Integrated Safety and Security Engineering for Mission-Critical Systems

Problem

Software increasingly dominates safety- and mission-critical system development. Issues are discovered long after they are created.

Solutions

Our three-year project aims to make systems safer and more secure by enabling early discovery of system-level issues through virtual integration and incremental analytical assurance. This project consists of four efforts, all of which use the Architecture Analysis and Design Language (AADL), an SEI-created, internationally standardized language for designing software-centric critical systems.

Security Requirements

A new security annex to AADL and verification plugins

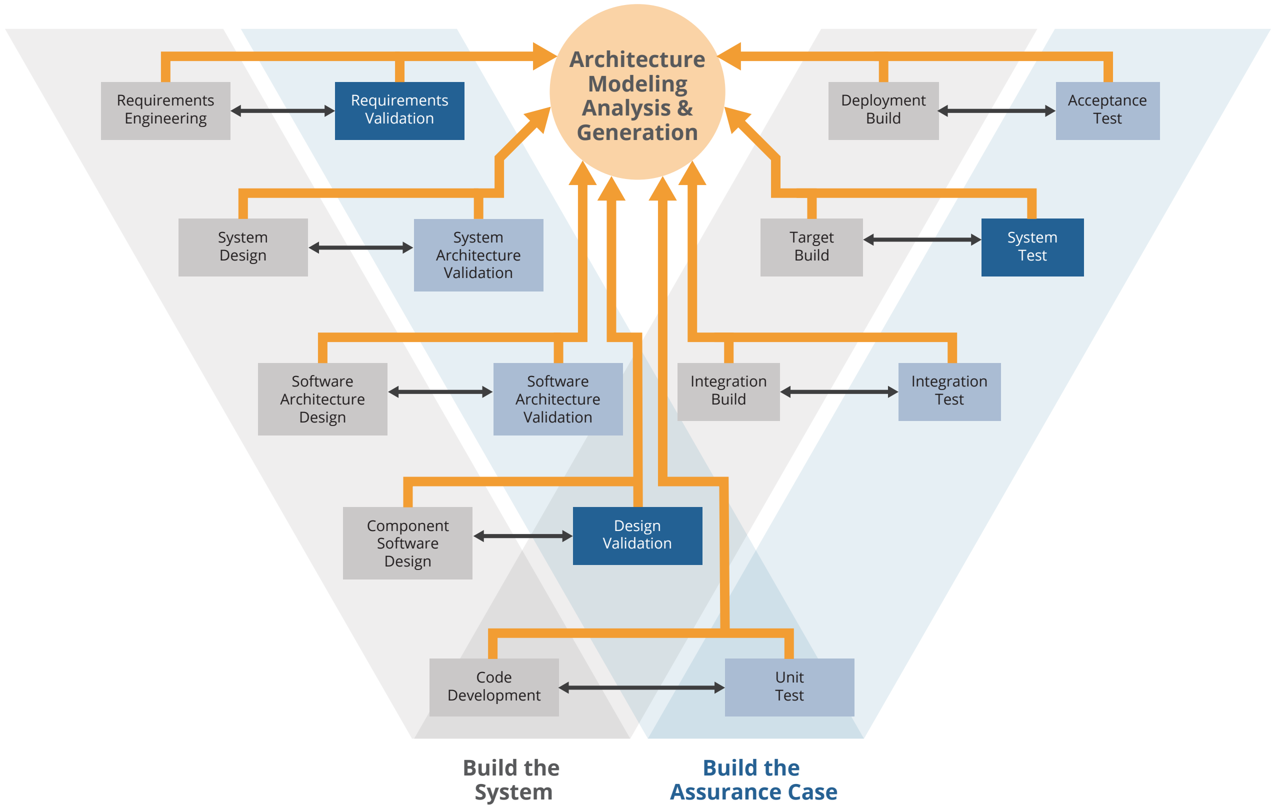
We developed an extension to AADL that enables system designers to describe how their system meets security goals by, for example, encrypting information or dealing with private keys. We also developed tools to verify that a system conforms to various policies, and we are publishing papers and documentation on how to use them.

Reusable Safety Patterns

A collection of patterns expressed using AADL

We proposed a library of safety design patterns that capture key safety architecture fragments. Each pattern is described using AADL, complemented by a machine-readable description of applicable error scenarios, a behavioral description of the nominal case, and a verification plan defined using custom tooling and AGREE / Resolute (tooling developed by Collins Aerospace). These formalizations are AADL implementations of existing patterns, and they equip system architects with modeling techniques and verification methods that are adaptable to various domains.

We're making it easier to **specify, design,** and **assure** critical systems that are safer and more secure.



Safety and Security Across the System Development Lifecycle

Architecture-Supported Audit Processor

A collection of system viewpoints for certification authorities

Performing a hazard analysis is a common way of examining a system for safety or security issues. This effort integrates a number of sources of system information—system architecture, error behavior, Kansas State's AWAS technology, and more—into a set of dynamic reports. The Architecture-Supported Audit Processor (ASAP) will allow system analysts to query interesting portions of a system's architecture interactively, rather than read only what an analysis format specifies.

[Off-]Nominal Behavior

Unified behavioral description

There are several ways to specify behavior in AADL, depending on what is being specified: (nominal) component behavior, off-nominal (i.e., erroneous) behavior, or mode-transition semantics. We produced a proposal to unify behavior specifications, which will make the language simpler and enable more powerful automated analyses.

AADL has been used in a variety of safety-critical domains, including medical devices, automotive components, and military and commercial aviation.

Carnegie Mellon University
Software Engineering Institute

Sam Procter, Peter Feiler,Dave Gluch, Aaron Greenhouse, Jerome Hugues, Lutz Wrage, Joe Seibel (info@sei.cmu.edu)
Kansas State University: John Hatcliff, Eugene Vasserman, Robby, Hari Thiagarajan, Jason Belt