Monolith First

IPC Spring Edition 2017

Benjamin Eberlei & Tobias Schlitt (@qafoo)

May 31, 2017

Qafoo

passion for software quality

international PHP2017 conference

- spring edition -

Moin

Benjamin Eberlei
@beberlei

Tobias Schlitt
@tobySen

@qafoo

@tideways

Qafoo
passion for software quality

We Learned the Hard Way

Almost every task is simpler with a Monolith than with Microservices

Qafoo
passion for software quality

The Microservices Cargo Cult

Where things go astray is when people look at, say, Amazon or Google or whoever else might be commanding a fleet of services, and think, hey it works for The Most Successful, I'm sure it'll work for me too. Bzzzzzzzzt!! Wrong!

*(*David Heinemeier Hansson, The Majestic Monolith*)*

passion for software quality

Monolith First, Microservices Second

You shouldn't start a new project with microservices, even if you're sure your application will be big enough to make it worthwhile. *(*Martin Fowler, Monolith First*)*

Outline

passion for software quality

Microservices vs Monolith Prerequisites

Technology

Provisioning

Software

Deployment

Resilience

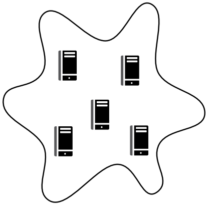
Monitoring

Consistency

Tracing & Debugging

Project Infrastructure

Organization



Software

Software

- Small(er) units of code are easier to understand and test
  - Better separation of work in large teams
  - Mix multiple languages and technologies which get job done
- Bounded Contexts

Deployment

Deployment

- Continuous integration and deployment for all services
- Scalable, robust, fast, centralized
- Common Configuration / 12 factor applications

Provisioning

Provisioning

- Reproducible machines with configuration management
- Infrastructure for simple up/down scaling of individual services
- Simple process to integrate a new service

Monitoring and Tracing

Monitoring and Tracing

- Googles Four Golden Signals
- Centralized logging infrastructure
- Correlation identifier propagation
- Distributed Tracing and profiling across machines

Project Infrastructure

Project Infrastructure

- Contracts between services / teams
- Cross-functional requirements
- Knowledge exchange
- Issue tracker, code repositor(y—ies), . . .

Resilience

Resilience

- Handle failure in distributed systems
  - Retries
  - Circuit Breakers
  - Throttling and Load-Balancing
- Latency between services
- Design for partial outages

Consistency

Consistency

- Data model segregation
- Distributed and potentially duplicated data
- Multi-phase commits
- Eventually (hopefully) consistent

Organization

Organization

Conway's Law:

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

- Crossfunctional teams / Devops
- Microservices were invented in companies with hundrets of developers

Technology

Technology

- Learning a large stack of new technology
  - Containers
  - Cluster and Container Orchestration
  - Service Discovery
  - Monitoring
- Use AWS, Google Cloud, Azure, Heroku...

Microservices Costs

Every topic requires a significant time-investment

Microservices Costs

Your team is probably too small to learn all this at once and still be productive!

Outline

Microservices vs Monolith Prerequisites

From Monolith to Microservices

Modular Design and Isolated Code

- Separate core and supporting domains
- Code in one module must not use code from other modules
- Be careful with frameworks that introduce inter-module dependencies
- DRY considered harmful: Repeat yourself!

Avoid Dependencies through Database(s)

- Isolate database tables and systems from each other
- Be careful with ORMs that introduce inter-module dependencies

RPC-able Interfaces

- Introduce interfaces at module boundaries
- Think of them as RPC Client wrapper
  - Data Transfer Objects as Arguments
  - Data Transfer Objects as Return Values
- Information Hiding Principle: Don’t expose internals

When should we split a module into a microservice?

- Do we need to scale the module independently from the monolith?
- Does the module require constant dev and ops work from a dedicated team?
- Does the module have its own, vastly different release schedule than the monolith?
- Does the module have a different uptime and availability requirement than the monolith?
- Is centralized application level monitoring and logging in place for the module?

Responsible Introduction of Microservices

1. A single Monolith
2. Fully automate CI/monitoring/provisioning
3. Extract module as a microservice
4. Repeat Step 3

The Harsh Reality

If you can’t write a modular monolith, then you will fail at microservices

Resources

- https://martinfowler.com/bliki/MicroservicePrerequisites.html
- https://martinfowler.com/bliki/MonolithFirst.html
- https://m.signalvnoise.com/the-majestic-monolith-29166d022228
- https://aadrake.com/posts/2017-05-20-enough-with-the-microservices.html
- http://www.russmiles.com/essais/8-ways-to-lose-at-microservices-adoption

https://qafoo.com/newsletter

passion for software quality

THANK YOU

Rent a quality expert
qafoo.com