

How to Become A Great Software Architect

Eberhard Wolff

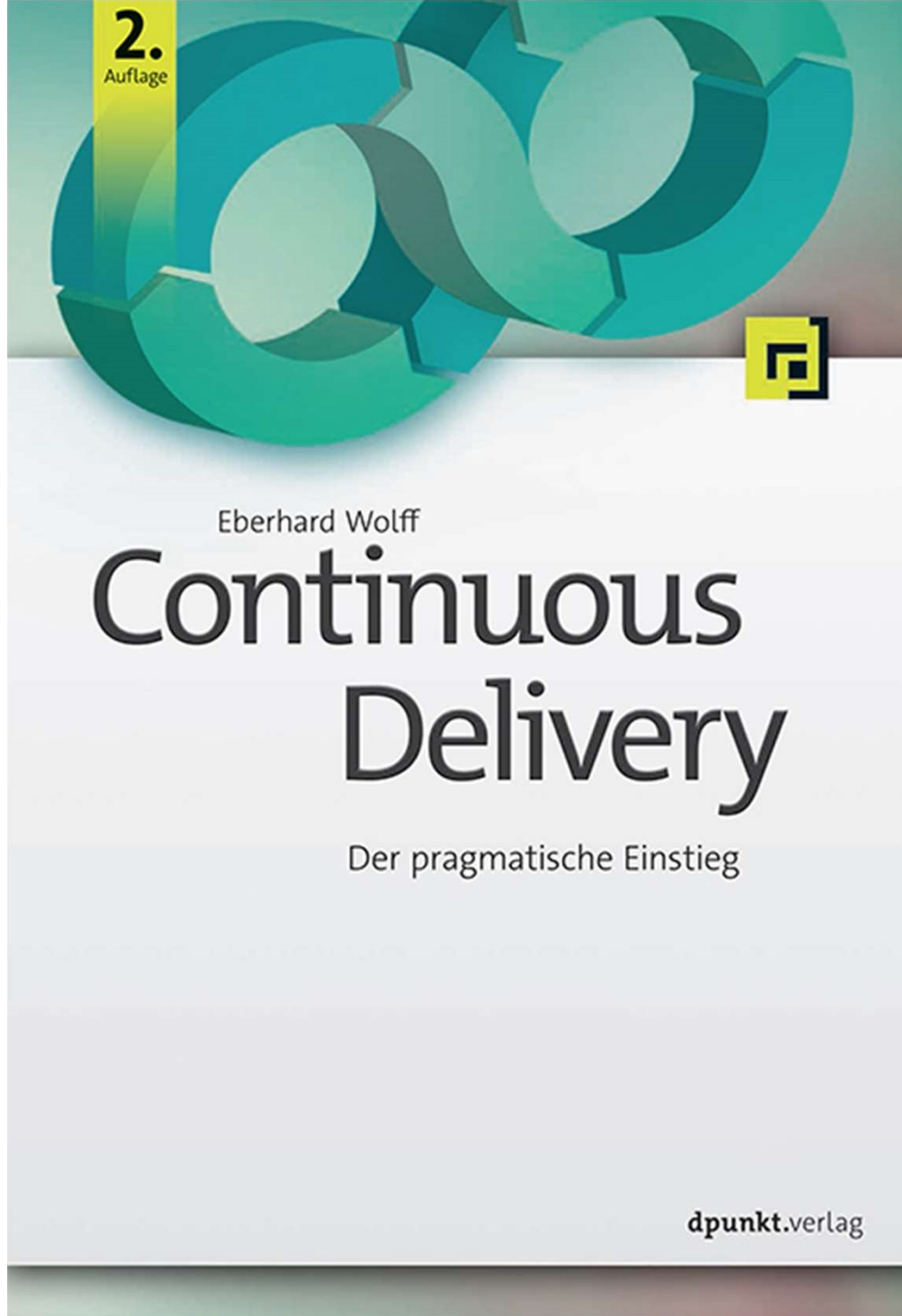
Fellow

INNOQ

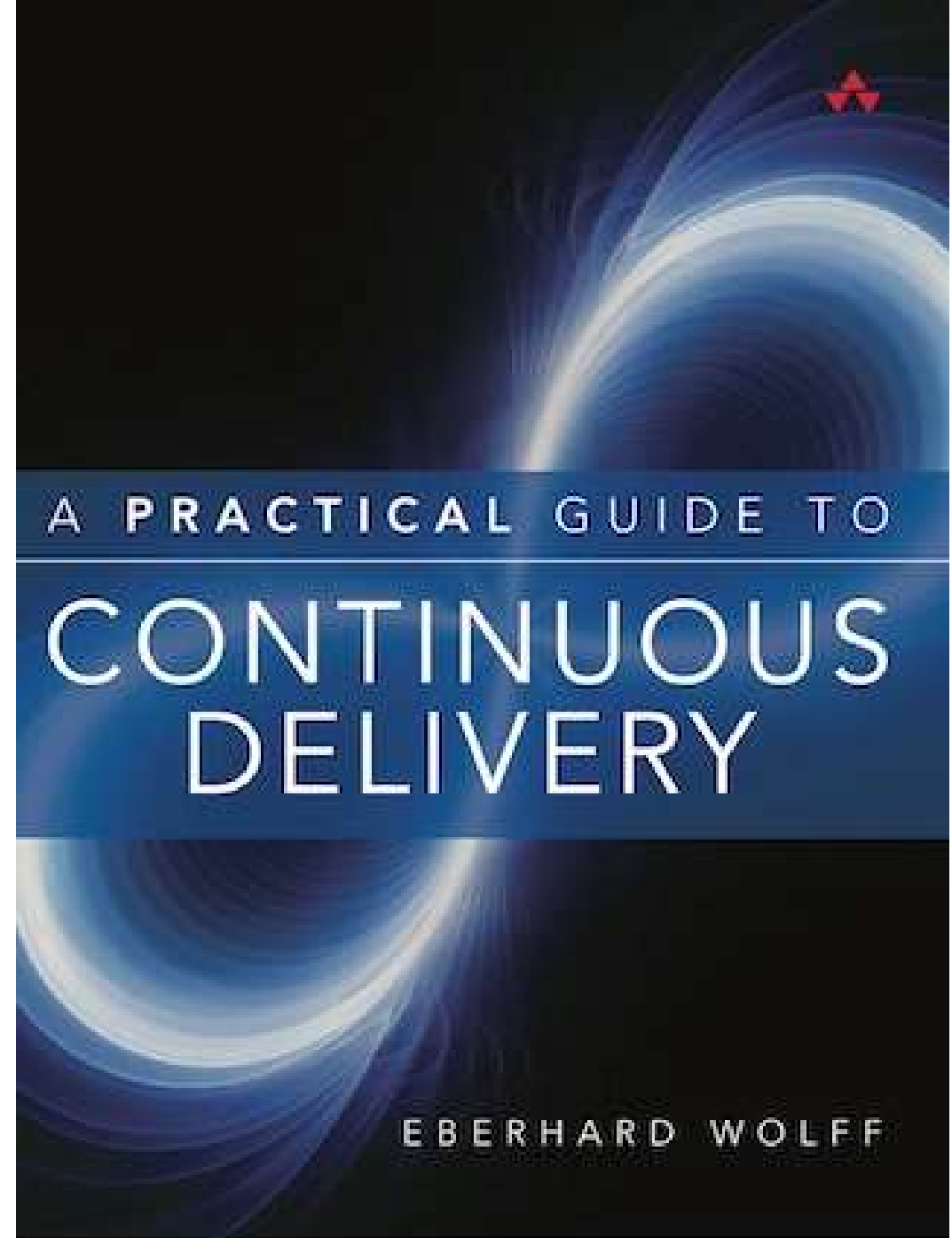
@ewolff

<http://ewolff.com>

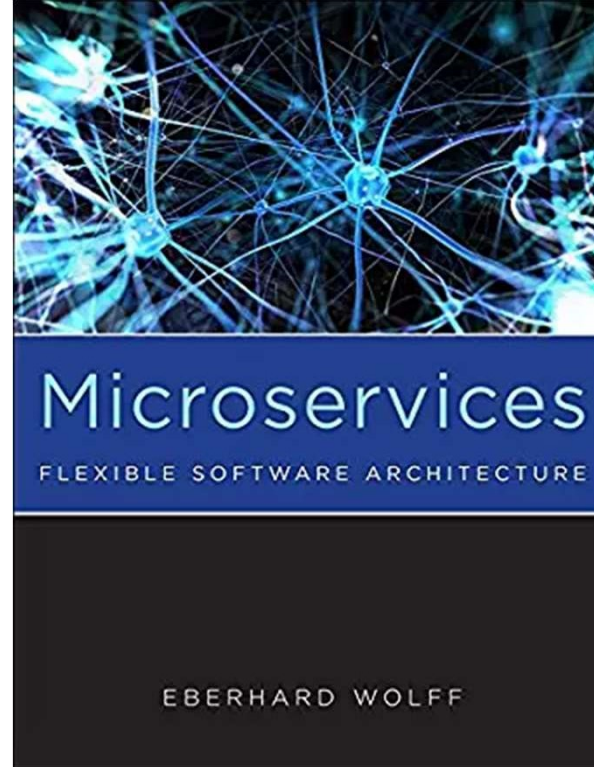




<http://continuous-delivery-buch.de/>



<http://continuous-delivery-book.com/>



<http://microservices-buch.de/>

<http://microservices-book.com/>



Eberhard Wolff

Microservices

Ein Überblick

INOQ



Eberhard Wolff

Microservices Primer

A Short Overview

INOQ

[http://microservices-buch.de/
ueberblick.html](http://microservices-buch.de/ueberblick.html)

[http://microservices-book.com/
primer.html](http://microservices-book.com/primer.html)

FREE!!!!



Das Eberhard Wolff Microservices- Praxisbuch

Grundlagen, Konzepte und Rezepte

dpunkt.verlag

Microservices



A Practical Guide
2nd Edition

Principles, Concepts, and
Recipes

Eberhard Wolff

<http://microservices-praxisbuch.de/>

<http://practical-microservices.com/>



Eberhard Wolff

Microservices Rezepte

Technologien im Überblick

INNOQ



Eberhard Wolff

Microservices Recipes

Technology Overview

INNOQ

[http://microservices-praxisbuch.de/
rezepte.html](http://microservices-praxisbuch.de/rezepte.html)

[http://practical-microservices.com/
recipes.html](http://practical-microservices.com/recipes.html)

FREE!!!!



FREE!!!!

<http://ddd-referenz.de/>
<https://domainlanguage.com/ddd/reference/>



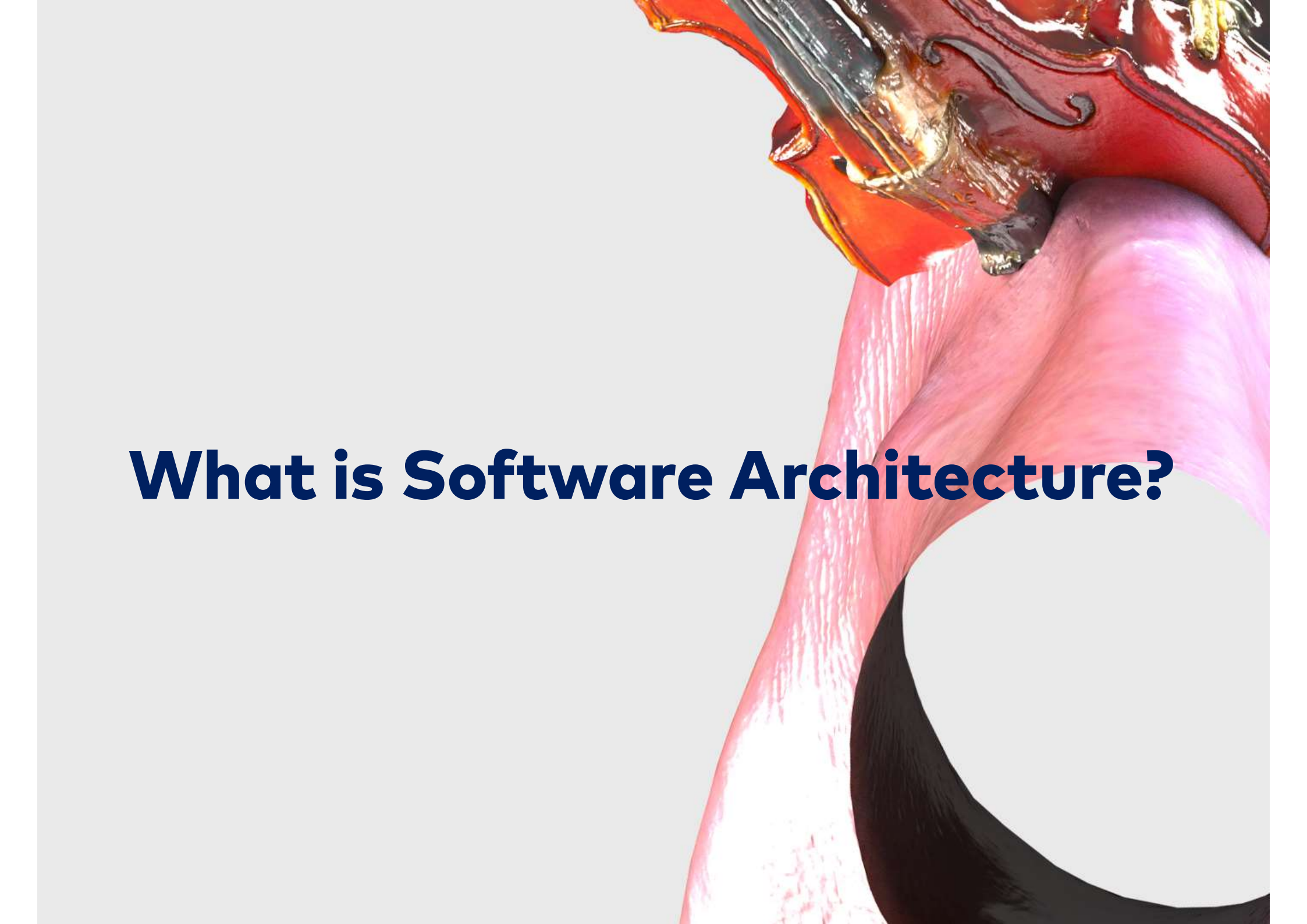
FREE!!!!

<http://leanpub.com/service-mesh-primer/>



**International Software Architecture
Qualification Board**

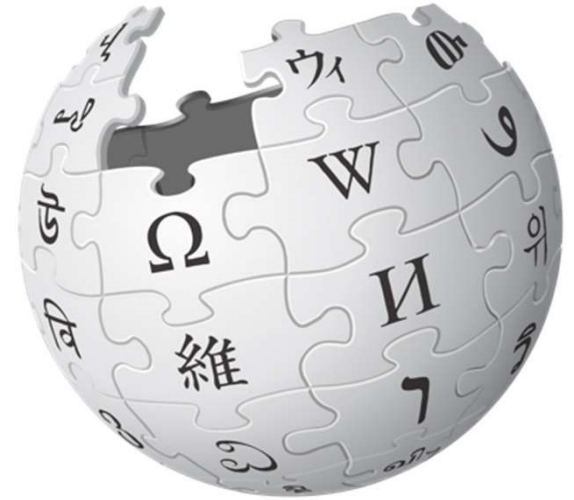
<https://www.isaqb.org/>



What is Software Architecture?

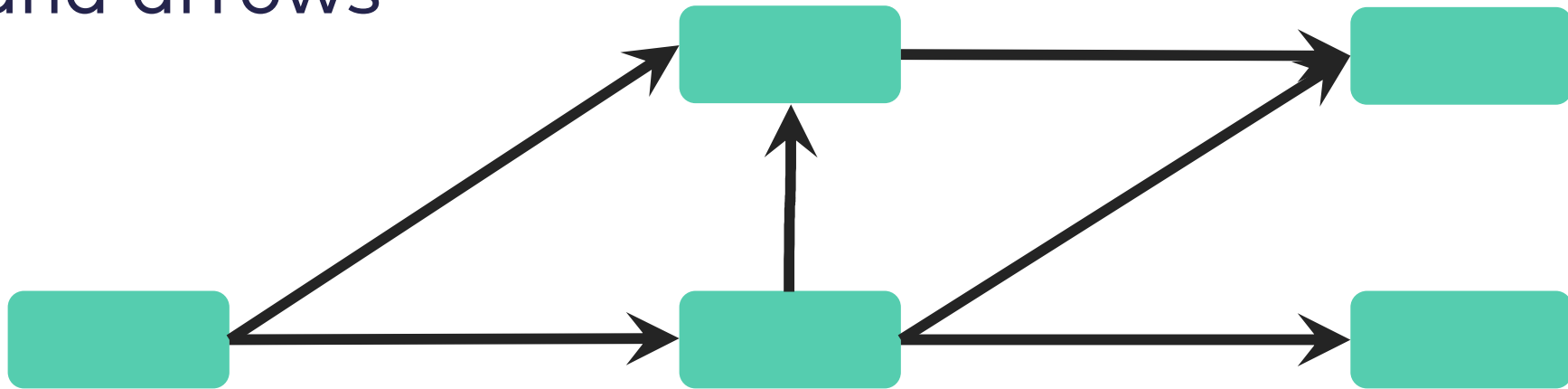
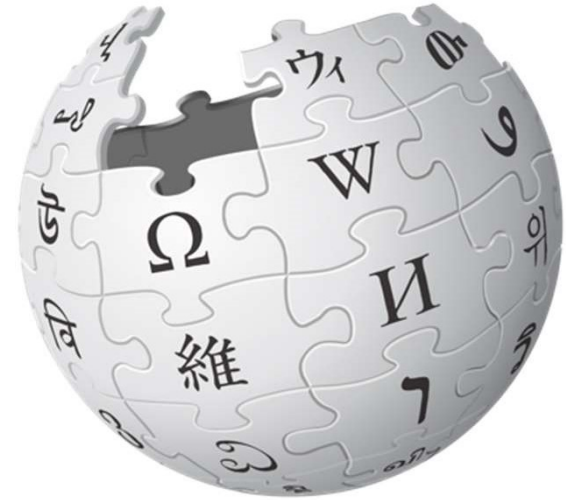
Wikipedia

- Software architecture =
high level structures
- Structure =
Software elements +
Relations among them +
Properties of elements and relations



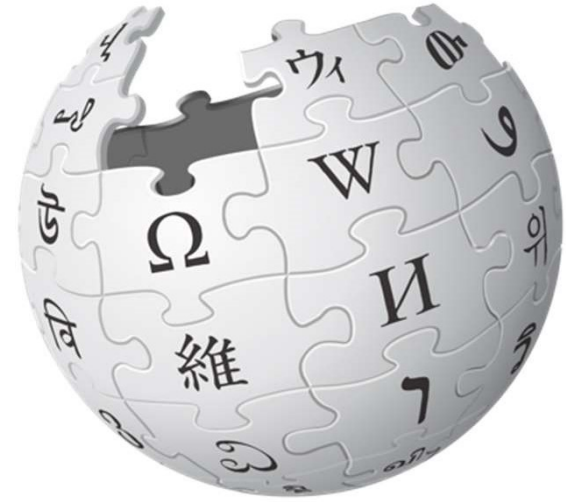
Wikipedia

- Software architecture = structures
 - Boxes
- ... and arrows



Wikipedia

- Common definition of architecture
...but does it catch all?



Architecture Fail?

- Software doesn't go into production

Security problem

Compliance problem

- Fail caused by structure?
- Successful architecture?

Martin Fowler

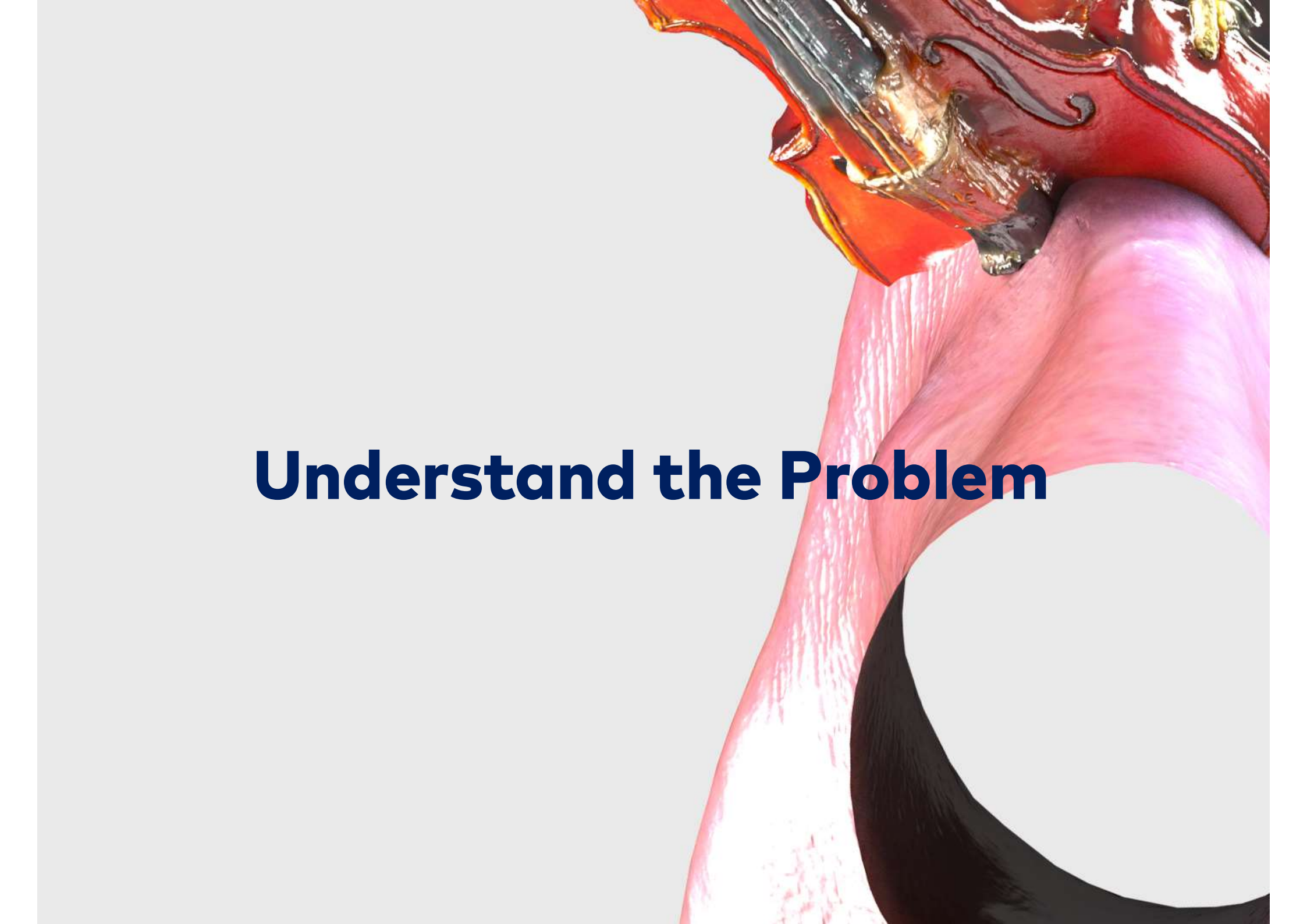
- Software architecture = Important and hard to change decisions
- How to know in advance?



Photo: Webysther Nunes

Software Architecture

- Find technical solutions
...to the problem at hand.
- Home-grown definition
- Broad definition
- Need to understand the problem!

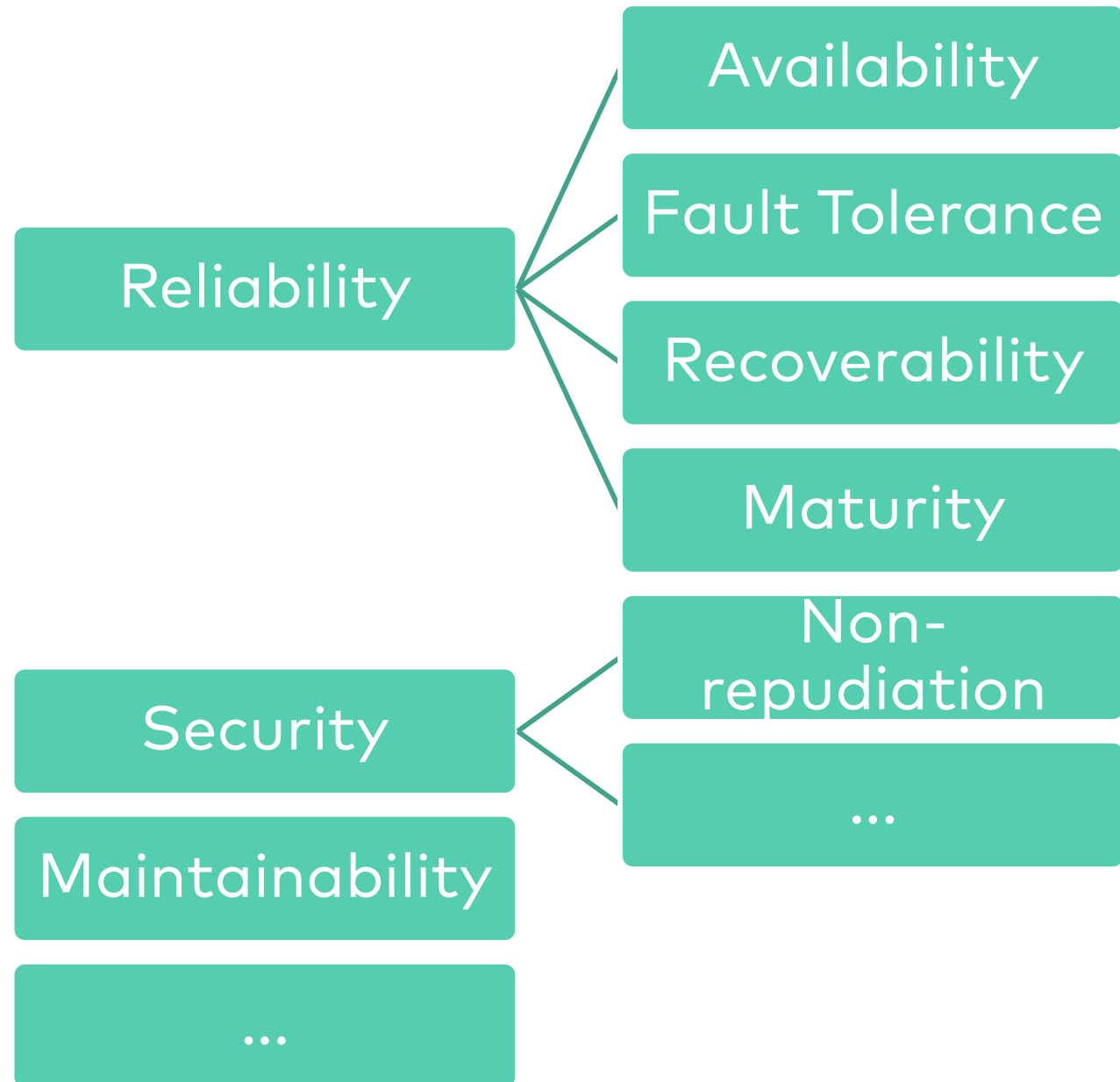


Understand the Problem

Quality Attributes / Tree

ISO 25010

Structure only
helps with
maintainability



Quality Attributes

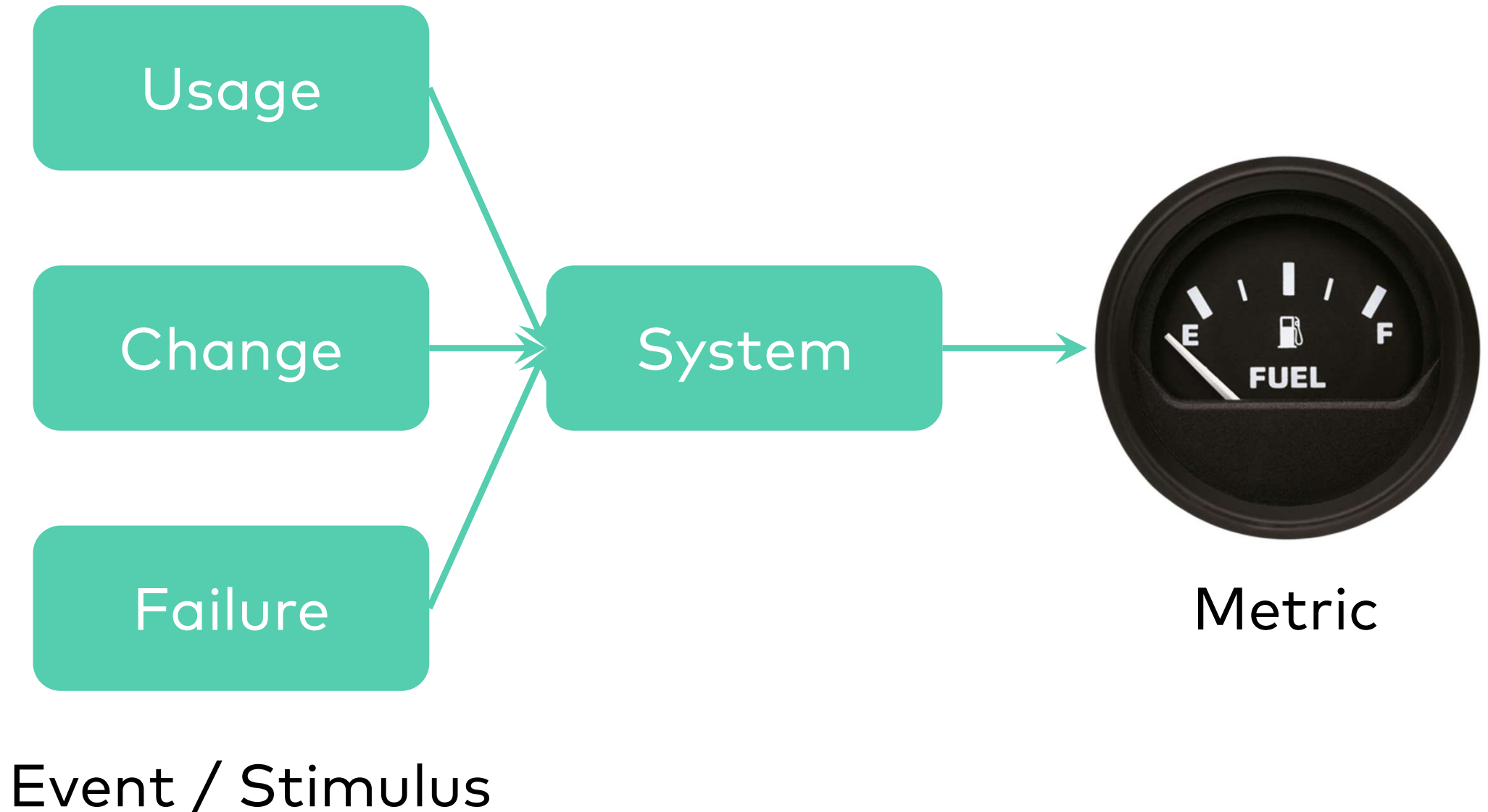
- Holistic view on quality
- Identify important attributes

- But

high-level

hard to verify

Quality Scenario



Quality Scenario

- Concrete
- Easy to verify - metric



Usage Scenario

- Stimulus

A new users registers

- Metric

Only one in 1.000 users calls the hotline.

Usability – Ease of Use

Usage: Solution

- Hire UX experts
- Usability tests
- No "classic" architecture work



Change Scenario

- Stimulus

A new language / locale should be support

- Metric

No code modification needed.

Takes two days

Maintainability - Modifiability

Change Solution

- Configuration files for language
- Code quality irrelevant



Failure Scenario

- Stimulus

A server crashes

- Metric

System might be unavailable for two hours.

No data might be lost.

Reliability – fault tolerance

Failure Solution

- RAID
- Backup
- Data center in different locations
- No need for a cluster of servers



Solutions

- Solutions must solve problems.
- Traditional measures like
high code quality,
clean architecture,
scalability,
your favorite framework or language
...solves none of the scenarios

**Traditional Architect
vs.
Modern Architect**

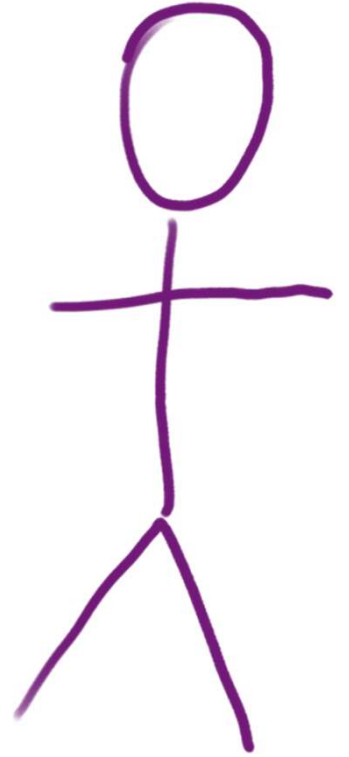
Traditional Architect

- A title



Modern Architect

- Architect does architecture
- A role
- Might be different persons
- Might change over time
- Might or might not have title "software architect"



Traditional Architect

- Most experienced technical person



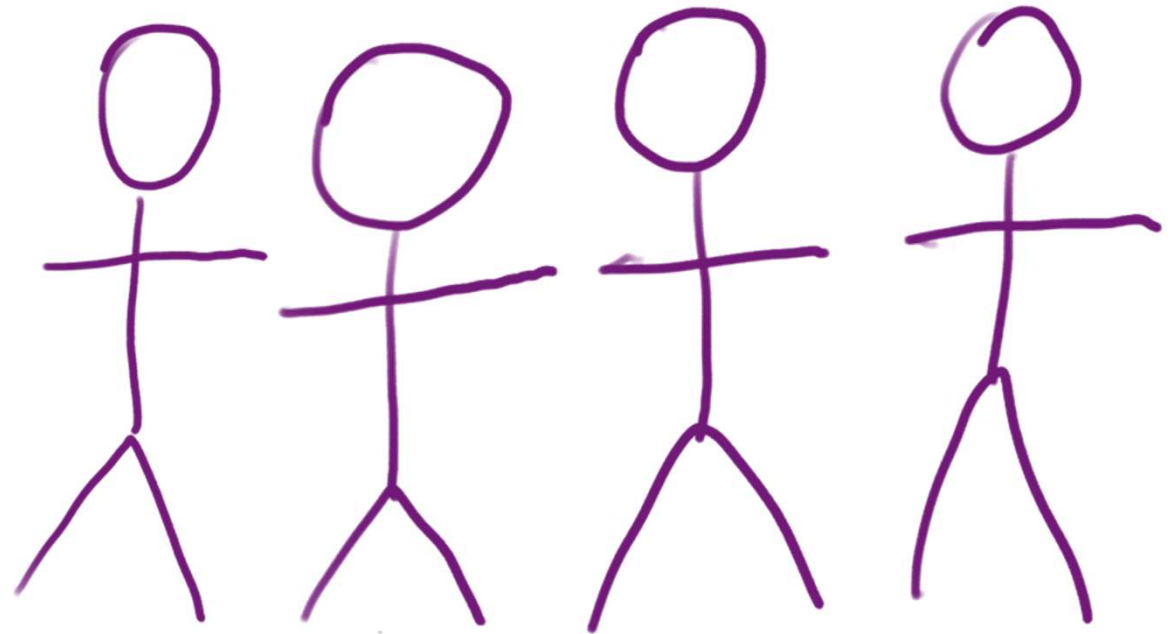
Most Experienced Person

- For every detail?
- For every technology?
- Unrealistic
- Making all important decisions
- Puts enormous pressure on the architect.
- Why do you want to play such role?



Most Experienced Person

- Quite sad
- Use the technical expertise of the team
- ...and grow the team



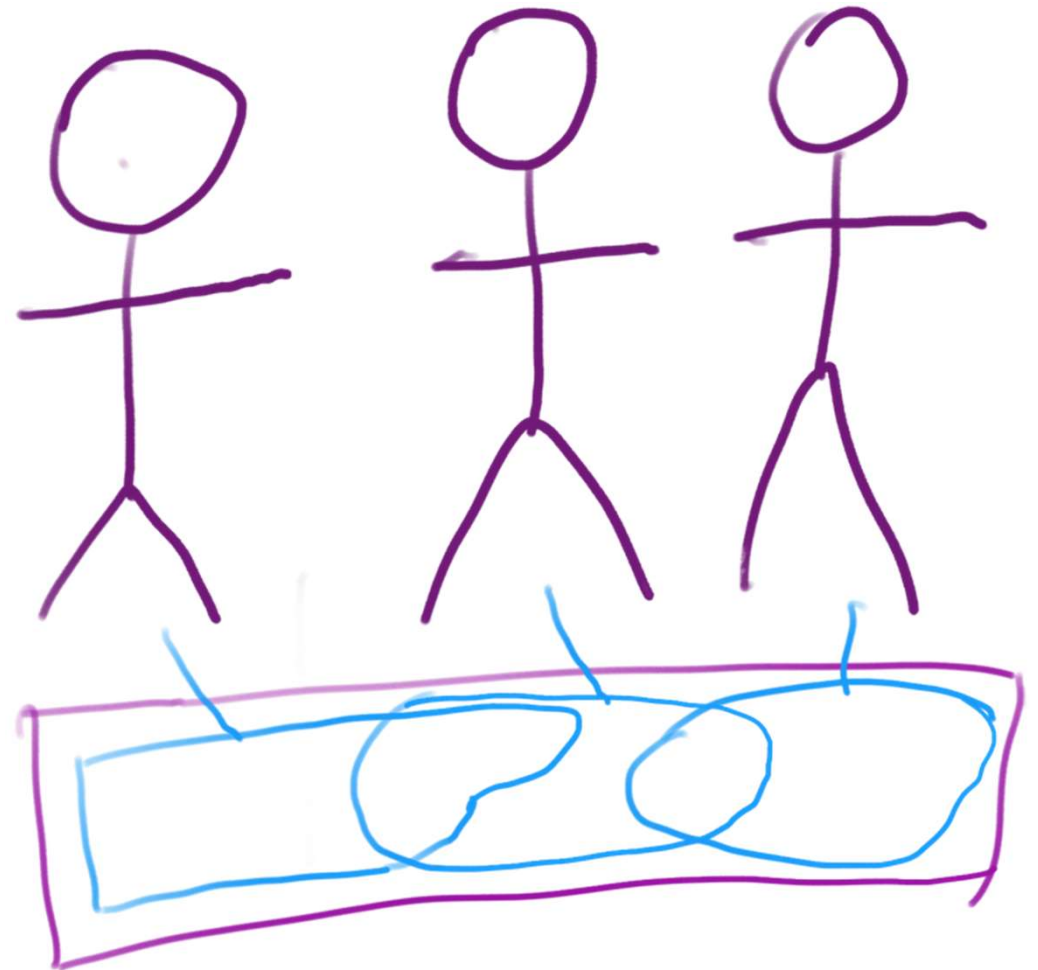
Traditional Architect

- Understand & guard whole system



Understanding the System

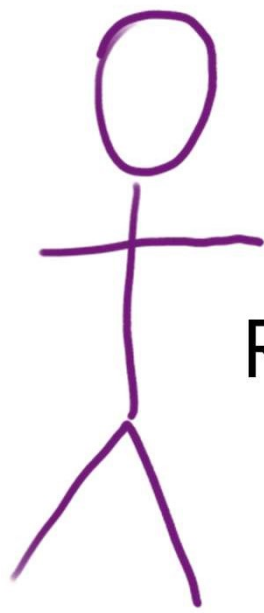
- Architects need to understand the system.
- Systems too large for one person.
- So use the expertise of the team.



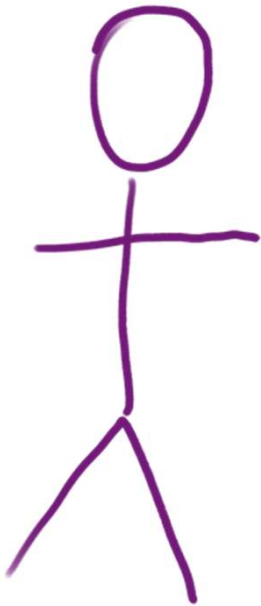
Traditional Architect

- Define the architecture
- i.e. make all relevant decisions
- Enforce the architecture

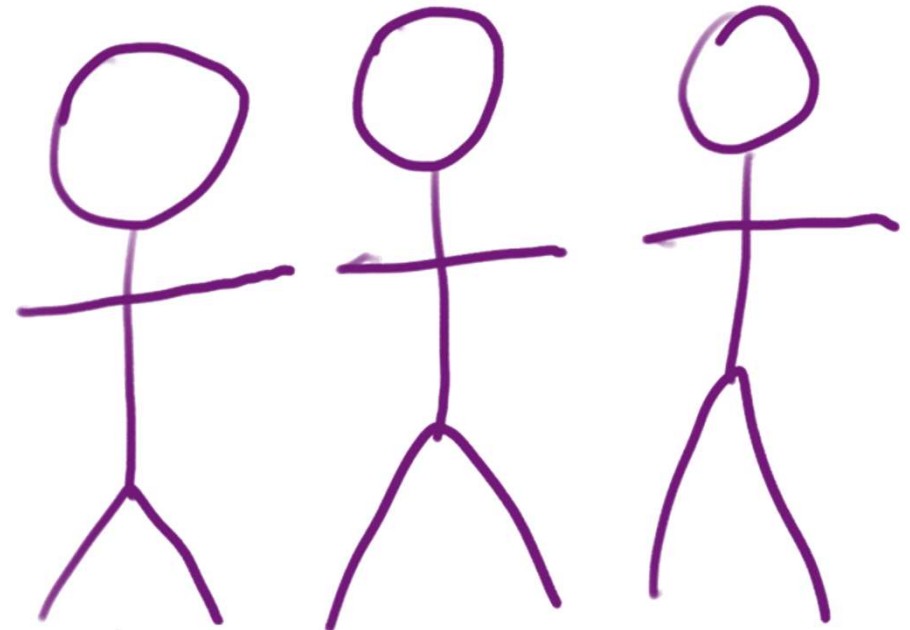
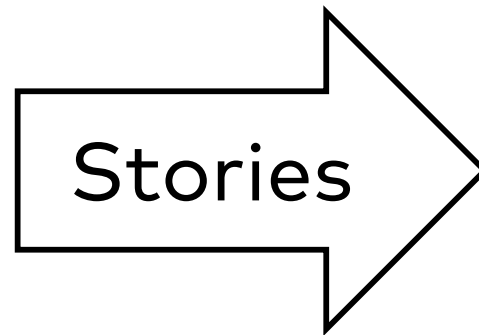




Scrum Master
Removes obstacles
Enforces rules



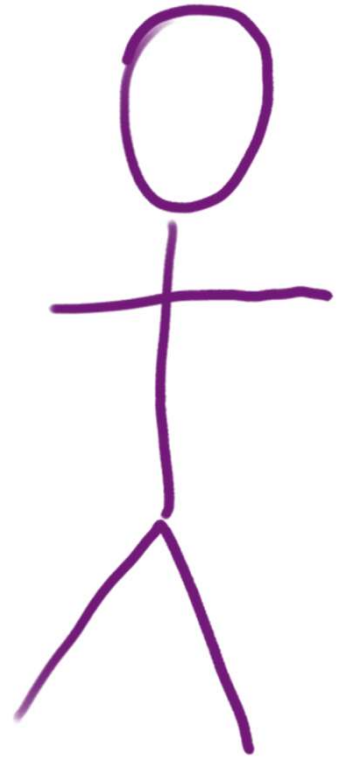
Product
Owner



Team
Self-organizing
Implements stories

Self-organizing Team

- Architect is "just" a team member
 - No way to "enforce" architecture
 - Actually, the team decides how to work.
-
- How can the architect define the architecture?



Traditional Architect

- Enforces the architecture

...but probably doesn't know whether it is truly implemented.

...because he / she can't understand the whole system.



- How do you architect if you can't understand the system?
- What is being "enforced"????

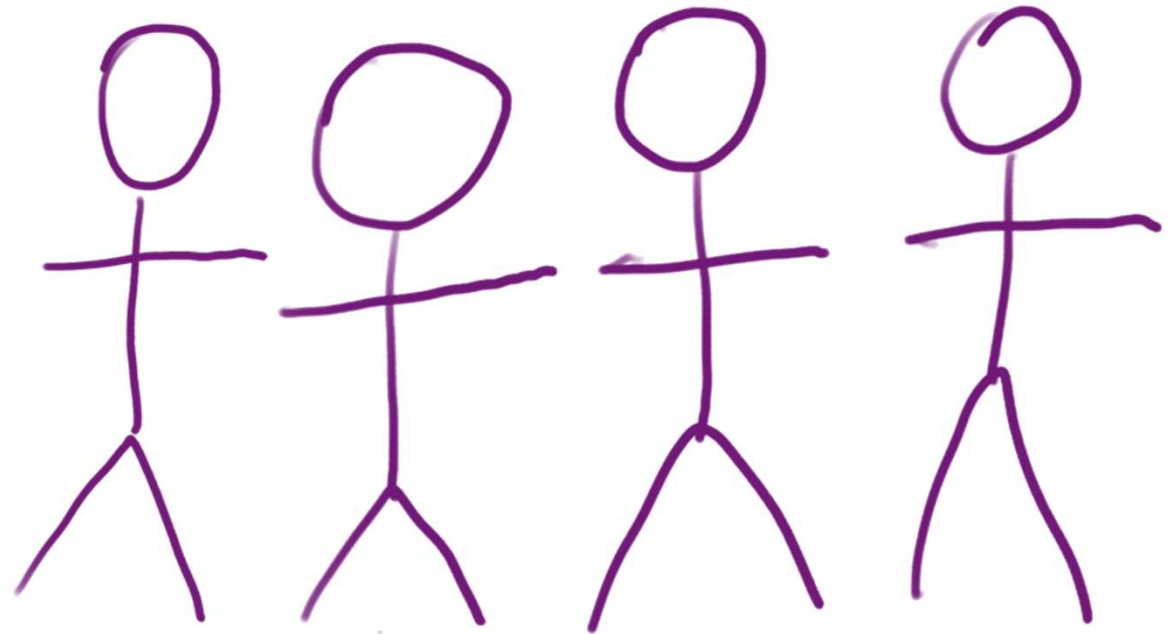
Ivory Tower

- Just because architects think they do architecture doesn't mean it has any impact.
- Information about issues might be wrong.
- Enforcement might be an illusion.



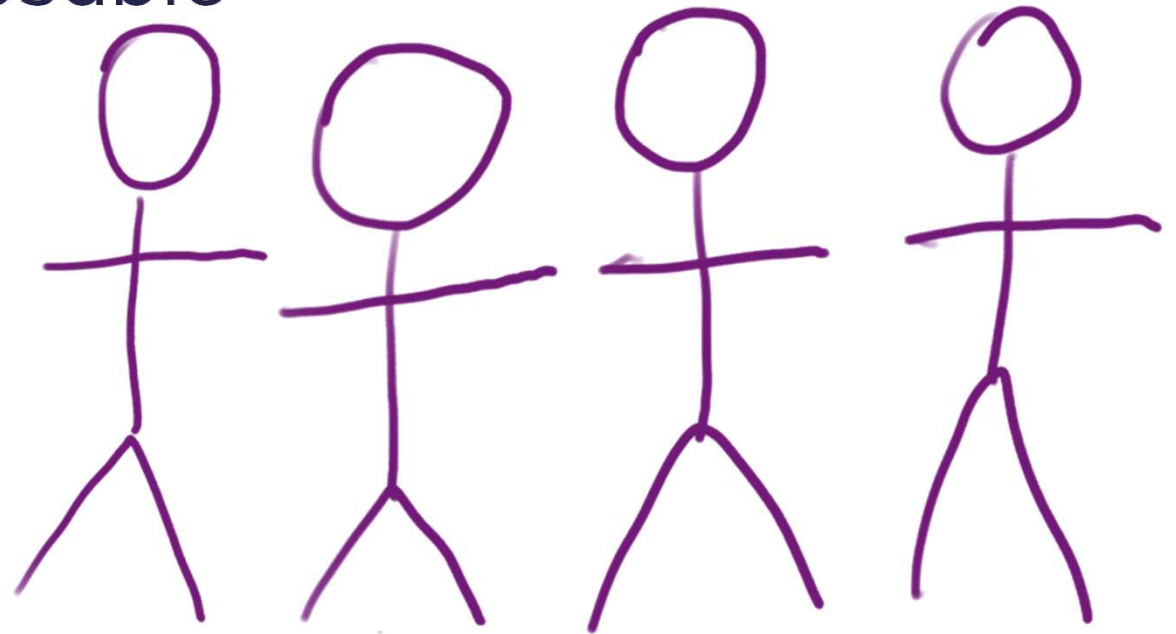
Technical Decisions

- Understand the team's problems
- Seek feedback about possible decision
- Communicate decisions



Technical Decisions

- Moderate, don't enforce
- Make your expertise usable
- General rule for managing self-organization



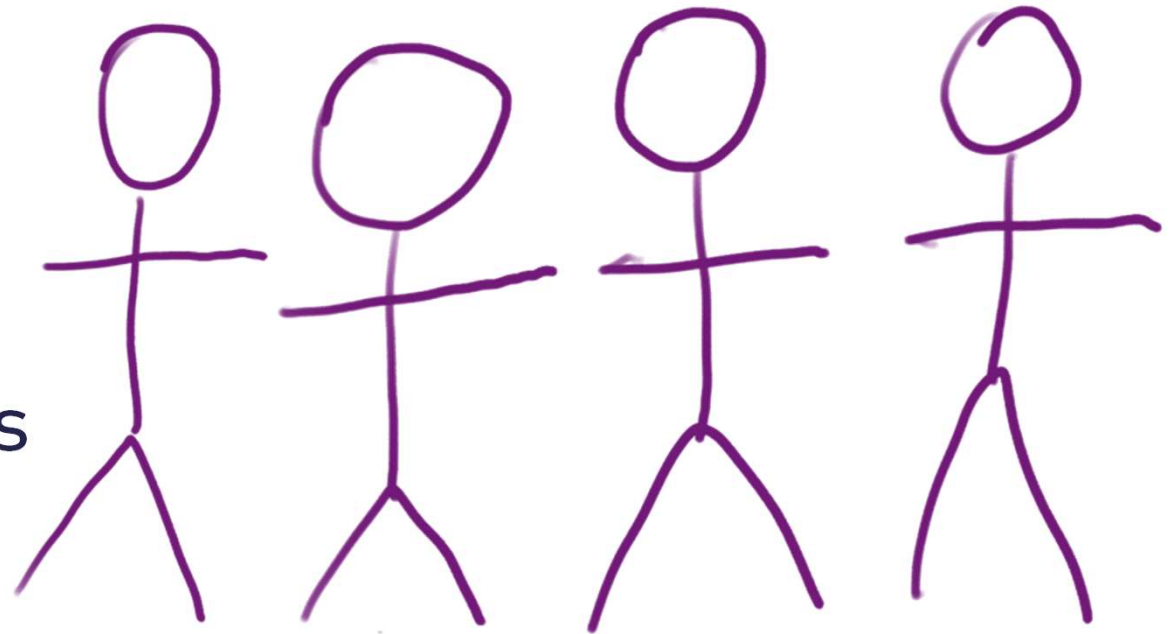
Moderate Technical Decisions

- Use expertise of the full team

- Results:

Better decisions

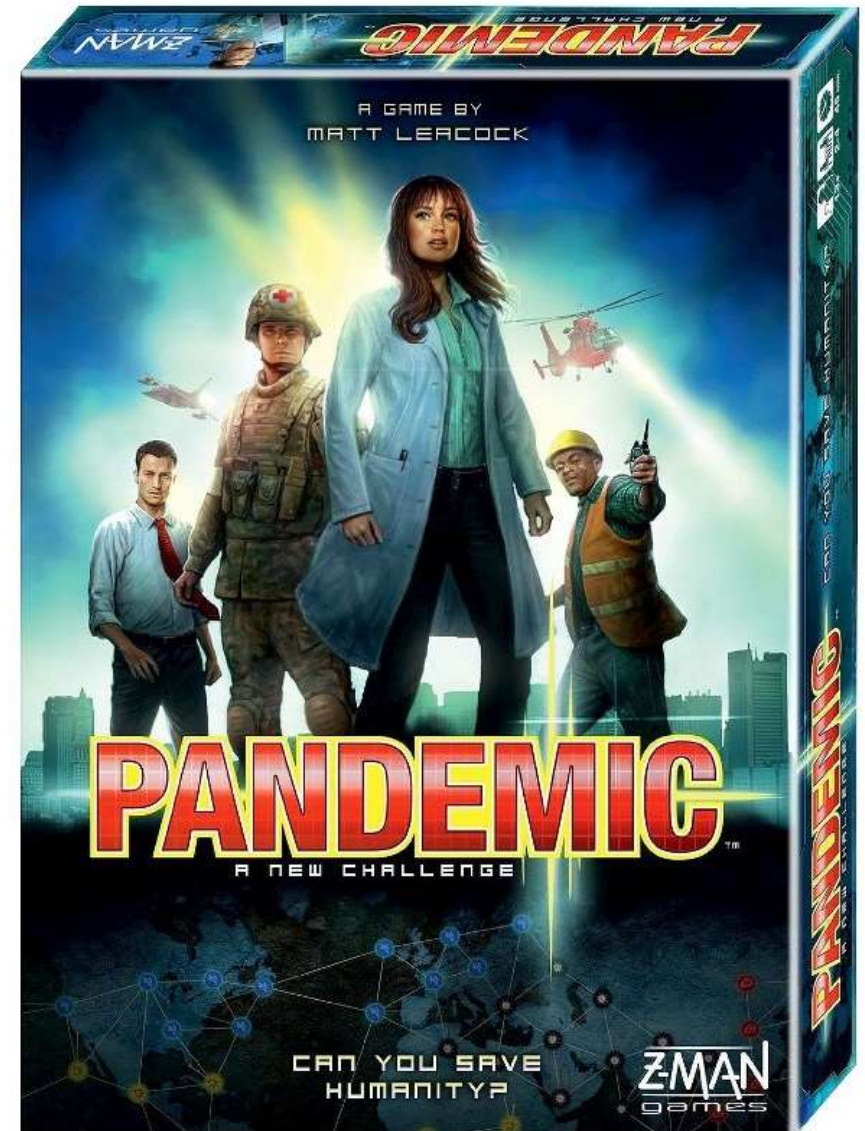
Team actually executes
decisions



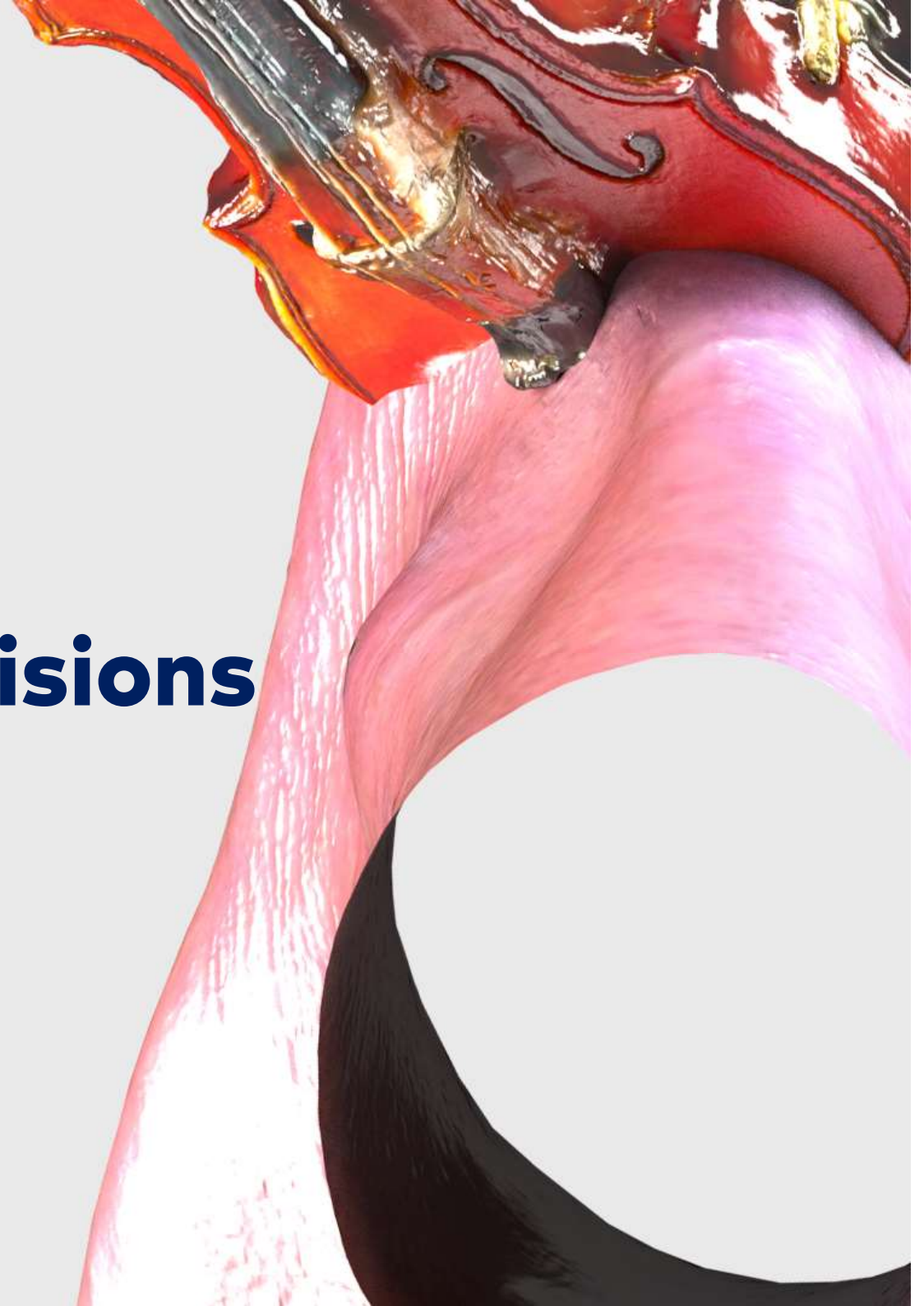
- No ivory tower

Software Architecture = Collaborative Game

- All lose or win together
- Everyone has a specific role
- Communication is essential



Decisions



Understanding the System

- Information about the system is incomplete
...so are requirements
- More information might be hard to get.
...or too late
...or too expensive
- You will still need to make decisions.
- Be courageous.



Software architecture
= making decisions
...constantly
...without enough
information

Decisions

- Often decisions are made too early
- Work on unimportant stuff
- Too little information

Decisions

- When do you need to make the decision?
- Example: Issue invoices as soon as there is revenue
...and also payment, book keeping etc
- No need for a detailed architecture, yet.

Decisions

- What if you do nothing?
- What if you make the decision too late?

- No revenue

...only bad if there are customers

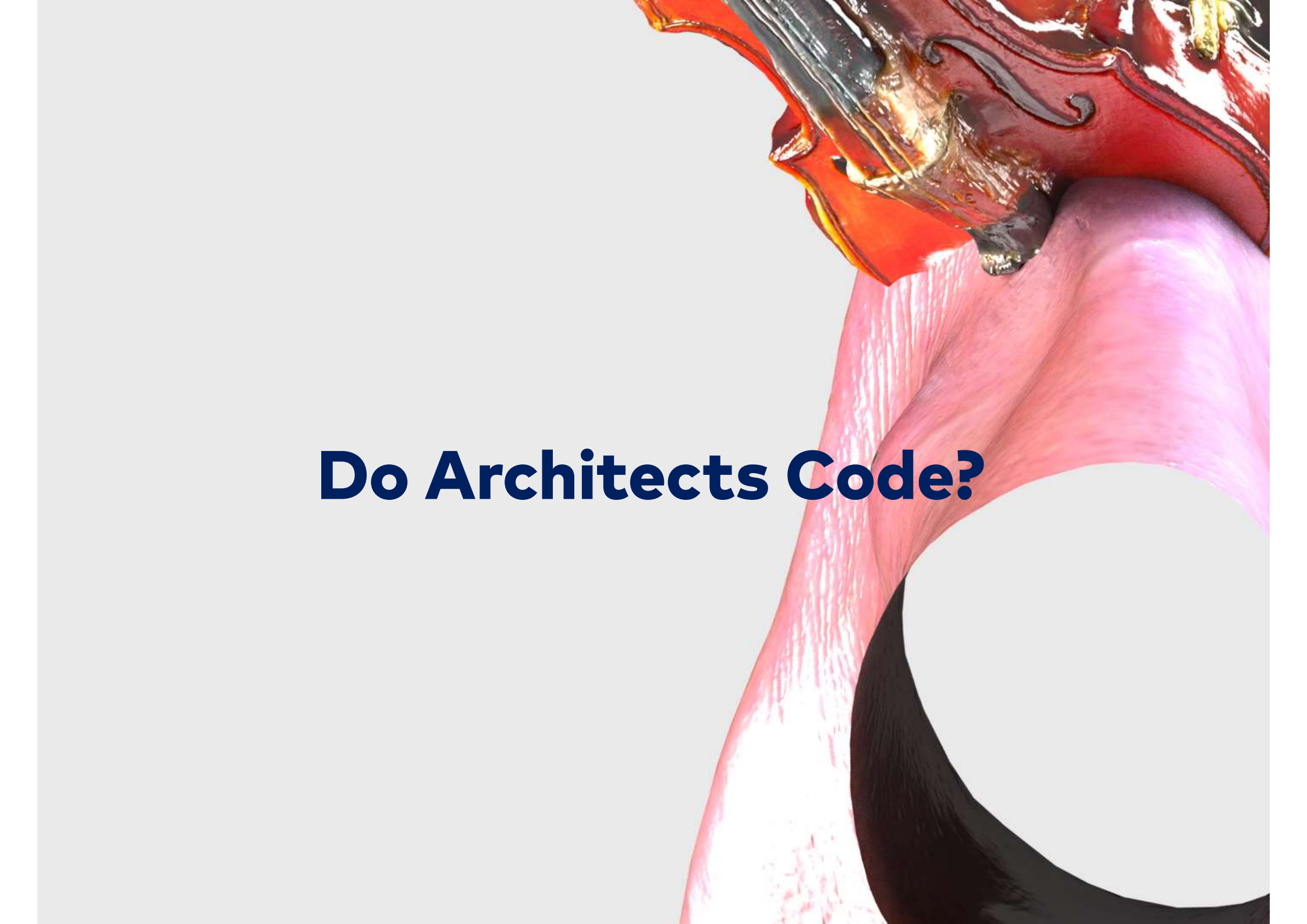
- Manual invoicing

...might be fine

...unless you are truly successful

The best decisions
are the ones not made
yet.

Be prepared
to revise decisions



Do Architects Code?

Traditional Architect

- Doesn't code

...because architects do architecture

- Problem: Ivory tower

...but that is solved

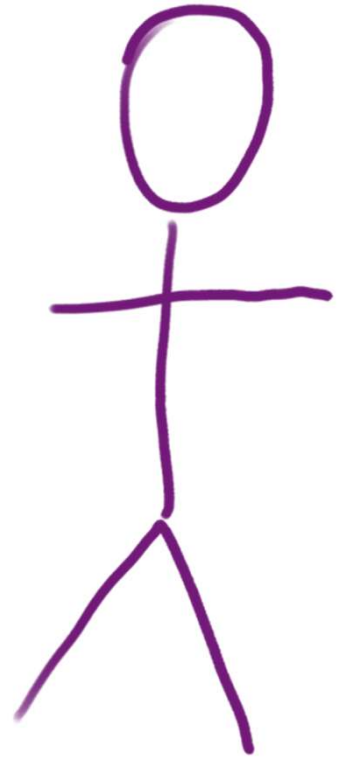
- Problem: Detached from reality

...can be solved by communication



Modern Architect

- What if architecture is a full-time job?
- No time to code
- Architect is a role.
- Anyhow collaborative
- Either have several coding architects
...and communication overhead between them.
- Or a full time architect



Soccer Coach

- Knows how to play soccer

...but doesn't do it any more.

- In particular won't try to score a goal.

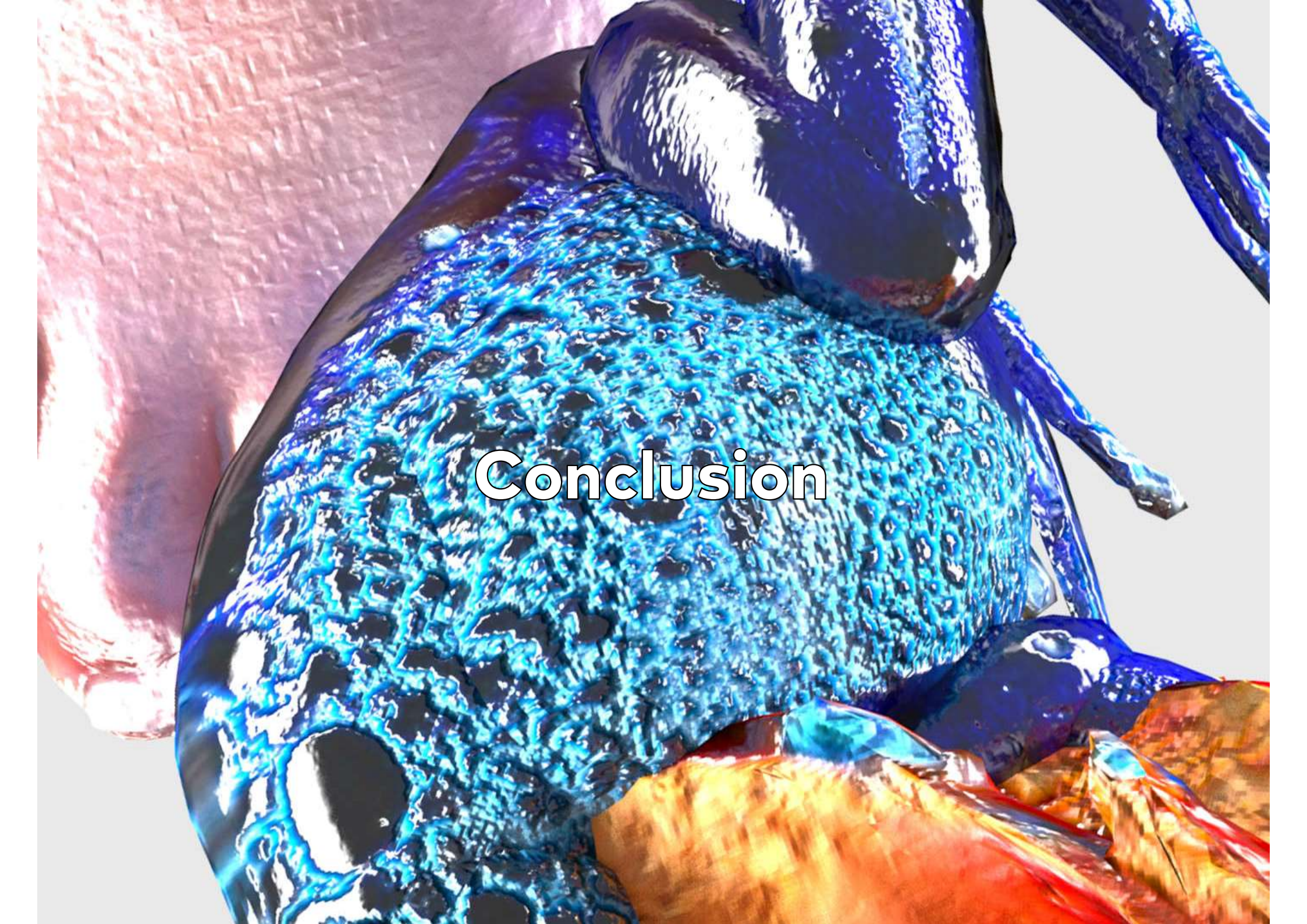
- A role model for software architects?

- Knows how to code

...but doesn't do it.

- In particular not for complex code.

Architects May or May Not
Code.



Conclusion

Conclusion

- Software architecture = solve technical problems
- Quality attributes / tree / scenarios to understand problem
- Often solution is not traditional architecture
- Software architecture is a collaborative game
- Don't be afraid to make decision later
- Don't be afraid to make decisions with unclear information

Send email to
jugch2019@ewolff.com

Slides

- + Microservices Primer DE / EN
- + Microservices Recipes DE / EN
- + DDD Reference DE
- + Service Mesh Primer EN
- + Sample Microservices Book DE / EN
- + Sample Practical Microservices DE/EN
- + Sample of Continuous Delivery Book DE

Powered by Amazon Lambda
& Microservices

Email address logged for 14 days,
wrongly addressed emails handled manually

