software architecture

Views for building software?
- functional (data flow) – related to organizational groups
- concurrency view – process/threads – interaction via data flow, events, synchronization
- code view – what programmer sees / methods/ subclasses/interfaces/ design patterns / UML
- development/impl view – code as repositories – check in/out
- physical view – hw/sw codesign – cpu as component

Views for building a house?
- carpenter view
- electrician view
- plumber view
- gardener/sprinkler/
- --all hinge on floor plan – tied to physical world

What is software architecture?
definition?
software architecture

- a coherent, justified collection of the system’s earliest set of design decisions

software architecture affects the final system…

(1) early design decision allow or preclude nearly all of the system’s quality attributes
- performance: inter-component execution and communication times
- security: data flow secured by secure kernel / encryption
- reliability: warm/hot restart protocols
- also the implementation itself (e.g. TCP/IP)

(2) affects the organization
- concurrency view may require special team
- code view may required modules best handled by specialized teams
- quality modules characterized by high internal coupling and low external coupling
- Conway’s Law: software should match organizational structure

(3) constrains the implementation
- using .NET precludes J2EE

early architectural decisions should center around forms of the infrastructure:

- how components communicate, pass/share data, initialize, shut down, self test
architectural styles?

• independent processes/ communicating processes
• data flow – pipe and filter
• data centered – repository/blackboard
• virtual machine: interpreter/rule based
• call return/ main and subroutines / object based / layered

• -- can be mixed and matched

the big-Q is…

• How do you know you have chosen the right architecture?
• Are you on the road to success or the road to calamity?

architecture is a bet/wager on the future success of your system

all architectures have..

• components (what kind)
• connectors / interaction mechanisms
• topological layout
• set of constraints on topology
• informal description of costs and benefits of style (e.g. pipes: good for reuse, performance not an issue)

• to be architectural is to be the most abstract depiction of the system that enables reasoning about critical requirements and constrains all subsequent refinements

– -- Mark Klein
### quality attributes

- performance
- reliability
- availability
- security
- modifiability
- portability
- functionality
- variability
- subsetability

### ATAM Steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Present Business drivers</td>
</tr>
<tr>
<td></td>
<td>- key requirements/stakeholders/current needs/how system will meet needs</td>
</tr>
<tr>
<td></td>
<td>- constraints – hw/sw/COTS</td>
</tr>
<tr>
<td></td>
<td>- quality attributes requirements and mapping back to business needs</td>
</tr>
<tr>
<td>2.</td>
<td>Present Architecture</td>
</tr>
<tr>
<td></td>
<td>- details depends on time</td>
</tr>
<tr>
<td></td>
<td>- list OS/hardware/middleware</td>
</tr>
<tr>
<td></td>
<td>- other systems interacted with</td>
</tr>
<tr>
<td></td>
<td>- architectural approaches used to meet needs</td>
</tr>
<tr>
<td>3.</td>
<td>Generate Quality attribute Utility tree</td>
</tr>
<tr>
<td></td>
<td>- based on attributes in step 2</td>
</tr>
<tr>
<td></td>
<td>- forces architect and customer to define requirements precisely</td>
</tr>
<tr>
<td>4.</td>
<td>Evaluate architectural approaches with high priority quality requirements</td>
</tr>
<tr>
<td></td>
<td>- identify</td>
</tr>
<tr>
<td></td>
<td>- risks</td>
</tr>
<tr>
<td></td>
<td>- non-risks</td>
</tr>
<tr>
<td></td>
<td>- tradeoffs</td>
</tr>
<tr>
<td></td>
<td>- utility tree serves as basis to probe architecture</td>
</tr>
</tbody>
</table>

### ATAM

- Architecture Tradeoff Analysis Method
- based on:
  - architectural styles
  - quality attributes
  - software architecture analysis method (SAAM)
utility tree

- start with 5 quality attributes / add or remove dependent of problem
  - performance
  - modifiability
  - availability
  - security

drill down...

- performance
  - data latency
    - minimize storage latency on DB to 20 ms.
    - deliver video in real time
  - transaction throughput
    - maximize average throughput to authentication server

drill down...

- performance
  - data latency
    - minimize storage latency on DB to 20 ms.
    - deliver video in real time
  - transaction throughput
    - maximize average throughput to authentication server

priority by importance to success of system -- degree of difficulty

- performance
  - data latency
    - minimize storage latency on DB to 20 ms.
    - deliver video in real time
  - transaction throughput
    - maximize average throughput to authentication server

H,M is prime candidate for scrutiny
6. Brainstorm Scenarios
   – Three kinds of scenarios
   – Use Case Scenarios
     • how will system be used
   – Growth Scenarios
     • how will system grow in near term
   – Exploratory Scenarios
     • extreme forms of growth/new performance requirements/new infrastructure
     • where will system be stressed