ATAM: Method for Architecture Evaluation

- Attribute-Based Architectural Styles
- The Steps of the ATAM

ATAM: Method for Architecture Evaluation

- Focuses on quality attribute requirements.
- Critical to have precise characterizations for each quality attribute.
- Quality attribute characterizations answer the following questions about each attribute:
  - What are the stimuli to which the architecture must respond?
  - What is the measurable or observable manifestation of the quality attribute by which its achievement is judged?
  - What are the key architectural decisions that impact achieving the attribute requirement?
- Architectural styles are the main determiners of architectural quality attributes.
  - Identification of
    - business goals which lead to
    - quality attribute goals.

Based upon the quality attribute goals,

ATAM steps

1. **Present the ATAM.**
   The method is described to the assembled stakeholders
   - customer representatives,
   - the architect or architecture team,
   - user representatives,
   - mountaineers,
   - administrators,
   - managers,
   - testers,
   - integrators,
   - etc.

2. **Present business drivers.**
   The project manager describes what business goals are motivating the development effort and hence what will be the primary architectural drivers e.g.,
   - high availability or
   - time to market or
   - high security.

3. **Present architecture.**
   The architect will describe the proposed architecture, focusing on how it addresses the business drivers.
   - Investigation and
   - Analysis

4. **Identify architectural approaches.**
   Architectural approaches are identified by the architect, but are not analyzed.
5. Generate quality attribute utility tree.
   The quality factors that comprise system “utility”
   - performance,
   - availability,
   - security,
   - modifiability,
   - etc.
   are elicited, specified down to the level of scenarios, annotated with stimuli and responses, and prioritized.

6. Analyze architectural approaches.
   Based upon the high-priority factors identified in Step 5, the architectural approaches that address those factors are elicited and analyzed e.g.
   - an architectural approach aimed at meeting performance goals will be subjected to a performance analysis
   During this step
   - architectural risks,
   - sensitivity points, and
   - trade-off points are identified

7. Brainstorm and prioritize scenarios.
   Based upon the exemplar scenarios generated in the utility tree step, a larger set of scenarios is elicited from the entire group of stakeholders. This set of scenarios is prioritized via a voting process involving the entire stakeholder group.

8. Analyze architectural approaches.
   This step reiterates step 6, but here the highly ranked scenarios from Step 7 are considered to be test cases for the analysis of the architectural approaches determined thus far. These test case scenarios may uncover additional architectural approaches, risks, sensitivity points, and trade-off points which are then documented.

   Based upon the information collected in the ATAM
   - styles,
   - scenarios,
   - attribute-specific questions,
   - the utility tree,
   - risks,
   - sensitivity points,
   - trade-offs
   the ATAM team presents the findings to the assembled stakeholders and potentially writes a report detailing this information along with any proposed mitigation strategies.

Questions to consider

- Are the servers single- or multi-threaded?
- How are priorities assigned to processes?
- How are processes allocated to hardware?
- What is the physical location of the hardware and its connectivity?
- What are the bandwidth characteristics of the network?
- How is queuing and prioritization done in the network?
- Do you use a synchronous or an asynchronous protocol?
- What is the impact of uni-cast or multicast broadcast protocols?
- What is the location of firewalls and their impact on performance?
- What information is cached versus regenerated?
- Based upon what principles?
What is the performance impact of a thin versus a thick client?

How are resources allocated to service requests?

How do we characterize client loading, (e.g., how many concurrent sessions, how many users)?

What are the performance characteristics of the middleware: load balancing, monitoring, reconfigure services to resources?

If this architecture includes layers/facades,
• are there any places there where the layers/facades are circumvented?

If this architecture includes a data repository,
• how many distinct locations in the architecture have direct knowledge of its data types and layout?

If a shared data type changes,
• how many parts of the architecture are affected?

If there are multiple processes competing for a shared resource,
• how are priorities assigned to these processes and the process controlling the resource?

If there are multiple pipelines of processes/threads,
• what is the lowest priority for each process/thread in each pipeline?

If multiple message streams arrive at a shared message queue,
• what are the rates and distributions of each stream?

Are there any relatively slow communication channels along an important communication path?
• e.g., a modem

If redundancy is used in the architecture,
• what type of redundancy
  - analytic,
  - exact,
  - functional
• how is the choice made between redundant components?

How are failures identified?
• Can active as well as passive failures be identified?

If redundancy is used in the architecture,
• how long does it take to switch between instances of a redundant component?
### Attribute-Based Architectural Styles

- A style can be thought of as a set of constraints on an architecture, constraints on component types and their interactions, and these constraints define the set or family of architectures that satisfy them.
  - By locating architectural styles in an architecture, we see what strategies the architect has used to respond to the system’s:
    - driving performance goals,
    - modifiability goals,
    - availability goals,
    - and so forth.

The ATAM uses a particular specialization of this called attribute-based architectural styles, or ABASs.

#### Examples of ABASs include

- **Modifiability Layering ABAS:**
  - analyzes the modifiability of the layers architectural style by examining potential effects of various modification scenarios.
- **Performance Concurrent pipelines ABAS:**
  - applies rate monotonic analysis to multiples pipelines on a single processor, each of which have real-time deadlines.
- **Reliability Tri-modular redundancy ABAS:**
  - applies Markov modeling to a classical style of redundancy used to enhance system reliability.

### Outputs of the ATAM

#### Risks and Non-Risks

- Risks are potentially problematic architectural decisions.
- Non-risks are good decisions that rely on assumptions that are frequently *implicit* in the architecture.

Both should be understood and explicitly recorded.

- The documenting of risks and non-risks consist of:
  - an architectural decision (or a decision that has not been made)
  - a specific quality attribute response that is being addressed by that decision along with the consequences of the predicted level of the response
  - a rationale for the positive or negative effect that decision has on meeting the quality attribute requirement
An example of a risk is:

- The rules for writing business logic modules in the second tier of your three-tier client server style are not clearly articulated
  - a decision that has not been made.

This could result in replication of functionality thereby compromising modifiability of the third tier
  - a quality attribute response and its consequences.

Unarticulated rules for writing the business logic can result in unintended and undesired coupling of components
  - rationale for the negative effect.

The Steps of the ATAM

- **Step 1 - Present the ATAM**
- **Step 2 - Present Business Drivers**
  - The system itself must be presented, initially at a high level of abstraction, typically describing:
    - its most important functional requirements
    - its technical, managerial, economic, or political constraints
    - its business goals and context
    - its major stakeholders
    - the architectural drivers (major quality attribute goals that shape the architecture)

- **Step 3 - Present Architecture**
  - In this presentation the architecture should cover
    - technical constraints such as an OS, hardware, or middleware prescribed for use
    - other systems with which the system must interact
    - architectural approaches used to meet quality attribute requirements

- **Step 4 - Identify Architectural Approaches**
- **Step 5 - Generate Quality Attribute Utility Tree**
- **Step 6 - Analyze Architectural Approaches**
  - The questions help us to
    - understand the approach
    - look for well-known weaknesses with the approach
    - look for the approach’s sensitivity points
    - find interactions and trade-offs with other approaches

- **Step 7 - Brainstorm and Prioritize Scenarios**
  - Scenarios are examples of architectural stimuli used to both
    - represent stakeholders’ interests
    - understand quality attribute requirements

- **Step 8 - Analyze Architectural Approaches**

- **Step 9 - Present Results**
  - the set of ATAM outputs:
    - the architectural approaches/styles documented
    - the set of scenarios and their prioritization
    - the set of attribute-based questions
    - the utility tree
    - the risks discovered
    - the non-risks documented
    - the sensitivity points and trade-off points found
The Two Phases of ATAM

Phase 1:
A small meeting, typically between a small subset of both the evaluation team and the customer team: gather as much information as possible to determine
- if the remainder of the evaluation is feasible and should proceed
- if more architectural documentation is required and, if so, precisely what kinds of documentation and how it should be represented
- what stakeholders should be present for Phase 2

Phase 1 Issues
- What is the structure of the message-handling software i.e., how is the functionality is broken down in terms of
  - modules,
  - functions,
  - APIs,
  - layers,
  - etc.?
- What facilities exist in the software architecture (if any) for self-testing and monitoring of software components?
- What facilities exist in the software architecture (if any)
  - for redundancy,
  - liveliness monitoring,
  - fail over,
  - and how data consistency is maintained
  - so that one component can take over from another and be sure that it is in a consistent state with the failed component
- What is the process and/or task view of the system, including mapping of these processes/tasks to hardware and the communication mechanisms between them?
- What functional dependencies exist among the software components?
  often called a “uses” view
- What data is kept in the database,
  - how big is it,
  - how much does it change, and
  - who reads/writes it?
- What is the anticipated frequency and volume of data being transmitted among the system components?

Phase 2:
At this point, it is assumed that the architecture has been documented in sufficient detail to support verification of the analysis already performed, and further analysis if needed.
- The appropriate stakeholders have been gathered and have been given advance reading materials such as
  - a description of the ATAM,
  - the seed scenarios, and
  - system documentation including
    - the architecture,
    - business case, and
    - key requirements.
These reading materials aid in ensuring that the stakeholders know what to expect from the ATAM.
A Sample Utility Tree

(M,L) Minimize storage latency on customer DB to 200 ms.

(H,M) Deliver video in real time

(L,H) Add CORBA middleware in < 20 person-months

(H,L) Change web user interface in < 4 person weeks

(M,M) Restart after disk failure in < 5 mins

(L,H) Power outage at Site 1 requires traffic re-direct to Site 2 in < 3 secs

(H,M) Network failure is detected and recovered in < 1.5 mins

(L,H) Credit card transactions are secure 99.999% of time

(L,H) Customer database authorization works 99.999% of time