Software Engineering Institute
Carnegie Mellon University

GUIDE TO SOFTWARE ARCHITECTURE TOOLS

Tools and Methods for Analyzing the Architecture

System Analysis

During its research projects, the Software Engineering Institute has developed several tools for system design, analysis, and validation. Among them several tools were designed for analyzing performance criteria, such as latency or bus load. Other analyses are specific to the avionics domain, such as the ARINC653 validation framework that aims at validating system properties related to avionics system (space isolation across partitions, validation of system configuration, analysis of partition communication policy, etc.).

Safety Analysis

Recent focus of the SEI work has been on tools for analyzing system safety in support of industry practice standards (such as SAE ARP4761). Support includes Functional Hazard Assessment (FHA), Failure Mode and Effect Analysis (FMEA), Fault Tree Analysis (FTA), stochastic Dependency Diagram (DD) or Reliability Block Diagram (RBD), and Markov Chain analysis. Automation of these currently largely manual practices allow for repeated analysis and trade studies of design alternatives.

Open Source AADL Tool Environment (OSATE)

The Open Source AADL Tool Environment is an Eclipse-based modeling framework for using AADL. It brings AADL support within the Eclipse environment so that architecture practitioners can write their models using the AADL textual syntax. Users can also visualize their model using the graphical notation. OSATE currently supports the latest revision of the language and integrates several validation and analysis plug-ins. The Software Engineering Institute has developed several of them for analyzing system security, performance, or safety. These plug-ins are available under the Eclipse Public License (EPL); their source code is available in the Github repository.

In addition, OSATE supports the Error-Model Annex of AADLv2 for specifying architecture faults and errors. Engineers can specify error occurrence and propagation in their architecture using the textual notation of the language. OSATE includes several functions for processing this additional information and generating validation materials required by validation standards, such as:

- Functional Hazard Assessment: description of faults that occurs in each system function
- Fault-Tree Analysis: hierarchical dependencies between faults occurrence within the architecture. OSATE currently supports the generation of FTA that targets commercial and open-source tools.
- Markov Chain Model: analysis of fault-occurrence according to system specification. OSATE actually supports the PRISM tool for Markov-Chain analysis.

OSATE Release Process

OSATE is distributed in two releases: the stable and testing releases. A stable release is published every 3 months. The testing release is updated in a daily build and contains experimental features.

AADL/OSATE Community

OSATE is an open-source project; its code is released under the Eclipse Public License (EPL). The AADL community has built a number of tools on top of OSATE.

Much of the work has been published and can be found at https://wiki.sei.cmu.edu/aadl/index.php/AADL_Related_Publications.

AADL Examples

We have made available a set of example AADL models on our public Github area. This repository hosts models grouped into separate AADL projects that can be imported within your OSATE workspace to try, test, and experiment with OSATE. Also, this is a good approach to have some AADL examples and learn how to use modeling patterns.

Tools and Methods for Establishing Requirements

Do you want to…	***Then look at…***
Identify important quality attributes and clarify system requirements before there is a software architecture to evaluate?	Quality Attribute Workshop
Bring together system of systems (SoS) stakeholders to augment existing mission threads with quality attribute considerations that will shape the SoS architecture and identify SoS architectural challenges?	Mission Thread Workshop

Tools and Methods for Evaluating the Architecture

Do you want to…	***Then look at…***
Evaluate and improve software architectures relative to quality attribute goals?	Architecture Improvement Workshop
Apply the leading method for evaluating a software architecture?	Architecture Tradeoff Analysis Method
Use mission threads to evaluate system and software architectures?	System Architecture Tradeoff Analysis Method
Provide an initial identification of SoS architectural risks and quality attribute inconsistencies across the constituent systems?	System of Systems (SoS) Architecture Evaluation
Find out if a software architecture for a system is suitable without having to build it first?	Active Reviews for Intermediate Design

How do you know if a software architecture is deficient or at risk relative to its target system qualities?

The answer is to conduct an evaluation of it. A formal software architecture evaluation should be a standard part of the architecture-based software development life cycle. Architecture evaluation is a cost-effective way of mitigating the substantial risks associated with this highly important artifact.

The achievement of a software system's quality attributes depends much more on the software architecture than on code-related issues such as language choice, fine-grained design, algorithms, data structures, testing, and so forth. Most complex software systems are required to be modifiable and have good performance. They may also need to be secure, interoperable, portable, and reliable. But for any particular system, what precisely do these quality attributes—modifiability, security, performance, reliability—mean? Can a system be analyzed to determine these desired qualities? How soon can such an analysis occur? What happens when these quality attributes are in conflict with each other? How can the tradeoffs be examined, analyzed, and captured?

The tools and methods in the table above can be used alone or in combination to obtain early and continuous benefits to any software development project.

Tools and Methods for Defining the Architecture

Do you want to…	***Then look at…***
Design the software architecture of a software-reliant system?	Attribute-Driven Design Method
Evaluate and improve software architectures relative to quality attribute goals?	Architecture Improvement Workshop

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu | www.cert.org
Email: info@sei.cmu.edu