

SA Forum specs come of age: New versions based on feedback from implementers



The SA Forum has released new versions of the Hardware Platform Interface (HPI) and the Application Interface Specification (AIS). The new specifications are part of the SA Forum plan for adding new services. The ultimate goal for the SA Forum roadmap is a comprehensive set of specifications that are functionally competitive with current proprietary products.

Charlene and Louise highlight some of the modifications that were made to the original versions of the HPI and AIS specifications to obtain the new versions, HPI B.01.01 and AIS B.01.01. Currently, the SA Forum is working on a Systems Management Specification and an AdvancedTCA to HPI Mapping, which are discussed briefly in this article.

Hardware Platform Interface

The Hardware Platform Interface (HPI) is an interface between the application software and middleware and the underlying hardware, allowing portability of software building blocks across different hardware platforms. With the HPI, programmers of applications and middleware can write software that is independent of the particular hardware platform.

The HPI specification represents the platform-specific characteristics of the physical hardware in an abstract model. In addition, the specification provides standard function calls for monitoring and controlling the physical hardware. A vendor's implementation of the HPI represents the physical hardware in terms of this abstract model and translates the function calls defined by the specification into appropriate actions of the physical hardware.

The HPI middleware monitors and controls the physical hardware and provides services to the applications and other middleware, independent of the hardware platform. It discovers the capabilities of the hardware platform. Then the middleware maps those capabilities into a system model, which the middleware maintains and presents to the applications and other middleware.

The modifications to the original version of the HPI specification, HPI A.01.01, simplify the use of the HPI, add several new features, and clarify items that adopters found ambiguous. More specifically, the modifications to the original version that resulted in the new version, HPI B.01.01, include the following items:

- Improvement of the usability of sensor representations by eliminating RAW formats, consolidation of sensor structures, and revision of sensor enabling, based on usage models from HPI users
- Update of the Inventory Data Repository concept, so that it supports AdvancedTCA systems and is easier to use
- Clarification of the notion of HPI domains, making it easier for HPI implementers and users to understand the intended use of HPI domains
- Enumeration of error behaviors and return codes for each API
- Elimination of the *saHpiInitialize()* and *saHpiFinalize()* functions; implementations are expected to handle the initialize and finalize operations automatically, as needed, when a session is opened and all sessions are closed
- Removal of the Entity Schemas concept, which did not serve any useful purpose
- Addition of support for reporting event queue overflows
- Formalization of the relationships between domains in a new Domain Reference Table, identifying domains as peers or having a parent/child relationship
- Addition of support for both manual and automatic mode operations on controls
- Addition of support for reporting failed resources
- Replacement of the Significant Asserted State in event subscriptions with a new Domain Alarm Table to report current alarm conditions in the domain
- Addition of a new management instrument type, the Annunciator, to provide an abstract interface to annunciation hardware

The new version, HPI B.01.01, also incorporates clarifications and resolutions of other small items and issues.

Application Interface Specification

The Application Interface Specification (AIS) defines an interface for high availability applications that is independent of different vendors' implementations of the high availability middleware. The AIS allows application programmers to write application software that is portable across different vendors' implementations of the high availability middleware.

The AIS represents the high availability characteristics of the system in an abstract model and defines APIs for functions that support that model. A vendor's implementation of the AIS represents the high availability middleware in terms of this abstract model and translates the APIs defined by the specification into appropriate actions of the high availability middleware.

The AIS middleware monitors and controls the applications and the middleware and detects and responds to faults. The AIS defines extensive APIs, which an application programmer can use in conjunction with a vendor's implementation of the AIS. In particular, it defines APIs for the Availability Management Framework, Cluster Membership Service, Checkpoint Service, Event Service, Message Service, and Lock Service.

The original version of the AIS specification, AIS A.01.01, has been restructured into seven volumes in the new version, AIS B.01.01. Volume 1, the Overview document, describes the objectives of the AIS specification and the system model, including the physical and logical entities that make up the system, and also the programming model. It defines abbreviations, concepts, and terms. In addition the specification presents an overview of the different parts of the specification. Volumes 2-7 contain the APIs for the:

- Availability Management Framework
- Cluster Membership Service
- Checkpoint Service
- Event Service
- Message Service
- Lock Service

The revised specification, AIS B.01.01, provides significant enhancements to the Availability Management Framework, including introduction of a Component Lifecycle Interface. It aligns the APIs of the various services, so that they have common signatures and a uniform programming style. Specific modifications to the original specification are categorized below as newly added topics, clarifications, cleanup of APIs, and removed topics.

Newly added topics

- Introduction of the Availability Management Framework's Component Life Cycle Interface, consisting of the commands INSTANTIATE, TERMINATE, CLEANUP, AM_START, and AM_STOP
- Description of the Availability Management Framework's recovery escalation policies
- Description of dependencies among service instances and component service instances, as well as dependencies among components in a service group
- Introduction of the quiescing state as an HA state to enable lock/unlock operations on service instances
- Support for local proxied components
- Introduction of pre-instantiable and non-pre-instantiable components and service units, including their presence states
- Introduction of the notion of registered process
- Introduction of new interfaces for passive process monitoring
- Introduction of new interfaces to start and stop healthchecks, and support for component-invoked healthchecks
- Adoption of LDAP Naming Conventions for AIS objects (*SaNameT*)
- Introduction of new interfaces to check optional features of the Lock Service

Clarifications

- Extended description of the Availability Management Framework redundancy models
- Revision of the state model for the Availability Management Framework
- Clarification of which Availability Management Framework APIs are limited to registered processes
- Clarification of the versioning scheme to be used when initializing AIS libraries
- Clarification of the event semantics (such as default attribute values, associating an event with an event channel, republishing an event)
- Clarification of the notion of Lost Event
- Clarification of the notions of Collocated Checkpoints and Active Replicas
- Clarification of the semantics of the Unlock operation
- Provision of partial support for the X.731 state model

Cleanup of APIs

- Association of each API call with a library handle obtained through *sa<Area>Initialize()*, either directly or indirectly
- Homogenization of *sa<Area>Initialize()* and *sa<Area>Finalize()*

- Replacement of opaque types with 64 bit integers
- Making APIs uniform (such as handle usage, error code usage, parameter types)
- Support for use of the NULL pointer as a buffer address, to indicate that the library is responsible for allocating the buffer and the process is responsible for freeing the buffer
- Simplification of component error reporting, by replacing *saAmfErrorReport()* with *saAmfComponentErrorReport()* and by dropping extra fields that support X.733 Alarm Management
- Modification of the track APIs to allow for a synchronous return of the current state of the group being tracked
- Renaming of the *saAmfExternalComponentRestartCallback()* function to *saAmfProxiedComponentInstantiateCallback()* and introduction of a new function, *saAmfProxiedComponentCleanupCallback()*

Removed topics

- Removal of the administrative state for components
- Removal of the usage state
- Removal of the pending operation callback functions of the Availability Management Framework, such as *saAmfPendingOperationConfirmCallback()*, *saAmfPendingOperationExpiredCallback()*, and *saAmfPendingOperationGet()*
- Removal of the *saAmfExternalComponentControlCallback()* function
- Removal of the *saAmfComponentCapabilityModelGet()* function
- As a result of not exposing the readiness state to components, removal of the *saAmfReadinessStateGet()* and *saAmfReadinessStateSetCallback()* functions
- Removal of the *saMsgMessageReceivedGet()* function
- Removal of the requirement that messages must be preserved during a failover and, consequently, removal of the SA_MSG_PRESERVE_MESSAGES flag
- Removal of the SA_MSG_QUEUE_SELECTION_OBJECT_SET flag from the *SaMsgQueueOpenFlagsT* type
- Removal of the *ackFlags* parameter from the *saMsgMessageSend()* and *saMsgMessageReply()* functions
- Removal of the cluster name from the *SaClmClusterNodeT* structure and inclusion of the initial view number in that structure
- Removal of Section 4.5, *Component Support of Availability and HA State*, which was redundant with Section 4.6 of AIS A.01.01

HPI and AIS implementations

Many companies in the computer and communications industries are using the HPI and AIS of the Service Availability Forum to produce high availability infrastructure products, systems and services for telecommunications, data communications and networked applications. Companies that implement the specifications can register their products as *SA Forum Registered* on the SA Forum Web site at www.saforum.org. SA Forum registered products will be demonstrated in the SA Forum booth at CTIA Wireless 2005.

Looking ahead

The SA Forum Systems Management Working Group is currently developing a Systems Management Specification (SMS). The SMS provides a Simple Network Management Protocol (SNMP) and Web based interface that supports distributed access, monitoring and control of the HPI and AIS management functional-



ity using Management Information Bases (MIBs), and Common Information Model (CIM) Schemas. The SA Forum will release the SMS in 2005.

The HPI supports management of highly available hardware platforms and, thus, provides a consistent and comprehensible interface for PICMG AdvancedTCA Shelf Management. The HPI along with the System Management Specification for HPI-based SNMP MIBs and CIM Schemas can be used to construct a platform manageability stack for AdvancedTCA systems that provides a flexible interface to the Shelf Managers.

To realize this objective, the SA Forum HPI Working Group is developing an AdvancedTCA to HPI Mapping. This specification will establish an industry standard means of modeling AdvancedTCA features using HPI constructs. Each AdvancedTCA PICMG command, Field Replaceable Unit (FRU) information item, etc. can be modeled by the generic HPI constructs in several different ways. The AdvancedTCA to HPI Mapping will establish a standard set of HPI resources, entities, controls, sensors, and inventory data fields, so that AdvancedTCA systems and their Shelf Management functionality can be modeled consistently. By establishing such a standard, even greater portability of high availability middleware across a variety of AdvancedTCA platforms will be achieved.

Figure 1 illustrates the effect of coupling the SA Forum and AdvancedTCA standards. The AdvancedTCA standards define a robust set of Shelf Management functionality. This Shelf Management functionality can be exposed by an HPI implementation, compliant with the standard AdvancedTCA to HPI Mapping. A vendor of such an implementation can exploit the SA Forum specifications to create a variety of solutions. In particular, using the SA Forum HPI APIs directly allows the high availability middleware to manage the service availability of the system proactively. For example, using the SA Forum HPI CIM Schema definition, along with the HPI Providers and CIM Object Managers, enables a standard CIM-based remote access interface. Using the SA Forum HPI MIB definition, along with the appropriate SNMP agents and sub-agents, provides a standard SNMP remote access interface.

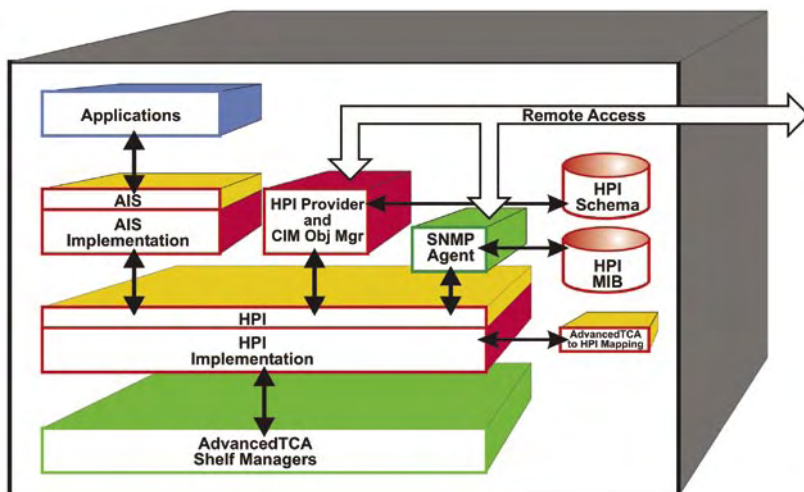


Figure 1

Summary

The SA Forum Hardware Platform Interface and Application Interface Specification allow application developers to write application software that is portable across different hardware platforms, operating systems, and high availability middleware. As open standards, they result in better service for the end users, fewer integration problems for system providers, and shorter development times and reduced costs for hardware and software vendors.

The new versions of the Hardware Platform Interface, HPI B.01.01, and the Application Interface Specification, AIS B.01.01, will serve as the baseline for future certification testing and for other Service Availability Forum specifications, such as the Systems Management Specification. Moreover, the AdvancedTCA to HPI Mapping will establish an industry-standard means of modeling AdvancedTCA features using HPI constructs. The SA Forum specifications will enable the use of COTS building blocks in the creation of high availability network infrastructure products, systems, and services.

The SA Forum HPI B.01.01 specification and the AIS B.01.01 specification are freely available for all to download from the SA Forum website at www.saforum.org/specification/download.



Louise Moser is a professor in the Department of Electrical and Computer Engineering at the University of California, Santa Barbara. She is an active participant in various standards organizations, including the Service Availability Forum, where she was an editor of the Application

Interface Specification. She received a Ph.D. in Mathematics from the University of Wisconsin.

Charlene Todd is a software architect for Intel, where she focuses on telecommunications systems and standards. Participation in the Service Availability Forum is one of Charlene's primary tasks. Charlene actively participates in many of the Service Availability Forum's technical



work groups, and has served as the technical editor for the Hardware Platform Interface Specification. Charlene holds an MS in Computer Science from Arizona State University, and a BS in Electrical Engineering from Colorado State University.

For further information, contact the authors at:

Service Availability Forum
E-mail: marketing@saforum.org
Website: www.saforum.org