Roadmap of the course

- What is software architecture?
- Designing Software Architecture
  - Requirements: quality attributes or qualities
  - How to achieve requirements: tactics
  - How do tactics lead to architectural styles
  - Case studies on architectural styles, observe the achieved qualities
  - The ADD method

Today: Evaluating an architecture
Architectural evaluation - why

- Architecture tells about system properties
  - Effects of design decisions are predictable => architecture is analyzable
- Architecture drives the software system => economic value
- Good evaluation methods => low-cost risk mitigation
- Architecture evaluation → good to be standard part of every architecture-based development method
Architectural evaluation - when

- Cost-effective: early in lifecycle
  - Easier to correct problems
  - Quality cannot be appended to a system later in lifecycle
  - Choose among alternatives or competing architectures

- Other times in lifecycle
  - Completed architecture: validate it before development
  - Legacy system, inherited system, large software system to be acquired

- Rule for "when"
  - When development teams start to make decisions based on the architecture
  - When the cost of undoing such decisions > evaluation cost
Discovery review

- Very early mini-evaluation
- Establishes and prioritizes problematic requirements
- Fewer stakeholders
  - People with decision power on the requirements
- Output
  - Stronger set of requirements
  - Initial approach to satisfy the requirements
Architectural evaluation - who

- Evaluation team
- Stakeholders
  - Project decision maker
    - Architect
    - Component designers
    - Management
  - Customer, sponsor (not always)
Why should they believe you?

- Evaluators = outsiders
- Stakeholders can be
  - Scared
  - Skeptical
    - They are the experts, evaluators cannot teach them about their system
- What to do
  - Counteract the fear
    - Purpose is architecture improvement, not blame pointing
  - Counteract the skepticism
    - Architectural approaches for analyzing QA do not vary much
    - Fresh eyes, new perspective
Results of Architectural evaluation

- Concretely: a report
- Most importantly: gives information
  - Is the architecture **suitable** for the system for which it was devised?
  - Which of two competing architectures is **most suitable** for the system at hand?
- Architecture is suitable if
  - The system that results from it will meet its quality goals
  - The system can be built using the resources at hand (architecture is **buildable**)

18 November 2013
Architecture suitable with respect to...

- A system is **modifiable** or not wrt a specific kind of change
- A system is **secure** or not wrt a specific kind of threat
- A system is **reliable** or not wrt a specific kind of fault occurrence
- A system **performs** well or not wrt specific performance criteria
- An architecture is **buildable** or not wrt specific time and budget constraints
Architectural evaluation - cost

- Cost = staff time required of the participants
- Approximative cost for AT&T: 70 staff-days
  - 300 full scale architecture reviews for projects requiring minimum 700 staff-days
- ATAM reviews: about 36 staff-days
  - For evaluation team
  - Other stakeholders’ time counts too
- Time included for training the evaluation team!
Architectural evaluation - benefits

1. Financial
2. Forced preparation for the review
3. Captured rationale
4. Early detection of problems
5. Validation of requirements
6. Improved architectures

- Overall: increased quality, controlled cost, decreased budget risk
Architectural evaluation - techniques

- Questioning techniques
  - Rely on thought experiments to check architecture suitability
  - Hypothetical architectures too
  - Scenario-based
    - ATAM
    - CBAM
  - Checklist- or questionnaire-based (for similar architectures)

- Measuring techniques
  - Rely on quantitative measures over existing artifact
  - Architectural metrics
  - Simulations, prototypes
ATAM

**Architecture Tradeoff Analysis Method**
- How well an architecture satisfies particular goals?
- Provides insight into how quality goals interact, how they *trade off*
- Has its origins in
  - SAAM (Software Architecture Analysis Method) from the early 1990s
  - Architectural styles
  - Quality attribute analysis communities
Participants in ATAM

• **The evaluation team.**
  – External to the project whose architecture is being evaluated.
  – Three to five people; a single person may adopt several roles in an ATAM.
  – They need to be recognized as competent, unbiased outsiders.

• **Project decision makers.**
  – These people are empowered to speak for the development project or have the authority to mandate changes to it.
  – They usually include the project manager, and if there is an identifiable customer who is footing the bill for the development, he or she may be present (or represented) as well.
  – The architect is always included – the architect must willingly participate.

• **Architecture stakeholders.**
  – Stakeholders have a vested interest in the architecture performing as advertised.
  – Stakeholders include developers, testers, integrators, maintainers, performance engineers, users, builders of systems interacting with the one under consideration, and, possibly, others.
  – Their job is to articulate the specific quality attribute goals that the architecture should meet.
  – Expect to enlist 12 to 15 stakeholders for the evaluation of a large enterprise-critical architecture.
## ATAM Evaluation Team Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team leader</td>
<td>Sets up the evaluation; coordinates with client, making sure client’s needs are met; establishes evaluation contract; forms evaluation team; sees that final report is produced and delivered (although the writing may be delegated)</td>
</tr>
<tr>
<td>Evaluation leader</td>
<td>Runs evaluation; facilitates elicitation of scenarios; administers scenario selection/prioritization process; facilitates evaluation of scenarios against architecture; facilitates on-site analysis</td>
</tr>
<tr>
<td>Scenario scribe</td>
<td>Writes scenarios on flipchart or whiteboard during scenario elicitation; captures agreed-on wording of each scenario, halting discussion until exact wording is captured</td>
</tr>
<tr>
<td>Proceedings scribe</td>
<td>Captures proceedings in electronic form on laptop or workstation: raw scenarios, issue(s) that motivate each scenario (often lost in the wording of the scenario itself), and resolution of each scenario when applied to architecture(s); also generates a printed list of adopted scenarios for handout to all participants</td>
</tr>
<tr>
<td>Questioner</td>
<td>Raises issues of architectural interest, usually related to the quality attributes in which he or she has expertise</td>
</tr>
</tbody>
</table>
Summary of ATAM

- Preparation (presentations)
  1. Present the ATAM
  2. Present business drivers
  3. Present architecture

- Investigation and analysis (assessing key QAs)
  4. Identify architectural approaches
  5. Generate quality attribute utility tree
  6. Analyze architectural approaches

- Testing (results to date against stakeholders)
  7. Brainstorm and prioritize scenarios
  8. Analyze architectural approaches

- Reporting
  9. Present the results
Step 1: present ATAM

- Evaluation team leader
  - ATAM steps in brief
  - Techniques to be used for elicitation and analysis
    - Utility tree generation
    - Architecture-based elicitation and analysis
    - Scenario brainstorming and prioritization
  - Outputs of evaluation
    - Elicited and prioritized scenarios
    - Questions used to understand and evaluate the architecture
    - Utility tree of QA
    - Discovered risks and non-risks
    - Sensitivity points and tradeoffs
Step 2: Present business drivers

- Describe
  - The system’s most important functions
  - Any relevant technical, managerial, economic, or political constraints
  - The business goals and context as they relate to the project
  - The major stakeholders
  - The architectural drivers (the major quality attribute goals)
Step 3: Present architecture

- By the lead architect (or architecture team)
- What: *(the essence)*
  - technical constraints (operating system, hardware, middleware)
  - other systems with which the system must interact
  - architectural approaches (tactics) used to meet the requirements
  - Driving architectural requirements, measurable quantities associated with these
  - COTS and their integration
  - Most important use case scenarios
  - Most important change scenarios
  - Issues/risks wrt meeting the driving requirements
Step 4: Identify architectural approaches

- Catalog the evident styles and other approaches
  - Based on step 3
  - Serves as the basis for later analysis
- Basically
  - Evaluation team acknowledge what they have got so far
Step 5: Generate quality attribute utility tree

- What: QA goals are articulated in detail via a quality attribute utility tree
- Quality attribute goals are
  - Identified, prioritized, refined
  - Expressed as scenarios
- "Utility" is an expression of the overall goodness of the system
  - Quality attributes form the second level being components of utility
Step 5: Generate quality attribute utility tree cont

- Scenarios are prioritized (X,Y)
  - X denotes how important they are
  - Y denotes how difficult it will be for the architecture to satisfy the scenario
- X, Y ∈ {H(igh), M(edium), L(ow)}
<table>
<thead>
<tr>
<th>Quality Attribute</th>
<th>Attribute Refinement</th>
<th>Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td><strong>Transaction response time</strong></td>
<td>A user updates a patient’s account in response to a change-of-address notification while the system is under peak load, and the transaction completes in less than 0.75 second. (H,M) A user updates a patient’s account in response to a change-of-address notification while the system is under twice the current peak load, and the transaction completes in less than 4 seconds. (L,M)</td>
</tr>
<tr>
<td><strong>Throughput</strong></td>
<td><strong>Generating reports</strong></td>
<td>At peak load, the system is able to complete 150 normalized transactions per second. (M,M) No scenarios suggested.</td>
</tr>
<tr>
<td><strong>Usability</strong></td>
<td><strong>Proficiency training</strong></td>
<td>A new hire with two or more years experience in the business becomes proficient in Nightingale’s core functions in less than 1 week. (M,L) A user in a particular context asks for help, and the system provides help for that context. (H,L)</td>
</tr>
<tr>
<td><strong>Normal operations</strong></td>
<td></td>
<td>A hospital payment officer initiates a payment plan for a patient while interacting with that patient and completes the process without the system introducing delays. (M,M)</td>
</tr>
<tr>
<td><strong>Configurability</strong></td>
<td></td>
<td>A hospital increases the fee for a particular service. The configuration team makes the change in 1 working day; no source code needs to change. (H,L)</td>
</tr>
<tr>
<td><strong>Maintainability</strong></td>
<td></td>
<td>A maintainer encounters search- and response-time deficiencies, fixes the bug, and distributes the bug fix. (H,M) A reporting requirement requires a change to the report-generating metadata. (M,L) The database vendor releases a new version that must be installed in a minimum amount of time. (H,M)</td>
</tr>
<tr>
<td><strong>Extensibility</strong></td>
<td><strong>Adding new product</strong></td>
<td>A product that tracks blood bank donors is created. (M,M)</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td><strong>Confidentiality</strong></td>
<td>A physical therapist is allowed to see the part of a patient’s record dealing with orthopedic treatment, but not other parts nor any financial information. (H,M)</td>
</tr>
<tr>
<td><strong>Integrity</strong></td>
<td></td>
<td>The system resists unauthorized intrusion. (H,M)</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td></td>
<td>The database vendor releases new software, which is hot-swapped into place. (H,L) The system supports 24/7 Web-based account access by patients. (L,L)</td>
</tr>
</tbody>
</table>
Step 6: Analyze architectural approaches

- Examine
  - Highest ranked scenarios (step 5)
  - Architectural styles proposed to realize them (step 4)

- The goal is for the evaluation team to be convinced that the approach is appropriate for meeting the attribute-specific requirements

- Scenario walkthroughs

- Identify and record a set of sensitivity points and tradeoff points, risks and non-risks
  - Sensitivity and tradeoff points are candidate risks
Step 6 outputs

- Architectural approaches relevant to each high-priority utility tree scenario
- Analysis questions associated with each approach
  - Pointed to the QA associated to the scenario
- The architect’s responses to the questions
- Risks, non-risks, sensitivity points, tradeoffs associated with the achievement of one/more QA wrt the QA questions that probed the risk
More on Step 6

- Utility tree shows how to probe the architecture
- Architect responds with the architectural approach that answers this need
- Evaluation team can use the QA-specific questions to probe the approach more deeply
- The QA-specific questions help team to
  - Understand the approach in detail and how it was applied in the instance
  - Look for well-known weaknesses with the approach
  - Look for the approach’s sensitivity and tradeoff points
  - Find interactions and tradeoffs with other approaches
Example of Analysis

<table>
<thead>
<tr>
<th>Scenario: A12</th>
<th>Scenario: Detect and recover from HW failure of main switch.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute(s)</td>
<td>Availability</td>
</tr>
<tr>
<td>Environment</td>
<td>Normal operations</td>
</tr>
<tr>
<td>Stimulus</td>
<td>One of the CPUs fails</td>
</tr>
<tr>
<td>Response</td>
<td>0.999999 availability of switch</td>
</tr>
<tr>
<td>Architectural decisions</td>
<td>Sensitivity  Tradeoff  Risk  Nonrisk</td>
</tr>
<tr>
<td>Backup CPU(s)</td>
<td>S2  R8</td>
</tr>
<tr>
<td>No backup data channel</td>
<td>S3  T3  R9</td>
</tr>
<tr>
<td>Watchdog</td>
<td>S4  N12</td>
</tr>
<tr>
<td>Heartbeat</td>
<td>S5  N13</td>
</tr>
<tr>
<td>Failover routing</td>
<td>S6  N14</td>
</tr>
</tbody>
</table>

**Reasoning**

Ensure no common mode failure by using different hardware and operating system (see Risk 8)

Worst-case rollover is accomplished in 4 seconds as computing state takes that long at worst

Guaranteed to detect failure within 2 seconds based on rates of heartbeat and watchdog

Watchdog is simple and has proved reliable

Availability requirement might be at risk due to lack of backup data channel ... (see Risk 9)

**Architecture diagram**

[Diagram showing the flow of data and components including Primary CPU (OS1), Backup CPU (OS2), Switch CPU (OS1), Heartbeat, and various decision points.]
Step 7: Brainstorm and prioritize scenarios

- Utility tree shows the architect’s view on the quality attributes
- Here the focus is on the other stakeholders’ view on the quality attributes and scenarios based on these
  - Which are the most meaningful and important scenarios wrt users etc.
- Scenario brainstorming works well in larger groups
- The prioritized list of scenarios compared to utility tree scenarios
Step 7: Brainstormed scenarios

- Use case scenarios
  - How stakeholders expect system to be used

- Growth scenarios
  - How the architecture is expected to accommodate growth and change
    - Expected modifications, changes in performance or availability, porting to other platforms, integration with other software

- Exploratory scenarios
  - Extreme forms of growth, how the architecture might be stressed by changes
    - Dramatic new performance or availability requirements, major changes in infrastructure or system mission
Step 7: Prioritizing scenarios

- First stakeholders merge scenarios they believe represent the same behavior / QA concern.
- Then stakeholders vote on scenarios they believe are most important.
  - Vote is public.
  - Evaluation leader sorts the scenarios, detects the sharp drop-off in number of votes, and draws the line there.
Vehicle dispatching system exp

4. Dynamically replan a dispatched mission within 10 minutes. [28]

27. Split the management of a set of vehicles across multiple control sites. [26]

10. Change vendor analysis tools after mission has commenced without restarting system. [23]

12. Retarget a collection of diverse vehicles to handle an emergency situation in less than 10 seconds after commands are issued. [13]

14. Change the data distribution mechanism from CORBA to a new emerging standard with less than six person-months’ effort. [12]
Highly ranked scenarios with QA annotations

<table>
<thead>
<tr>
<th>Scenario</th>
<th>#Votes</th>
<th>Quality Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>28</td>
<td>Performance</td>
</tr>
<tr>
<td>27</td>
<td>26</td>
<td>Performance, Modifiability, Availibility</td>
</tr>
<tr>
<td>10</td>
<td>23</td>
<td>Integrability</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>Performance</td>
</tr>
<tr>
<td>14</td>
<td>12</td>
<td>Modifiability</td>
</tr>
</tbody>
</table>
Step 7: Comparison

- Scenario prioritization compared to utility tree
  - Agreement or disagreement
- Each high-priority brainstormed scenario is placed in appropriate leaf node in utility tree
  - Prior to this, identify the QAs the scenario addresses
- When brainstormed scenario placed in utility tree:
  - Scenario matches and duplicates already existing leaf node
  - Scenario goes onto a new leaf of existing branch
  - Scenario fits in no branch of the tree (QA not previously accounted for)
Utility tree vs Scenario Brainstorming

<table>
<thead>
<tr>
<th></th>
<th>Utility trees</th>
<th>Scenario Brainstorming</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stakeholders</strong></td>
<td>Architects, project leader</td>
<td>All stakeholders</td>
</tr>
<tr>
<td><strong>Typical group size</strong></td>
<td>Evaluators, 2-3 project personnel</td>
<td>Evaluators, 5-10 project related personnel</td>
</tr>
<tr>
<td><strong>Primary goals</strong></td>
<td>Elicit, make concrete and prioritize the driving QA; provide focus for the rest of the evaluation</td>
<td>Foster stakeholder communication to validate QA goals elicited via utility tree</td>
</tr>
<tr>
<td><strong>Approach</strong></td>
<td>General to specific; begin with QA, refine until scenarios emerge</td>
<td>Specific to general; begin with scenarios, then identify QAs they express</td>
</tr>
</tbody>
</table>
Stakeholders and Architect

- Architect should be present at evaluation!
  - Cannot identify architect: trouble
  - Architect sees all stakeholders together, important step
  - Architect takes items to work on from evaluation
  - Has to present the architecture
- Identify *kinds* of stakeholders and names!
  - Evaluators suggest kinds
  - Client provides names
Step 8: Analyze architectural approaches

- Highest ranked scenarios from step 7 are presented to the architect
  - Architect explains how relevant architectural decisions contribute to realizing each one
  - Previously discussed architectural approaches should come in here
- Same activities as in Step 6
- Ideally: just testing
  - If step 7 did not produce any high-priority scenarios not already covered
Step 9: Present results

- Verbal report and slides
- Written report

Outputs:
- The architectural approaches documented
- The set of scenarios and their prioritization from the brainstorming
- The utility tree
- The risks discovered
- The non-risks documented
- The sensitivity points and tradeoff points found
Step 9: Risk themes

- Risks can be grouped together based on some common underlying concern, systemic deficiency
- Exp
  - Documentation given insufficient consideration
    - Inadequate doc
    - Out-of-date doc
  - Insufficient attention to backup capability or provision of high availability
    - System cannot function when various SW/HW fails
- Risk theme <-> affected business drivers
  - Closure and risks pointed out to management
Outputs of the ATAM

- A concise presentation of the architecture
- Articulation of the business goals
- Quality requirements in terms of a collection of scenarios
- Mapping of architectural decisions to quality requirements
- A set of identified sensitivity and tradeoff points
- A set of risks and non-risks
- A set of risk themes
Other outputs

- Secondary outputs
  - Architecture representation survives evaluation
  - Scenarios too
  - Analysis can serve as statement of rationale for architectural decisions
    - Made or not
  - Mitigation strategies
- Intangible goals
  - Social, community sense
  - Better communication
  - Improved understanding
# Phases of the ATAM

<table>
<thead>
<tr>
<th>Phase</th>
<th>Activity</th>
<th>Participants</th>
<th>Typical duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Partnership and preparation: Logistics, planning, stakeholder recruitment, team formation</td>
<td>Evaluation team leadership and key project decision-makers</td>
<td>Proceeds informally as required, perhaps over a few weeks</td>
</tr>
<tr>
<td>1</td>
<td>Evaluation: Steps 1-6</td>
<td>Evaluation team and project decision-makers</td>
<td>1-2 days followed by a hiatus of 2-3 weeks</td>
</tr>
<tr>
<td>2</td>
<td>Evaluation: Steps 7-9</td>
<td>Evaluation team, project decision makers, stakeholders</td>
<td>2 days</td>
</tr>
<tr>
<td>3</td>
<td>Follow-up: Report generation and delivery, process improvement</td>
<td>Evaluation team and evaluation client</td>
<td>1 week</td>
</tr>
</tbody>
</table>
Lightweight Architectural Evaluation

- ATAM is a substantial undertaking
  - 20 to 30 person-days of effort from an evaluation team, plus even more for the architect and stakeholders
  - makes sense on a large and costly project, where the risks of making a major mistake in the architecture are unacceptable.
- Lightweight Architecture Evaluation method, based on the ATAM, for smaller, less risky projects
  - May take place in a single day, or even a half-day meeting.
  - May be carried out entirely by members internal to the organization.
  - Of course this lower level of scrutiny and objectivity may not probe the architecture as deeply.
- Because the participants are all internal to the organization and fewer in number than for the ATAM, giving everyone their say and achieving a shared understanding takes much less time.
- The steps and phases of a Lightweight Architecture Evaluation can be carried out more quickly.
## Typical Agenda: 4-6 Hours

<table>
<thead>
<tr>
<th>Step</th>
<th>Time</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Present the ATAM</td>
<td>0 hours</td>
<td>Participants already familiar with process.</td>
</tr>
<tr>
<td>2. Present business drivers</td>
<td>0.25 hours</td>
<td>The participants are expected to understand the system and its business goals and their priorities. A brief review ensures that these are fresh in everyone’s mind and that there are no surprises.</td>
</tr>
<tr>
<td>3. Present architecture</td>
<td>0.5 hours</td>
<td>All participants are expected to be familiar with the system. A brief overview of the architecture, using at least module and C&amp;C views, is presented. 1-2 scenarios are traced through these views.</td>
</tr>
<tr>
<td>4. Identify architectural approaches</td>
<td>0.25 hours</td>
<td>The architecture approaches for specific quality attribute concerns are identified by the architect. This may be done as a portion of step 3.</td>
</tr>
<tr>
<td>5. Generate QA utility tree</td>
<td>0.5-1.5 hours</td>
<td>Scenarios might exist: part of previous evaluations, part of design, part of requirements elicitation. Put these in a tree. Or, a utility tree may already exist.</td>
</tr>
<tr>
<td>6. Analyze architectural approaches</td>
<td>2-3 hours</td>
<td>This step—mapping the highly ranked scenarios onto the architecture—consumes the bulk of the time and can be expanded or contracted as needed.</td>
</tr>
<tr>
<td>7. Brainstorm scenarios</td>
<td>0 hours</td>
<td>This step can be omitted as the assembled (internal) stakeholders are expected to contribute scenarios expressing their concerns in step 5.</td>
</tr>
<tr>
<td>8. Analyze architectural approaches</td>
<td>0 hours</td>
<td>This step is also omitted, since all analysis is done in step 6.</td>
</tr>
<tr>
<td>9. Present results</td>
<td>0.5 hours</td>
<td>At the end of an evaluation, the team reviews the existing and newly discovered risks, nonrisks, sensitivities, and tradeoffs and discusses whether any new risk themes have arisen.</td>
</tr>
</tbody>
</table>
Other methods

- CBAM
  - Cost-Based Analysis Method
  - Uses ATAM

- Measuring technics
  - Answer specific questions about specific qualities
  - Need the presence of a design/implementation artifact (the object to measure)
  - RMA – rate monotonic analysis: quantitative technique for ensuring that a set of fixed-priority processes can be scheduled on a CPU
    - Can be performed as architecture is being evolved
  - ADL, formal notations and languages
CBAM

- Biggest trade-offs influence economics
  - Resources
- Earlier: costs
  - Of building system, not long term
- Now also: benefits
- Economic models needed
  - Consider costs, benefits, risks, schedule implications
- Basic idea of CBAM
  - Architectural strategies $\rightarrow$ quality attributes $\rightarrow$ benefits for stakeholders (utilities)
CBAM Utilities

- Architectural strategy
  - Provides specific level of utility to stakeholders
  - Has cost
  - Takes time to implement
- Return On Investment (ROI)
  - Ratio of benefit to cost
- Utility-response curves
  - Depicts how the utility derived from a particular response varies as the response varies
  - Best-case, worst-case, current-case, desired-case response
    - interpolation
  - Side effects
Some formulas

- Overall utility of architectural strategy across scenarios
  - Strategy \( i \)
  - Scenario \( j \)
  - Benefit \( B_i \)
  - Benefit \( b_{i,j} \)
  - Weight \( W_j \)
  - Utility \( U \)
  - Return over investment \( R_i \), cost \( C_i \)
  - \( B_i = \sum_j (b_{i,j} \times W_j) \)
  - \( b_{i,j} = U_{\text{expected}} - U_{\text{current}} \)
  - \( R_i = \frac{B_i}{C_i} \)
Properties of successful evaluation

- Clear goals and requirements for architecture
- Controlled scope
- Cost-effectiveness
- Key personnel availability
- Competent evaluation team
- Managed expectations
- Final report