Individual Research: Proposal

Topic: Architecture Tradeoff Analysis Method (ATAM)

Student: Dazhi Li

## Abstract

*The Architecture Tradeoff Analysis Method (ATAM) is a structured approach used to evaluate software architectures based on their ability to meet quality attributes. This proposal explores the purpose of ATAM, its systematic evaluation steps, and its role in assessing software quality. The study highlights how ATAM provides a comprehensive framework for improving architectural decision-making and contributes to the overall quality assurance of software systems.*

## Relevance to Class Objectives

Here are some aspects that I found ATAM is strongly related to our course materials

1. Related to our course objectives. Both ATAM and our course have a common target of learning how to achieve high quality software design. By systematically evaluating architectural choices and their impact on software quality, ATAM ensures that software systems are designed with a strong foundation, reducing technical debt and enhancing overall quality.
2. Related to the quality attribute relationship. For an example, improving security is has a negative influence on performance and so forth. ATAM is an explicit methodology to find out the relationship in a software structure, that is what they called tradeoff point in a structure. By finding such tradeoff point, architect could make their decision on which quality attribute to improve while another quality attribute to be impaired.
3. A supplement for our reading materials. Unfortunately, ATAM is mentioned and interpreted in our reading material [3] on Chapter 2. However, my research contents are quite distinguishing. [3] focuses on describing communitive steps of ATAM in stakeholders and some personal experience to demonstrate it. My concentrate on the first publishment of ATAM [1] with some examples to understand its running principles and terminology. My research is another perspective on how to find a tradeoff point from architect view and utilize it on decision making.

## What is ATAM and important terminology

Before introducing ATAM, knowing the background when it was born is very important. This methodology was first published during 1998. People at that time were getting familiar to quality. However, only experienced designers knew tradeoff exists in those quality attributes. And there is no principled method for characterizing them and, in particular, for characterizing the interactions among attributes. To address this problem, ATAM was born in an explicit structured way.

**ATAM** is a method for evaluating architecture-level designs that considers multiple quality attributes such as modifiability, performance, reliability, and security in gaining insight as to whether the fully fleshed out incarnation of the architecture will meet its requirements. The method identifies tradeoff points between these attributes, facilitates communication between stakeholders (such as user, developer, customer, maintainer) from the perspective of each attribute, clarifies and refines requirements, and provides a framework for an ongoing, concurrent process of system design and analysis.

Before starting to exploring the actionable process of ATAM, understanding some important terminology is necessary.

**Sensitivity points** [2]: A sensitivity point is a property of one or more components (and/or component relationships) that is critical for achieving a particular quality attribute response. For an example, the level of confidentiality in a virtual private network might be sensitive to the number of bits of encryption.

**Tradeoff points** [2]: A tradeoff point is a property that affects more than one attribute and is a sensitivity point for more than one attribute. For example, changing the level of encryption could have a significant impact on both security and performance.

More terminology including **utility tree** and **priority scenarios** could be found in [3]. I suppose that all the all the authors have read the course reading material. Besides, this paper is not digging into that part for distinguishing course materials.

## Overall Actionable Process of ATAM (communitive)

This part is describing on overall steps of conduct ATAM. It is more like the communitive actions among the stakeholders that require everyone to take part in. This part is mainly based on [2]. This is essential for understanding ATAM is a large communitive architecture analysis method that required everyone involved.

**Step1 - Present the ATAM**

A presentation lead by the evaluation team to the assembled stakeholders. It includes ATAM process that everyone will be following and what information will be collected.

**Step2 - Present Business Drivers**

In this step the project manager presents a system overview from a business perspective. This could be highly abstract, typically describing.

**Step 3 - Present Architecture**

The architecture will be presented by the lead architect (or architecture team) at an appropriate level of detail. This includes technical constraints, other systems with which the system must interact and architectural approaches used to meet quality attribute requirements.

**Step 4 - Identify Architectural Approaches**

Architecture candidates are captured by the analysis team, including which highest priority quality attribute each will resolve.

**Step 5 - Generate Quality Attribute Utility Tree**

In this step the evaluation team works with the architecture team, manager, and customer representatives to identify, prioritize, and refine the system’s most important quality attribute goals. The output of the utility tree generation process is a prioritization of specific quality attribute requirements, realized as scenarios.

**Step 6 - Analyze Architectural Approaches (My research focus)**

In this step the evaluation team probe for the architectural approaches that realize the important quality attributes. This is done with an eye to documenting these architectural decisions and identifying their risks, sensitivity points, and tradeoffs.

**Step 7 - Brainstorm and Prioritize Scenarios**

This step requires all the stakeholders to brainstorm use case scenarios first. Similar use case will be combined. Then all the stakeholders start to voting on those scenarios based on their thinking of priority.

**Step 8 - Analyze Architectural Approaches**

In this step we reiterate Step 6, mapping the highest ranked newly generated scenarios onto the architectural artifacts thus far uncovered. Assuming Step 7 didn’t produce any high-priority scenarios that were not already covered by previous analysis, Step 8 is a testing activity: We hope and expect to be uncovering little new information. This is a testing activity: at this point we hope and expect to be uncovering little new information.

**Step 9 - Present Results**

Finally, the collected information from the ATAM needs to be summarized and presented back to the stakeholders. This presentation typically takes the form of a verbal report accompanied by slides but might, in addition, be accompanied by a more complete written report delivered subsequent to the ATAM.

## Specific Actionable Process on Finding Tradeoff Points

**Step 1 - Proposing Architectures**

Prepare some architectures that are potential to solve the client’s problem. At least functionally working.

**Step 2 - Quality Attribute Analysis**

Analysis these architectures by different quality attributes. There attributes include but not exclude performance, availability, security and so on. As for how to connect the architecture to a single quality attribute and visualize/quantify it, the methods are not limited, but better be quantified by some metrics.

**Step 3 - Find Sensitivity Points**

By observing the quality attribute analysis result, find out huge quality attribute variance when one factor changes. Identify those factors to be sensitivity points and record.

**Step 4 - Find Tradeoff Points**

By observing those sensitivity points, find out which sensitivity points could cause more than two quality attributes changes. And if those two changes inversely, i.e. one improving while the other impairing, identify this point as tradeoff points.

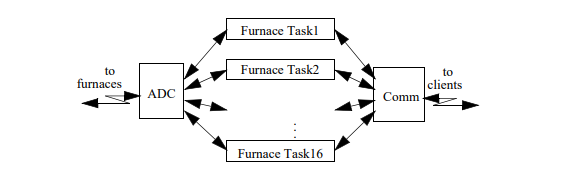
**Step 5 - Architecture evolution**

If the stakeholders have some tolerance to lose some quality attribute, this step could be skipped. However, usually architect could try to find another way to deal with the tradeoff points. At lease let the negative impact being smaller. Further architecture evolution requires another cycle of analysis and identification of sensitivity and tradeoff points.

## A Case Study: Finding tradeoff points and evolution

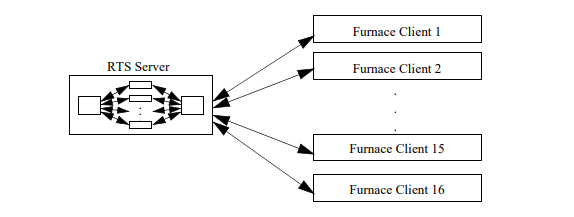
Here is a very classic example from [1] that could be applied with our ATAM process.

To begin with, describing the requirements and system are necessary. The RTS (remote temperature sensor) system exists to measure the temperatures of a set of furnaces, and to report those temperatures to an operator at some other location. The remote temperature sensor functionality is encapsulated in a server that serves some number of clients. To remain consistent with the original problem, our analysis will assume that there are 16 clients: one per furnace. And the architecture of the furnace server is given below.

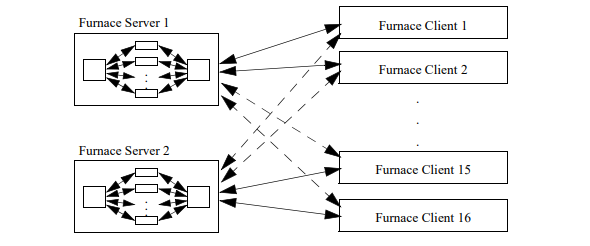


Now let us follow our **step 1**. Three different software architectures are given below.

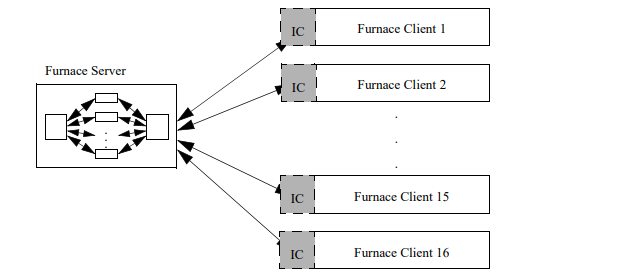
Option 1 is the baseline: a simple and inexpensive client-server architecture, with a single server serving all 16 clients.



Option 2 differs from option 1 in that it adds a second server to the system architecture. These servers interact with clients as a “primary” server (indicated by the solid lines between servers and clients) or as a “backup” server (indicated by the dashed lines). As shown below, under normal operation each server only serves 8 of the 16 clients.



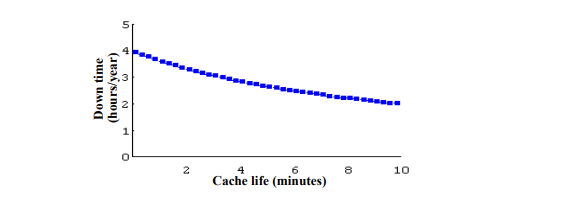
Option 3 differs from option 1 in only one way: each client has a “wrapper” that intercedes between it and the server. This wrapper is an “intelligent cache,” shown as IC below. The cache works as follows: it intercepts periodic temperature updates from the server to the client, builds a history of these updates, and then passes each update to the client. In the event of a service interruption, the cache synthesizes updates for the client. It is an intelligent cache because the updates it provides take advantage of historical temperature trends to extrapolate plausible values into the future.



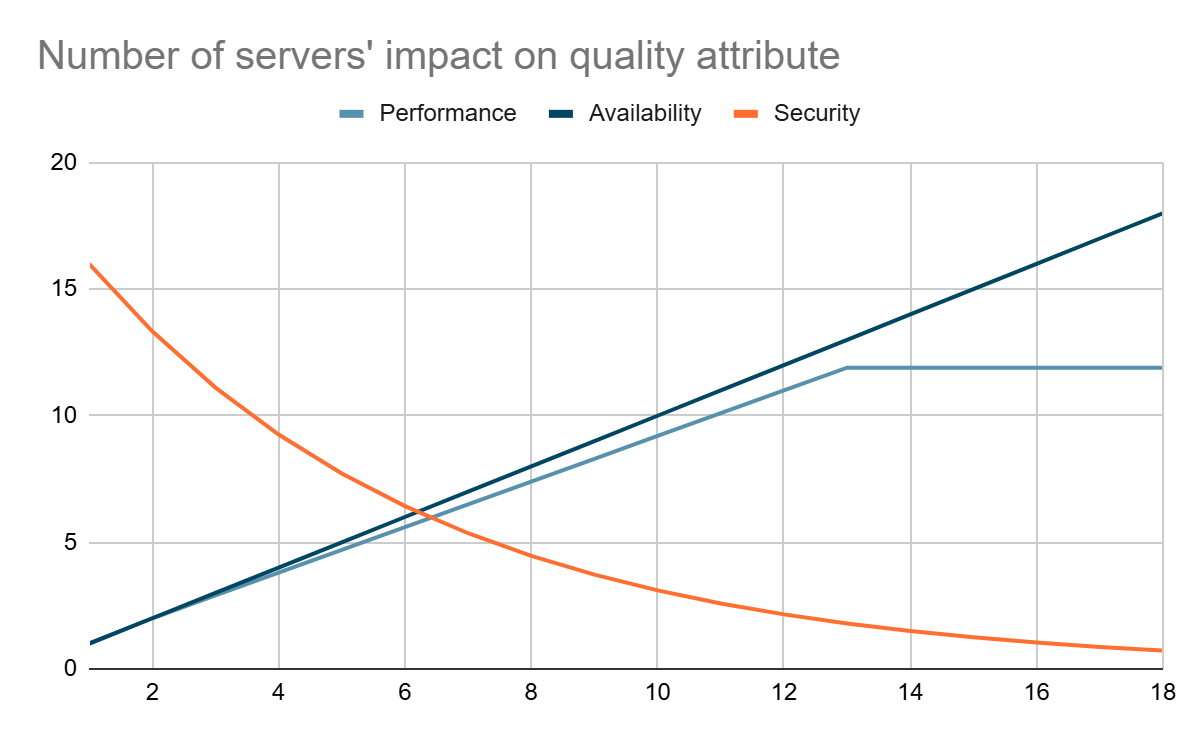
Then let us do the **Step2**. Suppose that our clients require performance, availability and security. And the result of the analysis is shown below.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Single Server | Double Server | Intelligent Cache |
| Performance (average latency) | 5,100 ms | 2,550 ms | 5,200 ms |
| Availability (Unavailable duration) | Hardware Down(Hrs):  278.833  Software Down(Hrs):  3.9982 | Hardware Down(Hrs): 17.7327  Software Down(Hrs): 0.003649 | Hardware Down(Hrs):  276.91  Software Down(Hrs): 2.66545 |
| Security (spoof attack in 60 mins) | 2.70072 | 3.8766 | 2.70072 |

Then let us do **Step3** to identify some sensitivity points. As what we can see above, if we try to increase the number of servers from 1 to 2, the performance, availability and security changes manifestly. Then we could make sure that performance, availability and security is sensitive to number of servers. However, if we try to increase the life time of intelligent cache, the story is totally different. The performance and security do not change at all while the availability changes a little bit. As shown below, the impact of increasing the life time is getting vanished when the life time is even larger.



Then we do **step 4** to identify tradeoff point among those sensitivity points. Luckily, the number of servers is impacting more than two quality attributes negatively and positively. The trending of how number of servers influences those quality attributes are shown below.



Finally, we will do **step5**. If our clients say that security is an importance requirement that they demand, architect could evolve the multi-server architecture by applying encryption and detection modules to solve this and do another round of analysis.

## How does ATAM help in decision making

In my perspective, the tradeoff points that are identified during the architecture analysis could only help architect document them and raise the alert of risks in the community. Only some architectures which are influenced too much by tradeoff points that some qualities cannot be satisfied could utilize tradeoff points to cut some quality attribute. Most cases, these sensitivity points and tradeoff points are going to be presented to the stakeholders and documented for decision making process. And what we should keep in mind is that, ATAM is a communitive evaluation process, these decisions are usually made by the stakeholders all together by what are discovered during the ATAM process. As [2] said the point of this example is not to show which alternative the contractor chose, for that is relatively unimportant and relied on their organizational and mission-specific constraints. This, in turn, focused design activity in the areas of highest risk and caused a major iteration within the spiral process of design and analysis.

## Recommendations

When thinking of a solution to a single problem, do not limit the answer to a single architecture. We could come up with a shole candidates of architecture and evaluate them in isolation. Then connect them all together to find out what is a sensitive element to the quality attributes. The sensitive elements are highly likely to be the tradeoff points.

Architecture evaluation is a communitive process. More stakeholder involvements lead to a better result. This process also needs a structured method and schedule for being executed successfully.

## Application of Proposal

There are many real examples that ATAM could be applied to our software architecture evaluation.

Common Avionics Architecture System (CAAS) [4]: The U.S. Army's Technology Applications Program Office (TAPO) collaborated with Rockwell Collins to develop the CAAS for a fleet of Special Operations helicopters. To ensure the architecture met critical quality attributes like modifiability, performance, and reliability, an ATAM evaluation was conducted. This assessment identified potential risks and trade-offs, guiding the development of a robust and flexible avionics system.

VAN-Core System for Online Credit Card Transactions [5]: A study applied ATAM to evaluate and enhance the architecture of the VAN-Core system, which processes online credit card transactions continuously. Given the system's requirement for high availability and reliability, the ATAM evaluation focused on identifying architectural risks and trade-offs related to performance and fault tolerance. The insights gained led to improvements that ensured the system could operate effectively 24/7.

## Conclusion

Although ATAM is published two decades ago, it is still a useful method in architecture evaluation. The actionable process of ATAM is very clear and structured that anyone could easily follow. Moreover, the process of finding sensitivity points and tradeoff points signifies the importance of understanding the risk in our architecture. Also, ATAM is widely used in our real-world cases. It is valuable for us to study this method to make high quality design.

## References

[1] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and S. Carriere, "The Architecture Tradeoff Analysis Method," *Carnegie Mellon University, Software Engineering Institute's Digital Library*. Software Engineering Institute, Technical Report CMU/SEI-98-TR-008, 1-Jul-1998 [Online]. Available: https://insights.sei.cmu.edu/library/the-architecture-tradeoff-analysis-method/.

[2] R. Kazman, M. Klein, and P. Clements, "ATAM: Method for Architecture Evaluation," Carnegie Mellon University, Software Engineering Institute's Digital Library. Software Engineering Institute, Technical Report CMU/SEI-2000-TR-004, 1-Aug-2000 [Online]. Available: https://insights.sei.cmu.edu/library/atam-method-for-architecture-evaluation/.

[3] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 4th Edition. Carnegie Mellon University, Software Engineering Institute’s Digital Library, 2021. [Online]. Available: <https://insights.sei.cmu.edu/library/software-architecture-in-practice-fourth-edition/>

[4] M. Barbacci, P. Clements, A. Lattanze, L. Northrop, and W. Wood, "Using the Architecture Tradeoff Analysis Method (ATAM) to Evaluate the Software Architecture for a Product Line of Avionics Systems: A Case Study," *Carnegie Mellon University, Software Engineering Institute's Digital Library*. Software Engineering Institute, Technical Note CMU/SEI-2003-TN-012, 1-Jul-2003 [Online]. Available: <https://doi.org/10.1184/R1/6585797.v1>.

[5] J. Lee, S. Kang, H. Chun, B. Park and C. Lim, "Analysis of VAN-Core System Architecture- A Case Study of Applying the ATAM," 2009 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing, Daegu, Korea (South), 2009, pp. 358-363, doi: 10.1109/SNPD.2009.90.