

Integrating Analysis and Design Methods for the Software Life Cycle

Mario R. Barbacci

In the last issue of The Architect, “Rethinking the Software Life Cycle” (Volume 6, Number 3, Third Quarter 2003), the authors described architecture-centric analysis and design methods that have been emerging for a number of years. A number of these methods have been described in previously in The Architect. The methods share a set of common characteristics and activities that suggest that the methods could be used in combination. In some cases activities are redundant; if this duplication is eliminated, using the methods in sequence can save cost and time. This column continues the theme and illustrates examples of combinations of the methods and when the combinations could be applied during the life cycle.

Most of these methods have a precise description of the documents and process required, and usually the process is fairly rigid, not allowing for changes or deviations. We are now conducting an integration study, looking at how the various methods could be integrated and at the benefits of the integration. There are several technical reports to be published soon. This column is an invitation to readers to send comments and to check the SEI Web site at http://www.sei.cmu.edu/ata/ata_eval.html for new publications.

The Methods

Quality Attribute Workshop (QAW) [1] collects and organizes software quality attribute requirements. The QAW collects, prioritizes, and refines scenarios that can be used to test if the architecture will meet the requirements. The analysis proper is not part of the QAW process but can be performed as a step in some of the other methods. The inputs, outputs, and activities in QAW are illustrated in Figure 1.

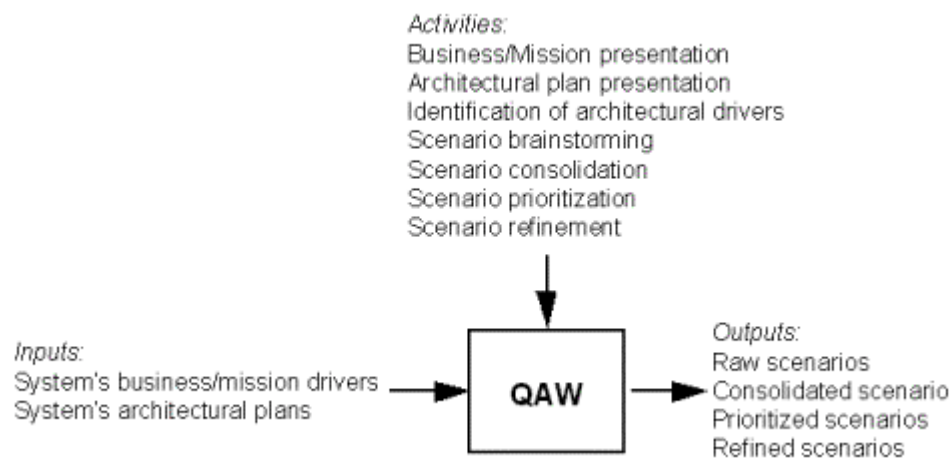


Figure 1: QAW inputs, outputs, and activities

Attribute-Driven Design (ADD) [2] defines a design process on the quality attributes the software must fulfill. ADD documents a software architecture in a number of views and depends on understanding the system's constraints and requirements, represented as scenarios. The ADD inputs, outputs, and activities are illustrated in Figure 2.

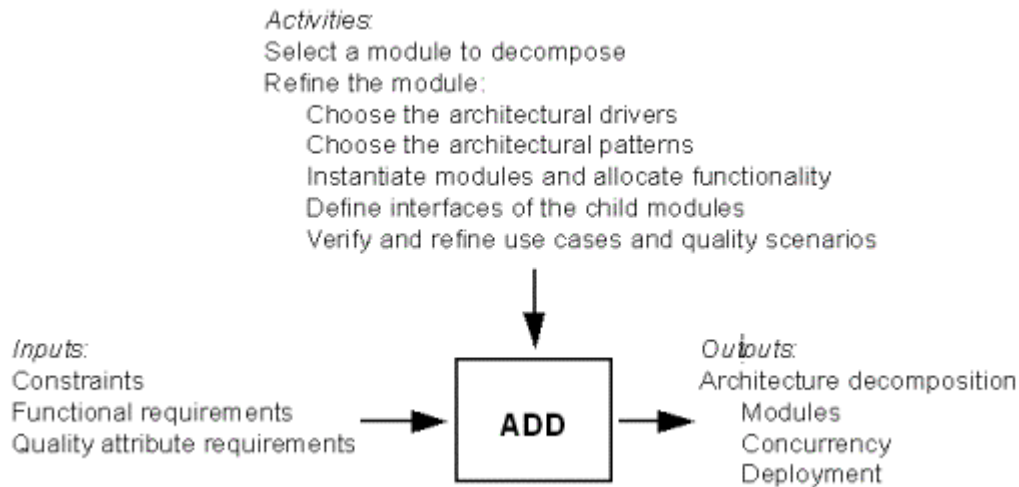


Figure 2: ADD inputs, outputs, and activities

Architecture Tradeoff Analysis Method (ATAM) [3] illuminates the consequences of architectural decisions with respect to quality attribute requirements. These consequences are illustrated in a set of risks, sensitivities, and tradeoffs. The ATAM inputs, outputs, and activities are illustrated in Figure 3.

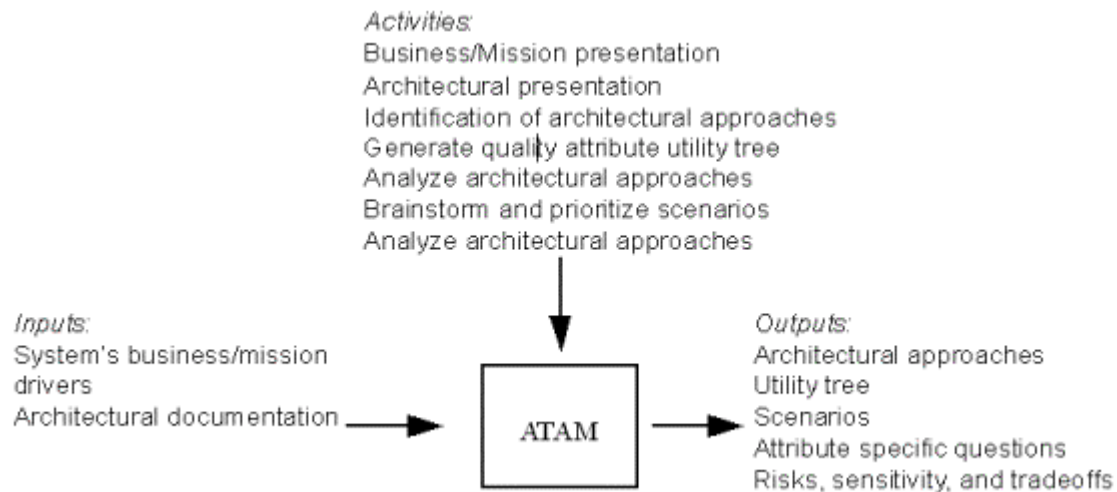


Figure 3: ATAM inputs, outputs, and activities

Active Reviews for Intermediate Designs (ARID) [4] blends active design reviews with ATAM into a technique for investigating designs that are practically complete. Like ATAM, ARID engages the stakeholders to create a set of scenarios that are used to test whether the design can be used by the software engineers who must work on it. The ARID inputs, outputs, and activities are illustrated in Figure 4.

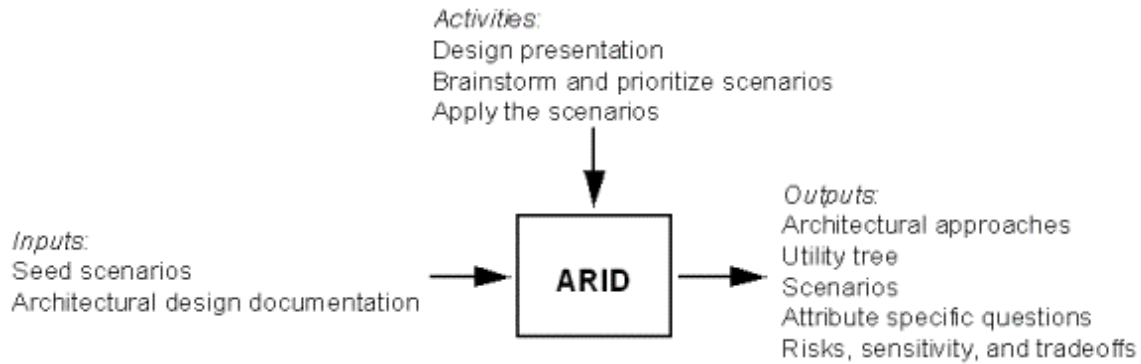


Figure 4: ARID inputs, outputs, and activities

Cost-Benefit Analysis Method (CBAM) [5] facilitates architecture-based economic analysis of software-intensive systems. This method helps the system stakeholders to choose among architectural alternatives during the design or maintenance phases. The CBAM inputs, outputs, and activities are illustrated in Figure 5.

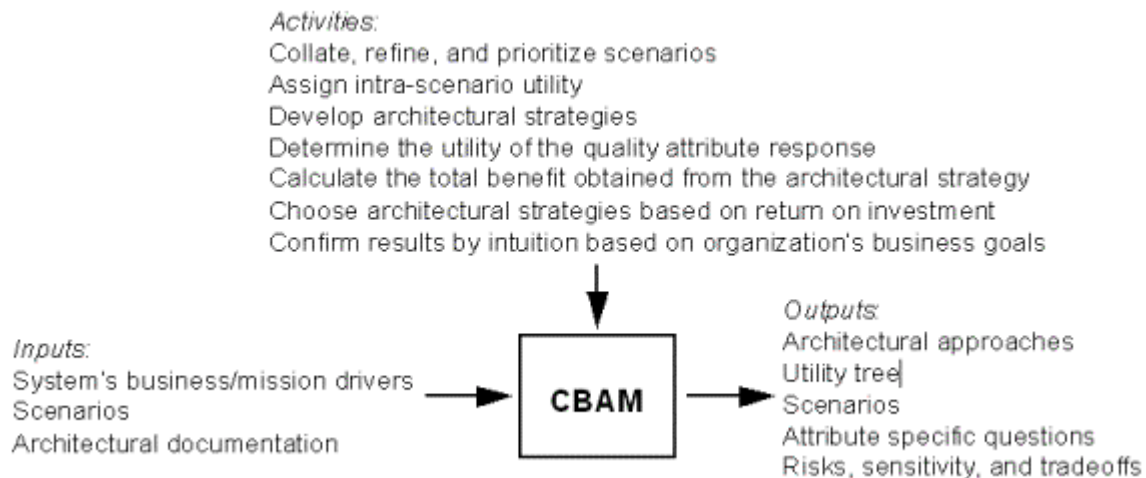


Figure 5: CBAM inputs, outputs, and activities

Combinations of Methods

Depending on the stage of the life cycle, the methods can be combined by taking the results of a method and using them as input to another method, perhaps eliminating redundant activities. Table 1 shows the methods and activities, and notes which artifacts are inputs to the method, outputs from the method, or both. Figure 6 shows a sequence of applications of the methods.

Table 1: *Methods and Life-Cycle Stages*

Life-Cycle Stage	QAW	ADD	ATAM	CBAM	ARID
Business needs and constraints	Input	Input	Input	Input	
Requirements	Input; output	Input	Input; output	Input; output	
Architecture design		Output	Input; output	Input; output	Input
Detailed design					Input; output
Implementation					
Testing					
Deployment					
Maintenance				Input; output	

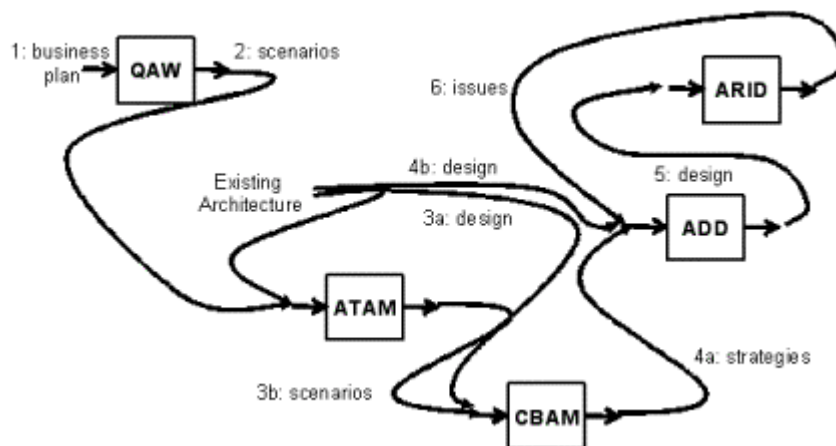


Figure 6: *Combination of Methods*

References

1. 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.
<<http://www.sei.cmu.edu/publications/documents/03.reports/03tr016.html>>.
2. Bachmann, F.; Bass, L.; Chastek, G.; Donohoe, P.; & Peruzzi, F. *The Architecture Based Design Method* (CMU/SEI-2000-TR-001, ADA37581). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.
<<http://www.sei.cmu.edu/publications/documents/00.reports/00tr001.html>>.
3. Kazman, R.; Klein, M.; Clements, R. *ATAM: A Method for Architecture Evaluation* (CMU/SEI-2000-TR-004, ADA382629). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.
<<http://www.sei.cmu.edu/publications/documents/00.reports/00tr004.html>>.
4. Clements, P. *Active Reviews for Intermediate Designs* (CMU/SEI-2000-TN-009, ADA383775) Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.
<<http://www.sei.cmu.edu/publications/documents/00.reports/00tn009.html>>.
5. Kazman, R.; Asundi, J.; & Klein, M. *Making Architecture Design Decisions: An Economic Approach* (CMU/SEI-2002-TR-035, ADA408740). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002.
<<http://www.sei.cmu.edu/publications/documents/02.reports/02tr035.html>>

About the Author

Mario Barbacci is a Senior Member of the staff at the Software Engineering Institute (SEI) at Carnegie Mellon University. He was one of the founders of the SEI, where he has served in several technical and managerial positions, including Project Leader (Distributed Systems), Program Director (Real-time Distributed Systems, Product Attribute Engineering), and Associate Director (Technology Exploration Department). Prior to joining the SEI, he was a member of the faculty in the School of Computer Science at Carnegie Mellon University.

His current research interests are in the areas of software architecture and distributed systems. He has written numerous books, articles, and technical reports and has contributed to books and encyclopedias on subjects of technical interest.

Barbacci is a Fellow of the Institute of Electrical and Electronic Engineers (IEEE) and the IEEE Computer Society, a member of the Association for Computing Machinery (ACM), and a member of Sigma Xi. He was the founding chairman of the International Federation for Information Processing (IFIP)

Working Group 10.2 (Computer Descriptions and Tools) and has served as chair of the Joint IEEE Computer Society/ACM Steering Committee for the Establishment of Software Engineering as a Profession (1993-1995), President of the IEEE Computer Society (1996), and IEEE Division V Director (1998-1999).

Barbacci is the recipient of several IEEE Computer Society Outstanding Contribution Certificates, the ACM Recognition of Service Award, and the IFIP Silver Core Award. He received bachelor's and engineer's degrees in electrical engineering from the Universidad Nacional de Ingenieria, Lima, Peru, and a doctorate in computer science from Carnegie Mellon.

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.