



# GoAhead SelfReliant

## SelfReliant Advanced Suite (SR-AS)

SelfReliant Advanced Suite (SR-AS) is a full suite of standards-based, platform-independent high availability (HA) middleware that delivers a pre-integrated, carrier-grade application ready platform. SR-AS helps development teams do the following:

- Shorten the time it takes to build highly available systems
- Reduce initial development costs by leveraging commercial, off-the-shelf technology
- React more quickly to technology and platform changes
- Focus development efforts on functionality that delivers competitive differentiation
- Reduce project risk by building on a platform that provides pre-tested, pre-integrated hardware and software

With its modular design, developers can select the level of availability management functionality that is most important for their product. SR-AS includes a broad array of services including messaging services, platform services, and systems management services. With an extensive library of APIs, SelfReliant can be completely customized to meet the unique market requirements of a project. In addition, user-written APIs can be developed to extend SelfReliant's capabilities even further.

SelfReliant is completely standards-based and platform independent and therefore protects organizations from being locked into particular hardware or software. SelfReliant supports multiple operating systems as well as multiple CPU and system architectures. SelfReliant also supports the Service Availability Forum (SA Forum) Hardware Platform Interface (HPI) specification, enabling availability management down to the hardware component levels. GoAhead has also released an early access release version that supports the SA Forum Application Interface Specification (AIS) and has announced plans to release a Beta version later this year.

SelfReliant is pre-integrated and pre-tested with several third party hardware platform providers as well as software components such as in-memory databases, RDBMS, protocol stacks and storage management. GoAhead helps development teams avoid the resource drain associated with getting various commercial-off-the-shelf building blocks to work together. Applications that are built on SelfReliant are largely insulated from changes that occur in underlying hardware and software.

### Key SR-AS Benefits

- **Complete, integrated availability management, distributed messaging, and embedded systems management delivering availability of 99.999% or better**
- **Pre-integration with third party hardware and software insulates development teams from changes in underlying hardware and software and allows resources to be focused on competitive differentiators**
- **Support for the Service Availability Forum HPI specification enables complete hardware platform management and clear roadmap for AIS support**
- **Industry-proven software lowers overall project risk**
- **Platform-independence and support for multiple operating systems allows development teams to quickly respond to new technologies and platforms**
- **Millisecond stateful failover, fast messaging, and low CPU usage ensure optimal systems performance**
- **Extensive library of APIs enables development teams to customize functionality to meet unique market requirements**

### Who Uses SelfReliant Advanced Suite

SR-AS can be used in a wide variety of equipment and applications, including the following:

- **Communications** – Implementations span wireless base stations, base station controllers, media gateways and controllers, push-to-talk solutions, SIP servers, softswitches, and network storage devices
- **Military and Aerospace** – Projects include command, control and communications systems that require five and six-nines of availability
- **Industrial automation** – Industrial applications include real-time, distributed process control applications where downtime is unacceptable

SelfReliant Advanced Suite is ideal for...

- Developers who need to deliver sophisticated or highly customized high availability services including:
  - 99.999% high availability or better
  - Sub-second, stateful failover
  - Fast, distributed messaging
  - Low system resource consumption (memory, CPU)
  - A pre-integrated, application ready HA platform
- Project teams who need a phased approach to HA functionality - starting with fundamental HA capabilities and moving to more sophisticated HA capabilities as time goes on
- Architects planning to ensure their HA solutions are based on standards such as SA Forum's HPI and AIS specifications

## SelfReliant Advanced Suite Overview

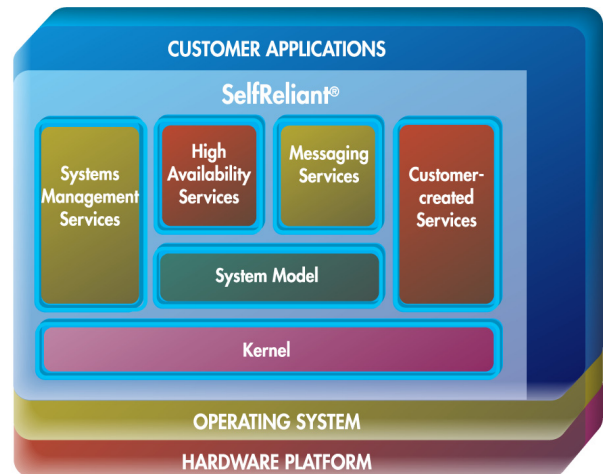
SR-AS manages redundant components including applications, hardware, operating system, other middleware, and itself. SR-AS is a high performance product featuring extremely fast failover and scalability of up to 64 nodes.

The product presents several different options for availability management. For non-intrusive stateless application failover, SR-AS offers quick and easy HA with no programming. With this method, applications can participate in the HA framework without code modification. In addition, a simple set of APIs allows developers to checkpoint applications to preserve state for paired applications and/or nodes. For multi-node failure scenarios that preserve state, SR-AS provides the full SelfReliant API libraries to control devices down to the hardware component and operating system levels.

To facilitate understanding of SelfReliant's breadth and depth of features, the software offers the following service categories:

- SelfReliant Kernel
- Messaging
- System Model
- High Availability
- Systems Management
- Integrated Platforms
- Customer-created

These services include subsidiary services that together deliver a comprehensive high availability and systems management solution.



## SelfReliant Advanced Suite Services

### SelfReliant Kernel Services

The SelfReliant Kernel is the small, reliable, cross-platform foundation for all SelfReliant services. It is multi-threaded, supporting thread creation, priorities and synchronization. The Kernel loads the other SelfReliant services as well as customer-specific APIs. This portability layer limits users' dependencies on the underlying operating system and hardware.

### Messaging Services

Messaging services include basic functionality common to both single node and distributed systems. Because more advanced services frequently depend upon these underlying features, messaging and database services are often the first capabilities project teams implement.

The **Distributed Messaging Service (DMS)** provides the underlying inter- and intra-node communications framework for distributed and single node systems. High performance DMS supports heartbeating, congestion detection, message prioritization, checkpointing, transparent network failover, and resource discovery. Communications methods include a choice of point-to-point, publish and subscribe, and server pooling. Routing options are one-to-one, one-to-many, many-to-one and many-to-many. DMS provides the following functionality:

- Conducts event, fault, and error notification
- Manages client/server communication
- Automates communication monitoring
- Provides the framework for remote procedure calls
- Provides high performance delivery (40,000+ messages/sec) with low resource requirements



## Availability System Model

The System Model is a unique feature in SelfReliant. Each system resource to be managed is represented as a managed object in the System Model. **The System Model is a key architectural component that is often neglected in proprietary projects.** The SelfReliant System Model can scale from a simple architecture to a model that captures the complexities of even the most sophisticated systems. It also reflects resource dependencies, including parent-child relationships that form a given service. The System Model ensures that project teams can easily maintain and update the system configuration in the future without modifying the hardware or application.

- Managed Objects
  - Representation of resources based on ITU X.731 states
  - Attributes: health, operation, administrative status, role
  - Methods: access/control, monitoring, configuration
  - System Model scales to 10,000+ objects
- States of Managed Objects (State Model)
  - Records the state of each object, such as active/standby, locked/unlocked and healthy/failed
  - Enables Availability Management Service to make intelligent recovery decisions based on each resource's attributes and methods
  - AMS replicates the System Model to the standby availability manager in case the active availability manager node fails
- Service Groups: Logical representation of redundant resources and service units
- Resiliency: System Model is replicated to a hot standby node
- Redundancy Policies: 2N, N+1, N+M, Active/Active, Custom
- System Model Export/Import Tool: Extract managed object data from an existing System Model, update the model as desired, and import the updated version to provision all or parts of a new System Model. This tool greatly simplifies the System Model provisioning process, encourages collaborative development, and assists with debugging.

## High Availability Services

High Availability Services interact with the Messaging Services, System Model, and Systems Management Services to ensure 99.999% or better availability of equipment and applications. SelfReliant operates in heterogeneous hardware environments, including pedestal, rack and blade servers, as well as cPCI and ATCA chassis and proprietary hardware. It also supports the most popular operating systems and processor architectures.

- **Availability Management Service (AMS)** – Provides SelfReliant's core availability management framework. Using the System Model and State Model, AMS centrally manages abstractions of the hardware and software to eliminate downtime. Managed resources can include applications,

operating system, chassis, I/O cards, redundant CPUs, networks, peripherals, clusters and other middleware. Virtual IP addresses can be dynamically assigned to these active resources. AMS also supports administrative operations such as initiating a node-level switchover, allowing resources to gracefully transition work in progress to standby resources. The availability manager is redundant with a hot standby on a separate node. Recovery actions and notifications are based on business rules, event priority structures, and user-defined policies. AMS is responsible for role assignments (N+M) and active/standby/spare identification. It also conducts health monitoring and uses very little overhead.

- **Cluster Management Service (CMS)** – Manages the physical nodes or instances of SelfReliant, while AMS manages virtual elements. Using DMS, it monitors the health and status of each node, reporting any failures to AMS and interested users. CMS supports redundant network interfaces as well as multiple network topologies. A cluster may be assigned a virtual IP address.
- **Replicated Database Service (RDBS)** – Provides replication of the database in a redundant system, supporting AMS' ability to conduct fast, seamless switchover. RDBS includes a configurable replication throttle to control the data rate of the replication process.
- **Transparent Application Management Service (TAMS)** – Delivers HA to applications that do not require modification or are impossible to modify. Examples include legacy applications, commercial applications with no access to source code and applications that preserve state through other methods. TAMS supports virtual IP address switchover. TAMS can also be used with modified applications to provide application startup, monitoring, shutdown, etc.
- **Simplified Availability Management Service (SAMS)** – Provides fast failover with preservation of application state. Simple HA APIs enable start, stop, restart and millisecond switchover for pairs of applications and/or nodes in active/standby and active/active configurations. SAMS includes support for individual, group, service group and dependency failover scenarios.
- **Platform Resource Management Service (PRMS)** – SelfReliant supports the SAs Forum's HPI specifications. This industry-driven standard provides APIs for managing platform availability to reduce development time and effort. On platforms that supply an HPI service, PRMS provides automated discovery of HPI-enumerated hardware resources and populates those resources in the System Model. Using the HPI hardware management capabilities, it defines and provides the default behavior for hardware resources.

## Systems Management Services

Systems Management Services are the fundamental building blocks for developers creating their own system-specific management solution. These services can be used in either



stand alone or redundant systems. Along with messaging and database services, project teams often design and implement this set of services early in a project.

- **SelfReliant Management Console** – Provides a browser-based, single point of management for the entire cluster. Includes straightforward configuration of cluster topologies and recovery policies.
- **SNMP Agent** – Provides two agent integration methods: a master/subagent implementation, and a standalone SNMP agent. A MIB compiler and other tools are also provided.
- **SNMP Object MIB Module** - Makes the AMS System Model accessible via SNMP allowing remote users to view the state of resources, perform administrative operations, configure trap notifications and publish AMS object subclasses using standard SNMP MIB browser.
- **HPI MIB Module** - Extends the existing systems management provided by the Object MIB Module to expose Hardware Platform Interface (HPI) data for hardware resources through an SNMP interface, enabling an external system manager to access hardware data and functionality without pre-knowledge of or integration with the hardware.
- **Embedded Web Server** – Leverages the GoAhead open source WebServer, providing HTML remote access, control and status/alert display.
- **Management Data Store Service** – Offers a high performance in-memory data store that standardizes data via XML in a hardware independent manner. Intended for system management functions (not application database use), the data store is optimized for read and access times (<3 µsec), while preserving a small memory footprint. It supports background persistence to disk and peer-to-peer replication.

### Integrated Platform Services

- **Hot Swap Management Service** – Provides built-in, extensible hot swap management policies for your system, ensuring a graceful transition when resources are inserted or extracted.
- **Alarm Management Service** – Provides built-in, extensible alarm management policies for both hardware and software resources that can be configured to match the overall system's alarm management policies.
- **Storage Management Services (SMS)** – Built to handle storage subsystems as managed resources to be easily included into a system's availability management policies.

### Customer-Created Services

SR-AS opens access to all SelfReliant APIs. Developers can extend SR-AS built-in features with customized functions. Users can manage each resource at a more granular level, further

improving reliability and availability. Other advanced features include increasing the number of states and creating custom service groups unique to your application. Because SelfReliant was built hand-in-hand with a large network equipment provider for all types of network equipment, its flexibility, depth and breadth handle the toughest availability challenges.

### SR-AS Specifications *(measured on Linux CGE 3.1)*

#### System Requirements

- Run-time memory: 5 MB
- Run-time disk space: approximately 17 MB
- Development disk space: 200 MB
- Supported operating systems/CPU platforms include:
  - RedHat Enterprise Linux 3.0 (IA)
  - MontaVista Linux CGE 3.1 (IA)
  - MontaVista Linux Professional Edition 3.1/4.0 (PPC)
  - Solaris 8.0/9.0 (SPARC)
  - Windows Server 2003 (IA)
  - Debian 3.0 Linux (IA64)
  - VxWorks 5.5.1(PPC)

#### Performance and Scalability

- High Availability
  - Modified application failover: as low as 10 ms
  - Unmodified application failover: as low as 300 ms
- System Model
  - Scales to: >10,000 managed objects
  - Object state transitions per second: >2500
- Messaging
  - Point-to-point connection rate (64 byte message): >40,000 messages per second
  - Publish-subscribe connection rate (64 byte message): >18,000 messages per second
  - Roundtrip latency pub/sub with 100 connections: 0.7 ms
  - Number of connections per node: >1,000
- Database
  - Read time of 64 byte record: 3 µs
  - Write time of 64 byte record: 19 µs
- Scalability: Number of nodes: up to 64
- Embedded WebServer
  - 60KB footprint
  - HTTP 1.0 support
  - Persistent connections (consistent with HTTP 1.1)
  - 65 connections per second
- Steady-state CPU usage: <0.1%

#### Languages supported

C and C++

[www.goahead.com](http://www.goahead.com)

10900 NE 8th Street, Suite 1200 Bellevue, Washington 98004-1455  
PHONE +1.425.453.1900 FAX 1+.425.636.1117 CONTACT [info@goahead.com](mailto:info@goahead.com)