CSE 218: Advanced Topics in Software Engineering
Software Architecture for Distributed and Reactive Systems

Ingolf H. Krueger
ikrueger@ucsd.edu
“Buy, don’t build« is the anthem of the software community today. But buying means less control over every aspect of a system’s development. How can this loss of control be reconciled with our desire for quality? Part of the answer lies in our assertion that, for large systems, quality lives primarily in the architecture.”

[BCK98]
Overview

- The Notion of Architecture
- Architectural Aspects
- Architecture vs. Development Process
- Influence of Architecture on System Properties
The Notion of Architecture

Is Software Architecture the Feng Shui of Software Engineering?
A **Software-Architecture** describes the decomposition of a system into **units / components / subsystems**, and their **connections / interactions / relationships** observing **quality requirements / design guidelines / constraints**

More precisely: interfaces (offered/used)
- Syntax (signals/methods, typing)
- Behavior (protocols)

Often: (arbitrary) distinction between “functional” and “non-functional” properties
Who is involved? Stakeholders!

- Customer
- Marketing
- Developer
- Architect
- End User
- Maintainer
Forces Influencing Architectures

adapted from [H03]
Overview

- The Notion of Architecture
- Architectural Aspects
- Architecture vs. Development Process
- Influence of Architecture on System Properties
Architectural Aspects

Functional requirements
Models of structure and behavior

logical view

implementation view

process view

service view

deployment view

Source code organization
File structure
Configuration information

Aspects of distribution and concurrency
Response times
Throughput

see also: [RUP98]

Mapping of executables to processors
System platform
Installation

...
## Architectural Aspects and Models

<table>
<thead>
<tr>
<th>Model view</th>
<th>Elements Considered</th>
<th>Description Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Data Structures</td>
<td>Entity Relationship-/Class-Diagrams</td>
</tr>
<tr>
<td>Process</td>
<td>Processes, Communication</td>
<td>Dataflow Diagram</td>
</tr>
<tr>
<td>State Transition</td>
<td>Events, States, Control Flow</td>
<td>Statecharts</td>
</tr>
<tr>
<td>Structure</td>
<td>Objects, Modules, Classes, Components</td>
<td>Object-/Class-/Component-Diagrams</td>
</tr>
<tr>
<td>Behavior</td>
<td>Interfaces, Event Traces, Message Histories, Control Flow</td>
<td>Statecharts, Message Sequence Charts</td>
</tr>
</tbody>
</table>

Kinds of Architecture

Application-specific Architecture

- Logical component structure
- Distribution of “business logic”
- Largely independent of technology issues

Application-independent architecture

- Infrastructure
- Libraries
- Tools
- ...
Overview

- The Notion of Architecture
- Architectural Aspects
- Architecture vs. Development Process
- Influence of Architecture on System Properties
Architecture vs. Process

Pitfalls of the “Waterfall” Development Process

- Requirements Analysis
- Design
- Code and Unit Testing
- Subsystem Testing
- System Testing

Recall:
- 55% of all errors are made, but less than 10% are detected during RA
Facts of the trade:


- Requirement deficiencies are the prime sources of project failures.

Denver International Airport (DIA) was scheduled to open on October 31, 1993 with all three of its concourses fully running on a newly developed, complex, automated baggage handling system. It took until February 28, 1995 for DIA to finally open (sixteen months late!).
Architecture vs. Process

Iterative and Incremental Development

Analysis

Specification

Design

Implementation
Architecture vs. Process

Domain Model, Architecture

Business Processes, Use Cases, User Stories, Requirements, Risks

Architecture Document

Implementation
Overview

- The Notion of Architecture
- Architectural Aspects
- Architecture vs. Development Process
- Influence of Architecture on System Properties
Architecture Influences System Properties

see also: [DW98]

reusable

changeable

System

portable

testable

implementable
Architecture Influences System Properties

see also: [DW98]

- functional
- extensible
- user friendly
- performant
- economical
- available
- safe/secure
- dependable
- scalable
Example Application: “Click-to-Dial”

adapted from [PKB+02]
Literature


Literature

[GOF95] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns – Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995


