Agile Cambridge 2019

Agile Architecture and Innovation

Stefan Tilkov
stefan.tilkov@innoq.com
@stilkov

INNOQ

Topics we'll ignore today:

Whether you need "Architecture"

(you have no choice)

Whether your Agile project needs an "Architect"

(no, but someone will do that architecting)

What to document

(structure, dependencies, principles, choices)

Whether there should be "Architecture Stories"

(no)

Whether the whole discussion might actually be quite pointless

(possibly)

Instead:
Architecture

Architecture Manifesto Attempt

Conscious trade-offs over **emerging architecture**
Documented rationale over **quick ad-hoc decisions**
Sustained changeability over **easy initial development**
Design for replacement over **design for re-use**
Simplicity over **fast delivery**

(Software) Architecture Definitions

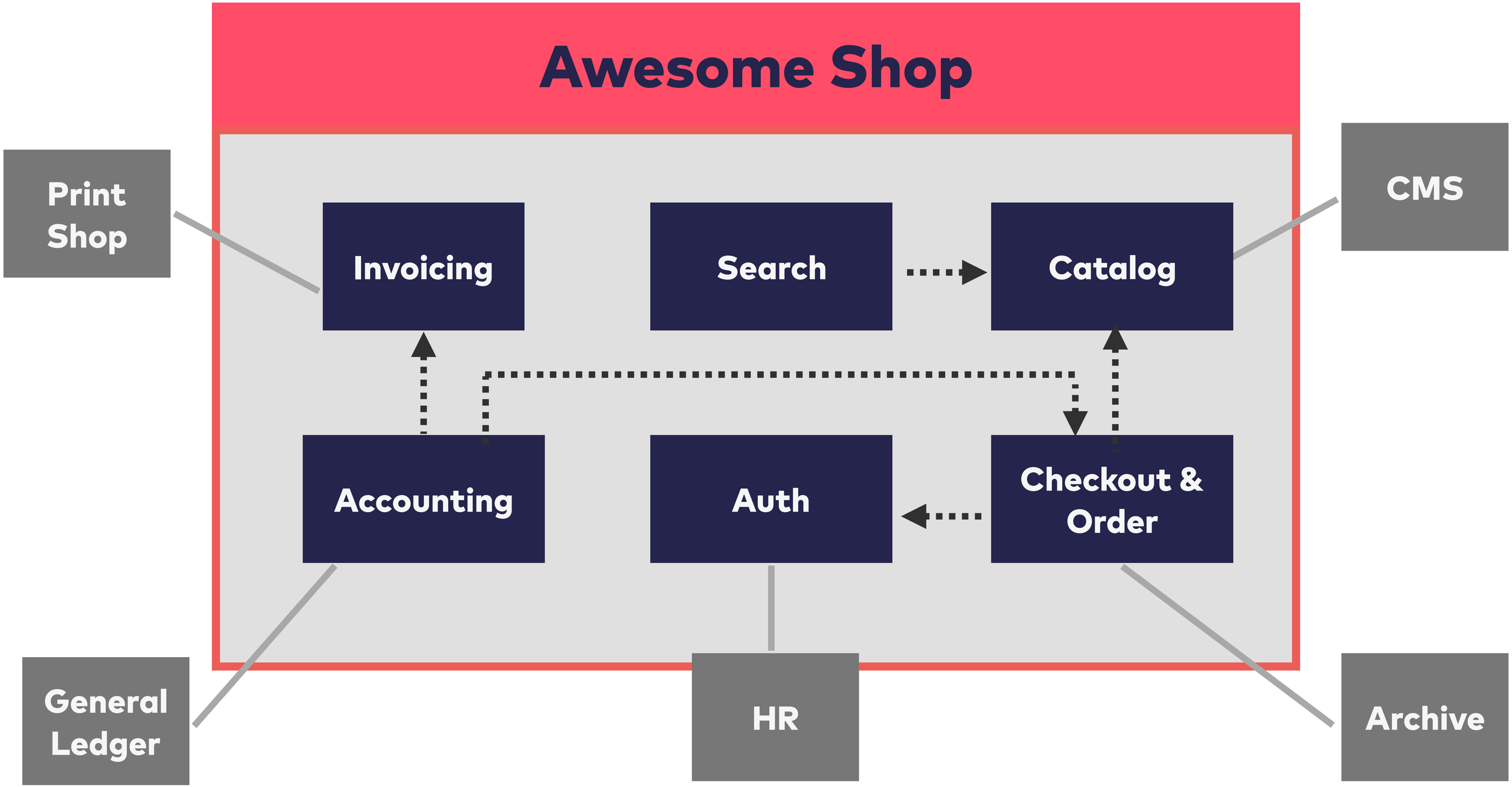
A system's elements, their relationships, and the rules and principles that govern their design and evolution

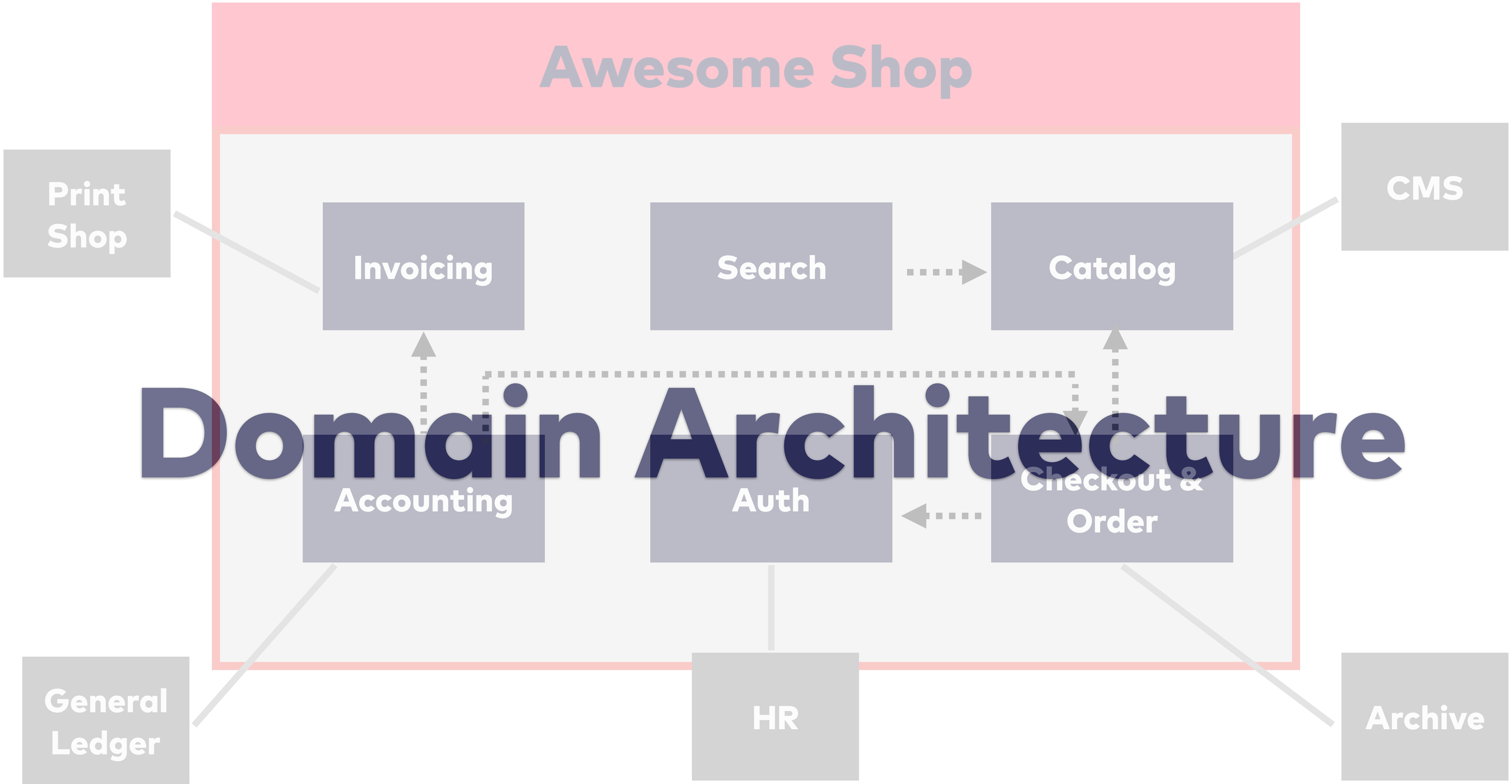
Whatever the architect considers important enough to merit their attention

Decisions that you want to be correct because they are costly to change

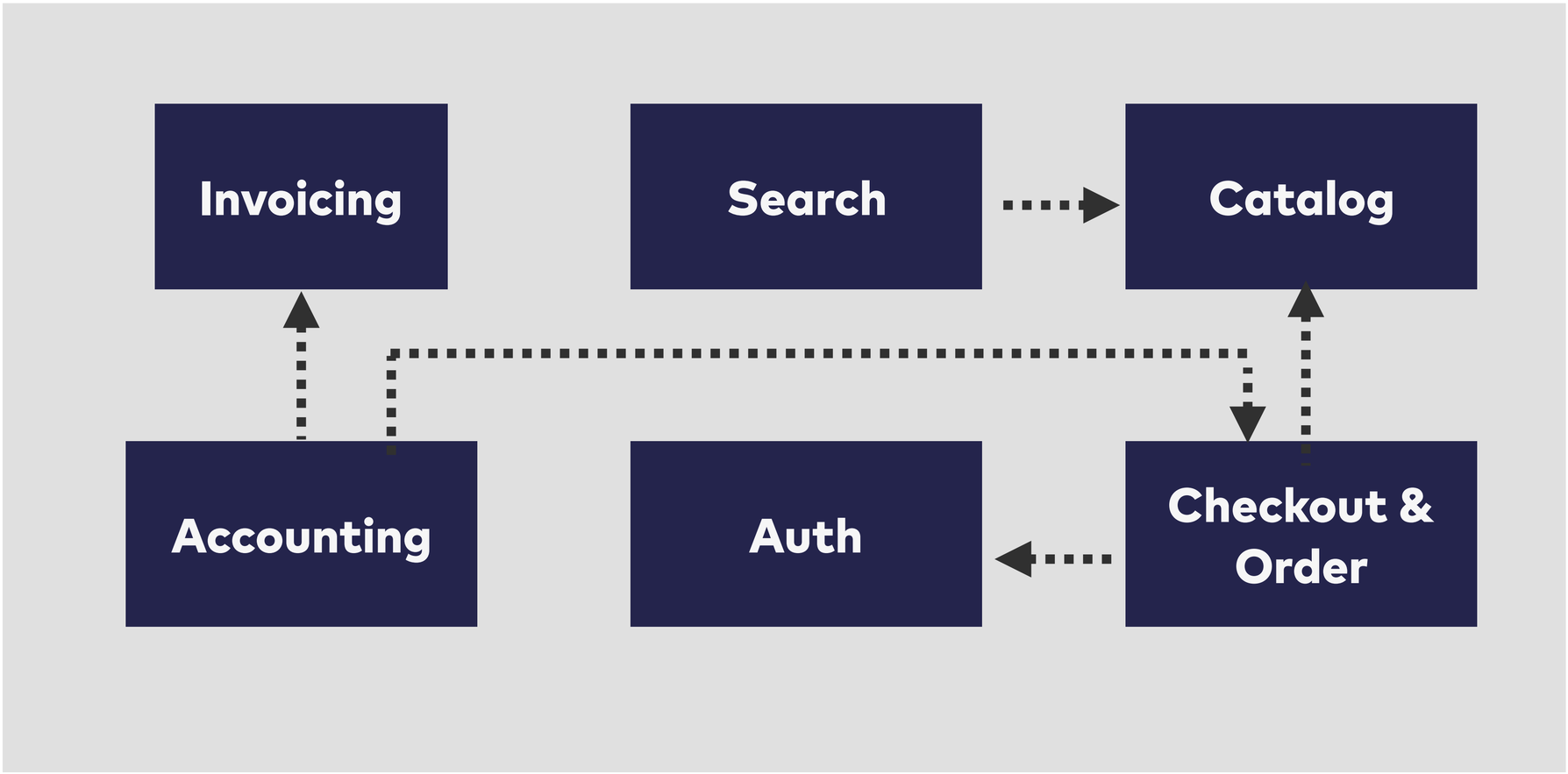

Print Shop
CMS
Awesome Shop
General Ledger
Archive
HR


Print Shop
CMS
Awesome Shop
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

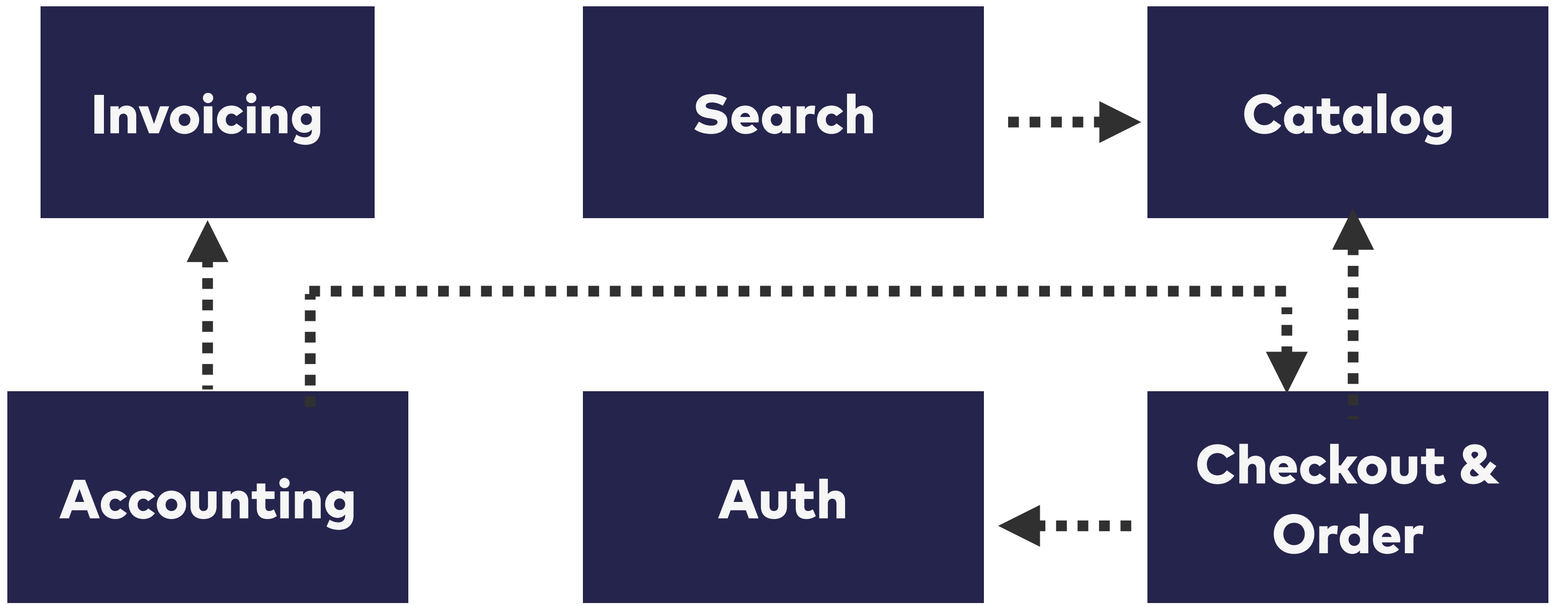
COTS

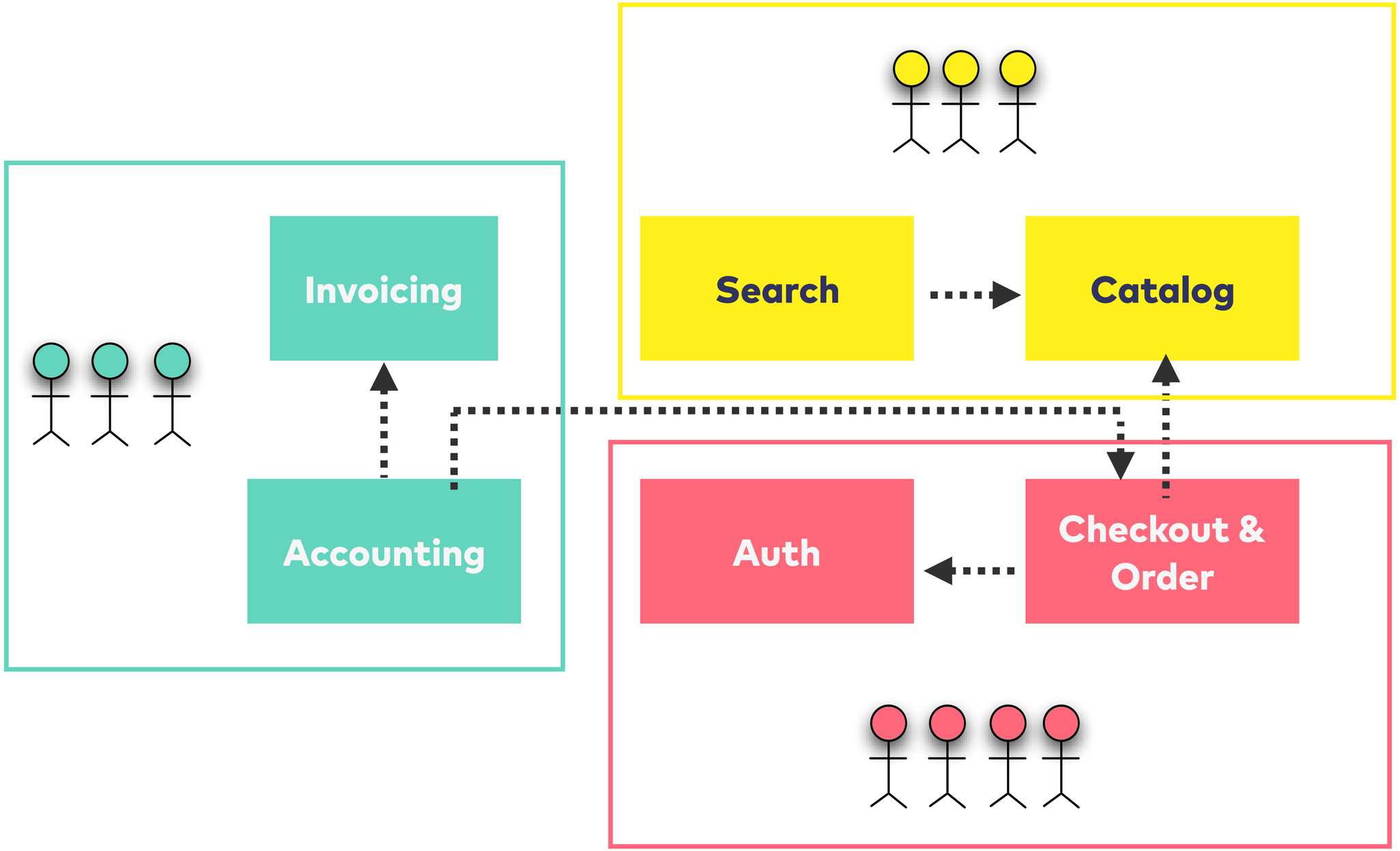
Java
Spring Boot

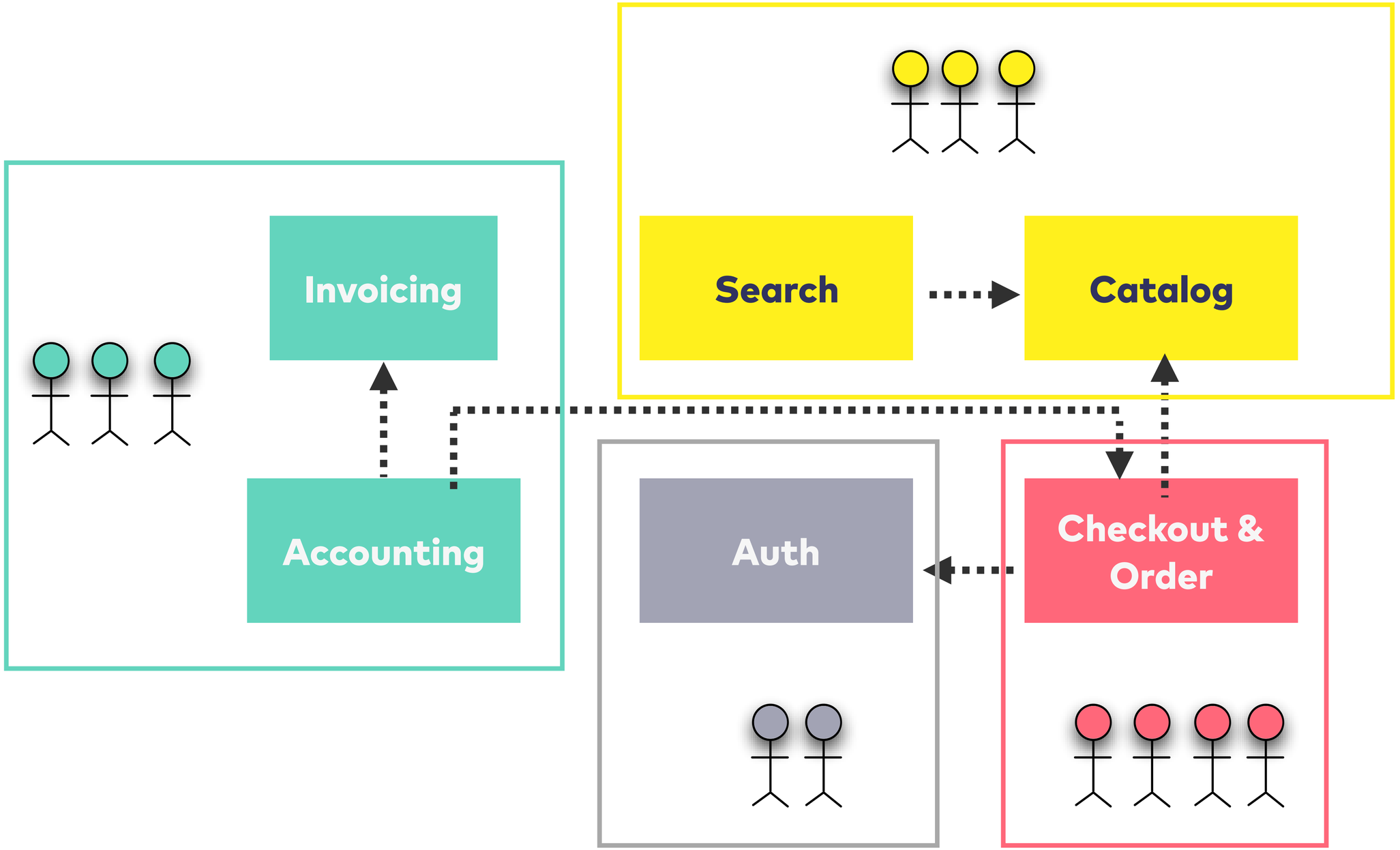
OSS Product

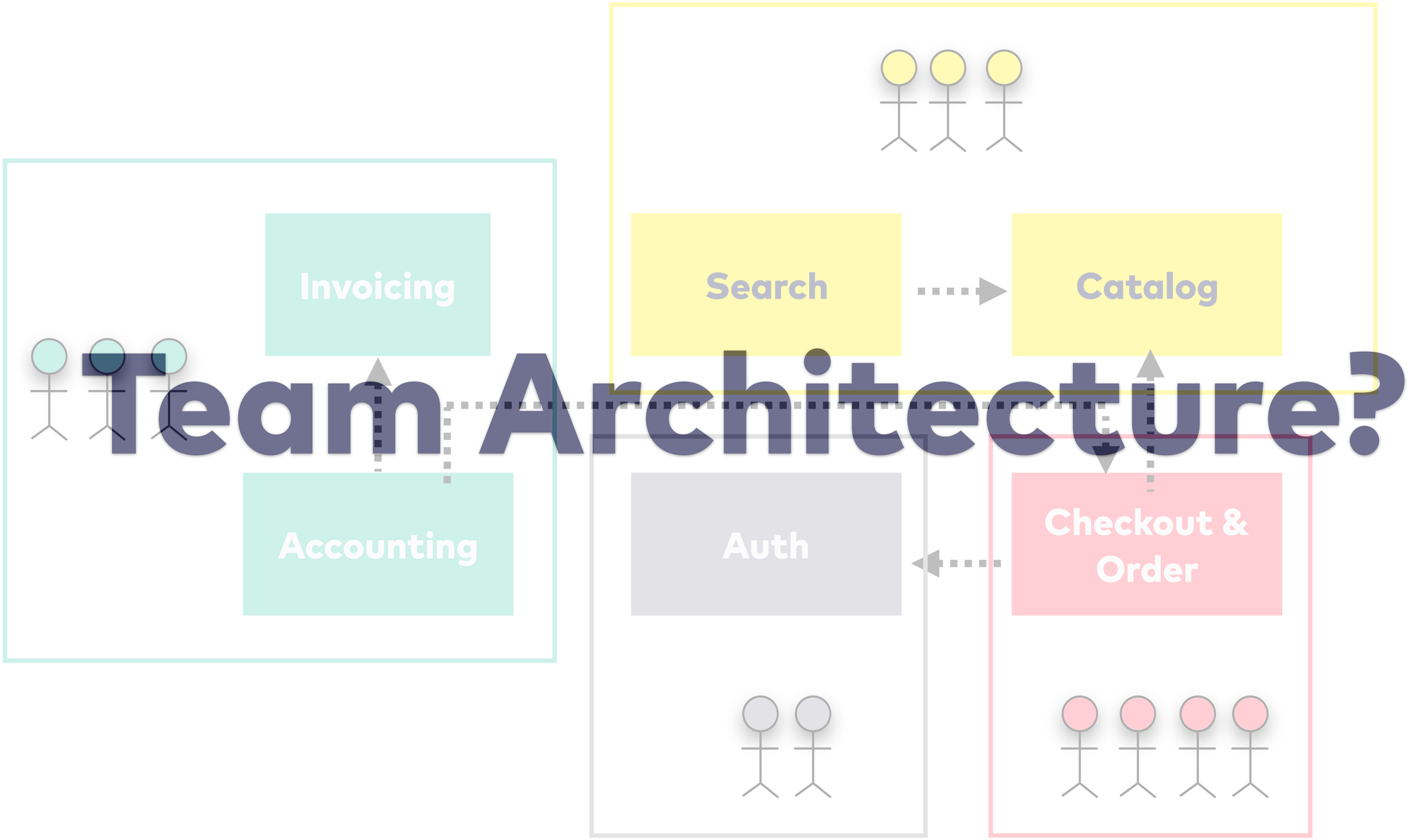
Java
Spring Boot


Ruby on Rails
MySQL
NodeJS
ElasticSearch
COTS
Micro Architecture
Java
Spring Boot
OSS Product
Java
Spring Boot


Invoicing
Search
Catalog
Accounting
Auth
Checkout &
Order


Invoicing
Accounting
Search
Catalog
Auth
Checkout &
Order


Invoicing
Accounting
Search
Catalog
Auth
Checkout &
Order



Team Architecture?

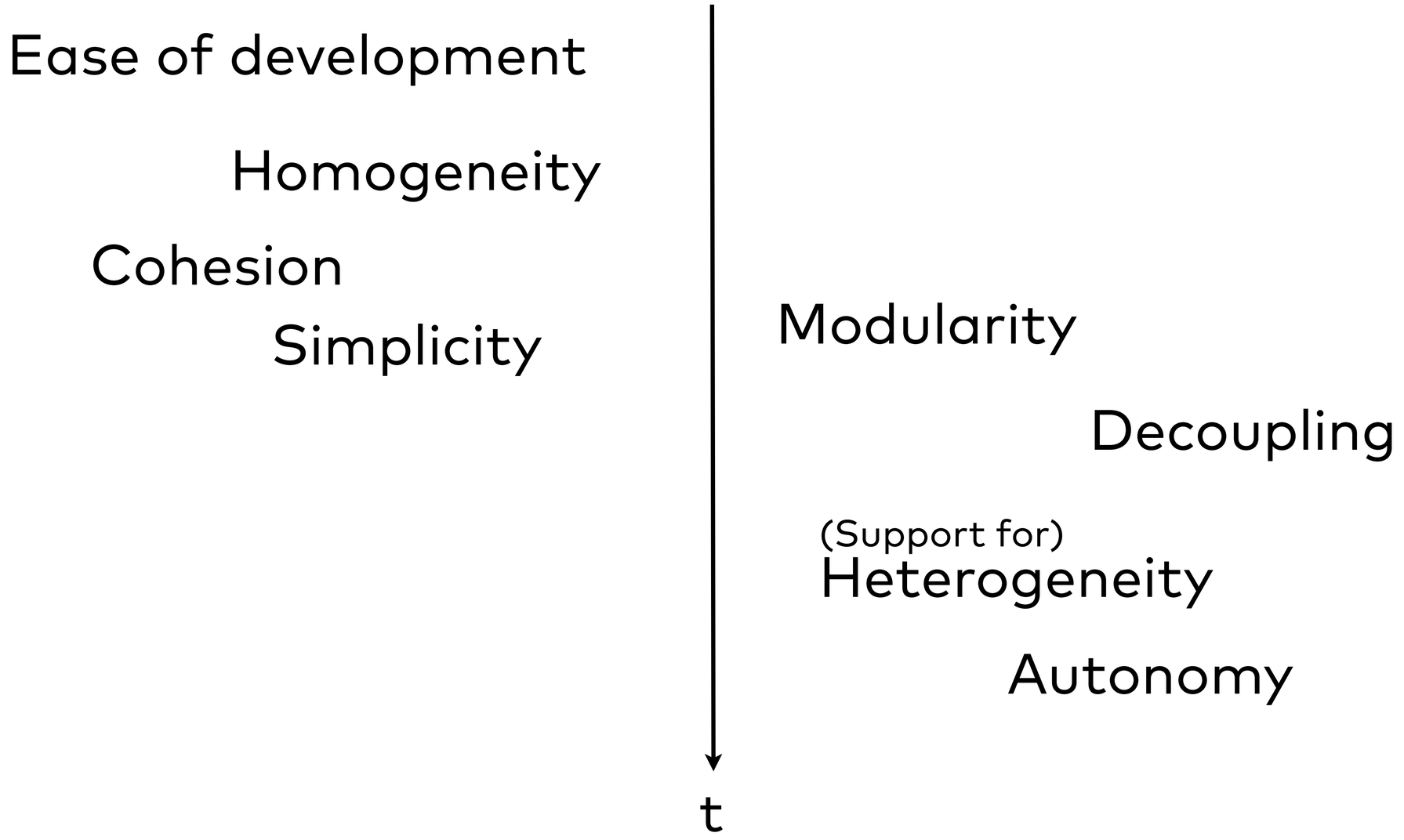
If your goal is to support autonomous teams, architecture is an essential ingredient

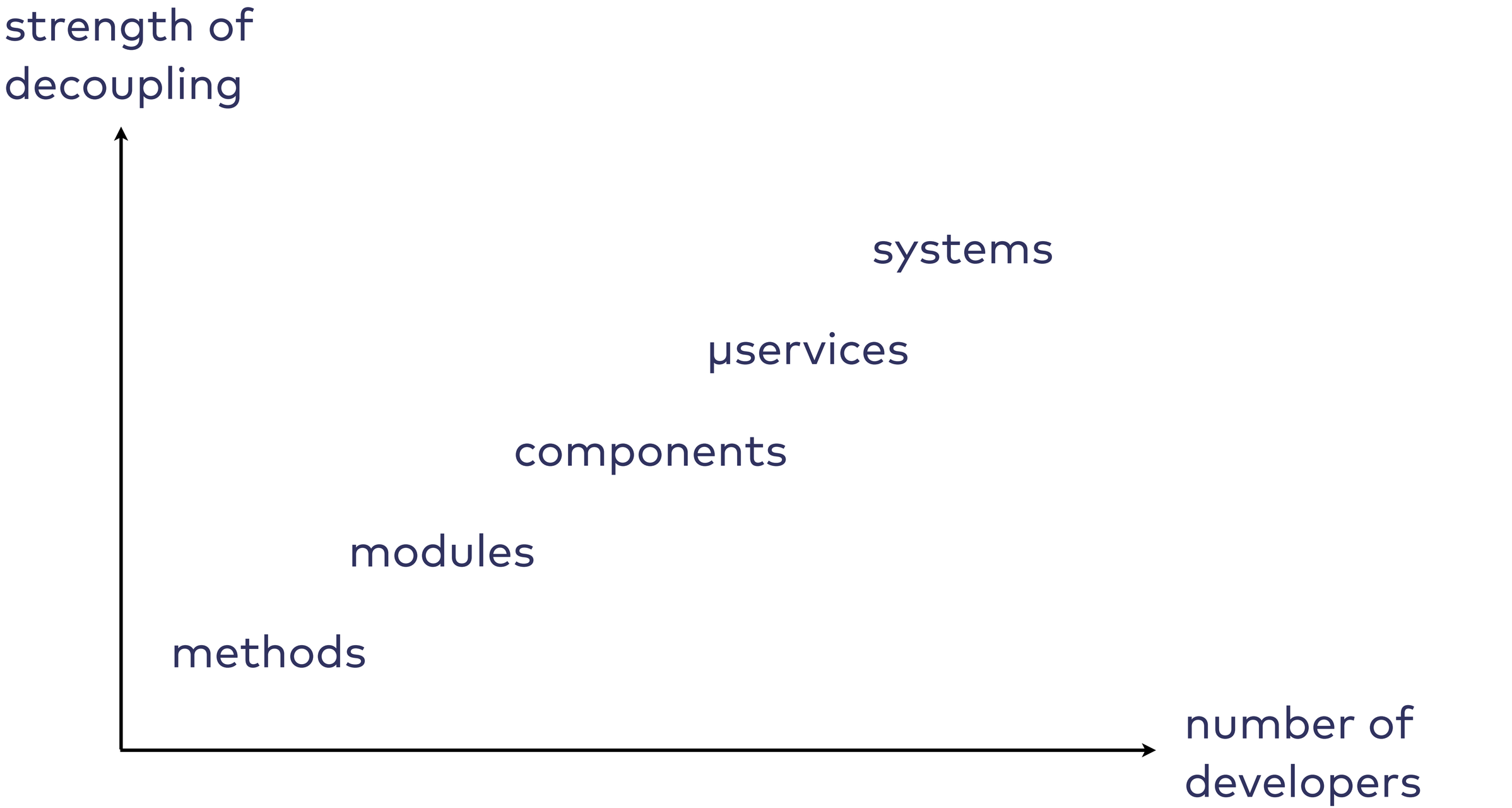
Size is the #1 enemy of agility. Keep your systems as small as you can.

@stilkov

Coming up with the “right” system boundaries is an architecture activity that must be done first

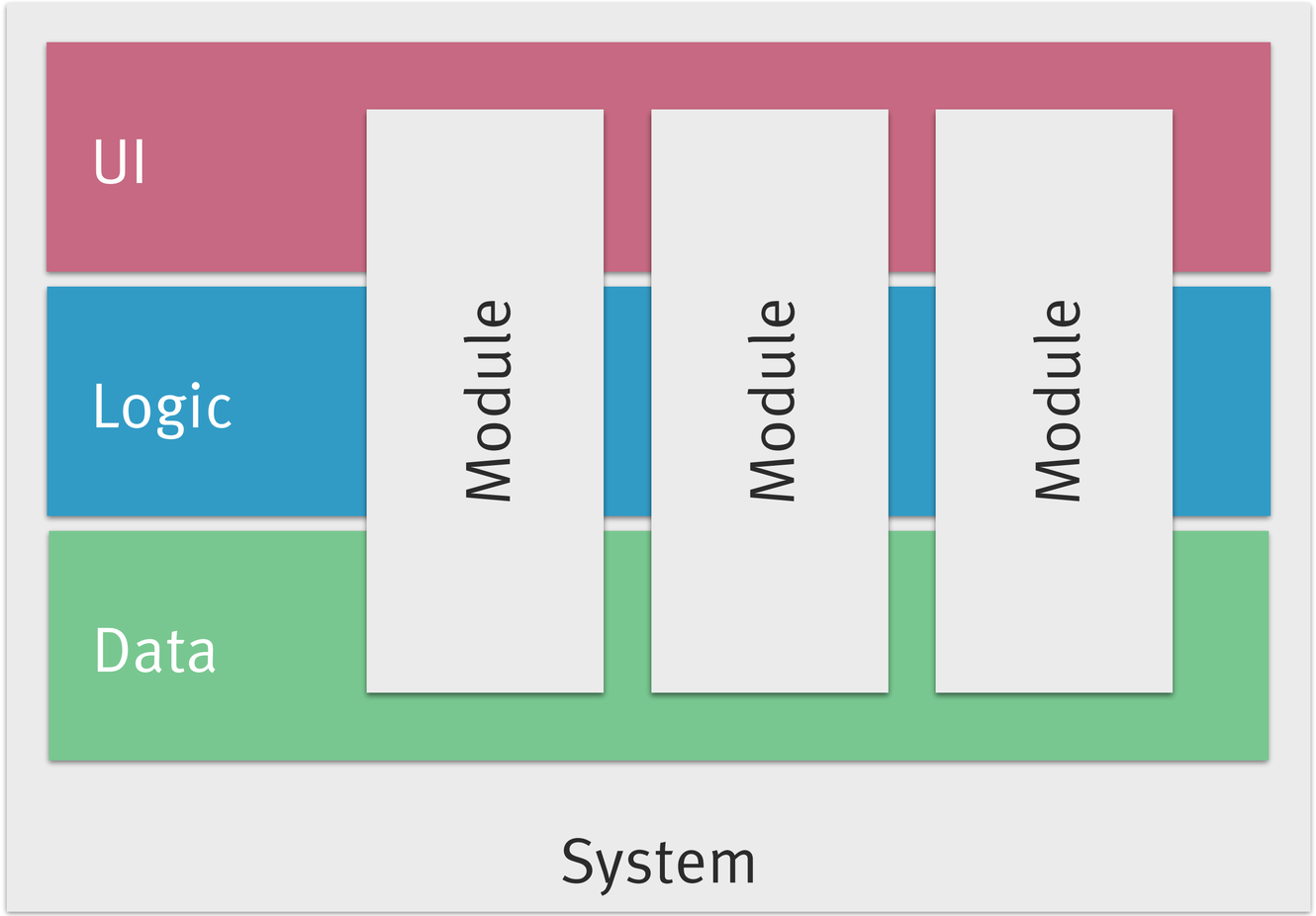
Managing dependencies is the most important ongoing architecture task


Ease of development
Homogeneity
Cohesion
Simplicity
Modularity
Decoupling
(Support for)
Heterogeneity
Autonomy
t

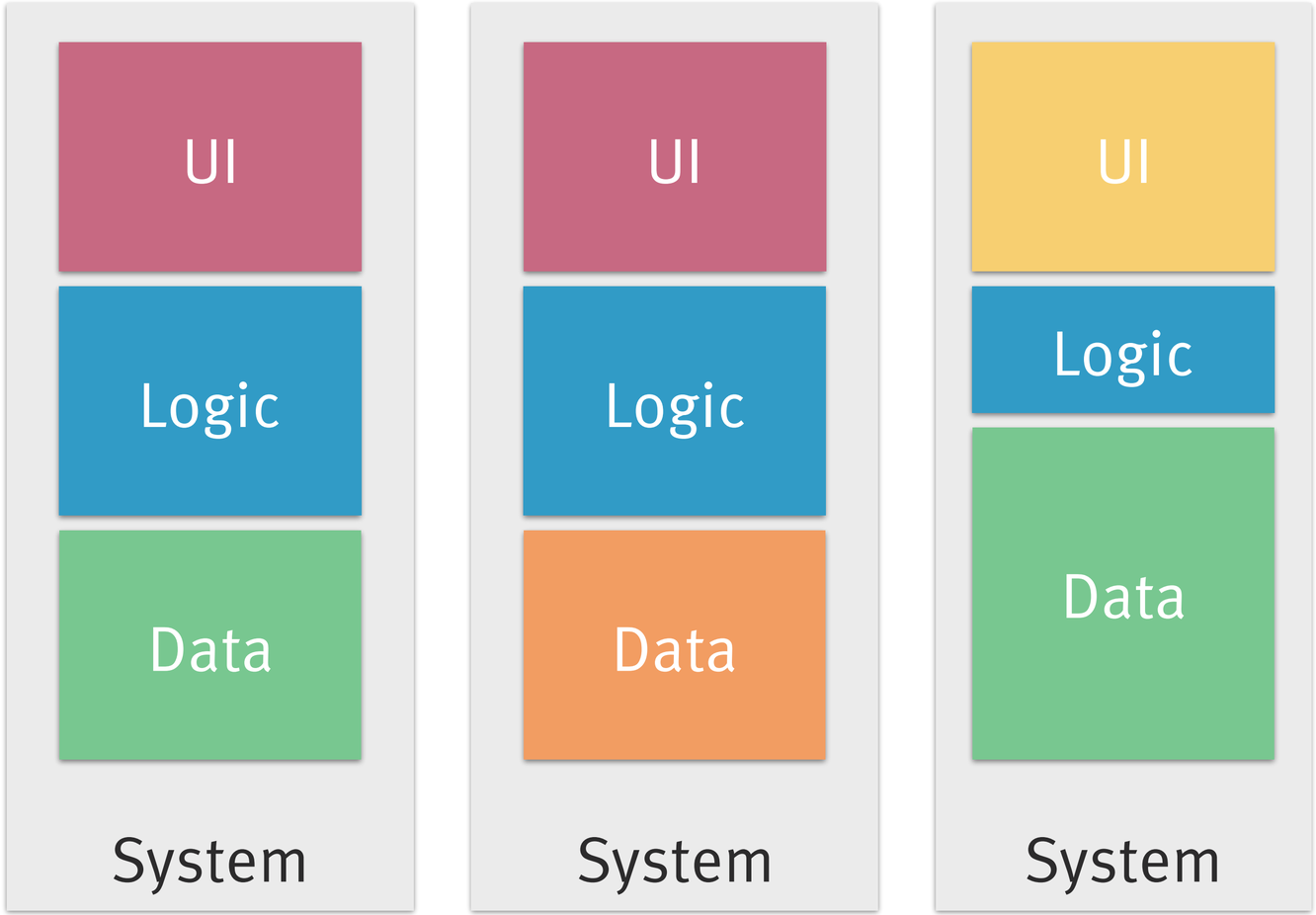

strength of
decoupling
systems
μservices
components
modules
methods
number of
developers

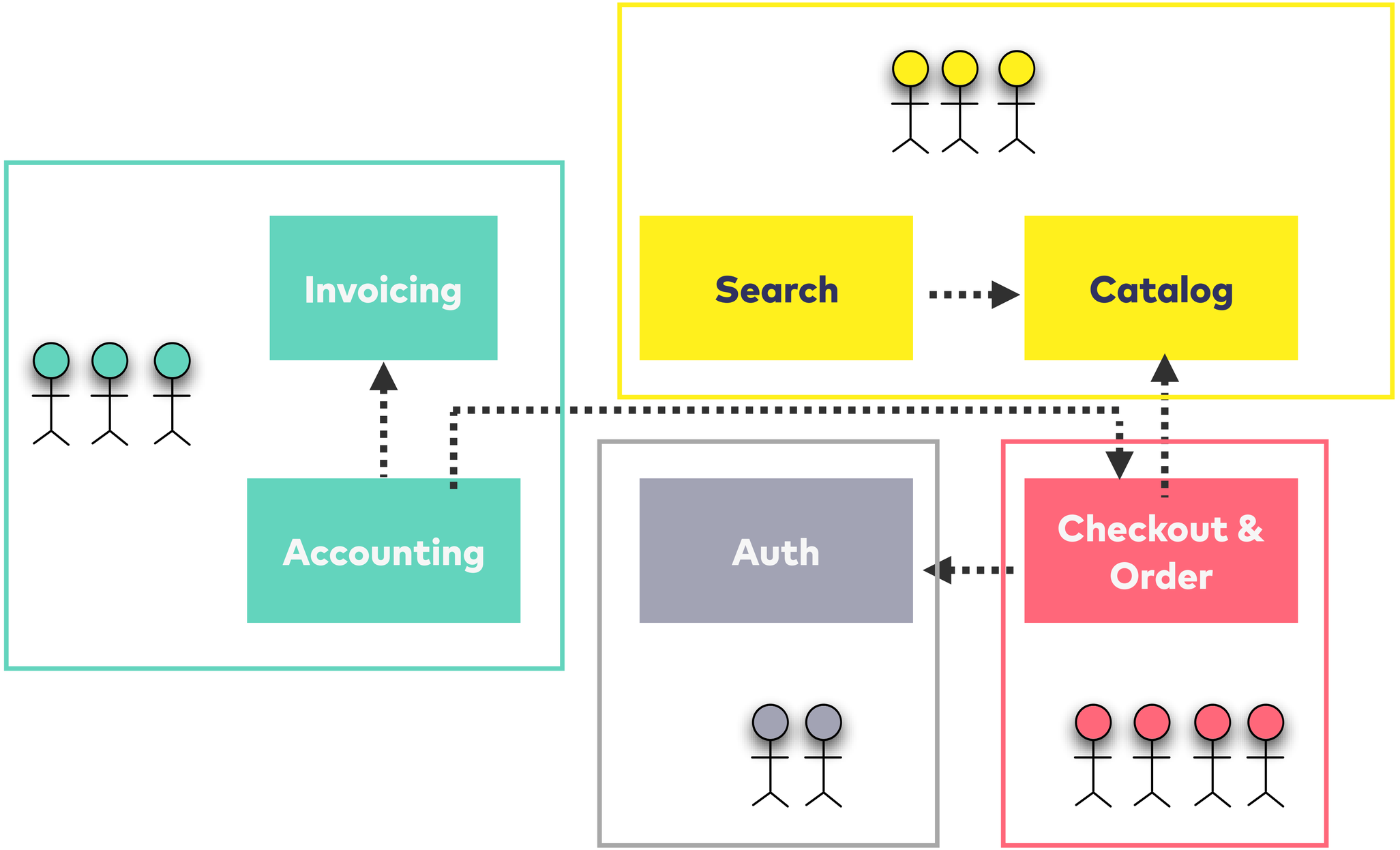
@stilkov

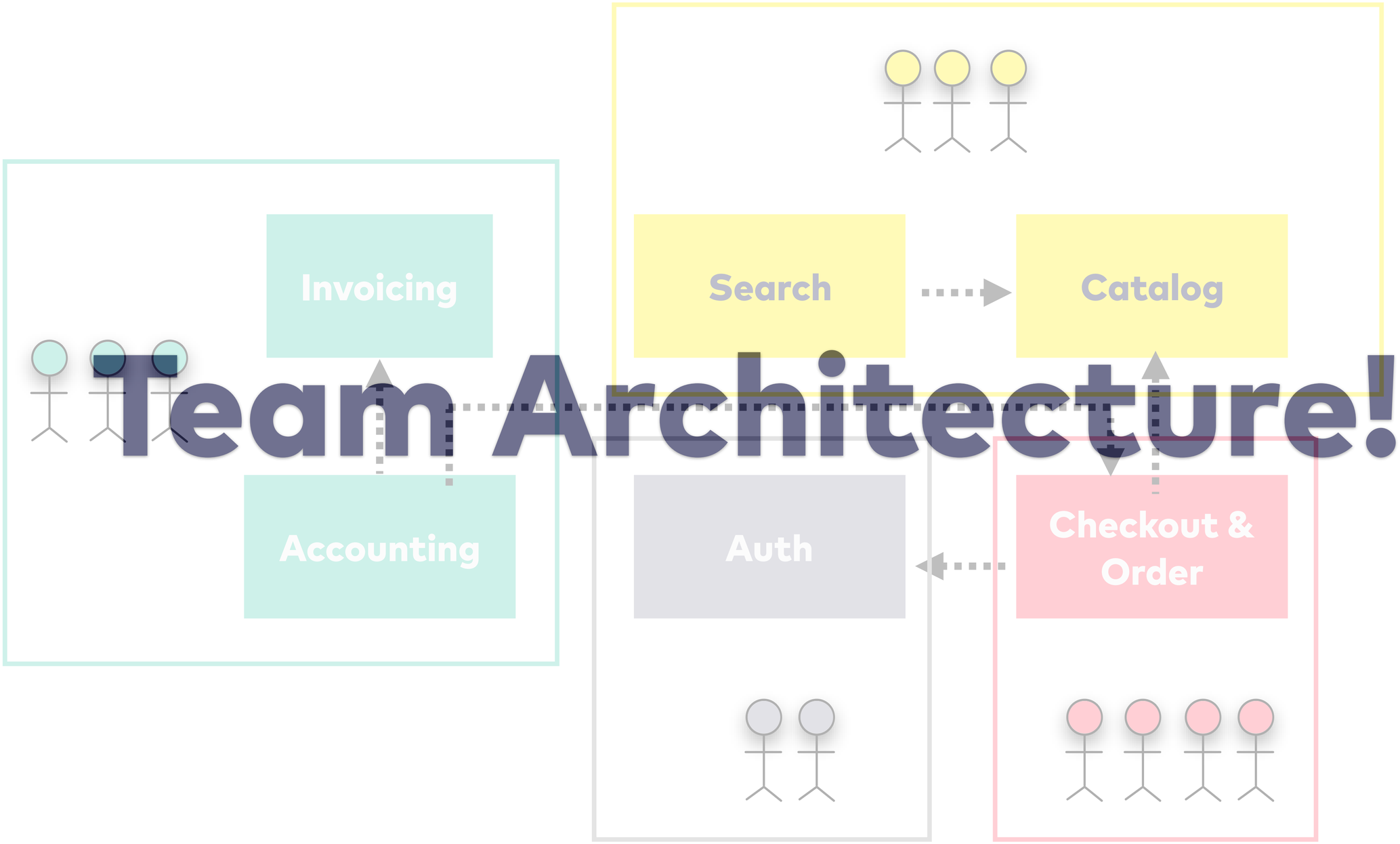
From a layered system ...



… to a system of systems




Invoicing
Accounting
Search
Catalog
Auth
Checkout &
Order



Team Architecture!

Benefits

1. Isolation
2. Autonomy
3. Scalability
4. Resilience
5. Speed
6. Experimentation
7. Rapid Feedback
8. Flexibility
9. Replaceability
10. Ecosystem

Centralization vs. Autonomy: Cases

Context:

- …

Observation(s):

- …

Lesson(s) learned:

- …

Context:

- E-Commerce/Online shop (Retail)
- 100-120 developers, ~10 teams

Observation(s):

- Lack of front-end expertise led to central UI/design team, bottleneck for development, deployment, operations, evolution

Lesson(s) learned:

- Local optimization needs can trigger centralization
- Full stack teams require full stack capabilities

A general lack of specific skills, combined with a select few who have it, will sabotage any attempt at decentralizing anything requiring it

Context:

- E-Commerce/Online shop (Retail)
- 100-120 developers, ~10 teams

Observation(s):

- Extremely inefficient UI integration runtime due to lack of standardization
- Vast differences in API style, formats, documentation

Lesson(s) learned:

- Complete lack of guidance creates unproductive diversity

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)

Context:

- Insurance customer portal
- 10-15 developers, 1 team

Observation(s):

- Potential for independent decisions in separated systems (almost) never exploited
- Engineering effort spent on coordination

Lesson(s) learned:

- Premature modularization can lead to increased effort without matching benefits

Context:

- E-Commerce/Online shop (Retail)
- 100-120 developers, ~10 teams

Observation(s):

- Common standard micro architecture at start of project
- Gradual increase in degrees of freedom
- Increase in actual diversity of tools, languages, architecture

Lesson(s) learned:

- Increased maturity allows for less dogma/fewer rules

Pattern: Regulated Market

Dreyfus model of skill acquisition

Stage / Quality	Novice	Advanced Beginner	Competence	Proficient	Expert
Recollection	Non-Situational	Situational	Situational	Situational	Situational
Recognition	Decomposed	Decomposed	Holistic	Holistic	Holistic
Decision	Analytical	Analytical	Analytical	Intuitive	Intuitive
Awareness	Monitoring	Monitoring	Monitoring	Monitoring	Absorbed

@stilkov

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

Takeaways

1.
There is no conflict between Agile and Architecture

2.
Size matters

3.
Architecture must match Organization

That's all I have.
Thanks for listening!

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

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

Problems Some People Have

Building features takes too long

Technical debt is well-known, yet not addressed

Deployment is way too complicated and slow

Architectural quality has degraded

Scalability has reached its limit

"-ility" problems abound

Replacement would be way too expensive