Analyzing Architectures

• Why
• When
• Cost
• Benefits
  1. Financial
  2. Forced preparation for the review
  3. Captured rationale
  4. Early detection of problems with the existing architecture
  5. Validation of requirements
  6. Improved architectures
Analyzing Architectures
cont.

• Techniques
• Planned or Unplanned
• Preconditions
  1. Clearly articulated goals and requirements for the architecture
  2. Controlled scope
  3. Cost-effectiveness
  4. Key personnel availability
  5. Competent evaluation team
  6. Managed expectations

• Results
The ATAM:
A Comprehensive Method for Architecture Evaluation

• **Architecture Tradeoff Analysis Method**
• A thorough and comprehensive way to evaluate a software architecture

• **This is hard:**
  – Large systems are very complicated
  – Evaluation needs to compare business goals and technical decisions
  – Careful management of an evaluation process is needed to consider multiple stakeholders in a review format
Participants in the ATAM

• Three basic groups:
  – The evaluation team – 4-5 people
  – Project decision makers – PM, customer, architect
  – Architecture stakeholders – recommend 12-15
Outputs

• A concise presentation of the architecture
• Defined business goals
• Quality requirements via a collection of scenarios
• Mapping of architectural decisions to quality requirements
• A set of risks and non-risks
• A set of risk themes
<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</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</td>
<td>Evaluation team and project decision makers</td>
<td>1 day followed by a hiatus of 2 to 3 weeks</td>
</tr>
<tr>
<td>2</td>
<td>Evaluation (continued)</td>
<td>Evaluation team, project decision makers, and stakeholders</td>
<td>2 days</td>
</tr>
<tr>
<td>3</td>
<td>Follow-up</td>
<td>Evaluation team and evaluation client</td>
<td>1 week</td>
</tr>
</tbody>
</table>
Phases

- **Phase 0**: partnership and preparation

- **Phase 1 and 2**: evaluation
  - Phase 1 has 6 steps
  - Phase 2 has 3 steps with all stakeholders

- **Phase 3**: follow-up via written report
Phase 1

1. Present the ATAM process
2. Present the business drivers
3. Present the architecture
4. Identify architecture approaches
5. Generate quality attribute tree
   - Table 11.5
6. Analyze the architectural approaches
Step 1: Present the ATAM

- The first step calls for the evaluation leader to present the ATAM to the assembled project representatives.

- This time is used to explain the process that everyone will be following, to answer questions, and to set the context and expectations for the remainder of the activities.

- Using a standard presentation, the leader will describe the ATAM steps in brief and the outputs of the evaluation.
Step 2: Present Business Drivers

The presentation should describe the following:

- 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 (that is, the major quality attribute goals that shape the architecture)
Step 3: Present Architecture

**Example of a template for Architecture Presentation (~20 slides; 60 minutes)**

- Driving architectural requirements (2–3 slides)
- Important Architectural Information (4–8 slides)
- Architectural approaches, patterns, or tactics employed (3–6 slides)
- Use of commercial off-the-shelf (COTS) products and how they are chosen/integrated (1–2 slides)
- Trace of 1 to 3 of the most important use case scenarios (1–3 slides)
- Trace of 1 to 3 of the most important change scenarios (1–3 slides)
- Architectural issues/risks with respect to meeting the driving architectural requirements (2–3 slides)
- Glossary (1 slide)
Step 4: Identify Architectural Approaches

- The ATAM focuses on analyzing an architecture by understanding its architectural approaches. Architectural patterns are useful for (among other reasons) the known ways in which each one affects particular quality attributes. A layered pattern tends to bring portability to a system, possibly at the expense of performance. A data repository pattern is usually scalable in the number of producers and consumers of data. And so forth.

- By now, the evaluation team will have a good idea of what patterns and approaches the architect used in designing the system. During that step, the architect is asked to explicitly name the patterns and approaches used, but the team should also be adept at spotting ones not mentioned.

- In this short step, the evaluation team simply catalogs the patterns and approaches that are evident. The list is publicly captured by the scribe for all to see and will serve as the basis for later analysis.
Step 5—Generate Quality Attribute Utility Tree

- Here, the evaluation team works with the project decision makers to identify, prioritize, and refine the system's most important quality attribute goals, which are expressed as scenarios.

- A utility tree begins with utility as the root node. Utility is an expression of the overall "goodness" of the system. Quality attributes form the second level because these are the components of utility.

- The output of utility tree generation is a prioritized list of scenarios that serves as a plan for the remainder of the ATAM evaluation. It tells the ATAM team where to spend its (relatively limited) time and, in particular, where to probe for architectural approaches and risks. The utility tree guides the evaluators toward the architectural approaches for satisfying the high-priority scenarios at its leaves.
# Example: QA Utility Tree

<table>
<thead>
<tr>
<th>Quality Attribute</th>
<th>Attribute Refinement</th>
<th>Scenarios (Importance, Difficulty)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Throughput</td>
<td>At peak load, the system is able to complete 150 normalized transactions per second. (M,M)</td>
</tr>
<tr>
<td>Usability</td>
<td>Proficiency training</td>
<td>A user in a particular context asks for help, and the system provides help for that context. (H,L)</td>
</tr>
<tr>
<td>Availability</td>
<td></td>
<td>The system supports 7/24 Web-based account access by patients. (L,L)</td>
</tr>
</tbody>
</table>
Step 6: Analyze Architectural Approaches

Here the evaluation team examines the highest-ranked scenarios one at a time; the architect is asked to explain how the architecture supports each one.

For well-known approaches, the team asks how the architect overcame known weaknesses in the approach or how the architect gained assurance that the approach sufficed.

For example, the number of simultaneous database clients will affect the number of transactions that a database can process per second. Thus, the assignment of clients to the server is a sensitivity point with respect to the response as measured in transactions per second.

Some assignments will result in unacceptable values of this response—these are risks. When it turns out that an architectural decision is a sensitivity point for more than one attribute, it is designated as a tradeoff point.
Example of architectural approach 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>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Architectural decisions</th>
<th>Sensitivity</th>
<th>Tradeoff</th>
<th>Risk</th>
<th>Nonrisk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup CPU(s)</td>
<td>S2</td>
<td></td>
<td>R8</td>
<td>N12</td>
</tr>
<tr>
<td>No backup data channel</td>
<td>S3</td>
<td>T3</td>
<td>R9</td>
<td>N13</td>
</tr>
<tr>
<td>Watchdog</td>
<td>S4</td>
<td></td>
<td></td>
<td>N14</td>
</tr>
<tr>
<td>Heartbeat</td>
<td>S5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failover routing</td>
<td>S6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures no common mode failure by using different hardware and operating system (see Risk 8)</td>
</tr>
<tr>
<td>Worst-case rollover is accomplished in 4 seconds as computing state takes that long at worst</td>
</tr>
<tr>
<td>Guaranteed to detect failure within 2 seconds based on rates of heartbeat and watchdog</td>
</tr>
<tr>
<td>Watchdog is simple and has proved reliable</td>
</tr>
<tr>
<td>Availability requirement might be at risk due to lack of backup data channel ... (see Risk 9)</td>
</tr>
</tbody>
</table>

Architecture diagram:

```
Primary CPU (OS1)

heartbeat (1 sec.)

Backup CPU with Watchdog (OS2)

Switch CPU (OS1)
```
Phase 2

7. Brainstorm and prioritize scenarios
   • Table 11.6
8. Analyze the architectural approaches
9. Present results
Step 7—Brainstorm and Prioritize Scenarios

- While utility tree generation is used primarily to understand how the architect perceived and handled quality attribute architectural drivers, the purpose of scenario brainstorming is to take the pulse of the larger stakeholder community.

- The prioritized list of brainstormed scenarios is compared with those from the utility tree exercise. In this step, the evaluation team asks the stakeholders to brainstorm scenarios that are operationally meaningful with respect to the stakeholders' individual roles.

- Each stakeholder casts his or her votes publicly; our experience tells us it is more fun that way and builds unity among the participants. Once the votes are tallied, the evaluation leader orders the scenarios by vote total and looks for a sharp drop-off in the number of votes.
## Example: Brainstormed Scenarios

<table>
<thead>
<tr>
<th>Number</th>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Previously public data is made private, and access is adjusted accordingly.</td>
</tr>
<tr>
<td>6</td>
<td>Decide to support German.</td>
</tr>
<tr>
<td>9</td>
<td>A user requests a new field for asynchronous queries.</td>
</tr>
<tr>
<td>21</td>
<td>Introduce a new workflow process for patient check-in and check-out.</td>
</tr>
<tr>
<td>29</td>
<td>A fire in the data center forces the information hub to be moved to a new location.</td>
</tr>
<tr>
<td>33</td>
<td>The phone company changes an area code.</td>
</tr>
</tbody>
</table>
Step 8—Analyze Architectural Approaches

• After the scenarios have been collected and prioritized, the evaluation team guides the architect in the process of carrying out the highest ranked scenarios from step 7. The architect explains how relevant architectural decisions contribute to realizing each one. Ideally this activity will be dominated by the architect's explanation of scenarios in terms of previously discussed architectural approaches.

• In this step the evaluation team performs the same activities as in step 6, mapping the highest-ranked, newly generated scenarios onto the architectural artifacts uncovered thus far.
Step 9: Present Results

The following outputs are presented:

• The architectural approaches documented
• The set of scenarios and their prioritization from the brainstorming
• The utility tree
• The risks discovered
• The nonrisks documented
• The sensitivity points and tradeoff points found
ATAM Summary

• Not an evaluation of requirements
• Not a code evaluation
• Does not include actual system testing
• Not precise, but identifies possible risk areas within the architecture
• Actual practice: amazement that so many risks can be found in such a short time.
The Nightingale System:  
A Case Study in Applying the ATAM

PHASE 0: PARTNERSHIP AND PREPARATION

The client or organization for the evaluation, which had approached us after reading about the ATAM on our Web site, was a major producer of health care systems software, aimed at the hospital, clinic, and HMO markets. The system under consideration was called Nightingale. We learned that it was a large system expected to comprise several million lines of code and that it was well into implementation. Nightingale already had its first customer, a hospital chain with forty-some hospitals throughout the southwestern United States.
## Evaluation Team Role Assignments

<table>
<thead>
<tr>
<th>Member</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Team leader, evaluation leader, questioner</td>
</tr>
<tr>
<td>2</td>
<td>Evaluation leader, questioner</td>
</tr>
<tr>
<td>3</td>
<td>Timekeeper, questioner</td>
</tr>
<tr>
<td>4</td>
<td>Scenario scribe, questioner, data gatherer</td>
</tr>
<tr>
<td>5</td>
<td>Questioner, process enforcer</td>
</tr>
<tr>
<td>6</td>
<td>Proceedings scribe, process observer</td>
</tr>
</tbody>
</table>
PHASE 1: EVALUATION

• Step 1: Present ATAM

• Step 2: Present Business Drivers
  • support for their kickoff customer's diverse uses (e.g., treatment tracking, payment histories, trend spotting, etc.).
  • creation of a new version of the system (e.g., to manage doctors' offices) that the development organization could market to customers other than the kickoff customer.
The following quality attributes were identified as high priority:

- **Performance.** Health care systems require quick response times to be considered useful. The 5-second transaction response time of the legacy system was too slow, as were the legacy response times for online queries and reports.

- **Usability.** There was a high turnover of users of the system, so retraining was an important customer issue.

- **Maintainability.** The system had to be maintainable, configurable, and extensible to support new markets (e.g., managing doctors' offices), new customer requirements, changes in state laws and regulations, and the needs of the different regions and cultures.
The manager identified the following quality attributes as important, but of somewhat lower priority:

- **Security.** The system had to provide the normal commercial level of security (e.g., confidentiality and data integrity) required by financial systems.

- **Availability.** The system had to be highly available during normal business hours.

- **Scalability.** The system had to scale up to meet the needs of the largest hospital customers and down to meet the needs of the smallest walk-in clinics.

- **Modularity.** The developing organization was entertaining the possibility of selling not just new versions of Nightingale but individual components of it. Testability and supportability.
Step 3: Present Architecture

- Nightingale consisted of two major subsystems: OnLine Transaction Manager (OLTM) and Decision Support and Report Generation Manager (DSRGM).

- Nightingale was built to be highly configurable.

- The OnLine Transaction Manager subsystem was strongly layered.

- Nightingale was a repository-based system; a large commercial database lay at its heart.

- Nightingale relied heavily on COTS software, including the central database, a rules engine, a work flow engine, CORBA, a Web engine, a software distribution tool, and many others.

- Nightingale was heavily object oriented, relying on object frameworks to achieve much of its configurability.
Layered view of the OLTM in the architect's informal notation
A view showing communication, data flow, and processors of the OLTM
Data flow architectural view of the OLTM
Step 4: Catalog Architectural Approaches

After the architecture presentation, the evaluation team listed the architectural approaches they had heard, plus those they had learned about during their pre-evaluation review of the documentation. The main ones included

- layering, especially in OLTM.
- object orientation.
- use of configuration files to achieve modifiability without recoding or recompiling.
- client-server transaction processing.
- a data-centric architectural pattern, with a large commercial database at its heart.
Step 5: Generate Quality Attribute Utility Tree

Table 11.5.
Tabular Form of the Utility Tree for the Nightingale ATAM Exercise
Step 6: Analyze Architectural Approaches

• Replacing the commercial database with a database supplied by another vendor would be difficult.

• Replacing one operating system with another would be a reasonably straightforward change.

• Changing the rules engine raised several issues of concern.
Step 7: Brainstorm and Prioritize Scenarios

The stakeholders were a productive group, contributing a total of 72 scenarios during this step. More than a dozen of those scenarios were found at the leaves of step 5's utility tree but were not analyzed during phase 1. This was not only proper but encouraged.

Table 11.6. Brainstormed Scenarios
Step 8: Analyze Architectural Approaches

• During step 8, we analyzed seven additional scenarios, a number slightly above average for an ATAM exercise.

• In deference to space limitations, the Scenario 15 sidebar summarizes the analysis for just one of them.
Step 9: Present Results

• Step 9 is a one- to two-hour presentation summarizing the results and findings of the exercise.

• It begins with a boilerplate set of slides that contains a method recap and blank template slides that can be filled in with:
  • the business drivers summary,
  • the architecture summary,
  • the list of approaches, the utility tree,
  • the scenario analysis, and
  • the list of analysis outputs.
PHASE 3: FOLLOW-UP

- The tangible output of the ATAM is a final report that contains a list of risks, nonrisks, sensitivity points, and tradeoff points.
- It also contains a catalog of architectural approaches used, the utility tree and brainstormed scenarios, and the record of analysis of each selected scenario.
- Finally, the final report contains the set of risk themes identified by the evaluation team and an indication of which business drivers are jeopardized by each one.