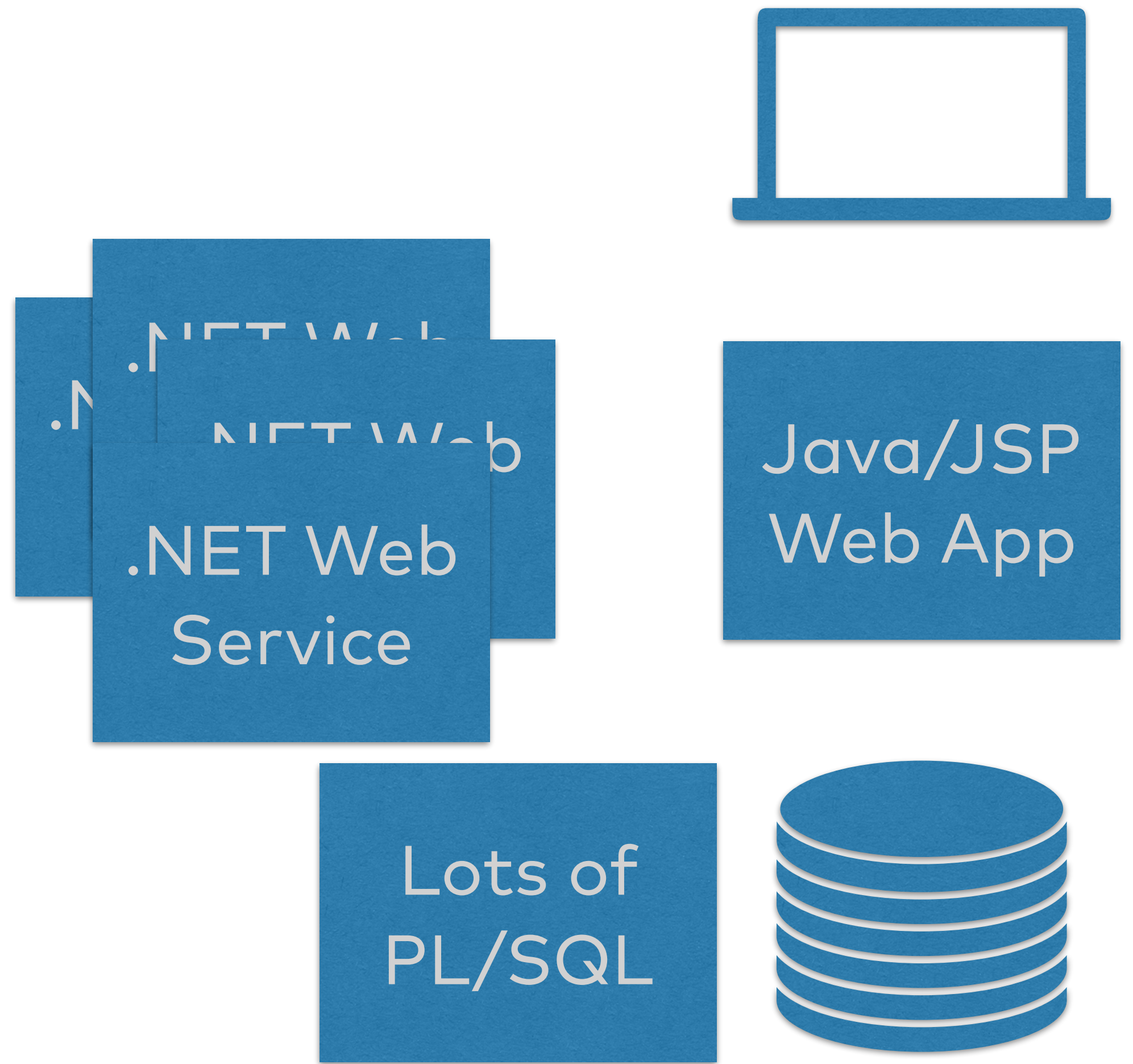


Java/JSP
Web App

Lots of
PL/SQL



Oracle DB



Oracle DB

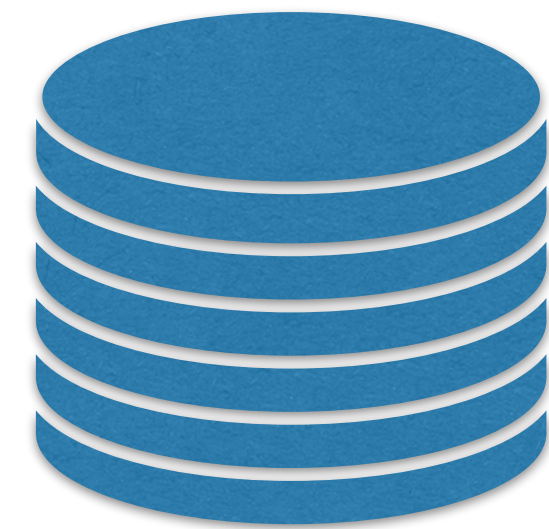
Company A



Java/JSP
Web App

.NET Web
Service

Lots of
PL/SQL



Oracle DB

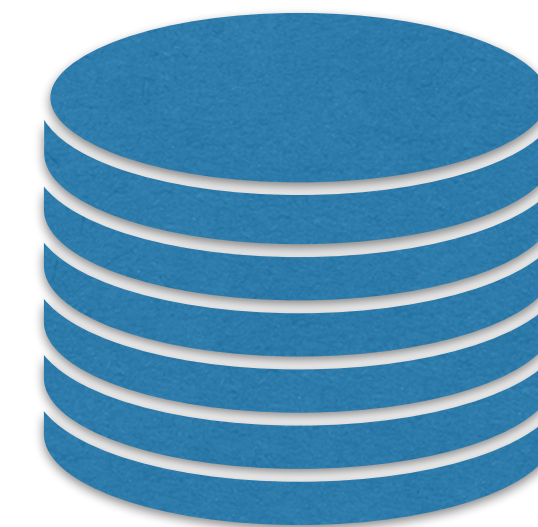
Company B



Java/JSP
Web App

.NET Web
Service

Lots of
PL/SQL



Oracle DB

Company A



Java/JSP
Web App

.NET Web
Service

Lots of
PL/SQL

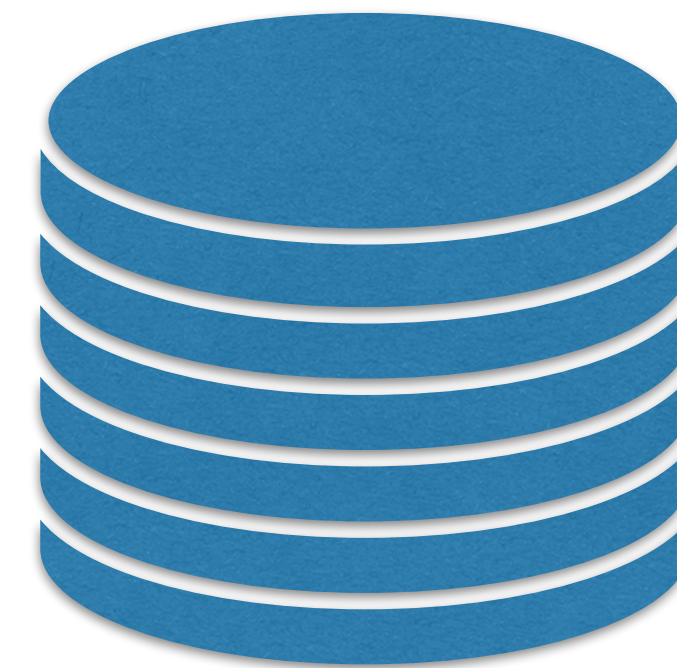
Company B



Java/JSP
Web App

.NET Web
Service

Lots of
PL/SQL

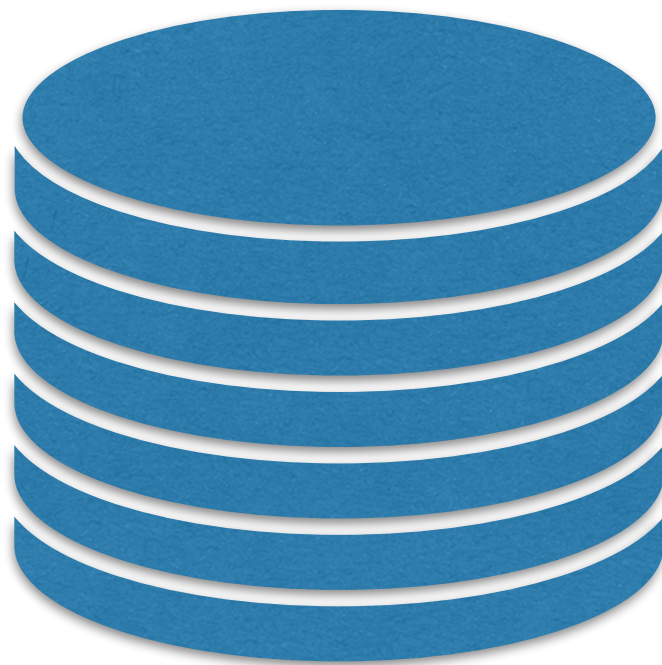
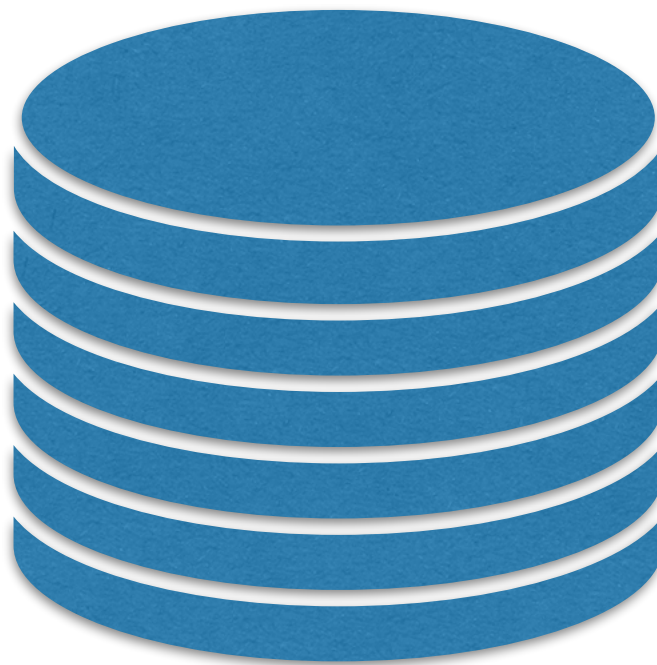
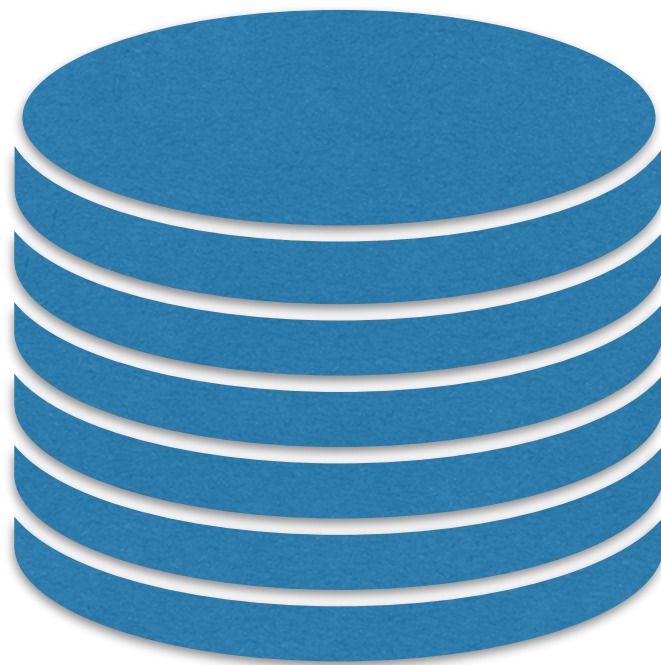
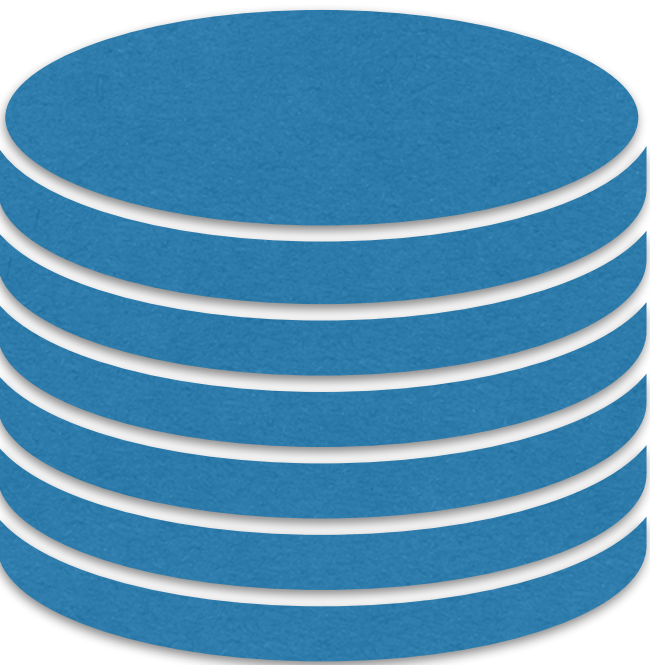
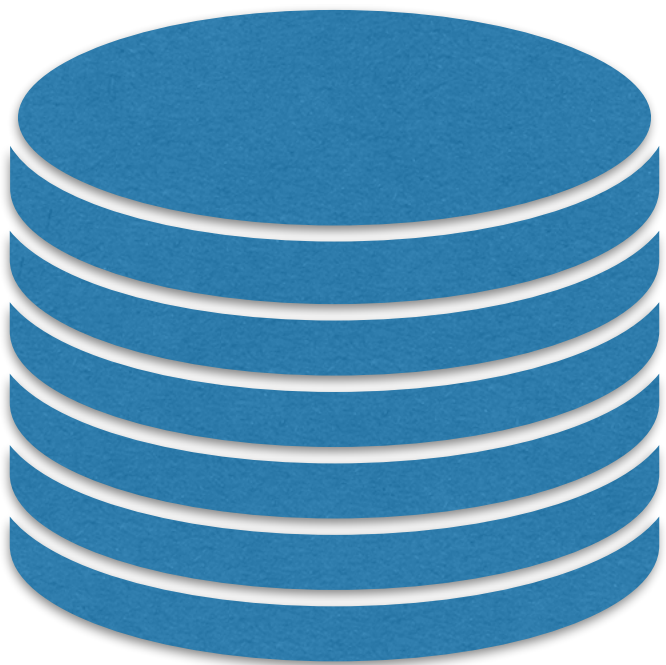
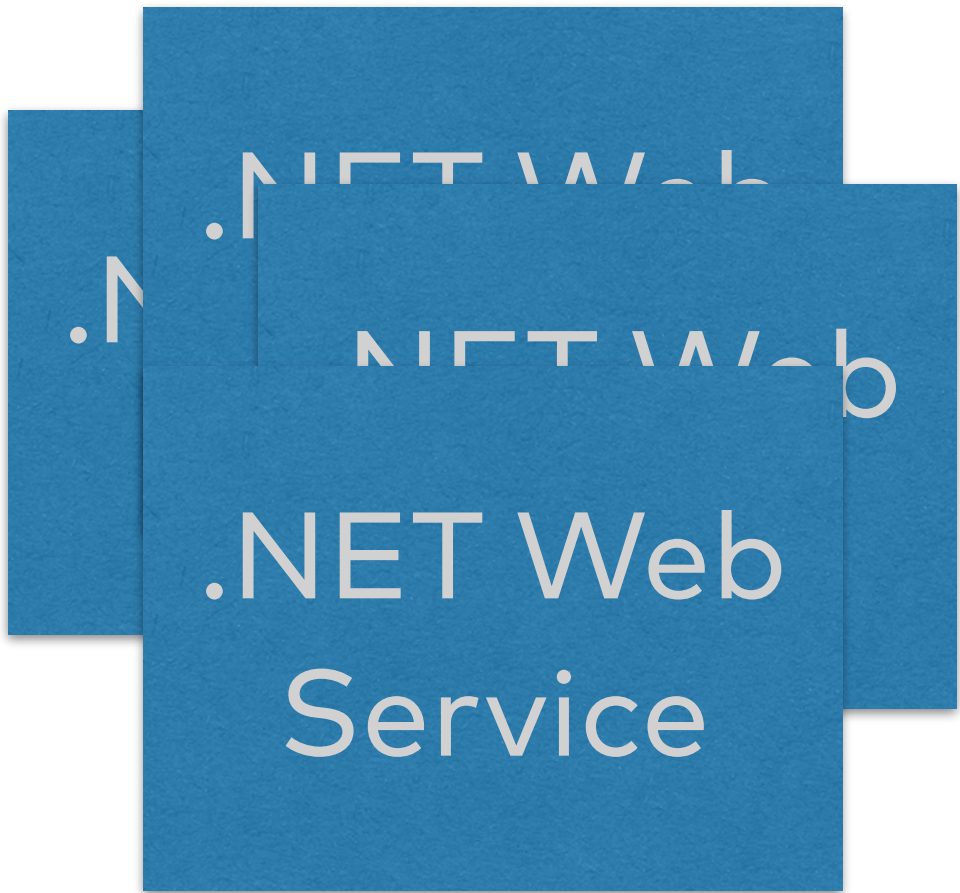


Oracle DB

Company A



Java/JSP
Web App



Couch/Pouch

Mongo

Oracle DB

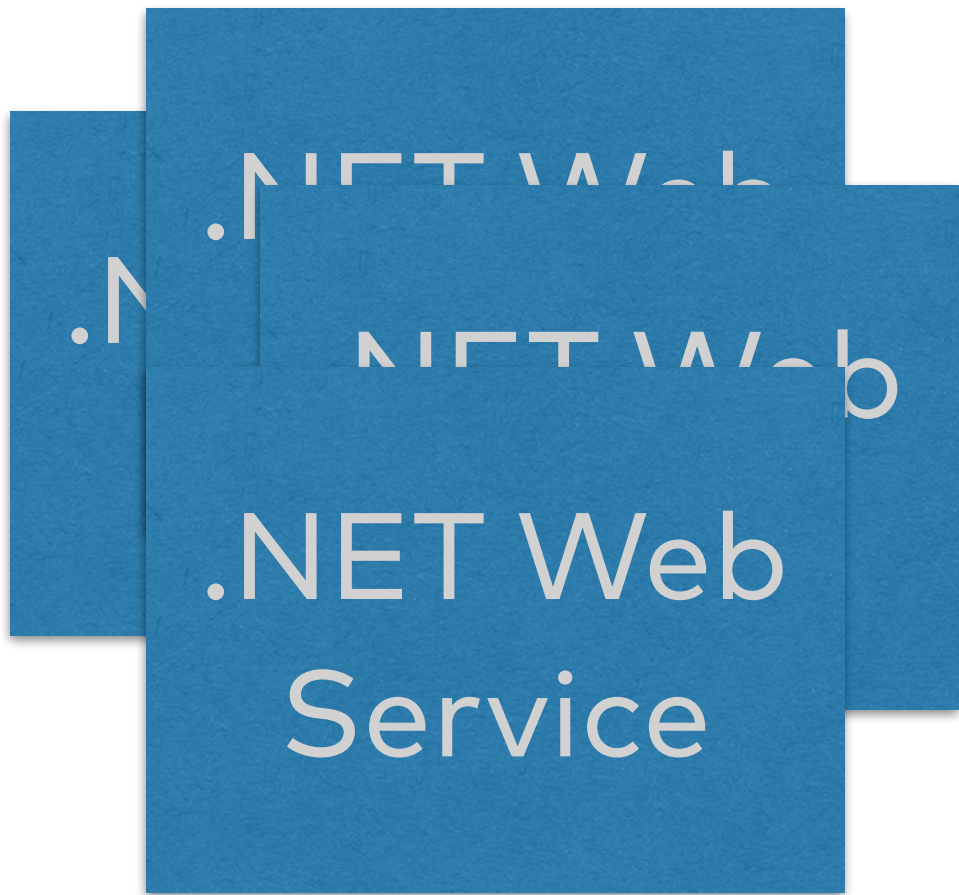
Mongo

MySQL

Company B



Java/JSP
Web App



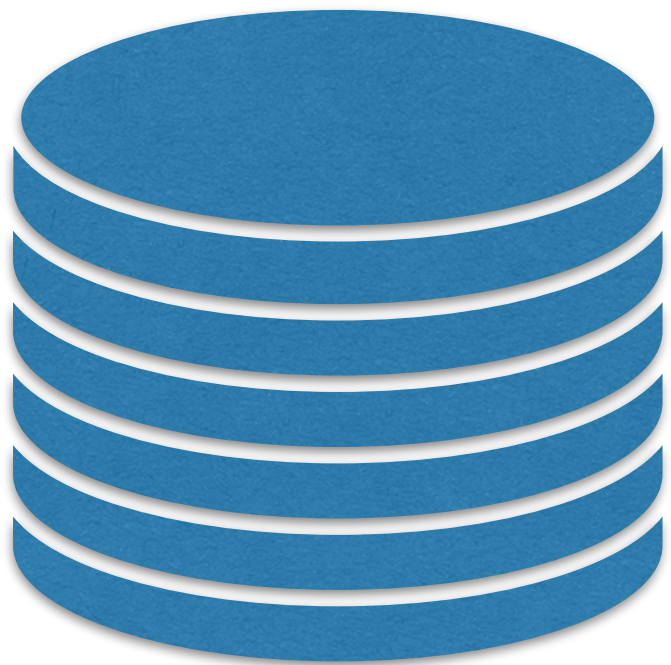
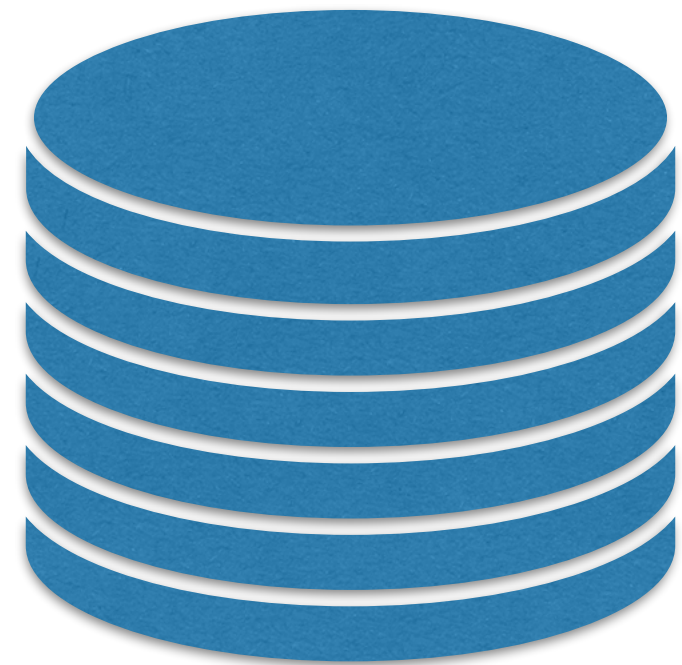
Company A

C++ Encryption Lib



.NET Web Service

Java/JSP Web App



Couch/Pouch

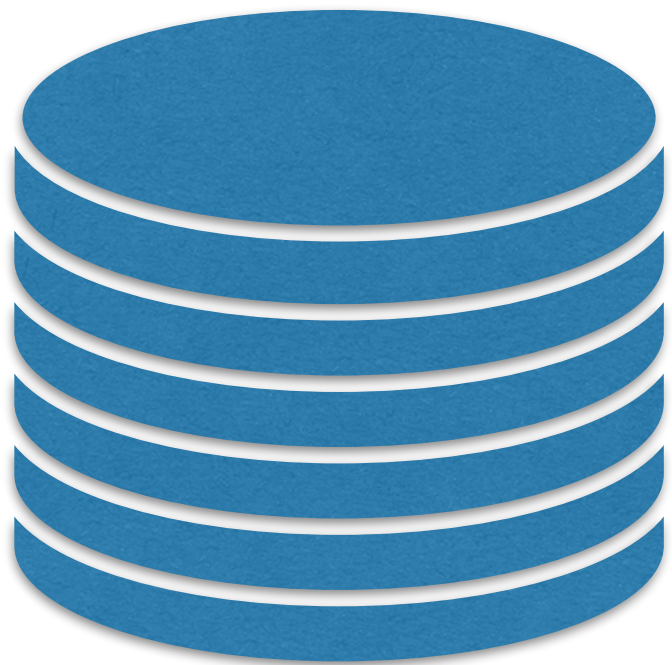
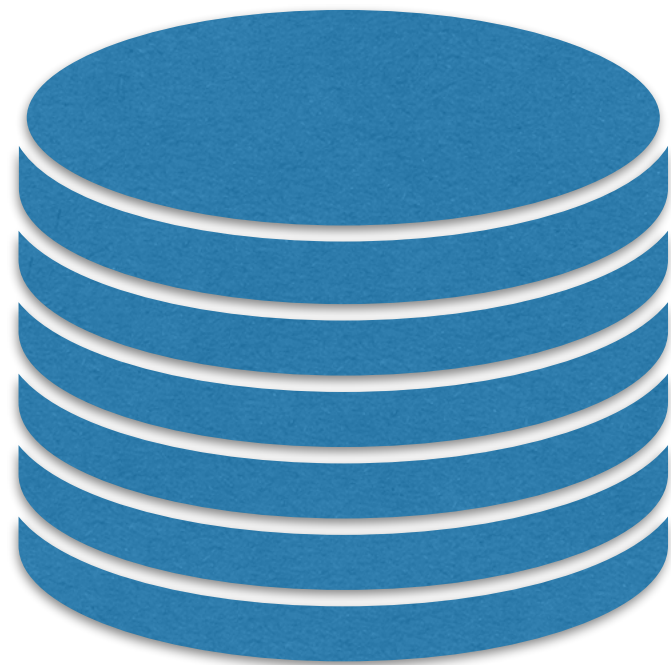
Mongo

Company B



Java/JSP Web App

.NET Web Service



Mongo

MySQL

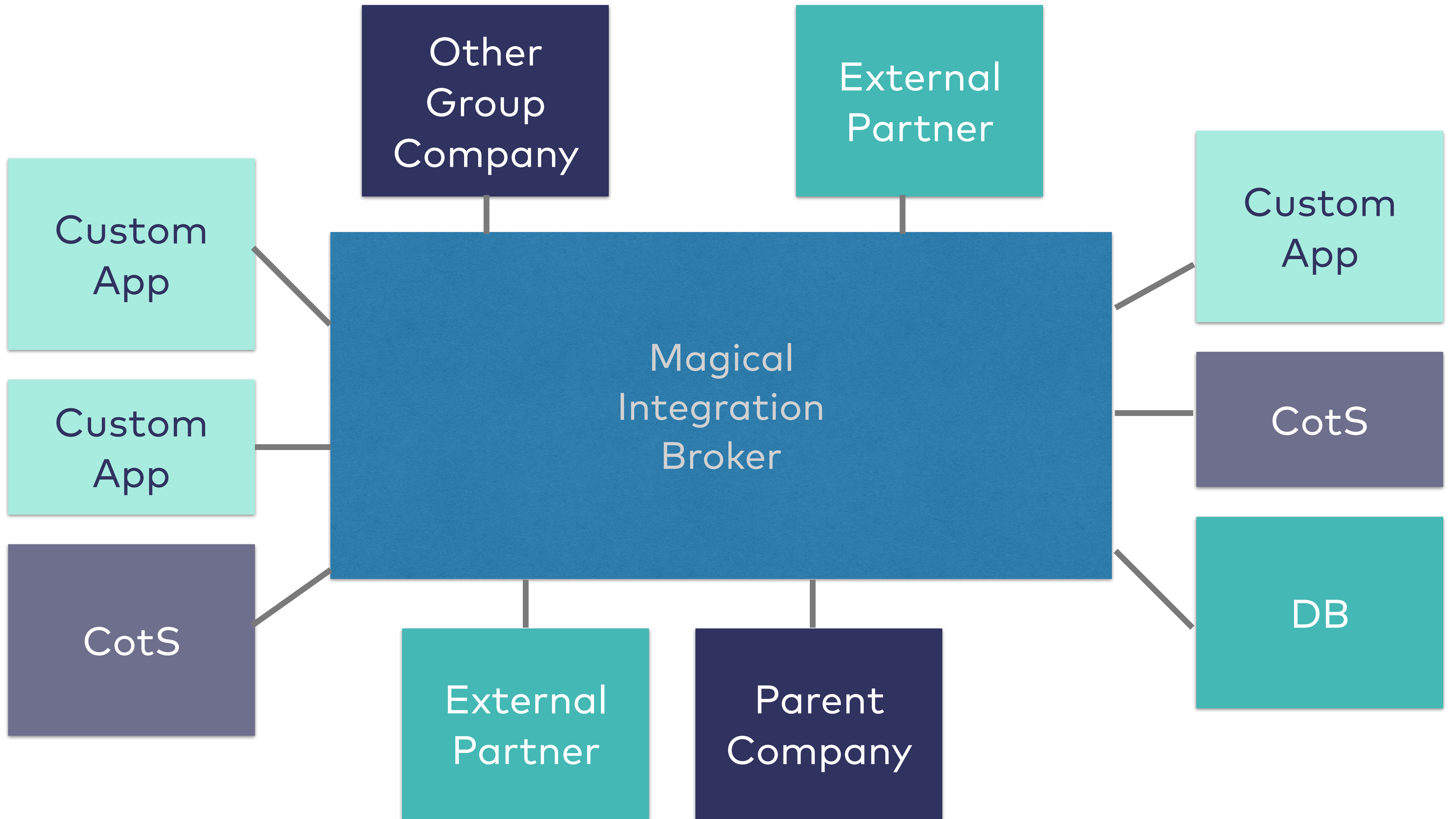
Lessons learned

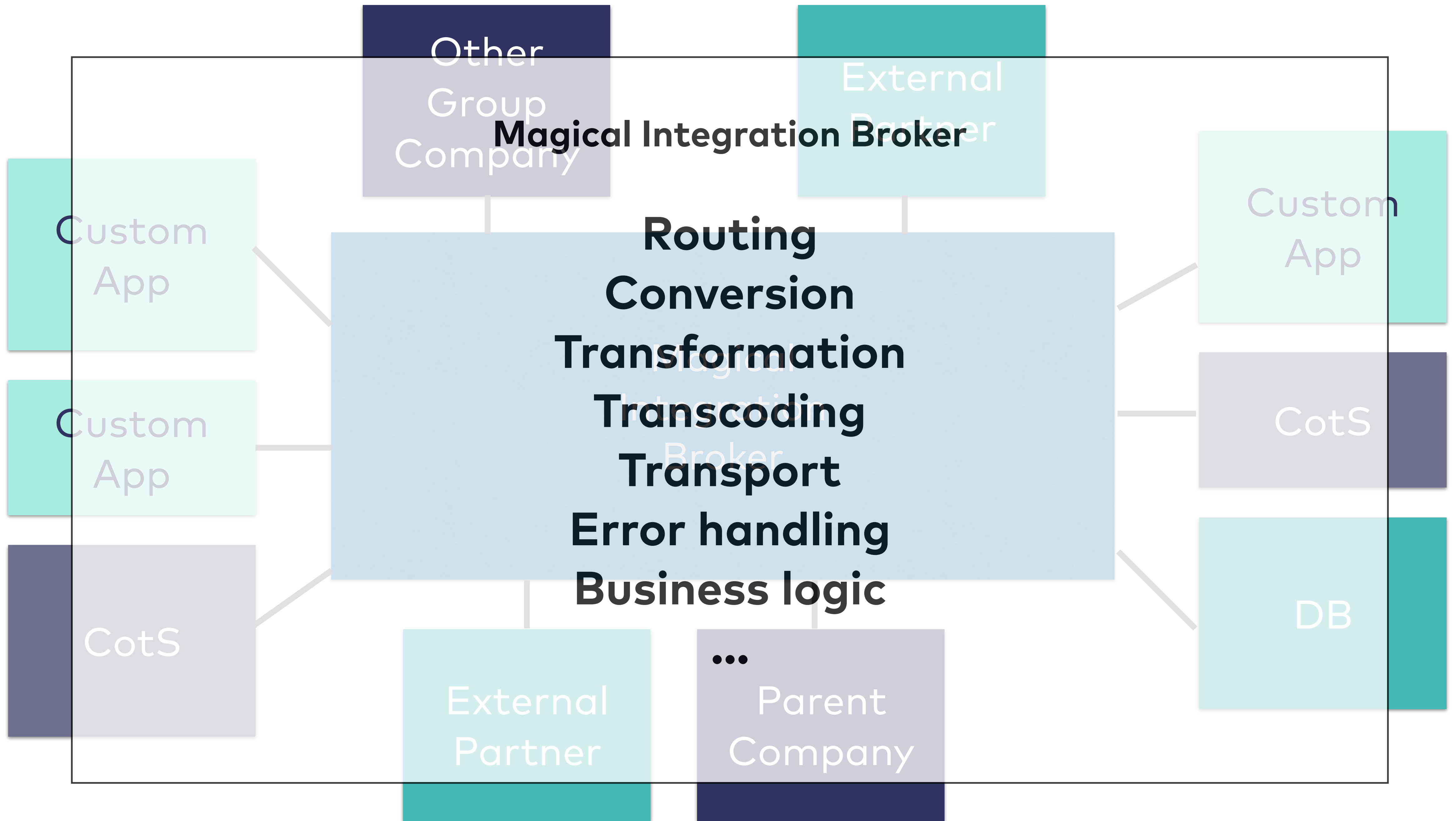
- **Successful systems often end up the worst architecture**
- **Unmanaged evolution will lead to complete chaos**
- **Don't be afraid of some light architectural governance**

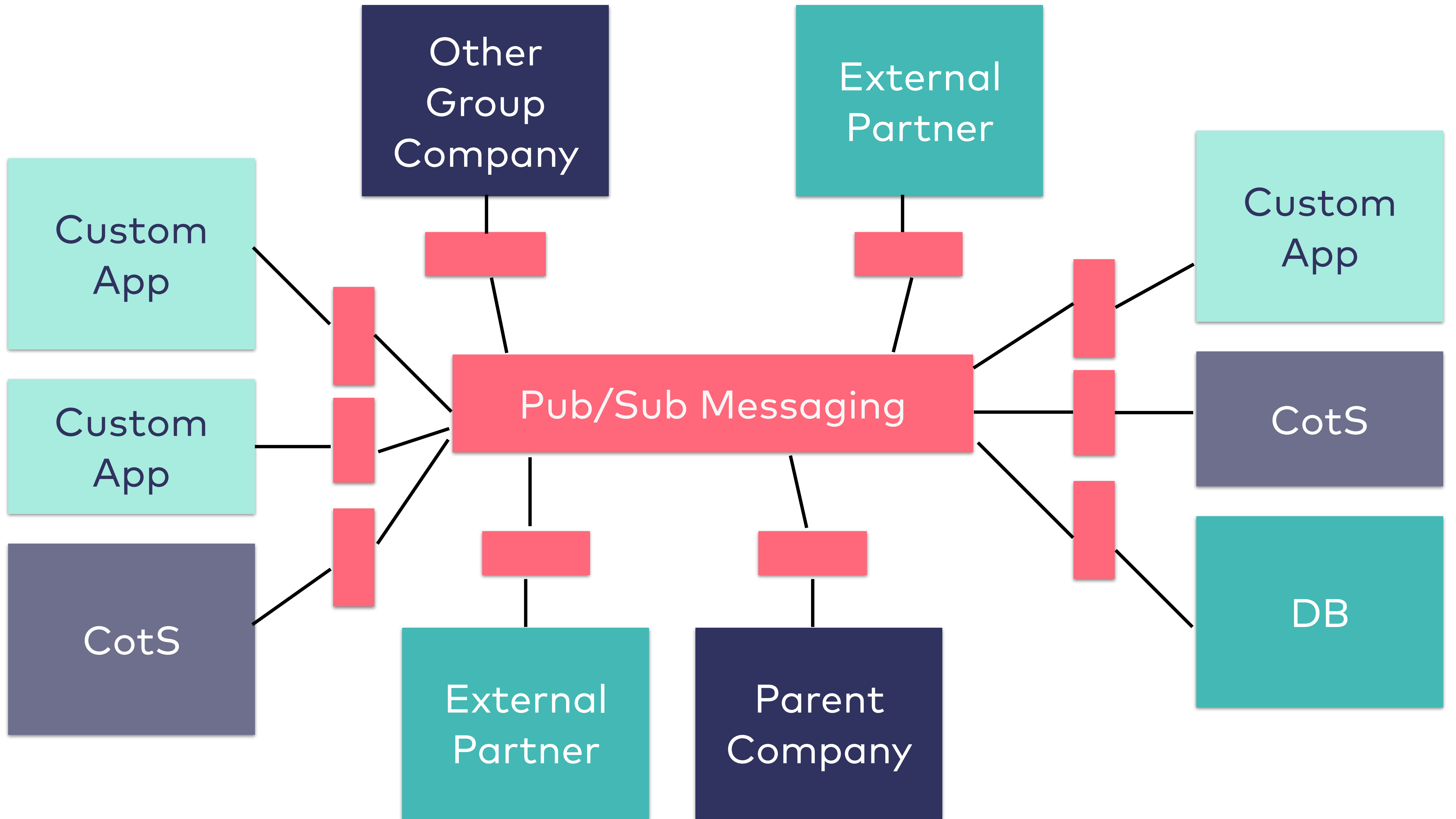
#6: Improve with less intelligence

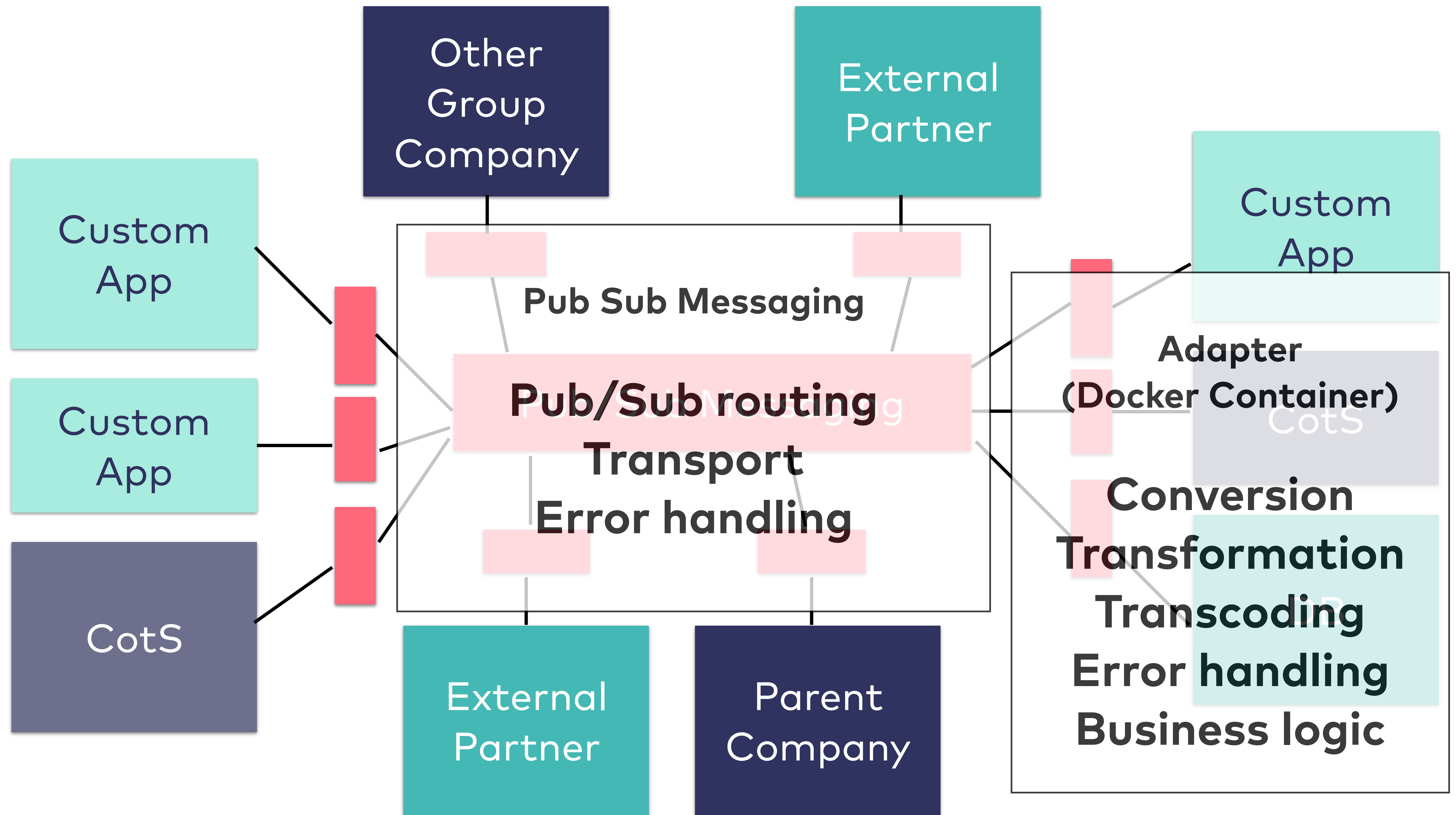
Context

- **Bank with multiple CotS systems**
- **Highly proprietary integration solution phased out by vendor**
- **Project launched to replace commercial product with open source solution**









Lessons learned

- **Smart endpoints, dumb pipes (cf. Jim Webber)**
- **Value of specific over generic solutions**
- **Micro architecture with blueprints**

Takeaways

1.

**Don't be afraid of
architecture**

2.

**Choose the simplest thing
that will work**

3.

Create evolvable structures

4.

**Manage your *system's*
architectural evolution**

5.

**Don't build road blocks –
create value and get out of
the way**

That's all I have. Thanks for listening!

Stefan Tilkov
@stilkov
stefan.tilkov@innoq.com
Phone: +49 170 471 2625



innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
Phone: +49 2173 3366-0

Ohlauer Straße 43
10999 Berlin
Germany

Phone: +49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany

Phone: +49 2173 3366-0

Kreuzstraße 16
80331 München
Germany

Phone: +49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
Phone: +41 41 743 0116

@stilkov



www.innoq.com

SERVICES

Strategy & technology consulting
Digital business models
Software architecture & development
Digital platforms & infrastructures
Knowledge transfer, coaching & trainings

FACTS

~150 employees
Privately owned
Vendor-independent

OFFICES

Monheim
Berlin
Offenbach
Munich
Hamburg
Zurich

CLIENTS

Finance
Telecommunications
Logistics
E-commerce
Fortune 500
SMBs
Startups

Growing architectural maturity means less guidance and rules are needed

The more experienced you are at (active and passive) architectural governance, the less you can do of it