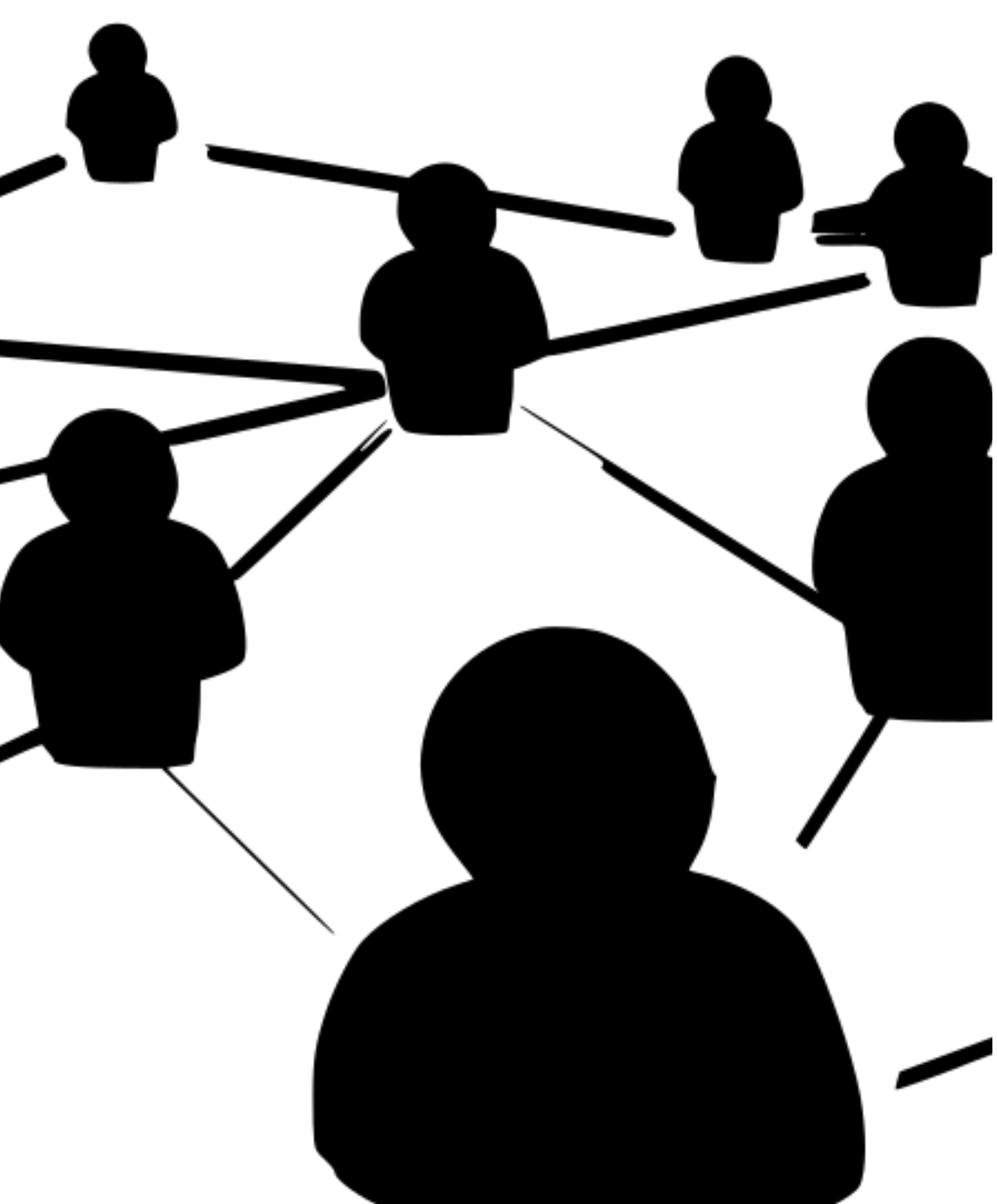


Architecture, Organization, Processes – and Humans

YOW! 18 August 2020

Stefan Tilkov, @stilkov
stefan.tilkov@innoq.com

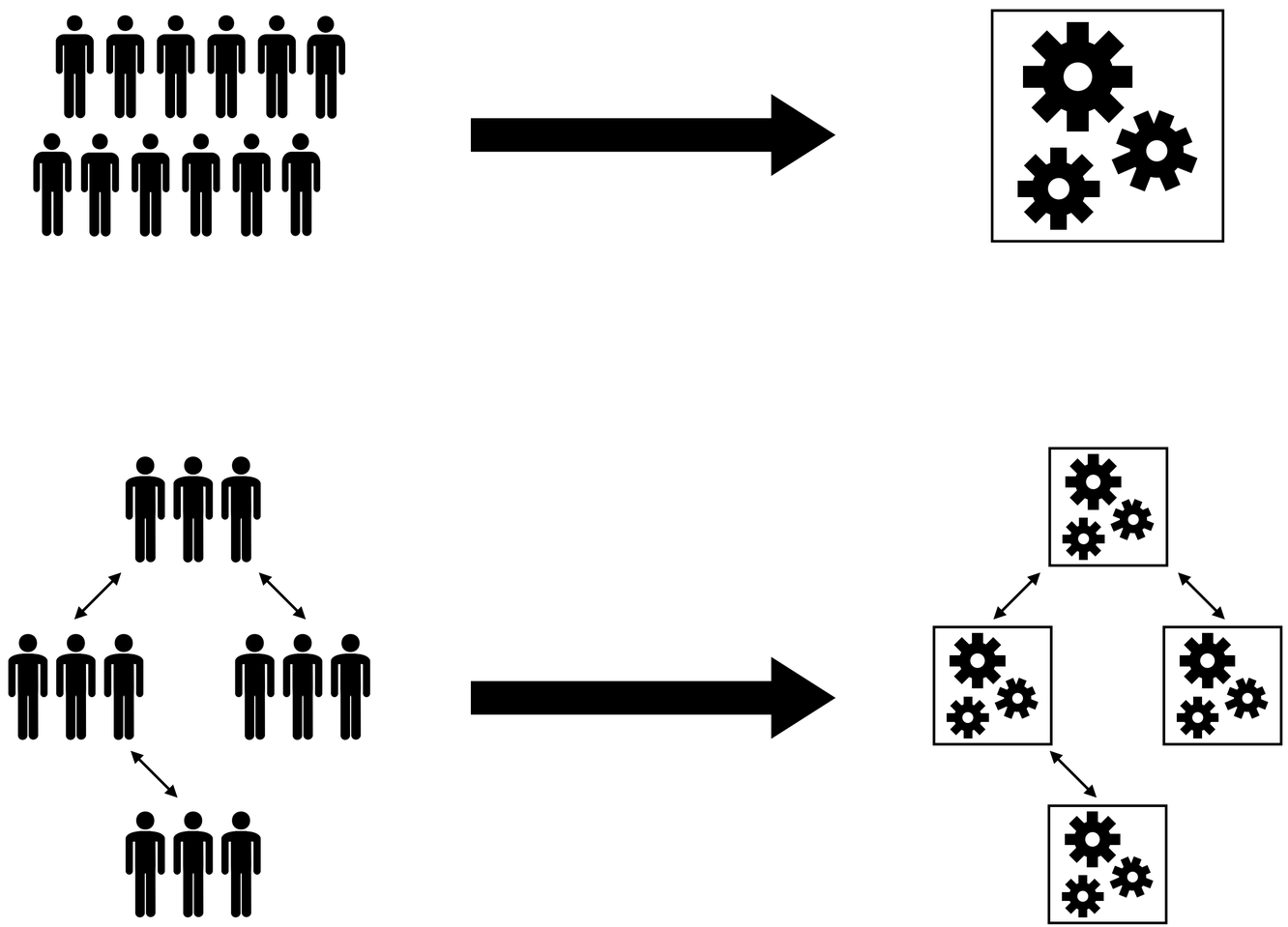
INNOQ

Architecture & Organization

Conway's Law: Organization → Architecture

"Organizations which design systems are constrained to produce systems which are copies of the communication structures of these organizations."
– M.E. Conway

Conway's Law Illustrated



Conway Reversal 1: Organization ← Architecture

Any particular architecture approach constraints organizational options – i.e. makes some organizational models simple and others hard to implement.

Conway Reversal 2: Organization ← Architecture

Choosing a particular architecture
can be a means of optimizing for
a desired organizational structure.

The “Tilkov wants a law, too” slide

The quality of a system’s architecture
is inversely proportional to the number
of bottlenecks limiting its evolution,
development, and operations

The “Tilkov wants a law, too” slide*

In a digital company, architecture, organization & processes can only evolve together

*Attempt #2 in case the 1st one doesn’t catch on

Let's talk about patterns

Pattern: <Name>

Description	Approach	Consequences
...	...	...

Pattern: Microservices

Description

Design modules as separate deployment and operation units, with large degrees of freedom for their implementation

Approach

Former technical detail (deployment architecture) as first class architectural design principle
Network communication as hard-to-cross boundary, enforcing encapsulation

Consequences

Isolation
Autonomy
Scalability
Resilience
Speed
Experimentation
Rapid Feedback
Flexibility
Replaceability

Antipattern: <Name>

Description	Reasons	Consequences
...	...	...

Antipattern: Microservices
(a.k.a. “Distributed Monolith“)

Description	Reasons	Consequences
System made up of arbitrarily sized, tightly coupled modules communicating over network interfaces	Hype-driven architecture Conference-driven development Missing focus on business domain Infrastructure over-engineering	“Ripple” effect of changes Complex environment Massive network overhead Performance issues Wild mix of technologies, products & frameworks Hard to understand & maintain

Antipatterns

Antipattern: Conference-driven Architecture

Antipattern: Conference-driven Architecture

Description

Hypes are accepted as gospel, and applied to problems regardless of whether they match requirements or not

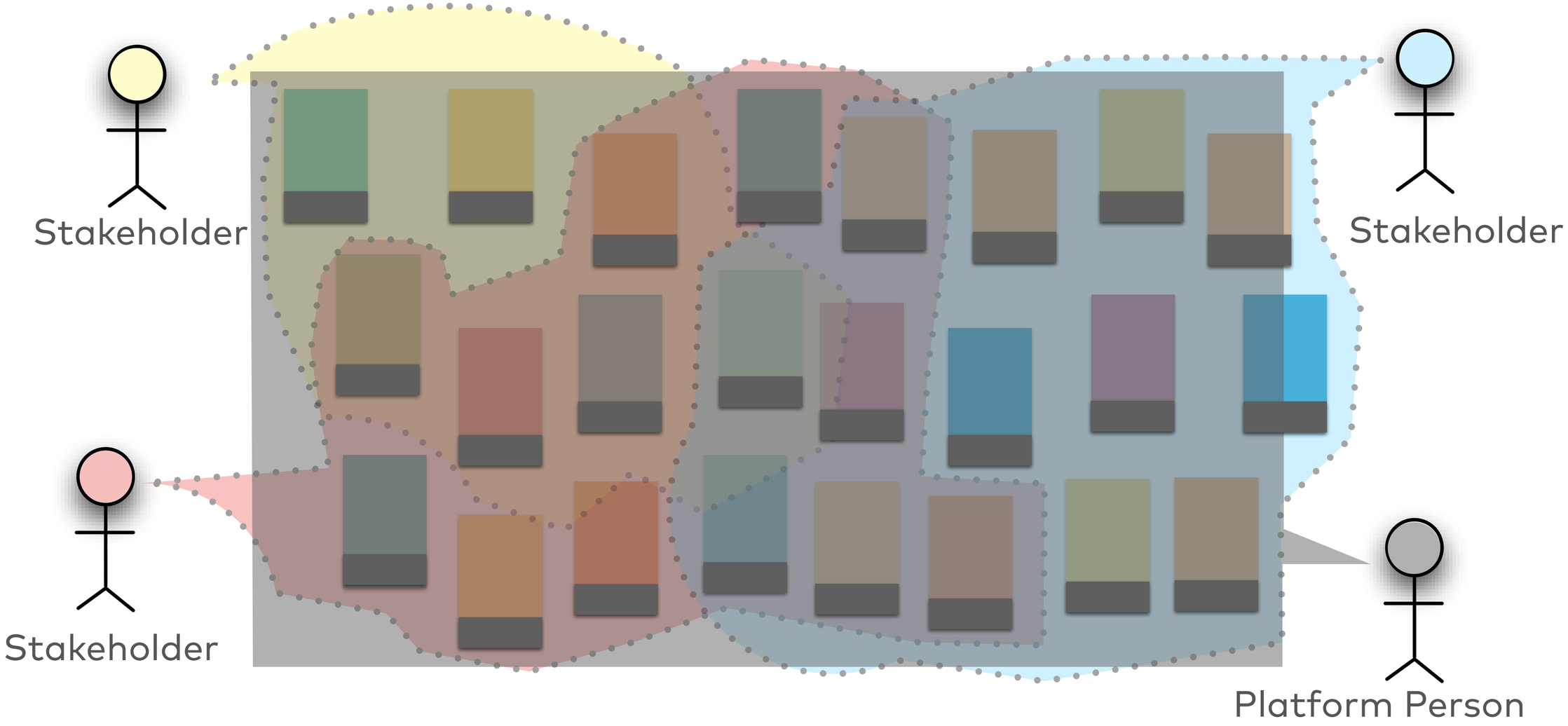
Reasons

- Hot and shiny toys!
- Community respect
- Search for guidance

Consequences

- Occasional successes
- Motivated developers
- Half-time of solutions matches conference cycle time
- Acceptance of architecture directly related to # of conference visits

Antipattern: Decoupling Illusion



Antipattern: Decoupling Illusion

Description

Technical separation into subsystems/services does not match business domain separation

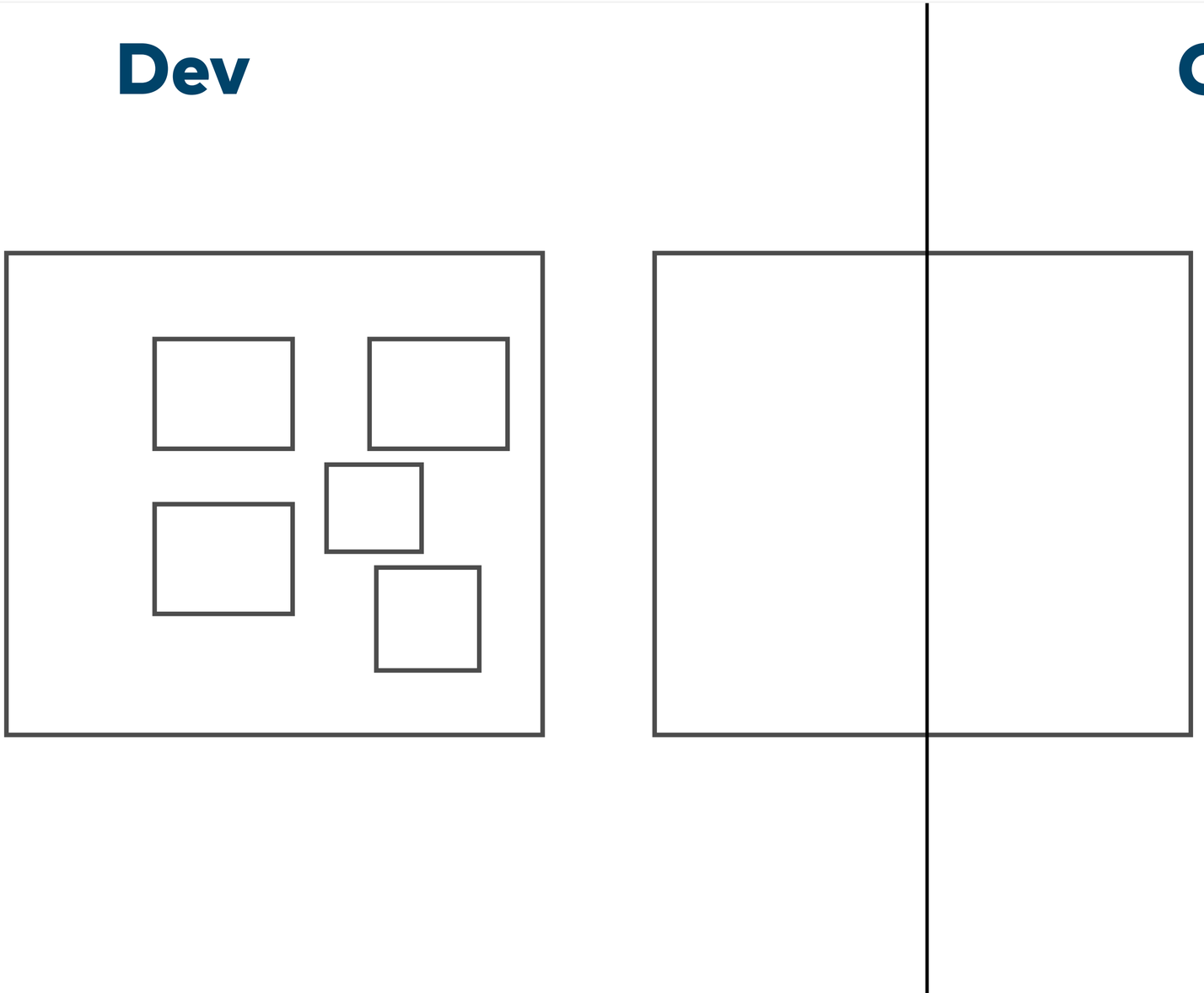
Reasons

- Technical drivers prioritized over business drivers
- Lack of awareness for stakeholder needs
- Reuse driver furthers single platform approach
- Microservices hype

Consequences

- Technical complexity
- Conflicting stakeholder needs require coordination
- Organizational bottlenecks due to centralized components with highly concurrent requests

Antipattern: Half-hearted Modularization



Antipattern: Half-hearted Modularization

Description

Modularization is performed in one aspect of the lifecycle only

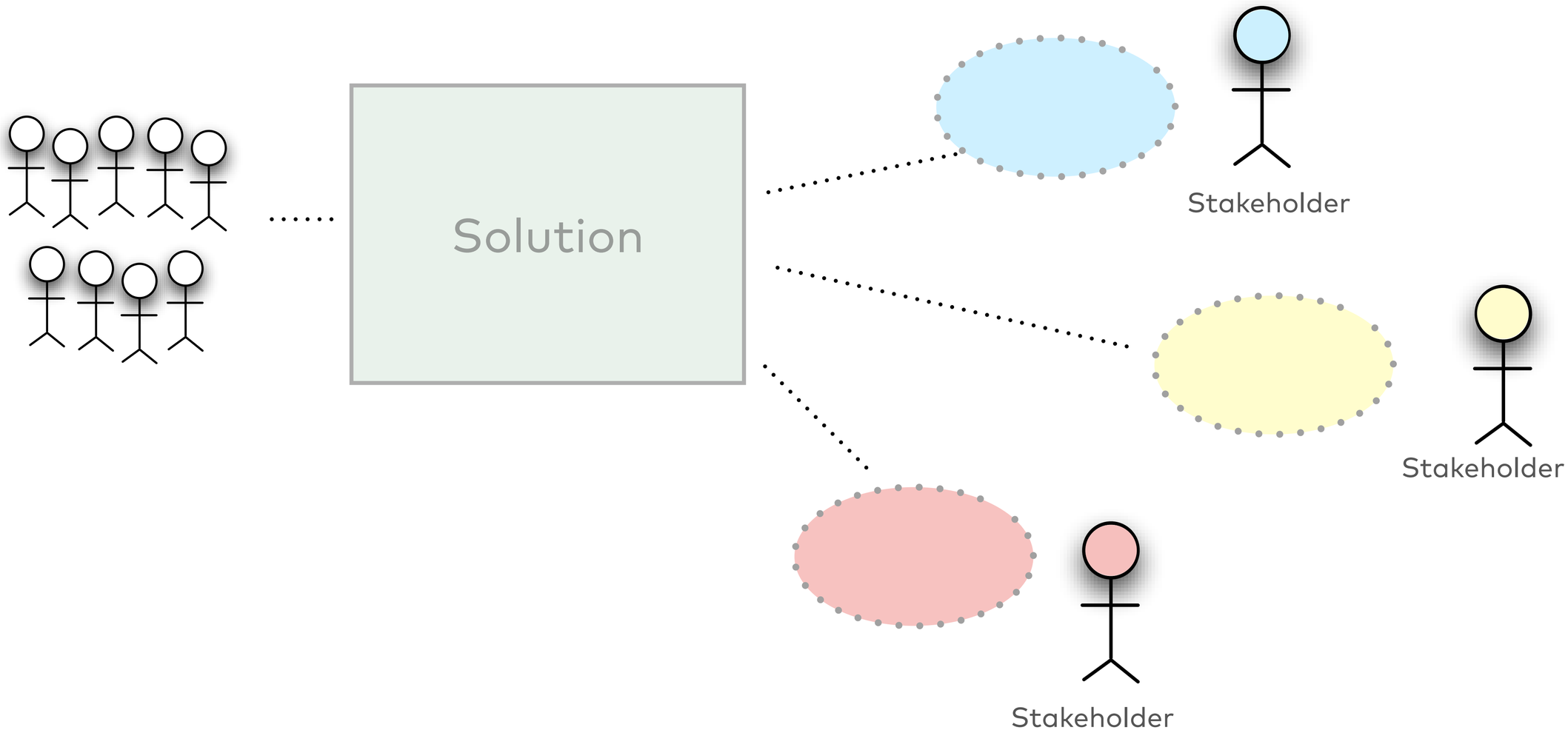
Reasons

- Resistance of one group to participate
- Lack of understanding of lifecycle aspects by initiators

Consequences

- Added complexity, limited value
- Delivery inhibited by existing processes
- New approach might be “burned” for future attempts

Antipattern: Solution Centrism



@stilkov

Antipattern: Solution Centrism

Description

Implementation solution as unifying factor

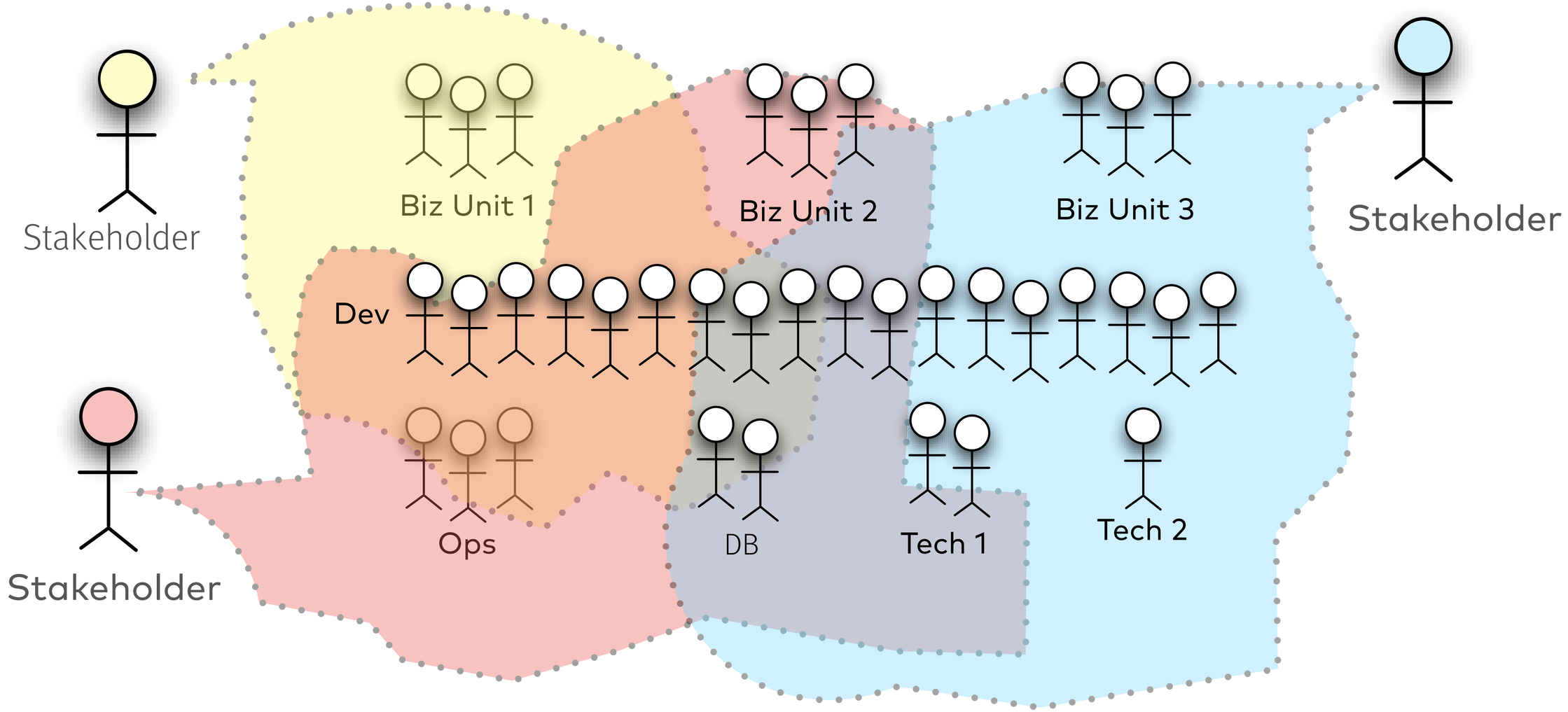
Reasons

- Vendor influence
- Experience drives selection of technology
- Sunk cost fallacy

Consequences

- Inefficiency due to hammer/nail problem
- Bottleneck by definition
- Technology, not domain as unifying factor
- Developer frustration
- Skills shortage in market
- Hard to motivate people to train in proprietary tech

Antipattern: Domain-last Approach



Antipattern: Domain-last Approach

Description

Major driver for organizational structure is roles and technical capabilities, not business domain

Reasons

- Matches classical company structure
- Division of labor in divisions, department, teams
- Projects as exceptions to change something that works

Consequences

- Inter-departmental politics over business needs
- Conflicting project and disciplinary hierarchies and stakeholders
- Blameshifting

Antipattern: Uncreative Chaos

Antipattern: Uncreative Chaos

Description

Lack of architectural structure & repeatable process for architectural decisions

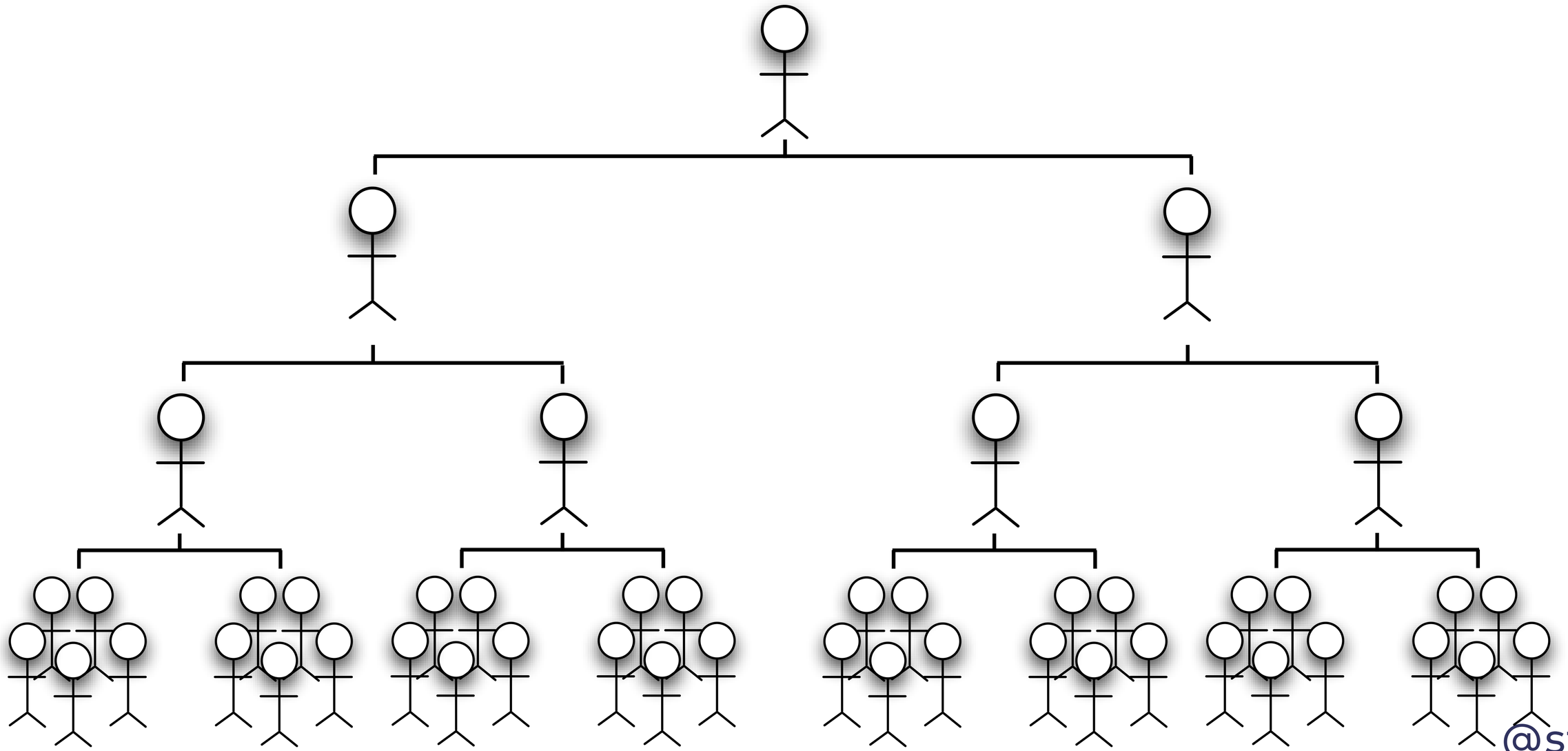
Reasons

- No (effective) centralized governance
- Non-technical senior management
- Focus on unnecessary standardization
- Strong business leaders, weak tech leaders

Consequences

- Redundancy in all aspects
- Frequent technology discussions between teams
- High integration costs and technical debt
- Slow delivery capability due to complexity
- Complex and

Antipattern: Authoritarian Regime



Antipattern: Authoritarian Regime

Description

Centralized decision making, strong standardization, homogeneous environment

Reasons

- Unpopular decisions (cost savings, product standardization, ...)
- (Perceived or real) lack of skills in “lower levels”
- Possibly due to company culture

Consequences

- Frustration and developer exodus
- Lack of innovation & speed because of bottlenecks
- Technology paralysis

Patterns

Pattern: Developer Self-Service

Pattern: Developer Self-service

Description

Project developers can access allocate resources without asking for permission

Approach

- API-based access to resources (computation, storage, network, services)
- Fine-grained security controls
- Rate-limiting
- Public, private or hybrid cloud-based

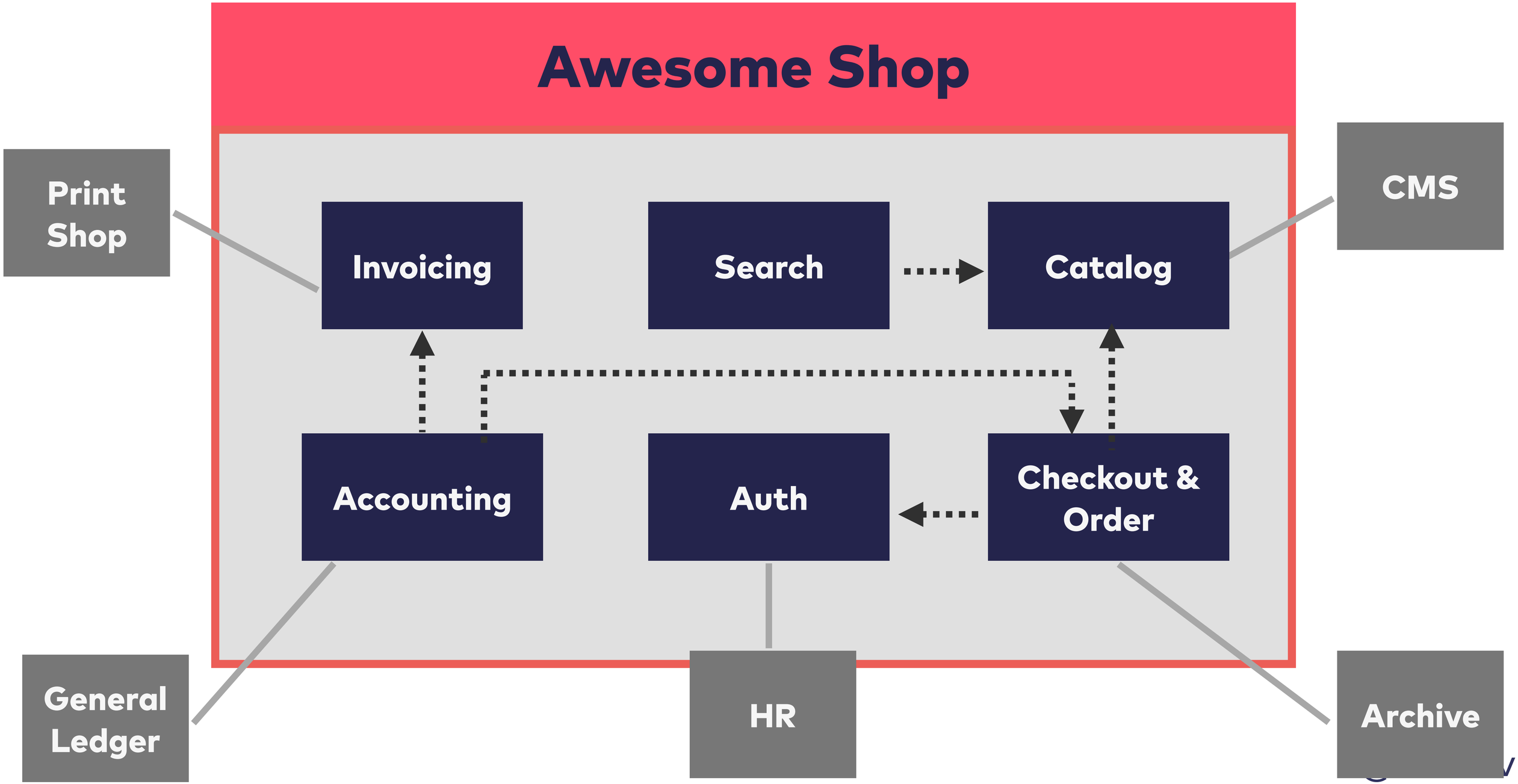
Consequences

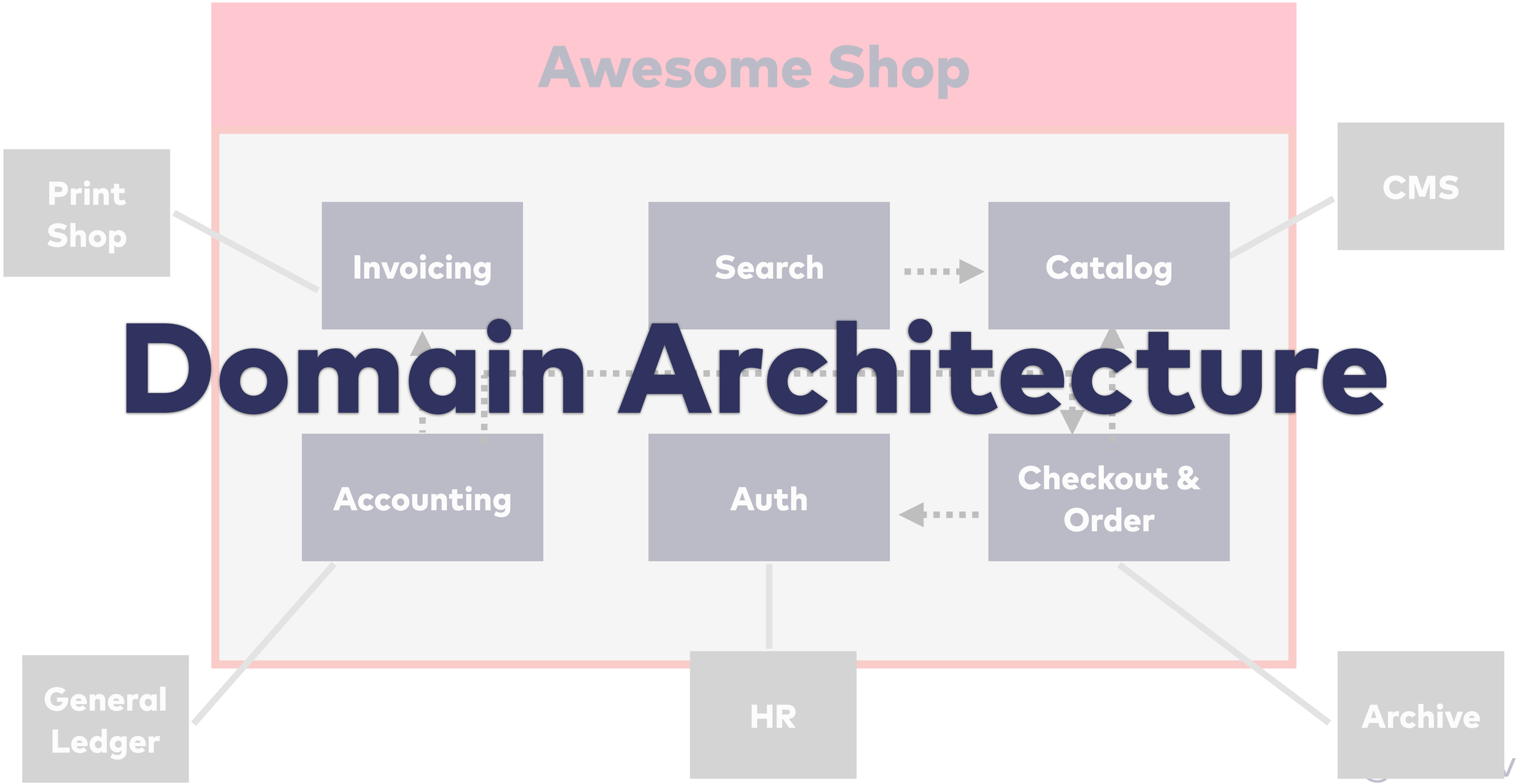
- Shorter delivery/ deployment cycles
- Support for experimentation
- Easier test setup

Pattern: Regulated Market

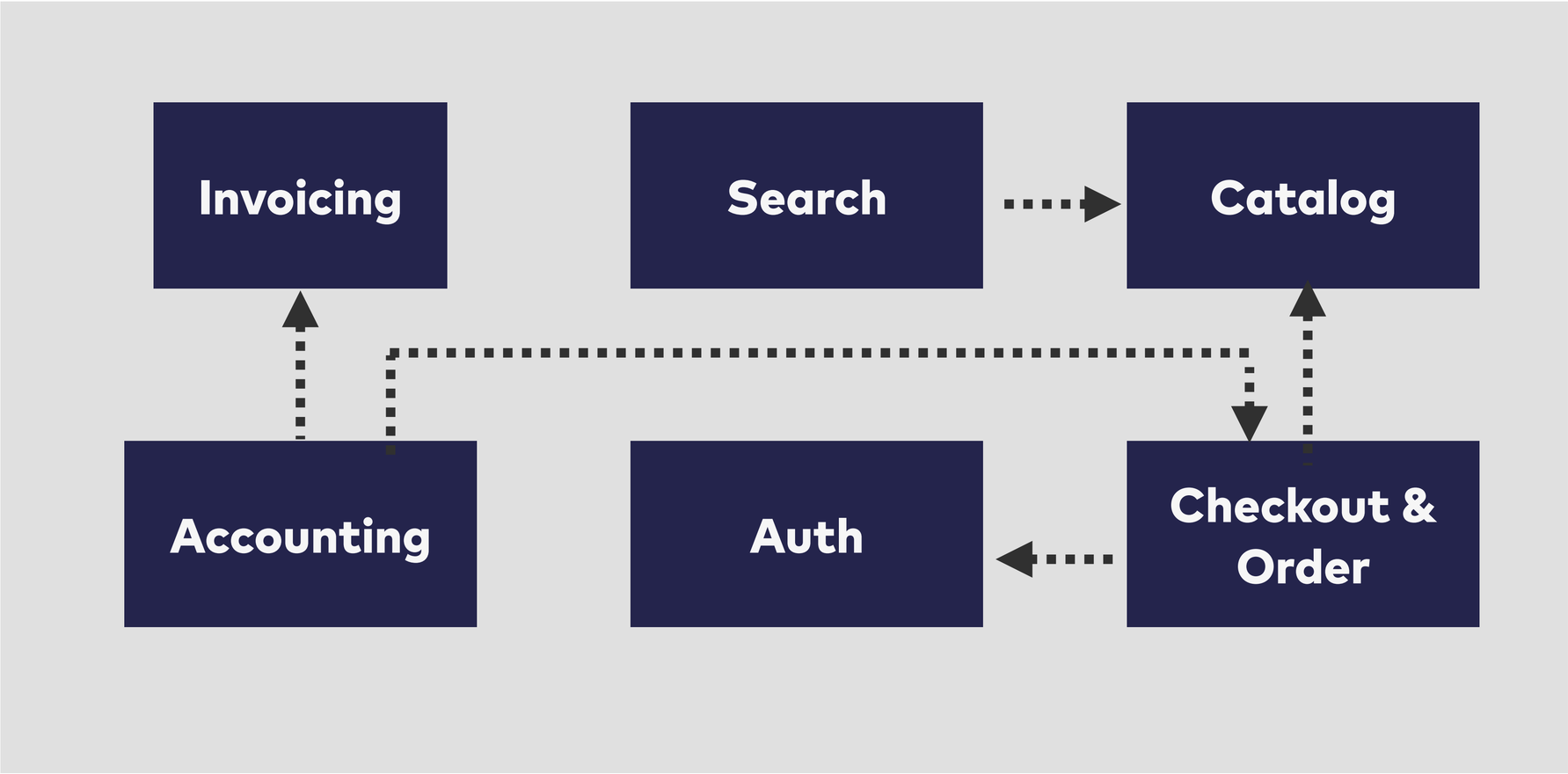

Print Shop
CMS
Awesome Shop
General Ledger
Archive
HR

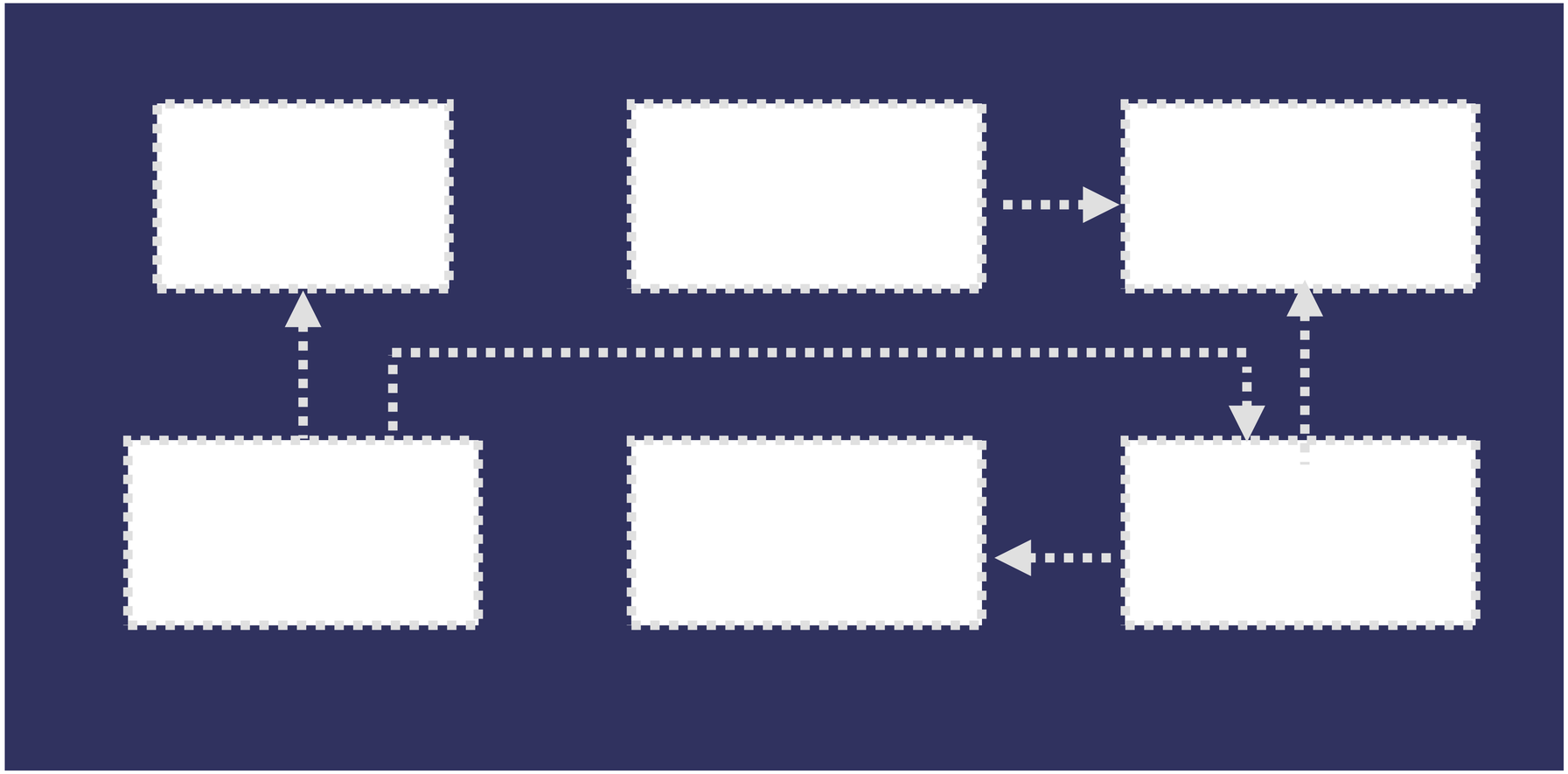

Print Shop
CMS
Awesome
Context
General Ledger
Archive
HR


Awesome Shop
Print Shop
Invoicing
Search
Catalog
CMS
Accounting
Auth
Checkout & Order
General Ledger
HR
Archive



Domain Architecture


Invoicing
Search
Catalog
Accounting
Auth
Checkout &
Order



Macro Architecture

Ruby on Rails
MySQL

NodeJS
ElasticSearch

COTS

Java
Spring Boot

OSS Product

Java
Spring Boot

Ruby on Rails
MySQL

NodeJS
ElasticSearch

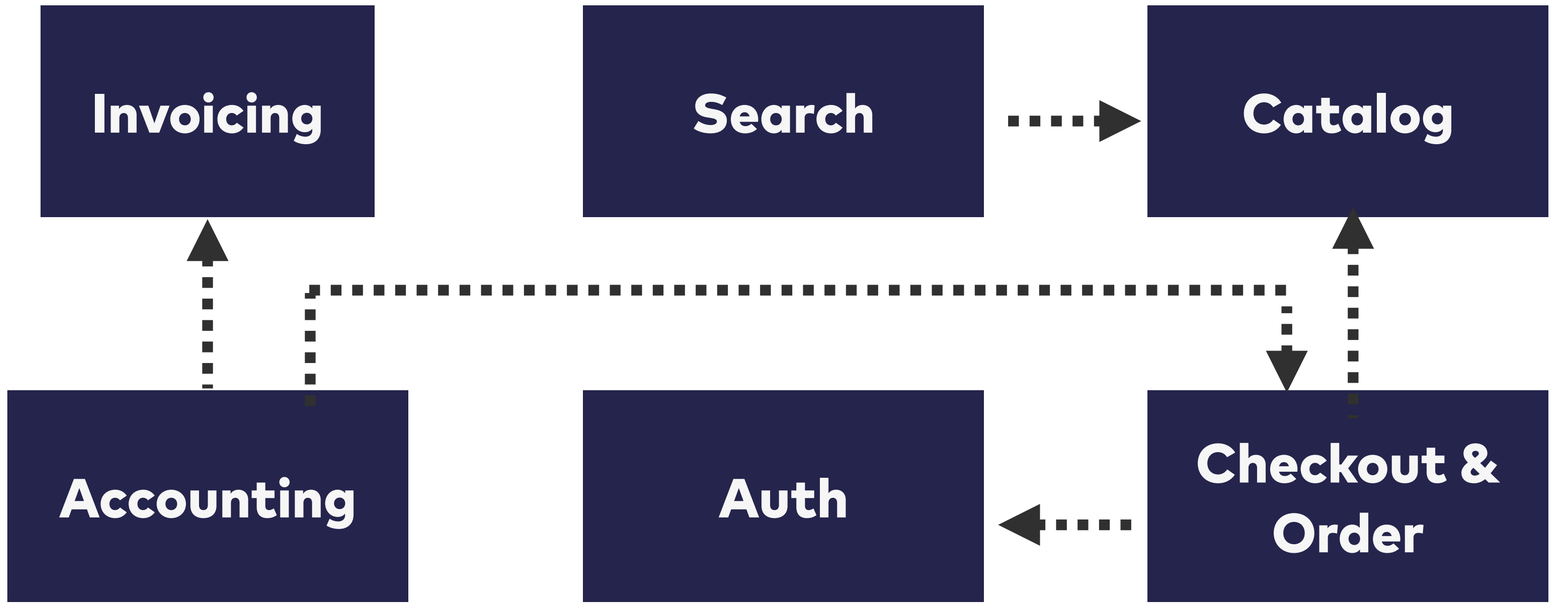
COTS

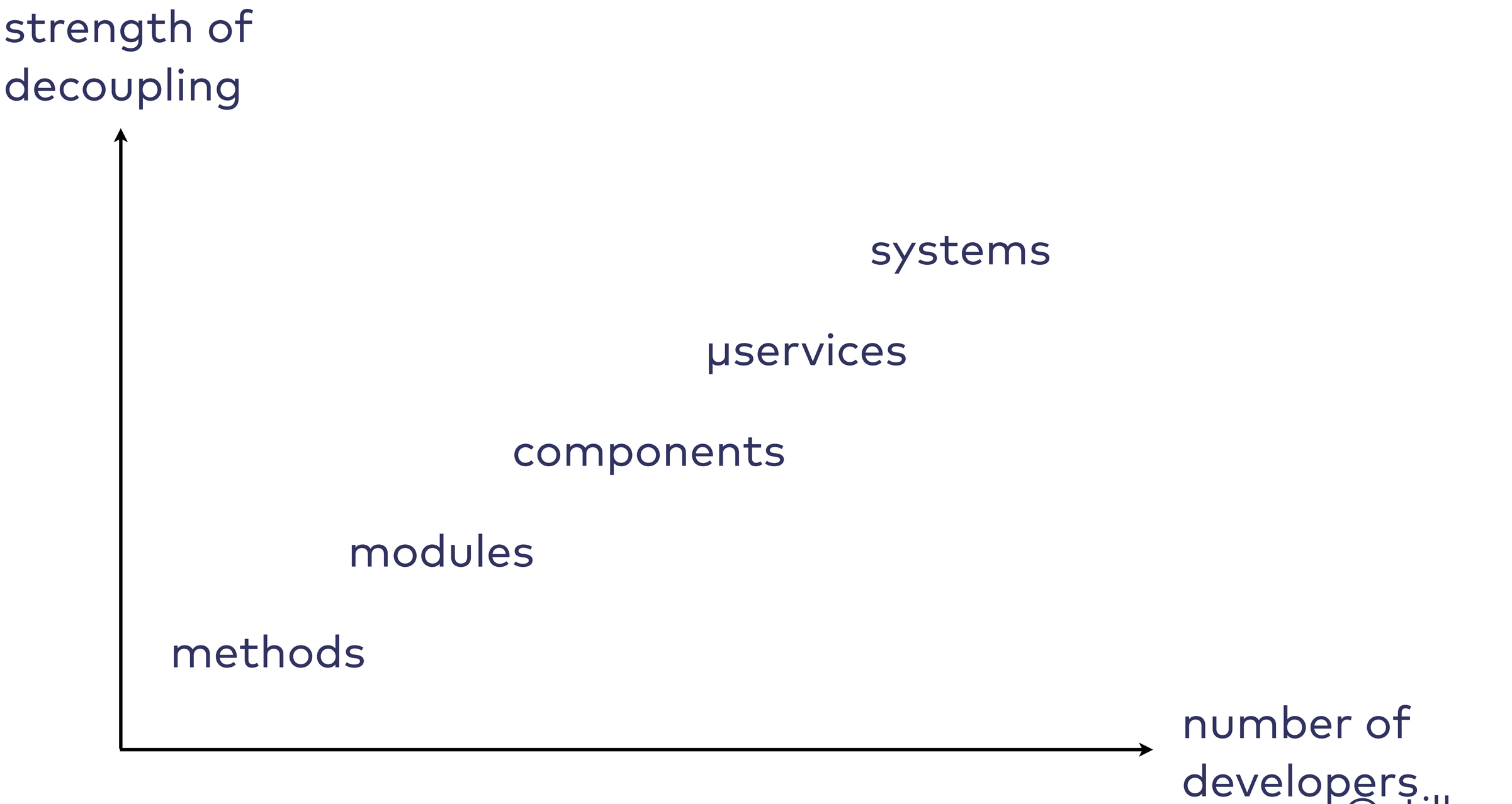
Micro Architecture

Java
Spring Boot

OSS Product

Java
Spring Boot


Invoicing
Search
Catalog
Accounting
Auth
Checkout &
Order


strength of
decoupling
systems
μservices
components
modules
methods
number of
developers

Macro Architecture

Pattern: Regulated Market

Description

Let “the free market of ideas” decide what works best, but provide a framework of rules for interoperability

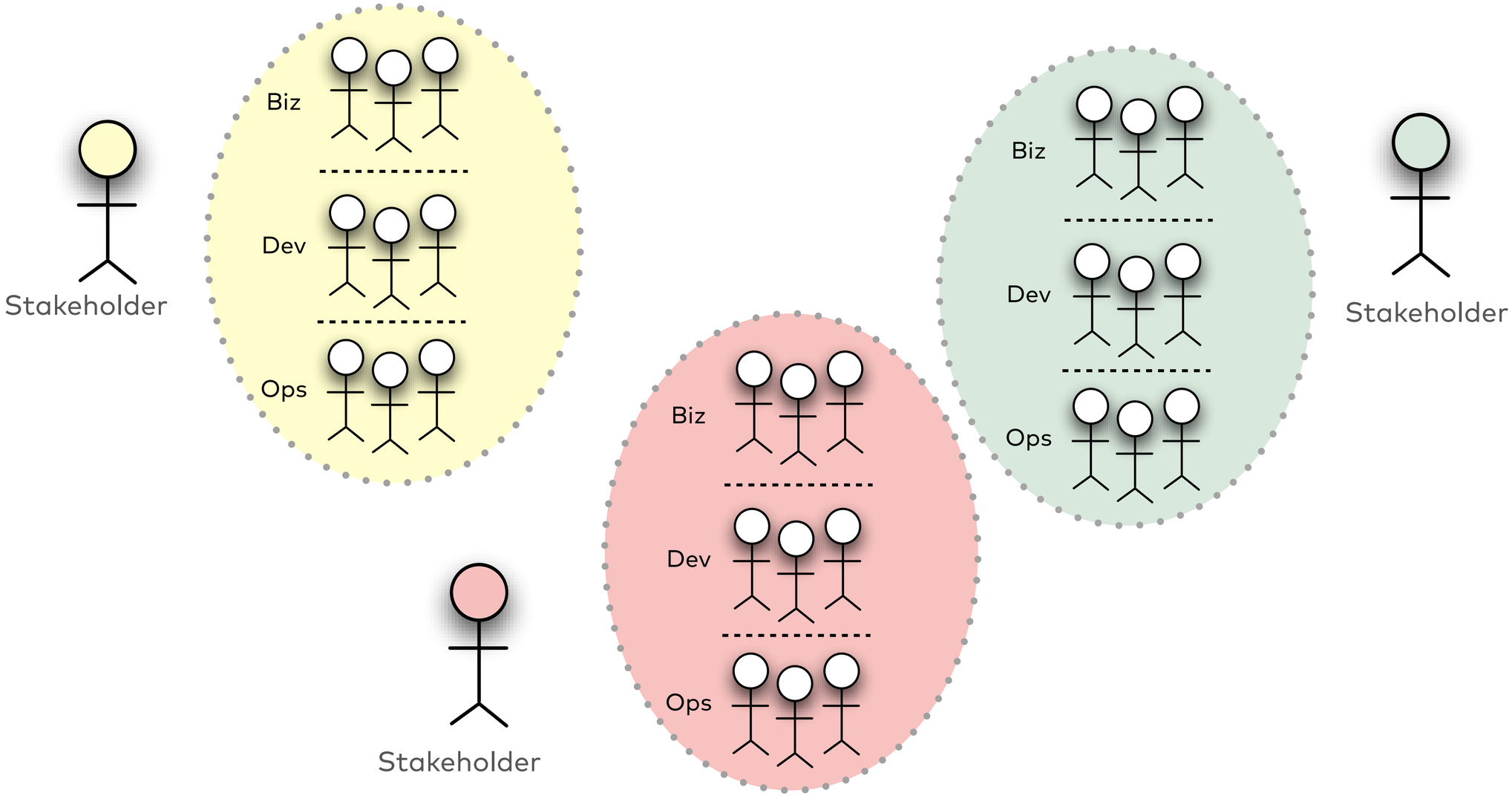
Approach

- Separate micro & macro architecture
- Strictly enforced rules for macro architecture
- Loose, minimal governance for micro architecture

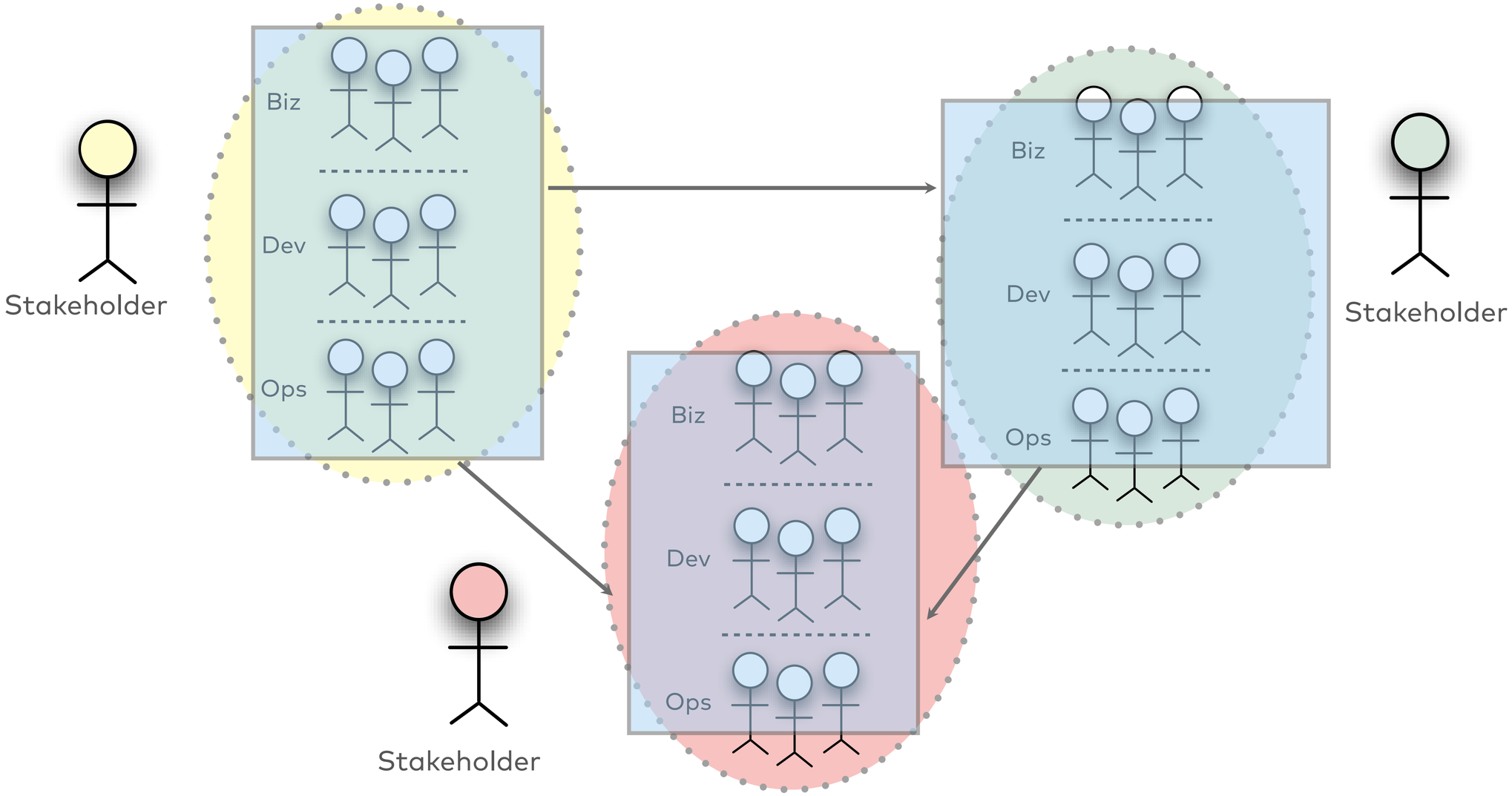
Consequences

- Motivated developers
- Experimentation with different micro architecture approaches possible
- Best-of-breed approach
- Local optima

Pattern: Autonomous Cells



Pattern: Autonomous Cells



Pattern: Autonomous Cells

Description

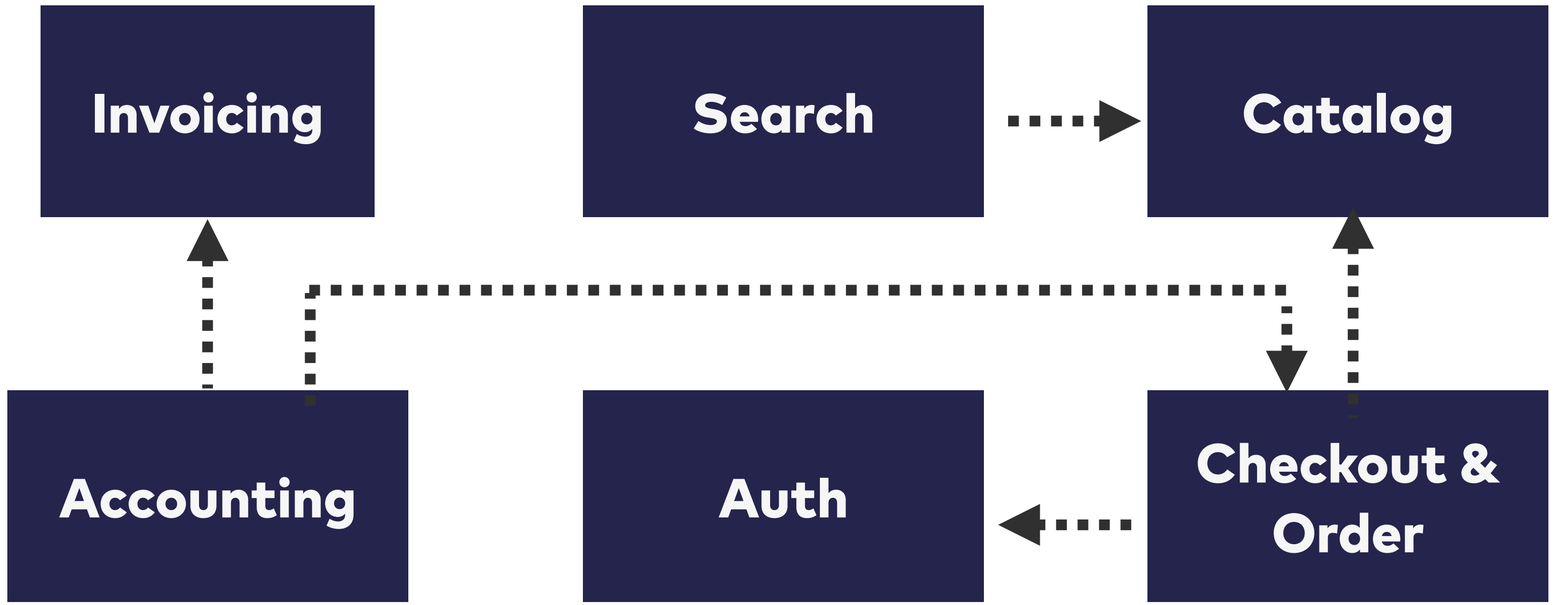
Decentralized, domain-focused cells with maximum authority over all aspects of a set of capabilities

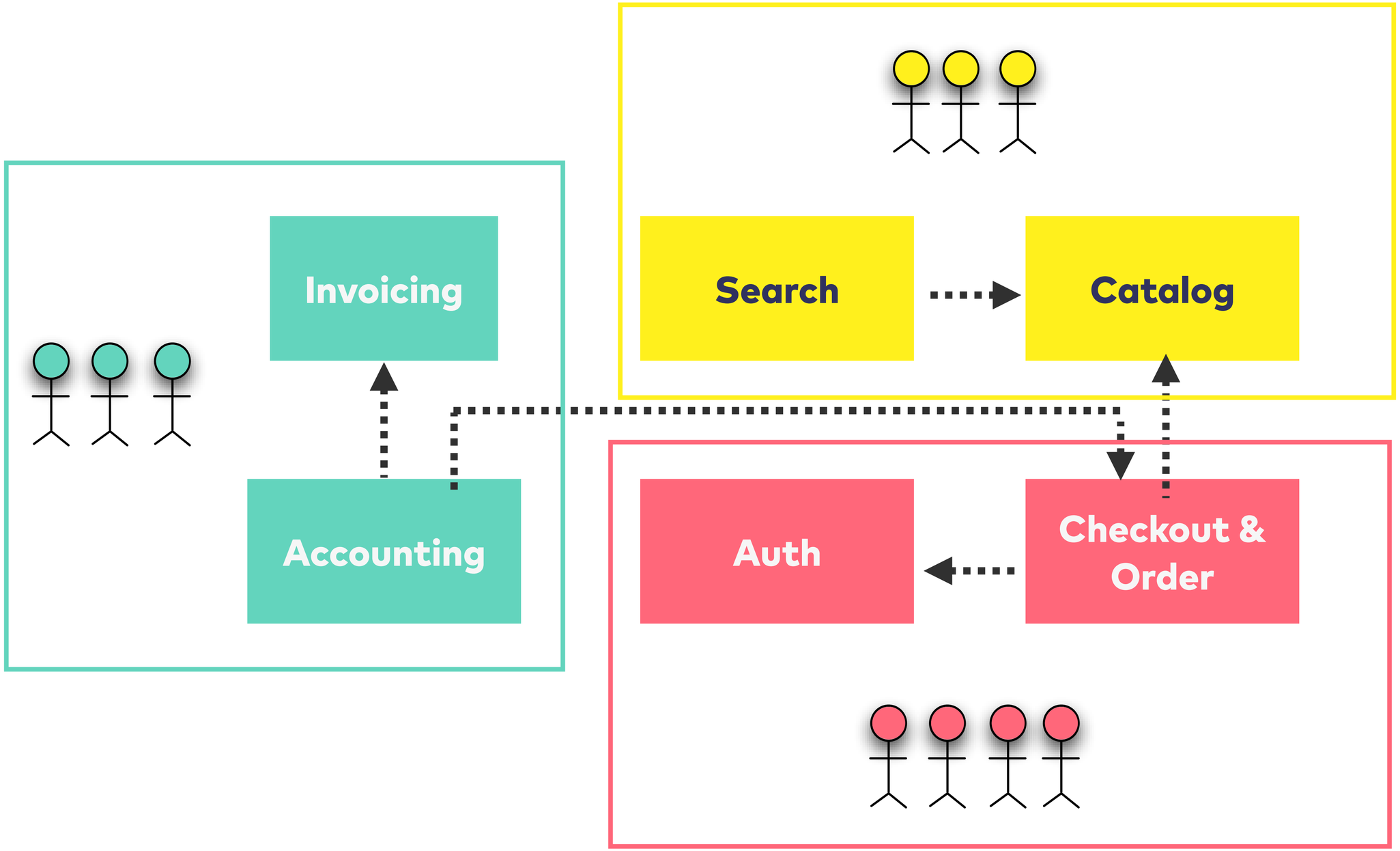
Approach

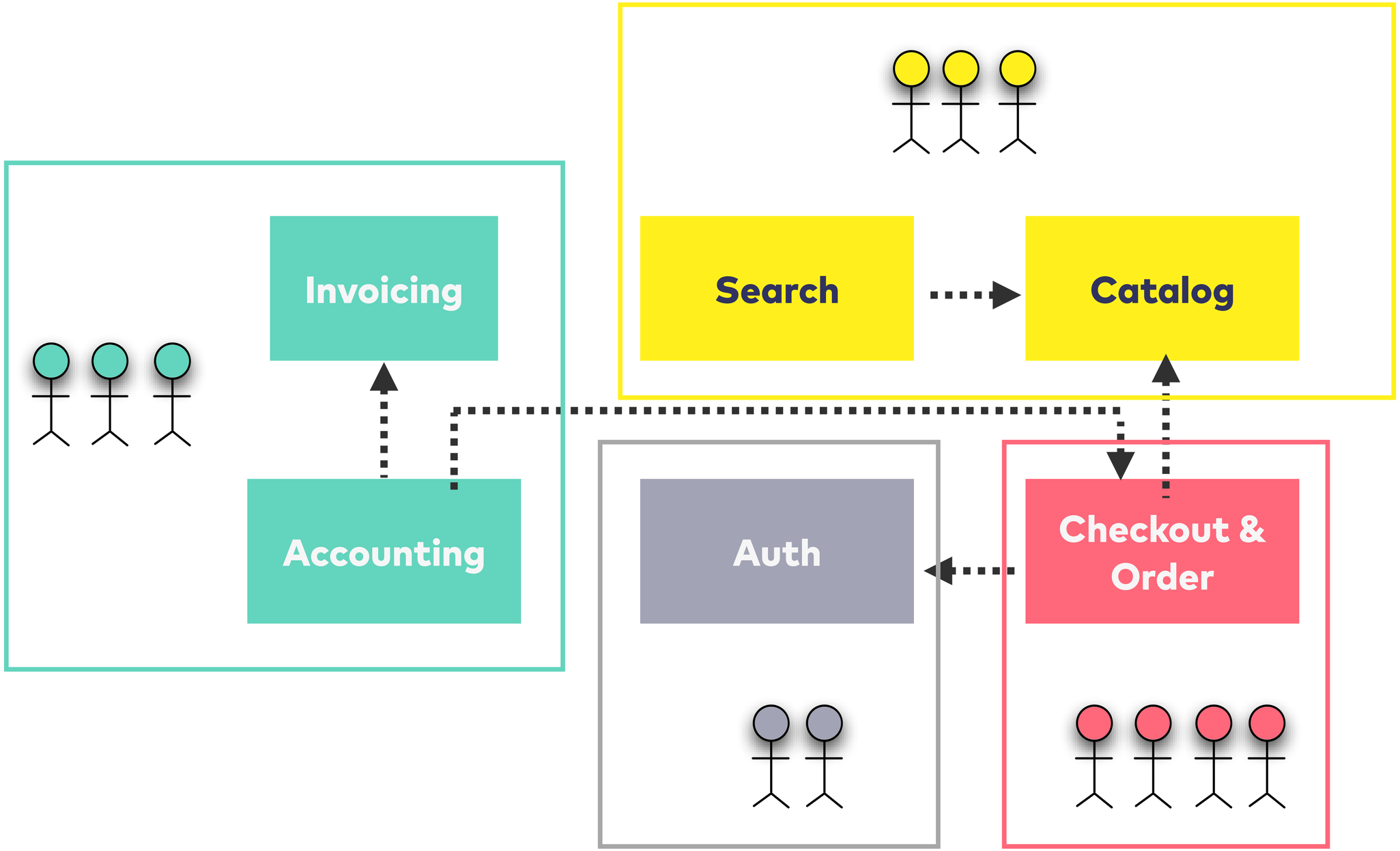
- Decisions are made locally on all aspects of a solution
- Success is measured via customer-oriented KPIs
- Cross-functional team with biz, dev, ops skills

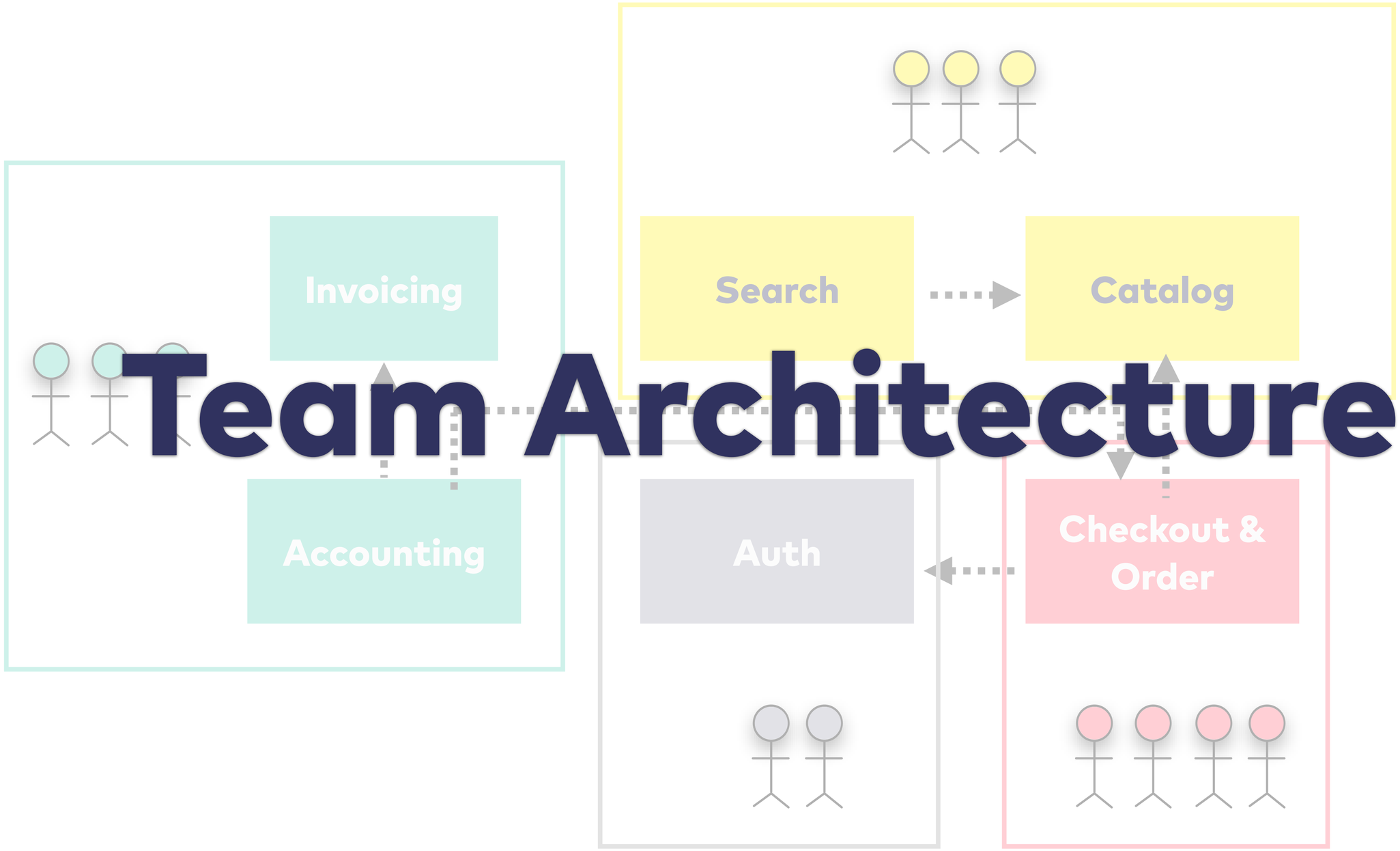
Consequences

- Customer/end user focus
- Decentralized delivery capability
- Speed as #1 priority
- “Full-stack” requirement for developers and other roles
- Redundancy instead of centralization


Invoicing
Search
Catalog
Accounting
Auth
Checkout &
Order


Invoicing
Accounting
Search
Catalog
Auth
Checkout &
Order


Invoicing
Search
Catalog
Accounting
Auth
Checkout &
Order



Team Architecture

@stilkov

Pattern: Evolutionary Architecture

Pattern: Evolutionary Architecture

Description

Architecture is constructed so it can evolve as much as possible over the course of (ideally indefinite) time

Approach

- Separation of large domain into “islands of change”
- Design for replacement, not for re-use
- Minimization of shared dependencies

Consequences

- Cell metaphor: Renewal over time
- Experimentation with different micro architecture approaches possible

Pattern: Marketing-based Governance

Pattern: Marketing-based Governance

Description

Architectural approaches are evangelized instead of mandated

Approach

- Disseminate information via blogs, brown-bag sessions, public talks
- Architects as communicators
- Integration with public/community work

Consequences

- More heterogeneity
- Similarity to industry
- Decisions made based on a solution's merit
- Bottom-up modernization

Recommendations

1.
Acquire domain knowledge and focus on business value

2.
Partner with business stakeholders

3.
Create evolvable structures

4.
Be aware of the interplay between architecture, processes, organization, and humans

5.
Create value – or get out of the way of those who do as quickly as possible

Thank you! Any questions?

INNOQ
www.innoq.com

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

innoQ Deutschland GmbH

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

Ohlauer Str. 43
10999 Berlin
Germany
+49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany
+49 2173 3366-0

Kreuzstr. 16
80331 München
Germany
+49 2173 3366-0

Hermannstrasse 13
20095 Hamburg
Germany
+49 2173 3366-0

innoQ Schweiz GmbH

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

INNOQ

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

@stilkov

Image Credit

https://pixabay.com/en/chaos-room-untidy-dirty-messy-627218/

https://commons.wikimedia.org/wiki/File:Wroclaw_Daily_Market.jpg

https://pixabay.com/en/smartphone-face-man-old-baby-1790833/

https://pixabay.com/en/marketing-customer-center-2483856/

http://maxpixel.freegreatpicture.com/Board-Arrow-Note-Garden-Shield-Wc-Self-Service-863157

https://commons.wikimedia.org/wiki/File:Chicago_Campus_Conference.JPG