Tim Transformation Pitfalls And Antipatterns

People, Technology and Process

Antipattern: Getting more and more resources to deliver faster

People treated as resources.
management -> ++developers == ++features.

People != resources

Split 28 team size into 3 teams nearly balanced in skills. We lost best technical people as they moved to team leaders (new antipattern)

Minimise communications

Teach Java 8 and PCF skills

People

Pitfall: Pairing is a Waste of Resources

Stakeholders didn’t accept pairing

Code review

Team was occasionally pairing/mobbing by nature to fix issues or knowledge transferring

People

Pitfall: Absence of OKRs

Every planning meeting, PMs were shouting “We have to commit”!. Target is 20+ MS a month.

Define OKRs (Concept introduced)

Stop releasing in quantity and focus on quality

Introducing modern DevOps ways of working to stakeholders

People

Pitfall: Fixed Mindset

Key stakeholders have a fixed mindset and pathological culture.
Lack of technical skills in engineering.

Show, Not Tell

The smell of success is the best deodorant

Internal marketing

People

Antipattern: Everything is SCDF

The whole project architecture built entirely around SCDF,
every feature is a source, processor and sink.

Rather than centered architecture universe around SCDF, use it for data processing.

Use DDD to define microservices around business capabilities.

Stop using SCDF for user interactions.

Utilise PCF services (use the right tool for the job).

Technology

Pitfall: Absence of Development Environment

There was no development environment sandbox.

Rather than using shared PCF testing environment for testing and debugging, invest in Environment Parity.

Use Docker and PCF dev.

Issues can be replicated in local computer

Technology

Antipattern: Absence of Enterprise Architect Role

Team was involved in an implementation of project, which has no EA. (shifting from Java to nodejs)

EA is required to garden organization, product teams, microservices and platform operators.

EA needs to fully understand the whole system components and their interactions.

Technology

Pitfall: Absence of Automated Tests

There was no unit, integration tests, depends on manual tests.

Rather than depending on manual testing team, adapting TDD.

Integration and smoke automated tests.

QA can still do exploratory tests.

Technology

Pitfall: Lack of 15 factors app

While PCF motives and ease 15 factors app development, these factors were not in team radar.

One repo per microservice

Environment parity

Security

Monitoring

Technology

Pitfall: No Visible Backlog

Backlog was maintained in excel file, delivered through emails, with estimated and assigned tasks.
Features written as title only with no description.

Jira for backlog

IPM

Team to estimate effort

Stories order by business value

Process

Antipattern: Ignoring Security

Credentials (including OpsManager admin) were pushed to git in plain text.

Start installing Vault

Highlighted the risk

Introduced least privilege concept

Process

Pitfall: No CI/CD

Manual deployment through copy/paste commands.
Works on my machine!
Few people know how to deploy to SCDF

Concourse pipelines

Deployment scripts

Process

Lessons learned

- People learn the hard way
- People requires patient and open mindset
- Show, no tell
- Microservices and cloud native requires skills
- Strangle is working
- Language is a real parrier
- Social and outing with team, helps a lot
- Alignment and OKR is a key
- Feedback

Lessons