OpenSAF Platform Management

Anil Sadineni

GoAhead Software

Agenda

- Features supported/unsupported in OpenSAF 4.0
- PLM Architecture
- PLM Design
- Admin Operation Flow
- Information Modeling/Configuration
- Demonstration

OpenSAF™ The Open Service Availability Framework

PLM in a big picture

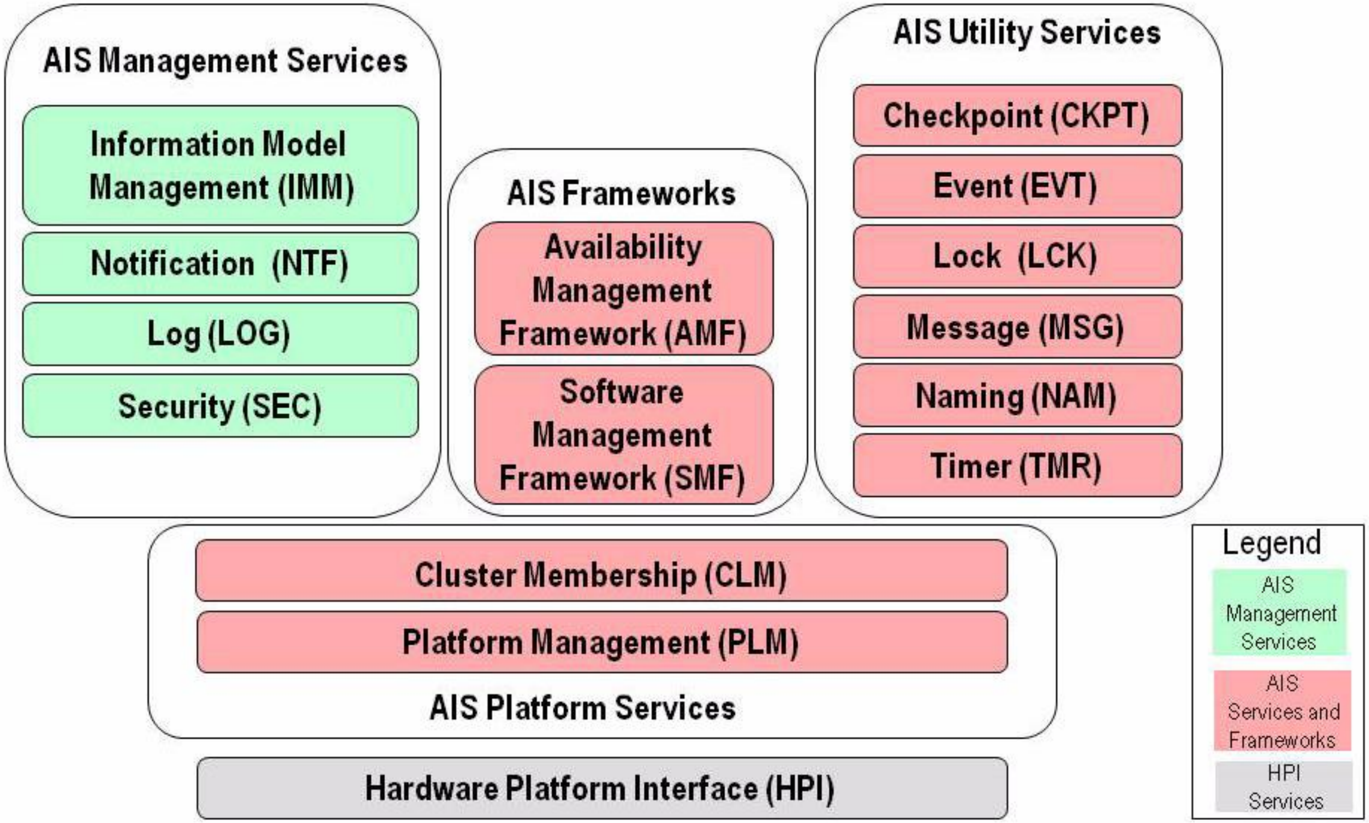
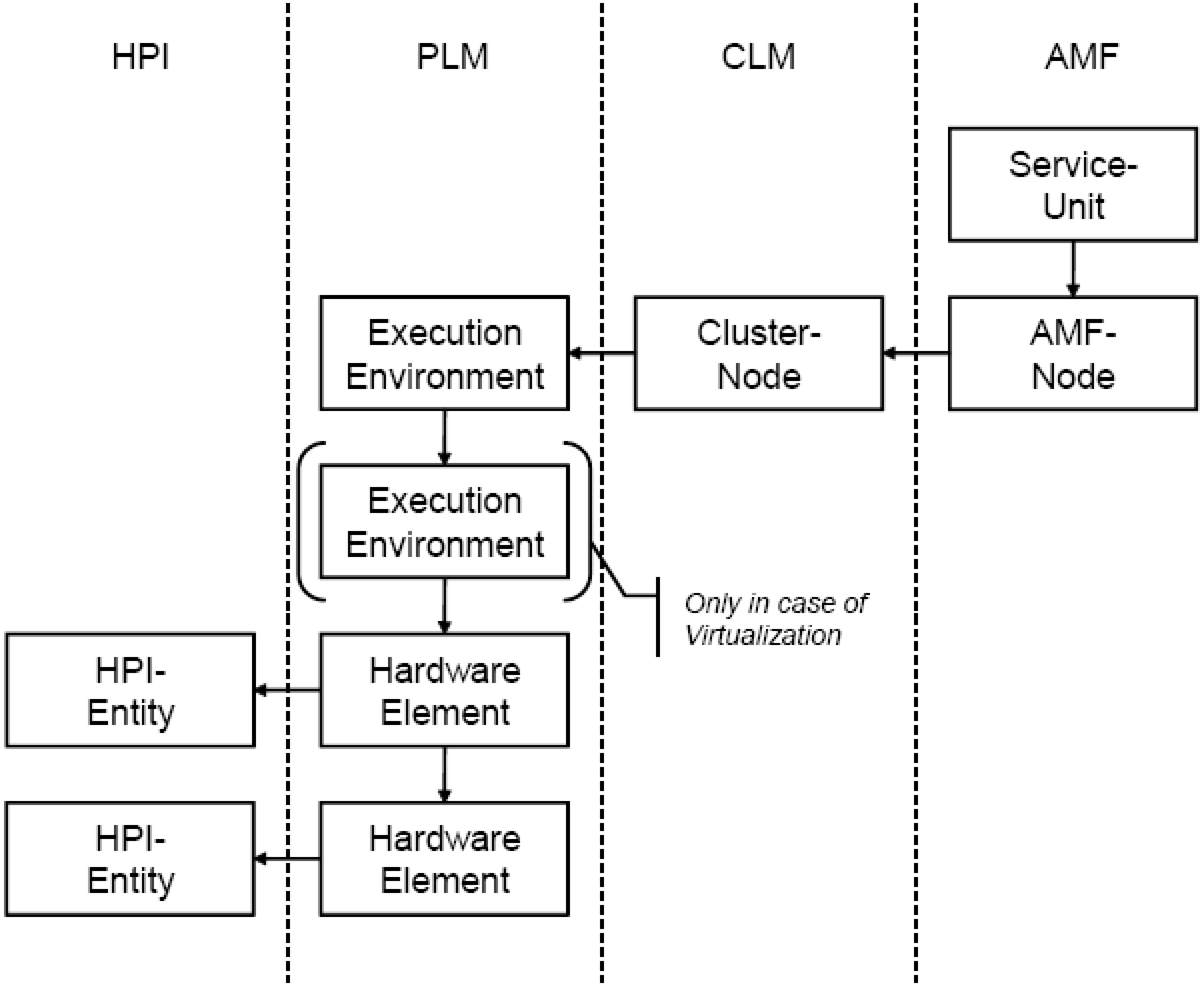


Figure 2: Overview of HPI and AIS Services of SAI-Overview-B.05.01 Section 3.2.2.2

PLM Position in the Information Model



Supported features

- Modeling of hardware entities as HE objects using SAF IMM model.
- Modeling of execution environment EE as child of HE or child of Domain object using SAF IMM model.
- Discovery of hardware and management of hot swap information of hardware using HPI-B interface.
- Verification of discovered hardware entities and execution environments against configured HE and EE objects respectively
- HE and EE states management (Presence, Operational, Readiness, Administrative states and Readiness Flags)
- Administrative operations on HEs and EEs.
- Fault isolation of HEs and EEs

Supported features contd..

- Entity group management, Readiness status tracking and Readiness Impact operations.
- Supports all kinds of dependencies.
- Supports following hot-swap models
  - TWO_HOTSWAP_MODEL
  - THREE_HOTSWAP_MODEL
  - PARTRIAL_FIVE_HOTSWAP_MODEL
  - FULL_FIVE_HOTSWAP_MODEL
- Ability to run even in absence of HPI daemon.
- State Change notifications and HPI event notifications using SAF Notification service.
- HE and EE Alarms for hardware faults (only physical extractions and isolated faults) and unmapped HE Alarms using SAF Notification service.

Unsupported features

- Health monitoring of HEs and EEs
- EE Virtualization
- HE and EE security alarms
- Multi-chassis systems
- Multi-domain systems
- Configuration of entity path fragments and wild-card characters in entity-paths attribute of HE object
- Simultaneous administrative operations on HE/EE and its children

PLM Architecture

- PLM Service is designed and implemented in 2-tier architecture (Server and agent)
- PLM Server (PLMS) runs only on controllers and PLM Agent (PLMA) runs as part of application
- PLM Coordinator (PLMc) runs on each EE

PLM Server

- Uses HPI to interact with hardware entities (HEs)
- Uses PLMc to interact with execution environments (EEs)
- Reads the configuration of PLM objects from IMM as Object Manager
- Does the verification of HEs and EEs
- Manages HE and EE states
- Performs administrative operations invoked on HEs and EEs
- Maintains the information of entity groups and the clients/applications tracking the changes to these groups
- Registers with IMM as Object Implementer for HEs and EEs

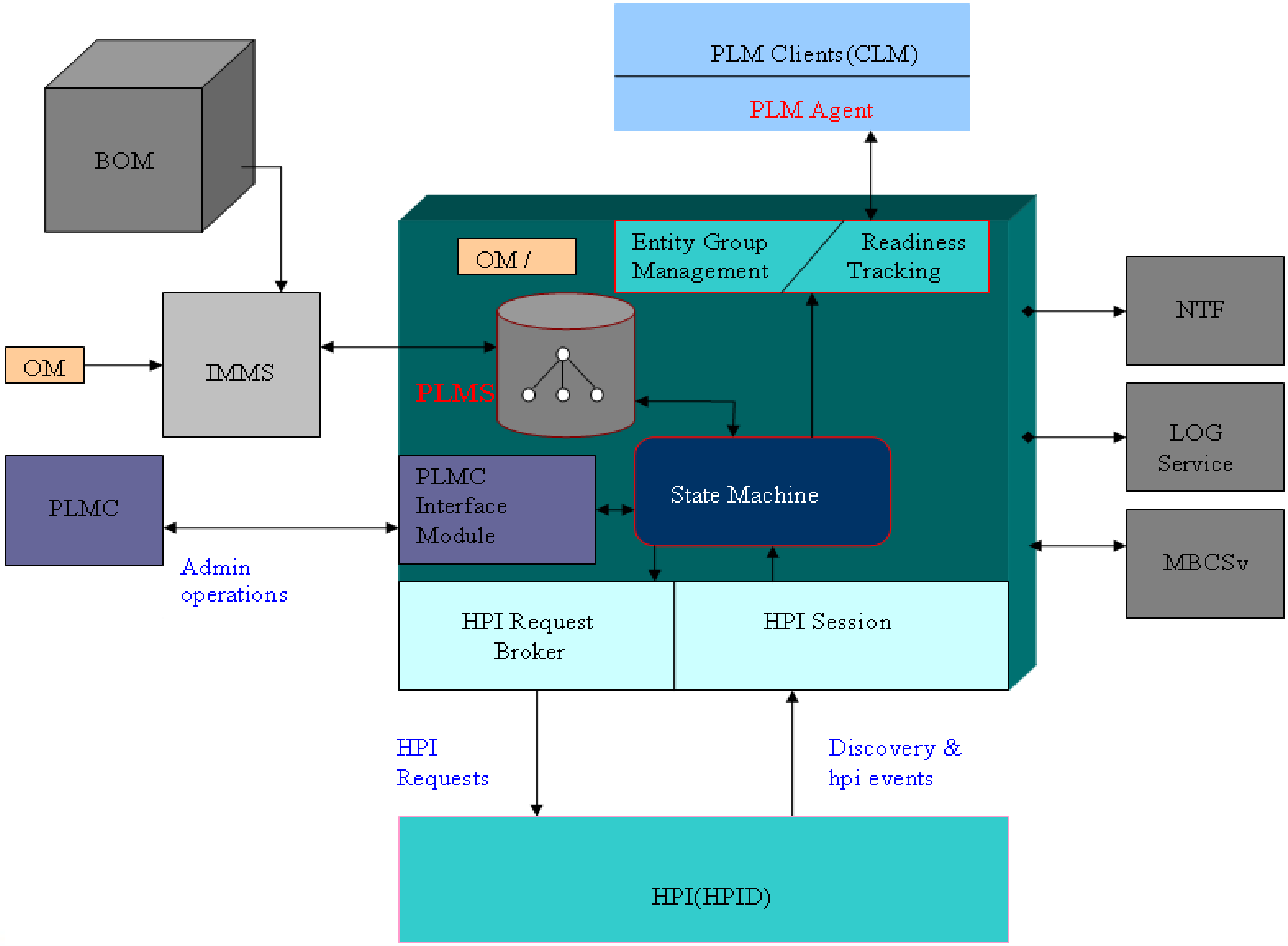
PLM Agent

- Implements PLM Library Life Cycle APIs
- Allows applications to create/delete entity groups, add/remove members in the entity groups
- Allows applications to start and stop tracking the changes to the entities in the groups
- Allows applications to inform readiness state changes of entities to PLM using Readiness Impact API

PLM Coordinator

- Coordinates EE verification by reading configuration information of EE and sending it to PLMS
- Coordinates administrative operations on EE
- Uses TCP/UDP socket communication with PLMS
- Does not use any infrastructure services of OpenSAF

PLM Design


PLM Clients(CLM)
PLM Agent
BOM
OM /
Entity Group Management
Readiness Tracking
OM
IMMS
PLMS
NTF
LOG Service
PLMC
PLMC Interface Module
State Machine
MBCSv
Admin operations
HPI Request Broker
HPI Session
HPI Requests
Discovery & hpi events
HPI(HPID)

IMM interface

- Registers with IMM as Object Manager, reads the PLM configuration at startup on active and when standby becomes active (during failovers and switchovers)
- Builds the internal data structures with the configured information.
- Registers with IMM as Object Implementer for PLM objects and processes any CCB create/delete/modify requests on these objects

HPI Interface

- PLMS uses HPI for resource discovery and hardware management
- HPI interface module has following sub-modules
  - HPI Session Manager (HSM)
  - HPI Request Broker (HRB)
- Implemented as a dynamic library which can be loaded at runtime

HPI Session Manager (HSM)

- Opens a session with HPI daemon.
- Discovers the hardware resources available.
- Receives HPI events and forwards hot-swap events to PLMS.
- Discovers hot-swap model supported by each resource dynamically.
- Retrieves FRU Inventory data on receiving Insertion-pending or Active hot swap event

HPI Request Broker (HRB)

- Provides a set of routines to PLMS to perform HPI operations.
- Receives HPI requests from PLMS and invokes corresponding HPI API

PLMc Interface Module

- Provides a set of API to PLMS to communicate with PLMc.
- Uses sockets to communicate with PLMc.
- Listens on a configured ports to receive connections from PLMc on EEs.

Notifications

- PLMS generates the notifications for the following
  - Any state (presence, readiness, operational, administrative) changes of entities
  - For every HPI hot swap event of entities
  - Unmapped HE Alarm

Entity Group Management

- Applications can use following API to manage entity groups
  - saPlmEntityGroupCreate()
  - saPlmEntityGroupAdd()
  - saPlmEntityGroupRemove()
  - saPlmEntityGroupDelete()

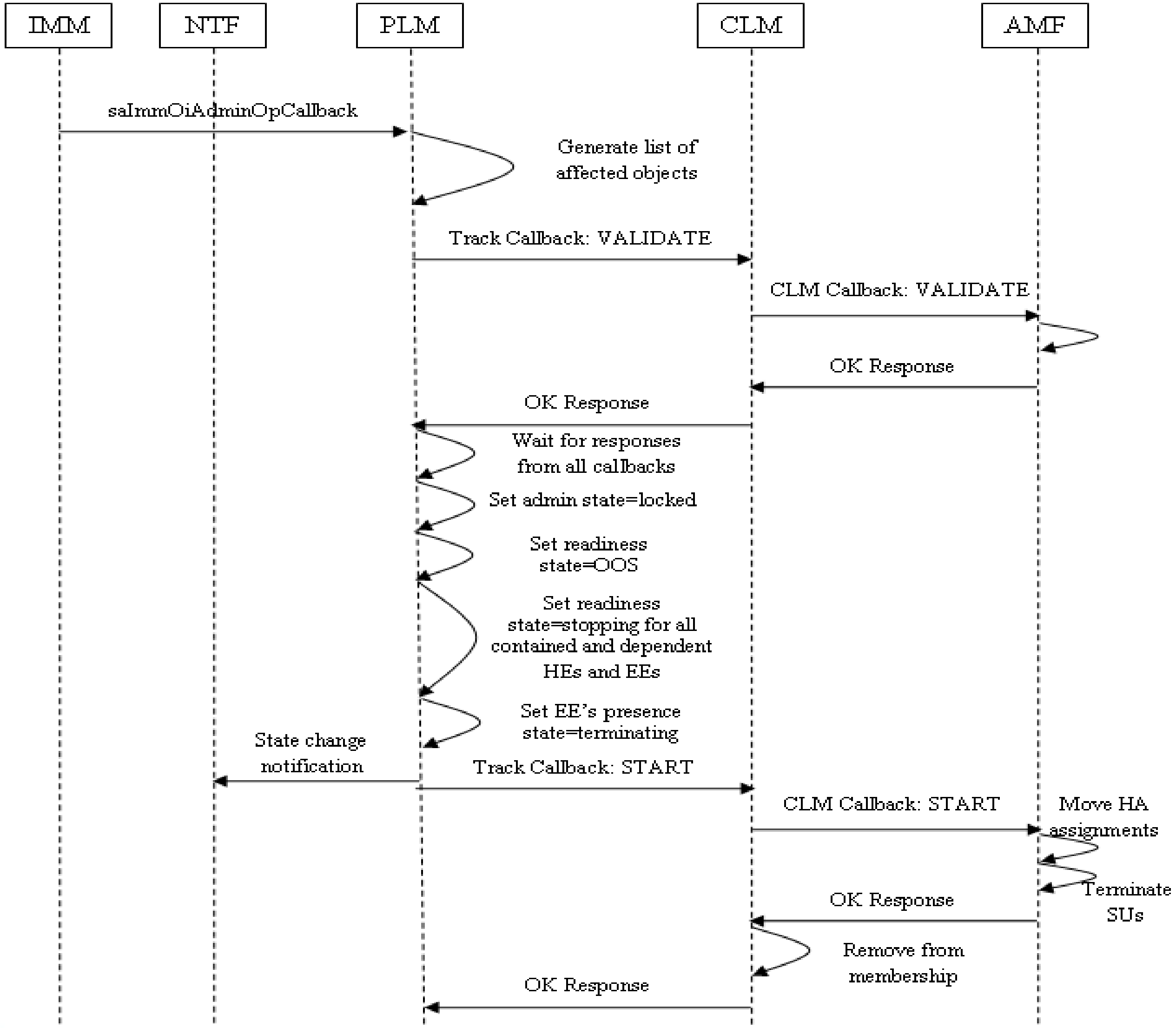
Readiness Status Tracking

- Applications can track the readiness status changes of the entities using Readiness Track API
  - saPlmReadinessTrack()
  - SaPlmReadinessTrackCallbackT
  - saPlmReadinessTrackResponse
  - saPlmReadinessTrackStop
- Three kinds of flags can be used to track the readiness state
  - TRACK_CURRENT
  - TRACK_CHANGES
  - TRACK_CHANGES_ONLY
- PLM responds with the readiness state changes synchronously (in one instance of TRACK_CURRENT flag usage) or asynchronously using the Readiness Callback API in the SA_PLM_CHANGE_COMPLETED step

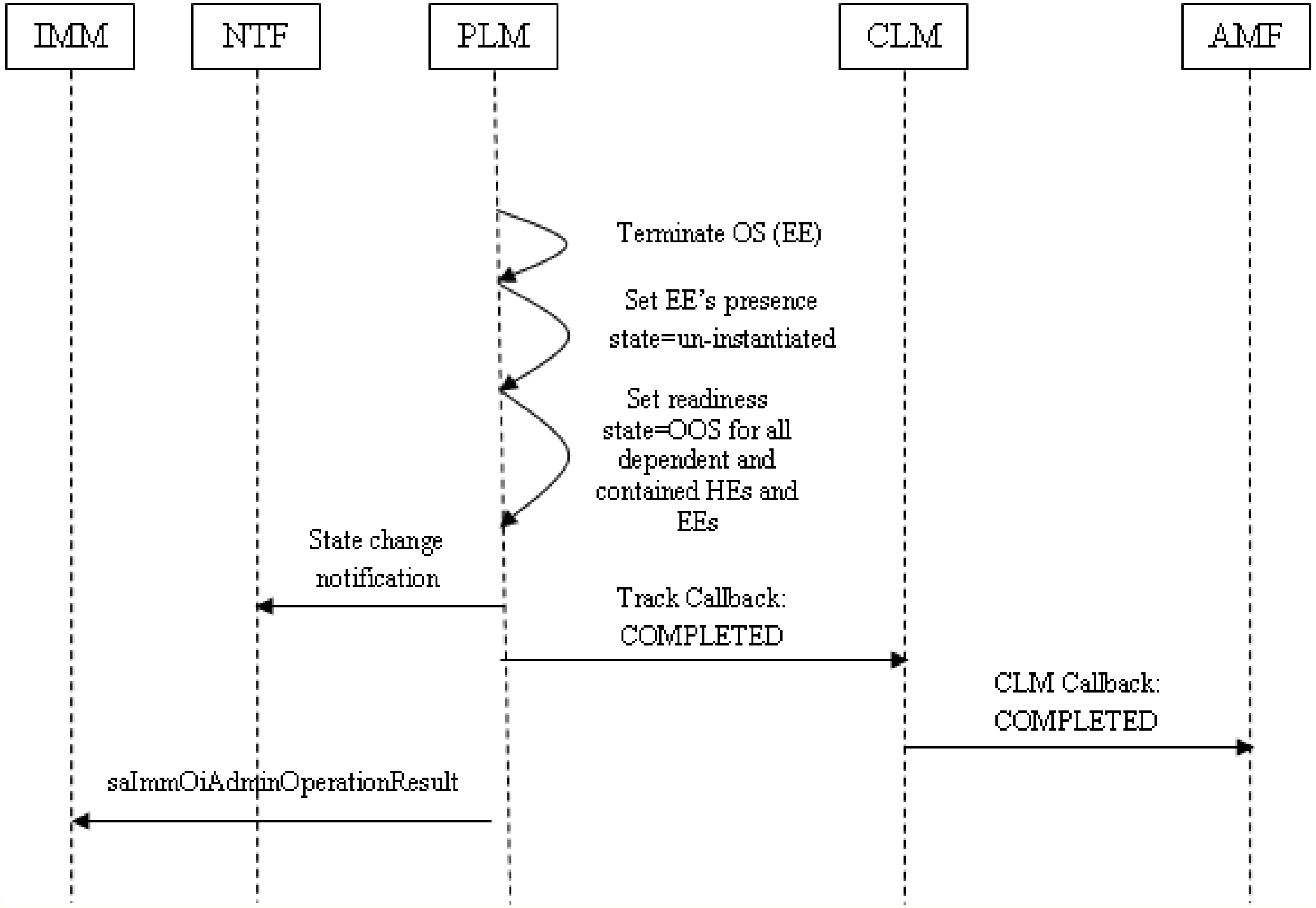
Readiness Status Tracking contd..

- Applications can use the flags SA_TRACK_VALIDATE_STEP and SA_TRACK_START_STEP to get a chance to validate the change and get ready before the change respectively.
- Readiness Callback API is invoked with maximum of three steps depending on the cause which triggered the readiness change
  - SA_PLM_CHANGE_VALIDATE
  - SA_PLM_CHANGE_START or SA_PLM_CHANGE_ABORTED
  - SA_PLM_CHANGE_COMPLETED
- Applications can use saPlmEntityReadinessImpact() API to report the change in state of health of an entity

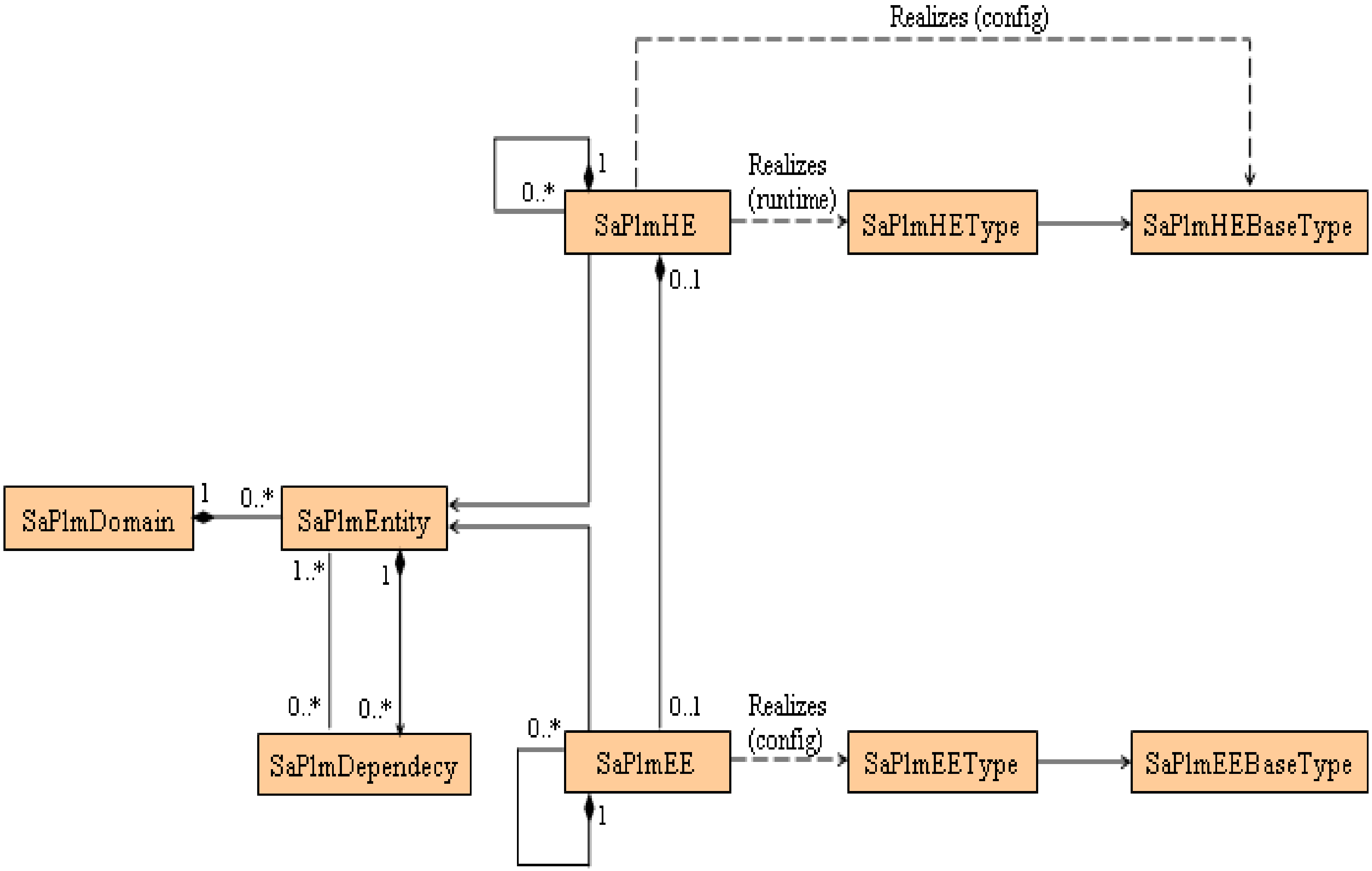
Lock operation on HE



Lock operation on HE contd..


IMM
NTF
PLM
CLM
AMF
Terminate OS (EE)
Set EE's presence state=un-instantiated
Set readiness state=OOS for all dependent and contained HEs and EEs
State change notification
Track Callback: COMPLETED
CLM Callback: COMPLETED
saImmOiAdminOperationResult

Information Modeling



Information Modeling contd..

- Typically each FRU (Field Replaceable Unit) is modeled as HE
- SaPlmHEBaseType and SaPlmEEBaseType objects are configured to represent the functionality of HE and EE respectively
- SaPlmHEType and SaPlmEEType objects reflect the implementation specific aspects of HE and EE respectively
- Each HE can have at most one EE as a child which is booted when HE starts up
- EE can be configured directly under the SaPlmDomain object if no HEs are configured

Information Modeling contd..

- Configure the saPlmHEBaseHEType attribute of HE with the value which is the DN of the configured HEBaseType object
- Configure the saPlmHEEntityPaths attribute of HE with the value/values which will be the entity path published by the HPI
- Configure IDR information in the saPlmHetIdr attribute of HE as a string as mentioned in the PLM specification
- Configure saPlmEEType attribute of EE with the value which is the DN of the configured EEType object

Sample configuration - HEs

```
<object class="SaPlmHEBaseType">
            <dn>safHEType=atca-7150</dn>
            <attr>
                 <name>saPlmHetHpiEntityType</name>
                <value>SAHPI_ENT_FRONT_PANEL_BOARD</value>
            </attr>
</object>

<object class="SaPlmHEType">
            <dn>safVersion=7150_ver1,safHEType=atca-7150</dn>
            <attr>
                 <name>saPlmHetIdr</name>
                 <value>PRODUCT/MANUFACTURER=MOTOROLA,PRODUCT
                    /SERIAL_NUMBER=2130125A3KWJ,BOARD/PRODUCT_NA
                    ME=ATCA-7150</value>
            </attr>
</object>
```

Sample configuration - HEs contd..

```
<object class="SaPlmHE">
    <dn>safHE=7150_slot_3,safDomain=domain_1</dn>
    <attr>
            <name>saPlmHEEntityPaths</name>
            <value>{{SAHPI_ENT_PICMG_FRONT_BLADE,0},{SAHPI_ENT_
            PHYSICAL_SLOT,3},{SAHPI_ENT_ADVANCEDTCA_CHASSIS,2},
            {SAHPI_ENT_ROOT,0}}</value>
    </attr>
    <attr>
          <name>saPlmHEBaseHEType</name>
          <value>safHEType=atca-7150</value>
    </attr>
</object>
```

Sample configuration -EEs

```
<object class="SaPlmEEBaseType">
        <dn>safEEType=Linux_os</dn>
        <attr>
            <name>saPlmEetProduct</name>
            <value>os</value>
        </attr>
        <attr>
            <name>saPlmEetVendor</name>
            <value>Windriver</value>
        </attr>
        <attr>
            <name>saPlmEetRelease</name>
            <value>2.6.18-8.el5</value>
        </attr>
</object>
```

Sample configuration – EEs contd..

```
<object class="SaPlmEEType">
                <dn>safVersion=1,safEEType=Linux_os</dn>
                <attr>
                        <name>saPlmEetDefInstantiateTimeout</name>
                        <value>1000000</value>
                </attr>
                <attr>
                        <name>saPlmEetDefTerminateTimeout</name>
                        <value>1000000</value>
                </attr>
</object>
```

Sample configuration - EEs contd..

```
<object class="SaPlmEE">
    <dn>safEE=os,safHE=7150_slot_3, safDomain=domain_1</dn>
    <attr>
                <name>saPlmEEType</name>
                <value>safVersion=1,safEEType=Linux_os</value>
    </attr>
    <attr>
                <name>saPlmEEInstantiateTimeout</name>
                <value>180000000000</value>
    </attr>
    <attr>
                <name>saPlmEETerminateTimeout</name>
                <value>180000000000</value>
    </attr>
</object>
```

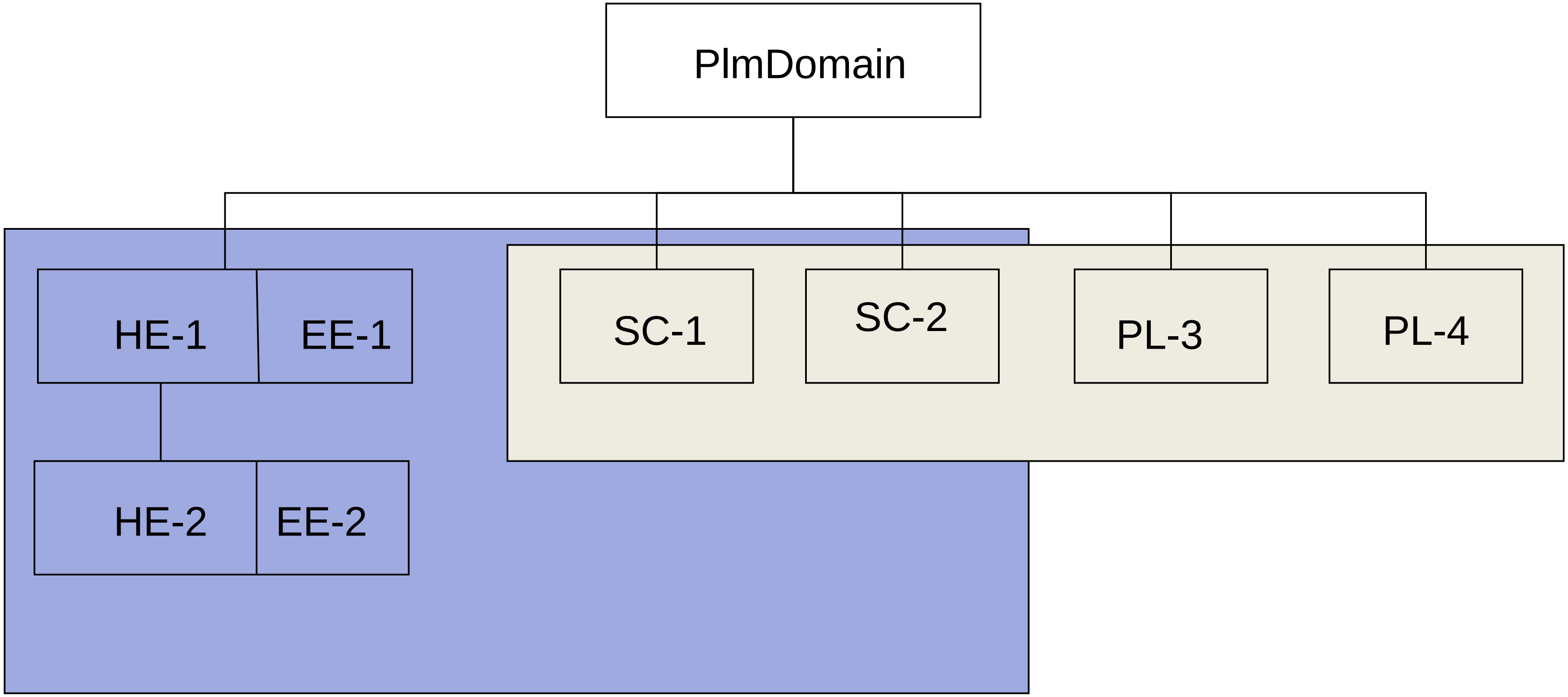
Sample configuration - Dependency

```
<object class="SaPlmDependency">

    <dn>safDependency=dep_1,safEE=os,safHE=7150_slot_3,safDomain=domain_1</dn>

           <attr>
                 <name>saPlmDepNames</name>
                 <value>safEE=os,safHE=slot_1,safDomain=domain_1</value>
                 <value>safEE=os,safHE=slot_2,safDomain=domain_1</value>
           </attr>
           <attr>
                 <name>saPlmDepMinNumber</name>
                 <value>1</value>
           </attr>
</object>
```

Demonstration Overview

- plm_demo is a trivial application that tracks readiness status of 6 entities (2 controllers, 2 HEs and 2 EEs on those HEs)
- Track flags are
  SA_TRACK_CURRENT | SA_TRACK_CHANGES
- It exercises the following API
  - saPlmInitialize
  - saPlmSelectionObjectGet
  - saPlmEntityGroupCreate
  - saPlmEntityGroupAdd
  - saPlmReadinessTrack
  - saPlmDispatch
- lock/unlock admin operations on an EE

Topology of Entities in Demo



Thank You