Notes:

The Architecture Analysis & Design Language (AADL): An Introduction

Peter Feiler, David Gluch, John Hudak

2.1 Abstraction of Components

Component has:

* Identity (unique name and runtime essence)
* Interfaces with other components
* Distinguishing properties
* Subcomponent definitions
* Subcomponent interactions

Component Categories:

* Application software – thread, thread group, process, data, subprogram
* Hardware – processor, memory, device, bus
* Composite – system

Runtime Semantics for Interactions:

* Message passing
* Event passing
* Synchronized access to shared components
* Thread scheduling protocols
* Timing requirements
* Remote procedure calls

2.2 Architectural Analysis

Execution semantics for threads:

* Periodic
* Aperiodic (event-driven)
* Background (dispatched once and run to completion)
* Sporadic (bounded by a maximum rate)

Properties can be added to components, ports, and connections to support different kinds of analysis.

Interface structures for specifying interactions:

* Data ports for unqueued state data
* Event data ports for queued message data
* Event ports for asynchronous events
* Synchronous subprogram calls
* Explicit access to data components

Connections explicitly specify interactions. They can be immediate (midframe) or delayed (phase-delayed). Data transfers are assumed to be deterministic. The sampling rate of a receiving thread is always constant.

Properties:

* Period
* WCET
* Deadlines
* Space requirements
* Arrival rates
* Characteristics of data and event streams
* Source code and data which implement the specified component
* Constraints for binding threads to processors, source code, and data onto memory (or to a specific processor or memory type). Constraints can also prevent colocation.

Questions so far:

1. What is a runtime essence?

**AADL Models of Fault Tolerant Systems (Behavior Annex)**

Lasnier, G.; Robert, T.; Pautet, L.; Kordon, F.; , "Behavioral modular description of fault tolerant distributed systems with AADL Behavioral Annex," *New Technologies of Distributed Systems (NOTERE), 2010 10th Annual International Conference on* , vol., no., pp.17-24, May 31 2010-June 2 2010  
doi: 10.1109/NOTERE.2010.5536853  
URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5536853&isnumber=5536542>

AADL-BA (Behavior Annex) – automata-based AADL extension describing component behavior

The authors give an example, modeling primary backup replication (PBR) using AADL-BA. In PBR, N backup replicas store the execution state updates from the primary replica, but do not run unless the primary crashes and a new primary is elected. The new primary then starts up from the most recent execution context update sent by the original primary, and the networked application continues to function. They give behavioral models for three functions:

1. The replica controller (states: primary, backup, and election).
2. Thread behavior – rendezvous synchronization on dispatch events.
3. Heartbeat timeout behavior (starting from dispatch and ending on the task deadline).

**Modeling Faults within AADL (Error Model Annex)**

[**Automatic Generation of Fault Trees from AADL Models**](http://www-users.cs.umn.edu/%7Eajoshi/AeroSE07-JoshiBinnsVestal.pdf), Anjali Joshi, Pam Binns, and Steve Vestal. In ICSE Workshop on Aerospace Software Engineering, Minneapolis 2007.

Formal methods work in this area thus far fails to take into account the hardware architecture and its potential failure modes, as most of the existing work focuses on software behavioral models.

The Error Model annex supports the description of error states for AADL components, the definition of their occurrence patterns, the description of their propagation through the system, and the association of those error states and transitions with components. These take the form of associations, guards, occurrence patterns, and responses. In the example model the associations describe how the error state of the corresponding hardware and the arrival of error event propagations affect the error state of the software component.

Finally, they describe the generation process required to transform AADL error models into fault tree models.

[**Behavioral Fault Modeling for Model-based Safety Analysis**](http://www-users.cs.umn.edu/%7Eajoshi/Joshi-BehavioralFaultModel.pdf), Anjali Joshi, and Mats P.E. Heimdahl. Proceedings of 10th IEEE High Assurance Systems Engineering Symposium (HASE'07), Dallas, November 2007.

How can we model the particular fault behaviors without incorporating them into the original model? Faults should be specified separately from the nominal model behavior, and composed to model particular fault scenarios.

They propose an extension to Lustre (*LustreFM*) for handling domain-specific fault modeling extensions. Their example is good, it covers a wheel brake system to provide braking pressure for aircraft wheels. They have a concrete requirement spec as well, that the loss of all wheel braking shall be less probably than 5e-7 per flight. The braking system is hydraulic and dual-redundant, with failover in the case of primary brake system failure.