Resource Functions II.

High Availability and Timeliness in Linux
Gyula Bajor
09.04.2003.

Discovery of Resources

The discovery of resources is accomplished by accessing two data structures: The Resource Presence Table (RPT) associated with a domain, and the Resource Data Records (RDRs) associated with resource.

The RPT is maintained for each domain defined in the interface. In RPT there is an entry for each resource that is a member of the corresponding domain. An RPT may also contains entries that reference additional domains available through the interface. The user can discover all the resources available in the platform, even if some of them are included in a different domain. In order to support this full resource discovery each RPT includes domain reference entries. If the user accesses the "default domain" defined in the specification then accesses all the domains referenced in the RPT of that domain, and continues this process for all those domains. Therefore all domains available in the system will be discovered.

Service Availability middleware or other carrier-grade software will need to create an internal model of the entire system; that is the "physical view" of the system. This complete discovery can be done by invoking saHpiResourcesDiscover function.

This function may be called during operation to regenerate the RPT table. For those FRUs that must be discovered by polling, latency between FRU insertion and actual addition of the resource associated with that FRU to the RPT exists. To overcome this latency, a discovery of all present resources may be forced by calling saHpiResourcesDiscover.

Event Generation and Logging

Users of the HPI need subscribe to receive events from the HPI on a domain-by-domain basis. After subscribing to events the HPI sets up an event queue for the user, and places a copy of all events related to any resource that is member of that domain on the queue as they are generated. A user may read events from the queue via a blocking or non-blocking call. If a user process makes a blocking call, that process will wait until an event is received, and will then be "woken up" to process, the received event.

Users need not poll the various management instruments in the system, because event messages are generated for all “significant occurrences” in the system platform. In order to operate this way, the user must be aware of the “initial state” of the system. The HPI provides a special feature that allows the user to learn this initial state via processing event messages. When the user subscribes to receive events from a domain, a flag may be set that requests the HPI to place events on the user's event queue for all active alarms in the system. This effectively recreates events that occurred prior to the subscription request, but that the user “needs to know about” if it is not going to poll all sensors for their current state.

Exactly what events are placed in the resource and domain system event logs is implementation-specific. All system event logs, either the domain system event log, or a resource system event log may be managed by the user through these HPI functions. Management of a system event log includes activities such as reading records from it, writing records to it, clearing it, setting the timestamp clock, etc.

In addition to forwarding events to the subscribers, the HPI also logs events in a non volatile memory associated with each domain. This event log is accessible by users through the HPI at any time, to retrieve a historical record of events that have been generated for a domain.

Hot Swap Capabilities

Managing hot swappable devices and a changing platform configuration is one of the key features of the Service Availability Forum Platform Interface. A component that can be added or removed from the system is called a Field Replaceable Unit (FRU) in the specification. FRUs are always modeled as a separate resource allowing the interface to dynamically adjust as the hardware platform configuration changes. Resources which support hot swap events and functions will have the “Managed Hot Swap” capability set in their RPT entry. Including “Managed Hot Swap” flag indicates that the resource follows the hot swap model and usage described in this section and can be managed using the functions described below.

When a FRU is inserted in the system, the HPI will add a corresponding resource to a domain, and will generate an event message. User software can then access the RDRs in that newly added resource to discover its capabilities.

Similarly, when a FRU is removed from the system, the corresponding resource is removed from any domains in which it had membership, and appropriate events are sent. User software thus is aware that all the management instruments hosted by that resource are no longer available in the system, and the entities they are associated with have been removed.

The HPI supports two models of the actual hot-swap activity. For FRUs that require special processing related to their insertion or extraction, the full hot swap model may be used, as shown in Figure 1. This model includes five states: Not Present, Insertion Pending, Present, Extraction Pending, and Inactive.

A simplified hot-swap model is used for FRUs that do not require any special processing as they are inserted or removed. The state diagram for this model is shown in Figure 2.

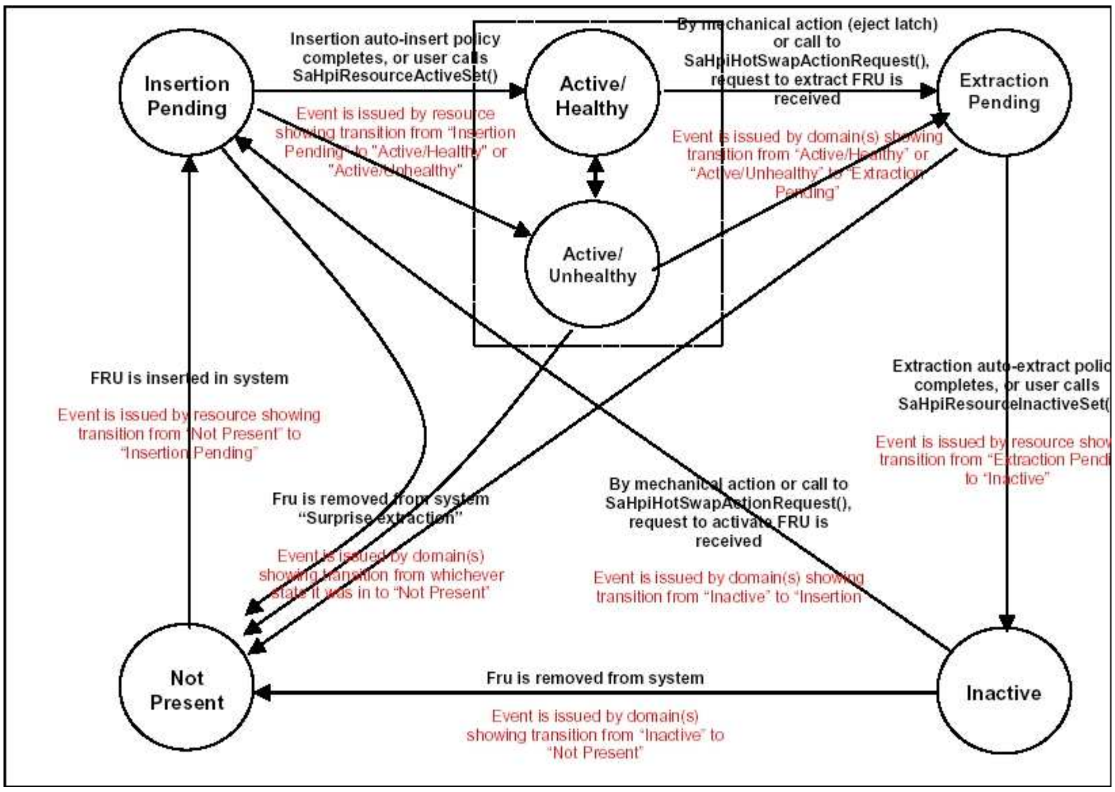
Insertion Pending
Active/ Healthy
Extraction Pending
Active/ Unhealthy
Not Present
Inactive
Insertion auto-insert policy completes, or user calls SaHpiResourceActiveSet()
Event is issued by resource showing transition from "Insertion Pending" to "Active/Healthy" or "Active/Unhealthy"
By mechanical action (eject latch) or call to SaHpiHotSwapActionRequest(), request to extract FRU is received
Event is issued by domain(s) showing transition from "Active/Healthy" or "Active/Unhealthy" to "Extraction Pending"
FRU is inserted in system
Event is issued by resource showing transition from "Not Present" to "Insertion Pending"
Extraction auto-extract polic completes, or user calls SaHpiResourceInactiveSet(
Event is issued by resource show transition from "Extraction Pendi to "Inactive"
Fru is removed from system "Surprise extraction"
Event is issued by domain(s) showing transition from whichever state it was in to "Not Present"
By mechanical action or call to SaHpiHotSwapActionRequest(), request to activate FRU is received
Event is issued by domain(s) showing transition from "Inactive" to "Insertion
Fru is removed from system
Event is issued by domain(s) showing transition from "Inactive" to "Not Present"

Figure 1. Full Hot Swap model

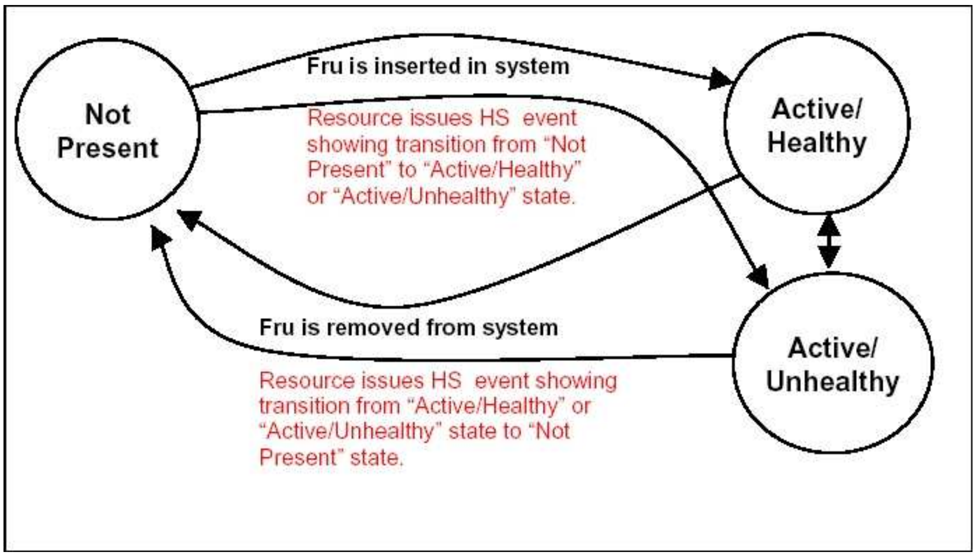
Not Present
Active/ Healthy
Active/ Unhealthy
Fru is inserted in system
Resource issues HS event showing transition from "Not Present" to "Active/Healthy" or "Active/Unhealthy" state.
Fru is removed from system
Resource issues HS event showing transition from "Active/Healthy" or "Active/Unhealthy" state to "Not Present" state.

Figure 2. Simplified Hot Swap Model

*NOT PRESEN*T

The NOT PRESENT state is actually a virtual state that represents a resource that is not currently present in the domain, because the FRU associated with that resource is not currently present in the system. A resource is in this state before the FRU is physically inserted into the system or if it has been removed from the system. A resource typically transitions to this state from the INACTIVE state, but a resource can transition to this state from any state due to a surprise extraction of the FRU. The HPI implementation should generate a hot swap event with SAHPI_HS_STATE_NOT_PRESENT when the resource transitions from any state to the NOT PRESENT state. The event can indicate a normal transition from INACTIVE to NOT PRESENT or a surprise extraction from any other state.

INSERTION PENDING

The INSERTION PENDING state is entered after a resource has been added to the domain, as a result of the associated FRU being physically inserted into the system. This state indicates the resource is transitioning from a NOT PRESENT or INACTIVE state into the ACTIVE state. When transitioning into the INSERTION PENDING state, the resource should generate a hot swap event with SAHPI_HS_STATE_INSERTION _PENDING. The event can be generated when the ejector latch is shut (cPCI) or the device is seated in a slot. Upon receiving the event, middleware has the opportunity to discover the capabilities of the resource before allowing the FRU associated with the resource to power on and become an active component in the system. During this state, the FRU can be commanded to power on or de-assert reset.

ACTIVE / HEALTHY, ACTIVE / UNHEALTHY

The ACTIVE/HEALTHY and ACTIVE/UNHEALTHY states indicate that a resource is now an active member of the domain. After a FRU completes the hardware connection process, the associated resource enters an ACTIVE/HEALTHY state if no faults are present. This does not mean that the FRU is now active at the software level, but merely indicates that the FRU is now active in the system, is healthy, and that it should not be abruptly removed. The HPI implementation generates a hot swap event with HotSwapState = SAHPI_HS_STATE_ACTIVE_HEALTHY when the resource transitions to the ACTIVE_HEALTHY state from any state other than ACTIVE/UNHEALTHY. The HPI implementation generates a hot swap event with HotSwapState = SAHPI_HS_STATE_ACTIVE _UNHEALTHY when the resource transitions to the ACTIVE/ UNHEALTHY state from any state other than ACTIVE/HEALTHY. This indicates that the FRU is now available for use, but contains a fault that may or may not prevent the FRU from functioning properly. A resource will transition between ACTIVE/HEALTHY and ACTIVE/UNHEALTHY as fault conditions are asserted or cleared on the FRU. No events are generated on transitions between ACTIVE/HEALTHY and ACTIVE/UNHEALTHY because it is assumed that other events related to the specific fault being asserted or cleared will be issued, and the hot swap event noting this state transition would be redundant.

EXTRACTION PENDING

The EXTRACTION PENDING state indicates that the resource has requested extraction of the associated FRU. Typically, a resource enters an extraction pending state when an ejector latch is opened (PICMG 2.1) or when a hot swap button is pressed. The HPI implementation should generate a hot swap event with HotSwapState = SAHPI_HS_STATE _EXTRACTION_PENDING when a resource that supports Managed Hot Swap requests extraction. Upon receiving the event, middleware has the opportunity to unload software drivers, relocate processes, or unmount file systems (software disconnect) before allowing a FRU to power down and disconnect from the system.

INACTIVE

The INACTIVE state indicates that the FRU is no longer active in the system, and that it has completed the extraction process. When a FRU completes the hardware disconnection process, it is logically and electrically disconnected or isolated from the platform but still physically located in the platform, so the associated resource remains in the domain. Typically, a FRU will be powered off or held in reset when in this state. The HPI implementation should generate a hot swap event with HotSwapState = SAHPI_HS_STATE_INACTIVE when the resource is transitioning to an INACTIVE state.

Hot Swap Functions

HPI defines a set of routines for managing the hot swap connection/ disconnection process.

Function **SaHpiHotSwapControlRequest** allows the caller, after receiving a hot swap event with HotSwapState equal to SAHPI_HS_STATE_INSERTION_PENDING or SAHPI_HS_STATE_EXTRACTION_PENDING, to request control of the hot swap policy and prevent the default policy from being invoked. Because a resource that supports the simplified hot swap model will never transition into Insertion Pending or Extraction Pending states, this function is not applicable to those resources. A resource supporting hot swap typically supports default policies for insertion and extraction. On insertion, the default policy may be for the resource to turn the associated FRU s local power on and to de-assert reset. On extraction, the default policy may be for the resource to immediately power off the FRU and turn on a hot swap indicator.

Function **saHpiResourceActiveSet** can be used to request a resource to return to the ACTIVE/HEALTHY or ACTIVE/UNHEALTHY state from the EXTRACTION PENDING state in order to reject an extraction request. During insertion, a resource supporting hot swap will generate an event to indicate that it is in the INSERTION PENDING state. If the management middleware or other user software calls saHpiHotSwapControlRequest() before the resource begins an auto-insert operation, then the resource will remain in INSERTION PENDING state while the user acts on the resource to integrate it into the system. During this state, the user can instruct the resource to power on the associated FRU, to de-assert reset, or to turn off its hot swap indicator using the saHpiResourcePowerStateSet(), saHpiResourceResetStateSet(), or saHpiHotSwapIndicatorStateSet() functions, respectively. Once the user has completed with the integration of the FRU, this function must be called to signal that the resource should now transition into ACTIVE/HEALTHY or ACTIVE/UNHEALTHY state (depending on whether or not there are active faults). Because a resource that supports the simplified hot swap model will never transition into Insertion Pending or Extraction Pending states, this function is not applicable to those resources.

Function **saHpiResourceInactiveSet** can be used to request a resource to return to the INACTIVE state from the INSERTION PENDING state to abort a hot-swap insertion action. During extraction, a resource supporting hot swap will generate an event to indicate that it is in the EXTRACTION PENDING state. If the management middleware or other user software calls saHpiHotSwapControlRequest() before the resource begins an auto-extract operation, then the resource will remain in EXTRACTION PENDING state while the user acts on the resource to isolate the associated FRU from the system. During this state, the user can instruct the resource to power off the FRU, to assert reset, or to turn on its hot swap indicator using the saHpiResourcePowerStateSet(), saHpiResourceResetStateSet(), or saHpiHotSwapIndicatorStateSet() functions, respectively. Once the user has completed the shutdown of the FRU, this function must be called to signal that the resource should now transition into INACTIVE state. Because a resource that supports the simplified hot swap model will never transition into Insertion Pending or Extraction Pending states, this function is not applicable to those resources.

Function **saHpiAutoInsertTimeoutGet** allows the caller to request the auto-insert timeout value. This value indicates how long the HPI implementation will wait before the default auto-insertion policy is invoked.

Function **saHpiAutoInsertTimeoutSet** allows the caller to configure a timeout for how long to wait before the default auto-insertion policy is invoked. This function accepts a parameter instructing the implementation to impose a delay before a resource will perform its default hot swap policy for auto-insertion. The parameter may be set to SAHPI_TIMEOUT_IMMEDIATE to direct resources to proceed immediately to auto-insertion, or to SAHPI_TIMEOUT_BLOCK to prevent auto-insertion from ever occurring. If the parameter is set to another value, then it defines the number of nanoseconds between the time a hot swap event with HotSwapState =

SAHPI_HS_STATE_INSERTION_PENDING is generated, and the time that the auto-insertion policy will be invoked for that resource. If, during this time period, a saHpiHotSwapControlRequest() function is processed, the timer will be stopped, and the auto-insertion policy will not be invoked. Once the auto-insertion process begins, the user software will not be allowed to take control of the insertion process; hence, the timeout should be set appropriately to allow for this condition. Note that the timeout period begins when the hot swap event with HotSwapState = SAHPI_HS_STATE_INSERTION_PENDING is initially generated; not when it is received by a caller with a saHpiEventGet() function call, or even when it is placed in a session event queue.

Functions **saHpiAutoExtractTimeOutGet/Set** do the very same as the previous two do, but in the reverse direction. Due to the fact these functions are called when a resource is present in the RPT, there is a parameter, called ResourceID addressing the corresponding resource.

Function **saHpiHotSwapStateGet** allows the caller to retrieve the current hot swap state of a resource. The returned state will be one of the five, well known hot swap state:

- SAHPI_HS_STATE_INSERTION_PENDING
- SAHPI_HS_STATE_ACTIVE_HEALTHY
- SAHPI_HS_STATE_ACTIVE_UNHEALTHY
- SAHPI_HS_STATE_EXTRACTION_PENDING
- SAHPI_HS_STATE_INACTIVE

The state SAHPI_HS_STATE_NOT_PRESENT will never be returned, because a resource that is not present cannot be addressed by this function in the first place.

Function **saHpiHotSwapActionrequest** allows the caller to invoke an insertion or extraction process via software. A resource supporting hot swap typically requires a physical action on the associated FRU to invoke an insertion or extraction process. An insertion process is invoked by physically inserting the FRU into a chassis. Physically opening an ejector latch or pressing a button invokes the extraction process.

Function **saHpiResourcePowerStateGet/Set** allows the caller to get/set the current power state of the FRU associated with the specified resource.

A typical resource supporting hot swap will have to ability to control local power on the FRU associated with the resource. During insertion, the FRU can be instructed to power on. During extraction the FRU can be requested to power off. Not all resources supporting managed hot swap will necessarily support this function. In particular, resources that use the simplified hot swap model may not have the ability to control a FRU hot swap indicator (it is likely that none exists). An appropriate error code will be returned if the resource does not support control of a hot swap indicator on the FRU.

Function **saHpiHotSwapIndicatorStateSet/Get** allows the caller to set/get the state of the indicator (e.g. an LED) may be present on the resource. Valid states include SAHPI_HS_INDICATOR_OFF or SAHPI_HS_INDICATOR_ON. This function will set/get the indicator regardless of what hot swap state the resource is in, though it is recommended that this function be used only in conjunction with moving the resource to the appropriate hot swap state. Not all resources supporting managed hot swap will necessarily support this function. In particular, resources that use the simplified hot swap model may not have the ability to control a FRU hot swap indicator (it is likely that none exists). An appropriate error code will be returned if the resource does not support control of a hot swap indicator on the FRU.

Configuration

The can be used on any resources that have the "Configuration" capability (SAHPI_CAPABILITY_CONFIGURATION), set in their RPT entries.

Function **saHpiParamControl** allows the user to save and restore parameters associated with a specific resource. The restoring of the parameters can be done by either reinitiating them with factory default settings (sensor thresholds and configurations, and resource specific configuration parameters) or loading configuration parameters from non-volatile storage.

Reset Functions

The user can set/get the current reset state of a certain entity with the following functions. The reset actions for the get function are SAHPI_RESET_ASSERT (put the entity into reset state and hold reset asserted, e.g., for hot swap insertion/extraction purposes) and

SAHPI_RESET_DEASSERT. The additional actions for the set functions are SAHPI_COLD_RESET (perform a Cold Reset on the entity (pulse), leaving reset de-asserted) and SAHPI_WARM_RESET (perform a Warm Reset on the entity (pulse), leaving reset de-asserted).

Entities may be reset for a variety of reasons. A misbehaving entity may be reset to bring it to a known state. In these cases, either a warm reset or a cold reset may be performed. A warm reset preserves entity state, whereas a cold reset does not. Both of these reset types are pulsed asserted and then de-asserted by the HPI implementation. This allows the HPI implementation to hold the reset asserted for the appropriate length of time, as needed by each entity. saHpiResourceResetStateSet() can also be used for insertion and extraction scenarios. A typical resource supporting hot swap will have to ability to control local reset within the FRU. During insertion, a resource can be instructed to assert reset, while the FRU powers on. During extraction a resource can be requested to assert reset before the FRU is powered off. This function allows the caller to set the reset state of the specified FRU.

Function **saHpiResourceResetStateGet** gets the reset state of an entity, allowing the user to determine if the entity is being held with its reset asserted. If a resource manages multiple entities, this function will address the entity which is identified in the RPT entry for the resource.

Function **saHpiResourceResetStateSet** directs to perform the specified reset type on the entity that it manages. If a resource manages multiple entities, this function addresses the entity that is identified in the RPT entry for the resource.

A carrier-grade hardware – Sun Netra(TM) ct 800 servers

The Netra ct server is a CompactPCI-based, rack mountable server. The server offers extremely attractive serviceability and modularity, particularly for I/O-intensive applications, as are found in the telco and internet service provider environments. The Netra ct server chassis is highly configurable. Within its chassis, you can have a minimal configuration of a single computer with two hot-swappable I/O slots. You can also have a maximum configuration of up to four independent computers with two I/O slots in each.

Some components in the Netra ct 800 server server are hot-swappable. A hot-swappable component is a component that you can install or remove and replace while the system continue to operate, without interrupting the operating system. You are only required to enter software commands before and after an installation or removal/replacement procedure. The major components, which are the Field Replaceable Units are:

- CPU card
- Power Supply Units (hot swappable)
- System Status Panel (hot swappable)
- System Controller Board
- Main Air Filters (hot swappable)
- Power Supply Unit Air Filters (hot swappable)
- Fan Trays and Fans (hot swappable)
- Hard Disk Drives (hot swappable)
- Removable Media Module (hot swappable)
- Alarm Card
- Midplane and Server Mechanical

References

Service Availability Forum, Forum Platform Interface, 2002.
SA Forum, Hardware Platform Interface Specification, 2002.
Sun Microsystems, Inc., Netra(TM) ct 400/800 server White Paper, 2001.