MicroServices

meet Real World projects
MicroServices

meet Real World projects

www.innoQ.com  #gotober #innoQ
Alexander Heusingfeld
Senior Consultant @ innoQ
alexander.heusingfeld@innoq.com
@goldstift
Architecture Consulting...
“We’d like to have a microservice architecture!”
— Customer X
When reviewing a monolithic application ...
...and taking a look into the black box...
...you’ll likely find it consists of multiple Bounded Contexts.
If you cut a monolithic system along its very domains ...
... and wrap every domain in a separate, replaceable web application ...
... then that application can be referred to as a self-contained system (SCS).
more information on self-contained systems (SCS) can be found at

http://scs-architecture.org/

#gotober #innoQ
Architectural Decisions
Architectural Decisions

› Domain Architecture
Architectural Decisions

› Domain Architecture

› Macro Architecture
Architectural Decisions

› Domain Architecture

› Macro Architecture

› Micro Architecture
Isn't there more than that...
At a project meeting...
Did you think about the people who make your architecture exist?
us vs. them
“Don’t care about this, it’s our business!”

“Alarming is our concern, don’t bother about it!”

“No need for a discussion, we always fix that during deployment.”

“That’s part of the handover to operations.”
overcome “us vs. them”
overcome “us vs. them”

› cross-functional != cross-department
overcome “us vs. them”

› cross-functional != cross-department

› have one manager to decide on a team’s targets
overcome “us vs. them”

› cross-functional != cross-department
› have one manager to decide on a team’s targets
› don’t neglect team-building
overcome “us vs. them”

- cross-functional != cross-department
- have one manager to decide on a team’s targets
- don’t neglect team-building
- trust is not optional
well-known pros are subjective
“Operating a monolith is easier!”
Of course it’s easier...
It’s always easier...

...if the complexity is on someone else’s desk.
“Operational costs are increased!”
Monolith
Microservices?
Microservices?

A broken Monolith?!
“Separating teams duplicates work!”
The manageable, domain specific scope enables the development, operation and maintenance of an SCS by an **autonomous team**.
share ideas

share concepts

don’t share functional code
“Deployments cannot be faster, we have an established process!”
Sample of a deployment-pipeline

- **Dev**
  - Git
  - Jenkins
    - Unit test
    - Integration test
    - devtest box
      - Acceptance test devtest-config
      - Acceptance test testing-config
  - push

- **Manual tests**
  - When all tests of build-X are successful, deployment to **shadow-box** in PROD triggered manually after signoff

- **Production**
  - **prod** active box
  - **prod** shadow box
  - application-specific Monitoring & Metrics to detect changes
  - Manual switch of active box triggered by product owner

- **Switch of active box triggered manually by tester**
  - THIS switch triggers the "RC changes" email
Sample of a deployment-pipeline

- **Dev**
  - push
  - Git
  - Jenkins
    - on success -> deploy
    - triggered by Jenkins
    - Unittest
    - Integrationtest

- **Jenkins**
  - on success -> deploy
  - triggered by Jenkins
  - devtest box
  - devtest-config

- **Acceptancetest**
  - on success -> deploy to "testing" shadow box

- **Manual tests**
  - when all tests of build-X are successful, deployment to shadow-box in PROD triggered manually after signoff

- **Production**
  - "prod" active box
  - "prod" shadow box
  - Acceptancetest prod-config

- **Application-specific Monitoring & Metrics** to detect changes

- **Take metrics when running your test suites on ALL environments**

- **Switch of active box triggered by tester**
  - THIS switch triggers the "RC changes" email
What this taught us
What this taught us

› enable fast feedback for your team
What this taught us

› enable fast feedback for your team

› automate what’s next to you first
What this taught us

› enable fast feedback for your team
› automate what’s next to you first
› do your homework before you teach others
What this taught us

› enable fast feedback for your team
› automate what’s next to you first
› do your homework before you teach others
› other people will notice the benefits
What this taught us

› enable fast feedback for your team
› automate what’s next to you first
› do your homework before you teach others
› other people will notice the benefits
› complex processes can be adopted, divide them and take one step at a time
“pets vs. cattle”
summarized: change perspectives!
A company which embraced and evolved
How the customer explained it
How the programmer wrote it
What operations installed
What the customer really needed
Modernisation Strategies
Big Bang
Change via Copy

Diagram: A series of shapes transforming from round to angular forms.
Change via Extraction
Strangulate Bad Parts
more information on software modernisation can be found at

http://aim42.org/

#gotober #innoQ
conclusion
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
Summary

#gotober #innoQ
Summary

› distributed systems are hard - **organizational impact**, too
Summary

- distributed systems are hard - organizational impact, too
- don't forget: there's always at least one other perspective

#gotober #innoQ
Summary

- Distributed systems are hard - organizational impact, too
- Don't forget: there's always at least one other perspective
- Don't overwhelm people, change one thing at a time

#gotober #innoQ
Summary

› distributed systems are hard - **organizational impact**, too
› don't forget: there's always at least one **other perspective**
› **Don’t overwhelm people**, change one thing at a time
› not everyone who **wants** microservices is immediately **capable** to establish them
Thank you!
Questions?
Comments?

https://www.innoq.com/en/timeline/?tag=scs