Industrial Applications of AADL

“AADL Avionics Case Studies and Concepts for Integrating AADL into System Development”

January 29, 2007

Prepared by John Mettenburg, Advanced Technology Center
jjmetten@rockwellcollins.com
Agenda

Introduction
Case Studies
Concepts for Integration
Summary & Lessons Learned

Integrate, then build.
Business Profile

Communications and Aviation Electronics Systems for Commercial and Military Applications Worldwide

2006 Sales $4.2 Billion

Government Systems
- Integrated Systems
- Sensor Systems
- Displays
- Communication Systems

Commercial Systems
- Air Transport
- Business & Regional
- Cabin Systems
- eFlight

Engineering & Technology

Advanced Technology Center
The Advanced Technology Center (ATC) identifies, acquires, develops and transitions value-driven technologies to support the continued growth of Rockwell Collins.
Embedded Information Systems

Automated System Analysis / Verification

- Predictive Tools for Complex Architectures
- Information Flow Modeling & Analysis
- Model-Based Verification

Secure Computing Systems

- Gigabit Encryption
- Assured Information Sharing
- System Level Anti Tamper

Persistent ISR

- System Framework
- Knowledge Management
- Information Analysis & Fusion
Predictive Tools for Complex Architectures
Project Goals and Key Issues

Allow System Developers to Design the System Architecture & Predict Its Behavior from Bid and Proposal through Maintenance

• What Needs to be Predicted?
  – Performance, capacity, latency, safety, security, …
• What Needs to be Modeled?
  – Logical and physical architecture,
  – CPU cycles, bus bandwidth, …
• How is the Analysis Done?
  – Custom Tools, commercial tools, open source tools, …
• How Can We Add Value?
  – Focus on back end analysis, build competitive advantages,…
• What is the Transition Path to Product Areas?
  – Use existing tools, open standards, use existing editors on front end, …

Integrate, then Build.
Systems Development Process Context

Focusing on Design Phase with the intent to Support “Early Integration” through Predictive Analysis
Agenda

Introduction

Case Studies

- SE Tool Study (2005)
- AADL Integration Pilot (2006)

Concepts for Integration

Summary & Lessons Learned

Integrate, then build.
**AADL IMA System Analysis for Trade Studies**

**Display System Graphical AADL**

**Notes:**
Identifiers with angle-bracketed terms are replicated for each unique set of terms, where terms are defined as:
- `<cpmid>`: Longer name of cpm
- `<srccpm>`: Source common processing machine
- `<destcpm>`: Destination common processing machine
- `<vmr>`: Virtual machine and rate, indicating thread name
- `<ndo>`: Network data object name
- `<sw>`: ASL switch side identifier

**Automated Analysis**
- 5 CPMs
- 13 Virtual Machines
- 90 Threads
- 165 End-End Flows
SE Tools Study

- **UML Based**
  - Artisan Realtime Studio from Artisan (UML 2.0/OO)
  - Enterprise Architect (UML 2.0/OO)
  - Rhapsody from Ilogix (UML 2.0/OO)
  - Tau from Telelogic (UML 2.0/OO)
  - System Architect/SW Modeler from Rational/IBM (UML 2.0/OO Eclipse)
- **Other**
  - ARIS from Ids Scheer
  - CORE from Vitech (DoDAF)
  - Metis from Computas (Zachman/DoDAF/UML 2.0)
  - MIAA (Military Information Architecture Acceleration) from Ptechinc (DoDAF/CADM)
  - NetViz from NetViz (DoDAF)
  - Proforma from Proforma (Business model focus /UML/DoDAF/OO/Structured Decomposition)
  - TeamCenter Architect from UGS (Partial UML, DoDAF, PIM/PSM)
  - VisualSim from Mirabilis Design
  - Cradle from 3SL (UML, DoDAF, PIM/PSM)
AADL Visio Stencil Integration
Note the Introduction of AADL Stereotypes

AADL Provides Backbone to UML For Real-Time Embedded Systems…
AADL Integration Pilot Evaluation of End-End Latency

Architecture Visualization

Teamcenter

XML

Physical Architecture Capture

Database

XML

Architecture Analysis

OSATE

```
Display_System . ( , . . . , )
  Function\Wrappers
    CDU - Left - ( , CDU-4500 )
    CDU - Right - ( , CDU-4500 )
```

```
properties
  system::Name => "Display_System";

subcomponents
  pCDU_d_Left : process TP_CD Ud4500.I_CD Ud4500;
  pCDU_d_Right : process TP_CD Ud4500.I_CD Ud4500;
```
AADL Integration Pilot
Latency Allocation & Validation Using AADL

Avionics System

Appl 1
Process
CPU
Display

Appl 2
Process
CPU
Keypad

Latency Budget
Comm 2 + SW Time 2 + Switch Delay + SW Time 1 + Comm 1
Analyze Latency…

End-to-end flow latency 425 ms exceeds specified latency 350 ms
AADL Integration Pilot
UGS Teamcenter (Alternative Flow Analysis/Visualization)
AADL Integration Pilot - Conclusion

• Realized the value of a intermediate architecture form
  – Instead of publishing the internal schema of the tool
    • This is the AP/STEP approach
  – To simply document the semantics (as part of the translator)
    • Promotes a common understanding and best practices (common AND unique)
  – Found AADL to be well-matched for the task
    • Correctness, Visualization, Modularization & Analysis Infrastructure
  – QVT showing some promise
    • Communication & Automation

• Realized the power of Teamcenter
  – Office integration proved workable and valuable
    • e.g Reuse of existing Spreadsheets, replication of existing diagrams & reports
  – Usable version control
  – Much more capability than needed for visualization
  – Lack of syntax and semantics at this level could be a benefit (“dumbed down diagrams”)

• Demonstrated the value of Eclipse
  – Reused the latency analysis plug-in (OSATE)
  – Analysis plug-ins were easy to develop
  – But still need a scripting language…

• Need to watch Scalability & Scope
  – For example, XML is verbose (DOM XML parser approach is inadequate)
  – One size does NOT fit all (focus on IMA helped)
  – Need to incorporate modularity in Exchange mechanism to support the “24 hr turn”
AADL “Forms IO” Evaluation

SEI Forms Based AADL Import/Export Concept

Input Data ➔ Excel Templates ➔ Access Tables ➔ XML ➔ OSATE

Reformat Existing Data into Excel Worksheets

Process using the Forms IO Tools

Analyze/Operate using The AADL IDE (OSATE)
Agenda

Introductions
Case Studies
Concepts for Integration
Summary & Lessons Learned
Concepts for Integration
Architecture Modeling & Analysis - Business Adoption

- **Top-Down**
  - Enterprise Initiatives
  - Product Line
  - Customer Requirements/Constraints

- **Bottom-Up**
  - Proof of Concept
  - Pursuit Strategy (cost & schedule)
  - Innovative People
  - Communities of Practice

- **Middle-Out**
  - Subsystems are the common denominator
    - Good place to standardize the notation

- **Underneath**
  - AADL provides common solution space for Architecture models
    - Build DSL on top
Concepts for Integration - AADL at Rockwell Collins

- Actively participates in the AADL standards development
- Rockwell Collins published AADL papers and a modest AADL example model ([www.aadl.info](http://www.aadl.info))
- AADL XML is used for architecture exchange behind on-going programs
- AADL Based Analysis/Deployment Tools

- AADL Integration Into the Development Process…
  - Standards & Open Source
  - Common Architecture Model
  - Interface Data Specification
- AADL Tool Integration Efforts
  - UML
  - Simulink™
  - Custom Tools
Concepts for Integration
Accelerating Design using Standards & Open Source

• Benefits of Standards
  – Enable communication between tools in a framework
  – Provide stability to an architecture framework. I.e. they shield against tool obsolescence and its impact.
  – Protect intellectual property
  – Promote Innovation
  – Supports use of “best fit” tool across disciplines, design cycle & life cycle

• Open Source
  – Accelerating Standards Adoption
  – Reducing Cost of Technology
  – Reinforces need for standards
  – Eclipse Platform
  – Domain Specific Languages
  – Internal Open Source
**Problem**
Ad-hoc approach to design and implementation of translators discourages reuse, couples tool versions and prevents change

**Solution**
Standardized Architecture Exchange approach provides standardized template, minimizes number of translators and promotes agility

**Emphasis on Standard Languages and Translators – Not tools**
Concepts for Integration - Common Architecture Model
Approach

- Standards based approach
  - AADL as an intermediate form (reference meta-model)
  - UML (Unified Modeling Language) Meta-Models
  - XML/XMI (metamodel representation)
  - QVT (model transformation language)
  - MARTE approach for a Property Model

- Declarative mappings
  - Human understandable/checkable
  - Defined as high level QVT transformations

- Automated Report Generation (Translator Spec)

- Path to Model based Transformations
  - ATL, Express-X …
• Challenges
  – Number of mappings to be defined
  – Lack of documentation and a well defined semantics for many of the models to be mapped
  – Potentially significant mismatches between meta-models, at a conceptual level
  – Development of a process for maintenance/evolution of metamodels & mappings over time
  – Scaleability for large models/mappings → incremental updates
  – AP233/SysML Integration

It’s a lot of work...
Concepts for Integration - Common Architecture Model

• **Benefits**
  – Interoperability between processes and tools
    • Internal processes and tools
    • Industry standard, external processes and tools
  – Common representation for analysis
  – A Reference for common, well-defined semantics
  – Ability to more easily evolve models and mappings
  – Ability to adapt models and model generated code to specific domains, platforms, applications, etc.

**Lower the Cost and Improve Quality**
Concepts for Integration - Interface Data Specification

• **Interface design data can be captured and exchanged with AADL**
  - XML is popular as exchange format
  - Property support is well suited for the problem
  - Text & Graphicals forms are readable
    - Documentation, Review
    - Debug and Integration
  - Ease of Integration with System and Software Models

• **Standardized Views are possible**
  - Extensible Data Editor
  - Report Generator

• **Issues**
  - Not Invented Here
  - Verbosity
  - XML with a robust schema is not light weight

• **AADL currently “under the covers” for an Enterprise, component modeling toolset**
The Data Dictionary is the information requirement.
data ReceiverMode
properties
  DD_Properties::Initial_Value => "State1";
  DD_Properties::Element_Type => MLEnumeration;
  DD_Properties::Element_Values => ("State1", "State2", … "State10");
end ReceiverMode;

data implementation ReceiverMode.serial
subcomponents
  rcvrmode: data INT16 {
    Element_Properties::Initial_Value => "0";
    Element_Properties::Element_Enum_Values => (1, 2, 3, 4, 5, 10);
  };
properties
  Element_Properties::Version => "12.2.a";
end ReceiverMode.serial;
data Latitude
properties
  DD_Properties::Numeric_Type => Float;
end Latitude;

data implementation Latitude.format2

subcomponents

  field1: data {
    Element_Properties::Length => 16 Bits;
    Element_Properties::Word_Offset => 0 W;
    Element_Properties::LSB_Position => 0 W;
    Element_Properties::LSB_Weight => value(\text{Constants::M\_PI\_Div\_2\_15});
    Element_Properties::Element_Units => RADIANS;
  };

  field2: data {
    Element_Properties::Length => 16 Bits;
    Element_Properties::Word_Offset => 1 W;
    Element_Properties::LSB_Position => 0 W;
    Element_Properties::LSB_Weight => value(\text{Constants::M\_PI\_Div\_2\_31});
    Element_Properties::Element_Units => RADIANS;
  };

  properties
  Element_Properties::Is_Packed => true;
end Latitude.format2;

end Longitude.format2;

Representation of Longitude is the same...
Data Dictionary

Shared Data Definitions, Or Implementations

data Latitude
  properties
    Data_Dictionary::data_type => FLOAT;
    Latitude;

system implementation icd.rs422
  subcomponents
    hdr : data pl_common::Copyright_String;
    msg1 : data pl_sp_messages::message1.format1;
    msg2 : data pl_sp_messages::message2.format1;
  properties
    ICD::version => "1.0";
end icd.rs422;

data implementation latitude.format1
  properties
    Element::Data_Type => FIXEDPT;
  end latitude.format1;

data latitude extends pl_dd_components::latitude
  end latitude;

data implementation message1.format1
  subcomponents
    header : data header;
    field1 : data pl_data_defs::latitude.format1;
    field2 : data pl_data_defs::latitude.format2;
  end message1.format1;

"pl" == Product Line

Independent Files for CM
Concepts for Integration – Custom Tools
Multi-Objective System Trade-off and Analysis (MOSTA)
Concepts for Integration
AADL Tool Integration Efforts – UML & Simulink

• UML
  – AADL profile validation/integration (Rhapsody)
  – AADL <-> UML w/AADL Profile Translator (XMI2.1 w/vendor adaptors)
  – SysML & Domain Specific Profiles
  – Component Integration

• Simulink™
  – Custom Integration
AADL Applications Summary - Lessons Learned

• **“Standard” sounds good to a funding source**
  – But maybe not so good for the funding target, Definitely not so good for some vendors
  – Be prepared to pay for it…

• **Do not under-estimate change to a Business area**
  – Look for allies (not just once! periodically)
  – Incremental, while less efficient, is more practical
  – Find the brief periods of intense pain – and exploit them

• **Know the Users**
  – Sometimes they are the last ones we ask

• **Evaluation and Pilot**
  – Let the customer decide

• **Common Architecture Model (CAM)**
  – Common Property Model is the real work…
  – Source & Destination Schema Shared Ownership

• **Translators**
  – Transfer what you need, Develop Role based views
  – Break up the transform if necessary
  – Make performance a requirement

• **Future Work**
  – Extend the CAM “down” for Component Modeling Integration
  – Extend the CAM “up” for Functional Architecture using AP233
    • Requirements, Functional Architecture, Configurations
  – Formal Methods Integration