Software Architecture in Industry

Erwann Wernli
Software Architect
erwann.wernli@sbb.ch
Agenda

- Textbook vs. Real-World Software Architecture
- Architecture in Practice: RCS
- Architecture and Trade-offs
Software Architecture refers the fundamental structures of a software system and the discipline of creating such structures and systems.

Each structure comprises software elements, relations among them, and properties of both elements and relations.

https://en.wikipedia.org/wiki/Software_architecture
Software Architecture from Textbooks
Software Architecture in Practice

→ Diagrams are a mix of various concerns
Software Architecture in Practice

→ Diagrams are a mix of various concerns
Software Architecture in Practice

> Diagrams are a mix of various concerns

Functional aspects

Technical aspects
Documenting Software Architecture

As an industry, we do have the Unified Modeling Language (UML), ArchiMate and SysML, but asking whether these provide an effective way to communicate software architecture is often irrelevant because many teams have already thrown them out in favour of much simpler "boxes and lines" diagrams. Abandoning these modelling languages is one thing but, perhaps in the race for agility, many software development teams have lost the ability to communicate visually.

https://c4model.com/
Functional Architecture
Server Processes
NFR

➔ High availability
  - The system must be highly available (99.8% incl. Maintenance windows). It must support Disaster Recovery.

➔ Performance
  - Critical data processing must take no more than 1 second in 90% of the cases.
Technology Stack

Persistenz & Caching
- Coherence
- Hibernate
- Oracle DB

Client
- Eclipse RCP

Messaging
- Tibco Rendezvous

Runtime
- Java SE 7.0
- OSGi / Equinox
Architecture Principles

- Non-reliable communication with Tibco Rendezvous
- Distributed Caching with Coherence
- Leader election for fault-tolerance
Communication: Multicast und Routing

Tibco RV

Typical Bus System

Client Subnet

Server Subnet

RVRD

RVD

TCP/IP

Filtering

Topic

Multicast

TCP/IP

Betriebslage

TIBCO Rendezvous

(*many to many*)

(one to few)
Caching: Coherence

RCS DB Server

Coherence Data Grid

RCS Server

RCS process

Read

Write
High availability: Redundancy

Node 1

Node 2

Node 3

Processes of the application

Cache
High availability: Redundancy

Node 1

Node 2

Node 3

Processes of the application

Cache

✓
Codebase: Layering

- Service
- Function
- Integration
- Access
- Model
Architecture and Trade off
Architecture is about tradeoff

1. There are many ways to decompose a system functionally
2. There are many technologies to choose from
3. There are many NFR that comes into play
Example: RCS

- Availability
- Performance
- Maintainability

Software Architecture

- Services with Shared State, Active/Passive
- Fat Client

Messaging
Caching
Leader Election
NFR

Product NFR
Customer NFR
Decomposition Strategies

- Self-contained services
- Microservices
- Monolith
- Layers
- Pipe and Filter
- Workflow
- Event-Driven
- Actors
- Microfrontend
- Fat Client
- Thin Client
- Backend for Frontend
- RESTful Services
- Orchestration
- Choreography
- Pub/Sub
- RPC
- Event Sourcing
- 3-Tier Architecture
- …
Decomposition Strategies

Why do people advise against stored procedure?
Technologies

And many many more…

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNDF Projects representing a particularly well-traveled path.

github.com/cnclf/landscape
Arc42: Solution Strategy

Contents

A short summary and explanation of the fundamental decisions and solution strategies, that shape the system’s architecture. These include

1. technology decisions

2. decisions about the top-level decomposition of the system, e.g. usage of an architectural pattern or design pattern

3. decisions on how to achieve key quality goals

https://docs.arc42.org/home/
The simple house example also highlights another important property of architectures: rarely is an architecture simply "good" or "bad". Rather, architecture tends to be fit or unfit for purpose.

https://www.enterpriseintegrationpatterns.com/ramblings/86_isthisarchitecture.html