Give the Stakeholders What They Want: Design Peer Reviews the ATAM Style

Felix Bachmann, Software Engineering Institute

Abstract. The Architecture Tradeoff Analysis Method® (ATAM®) is used to evaluate the architecture of a software intensive system to determine if it meets the organization's business and mission goals [1]. ATAM is typically applied at the end of the architecture design process. Looking at the results of many past ATAM evaluations, it becomes apparent that some mechanism is needed to better guide the architecture design process. Many ATAM evaluations show major issues with the system that could have been avoided with the right design approach.

In a recent project, SEI guided architects of an organization through the design of a major system for the financial market. We used a Quality Attribute Workshop (QAW) to generate the first set of important quality attribute scenarios that interested stakeholders. The design was created utilizing the SEI's Attribute Driven Design method (ADD) [2] in which the generated scenarios were transformed into an architecture design. On a bi-weekly basis, ATAM style peer reviews were conducted with the architects to ensure that the design actually addressed the requirements. This combination of methods achieved some interesting results for this project:

* First, an architecture evaluation using ATAM conducted at the end of the architecture design process was completed in half the time than comparable ATAMs done on large software systems and did not uncover any unexpected risks. In short, the evaluation showed that the system indeed provides what the stakeholders want.

* Secondly, the ADD method combined with the ATAM style peer review made the architecture design tasks transparent for both project management and the stakeholders. Instead of trying to explain architecture diagrams, scenarios (from the QAW) with their associated risks (from the ATAM style peer review) were reviewed on a biweekly basis. Seeing the risks being mitigated over time convinced the stakeholders that the project was on the right track.

* Finally, the architecture team never had to be pushed to document their architecture. Just the fact that the architects had to prepare for the biweekly peer reviews was sufficient incentive to write down how the current design would fulfill the stakeholder scenarios. Basically, the architecture documentation was created continuously during the design with no additional effort.

Quality Attribute Scenarios

Properly designing software architecture means that, aside from the necessary functionality, the system will meet the required quality attribute requirements such as modifiability, interoperability, and security, to name just a few [3]. In fact, it is the system's architecture that determines if the system meets the quality attribute requirements. That is why SEI created architecture methods like QAW [4] and ADD [5], as well as the ATAM [2], that are centered on utilizing quality attribute scenarios as a more precise way of specifying the quality attribute requirements a system has to fulfill.

During a QAW, the stakeholders articulate and prioritize quality attribute scenarios based on their business and mission goals. It is the architect's job to then take those scenarios and transform them into a design that will support these goals. This is exactly what SEI's ADD method is used for. Performing an ATAM at the end of the design process involves reviewing the quality attribute scenarios again, verifying with the stakeholders that the scenarios are still valid, and then verifying that the architecture supports these scenarios.

This sounds like a valid process, but a surprising number of ATAMs reveal that the architecture does not fulfill the requirements. Independent of the reasons why, in many cases there is no time in the schedule to actually go back and redesign the architecture. The only remaining alternatives are to either end the project or to move forward with a system designed with inherent risk hoping that nothing bad will happen.

ATAM Style Design Peer Reviews

During the design process there have to be some checkpoints that allow verification that the design will fulfill the stakeholder's expectation. Conducting peer reviews is a common method for doing so. As the ATAM results show, in many cases those peer reviews do not ensure that the appropriate system is developed. To overcome this weakness we introduced a peer review process that utilizes the same techniques ATAM evaluation uses. This makes the design process similar to Kent Beck's Test Driven Development (TDD) [6]. In a test-driven development, tests are created first, then the part of the system that is executed by the test is developed and then the tests are run. If a test fails, the developed code is corrected and the test is run again. These steps are repeated until all tests pass.

In an architecture design, the tests are actually the quality attribute scenarios. The architecture design must fulfill those scenarios to be accepted by the stakeholders as a good design. As was stated above, the scenarios are already defined during a QAW before the architecture is designed. The architects ensure that the current design is checked in a periodic fashion to see if the scenarios are continuing to be fulfilled. Running tests on the current design means performing a peer review using the techniques of the ATAM.

The timing and the scope of the peer review strongly depend on system complexity and the quality attribute scenarios. In our case, we decided to conduct an ATAM style design peer review every two weeks. We allocated three hours for the review and we were able to review two scenarios during those three hours. At the beginning of a two-week cycle, the architects decided on the two scenarios to focus on for that cycle. The architects then had two weeks to design the system that would support those two scenarios. During that time period, the architects had to produce a documented design including evidence that the design was appropriate and that past scenarios, already checked in earlier reviews, were not now being violated by the updated design. At the end of the cycle, a peer review was conducted with ATAM-trained SEI architects.

Steps of the ATAM Style Peer Review

An ATAM-style peer review is done by building a review team that consists of the system's architecture team and two other architects to act as reviewers. In our case we used architects from SEI, but any architect, not involved in the project and knowledgeable in the ATAM method would be able to do the review.

It is the responsibility of the architecture team to provide all necessary documentation to reviewers at the beginning of the review, explaining why the chosen scenarios are well supported. Typically the architecture team leader is the main speaker, but the other architects also provide information whenever necessary. At least one architect needs to have a good understanding of what the stakeholders actually meant when they created the scenarios. This knowledge helps to identify when a scenario was written ambiguously.

The reviewers' main responsibility is to ask questions that help the architecture team uncover issues in their design. One of the reviewers acts as a facilitator, responsible for guiding the whole review team through the review process. The other reviewer acts as a scribe, writing down the approaches, risks and the to-do items.

Let us have a more detailed look into the ATAM style design peer review process.

Step 1: Select the scenario to analyze.

A design peer review needs to have a clear focus. Instead of analyzing the whole architecture—which could be a very tedious task—the review only needs to uncover the risks associated with one scenario. The peer review starts by selecting the scenario to review. This is usually one of the scenarios selected at the beginning of the two-week design cycle, but could also be any other scenario. It often happens that, during the design process, a scenario will get refined into multiple, more detailed scenarios addressing different aspects of the requirements. For example, the starting scenario could have been one stating that the system has to be available 24 hours, seven days per week. During the design process, the architects may have discovered that hardware failures and software failures need to be treated differently. Therefore, the availability scenario might have been broken into two scenarios, each addressing different aspects of availability.

As a rule of thumb, we saw that every scenario created by stakeholders during a QAW was typically divided into three to five more specific scenarios.

A design peer review needs to have a clear focus. Instead of analyzing the whole architecture—which could be a very tedious task—the review only needs to uncover the risks associated with one scenario.

Step 2: Elicit the architecture approaches.

An architecture approach is a pattern or tactic [2] used in the architecture to support the chosen quality attribute scenario. On one hand, eliciting the approaches allows reviewers to very quickly see if there is sufficient support for that scenario. On the other hand, it also allows the reviewers to ask questions about the possibly negative consequences the approach has on other scenarios. The scribe writes down the approaches including the rationale on why they were chosen.

Step 3: Analyze architecture approaches.

The analysis of the architecture approaches is done as a question and answer session where the reviewers ask questions about the solution and the architects answer the questions by pointing to the parts of the architecture documentation that provide the answers. Here are clues about how to treat the answers:

* If a question cannot be answered, the scribe writes it down as a risk.

* If the provided answer is problematic because it might violate some other scenarios, it is written down as a risk.

* If the answer is that this is still an open issue, it is written down as a to-do item.

* If the answer satisfies the reviewers, it is written down as evidence with a pointer to the supporting documentation if it was not already done. The scribe also notes every piece of documentation, such as structural diagrams (module views, component and connector views, deployments, views, etc.) or behavioral diagrams (sequence charts, state diagrams, etc.) that was used during the review of the scenario.

Step 4: Review results.

Step 3 usually results in a list of five to 10 risks per scenario. This may sound like a big number, but this level is normal when the scenario is reviewed for the first time. It also means that some sort of redesign has to follow. In Step 4 the architecture team and the reviewers analyze the captured list of risks as well as to-do items and decide on appropriate actions. The best case would be that the scenario is solved and no further action is required. More commonly, the appropriate actions are to adjust the architecture, to build a prototype to provide better insights, or have a discussion with the stakeholders because the scenario might be impossible to achieve.

After Step 4, the architecture team should have a clear view about what to do next with the reviewed scenarios.

Performing ATAM style design peer reviews every second week was never seen as a burden by the architects. They were actually looking forward to the next review because the reviews provided them with valuable input and they could see progress when the list of risks and the to-do list became smaller and smaller over time.

Conclusion

In our experience, performing ATAM style design peer reviews every second week was never seen as a burden by the architects. They were actually looking forward to the next review because the reviews provided them with valuable input and they could see progress when the list of risks and the to-do list became smaller and smaller over time. The architects also saw the value of early feedback. Even if they went down a wrong path, the most time they would lose was two weeks.

These benefits are apparent, but there were other positive side effects. The peer reviews trained the architects to think in terms of uncovering risks and mitigating them. This enabled the architects to have more productive discussions with stakeholders, such as the project manager or the program office about their requirements using scenarios with their attached risks. When product development goes into the architecture design phase, outsiders often perceive that nothing is happening even if the architects show diagrams and pictures. For an outsider those pictures do not mean anything, but talking about scenarios and risks makes the whole architecture design process transparent. Even if you do not understand what those architecture diagrams mean, you can clearly track risks and see progress as those risks are slowly being mitigated.

We also did an ATAM at the end of the architecture design, just to make sure that nothing was missed. The ATAM was done by a completely independent SEI team that was not involved in the design. Since the architecture team was able to provide to the ATAM team all the artifacts that are usually created during Phase 1 of the ATAM, the ATAM team could focus on Phase 2 only. This expedited verification with the stakeholders that indeed the architecture fulfilled their needs. The result was that the ATAM did not find any unknown or unaddressed risk. Basically, the ATAM acknowledged that the stakeholders would get the system they want.

It is also noteworthy that in the project plans there was never a “documentation” task. The fact that the architects had to deliver proof every two weeks automatically led them to document their design immediately. They knew that if an important concept was not written down it would end up as a risk. ❖

Disclaimer:

Architecture Tradeoff Analysis Method® and ATAM® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

ABOUT THE AUTHORS

Felix H. Bachmann is a Senior Member of the Technical Staff at the Software Engineering Institute (SEI) working in the Architecture Centric Engineering Initiative. He is co-author of the Attribute-Driven Design Method, a contributor to and instructor for the ATAM Evaluator Training, and a co-author of Documenting Software Architectures: Views and Beyond.

Before joining SEI he was a software engineer at the Robert Bosch GmbH in Corporate Research, where he worked with software development departments to address the issues of software engineering in small and large embedded systems.

Software Engineering Institute
Felix Bachmann
4500 5th Avenue
Pittsburgh, PA 15213
Phone: (412) 268-6194
Fax: (412) 268-6257
E-mail: fb@sei.cmu.edu

REFERENCES

1. Clements, P.; Kazman, R.; Klein, M. Evaluating Software Architectures: Methods and Case Studies Addison Wesley, 2002. Engineering Institute, Carnegie Mellon University, 2002.
2. Bass, L., Clements, P., and Kazman, R. 2003. Software Architecture in Practice, Second Edition. Boston, MA: Addison-Wesley.
3. Bachmann, F.; Bass, L.; Klein, M. Illuminating the Fundamental Contributors to Software Architecture Quality (CMU/SEI-2002-TR-025). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002.
4. Barbacci, M.; Ellison, R.; Lattanze, A.; Stafford, J.; Weinstock, C.; Wood, W. Quality Attribute Workshops (QAWs), Third Edition (CMU/SEI-2003-TR-016). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003.
5. Wojcik, R.; Bachmann, F.; Bass, L.; Clements, P.; Merson, P.; Nord, R.; Wood, W. Attribute Driven Design (ADD), Version 2.0 (CMU/SEI-2006-TR-023). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.
6. Beck, K. Test-Driven Development: By Example. Addison-Wesley, November 2002.