Architectural Analysis
Curt Clifton
Rose-Hulman Institute of Technology
Today

- Intro. to Architectural Analysis
- Design Studio: Smart Storage Solution
Software architecture: the large-scale motivations, constraints, organization, patterns, responsibilities, and connections of a system.

- Structure and connections
- Components, connectors, and topology
One View of Architectural Analysis

- A specialization of requirements analysis focused on those requirements that strongly influence the large-scale structure and connections of the system
- Typically focused on “ilities”
Why does Architectural Analysis matter?

- Reduce the risk of missing something important
- Avoid applying too much effort to low priority issues
- Align the product with business goals

Just an intro today.
AA is a main focus of CSSE 377
When do we do Architectural Analysis?

- Before the first iteration
- Waterfall?
- Nope, risk management
- Between every iteration
Points of Change

- **Variation point**: points of change in the existing system or requirements
  - E.g., multiple tax calculators

- **Evolution point**: points of change that may arise in the future but aren’t currently present
  - E.g., hand-held POS devices
Goals of Architectural Analysis

- Identify and resolve non-functional requirements
- Identify variation points
- Identify *most probable* evolution points
Example Questions

- How do reliability requirements affect the design?
- How do licensing costs of subcomponents affect profitability?
- How do adaptibility/configurability requirements affect the design?
- How does branding affect the architecture?
Cartoon of the Day

I just thought of an idea for an iPhone app!
Tell me about it!
Well you know how...
RING! RING!
Hello?
Oh hi. I was just---
Really? Darn.
They rejected it.
They're getting really efficient!
Common Steps

- Identify the architectural factors
  - Non-functional requirements that have an impact on the architecture
  - Also functional requirements related to variation/evolution points
- Analyze alternatives and create solutions → architectural decisions
- Document these decisions—technical memos
Architectural Factors

- FURPS+ Requirements:
  - Functional
  - Usability
  - Reliability
  - Performance
  - Supportability
  - +

- Typically the non-functional quality attributes drive the architecture
- Why?
Quality Scenarios

- Measure quality requirements
- Usually of the form `<stimulus> <measurable response>`
  - E.g., “When the completed sale is sent to the remote tax calculator, the result is returned within 2 seconds, measured in a production environment under average load conditions”
### Factor Table in Supplementary Spec.

<table>
<thead>
<tr>
<th>Factor</th>
<th>Measures and quality scenarios</th>
<th>Variability (current flexibility and future evolution)</th>
<th>Impact of factor (and its variability) on stakeholders, architecture and other factors</th>
<th>Priority for Success</th>
<th>Difficulty or Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reliability—Recoverability</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recovery from remote service failure</td>
<td>When a remote service fails, reestablish connectivity with it within 1 minute of its detected re-availability, under normal store load in a production environment.</td>
<td>current flexibility - our SME says local client-side simplified services are acceptable (and desirable) until reconnection is possible. Evolution - within 2 years, some retailers may be willing to pay for full local replication of remote services (such as the tax calculator). Probability? High.</td>
<td>High impact on the large-scale design. Retailers really dislike it when remote services fail, as it prevents them from using a POS to make sales.</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>Recovery from remote product database failure</td>
<td>as above</td>
<td>current flexibility - our SME says local client-side use of cached “most common” product info is acceptable (and desirable) until reconnection is possible. Evolution - within 3 years, client-side mass storage and replication solutions will be cheap and effective, allowing permanent complete replication and thus local usage. Probability? High.</td>
<td>as above</td>
<td>H</td>
<td>M</td>
</tr>
</tbody>
</table>
Technical Memos: Documenting Decisions

- Summarize the issue
- List the relevant architectural factors
- Describe the chosen solution
- Give the motivation for choosing the solution
- Note any unresolved issues
- Identify alternatives considered

Including rationale for rejecting alternatives

Lots of sample memos in the book
Architectural Decisions and Priorities

- First: inflexible constraints, safety and legal compliance
  - E.g., NextGen POS tax law compliance
- Second: business/organization goals
  - E.g., planned expansion into European market
- Third: other goals, including evolution points ordered by estimated probability
Separation of Concerns

- Some large-scale techniques for SOC:
  - Modularize into separate components
    - E.g., persistence service/façade, layered arch.
  - Use decorators
  - Use post-compilers or aspect-oriented techniques
  - Architecture description languages (ADLs)
Summary

- Architectural concerns are related to non-functional requirements, including business/organization goals.
- Architectural concerns involve system-level, large-scale problems. Solutions involve large-scale design decisions.
- Architectural analysis deals in interdependencies and trade-offs.
- Architectural analysis requires evaluation of alternative solutions.
## Design Studio: Smart Storage System

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team describes problem and perhaps current solution (if any)</td>
<td>~5 min.</td>
</tr>
<tr>
<td>Class thinks about questions, alternative approaches. <strong>Q7</strong></td>
<td>~3 min.</td>
</tr>
<tr>
<td>On-board design</td>
<td>~12 min.</td>
</tr>
</tbody>
</table>
