Management of Applications in Large and Heterogeneous Systems

Dario Di Crescenzo
Fabrizio Morciano
SELEX Sistemi Integrati
Agenda

- AMSM: state of problem
- AMSM: spec overview
- SELEX Sistemi Integrati experience
- Cardamom vs AMSM
Problem Statement

- Naval Combat Management Systems
- Built upon huge variety of underlying computing platforms
  - hardware and software
- Need for consistency amongst Application Management platforms
  - enable integrators to abstract from platform dependencies
- Specific naval CMS QoS constraints
The design of the specification follows the following principles:

- Maximum use possible of existing standard DMTF CIM (it is a widely accepted standard for management of software and hardware systems)
  - The specification selects a relevant subset of CIM for the CMS domain and extends it where needed.
- Inclusion of HPI-based hardware monitoring as optional PSM.
  - CIM does not model hardware elements to the detail level required by AMSM RFP.
  - A set of PSMs covering a variety of platform technologies:
    - CORBA, DDS/DCPS, XML, DMTF CIM Managed Object Format (MOF), HPI.
    - A hierarchical 3-level model of software systems, applications and software executable elements.
    - A division between design-time and run-time information of software and hardware entities.
    - A flexible deployment model allowing user defined conditions and actions to be defined.
AMSM History

- RFP issued: June 25\textsuperscript{th}, 2004
- Initial submission deadline: May 30\textsuperscript{th}, 2005, responses from:
  - Thales, SELEX, Themis, Progeny and Atlas
  - Thales, SELEX, Themis and Progeny
AMSM: spec overview
Submission overview

- Form of the submission:

  • **Platform Independent Model (PIM)**
    - in UML for data models and service interfaces

  • **5 Platform Specific Models (PSM):**
    - CORBA/IDL
    - DMTF/CIM (⇒ DMTF/WBEM)
    - DDS/DCPS
    - XML (initialisation files)
    - HPI (cross-reference with HPI initialisation files)
## Conformance Profiles

### Profiles

<table>
<thead>
<tr>
<th>Profiles</th>
<th>Normal</th>
<th>HW System Management</th>
<th>Fault Tolerance</th>
<th>Load Balancing</th>
<th>Maximum Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>LW Logging</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>AMSM Management</td>
<td>P</td>
<td>P</td>
<td>+ opt. classes and Assoc.</td>
<td>+ opt. classes and Assoc.</td>
<td>F</td>
</tr>
<tr>
<td>Supported Application Model</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>Application</td>
<td>P</td>
<td>P</td>
<td>+ opt. classes and Assoc.</td>
<td>+ opt. classes and Assoc.</td>
<td>+ opt. methods</td>
</tr>
<tr>
<td>Application Spec</td>
<td>P</td>
<td>P</td>
<td>+ opt. Classes</td>
<td>+ opt. Classes</td>
<td></td>
</tr>
<tr>
<td>Application Deployment</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>Application Deployment Spec.</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>Logical Hardware</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logical Hardware Spec.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Profile vs. Packages rules

- **A**: At least one among them
- **P**: Partial
- **F**: Full

© 2007 SELEX Sistemi Integrati. All rights reserved
## Profiles vs. Implementation rules

<table>
<thead>
<tr>
<th>Profiles vs. Implementation rules</th>
<th>Profiles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal</td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
</tr>
<tr>
<td>Must</td>
<td></td>
</tr>
<tr>
<td>May</td>
<td></td>
</tr>
</tbody>
</table>

## Implementation vs. PSMs rules

<table>
<thead>
<tr>
<th>Implementation vs. PSMs rules</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Must</td>
</tr>
<tr>
<td>Core PSMs</td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td>X</td>
</tr>
<tr>
<td>IDL</td>
<td>X</td>
</tr>
<tr>
<td>CIM</td>
<td>A</td>
</tr>
<tr>
<td>DCPS/ f</td>
<td></td>
</tr>
<tr>
<td>DCPS/ m</td>
<td></td>
</tr>
<tr>
<td>HPI</td>
<td>X</td>
</tr>
</tbody>
</table>
PIM Packages Structure

- In order to break down the overall model in a modular way such that interdependencies and complexity are minimized, two dimensions were considered:
  - Hardware vs. Software vs. Deployment (i.e., Software on Hardware)
  - Run-Time (monitoring) classes vs. Specification Classes

<table>
<thead>
<tr>
<th>Package</th>
<th>Hardware</th>
<th>Software</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run-Time</td>
<td>Logical Hardware</td>
<td>Application</td>
<td>Application Deployment</td>
</tr>
</tbody>
</table>
Packages overview
AMS Management Package

- 8 classes that provide the entry points of the AMSM service with the operations that give access to the remainder of the model
  - AMS_HWManagement and AMS_PhysicalHWManagement for hardware
  - AMS_DeploymentConfManagement for the deployment configurations
  - AMS_SystemManagement for systems
  - AMS_ApplicationManagement for applications
  - AMS_ESEManagement for executable software Elements
  - AMS_LoadBalancingManagement for load balancing groups
  - AMS_RedundancyGroupManagement for redundancy groups
AMS Application Package

• The "Application" package groups the classes needed to manage and monitor applications while they are running (information for the application definition are in “Application Specification” package).
• Since some items needed to manage applications are defined beforehand, there are links from this package to the "Application Specification" package.
• Roughly AMS_Application is designed as a set of executable software elements and/or redundancy groups and/or load balanced groups.
  – A redundancy group (AMS_RedundancyGroup) gathers executable software elements which are executed in a redundant way.
  – A load balancing group (AMS_LoadBalancingGroup) gathers executable software elements, which are executed in a load, balanced way
AMS Application – LB and FT
Starting up an application

1: GetSystem(filter:AMS_BWFilter())

2: Startup()

3: Subscribe(SystemStatus(delay:uint 6 filter:AMS_BWFilter, out subscriptionID:uint32))

4: Loop()

5: NotifyBW()

6: Unsubscribe(subscriptionID:uint32())

Platform dependent method to receive notifications.

A client that uses AMSM to perform startup and obtain status change notifications.

A filter to maintain information about application to observe has been instantiated.

Internal processing of user's application.
The "Application Specification" package groups the classes needed to model applications so they can be deployed subsequently.

- This package is a configuration view of applications.
- The main entity is the specification of an executable software element: an AMS_ESESpec
- An executable software element specification is the object which the AMSM service needs to deploy applications (i.e., create an executable software element from its specification)

- An actual executable software element will not hold a lot of information in itself since it will use its specification to keep them. Of this application information, the most important are the checks (CIM_Check) and actions (CIM_Action).
AMS Application Specification Package
CIM_Action are operations that are part of a process to start or shutdown (or deploy) a software element
Check Specification Classes

CIM_Check are conditions or characteristics that have to be true so as to deploy a Software Element.

An association class between AMS_ESESpec and either CIM_Action or CIM_Check specifies the condition (start, stop, deploy) in which the action or check will take place.
It represents the "Hardware" sub-package describing the effective hardware topology.

- These classes permit the representation of an actual network.
- The essential class is AMS_ComputerSystem which represents:
  - A computer as an aggregation of hardware elements.
  - A computer as a node in a network.
- An AMS_ComputerSystem aggregates a hardware configuration (CIM_LogicalDevice)
  - processor, memory, file systems, and gets some operating systems (AMS_OperatingSystem) which support application models
All the AMS_ComputerSystem class and sub-classes are interfaces offering monitoring methods.
There are different ways in which this PSM may be utilized is:

- Browsing software system structures: application, groups, and ESEs.
- Browsing networks and computer systems.
- Discovery and configuration of networks and computers.
- Software data inventory.
- Display of computers and/or applications statuses.
- GUI-based management of applications and computers.

These uses may be gathered in two main purposes:

- Getting information from a database of software and hardware
  - Need of interfaces to get all the attributes and to iterate on all the associations

- Managing some of these element: applications, ESEs, computers
  - Need for a way to quickly retrieve elements of the object
The PSM is divided into two parts:

- The DCPS/f ('/f' stands for full) PSM is intended to be equivalent to the CORBA/IDL and CIM profile, and thus compliant implementations are not required to deploy any elements of the CORBA and/or CIM profiles.
- The DCPS/m ('/m' stands for monitoring) PSM is a subset of the DCPS/f PSM, and contains those elements which are required for asynchronous monitoring of states of the different (software and hardware) elements.
  - The DCPS/m PSM is defined to allow other PSMs (CORBA and/or CIM) to import it and use it for asynchronous monitoring tasks.
  - The inclusion of DCPS/m profile in CORBA and/or CIM PSM is not a mandatory, but an optional (convenience) mechanism.
- Typically, the integration will be implemented through the various “subscribe” methods in the CORBA and CIM PSM, which, in case when DCPS/m PSM is included, will result in subscription (registration of interest) to the relevant DCPS topics from the DCPS/m profile.
XML PSM

- Normalize the format of the files which can be read or written by an AMSM service
- The uses of these files by an AMSM service are threefold:
  - May be the configuration files allowing the user of the AMSM service (integrator…) to initialize the service with
    - Software system specifications
    - Application specifications
    - Deployment specifications,
    - A (first) drawing of the network.
  - May be used as a backup capability allowing an AMSM service to be re-started with its previously recorded state.
  - May be used to exchange data amongst multiple instantiations of the AMSM service.
The whole picture
What’s next

– Now undergoing finalisation
– Possible scope for future revisions of the standard
  • **dynamic** deployment and creation of software specification
  • **dynamic** hardware discovery
  • **dynamic** creation of software specification
  • multiple **cooperating or competing** AMSM services
    – API to **exchange** data among AMSM services
  • “**low-weight**” profile
    – hardware definition cuts to the minimum
    – no “check” classes …
  • new “Action” type: On Error
  • new specific “Action” classes in order to deploy and run **component packages** (CCM D&C, J2EE)
  • new classes of devices
So what is it?

- A **complete solution** for application management and system monitoring of near real-time (naval) CMS and C4I systems

- Object model based on **worldwide know-how** on naval CMS
  - about 60 specific classes
  - 50 classes extracted from DMTF/CIM standard

- Several kinds of implementation foreseen
  - CIM/HPI: **interoperability with today’s management tools**
  - CORBA and DDS: **integration in today’s systems**
  - XML: **initialisation and exchange streams**
Expected System Management capabilities:

- **System Configuration**
  - initial definition of the system configuration
  - runtime modification of system configuration (where applicable)

- **System Control**
  - start-up and stop of the whole system or of a system subset
  - shutdown and reboot of nodes

- **System Monitoring**
  - Monitoring of processes
  - Monitoring of nodes

- **System State report and Notification**
  - get the system state upon request (Administration HMI)
  - be automatically notified when a specific event happens (Observer)
Complex System: a real case

This picture shows the complexity and hierarchy of elements that a System Management framework should be aware of and should be able of:

1) configure,
2) control,
3) monitor,
4) generate reports or alerts about.
The Cardamom Solution

- Cardamom a open source, CORBA based middleware to deploy near real-time application
- Jointly developed by SELEX and Thales
- Addresses the problem to perform AMSM via CORBA
The Big Picture

Abstraction Layers
- ORB
- OS

Container

Abstraction Layers

Pluggable Services

FAULT TOLERANT OBJECTS & COMPONENTS

Basic Services
- System Management

Load Balancing  Trace  Enhanced View Of Time  Recording  Transaction  Event  DDS  Life Cycle  Naming
Cardamom Approach in OMG

ORB

CARDAMOM Services

CORBA Standard Services

CARDAMOM Prototyping Validation

OMG RFP

© 2007 SELEX Sistemi Integrati. All rights reserved
What in Cardamom

What in AMSM

ManagedElement

CIM_Configuration

CIM_Setting

CIM_ManagedSystemElement

ASM_ApplicationSpecification

ASM_ExecutableSoftwareElementSpecification

ASM_SoftwareSystemSpecification

CIM_LogicalElement

RedundancyGroup

CIM_EnableLogicalElement

CIM_Software

ASM_ExecutableSoftwareElement

Process

ASM_ManagedObject

CIM

CIM_OperatingSystem

CIM_System

CCMComponent

System

Application

ComponentServer

CIM_J2eeApplication

CIM_J2eeServer
AMS and Cardamom

• **PRO**
  – Uses a CORBA-like profile
  – Addresses the AMSM capabilities over Large Systems
  – Provides functionalities to use CCMComponents

• **CONS**
  – Hardware management
Thanks for your attention