SAE AADL: An Industry Standard for Embedded Systems Engineering

Peter Feiler
Software Engineering Institute
phf@sei.cmu.edu
412-268-7790
SAE AADL Standard
An Enabler of Predictable Model-Based System Engineering

• Notation for specification of task and communication architectures of Real-time, Embedded, Fault-tolerant, Secure, Safety-critical, Software-intensive systems

• Fields of application: Avionics, Automotive, Aerospace, Autonomous systems, …

• Based on 15 Years of DARPA funded technologies

• Standard approved by SAE in Sept 2004

• www.aadl.info
SAE AS-2C AADL Subcommittee

- Bruce Lewis (US Army AMRDEC): Chair
- Peter Feiler (SEI): technical lead, author & editor
- Steve Vestal (Honeywell): co-author
- Ed Colbert (USC): UML Profile of AADL
- Joyce Tokar (Pyrrhus Software): Ada & C Annex

Other Voting Members
- Boeing, Rockwell, Honeywell, Lockheed Martin, Raytheon, Smith Industries, General Dynamics, Airbus, Axlog, European Space Agency, TNI, Dassault, EADS, High Integrity Solutions

Coordination with
- NATO Aviation, NATO Plug and Play, French Government COTRE, SAE AS-1 Weapons Plug and Play, OMG UML & SysML
Potential Users

- Airbus
- European Space Agency
- Rockwell Collins
- Lockheed Martin
- Smith Industries
- Raytheon
- Boeing FCS
- Common Missile
- System Plug and Play

New System Engineering Approach incorporates AADL
Modeling of Satellite Systems, Architecture Verification - ASSERT
Modeling of Avionics Computer System
Embedded System Engineering & AADL
Apply AADL for systems integration modeling & analysis
NATO/SAE AS1 Weapon System Integration
AADL-Based Engineering

System Analysis
- Schedulability
- Performance
- Reliability
- Fault Tolerance
- Dynamic Configurability

System Integration
- Runtime System Generation
- Application Composition
- System Configuration

Software System Engineer

Architecture Modeling
- Abstract, but Precise

Application Software

Execution Platform

Composable Components

Guidance & Control

Supply

Ambulate & Signal Processing

Information Fusion

Hardware

GPS DB HTTPS Ada Runtime

Devices Memory Bus Processor

www.aadl.info
A Partitioned Portable Architecture

Strong Partitioning
- Timing Protection
- OS Call Restrictions
- Memory Protection

Interoperability/Portability
- Tailored Runtime Executive
- Standard RTOS API
- Application Components
MetaH: Proof of Concepts for AADL

1991 DARPA DSSA program begins
1992 Partitioned PFP target (Tartan MAR/i960MC)
1994 Multi-processor target (VME i960MC)
1995 Slack stealing scheduler
1998 Portable Ada 95 and POSIX middleware configurations
1998 Extensibility through MetaH-ACME Mapping
1998 Reliability modeling extension
1999 Hybrid automata verification of core middleware modules

Numerous evaluation and demonstration projects, e.g.
- Missile G&C reference architecture, demos, others (AMCOM SED)
- Hybrid automata formal verification (AFOSR, Honeywell)
- Missile defense (Boeing)
- Fighter guidance SW fault tolerance (DARPA, CMU, Lockheed-Martin)
- Incremental Upgrade of Legacy Systems (AFRL, Boeing, Honeywell)
- Comanche study (AMCOM, Comanche PO, Boeing, Honeywell)
- Tactical Mobile Robotics (DARPA, Honeywell, Georgia Tech)
- Advanced Intercept Technology CWE (BMDO, MaxTech)
- Adaptive Computer Systems (DARPA, Honeywell)
- Avionics System Performance Management (AFRL, Honeywell)
- Ada Software Integrated Development/Verification (AFRL, Honeywell)
- FMS reference architecture (Honeywell)
- JSF vehicle control (Honeywell)
- IFMU reengineering (Honeywell)
AADL in Context

Research ADLs

- **MetaH**
  - Real-time, modal, system family
  - Analysis & generation
  - RMA based scheduling

- **Rapide, Wright, ..**
  - Behavioral validation

- **ADL Interchange**
  - ACME

Industrial Strength

- **UML 2.0, UML-RT**
- **HOOD/STOOD**
- **SDL**

**Basis**

**Extension**

**Influence**

**Alignment**

**Enhancement**

**DARPA Funded Research since 1990**

**AADL Extensible Real-time Dependable**

**UML Profile**

**Airbus & ESA**

© 2004 by Carnegie Mellon University
AADL: The Language

Components with precise semantics
  – Thread, thread group, process, system, processor, device, memory, bus, data, subprogram

Completely defined interfaces & interactions
  – Data & event flow, synchronous call/return, shared access
  – End-to-End flow specifications

Real-time Task Scheduling
  – Supports different scheduling protocols incl. GRMA, EDF
  – Defines scheduling properties and execution semantics

Modal, configurable systems
  – Modes to model transition between statically known states & configurations

Component evolution & large scale development support

AADL language extensibility
Thread Execution Semantics

- Dispatch protocols
- Nominal & recovery
- Fault handling
- Resource locking
- Mode switching
- Initialization & finalization
Execution Platform Bindings

Co-location constraints in support of redundant systems

Processor, memory, and connection bindings

High speed network

1553 bus

Mission Processor

Display Processor

Pilot Display

CoPilot Display

1553

Flight Mgr

Weapons Mgr

Warnings Annunciations

MFD Pilot

MFD Copilot

1553 bus
An Avionics System Case Study

- Migration from static timeline to preemptive scheduling
  - Identified issues with shared variable communication
  - Migration potential from polling tasks to event-driven tasks
- Flexibility, predictability & efficiency of port-based communication
  - Defined communication timing semantics
  - Support for deterministic transfer & optimized buffers
- Effectiveness of connection & flow semantics
  - Support end-to-end latency analysis
- Analyzable fault-tolerant redundancy patterns
  - Orthogonal architecture view without model clutter
A Naïve Thread-based Design

From other Partitions

Periodic I/O

Navigation

Sensor Processing

20Hz

Pr 1

Pr 2

Pr 3

Integrated Navigation

Shared data area

Guidance Processing

20Hz

Pr 4

Flight Plan Processing

5Hz

Pr 6

Aircraft Performance Calculation

2Hz

Pr 9

To other Partitions

Potential non-deterministic communication due to preemption

Potential priority inversion due to priority assignment

Tasks must complete within frame => cyclic executive behavior
Flight Manager in AADL

From Partitions

Navigation Sensor Processing

20Hz

Integrated Navigation

10Hz

Guidance Processing

20Hz

Flight Plan Processing

5Hz

Fuel Flow

Nav data

Nav signal data

Nav sensor data

FP data

Aircraft Performance Calculation

2Hz

Performance data

www.aadl.info
Data Stream Latency Analysis

• Flow specifications in AADL
  – Properties on flows: expected & actual end-to-end latency
  – Properties on ports: expected incoming & estimated output latency

• End-to-end latency contributors
  – Delayed connections result in sampling latency
  – Immediate periodic & aperiodic sequences result in cumulative execution time latency

• Phase delay shift & oscillation
  – Noticeable at flow merge points
  – Variation interpreted as noisy signal to controller

Potential hazard

Latency calculation & jitter accumulation
Other Flow Characteristics

• Miss rate of data stream
  – Accommodates incomplete sensor readings
  – Allows for controlled deadline misses
• State vs. state delta communication
  – Data reduction technique
  – Implies requirement for guaranteed delivery
• Data accuracy
  – Reading accuracy
  – Computational error accumulation
• Message acknowledgment semantics
  – In terms of flow steps
Redundancy Specification

- Redundancy abstraction
- Co-location constraints on execution platform binding

Redundancy characteristics as properties
Primary/Backup Patterns

Passive Backup
- CSS1 Primary
  - SS1.1
  - SS1.2
- CSS1 Backup
  - SS1.1
  - SS1.2

Hot Standby
- CSS1 Primary
  - SS1.1
  - SS1.2
- CSS1 Backup
  - SS1.1
  - SS1.2

Continuous State Exchange
- CSS1
  - SS1.1
  - State
  - SS1.2

Voted Output
- CSS1
  - SS1.1
  - SS1.2
  - SS1.3
Primary Backup Synchronization

• External and internal mode control
• Errors reported as events
• Supports reasoning about Primary/Backup logic
AADL Language Extensions

- New properties through property sets
- Sublanguage extension
  - Annex subclauses expressed in an annex-specific sublanguage
- Project-specific language extensions
- Language extensions as approved SAE AADL standard annexes
- Examples
  - Reliability modeling
  - ARINC 653
  - Behavior
  - Constraint sublanguage
Example Annex Extension

THREAD t
FEATURES
   sem1 : DATA ACCESS semaphore;
   sem2 : DATA ACCESS semaphore;
END t;

THREAD IMPLEMENTATION t.t1
PROPERTIES
   Period => 13.96ms;
   cotre::Priority => 1;
   cotre::Phase => 0.0ms;
   Dispatch_Protocol => Periodic;

COTRE thread properties

ANNEX cotre.behavior {**

STATES
   s0, s1, s2, s3, s4, s5, s6, s7, s8 : STATE;
   s0 : INITIAL STATE;

TRANSITIONS
   s0 -{}-> s1 { PERIODIC_WAIT };
   s1 -{}-> s2 { COMPUTATION(1.9ms, 1.9ms) };
   s2 -[ sem1.wait ! (-1.0ms) ]-> s3;
   s3 -{}-> s4 { COMPUTATION(0.1ms, 0.1ms) };
   s4 -[ sem2.wait ! (-1.0ms) ]-> s5;
   s5 -{}-> s6 { COMPUTATION(2.5ms, 2.5ms) };
   s6 -[ sem2.release ! ]-> s7;
   s7 -{}-> s8 { COMPUTATION(1.5ms, 1.5ms) };
   s8 -[ sem1.release ! ]-> s0;
**);
END t.t1;

COTRE behavioral annex

Courtesy of cotre
Reliability Modeling Approach

Error state & occurrence model as AADL extension
- Error states and transitions
- Fault events & occurrence rates
- Error propagation rates
- Masking of subcomponent and propagation errors

Architecture model provides
- Dependency information
- Isolation analysis
- Basis for stochastic process model generation

Reflects hazard analysis, component failure modes & effects analysis
An XML-Based AADL Tool Strategy

- Textual AADL
- Graphical AADL Editor
  - Declarative AADL XML
  - Architecture Import/Export
  - AADL Instance XML
  - AADL Runtime Generator
- Semantic Checking
- Execution Platform Binding
- Scheduling Analysis
- Commercial Tool like TimeWiz
- Reliability Analysis
- Safety Analysis
- Project-Specific In-House
- Filter to Markov Analysis

© 2004 by Carnegie Mellon University
www.aadl.info
Open Source AADL Tool Environment

• OSATE is
  – Developed by the Software Engineering Institute
  – Available at under a no cost Common Public License (CPL)
  – Implemented on top of Eclipse Release 3 (www.eclipse.org)
  – Generated from an AADL meta model using the Eclipse Modeling Framework (EMF)
  – A textual & graphical AADL front-end with semantic & XML/XMI support
  – Extensible through architecture analysis & generation plug-ins

• OSATE offers
  – Low cost entrypoint to the use of SAE AADL
SAE AADL and OSATE: Enablers of Embedded Systems Research

• Industry standard architecture modeling notation & model interchange format facilitates
  – Interchange of architecture models between contractors & subcontractors
  – Common architecture model for non-functional system property analysis from different perspectives
  – In-house prototyping of project specific architecture analysis & generation
  – Architecture research with access to industrial models & industry exposure to research results