CLOUD NATIVE TRANSFORMS AGILE & ORGS

Björn Schotte bjoern.schotte@mayflower.de

AGENDA

1. Cloud native Basics
2. Impact on Agile Roles, Tech, Teams, Process, Org
3. Impact on Software Modernization
4. Do we need Scrum anymore?

MAYFLOWER

100 people (Munich, Würzburg, Berlin)

Individual Software Development since 2000

Agile since 2005

Cloud-native, AI, Modernisation

Agile / Tech Consulting

WHAT IF …

your usage of Scrum impedes your product capability?

(or: Scrum isn’t needed anymore!)

CLOUD NATIVE

PRINCIPLES

1. Scalability
2. Resilience (let’s break it!)
3. Manageability
4. Observability (baked into from the beginning)
5. Automation

SCALABILITY

A scalable system makes it easy to add more capacity.

Biggest motivator for moving into Cloud.

Growth factor!

RESILIENCE

A system can be considered resilient if it can continue operating correctly—possibly at a reduced level—rather than failing completely when some part of the system fails.

MANAGEABILITY

Components have a UI

Status delivered via API

Describe environment as code (i.e. via Terraform)

OBSERVABILITY

Observable systems yield meaningful, actionable data.

Faster incident response, increased developer productivity.

AUTOMATION

GitOps / Infra-as-code

CI/CD

Scale up / down automatically

Monitoring & automated recovery

CLOUD NATIVE

MATURITY LEVEL

1. Build: Baseline, PoC
2. Operate: Moving to production
3. Scale: Growing competency
4. Improve: Security + Governance
5. Optimize: Monitor for optimization

BENEFITS OF CN

Building resilient and highly available applications
Respond to market changes ultra-fast

BENEFITS OF AGILE

Iterative development Respond to market changes

MATCH MADE IN HEAVEN?

YES, BUT …

FROM AGILE TO CN

Stage	Agile	Cloud native
Culture	Iterative	Collaborative
Product / Service Design	Feature driven	Data driven
Team	Cross-functional Teams	DevOps / SRE
Process	Agile (Scrum/Kanban)	Design Thinking + Agile + Lean

IMPACTS ON

CULTURE

1. Broad high-level goals
2. Achieve small goals quick
3. Errors are options (not failures)
4. Generate own ideas for Product

IMPACTS ON

PRODUCT / SERVICE DESIGN

1. Ability to release planned features earlier (bc of data driven decisions)
2. Teams decide if feature was an advancement (through monitoring)

IMPACTS ON

TEAM

1. You build it, you run it (CapEx to OpEx!)
2. Teams communicate through APIs (Team Topologies)
3. Social density+ inside team, but not between teams
4. Mandatory capability: Deliver features on your own
5. Release early & often (n times a day to days)
6. DDD & Dynamic Reteaming is your friend

IMPACTS ON

PROCESS

1. Culture of experimentation
2. Continuous delivery of features / releases
3. No / reduction of dependencies (hello SAFe!)
4. Free decisions on tech stack / Teams define their way of working
5. Automation everywhere in the process
6. Flexible processes like IT-Kanban

IMPACTS ON

PRODUCT OWNERS

1. More tech know-how to know *what* can be built
2. Lives a data driven decision culture
3. Needs greater empowerment in order to embrace CN potential

IMPACTS ON

SCRUM MASTERS

1. More tech know-how to know how the humans build IT systems
2. Improve ability of exchange between teams
3. IT is different: More on the tech / functional / data side

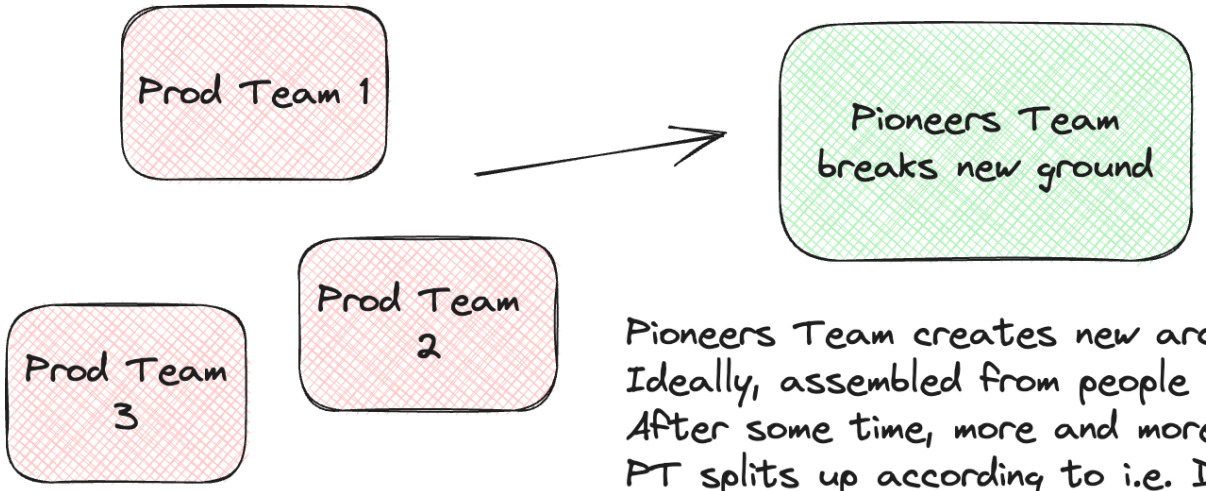
… improves respect inside team

IMPACTS ON

DEVELOPERS

1. Learn new ways of development (Pairing, Ensemble, lightning talk, …)
2. Experimental, playful culture
3. Learning & sharing
4. Evolutional architecture
5. Pioneers team builds new ground, draws new team members in

PIONEERS EVOLUTION



Pioneers Team creates new architecture / software.
Ideally, assembled from people from Team 1-3.
After some time, more and more people join "Pioneers Team".
PT splits up according to i.e. DDD, lot of Pairing / Ensemble involved.

CLOUD NATIVE PATTERNS

https://www.cnpatterns.org/patterns-library

IMPACTS ON

ORGANISATIONS

TRUE end2end team responsibility as **Org KPI**

NO “DevOps” teams - releases done by product teams

Business departments are REAL PART of prod. team

PMO consists of delegates from prod. teams

Give devs **slacktime** for learning new tech

SCRUM: DO WE NEED IT IN CN?

No!

But: Agility + foundations of iterative development + empirical process control

Iterations < 1day

FROM LEGACY TO CLOUD (NATIVE)

Use modernization patterns (i.e. Pioneers Team)

DDD & Team Topologies to the rescue

Platform Teams: Internal dev service teams

Platform-as-a-Product

Be aware of different paradigms (i.e. Desktop2Cloud)

WHAT ABOUT AI?

1 year in AI = 2-3 weeks in reality

WHAT ABOUT AI?

1. Experiment, Experiment, Experiment
2. Big Picture Event Storming to the rescue
3. Identify cost savings and innovative usage
4. Autonomous Agents are key

HOW TO PROGRAM WITH AI?

At the “meta layer” (instructions)

Agents = Task + Tools

CREATE *YOUR* FUTURE WITH CLOUD NATIVE

Let’s talk.

bjoern.schotte@mayflower.de

+49-151-22668191