Functional Specification
Sun Communication Application Server

Author(s): binod.pg@sun.com, sreeram.duvur@sun.com

Version: 1.0

Revision History

Version	Description	Date	Author
1.0	First version	10th Oct 2007	Sreeram, Binod
1.1	Incorporated comments from Kristoffer	25th Oct 2007	Binod

1 Introduction

<List proposed feature(s). Introduce the basic vocabulary. Why is this interesting? List capabilities that may be normally expected, but are not being supported. Are there any limitations and caveats that need to be disclosed?>

Sun Java System Communication Application Server (SJS CAS or SCAS) adds SIP and Telco related capabilities on top of Sun Java System Application Server (SJS AS). SJS AS is developed as project GlassFish (http://glassfish.dev.java.net), and SCAS is developed as project SailFin (http://sailfin.dev.java.net). SailFin is an affiliated member of GlassFish community in java.net.

SCAS is a full-featured Telco Application Server with load balancing, clustering and failover and administration features.

1.1 Terminology

DAS
Domain Administration Server of the Application Server

IMS
The IP Multimedia Subsystem (IMS) is an architectural framework for delivering internet protocol (IP) multimedia to mobile users.
http://en.wikipedia.org/wiki/IP_Multimedia_Subsystem

CSCF
The CSCF provides session control for subscribers accessing services within the IP Multimedia Subsystem
http://en.wikipedia.org/wiki/IP_Multimedia_Subsystem

TIPC
Transparent Inter-process Communication (TIPC) is a communications protocol for inter-process communication (IPC) that was specially designed for intra-cluster communication.
http://en.wikipedia.org/wiki/TIPC

SNMP
SNMP is used by network management systems to monitor network-attached devices for conditions that warrant administrative attention.
http://tools.ietf.org/html/rfc1157

GMS (Project Shoal)
Project Shoal is a Group Management System heavily used in SJSAS but it can also be used in other contexts.

1.2 Features supported by 1.0 release of SCAS

High level features supported by SCAS are enumerated below. For details of each functionality, please refer to individual functional specifications.

1.2.1 Java EE 5

SCAS is developed as layer on top of SJSAS. It will support all the features available in SJSAS including Java EE 5 compatibility.

1.2.2 JSR 289 and JSR 116

SCAS will contain a sip stack that is compliant with RFCs 3261, 3262, 3265, 3311, 3515, 3903 etc. It will also expose the SIP Servlets Java APIs defined by JSR 116 and JSR 289. The JSR 289 container will be integrated to SCAS as a listener to the tomcat container.

1.2.3 Converged Sip and Http Sessions

SCAS will support converged HTTP and SIP session functionality as defined by JSR 289 and JSR 116.

1.2.4 Converged Http and Sip Load Balancer

SCAS will provide a built-in load balancer that support converged Sip and Http applications. The load balancer will not be a separate java process nor a plug-in to a Web Server. It will be implemented as part of SCAS instance itself. This is a novel approach and a significant new direction compared to GlassFish. Over the long term this architecture may become useful in GlassFish as well, but that is not an immediate goal for this project.

1.2.5 Session Replication for Sip and Http Sessions

Session replication capability in SCAS will support both SIP and HTTP sessions. It will also support replication of Sip Timers, SIP Dialog state and SipApplicationSession objects as defined by JSR 289.

The current session replication framework in SJSAS will be extended to support usecases demanded by the Sip application state. HADB will not be supported in this release.

1.2.6 Grizzly Integration

Grizzly NIO framework will be used in the SIP container for server side socket listeners. Grizzly is being enhanced to support symmetric client and server operation, which is required in Sip Application Servers. Grizzly is enhanced to add additional functionality like client side TCP, client side UDP, and server side UDP with high performance and TLS support.

1.2.7 Security Enhancements

Digest Authentication will be supported for both HTTP and SIP protocols.
P-Asserted-Identity will be supported for SIP protocol. The Sip container will be enhanced to support SIPS over TLS.

1.2.8 Deployment of Sip and Converged applications

SCAS will support deployment of pure Sip applications and converged applications using the extension-module mechanism in GlassFish. It will also support sun-sip.xml as the sun specific deployment descriptor for Sip applications.

1.2.9 Administration Support for SCAS

Asadmin CLI commands and GUI will be developed to support the functionality introduced in SCAS. This will cover both Sip container and Converged load balancer. The current monitoring capabilities of the SJSAS will be extended to support SCAS specific monitoring and AMX. The administration support will be an extension on top of the current glassfish administration framework.

1.2.10 Netbeans Tooling

Support for Sip Servlet applications based on JSR 289 will be provided on top of Netbeans 6.0. A simple Sip client will be added to enable development time testing. This plug-in will be provided via the netbeans update center.

1.2.11 Hot deployable application router

SCAS will support hot deployment of Application router archives (jar files) using the extension module mechanism. An application router that follow a simple alphabetical rule for routing the requests will be provided by default. JSR 289 specification describes semantics of default application router. That will be provided as well.

1.2.12 Enhancements for supporting a SAF based management framework.

Enhancements to the current administration and management framework will be done so that an external management framework (eg: a SAF based tool) will be able to manage the application server at a steady state. This capability will translate to the ability to start and stop application server instances and the ability to start and stop applications in the server instances by passing DAS (domain administration server) and node agent. The watchdog functionality of the nodeagent will also be in use, since in the steady state, node agent will not be running. Note that this is not the default behaviour of the application server, but are enhancements to allow a management framework to achieve the described functionality. The newly filed JSR 319 is aiming standardization in this space.

1.2.13 Packaging and File Layout

SCAS will be delivered as a File based installer and also as an addon package. File layout preserves GlassFish layout with overlays for Sip servlets support.

2 Design Overview

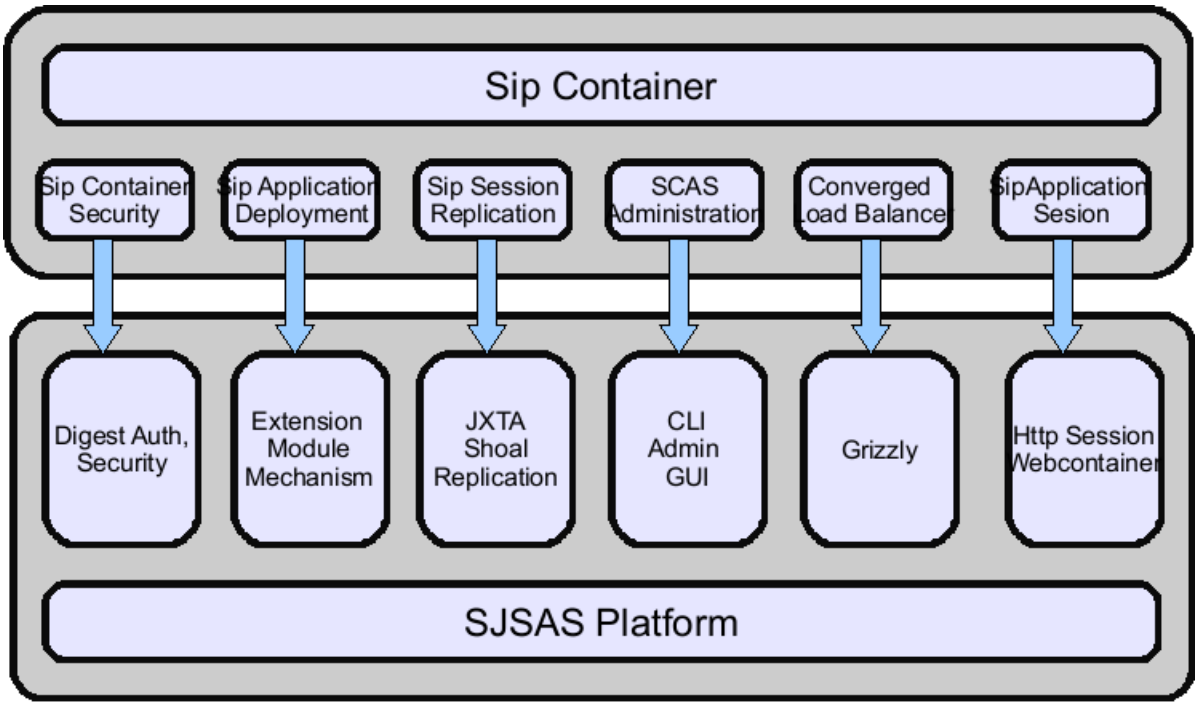
<Discuss the core concepts and design. Provide conceptual diagrams, if they would be helpful. Show how this sub-system/feature co-exists with other sub-systems. You may write 1-4 pages (can be shorter or longer). This section is should be a map to navigate well documented code!>

2.1 Relationship with SJSAS

Since SCAS is meant to add functionality on top of SJSAS, the product will be implemented as a layer on top of SJSAS. Necessary extensions will be implemented in base SJSAS so that the new functionality can be plugged in. For example, there will be an interface defined in the SJSAS administration CLI to look for additional commands defined in another descriptor file also. Different functional specifications describe their interfaces with SJSAS where applicable.

All of the SJSAS functionality will be available in SCAS as well. It also mean that where not specified the architecture assume the SJSAS architecture and thus SCAS will inherit all the advantages and limitations of the SJSAS.

The following picture capture the current list of interfaces.



All such interfaces will be implemented in the SJSAS 9.2 release.

2.1.1 Profiles

SJSAS support developer and cluster profiles. SCAS will continue to support them. Basic Sip Servlet Container will be available by default in developer profiles. User will be able to deploy and run a pure Sip or Converged application in a single instance. Converged Load Balancer and Sip Session replication will be available only in cluster profile.

SCAS will not support enterprise profile of the SJSAS. So, the HADB based session persistence will not be supported in the first release of SCAS. In a future release of SCAS enterprise profile will be considered for support.

2.1.2 Domain.xml

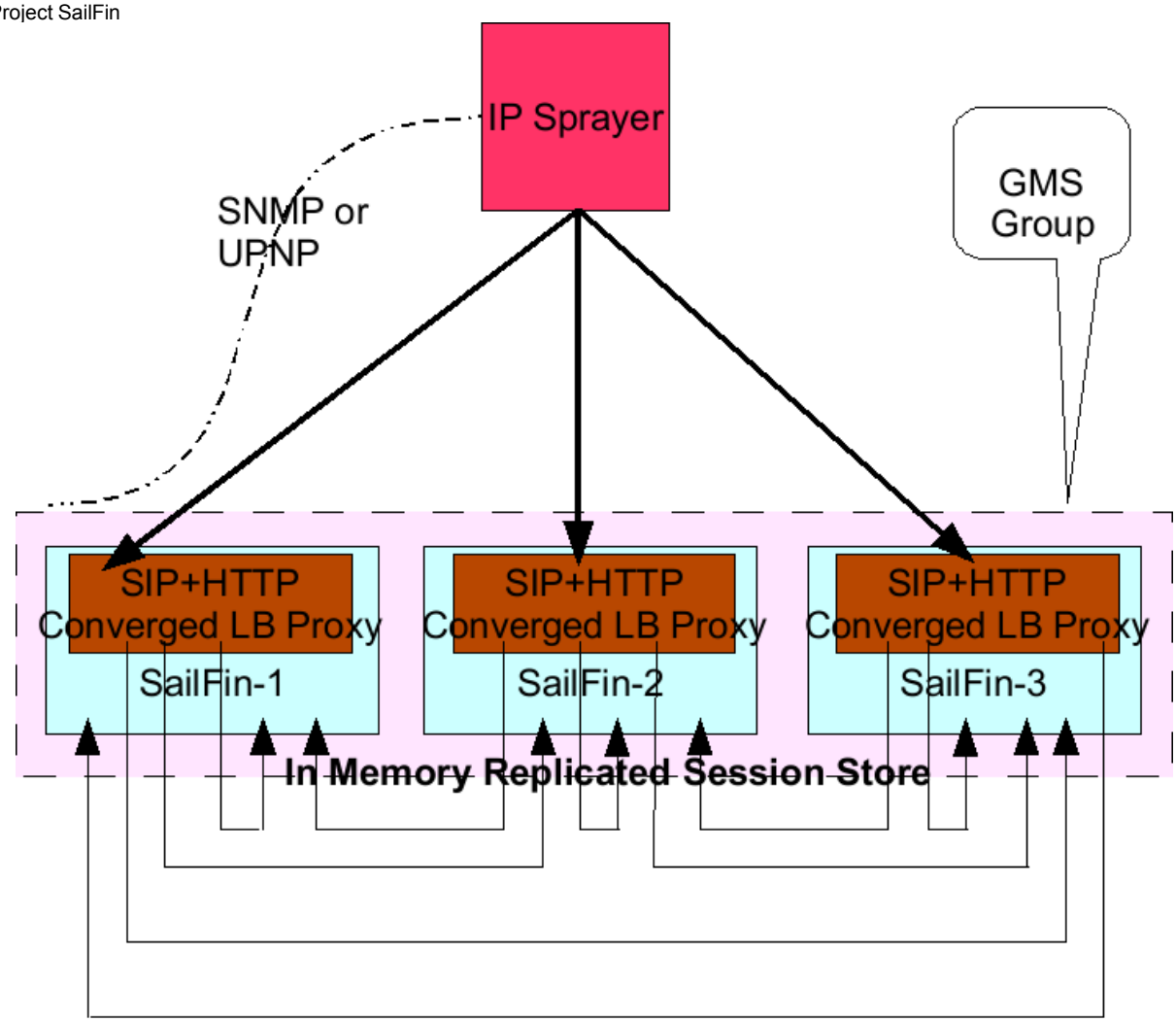
The normal practice of any product that sit on top of SJSAS is to use a different configuration file. However given that next major version of the SJSAS is planning to provide extensibility of domain.xml, SCAS will use domain.xml for its configuration. It means that the new version of domain.xml DTD will be present in SCAS. SJSAS 9.1.1 will be enhanced to dynamically change the DTD to a newer version.

2.2 Product Deployment Topology.

SCAS can be deployed to run Java EE Applications, Sip applications or Converged Sip and Java EE applications. These applications can use any of the Java EE and other functionalities supported by JSR-289 and by developer or cluster profiles of the SJSAS.

It is expected that for most of the deployments of the SCAS, the Converged Load Balancer will be used. However, it would be possible to use the native load balancer plug-in also with SCAS using a manual setup process. Native load balancer can be used only for HTTP load balancing.

It will be possible to use the converged load balancer as part of the cluster instances itself. Each instance of the cluster will act as both load balancing proxy and the backend server instance.



It is very likely that the load balancer tier is fronted by a hardware IP level load balancer (or IP sprayer).

The IP sprayer abstraction enables a single IP view of SCAS cluster to external clients. Examples of the IP Sprayer could be an IP Load Balancer appliance, IP virtualization capabilities such as those found in Sun Cluster or TIPC/Linux. The IP sprayer will be able to view the shape of the load balancer cluster using protocols like SNMP or its own custom policies. An agent can also be written (deployed as a lifecycle module on SCAS) which could use GMS apis to get notified of the cluster view changes and then inform the IP sprayer of the cluster view changes using protocols like SNMP.

Usage of SipSessionUtil in the applications (eg MDBs using non-JMS resource adapters) deployed in a cluster depends on the ability of the protocol adapter of the

incoming messages to do the load balancing using a hashing algorithm similar to consistent hashing.

It is expected that all the instances of the SCAS cluster are in the same subnet. A cluster spanning multiple subnets will not be supported in this release.

2.3 Security

Digest authentication will be implemented in both Http and Sip containers. The current JDBC realm will be enhanced to support Digest authentication. In most of the IMS deployment, if the application server is fronted by a CSCF, the initial authentication will be done by CSCF. In that case, a special p-asserted-identity header will be added to the Sip Message to indicate that it is from a trusted source. SCAS will support p-asserted-identity.

There will be a way to configure a trusted intermediate information to administratively determine an incoming or outgoing sip traffic is from a trusted entity or not. This is in addition to the TLS support in SIP socket listeners.

The Sip Servlet authentication will be integrated to SJSAS security framework so that the security context from the Sip Servlet will be propagated to EJBs.

3 Availability

<How do you handle availability concerns? Does this feature introduce any new failure modes? List testing and failure scenarios that quality team needs to worry about.>

The current session persistence framework that is based on a ring topology will be used for replication of Sip timers, dialogs, dialog fragments, sessions , Converged Http Sessions and Sip Application Sessions. The state of the application at any point is represented by this inter-related complex of objects. Since these objects are replicated to another instance, the memory requirement for session storage will increase by 100%.

The availability story will be based on the converged load balancer, session replication and shoal cluster membership service. Shoal framework will be enhanced to support a NIO based discovery mechanism as against the current UDP based communication.

SCAS will support only asynchronous mode of the replication. It means that any replication activity will be asynchronously replicated to the its replica buddy and no acknowledgment mechanism for the replication will be implemented.

HADB based session persistence will not be supported in this release of SCAS. This is not really because the enterprise profile or HADB is not applicable for SCAS, but because of the schedule commitments and to be able to meet the schedule.

3.1 Rolling Upgrade

Upgrading an application in SCAS will be by following exact same procedure of SJSAS. Only compatible versions of the applications will be supported. That means the new application should use the same same database schemas and session information and should have a compatible business logic.

Following are the steps for upgrading the application.

1. Backup the domain.xml using the command line interfaces.
2. Switch off dynamic reconfiguration for the cluster.
3. Deploy the new version of the application. Since the dynamic reconfiguration is switched off, the new bits wont be synchronized to the cluster instances and they will just remain in the domain administration server.
4. Disable one server instance from the load balancer using the asadmin command so that load balancer quiesce the requests to the instance. During this period, load balancer will make sure that all the new requests are routed to a different instance.
5. After the quiescing period is over, restart the instance. During restart, the instance will get synchronized with DAS and thus the application bits will be upgraded. When the instance is restarted, it's sessions will be migrated back from the other instance. Also, the load balancer will start routing all the requests to the failed over instance again as per the consistent hashing algorithm.
6. Repeat the above steps for all the instances.

It will also be possible to do a minor compatible upgrade of the application server using similar steps.

4 Performance

<How do you want performance team to measure this sub-system? Any micro benchmarks necessary?Any goals? Anticipated scalability limits or goals?>

SCAS will be tested for a minimum of 10 instances in a cluster. Exact performance goals of SCAS is documented at the “Performance Requirement Documentation”

5 Management and Monitoring

<Describe how performance, management status, and diagnostic information is exposed. How does this feature handle dynamic configuration changes?>

SJSAS administration/management framework will be extended to support SCAS. The extensions will be for the following.

- Asadmin CLI framework to honor an additional CLI descriptor.
- GUI to insert new screens SCAS.
- Mbean framework to pickup additional descriptor files.
- Dynamic reconfiguration framework to honour an extended domain.xml file.
- AMX.

There will be enhancements to domain.xml for the following.

- New sip-service element, similar to http-service.
- New sip-container element to configure the container
- New converged-load-balancer related elements.
- Sip-container availability related elements.
- Security Related Elements for Identity Assertion.

There will be new asadmin commands and GUI screens to manage these. More details are available in the Administration Functional Specification.

The current monitoring framework of SJSAS will be extended to expose Sip Servlet monitoring capabilities. The exact list of monitoring attributes are listed in the Administration functional specification. The sip container will also be enhanced to support callflow.

5.1 Formal Interfaces

<How is this feature(s) configured by administrator? Does it introduce new commands or modify existing ones? Show syntax of expected administrative commands and response codes. What is the schema/syntax for new configuration in domain.xml? Show the DTD snippets later in this section. What are their default values? What are the validation rules? Think about the stability level for each of the above. Are you expecting that the proposed design will change?]

The exact exposed interfaces are described in different functional specifications.

5.1.1 Reference Documents

- Administration Functional Specification
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/sailfin_admin.doc
- Load balancer Functional Specification

- http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/ConvergedLB-FSD.pdf
- Converged LB Proxy Functional Specification.
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/lb_proxy_fsd.doc
- Session Replication Functional Specification.
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/FSD-SSR-2.3.odt
- Deployment Functional Specification.
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/sailfin_fsd_deployment.doc
- Container Integration Functional Specification.
  -
- Security Functional Specification.
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/sailfin_fsd_security.odt
- Converged Http Session Functional Specification.
  - http://wiki.glassfish.java.net/attach/FunctionalSpecsOnePagers/ConvergedHttpSession_FunctionalSpec.doc
- Installer Functional Specification
  -

6 Packaging, Files, and Location

<Does this feature add new jar files or extend existing ones? Where are they located?>

The SCAS will be delivered as a file based installer that will install SJSAS and then the additional SCAS functionality on top of it. The file based installer will contain the J2SE 6.0 version that will be used for testing SCAS.

SCAS installer will add SCAS jar files to the lib directory of the SJSAS so that they will be picked up by SJSAS classloader. The asadmin script will be modified to pickup a SCAS client jar file as well.

To allow someone to install SCAS on top of an existing SJSAS installation, there will be a separate bundle available. This bundle will be packaged as per the SJSAS addon packaging guidelines. User will be able to install that using the SJSAS update center.

7 Documentation Requirements

<List the required documentation to support this product feature.>

Following are the set of documentation that will need to be either enhanced or newly written

- Developers Guide.
- Administration Guide.
- HA administration Guide.
- Performance Tuning Guide.
- Deployment Planning Guide.

8 Open Issues