Microservices for Scalability

Keynote at ICPE 2016, Delft, NL

Prof. Dr. Wilhelm (Willi) Hasselbring

Software Engineering Group, Kiel University, Germany
http://se.informatik.uni-kiel.de/
Competence Cluster Software Systems Engineering
http://kosse-sh.de/
Agenda

1. Integrated Information Systems
   – Including its Limits to Scalability
2. Information Systems Integration
   – Including its Anti-Patterns to Scalability
3. Microservice Architectures for Scalability
   – Performance and Elasticity
   – Software Development Scalability
4. Takeaways
Integrated Information Systems?

Why not employing an integrated information system?

Example: ARIS
Architecture of Integrated Information Systems

Source: [Scheer 1994]
Web Information Systems
Cache Architecture

Source: [Abbott & Fisher 2015]

Approaches to Scalability on the database layer:
• Big enterprise server
• Database replication
• Database sharding

However, you have to scale everything to scale anything!
Agenda

1. Integrated Information Systems
   – Including its Limits to Scalability

2. Information Systems Integration
   – Including its Anti-Patterns to Scalability

3. Microservice Architectures for Scalability
   – Performance and Elasticity
   – Software Development Scalability

4. Takeaways
Information Systems Integration?

Source: [Conrad et al. 2005]
Architecture and Integration Layers for Business Information Systems

Source: [Hasselbring 2000]
Integration Dimensions

Source: [Hasselbring 2000]
General System Architecture of Federated Database Systems

Source: [Hasselbring 2015]
Five-level schema architecture for federated database systems

Source: [Sheth & Larson 1990, Hasselbring 2015]

Result: Tight coupling between integrated databases!
Some Anti-Patterns to Scalability of Information Systems

1. One central database
2. Distributed transactions
3. Schema-based integration
4. Limited capacity
5. Shared code

Not meant to be exhaustive, but discussed in this talk.
Agenda

1. Integrated Information Systems
   – Including its Limits to Scalability

2. Information Systems Integration
   – Including its Anti-Patterns to Scalability

3. Microservice Architectures for Scalability
   – Performance and Elasticity
   – Software Development Scalability

4. Takeaways
The Scale Cube

Based on [Abbott & Fisher 2015]
Y-Axis Scaling via Independently Deployable Microservices

Based on [Bas et al. 2015].
Polyglot Persistence

Page Assembly

 Backend Integration

Product

Search

Recommendation

Order

Tracking

Wide Column Database

Full-Text Database

Graph Database

Relational Database

Time Series Database

Wide Column Database: cassandra

Full-Text Database: elastic

Graph Database: neo4j

Relational Database: MySQL

Time Series Database: KairosDB
Verticals for Business Functions
Example: otto.de

Based on [Kraus et al. 2013 Steinacker 2014]
Verticals and Microservices

Based on [Steinacker 2014]
“Scalability is managed by each service individually and is included in its SLA in the form of a guaranteed response time given a particular load.”

[Bas et al. 2015, Chapter 4]

“The trade-off between many small components and a few large components must be considered in component and system design.”

[Hasselbring 2002]
Vertical and Horizontal Scalability

There are two primary approaches to scaling:

• Vertical scaling is also known as scaling up, which means to
  – increase the overall application capacity of individual nodes through hardware improvements, e.g., change to other nodes with higher memory, or increase the number of CPU cores.

• Horizontal scaling is also called scaling out, which means to
  – increase the overall application capacity by adding more nodes, each additional node typically has the equivalent capacity, such as the same amount of memory and the same CPU.

→ Elasticity required
Manage a cluster of containers for horizontal scalability

http://kubernetes.io/
SLAStic: Online Capacity Management

Essential in this Context: Continuous Monitoring

[Fittkau et al. 2013, 2015a]

[van Hoorn et al. 2012]
Monitoring for Online Capacity Management
But also Scalable Monitoring Trace Processing

Processing Capabilities:
✓ Cost efficient
✓ Scalable to millions of monitored methods per second

[Fittkau et al. 2015b]
Adaptive Monitoring: Adjust Instrumentation Coverage at Runtime

Integration of Adaptation and Evolution

Models @ Runtime [Heinrich et al. 2014, 2015]
Agenda

1. Integrated Information Systems
   – Including its Limits to Scalability
2. Information Systems Integration
   – Including its Anti-Patterns to Scalability
3. Microservice Architectures for Scalability
   – Performance and Elasticity
   – Software Development Scalability
4. Takeaways
DevOps & Software Architecture

“The deployment pipeline is the place where the architectural aspects and the process aspects of DevOps intersect.”

[Bas et al. 2015]
Deployment Pipelines for Continuous Deployment
Example Deployment Pipeline @ Otto.de

Source: [Breetzmann et al. 2014]
Automated Quality Assurance
Example: Regression Benchmarking

Integrated into Continuous Integration Setup
[Waller et al. 2015]

Should include automated anomaly detection

https://build.se.informatik.uni-kiel.de/jenkins/job/kieker-nightly-release/plot/
Conway’s Law

“The basic thesis of this article is that organizations which design systems [...] are constrained to produce designs which are copies of the communication structures of these organizations”

[Conway 1968]

If the organizational structure is decomposed vertically and according to the microservices structure into cross-functional feature teams,

• **scaling** development capacities according to changing business requirements is enabled.
• The **feature teams** should be highly independent, having members of all roles and skills that are required to build and maintain their microservice.
→ Decoupling teams as relevant as decoupling software modules
Component vs. Middleware Reuse

Component A

Component B

Shared Component
<use>•

Component A

Component B

Open Source Middleware
<import>•

Example:
https://github.com/otto-de/
From Monoliths towards Microservices

Yesterday, at the ICPE 2016 Doctoral Symposium
- Holger Knoche: “Sustaining Runtime Performance while Incrementally Modernizing Transactional Monolithic Software towards Microservices”
Anti-Patterns and Solutions to Scalability of Information Systems

1. One central database → Polyglott persistence
2. Distributed transactions → Eventual consistency
3. Schema-based integration → Loose coupling via asynchronous messaging
4. Limited capacity → Continuous monitoring for elastic capacity management
5. Shared code → Open source frameworks

Microservices offer such solutions.

Scalability for both, runtime performance and development performance (DevOps).

However, be aware of the imposed costs!
Advertisements

- **Softwareforen Leipzig**, April 12-13, 2016
  Microservice Architectures and Continuous Delivery
  http://www.softwareforen.de/goto/sar

- DevOpsDays Kiel, May 12-13, 2016
  http://www.devopsdays.org/events/2016-kiel/

- KoSSE Day on DevOps, June 1, 2016
  http://kosse-sh.de/

- Symposium on Software Performance
  November 08–09, 2016 in Kiel
  (Descartes/Kieker/Palladio Days 2016)
  http://www.performance-symposium.org/
References