Conway's Law Extended

Peter Gfader

Beyond Agility

Abstract

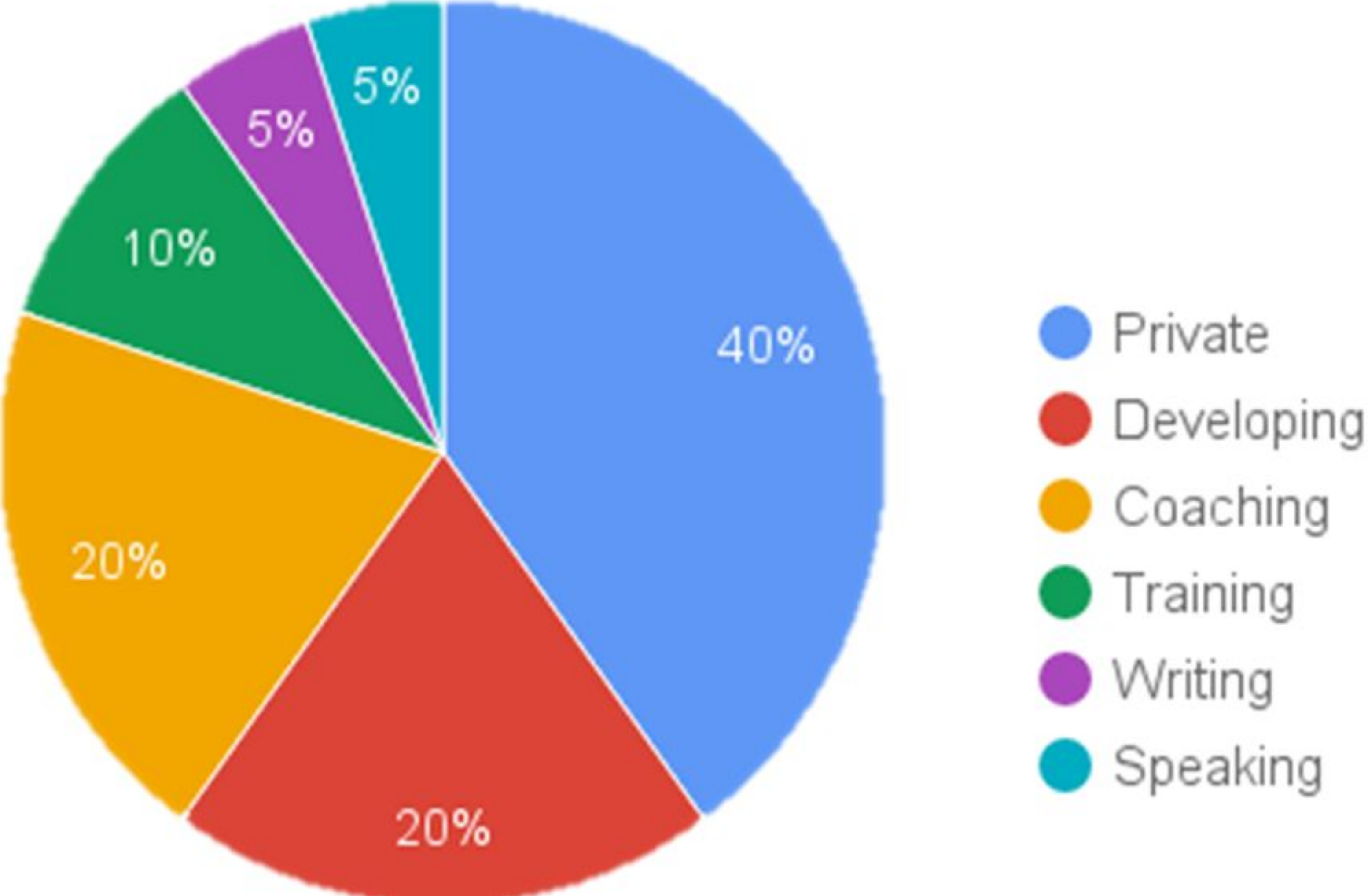
Jeder kennt wahrscheinlich Conway’s Law und schlaue Software- und Organisatons-Architekten nutzen es zu ihrem Vorteil.

Es gibt jedoch einen 3. Baustein, der in Conway’s Law nicht vorkommt, den ich aber als sehr relevant sehe für Software Teams. Ich habe diesen 3. Baustein in verschiedensten Kundenbegegnungen mit skalierten Teams verwendet und jetzt in ein Modell gegossen: "Conway's Law Extended".

Es gibt Wechselwirkungen zwischen diesen 3 Elementen im "Conway’s Law Extended", das wirkt un-intuitiv und macht es spannend!

Ich erkläre das Modell, die herrschenden Effekte, was man tun kann, um einen positiven Effekt in Zusammenarbeit zu erzeugen, und was das mit Entscheidungen macht.

Ich freue mich auf einen Erfahrungsaustausch zum Modell am Event.



lke
ng ideas

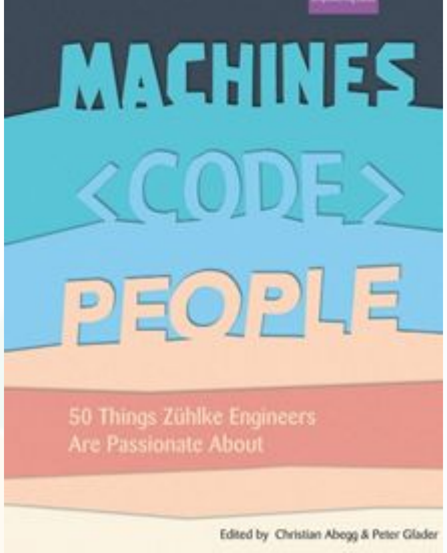
csolutions gmbh

ec.ch

XUS

Software Crafters Meetup

https://www.meetup.com/Software-Craftsmanship-Zurich/



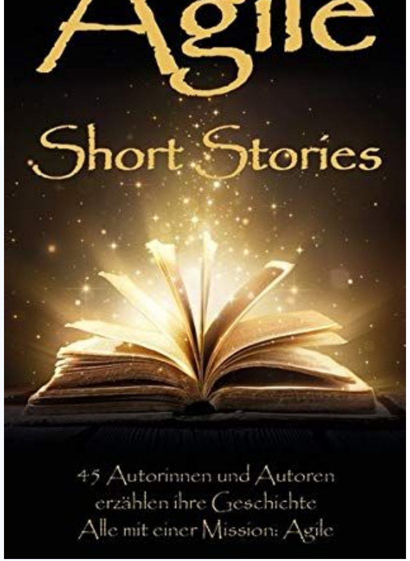
MACHINES

<CODE>

PEOPLE

50 Things Zühlke Engineers Are Passionate About

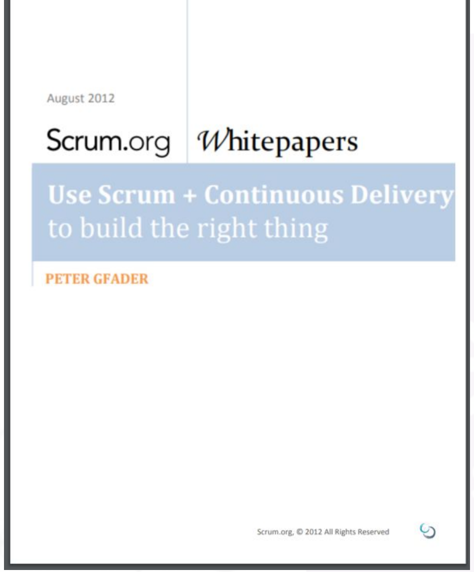
Edited by Christian Abegg & Peter Gfader



Agile

Short Stories

45 Autorinnen und Autoren
erzählen ihre Geschichte
Alle mit einer Mission: Agile



August 2012

Scrum.org Whitepapers

Use Scrum + Continuous Delivery
to build the right thing

PETER GFADER

Scrum.org, © 2012 All Rights Reserved

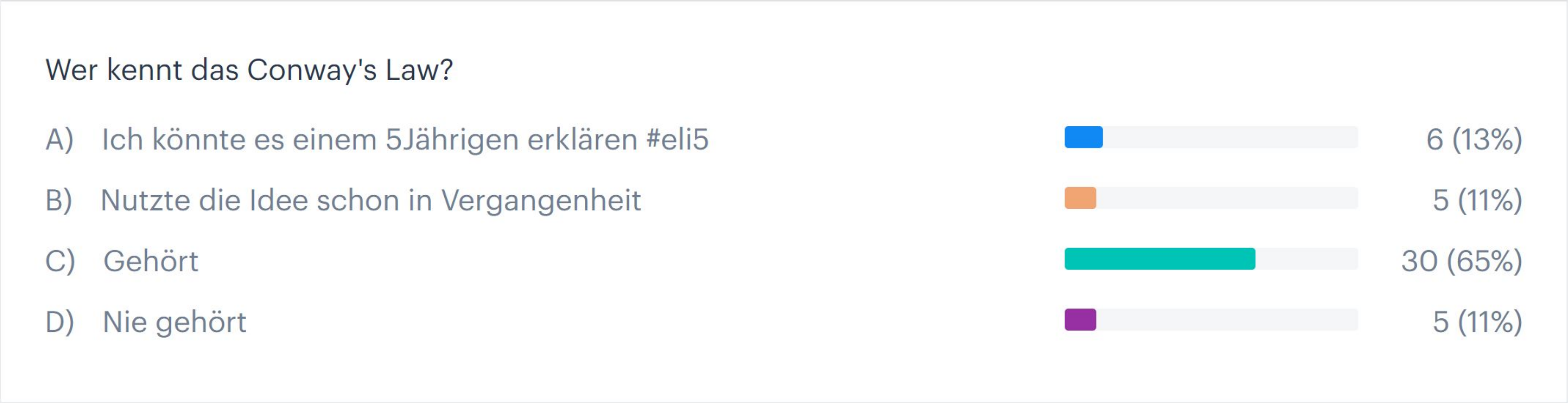
https://beyond-agility.com/books

📚 Reading? → https://beyond-agility.com/blog

🎞️ Watching? → https://beyond-agility.com/videos

Conway's Law

Conway's Law



Wer kennt das Conway's Law?

A)	Ich könnte es einem 5Jährigen erklären #eli5	6 (13%)
B)	Nutzte die Idee schon in Vergangenheit	5 (11%)
C)	Gehört	30 (65%)
D)	Nie gehört	5 (11%)

Conway's Law

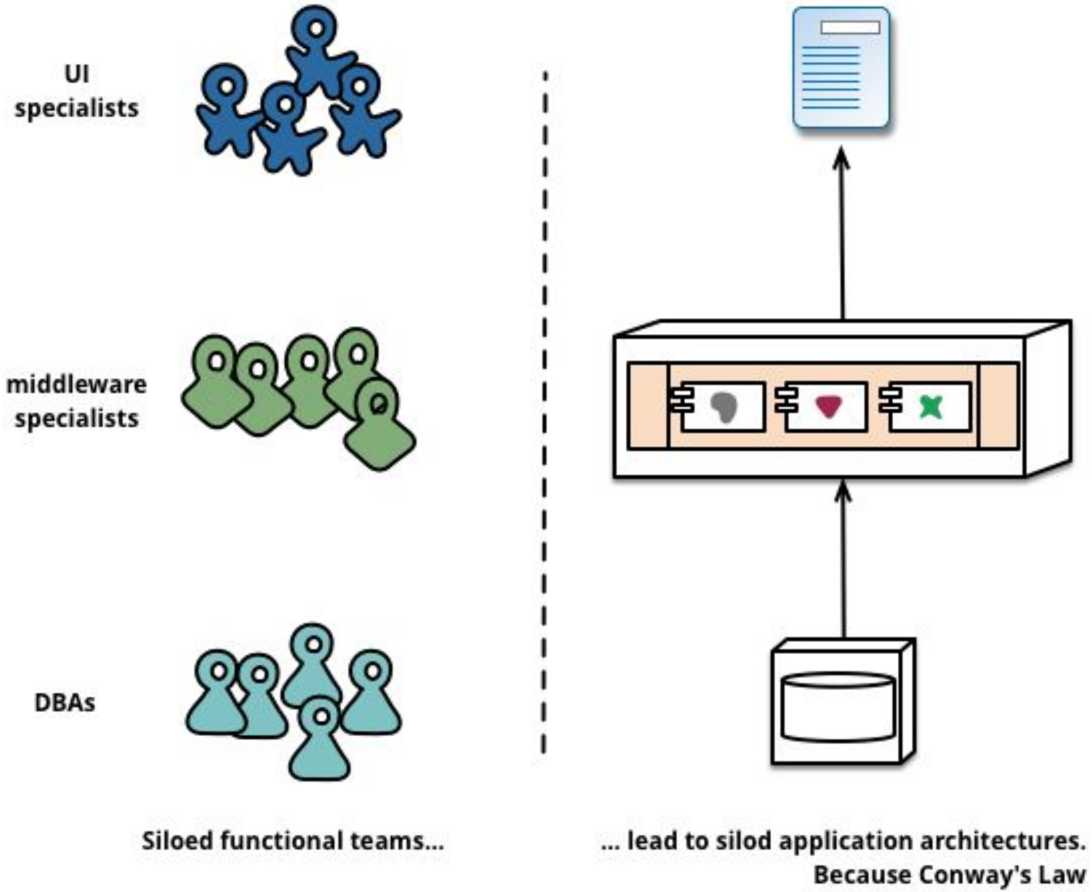
Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.

-- Melvyn Conway, 1967

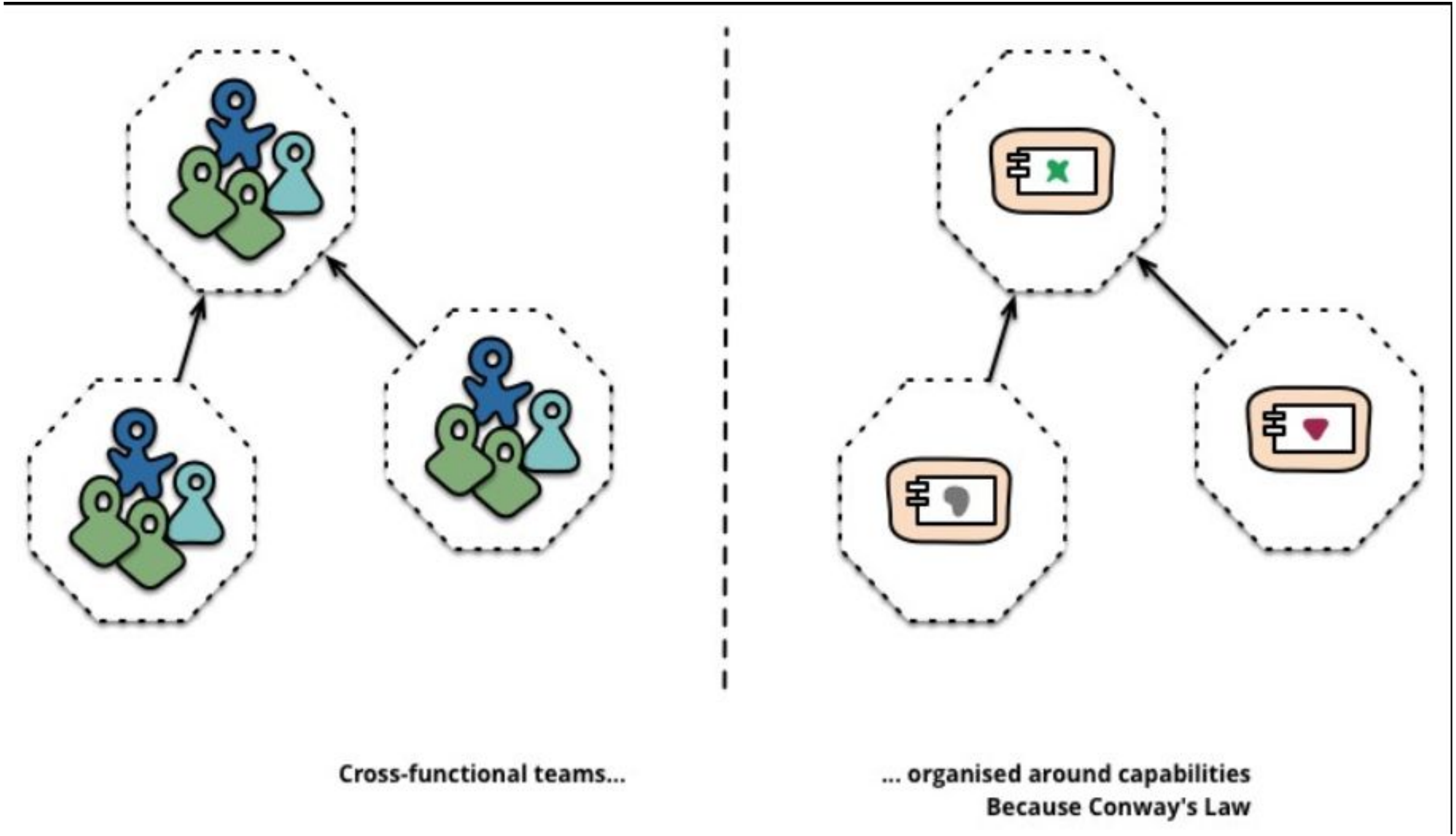
Conway's Law

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

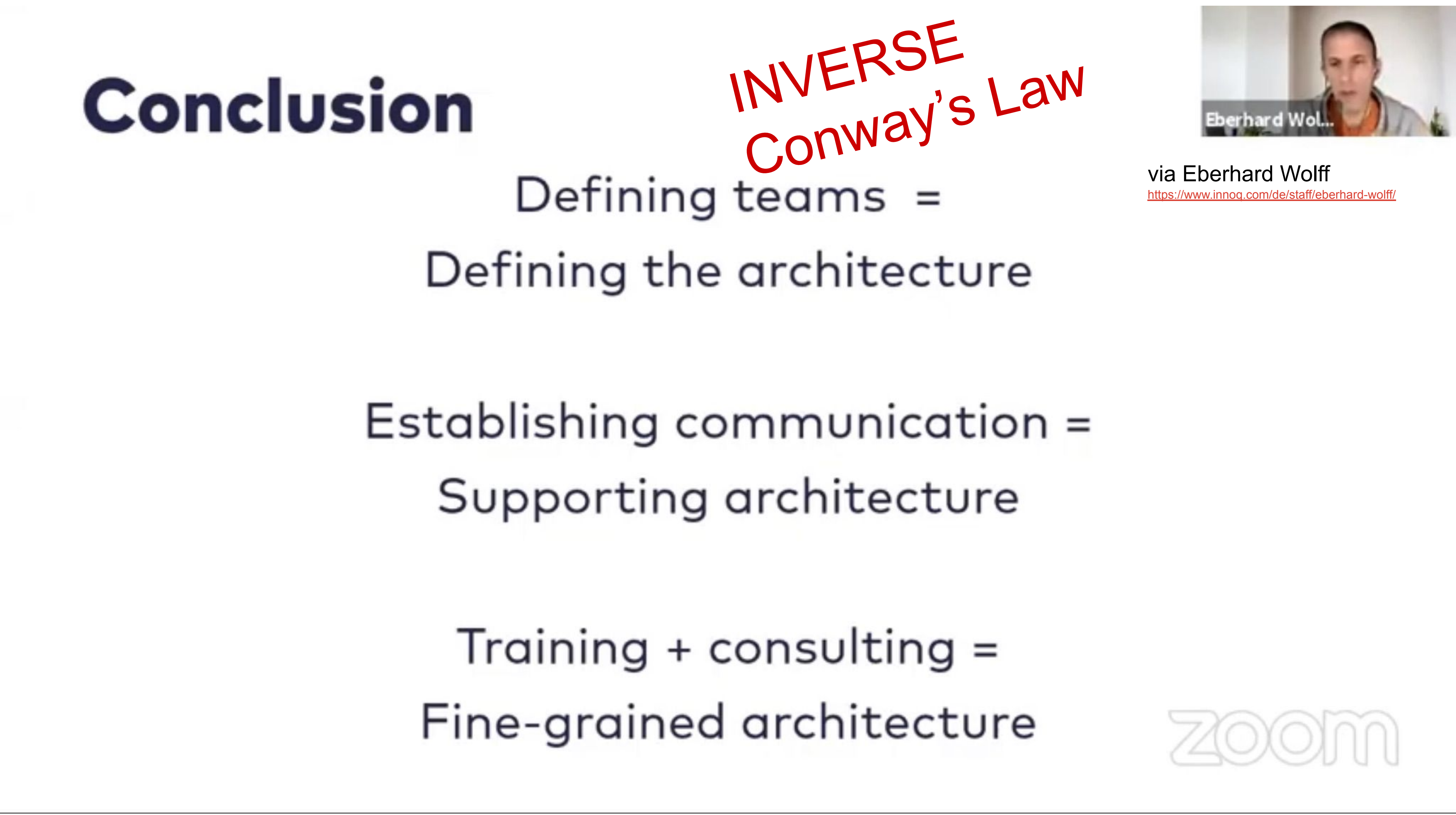
Wikipedia


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

From https://martinfowler.com/articles/microservices.html



From https://martinfowler.com/articles/microservices.html



Conclusion

INVERSE
Conway's Law

via Eberhard Wolff
https://www.innoq.com/de/staff/eberhard-wolff/

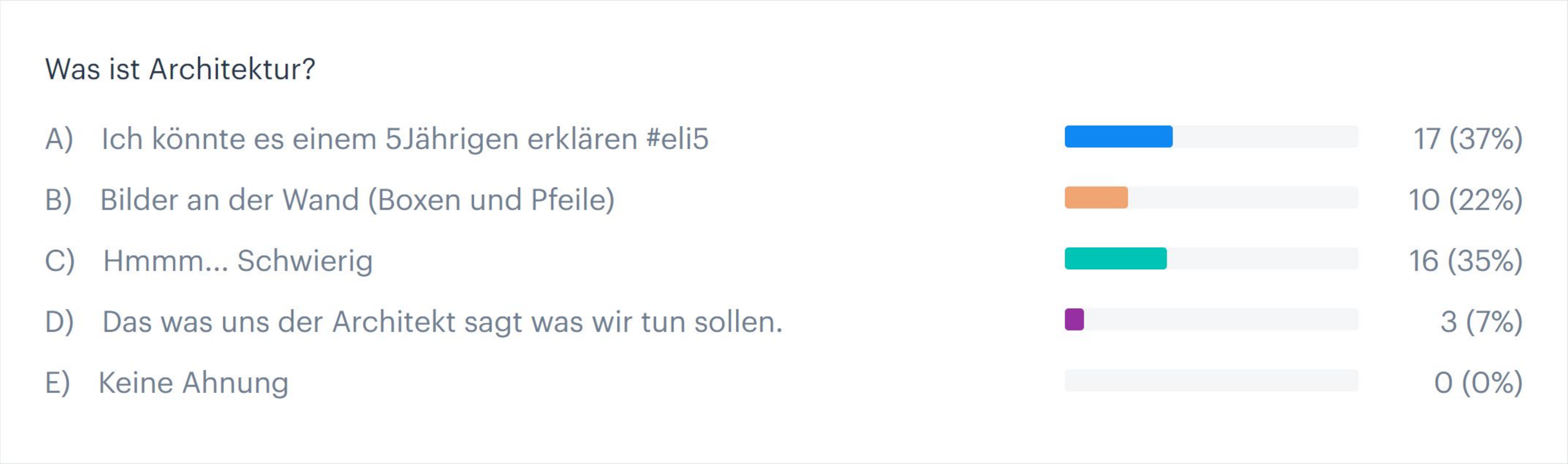
Defining teams =
Defining the architecture

Establishing communication =
Supporting architecture

Training + consulting =
Fine-grained architecture

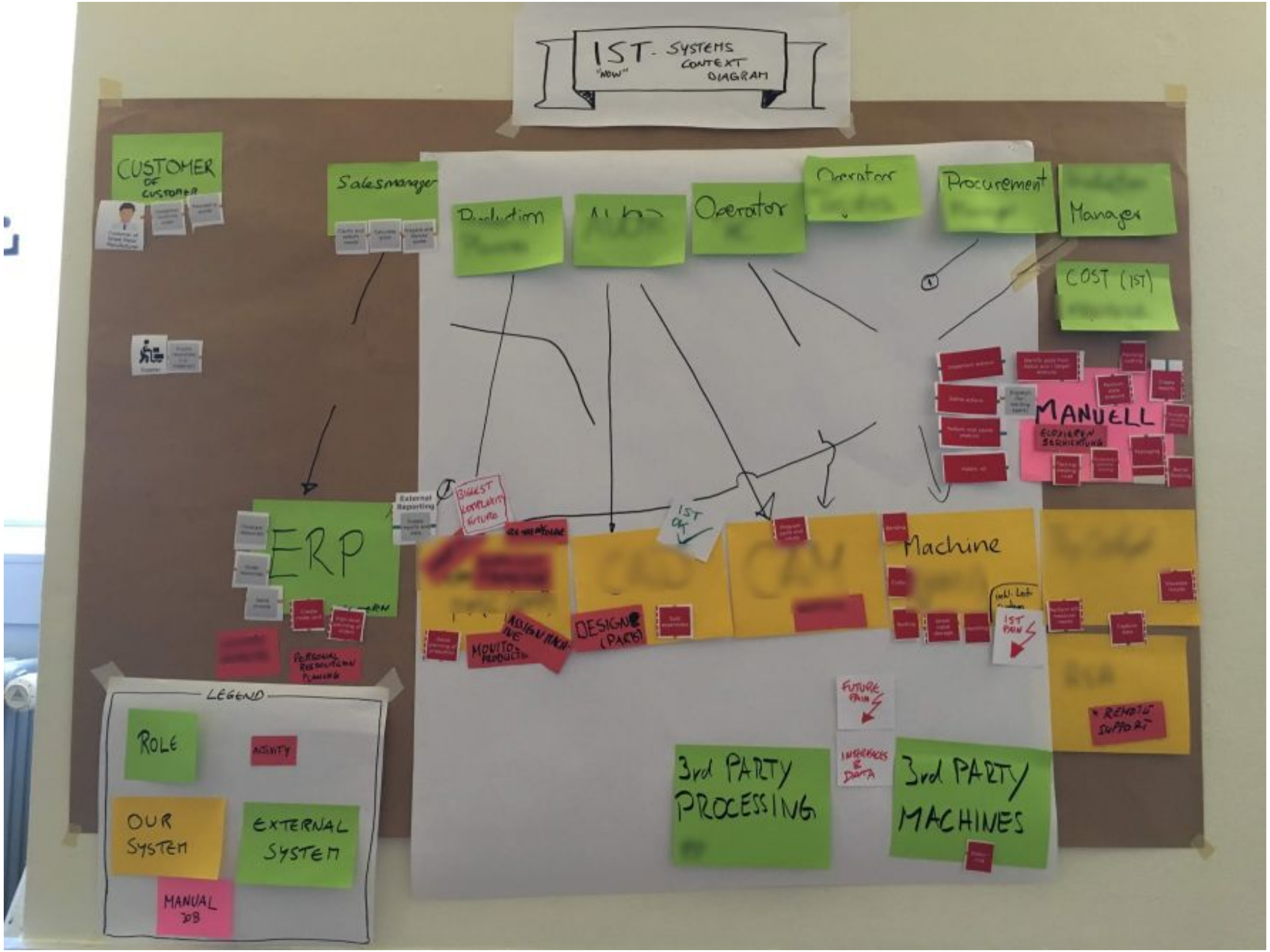
Architecture

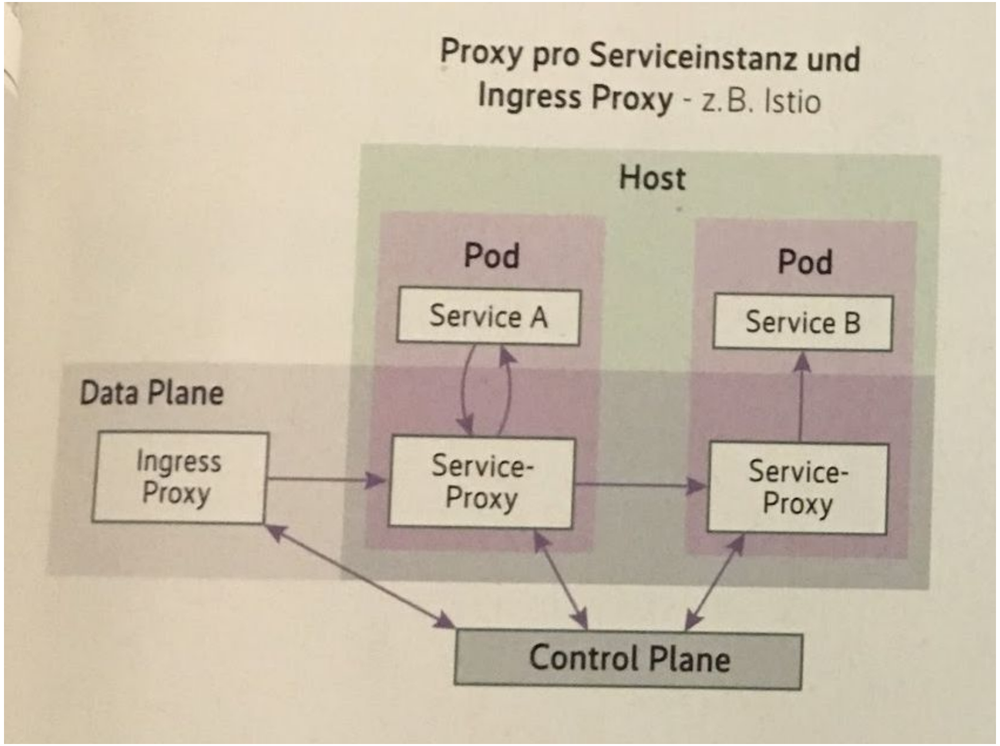
What is Architecture?



Was ist Architektur?

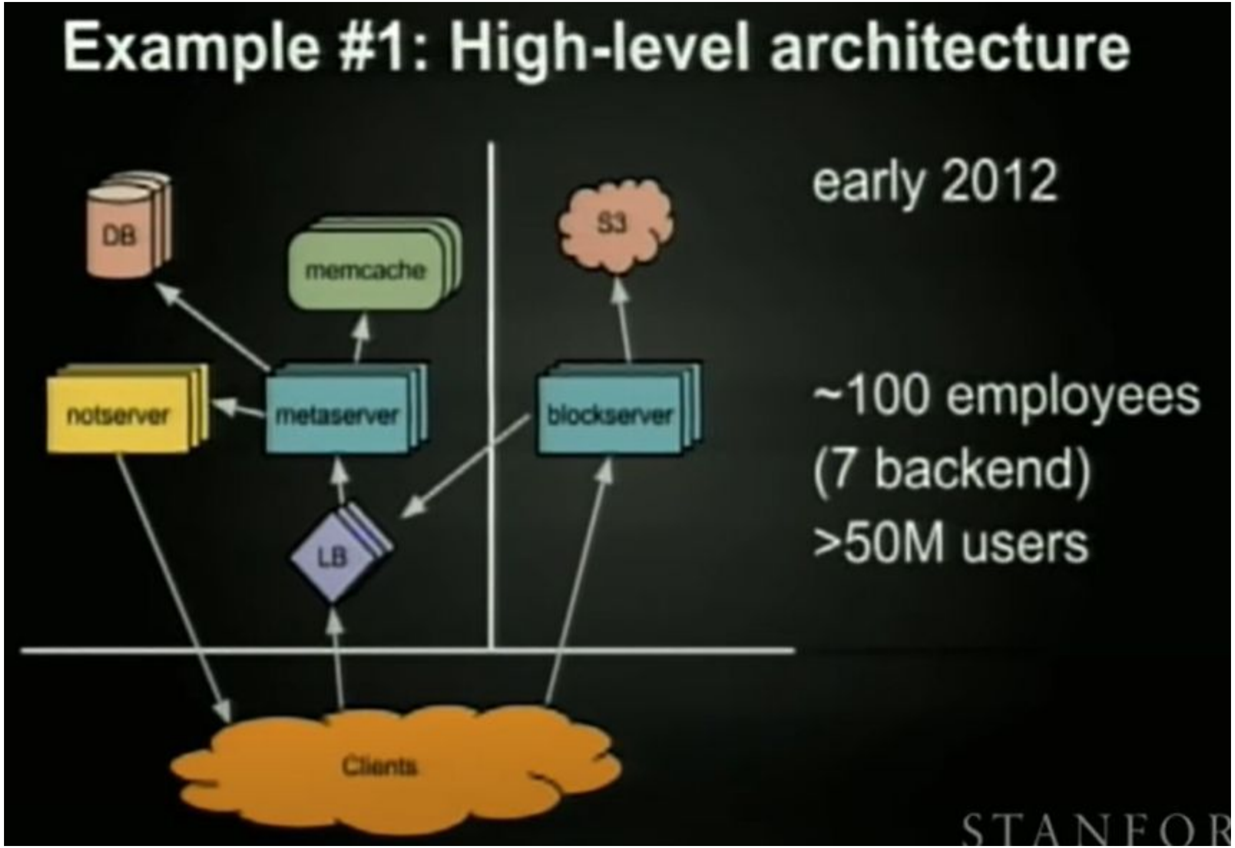
A) Ich könnte es einem 5Jährigen erklären #eli5	17 (37%)
B) Bilder an der Wand (Boxen und Pfeile)	10 (22%)
C) Hmmm... Schwierig	16 (35%)
D) Das was uns der Architekt sagt was wir tun sollen.	3 (7%)
E) Keine Ahnung	0 (0%)


IST- "NOW" SYSTEMS CONTEXT DIAGRAM
CUSTOMER OF CUSTOMER
Salesmanager
Production
Operator
Operator
Procurement
Manager
COST (IST)
MANUELL
ELOXIEREN BESCHICHTUNG
External Reporting
ERP
BIGGEST COMPLEXITY FUTURE
IST OK
Machine
DESIGN (PARTS)
ASSIGN MACHINE
MONITOR PRODUCTION
PERSONAL RESSOURCEN PLANNING
IST PAIN
FUTURE PAIN
INTERFACES & DATA
3rd PARTY PROCESSING
3rd PARTY MACHINES
* REMOTE SUPPORT
LEGEND
ROLE
ACTIVITY
OUR SYSTEM
EXTERNAL SYSTEM
MANUAL JOB



Proxy pro Serviceinstanz und Ingress Proxy - z.B. Istio

Dropbox 2012



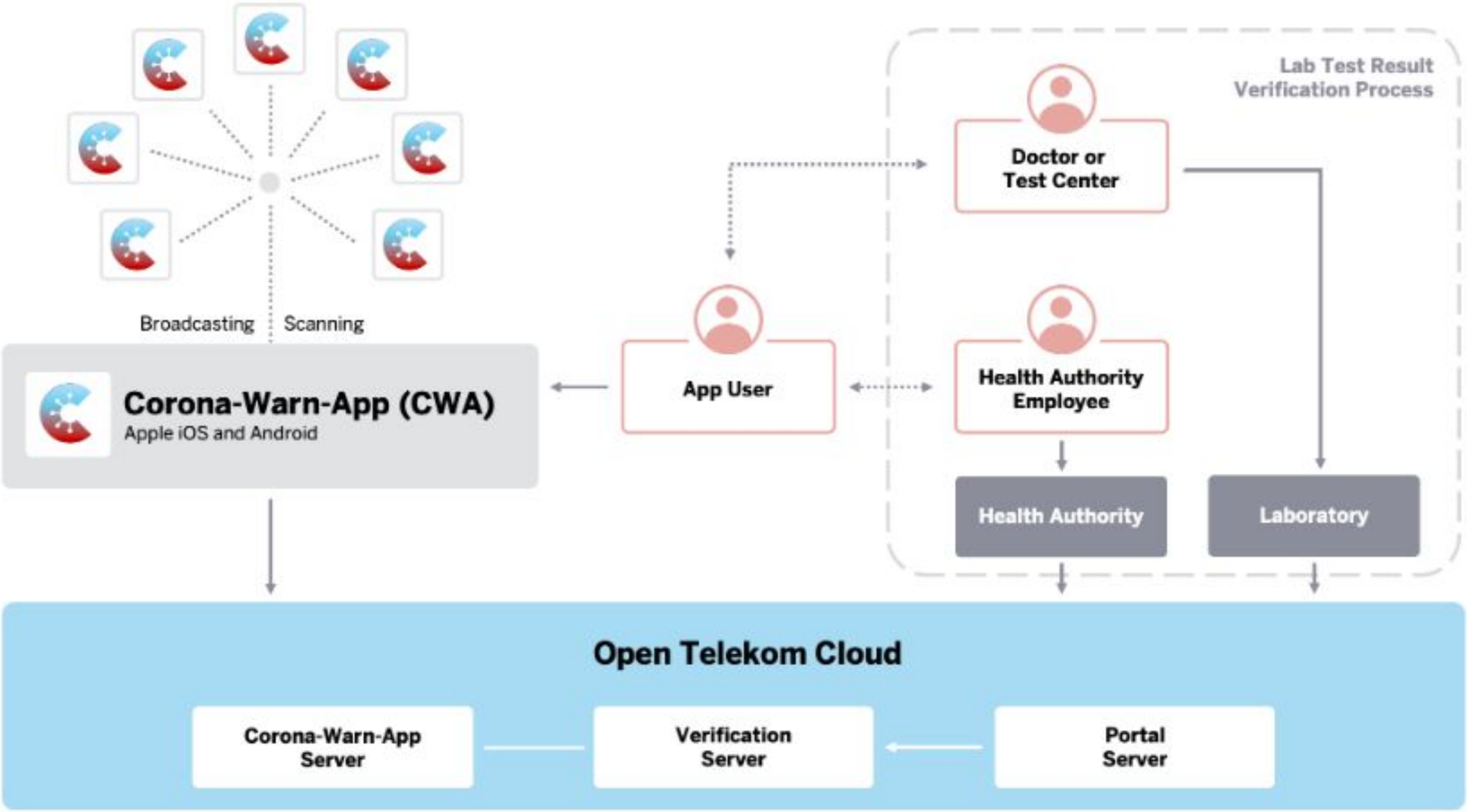
Example #1: High-level architecture

early 2012

~100 employees
(7 backend)
>50M users

How We've Scaled Dropbox
https://www.youtube.com/watch?v=PE4gwstWhmc

PRIO MGMT
GRUPPE
FINE PLANNING
DATA
MONITORING
STORAGE
MACHINE
OPERATION



Corona Warn App (Germany)

https://github.com/corona-warn-app/cwa-documentation/blob/master/solution_architecture.md

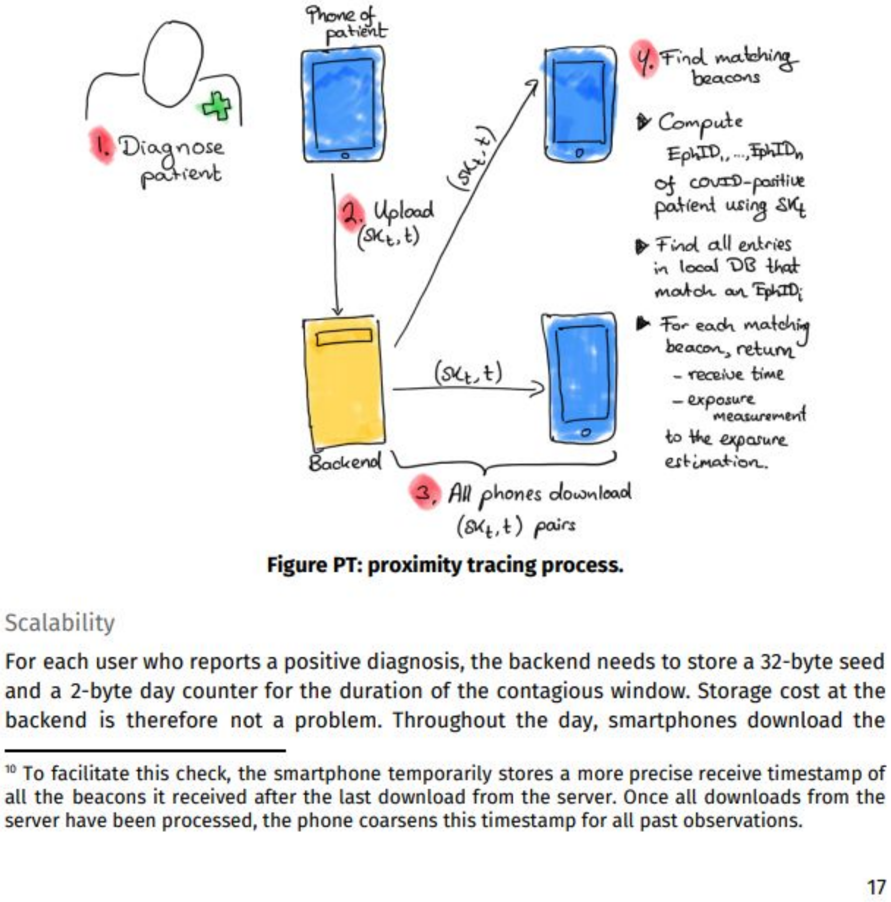

Phone of patient
1. Diagnose patient
2. Upload (SK_t, t)
(SK_t, t)
4. Find matching beacons
Compute $EphID_1, \ldots, EphID_n$ of COVID-positive patient using SK_t
Find all entries in local DB that match an $EphID_i$
For each matching beacon, return
- receive time
- exposure measurement
to the exposure estimation.
Backend
(SK_t, t)
3. All phones download (SK_t, t) pairs

Figure PT: proximity tracing process.

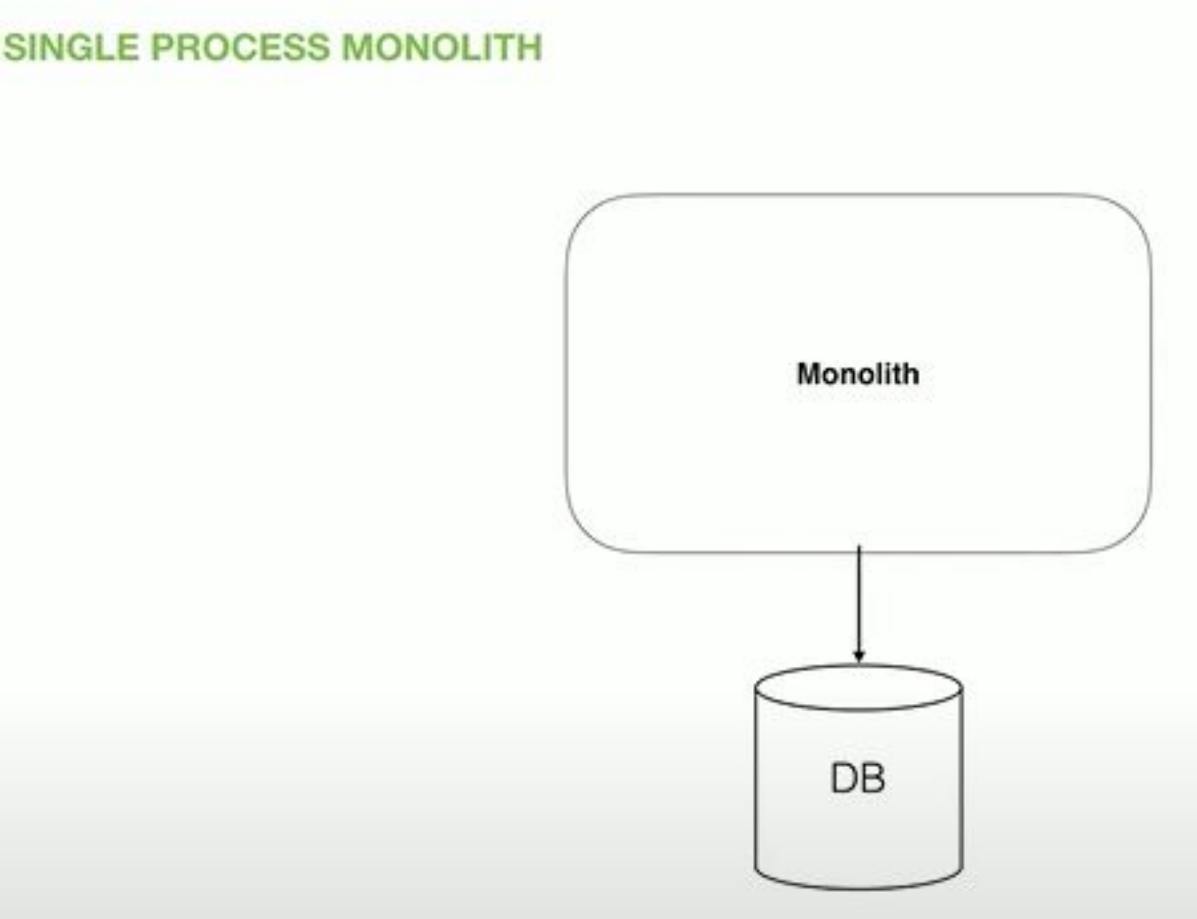
Scalability

For each user who reports a positive diagnosis, the backend needs to store a 32-byte seed and a 2-byte day counter for the duration of the contagious window. Storage cost at the backend is therefore not a problem. Throughout the day, smartphones download the

[10] To facilitate this check, the smartphone temporarily stores a more precise receive timestamp of all the beacons it received after the last download from the server. Once all downloads from the server have been processed, the phone coarsens this timestamp for all past observations.

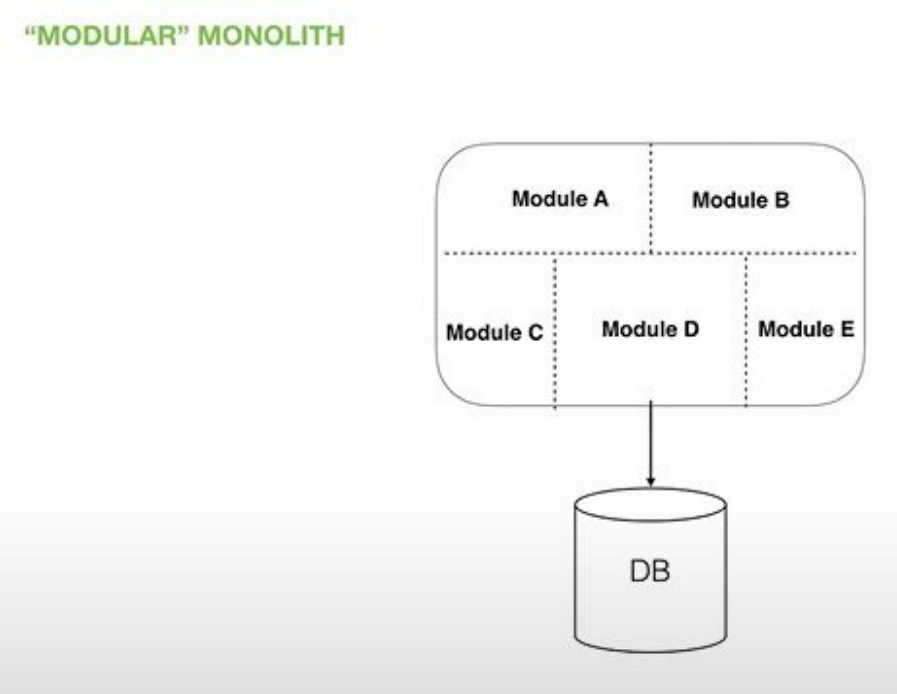
Decentralized Privacy-Preserving Proximity Tracing Switzerland (DP3T)

https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf



SINGLE PROCESS MONOLITH

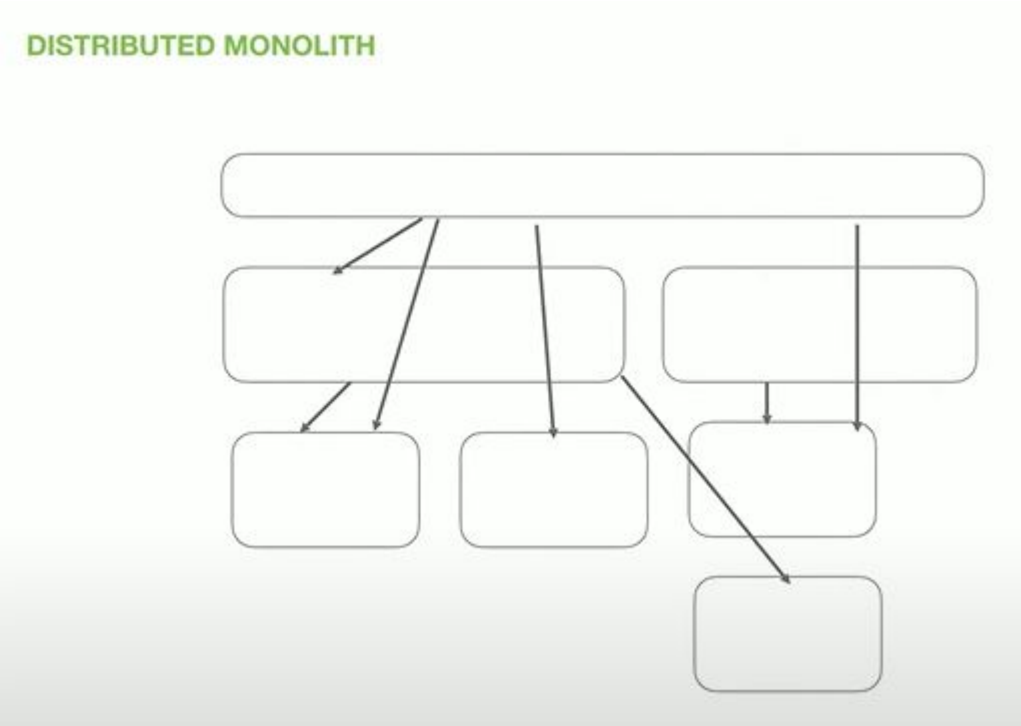
Monolith Decomposition Patterns - Sam Newman
https://www.youtube.com/watch?v=64w1zbpHGTg



“MODULAR” MONOLITH

Monolith Decomposition Patterns - Sam Newman

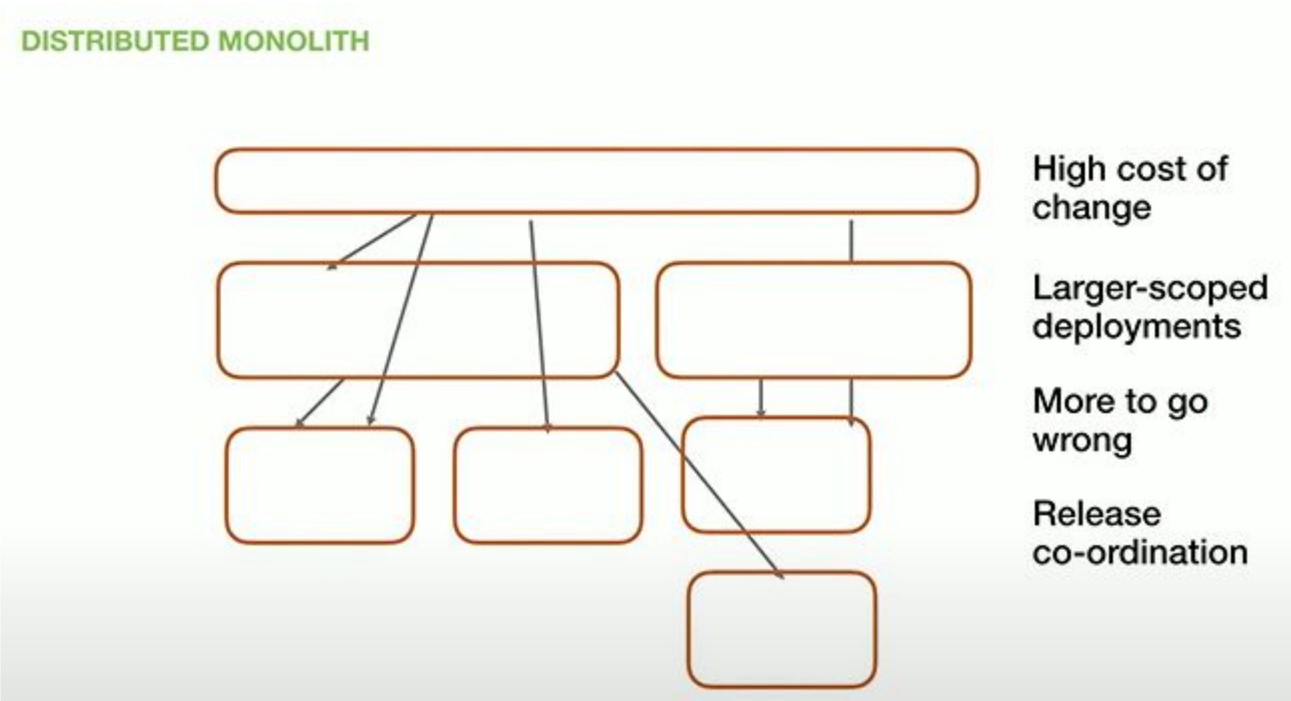
https://www.youtube.com/watch?v=64w1zbpHGTg



DISTRIBUTED MONOLITH

Monolith Decomposition Patterns - Sam Newman

https://www.youtube.com/watch?v=64w1zbpHGTg



DISTRIBUTED MONOLITH

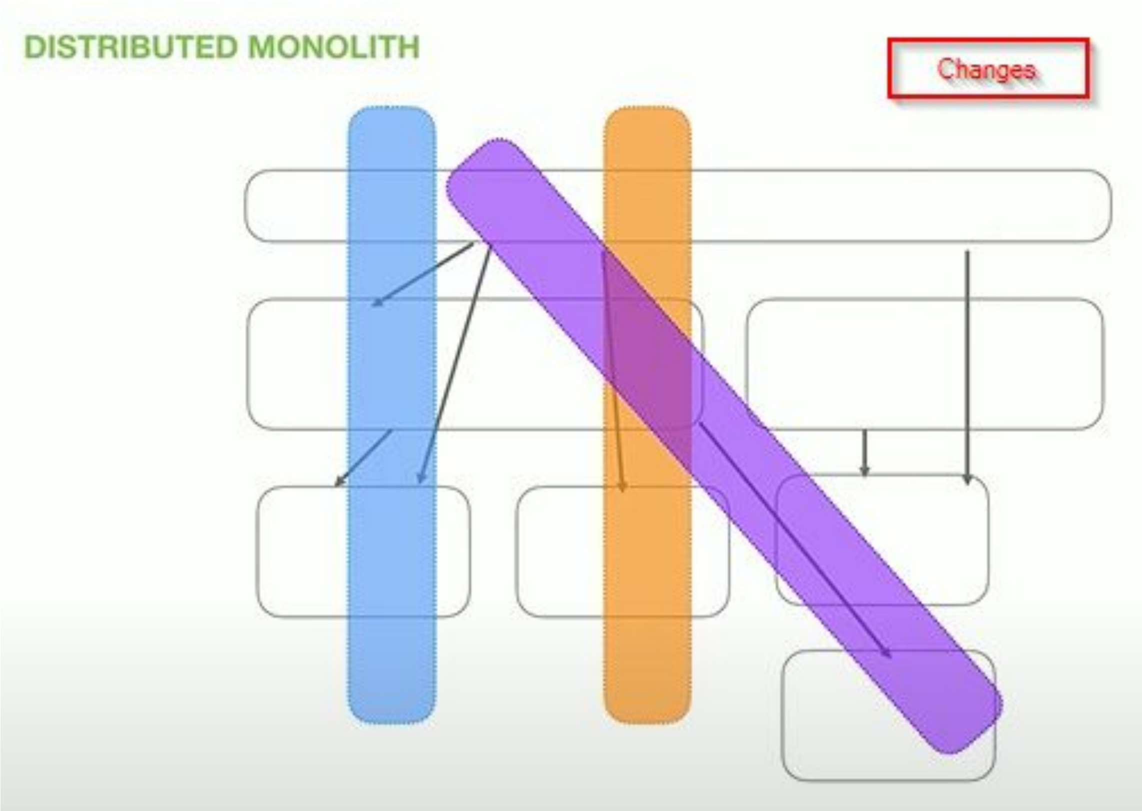
High cost of change

Larger-scoped deployments

More to go wrong

Release co-ordination

Monolith Decomposition Patterns - Sam Newman
https://www.youtube.com/watch?v=64w1zbpHGTg



Monolith Decomposition Patterns - Sam Newman
https://www.youtube.com/watch?v=64w1zbpHGTg

What is Architecture?

What is Architecture?

“Decisions that have a bigger impact.”

Peter Gfader

Why Architecture?

With architecture we attempt
  to minimize the cumulative costs of a system over its lifetime
  while ensuring the correctness of behavior at runtime

via Uwe Friedrichsen

What is GOOD Architecture?

“Good architecture attracts people”

What is GOOD Architecture?

“Good architecture enables fast high quality feedback”

Continuous Delivery

2 Laws Of Architecture (via Neal Ford)

1. Everything is a Tradeoff

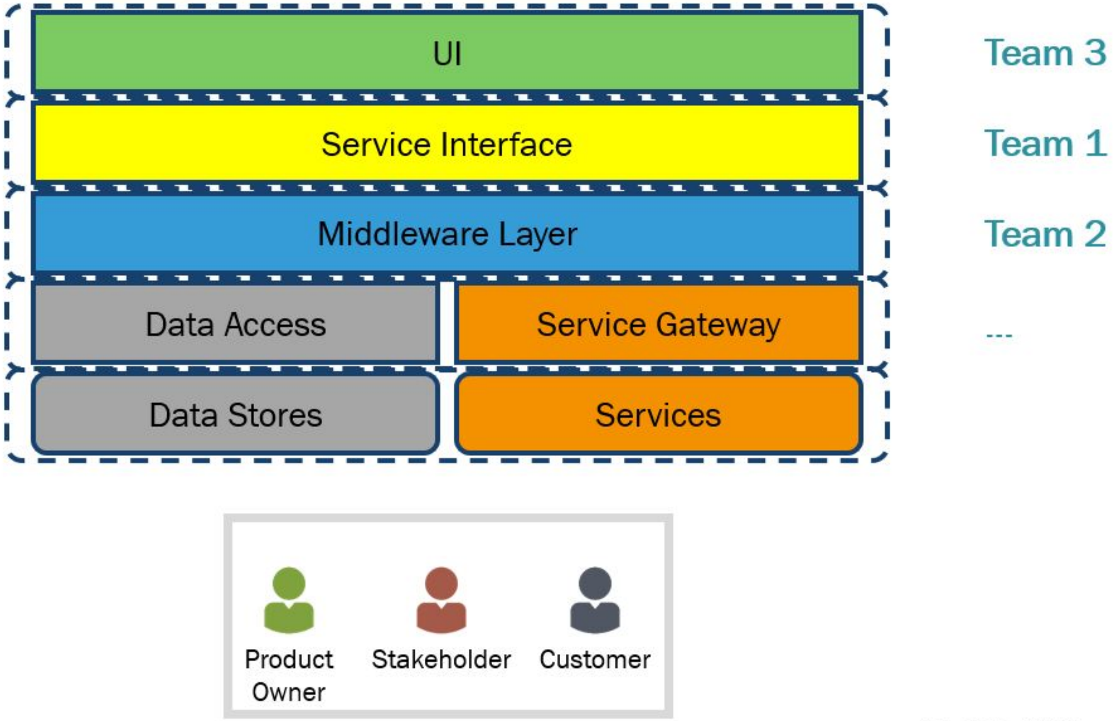
corolal: you havent identified the tradeoff yet.

2. Why is more important than the How

architecture diagrams
ADR are more important
why you choose something... tradeoffs.
decisions always in a context

Communitaion Team Setup

Component or Layer Teams Face Additional Complexities



UI

Service Interface

Middleware Layer

Data Access

Service Gateway

Data Stores

Services

Team 3

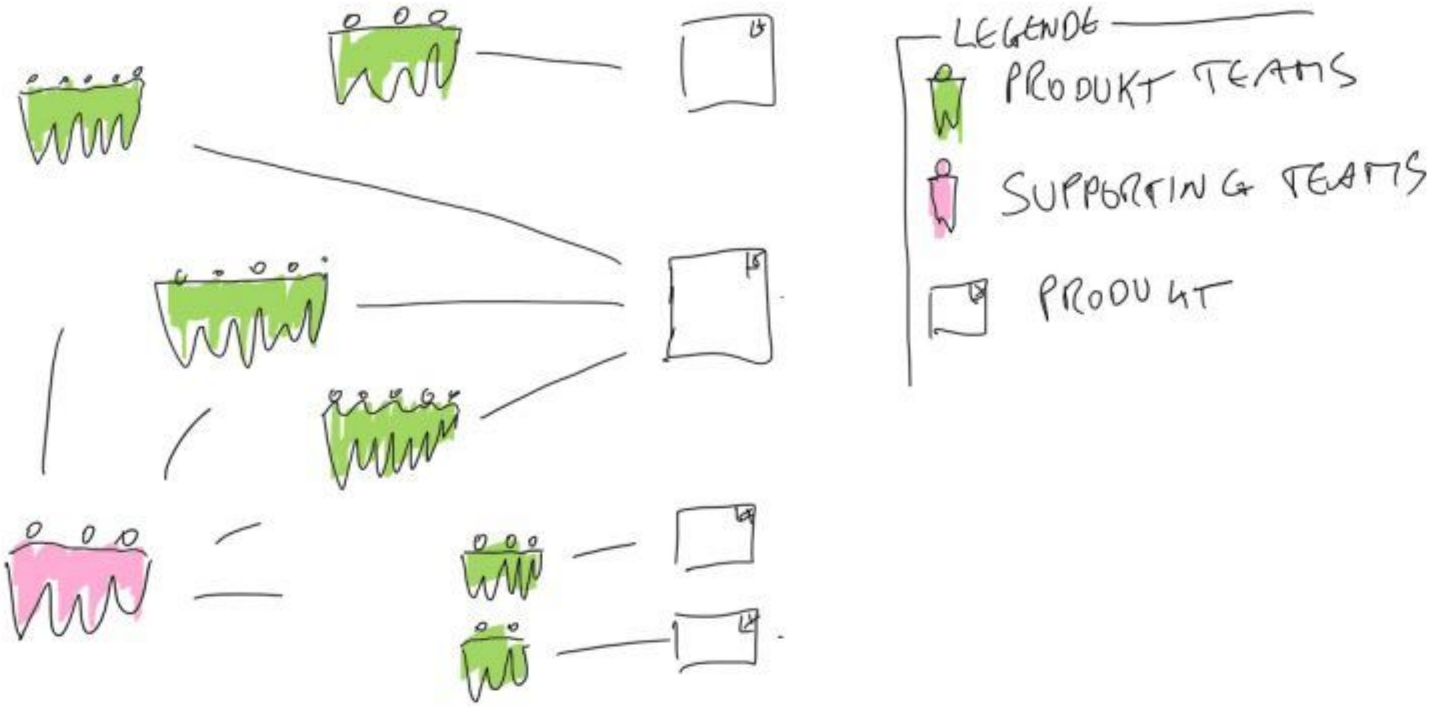
Team 1

Team 2

Product Owner

Stakeholder

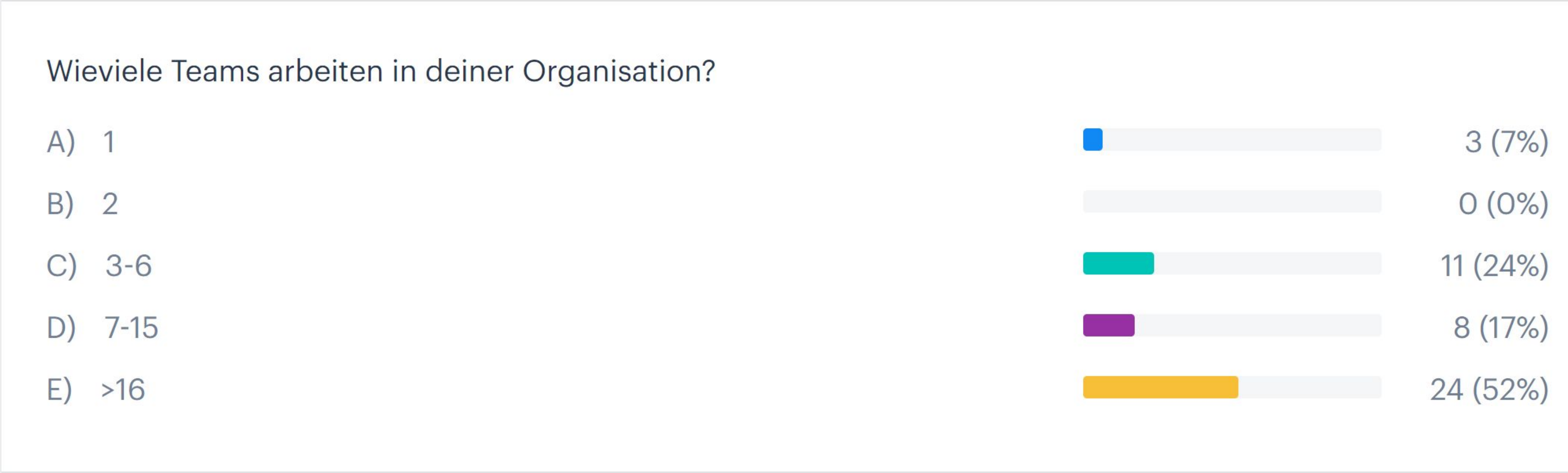
Customer



More info here: Trust in Product Teams

https://www.digicomp.ch/blog/2019/06/24/wieso-das-management-development-teams-nicht-vertraut

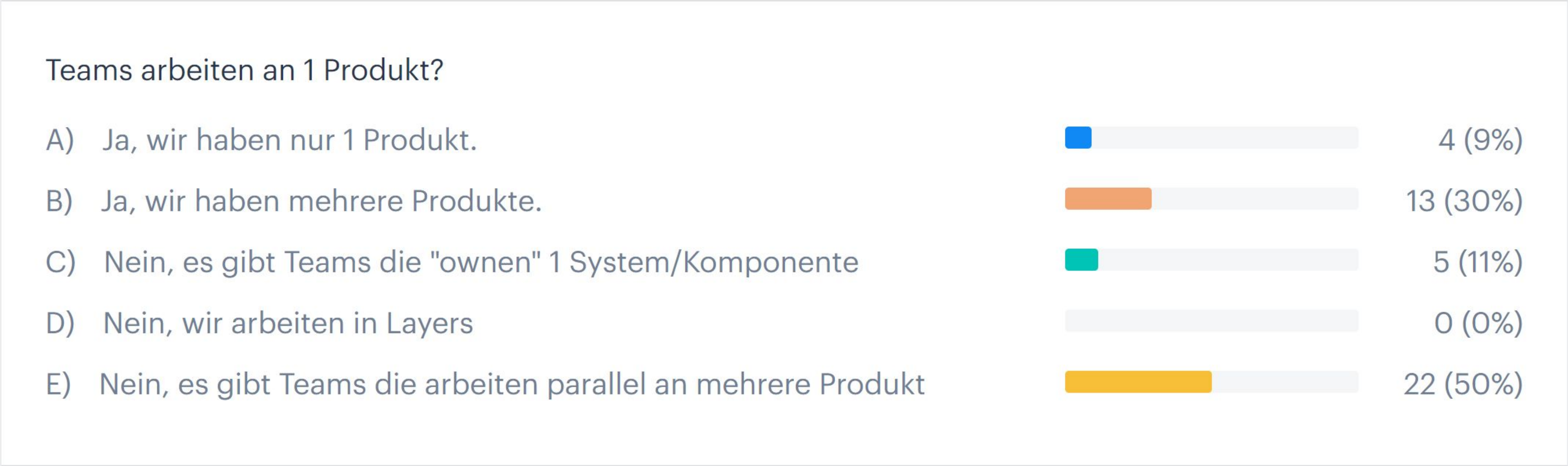
How many Teams?



Wieviele Teams arbeiten in deiner Organisation?

A) 1	3 (7%)
B) 2	0 (0%)
C) 3-6	11 (24%)
D) 7-15	8 (17%)
E) >16	24 (52%)

What kind of Team Setup do you have?



Teams arbeiten an 1 Produkt?

A)	Ja, wir haben nur 1 Produkt.	4 (9%)
B)	Ja, wir haben mehrere Produkte.	13 (30%)
C)	Nein, es gibt Teams die "ownen" 1 System/Komponente	5 (11%)
D)	Nein, wir arbeiten in Layers	0 (0%)
E)	Nein, es gibt Teams die arbeiten parallel an mehrere Produkt	22 (50%)

Work

Work

What a Team does? (and owns)

What a Team works on?

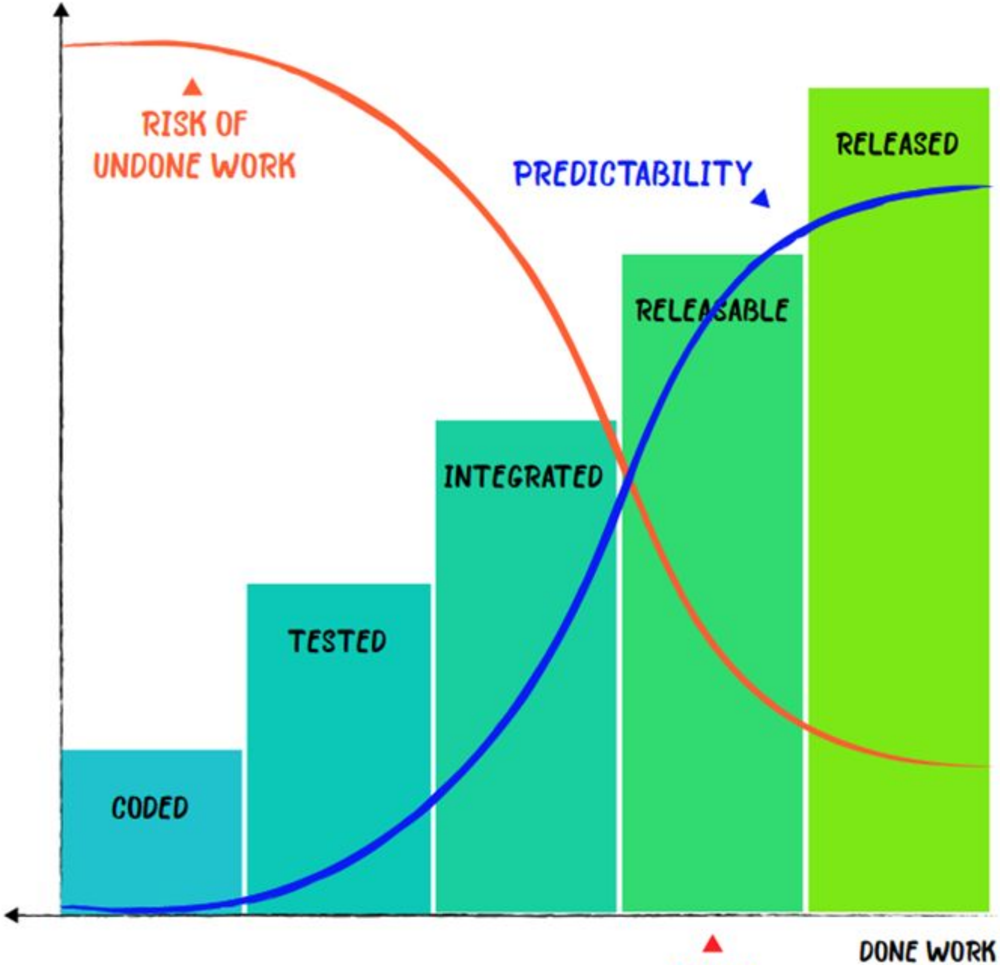
How work comes to the team?

How Testing is Different in an Agile Project?

HOW TESTING IS DIFFERENT IN AGILE THAN IN A TRADITIONAL PROJECT
PETER GFADER
0:09 / 4:33

https://www.youtube.com/watch?v=xdak981_v3g

What a Team Does

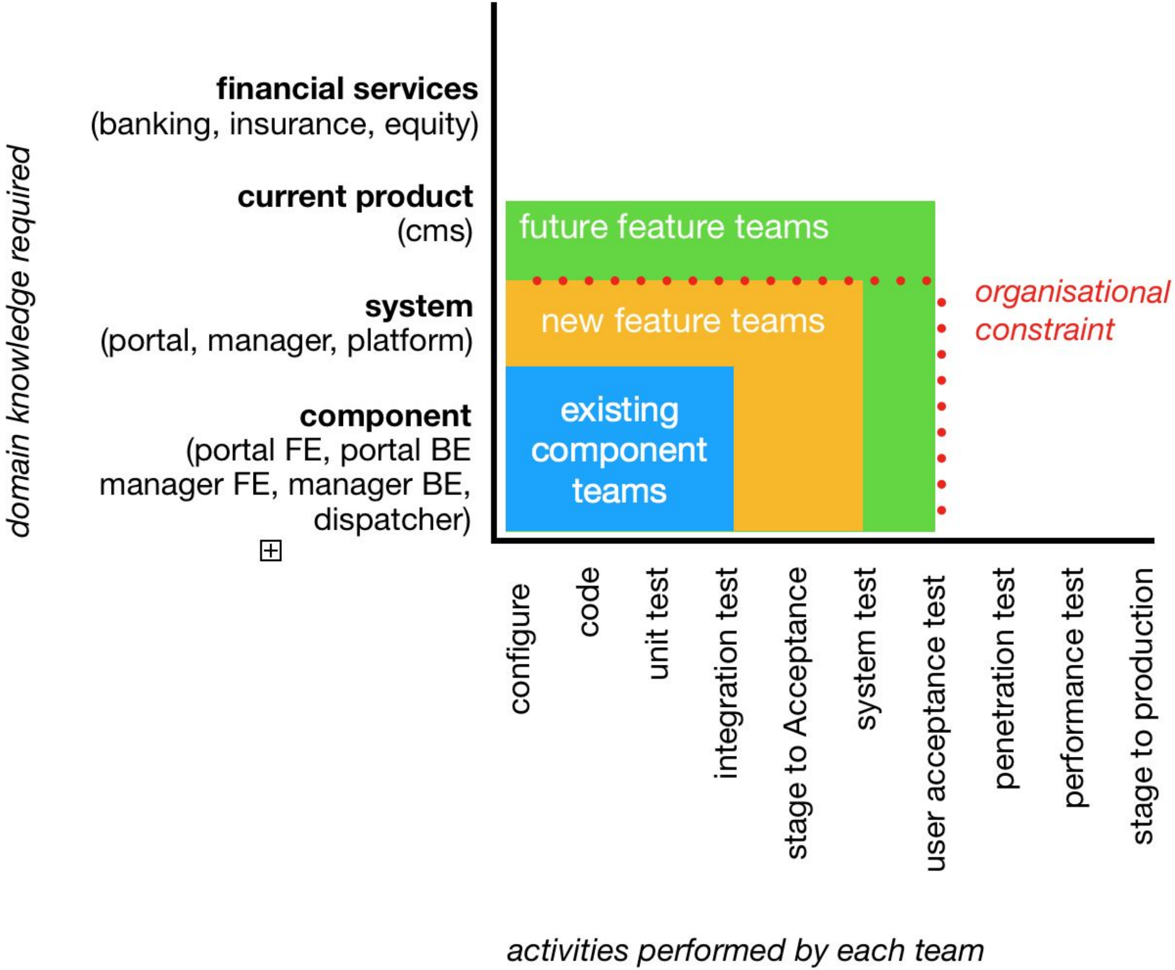


Deployed VS Released

Can You Distinguish between Deployment and Release?

3 minute video

https://www.youtube.com/watch?v=RBQz9QhFfdc



domain knowledge required

financial services
(banking, insurance, equity)

current product
(cms)

system
(portal, manager, platform)

component
(portal FE, portal BE
manager FE, manager BE,
dispatcher)

future feature teams

new feature teams

existing component teams

organisational constraint

configure

code

unit test

integration test

stage to Acceptance

system test

user acceptance test

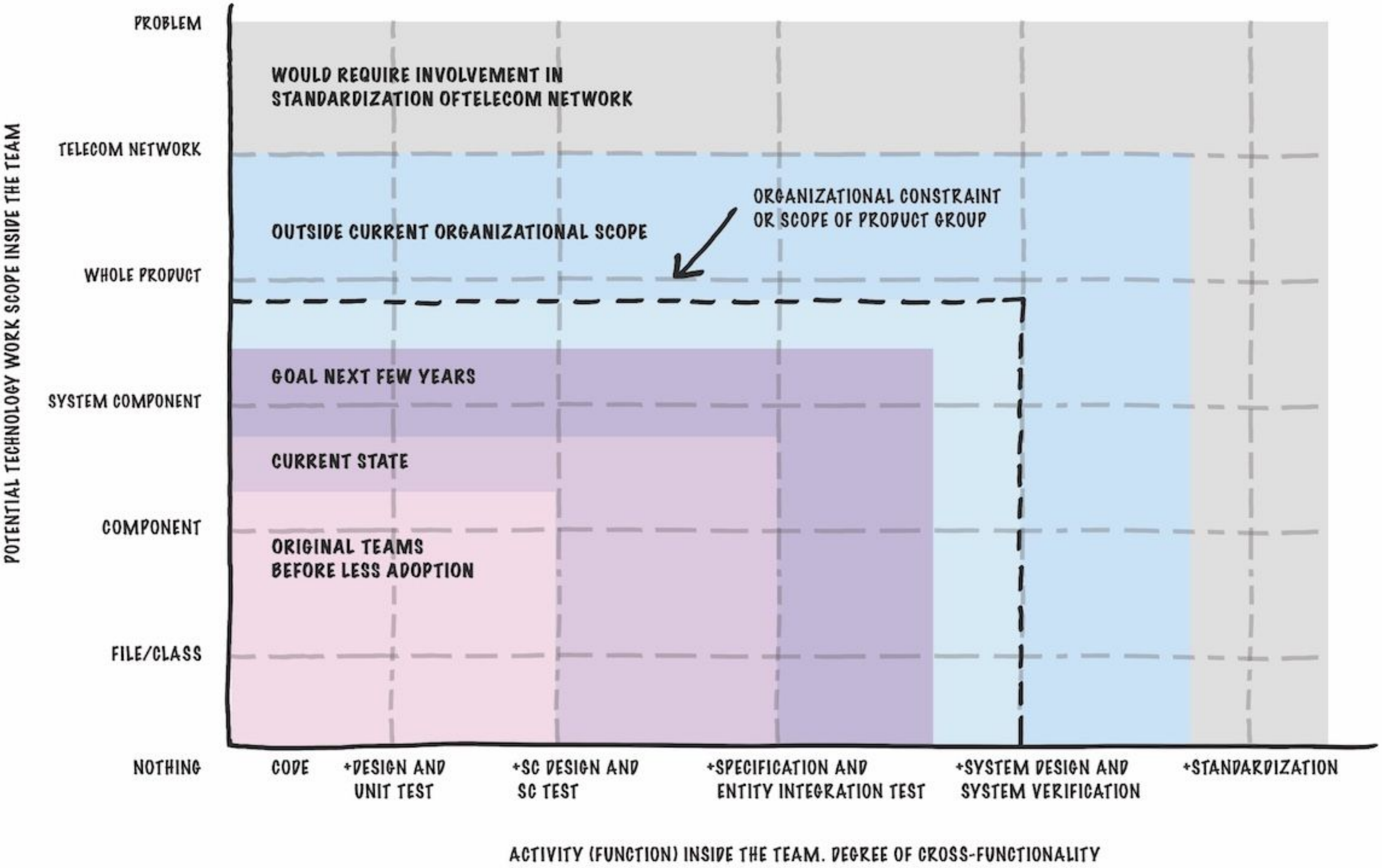
penetration test

performance test

stage to production

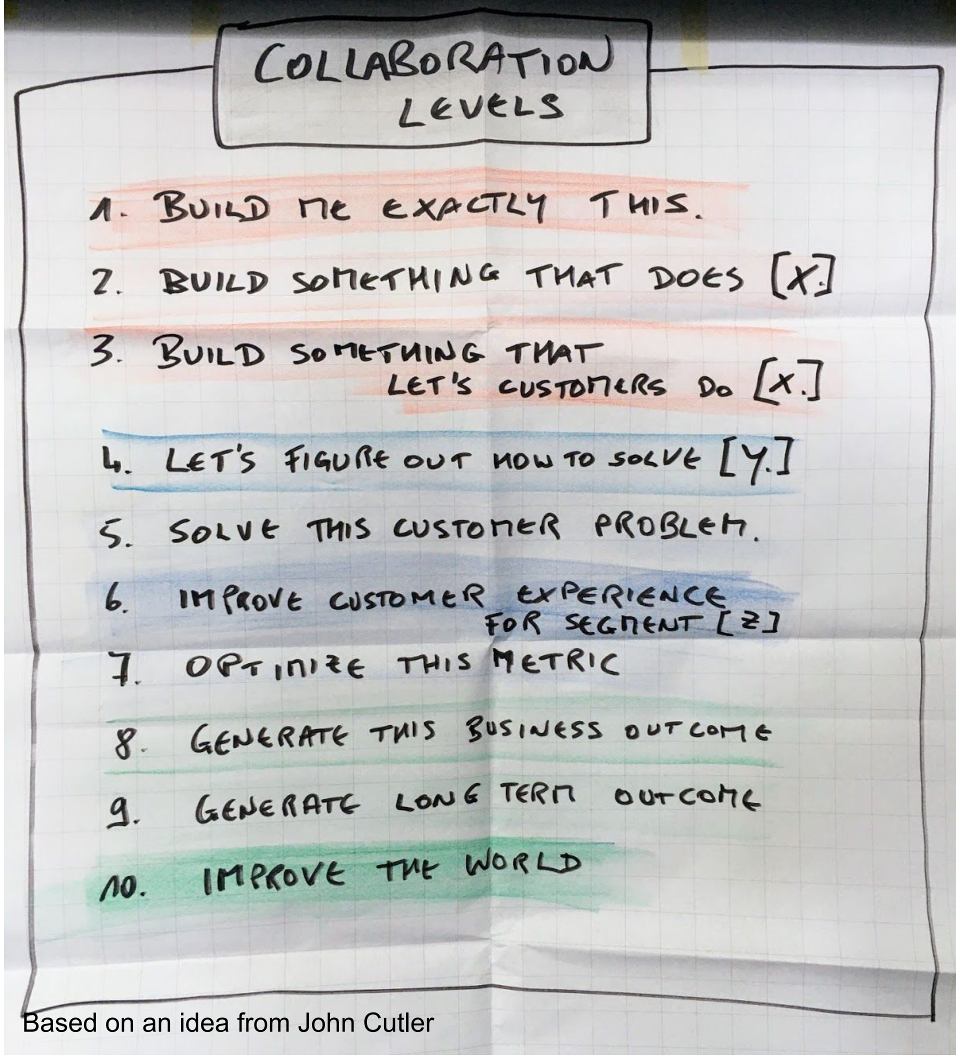
activities performed by each team

https://less.works/less/adoption/feature-team-adoption_map


PROBLEM
TELECOM NETWORK
WHOLE PRODUCT
SYSTEM COMPONENT
COMPONENT
FILE/CLASS
NOTHING
POTENTIAL TECHNOLOGY WORK SCOPE INSIDE THE TEAM
WOULD REQUIRE INVOLVEMENT IN STANDARDIZATION OFTELECOM NETWORK
OUTSIDE CURRENT ORGANIZATIONAL SCOPE
ORGANIZATIONAL CONSTRAINT OR SCOPE OF PRODUCT GROUP
GOAL NEXT FEW YEARS
CURRENT STATE
ORIGINAL TEAMS BEFORE LESS ADOPTION
CODE
+DESIGN AND UNIT TEST
+SC DESIGN AND SC TEST
+SPECIFICATION AND ENTITY INTEGRATION TEST
+SYSTEM DESIGN AND SYSTEM VERIFICATION
+STANDARDIZATION
ACTIVITY (FUNCTION) INSIDE THE TEAM. DEGREE OF CROSS-FUNCTIONALITY

https://less.works/less/adoption/feature-team-adoption_map

How does work come to Teams?

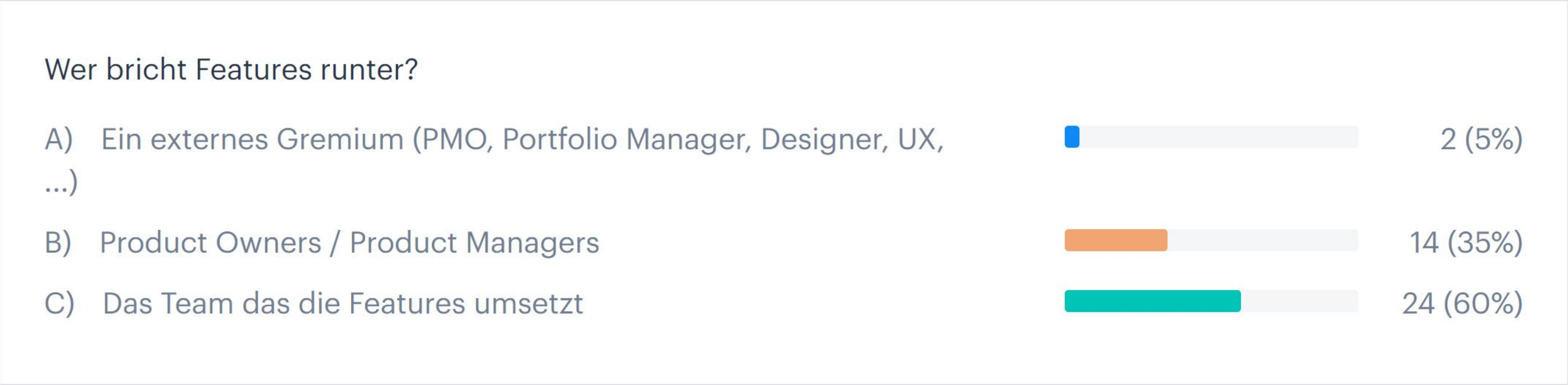


COLLABORATION LEVELS

1. BUILD ME EXACTLY THIS.
2. BUILD SOMETHING THAT DOES [X.]
3. BUILD SOMETHING THAT LET'S CUSTOMERS DO [X.]
4. LET'S FIGURE OUT HOW TO SOLVE [Y.]
5. SOLVE THIS CUSTOMER PROBLEM.
6. IMPROVE CUSTOMER EXPERIENCE FOR SEGMENT [Z]
7. OPTIMIZE THIS METRIC
8. GENERATE THIS BUSINESS OUTCOME
9. GENERATE LONG TERM OUTCOME
10. IMPROVE THE WORLD

Based on an idea from John Cutler

Work breakdown VS Slicing



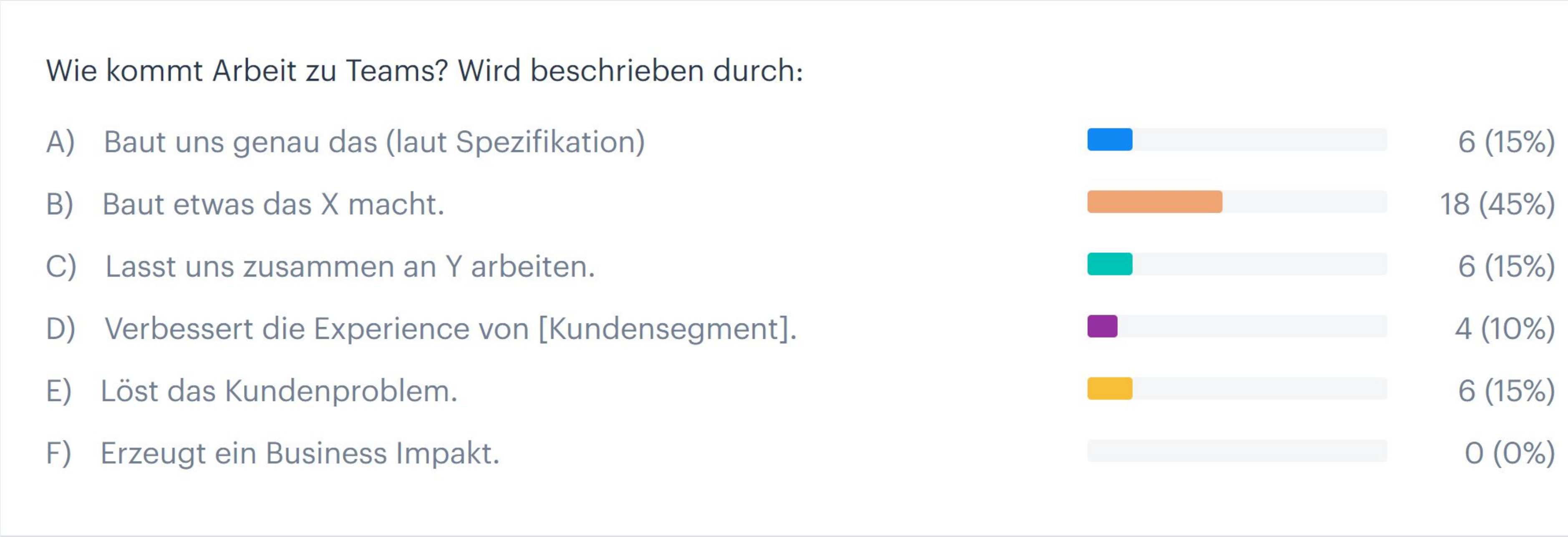
Wer bricht Features runter?

A) Ein externes Gremium (PMO, Portfolio Manager, Designer, UX, ...) 2 (5%)

B) Product Owners / Product Managers 14 (35%)

C) Das Team das die Features umsetzt 24 (60%)

How does work come to Teams?



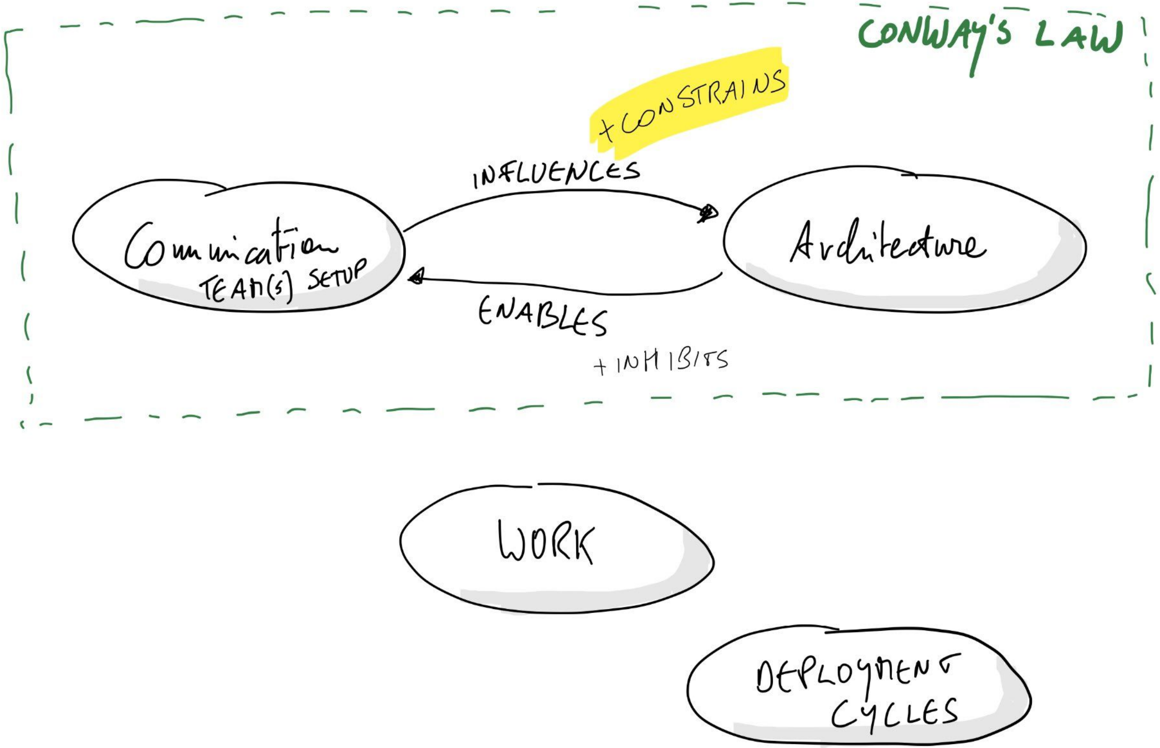
Wie kommt Arbeit zu Teams? Wird beschrieben durch:

A) Baut uns genau das (laut Spezifikation)	6 (15%)
B) Baut etwas das X macht.	18 (45%)
C) Lasst uns zusammen an Y arbeiten.	6 (15%)
D) Verbessert die Experience von [Kundensegment].	4 (10%)
E) Löst das Kundenproblem.	6 (15%)
F) Erzeugt ein Business Impakt.	0 (0%)

Conway's Law

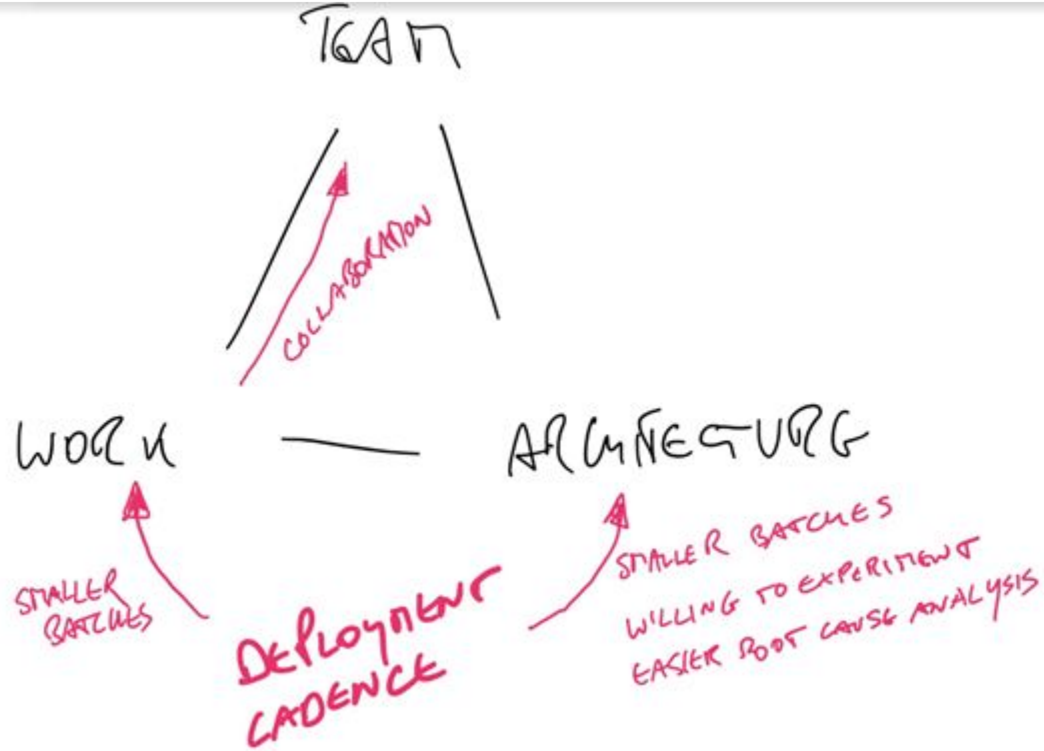
Extended

Conway's law EXTENDED



- **Decision on these 3 topics -> in same room**
  Have Architecture Teams for some time… and then go back into previous setup
- **Architecture <-> Team Communication**
  - Self Org Teams ← Is that even possible?
  - Scale up.. Multiple teams
- Who changes & decides team setup? Slicing of work?
- Who decides the work for teams? What size?
- Who decides the architecture?
- Project VS Maintenance
- Component VS Layer VS Feature Teams
- *“iOS Devs don’t want to do something else”*
- Microservices
- How we work? Pairing, Mob Programming?

Conway's law EXTENDED



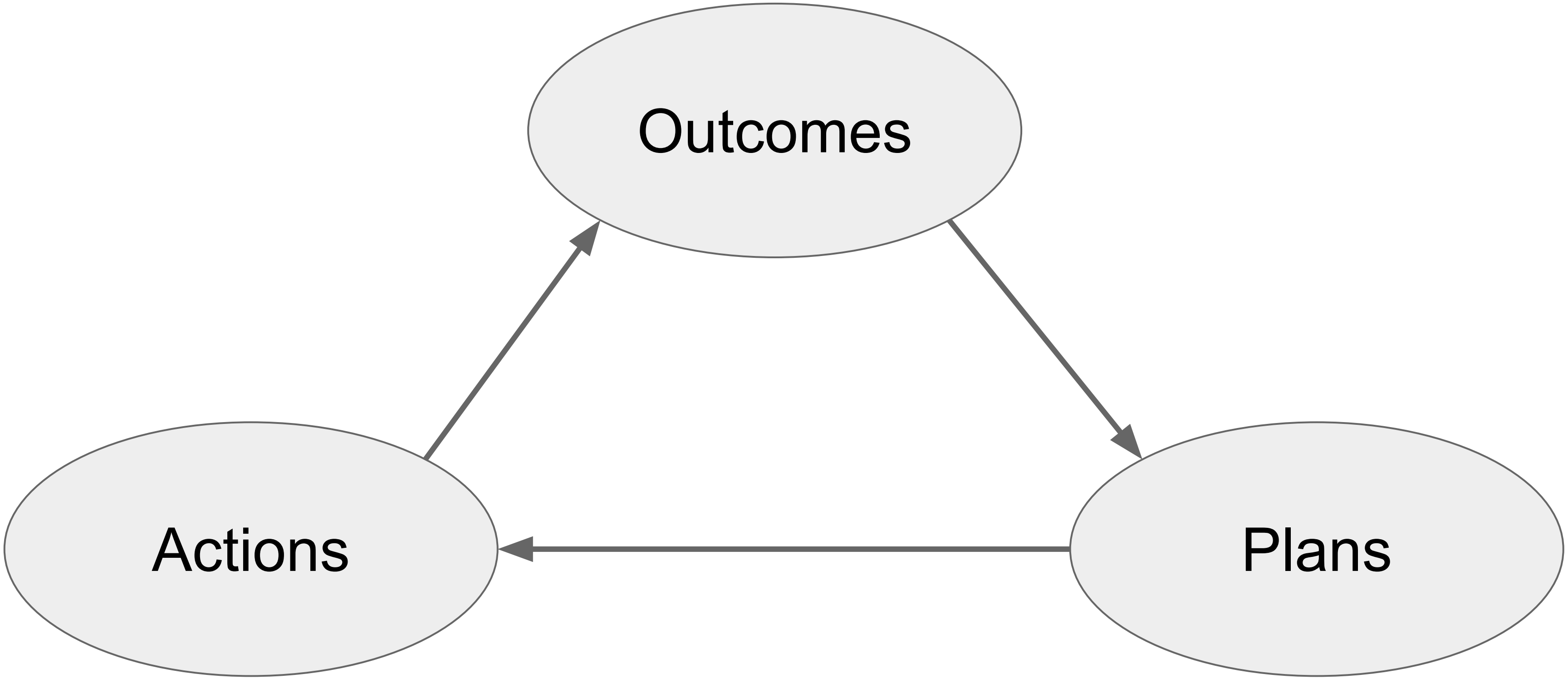
Conway's law Extended

Organizations which design systems … are constrained to produce designs,
which are copies of the communication structures of these organizations ***and the work****.*

Not on Wikipedia yet :)

What Now??

The Nature Of The Problem



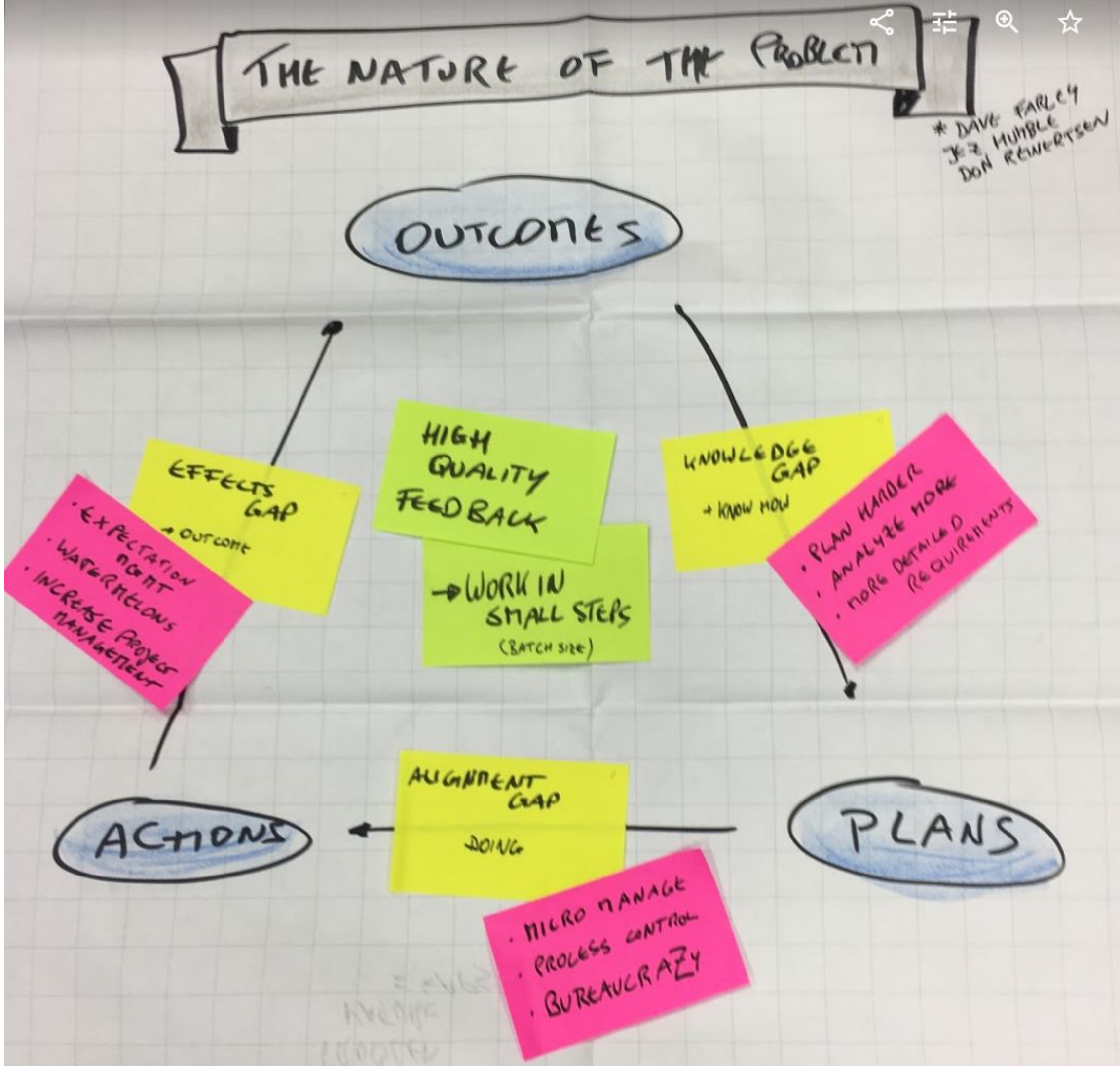
Source: "The Art of Action" by Stephen Bungay

From “The Art of Action”
- Stephen Bungay

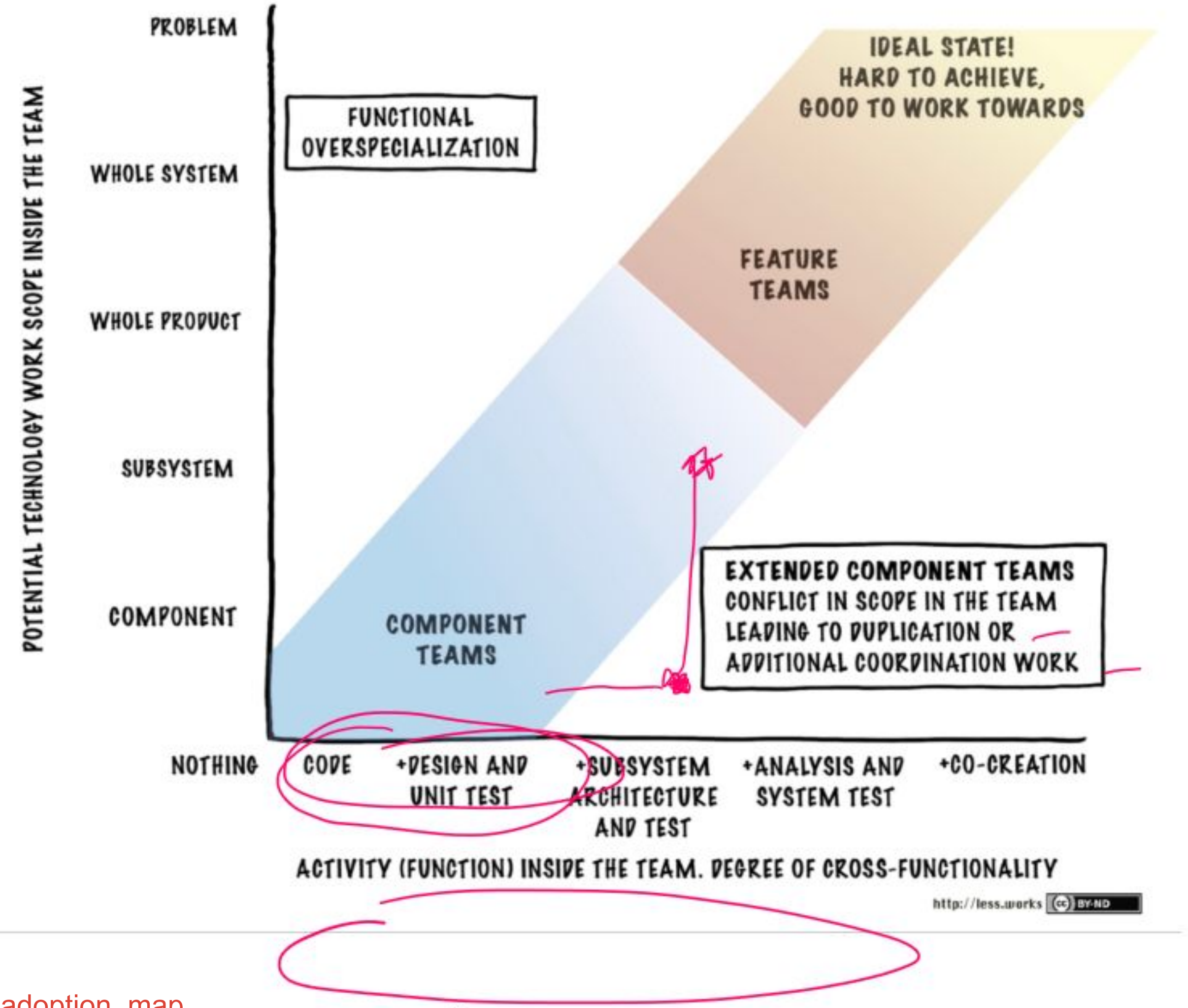
3 minute video
https://youtu.be/FVEWdatM8Uk?t=1075

via Jez Humble and Dave Farley

Min 17:58 until 21:00



Feature Team Adoption Map


POTENTIAL TECHNOLOGY WORK SCOPE INSIDE THE TEAM
PROBLEM
WHOLE SYSTEM
WHOLE PRODUCT
SUBSYSTEM
COMPONENT
FUNCTIONAL OVERSPECIALIZATION
IDEAL STATE! HARD TO ACHIEVE, GOOD TO WORK TOWARDS
FEATURE TEAMS
COMPONENT TEAMS
EXTENDED COMPONENT TEAMS
CONFLICT IN SCOPE IN THE TEAM LEADING TO DUPLICATION OR ADDITIONAL COORDINATION WORK
NOTHING
CODE
+DESIGN AND UNIT TEST
+SUBSYSTEM ARCHITECTURE AND TEST
+ANALYSIS AND SYSTEM TEST
+CO-CREATION
ACTIVITY (FUNCTION) INSIDE THE TEAM. DEGREE OF CROSS-FUNCTIONALITY
http://less.works
BY-ND
RELEASE CYCLES

https://less.works/less/adoption/feature-team-adoption_map

Q&A

Was hältst du von der These: Teams die nicht miteinander kommunizieren bauen lose gekoppelte Systeme. Daher sollten wir die Kommunikation zwischen den Teams reduzieren oder gleich verbieten ;)

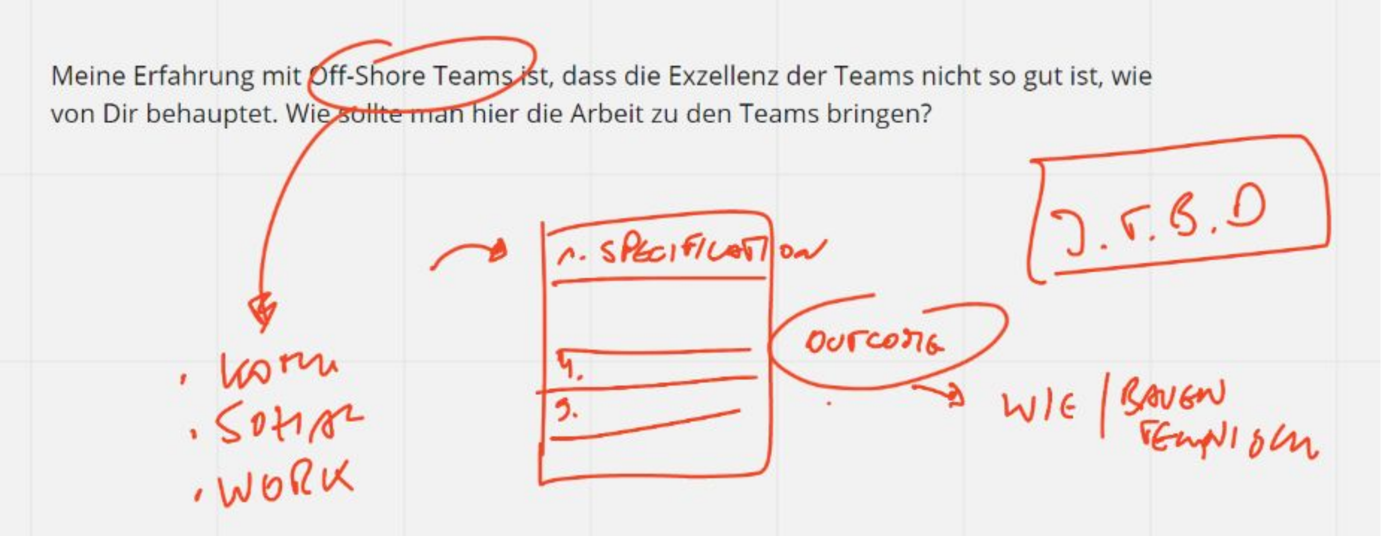
Tönt nach einem spannenden Experiment :)

Empfehlung:

- Reflektion im Team(s)→ “Self Designing Team Workshop”

Meine Erfahrung mit Off-Shore Teams ist, dass die Exzellenz der Teams nicht so gut ist, wie von Dir behauptet. Wie sollte man hier die Arbeit zu den Teams bringen?

Challenge Kommunikation

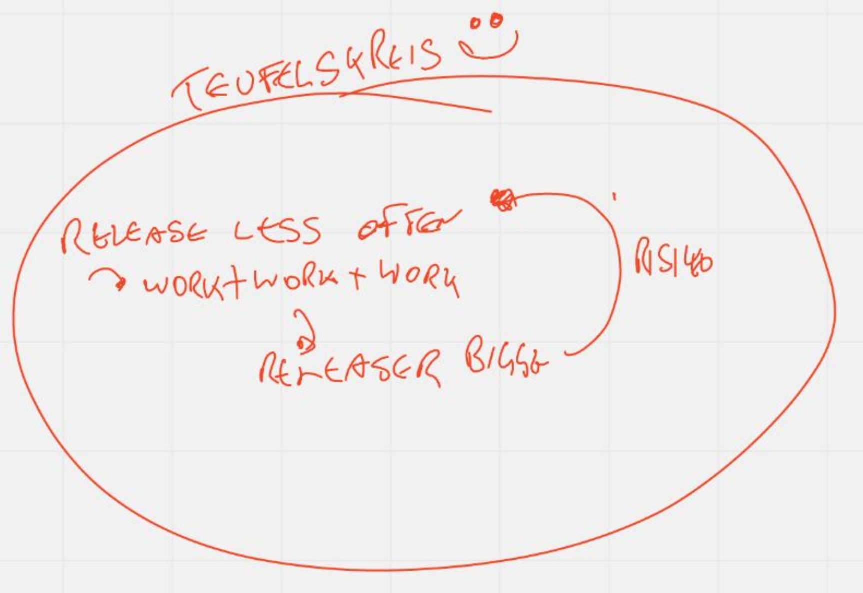


Meine Erfahrung mit Off-Shore Teams ist, dass die Exzellenz der Teams nicht so gut ist, wie von Dir behauptet. Wie sollte man hier die Arbeit zu den Teams bringen?

Was wenn der lila Pfeil in die andere Richtung geht? Also wenn ein Architekturgremium "mehr Power" hat und Druck ausübt, seltener zu releasen, anstatt umgekehrt? Gibt's da Tipps?

“If its hard to it more often”

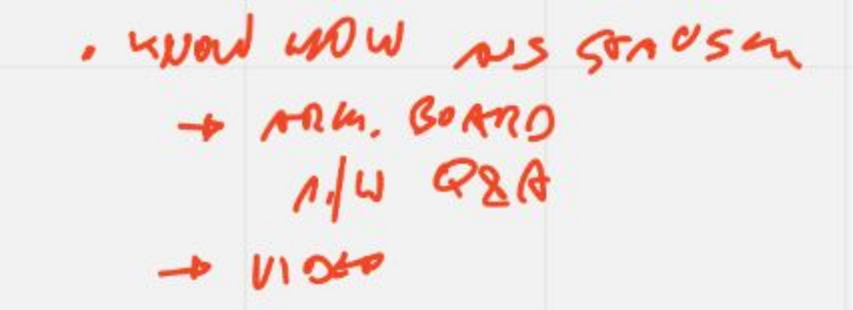
Counter intuitive



Muss man die Teams ausbilden bzgl. Architektur?

Architektur als Skill Fertigkeit → JA

Wie?

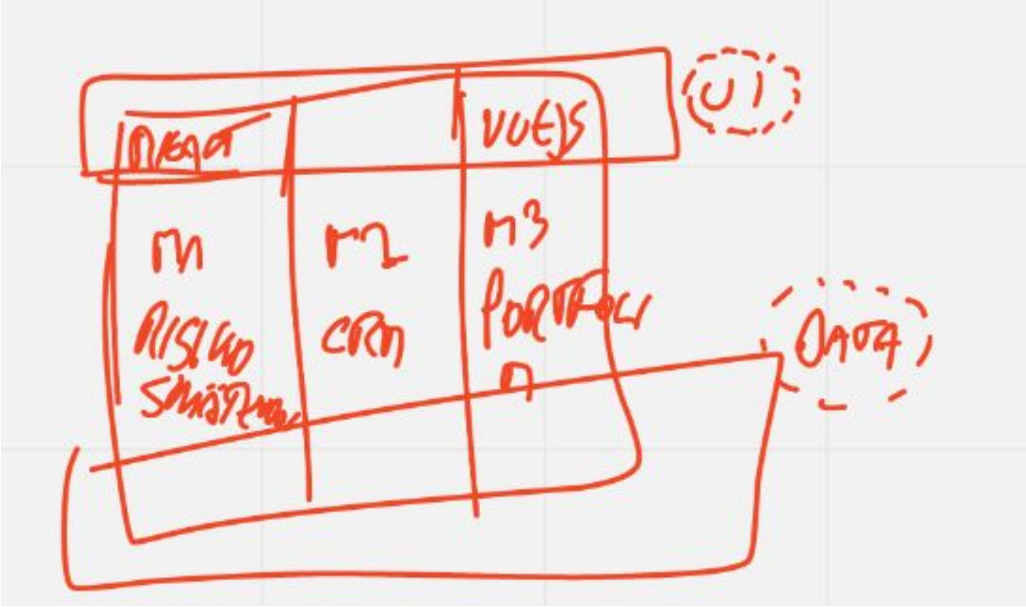


- know how aus streuen
  - → Arch. Board
    m/w Q&A
  - → Video

Wie siehst DU eine Aufgabenaufteilung im Kontext von einem Produkt (Value Stream) aber mit skalierten Scrum-Teams ?

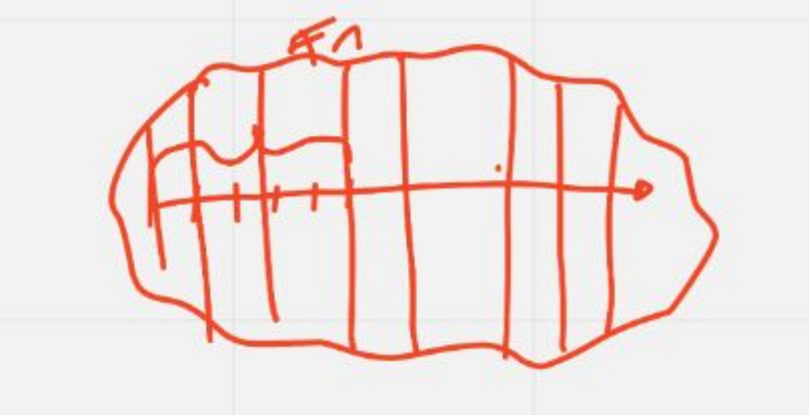
Strive for Autonomy

Mastery + Purpose



Kürzere Zyklen -> Weniger Features implementiert weil Deployment- .Arbeit nicht reduziert aber vervielfacht wird, oder?

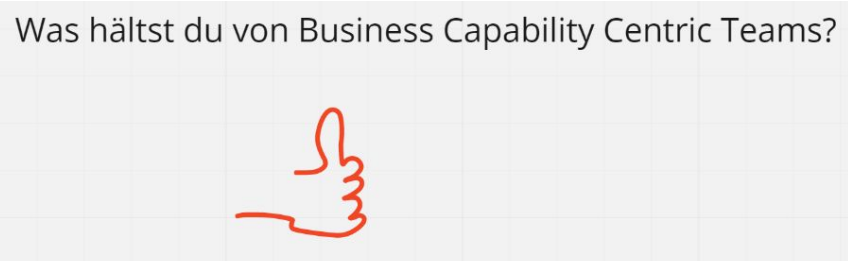
If its hard, do it more often → counter intuitive (weil wir in einem komplexen Umgebung sind? Problem haben?)



Was hältst du von Business Capability Centric Teams?

https://www.martinfowler.com/bliki/BusinessCapabilityCentric.html

Ohne Details zu kennen → Tönt gut



Was hältst du von Business Capability Centric Teams?

What will you do different tomorrow?

Continue the conversation

@peitor

Connect with me

https://twitter.com/peitor

https://www.linkedin.com/in/petergfader/

peter@beyond-agility.com

Google Group

https://beyond-agility.com

Scan me

Peter Gfader