The SAE Architecture Analysis and Description Language (AADL) Standard: A Basis for Architecture-Driven Embedded Systems Engineering

DSN 2006 Workshop on Architecting Dependable Systems (WADS)
27 June 2006
Philadelphia, PA

Joyce L Tokar, PhD
Pyrrhus Software
tokar@pyrrhussoft.com
Objectives

• *Introduce* architecture-based development concepts and benefits.

• *Provide* a summary of the SAE AADL Standard

• *Provide* a summary of AADL’s capabilities.

• *Demonstrate* the benefits of AADL in real-time systems design.

• *Provide* an overview of the AADL development environment.
The SAE AADL Standard

- Sponsored by the Society of Automotive Engineers (SAE)
  - Avionics Systems Division (ASD)
    - Embedded Systems (AS2)
      - Avionics Architecture Description Language Subcommittee (AS2C)

- Status
    - Graphical Annex
    - XML Annex
    - Programming Language Annex
    - Error Annex
  - UML Annex to be balloted in 2006.

- Coordination with
  - NATO Aviation, NATO Plug and Play, French Government COTRE, ASSERT, SAE AS-1 Weapons Plug and Play, OMG UML

http://www.aadl.info

email: info@aadl.info
SAE AS-2C AADL Subcommittee

- **Key Players:**
  - Bruce Lewis (AMCOM): Chair, technology user
  - Steve Vestal (Honeywell): MetaH originator, co-author
  - Peter Feiler (SEI): Technical lead, author, co-editor, technology user
  - Ed Colbert (USC): AADL & UML Mapping
  - Joyce Tokar (Pyrrhus Software): Programming Language Annex, co-editor

- **Members:**
  - Boeing, Rockwell, Honeywell, Lockheed Martin, Raytheon, Smith Industries, Airbus, Axlog, Dassault, EADS, High Integrity Solutions
  - NAVAir, Open Systems JTF, British MOD, US Army
  - European Space Agency

- **Coordination with:**
  - NATO Aviation, NATO Plug and Play, ESA, French Government CÔTRE, OMG-UML & SysML, SAE AS-1 Weapons Plug-n-Play
Model-Based System Engineering

Predictive Analysis Early In & Throughout Life Cycle

Architecture Modeling & Analysis

Requirements Analysis

Architecture-Driven Development

System Integration

Rapid Integration
Predictable Operation
Upgradeability
Reduced Cost
What is Architecture?

- Architecture is the fundamental organization of a system as embodied in
  - its components,
  - their relationships to each other and the environment,
  - the principles governing its design and evolution.

- The architecture of a program or computing system is
  - the structure or structural arrangements of its composite elements, both hardware and software,
  - the externally visible properties of those elements,
  - the relationships among them.

Architecture is the foundation of good software & systems engineering
What is an Architecture Description Language (ADL)?

- The *architecture* of a system defines its high-level structure and exposes its gross organization as a collection of interacting components.
- An *Architecture Description Language (ADL)* focuses on the high-level structure of the overall application rather than on the implementation details of any specific component.
- ADLs and their accompanying toolsets support architecture-based development, formal modeling, and analysis of architectural specifications.
- The *AADL* is an architecture description language that includes support for the inclusion of both the software components and the execution platform components in the system architectural specification.
Architecture Description Languages

Research ADLs

- MetaH
  - Real-time, modal, system family
  - Analysis & generation
  - RMA based scheduling
- Rapide, Wright, ..
  - Behavioral validation
- ADL Interchange
  - Acme, xADL
  - ADML (MCC/Open Group, TOGAF)

Industrial Strength

- HOOD/Stood
- SDL
- UML 2.0, UML-RT
The SAE Architecture Analysis and Design Language (AADL)

- A language for

  - abstract and precise description of real time, performance critical architectures including both hardware and software components.

  - incrementally integrating multiple dimensions of analysis (time, safety, dependability, schedulability, utilization, fault tolerance etc) through component properties for system engineering analysis.

  - taking a specification of the architecture and using it to auto-integrate a compliant system from compliant components.
SAE Architecture & Analysis Description Language (AADL)

• Specification of
  - Real-time
  - Embedded
  - Fault-tolerant
  - Securely partitioned
  - Modal & dynamically configurable

• Software task and communication architectures

• Bound to
  - Distributed multiple processor hardware architectures

• Fields of application
  - Avionics, Aerospace, Automotive, Autonomous systems, …
AADL-Based System 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

Predictive Embedded System Engineering
Reduced Development & Operational Cost

SAE AADL

AADL-Based System Engineering

SSTC 2006
AADL: The Language

• The AADL defines standard *categories* of *components*:
  - Software: data, subprogram, thread, thread group, process
  - Execution platform: device, memory, bus, processor
  - Composite: System

• A *connection* between component *ports* declares a *flow* of control and/or data between components.
  - Ports: data, event, event data
  - Connections: port-to-port, subprogram calls.

• The relationship between software and execution platform components is represented through the use of *bindings*. 
AADL: The Language

- **Property associations** are used to constrain the model, for example, the legal and required bindings, but bindings need not be completely and explicitly declared by the developer.

- A component may have an **implementation**, an internal sub-architecture declared as a set of connected sub-components.

- A **package** provides a way to organize components and port group types into a related sets of declarations.

- **Modes** may be used to model transition between statically known states & configurations.
AADL: The Language

• **Component Type** -- specifies the interface to the component.

• **Component Implementation** -- zero or more specifications of the component’s internal representation.
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 data access
- End-to-end flow specification

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

Modal, reconfigurable systems
- Mode to mode transition between statically known states & configurations

Component evolution & large scale development support
- Inheritance for types and implementations
- Component packages provide subcontractor support

Language extensibility
- Standard typing sublanguage for user defined types
- User/vendor/industry/standard Annex sublanguages
Multiple Viewpoints of SAE AADL

- **Component View**
  - Model of system composition & hierarchy.
  - Well-defined component interfaces.

- **Concurrency & Interaction View**
  - Time ordering of data, messages, and events.
  - Dynamic operational behavior.
  - Explicit interaction paths & protocols.

- **Execution View**
  - Execution platform as resources.
  - Specification & analysis of runtime properties
    - timeliness, throughput, reliability, graceful degradation, ...
  - Binding of application software.

- **User-defined View**
  - Analysis-oriented.

- **Logical View**
  - Specification of relationships between software and execution platform components.

Primary target was the concepts and viewpoints associated with an operational system.
The AADL in a Nutshell

**Application**
- Thread, Process, System Execution Platform
  - Execution engine
  - Memory, Bus, Device

**Components**
- Specifications
- Variant implementations
- Ports
- Connections
- Domain data objects
- Behaviors

**System Architecture**
- Performance-Critical
- Layering & Composition

**Implementation**
- Hardware
  - Modeling and bindings fully supported by AADL (auxiliary to the SW API)

**Key Features**
- **Reusable**
  - Very portable. Function/non-functional requirements. Ideal isolation from hardware.

- **Real-time**
  - User specifies timing requirements, analyzers available, concurrency handled automatically!

- **Reliability, Safety, Security, Support**
  - User specifies requirements; analyzers available

- **Hardware Independent**
  - No implementation specified in SW API.

- **Extensible/Scalable**
  - Multi-processor/multi-process, easily add/change and see effects. User defined domain specific functions.

- **Flexible**
  - System spec used to change implementation. Interface with any standard or application

- **Generic**
  - Modular, scalable, system “block diagram” with semantics

- **Object-Oriented**
  - Clearly defined object, messaging, properties, decomposition

- **Open**

- **Usable and Available**
  - Approach/formalism is SIMPLE/UNIFORM, PRACTICAL, and EASY TO USE, LEARN, AND INTERFACE WITH OTHER APPROACHES!

- **Hardware Modeling and Bindings**
  - Fully supported by AADL (auxiliary to the SW API)

- **Verifiable**
  - Strong support for predictable real-time architectures exhibiting high-reliability

- **Formal, Rich Semantics**
  - Models can span high-level system to detailed interfaces
The SAE AADL Standard

• Provides a standard & precise way to describe the architecture of embedded computer systems.
• Provides a standard way to describe components, assemblies of components, and interfaces to components.
• Describes how components are composed together to form complete system architectures.
• Describes the runtime semantics and thread scheduling protocols.
• Describes the mechanisms to exchange control and data between components.
• Describes dynamic run-time configurations.
AADL: The Language

System Scheduling

- Supports different scheduling protocols including Rate Monotonic Analysis (RMA), Earliest Deadline First (EDF), user-defined
- Defines scheduling properties and execution semantics
- Hardware and Software binding constraints support system optimization, product-lines, safety

Scaleable

- From software subprogram
- To hardware and software System of Systems

Component evolution & large scale development support

- Inheritance for types and implementations
- Component packages provide subcontractor support

AADL language extensibility

- Standard typing sublanguage for user defined types
- User/vendor/industry/standard Annex sublanguages
Application Components

- System: hierarchical organization of components
  - System

- Process: protected virtual address space
  - process

- Thread group: organization of threads in processes
  - Thread group

- Thread: a schedulable unit of concurrent execution
  - Thread

- Data: potentially sharable data
  - data

- Subprogram: Callable unit of sequential code
  - Subprogram
Execution Platform Components

- Processor – Provides thread scheduling and execution services
  ![Processor](image)
- Memory – provides storage for data and source code
  ![Memory](image)
- Bus – provides physical connectivity between execution platform components
  ![Bus](image)
- Device – interface to external environment
  ![Device](image)
Graphical & Textual Notation

system Data_Acquisition
features
  speed_data: in data port metric_speed;
  GPS_data: in data port position_carthesian;
  user_input_data: in data port user_input;
  s_control_data: out data port state_control;
end Data_Acquisition;
Flow Specification in AADL

System S1

Flow Specification
- flow path F1: pt1 -> pt2
- flow path F2: pt1 -> pt3

Process P1

Flow Implementation
- flow path F1: pt1 -> C1 -> P2.F5 -> C3 -> P1.F7 -> C5 -> pt2

Connections are physical

Flows are logical
Faults and Modes

- AADL provides a fault handling framework with precisely defined actions.
- AADL supports runtime changes to task & communication configurations.
- AADL defines timing semantics for task coordination on mode switching.
- AADL supports specification of mode transition actions.
- System initialization & termination are explicitly modeled.
- Error Annex provides support for error models and analysis.
An Avionics System Case Study

- Migration from static timeline to preemptive scheduling
- Towards distributed partitioned system
- Software & hardware redundancy
- Access to detailed design & performance data

- Pattern-based analysis of architecture
  - Abstract, but precise architecture models
  - Identify potentially systemic issues

- High-fidelity analysis of network workload
  - Model generated from design data
  - Tool-based analysis of full-scale model
System Timing Concerns

• Critical flows: application perspective
  - Unqueued data streams, event streams, queued message streams
  - Sampling of stream, throttling of processing
  - End-to-end latency, throughput
  - Variability & upper bounds
  - Hybrid control systems & modal operation

• Critical flows: embedded software perspective
  - Periodic & aperiodic threads
  - Efficient communication mechanisms
  - Time & space partitioning
  - Schedulability of processor & buses/networks
  - Hybrid & modal task architectures
Flight Manager: Principal Functionality

- **Navigation Sensor Processing**
- **Integrated Navigation**
- **Guidance Processing**
- **Flight Plan Processing**
- **Aircraft Performance Calculation**

**From other Partitions**
- Periodic I/O
- From other Partitions

**To other Partitions**
- Flight Manager State

- **Nav Radio**
- **Auxiliary service**

- **Processing functions**
- **20Hz**
- **10Hz**
- **5Hz**
- **2Hz**

**SSTC 2006**
A Cyclic Executive Implementation

Switch clock mod
Hyperperiod
Case 20Hz:
call PIO
call NSP
call GP
Case 2*20Hz: -- 10Hz
call PIO
call NSP
call IN
call GP
Case 3*20Hz:
. . .
Case 4*20Hz: -- 5Hz

Tasks must complete within frame => cyclic executive behavior

From other Partitions

Periodic I/O

20Hz

Navigation Sensor Processing

20Hz

Integrated Navigation

10Hz

Shared data area

Potential non-deterministic communication due to preemption

To other Partitions

Potential priority inversion due to priority assignment

Guidance Processing

Flight Plan Processing

Aircraft Performance Calculation

20Hz

2Hz

5Hz

1

2

3

4

5

6

From other Partitions

Periodic I/O

Navigation Sensor Processing

Integrated Navigation

Shared data area

Potential non-deterministic communication due to preemption

To other Partitions

Potential priority inversion due to priority assignment

Guidance Processing

Flight Plan Processing

Aircraft Performance Calculation

20Hz

2Hz

5Hz
Priority Inversion Checker

• Analysis of AADL models
  ß User assigned priorities
    • Modeled as new property
  ß Potential red flag
    • Recording & reporting of analysis results

• Tool support
  ß Checker operates on system instance bound to execution platform
  ß External tool processes XML document
  ß Plug-in to Open Source AADL Tool Environment

Potential priority inversion identifiable by analysis tool
Non-deterministic Phase Delay

• Variable phase delay of data elements
  - Variable timing of user-level send/receive calls
  - Variable send/receive ordering due to preemption
  - Results in variable frame delay of data element

• Does it matter?
  - Data stream as controller input
    - Latency jitter viewed as noise in data stream
    - Software induced jitter engineered away
  - Data stream as display output
    - Phase delay oscillation results in blurred display
  - Time stamping of data elements
    - Time synchronization of data streams
Intended Data Flow

From other Partitions

Periodic I/O

20Hz

To other Partitions

Implemented via shared data
Achieved via precedence ordering

External I/O

20Hz

Shared data area

Decreasing Priority

Integrated Navigation

10Hz

Guidance Processing

20Hz

Flight Plan Processing

5Hz

Aircraft Performance Calculation

2Hz
Flight Manager in AADL

AADL connections have precise timing semantics

Immediate & delayed data port connections preserve determinism

Flight Plan Processing

Navigation Sensor Processing

Integrated Navigation

Guidance Processing

Aircraft Performance Calculation

Fuel Flow

From Partitions

To Partitions

Nav signal data

Nav sensor data

Nav sensor data

Nav data

Nav data

FP data

FP data

20Hz

10Hz

20Hz

5Hz

2Hz

Sale Software
Enduring Solutions

SSTC 2006
Analyzable and Reconfigurable AADL Specifications for IMA System Integration

- Not modeled for this AADL example
Graphical Software (Logical) View

```
system
  CDU_Processor_Software_Impl
    p_CDU_Display_Manager: process
      CDU_Display_Manager_Impl
        <ndo>_to_<destcpm>_<sw>_Out_Socket
        <ndo>_to_<destcpm>_<sw>_Out
        <rate>
        thread
        <vm>_<rate>
        <rate>
        thread
        <vm>_<rate>
        <rate>
        thread
        <vm>_<rate>
    p_CDU_IO_Manager: process
      CDU_IO_Manager_Impl
        <ndo>_to_<destcpm>_<sw>_Out
        <ndo>_from_<srccpm>_<sw>_In
    p_Communications_Manager: process
      Communications_Manager_Impl
        <ndo>_to_<destcpm>_<sw>_Out_Socket
        <ndo>_to_<destcpm>_<sw>_Out
        <ndo>_to_<destcpm>_<sw>_Out_Socket
        <ndo>_from_<srccpm>_<sw>_In
        <ndo>_from_<srccpm>_<sw>_In
        <ndo>_from_<srccpm>_<sw>_In
        <ndo>_from_<srccpm>_<sw>_In
        <ndo>_from_<srccpm>_<sw>_In_Group
        <ndo>_from_<srccpm>_<sw>_In_Group
        <ndo>_from_<srccpm>_<sw>_In_Group
        <ndo>_from_<srccpm>_<sw>_In_Group
    p_Flight_Manager: process
      Flight_Manager_Impl
        <ndo>_to_<destcpm>_<sw>_Out
        <ndo>_to_<destcpm>_<sw>_Out
        <ndo>_to_<destcpm>_<sw>_Out

```
Overall System Integration

Notes:

Identifiers with angle-bracketed terms are replicated for each unique set of terms, where terms are defined as:

- `<cpm>`: Common processing machine name
- `<cpmid>`: Longer name of cpm
- `<ndo>`: Network data object name
- `<sw>`: ASL switch side identifier
- `<vmr>`: Virtual machine and rate, indicating thread name

Prototype system of a Generic Display System Using a switched ethernet LAN.

5 Common Processing Modules
13 Virtual Machines
90 Threads
165 End-to-end Data Flows
Analysis and Reconfiguration Tool

• System generation from Translated XML AADL
  - Automatic schedule generation
  - Allocation of VMs to hosts
• System analysis
  - Schedulability, rate-monotonic analysis
  - Network analysis
• Editing and visualization
  - Direct manipulation, tree view
  - Graphs, tables, trade studies
Some Other AADL Users

Model Based Systems Engineering at DARP

Modelling of PnP Weapon Systems with AADL – Protocol Behaviour

The University of York

Service-oriented architectures and AADL modeling

Oleg Sokolsky
Real-Time System Group
University of Pennsylvania

SAE AADL Working Group Meeting
January 24-27, 2006

Automating Timing and Safety from Architecture Specific

Steve.Vincent@Honeywell.com
13 April 2005

SAE Aircraft/Store System Roadmap

Towards solving: Binding for AADL Using Constraint Programming

Christophe Guettler, Dr
System Architect

AADL Standardisation Committee,
Arcueil 19 October 2005

Honeywell Aerospace

SSTC 2006
An Extensible Engineering Environment

AADL Extensions
- Error model
- Concurrency behavior

Architecture
- Consistency
- Plug-ins

OMNET++
- Network simulation

Object Model Interface
- Network model

Model Export Filters
- Timing model

TimeWeaver (CMU)
- Distributed resource allocation
- Multi-platform runtime system generation

Architecture Import & Extraction
- Architecture Export MetaH, TTA

MetaH Toolset
(Honeywell)
- Scheduling analysis
- Reliability analysis
- Isolation analysis
- Runtime system generation

TimeWiz Commercial Tool
- Scheduling analysis
- Execution trace analysis

SSTC 2006
Benefits of Model-Driven Development

- Model-based system engineering benefits
  - Analyzable architecture models drive development
  - Predictable runtime characteristics at different modeling fidelity
  - Model evolution & tool-based processing
  - Prediction early and throughout lifecycle
  - Reduced integration & maintenance effort

- Benefits of AADL as SAE standard
  - Common component definitions across teams, documents
  - Single architecture model augmented with analysis properties
  - Interchange & integration of architecture models
  - Tool interoperability & extensible engineering environments
  - Aligned with UML engineering, potential adoption by UML

The SAE AADL
– as an industry standard –
provides a stable common framework for contractors
to cooperatively evolve large-scale systems and
for tool vendors to provide tools for a
common architecture representation.

Reduction in errors in the final system through early analysis and automatic system generation.
## Acronyms

- **AADL** – Architecture Analysis and Description Language
- **ADL** – Architecture Description Language
- **ADML** – Architecture Description Markup Language
- **ASD** – Avionics Systems Division
- **AS2** – ASD Embedded Systems Subcommittee
- **AS2C** – ASD AS2 Avionics Architecture Description Language Subcommittee
- **CMU** – Carnegie Mellon University
- **EDF** – Earliest Deadline First
- **HOOD** – Hierarchical Object Oriented Design
- **IMA** – Integrated Modular Architecture
- **LAN** – Local Area Network
- **MCC/Open Group** – The Open Group Micro-electronics and Computer technology Consortium
- **OSATE** – Open Source AADL Tool Environment
- **RMA** – Rate Monotonic Analysis
- **SAE** – Society of Automotive Engineers
- **SDL** – Specification and Description Language
- **SEI** – Software Engineering Institute
- **STOOD** – S Object Oriented Design
- **TOGAF** – The Open Group Architecture Framework
- **TTA** – Time Triggered Architecture
- **UML** – Unified Modeling Language
- **xADL** – Highly Extensible Architecture Description Language
- **XML** – Extensible Markup Language

Thank You!

Questions?