AADL and MDA

Early Experience Applied to Aircraft-Weapon Integration

Yves LaCerte
Yves.LaCerte@gd-ais.com

3 February 2005
Agenda

- Introduction
- Weapon Management System - WMS
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Description Language - AADL
- Model Driven Architecture - MDA
Introduction

- The aircraft-weapon integration challenge is part of a larger integration problem, i.e.
  - Independent system-specific models often create unsolvable interoperability problems

- Tower of Babel
  - Build a tower to the sky
  - Workers speak different languages
  - Cooperation collapsed
Introduction

• Semiotics, a field of linguistics
  ➔ A science which studies the role of signs as part of social life *
    ■ Syntax: structure of signs
    ■ Semantics: meaning of signs as captured by syntax
    ■ Pragmatics: adaptable interpretation of signs based on context

Agenda

- Introduction
- Weapon Management System - WMS
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Description
- Language - AADL
- Model Driven Architecture - MDA

Primary electrical, functional and logical interface between the Mission Management Computer(s), weapons, launchers and other equipment used to release and deliver stores
Motivation

- Today, $100M (typically)*

- 40% to 60% is software updates (typically)*

* Source: AFRL/MN, 1999
Test and Integration Challenges

- **Platform Perspective**
  - Provide relevant functions and data
  - Observe resource, performance and timing constraints
  - Do not change platform software

- **Weapon Perspective** *
  - Identify/define relevant functions and data
  - Identify/define resource, performance and timing constraints

* Weapon software does change over time, which may impact the platform
Cost / Schedule Perspective (Notional)

WMS Development Costs

- Traditional
- New

Number of Months to Integrate a New Weapon

- Traditional
- New

- Weapon 1
- Weapon 2
- Weapon 3
- New WMS

MDA + AADL

No need to take down the whole squadron

Do not change platform software
Agenda

- Introduction
- Weapon Management System - WMS
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Description Language - AADL
- Model Driven Architecture - MDA
State of the Art

- Increasing complexity / decreasing productivity
  → six (or fewer!) lines per day *

- The inefficiency of the embedded software development process will prevent novel technologies from entering the marketplace in time

* Typical of embedded software industry
Characteristics

- Resource constrained
  - CPU, memory, size, weight and power
- Hard Real Time
  - Functional and non-functional aspects
- Safety Critical
  - Dependable / Certifiable
- Long Lifetime
  - Several upgrades / increasing complexity
Embedded Systems: Solutions / Trends

• Components
  - Less dedication to specific functions
  - Design - Improved abstraction
  - Synthesis - Auto code generation
  - Models - Assess before final implementation
  - Specification languages
    ■ Unambiguous representation of behavior and constraints - Rigorous semantics
    ■ Widely accepted
Embedded Systems: Solutions / Trends

- Dynamic Reflective Systems
  - Change internal behavior depending upon attached devices
  - Capable of integrating devices which provide new functions
  - Capable of providing unforeseen functionality
  - Foundation of aspect-oriented programming
Semantic Interface Specification

Syntax can be performed by any type of Interface Definition Language (IDL, XML)
Assess semantic properties of an interface by an executable interface model
Assess interoperability by analyzing provided and required interfaces, and contracts
Adapt interface to improve interoperability
Embedded Systems: Solutions / Trends

- Applied to Aircraft-Weapon Integration
  - Integrate new capabilities within a given design space (domain)
  - Prevent waiting for an aircraft upgrade cycle to integrate new weapons
- But
  - Likely difficult to implement a dynamic system that meets performance constraints
  - Aircraft-Weapon interface standards is a recent development, change is slow
Agenda

- Introduction
- Weapon Management System
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Descriptive Language - AADL
- Model Driven Architecture - MDA
The Plug and Play Concept

- Demonstrate interoperability at design time
  - In terms of Functionality and Data
    - Open system approach via standards
  - In terms of Non-Functional Quality Attributes
    - Safety, real-time, reliability, fault tolerance, security….

- A system that can exchange information and services with multiple systems is more interoperable than one that can't
  - Assess the quality of interoperability
  - Formulate strategies to improve interoperability
The WMS accepts generic missile commands. These are subsequently passed on to the missile.
The WMS offers mission services that are non-missile specific. Configuration data is subsequently passed on to the missile.
Domain Model - Life-Cycle View
State Machines

- Preferred for the specification of controllers
  - Useful for
    - Verification against requirements
    - Test-case generation
    - Automatic code generation

API and events used to cause state changes
Weapon requires power.... Power provided by WMS
Agenda

- Introduction
- Weapon Management System - WMS
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Description Language - AADL
- Model Driven Architecture - MDA
Platform Challenge

- Component models and supporting frameworks often rely on the specifics of the underlying platform.

- There is a need for techniques to handle functional and non-functional properties of components and systems.

AADL addresses non-functional properties.
AADL Summary

- Purpose
  - Analyze embedded systems

- Focus
  - Software system architecture model of an execution platform
    - Software components bound to hardware components

<table>
<thead>
<tr>
<th>Standardization Efforts</th>
<th>SAE AS5506</th>
<th>UML profiles</th>
</tr>
</thead>
</table>
UML Profiles

- UML 2.0
  - Embedded Systems
  - AADL
  - SysML
    - Activity Diagrams
    - Assembly Diagrams
    - Parametric Diagrams
    - Requirement Diagrams
AADL Components Types

UML 1.x and UML 2.0 profile... work in progress...
AADL Components Types

- **device**: Physical devices that interface with an external environment
- **memory**: Randomly accessible physical storage such as RAM or ROM
- **bus**: Communication channels that can exchange control and data between processors, memories, and devices
- **processor**: Abstraction of hardware and software that is responsible for scheduling and executing threads
- **System**: Composition of software and execution platform components
- **Process**: Space partitions in terms of virtual address spaces
- **ThreadGroup**: Structural grouping of threads within a process
- **Thread**: Units of concurrent execution
- **Data**: Static data in source text
- **Subprogram**: Source text that is executed sequentially

*Icons from SAE AS5506 AADL Annex*
System Example
**System Construction Example**

```plaintext
system MILSystem
end MILSystem;

system implementation
MILSystem.MissilePlatform
subcomponents

-- buses
Power_A : bus MILPower;

-- components
WMS1 : system WMS.WMS;
FCS : system FireControlSystem.FCS;

connections

-- bus access
bus access Power_A -> Power1.IFPower_A;
bus access Power_A -> WMS1.IFPower_A;

-- ports
port group WMS1.Out1553_A -> Launcher1.In1553_A;
```

---

[Diagram of System Construction]
system FireControlSystem
features
  IFGeneric: requires bus access MILGeneric;
  InMission: port group Rx_Port;
  OutMission: port group Tx_Port;
end FireControlSystem;

system implementation FireControlSystem.FCS
end FireControlSystem.FCS;
System Construction Example

Black Box

system WMS
features
-- Buses
IF1553_A: requires bus access MIL1553;
IFGeneric: requires bus access MILGeneric;
IFPower_A: requires bus access MILPower;
IFDiscrete_A: requires bus access MILDiscrete;
-- Ports
In1553_A: port group Rx_Port;
Out1553_A: port group Tx_Port;
InDiscrete_A: port group Rx_Port;
OutDiscrete_A: port group Tx_Port;
OutPower_A: port group Tx_Port;
InMission: port group Rx_Port;
OutMission: port group Tx_Port;
end WMS ;

port group Rx_Port
features
Rx: in data port;
inverse of Tx_Port
end Rx_Port;

port group Tx_Port
features
Tx: out data port;
end Tx_Port;
WMS System Implementation Example

White Box

```
subcomponents
--- processors
--- processes
--- bindings

connections
--- port groups

modes
MainMode: initial mode;
BackupMode: mode;
```

system implementation WMS.WMS
subcomponents

connections

modes
end WMS.WMS ;
process PlugandPlayDispatcher
features
  InGeneric: port group Rx_Port;
  OutGeneric: port group Tx_Port;
  InLauncher: port group Rx_Port;
  OutLauncher: port group Tx_Port;
  InWeapon: port group Rx_Port;
  OutWeapon: port group Tx_Port;
end PlugandPlayDispatcher;
Process Implementation

process implementation PlugandPlayDispatcher.WMS

subcomponents

  Bus_Listener: thread Listener.Bus_Listen;
  PnP_Dispatcher: thread PnPDispatcher.WMS;

connections

  port group InGeneric -> Bus_Listener.Bus_Listen;
  port group PnP_Dispatcher.OutLauncher -> OutLauncher;
  port group InLauncher -> PnP_Dispatcher.InLauncher;
  port group PnP_Dispatcher.OutWeapon -> OutWeapon;
  port group InWeapon -> PnP_Dispatcher.InWeapon;

end PlugandPlayDispatcher.WMS;
Time-Space Partitioning

Processor

Partition 1
Partition 2
... Partition n

Processor PPC
End PPC;
Processor implementation PPC.two
Subcomponents
  partition1: processor RMA.impl;
  partition2: processor RMA.impl;
End PPC.two;
System implementation ESys.impl
Subcomponents
  platform: processor PPC.two;
  app: system userApp;
Properties
  Binding => platform applies to app;
End ESys.impl;
Thread

Producer – Consumer

features

Buffer_Listener: port group Rx_Port;
Bus_Dispatcher: port group Tx_Port;
Buffer_2: requires data access Buffer;

properties

Source_Text => "abc";
Source_Code_Size => 100 kb;
Source_Data_Size => 10 kb;
Source_Stack_Size => 10 kb;
Source_Heap_Size => 10 kb;
Dispatch_Protocol => periodic;
Period => 100 ms;
Deadline => 100 ms;
Compute_Execution_Time => 5 ms;
Trade-offs

Producer – Consumer
Event-Driven with
Publish-Subscribe
Tools - OSATE

- Open source AADL tool environment
  - Software Engineering Institute
  - www.aadl.info

- Set of plug-ins on top of Eclipse
  - www.eclipse.org
OSATE Plug-ins

- Analysis
  - Schedulability
  - System scalability
  - Safety
  - End-to-end flow analysis

- AADL support from UML vendors
Agenda

- Introduction
- Weapon Management System - WMS
- Embedded Systems
- Plug and Play Concept
- Architecture Analysis and Description - Language - AADL
- Model Driven Architecture - MDA
Approach to MDA

TE = Transformation Engine
CG = Code Generator
VE = Validation Engine
Approach to MDA / AADL

Derive system properties

Prove system properties

UML Model

Platform Requirements

Product Instance

CG

Executable System

Prove system properties

AADL Model

Platform Requirements

Product Instance

CG

Executable System

Prove system properties

Validated System

Validated Product

CG

Executable Product
MDA Work Products

Application

Context View
Requirements View
Analysis View
Design View

… Queries …

Platform Independent Model

Guidelines and Rules
Platform Model
Libraries
Patterns
Producer/Consumer

Product Instance

Platform Requirements
Platform Specific Model

AADL Model ready for analysis
UML Model ready for code generation

GENERAL DYNAMICS
Advanced Information Systems
MDA / AADL Expectation

Development Costs for New Weapons

Development Costs for New Weapons

Traditional | New

Weapon 1 | Weapon 2 | Weapon 3

New WMS | MDA

General Dynamics
Advanced Information Systems

47