Microservices in a box

Tobias Vogel @ SEA Away Day 2018

Past engagements

Java / Javascript
monolith web apps
on premise

Java / Javascript
monolith web app
on premise

Scala / Javascript
microservices
web app
AWS

C#
microservices
backend service
on premise

Java/Clojure/JS
microservices web
app/backend services
on premise

JS/React Native
microservices web
app/backend services
AWS

< 2013 → now

Comparison

Monolith

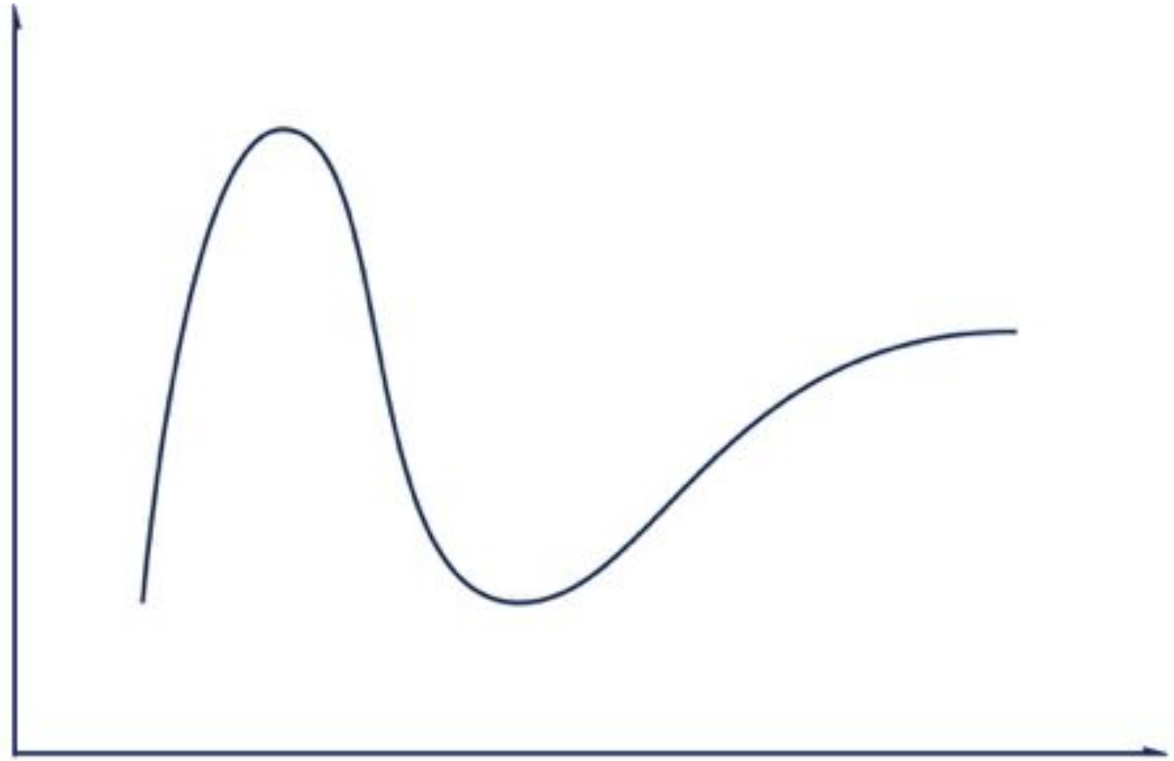
- Too complex to understand
- Hard to change/refactor
- Often outdated frameworks/libraries
- Big ball of mud

Microservices

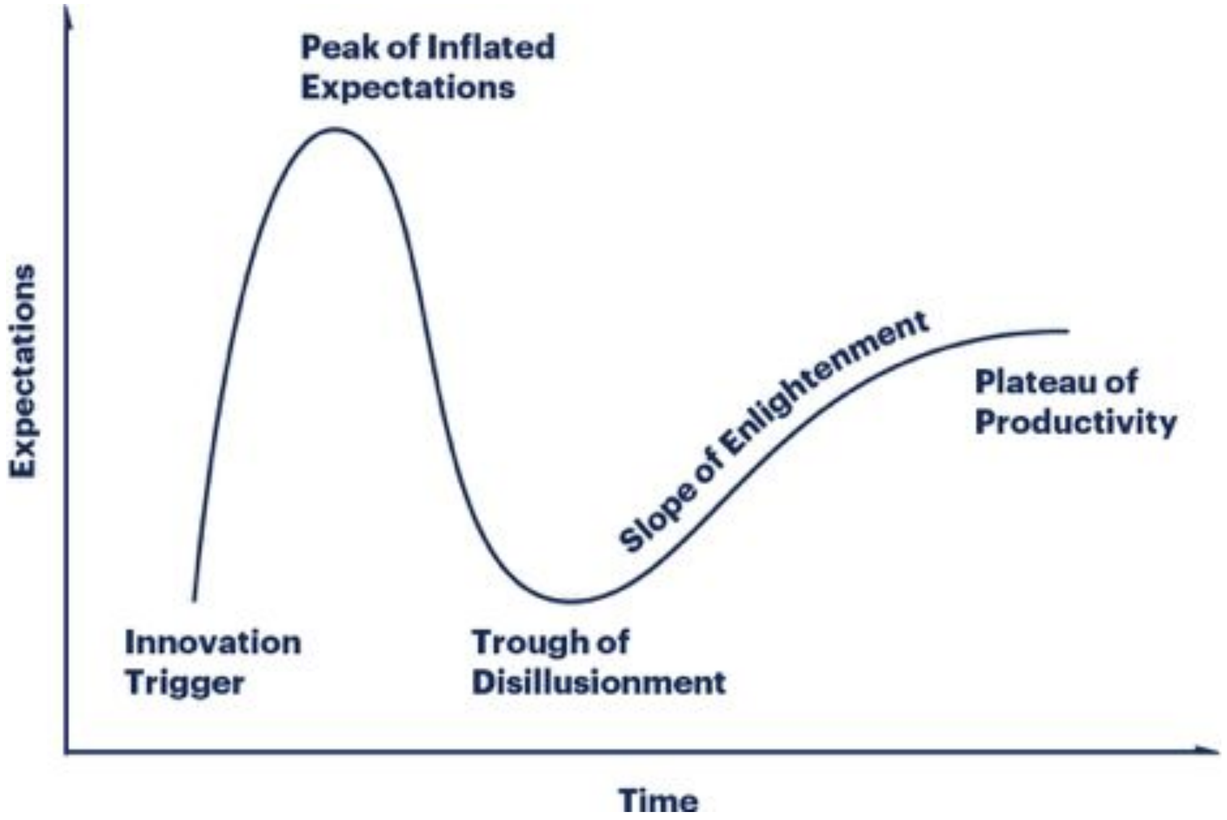
- Clear boundaries (DDD enforced)
- Independent teams and deployment
- Ployglot
- Highly scalable
- Amazon, Facebook, Google/Alphabet, Netflix, Spotify

Why am I here?

…because of this



Gartner Hype Cycle



- Hype Cycle Research Method: https://www.gartner.com/en/research/methodologies/gartner-hype-cycle

Why do I think so?

…not every company is Netflix

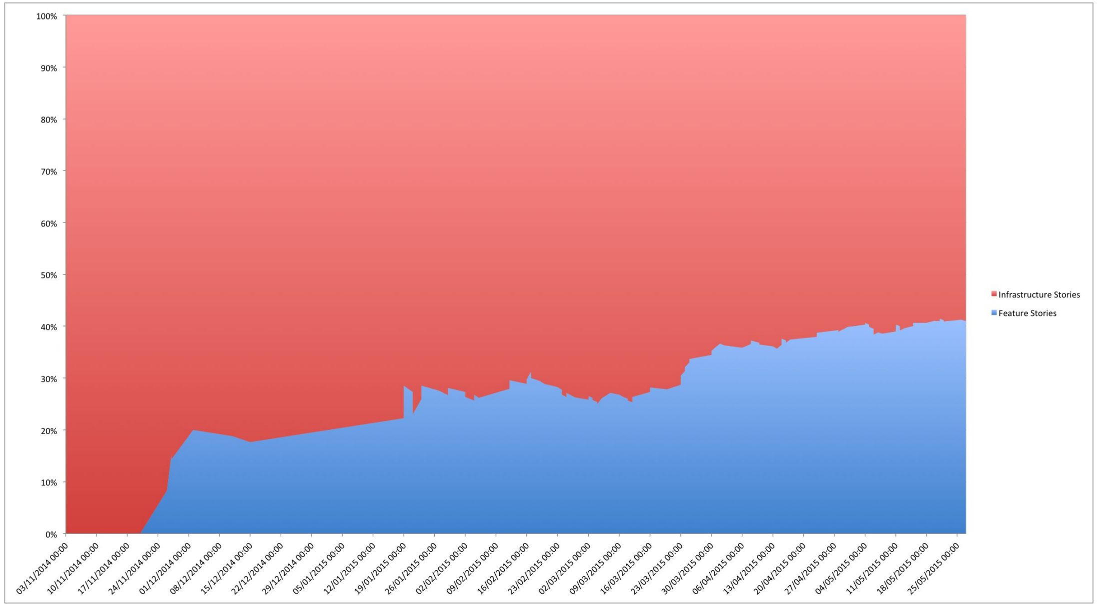
- Microservices Preconditions
- Microservices in Adopt
- The rise of non-microservices architectures
- Monolith first

- Preconditions: https://www.martinfowler.com/bliki/MicroservicePrerequisites.html
- Microservices in Adopt: https://www.thoughtworks.com/insights/blog/microservices-adopt
- Rise of non-MS arch: https://developers.redhat.com/blog/2018/09/10/the-rise-of-non-microservices-architectures/
- Monolith first: https://www.martinfowler.com/bliki/MonolithFirst.html

Some experiences I made

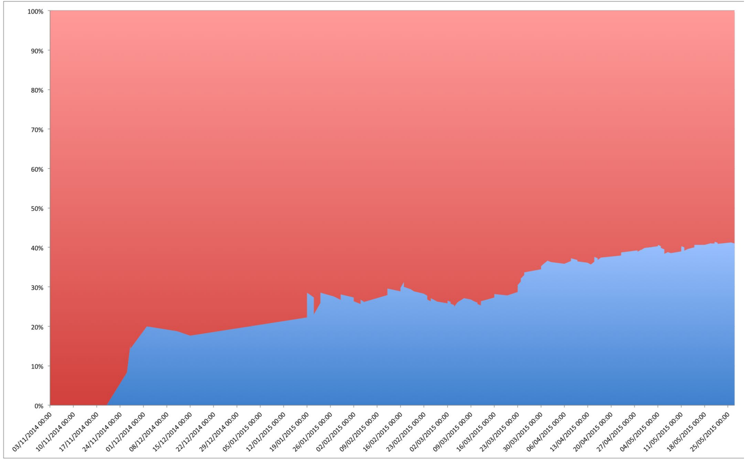
Some experiences

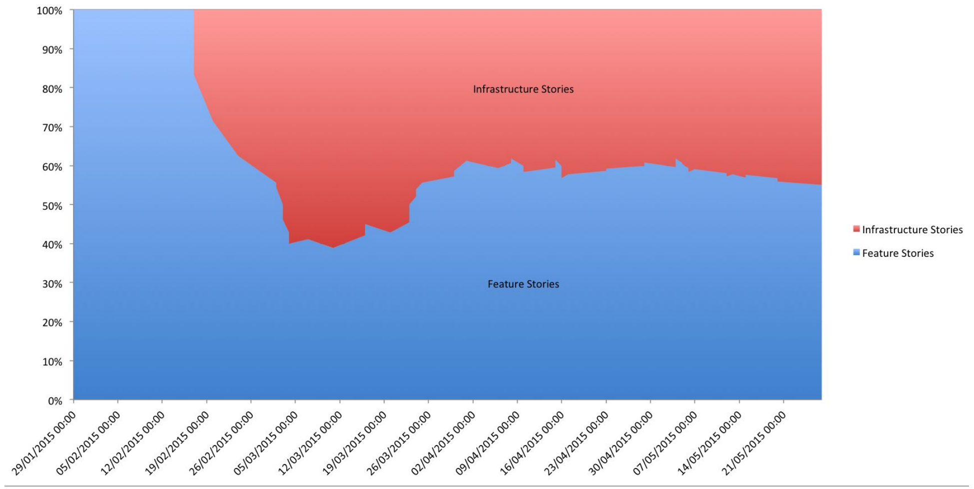
- Initial investment is high



Some experiences

- Initial investment is high





Some experiences

- Initial investment is high
- Complexity gets moved into the integration layer

Some experiences

- Initial investment is high
- Complexity gets moved into the integration layer
- API changes require more effort

Some experiences

- Initial investment is high
- Complexity gets moved into the integration layer
- API changes require more effort
- Consumer Driven Contract tests

Some experiences

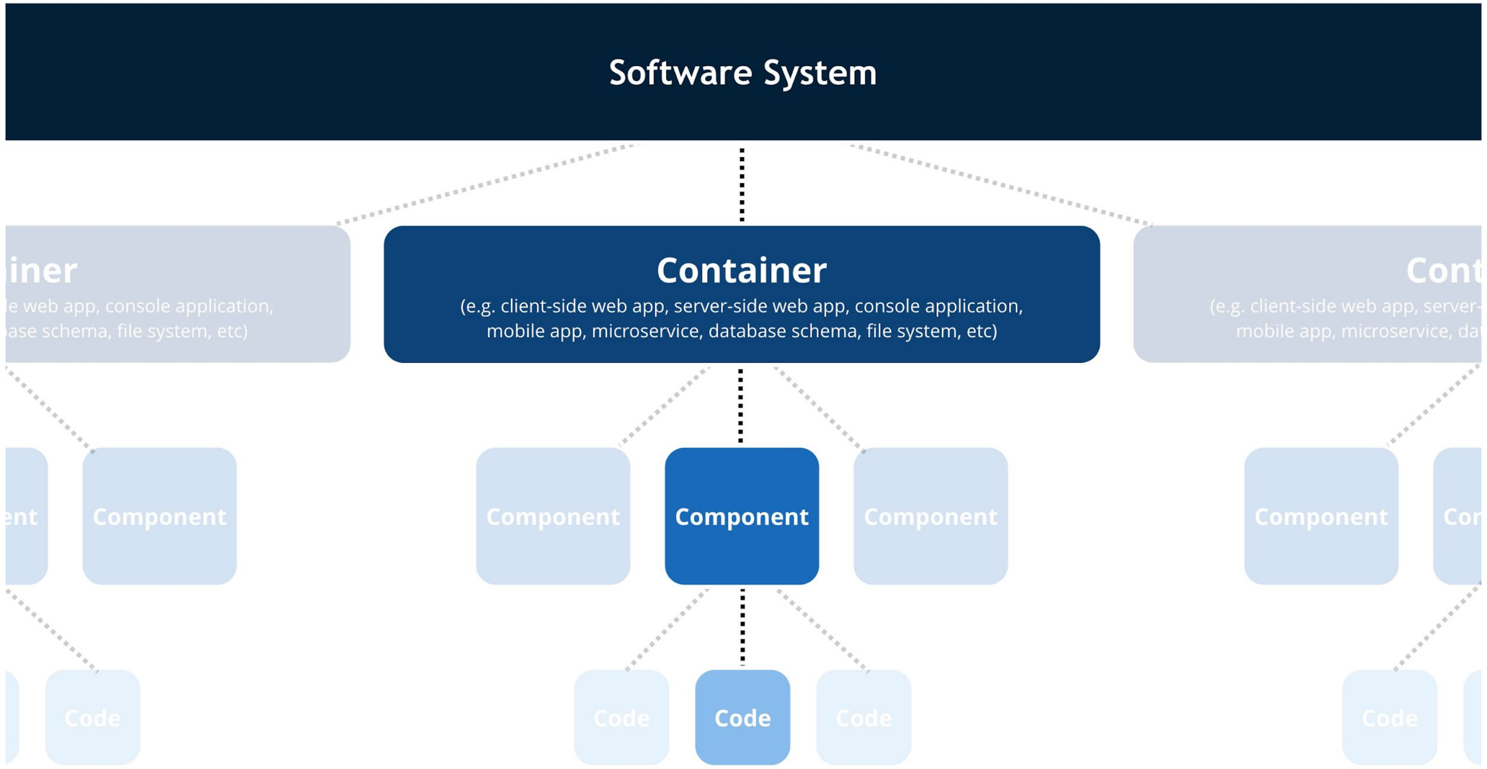
- Initial investment is high
- Complexity gets moved into the integration layer
- API changes require more effort
- Consumer Driven Contract tests
- Debugging microservices

Some experiences

- Initial investment is high
- Complexity gets moved into the integration layer
- API changes require more effort
- Consumer Driven Contract tests
- Debugging microservices
- Scaling for...what?

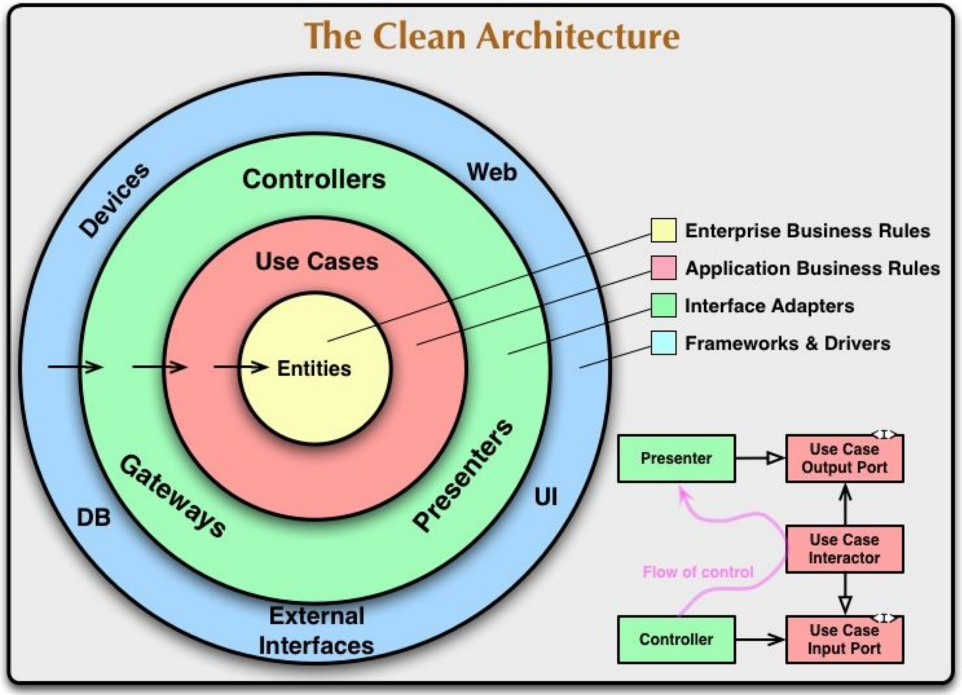
How to structure a monolith?

Follow existing patterns


Software System
iner
e web app, console application,
ase schema, file system, etc)
Container
(e.g. client-side web app, server-side web app, console application,
mobile app, microservice, database schema, file system, etc)
Cont
(e.g. client-side web app, server-
mobile app, microservice, da
ent
Component
Component
Component
Component
Component
Co
Code
Code
Code
Code
Code

- C4 Model: https://c4model.com/

Follow existing patterns



- Clean Architecture: https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

…and apply learnings from microservice applications

- Treat services in a monolith as microservices
- Communication between services in the outer layers
- Enforce service boundaries

- Evolutionary Architecture: https://www.thoughtworks.com/insights/blog/microservices-evolutionary-architecture

Service Communication

Sync

- Like HTTP but without the network
- Caller requires knowledge of destination service (coupling)

Async

- In-Memory Messaging
- Event Producers don’t need to know Consumers
- Domain events live outside of Services
- Event Storming can help you explore your domain

- Event Storming: https://en.wikipedia.org/wiki/Event_storming

How does this look like in code?

- zoo-application: https://github.com/tobivogel/zoo-application

Experiences with it

Pros

- Multiple teams can work independently
- Simpler to change API until “it’s right”
- Service integration tests are straight forward
- Splitting out a new service can be a matter of days

Cons

- Choice of technology is restricted
- Broken builds block multiple teams
- Feature Toggles become more important
- Library changes are not always simple

Our highest priority is to satisfy the customer

through early and continuous delivery

of valuable software.

THANK YOU

For questions or suggestions:

Tobias Vogel

tobias.vogel@ThoughtWorks.com

ThoughtWorks®