AADL and the Plug and Play Weapon
Early Experience Using the Architecture Analysis & Design Language
TC04

Yves LaCerte
3 November 2004
A Weapon Management System

Mission System
Video
Radar

1553 Eth
Video
RS-232

1553

RINU

Aircraft Discretes

1553

Eth

Video
RS-232

Starboard Wing Stations

Racks 1760 ASI Legacy ASI

Port Wing Stations

Weapons Bay

Starboard

Port
Introducing a New Weapon
Test and Integration Challenges

- **Platform Perspective**
  - Do not change the aircraft or store application software
  - Provide relevant functions and data
  - Observe resource, performance and timing constraints

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

- Demonstrate interoperability at design time
  - In terms of Functionality and Data
    - Open system approach via standards
  - In terms of Quality of Service (QoS)
    - 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
  - Can we measure the quality of interoperability?
  - Can we formulate strategies to improve interoperability based on measured quality?
The Plug and Play Concept - Standards

**APIs / Protocols**

<table>
<thead>
<tr>
<th>Store Control API (SCA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mil-Std-1760</td>
</tr>
<tr>
<td>Universal Armament Interface (UAI)</td>
</tr>
<tr>
<td>Generic Aircraft-Store Interface Framework, SAE AIR5532</td>
</tr>
<tr>
<td>Mil-Std-1553</td>
</tr>
</tbody>
</table>

**Data**

```xml
<?xml version="1.0" ?>
<catalog>
  <carrier id="Carriage Store M299">
    <Emergency_jettison_permitted>Yes</Emergency_jettison_permitted>
    <IBIT_available_>Yes</IBIT_available_>
    <IBIT_completion_sign>event name xyz</IBIT_completion_sign>
  </carrier>
  <store id="JCM xyz">
    <Emergency_jettison_permitted>Yes</Emergency_jettison_permitted>
    <IBIT_available_>Yes</IBIT_available_>
    <IBIT_completion_sign>event name xyz</IBIT_completion_sign>
  </store>
</catalog>
```
Store Control API (SCA)

Setup
- Identification
- Initialization

IBIT Available Period
- Data Harmonization
- Prepare For Release

IBIT Control
- Not Running IBIT
- Running IBIT

Selection Control
- Not Selected For Release
- Selected For Release

Operate
- Identification
- Initialization
- PUBIT

Relieving State
- Not Releasing
- Preparing To Release
- Releasing

Jettison State
- Not Doing Jettison
- Doing Jettison

IBIT State
- Not Doing IBIT
- Doing IBIT

Retrieve Carriage Store IBIT Results
- Retrieve Carriage Store Identity
- Retrieve Carriage Store PBIT Results
- Retrieve Carriage Store PUBIT Results
- Retrieve Mission Stores Selected For Release
- Retrieve Plug-And-Play Parameters From Carriage Store
- Set Up Release Package
- Store Present

Carriage Store Control Operations
- Apply Operational Power To Carriage Store
- Get Carriage Store State
- Get Communications Status Of Carriage Store
- Get Power Interruption Indicator Status Of Carriage Store
- Jettison Carriage Store
- Perform Carriage Store IBIT
- Provide Plug-And-Play Parameters To Carriage Store
- Remove Operational Power From Carriage Store
- Reset Power Interruption Indicator Status Of Carriage Store
- Retrieve Carriage Store Inventor
- Retrieve Carriage Store IBIT Results
- Retrieve Carriage Store PBIT Results
- Retrieve Carriage Store PUBIT Results
- Retrieve Mission Stores Selected For Release
- Retrieve Plug-And-Play Parameters From Carriage Store
- Set Up Release Package
- Store Present

Mission Store Control Operations
- Apply Fuzing To Mission Store
- Apply Conditioning Power To Mission Store
- Apply Operational Power To Mission Store
- Change Mission Store Selected For Release
- Commit Mission Store To Release
- Get Communications Status Of Mission Store
- Get Mission Store State
- Get Power Interruption Indicator Status Of Mission Store
- Harmonization Data
- Jettison Mission Store
- Mission Store Harmonization Level
- Perform Initialization Of Mission Store
- Prepare Mission Store For Jettison
- Prepare To Release Mission
- Release Mission Store
- Remove Conditioning Power From Mission Store
- Remove Operational Power From Mission Store
- Reset Power Interruption Indicator Status Of Mission Store
- Retrieve Mission Store IBIT Results
- Retrieve Mission Store PBIT Results
- Retrieve Mission Store PUBIT Results
- Retrieve Mission Store Targeting Info
- Set Release Parameters
- Store Present

Carriage Store Control Operations
- Apply Operational Power To Carriage Store
- Get Carriage Store State
- Get Communications Status Of Carriage Store
- Get Power Interruption Indicator Status Of Carriage Store
- Jettison Carriage Store
- Perform Carriage Store IBIT
- Provide Plug-And-Play Parameters To Carriage Store
- Remove Operational Power From Carriage Store
- Reset Power Interruption Indicator Status Of Carriage Store
- Retrieve Carriage Store Inventor
- Retrieve Carriage Store IBIT Results
- Retrieve Carriage Store PBIT Results
- Retrieve Carriage Store PUBIT Results
- Retrieve Mission Stores Selected For Release
- Retrieve Plug-And-Play Parameters From Carriage Store
- Set Up Release Package
- Store Present
The Plug and Play Concept - QoS

- Architecture Analysis and Design Language
  - Walk through architecture notions of a Stores Management System
  - Ask yourself “What analyses will I do?”
  - The answer will drive your modeling approach
Diagram created from ROSE using Dr. Colbert’s add-in

Rich device feature set? Enough to completely characterize a new weapon? Consider data .... Consider action language...

Can the system scale up? Capacity Planning... Queueing Network Model....

Interoperable models
Can the XML Descriptor be used to characterize a device? Consider a weapon ontology..... Process vs partitions
Weapon Process Implementation

Diagram adapted from Rhapsody

Asynchronous vs Synchronous

Notification vs polling

11
Weapon Task Implementation

Diagram adapted from Rhapsody

Safety Implications

Ordering

Acknowledgements
AADL and MDA

Application Requirements → Platform Independent Model

Platform Requirements → Transformation Engine

Validation Engine → Platform Specific Model

Non-modeled elements → Executable System

GENERAL DYNAMICS
Advanced Information Systems
Backup
Application Components

- System
  - Hierarchical organization of components

- Process
  - Protected virtual address space

- Thread group
  - Organization of threads in processes

- Thread
  - A schedulable unit of concurrent execution
Application Components

- Data
  - Potentially sharable data

- Subprogram
  - Callable unit of sequential code
Execution Platform Components

- **Processor**
  - Provides thread scheduling and execution services

- **Memory**
  - Provides storage for data and source code

- **Bus**
  - Provides physical connectivity between execution platform components

- **Device**
  - Interface to external environment