CSCI 3130
Software
Architectures

February 12, 2013
Software Architecture

- Helps with:
  - Communication and Understanding
  - Reuse
  - Construction and Evolution
  - Analysis

- Architecture is described using views for different perspectives
  - Most common view: Component & Connector

- Architecture Styles are “Design Patterns” for Software Architectures
Architecture Integrity

- Why listen to the architect?
  - Architecture imposes constraints
  - Constraints allow to make assumptions in other parts of the system
  - If the constraints are not respected, other parts of the system may no longer be compatible
  - Deviation impacts communication, evolution, reuse, analysis
Example: Word Count

Intended Architecture:

Tokenize → Sort → Count → Word Count
Example: Word Count

Deviating Implementation 1:

- Main
  - Tokenize
  - Sort
  - Count
  - Word Count
Example: Word Count

Deviating Implementation 2:

1. get word
2. compare word
3. increment count

Word Count

Count frequencies
Deployment: Allocation View

- How to allocate system components to resources:
  - Hardware
  - Network infrastructure
  - Schedules

- Performance analysis and tradeoffs

- Reliability (redundancy ↔ performance)

- Costs
Architecture Documentation

- Diagrams are not sufficient documentation
- Documentation needs to satisfy all stakeholders
- Primary goal is to communicate the architecture:
  - Structure and formulate the documentation with that in mind
Architecture Documentation

Sample Outline:

● Context (diagram)
  • How does the system fit into its environment?
  • Who interacts with the system?

● Relevant Views (C&C, module, allocation)
  • Diagram
  • Describe the elements/components in the view in detail
  • Describe the interfaces between elements/components
  • Rationale for the decisions reflected in the architecture
  • Describe behaviour and processes
  • Combine views if suitable (e.g. C&C + allocation)
Architecture Documentation

- Formal languages:
  - Acme
  - Wright
  - UML (good choice for diagrams)
- English works too
- Don't constrain yourself
  - Use whatever gets the point across.
- Don't overload it
  - One cloud is enough, and it does not need to be sparkly.
Architecture Analysis & Evaluation

- Significant impact on qualitative properties:
  - Performance
  - Reliability
  - Modifiability
  - Portability

- More important than decisions at the implementation level:
  - A faster sorting algorithm only makes the chosen architecture faster, but not better.

- Evaluate an architecture w.r.t. individual properties
Architecture Analysis & Evaluation

- Formal simulation models can help:
  - Difficult to capture all the information to have a representative model.
  - Better choice for increasing system complexity (cost ↔ benefit)

- Alternative: Procedural Approach
  - List attributes to be evaluated
  - Assign an experience-based subjective assessment of the quality to each attribute (e.g. letter grades)
ATAM  Architectural Tradeoff Analysis Method

1. Collect Scenarios
   - Use Cases, Error Cases, Exceptional Cases (e.g. high load)
   - Attributes of interest

2. Collect Requirements and Constraints
   - Check SRS for QoS requirements/expectations for each use case / attribute
   - Find quantitative measures

3. Describe architectures that are subject to analysis

4. Analyze attributes w.r.t. Requirements

5. Identify Sensitivity and Tradeoffs
   - Points with most significant impact when changes
   - Impact on other components
ATAM Example