Quality Attribute Workshops

Mario R. Barbacci
Robert Ellison
Judith A. Stafford
Charles B. Weinstock
William G. Wood

May 2001

TECHNICAL REPORT
CMU/SEI-2001-TR-010
ESC-TR-2001-010

Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Quality Attribute Workshops

CMU/SEI-2001-TR-010
ESC-TR-2001-010

Mario R. Barbacci
Robert Ellison
Judith A. Stafford
Charles B. Weinstock
William G. Wood

May 2001

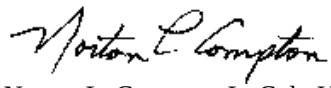
Architecture Tradeoff Analysis Initiative

This report was prepared for the

SEI Joint Program Office
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER



Norton L. Compton, Lt Col., USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-00-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Tables

List of Figures

Abstract

Quality attribute workshops (QAW) provide a method for evaluating the architecture of a software-intensive system during the acquisition phase of major programs. The architecture is evaluated against a number of critical quality attributes, such as availability, performance, security, interoperability, and modifiability. The evaluation is based on test cases that capture questions and concerns elicited from various stakeholders associated with the system. The process of eliciting questions allows stakeholders to communicate directly, thereby exposing assumptions that may not have surfaced during requirements capture. Our experience to date includes twelve quality attribute workshops that were held with three different U.S. Government acquisition programs. In this report, we provide a rationale for developing the process and describe it in detail. We follow this with a list of lessons learned and discuss how these lessons have helped us evolve the process to its current state.

1 Introduction

An architectural description is the earliest representation of a system that can be analyzed for desired quality attributes, such as performance, availability, modifiability, interoperability, and security. The architecture also provides insights as to how these attributes interact, forming a basis for making tradeoffs between these attributes. The quality attribute workshop (QAW) is a method for analyzing a system architecture for these attributes through test cases. The QAW complements the Architecture Tradeoff Analysis Method[SM] (ATAM[SM]) developed by the SEI.

In an ATAM evaluation, an external team facilitates sessions during which scenarios are developed representing the quality attributes of the system. These scenarios are then prioritized, and the highest priority scenarios are analyzed against the architectural approaches chosen for the system. The results are expressed as risks, sensitivity points, and tradeoffs.

In addition to identifying risks, sensitivity points, and tradeoffs, an ATAM evaluation encourages communication among diverse stakeholders (e.g., sponsors, users, designers, maintainers, and acquirers). The results of an ATAM evaluation can show stakeholders where the architecture could be improved before embarking on expensive implementations.

The ATAM has been developed over the past four years and has been applied to over 20 systems. In order to conduct an ATAM evaluation, an articulation of the business drivers and at least an initial draft of the architecture are required. However there are times when an ATAM evaluation would be of limited value, for example, during the acquisition of a large-scale system. In this case,

- Competing teams will likely generate different scenarios making it difficult to use ATAM to compare proposals.
- There might not be a reasonable draft of the architecture available early in the acquisition cycle necessary to conduct an ATAM.

The QAW is a variation of ATAM that addresses these situations. In a QAW, the highest priority stakeholder-generated scenarios are turned into test cases by adding additional details (e.g., context, assets involved, sequence of activities, and others). The architecture team then

---

[SM] Architecture Tradeoff Analysis Method and ATAM are service marks of Carnegie Mellon University.

For additional information, visit the SEI Architecture Tradeoff Analysis Initiative website: www.sei.cmu.edu/ata/ata_init.html

independently analyzes the test cases against the architecture and documents the results. The test case creation and analysis phase often takes place over an extended period of time. In the final phase, the architecture team presents the results to the sponsors and stakeholders.

There are a number of benefits to the QAW approach:

- The process provides opportunities for communication among the architecture team and the other stakeholders.
- Scenario and test case generation can be done before the architecture is in place.
- The test cases provide a mechanism for analyzing the consequences of architectural decisions.
- The results of the analysis provide a mechanism for improving the architecture before presenting it to stakeholders.
- Additional scenarios can be developed into new test cases for further analyses in a continuous process.
- The process gives the sponsor or other significant stakeholders early insight into the capabilities of the architecture team, and builds confidence in the architecture team's understanding of the challenges involved with developing the system.

The remainder of this report describes the QAW approach. Section 2 covers the overall process of generating scenarios, developing test cases, and analyzing an architecture using test cases. The section also describes the sequence of activities in a QAW. These activities include both workshop and analytical exercises that stakeholders perform between workshops. Section 3 describes variations in QAWs implemented for different organizations. Section 4 captures lessons learned during the initial QAWs and summarizes our experience with the QAW approach. The appendix provides an example scenario, its refinement, and a test case derived from the refined scenario.

2 Process Outline

The QAW process is shown in Figure 1. The process can be organized into four distinct segments: (1) scenario generation, prioritization, and refinement; (2) test case development; (3) analysis of test cases against the architecture; and (4) presentation of the results. These are the four gray ovals in the figure.

The first and last segments of the process occur in facilitated one-day meetings. The middle segments take place off-line and can continue over an extended period of time.

The process is iterative in that the test case analyses might lead to the development of additional test cases or to architectural modifications. Architectural modifications might prompt additional test case analyses, etc. There is a further iteration, not shown in the figure, in which test cases are developed in batches, resulting in a continuous cycle of analyses and architectural modifications.

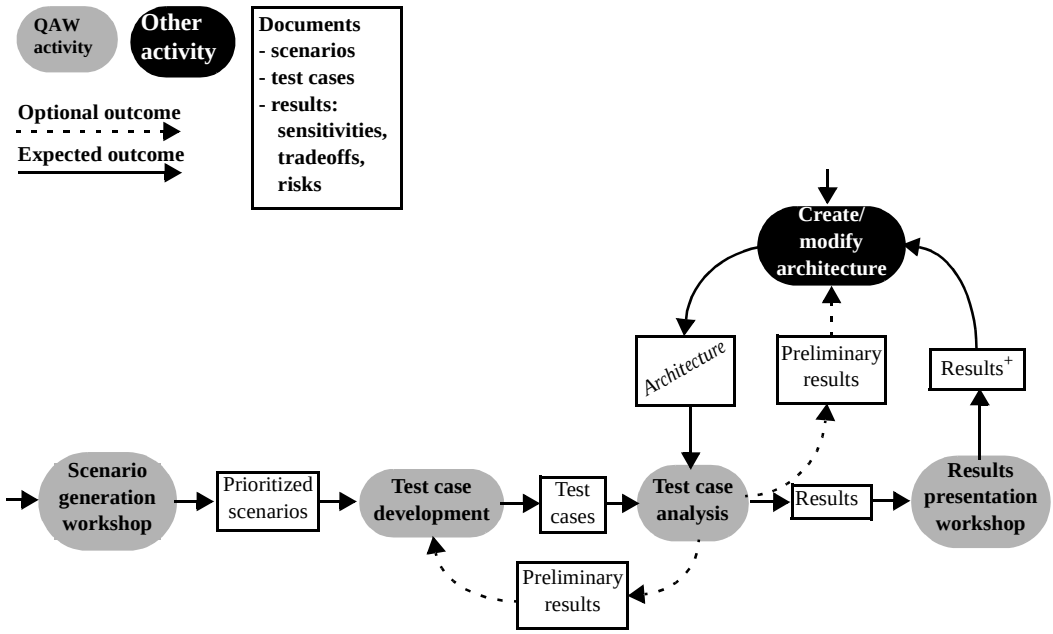


Figure 1: The QAW Process

2.1 Scenario Generation Workshop

The first activity in the QAW process is to carry out scenario generation, prioritization, and refinement. The scenarios are generated during a brainstorming workshop attended by facilitators, stakeholders, and the architecture team. The stakeholders are provided in advance with a workbook containing descriptions of several quality attributes, sample questions to assist in generating scenarios, example scenarios for each quality attribute, and examples of refined scenarios. A typical agenda for this workshop is shown in Table 1.

Time	Activity
8:30 a.m.	Start workshop
	Welcome and introductions
	Quality attribute workshop overview
	Business drivers
	Architecture plans
10:00 a.m.	Scenario generation and prioritization
12:00 noon	Lunch break
1:00 p.m.	Scenario refinement
3:00 p.m.	Wrap up
	Review scenarios
	Action items
4:00 p.m.	End workshop

Table 1: Agenda for Scenario Generation Workshop

The workshop starts with a facilitation team presenting an overview of the QAW process including QAW activities (the gray ovals in Figure 1) and the results of these activities. A customer representative then describes the system's business drivers including business context for the system, architectural drivers (quality attributes that "shape" the architecture), critical requirements (quality attributes most central to the system's success), etc. The presentation of the business drivers is followed by an overview of the architecture plans (to the extent they might exist). The overview addresses technical constraints such as an operating system, hardware, or middle-ware prescribed for use, other systems with which the system will interact, and planned architectural approaches to address quality attribute requirements. These three presentations set the context for the activities to follow.

During the scenario generation segment of the workshop, individual stakeholders, in a round-table fashion, propose scenarios or ask questions about the way in which the architecture will respond to various situations. Scenarios are used to represent stakeholders' interests and quality attribute requirements. Scenarios should cover a range of anticipated uses of (use case

scenarios), anticipated changes (growth scenarios), and unanticipated stresses (exploratory scenarios) to the system.

Stakeholders are encouraged to generate as many scenarios as possible to represent a wide range of concerns. Questions to clarify scenarios are allowed but challenges or arguments about the importance of a scenario are discouraged at this point because the subsequent scenario prioritization takes care of this issue.

A good scenario makes clear what the stimulus is that causes it and what responses are of interest. For example, a use case scenario might be: *"Remote user requests a database report via the Web during peak period and receives it within 5 seconds."* A growth scenario might be: *"Add a new data server to reduce latency in scenario 1 to 2.5 seconds within 1 person-week,"* and a exploratory scenario might be: *"Half of the servers go down during normal operation without affecting overall system availability."* Scenarios should be as specific as possible, identifying stimuli, responses, and environment. Scenario generation continues until the allotted time is exhausted or until the stakeholders generate no additional scenarios. This activity takes between one and two hours. Typically, 30 to 40 scenarios are generated.

Only a small number of scenarios can be refined during a one-day workshop. Thus, stakeholders must prioritize the scenarios generated previously. The stakeholders use a voting process. Typically, they are given multiple votes (usually about 30% of the total number of scenarios). They can distribute those votes across multiple scenarios or apply all votes to a single scenario. Before the vote, the stakeholders are free to spend some time perusing the scenarios, discussing them in small groups, and combining scenarios that express equivalent concerns. This activity typically takes 30 minutes.

The top three or four scenarios are then refined by the stakeholders to provide a better understanding of their context and detail. For example, a scenario like "a communication relay node failed," does not really capture the consequences or implications of the failure. The refinement activity elicits further details such as the expected operational consequences, the system assets involved, the end-users involved, the potential affects of the scenario on system operation, and the exceptional circumstances that may arise. For the above scenario, such details would include which facility or node detects failure, what is the expected automated response to failure (if any), what is the expected manual intervention, which capabilities will be degraded during the outage, and the expected actions taken to return the relay to service. This activity typically takes two hours.

The result of this workshop is a prioritized list of scenarios and the refined description of the top three or four scenarios. Each description consists of a simple scenario statement and a list of bulleted items extending the statement.

2.2 Test Case Development

The objective of this activity is to transform each refined scenario from a statement and list of bullets into a well-documented test case. The test cases may combine and/or extend the scenarios, add assumptions and clarifications, define issues, and pose relevant questions.

A test case has a context section outlining the important aspects of the case, an issues and questions section stating the various architectural concerns, and a utility tree that summarizes the issues and questions.

2.2.1 Context

The context section describes the mission or activity, the assets involved, the geographical region, the operational setting, the players, etc. It includes a description over a time interval allowing the operation to play out. In a test case involving a communication relay node failure, for example, the test case may define

- the operation at the time of failure
- what happens immediately after, when the system is reacting to the failure
- degraded operation during the interval when repair is underway
- restoring the system to normal operation

2.2.2 Issues and Questions

The issues and questions section defines the issues implied by the context and proposes questions that connect these issues to quality attributes. For example the issue may be: “how failure is detected.” In this case, the questions might be:

- What subsystem detects the failure?
- How long does it take to detect the failure?
- What happens during this interval?

There should be from five to ten issues and questions for each context, covering various quality attributes. In the failure context, it is relatively straightforward to discuss

- performance issues, such as time to detect failure
- availability issues, such as degraded mode of services provided
- interoperability issues, such as how an alternative service might be introduced
- security issues, such as the impact on data integrity

Each test case should cover a few quality attributes, as well as assets, geography, missions, and activities. A checklist could be developed to aid in preparing the test cases to assure coverage across these topics.

2.2.3 Utility Tree

Each test case has a utility tree that links quality attributes first to specific attribute issues and then to specific questions (Table 2).

Root	Quality Attribute	Specific Attribute Issue	Question
Utility	Performance	... time to send a message...	... frequency of messages?
			... latency of messages?
			... priority of messages?
		... time to perform a critical function...	*... question?*
			... question?
		... time to restart a service...	*... question?*
			... question?
	Security	*... issue...*	*... question?*
			... question?
	attribute	*... issue...*	*... question?*
			... question?

Table 2: Utility Tree Format

2.3 Test Case Analysis

This activity requires the test cases and the architectural representations needed for the analysis of each test case. In some cases, the analyst must extend the architectural representations while performing the analysis. In the previous example, when “a communication relay node failed,” the analyst might build a sequence diagram showing the behavior of the major system components and the sequences of messages passing between them.

The analysis should be done at the highest reasonable level of abstraction, but should include components that make sense under the test case circumstances. In the example mentioned above, the analyst might respond in one of two ways:

1. The analyst shows a sequence diagram which includes an architectural component labeled “traffic manager” which send messages to other defined components to divert their message traffic, discusses the level of degradation of the network traffic during the failure, and also includes a “load shedding” component.

2. The analyst points to a component labeled “control center” and asserts that this node will contain a “traffic manager” and a “load shedder” which will take the appropriate actions.

Obviously the first case includes sufficient detail to evaluate the problem, whereas the second case is much more in the “trust me” category, which the method is trying to overcome.

A typical sequence of steps for a test case analysis would include the following:

1. Review the capabilities of the assets in the test case context and determine how the system will react to the situation.
2. Make and document any assumptions necessary to proceed with the analysis.
3. Determine which architectural views (for example operational, system, technical, process, behavioral, structural) can best describe how the system will address the issues and their associated questions.
4. If necessary refine the architecture to help answer the questions.
5. Document the answers as specifically as possible.

The analysis of a test case might identify specific quality attribute sensitivity points, tradeoffs, and risks, which in turn might lead to architectural changes. It is important to record not only risks, i.e., potentially problematic architectural decisions, but also non-risks, i.e., good decisions that avoid a potential quality attribute deficiency.

An example risk might be: *“Rules for writing business logic modules in the second tier of your three-tier architecture are not clearly articulated. This could result in replication of functionality thereby compromising modifiability of the third tier.”* An example non-risk might be: *“Assuming message arrival rates of once per second, a processing time of less than 30 ms, and the existence of one higher priority process, then a one-second soft deadline seems reasonable.”* Both risks and non-risks should be documented with specific architectural decisions, quality attribute requirements, and rationale.

2.4 Results Presentation Workshop

The results workshop presents the test case analysis. At the same time, it provides an opportunity for the architecture team to demonstrate that it completely understands the test cases, its architecture is able to handle these cases correctly, and its members have the competence required to continue analyzing test cases as part of the architecture development effort.

A typical workshop agenda is shown in Table 3.

Time	Activity
8:30 a.m.	Start workshop
	Welcome and introductions
	Quality attribute workshops overview
	Business drivers
	Architecture plans
	Purpose of workshop and expected outcomes
9:30 a.m.	Presentation of analysis of test cases #1 and #2
11:45 a.m.	Review of first two test cases and tailoring of afternoon objectives
12:00 noon	Lunch
1:00 p.m.	Presentation of analysis of two or more additional test cases
4:00 p.m.	Wrap-up
	Review of test cases
	Summary
	Review future activities
5:00 p.m.	End workshop

Table 3: Agenda for Presentation of Results Workshop

Notice that the beginning of the workshop recapitulates the QAW process, business drivers, and architectural plans that were presented during the scenario generation workshop. Since the presentation of results might take place a few months after the first workshop, this review serves to remind participants who were involved in the scenario generation and to introduce the concepts to new participants.

There are two formats for this workshop: a dry-run (rehearsal) workshop and a full-scale workshop. The dry-run workshop is often conducted when the architecture team is unsure about a number of issues, such as

- What level of detail is sufficient?
- How precise should the answers be?

- How can other analysis results (for example reliability, availability and maintainability analysis, or network loading analysis) be used?
- What additional architectural views should be built?

During the dry-run workshop the architecture team reports the results of the test cases analyses as they exist. At the conclusion of this workshop, there likely will be issues that should be resolved by further analysis, and the architecture team may need to modify the architecture to mitigate risks identified during the analysis.

The architecture team presents its final briefing at a full scale workshop. It takes place after "cleaning up" the results of the dry-run workshop. Risks that arise in the full scale workshop represent significant threats to the architecture.

In both formats of the workshop, the facilitators and stakeholders should probe architectural approaches from the point of view of specific quality attributes to identify additional risks. During the workshops, the participants identify the approaches that pertain to the highest priority quality attribute requirements, generate quality-attribute specific questions for each high priority quality attribute requirement, and identify and record additional risks and non-risks, sensitivity points, and tradeoffs.

3 Variants

We have described a generic process for performing a QAW. However customers have required variations, as outlined below:

In one case, a QAW was used in the competitive phase (three industry teams competing) of the acquisition of a large-scale Command, Control, Communications, Computer, Intelligence, Surveillance, Reconnaissance (C4ISR) system. In this instance, the QAW process was customized as follows:

- The scenario generation workshop was conducted with each contractor separately. As a result of the workshop, participants gained an understanding of the process, a list of prioritized scenarios, and refined high-priority scenarios.
- A government technical assessment team (TAT) used these scenarios to develop a number of test cases. Changes were made to hide the identity of the teams and extend the coverage of the scenarios over a set of assets, missions, and geographical regions. An example was developed to make the process more understandable and copies were distributed to all industry teams.
- The industry teams performed the analysis and presented the results in a dry-run presentation of a results workshop. There was a large variation in the presentations for these workshops, ranging from performing only one test case in great detail, to performing all test cases but in insufficient detail. Each industry team was then informed of how well it did and how it could improve its analysis.
- The industry teams then completed the analysis in a final presentation of the results workshop, allowing them to correct any flaws.

In another case, the customer only wished to generate, prioritize, and refine scenarios as an exercise for the stakeholders. Since the architectural plans were still uncertain, the customer intended to develop and analyze test cases later, after a draft architecture became available.

In yet another case, QAW was used in a pre-competitive phase for a large system. The architecture team (from various facilities) was building the architecture for a shared communications system. Stakeholders from the different facilities held separate workshops to generate, prioritize, and refine scenarios. The architecture team then turned these refined scenarios into test cases and analyzed the proposed architecture against them. The architecture team then presented the results of the analysis, first to a review team, and later, to the original stakeholders.

4 Experience and Conclusions

As a result of conducting a number of quality attribute workshops, we learned a number of lessons. Most of these lessons were integrated into the method incrementally:

We learned, for example, that the scenario generation workshop is a useful communications forum to familiarize stakeholders with the activities and requirements of other stakeholders. In several cases, the architecture team members were unaware of considerations brought up by those with responsibility for maintenance, operations, or acquisition.

We also learned that the facilitation team has to be flexible and adapt to the needs of the customer, as the following observations indicate:

- We originally developed the method to analyze *software* architectures. In the case of one customer, however, our initial efforts were directed towards a *system's* architecture early in its development stage, when the software was a "gleam in the eye" of developers and, hence, poorly represented. The generated scenarios emphasized operational conditions, missions, assets, and geographical regions. We adjusted the method and were able to generate and refine scenarios applicable to both software and systems architectures.
- The process of generating scenarios in the brainstorming sessions is usually inclusive, but the process for refining the high priority scenarios might not be. Some stakeholders might feel left out of the refining effort if other, more vocal stakeholders dominate the process. It is the responsibility of the facilitators to make sure that everyone can contribute.
- The approach relies on getting the right stakeholders to do some preparatory reading and attend the workshop for a day. In one case, the burden for attracting these stakeholders fell completely on the architecture team and this created some awkward situations. The team needed a process to attract the right people to the workshop. This could include forms for inviting stakeholders and explaining the advantages (carrots), and suggested approaches to upper management to cause interest in attending (sticks).
- The scenarios generated in a workshop can be checked against the requirements in two ways. First, unrefined scenarios can be sensitivity-checked against system requirements. Second, refined scenarios can lead to a better understanding of some requirements. Undocumented requirements can be discovered by both means.
- The workshop can open lines of communication between the architecture team and the stakeholders. In one case, potential critics of the project became advocates by virtue of seeing their concerns addressed through the QAW process.

- The scenarios can be checked against the expected evolution of the system over time, ensuring that the projected deployment of assets and capabilities match the scenarios and test cases.

In the second workshop, the architecture team presents the results of the analysis. However, questions from the stakeholders often lead to discussions of alternatives and quick “back of the envelope” analyses. In some cases, the analysis might reveal flaws in the architectures and cause the architecture team to change the design.

Like the first workshop, participants are provided with a handbook before the meeting. The handbook presented the results of the first workshop, including the complete list of scenarios, the priorities, and the refined high priority scenarios. The handbook includes the test cases and provides a test case analysis example so the participants know what to expect at the workshop. The following observations are derived from conducting a number of these workshops:

- In one case, the initial example given in the participants’ handbook was too general and did not contain enough detail. This reduced the level of “buy-in” from participants. We corrected this by developing another example: one that presented the right level of detail.
- The test cases generated by the QAW process often extend the existing system requirements. In one case, the new requirements seemed to challenge the requirements elicitation effort and raised concerns of the architecture team. A typical response was “the system wasn't meant to do that.” Some judgment must be made as to which test cases can be handled and at which phase of system deployment. While this topic can lead to extended arguments within the team, it is a useful exercise, since these concerns eventually must be resolved.
- In another case, the stakeholders were concerned because the process only analyzed a few test cases out of a large collection of scenarios. They wanted to know what was to be done with the remaining scenarios. This issue should be resolved in advance of the first QAW workshop. One approach is to analyze the architecture incrementally against an ever-expanding set of test cases and, if necessary, adjust the architecture in each increment. However, this approach is constrained by budgets, expert availability, and participants’ schedules.

In conclusion, the process for conducting quality attribute workshops is solidifying as we continue to hold them with additional customers, in different application domains, and at different levels of detail. The approach looks promising; the concept of checking for flaws in the architecture before committing to development should reduce re-work in building the system.

Appendix A Example Test Case

A.1 Scenario Refinement

A test case usually starts with a short scenario such as, “Mars orbital communications relay satellite fails.” The scenario is refined by generating a list of bulleted statements that further explain the circumstances associated with the scenario:

- one of three aero-stationary satellites fails
- reports failure to Earth control element
- Mars surface elements and Mars satellites know that it failed
- service assessment done at control center (two days)
- traffic re-routing is to be performed
- network reconfiguration dictated by flight director, perhaps postponed to limit possibility of further failure
- multiple authorities in multiple organizations and control centers
- well defined decision making process leading to mission director (final authority)
- multiple missions will be running simultaneously. This is a big deal. Currently, we are doing two, which is difficult and taxing. Coordination is complex.

A.2 Test Case Components

The refined scenario is now used to develop a test case.

A.2.1 Test Case Context and Activities

Humans and robotic missions are present in the Mars surface when one of three stationary-stationary satellites has a power amplifier failure. The primary communications payload is disabled for long-haul functions but the proximity link to other relay satellites and customers on orbit/surface still works. Secondary Telemetry and Tele-Command (TTC) for spacecraft health is still working for direct-to-earth with a low data rate. The remaining two satellites are fully functional. Communications with the crew has been interrupted. The crew is not in an emergency situation at the time of the failure, but reconnection is needed "promptly." The crew on the surface is concentrated in one area and the other missions in the Mars vicinity are in normal operations, non-emergencies, or performing mission-critical events. The event occurs late in the development of the communications network, so the system is well developed.

A.2.2 Stimulus

Detection of failure, i.e., power amplifier failed disabling the long-haul functions but the proximity link to other relay satellites and customers on orbit/surface is still working.

A.2.3 Quality Attribute Issues and Questions

The test case then lists a number of questions about the events to be answered by the analysis. To help focus the analysis, each question is tagged by the quality attribute it addresses: performance (P), security (S), availability (A), modifiability (M), and interoperability (I), and with a specific issue of concern within that quality attribute. This section would also include the utility tree associated with the test case.

1. **Issue**: Mission safety requires consistent and frequent communications between the crew and earth (P, A)
    a. Question: How long does it take to detect the failure?
    b. Question: How long does it take to reconfigure the system to minimize the time the crew is without communication?
2. **Issue**: System operation will be degraded (P, A)
    a. Question: Is there a way for the customer to simplify their procedures so they can handle a larger number of missions with less trouble than coordinating two as they do now?
    b. Question: What redundancy is required?

c. Question: Is there a way to send information about the degraded satellite back to Earth for analysis?

3. **Issue**: System recovery (P, A)

   a. Question: Can the crew participate in the repair?

   b. Question: Is there any expectation for a human interface between Mars and the Earth (e.g., crew in space station)?

   c. Question: Can the customer participate in the notification (e.g., “Please send a message to the other satellite”)?

A.2.4 Utility Tree

Root	Quality Attribute	Specific Attribute Issue	Question
Utility	Performance	... of communications...	(1b) how long to reconfigure?
		... degraded operation...	(2a) can decisions be simplified?
			(2c) how is information sent back?
	Availability	... mission safety...	(1a) How long to detect the failure?
		... redundancy...	(2b) What redundancy is required?
		... recovery...	(3a) can the crew help?
			(3b) can space station help?
			(3c) can other assets help

Table 4: Test Case Example Utility Tree

A.3 Analysis Results: Sensitivities, Tradeoffs, and Risks

The analysis of a test case might identify specific quality attribute sensitivity points, tradeoffs, and risks, which in turn might lead to architectural changes.

In our example, there are two sensitivity points for availability and crew safety: satellite locations and monitoring the health of the satellites.

A.3.1 Satellite Locations

The initial architecture has three operational aero-stationary satellites. Each satellite has visibility over a fixed area of Mars. When the satellite fails, its area of responsibility is no longer in communication with Earth and this area contains the crew.

A satellite failure risks having inadequate communications with the crew on Mars. This is a serious problem but can be alleviated by a number of different architectural approaches:

- Move one or two of the other stationary satellites to provide communications with the crew. This will degrade communications between the relocated satellites and the assets in their original area of responsibility. This solution constitutes a tradeoff between quality of communications in different areas of responsibility.
- Place one or more in-orbit spare satellites. A spare can be moved to the location of the failed satellite and take over its area of responsibility. This solution constitutes a tradeoff between cost (additional satellites) and speed of “repair” (the more spares, the quicker one could be moved to the desired location).
- Place “feeder antennas” on the surface of Mars to relay communications from the crew to (eventually) another satellite, i.e., rerouting the traffic in case of failure. This solution is a tradeoff between cost (the antennas) and the quality of the communications (the volume and latency of messages).
- Place the satellites in a “slow drift” orbit, such that the loss of a satellite will not cause a complete communication failure. There will be times when one or more of the other satellites will be in sight and there will be times where no satellite is in sight. This is probably the least disruptive solution, provided the periodic (but predictable) loss of communications can be tolerated. In case of a satellite failure, the crew will have communications when the next satellite drifts over the area.

A.3.2 Health of the Satellites

Monitoring the health of the satellites will improve the availability of the communications with the crew. A health monitor could help predict imminent failures, detect recent failures, and plan corrective actions.

Without a health monitoring system, there is a risk that satellites failures could go unnoticed until they are needed to provide communications to the crew on the surface. There is also an additional risk of having extended down-times due to the long transit time from Earth (e.g., sending a replacement satellite could take months). There are a number of alternatives, which are not mutually exclusive:

- The satellites could exchange periodic health messages among them (e.g., pushing “I am here!” messages and pulling “Are you there?” messages) and inform the ground stations of their health states. This is a tradeoff with performance because of the additional message traffic.
- The satellites could run self-tests and inform the ground stations whenever a problem is detected. This is a tradeoff with performance and probably cost (i.e., extra components to conduct the self-tests).
- The mission customers could notice degradation in communications and alert the communication system operators. This is a tradeoff with usability (e.g., how to avoid false alarms).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. **AGENCY USE ONLY** (leave blank)	2. **REPORT DATE** May 2001	3. **REPORT TYPE AND DATES COVERED** Final
4. **TITLE AND SUBTITLE** Quality Attribute Workshops		5. **FUNDING NUMBERS** C — F19628-00-C-0003
6. **AUTHOR(S)** Mario Barbacci, Robert Ellison, Judith Stafford, Charles Weinstock, William Wood		
7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		8. **PERFORMING ORGANIZATION REPORT NUMBER** CMU/SEI-2001-TR-010
9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116		10. **SPONSORING/MONITORING AGENCY REPORT NUMBER** ESC-TR-2001-010
11. **SUPPLEMENTARY NOTES**		
12.a **DISTRIBUTION/AVAILABILITY STATEMENT** Unclassified/Unlimited, DTIC, NTIS		12.b **DISTRIBUTION CODE**

13. **ABSTRACT** (maximum 200 words)

Quality attribute workshops (QAW) provide a method for evaluating the architecture of a software-intensive system during the acquisition phase of major programs. The architecture is evaluated against a number of critical quality attributes, such as availability, performance, security, interoperability, and modifiability. The evaluation is based on test cases that capture questions and concerns elicited from various stakeholders associated with the system. The process of eliciting questions allows stakeholders to communicate directly, thereby exposing assumptions that may not have surfaced during requirements capture. Our experience to date includes twelve quality attribute workshops that were held with three different U.S. Government acquisition programs. In this report, we provide a rationale for developing the process and describe it in detail. We follow this with a list of lessons learned and discuss how these lessons have helped us evolve the process to its current state.

14. **SUBJECT TERMS** Quality attributes, software system acquisition, software architecture, evaluation, attribute requirements			15. **NUMBER OF PAGES** 33 16. **PRICE CODE**
17. **SECURITY CLASSIFICATION OF REPORT** UNCLASSIFIED	18. **SECURITY CLASSIFICATION OF THIS PAGE** UNCLASSIFIED	19. **SECURITY CLASSIFICATION OF ABSTRACT** UNCLASSIFIED	20. **LIMITATION OF ABSTRACT** UL