MITRE's Architecture Quality Assessment

Richard F. Hilliard II
Michael J. Kurland
Steven D. Litvintchouk
The MITRE Corporation
rh | kurland | sdl @mitre.org
617 271 2724
617 271 8500 (fax)

ABSTRACT

MITRE's Architecture Quality Assessment (AQA) is intended to provide an objective and repeatable technique for the evaluation of system architectures. This paper describes the scope of AQA and its intended use; the evaluation methodology embodied in the AQA; the process of conducting an architecture assessment using the AQA; the current status of the AQA; and a comparison with other approaches to architectural assessment.

INTRODUCTION

Current interest in architecture is motivated by the objectives of building systems faster, better and cheaper (Horowitz, 1991). This has led to attempts to create a discipline of systems architecture exemplified by work such as (Rechtin, 1991) and (Maier, 1996), and by the growth of architectural methods such as (Kruchten, 1996) and (Emery, Hilliard, Rice, 1996) to support this discipline. Within large enterprises, such as the US Department of Defense, the promise of architecture has resulted in a move toward *architecture-based procurement* of defense systems. This is evidenced by the definition of enterprise-wide "building codes" and standards such as the Technical Architecture for Information Management (TAFIM) and the Joint Technical Architecture (JTA) and by the emergence of industry and formal architectural standards such as Open/X and the IEEE *Recommended Practice for Architectural Description* (Ellis *et al*., 1996).

A key aspect of making systems architecture a discipline is the means to evaluate and critique architectures. Criticism and independent assessment play a significant role in traditional building architecture in both the training of new architects and in the day-to-day conduct of the practice. MITRE, as system engineer for our sponsors, is increasingly called upon to specify or review a system's architecture as a part of system development. Since the architecture can have far-reaching implications throughout the life cycle, it is critical to be able to evaluate that architecture early. To that end, we have undertaken the development of the *Architecture Quality Assessment*.

Purpose. The purpose of Architecture Quality Assessment (AQA) is to provide a means to conduct architectural evaluations. The AQA is intended to provide an objective and repeatable basis for assessing architectural quality in its general and program-specific aspects. The results of an AQA are meant to be a direct input to the decision maker – such as a program executive officer (PEO) or program manager (PM) – for architecture-related issues on major system developments. The assessment can then be used to highlight potential risks from which to formulate risk reduction strategies, or as a basis for other programmatic decisions relating to acceptance of the architecture, program schedule, award fee or progress payments.

The AQA is designed to be applied in various ways; it may be used to: evaluate a candidate architecture; review the technical progress of an ongoing architecture development; assess a complete, delivered architecture prior to its acceptance and system implementation; or, compare two or more alternate architectures in a consistent fashion.

Scope. The scope of the AQA is the architecture-related activity of a major system program, which encompasses development, delivery, maintenance and evolution. We use the term “architecture activity” to emphasize that the AQA does not assume any particular life cycle model, life cycle phases, or acquisition policies. The architecture activity takes place in a context determined by the system requirements and other programmatic goals and constraints; the resulting architectural description is the primary input to the overall system design. It also influences other considerations, such as the development environment and approach. The architecture activity may be carried out by the Government or its agents (such as MITRE), by a contractor, or by a joint team. The AQA is equally applicable in any of these cases, making no assumption as to who is involved in the architecture activity.

The AQA is narrowly focused on architecture-related artifacts or deliverables. It is not intended to assess the capability of the architect or organization – only the resulting products. The AQA specifically excludes other concerns and sources of risk not related to architecture, including: the overall system development process, the capabilities of the developers to carry out that process, the validity and relevance of requirements which led to the architecture, and the resulting system design and implementation. These, and other considerations such as design quality and code quality, are already addressed by other techniques.

AQA Elements. The AQA consists of the following elements, which are described below:

- a conceptual foundation, or frame of reference, for talking about architectures, architectural descriptions and architectural methods. This frame of reference is used to pose questions about the architecture under review;
- the identification of architectural quality areas and factors;
- a set of measures to be used to assess an architecture against the identified quality factors;
- a methodology for applying the assessment in a repeatable fashion to address program-specific and general quality concerns;
- a technique for interpretation of assessment results and their correlation to architecture-related system and program risks.

ARCHITECTURAL FOUNDATIONS

Because architectural methods are in their infancy, there is a lack of standardized terminology and concepts applicable to architecture. Yet, for purposes of objective evaluation, we needed some uniform vocabulary to frame questions in the AQA. There are a number of popularly cited “architectural frameworks” such as the TAFIM, the JTA and the Zachman framework. Despite their popularity, we did not find in any of these works a suitable terminological basis on which to do solid engineering analyses. Therefore, for the purposes of architecture assessment, we adopted the terminology of an architecture metamodel, originally developed by MITRE for the Army’s Sustaining Base Information Services (Emery, Hilliard, Rice, 1996). MITRE’s metamodel, and related work by P. Kruchten (Rational), is now the starting point for a unified architectural metamodel being created as a part of the IEEE’s *Recommended Practice for Architectural Description*. A few of the terms are shown in figure 1.

1. An *architecture* is the highest-level concept of a system in its environment.
2. An architecture is documented as an *architectural description*.
3. An architectural description consists of one or more views.
4. Each *view* presents the system from a well-defined vantage point to address a specific set of concerns. Each view is documented in terms of a number of architectural elements.
5. An *architectural element* may be a component, connection or constraint.
6. A *component* depicts a major element of a view.
7. A *connection* depicts a relationship between components.
8. A *constraint* depicts a law which a component or connection must obey.

Figure 1. Some representative definitions

METHODOLOGY

The evaluation methodology used in the AQA follows that of MITRE’s Software Quality Assessment Exercises (SQAE). Specifically, AQA uses the SQAE’s three-layered framework of (1) quality areas, (2) factors and (3) measures to organize the assessment. We identified six *quality areas* which contribute to overall architectural quality (see figure 2).

Understandability. Is the architecture recorded such that it may be communicated to others? Architecture understandability is a prerequisite to each of the quality areas below – if an architecture cannot be understood, it cannot be readily implemented, maintained or evolved.
Feasibility. Does the architecture provide a sufficient basis upon which to develop a system which is an instance of that architecture and satisfies program needs? If the architecture does not exhibit feasibility, the system may not be readily or economically implemented; stakeholder needs may not be met. Feasibility is a prerequisite to the system's implementability, maintainability and evolvability.
Openness. Does the architecture yield a system which can operate in an open system environment? An open system is built to exploit public interfaces and services, exhibiting minimal dependence on proprietary systems.
Maintainability. Does the architecture yield a system which is maintainable?
Evolvability. Does the architecture exhibit a degree of changeability to meet new user or client needs, and may be adapted to new, unanticipated missions, while preserving the integrity of the original architecture? Evolvability is measured against the vision statement. Prerequisites: Feasibility, Openness and Maintainability.
Client Satisfaction. Does the architecture result in a system which meets its requirements in the context of its mission. These concerns are necessarily project- and system-specific?

Figure 2. Synopsis of Quality Areas

Each quality area is cross-classified by eight *quality factors* which contribute in varying weights to those areas: Conceptual Foundations, Architecture Documentation, Architecture Description, Analyzability, Emergent Properties, Patterns and Idioms, Quality of Description, and Management Context. (See figure 3.) Each factor is assessed with a number of *measures* – specific questions based on the architectural information available.

Conduct of an Assessment. The steps of an AQA are as follows:

1. Perform Needs Analysis (when one is not available).
2. Gather relevant documents and other artifacts related to the architecture.
3. Evaluate documentation against measures and score results.
4. Interpret results and identify architecture-related risks.
5. Document results for client.

Conceptual Foundations. Is the architecture being developed based upon a well-defined set of concepts or conceptual foundation?
Architecture Documentation. What is the quality of the architectural documentation set relative to the conception of that architecture and the anticipated usage of the documentation by the client, system developers, and others?
Architecture Description Techniques. What is the quality of the means (notations and conventions) by which architectural information is expressed within the architecture documentation?
Analyzability. Can the architecture description, as developed using the notations and techniques of the previous factor, be analyzed for consistency, completeness and other characteristics?
Emergent Properties. Does architecture address key client concerns ("ilities")?
Patterns and Idioms. Does the architecture exhibit regular properties addressing system concerns?
Quality of Description. Does architecture description exhibit: simplicity, descriptive consistency, and descriptive completeness?
Management Context. Is architecture documentation useful in the larger system life cycle?

Figure 3. Synopsis of Quality Factors

Needs Analysis. An AQA takes place in the context of the system- and program-specific concerns for that architecture. For the purposes of an objective evaluation, these concerns must be recorded in some consistent fashion, prior to the actual evaluation of architecture documentation. This is accomplished through a *needs analysis*. Needs analysis produces three work products: a needs repository, a goals statement and a vision statement. The needs repository contains needs of the system. Needs may be thought of as distilled, architecture-relevant system requirements. The goals statement captures the client's success criteria. The vision statement reflects the client's long-term direction for system and its evolution. The needs analysis is then used to: inventory system-specific items for assessment; determine relevance and applicability of measures (specific questions) to the system under assessment; and determine weighting, thresholds and prioritization of measures and factors for the system under assessment.

Architecture Documentation. To achieve AQA's goals of objectivity and repeatability, the assessment is limited to the actual architecture-related artifacts (architectural documentation, simulations, models and analyses) for the system. Undocumented architectural information, such as what might be learned from conversation or "hearsay," is not considered admissible by the AQA. Architectural documentation may take many forms – therefore the AQA does not presume or prescribe the form or content of architecture documents. At present, much of this information is likely to be in an informal graphical or textual medium. In the future, more formal means of capturing architectural expression may exist to facilitate these assessments.

The AQA evaluators need to gather the relevant architectural documentation for assessment of the system architecture. Ideally, the architectural documentation should be identified in concert with the architect, and should be focused on architectural information. Since there is no standard or required set of documents in which architectural information resides, this step is crucial in obtaining appropriate information for the AQA.

Evaluation and Scoring. When the AQA evaluators are sufficiently familiar with the architectural documentation, the actual assessment can begin. This involves evaluating each of over two hundred measures. Measures take the form of questions. There are two types of questions in the AQA: iterators and discretes. *Iterators* are questions whose answers take the form of a list. The results of an iterator are then typically used to instantiate one or more other questions. For example:

> What views are evident in the architecture description? (*List views*.)
> *For each view:*
> - How well does the view represent a well-defined purpose and an easily stated viewpoint?
> - How well does the view identify the major concerns it is intended to address?
> - How well does the view identify the major stakeholder classes it is intended to address?
> - How well does the view address the concerns of the stakeholder classes it is intended to address?
> - Does the view comprise elements of a few related element types?

To simplify the scoring, aggregation and interpretation of results, all questions (other than iterators) are *discretes*; i.e., responses to these questions range over a discrete set of values shown in table 1. For analysis purposes, each of the discretes is coded as a numerical value.

Table 1. AQA Scoring Regime

Value	Description
IDEAL	*Subject is explicitly addressed, and receives best possible treatment.*
GOOD	*Sound treatment of the subject relative to expectations.*
MARGINAL	*Treatment of subject meets minimal expectations*
UNACCEPTABLE	*Treatment of subject does not meet minimal expectations.*
INCOMPLETE	*Treatment of subject exhibits a level of detail which is insufficient to make a judgment.*
NON-APPLICABLE	*This measure is not applicable to the subject of interest.*

Interpretation of Results. Once all applicable questions have been answered, the results of individual measures are aggregated to yield scores for each quality factor. Using the weightings determined from the needs analysis, the quality area scores are then calculated using the factor inputs. At this stage, architecture-related risks may be identified.

Assessment Products. There are three primary products of the AQA review: an Executive Summary, a Detailed Evaluation and Interpretation of results, and a set of Open Issues and Questions. The Executive Summary provides an overview of the overall quality results and any outstanding issues, relative to the system's needs analysis. The Detailed Evaluation provides the scores at all levels of aggregation, the weightings used, and the identified risks. In some cases, clients want to take the results of the AQA back to the architect to provide feedback, or to highlight unresolved issues in the architecture. For this purpose, Open Issues and Questions are documented in a form that may be submitted to the architect.

RELATION TO OTHER WORK

There is at present considerable interest in architecture evaluation, particularly for software architectures. AT&T developed a checklist for the conduct of architecture reviews of telecommunications products (AT&T, 1991). Unlike AQA, it is domain-specific and is performed "interactively" with the system architects, rather than via architecture documentation. AT&T cites a savings of about ten percent of development costs from such reviews.

The Software Architecture Analysis Method (SAAM), uses a scenario-based approach to assess a system's desired operational and evolutionary quality attributes (Kazman, Abowd, Bass, Clements, 1996). Unlike AQA, SAAM does not assume that an architectural description exists – developing that description is the initial activity of the analysis. SAMM takes place against a set of user and other stakeholder scenarios defined for each system –rather than against a predefined set of measures. Weighting of scenarios and overall ranking is determined *after* evaluation through a process involving all stakeholders. This contrasts with the AQA approach which documents stakeholder priorities *before* evaluation through the needs analysis.

Initial attempts to define architectural metrics (Lung, Kalaichelvan, 1996) are limited to software architectures and are not yet general enough for our purposes, presuming as they do the nature of the computational constructs to be counted.

STATUS

AQA development was begun in February 1996 under special corporate funding. Between February and August, we developed a "Proof of Concept" version of the AQA, and subjected it to trial use on the Air Mobility Command (AMC) Information Processing System (IPS). Portions of the AQA have been applied on: FAA's Standard Terminal Automation Replacement Systems (STARS), National Missile Defense, and Theater Battle Management Core Systems. The AQA has been adopted for use by the Chief Architect's Council of the Air Force PEO for Battle Management portfolio.

The development of the AQA was carried out by a small team in consultation with systems engineering specialists throughout MITRE. These specialists contributed specific measures in specialized areas of architecture, such as communications, reliability and availability, safety, and security. AQA development was conducted on MITRE's "intranet" to facilitate the team's communication and coordination and permit frequent peer review.

Our initial experience with the AQA appears to validate our basic approach. Through trial use, we were able to refine and simplify our set of measures. However, we do need to improve our techniques for scoring and aggregating measures into complete results. Our experience with AMC IPS also confirmed something else we had long suspected: better techniques are needed to help MITRE systems engineers absorb and correlate the tremendous volume of documentation and other information generated by a major DoD procurement and there is a good opportunity for automation in this regard.

Future Work. With a prototype version of the AQA now complete, we plan to employ it on additional projects. Experience gained on these projects will help us increase the depth and breadth of architectural coverage. As noted earlier, we have begun to investigate automated support for the AQA process and for the information management problems implied by its use, including automated techniques for architectural description and a repository for the results of assessments.

Acknowledgments. The authors which to thank Tim Rice and Steve Schwarm for their contributions, Ed Green and John Maurer for their support, and especially Don Neuman for his encouragement and sponsorship of this project.

REFERENCES

AT&T Bell Laboratories. *Best Current practices: software architecture validation*, 1991.

Walter J. Ellis, Richard F. Hilliard, Peter T. Poon, David Rayford, Thomas F. Saunders, Basil Sherlund, and Ronald L. Wade. Toward a recommended practice for architectural description. In *Proceedings of 2nd IEEE International Conference on Engineering of Complex Computer Systems*, Montreal, Quebec, Canada, October 21-25, 1996.

David E. Emery, Richard F. Hilliard, and Timothy B. Rice. *Experiences applying a practical architectural method*. In Alfred Strohmeier, editor, *Reliable Software Technologies – Ada-Europe'96*, number 1088 in Lecture Notes in Computer Science. Springer, 1996.

Richard F. Hilliard, Michael J. Kurland, Steven D. Litvintchouk, Timothy B. Rice, and Stephen C. Schwarm. *Architecture quality assessment*. Not currently in the public domain. **Now available from MITRE or from Rich Hilliard.**

Barry M. Horowitz. *The importance of architecture in DOD software*. Technical Report M91-35, The MITRE Corporation, July 1991.

Rick Kazman, Gregory Abowd, Len Bass, and Paul Clements. Scenario-based analysis of software architecture. *IEEE Software*, **29**(11): 47-55, November 1996.

Philippe B. Kruchten. The 4+1 view model of architecture. *IEEE Software*, **28**(11):42-50, November 1995.

Chung-Horng Lung and Kalai Kalaichelvan, Metrics for software architecture analysis: a case study in telecommunications. Submitted to ICSE'97.

Mark W. Maier. System architecture: An emergent discipline? In *1996 IEEE Aerospace Applications Conference Proceedings*, **3**: 231-246, 1996.

Eberhardt Rechtin. *Systems architecting: creating and building complex systems*. Prentice-Hall, 1991.