

**452 / SPACE NETWORK PROJECT
(MISSION SERVICES PROGRAM)**

**Space Network (SN)
Web Services Interface (SWSI)
Server Operator's Guide
Release 04.1**

(CCB Review Copy – 02/04/04)

DCN 1

Effective Date: February 2004

Expiration Date: November 2008



National Aeronautics and
Space Administration

————— Goddard Space Flight Center —————
Greenbelt, Maryland

**CHECK THE GSFC CENTRALIZED CONFIGURATION MANAGEMENT SYSTEM AT:
<http://gdms.gsfc.nasa.gov>
PRIOR TO USE TO VERIFY THAT THIS IS THE CORRECT VERSION**

Space Network (SN) Web Services Interface (SWSI) Server Operator's Guide Release 04.1

February 2004

Prepared by:

Mr. Scott Robinson Date
Computer Science Corporation
Consolidated Space Operations Contract

Prepared by:

Mr. E. Joseph Stevens Date
Space Network Project, Code 452

Prepared by:

Mr. Thomas Sardella Date
SWSI Product Manager
Space Network Project, Code 452

Approved by:

Mr. Keiji Tasaki Date
Project Manager
Space Network Project, Code 452

Goddard Space Flight Center
Greenbelt, Maryland

Preface

This document contains Local Operating Procedures (LOP) and overview material necessary for operation of the Space Network (SN) Web Services Interface (SWSI) Server.

This document is currently under the configuration management of the Flight Programs and Projects Directorate's Space Network Project (Code 452) Configuration Control Board (CCB). Configuration Change Requests to this document may be submitted to the Space Network Project's CCB along with supportive material justifying the proposed change. Changes to this document shall be made by Documentation Change Notice (DCN) or by complete revision.

NOTE TO USERS

This document has been approved and published for SWSI Product baseline documentation purposes. It is intended that this document be transitioned to the designated contractor upon final delivery and thusly no longer under the purview of the Space Network Project Configuration Control Board to manage, maintain, or further change. At such time this guide shall become a reference document, and questions should be addressed to the designated contractor and/or:

Space Network Project Manager
Code 452
Goddard Space Flight Center
Greenbelt, MD 20771

Abstract

The primary function of the Space Network (SN) Web Services Interface (SWSI) is to provide a standards-based cross-platform customer interface for performing Tracking and Data Relay Satellite (TDRS) and Demand Access System (DAS) scheduling and real-time service monitoring and control. A secure interface is provided to allow these functions to be performed either from the NASA Integrated Services Network (NISN) Internet Protocol (IP) Operational Network (IONET) or from the Internet.

This Operator's Guide provides the necessary information and guidelines for Data Services Management Center (DSMC) operators, system administrators, and database administrators to perform the required steps for configuration and operation of the servers in support of daily SWSI operations.

Keywords: *SWSI, NCCDS, SN, TDRS, DAS, DASCON*

Change Information Page

List of Effective Pages			
Page Number	Issue		
Cover	DCN 1		
Signature Page	DCN 1		
Preface	Original		
Abstract	Original		
Change Information Page	DCN 1		
DCN Control Sheet	DCN 1		
xi through xxiii	DCN 1		
1-1 through 1-3/1-4	DCN 1		
2-1 through 2-7/2-8	Original		
3-1 through 3-49	DCN 1		
4-1 through 4-6	DCN 1		
5-1 through 5-4	Original		
6-1 through 6-12	Original		
7-1 through 7-40	DCN 1		
8-1 through 8-4	Original		
9-1 through 9-19	DCN 1		
A-1 through A-5/A-6	Original		
B-1 through B-4	DCN 1		
Document History			
Document Number	Status/Issue	Publication Date	CCR Number
452-SOG-SWSI	Original	September 2003	452/_____
452-SOG-SWSI	DCN 1	February 2004	

Contents

Preface	iii
Abstract	v
Section 1. Introduction	1-1
1.1 Purpose	1-1
1.2 Background	1-1
1.3 Document Organization.....	1-2
1.4 Applicable Documents.....	1-3
Section 2. SWSI Overview	2-1
2.1 SWSI System Description	2-1
2.2 System Environment	2-3
2.2.1 General.....	2-3
2.2.2 Network Control Center (NCC) Data System (NCCDS) Operations.....	2-3
2.2.3 Auxiliary Network Control Center (ANCC).....	2-3
2.2.4 Service Planning Segment (SPS).....	2-3
2.2.5 Communications and Control Segment (CCS).....	2-3
2.2.6 NCCDS Protocol Gateway (NPG)	2-4
2.2.7 NCCDS TDRSS Unscheduled Time (TUT) Server	2-4
2.2.8 Demand Access System (DAS).....	2-4
2.2.9 DAS Controller (DASCON).....	2-4
2.2.10 NISN Secure Gateway	2-4
2.3 Server Hardware	2-5
2.4 Server Software.....	2-6
Section 3. Server Configuration	3-1
3.1 Overview	3-1
3.2 Application Server	3-3
3.2.1 Application Server Invocation and Usage.....	3-3
3.2.2 Application Server Main Properties	3-3
3.2.3 Application Server Secure Socket Layer (SSL) Properties.....	3-8
3.2.4 Application Server Certificate Authority (CA) Properties	3-9
3.3 Isolator.....	3-9
3.3.1 Isolator Invocation and Usage.....	3-9

	3.3.2	Isolator Properties	3-10
3.4		SWSI-NCCDS Interface	3-15
	3.4.1	SNIF Invocation and Usage.....	3-15
	3.4.2	SNIF Configuration	3-16
3.5		SWSI-DAS Interface.....	3-18
	3.5.1	SDIF Invocation and Usage.....	3-18
	3.5.2	SDIF Configuration	3-19
3.6		TUT Proxy Sender.....	3-24
	3.6.1	TUT Proxy Sender Invocation and Usage	3-24
	3.6.2	TUT Proxy Sender Properties	3-25
3.7		TUT Proxy Receiver	3-27
	3.7.1	TUT Proxy Receiver Invocation and Usage.....	3-27
	3.7.2	TUT Proxy Receiver Properties.....	3-28
3.8		Certification Generation Tool.....	3-30
	3.8.1	Certificate Generation Tool Invocation and Usage	3-30
	3.8.2	Certificate Generation Tool CGI Configuration Parameters ..	3-31
	3.8.3	Certificate Generation for Client Users	3-35
	3.8.4	Certificate Generation for Server Processes	3-38
	3.8.5	Certificate Generation for SWSI CA	3-39
3.9		High Availability Application.....	3-40
	3.9.1	General.....	3-40
	3.9.2	HA System Startup Script.....	3-41
	3.9.3	Operator cshrc Script	3-42
	3.9.4	HA Configuration File	3-43
	3.9.5	HA Services File	3-44
	3.9.6	HA Startup Script.....	3-45
	3.9.7	HA Shutdown Script	3-45
	3.9.8	Service Scripts	3-46
3.10		NISN Secure Gateway	3-48

Section 4. User Environment 4-1

	4.1	Overview	4-1
	4.2	Backend Server Menus	4-2
	4.3	Open Server Menus	4-3
	4.4	Logging Off.....	4-5

Section 5. Customer and User Setup..... 5-1

	5.1	General.....	5-1
	5.2	Adding Customers.....	5-1
	5.3	Adding SWSI Client Users.....	5-3

5.4	IP Address Filtering	5-3
5.5	SSC Management	5-3
5.6	Client User Login Problems.....	5-4
Section 6. Database Design and Management.....		6-1
6.1	Database Schema.....	6-1
6.2	Server Configuration	6-6
6.2.1	Overview	6-6
6.2.2	Parameter Files	6-6
6.2.3	System Parameters.....	6-7
6.2.4	SQL*Net Configuration Files	6-7
6.2.5	Table Spaces	6-7
6.2.6	Control Files	6-8
6.2.7	Redo Logs.....	6-9
6.2.8	Rollback Segments	6-9
6.3	Backup and Recovery	6-10
6.3.1	Hot Backup.....	6-10
6.3.2	Troubleshooting.....	6-10
6.3.3	Loss of Redo Log	6-11
6.3.4	Recovery from Operator Error.....	6-11
6.3.5	Recovery from Server Hardware Crash.....	6-11
6.3.6	Recovery from RAID Hardware Crash	6-11
6.3.7	Recovery from Loss of Data File(s)	6-11
6.3.8	Recovery from Loss of Control File(s)	6-12
6.4	Failover.....	6-12
6.5	Oracle Accounts	6-12
Section 7. Database Administration.....		7-1
7.1	Overview	7-1
7.2	User Administration	7-2
7.2.1	General.....	7-2
7.2.2	Display user Menu Option	7-3
7.2.3	Display users logged in Menu Option	7-3
7.2.4	Display activity log Menu Option	7-5
7.2.5	Mark a user as logged off Menu Option	7-6
7.2.6	Deactivate a user Menu Option.....	7-6
7.2.7	Reactivate a user Menu Option.....	7-7
7.2.8	Change user privilege Menu Option	7-7
7.2.9	Change user contact information Menu Option	7-7
7.2.10	Set a user's password expiration date Menu Option	7-7

7.2.11	Set a user's account expiration date Menu Option	7-8
7.2.12	Add a user Menu Option.....	7-8
7.2.13	Change user password Menu Option	7-8
7.2.14	Enter encrypted user password Menu Option	7-9
7.2.15	Delete a user Menu Option.....	7-9
7.2.16	Display SICs authorized for each user Menu Option.....	7-9
7.2.17	Display users authorized for each SIC Menu Option.....	7-10
7.2.18	Add a SIC for a user Menu Option	7-10
7.2.19	Remove a SIC for a user Menu Option	7-10
7.3	NCCDS Schedule Connection Administration.....	7-11
7.3.1	General.....	7-11
7.3.2	Display Schedule Connections Menu Option	7-11
7.3.3	Monitor Connection Status Menu Option.....	7-12
7.3.4	Add Schedule Connection Menu Option	7-13
7.3.5	Delete Schedule Connection Menu Option.....	7-13
7.3.6	Update Schedule Connection Menu Option	7-14
7.3.7	Display Schedule Connection SIC Assignments Menu Option.....	7-14
7.3.8	Add a SIC to a Schedule Connection Menu Option.....	7-14
7.3.9	Remove a SIC from a Schedule Connection Menu Option ...	7-15
7.3.10	Enable Schedule Status Connection Menu Option.....	7-15
7.3.11	Disable Schedule Status Connection Menu Option.....	7-15
7.4	NCCDS Realtime Connection Administration.....	7-16
7.4.1	General.....	7-16
7.4.2	Display Realtime Connections Menu Option	7-16
7.4.3	Monitor Connection Status Menu Option.....	7-17
7.4.4	Add Realtime Connection Menu Option	7-17
7.4.5	Delete Realtime Connection Menu Option	7-18
7.4.6	Update Realtime Connection Menu Option	7-18
7.4.7	Display Realtime Connection SIC Assignments Menu Option.....	7-18
7.4.8	Add a SIC to a Realtime Connection Menu Option	7-19
7.4.9	Remove a SIC from a Realtime Connection Menu Option	7-19
7.5	SIC Administration.....	7-19
7.5.1	General.....	7-19
7.5.2	Display SICs Menu Option	7-20
7.5.3	Update a SIC Menu Option	7-21
7.5.4	Add a SIC Menu Option	7-21
7.5.5	Remove a SIC Menu Option.....	7-21
7.5.6	Purge schedule requests for a SIC Menu Option	7-22
7.5.7	Delete a specific schedule request Menu Option	7-22
7.5.8	Purge all schedule requests for a SIC Menu Option.....	7-22
7.5.9	Purge active events for all SICs Menu Option	7-23

7.6	Prototype Event Code Administration.....	7-23
7.6.1	General.....	7-23
7.6.2	Display Prototype Event Codes Menu Option	7-24
7.6.3	Add a Prototype Event Code Menu Option.....	7-24
7.6.4	Remove a Prototype Event Code Menu Option.....	7-24
7.7	SUPIDEN Administration.....	7-25
7.7.1	General.....	7-25
7.7.2	Display SUPIDENs Menu Option	7-25
7.7.3	Add a SUPIDEN Menu Option.....	7-26
7.7.4	Remove a SUPIDEN Menu Option.....	7-26
7.8	TDRS Name Administration	7-26
7.8.1	General.....	7-26
7.8.2	Display TDRS Names Menu Option	7-27
7.8.3	Display TDRS Groups Menu Option.....	7-28
7.8.4	Add TDRS Name Menu Option	7-28
7.8.5	Remove TDRS Name Menu Option	7-29
7.8.6	Add TDRS Group Menu Option.....	7-29
7.8.7	Remove TDRS Group Menu Option.....	7-30
7.8.8	Add TDRS Names to Group Menu Option.....	7-30
7.8.9	Remove TDRS Names from Group Menu Option	7-30
7.8.10	Update TDRS Name or Group Menu Option	7-31
7.9	SSC Administration	7-31
7.9.1	General.....	7-31
7.9.2	Display NCC SSCs Menu Option	7-33
7.9.3	Display DAS SSCs Menu Option.....	7-33
7.9.4	Add an NCC SSC Menu Option	7-34
7.9.5	Add a DAS SSC Menu Option.....	7-34
7.9.6	Add initial 10 DAS SSC's (001-010)	7-35
7.9.7	Duplicate an SSC Menu Option.....	7-36
7.9.8	Remove an SSC Menu Option	7-36
7.9.9	Make an SSC editable Menu Option	7-36
7.9.10	Make all DAS SSCs editable Menu Option.....	7-36
7.9.11	Make all NCC SSCs editable Menu Option	7-37
7.9.12	Make an SSC not editable Menu Option	7-37
7.9.13	Make all DAS SSCs not editable Menu Option.....	7-37
7.9.14	Make all NCC SSCs not editable Menu Option	7-37
7.9.15	Add missing SSC parameters	7-37
7.10	Active Schedule Upload Administration.....	7-38
7.10.1	General.....	7-38
7.10.2	Display Active Schedule Upload Parameters entries Menu Option.....	7-39

- 7.10.3 Update Active Schedule Upload Parameters entry Menu Option7-39
- 7.10.4 Add Active Schedule Upload Parameters entry Menu Option 7-40
- 7.10.5 Remove Active Schedule Upload Parameters entry Menu Option7-40

Section 8. Digital Certificate Management..... 8-1

- 8.1 Overview 8-1
- 8.2 Certificate Authority 8-1
- 8.3 Web Server Certificates..... 8-3
- 8.4 Application Server Certificates 8-4
- 8.5 SWSI Client User Certificates..... 8-4

Section 9. System Administration Procedures 9-1

- 9.1 Server Accounts 9-1
- 9.2 root Account Environment 9-1
 - 9.2.1 General..... 9-1
 - 9.2.2 Switch User (su) Command Usage 9-1
 - 9.2.3 System Control..... 9-2
 - 9.2.4 root Account Aliases..... 9-2
- 9.3 Account Management 9-2
 - 9.3.1 Account Creation..... 9-2
 - 9.3.2 Setting Initial Password with passwd Command 9-3
 - 9.3.3 Setting Initial Password with admintool Application..... 9-3
 - 9.3.4 Account Deletion with userdel Command..... 9-5
 - 9.3.5 Account Deletion with admintool Application 9-5
- 9.4 Backup and Recovery 9-5
 - 9.4.1 Backup and Recovery Overview 9-5
 - 9.4.2 Full System Backup..... 9-6
 - 9.4.3 Full System Recovery 9-6
 - 9.4.4 Database Backup 9-9
 - 9.4.5 Database Recovery..... 9-9
 - 9.4.6 Tape Handling..... 9-10
- 9.5 IPFilter..... 9-11
 - 9.5.1 IPFilter Overview 9-11
 - 9.5.2 Adding an IP Address..... 9-13
 - 9.5.3 Removing an IP Address..... 9-13
 - 9.5.4 Adding or Removing an IP Address Interactively 9-15
 - 9.5.5 Listing All IP Addresses..... 9-15
 - 9.5.6 Searching for a Single IP Address..... 9-16
- 9.6 RAID management..... 9-16
- 9.7 cron Procedures 9-19

Appendix A. SNIF Log Messages	A-1
--	------------

Appendix B. Abbreviations and Acronyms	B-1
---	------------

List of Figures

Figure 2-1. High Level SWSI Architecture	2-2
Figure 3-1. Server Interprocess Communication	3-1
Figure 4-1. Backend Server CDE Toolbar	4-1
Figure 4-2. Backend Server Toolbar Menus	4-3
Figure 4-3. Open Server CDE Toolbar.....	4-5
Figure 4-4. Open Server Toolbar Menus	4-6
Figure 6-1. SWSI Database Scheme (1 of 3).....	6-3
Figure 6-2. SWSI Database Scheme (2 of 3).....	6-4
Figure 6-3. SWSI Database Scheme (3 of 3).....	6-5
Figure 9-1. Solaris admintool Application.....	9-4
Figure 9-2. RAID Manager Main Control Window	9-18
Figure 9-3. RAID Manager Configuration Window	9-18

List of Tables

Table 3-1, Secure Gateway Rules	3-49
Table 6-1. SWSI Database Tables	6-2
Table 6-2. Logical Volumes	6-8
Table A-1. SNIF Log Messages (1 of 5).....	A-1

Section 1. Introduction

1.1 Purpose

This Space Network (SN) Web Services Interface (SWSI) Server Operator's Guide provides instructions for configuring and operating the SWSI servers located in the Data Services Management Center (DSMC) at the White Sands Complex (WSC).

1.2 Background

The primary function of SWSI is to provide a network-based web interface to the Network Control Center (NCC) Data System (NCCDS) and to the Demand Access System (DAS) to perform SN customer scheduling, real-time service monitoring and control, and state vector storage. The SWSI provides the following capabilities:

- Standards-based customer interface for performing TDRS scheduling, real-time service monitoring and control
- Access from the Internet and NASA Integrated Services Network (NISN) Open & Closed Internet Protocol (IP) Operational Network (IONet)
- Secure access through encryption, certification, and authentication
- Cross-platform compatible client application
- Java-based Graphical User Interface (GUI)
- Supports full NCCDS/Mission Operations Center (MOC) interface, including flexible scheduling
- Ability to transmit customer state vectors to SN
- Orbiting or stationary state vector generation based on user input of geocentric (position & velocity) or geodetic (latitude, longitude, & altitude) coordinates
- Internet and Open IONet access to TDRSS Unscheduled Time (TUT)
- Test mode for performing Engineering Interface (EIF) testing and user training
- Minimal user requirements – Windows or Unix workstation with Java Virtual Machine (freeware), web browser, and SWSI client application software

1.3 Document Organization

This document is organized into 9 sections and 2 appendices. Following the Introduction (Section 1), this document presents procedures and reference material on the specified topics in the following order:

- SWSI Overview (Section 2)
- Server Configuration (Section 3)
- User Environment (Section 4)
- Customer and User Setup (Section 5)
- Database Design and Management (Section 6)
- Database Administration (Section 7)
- Digital Certificate Management (Section 8)
- System Administration Procedures (Section 9)
- SNIF Log Messages (Appendix A)
- Abbreviations and Acronyms (Appendix B)

1.4 Applicable Documents

1. *High Availability (HA) User's Guide*, 451-HAUG/NCC98
2. *Space Network (SN) Web Services Interface (SWSI) Client Software User's Guide*, 452-UG-SWSI
3. *NCCDS Protocol Gateway (NPG) Operator's Guide*, 451-NPGUG/NCC98
4. *Space Network (SN) Web Services Interface (SWSI) System Design Specification*, 452-SDS-SWSI
5. *Space Network (SN) Web Services Interface (SWSI) Security Plan*, 452-SP-SWSI
6. *Interface Control Document Between the Network Control Center Data System and Mission Operations Center*, 451-ICD-NCCDS/MOC
7. *Interface Control Document Between Demand Access Service (DAS) and Space Network Web Services Interface (SWSI)*, 453-ICD/DAS-SWSI
8. *Sun StorEdge RAID Manager 6.22 User's Guide*, Sun Part #806-0478-10, September 1999, Revision A
9. *Sun StorEdge RAID Manager 6.22 Installation and Support Guide*, Sun Part #805-7756-10, September 1999, Revision A
10. *Sun StorEdge RAID Manager 6.22 Release Notes*, Sun Part #805-7758-11, November 1999, Revision A
11. *Sun StorEdge A1000 and D1000 Installation, Operations, and Service Manual*, Sun Part No. 805-2624-10, Revision A, February 1998
12. *Oracle 8i Backup and Recovery Handbook*, Oracle Press

Section 2. SWSI Overview

2.1 SWSI System Description

A block diagram showing the high level SWSI architecture is shown in Figure 2-1. The main hardware components of SWSI are as follows:

- Client Workstation - user's desktop workstation, which can be any desktop that supports Sun Microsystems' Java Virtual Machine (JVM) 1.4.1.
- Backend Server – hosts most of the SWSI server applications; manages user login sessions, database storage, and the communications with NCCDS and DAS.
- Open Server – proxy server to allow Open IONet and Internet-based users to connect to SWSI and to access TUT. User requests are directed to Backend Server through the NISN Secure Gateway using a single predefined set of rules. This allows for the addition of new customers and users without the need for adding new Secure Gateway rules.

The main software components of SWSI are as follows:

- Client – executes on Client Workstation, provides Graphical User Interface (GUI) for performing SWSI client operations.
- Application Server – server process that the Client connects to in order to access SWSI services; keeps track of user requests and provides responses back to the Client. The Application Server runs on both the Open Server and the Backend Server.
- Isolator – server process that provides an interface for the Client with the SWSI Database; processes user requests and generates responses; communicates with the Client through the Application Server. A separate Isolator is required for each Application Server.
- SWSI-NCCDS Interface (SNIF) – server process that communicates with the NCCDS using the messaging protocol defined in the *NCCDS/Mission Operations Center (MOC) Interface Control Document (ICD)*. A separate SNIF is required to communicate with each NCCDS (operations and test).
- SWSI-DAS Interface (SDIF) – server process that communicates with the Demand Access System (DAS). Only one SDIF is required since there is no test DAS.
- Database – backend data storage; holds all customer configuration and scheduling data; allows access to customer schedules from any Client Workstation from any IP network for any authorized user.
- Open TUT Server – web server that mirrors the TUT web service provided by NCCDS on the Closed IONet. The Open TUT Server data is updated hourly.

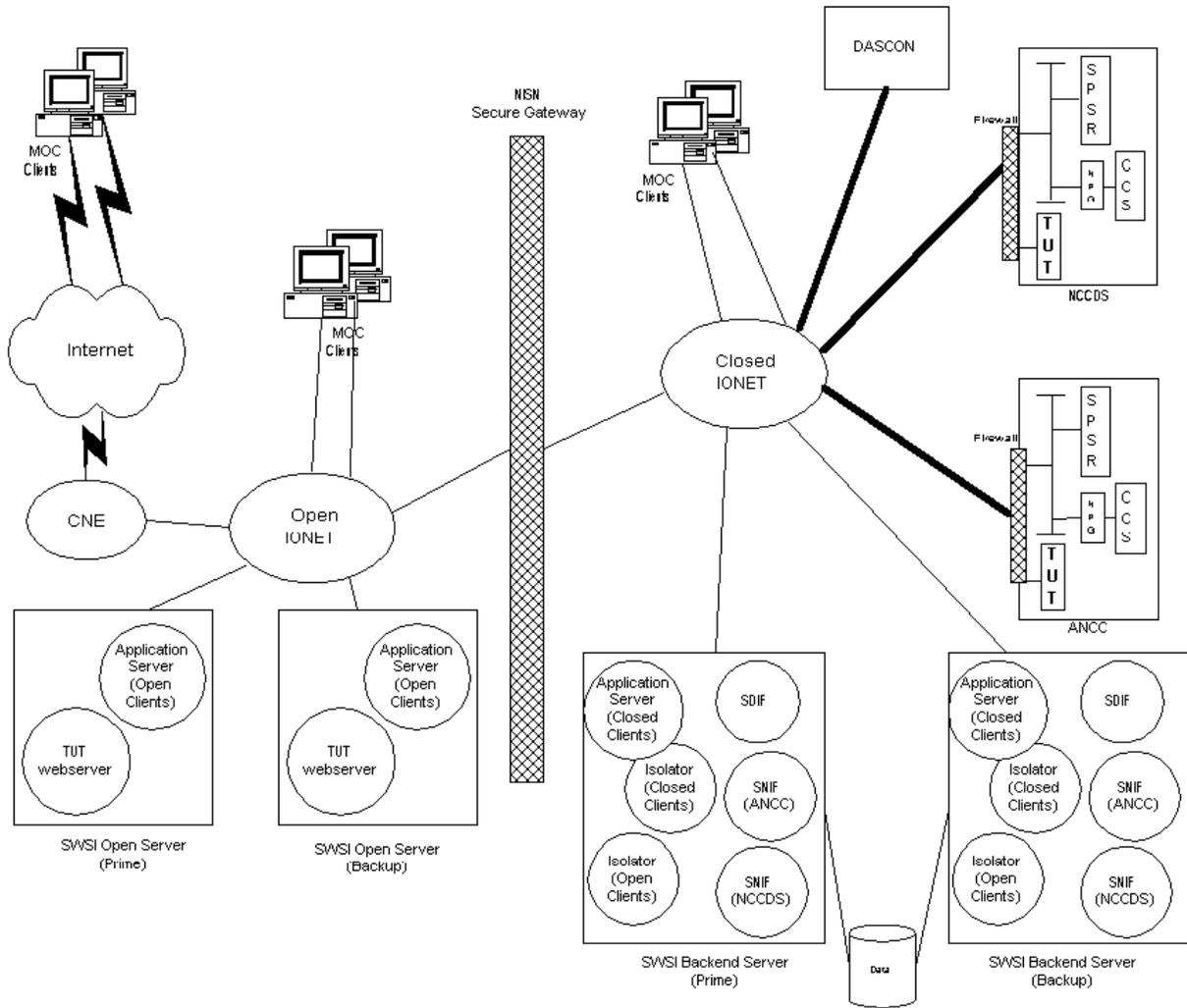


Figure 2-1. High Level SWSI Architecture

2.2 System Environment

2.2.1 General

This section describes the environment in which the SWSI operates and briefly discusses the interactions with external systems with which the SWSI interfaces.

2.2.2 Network Control Center (NCC) Data System (NCCDS) Operations

The NCC serves as the central control facility of the Spaceflight Tracking and Data Network (STDN), which consists of the Space Network (SN) and Ground Network (GN). The SN includes the Tracking and Data Relay Satellites (TDRSs) and two ground terminals, the White Sands Ground Terminal (WSGT) and the Second TDRSS Ground Terminal (STGT). The NCC schedules, controls, and ensures the reliability of the SN. The NCC is located within the Data Services Management Center (DSMC) at STGT. The SWSI communicates with the operations NCCDS on behalf of SWSI customers through implementation of the *NCCDS/MOC Interface Control Document (ICD)* protocol. All communications use Transmission Control Protocol (TCP) and are limited to those messages designated for full support customers.

2.2.3 Auxiliary Network Control Center (ANCC)

The ANCC serves primarily as a test facility for testing new NCCDS software releases and for performing Engineering Interface (EIF) tests with customer MOCs. ANCC is located at WSGT. SWSI interfaces with the ANCC to allow SWSI customers to perform interface testing and user training.

2.2.4 Service Planning Segment (SPS)

The SPS is the primary NCCDS subsystem used for performing SN service planning. SPS receives and validates customer service requests, generates and maintains the schedule, and disseminates the schedule to the appropriate SN elements and customers. The SPS also receives acquisition data from the Flight Dynamics Facility (FDF) and SN customers, stores the data, and disseminates acquisition data to WSGT and STGT. The SWSI maintains TCP connections with SPS for performing scheduling and vector storage on behalf of each SWSI customer.

2.2.5 Communications and Control Segment (CCS)

The CCS is the primary NCCDS subsystem used for performing SN service control and service assurance. Customers are able to perform real-time reconfiguration of an ongoing service through the use of Ground Control Message Requests (GCMRs). CCS is used to monitor the performance of active events and passes this information to customers in the form of User Performance Data (UPD) messages.

2.2.6 NCCDS Protocol Gateway (NPG)

The NPG performs message protocol translation between legacy entities that communicate in 4800 BBs and newer entities that use TCP messages. Since CCS communicates using 4800 BB protocol and the SWSI communicates using TCP, the SWSI establishes real-time connections with the NPG, using the NPG as a TCP proxy for the CCS.

2.2.7 NCCDS TDRSS Unscheduled Time (TUT) Server

The TUT World Wide Web (WWW) Server provides information about unscheduled TDRS resources. It consists of start and stop times of unscheduled use of the Single Access (SA), Multiple Access Forward (MAF), and S-band Multiple Access Forward (SMAF) antennas, and Multiple Access Return (MAR) and S-band Multiple Access Return (SMAR) links for each TDRS. This data is essentially the unused time in the schedule. The NCCDS TUT Server provides this service only to customers located on the Closed IONET.

2.2.8 Demand Access System (DAS)

The DAS expands the existing TDRSS Multiple Access Return (MAR) capabilities by building upon the Third Generation Multiple Access Beamforming Subsystem (TGBFS). The existing TDRSs provide pre-scheduled communication service to customers by using ground-based electronics to process signals emanating from customers that are relayed by the TDRS on-board phased array antenna systems. The TGBFS expands the capability of the TDRSs MAR system and allows service to be provided on a demand basis rather than on a pre-scheduled basis.

2.2.9 DAS Controller (DASCON)

The DASCON is responsible for scheduling and controlling all DAS-related hardware at the White Sands Complex (WSC). The SWSI communicates with the DASCON on behalf of SWSI customers through implementation of the *DAS/SWSI Interface Control Document (ICD)* protocol. All communications use Transmission Control Protocol (TCP).

2.2.10 NISN Secure Gateway

The NISN Secure Gateway is a rule-based firewall used to prevent penetration of hosts on the Closed IONET from less secure networks. A small number of rules are used to allow connection between the Open Server and the Backend Server components. All message traffic is channeled through this path. The rule set is static, meaning that Secure Gateway changes are not required in response to SWSI customers being added or removed.

2.3 Server Hardware

The SWSI servers are hosted on Sun UltraSparc workstations. Two workstations are provided in a redundant configuration for the backend and open servers, resulting in a total of four workstations. These workstations are intended to operate with minimal operator intervention. The backend servers consist of the following hardware:

- Sun Microsystems Blade 1000 desktop workstation
- 36 GByte SCSI disk drive
- 2 GByte Random Access Memory (RAM)
- Built-in 10/100 Mbps Ethernet interface
- Quad 10/100 Mbps Ethernet interface expansion card
- Differential SCSI expansion card
- DVD-ROM drive
- 21" color monitor
- 4mm 20 Gbyte DDS-4 tape drive

A RAID array connected to both backend servers is used for database storage. It consists of the following:

- Sun Microsystems 72 Gbyte Storedge A1000 External Raid Array

The open servers consist of the following:

- Sun Microsystems Ultra 2 desktop workstation
- 9 GByte SCSI disk drive
- 640 MByte RAM
- Built-in 10/100 Mbps Ethernet interface
- Quad 10/100 Mbps Ethernet interface expansion card
- CD-ROM drive
- 21" color monitor
- External 12 Mbyte 4mm DDS-3 tape drive

2.4 Server Software

Each server contains the following Commercial Off-The-Shelf (COTS) or freely available software that are essential to running the SWSI applications:

- Sun Microsystems Solaris 8 Operating System
- Java Runtime Environment (JRE) version 1.4.1_02
- Java Development Kit (JDK) version 1.4.1 Java archiver (jar)
- Oracle Database Management System (DBMS) version 8.1.6 (backend server only)
- Oracle JDBC Driver version 9.0.1
- Phaos J/CA Toolkit version 1.11-4
- Phaos SSLava Tookit 1.3
- Apache 1.3.27
- OpenSSL Ben-SSL 1.48
- CohProg SaRL Network Consulting (<http://www.cohprog.com/>) Apache Mod_bandwidth version 2.0.4
- Sun StorEdge RAID Manager version 6.22 (backend server only)
- TCPWrappers version 7.6
- IPFilter version 3.4.31
- wget version 1.8.2

Each server contains the following Government Off-The-Shelf (GOTS) applications:

- High Availability (HA) Application – used to control the execution of critical server processes. Ensures that only one server in an HA pair is executing the processes at any one time. This application was developed as part of the Network Control Center (NCC) 1998 system release to execute on Sun Microsystems platforms.
- HA Graphical User Interface (GUI) – used to monitor status of the HA application
- NCCDS Protocol Gateway (NPG) Delogger – used to view SNIF logs in real time. This application was developed as part of the NPG system for NCC 98.

Each server contains the following SWSI applications:

- Application Server – executed under HA control.
- Isolator (backend server only) – executed under HA control.

- SNIF (backend server only) – executed under HA control
- SDIF (backend server only) – executed under HA control
- TDRSS Unscheduled Time (TUT) Proxy Sender (backend server only) – executed under *cron* control. Periodically retrieves TUT data files from NCCDS operations and ANCC TUT servers and transmits them to the SWSI open servers. Also receives user-generated digital certificates from open servers for archival on backend servers.
- TUT Proxy Receiver (open server only) – receives TUT data files transmitted by TUT Proxy Sender from backend server and stores files in appropriate locations on the open server so that they are accessible to users through the TUT web page. Also sends user-generated digital certificates to backend server for archival.
- SWSI Web Page – contains user digital certificate generation tool and forms, as well as SWSI Client software for users to download.
- TUT Web Page (open server only) – a mirror of the TUT web page provided by the NCC. Allows user to access TUT via the Internet and Open IONet.
- Certificate Generator – accessed by user via the SWSI Web Page. Used to generate digital certificates for SWSI Client users, SWSI server processes, and the SWSI Certificate Authority (CA).

Section 3. Server Configuration

3.1 Overview

This section contains detailed information on how to configure and execute the various server applications. The Application Server, Isolator, SWSI-NCCDS Interface (SNIF), and SWSI-DAS Interface (SDIF) must be configured so as to communicate with each other using predetermined Internet Protocol (IP) addresses, Transmission Control Protocol (TCP) ports, and User Datagram Protocol (UDP) ports. The diagram shown in Figure 3-1 depicts this communication. For TCP connections, the arrows indicate the direction for connection establishment (client to server). For UDP connection, the arrows indicate actual data flow. The ports that each process listens on as shown in the diagram are the same as shown in the example property files to follow in later subsections. This port assignment is meant only as an example and does not preclude configuration on ports other than those shown in the diagram.

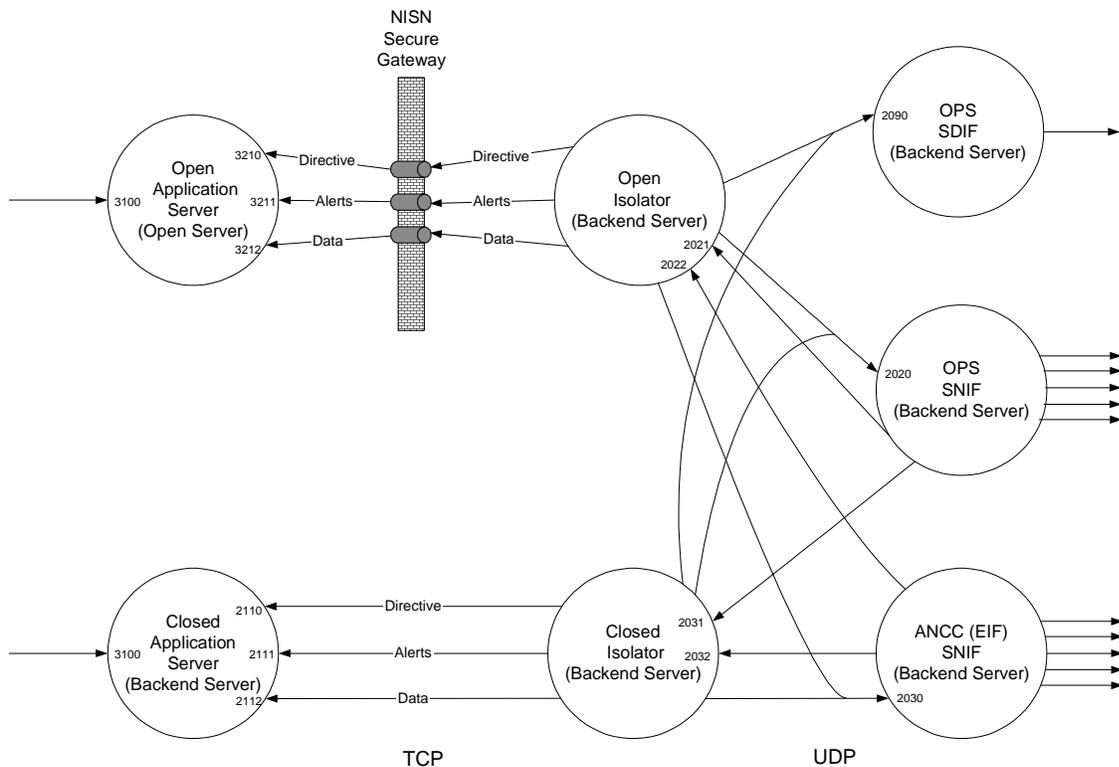


Figure 3-1. Server Interprocess Communication

The communication channels are defined as follows:

- Client-Application Server TCP Connection – the user Client software connects to the SWSI server via a single TCP connection to the Application Server. Separate Application Servers are provided for connection from the Internet or Open IONet, and from the Closed IONet. The port that the Client connects to is defined by the Application Server *clientServerConnectionPort* property.
- Application Server-Isolator TCP Connections – a separate Isolator is associated with each Application Server. The Isolator connects with its associated Application Server via three TCP connections defined as follows:
 - Directive Port – directives or requests sent by Clients and forwarded by Application Server to Isolator.
 - Events (Alerts) Port – alerts and User Performance Data UPD generated or forwarded by the Isolator to the Application Server.
 - Data Port – responses to directives or other data, such as Time Transfer Messages (TTMs), generated or forwarded by the Isolator to the Application Server.

These ports are defined on the Isolator by specifying a base port for the Directive Port, with the Events and Data ports implied by adding one and two respectively to the base port. The host and base port are defined on the Isolator by the *SWSIServerName* and *SWSIServerPort* properties. The three ports on the Application Server are defined separately by the *isolatorServerDirectivePort*, *isolatorServerEventsPort*, and *isolatorServerDataPort* properties.

- Isolator-SNIF UDP Channels – UDP, a connectionless protocol, is used for the interface between the Isolators and the SNIFs. Separate SNIFs are used for communication with the operations NCCDS and with the Auxiliary NCC (ANCC). Each Isolator is required to communicate with both SNIFs and vice versa. This provides access to both the operations NCCDS and the ANCC for Clients on both the Open IONet and the Closed IONet. Each SNIF listens on a single UDP port. Each Isolator listens on a separate UDP port for each SNIF. The hosts and ports on the Isolator are defined by the *SNIFHostName*, *SNIFnormPortNumber*, *SNIFnormInPortNumber*, *SNIFeifPortNumber*, and *SNIFeifInPortNumber* properties. The hosts and ports on the SNIF are defined by the *IsolatorReadHost*, *IsolatorReadPort*, *Isolator1WriteHost*, *Isolator1WritePort*, *Isolator2WriteHost*, and *Isolator2WritePort* properties.
- SNIF-NCCDS TCP Connections – SNIF communicates with NCCDS or ANCC using the TCP messaging protocol as defined in the NCCDS/MOC ICD. A separate set of connections is maintained on behalf of each SWSI customer SIC.
- SDIF-DASCON TCP Connection – SDIF communicates with the Operational DASCON or the DASCON Testbed using the TCP messaging protocol as defined in the DAS/SWSI ICD. A single connection is maintained and used for all SWSI customer SICs. DASCON can only accept a single connection from SDIF at time.

The remaining subsections describe in detail how to configure and invoke each application. Property files are used to configure Java-based applications. These are essentially configuration or preference files that are used to set up and control the execution of the server application. In general, the property files may be changed with any text editor. A “:” or “=” character separates the property name from its assigned value. A “!” or “#” character at the beginning of a line is used to enter a comment. Boolean properties, which should be evident from examining the file(s), can have a value of *true* or *false*.

The SNIF configuration files follow a different format that will be described separately in the SNIF subsection.

3.2 Application Server

3.2.1 Application Server Invocation and Usage

The Application Server is normally executed under HA control using preconfigured scripts and property files, but may be invoked interactively as follows:

```
java -Duser.timezone=GMT -classpath SWSI-OPS-cots.jar:SWSIServer.jar \
gov.nasa.gsfc.swsi.server.net.SWSIServer <propfile>
```

Where *propfile* is the name of the main property file, usually *serverMain.prop*. If a *propfile* is not given, then the Application Server looks for a file named *serverOther.prop* in the current working directory.

If invoked with a *-v* option as follows:

```
java -classpath SWSI-OPS-cots.jar:SWSIServer.jar gov.nasa.gsfc.swsi.server.net.SWSIServer -v
```

Then the Application Server just displays the version number and exits, similar to the following:

```
SWSI Server version Build 4 Patch 02 04/21/2003
```

3.2.2 Application Server Main Properties

The property file name passed as an argument in the invocation line is the primary Application Server configuration file. Following is a sample file:

```
! @(#)other.prop 1.4 06/27/00
! other (non-SSL) SWSI server properties
!
ProcessIDFile : /tmp/aserver.pid
GetPIDCommand : /export/home/swsiops/SWSI03.1/bin/write_pid

clientServerConnectionPort : 3100
isolatorServerDirectivePort : 3210
isolatorServerEventsPort : 3211
isolatorServerDataPort : 3212

CompressDirSocket : true
CompressMnemonicSocket : true
CompressAlertSocket : true
CompressClientSocket : true
```

```

maxUpdateFrequency : 2
maxQueuePriority : 5
queueSize : 200
resetCount : 50
resetTime : 60
! resetTime in min
resetDelay : 500
!resetDelay in msec
debugOn : true
!
! path to SSL property file
sslPropFilePath : /export/home/swsiops/SWSI03.1/properties/serverSSL.prop
!
! path to CA fingerprints property file
caPropFilePath : /export/home/swsiops/SWSI03.1/properties/serverCA.prop
!
! log file directories and options
currentLogDir : /export/home/swsiops/SWSI03.1/logs/server/current
archiveLogDir : /export/home/swsiops/SWSI03.1/logs/server/archive
maxActivityLogSize : 10000000
maxBadLoginLogSize : 10000000
logEvents : false
logMnemonics : false
!
! option to handle logins by server
handleLoginLocally : false
passwordFile : password.pwd
passPhraseMinSize : 30
numLoginsAllowed : 3
numDaysBeforePsswdExpire : 30
numDaysBeforePssPhraseExpire : 60
!
! password validation handled locally
minPasswordRequirements : 3
passwordMinSize : 8
!
! socket buffer sizes
ClientOutputBufferSize : 2048
ClientInputBufferSize : 512
IsoDirInputBufferSize : 512
IsoDirOutputBufferSize : 512
IsoAlertsInputBufferSize : 512
IsoAlertsOutputBufferSize : 512
IsoDataInputBufferSize : 2048
IsoDataOutputBufferSize : 512
!
! Thread priorities (1=min, 10=max, 5=normal)
! normally set EIF client below others
IsoDirThreadPriority : 5
IsoMnemonicThreadPriority : 5
IsoAlertThreadPriority : 5
opsClientPriority : 5
eifClientPriority : 3

```

The meaning of each property is given below:

- ProcessIDFile – file to contain the Unix ID of the Application Server process.
- GetPIDCommand – command to run to determine the Application Server Unix process ID.
- clientServerConnectionPort – the server port that Clients connect to.

- *isolatorServerDirectivePort* – one of three server ports that the Isolator connects to. Directives or requests sent by Clients to the Application Server are forwarded to the Isolator over this port. The value should match the value of the Isolator *SWSIServerPort* property
- *isolatorServerEventsPort* - one of three server ports that the Isolator connects to. Alerts generated or forwarded by the Isolator are sent to this port. The value should be the value of the *isolatorServerDirectivePort* property plus one.
- *isolatorServerDataPort* - one of three server ports that the Isolator connects to. Responses to requests or other data, such as UPD, generated or forwarded by the Isolator to the Application Server are sent to this port. The value should be the value of the *isolatorServerDirectivePort* property plus two.
- *CompressDirSocket* – indicates whether incoming data on Directive socket is compressed. The value should match the value of the Isolator *CompressDirSocket* property. This property actually has no effect since no data is received on the Directive socket. It is provided only for completeness.
- *CompressMnemonicSocket* - indicates whether incoming data on Mnemonic socket is compressed. The value should match the value of the Isolator *CompressMnemonicSocket* property.
- *CompressAlertSocket* - indicates whether incoming data on Alert socket is compressed. The value should match the value of the Isolator *CompressAlertSocket* property.
- *CompressClientSocket* - indicates whether outgoing data on Client socket is compressed.
- *maxUpdateFrequency* – unused.
- *maxQueuePriority* – number of the lowest priority alert that will be passed on to the Clients by the Application Server. One (1) is the highest priority (critical). The default value for this property is 5.
- *queueSize* – maximum number of alerts that can be queued by the server before previous alerts are discarded. The default value is 200.
- *resetCount* – count of acknowledgments from the Client. When this count is reached the *ObjectOutputStream* to the Client is reset and the count is restarted. The default value is 50. The reset is needed to keep memory from leaking on the Client.
- *resetTime* – time in minutes when the *ObjectOutputStream* associated with the Directive port to the Isolator (given by *isolatorServerDirectivePort*) is reset and the time restarted. The default value is 60. The reset is needed to keep memory from leaking on the Isolator.

- `resetDelay` – time in milliseconds that the Application Server prevents new directives from being written to the Isolator before resetting the `ObjectOutputStream`. The default value is 500. This delay allows the Isolator to read any remaining directives off the Directive port before the `ObjectOutputStream` is reset. Once the `ObjectOutputStream` is reset, any unread directives are lost.
- `debugOn` – defines whether debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems.
- `sslPropFilePath` – path and name of the SSL property file. The default value is `./serverSSL.prop`. The contents of this file are described in detail in Section 3.2.3.
- `caPropFilePath` - path and name of the Certificate Authority property file. The contents of this file are described in detail in Section 3.2.4.
- `currentLogDir` – path to the directory for current (active) logs. The default value is the current directory.
- `archiveLogDir` - path to the directory to which previous logs are moved. The default value is the current directory.
- `maxActivityLogSize` – maximum activity log file size in bytes. When the log file reaches this size, the current log file is closed and a new log file is started. The default value is 10000000.
- `maxBadLoginLogSize` - maximum bad login log file size in bytes. When the log file reaches this size, the current log file is closed and a new log file is started. The default value is 10000000.
- `logEvents` – defines whether to log alert activations in the activity log. This logs the Client’s initial request to receive alerts. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- `logMnemonics` – defines whether to log Client data requests (mnemonic activations and deactivations) in the activity log. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- `handleLoginLocally` – defines whether to perform user authentication locally rather than to forward login requests to the Isolator. The default value is *false*.
- `passwordFile` – name of file containing user Ids, passwords, etc. for when *handleLoginLocally* is *true*.
- `passPhraseMinSize` – minimum size allowed for a new passphrase. This is sent to the Client when the Client’s passphrase expires and the server forces a change. This property is only used if *handleLoginLocally* is *true*. The default value is 30.

- numLoginsAllowed – number of failed login attempts allowed before the user ID is deactivated. This property is only used if *handleLoginLocally* is *true*. The default value is 3.
- numDaysBeforePsswdExpire – number of days after which the password associated with a user ID needs to be changed. This property is only used if *handleLoginLocally* is *true*. The default value is 30.
- numDaysBeforePssPhraseExpire - number of days after which the passphrase associated with a user's private key needs to be changed. This property is only used if *handleLoginLocally* is *true*. The default value is 30.
- minPasswordRequirements – minimum number of different character types needed (upper case, lower case, numeric, or special characters) in a new user password. This property is only used if *handleLoginLocally* is *true*. The default value is 3.
- passwordMinSize – minimum number of characters required in new user passwords. This property is only used if *handleLoginLocally* is *true*. The default value is 8.
- ClientOutputBufferSize – buffer size in bytes for the object output stream to a Client. Buffer sizes should be in increments of 512. The default value is 2048.
- ClientInputBufferSize - buffer size in bytes for the object input stream from a Client. Buffer sizes should be in increments of 512. The default value is 512.
- IsoDirInputBufferSize - buffer size in bytes for the directive object input stream from the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoDirOutputBufferSize - buffer size in bytes for the directive object output stream to the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoAlertsInputBufferSize - buffer size in bytes for the alert object input stream from the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoAlertsOutputBufferSize - buffer size in bytes for the alert object output stream to the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoDataInputBufferSize - buffer size in bytes for the data object input stream from the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoDataOutputBufferSize - buffer size in bytes for the data object output stream to the Isolator. Buffer sizes should be in increments of 512. The default value is 512.
- IsoDirThreadPriority – priority for the thread that reads the directives socket from the Isolator. The default value is 5 (1=minimum, 5=normal, 10=maximum).
- IsoMnemonicThreadPriority - priority for the thread that reads the mnemonic data socket from the Isolator. The default value is 5 (1=minimum, 5=normal, 10=maximum).

- IsoAlertThreadPriority - priority for the thread that reads the alerts data socket from the Isolator. The default value is 5 (1=minimum, 5=normal, 10=maximum).
- opsClientPriority – priority for threads that read or write data to Client that are in operations mode. The default value is 5 (1=minimum, 5=normal, 10=maximum).
- eifClientPriority - priority for threads that read or write data to Client that are in test (EIF) mode. The default value is 5 (1=minimum, 5=normal, 10=maximum).

3.2.3 Application Server Secure Socket Layer (SSL) Properties

The Application Server SSL property file is specified by the *sslPropFilePath* property in the main property file. The SSL property file contains the properties used for providing encrypted connections to the Client. Following is a sample file:

```
! @(#)SSL.prop 1.1 02/15/00
!SSL settings
!
CipherSuite1 : SSL_RSA_EXPORT_WITH_RC4_40_MD5
CipherSuite2 : SSL_RSA_WITH_3DES_EDE_CBC_SHA

SetDebug : false

UseServerSSL : true

PrivateKeyFile : /export/home/swsiops/SWSI03.1/certs/enc-SWSI-appserver-key.der.10yr
ServerCertificateFile : /export/home/swsiops/SWSI03.1/certs/SWSI-appserver-cert.der.10yr
CertificateAuthorityFile : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-cert.der.10yr

PassPhraseName : This is the server passphrase

! key renegotiation delay in minutes
RenegotiationDelay : 30
```

The meaning of each property is given below:

- CipherSuites – defines the algorithms used to establish the SSL connection. Multiple cipher suites may be allowed by using successive values of *n*. The Client cipher suite must be set to match one of these values.
- SetDebug – defines whether debug output is generated by the Phaos security system. This is normally only useful to a developer when trying to troubleshoot problems.
- UseServerSSL – defines whether the socket connection to the Client will use SSL. This should always be *true*.
- PrivateKeyFile – file name of the Application Server’s encrypted private key file. This is used to digitally sign the server’s public digital certificate for presentation to the Client.
- ServerCertificateFile – file name of the Application Server’s public digital certificate.

- `CertificateAuthorityFile` – file name of the certificate authority’s public digital certificate. This is chained to the Application Server’s public digital certificate to allow the Client to check the digital signature on the digital certificate presented by the Application Server for authentication.
- `PassPhraseName` – the passphrase used to decrypt the Application Server’s private key file.
- `RenegotiationDelay` – delay in minutes after which key renegotiation is performed on the Client socket. The minimum allowed is 30 and the maximum is 120. If the specified value is outside this range, then the default value of 30 is used.

3.2.4 Application Server Certificate Authority (CA) Properties

The Application Server CA property file is specified by the `caPropFilePath` property in the main property file. The CA property file contains properties used for authenticating connections from the SWSI Client. These properties are digital fingerprints of the approved CA. Following is a sample file:

```
caFingerprint1 : 36D05CCE071A825ED57A62B0A6191F8F
caFingerprint2 : 49EFA74C23DF976F492408BEFBEEED497
```

The meaning of each property is given below:

- `caFingerprint1` – specifies the first digital fingerprint the Application Server will accept from authentic clients. This value should only change if the SWSI CA changes.
- `caFingerprint2` - specifies the 2nd digital fingerprint the Application Server will accept from authentic clients. This value should only change if the SWSI CA changes.

3.3 Isolator

3.3.1 Isolator Invocation and Usage

The Isolator is normally executed under HA control using preconfigured scripts and property files, but may be invoked interactively as follows:

```
java -Duser.timezone=GMT -classpath SWSI-OPS-cots.jar:SWSIisolator.jar \
gov.nasa.gsfc.swsi.isolator.Isolator <propfile>
```

Where *propfile* is the name of the property file. If a *propfile* is not given, then the Isolator looks for a file named *IsolatorProperties* in the current working directory.

If invoked with a `-v` option as follows:

```
java -classpath SWSI-OPS-cots.jar:SWSIisolator.jar gov.nasa.gsfc.swsi.isolator.Isolator -v
```

Then the Isolator just displays the version number and exits, similar to the following:

```
SWSI Isolator version Build 4 Patch 02 04/21/2003
```

3.3.2 Isolator Properties

The property file name passed as an argument in the invocation line is the Isolator configuration file. Following is a sample file:

```
#Properties for the ServInterface
SWSIserverName=swsi-server.nascom.nasa.gov
SWSIserverPort=3210
SWSIServerID=open

CompressDirSocket=true
CompressMnemonicSocket=true
CompressAlertSocket=true

#Properties for the SnifInterface
SNIFHostName=localhost
!Port number for Normal/Operation SNIF
!Write to SNIF
SNIFnormPortNumber=2020
!Read from SNIF
SNIFnormInPortNumber=2021
!Delta time in milliseconds to check if SNIFnorm is still up
SNIFnormDeltaTime=2000
!Number of times to attempt a reconnect prior to sending the client a disconnected status
SNIFnormReconnectAttempts=1

!Port number for EIF/Test SNIF
!Write to SNIF
SNIFeifPortNumber=2030
!Read from SNIF
SNIFeifInPortNumber=2022
!Delta time in milliseconds to check if SNIFeif is still up
SNIFeifDeltaTime=2000
!Number of times to attempt a reconnect prior to sending the client a disconnect status
SNIFeifReconnectAttempts=1

#Properties for the SdifInterface
SDIFHostName=localhost
!The connection between the Isolator and SDIF works like this:
!Isolator is the client; SDIF is the server (one SDIF will be running as EIF and the other
will
!be running as NORM)
!Only one port is used for reading and writing (between one instance of SDIF and the
Isolator)
SDIFNormportNumber=2141
SDIFeifportNumber=2140

!Directory to write the TSW file to, SNIF will poll this directory to see if it missed any
files
TSWnormDirPathname=/export/home/swsiops/SWSI03.1/file_messages/ops/TSWIn
TSWeifDirPathname=/export/home/swsiops/SWSI03.1/file_messages/ancc/TSWIn

!Auto Storage of Active Schedule on Client file directory to find file to
!pass to the client; make sure that the dir props are different in the
! IsolatorProperties file for the closed Isolator vs. the open Isolator

!this directory should match the snif config file value for ActiveSchDirectory
AutoActSchNORMdir=/export/home/swsiops/SWSI03.1/file_messages/ops/ActiveSchedule
AutoActSchEIFdir=/export/home/swsiops/SWSI03.1/file_messages/ancc/ActiveSchedule

!Auto Storage of Active Schedule on Client process...link file directory for NORM
AutoActSchLnNORMdir=/export/home/swsiops/SWSI03.1/file_messages/ops/Isolatortemp.open
!Auto Storage of Active Schedule on Client process...link file directory for EIF
AutoActSchLnEIFdir=/export/home/swsiops/SWSI03.1/file_messages/ancc/Isolatortemp.open
```

```

#Properties for the DbInterface
SWSIdbDriver=oracle.jdbc.driver.OracleDriver
SWSIdbAccount=oracle_username
SWSIdbPassword=oracle_password
!URL for the Normal Operation Database (NORM)
NORMdbUrl=jdbc:oracle:thin:@localhost:1521:OPS
!URL for the test Database (EIF)
EIFdbUrl=jdbc:oracle:thin:@localhost:1521:EIF

#Properties for the Message log

currentActivityLogFile=/export/home/swsiops/SWSI03.1/logs/isolator/open/current
archiveActivityLogFile=/export/home/swsiops/SWSI03.1/logs/isolator/open/archive
currentErrorLogFile=/export/home/swsiops/SWSI03.1/logs/isolator/open/current
archiveErrorLogFile=/export/home/swsiops/SWSI03.1/logs/isolator/open/archive
!max size the file should grow before archiving the file
maxActivityLogSize=1000000
maxErrorLogSize=1000000

!Set the debug mode to true or false
DbDebugOn=true
SnifDebugOn=true
SdifDebugOn=true
ServDebugOn=true
AllDebugOn=true

#Properties for Passwords/Logins
MinPasswordLength=9
MinPasswordRequirements=2
NumLoginsAllowed=3
NumDaysPasswordValid=60
MinPassPhraseLength=10

#
# Process ID
#

ProcessIDFile=/tmp/iso-open.pid
GetPIDCommand=/export/home/swsiops/SWSI03.1/bin/write_pid

```

The meaning of each property is given below:

- **SWSIServerName** – fully qualified domain name or IP address of host running the Application Server.
- **SWSIServerPort** – base TCP port (directive) for connection to the Application Server. The value should match the value of the Application Server *isolatorServerDirectivePort* property.
- **SWSIServerID** – ID of the Application Server (*open* or *closed*) that the Isolator is connecting to.
- **CompressDirSocket** – indicates whether outgoing data on Directive socket is compressed. The value should match the value of the Application Server *CompressDirSocket* property. This property actually has no effect since no data is transmitted on the Directive socket. It is provided only for completeness.

- *CompressMnemonicSocket* - indicates whether outgoing data on Mnemonic socket is compressed. The value should match the value of the Application Server *CompressMnemonicSocket* property.
- *CompressAlertSocket* - indicates whether outgoing data on Alert socket is compressed. The value should match the value of the Application Server *CompressAlertSocket* property.
- *SNIFhostName* - fully qualified domain name or IP address of host running both the operations and test (EIF) SNIFs.
- *SNIFnormPortNumber* – UDP port number to write to for operations SNIF communications. The value should match the value of the operations SNIF *IsolatorReadPort* property.
- *SNIFnormInPortNumber* – UDP port to read from for operations SNIF communications. The value should match the value of either the operations SNIF *Isolator1WritePort* or the *Isolator2WritePort* property.
- *SNIFnormDeltaTime* – number of milliseconds to wait after not receiving a message from the operations SNIF before sending an echo request to operations SNIF to determine if Isolator/operations SNIF communications is still active.
- *SNIFnormReconnectAttempts* – number of times to send an echo request to operations SNIF before declaring that Isolator/operations SNIF communications is inactive.
- *SNIFeifPortNumber* - UDP port number to write to for EIF SNIF communications. The value should match the value of the EIF SNIF *IsolatorReadPort* property.
- *SNIFeifInPortNumber* - UDP port number to read from for EIF SNIF communications. The value should match the value of either the EIF SNIF *Isolator1WritePort* or the *Isolator2WritePort* property.
- *SNIFeifDeltaTime* - number of milliseconds to wait after not receiving a message from the EIF SNIF before sending an echo request to EIF SNIF to determine if Isolator/EIF SNIF communications is still active.
- *SNIFeifReconnectAttempts* - number of times to send an echo request to EIF SNIF before declaring that Isolator/EIF SNIF communications is inactive.
- *SDIFhostName* – fully qualified domain name or IP address of host running both the operations and EIF SDIFs.
- *SDIFNormportNumber* – TCP port number to write to for operations SDIF communications. The value should match the value of the operations SDIF *isolatorSocketPort* property.
- *SDIFEifportNumber* – TCP port number to write to for EIF SDIF communications. The value should match the value of the EIF SDIF *isolatorSocketPort* property.

- TSWnormDirPathname – directory in which to store TDRSS Scheduling Window (TSW) files for later retrieval by the operations SNIF. The value should match the value of the operations SNIF *TSWInputDirectory* property.
- TSWeifDirPathname - directory in which to store TSW files for later retrieval by the EIF SNIF. The value should match the value of the EIF SNIF *TSWInputDirectory* property.
- AutoActSchNORMdir – directory from which to read Active Schedule files stored by the operations SNIF. The value should match the value of the operations SNIF *ActiveSchDirectory* property.
- AutoActSchEIFdir - directory from which to read Active Schedule files stored by the EIF SNIF. The value should match the value of the EIF SNIF *ActiveSchDirectory* property.
- AutoActSchLnNORMdir – temporary directory used for processing Active Schedule files received from the operations SNIF.
- AutoActSchLnEIFdir - temporary directory used for processing Active Schedule files received from the EIF SNIF.
- SWSIdbDriver – location of Oracle JDBC driver.
- SWSIdbAccount – Oracle username.
- SWSIdbPassword – Oracle password.
- NORMdbUrl – location and name of the Oracle database instance to be used for operations support. The format is of the form `jdbc:oracle:thin:@<hostname>:<port>:<instance_name>`.
- EIFdbUrl - location and name of the Oracle database instance to be used for EIF support. The format is of the form `jdbc:oracle:thin:@<hostname>:<port>:<instance_name>`.
- currentActivityLogFile - path to the directory for current (active) activity logs.
- archiveActivityLogFile – path to the directory to which previous activity logs are moved.
- currentErrorLogFile - path to the directory for current (active) error logs.
- archiveErrorLogFile - path to the directory to which previous error logs are moved.
- maxActivityLogSize - maximum activity log file size in bytes. When the log file reaches this size, the current log file is closed and a new log file is started.
- maxErrorLogSize - maximum error log file size in bytes. When the log file reaches this size, the current log file is closed and a new log file is started.
- DbDebugOn - defines whether database debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems.
- SnifDebugOn - defines whether SNIF debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems.

- SdifDebugOn - defines whether SDIF debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems.

- ServDebugOn - defines whether Application Server debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems.
- AllDebugOn - defines whether all debug output is generated, regardless of the setting of other debug properties. This is normally only useful to a developer when trying to troubleshoot problems.
- MinPasswordLength – obsolete.
- MinPasswordRequirements – obsolete.
- NumLoginsAllowed - number of failed login attempts allowed before the user ID is deactivated.
- NumDaysPasswordValid - number of days after which the password associated with a user ID needs to be changed.
- MinPassPhraseLength – obsolete.
- ProcessIDFile – file to contain the Unix ID of the Isolator process.
- GetPIDCommand – command to run to determine the Isolator Unix process ID.

3.4 SWSI-NCCDS Interface

3.4.1 SNIF Invocation and Usage

SNIF is normally executed under HA control using preconfigured scripts and property files, but may be invoked interactively as follows:

```
snif -c <config_filename>
```

Where *config_filename* is the name of the configuration or property file.

If invoked with a *-v* option as follows:

```
snif -v
```

Then the SNIF just displays the version number and exits, similar to the following:

```
SNIF Version: Build 4 Patch 02
```

If invoked with an invalid option, such as follows:

```
snif -h
```

Then the SNIF displays usage information and exits, similar to the following:

```
Error starting SNIF, invalid parameter: -h
Usage: snif -v -d -c <config_filename> [-p <PID_filename>]
    d = debug mode
    v = display version
    p = process ID filename (opt)
```

3.4.2 SNIF Configuration

In general, the configuration file may be changed with any text editor. White space (any combination of spaces and tabs) separates the property name from its assigned value. A “#” character at the beginning of a line is used to enter a comment. Following is a sample EIF SNIF file:

```
#
# SNIF Configuration File
#
# File format:
#
#   keyword <tab> value
#
#
# No white space allowed in System ID
#
NCCDSSystemID           ANCC

#
# Single character NCCDS ID Code used in messages exchanged with
# Isolator. N = Ops NCC, E = EIF
#
NCCDSIDCode             E

NCCDSDomain             ncc.ops.nascom.nasa.gov

#
# Service names, to allow for entry of EIF services.
# These entries are not mandatory. Default is the standard
# ops service names.
#
schReqName              EschReq
schStatusName           EschStatus
pmDataName              EpmData
reconfigName            Ereconfig
acqStoreName            EacqStore
tswStoreName            EtswStore

DatabaseName           EIF

OracleUsername          oracle_username
OraclePassword          oracle_password

IsolatorReadHost       localhost
IsolatorReadPort       2030

Isolator1WriteHost     localhost
Isolator1WritePort     2022

Isolator2WriteHost     localhost
Isolator2WritePort     2032

#
# File Storage Locations
#
RCTDMDirectory          /export/home/swsiops/SWSI03.1/file_messages/ancc/RCTDM
TTMDirectory           /export/home/swsiops/SWSI03.1/file_messages/ancc/TTM
TSWInputDirectory      /export/home/swsiops/SWSI03.1/file_messages/ancc/TSWIn
TSWArchiveDirectory    /export/home/swsiops/SWSI03.1/file_messages/ancc/TSWArc
```

```

ActiveSchDirectory      /export/home/swsiops/SWSI03.1/file_messages/ancc/ActiveSchedule
TempDirectory          /export/home/swsiops/SWSI03.1/file_messages/ancc/SNIFtemp

#
# Logging parameters
#

MaxLogFileSize         1000000
LogFileDirectory       /export/home/swsiops/SWSI03.1/logs/snif/ancc/current
LogFileArcDirectory    /export/home/swsiops/SWSI03.1/logs/snif/ancc/archive

```

The meaning of each property is given below:

- NCCDSSystemID – name of NCCDS system with which SNIF is communicating. Usually set to *NCCDS* or *ANCC*.
- NCCDSIDCode – single letter code used in communication between SNIF and Isolator. Should be set to either N for operations SNIF or E for EIF SNIF.
- NCCDSDomain – NCCDS domain name as specified in Section 4.3.1 of the *NCCDS/MOC ICD*.
- schReqName – service name for Schedule Request service. The default value is schReq.
- schStatusName - service name for Schedule Status service. The default value is schStatus.
- pmDataName - service name for Performance Data service. The default value is pmData.
- reconfigName - service name for Reconfiguration service. The default value is reconfig.
- acqStoreName - service name for Acquisition Data Storage service. The default value is acqStore.
- tswStoreName - service name for TSW Storage service. The default value is tswStore.
- DatabaseName – name of Oracle database instance.
- OracleUsername - Oracle username.
- OraclePassword - Oracle password.
- IsolatorReadHost – fully qualified domain name or IP address to listen on for receiving UDP messages from both the open and closed isolators.
- IsolatorReadPort – UDP port to read from for both open and closed Isolator communications. The value should match the value of either the SNIFnormPortNumber or SNIFeifPortNumber for both Isolators.
- Isolator1WriteHost – fully qualified domain name or IP address of host running Isolator #1.

- Isolator1WritePort – UDP port number to write to for Isolator #1 communications. The value should match the value of either the SNIFnormInPortNumber or SNIFeifInPortNumber of Isolator #1.
- Isolator2WriteHost - fully qualified domain name or IP address of host running Isolator #1.
- Isolator2WritePort – UDP port number to write to for Isolator #2 communications. The value should match the value of either the SNIFnormInPortNumber or SNIFeifInPortNumber of Isolator #2.
- RCTDMDirectory – directory in which to store Return Channel Time Delay Messages (RCTDMs) for later retrieval by both Isolators.
- TTMDirectory - directory in which to store Time Transfer Messages (TTMs) for later retrieval by both Isolators.
- TSWInputDirectory – directory from which to read TSW files stored by both Isolators. The value should match the value of either the TSWnormDirPathname or TSWeifDirPathname property of both Isolators.
- TSWArchiveDirectory – directory to which TSW files should be moved after they have been successful transmitted to NCCDS.
- ActiveSchDirectory – directory in which to store Active Schedule files for later retrieval by both Isolators. The value should match the value of the TSWnormDirPathname or TSWeifDirPathname property of both Isolators.
- TempDirectory – temporary directory for processing Active Schedule files being prepared for transmission to both Isolators.
- MaxLogFileSize - maximum log file size in bytes. When the log file reaches this size, the current log file is closed and a new log file is started.
- LogFileDirectory - path to the directory for current logs.
- LogFileArcDirectory - path to the directory to which previous error logs are moved.

3.5 SWSI-DAS Interface

3.5.1 SDIF Invocation and Usage

The SDIF is normally executed under HA control using preconfigured scripts and property files, but may be invoked interactively as follows:

```
java -Duser.timezone=GMT -DSWSIROOT=$SWSIROOT -classpath SWSI-OPS-cots.jar:SWSIsdif.jar \
gov.nasa.gsfc.swsi.sdif.SDIFMain <propfile>
```

Where *propfile* is the name of the property file.

If invoked with a -v option as follows:

```
java -Duser.timezone=GMT -DSWSIROOT=$SWSIROOT -classpath $CLP \
gov.nasa.gsfc.swsi.sdif.SDIFMain -v
```

Then the SDIF just displays the version number and exits, similar to the following:

```
SWSI SDIF version Build 5 Patch 05 12/31/2003
```

3.5.2 SDIF Configuration

The property file name passed as an argument in the invocation line is the SDIF configuration file. Following is a sample file:

```
#
# SWSI-DAS Interface (SDIF) Property File
#
# This file contains the properties used for proper operation of SDIF
# subsystem.
#
# NOTE: The property file may be changed with any text editor. Properties
# may be provided in any order. A ":" character separates the
# property name from its assigned value. An "#" or "!" character at
# the beginning of a line is used to comment an entire line. Boolean
# properties can have a value of true or false.
#
# All properties are static.
# SDIF needs to be restarted in order for any changes to take effect.
#####
##### Database connectivity related properties #####
#####
# EIFMode -
# Specifies the operational mode SDIF is running: EIF mode or OPS mode.
# A false value denotes OPS mode. A true value denotes EIF mode.
# Values: true | false
# Default: false
EIFMode : false
# dbConnectionString -
# JDBC Connection string specifying which instance of the database to use.
# Default: jdbc:oracl:thin:@localhost:1521:ops
dbConnectionString : jdbc:oracl:thin:@localhost:1521:ops
# dbLogin -
# Database Username.
dbLogin : database_username
# dbPassword -
# Database password.
dbPassword : database_password
##### End of Database connectivity related properties #####
#####
##### Network connectivity properties #####
#####
```

```

# isolatorSocketPort -
# Specifies the server port that SDIF listens for Isolator connections.
# Default: 3140

isolatorSocketPort : 2141

# dasHost -
# Specifies the hostname or IP address of DASCON
# Default: localhost

dasHost : 123.123.123.123

# dasSocketPort -
# Specifies the TCP port for communicating with DASCON
# Default: 2100

dasSocketPort : 5999

#####          End of network connectivity related properties          #####

#####
#####          Logging related properties          #####
#####

# message log -
# Controls logging message information to the log file.
# Values: true | false
# Default: true

messageLog : true

# logFileName -
# Name of log file (including path if any)
# This property is only used when the value of the messageLog property is true.
# Default: $SWSIROOT/logs/sdif/sdifmessage.log

messageLogFileName                                     :
/export/home/swsiops/current_release/logs/sdif/ops/message/sdifmessage.log

# data log -
# Controls logging data information to the log file.
# Values: true | false
# Default: true

dataLog : true

# logFileName -
# Name of log file (including path if any)
# This property is only used when the value of the dataLog property is true.
# Default: $SWSIROOT/logs/sdif/sdifdata.log

dataLogFileName : /export/home/swsiops/current_release/logs/sdif/ops/data/sdifdata.log

# console -
# Controls logging information to console (stdout).
# Values: true | false
# Default: true

console : true

# telnet -
# Controls whether to log information to a telnet port. If value is true.
# telnet server is created on the socket specified by telnetLoggerPort property.
# Whenever, a telnet client connects to that port on the host where SDIF is running,
# Logging information is printed to stdout of that client.
# Values: true | false
# Default: false

telnet : true

```

```

# telnetLoggerPort -
# Specifies the TCP port for telnet server to listen for connections.
# This property is only used when the value of the telnet property is true.
# Default: 23

telnetLoggerPort : 2399

#####          End of Logging related properties          #####

#####
#####          XML-based processing related properties          #####
#####

# swsiDASXMLSchemaLocation -
# Location of XML Schema. The property can be a local path or a URL.
# Default: $SWSIROOT/xmlschema/swsi_das.xsd

swsiDASXMLSchemaLocation : /export/home/swsiops/current_release/xmlschema/swsi_das.xsd

# validateXML -
# Controls whether XML validation is enabled.
# Values: true | false
# Default: false

validateXML : true

# validateXMLWithSchema -
# Controls whether XML validation is done via XML schema or DTD.
# This property is only used when validateXML property is true.
# Values: true for schema, false for DTD
# Values: true | false
# Default: true

validateXMLWithSchema : true

# xmlTemplatesDir -
# Path to the root directory of xml templates directories.
# Default: $SWSIROOT/xmltemplates

xmlTemplatesDir : /export/home/swsiops/current_release/xmltemplates

# defaultXMLSchemaInstanceNamespaceURI -
# Specifies default URI to W3C XML schema..
# Default: http://www.w3.org/2000/10/XMLSchema-instance

defaultXMLSchemaInstanceNamespaceURI : http://www.w3.org/2000/10/XMLSchema-instance

# defaultXMLNamespacePrefix -
# Specifies the default XML namespace to use.
# Default: xsi

defaultXMLNamespacePrefix : xsi

#####          End of XML-based processing related properties          #####

#####
#####          Miscellaneous properties          #####
#####

# ProcessIDFile -
# Specifies the path to the SDIF Process ID file.
# Default: /tmp/SDIF.pid

ProcessIDFile : /tmp/ops-sdif.pid

# GetPIDCommand -
# Specifies the path to the Process ID producing native executable.
# Default: write_pid

GetPIDCommand : /export/home/swsiops/current_release/bin/write_pid

```

The meaning of each property is given below:

- EIFMode - specifies the operational mode SDIF is running: EIF mode or OPS mode. A false value denotes OPS mode. A true value denotes EIF mode.
- dbConnectionString - JDBC Connection string specifying the location and name of the Oracle database instance to use. The format is of the form jdbc:oracle:thin:@<hostname>:<port>:<instance_name>.
- dbLogin – Database username.
- dbPassword – Database password.
- isolatorSocketPort - specifies the server port that SDIF listens for Isolator connections. This value should match the value of either the SDIFNormportNumber or the SDIFEifportNumber for both of the Isolators.
- dasHost – fully qualified hostname or IP address of DASCON.
- dasSocketPort – specifies the TCP port for communicating with DASCON.
- messageLog – controls logging message information to the message log file.
- messageLogFileName – name of the message log file (including the path if any).
- dataLog – controls logging data information to the data log file.
- dataLogFileName – name of the data log file (including the path if any).
- console – controls logging information to the console (stdout).
- telnet - controls whether to log information to a telnet port. If value is true, a telnet server is created on the socket specified by the telnetLoggerPort property. Whenever, a telnet client connects to that port on the host where SDIF is running, logging information is printed to stdout of that client.
- telnetLoggerPort - specifies the TCP port for the telnet server to listen for connections. This property is only used when the value of the telnet property is true.
- swsiDASXMLSchemaLocation – specifies the location of the XML Schema. The property can be a local path or a URL.
- validateXML – controls whether XML validation is enabled.
- validateXMLWithSchema - controls whether XML validation is done via XML schema or DTD. This property is only used when validateXML property is true. Values: true for schema validation, false for DTD validation.
- xmlTemplatesDir –path to the root directory of xml templates directories.
- defaultXMLSchemaInstanceNamespaceURI - specifies the default URI to W3C XML schema. The default value is <http://www.w3.org/2000/10/XMLSchema-instance>.

- defaultXMLNamespacePrefix - specifies the default XML namespace to use. The default value is xsi.
- ProcessIDFile – specifies the path and file to contain the Unix ID of the SDIF process.
- GetPIDCommand – command to run to determine the SDIF Unix process ID.

3.6 TUT Proxy Sender

3.6.1 TUT Proxy Sender Invocation and Usage

The TUT Proxy Sender is normally run under *cron* control using preconfigured scripts and property files, but may be invoked interactively as follows:

```
SendTut.csh <propfile>
```

or individually as follows:

```
java -classpath SWSItut_proxy.jar URLGrab http://<tut server host>/data/newtut.dat \  
| grep -v "^2 " > newtut.dat  
java -classpath SWSItut_proxy.jar SendTut -prop <propfile> -infile newtut.dat.uue
```

where *propfile* is the name of the property file. The TUT Proxy Sender will not run without a reference to a property file.

If invoked with a *-v* option as follows:

```
java -classpath SWSItut_proxy.jar SendTut -v
```

Then the TUT Proxy Sender just displays the version number and exits, similar to the following:

```
TUT Proxy Sender: Build4 Patch 02 04/21/2003
```

If invoked with an invalid option or with the *-h* option, something similar to the following will be displayed:

```
TUT Proxy Sender: Build 4 Patch 02 04/21/2003  
Usage: java SendTut [OPTIONS] ...  
  
OPTIONS:  
-h, -help           Show this help  
-v, version         Displays the version  
-D, -debug          Enables Debug Output  
-host <HOST>       Server Host to connect  
-port <PORT>       Server Port to connect  
-infile <INPUT-FILE> Input file to send  
-prop <PROPERTY-FILE> Properties file to read  
-mode OPS,EIF,CERT Valid Operational Modes  
                    are defined by the Server host  
  
Examples:  
java SendTut -DEBUG -prop tut.prop.ops  
java SendTut -prop tut.prop.ops -infile newtut.dat  
java SendTut -prop tut.prop.ops -infile newtut.dat -mode OPS
```

Note: Command line arguments override the properties file.

3.6.2 TUT Proxy Sender Properties

The property file name passed as an argument in the invocation line is the TUT Proxy Sender configuration file. Following is a sample file:

```
! TUT Sender and Receiver Properties

DebugMode      : false
MultiRequestOn : false
ServerHost     : swsi-lil.nascom.nasa.gov
ServerPort     : 1234
ServerSyncMode : OPS,EIF,CERT
SyncModeOn     : true
ShowTransferStatus : false

HomeDir        : /export/home/swsiops
BuildDir       : SWSI03.1
CertDir        : /export/home/swsiops/SWSI03.1/locker
CertStorageDir : /export/home/swsiops/SWSI03.1/locker/storage
CertTrashDir   : /export/home/swsiops/SWSI03.1/locker/trash

DeleteCertAfterTx: true

! Transfer Certificate after Waiting N minutes
XferCertWaitTime: 30

! Compress/UUencode UUdecode/Decompress script
CompUUCodecCmd : /export/home/swsiops/SWSI03.1/bin/comp_uucodec

LogTag         : OPS
LogDir         : logs/tut
TempTutDir     : temp
CurrentLogDir  : current
ArchiveLogDir  : archive
LogFile        : tut-xfer.log

TUTWebDir      : WWW/tut
TUTDataDir     : data
TUTDataFile    : newtut.dat

TUTServerHost  : tuth
TUTServerDomain: ncc.ops.nascom.nasa.gov

ProcessIDFile  : /tmp/tuttx_ops-lil.pid
GetPIDCommand  : /export/home/swsiops/SWSI03.1/bin/write_pid
```

The meaning of each property is given below:

- **DebugMode** - defines whether debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **MultiRequestOn** - not used for TUT Proxy Sender.
- **ServerHost** - fully qualified domain name or IP address of the host running the TUT Proxy Receiver.
- **ServerPort** - TCP port for connecting to the TUT Proxy Receiver.
- **ServerSyncMode** - not used for TUT Proxy Sender.

- SyncModeOn - defines whether to use synchronization flags when communicating with the TUT Proxy Receiver. The default value is *true*.
- ShowTransferStatus - indicates whether to show a progress status when transferring data to the TUT Proxy Receiver. This is useful only when running the TUT Proxy Sender interactively and not as a background process. The default value is *false*.
- HomeDir - defines the base directory path where data files transferred will be stored.
- BuildDir - defines the top-level relative directory name for the current build of SWSI. This directory is relative to the *HomeDir* property setting. This property must be set in order for proper retrieval to occur.
- CertDir - defines the full directory path for storing transferred user certificates.
- CertStorageDir - defines the full directory path for storing user certificates before they are transferred.
- CertTrashDir - defines the full directory path for storing unneeded user certificates. Certificates in this directory will be deleted if *DeleteCertAfterTx* is set to *true*.
- DeleteCertAfterTx - not used for TUT Proxy Sender.
- XferCertWaitTime - not used for TUT Proxy Sender.
- CompUUCodecCmd - defines the full directory path to the Compress/Uuencode UUdecode/Decompress command.
- LogTag - defines which mode the TUT Proxy Sender is operating in. Valid values are OPS, EIF, or CERT. This option also defines the file name extension of the log files and TUT data files. This property ensures that the operational and EIF TUT data as well the certificate data are not clobbered during transmission.
- LogDir - defines the relative directory path to the log directory. This directory is relative to the *BuildDir* property setting.
- TempTutDir - defines the relative directory path to the temp directory for storing the TUT data file. This directory is relative to the *LogDir* property setting.
- CurrentLogDir - defines the relative directory path for the current log file. This directory is relative to the *LogDir* property setting.
- ArchiveLogDir - defines the relative directory path for the archived log file. This directory is relative to the *LogDir* property setting.
- TUTWebDir - defines the relative parent directory path for storing the TUT data file. This directory is relative to the *HomeDir* property setting. This directory is also a web server directory location.

- TUTDataDir - defines the relative directory path for storing the TUT data file. This directory is relative to the *HomeDir* property setting. This directory is also a web server directory location.
- TUTDataFile - defines the name of the TUT data file that is retrieved from the NCCDS TUT server. This file does not change.
- TUTServerHost - base hostname for the NCCDS TUT server. The TUT Proxy Sender will append 1-9 to the end of this property to complete the hostname. When *LogTag* is set to CERT, this property is ignored.
- TUTServerDomain - NCCDS domain name. This is appended to the *TUTServerHost* property. When *LogTag* is set to CERT, this property is ignored.
- ProcessIDFile - file that contains the UNIX process ID of the TUT Proxy Sender.
- GetPIDCommand - command to run to determine the TUT Proxy Sender process ID.

3.7 TUT Proxy Receiver

3.7.1 TUT Proxy Receiver Invocation and Usage

The TUT Proxy Receiver is run at boot time by *init* using preconfigured scripts and property files, but may be invoked interactively as follows:

```
ReceiveTut.csh <propfile>
```

or individually as follows:

```
java -classpath SWSItut_proxy.jar ReceiveTut -MULTI -prop <propfile>
```

where *propfile* is the name of the property file. The TUT Proxy Receiver will not run without a reference to a property file.

If invoked with a *-v* option as follows:

```
java -classpath SWSItut_proxy.jar ReceiveTut -v
```

Then the TUT Proxy Sender just displays the version number and exits, similar to the following:

```
TUT Proxy Receiver: Build4 Patch 02 04/21/2003
```

If invoked with an invalid option or with the *-h* option, something similar to the following will be displayed:

```
TUT Proxy Receiver: Build 4 Patch 02 04/21/2003
Usage: java ReceiveTut [OPTIONS] ...

OPTIONS:
  -h, -help          Show this help
  -v, -version       Displays the version
  -D, -debug         Enables Debug Output
  -host <HOST>      Server Host to connect
```

```

-Multi                Enable Handle Multiple Requests
-outfile <OUTPUT-FILE> Output file to write
-prop <PROPERTY-FILE> Properties file to read
-mode OPS,EIF,CERT    Valid Operational Modes
                     are defined server properties file.

```

Examples:

```

java ReceiveTut -DEBUG -prop tut.prop
java ReceiveTut -prop tut.prop.ops -outfile newtut.dat
java ReceiveTut -MULTI -prop tut.prop.ops -mode OPS

```

Note: Command line arguments override the properties file.

3.7.2 TUT Proxy Receiver Properties

The property file name passed as an argument in the invocation line is the TUT Proxy Receiver configuration file. Following is a sample file:

```

! TUT Sender and Receiver Properties

DebugMode      : false
MultiRequestOn : true
ServerHost     : swsi-lil.nascom.nasa.gov
ServerPort     : 5213
ServerSyncMode : OPS,EIF,CERT
SyncModeOn     : true
ShowTransferStatus : false

HomeDir        : /export/home/swsiops
BuildDir       : SWSI03.1
CertDir        : /export/home/swsiops/SWSI03.1/locker
CertStorageDir : /export/home/swsiops/SWSI03.1/locker/storage
CertTrashDir   : /export/home/swsiops/SWSI03.1/locker/trash

DeleteCertAfterTx: true

! Transfer Certificate after Waiting N minutes
XferCertWaitTime: 30

! Compress/UUencode UUdecode/Decompress script
CompUUCodecCmd : /export/home/swsiops/SWSI03.1/bin/comp_uucodec

LogTag         : OPS
LogDir         : logs/tut
TempTutDir     : temp
CurrentLogDir  : current
ArchiveLogDir  : archive
LogFile        : tut-xfer.log

TUTWebDir      : SWSI03.1/WWW/tut
TUTDataDir     : data
TUTDataFile    : newtut.dat

TUTServerHost  : tuth
TUTServerDomain: ncc.ops.nascom.nasa.gov

!
! Process ID
!

ProcessIDFile  : /tmp/tutrx.pid
GetPIDCommand  : /export/home/swsiops/SWSI03.1/bin/write_pid

```

The meaning of each property is given below:

- **DebugMode** - defines whether debug output is generated. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **MultiRequestOn** - defines whether the TUT Proxy Receiver accepts a single request and exits or multiple requests continually (server mode). This option should be *true* unless debugging. The default value is *false*.
- **ServerHost** - fully qualified domain name or IP address of the host running the TUT Proxy Receiver.
- **ServerPort** - TCP port on which the TUT Proxy Receiver listens for connections.
- **ServerSyncMode** - valid synchronization modes for which the TUT Proxy Receiver operates. This option is usually set to "OPS,EIF,CERT". More modes can be added to the list.
- **SyncModeOn** - defines whether to use synchronization flags when communicating with the TUT Proxy Sender. The default value is *true*.
- **ShowTransferStatus** - indicates whether to show a progress status when receiving data from the TUT Proxy Sender. This is useful only when running the TUT Proxy Receiver interactively and not as a background process. The default value is *false*.
- **HomeDir** - defines the base directory path where data files received will be stored.
- **BuildDir** - defines the top-level relative directory name for the current build of SWSI. This directory is relative to the *HomeDir* property setting. This property must be set in order for proper retrieval to occur.
- **CertDir** - defines the full directory path for storing transferred user certificates.
- **CertStorageDir** - defines the full directory path for storing user certificates before they are transferred.
- **CertTrashDir** - defines the full directory path for storing unneeded user certificates. Certificates in this directory will be deleted if *DeleteCertAfterTx* is set to *true*.
- **DeleteCertAfterTx** - indicates whether to delete user certificates after they are transferred. The default value is *true*.
- **XferCertWaitTime** - number of minutes to wait before transferring a certificate to the backend server. This property also defines how long after a user certificate is created it will remain available for download. The default time is 30 minutes.
- **CompUUCodecCmd** - defines the full directory path to the Compress/Uuencode UUdecode/Decompress command.

- LogTag – not used for TUT Proxy Receiver.
- LogDir - defines the relative directory path to the log directory. This directory is relative to the *BuildDir* property setting.
- TempTutDir - defines the relative directory path to the temp directory for storing the TUT data file. This directory is relative to the *LogDir* property setting.
- CurrentLogDir - defines the relative directory path for the current log file. This directory is relative to the *LogDir* property setting.
- ArchiveLogDir - defines the relative directory path for the archived log file. This directory is relative to the *LogDir* property setting.
- TUTWebDir - defines the relative parent directory path for storing the TUT data file. This directory is relative to the *HomeDir* property setting. This directory is also a web server directory location.
- TUTDataDir - defines the relative directory path for storing the TUT data file. This directory is relative to the *HomeDir* property setting. This directory is also a web server directory location.
- TUTDataFile - defines the name of the TUT data file that is retrieved from the NCCDS TUT server. This file does not change.
- TUTServerHost - not used for TUT Proxy Receiver.
- TUTServerDomain - not used for TUT Proxy Receiver.
- ProcessIDFile - file that contains the UNIX process ID of the TUT Proxy Receiver.
- GetPIDCommand - command to run to determine the TUT Proxy Receiver process ID.

3.8 Certification Generation Tool

3.8.1 Certificate Generation Tool Invocation and Usage

The Certificate Generation Tool invocation method depends on whether it is used to generate a Client user certificate, server certificate, or Certificate Authority (CA). Specific invocation methods are shown in the following subsections. If invoked with a *-v* option as follows:

```
certtool -v
```

or:

```
java -classpath certtool.jar CertTool -v
```

Then the Certificate Generation Tool just displays the version number and exits, similar to the following:

```
Build4 Patch 02 04/21/2003
```

If invoked with the -h option, such as follows:

```
certtool -h
```

Then certtool displays usage information and exits, similar to the following:

```
Build4 Patch 02 04/21/2003
Usage: java CertTool [OPTIONS] [property file]...

OPTIONS:
  -h, -help           Show this help
  -v, -version        Displays the version
  -D, -debug          Enables Debug Output
  -genca, -ca         Generate CA Certificate and Private key
  -pass              Disables Passphrase quality check
  -nospace            Disables Counting spaces in passphrase length
  -prop <PROPERTY-FILE> Properties file to read
  -rpp <PROPERTY-FILE> Read Passphrase from properties file

Examples:
  java CertTool -prop JohnQDoe.prop
  java CertTool -rpp JohnQDoe.prop
  java CertTool -ca CACert.prop
  java CertTool JohnQDoe.prop
```

3.8.2 Certificate Generation Tool CGI Configuration Parameters

When generating Client user certificates, the Certificate Generation Tool is normally launched as a Common Gateway Interface (CGI) script from a HyperText Markup Language (HTML) form on the SWSI Server. An additional configuration file called *certtool-cgi.conf* provides properties that are used by *certtool* in this process. Following is a sample *certtool-cgi.conf* file:

```
# Certificate Generation Tool CGI Configuration File

DebugALL : false
DebugCert : false
DebugProp : false

SystemVersion : Build 4 Patch 01
JREVersion : v 1.4.1
JREURL : http://java.sun.com/j2se/1.4.1/download.html
JarProg : /export/home/cots/j2re1.4.1_02/bin/jar
SWSI_URL : http://swsi.gsfc.nasa.gov/
MSP_URL : http://msp.gsfc.nasa.gov/

# Storage Directory for created certs and keys
LockerDir : /export/home/swsiops/SWSI03.1/locker

# Property File Directory for On-line generated certs
PropertyFileDir : /export/home/swsiops/SWSI03.1/locker/property

# Days before certificate expires
CertificateExpiration : 366

# Certificate Authority Directory
CertificateAuthorityDir : /export/home/swsiops/SWSI03.1/certs/CA

CertificateAuthorityFile : SWSI-ca-cert.der.10yr
CertificateAuthorityPrivateKeyFile : SWSI-ca-key.der.10yr
```

```

# Certificate Serial Number
CertSerialNumber      : 1001

# Certificate Strength in bits
Strength              : 1024

# Certtool Script paths and options, add -pass to skip passphrase quality check
#
# -nospace disables space counting
CertToolPath          : /export/home/swsiops/SWSI03.1/bin/certtool
CertToolOptions       : -rpp

# HTML Docs
HTMLDocsDir           : /download

# Relative URL to Download Generated Certificates
CertDownloadPath      : /download/certs/tmp

# Check the Passphrase Quality, -D[1-3] turns on debug, -nospace disables space counting
QualityCheckPath      : /export/home/swsiops/SWSI03.1/bin/checkPassQuality

# Relative URL to for cgi-bin location
CGI_DIR                : /certs

# Passphrase Quality Limits
#
# Minimum passphrase length
TotalCharMin           : 20

# Minimum Number of Unique Characters
TotalUniqueMin         : 6

# Minimum number of words
TotalWordMin           : 4

# Minimum Number of Character Groups that Must Be Satisfied
TotalCharGroupMin     : 3

# At Least 1 Character from each Character Group
IndividualCharMin     : 1

```

The meaning of each property is given below:

- **DebugALL** - defines whether debug output is generated for all certtool operations. This is normally only useful to a developer when trying to troubleshoot problems. A value of *true* has the effect of setting *DebugCert* and *DebugProp* to be *true*. The default value is *false*.
- **DebugCert** - defines whether debug output is generated for certificate generation operations. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **DebugProp** - defines whether debug output is generated for property reading operations. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **SystemVersion** - specifies the SWSI delivery version displayed on the SWSI software download website. This value is updated with each new patch or version release of SWSI.

- JREVersion - specifies the Java Runtime Environment (JRE) version required to run the SWSI Client.
- JREURL - specifies the URL to download the JRE direct from Sun Microsystems. This value may change if current link becomes stale or outdated.
- JarProg - full path to the Java Archiver (jar) command.
- SWSI_URL - specifies the URL to the SWSI Project website.
- MSP_URL - specifies the URL to the Mission Services Program (MSP) Office website. This is the management organization for the SWSI website.
- LockerDir - specifies the full path to the storage directory for user-generated certificates.
- PropertyFileDir - specifies the full path to the property directory where the web-generated property files are stored.
- CertificateExpiration - specifies how many days the certificate is valid. This value should be no more than 366 for SWSI Client users. For SWSI Server processes, it should be on the order of years.
- CertificateAuthorityDir - full path to the directory that contains the CA certificates.
- CertificateAuthorityFile - relative path to the public CA certificate file. This is used to sign user-generated certificates. It is one half of the private/public key pair for the CA. This should not change. Currently, it is scheduled for a pre-determined cycle. A different cycle can be set. However, changing the CA file will result in all SWSI Client users having to generate new certificates.
- CertificateAuthorityPrivateKeyFile - relative path to the private CA file. This is used to sign user-generated certificates. It is one half of the private/public key pair for the CA. This should not change. Currently, it is scheduled for a pre-determined cycle. A different cycle can be set. However, changing the CA file will result in all SWSI Client users having to generate new certificates.
- CertSerialNumber - serial number assigned to each certificate created.
- Strength - size of the generated private/public key pair in bits. Valid values are 512, 1024, or 2048. The default value is 1024.
- CertToolPath - full path to the certtool command.
- CertToolOptions - command line options to be included when calling the certtool from CGI. See Section 3.8.1 for details on valid options.
- HTMLDocsDir - specifies the top level URL path to download the SWSI software and certificates. This value should correspond to a valid alias directory defined in the webserver configuration file.

- CertDownloadPath - specifies the full URL path to download the SWSI software and certificates.
- QualityCheckPath - specifies full path to the checkPassQuality command. The following checkPassQuality options can be included after this command if needed:

```
-D[1-3], -nospace
```

For more options, checkPassQuality can be invoked with the -h option:

```
checkPassQuality -h
```

```
Usage: java PassphraseQuality [OPTIONS] [passphrase]
```

```
OPTIONS:
```

```
-h, -help           Show this help
-D, -D[1-3]        Enables Debug Output Level 1 to 3
-verbose           Enables Debug Output Level 1
-prompt            Prompts to enter passphrase
-nospace           Disables Counting spaces in passphrase length
-W                Minimum Number of Words
-C                Minimum Number of Characters
-U                Minimum Number of Unique Characters
-G                Minimum Match from Character Groups
-I                Minimum Individual Characters from each Group
```

```
Examples:
```

```
java PassphraseQuality -W 4 -C 20 -U 8 -G 3 -I 1 "The quick brown fox."
java PassphraseQuality -nospace "The quick brown fox."
java PassphraseQuality -prompt
```

- CGI_DIR - specifies the top level URL path for the CGI scripts. This value should also correspond to a valid alias directory defined in the webserver configuration file.
- PassphraseMinLen - defines the minimum length the user's passphrase must be, not including spaces. The default value is 20.
- TotalUniqueMin - defines the minimum number of unique character that must exist in a user's passphrase. The default value is 6.
- TotalWordMin - defines the minimum number of words a user's passphrase must contain. A word is defined to be any contiguous group of characters separated by spaces or a carriage return. The default value is 4.
- TotalCharGroupMin - defines the minimum number of character groups the user's passphrase must satisfy. The character groups are as follows: Uppercase Characters, Lowercase Characters, Numeric Characters, and Punctuation Characters. The default value is 3.
- IndividualCharMin - defines the minimum number of characters from each character group a user's passphrase must satisfy. The default value is 1.

3.8.3 Certificate Generation for Client Users

When generating Client user certificates, the Certificate Generation Tool is normally launched from the SWSI Server Website as a CGI script using an automatically generated property file, but may be invoked interactively as follows:

```
certtool <propfile>
```

or:

```
java -classpath certtool.jar CertTool <propfile>
```

where *propfile* is the name of the property file. Following is a sample file:

```
! Certificate Generation Properties
! Automatically Generated! Thu Apr 24 16:43:28 2003

DebugALL : false
DebugCert : false
DebugProp : false

Organization      : NASA
OrganizationUnit  : 451
Locality          : Greenbelt
StateOrProvince   : MD
Country           : US
Common            : John Doe, jdoe, 12345678901
EmailAddress      : john.q.doe@toetag.com

! Expiration given in days
CertificateExpiration: 366
CertSerialNumber   : 1001
Strength           : 1024

CertificateAuthorityFile      : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
cert.der.10yr
CertificateAuthorityPrivateKeyFile : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
key.der.10yr

ClientCertificateRequestFile : /export/home/swsiops/SWSI03.1/locker/nasa-jdoe-csr.der
ClientCertificateFile        : /export/home/swsiops/SWSI03.1/locker/nasa-jdoe-cert.der
ClientPrivateKeyFile         : /export/home/swsiops/SWSI03.1/locker/nasa-jdoe-key.der
ClientEncryptedPrivateKeyFile: /export/home/swsiops/SWSI03.1/locker/enc-nasa-jdoe-key.der

! Passphrase is no longer used.
ClientPassPhrase :

KeyArchiveDir      : /export/home/swsiops/SWSI03.1/locker
KeyArchiveFile     : /export/home/swsiops/SWSI03.1/locker/nasa-jdoe-200304241643-key.zip

JarProg           : /export/home/cots/j2re1.4.0_01/bin/jar

! Passphrase Quality Settings
PassphraseMinLen  : 20
TotalUniqueMin    : 6
TotalWordMin      : 4
TotalCharGroupMin : 3
IndividualCharMin : 1
```

The meaning of each property is given below:

- **DebugALL** - defines whether debug output is generated for all the *certtool* operations. This is normally only useful to a developer when trying to troubleshoot problems. A value of *true* has the effect of setting *DebugCert* and *DebugProp* to be *true*. The default value is *false*.
- **DebugCert** - defines whether debug output is generated for certificate generation operations. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **DebugProp** - defines whether debug output is generated for property reading operations. This is normally only useful to a developer when trying to troubleshoot problems. The default value is *false*.
- **Organization** - specifies the user's top-level organization.
- **OrganizationUnit** - specifies the user's lower level organization unit.
- **Locality** - specifies the city or town where the organization is located.
- **StateOrProvince** - specifies the state or province where the organization is located.
- **Country** - specifies the country where the organization is located.
- **Common** - specifies the SWSI Client user's full name, username, and telephone number.
- **EmailAddress** - specifies the SWSI Client user's email address.
- **CertificateExpiration** - specifies how many days the certificate is valid. This value should be no more than 366 for SWSI Client users. For SWSI Server processes, it should be on the order of years.
- **CertSerialNumber** - serial number assigned to each certificate created.
- **Strength** - size of the generated private/public key pair in bits. Valid values are 512, 1024, or 2048. The default value is 1024.
- **CertificateAuthorityFile** - full path to the public CA certificate file. This is used to sign user-generated certificates. It is one half of the private/public key pair for the CA. This should not change. Currently, it is scheduled for a pre-determined cycle. A different cycle can be set. However, changing the CA file will result in all SWSI Client users having to generate new certificates.
- **CertificateAuthorityPrivateKeyFile** - full path to the private encrypted CA file. This is used to sign user-generated certificates. It is one half of the private/public key pair for the CA. This should not change. Currently, it is scheduled for a pre-determined cycle. A different cycle can be set. However, changing the CA file will result in all SWSI Client users having to generate new certificates.

- ClientCertificateRequestFile - full path to the user's certificate signing requests (CSR). This property will be different for each user.
- ClientCertificateFile - full path to the user's public certificate. This property will be different for each user. It is one half the private/public key pair for the user.
- ClientPrivateKeyFile - full path to the user's unencrypted private certificate. This property will be different for each user. It is one half the private/public key pair for the user.
- ClientEncryptedPrivateKeyFile - full path to the user's encrypted private certificate. This property will be different for each user. It is one half the private/public key pair for the user.
- ClientPassPhrase - specifies the passphrase to encrypt the user's private key specified by *ClientPrivateKeyFile*. Depending on how certtool is invoked, this property may be blank. When certtool is configured to be invoked under CGI, the passphrase is read from this property but deleted once generation is complete. When certtool is invoked from the command line, this property is ignored and you will be prompted to enter the passphrase.
- KeyArchiveDir - specifies the full directory path to store user generated certificates.
- KeyArchiveFile - specifies the full path to the certificate key archive file.
- JarProg - full path to the Java Archiver (jar), a standard command that is available with the Java Development Kit (JDK). *jar* is a standard part of the JRE package for Unix platforms.

The following properties define the passphrase quality:

- PassphraseMinLen - defines the minimum length the user's passphrase must be, not including spaces. The default value is 20.
- TotalUniqueMin - defines the minimum number of unique character that must exist in a user's passphrase. The default value is 6.
- TotalWordMin - defines the minimum number of words a user's passphrase must contain. A word is defined to be any contiguous group of characters separated by spaces or a carriage return. The default value is 4.
- TotalCharGroupMin - defines the minimum number of character groups the user's passphrase must satisfy. The character groups are as follows: Uppercase Characters, Lowercase Characters, Numeric Characters, and Punctuation Characters. The default value is 3.
- IndividualCharMin - defines the minimum number of characters from each character group a user's passphrase must satisfy. The default value is 1.

3.8.4 Certificate Generation for Server Processes

The Certificate Generation Tool should be invoked as follows to generate certificates for server processes:

```
certtool -rpp <propfile>
```

or:

```
java -classpath certtool.jar CertTool -rpp <propfile>
```

where propfile is the name of the property file. Following is a sample file:

```
! Certificate Generation Properties
! SWSI Application Server (10yr)
! Date: 08/24/2002
! Modified by: Joe Stevens
! Rev: 09/06/2002

DebugALL : false
DebugCert : false
DebugProp : false

Organization      : NASA Goddard Space Flight Center
OrganizationUnit  : Mission Services Program Office, Code 450
Locality          : Greenbelt
StateOrProvince   : MD
Country           : USA
Common            : SWSI Application Server
EmailAddress      : http://swsi.gsfc.nasa.gov

! Expiration given in days (Application Server expiration should be long, e.g. 5-10 years)
! 5 years has max 2 leap years = 3650+2 days
! 10 years has max 3 leap years = 3650+3 days
! Certificate expires on September 30, 2012
CertificateExpiration: 3677
CertSerialNumber   : 1001
Strength           : 1024

CertificateAuthorityFile      : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
cert.der.10yr
CertificateAuthorityPrivateKeyFile : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
key.der.10yr

ClientCertificateRequestFile : /export/home/swsiops/SWSI03.1/certs/SWSI-appserver-csr.der
ClientCertificateFile        : /export/home/swsiops/SWSI03.1/certs/SWSI-appserver-
cert.der.10yr
ClientPrivateKeyFile         : /export/home/swsiops/SWSI03.1/certs/SWSI-appserver-key.der
ClientEncryptedPrivateKeyFile: /export/home/swsiops/SWSI03.1/certs/enc-SWSI-appserver-
key.der.10yr

ClientPassPhrase : This is the server passphrase

KeyArchiveDir    : /export/home/swsiops/SWSI03.1/certs
KeyArchiveFile   : /export/home/swsiops/SWSI03.1/certs/SWSI-appserver-200209-key-10yr.zip
```

The properties are defined the same as described in Section 3.8.3.

3.8.5 Certificate Generation for SWSI CA

The Certificate Generation Tool should be invoked as follows to generate the CA's certificate and private key:

```
certtool -genca <propfile>
```

or:

```
java -classpath certtool.jar CertTool -genca <propfile>
```

where propfile is the name of the property file. Following is a sample file:

```
! Certificate Generation Properties
! Certificate Authority ONLY
! Date: 03/14/2002
! Modified by: Joe Stevens
! Rev: 04/16/2003

DebugALL : false
DebugCert : false
DebugProp : false

Organization      : NASA Goddard Space Flight Center
OrganizationUnit  : Mission Services Program Office, Code 450
Locality          : Greenbelt
StateOrProvince   : Maryland
Country           : US
Common            : Space Network Web Services Interface (SWSI)
EmailAddress      : http://swsi.gsfc.nasa.gov/

! Expiration given in days (CA expiration should be long, e.g. 5-10 years)
! 5 years has max 2 leap years = 3650+2 days
! 10 years has max 3 leap years = 3650+3 days
! Certificate expires on September 30, 2012
CertificateExpiration: 3677
CertSerialNumber   : 1
Strength           : 1024

! Full path to the output files recommended
CertificateAuthorityFile      : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
cert.der.10yr
CertificateAuthorityPrivateKeyFile : /export/home/swsiops/SWSI03.1/certs/CA/SWSI-ca-
key.der.10yr
```

The properties are defined the same as described in Section 3.8.3.

3.9 High Availability Application

3.9.1 General

The High Availability (HA) application runs in the background as user *root*, and is started at system boot time by the */etc/rc3.d/S40HAD* init script. The HA application is configured by editing a number of configuration files and scripts that are stored in the *swsiops* account. A local operator can control the operation of the HA application by using the HA GUI application as described in Section 4. The *High Availability (HA) User's Guide* provides more detailed information on how to configure and operate the HA application.

Within an HA cluster, only one of the two computers will be executing the defined set of server processes at one time. These server processes depend on certain resources that are also controlled by one computer at a time.

The Open Server HA application manages the following applications and resources:

- Application Server process
- Open Virtual IP address that corresponds to the *swsi-server.nascom.nasa.gov* hostname

The Backend Server HA application manages the following applications and resources:

- Application Server process
- Open Isolator process
- Closed Isolator process
- OPS SNIF process
- ANCC SNIF process
- OPS SDIF process
- EIF SDIF Process
- Closed Virtual IP address that corresponds to the *swsi-server.ops.nascom.nasa.gov* hostname
- RAID disk array
- Oracle database processes

The virtual IP addresses mentioned above are floating addresses that allow Client users to connect to a single IP address, regardless of which server is prime.

The remainder of this subsection shows examples of HA configuration files and scripts, all of which are owned by the *swsiops* account and are stored in the */export/home/swsiops/ops* directory tree.

3.9.2 HA System Startup Script

The *HA.sh* script is sourced by the */etc/rc3.d/S40HAD* script at system boot time. It sets up the environment used by the HA background application. Following is a sample script:

```
#!/usr/bin/sh
#
# augment path with SWSI-specific stuff
#
PATH=/export/home/cots/java/bin:${PATH}
PATH=/export/home/swsiops/bin:${PATH}
PATH=/export/home/swsiops/current_release/bin:${PATH}
#
# set Base environment variables
#
read VERSION < /etc/ha_version

DBA_BASE_DIR=/export/home/swsiops
BASE_DIR=/export/home/swsiops/$VERSION
BIN_DIR=$BASE_DIR/bin
UTIL_DIR=$BASE_DIR/util
export DBA_BASE_DIR
export BASE_DIR
export BIN_DIR
export UTIL_DIR

PATH=${PATH}:${BIN_DIR}:${UTIL_DIR}
#
# set HA environment variables
#
HA_HOME_DIR=$BASE_DIR
HA_BIN_DIR=$HA_HOME_DIR/bin
HA_UTIL_DIR=$HA_HOME_DIR/util
HA_SCRIPTS_DIR=$HA_HOME_DIR/scripts
HA_BLCFG_DIR=$BASE_DIR/haBlConfig
HA_CONFIG_DIR=$BASE_DIR/haConfig
HA_LOG_DIR=$BASE_DIR/haLogFiles
HA_ARC_DIR=$BASE_DIR/haLogArchives
export HA_HOME_DIR
export HA_BIN_DIR
export HA_UTIL_DIR
export HA_SCRIPTS_DIR
export HA_BLCFG_DIR
export HA_CONFIG_DIR
export HA_LOG_DIR
export HA_ARC_DIR
#
# set Generic environment variables
#
CONFIG_DIR=$HA_HOME_DIR/haConfig
AUDIO_DIR=$HA_UTIL_DIR
NPG_UTIL_DIR=$HA_HOME_DIR/util
NPG_BIN_DIR=$HA_HOME_DIR/bin
NPG_CONFIG_DIR=$BASE_DIR/haConfig
export CONFIG_DIR
export AUDIO_DIR
export NPG_UTIL_DIR
export NPG_BIN_DIR
export NPG_CONFIG_DIR
```

3.9.3 Operator `cshrc` Script

The operator `.cshrc` script is sourced by each local operator's `.cshrc` script when the operator logs in. It sets up the environment so that the operator can run the GUI tools (HA GUI, HA DeLogger, etc.) on the CDE panel as described in Section 4. Following is a sample script:

```
# @(#)cshrc 1.11 89/11/29 SMI
#
# set file creation mask to rw-rw-r-- (rw- [owner] rw- [group] r-- [other user])
#
umask 002
#
# set file name completion when esc key is entered
#
set filec
#
# set all paths
#
set path = ($path ~)
set path = ($path ~/bin)
set path = ($path /export/home/swsiops/current_release/bin)
set path = ($path /usr/local)
set path = ($path /usr/local/bin)
set path = ($path /usr/local/sbin)
set path = ($path /usr/5bin)
set path = ($path /usr/sbin)
set path = ($path /opt/NeWSprint/bin)

set path = ($path /usr/bin/X11)
set path = ($path /usr/openwin/bin)
set path = ($path /usr/ccs/bin)

setenv MOZILLA_HOME /usr/local/netscape

if ( $?prompt ) then
    set history=50
    alias ll ls -la
    alias lt ls -lat
endif

#
# set Base environment variables
#
setenv VERSION `cat /etc/ha_version`
setenv DBA_BASE_DIR /export/home/swsiops
setenv BASE_DIR /export/home/swsiops/$VERSION
setenv BIN_DIR $BASE_DIR/bin
setenv UTIL_DIR $BASE_DIR/util

#
# set HA environment variables (Backward Compatible)
#
setenv HA_HOME_DIR $BASE_DIR
setenv HA_BIN_DIR $HA_HOME_DIR/bin
setenv HA_UTIL_DIR $HA_HOME_DIR/util
setenv HA_SCRIPTS_DIR $HA_HOME_DIR/scripts
setenv HA_BLCFG_DIR $BASE_DIR/haBlConfig
setenv HA_CONFIG_DIR $BASE_DIR/haConfig
setenv HA_LOG_DIR $BASE_DIR/haLogFiles
setenv HA_ARC_DIR $BASE_DIR/haLogArchives

#
# set Generic environment variables
#
```

```

setenv CONFIG_DIR $SHA_HOME_DIR/haConfig
setenv LOG_FILESYSTEM /export/home
setenv AUDIO_DIR $UTIL_DIR
setenv NPG_UTIL_DIR $SHA_HOME_DIR/util
setenv NPG_BIN_DIR $SHA_HOME_DIR/bin
setenv NPG_CONFIG_DIR $BASE_DIR/haConfig

#
# add additional paths for Utilities
#
set path = ($path $UTIL_DIR)
set path = ($path $BIN_DIR)

```

3.9.4 HA Configuration File

The HA configuration file configures the internal mechanisms of the HA application. Once configured, this file usually does not require changes. The IP addresses in the sample are from the RFC-1918 private IP address space, and must match the UNIX-level configuration of the server's heartbeat LAN ports. The Open and Backend Servers use different private networks.

Each computer with HA installed on it uses two Ethernet links for heartbeats. One of these networks is named Primary in the configuration file, and the other is named Secondary. Internally, HA does not make a distinction; it uses both interfaces for heartbeats at all times.

In the following example, the default role is set to PRIMARY. One of the two computers in an HA cluster should be configured as PRIMARY, and the other should be configured as BACKUP. Note: the default role specified in the configuration file can be overridden by a command line option in */etc/rc3.d/S40HAD*. The SWSI servers are initially configured this way with the initial role forced to HALTED. Following is a sample file:

```

!-----
!
! haConfiguration
!
! High Availability configuration file template.
!
! Edit fields only!!!!!!
! DO NOT change field labels.
!
!-----
!
! Formats, by parameter name:
!
!      DefRole      BACKUP      default role
!      PrimHBLocAddr 128.183.81.49 primary HB local IP address
!      PrimHBLocPort 7777      primary HB local UDP port
!      PrimHBRemAddr 128.183.52.48 primary HB remote IP address
!      PrimHBRemPort 7777      primary HB remote UDP port
!      SecHBLocAddr 128.183.81.49 secondary HB local IP address
!      SecHBLocPort 8888      secondary HB local UDP port
!      SecHBRemAddr 128.183.52.49 secondary HB remote IP address
!      SecHBRemPort 8888      secondary HB remote UDP port
!
!
!-----

```

```

!      MissedHBTol      0          missed heartbeat tolerance
!                                     <greater than or equal to 0>
!      HBTimeout        5          heartbeat timeout(sec)
!                                     <greater than 1>
!      ShrMemKey        7890       shared memory key
!
! Notes:
!      1. IP addresses are fully.qualified.domain.names
!      2. IP ports are either numeric or are "well-known" names
!
!-----
!param-name          parameter values
!-----
DefRole              PRIMARY
!
PrimHBLocAddr       192.168.115.3
PrimHBLocPort       7777
PrimHBRemAddr       192.168.115.4
PrimHBRemPort       7777
SecHBLocAddr        192.168.116.3
SecHBLocPort        8888
SecHBRemAddr        192.168.116.4
SecHBRemPort        8888
!
MissedHBTol         1
HBTimeout           5
ShrMemKey           7900
!

```

3.9.5 HA Services File

The HA services file controls which services (applications or resources) are managed by HA. Each managed service that has an entry in this file must also have an appropriate set of HA scripts in the *scripts* directory. For example, if a configuration entry is named *xyzzy*, HA expects that the scripts directory will contain *xyzzy.kill*, *xyzzy.restart*, and *xyzzy.test*. The name of the service is arbitrary; it is not required to exactly match the name of an application or resource. What constitutes an application or resource is determined entirely by the contents of the service's scripts.

The Open Server has a single application (*aserver*). The initial configuration of the Backend Server has entries for seven applications (*aserver*, *iso-open*, *iso-closed*, *ops-snif*, *ancc-snif*, *ops-sdif*, *eif-sdif*), an entry for the Oracle database service, and an entry that is used to manage ownership of the RAID. Following is a sample file:

```

!-----
!
! haServices
!
! High Availability services configuration file template.
!
! Formats, by parameter name:
!
!      service_name      number of      process check
!                       retries          delay(sec)
!
!-----
aserver                5              10

```

3.9.6 HA Startup Script

The HA startup script is a Bourne shell script which is executed by the HA background application whenever it decides to change the HA role to Primary. This script is responsible for setting up the run-time environment needed by the SWSI services. In this example, the script brings up the *swsi-server* floating virtual IP address on logical interface *qfe0:1* (and does some logging):

```
#!/bin/sh
#-----
# Name:          ha.startup
#
# Purpose:      High Availability startup script used to start
#              monitored services and configure interfaces
#
#-----

rm -f /tmp/ha.startup.scripts.out

echo "ha.startup executing, at: `date`" > /tmp/ha.startup.scripts.out
echo " $0" >> /tmp/ha.startup.scripts.out
set >> /tmp/ha.startup.scripts.out

# Configure qfe0 overlayed (qfe0:1) interface
ifconfig qfe0:1 plumb
ifconfig qfe0:1 inet swsi-server netmask + broadcast + up

echo "ha.startup finished, at: `date`" >> /tmp/ha.startup.scripts.out

exit 0
```

3.9.7 HA Shutdown Script

The HA shutdown script is a Bourne shell script which is executed by the HA background application whenever it decides to change the HA role from Primary to Halted. This script is responsible for shutting down the SWSI services in their proper order, and for bringing down the *swsi-server* floating virtual IP address on logical interface *qfe0:1*. The services are stopped by executing the appropriate *kill* scripts (e.g., *xyzy.kill*). Following is a sample script:

```
#!/bin/sh
#-----
# Name:          ha.shutdown
#
# Purpose:      High Availability Shutdown script used to stop
#              all monitored services and unconfigure interfaces
#
#-----

#-----#
# stop services #
#-----#

# stop aserver

$HA_SCRIPTS_DIR/aserver.kill

#-----#
# unconfigure required interfaces #
#-----#
```

```

# Unconfigure qfe0 overlaid (qfe0:1) interface

ifconfig qfe0:1 down

exit 0

```

3.9.8 Service Scripts

Each service managed by HA is required to have three scripts: *kill*, *restart*, and *test*. The examples below use a fictional service *xyzy*. In these examples, the script's logging is not shown; consult the actual scripts for details.

The *kill* script is responsible for shutting down the service in an appropriate manner. Following is a sample script:

```

#!/bin/sh
#-----
# Name:          xyzy.kill
#
# Purpose:      High Availability script for killing the xyzy Application
#
#-----

#
# xyzy.pid contains PID of currently or previously run instance of
# the xyzy Application.  If xyzy.pid does not exist, assume process is
# not already running and start it up.
#

if test ! -r /tmp/xyzy.pid
then
    exit 1
fi

#
# If xyzy.pid does exist, check to see if process is running
#

pid=`cat /tmp/xyzy.pid`
if [ `ps -p $pid | wc -l` -le 1 ]
then
    exit 1
fi

# fall through -- xyzy is running

kill $pid

exit 0

```

The *restart* script is responsible for starting the service. Note that the standard output and standard error files of the application are piped through *timetag* and written to a log file. The function *startcomponent* is used to separate the invocation of the application from the script setup. Following is a sample script:

```

#!/usr/bin/sh

# define function that starts the component

startcomponent ()
{
    cd $SWSIBIN
    CLP=$CLASSPATH:$SWSIJARS/SWSIxyzy.jar
    export CLP
    java -Duser.timezone=GMT -classpath $CLP gov.nasa.gsfc.swsi.xyzy.net.SWSIxyzy \
        ../properties/xyzy.prop \
        2>&1 | $DEBUGTAG &
}

#
# script execution begins here
#

PIDFILE=/tmp/xyzy.pid

umask 022

# set up common swsi env. vars and path

cd $SHA_SCRIPTS_DIR

. ./swsi-common.sh

APASSPHRASE="No! Not THAT button!"
export APASSPHRASE

DEBUGFILE=$SWSIROOT/debug/xyzy/xyzy.debug
DEBUGTAG="/export/home/swsiops/current_release/bin/timetag -f $SWSIROOT/debug/xyzy -a
$SWSIROOT/debug/xyzy -m 1000000 -g"

#
# xyzy.pid contains PID of currently or previously run instance of
# the xyzy Application.  If xyzy.pid does not exist, assume process is
# not already running and start it up.
#

if test ! -r $PIDFILE
then
    startcomponent
    exit 0
fi

#
# If xyzy.pid does exist, check to make sure process isn't already
# running before starting it.
#

pid=`cat $PIDFILE`
if [ `ps -p $pid | wc -l` -le 1 ]
then
    startcomponent
fi

exit 0

```

The *test* script is responsible for testing whether or not the service is still running. A return code of zero indicates that the service is running, while a non-zero return code indicates that the service is not running. Following is a sample script:

```
#!/bin/sh
#-----
# Name:          xyzzy.test
#
# Purpose:       High Availability test script for xyzzy Application
#
#-----
#
# xyzzy.pid contains PID of currently or previously run instance of
# xyzzy Application.  If xyzzy.pid does not exist, assume process is
# not already running.
#
if test ! -r /tmp/xyzzy.pid
then
    exit 1
fi

#
# If xyzzy.pid does exist, check to see if process is running
#
pid=`cat /tmp/xyzzy.pid`
if [ `ps -p $pid | wc -l` -le 1 ]
then
    exit 1
fi

# fall through -- xyzzy is running

exit 0
```

3.10 NISN Secure Gateway

Table 3-1 lists all the NISN Secure Gateway rules required to provide the needed communications between the Backend Servers and the Open Servers. The Isolator on the Backend Servers connects to the Application Server on the Open Servers using the Open Server Virtual Address so that communication always occurs with the primary Open Server. For TUT transfers, both Open Servers are always updated so that when an Open Server failover occurs, the new prime Open Server immediately contains the latest TUT data files.

Table 3-1. Secure Gateway Rules

Closed IONet Source IP	Open IONet Destination IP	Open IONet Destination TCP Port
Backend Server A Permanent Address	Open Server Virtual Address	Application Server Directive Port (<i>isolatorServerDirectivePort</i> property)
Backend Server A Permanent Address	Open Server Virtual Address	Application Server Alert Port (<i>isolatorServerEventsPort</i> property)
Backend Server A Permanent Address	Open Server Virtual Address	Application Server Data Port (<i>isolatorServerDataPort</i> property)
Backend Server B Permanent Address	Open Server Virtual Address	Application Server Directive Port (<i>isolatorServerDirectivePort</i> property)
Backend Server B Permanent Address	Open Server Virtual Address	Application Server Alert Port (<i>isolatorServerEventsPort</i> property)
Backend Server B Permanent Address	Open Server Virtual Address	Application Server Data Port (<i>isolatorServerDataPort</i> property)
Backend Server A Permanent Address	Open Server A Permanent Address	TUT Proxy Receiver Port (<i>ServerPort</i> property)
Backend Server A Permanent Address	Open Server B Permanent Address	TUT Proxy Receiver Port (<i>ServerPort</i> property)
Backend Server B Permanent Address	Open Server A Permanent Address	TUT Proxy Receiver Port (<i>ServerPort</i> property)
Backend Server B Permanent Address	Open Server B Permanent Address	TUT Proxy Receiver Port (<i>ServerPort</i> property)

Section 4. User Environment

4.1 Overview

Operator and Database Administrator (DBA) accounts on the SWSI servers are standard Unix accounts. The Common Desktop Environment (CDE) is used to provide buttons and menu options to monitor server operation and perform common functions. After logging into the server, a CDE desktop with a two-level toolbar appears, such as is shown in Figure 4-1 or Figure 4-3. Separate toolbar layouts are provided for the Backend and Open servers. The toolbars are designed similar to those provided in the NCCDS Protocol Gateway (NPG) as described in the *NPG Operator's Guide*. Each toolbar level contains several icons, buttons, and menus. The lower toolbar level contains icons and menus from the system default CDE configuration. The upper toolbar was added for the SWSI servers. In addition to the application icons, there is a Julian Clock in the upper level toolbar. For more information about the CDE default configuration and applications, see the CDE User's Guide or use the Help Manager on the lower toolbar level.

The remainder of the section will address primarily those CDE functions that are specific to the SWSI servers. Details on functions that are common to the NPG CDE environment may be found in the *NPG Operator's Guide*.

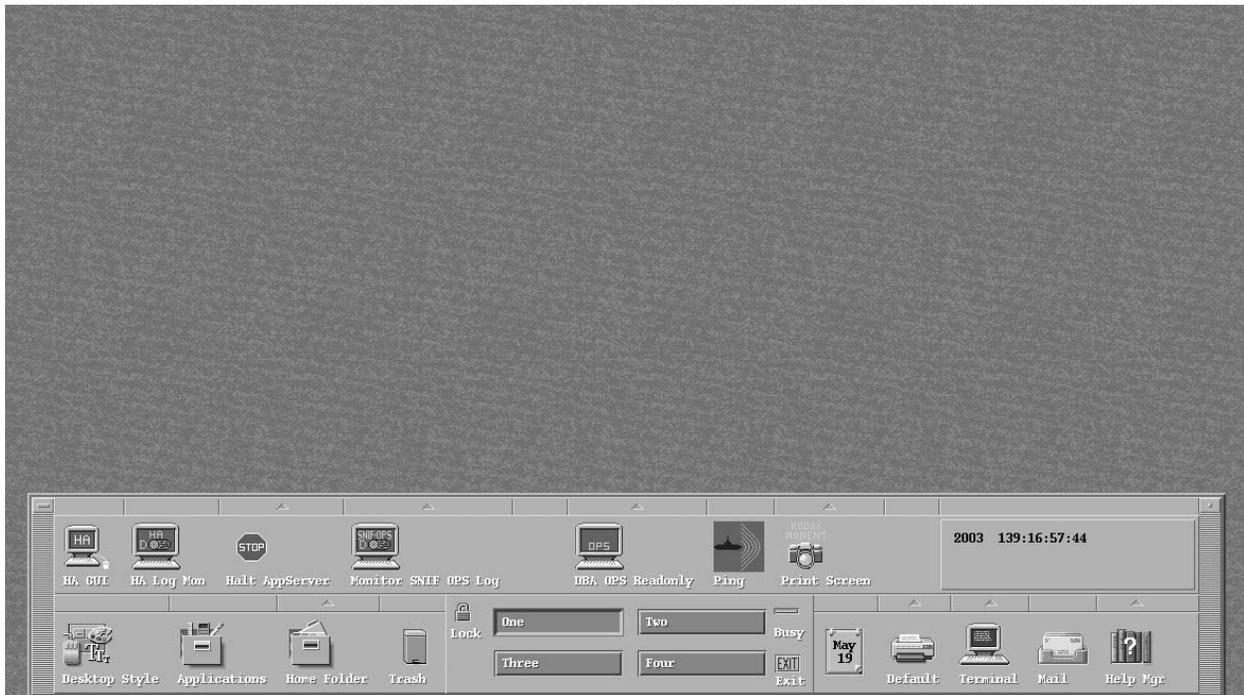


Figure 4-1. Backend Server CDE Toolbar

4.2 Backend Server Menus

The menus and buttons available on the sample Backend Server CDE toolbar as shown in Figure 4-2 are as follows [Note: The Backend Server CDE toolbar on your display may be slightly different due to the environment]:

- HA GUI Button – provides monitoring and control of the High Availability (HA) application. The HA GUI application is described in detail in Section 3 of the *NPG Operator's Guide*. It should be noted that the Backend Servers tend to be slow in changing roles due to the amount of time it takes to mount/dismount the RAID and to startup/shutdown Oracle. The following occurs for specific role changes:
 - PRIMARY to HALTED – this takes 1-2 minutes. There is no progress indication on the main HA GUI panel until the status shows HALTED.
 - BACKUP to PRIMARY – status is shown as PRIMARY immediately, but the transition takes 2-3 minutes to complete. One cycle of application failures may occur during this time. There needs to be patience during this startup as it will eventually complete successfully, even if there is no indication of completion. There should only be concern if the application failure counts continue to increment past one, thus resulting in a total transition failure.

In both these cases, monitoring messages in the HA log helps to give some indication of progress of the transition. Care must also be taken to allow a transition to complete on one server before initiating a role change on the other server. For example, if a server is in the process of changing from BACKUP to PRIMARY, don't try to change the other server from HALTED to BACKUP, as this may result in both servers trying to be PRIMARY at the same time. Wait until the first server has completed its transition to PRIMARY before changing the other server to BACKUP.

- HA Log Mon Button – provides a display of the HA log file using the NPG Delogger application. The Delogger is described in detail in Section 7 of the *NPG Operator's Guide*.
- Applications Control Menu – provides a means of halting individual SWSI Server applications when in Primary mode. These buttons are used to restart applications after a configuration or database change. The buttons in this menu halt the applications, while the HA application running in the background automatically restarts them.
- SNIF Delogging Menu – provides displays of the SNIF log files using the NPG Delogger application. The *SNIF OPS Logs* and *SNIF ANCC Logs* buttons open file browsers to the log file archive directories. The Delogger is described in detail in Section 7 of the *NPG Operator's Guide*. SNIF-specific log messages are described in Appendix A.
- DBA Tools Menu – invokes the SWSI Database Administration (DBA) tool described in Section 7. The *DBA OPS*, *DBA OPS2*, *DBA EIF*, and *DBA EIF2* buttons are used by a Database Administrator with an Oracle account with full update privilege as described in Section 6.5. The user is prompted for an Oracle username and password after selecting the appropriate button. Each button provides access to a different database instance as

described in Section 6.1. There are also corresponding *Readonly* buttons that allow a system operator to view system status (users logged in, NCCDS connection status, etc.), and to view but not update customer and configuration data. Oracle username and password are not required to use these buttons.

- Ping Button – this button is described in detail in Section 9 of the *NPG Operator’s Guide*.
- Utilities Menu – these menu options are described in detail in Section 8 of the *NPG Operator’s Guide*.



Figure 4-2. Backend Server Toolbar Menus

4.3 Open Server Menus

The menus and buttons available on the Open Server CDE toolbar as shown in Figure 4-3 are as follows:

- HA GUI Button – provides monitoring and control of the High Availability (HA) application. The HA GUI application is described in detail in Section 3 of the *NPG Operator’s Guide*. Unlike with the Backend Servers, transitions on the Open Servers are almost instantaneous.

- HA Log Mon Button – provides a display of the HA log file using the NPG Delogger application. The Delogger is described in detail in Section 7 of the *NPG Operator's Guide*.

- Applications Control Menu – provides a means of halting individual SWSI Server applications when in Primary mode. These buttons are used to restart applications after a configuration change. The buttons in this menu halt the applications, while the HA application running in the background automatically restarts them.
- Ping Button – this button is described in detail in Section 9 of the *NPG Operator's Guide*.
- Utilities Menu – these menu options are described in detail in Section 8 of the *NPG Operator's Guide*.

4.4 Logging Off

Before logging off of the server, the user should check all of the desktops and close any windows with running applications. If not, these processes may continue to run in the background after the user has logged off or may be restarted automatically when the user logs back in. Once started by the HA application, the SWSI applications will continue to run in the background after logging off the server. To logoff, click on the *EXIT* button in the middle of the lower toolbar. After the *Logout Confirmation* window appears, click on the *OK* button.

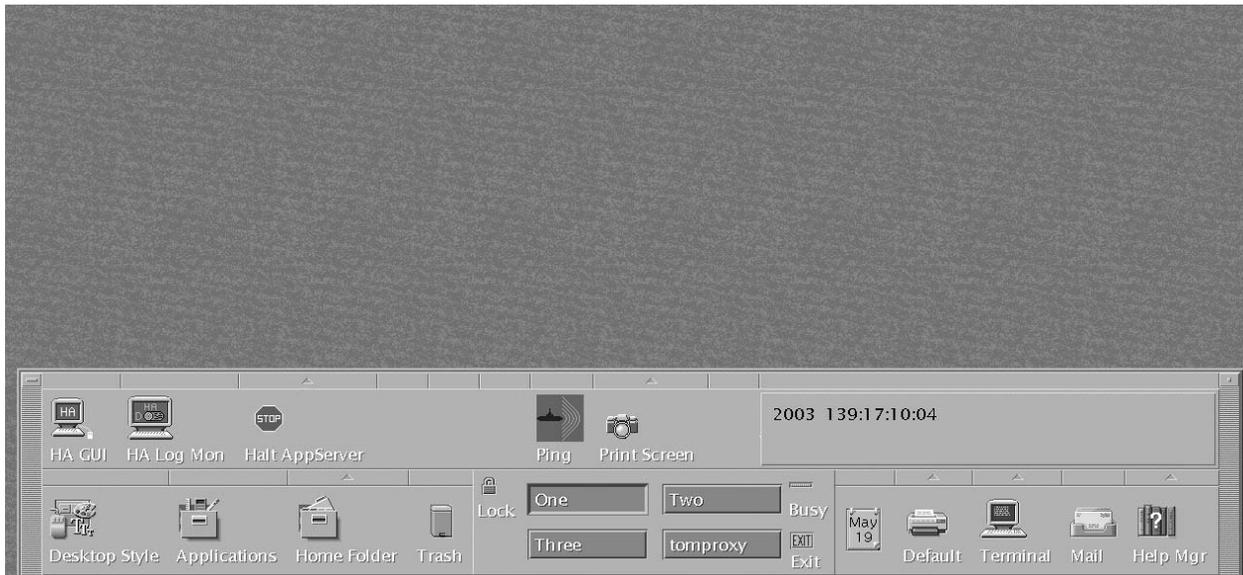


Figure 4-3. Open Server CDE Toolbar

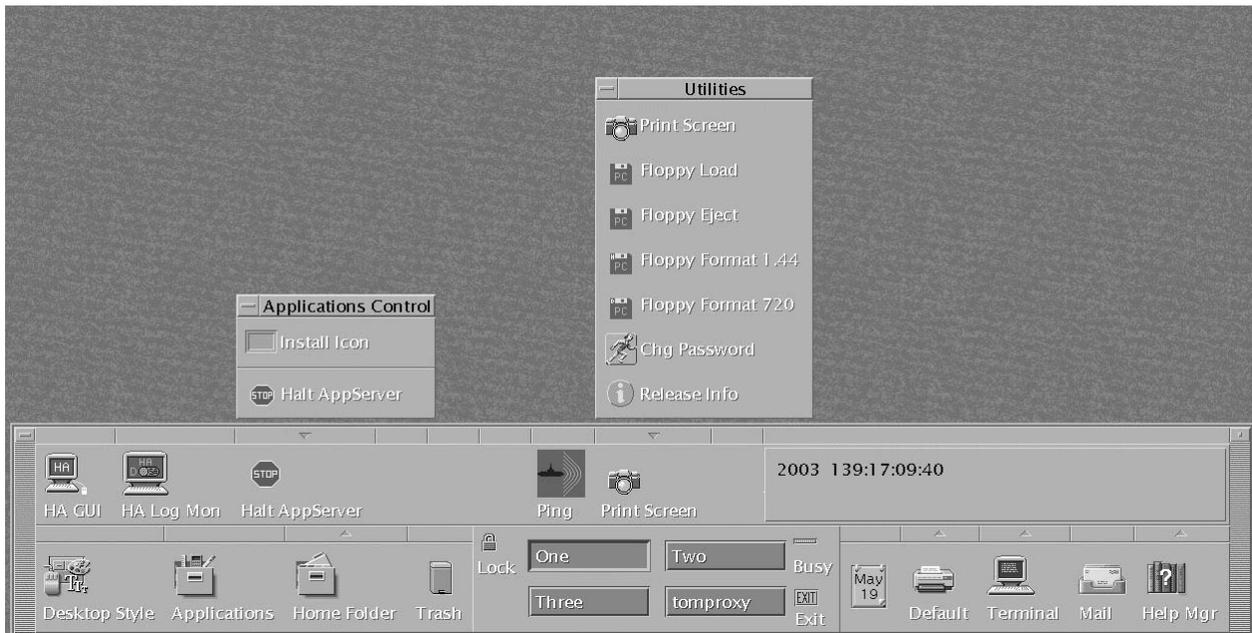


Figure 4-4. Open Server Toolbar Menus

Section 5. Customer and User Setup

5.1 General

This section contains general guidelines and procedures for managing data in support of the customer (mission) and user interfaces. Specific instructions for performing these operations is contained in other sections of this document, including Section 7, *Database Administration*, and Section 9, *System Administration Procedures*. In the procedures explained below that involve modifying the SWSI Database, note that SWSI maintains separate databases for operations (OPS) and test (EIF) modes, allowing separate settings (SSCs, user accounts, etc.) for each mode.

5.2 Adding Customers

Configuring the SWSI Servers for new customers (missions) consists of entering the following information from the appropriate NCCDS (OPS or ANCC) Database and from the customer:

1. Schedule Request purge time - establish with the customer how long after requested event start time to keep Schedule Requests before they are purged from the SWSI Database. This affects how many requests are displayed in the Client software Schedule Request Summary Panel. The purge time is entered into the SWSI Database along with the SIC as described in Step 2 below.
2. Spacecraft Identification Code (SIC) – entry of the SIC into the SWSI Database is described in Section 7.5.
3. SUPIDENs – entry of all associated SUPIDENs into the SWSI Database is described in Section 7.7.
4. Schedule Connection – a Schedule Connection is configured for connecting to the Service Planning Segment (SPS) as described in Section 7.3. An existing Schedule Connection entry in the SWSI Database may be used by adding the new SIC to that entry and ensuring that SPS is configured to send schedule results for the new SIC to the same Logical Destination. If creating a new SWSI Database Schedule Connection, then the NCCDS DBA must create a new Logical Destination, User ID, and Password. This information only needs to be entered into both the SPS and SWSI Databases and does not need to be shared with the customer.

5. Realtime Connection – a Realtime Connection is configured for connecting to the NCCDS Protocol Gateway (NPG) acting as a proxy TCP server for the Communications and Control Segment (CCS) as described in Section 7.4. An existing Realtime Connection entry in the SWSI Database may be used by adding the new SIC to that entry and ensuring that CCS is configured to send reconfiguration and performance data for the new SIC to the same destination. If creating a new SWSI Database Realtime Connection, then the NCCDS DBA must create a new User ID and Password. This information only needs to be entered into the NPG configuration and SPS, CCS, and SWSI Databases and does not need to be shared with the customer.
6. Prototype Event Codes – entry of all associated Prototype Event Codes into the SWSI Database is described in Section 7.6.
7. Service Specification Codes (SSCs) – entry of all SSCs and initial service parameter values into the SWSI Database is described in Section 7.9. Entry of a new SSC causes the default parameter values to be set the NULL. The SWSI DBA or MOC mission manager must log in from a SWSI Client to set the default parameters to their correct initial values.
8. Active Schedule Upload – establish with the customer whether or not they would like to receive an Active Schedule file on connected workstations as described in Section 8.8 of the *SWSI Client Software User's Guide*. The customer must also provide the following information about the upload process and file content:
 - Whether to send a new file when it changes and, if so, how often to check for changes, in minutes (default 5 minutes).
 - Whether to periodically send a new file regardless of whether there are changes and how often, in minutes (default 60 minutes).
 - Whether to include initial service parameter values.
 - For enumerated parameter types, whether to send the numeric value or an enumerated text string.

Entry of the Active Schedule Upload information into the SWSI Database is described in Section 7.10.

Note that all SWSI customers should be entered into the NCCDS or ANCC Database as full support customers.

5.3 Adding SWSI Client Users

Each SWSI Client user must have a separate account (group accounts are not allowed). The customer should provide the following information about new users:

1. Signed Rules of Behavior.
2. Contact information including full user name, company, mission name, geographic location, phone number, and email address.
3. Whether user should be allowed Mission Manager privileges. This allows the user to edit the initial SSC parameter values.

The SWSI DBA will need to assign a userid and temporary password and provide that information back to the user. The general convention on userid assignment is to use the first initial plus the last name (e.g., tsardella). Up to twenty characters are allowed for the userid. The SWSI DBA may either assign a temporary password or just use the SWSI DBA tool default, which is the same as the userid. When adding the new account, the password should also be set to expired so that the user is forced to enter a new password on initial login that meets the password criteria set by the SWSI applications. Entry of new user accounts is described in Section 7.2.

5.4 IP Address Filtering

The customer should provide a list of IP addresses for all client workstations requiring connection to the SWSI Server(s). The customer may request access for any combination of workstations located on the Closed IONet, Open IONet, or Internet. Though not always the case, IP addresses for all workstations connecting to the SWSI Servers should be treated as For Official Use Only (FOUO). Transmission of IP addresses to the DBA should be done by the following accepted means for FOUO information: voice, fax, or PGP electronic mail. Entry of IP addresses into the IPFilter configuration is described in Section 9.5. This configuration should be reviewed on a regular basis to remove IP addresses that may no longer be required.

5.5 SSC Management

Section 5.2, Step 7 above describes how to add SSC codes to the SWSI Database and to set default service parameter values. Additional functions for managing these codes are described in Section 7.9. Care must be taken when modifying SSCs and default parameter values in the NCCDS Database to be sure that the same changes are made in the SWSI Database so that the two databases are synchronized.

SWSI Client users with NCC Mission Manager privileges should also be made aware that any changes they make to SSC default parameter values should be closely coordinated with the NCCDS DBA to ensure the same synchronization. Coordination by the user with the DAS DBA is not required since DAS SSC codes are only managed locally to SWSI. Section 12 of the *SWSI Client Software User's Guide* describes the process of managing SSCs from the SWSI Client application in more detail.

5.6 Client User Login Problems

Occasionally, a SWSI Client user will experience problems logging into the SWSI Servers. When this happens, they will receive an error dialog that states the userid or password may be invalid, or that the user may already be logged in with that userid from the same IP address. If contacted by a user with this problem, the following steps should be followed by the SWSI DBA (see Section 7.2):

1. Check that the userid exists in the SWSI Database.
2. Check whether the account has been deactivated because of too many failed login attempts. This is often caused by a user forgetting his password. If so, then the user's password should be reset to a temporary value, the password expiration date set to expired, and the account reactivated. The temporary password should be given to the user. The user will be required by the SWSI Client to change password after a successful login.

Check to see if the user is already logged in. If so, inform the user that he may still have another SWSI Client application running on the same host and connected to the SWSI Server. If that is not the case, then the problem may be caused by a known bug in the SWSI Server applications. To fix the problem, the SWSI DBA may either mark the user as logged off as described in Section 7.2.5, or restart the appropriate Application Server as described in Section 4.

Section 6. Database Design and Management

6.1 Database Schema

Figures 6-1 through 6-3 show the SWSI database structure in entity-relationship notation. The relationships are implemented by using primary keys and foreign key constraints. The tables are divided into three general categories:

- Static – contents are defined by the software release and don't otherwise change. Examples are UPD and service descriptions.
- Semi-Static – contents are maintained by the SWSI Database Administrator (DBA). Updates are made in response to changes in customer/user requirements and system configuration.
- Dynamic – contents are changed by SWSI applications in response to normal operations. Examples are tables holding Schedule Request information.

A summary of all the database tables is given in Table 6-1. In general, only the semi-static tables are of importance to a server operator or DBA. The `ACTIVITY_LOG` and `USER_LOGIN` dynamic tables are also of interest because they provide status information to the operator. Maintenance of these tables is performed using the SWSI Database Administration tool described in Section 7.

Separate database instances (OPS, EIF) are used to maintain separate data for operations and test modes. A second set of instances (OPS2, EIF2) allows simultaneous operation of two different software releases on the same set of servers. This allows for minimal downtime when installing new releases that may have different schemas, and allows for a gradual transition of a large number of users to a new release.

Table 6-1. SWSI Database Tables

Table Name	Category	Description
ACTIVE_EVENTS_UPLOAD	Semi-Static	Parameters for periodic upload of active schedule file(s) to Client workstations
ACTIVE_SCHEDULE	Dynamic	Active (confirmed) events, derived from User Schedule Messages (USMs) received from NCCDS
ACTIVE_SCH_PARAM	Dynamic	Initial parameter values for services for confirmed events
ACTIVE_SCH_SERVICE	Dynamic	Services for confirmed events
ACTIVITY_LOG	Dynamic	Client login/logout events
ALERT_MESSAGE	Dynamic	Alerts generated by server processes and sent to Client workstations
DAS_RESULT_CODE	Static	Possible response codes received from DAS, along with explanation text
GCMR_PARAM	Dynamic	Parameter values for user reconfiguration requests (98/04) sent to NCCDS
GCMR_STATUS_CODE	Static	Possible GCMR Accept/Reject status codes received from NCCDS, along with explanation text
MAPREQUESTTOISOLATOR	Dynamic	Used internally by SDIF for routing response messages from DASCON to the Isolator
NCC_SV_FIXED_PARAM	Static	Fixed values for fields in State Vector (IIRV) messages sent to NCCDS
PLAYBACK	Dynamic	Playback requests sent to DAS
PROTOTYPE_EVENT_CODE	Semi-Static	Valid NCCDS Prototype Event Codes assigned to a SIC
REALTIME_CONNECTION	Semi-Static	NCCDS realtime connection (reconfig, pmData) configuration
REQUEST	Dynamic	NCCDS and DAS schedule requests
REQUEST_STATUS	Static	Schedule Request status codes (Transmitted, Queued, Granted, etc.)
REQUEST_TYPE	Static	Schedule Request types (SAR, SDR, RAR, etc.)
SAR	Dynamic	NCCDS Schedule Add Requests (SARs)
SCHEDULE_CONNECTION	Semi-Static	NCCDS scheduling connection (schReq, schStatus, etc.) configuration
SERVICE_LINK	Static	Lookup table containing the legal services and support types supported by NCCDS
SERVICE_PARAM	Static	Service parameter definitions
SERVICE_TYPE	Static	Service type (MAR, SMAF, DASMAR, etc.) information
SIC	Semi-Static	Support Identification Codes (SICs) for spacecraft supported by SWSI
SP_ENUM_VALIDATION	Static	Valid values for enumerated service parameters
SP_NUMERIC_VALIDATION	Static	Valid ranges of values for numeric service parameters
SRM_RESULT_CODE	Static	Possible result/explanation codes received from NCCDS in a Schedule Result Message (SRM), along with explanation text
SR_PARAM	Dynamic	Initial parameter values for services for a Schedule Request
SR_SERVICE	Dynamic	Services for a Schedule Request
SSC	Semi-Static	Valid Service Specification Codes (SSCs) assigned to a SIC
SSC_PARAM	Semi-Static	Default parameter values for an SSC
STATE_VECTOR	Dynamic	NCCDS or DAS State Vector submissions
SUPIDEN	Semi-Static	Valid Support Identifiers (SUPIDENs) for a SIC
SV_STATUS	Static	State Vector status codes (Transmitted, Queued, etc.)
SWSI_USER	Semi-Static	SWSI Client user information
SWSI_USER_SIC	Semi-Static	SWSI Client user SIC authorizations
TDRS_GROUP	Semi-Static	Valid TDRS group/set names
TDRS_IN_GROUP	Semi-Static	TDRS group/name assignments
TDRS_NAME	Semi-Static	Valid TDRS names
UPD	Static	User Performance Data (UPD) definitions
UPD_ENUM_VALIDATION	Static	Valid values for enumerated UPD parameters
UPD_LABEL	Static	Freetext labels for UPD detail displays
UPD_NUMERIC_VALIDATION	Static	Range check levels for UPD parameters
UPD_PARAM	Static	UPD parameter definitions
USER_LOGIN	Dynamic	Information about SWSI Client users who are or were logged in
USR_GCMR	Dynamic	User reconfiguration requests (98/04) sent to NCCDS
VALUE_TYPE	Static	Service parameter value types (enumerated, numeric, time, etc)
WAITLIST	Dynamic	NCCDS Wait List Requests

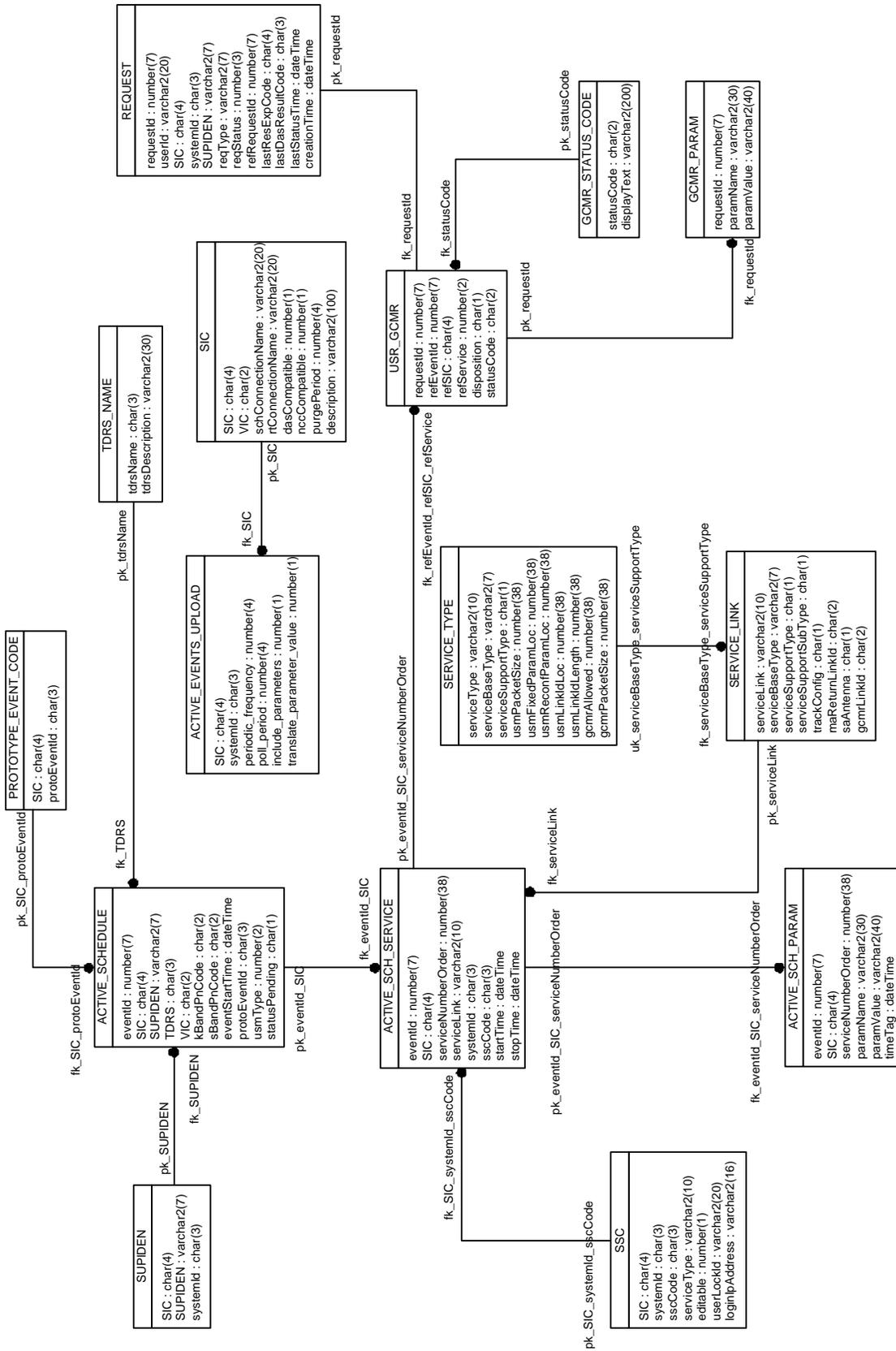


Figure 6-2. SWSI Database Schema (2 of 3)

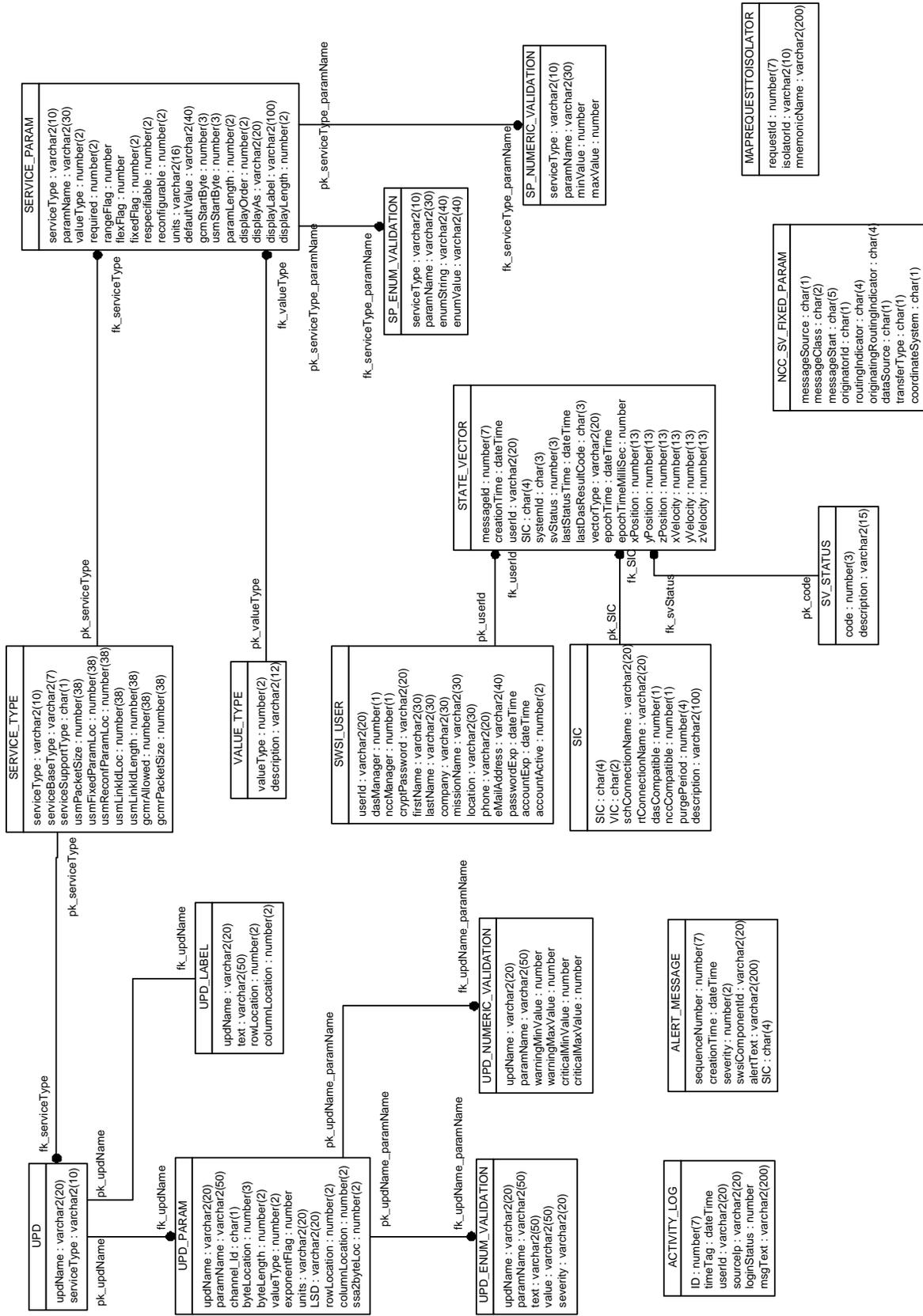


Figure 6-3. SWSI Database Schema (3 of 3)

6.2 Server Configuration

6.2.1 Overview

The 64-bit version of Oracle 8.1.6 Enterprise Edition (Oracle 8i Release 2) is installed on the internal disk drive of each backend server in the following directory:

```
/export/home/orasw/app/oracle/product/8.1.6
```

The following Oracle products are installed:

- Oracle 8 Database Server
- PL/SQL
- Oracle Server Manager
- SQL*Net
- SQL*Plus
- SQL*Loader
- Oracle JDBC Driver

Four database instances (OPS, EIF, OPS2, EIF2) are stored on the RAID disk so that it can be shared between both backend servers. Only the prime server as designated by the High Availability (HA) application has access at any one moment to the RAID and subsequently to the database instances. The instances were created using the *dbassist* tool provided by Oracle and are all the same size. *dbassist* provides a Graphical User Interface (GUI) for creating database instances and is documented in the Oracle 8.1.6 Enterprise Manager documentation.

6.2.2 Parameter Files

A set of parameter files as follows is included with the servers:

```
/export/home/orasw/app/oracle/admin/EIF/pfile/initEIF.ora  
/export/home/orasw/app/oracle/admin/EIF2/pfile/initEIF2.ora  
/export/home/orasw/app/oracle/admin/OPS/pfile/initOPS.ora  
/export/home/orasw/app/oracle/admin/OPS2/pfile/initOPS2.ora
```

These files are used to configure Oracle at startup time by the HA application

6.2.3 System Parameters

The following system parameters in the */etc/system* file are used by Oracle:

```
set shmsys:shminfo_shmmax=4294967295
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=100
set shmsys:shminfo_shmseg=10
set semsys:seminfo_semmni=100
set semsys:seminfo_semmsl=100
set semsys:seminfo_semmns=600
set semsys:seminfo_semopm=100
set semsys:seminfo_semvmx=32767
```

These parameters are the Solaris Operating System kernel parameters needed by Oracle to operate properly and efficiently.

6.2.4 SQL*Net Configuration Files

The Oracle SQL*Net configuration files (*sqlnet.ora*, *tnsnames.ora*, and *listener.ora*) modified for SWSI are included with the servers. The SQL*Net configuration files are located in the *\$ORACLE_HOME/network/admin* directory. The SQL*Net uses these files to configure the listener process. The listener process listens for requests to connect to a database instance. Once a request comes in, the listener establishes a connection to an Oracle server process and goes back to a listen state. SWSI communicates with the Oracle Database Server through SQL*Net via JDBC driver.

Following is an example of an entry in the *tnsnames.ora* file for the OPS instance on one of the backend servers:

```
OPS =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = reptar)(PORT = nnnn))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = OPS)
    )
  )
```

6.2.5 Table Spaces

Table spaces in Oracle are the logical names of where the data is stored. Each table space has a single data file that it references. The names of these seven table spaces are as follows:

- SYSTEM - used by Oracle for its data dictionary
- TEMP - used by Oracle for sorts request by the user

- TOOLS – Oracle tools
- RBS - used by Oracle for performing rollback functions
- SWSI_STATIC_DATA_TS – SWSI database static data, or data that doesn't change often
- SWSI_DATA_TS – SWSI database non-static data
- SWSI_INDEX_TS – SWSI database indices

The data files are the physical files where Oracle stores its data. Each instance has its own set of seven data files. These data files reside on the RAID. The RAID is divided into seven logical volumes. The logical volumes are mounted as shown in Table 6-2.

Table 6-2. Logical Volumes

Logical Volume	RAID Mount Point	Contents
LU0	/mnt/data0	SYSTEM, TOOLS, Control Files Set 1
LU1	/mnt/data1	RBS, Control Files Set 2
LU2	/mnt/data2	SWSI_DATA_TS, Control Files Set3
LU3	/mnt/data3	SWSI_INDEX_TS
LU4	/mnt/data4	REDO Log Files, LOG Files
LU5	/mnt/data5	SWSI_STATIC_DATA_TS, TEMP, archive files
LU6	/mnt/data6	SWSI Database backups

6.2.6 Control Files

Control files are used by Oracle to keep track of all the data files in an instance. The control file keeps track of system changes to all the files so that they are kept in sync. Oracle provides a way to mirror control files so that if one is lost or damaged a DBA can restore from the mirrored control file. In the four SWSI instances, the control files have been mirrored in two other locations besides the original. The control files for four SWSI database instances are as follows:

```

/mnt/data0/control1
EIF EIF2 OPS OPS2
/mnt/data0/control1/EIF/control01.ct1
/mnt/data0/control1/EIF2/control01.ct1
/mnt/data0/control1/OPS/control01.ct1
/mnt/data0/control1/OPS2/control01.ct1

/mnt/data1/control2
EIF EIF2 OPS OPS2
/mnt/data1/control2/EIF/control02.ct1
/mnt/data1/control2/EIF2/control02.ct1
/mnt/data1/control2/OPS/control02.ct1
/mnt/data1/control2/OPS2/control02.ct1

```

```
/mnt/data2/control3
EIF EIF2 OPS OPS2
/mnt/data2/control3/EIF/control03.ctl
/mnt/data2/control3/EIF2/control03.ctl
/mnt/data2/control3/OPS/control03.ctl
/mnt/data2/control3/OPS2/control03.ctl
```

6.2.7 Redo Logs

Oracle uses redo logs to record every transaction that is performed against the database tables. This information is used by Oracle to recover the database in case of failure. At least two redo logs are needed in order for Oracle to run. Oracle writes to these redo logs in a circular fashion. When one redo log file fills up, Oracle starts to write to the next one. In the SWSI environment, there are three redo log groups. Each log group has two files attached. Mirroring the redo logs for each instance helps prevent an Oracle shutdown if one of the redo logs becomes damaged. Also having three redo log groups allows Oracle time to commit work in the first redo log before it is overwritten. The location of the OPS redo logs is:

```
/mnt/data4/OPS
```

The other instance redo logs are in:

```
/mnt/data4/redo/EIF
/mnt/data4/redo/EIF2
/mnt/data4/redo/OPS2
```

Oracle also allows archival of the redo logs. Running each instance in archive mode provides the ability to make hot backups of the databases. These files are used to recover the database instance in case of an Oracle server crash and loss of files. Oracle applies the archived files to the data files that were last saved to bring the files to a current status. The archive files are archived to:

```
/mnt/data5/archredo/EIF/arch
/mnt/data5/archredo/EIF2/arch
/mnt/data5/archredo/OPS/arch
/mnt/data5/archredo/OPS2/arch
```

6.2.8 Rollback Segments

Rollback segments are used by the Oracle server process to keep track of changes being made in the system. If a transaction does not commit its work for any reason (e.g., system crash, software aborts), Oracle uses the information in the rollback segment to restore data to its original state. There are seven rollback segments in each of the four SWSI Oracle instances. One on each instance is the system rollback segment. The user processes use the other six rollback segments. The names of these rollback segments are SYSTEM, RBS0, RBS1, RBS2, RBS3, RBS4, RBS5 and RBS6. They all use the RBS table space and are contained in the data file attached to the table space.

6.3 Backup and Recovery

6.3.1 Hot Backup

Hot backups of the database instances are performed under *cron* control. The hot backups allow users to access the database while it is being backed up. Oracle uses its rollback segments to keep track of the user changes while a table space is in backup mode.

The script `/export/home/oracle/backupscripsh/hotbackup.sh` performs hot backups of the four database instances. This script overwrites the last backup that was done on the RAID. The hot backup script puts each table space into a backup mode and copies all the SWSI database files (.dbf) to its backup location on the RAID. It then performs a log switch so the archive files are current and makes a list of archive files to move to their backup location. Finally it creates a backup control file to its backup location on the RAID.

The four SWSI database files are backed up to the following locations:

```
/mnt/data6/backups/EIF2BU
/mnt/data6/backups/EIFBU
/mnt/data6/backups/OPS2BU
/mnt/data6/backups/OPSB
```

This allows a backup copy of all the database files to be on disk. Having a backup copy of the data files on disk allows for a faster recovery if needed. Once the database files are stored to disk on the RAID, other scripts will copy them to the internal disk of each server and then to tape. This feature is desirable in case of loss of the entire RAID. The scripts to copy the data files to the internal disk and then to tape are described in Section 9.4.4.

6.3.2 Troubleshooting

A good place to look for problems with Oracle is in the Oracle alert log that gets created and written to when the Oracle instance is started up. The trace files are also a good source of information. They are generated by the Oracle's background processes. The locations of these files are:

```
/mnt/data5/errorumps/EIF/bdump      -> alert log and trace files
/mnt/data5/errorumps/EIF2/bdump
/mnt/data5/errorumps/OPS/bdump
/mnt/data5/errorumps/OPS2/bdump
/mnt/data5/errorumps/EIF/udump      -> user dump trace files
/mnt/data5/errorumps/EIF2/udump
/mnt/data5/errorumps/OPS/udump
/mnt/data5/errorumps/OPS2/udump
```

Another source of information is to go into server manager line mode (*svrmgrl*) and try to start the instance that is having problems. Oracle will come back with an error message telling the DBA why it cannot open.

6.3.3 Loss of Redo Log

If a redo log is lost prior to archiving it, the database instance will require media recovery, because all the changes recorded in the redo log are lost. All the data files will have to be restored from a backup copy. Oracle's documented procedure for a point in time recovery or until cancel recover will have to be executed by the DBA. Oracle Press documents these procedures in the *Oracle8i Backup and Recovery Handbook*. When opening the database, use the alter database reset logs option. Some data loss will occur with this type of recovery.

6.3.4 Recovery from Operator Error

To recover from accidental deletion of SWSI tables or data, perform a restore using the last good backup. The DBA can recover the database to just before the data was deleted. This is called a point in time recovery. The procedure for doing this type of recovery is documented in the *Oracle8i Backup and Recovery Handbook* by Oracle Press.

6.3.5 Recovery from Server Hardware Crash

In most cases, Oracle can perform its own crash recovery when it is brought back up. Oracle will use its redo log information to roll the database forward to the point of failure. Oracle will then use its roll back segments to roll back any uncommitted transactions at the time of failure. Oracle will then open the database. This is all done automatically.

6.3.6 Recovery from RAID Hardware Crash

The physical database resides on RAID, which consists of multiple disks that are running mirroring and striping. The mirroring should prevent the loss of data. Loss of the RAID would require the Oracle data files and control files to be restored from a backup stored on the internal disk or tape.

6.3.7 Recovery from Loss of Data File(s)

If a single data file is lost, it can be restored from a backup copy. Files should be restored to the locations specified in this document. After restoring the data file, the DBA will have to connect to the server manager line mode as Oracle Unix user and run the recover the procedure for recovering data files, table spaces or database. Detailed procedures for recovering the database instance are in the *Oracle8i Backup and Recovery Handbook* by Oracle Press.

If all data files are lost, they can be restored from their backup copy. Files should be restored to the locations specified in this document. After restoring the data files, the DBA will have to connect to the server manager line mode as Oracle Unix user and run the procedure for recovering the database. Detailed procedures for recovering the database instance are provided in the *Oracle8i Backup and Recovery Handbook* by Oracle Press.

6.3.8 Recovery from Loss of Control File(s)

If Oracle loses one of its control files, it will shut down. To determine what went wrong, check Oracle's alert log. If there is no information in the alert log, then try to start up the database instance. Oracle will tell you why it cannot start. To repair this problem, delete the bad control file, copy one of the mirrored control files that is still good to where the bad control file was, and restart the Oracle instance. This type of recovery will work for control files as long as you have one good control file.

If all control files are lost, the following scripts can be run to create new control files:

```
/export/home/oracle/createBuCTRLEIF.sql  
/export/home/oracle/createBuCTRLEIF2.sql  
/export/home/oracle/createBuCTRLOPS.sql  
/export/home/oracle/createBuCTRLOPS2.sql
```

These scripts would be run from the server manager line mode (i.e., *svrmgrl*).

6.4 Failover

SWSI database files reside on the shared disk (RAID). When SWSI fails over from one server to another server by HA, the RAID gets un-mounted from the primary server and gets mounted on the backup server. The SWSI database instances are then brought up after a oracle database is started

6.5 Oracle Accounts

The SWSI database is configured with the following accounts:

- SWSIDB – owns the schema and has full privileges.
- SWSIOPS – readonly account for SWSI operator access. An operator may use this account with the Database Administration tool described in Section 7 for viewing data and system status, but not to modify data. The operator is not required to enter a password when selecting the DBA Tool (readonly) CDE menu option as described in Section 4
- ORASWSI – used by SWSI applications (Isolator, SNIF, SDIF) to access tables. The username and password for this account are entered into the property or configuration files for these applications as described in Section 3.
- DBA Accounts – these accounts have update, insert, and delete privileges and are assigned to individual Database Administrators for use in updating the database using the Database Administration tool described in Section 7.

Section 7. Database Administration

7.1 Overview

The SWSI Database Administration tool is used by the SWSI Database Administrator (DBA) to maintain customer and user semi-static data in the SWSI database as described in Section 6. The DBA tool is also used to monitor user login and NCCDS connection status. The DBA tool is invoked by selecting the appropriate Common Desktop Environment (CDE) menu options as described in Section 4. In readonly mode, the operator may view data and status, but may not modify data. An Oracle account with update privilege must be used to modify data.

The DBA tool may also be invoked interactively as follows:

```
dba -b <database_name> -u <oracle_username> -p <oracle_password>
```

If invoked with a `-v` option as follows:

```
dba -v
```

Then the DBA tool just displays the version number and exits, similar to the following:

```
SWSI DBA Version: Build 4 Patch 02
```

If invoked with an invalid option, such as follows:

```
dba -h
```

Then the DBA tool displays usage information and exits, similar to the following:

```
SWSI Database Administration Tool

Usage:
  dba [-dhv] -b <database> -u <username> -p <password>

  d = debug mode
  h = help (this message)
  v = display version
  b = Oracle database instance
  u = Oracle username
  p = Oracle password
```

Once invoked, the DBA tool main menu appears as follows:

```
SWSI DBA Version Build 4 Patch 02, EIF database instance
Main Menu

1 = User Administration
2 = NCCDS Schedule Connection Administration
3 = NCCDS Realtime Connection Administration
4 = SIC Administration
```

5 = Prototype Event Code Administration
6 = SUPIDEN Administration
7 = TDRS Name Administration
8 = SSC Administration
9 = Active Schedule Upload Administration

q = Quit

Enter command:

7.2 User Administration

7.2.1 General

The User Administration menu is used for maintaining user accounts and SIC authorizations for each user. This menu is used for maintaining or viewing information from the following Database tables:

- ACTIVITY_LOG - Client login/logout events.
- SWSI_USER – SWSI Client user information.
- SWSI_USER_SIC – SWSI Client user SIC authorizations.
- USER_LOGIN - Information about SWSI Client users who are or were logged in.

Following is the User Administration menu:

User Administration Menu

1 = Display user
2 = Display users logged in
3 = Display activity log
4 = Mark a user as logged off

5 = Deactivate a user
6 = Reactivate a user
7 = Change user privilege
8 = Change user contact information
9 = Set a user's password expiration date
a = Set a user's account expiration date

b = Add a user
c = Change user password
d = Enter encrypted user password
e = Delete a user

f = Display SICs authorized for each user
g = Display users authorized for each SIC
h = Add a SIC for a user
i = Remove a SIC for a user

7.2.2 Display user Menu Option

Selection of the *Display user* menu option provides a listing of either one or all Client user accounts from the SWSI_USER table. Upon selection of this option, the operator is prompted to choose which user to display:

```
Enter User ID (default all):
```

Following is a sample user listing:

```
SWSI Users, EIF database instance
-----
User ID          DAS Mgr?      Password Expiration   Account
Encrypted Password NCC Mgr?     Account Expiration    Active?
Name
Company          Scheduler?
Location         Submit GCMR?
Mission
Phone
Email
-----
sardella                2003/04/21 21:20 (expd)   Y
6pe0mFn7bBcww         2013/02/17 21:20 (3576)
Tom Sardella
NASA/GSFC
Greenbelt, MD        301-286-7686
LDB                  Tom.Sardella@nasa.gov

stevens                Y                2003/06/22 20:04 ( 48)   Y
5IC4dc3oJWzpz        2012/12/07 15:02 (3503)
```

The information displayed is self-explanatory, with the exception of the following:

- DAS Manager – user is allowed to edit Service Specification Code (SSC) default parameter values for DAS SSCs.
- NCC Manager – user is allowed to edit Service Specification Code (SSC) default parameter values for NCC SSCs.
- Scheduler – user is allowed to schedule.
- Submit GCMR – user is allowed to submit GCMR.
- Password Expiration – the user’s password expires on the indicated date. The number in parentheses indicates how many days in the future that the password expires. If already expired, then this is indicated as *expd*.
- Account Expiration - the user’s account expires on the indicated date. The number in parenthesis indicates how many days in the future that the account expires. If already expired, then this is indicated as *expd*.
- Account Active – indicates if the account is active. If not active, then this field is blank. An account may be inactive because it has expired or if there have been too many failed login attempts. The DBA may also manually deactivate an account.

7.2.3 Display users logged in Menu Option

Selection of the *Display users logged in* menu option provides a listing of all Client users who are logged in as indicated in the USER_LOGIN table, such as follows:

SWSI Users Logged In, EIF database instance
Mon May 5 17:10:54 GMT 2003

User ID Login Date IP Address
Server ID Logout Date Failed Attempts

```

sardella      2003/04/28 20:28:03 xxx.xxx.xxx.xxx (xyz.gsfc.nasa.gov)
open          2003/04/28 20:20:22 0

stevens      2003/04/28 20:21:33 yyy.yyy.yyy.yyy (abc.nascom.nasa.gov)
closed       2003/04/28 20:20:51 0

```

Control-C to exit

The display updates automatically every five seconds. Pressing *Control-C* interrupts the display and returns the operator to the User Administration menu. The following information is provided in this display:

- User ID – account name of the logged-in user.
- Server ID – server (*open*, *closed*) that the user is connected to.
- Login Date – date and time of most recent login.
- Logout Date – date and time of most recent logout, which would be for a previous session.
- IP Address – address of Client host from which user is logged in. If a fully qualified domain or host name is available, then this is displayed in parentheses
- Failed Attempts – number of times that user failed to log in prior to the current session, possibly because of an incorrectly entered password.

7.2.4 Display activity log Menu Option

Selection of the *Display activity log* menu option provides a listing of Client user login and logout activity from the ACTIVITY_LOG table. Upon selection of this option, the operator is prompted for which user for which a delog is desired, along with how far back in the log to display:

```

Enter User ID (default all): sardella
Enter how far back in log in days (default all entries): 25

```

Following is a sample activity log listing:

```

SWSI User Activity Log, EIF database instance
Mon May 5 18:55:39 GMT 2003
-----
Time           User ID      Action      IP Address
-----
04/10 21:25:55 sardella    Login      xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/10 21:34:09 sardella    Logout     xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/11 21:19:49 sardella    Login      xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/11 21:37:59 sardella    Logout     xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/17 19:02:10 sardella    Login failed xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/17 19:02:40 sardella    Login      xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
04/17 19:09:14 sardella    Logout     xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
05/05 18:49:51 sardella    Logout     xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
05/05 18:53:53 sardella    Passwd chg request xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
05/05 18:54:03 sardella    Passwd changed xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
05/05 18:54:03 sardella    Login      xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)
05/05 18:54:42 sardella    Logout     xxx.xxx.xxx.xxx (abc.gsfc.nasa.gov)

```

The information displayed is self-explanatory, with the exception of the following:

- Action – consists of one of the following:
 - Login – user successfully logged in.
 - Logout – user successfully logged out.
 - Login failed – user was unable to log in because of an inexistent account, an invalid password, or because the user is already logged in from that IP address (user is allowed to be logged in only once from any one IP address).
 - Logout failed – a system error occurred while trying to log the user off.
 - Passwd chg request – password change requested by user.
 - Passwd changed – password change was successful.
 - Passwd not changed - password change was unsuccessful.
- IP Address – address of Client host from which user is connected. If a fully qualified domain or host name is available, then this is displayed in parentheses

7.2.5 Mark a user as logged off Menu Option

Selection of the *Mark a user as logged off* menu option provides a means of clearing an error condition on the SWSI Server where a user is marked as logged on, but the user is actually no longer connected. Since the server only allows a user to be logged once from any one IP address, the user is subsequently unable to log in. Upon selection of this option, the operator is prompted for which user for to mark as logged off:

```
Enter User ID to log off: sardella
Enter Server ID: open
Enter IP Address: xxx.xxx.xxx.xxx
sardella logged off
```

7.2.6 Deactivate a user Menu Option

Selection of the *Deactivate a user* menu option provides a means of disabling a Client user account. Upon selection of this option, the operator is prompted for which user account to disable:

```
Enter User ID to deactivate: sardella
sardella deactivated
```

7.2.7 Reactivate a user Menu Option

Selection of the *Reactivate a user* menu option provides a means of re-enabling a Client user account. This is necessary as a result of an account being automatically deactivated because of too many failed login attempts, or in case an account needs to be reactivated after being manually deactivated. Upon selection of this option, the operator is prompted for which user account to enable:

```
Enter User ID to reactivate: sardella
sardella reactivated
```

7.2.8 Change user privilege Menu Option

Selection of the *Change user privilege* menu option provides a means of setting whether or not a user may edit SSC default parameter values. Upon selection of this option, the operator is prompted for which privileges to change:

```
Enter User ID: sardella
DAS Manager? (Y/N): y
NCC Manager? (Y/N): y
Scheduler? (Y/N): Y
Submit GCMR? (Y/N): Y
sardella privilege set to DAS Manager 1, NCC Manager 1
```

7.2.9 Change user contact information Menu Option

Selection of the *Change user contact information* menu option provides a means of changing a user's contact information. Upon selection of this option, the operator is prompted for which information to change:

```
Enter User ID: sardella
Enter First Name: Tom
Enter Last Name: Sardella
Enter Company: NASA/GSFC
Enter Mission Name: LDB
Enter Location: Greenbelt, MD
Enter Phone: 301-286-7686
Enter Email Address: Tom.Sardella@nasa.gov
sardella Contact Information updated
```

Pressing *Enter* for any field causes that field to remain unchanged.

7.2.10 Set a user's password expiration date Menu Option

Selection of the *Set a user's password expiration date* menu option provides a means of setting the date when a user's password will expire. Upon selection of this option, the operator is prompted for which user account to change and when to change it to:

```
Enter User ID: sardella
Enter days until expiration: 30
sardella password expiration date set
```

Enter a value of 0 for the number of days until expiration causes the password to be immediately expired. The user will still be able to log in, but will be prompted to change his password immediately.

7.2.11 Set a user's account expiration date Menu Option

Selection of the *Set a user's account expiration date* menu option provides a means of setting the date when a user's account will expire. Upon selection of this option, the operator is prompted for which user account to change and when to change it to:

```
Enter User ID: sardella
Enter days until expiration: 365
sardella account expiration date set
```

Enter a value of 0 for the number of days until expiration causes the account to be immediately expired, meaning that the user will no longer be able to log in.

7.2.12 Add a user Menu Option

Selection of the *Add a user* menu option provides a means of adding a new Client user account. Upon selection of this option, the operator is prompted for information about the new user account:

```
Enter User ID: sardella
Enter Password (default same as username):
Enter First Name: Tom
Enter Last Name: Sardella
Enter Company: NASA/GSFC
Enter Mission Name: LDB
Enter Location: Greenbelt
Enter Phone: 301-286-7686
Enter Email Address: Tom.Sardella@nasa.gov
Enter days until password expiration (default expired):
Enter days until account expiration (default 10 years):

Please wait while password is encrypted...

User sardella added
```

Contact information (First Name, Last Name, Company, Mission Name, Location, Phone, Email Address) is optional, though it is highly recommended that this information be entered. It is also recommended that the password expiration initially be set to expired so that the user is forced to enter a new password on initial login that meets the password criteria set by the SWSI applications.

7.2.13 Change user password Menu Option

Selection of the *Change user password* menu option provides a means of changing or resetting a user's password. Upon selection of this option, the operator is prompted for the name of the user account to change and the new password:

```
Enter User ID: sardella
Enter Password (default same as username):
Verify Password:

Please wait while password is encrypted...

sardella password changed
```

If the operator enters a password rather than allowing use of the default, then the operator is required to enter the password twice for verification.

7.2.14 Enter encrypted user password Menu Option

Selection of the *Change encrypted user password* menu option provides a means of changing a user's password to the encrypted value that is normally stored in the SWSI_USER table. This is useful when copying password between Database instances. Upon selection of this option, the operator is prompted for the name of the user account to change and the new encrypted password:

```
Enter User ID: sardella
Enter encrypted password: f3q2LJxlmOTAk
sardella password changed
```

7.2.15 Delete a user Menu Option

Selection of the *Delete a user* menu option provides a means of removing an existing Client user account. Upon selection of this option, the operator is prompted for the name of the user account to remove:

```
Enter User ID: sardella
Are you sure you want to remove user sardella?
This will also remove all associated records in USER_LOGIN and SWSI_USER_SIC
Enter Y to confirm: y
User sardella deleted
```

7.2.16 Display SICs authorized for each user Menu Option

Selection of the *Display SICs authorized for each user* menu option provides a listing of SICs authorized for each Client user as indicated in the SWSI_USER_SIC table, such as follows:

```
SWSI User SIC Assignments, EIF database instance
-----
User ID          SIC
-----
sardella         0338
                 1294
                 1398
stevens          1294
                 1419
                 1446
                 3301
```

7.2.17 Display users authorized for each SIC Menu Option

Selection of the *Display users authorized for each SIC* menu option provides a listing of Client users authorized for each SIC as indicated in the SWSI_USER_SIC table, such as follows:

```
SWSI SIC User Assignments, EIF database instance
-----
SIC  User ID
-----
0338 sardella

1294 sardella
     stevens

1398 sardella

1419 stevens

1446 stevens

3301 stevens
```

7.2.18 Add a SIC for a user Menu Option

Selection of the *Add a SIC for a user* menu option provides a means of authorizing a new SIC for a Client user account. Upon selection of this option, the operator is prompted for the name of the user account to change and the SIC to add:

```
Enter User ID: sardella
Enter SIC: 0338
SIC 0338 added for user sardella
```

7.2.19 Remove a SIC for a user Menu Option

Selection of the *Remove a SIC for a user* menu option provides a means of removing authorization for an existing SIC for a Client user account. Upon selection of this option, the operator is prompted for the name of the user account to change and the SIC to remove:

```
Enter User ID: sardella
Enter SIC: 0338
SIC 0338 deleted for user sardella
```

7.3 NCCDS Schedule Connection Administration

7.3.1 General

The NCCDS Schedule Connection Administration menu is used for maintaining configuration information for the SNIF to use in establishing and maintaining scheduling, state vector storage, and TDRS Scheduling Window (TSW) TCP connections with NCCDS. This menu is used for maintaining or viewing information from the following Database tables:

- SIC (schConnectionName column) - Support Identification Codes (SICs) for spacecraft supported by SWSI.
- SCHEDULE_CONNECTION – NCCDS scheduling connection (schReq, schStatus, etc.) configuration.

Following is the Schedule Connection Administration menu:

```
NCCDS Schedule Connection Administration Menu

1 = Display Schedule Connections
2 = Monitor Connection Status
3 = Add Schedule Connection
4 = Delete Schedule Connection
5 = Update Schedule Connection

6 = Display Schedule Connection SIC Assignments
7 = Add a SIC to a Schedule Connection
8 = Remove a SIC from a Schedule Connection

9 = Enable Schedule Status Connection
a = Disable Schedule Status Connection
```

7.3.2 Display Schedule Connections Menu Option

Selection of the *Display Schedule Connections* menu option provides a listing of all Schedule Connections from the SCHEDULE_CONNECTION table. Each Schedule Connection actually refers to a group of connections that are established with the NCCDS Service Planning Segment (SPS). Following is a sample listing:

Schedule Connections, EIF database instance

```
-----
Name                Logical Dest      User          schReq  acqStore  tswStore  schReq  tswStore
                    ID    Pass EN?  Timeout Timeout  Timeout Meter  Meter
-----
GP-B Scheduling     PROBE            ABC  DEFG  Y   600    600    600    1000  5000
LSAT-4/5 Scheduling LSM              HIJ  KLMN  600    600    600    1000  5000
```

The following information is provided in this display:

- Name – operator-assigned Schedule Connection name.
- Logical Dest – logical destination name, as defined in Table 7-8, Item 7 of the *NCCDS/MOC ICD*. This value is automatically inserted by SNIF into Schedule Result Request messages (99/28) sent to NCCDS.
- User ID – schedule request message Requester ID, as defined in Table 7-1, Item 5 of the *NCCDS/MOC ICD*. This value is automatically inserted by SNIF into Schedule Request messages (SAR, ASAR, etc.) sent to NCCDS.
- Pass – schedule request message Password, as defined in Table 7-1, Item 6 of the *NCCDS/MOC ICD*. This value is automatically inserted by SNIF into Schedule Request messages (SAR, ASAR, etc.) sent to NCCDS.
- Enabled – indicates whether the Schedule Status (schStatus) connection is enabled. This connection may be turned on and off in realtime by the operator to allow a customer to switch between using different scheduling systems; e.g., SWSI vs. the User Planning System (UPS). SPS does not support multiple scheduling systems to be simultaneously connected on the schStatus service.
- schReq Timeout – time in seconds after the last schedule request message (SAR, etc.) is transmitted to close the Schedule Request (schReq) connection.
- acqStore Timeout - time in seconds after the last state vector message is transmitted to close the Vector Storage (acqStore) connection.
- tswStore Timeout - time in seconds after the last TSW message is transmitted to close the TSW Storage (tswStore) connection.
- schReq Meter – time delay in milliseconds between successive transmissions of schedule request messages on the schReq connection.
- tswStore Meter - time delay in milliseconds between successive transmissions of TSW messages on the tswStore connection.

7.3.3 Monitor Connection Status Menu Option

Selection of the *Monitor Connection Status* menu option provides status information about all NCCDS connections, including realtime connections described in Section 7.4. Following is a sample display:

```

SWSI Connection Status, EIF database instance
Tue May 6 00:22:07 GMT 2003
-----
Name                schReq  schStatus  acqStore  tswStore  reconfig  pmData
-----
```

```

GP-B Scheduling                Active
LSAT-4/5 Scheduling

GP-B Realtime                  Active   Active
LSAT-4/5 Realtime              Active   Active

Control-C to exit

```

The display updates automatically every five seconds. Pressing *Control-C* interrupts the display and returns the operator to the Schedule Connection Administration menu. An Active status indicates that a TCP connection is established with NCCDS.

7.3.4 Add Schedule Connection Menu Option

Selection of the *Add Schedule Connection* menu option provides a means of adding a new Schedule Connection entry. Upon selection of this option, the operator is prompted for information about the new entry:

```

Enter Schedule Connection Name: LSAT-4/5 Scheduling
Enter Logical Destination Name: LSM
Enter NCC User ID: HIJ
Enter NCC Password: KLMN
Enter schReq Timeout (sec) (default 600):
Enter acqStore Timeout (sec) (default 600):
Enter tswStore Timeout (sec) (default 600):
Enter schReq Meter (msec) (default 1000):
Enter tswStore Meter (msec) (default 5000):

Schedule Connection LSAT-4/5 Scheduling added

EIF SNIF must be restarted for this change to take effect

```

Each field entry is described in Section 7.3.2 above. After entry is complete and SICs are assigned to the entry (see Section 7.3.8 below), the appropriate SNIF must be restarted using the procedure described in Section 4.2. Note also that new entries are added with the Schedule Status connection disabled. Section 7.3.10 describes how to enable that connection.

7.3.5 Delete Schedule Connection Menu Option

Selection of the *Delete Schedule Connection* menu option provides a means of removing an existing Schedule Connection entry. Upon selection of this option, the operator is prompted for the name of the Schedule Connection to remove:

```

Enter Schedule Connection Name: LSAT-4/5 Scheduling
Are you sure you want to remove Schedule Connection LSAT-4/5 Scheduling?
This will also remove all SIC assignments for that connection.
Enter Y to confirm: y

Schedule Connection LSAT-4/5 Scheduling deleted

EIF SNIF must be restarted for this change to take effect

```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.3.6 Update Schedule Connection Menu Option

Selection of the *Update Schedule Connection* menu option provides a means of changing the settings of an existing Schedule Connection entry. Upon selection of this option, the operator is prompted for which information to change:

```
Enter Schedule Connection Name: LSAT-4/5 Scheduling
Enter Logical Destination Name: newname
Enter NCC User ID:
Enter NCC Password:
Enter schReq Timeout (sec):
Enter acqStore Timeout (sec):
Enter tswStore Timeout (sec):
Enter schReq Meter (msec):
Enter tswStore Meter (msec):

Schedule Connection LSAT-4/5 Scheduling updated

EIF SNIF must be restarted for this change to take effect
```

Pressing *Enter* for any field causes that field to remain unchanged. In this case, only the Logical Destination Name was changed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.3.7 Display Schedule Connection SIC Assignments Menu Option

Selection of the *Display Schedule Connection SIC Assignments* menu option provides a listing of SICs assigned for each Schedule Connection entry, such as follows:

```
Schedule Connection SIC Assignments, EIF database instance
-----
Schedule Connection  SIC
-----
GP-B Scheduling      8603

LSAT-4/5 Scheduling  1294
                    1419
```

7.3.8 Add a SIC to a Schedule Connection Menu Option

Selection of the *Add a SIC to a Schedule Connection* menu option provides a means of assigning a new SIC for a Schedule Connection entry. Upon selection of this option, the operator is prompted for the name of the Schedule Connection to change and the SIC to add:

```
Enter Schedule Connection Name: LSAT-4/5 Scheduling
Enter SIC: 1419
SIC 1419 added to Schedule Connection LSAT-4/5 Scheduling

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.3.9 Remove a SIC from a Schedule Connection Menu Option

Selection of the *Remove a SIC from a Schedule Connection* menu option provides a means of deleting a SIC from a Schedule Connection entry. Upon selection of this option, the operator is prompted for the name of the Schedule Connection to change and the SIC to remove:

```
Enter Schedule Connection Name: LSAT-4/5 Scheduling
Enter SIC: 1419
SIC 1419 removed from Schedule Connection LSAT-4/5 Scheduling

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.3.10 Enable Schedule Status Connection Menu Option

Selection of the *Enable Schedule Status Connection* menu option provides a means of enabling the schStatus connection for a Schedule Connection entry in realtime. This allows a customer to switch between using different scheduling systems; e.g., SWSI vs. the User Planning System (UPS). SPS does not support multiple scheduling systems to be simultaneously connected on the schStatus service and care must be taken by the customer working together with the SWSI Server operator to ensure that this doesn't happen. Upon selection of this option, the operator is prompted for the name of the Schedule Connection for which schStatus should be enabled:

```
Enter Schedule Connection to enable: LSAT-4/5 Scheduling
LSAT-4/5 Scheduling Schedule Status Connection enabled
```

After the operation is complete, SNIF may take as long as one minute to establish the connection with NCCDS.

7.3.11 Disable Schedule Status Connection Menu Option

Selection of the *Disable Schedule Status Connection* menu option provides a means of disabling the schStatus connection for a Schedule Connection entry in realtime. Upon selection of this option, the operator is prompted for the name of the Schedule Connection for which schStatus should be disabled:

```
Enter Schedule Connection to disable: LSAT-4/5 Scheduling
LSAT-4/5 Scheduling Schedule Status Connection disabled
```

After the operation is complete, SNIF may take as long as one minute to break the connection with NCCDS.

7.4 NCCDS Realtime Connection Administration

7.4.1 General

The NCCDS Realtime Connection Administration menu is used for maintaining configuration information for the SNIF to use in establishing and maintaining Ground Control Message Request (GCMR) (reconfig) and performance data monitoring (pmData) TCP connections with NCCDS. This menu is used for maintaining or viewing information from the following Database tables:

- SIC (rtConnectionName column) - Support Identification Codes (SICs) for spacecraft supported by SWSI.
- REALTIME_CONNECTION – NCCDS realtime connection (reconfig, pmData) configuration.

Following is the Realtime Connection Administration menu:

```
NCCDS Realtime Connection Administration Menu

1 = Display Realtime Connections
2 = Monitor Connection Status
3 = Add Realtime Connection
4 = Delete Realtime Connection
5 = Update Realtime Connection

6 = Display Realtime Connection SIC Assignments
7 = Add a SIC to a Realtime Connection
8 = Remove a SIC from a Realtime Connection
```

7.4.2 Display Realtime Connections Menu Option

Selection of the *Display Realtime Connections* menu option provides a listing of all Realtime Connections from the REALTIME_CONNECTION table. Each Realtime Connection actually refers to a group of connections that are established with the NCCDS Protocol Gateway (NPG) acting as a proxy TCP server for the Communications and Control Segment (CCS). Following is a sample listing:

```
Realtime Connections, EIF database instance
-----
Name                User      UPDR
                   ID       Pass Meter
-----
GP-B Realtime       ABC  DEFG 1000
LSAT-4/5 Realtime   HIJ  KLMN 1000
```

The following information is provided in this display:

- Name – operator-assigned Realtime Connection name.
- User ID – realtime request message Requester ID, as defined in Table 8-1, Item 5 of the *NCCDS/MOC ICD*. This value is automatically inserted by SNIF into GCMR and User Performance Data Request (UPDR) messages sent to NCCDS.
- Pass – realtime request message Password, as defined in Table 8-1, Item 6 of the *NCCDS/MOC ICD*. This value is automatically inserted by SNIF into GCMR and User Performance Data Request (UPDR) messages sent to NCCDS.
- UPDR Meter – time delay in milliseconds between successive transmissions of UPDR messages on the pmData connection.

7.4.3 Monitor Connection Status Menu Option

Selection of the *Monitor Connection Status* menu option provides status information about all NCCDS connections, including scheduling connections described in Section 7.3. Following is a sample display:

```
SWSI Connection Status, EIF database instance
Tue May 6 00:22:07 GMT 2003
-----
Name                schReq  schStatus  acqStore  tswStore  reconfig  pmData
-----
GP-B Scheduling                Active
LSAT-4/5 Scheduling

GP-B Realtime                                Active   Active
LSAT-4/5 Realtime                        Active   Active

Control-C to exit
```

The display updates automatically every five seconds. Pressing *Control-C* interrupts the display and returns the operator to the Realtime Connection Administration menu. An Active status indicates that a TCP connection is established with NCCDS.

7.4.4 Add Realtime Connection Menu Option

Selection of the *Add Realtime Connection* menu option provides a means of adding a new Realtime Connection entry. Upon selection of this option, the operator is prompted for information about the new entry:

```
Enter Realtime Connection Name: LSAT-4/5 Realtime
Enter NCC User ID: HIJ
Enter NCC Password: KLMN
Enter UPDR Meter (msec) (default 1000):

Realtime Connection LSAT-4/5 Realtime added

EIF SNIF must be restarted for this change to take effect
```

Each field entry is described in Section 7.4.2 above. After entry is complete and SICs are assigned to the entry (see Section 7.4.8 below), the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.4.5 Delete Realtime Connection Menu Option

Selection of the *Delete Realtime Connection* menu option provides a means of removing an existing Realtime Connection entry. Upon selection of this option, the operator is prompted for the name of the Realtime Connection to remove:

```
Enter Realtime Connection Name: LSAT-4/5 Realtime
Are you sure you want to remove Realtime Connection LSAT-4/5 Realtime?
This will also remove all SIC assignments for that connection.
Enter Y to confirm: y

Realtime Connection LSAT-4/5 Realtime deleted

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.4.6 Update Realtime Connection Menu Option

Selection of the *Update Realtime Connection* menu option provides a means of changing the settings of an existing Realtime Connection entry. Upon selection of this option, the operator is prompted for which information to change:

```
Enter Realtime Connection Name: LSAT-4/5 Realtime
Enter NCC User ID:
Enter NCC Password:
Enter UPDR Meter (msec): 500

Realtime Connection LSAT-4/5 Realtime updated

EIF SNIF must be restarted for this change to take effect
```

Pressing *Enter* for any field causes that field to remain unchanged. In this case, only the UPDRMeter was changed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.4.7 Display Realtime Connection SIC Assignments Menu Option

Selection of the *Display Realtime Connection SIC Assignments* menu option provides a listing of SICs assigned for each Realtime Connection entry, such as follows:

```
Realtime Connection SIC Assignments, EIF database instance
-----
Realtime Connection  SIC
-----
GP-B Realtime        8603
LSAT-4/5 Realtime    1294
                    1419
```

7.4.8 Add a SIC to a Realtime Connection Menu Option

Selection of the *Add a SIC to a Realtime Connection* menu option provides a means of assigning a new SIC for a Realtime Connection entry. Upon selection of this option, the operator is prompted for the name of the Realtime Connection to change and the SIC to add:

```
Enter Realtime Connection Name: LSAT-4/5 Realtime
Enter SIC: 1419
SIC 1419 added to Realtime Connection LSAT-4/5 Realtime

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.4.9 Remove a SIC from a Realtime Connection Menu Option

Selection of the *Remove a SIC from a Realtime Connection* menu option provides a means of deleting a SIC from a Realtime Connection entry. Upon selection of this option, the operator is prompted for the name of the Realtime Connection to change and the SIC to remove:

```
Enter Realtime Connection Name: LSAT-4/5 Realtime
Enter SIC: 1419
SIC 1419 removed from Realtime Connection LSAT-4/5 Realtime

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.5 SIC Administration

7.5.1 General

The SIC Administration menu is used for maintaining the list of spacecraft supported by SWSI and for manually purging schedule requests and confirmed (active) events. This menu is used for maintaining or viewing information from the following Database tables:

- ACTIVE_SCHEDULE - Active (confirmed) events, derived from User Schedule Messages (USMs) received from NCCDS.
- REQUEST – NCCDS and DAS schedule requests.
- SIC (schConnectionName column) - Support Identification Codes (SICs) for spacecraft supported by SWSI.
- SUPIDEN - Valid Support Identifiers (SUPIDENs) for a SIC.

Following is the SIC Administration menu:

```
SIC Administration Menu

1 = Display SICs
2 = Update a SIC
3 = Add a SIC
4 = Remove a SIC
5 = Purge schedule requests for a SIC
6 = Delete a specific schedule request
7 = Purge all schedule requests for a SIC
8 = Purge active events for all SICs
```

7.5.2 Display SICs Menu Option

Selection of the *Display SICs* menu option provides a listing of all SICs from the SIC table, such as follows:

```
SICs, EIF database instance
-----
SIC  Description          VIC  Schedule Connection  DAS NCC Purge
      Description          Realtime Connection  Y   Y   Period
-----
1294 Landsat 4           01  LSAT-4/5 Scheduling   Y   Y   120
      Landsat 4           Realtime
1419 Landsat 5           01  LSAT-4/5 Scheduling   Y   Y   120
      Landsat 5           Realtime
8603 Gravity Probe B    01  GP-B Scheduling       Y   Y   120
      Gravity Probe B    Realtime
```

The following information is provided in this display:

- SIC – SIC code.
- Description – operator-assigned spacecraft or mission description.
- VIC – Vehicle Identification Code, as defined in Table 9-2, Item 17 of the *NCCDS/MOC ICD*. This value is automatically inserted into State Vector messages sent to NCCDS.
- Schedule Connection – Schedule Connection that this SIC is assigned to. See Section 7.3.
- Realtime Connection – Realtime Connection that this SIC is assigned to. See Section 7.4.
- DAS – indicates whether or not SIC is supported by DAS services.
- NCC – indicates whether or not SIC is supported by NCC services.
- Purge Period – when automatically purging Schedule Requests, delete those requests whose requested start time is older than this number of days.

7.5.3 Update a SIC Menu Option

Selection of the *Update a SIC* menu option provides a means of changing the settings of an existing SIC. Upon selection of this option, the operator is prompted for which information to change:

```
Enter SIC: 1294
Enter VIC:
DAS Compatible (Y/N): n
NCC Compatible (Y/N):
Enter Schedule Request Purge Period (days):
Enter Description:
SIC 1294 updated

EIF SNIF must be restarted for this change to take effect
```

Pressing *Enter* for any field causes that field to remain unchanged. In this case, only the DAS Compatible flag was changed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.5.4 Add a SIC Menu Option

Selection of the *Add a SIC* menu option provides a means of adding a new SIC. Upon selection of this option, the operator is prompted for information about the new entry:

```
Enter SIC: 1294
Enter VIC: 01
DAS Compatible (N): n
NCC Compatible (N): y
Enter Schedule Request Purge Period (days) (default 0): 90
Enter Description: Landsat 4
SIC 1111 added

EIF SNIF must be restarted for this change to take effect
```

Each field entry is described in Section 7.5.2 above. Regardless of whether the SIC is indicated as DAS Compatible, a DAS SUPIDEN of XnnnnNA (e.g., X1294NA) is always added. This helps to avoid a problem where a Resource Allocation Request is lost due to the DBA neglecting to enter a SUPIDEN value for a new SIC. After entry is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.5.5 Remove a SIC Menu Option

Selection of the *Remove a SIC* menu option provides a means of removing an existing SIC. Upon selection of this option, the operator is prompted for the SIC code to remove:

```
Enter SIC: 1294
Are you sure you want to remove SIC 1294?
This will also remove all associated SSCs, Schedule Requests, and Active Events.
Enter Y to confirm: y
Are you REALLY SURE you want to do this? This removal is not reversible.
Enter Y to confirm: y
SIC 1294 deleted

EIF SNIF must be restarted for this change to take effect
```

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.5.6 Purge schedule requests for a SIC Menu Option

Selection of the *Purge schedule requests for a SIC* menu option provides a means of purging old Schedule Requests from the Database. This helps to keep the number of requests that a Client user views in the Schedule Request Summary panel to a manageable level. Normally this purging is performed automatically by a cron job based on the Purge Period parameter in the SIC table, but it may also be performed manually from the DBA tool using this option. Upon selection of this option, the operator is prompted for which SIC to purge and how long ago to purge:

```
Enter SIC (default all): 1294
Enter how long ago to purge (days): 60
Schedule Requests prior to 60 days ago purged for SIC 1294
```

Schedule Requests whose requested start time is older than requested purge time are deleted.

7.5.7 Delete a specific schedule request Menu Option

Sometimes it is necessary to manually delete selected Schedule Requests, such as when a request is erroneously submitted with a start time in the distant future. In a case such as this, the normal purge process does not work because it will only purge requests that are in the past. The *Delete a specific schedule request* menu option allows an operator to do this. Upon selection of this option, the operator is prompted for the ID of the request to delete:

```
Enter Request ID: 800541
Request ID 800541 successfully deleted
```

7.5.8 Purge all schedule requests for a SIC Menu Option

Selection of the *Purge all schedule requests for a SIC* menu option is similar to the option described in Section 7.5.6, except that no purge period is specified. All schedule requests are purged, regardless of the requested start time. Upon selection of this option, the operator is prompted for which SIC to purge:

```
Enter SIC (default all): 1294
Are you sure you want to purge all Schedule Requests for SIC 1294?
Enter Y to confirm: y
All Schedule Requests purged for SIC 1294
```

7.5.9 Purge active events for all SICs Menu Option

Selection of the *Purge active events for all SICs* menu option provides a means of purging old Active Events from the Database. This has no effect on the Client users since they can only view confirmed events in the future, but it does help to reduce Database size and improve performance by eliminating old, unneeded data. Normally this purging is performed automatically by a cron job, but it may also be performed manually from the DBA tool using this option. Upon selection of this option, the operator is prompted for how long ago to purge:

```
Enter how long ago to purge (days): 60
Active Events prior to 60 days ago purged for all SICs
```

Active Events whose Event Start Time is older than requested purge time are deleted.

7.6 Prototype Event Code Administration

7.6.1 General

The Prototype Event Code Administration menu is used for maintaining the list of NCCDS Prototype Event Codes for a SIC. This menu is used for maintaining or viewing information from the following Database table:

- PROTOTYPE_EVENT_CODE - Valid NCCDS Prototype Event Codes assigned to a SIC.

Upon selection of this menu, the operator is first prompted for the SIC for which the Prototype Event Codes are desired:

```
Enter SIC: 1294
```

After entry of the desired SIC, the Prototype Event Code Administration menu is displayed:

```
SIC 1294 Prototype Event Code Administration Menu

1 = Display Prototype Event Codes
2 = Add a Prototype Event Code
3 = Remove a Prototype Event Code
```

7.6.2 Display Prototype Event Codes Menu Option

Selection of the *Display Prototype Event Codes* menu option provides a listing of all Prototype Event Codes for a SIC, such as follows:

```
Prototype Event Codes for SIC 1294, EIF database instance
-----
C21
C23
C31
C33
C35
D31
D33
E31
E33
M30
M31
M32
M33
M34
M35
S31
S33
```

7.6.3 Add a Prototype Event Code Menu Option

Selection of the *Add a Prototype Event Code* menu option provides a means of adding a new Prototype Event Code for a SIC. Upon selection of this option, the operator is prompted for the new code:

```
Enter Prototype Event Code: C21
Prototype Event Code C21 added for SIC 1294
```

7.6.4 Remove a Prototype Event Code Menu Option

Selection of the *Remove a Prototype Event Code* menu option provides a means of removing an existing Prototype Event Code from a SIC. Upon selection of this option, the operator is prompted for the code to remove:

```
Enter Prototype Event Code: C21
Prototype Event Code C21 removed for SIC 1294
```

7.7 SUPIDEN Administration

7.7.1 General

The SUPIDEN Administration menu is used for maintaining the list of SUPIDENs for a SIC. This menu is used for maintaining or viewing information from the following Database table:

- SUPIDEN - Valid Support Identifiers (SUPIDENs) for a SIC.

Upon selection of this menu, the operator is first prompted for the SIC for which the SUPIDENs are desired:

```
Enter SIC: 1294
```

After entry of the desired SIC, the SUPIDEN Administration menu is displayed:

```
SIC 1294 SUPIDEN Administration Menu

1 = Display SUPIDENs
2 = Add a SUPIDEN
3 = Remove a SUPIDEN
```

7.7.2 Display SUPIDENs Menu Option

Selection of the *Display SUPIDENs* menu option provides a listing of all SUPIDENs for a SIC, such as follows:

```
SUPIDENs for SIC 1294, EIF database instance
-----

SUPIDEN  System ID
-----
B1294CS  NCC
B1294EE  NCC
B1294MS  NCC
B1294PB  NCC
B1294SE  NCC
X1294NA  DAS
Y1294CS  NCC
Y1294EE  NCC
Y1294MS  NCC
Y1294PB  NCC
Y1294SE  NCC
```

7.7.3 Add a SUPIDEN Menu Option

Selection of the *Add a SUPIDEN* menu option provides a means of adding a new SUPIDEN for a SIC. Upon selection of this option, the operator is prompted for the new code and system ID (NCC or DAS):

```
Enter SUPIDEN: B1294CS
Enter System ID (1=NCC, 2=DAS): 1
SUPIDEN B1294CS added for SIC 1294
```

EIF SNIF must be restarted for this change to take effect

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.7.4 Remove a SUPIDEN Menu Option

Selection of the *Remove a SUPIDEN* menu option provides a means of removing an existing SUPIDEN from a SIC. Upon selection of this option, the operator is prompted for the code to remove:

```
Enter SUPIDEN: abcdefg
SUPIDEN abcdefg removed for SIC 1294
```

EIF SNIF must be restarted for this change to take effect

After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8 TDRS Name Administration

7.8.1 General

The TDRS Name Administration menu is used for maintaining the list of TDRS Names and Groups (or Sets) that are available to the Client user when creating Schedule Requests. This menu is used for maintaining or viewing information from the following Database tables:

- TDRS_NAME - Valid TDRS names
- TDRS_GROUP - Valid TDRS group/set names
- TDRS_IN_GROUP - TDRS group/name assignments

Following is the TDRS Name Administration menu:

```
TDRS Name Administration Menu

1 = Display TDRS Names
2 = Display TDRS Groups

3 = Add TDRS Name
4 = Remove TDRS Name
5 = Add TDRS Group
6 = Remove TDRS Group

7 = Add TDRS Names to Group
8 = Remove TDRS Names from Group
9 = Update TDRS Name or Group
```

There is no capability for changing the name of a TDRS. In cases where name changes are required, such as when changing from an alphanumeric name (e.g., TDE) to a numeric name indicating longitudinal position (e.g., 171), the new name must instead be added. This addition must be followed by a period of time where both the old and new names coexist in the SWSI Database because of constraints that require that Schedule Requests and Active Events referencing the old name be able to access that name. After these Schedule Requests and Active Events have been purged from the Database, then the old names may be removed. During this overlap period, an operational procedure must be employed to notify SWSI users that, even though the Client may still allow them to create Schedule Requests with the old name, this is in fact an invalid entry and will be rejected by NCCDS or DAS.

7.8.2 Display TDRS Names Menu Option

Selection of the *Display TDRS Names* menu option provides a listing of all TDRS Names, such as follows:

```
TDRS Names, EIF database instance
-----

TDRS      New
Name  DAS?  NCC?  Generation?
-----
171    Y     Y
275    Y     Y
TDE    Y     Y
TDH           Y
TDS    Y     Y
TDW    Y     Y
```

The following information is provided in this display:

- TDRS Name – TDRS identification.
- DAS – indicates whether name is used by DAS.
- NCC – indicates whether name is used by NCC.
- New Generation – indicates whether TDRS is an HIJ spacecraft. This is needed by SNIF in order to parse MAF/SMAF UPDs received from NCCDS.

7.8.3 Display TDRS Groups Menu Option

Selection of the *Display TDRS Groups* menu option provides a listing of all TDRS Groups or Sets, such as follows:

```
TDRS Groups, EIF database instance
-----

TDRS  TDRS      New
Group Name  DAS?  NCC?  Generation?
-----
047   TDS           Y
049   TDS           Y
150   TDE           Y
ANY   171           Y
      TDE           Y
      TDS           Y
      TDW           Y
TES   TDE           Y
      TDS           Y
TW7   171           Y
      TDW           Y
TWE   TDE           Y
      TDW           Y
```

The following information is provided in this display:

- TDRS Group – TDRS group name.
- TDRS Name – list of TDRS names assigned to that group.
- DAS – indicates whether group is used by DAS.
- NCC – indicates whether group is used by NCC.
- New Generation – indicates whether group is for HIJ spacecraft.

7.8.4 Add TDRS Name Menu Option

Selection of the *Add TDRS Name* menu option provides a means of adding a new TDRS Name. Upon selection of this option, the operator is prompted for information about the new name:

```
Enter TDRS Name: TDE
Used by DAS? (Y/N): n
Used by NCC? (Y/N): y
New Generation (HIJ)? (Y/N): n

TDRS Name TDE added

EIF SNIF must be restarted for this change to take effect
```

After entry is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8.5 Remove TDRS Name Menu Option

Selection of the *Remove TDRS Name* menu option provides a means of removing an existing TDRS Name. Upon selection of this option, the operator is prompted for the name to remove:

```
Enter TDRS Name: TDE
Are you sure you want to remove TDRS Name TDE?
You will be unable to remove the TDRS Name if there are
associated Schedule Request and Active Schedule Service entries.
Enter Y to confirm: y
TDRS Name TDE removed

EIF SNIF must be restarted for this change to take effect
```

If there are any Schedule Requests or Active Events still in the Database that reference this name, then an error message similar to the following is displayed:

```
Database Error -2292, ORA-02292: integrity constraint (SWSIDB.SAR_TDRS_FK) violated -
child record found
Error removing TDRS Name TDE, database error
```

All Schedule Requests and Active Events must first be purged as described in Section 7.5 before the removal operation can proceed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8.6 Add TDRS Group Menu Option

Selection of the *Add TDRS Group* menu option provides a means of adding a new TDRS Group or Set. Upon selection of this option, the operator is prompted for information about the new group:

```
Enter TDRS Group Name: ALL
Used by DAS? (Y/N): n
Used by NCC? (Y/N): y
New Generation (HLJ)? (Y/N): n
Enter first TDRS Name: 171
TDRS Group ALL added with initial TDRS Name of 171
Enter next TDRS Name: TDE
TDRS Name TDE added to Group ALL

EIF SNIF must be restarted for this change to take effect

Enter next TDRS Name: TDS
TDRS Name TDS added to Group ALL

EIF SNIF must be restarted for this change to take effect

Enter next TDRS Name:

EIF SNIF must be restarted for this change to take effect
```

The operator is continually prompted for TDRS Names to add to the group. Entry is considered complete when an *Enter* is entered in response to a *Enter next TDRS Name* prompt. After entry is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8.7 Remove TDRS Group Menu Option

Selection of the *Remove TDRS Group* menu option provides a means of removing an existing TDRS Group. Upon selection of this option, the operator is prompted for the group to remove:

```
Enter TDRS Group Name: ALL
Are you sure you want to remove TDRS Group Name ALL?
You will be unable to remove the TDRS Group Name if there are
associated Schedule Request and Active Schedule Service entries.
Enter Y to confirm: y
TDRS Group Name ALL removed
```

EIF SNIF must be restarted for this change to take effect

If there are any Schedule Requests or Active Events still in the Database that reference this name, then an error message similar to the following is displayed:

```
Database Error -2292, ORA-02292: integrity constraint (SWSIDB.SAR_TDRS_FK) violated -
child record found
Error removing TDRS Group Name ALL, database error
```

All Schedule Requests and Active Events must first be purged as described in Section 7.5 before the removal operation can proceed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8.8 Add TDRS Names to Group Menu Option

Selection of the *Add TDRS Names to Group* menu option provides a means of adding a new TDRS Name to an existing TDRS Group. Upon selection of this option, the operator is prompted for information about the names and the group to be added to:

```
Enter TDRS Group Name: ALL
Enter first TDRS Name: TDW
TDRS Name TDW added to TDRS Group ALL
Enter next TDRS Name:
```

The operator is continually prompted for TDRS Names to add to the group. Entry is considered complete when an *Enter* is entered in response to an *Enter next TDRS Name* prompt.

7.8.9 Remove TDRS Names from Group Menu Option

Selection of the *Remove TDRS Names from Group* menu option provides a means of removing TDRS Names from an existing TDRS Group. Upon selection of this option, the operator is prompted for information about the names to be removed and the group to be removed from:

```
Enter TDRS Group Name: ALL
Enter first TDRS Name: TDE
TDRS Name TDE removed from TDRS Group ALL
Enter next TDRS Name: TDW
TDRS Name TDW removed from TDRS Group ALL
```

EIF SNIF must be restarted for this change to take effect

Enter next TDRS Name:

EIF SNIF must be restarted for this change to take effect

The operator is continually prompted for TDRS Names to remove from the group. Entry is considered complete when an *Enter* is entered in response to an *Enter next TDRS Name* prompt. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.8.10 Update TDRS Name or Group Menu Option

Selection of the *TDRS Name or Group* menu option provides a means of changing the settings of an existing TDRS Name or Group. Upon selection of this option, the operator is prompted for which information to change:

```
Enter TDRS Name or TDRS Group Name: ALL
Used by DAS? (Y/N): y
Used by NCC? (Y/N):
New Generation (HIJ)? (Y/N):
TDRS (Group) Name ALL updated
```

EIF SNIF must be restarted for this change to take effect

Pressing *Enter* for any field causes that field to remain unchanged. In this case, only the *Used by DAS* flag was changed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.9 SSC Administration

7.9.1 General

The Service Specification Code (SSC) Administration menu is used for maintaining the list of SSCs for a SIC. This menu is used for maintaining or viewing information from the following Database tables:

- SSC - Valid Service Specification Codes (SSCs) assigned to a SIC.
- SSC_PARAM – Default parameter values for an SSC.

Upon selection of this menu, the operator is first prompted for the SIC for which the SSCs are desired:

```
Enter SIC: 1294
```

After entry of the desired SIC, the SSC Administration menu is displayed:

```
SIC 1294 SSC Administration Menu

1 = Display NCC SSCs
2 = Display DAS SSCs

3 = Add an NCC SSC
4 = Add a DAS SSC
5 = Add initial 10 DAS SSC's (001-010)
6 = Duplicate an SSC
7 = Remove an SSC

8 = Make an SSC editable
9 = Make all DAS SSCs editable
a = Make all NCC SSCs editable

b = Make an SSC not editable
c = Make all DAS SSCs not editable
d = Make all NCC SSCs not editable

e = Add missing SSC parameters
```

Following are the valid SWSI service types which may be associated with an SSC:

- MAF
- SSAF
- KSAF
- SMAF
- KASAF
- MAR
- SSAR
- KSAR
- SMAR
- KASAR
- KASARWB
- TRKN
- TRKC
- EETF
- EETR
- DASMAR

7.9.2 Display NCC SSCs Menu Option

Selection of the *Display NCC SSCs* menu option provides a listing of all NCC SSCs for a SIC, such as follows:

```
NCC SSCs for SIC 1294, EIF database instance
-----
```

SSC	Service Type	Editable?	User Lock ID	IP Address
A01	MAF	Y	sardella	xxx.xxx.xxx.xxx
A02	MAF	Y		
B02	MAR	Y		
B03	MAR	Y		
B21	MAR	Y		
G01	EETF	Y		
G02	EETF	Y		
H01	SSAF	Y		
H03	SSAF	Y		
H05	SSAF	Y		

The following information is provided in this display:

- SSC – SSC code.
- Service Type – type of service (MAF, MAR, SSAF, etc.)
- Editable – indicates whether the SSC default parameter values are editable from the Client by an authorized user (NCC Manager).
- User Lock ID – ID of the Client user who has the SSC locked for editing.
- IP Address – address of the Client user who has the SSC locked for editing.

7.9.3 Display DAS SSCs Menu Option

Selection of the *Display DAS SSCs* menu option provides a listing of all DAS SSCs for a SIC, such as follows:

```
DAS SSCs for SIC 1294, EIF database instance
-----
```

SSC	Service Type	Editable?	User Lock ID	IP Address
001	DASMAR	Y		
002	DASMAR	Y		
003	DASMAR	Y		
004	DASMAR	Y		
005	DASMAR	Y		
006	DASMAR	Y		
007	DASMAR	Y		
008	DASMAR	Y		
009	DASMAR	Y		
010	DASMAR	Y		

The following information is provided in this display:

- SSC – SSC code.
- Service Type – type of service. For DAS services, this is always DASMAR.
- Editable – indicates whether the SSC default parameter values are editable from the Client by an authorized user (DAS Manager).
- User Lock ID – ID of the Client user who has the SSC locked for editing.
- IP Address – address of the Client user who has the SSC locked for editing.

7.9.4 Add an NCC SSC Menu Option

Selection of the *Add an NCC SSC* menu option provides a means of adding a new NCC SSC. Upon selection of this option, the operator is prompted for information about the new SSC:

```
Enter SSC Code: A08
Enter Service Type: MAF
Editable (Y/N): y
SSC Default Parameter added for DATARATEMAXF =
SSC Default Parameter added for DOPC =
SSC Default Parameter added for DTR1 =
SSC Default Parameter added for FRQ1 =
SSC Default Parameter added for TSWS =
SSC Default Parameter added for UDAN =
SSC Default Parameter added for UIFCADDRESSF =
NCC SSC A08 added for SIC 1294
```

The default parameter values for the new SSC are initialized set to NULL. After the SSC is added by the DBA, a Client user with NCC Manager privilege must subsequently set the default parameters to their correct values.

7.9.5 Add a DAS SSC Menu Option

Selection of the *Add an DAS SSC* menu option provides a means of adding a new DAS SSC. Upon selection of this option, the operator is prompted for the new SSC code:

```
Enter SSC Code: 011
SSC Default Parameter added for Acq_Mode =
SSC Default Parameter added for Carrier_Freq_Ref =
SSC Default Parameter added for Data_Fmt_I =
SSC Default Parameter added for Data_Fmt_Q =
SSC Default Parameter added for Data_Rate_I =
SSC Default Parameter added for Data_Rate_Q =
SSC Default Parameter added for Data_class_ID = Not Applicable
.
.
.
DAS SSC 011 added for SIC 1294
```

The default parameter values for the new SSC are initialized set to NULL. After the SSC is added by the DBA, a Client user with DAS Manager privilege must subsequently set the default parameters to their correct values.

7.9.6 Add initial 10 DAS SSC's (001-010)

Selection of the *Add Initial 10 DAS SSC's (001-010)* menu option provides a means of populating the 10 default SSCs for this new SIC. This function is useful when a new DAS SIC is added to the SWSI database. To use this function, the operator would add the new SIC, then select the Add Initial 10 DAS SSCs menu option. These SSCs should initially contain the defaults specified in the DAS/SWSI ICD.

Upon selection of this option, the operator will see all 10 new SSC codes populated with their default values:

```
Enter command: 5
SSC Default Parameter added for Acq_Mode =
SSC Default Parameter added for Carrier_Freq_Ref =
SSC Default Parameter added for Data_Fmt_I =
SSC Default Parameter added for Data_Fmt_Q =
.
.
.
SSC Default Parameter added for VCP_Segregation_Q = Off

DAS SSC 001 added for SIC 1294
.
.
.
DAS SSC 010 added for SIC 1294
```

7.9.7 Duplicate an SSC Menu Option

Selection of the *Duplicate an SSC* menu option provides a means of adding a new SSC to have the same default parameter values as an existing SSC. This function is useful when creating SSCs that may be similar except for the default value of a few parameters. To use this function, the operator would add the initial SSC, edit its default parameter values from the Client, duplicate the initial SSC to a second SSC, and finally edit the default parameter values of the second SSC.

Upon selection of this option, the operator is prompted for information about the new SSC code:

```
Enter original SSC Code: A01
Enter new SSC Code: A10
Enter System ID (1=NCC, 2=DAS): 1
Editable (Y/N): y
SSC A10 added for SIC 1294
```

7.9.8 Remove an SSC Menu Option

Selection of the *Remove an SSC* menu option provides a means of removing an existing SSC. Upon selection of this option, the operator is prompted for the SSC to remove:

```
Enter SSC Code: A08
Enter System ID (1=NCC, 2=DAS): 1
Are you sure you want to remove SSC A08?
This will also remove all Schedule Request Service entries associated with that SSC.
You will be unable to remove the SSC if there are associated Active Schedule Service
entries.
Enter Y to confirm: y
SSC A08 removed for SIC 1294
```

7.9.9 Make an SSC editable Menu Option

Selection of the *Make an SSC editable* menu option provides a means of marking an SSC's default parameter values as editable from the Client by an authorized user (NCC or DAS Manager). Upon selection of this option, the operator is prompted for the SSC to make editable:

```
Enter SSC Code: A01
Enter System ID (1=NCC, 2=DAS): 1
SSC A01 made editable
```

7.9.10 Make all DAS SSCs editable Menu Option

Selection of the *Make all DAS SSCs editable* menu option provides a means of marking all DAS SSCs for a SIC as allowing their default parameter values to be edited from the Client by an authorized user (DAS Manager). Upon selection of this option, the following is displayed:

```
All DAS SSCs made editable
```

7.9.11 Make all NCC SSCs editable Menu Option

Selection of the *Make all NCC SSCs editable* menu option provides a means of marking all NCC SSCs for a SIC as allowing their default parameter values to be edited from the Client by an authorized user (NCC Manager). Upon selection of this option, the following is displayed:

```
All NCC SSCs made editable
```

7.9.12 Make an SSC not editable Menu Option

Selection of the *Make an SSC not editable* menu option provides a means of marking an SSC's default parameter values as not editable from the Client by an authorized user (NCC or DAS Manager). Upon selection of this option, the operator is prompted for the SSC to make non-editable:

```
Enter SSC Code: A01
Enter System ID (1=NCC, 2=DAS): 1
SSC A01 made not editable
```

7.9.13 Make all DAS SSCs not editable Menu Option

Selection of the *Make all DAS SSCs not editable* menu option provides a means of marking all DAS SSCs for a SIC as not allowing their default parameter values to be edited from the Client by an authorized user (DAS Manager). Upon selection of this option, the following is displayed:

```
All DAS SSCs made not editable
```

7.9.14 Make all NCC SSCs not editable Menu Option

Selection of the *Make all NCC SSCs not editable* menu option provides a means of marking all NCC SSCs for a SIC as not allowing their default parameter values to be edited from the Client by an authorized user (NCC Manager). Upon selection of this option, the following is displayed:

```
All NCC SSCs made not editable
```

7.9.15 Add missing SSC parameters

Selection of the *Add missing SSC parameters* menu option provides a means of adding the missing SSC parameters with a value of NULL. Upon selection of this option, the following is displayed:

```
Enter command: e
Processing SSC 001, Service Type DASMAR
Processing SSC 002, Service Type DASMAR
Processing SSC 003, Service Type DASMAR
Processing SSC 004, Service Type DASMAR
Processing SSC 005, Service Type DASMAR
Processing SSC 006, Service Type DASMAR
Processing SSC 007, Service Type DASMAR
Processing SSC 008, Service Type DASMAR
Processing SSC 009, Service Type DASMAR
Processing SSC 010, Service Type DASMAR
```

.
. .
.

7.10 Active Schedule Upload Administration

7.10.1 General

The Active Schedule Upload Administration menu is used for maintaining configuration information for the SNIF and SDIF to use in constructing and transmitting Active Schedule files to connected Clients. This menu is used for maintaining or viewing information from the following Database table:

- **ACTIVE_EVENTS_UPLOAD** – Parameters for periodic upload of active schedule file(s) to Client workstations.

Following is the Active Schedule Upload Administration menu:

```
Active Schedule Upload Administration Menu

1 = Display Active Schedule Upload Parameters entries
2 = Update Active Schedule Upload Parameters entry
3 = Add Active Schedule Upload Parameters entry
4 = Remove Active Schedule Upload Parameters entry
```

7.10.2 Display Active Schedule Upload Parameters entries Menu Option

Selection of the *Display Active Schedule Upload Parameters entries* menu option provides a listing of all entries from the ACTIVE_EVENTS_UPLOAD table, such as follows:

```
Active Schedule Upload Parameters, EIF database instance
-----
```

SIC	System	Periodic Freq	Poll Period	Include Parameters?	Translate Enumerated?
1294	NCC	60	5	Y	Y
1294	DAS	60	5	Y	Y
3301	NCC	60	5	Y	Y

The following information is provided in this display:

- SIC – SIC code.
- System – indicates whether schedule to be uploaded is for NCC or DAS.
- Periodic Frequency – how often in minutes to force upload of the current schedule, regardless of whether there have been changes.
- Poll Period – how often to poll the Active Schedule to determine if there have been changes since the last transmission. If the schedule has changed, then transmit a new file.
- Include Parameters – whether to include the initial service parameter values.
- Translate Enumerated – for enumerated parameter types, whether to send the numeric value or an enumerated text string.

7.10.3 Update Active Schedule Upload Parameters entry Menu Option

Selection of the *Update Active Schedule Upload Parameters entry* menu option provides a means of changing the settings of an existing ACTIVE_EVENTS_UPLOAD entry. Upon selection of this option, the operator is prompted for which information to change:

```
Enter SIC: 3301
Enter System ID (1=NCC, 2=DAS): 1
Enter Frequency of Periodic Uploads (min), 0 = no periodic uploads:
Enter Poll Period (min), 0 = no polling for changes: 10
Include parameters (Y/N):
Translate enumerated parameter values (Y/N):

SIC 3301 Active Schedule Upload parameters entry updated

EIF SNIF must be restarted for this change to take effect
```

Pressing *Enter* for any field causes that field to remain unchanged. In this case, only the Poll Period was changed. After the operation is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.10.4 Add Active Schedule Upload Parameters entry Menu Option

Selection of the *Add Active Schedule Upload Parameters entry* menu option provides a means of adding a new ACTIVE_EVENTS_UPLOAD entry. Upon selection of this option, the operator is prompted for information about the new entry:

```
Enter SIC: 1294
Enter System ID (1=NCC, 2=DAS): 1
Enter Frequency of Periodic Uploads (min), 0 = no periodic uploads: 60
Enter Poll Period (min), 0 = no polling for changes: 5
Include parameters (Y/N): n
Translate enumerated parameter values (Y/N): n

Active Schedule Upload parameters entry 1294 added

EIF SNIF must be restarted for this change to take effect
```

After entry is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

7.10.5 Remove Active Schedule Upload Parameters entry Menu Option

Selection of the *Remove Active Schedule Upload Parameters entry* menu option provides a means of removing an existing ACTIVE_EVENTS_UPLOAD entry. Upon selection of this option, the operator is prompted for information about the entry to be removed:

```
Enter SIC: 1294
Enter System ID (1=NCC, 2=DAS): 1

SIC 1294 System NCC Active Schedule Upload parameters entry deleted

EIF SNIF must be restarted for this change to take effect
```

After entry is complete, the appropriate SNIF must be restarted using the procedure described in Section 4.2.

Section 8. Digital Certificate Management

8.1 Overview

SWSI's digital certificate management is based upon the Public Key Infrastructure (PKI) process. In public key technology, each user has a pair of keys. The keys are not the same, yet they match up in a unique way. One key is kept secret by the user and is called the private key. The private key is that portion of a key pair used to decrypt or digitally fingerprint (sign) data. The other key is widely distributed and is called the public key. The public key is that portion of a key pair used to encrypt data or verify digital fingerprints. The public key is stored inside the user's certificate. A user's digital certificate is a computer-generated record that ties a user's identification with the user's public key in a trusted bond. It can be thought of as the electronic equivalent of a passport: both contain secure information that can be used to verify its owner's identity.

A digital fingerprint or signature is the result of data that is encrypted using a user's private key. The fingerprint provides a guarantee to a recipient of the "signed" data that it has not been modified since it was signed. It also verifies that the signed data came from the entity that sent it.

The Certificate Authority (CA) is responsible for the creation and management of certificates. The Registration Authority (RA) is responsible for the identification, authentication, and registration of certificate subscribers and performs certificate and key management functions on behalf of the CA. SWSI has rolled the functions of the CA and RA into one.

There are three types of certificates that are used within SWSI: CA certificates, application certificates, and user certificates. All three work together to provide a framework for authentication, data privacy, data integrity and non-repudiation of missions that use NASA's TDRSS services.

8.2 Certificate Authority

The SWSI Project acts as its own Certificate Authority. SWSI uses the Phaos J/CA toolkit to generate digital certificates, including the CA's public and private keys. However, the Phaos J/CA toolkit is not used for the web server certificates. Section 3.8.5 describes how to generate the CA's public and private keys.

Usually CAs are configured to have an extremely long lifetime as compared to the users they authorize. SWSI has been deployed with a CA that lasts for 10 years. There is also a backup CA that has an expiration of 20 years. These certificates were generated on September 6, 2002. The 10-year CA is scheduled to expire on September 30, 2012. The 20-year CA is scheduled to expire on September 30, 2022.

The process repeats for the next cycle.

8.3 Web Server Certificates

The SWSI Project acts as its own Certificate Authority and generates its own self-signed certificates for use in the secure web server. The Apache web server is configured to use OpenSSL Privacy Enhanced Mail (PEM) generated certificates.

The following steps are used to generate an OpenSSL certificate for the secure web server, replacing `xyzzzy.example.net` with the hostname of the certificate (e.g. `swwsi-server.nascom.nasa.gov`):

1. `cd /usr/local/ssl/certs`
2. `openssl rand -out /.rnd -rand /var/spool/prngd/pool 1024`
3. `openssl req -new > xyzzzy.csr`

This step leads to a series of questions that might have answers similar to these:

```
openssl req -new > xyzzzy.csr
Using configuration from /usr/local/ssl/openssl.cnf
Generating a 1024 bit RSA private key
.....+++++
.....+++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase: (enter a passphrase here without the parentheses)
Verifying password - Enter PEM pass phrase: (enter same passphrase again)
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name (DN).
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:                               US
State or Province Name (full name) [Some-State]:               Maryland
Locality Name (eg, city) []:                                    Greenbelt
Organization Name (eg, company) [Internet Widgits Pty Ltd]:   NASA-GSFC
Organizational Unit Name (eg, section) []:                     SWSI
Common Name (eg, YOUR name) []:                                xyzzzy.example.net
Email Address []:                                               .

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:                                        boo!
An optional company name []:                                    .
```

4. `openssl rsa -in privkey.pem -out xyzzzy.key`
5. `openssl x509 -in xyzzzy.csr -out xyzzzy.cert -req -signkey xyzzzy.key -days 3640`

The name of the certificate will be `xyzzzy.cert`. The certificate should be copied or moved to the proper directory as specified in the Apache web server configuration file.

8.4 Application Server Certificates

The SWSI Application Server has its own digital certificate. The SWSI Application Server has been deployed with a digital certificate that lasts for 10 years. The certificate was generated on September 6, 2002. The 10-year certificate is scheduled to expire on September 30, 2012. The certificates are manually generated according to the procedure in section 3.8.4.

The Application Server's digital certificate is used by the SWSI Client for authentication. The SWSI Client verifies the server's digital fingerprint from the server's public key. Then the Application Server and Client use the PKI process to establish a secure TCP session.

8.5 SWSI Client User Certificates

Each SWSI Client user must generate their own unique digital certificate using the web-based certificate generation tool described in section 3.8. User certificates expire 366 days after creation. Certificates generated via the Open SWSI web servers remain available for download by the user for 30 minutes. After this time, the certificates are transferred via the TUT Proxy to the SWSI Backend servers for permanent archival. Certificates generated via the SWSI Backend web servers remain available for download by the user for 30 minutes also. The certificates are moved to the archive directory after the time period has lapsed. User certificates can be generated manually by the SWSI Administrator or via the web interface by the users according to Section 3.8.3.

The SWSI Client's digital certificate is used by the Application Server for authentication. The Application Server verifies the Client's digital fingerprint from the Client's public key. Then the Client and Application Server use the PKI process to establish a secure TCP session.

Section 9. System Administration Procedures

9.1 Server Accounts

Each SWSI server has a small number of standard accounts (*root*, *swsiops*, *oracle*) and an individual account for each local user.

The *root* account is used only for systems administration activities. Logging in as *root* on the console should be reserved for those situations where console access as *root* is truly necessary (e.g., backups, restores, RAID management). A better practice is to use the switch user, or *su* command in a Terminal window while logged in as a regular user when root privileges are required.

The *swsiops* account owns the SWSI software. The SWSI software, configuration files, and log files are stored in directories that are located in the home directory of the *swsiops* account (*/export/home/swsiops*). In addition, the SWSI software applications are run by the High Availability (HA) application with the user ID of the *swsiops* account. Local operators rarely need to log in to the *swsiops* account. Software deliveries and maintenance might require use of the *swsiops* account by either logging in at the console, or using *su* in a Terminal window.

The *oracle* account owns the Oracle database software. The *oracle* account is also used to run the database software and for database maintenance.

Each local user of a SWSI server will be given their own Unix user account. Local users include operators, DBAs, system administrators, and software maintenance personnel. SWSI Client users (the end users of the system) do not get a Unix account. Local user accounts should be created by the system administrators using a command-line script (*makenewuser*) as described in Section 9.3.1 below.

9.2 root Account Environment

9.2.1 General

This subsection describes the root account user environment.

9.2.2 Switch User (su) Command Usage

su should always be used with the following option when switching user to any of the accounts on a SWSI server:

```
su -
```

9.2.3 System Control

If at all possible, set the HA mode to HALTED before rebooting or shutting down the system.

Use only the *init* command to change the operating level of the system. Do not use any of the other system control commands (*reboot*, *shutdown*) since they do not necessarily run all of the shutdown scripts in */etc/init.d*. The following lists the common *init* commands and their effects:

```
init 0    Shut down to the firmware level (the ok> prompt)
init 5    Power-off the system
init 6    Reboot the system
```

9.2.4 root Account Aliases

The following are some of the C-shell aliases that have been configured for the root account:

```
goipf          does a "cd" to the IP Filter configuration directory
checkipf       shows current IP Filter statistics
restartipf     restarts the IP Filter
checkapache    syntax checks the Apache web server's configuration file
restartapache  restarts the Apache web server
startapache    starts the Apache web server
stopapache     stops the Apache web server
goapache       does a "cd" to the base directory of Apache web server
gohome         does a "cd" to /export/home
gocots         does a "cd" to /export/home/cots
gotars         does a "cd" to /export/home/rootStuff/tars
rm6           runs the RAID management application
```

9.3 Account Management

9.3.1 Account Creation

Local user accounts are created by the system administrator using the following procedure:

1. Become *root*, either by logging in as *root* directly, or by using the following command in a Terminal window:

```
su -
```

In the above command, the minus sign is required. If not included, the root account's environment will not be properly initialized.

2. Run the *makenewuser* script:

```
makenewuser
```

The script will prompt the administrator to enter the account owner's full name and the Unix account name. Local conventions should be followed in deciding on the form of the user's full name and account name. Following is a sample session:

```
# makenewuser
Enter user's full name: Pete Moss
Pete Moss
Enter user's account name: pmoss
pmoss
13 blocks
```

After the account information has been entered, the script will create the user's UNIX account in */etc/passwd*, supplying the following standard values:

```
shell      /bin/csh
password   *locked*
UNIX group  swsiops
```

The user's home directory is created in */export/home* with the following files:

```
.cshrc
.dtprofile
```

Note that the account is created as locked. The administrator must unlock the account by giving it a password. This can be accomplished in one of two manners: by using either the command-line *passwd* command or the Solaris *admintool* application.

9.3.2 Setting Initial Password with *passwd* Command

Execute the following as the *root* user to set a user's password from the Unix command line:

```
passwd <accountname>
```

and supply a password. This method does not change the default password behavior for the account, which is to have a password that never expires. If the customary local procedures require password expirations, then the *admintool* application must be used.

9.3.3 Setting Initial Password with *admintool* Application

The Solaris *admintool* is a graphical application that can be used for account management. A sample *admintool* panel is shown in Figure 9-1. Since it is graphical, it requires access to the X Window server on the system's console. This access is automatic if the administrator logged in directly as *root* on the console (not recommended). If the administrator logged in with their own personal account, follow this procedure to enable the use of *admintool*:

1. Launch a Terminal window. Enter the following command:

```
xhost localhost
```

2. Launch a second Terminal window. Enter the following:

```
su -
setenv DISPLAY localhost:0.0
```

3. In the root privileged Terminal, enter:

```
admintool
```

Use the *admintool* GUI to give the account a password. If local convention dictates, enter customary values for the appropriate fields (Min Change, Max Change, etc.). When all changes have been made, click on the *OK* button.

The screenshot displays the Solaris *admintool* application window, which is organized into three main sections: USER IDENTITY, ACCOUNT SECURITY, and HOME DIRECTORY. At the bottom, there are five buttons: OK, Apply, Reset, Cancel, and Help.

- USER IDENTITY:**
 - User Name: pmoss
 - User ID: 1415
 - Primary Group: 106
 - Secondary Groups: (empty)
 - Comment: Pete Moss
 - Login Shell: Other /usr/bin/csh
- ACCOUNT SECURITY:**
 - Password: Account is locked
 - Min Change: days
 - Max Change: days
 - Max Inactive: days
 - Expiration Date: None None None
(dd/mm/yy)
 - Warning: days
- HOME DIRECTORY:**
 - Path: /export/home/pmoss

Figure 9-1. Solaris *admintool* Application

9.3.4 Account Deletion with `userdel` Command

The command-line `userdel` command may be used to delete a user account. To use this command, execute the following as `root`:

```
userdel -r <accountname>
```

9.3.5 Account Deletion with `admintool` Application

The Solaris `admintool` is a graphical application that can be used for account management. A sample `admintool` panel is shown in Figure 9-1. Since it is graphical, it requires access to the X Window server on the system's console. This access is automatic if the administrator logged in directly as `root` on the console (not recommended). If the administrator logged in with their own personal account, follow this procedure to enable the use of `admintool`:

1. Launch a Terminal window. Enter the following command:

```
xhost localhost
```

2. Launch a second Terminal window. Enter the following:

```
su -  
setenv DISPLAY localhost:0.0
```

3. In the root privileged Terminal, enter:

```
admintool
```

Once the `admintool` GUI window appears, select the account to be deleted by single-clicking on it. In the *Edit* pull-down menu, choose *Delete*. When the dialog box appears, be sure to check the *Delete Home Directory* option before clicking the *Delete* button.

9.4 Backup and Recovery

9.4.1 Backup and Recovery Overview

Since all data associated with the SWSI applications is stored on the RAID, data on the internal server disks is not considered critical enough to require daily backups. A full backup is required only if there is a major system change, such as a software delivery or system reconfiguration. Daily or incremental backups are not required. The only data stored on the RAID is from the Oracle databases, so procedures are provided for performing daily backups of that data to tape.

9.4.2 Full System Backup

Full system backups should be done in single-user mode, using the script `/backup/bin/backup.full`. To prepare a server for backup, first perform the following:

1. Use the HA GUI to set the HA mode to HALTED.
2. Eject any tapes in the tape drive.
3. Insert the backup tape into the tape drive.

To place the system in single-user mode, enter the following in a Terminal window:

```
su -  
init 0
```

These commands shut the system down to the firmware level. At the `ok>` prompt, enter the following:

```
boot disk -s
```

When prompted, supply the `root` password to enter maintenance mode. To perform the backup, enter:

```
/backup/bin/backup.full
```

When the backup is finished, reboot the system by entering:

```
init 6
```

9.4.3 Full System Recovery

The following procedure is performed to do a full system recovery:

1. If the server is running, do the following to place the system in firmware mode:
 - a. Use the HA GUI to set the HA mode to HALTED.
 - b. Enter the following in a Terminal window:

```
su -  
init 0
```

2. If the server is powered down, then do the following:
 - a. Turn on power
 - b. Press the STOP-A key combination at the appropriate time to prevent the system from booting.
3. Insert the Solaris 8 media into the CDROM drive.
4. Enter the following at the prompt:

```
ok> boot cdrom -s
```

5. Insert the latest full backup tape into the tape drive.
6. Format disk slices. For the Open Servers (Ultra 2s):

```
format
choose disk (usually 0)
partition
modify
base:                1
continue:            y
free hog [6]:        7
enter size of partition:
  0:                2gb
  1:                1gb
  3:                0
  4:                0
  5:                0
  6:                0
  7:                (5.43G)
ok?:                  y
table name:          "<SUN9.0G cyl 4924 alt 2 hd 27 sec 133>"
ready:               y
quit
quit
```

For the Backend Servers (SunBlades):

```
format
choose disk (usually 0)
partition
modify
base:                1
continue:            y
free hog [6]:        7
enter size of partition:
  0:                6gb
  1:                512mb
  3:                27.41gb
  4:                0
  5:                0
  6:                0
  7:                (1.41mb)
ok?:                  y
table name:          "<SUN36G cyl 24620 alt 2 hd 27 sec 107>"
ready:               y
quit
quit
```

7. Create file systems. For the Open Servers:

```
newfs /dev/rdisk/c0t0d0s0
fsck /dev/rdisk/c0t0d0s0
newfs /dev/rdisk/c0t0d0s7
fsck /dev/rdisk/c0t0d0s7
```

For the Backend Servers:

```
newfs /dev/rdisk/c1t1d0s0
```

```
fsck /dev/rdisk/clt1d0s0
newfs /dev/rdisk/clt1d0s3
fsck /dev/rdisk/clt1d0s3
```

8. Restore root (/) partition. For the Open Servers:

```
mount /dev/dsk/c0t0d0s0 /mnt
cd /mnt
tapes
ufsrestore rvf /dev/rmt/0n
rm restoresymtable
cd /
umount /mnt
fsck /dev/rdisk/c0t0d0s0
/usr/sbin/installboot
/usr/platform/`uname -i`/lib/fs/ufs/bootblk /dev/rdisk/c0t0d0s0
```

For the Backend Servers:

```
mount /dev/dsk/clt1d0s0 /mnt
cd /mnt
tapes
ufsrestore rvf /dev/rmt/0n
rm restoresymtable
cd /
umount /mnt
fsck /dev/rdisk/clt1d0s0
/usr/sbin/installboot
/usr/platform/`uname -i`/lib/fs/ufs/bootblk /dev/rdisk/clt1d0s0
```

9. Restore remaining partition. For the Open Servers:

```
mount /dev/dsk/c0t0d0s7 /mnt
cd /mnt
ufsrestore rvf /dev/rmt/0n
rm restoresymtable
cd /
umount /mnt
fsck /dev/rdisk/c0t0d0s7
```

For the Backend Servers:

```
mount /dev/dsk/clt1d0s3 /mnt
cd /mnt
ufsrestore rvf /dev/rmt/0n
rm restoresymtable
cd /
umount /mnt
fsck /dev/rdisk/clt1d0s3
```

10. Remove tape and reboot:

```
mt rewoffl
init 6
```

11. Perform a full backup as described in Section 9.4.2.

9.4.4 Database Backup

The Oracle database is backed up in a different manner than the system-level backups previously described. Database backups are performed daily, with the backup data being appended to a tape that is changed weekly.

The database backup scripts are run as scheduled in the *oracle* account's crontab. The crontab specifies that the script */export/home/oracle/backupscripts/hotbackup.sh* be executed on a daily basis. At time of initial delivery, this backup is scheduled to occur at 0600 UTC every day. The backup is scheduled on both of the Backend servers. Only the server that owns the RAID and is running Oracle will succeed in backing up the data.

The internal actions of the *hotbackup.sh* script are described in Section 6.3.1. When the actual backup of the database is complete, the data will have been stored in the */mnt/data6* filesystem on the RAID. The *hotbackup.sh* script then runs a second script, */export/home/oracle/backupscripts/db2disk.sh*, which creates a compressed archive copy of the data from the RAID on the internal disk of the server. This copy of the backup data is stored in the */export/home/ora-backups* directory, owned by the *oracle* user.

Once the backup data resides on the internal disk, the *db2disk.sh* script executes the */export/home/oracle/backupscripts/db2tape.sh* script. The *db2tape.sh* script appends the compressed archives onto the end of whatever tape happens to be in the server's tape drive. For this reason, the local operators should ensure that writable backup tapes are always loaded into the tape drives of both Backend Servers (since the HA status of the servers can change from day to day), and that the tapes be rotated on a weekly basis.

The output produced by the *hotbackup.sh* script and its secondary programs and scripts is written to a log file (*/var/tmp/hotbackup.log*). The oracle account's crontab runs one additional script, located at */export/home/oracle/backupscripts/rotateHotLogs.sh*. The *rotateHotLogs.sh* script manages the *hotbackup* logs by using the same file rotation scheme that is used on the UNIX system log files. Once a day, this script copies the set of saved *hotbackup* log files to higher index numbers, with the oldest log file being overwritten. At time of initial delivery, this script is scheduled to execute at 1000 UTC every day.

Additionally, both the *hotbackup* and the *rotateHotLogs* scripts log their actions in the system log file (*/var/adm/messages*). Their log entries are made using the "local2" facility identifier, which makes grepping for them easier.

9.4.5 Database Recovery

The database recovery procedures described in Section 6.3 assume that the database backup data is present in the */mnt/data6* filesystem on the RAID. If this data is not present, it will have to be loaded from the latest archives. If the latest archive is present on the internal disk of the server in the *oracle* account's *db-backups* directory, then the administrator can unarchive the data using *gunzip* and *tar* to the */mnt/data6* filesystem on the RAID before performing the Oracle recovery procedures. If the latest archive can only be found on a backup tape, the administrator must extract the archive from the tape into the *db-backups* directory before unarchiving.

9.4.6 Tape Handling

The script that backs up the SWSI Oracle database writes on whatever tape is in the tape drive. It does not position the tape (either forward or backward) before writing to the tape. For this reason, there are a few rules that must be followed:

1. The portion of the backup script that actually writes the data to the tape runs at the end of the daily backup. At time of initial delivery, this backup is scheduled to begin at 0600 UTC every day. However, the backup may take a considerable number of minutes (or even hours) before the data is ready to be written to the tape. Just how long the backup procedure takes depends on the amount of data to be saved, and on the processing load on the system (the backup procedure executes with a very low priority). The backup tape should be left alone during the entire backup procedure.

One way to detect whether the backup procedure is running or not is by finding the UNIX process ID of the system's *cron* process, and then using this process ID to show all processes that are children of the *cron*. Use the following procedure:

- a. Determine the cron process ID:

```
ps -ef | grep cron
```

- b. Use the following to list all running process that are the children of *cron*:

```
ptree <cronpid>
```

where *cronpid* is the process ID of *cron* as determined in Step a. If the backup is in progress, one of cron's children will be the *hotbackup.sh* script.

2. The backup tape must be rotated to offline storage before it fills up. The backup script is not smart enough to handle a backup that would span multiple tapes. At time of initial delivery, the plan is that the backup tape will be rotated weekly. Changing the tape is simple:
 - a. Make sure that the backup procedure is not running (see above).
 - b. Press the eject button on the tape drive. This will cause the tape in the drive to be rewound (to the beginning) and ejected.
 - c. Insert the replacement tape in the drive, and take the newly ejected tape to the tape storage location.

3. If, for some reason, the system's tape drive needs to be used for another purpose (e.g. system-level backups or a software delivery), care must be taken to ensure that the backup tape is returned to the same position that it had before it was ejected. Use the following procedure:
 - a. Before ejecting the backup tape, make sure that the backup procedure is not running (see above).
 - b. Press the eject button on the tape drive. This will cause the tape in the drive to be rewound to the beginning and ejected.
 - c. The tape drive is now free to be used for another purpose. However, remember that the backup procedure is run automatically by cron, and that it expects the backup tape to be properly positioned in the drive, so the other purpose must be completed before the scheduled backup time.
 - d. Once the other purpose has been completed, and the tape drive is once again empty, insert the backup tape in the drive.
 - e. Open a terminal window, and enter the following command:

```
mt eom
```

This command repositions the tape to the end of the previously recorded data, so that the next backup will write its data at the proper location.

9.5 IPFilter

9.5.1 IPFilter Overview

The SWSI servers use the IPFilter application to provide firewall services. IPFilter allows a system administrator to control what hosts have access to specific system services. This is used on the SWSI servers mainly to control what users have access to the secure HTTP service (HTTPS) for accessing the SWSI certificate generation tool, the SWSI Client software download section, and the TUT open web server, and for accessing the appropriate TCP port for connecting from the SWSI Client application.

IP addresses provided by the SWSI customer must be entered into the appropriate (backend or open server) IPFilter firewall table by the *root* user. All entries must be added to both the primary and secondary servers. The IPFilter table is located in the */etc/opt/ipf/ipf.conf* file on all the servers. The following root account aliases allow for easier management of the *ipf.conf* file:

```
goipf          does a "cd" to the IP Filter configuration directory
restartipf     restarts the IP Filter
```

Before editing the configuration file, it is good practice to save a copy of the file to a file of the same name with a version number on the end (e.g., *ipf.conf.10*). It is also necessary after changing the file to issue a *restartipf* command to make the changes take effect.

The format of the *ipf.conf* file is a bit obscure, so an IPFilter Configure utility is provided with the SWSI servers to provide a user-friendly interface for managing the filter rules. IPFilter Configure automatically backs up *ipf.conf* before committing changes and performs a *restartipf* after the system administrator completes a file change.

The IPFilter Configure command is invoked interactively by the *root* user as follows:

```
# ipfconfig -i
```

or by command line arguments such as follows:

```
# ipfconfig -a 192.168.1.3 "Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com"
```

If invoked with a *-v* option as follows:

```
# ipfconfig -v
```

Then the IPFilter Configure command just displays the version number for IPFilter and exits:

```
ipf: IP Filter: v3.4.31 (432)
Kernel: IP Filter: v3.4.31
```

If invoked with an invalid option, such as follows:

```
# ipfconfig -h
```

Then the IPFilter Configure command displays usage information and exits, similar to the following:

```
ipfconfig adds, removes, or lists IP Addresses from IPFilter

Usage:
ipfconfig -[hilv] -[arf] [ IP-Address ] -[n] bits -c [ "a comment" ]
-h = help (this message)
-a = Add an IP Address (NNN.NNN.NNN.NNN) to HTTPS, SWSI Client ports
-r = Remove an IP Address (NNN.NNN.NNN.NNN) from HTTPS, SWSI Client ports
-i = Interactive mode to add or remove an IP Address
-n = Number of bits in IP Address (e.g. 24 bit, 30 bit, default is 32 bit)
-c = Comment to be added with new entry
-f = Find a particular IP Address
-l = List current IP Addresses and ports
-v = Show IPFilter version
```

9.5.2 Adding an IP Address

To add access for an IP address to the IPFilter firewall table for the HTTPS and SWSI Client ports, invoke the IPFilter Configure command as follows:

```
# ipfconfig -a 192.168.1.3 -c "Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com"

Added 192.168.1.3 for port <https_port>
---192.168.1.3, <https_port>, 05/31/2003, 00:37:24, added by jstevens, Mission Alpha, Jenny
Doe, 555-867-5309, jenny.doe@toetag.com
Added 192.168.1.3 for port 3100
---192.168.1.3, 3100, 05/31/2003, 00:37:24, added by jstevens, Mission Alpha, Jenny Doe,
555-867-5309, jenny.doe@toetag.com
Restarting IP Filter... Done.
```

If you attempt to add an IP address that already exists, the following error message will be displayed:

```
# ipfconfig -a 192.168.1.3 -c "Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com"

A rule for 192.168.1.3 already exists.
---192.168.1.3, 3100, 05/31/2003, 00:03:45, added by jstevens, Mission Alpha, Jenny Doe,
555-867-5309, jenny.doe@toetag.com
---192.168.1.3, <https_port>, 05/31/2003, 00:14:00, added by jstevens, Mission Alpha, Jenny
Doe, 555-867-5309, jenny.doe@toetag.com
```

To add access for a 24 bit subnet IP address, invoke the IPFilter Configure command using the number of bits option as follows:

```
# ipfconfig -a 192.168.1.0 -n 24 -c "Mission Alpha, Jimmy Doe, 555-777-9311"

Added 192.168.1.0/24 for port <https-port>
---192.168.1.0/24, <https-port>, 12/31/2003, 13:21:00, added by jstevens, Mission Alpha,
Jimmy Doe, 555-777-9311, jimmy.doe@toetag.com
Added 192.168.1.0/24 for port 3100
---192.168.1.0/24, 3100, 12/31/2003, 13:21:00, added by jstevens, Mission Alpha, Jimmy
Doe, 555-777-9311, jimmy.doe@toetag.com
Restarting IP Filter... Done.
```

9.5.3 Removing an IP Address

To remove access for an IP address from the IPFilter firewall table for the HTTPS and SWSI Client ports, invoke the Configure IPFilter command as follows:

```
# ipfconfig -r 192.168.1.3

Removed 192.168.1.3 for port(s) <https_port> 3100
---192.168.1.3, <https_port>, 05/31/2003, 00:14:00, added by jstevens, Mission Alpha, Jenny
Doe, 555-867-5309, jenny.doe@toetag.com
---192.168.1.3, 3100, 05/31/2003, 00:03:45, added by jstevens, Mission Alpha, Jenny Doe,
555-867-5309, jenny.doe@toetag.com
Restarting IP Filter... Done.
```

If you attempt to remove an IP address that does not exist, the following error message will be displayed:

```
# ipfconfig -r 192.168.1.2

192.168.1.2 not found for port <https_port>.
```

192.168.1.2 not found for port 3100.

9.5.4 Adding or Removing an IP Address Interactively

The IPFilter Configure command in interactive mode allows the user to add and remove IP addresses for ports other than the HTTPS and SWSI Client ports. If invoked interactively, the IPFilter Configure command displays the following prompts for the user to input:

```
# ipfconfig -i

Entering Interactive Mode.
Enter Add or Remove (A/R)[A]: A
Enter IP Address: 192.168.1.3
Enter Port(s) [<https_port> 3100]: 3100
Enter comment: Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com
Added 192.168.1.3 for port 3100
---192.168.1.3, 3100, 05/31/2003, 00:03:45, added by jstevens, Mission Alpha, Jenny Doe,
555-867-5309, jenny.doe@toetag.com
Restarting IP Filter... Done.
```

Note, that the options shown in [] brackets are the default values. Interactive mode can be used to add or remove access for an IP address for a single port number or several port numbers:

```
# ipfconfig -i

Entering Interactive Mode.
Enter Add or Remove (A/R)[A]: R
Enter IP Address: 192.168.1.3
Enter Port(s) [<https_port> 3100]: 3100
Removed 192.168.1.3 for port(s) 3100
---192.168.1.3, 3100, 05/31/2003, 00:03:45, added by jstevens, Mission Alpha, Jenny Doe,
555-867-5309, jenny.doe@toetag.com
Restarting IP Filter... Done.
```

9.5.5 Listing All IP Addresses

The IPFilter Configure command can be invoked with the list option to show all the current IP addresses and the ports to which they have access:

```
# ipfconfig -l
```

The IPFilter Configure command displaying information similar to the following:

```
Port <https_port>
=====
172.16.3.15 ---172.16.3.15, <https_port>, 05/28/2003 17:32:36, added by jstevens,
John's Lab Host 301-777-9311
172.16.3.16 ---128.184.84.100, <https_port>, 05/28/2003 21:43:40, added by bsmith,
10.86.18.236 ---10.86.18.236, <https_port>, 05/30/2003, 19:46:18, added by jstevens,
192.168.1.3 ---192.168.1.3, <https_port>, 05/31/2003, 00:14:00, added by jstevens,
Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com
192.168.1.14 ---192.168.1.14, <https_port>, 05/30/2003, 19:31:12, changed by jstevens,
Testbed

Port 3100
=====
172.16.3.15 ---172.16.3.15, 3100, 05/28/2003 17:32:36, added by jstevens,
John's Lab Host 301-777-9311
172.16.3.16 ---172.16.3.16, 3100, 05/28/2003 21:43:40, added by bsmith,
10.86.18.236 ---10.86.18.236, 3100, 05/30/2003, 19:46:18, added by jstevens,
192.168.1.3 ---192.168.1.3, 3100, 05/31/2003, 00:03:45, added by jstevens,
Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com
192.168.1.14 ---192.168.1.14, 3100, 05/30/2003, 19:31:12, changed by jstevens,
Testbed
```

The output displays IP address sorted list with port number, date, time, and SWSI admin user, and a descriptive comment about the entry. Auditing of the IPFilter table should be done at least monthly to remove any unneeded IP addresses.

9.5.6 Searching for a Single IP Address

The IPFilter Configure command can be invoked with the find option to show only the IP address that you are interested in viewing:

```
# ipfconfig -f 192.168.1.3
```

The IPFilter Configure command displays information similar to the following:

```
---192.168.1.3/32, <https-port>, 05/28/2003, 00:14:00, added by jstevens, Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com  
---192.168.1.3/32, 3100, 05/28/2003, 00:03:45, added by jstevens, Mission Alpha, Jenny Doe, 555-867-5309, jenny.doe@toetag.com
```

9.6 RAID management

The Backend servers share a Sun A1000 RAID storage device. While the RAID is physically connected to both of the Backend servers at all times, only the server whose HA mode is set to PRIMARY is allowed to access the RAID. The servers and the RAID are all on the same SCSI bus. One of the servers (Spike) has a custom NVRAM script configured that changes its SCSI initiator ID from the standard value of 7 to a new value of 6.

The A1000 contains four 18 GB internal disks, for a total of 72 GB. The RAID has been divided into seven logical units (LUNs), each with a capacity of 7.386 GB. All of the LUNs have been configured for RAID Level 5.

The RAID is managed using the Sun StorEdge RAID Manager application, version 6.22. The RAID Manager application should only be run by *root*. Since it is a GUI application, the user should log in as root on the console of the server that owns the RAID (i.e., HA mode of PRIMARY). If neither server owns the RAID (neither server is in PRIMARY mode), then either may be used to manage the RAID.

When the RAID is owned by a server, the seven LUNs are mounted by an HA script in the directory */mnt*. The LUNs are mounted as *data0* through *data6*. All seven LUNs are used solely by the Oracle application for database storage.

When the RAID is unowned by either server, none of the LUNs are mounted. If it is necessary to mount the LUNs outside of HA control, the root user can use the */etc/init.d/mount-raid.sh* script to mount the LUNs. When finished, the LUNs can be unmounted by executing the */etc/init.d/umount-raid.sh* script. The LUNs must be unmounted before allowing either server to become PRIMARY.

The RAID Manager is run by logging in as *root* at the system console and executing the *rm6* alias in a Terminal window. Once the RAID Manager application has started, the Main Control Window as shown in Figure 9-2 will appear. Choosing *Configuration* from the Main Control Window brings up the Configuration Window as shown in Figure 9-3, which shows the seven LUN configuration of the RAID. Further information on use of the RAID Manager may be found in the *Sun StorEdge RAID Manager 6.22 User's Guide*.



Figure 9-2. RAID Manager Main Control Window

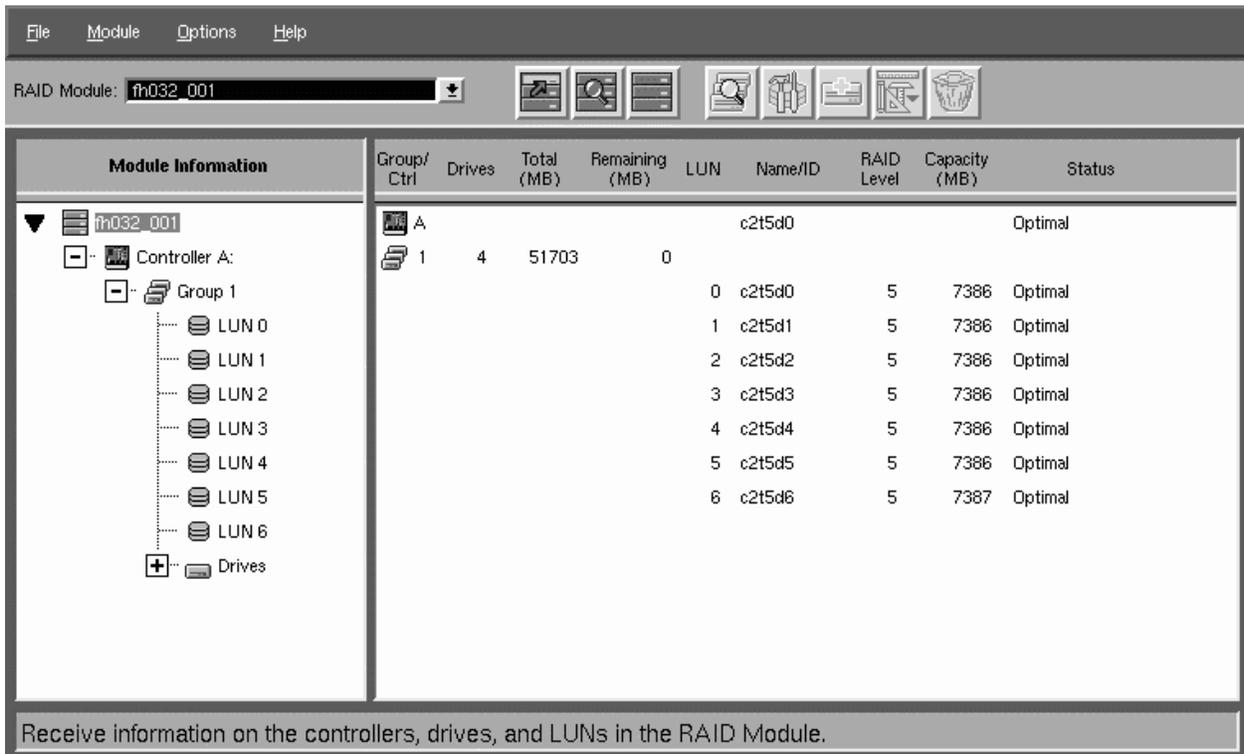


Figure 9-3. RAID Manager Configuration Window

9.7 cron Procedures

The following *cron* procedures are executed from the *root* account:

- *ntupdate* – run hourly on all servers to update the system time.

The following *cron* procedures are executed from the *oracle* account:

- *hotbackup.sh* – run daily only on Backend servers to backup Oracle databases and store on tape (see Section 9.4.4).
- *rotateHotLogs.sh* – run daily only on Backend server to manage log files produced by *hotbackup.sh*.

The following *cron* procedures are executed from the *swsiops* account:

- *s_runsenttut* – run hourly only on the Backend servers to send TUT data to the Open servers (see Section 3.6).
- *clean_tut_temp* – run daily only on the Open servers to remove temporary TUT web server files.
- *purge_databases* – run daily only on the Backend servers to purge old Schedule Requests and Active Events from all four SWSI database instances (see Sections 5.2, 7.5.6, and 7.5.9).

Appendix A. SNIF Log Messages

This appendix lists all the possible messages that can be logged by the SNIF and be displayed via the SNIF Delogger.

Table A-1. SNIF Log Messages (1 of 5)

Log Message	Explanation
<connection_name> <connection_type> connection established to <NCCDS_System_ID>	A connection was established with NCCDS; e.g., "LSAT-7 Realtime reconfig connection established to ANCC".
<connection_name> <connection_type> connection to <NCCDS_System_ID> closed	A connection with NCCDS was closed; e.g., "LSAT-7 Realtime reconfig connection to ANCC closed".
Bad queue message type received in <unit_name>: <message_type>	This is a software error that should be reported to SWSI development support.
Cycling <connection_name> pmData connection in preparation for upcoming event	An event is scheduled to occur within 5 minutes. The pmData connection has been closed and reopened to force the transmission of a UPD Request (UPDR) enable command to CCS.
Disabling Schedule Status Connection <connection_name>	A Schedule Status connection has been detected as having been disabled (see Section 7.3.10).
Duplicate value for MGCMR parameter <parameter_name>: <parameter_value>	This is a software error that should be reported to SWSI development support.
Enabling Schedule Status Connection <connection_name>	A Schedule Status connection has been detected as having been enabled (see Section 7.3.9).
Error in <unit_name> opening file <filename> for <reading/writing>	There was an error opening a TTM, RCDTM, or Active Schedule file for writing, or a TSW file for reading. One possible cause is a problem with one of the pathname properties (<i>RCTDMDirectory</i> , <i>TTMDirectory</i> , etc.) pointing to a directory that is not accessible by the SNIF application running as the <i>swsiops</i> user.
Error in <unit_name> writing to file <filename>: <error_condition>	There was an error writing to a TTM or RCDTM file. One possible cause is a problem with one of the pathname properties (<i>RCTDMDirectory</i> , <i>TTMDirectory</i> , etc.) pointing to a directory that is not accessible by the SNIF application running as the <i>swsiops</i> user.
Error in <unit_name>, Realtime Connection entry not found for SIC: <SIC>	An attempt was made to send a GCMR for a SIC which has not been configured for a Realtime Connection in the database (see Section 7.4).
Error in <unit_name>, Schedule Connection entry not found for SIC: <SIC>	An attempt was made to send a Schedule Request, State Vector, or TSW for a SIC which has not been configured for a Schedule Connection in the database (see Section 7.3).
Error in Isolator message: <message_contents>	This is a software error that should be reported to SWSI development support.
Error in Isolator MGCMR message: <message_contents>	This is a software error that should be reported to SWSI development support.
Error in make_ncc_connection looking up host <host_name>: <error_condition>	Unable to look up IP address for NCCDS hostname constructed from service name (<i>schReqName</i> , <i>schStatusName</i> , etc.) and <i>NCCDSDomain</i> properties.

Table A-1. SNIF Log Messages (2 of 5)

Log Message	Explanation
Error in poll_saved_requests, invalid TSW filename: <filename>	This is a software error that should be reported to SWSI development support.
Error in xxxxxxxx processing ID xxxxxxxx, Unknown Schedule Request type: xxxxxxxx	This is a software error that should be reported to SWSI development support.
Error initializing <connection_name> pmData connection, no SICs	A pmData connection could not be established with NCCDS because no SICs are assigned in the database to the connection (see Section 7.4.7).
Error initializing <connection_name> pmData connection, no SUPIDENs for SIC <SIC>	A UPD Request (UPDR) message could not be sent on a pmData connection to NCCDS because no SUPIDENs have been assigned for one of the SICs assigned to the connection (see Section 7.7).
Error parsing MGCMR parameters	This is a software error that should be reported to SWSI development support.
Error processing <SAR/ASAR/RR/SDR/WLR> ID #<request_id>, <error_condition>	This is a software error that should be reported to SWSI development support.
Error processing AFN message, invalid size: xxxxxxxx	An Acquisition Failure Notification message received from NCCDS was improperly formatted. SNIF discarded the message without processing it.
Error processing GCM Disposition message, <error_condition>	A GCM Disposition message from NCCDS was improperly formatted. SNIF discarded the message without processing and storing it.
Error processing GCM response, <error_condition>	A message received on the <i>reconfig</i> connection from NCCDS was improperly formatted. SNIF discarded the message without processing and storing it.
Error processing GCM Status message, <error_condition>	A GCM Status message from NCCDS was improperly formatted. SNIF discarded the message without processing and storing it.
Error processing Performance Data message, <error_condition>	A message received on the <i>pmData</i> connection from NCCDS was improperly formatted. SNIF discarded the message without processing it.
Error processing Schedule Request ID #<request_id>, <error_condition>	This is a software error that should be reported to SWSI development support.
Error processing Schedule Result, <error_condition>	A message received on the <i>schStatus</i> connection from NCCDS was improperly formatted. SNIF discarded the message without processing it and storing it.
Error processing SRM, <error_condition>	An SRM received from NCCDS was improperly formatted. SNIF discarded the message without processing it.
Error processing SRM, error updating status for ID <request_id>	Unable to update status for the Schedule Request referenced by the SRM, possibly because it doesn't exist in the SWSI database, or because of an invalid SRM result/explanation code. This may also happen if the customer has been configured for baseline support rather than full support.
Error processing State Vector ID #xxxxxxx, <error_condition>	This is a software error that should be reported to SWSI development support.
Error processing TSW File <filename>, SIC mismatch: SIC = <SIC>, SUPIDEN = <SUPIDEN>	This is a software error that should be reported to SWSI development support.
Error processing TSW File xxxxxxxx, <error_condition>	A Client user has submitted an invalid TSW. An alert has been sent to the user with a detailed description of the problem.
Error processing UPD ID #<message_id>, <error_condition>	A UPD message received from NCCDS was improperly formatted. SNIF discarded the message without processing it.

Table A-1. SNIF Log Messages (3 of 5)

Log Message	Explanation
Error processing User GCMR ID #<request_id>, <error_condition>	This is a software error that should be reported to SWSI development support.
Error processing USM, error adding event ID #<event_id> to Active Schedule	There was an error storing a USM in the database. SWSI development support should be notified.
Error processing USM: <error_condition>	A USM received from NCCDS was improperly formatted. SNIF discarded the message without processing and storing it.
Error processing USM: error storing USM	There was an error storing a USM in the database. SWSI development support should be notified.
Error renaming Active Schedule file <src_filename> to <dst_filename>: <error_condition>	There was an error moving an Active Schedule file to its permanent location. One possible cause is a problem with the <i>ActiveSchDirectory</i> property pointing to a directory that is not accessible by the SNIF application running as the <i>swsiops</i> user.
Error sending <request_type> on <connection_name> <connection_type>, assuming connection lost	There was an error sending a request message to NCCDS. The connection will be closed.
Error storing alert in database	This is a software error that should be reported to SWSI development support.
Error writing to Isolator #<isolator_number> socket: <error_condition>	There was an error writing a message to the Isolator. This error should not occur and should be reported to SWSI development support.
Fatal error allocating memory: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error connecting to Oracle	Unable to connect to Oracle for the configured database as specified by the <i>DatabaseName</i> , <i>OracleUsername</i> , and <i>OraclePassword</i> properties. Either the properties weren't properly specified, or there is some other problem preventing a connection, possibly because Oracle was not properly started.
Fatal error in <unit_name> creating <connection_type> socket: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> creating <thread_name> thread: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> detaching thread: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> initializing <condition_name> condition: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> initializing <mutex_name> mutex: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> locking <mutex_name> mutex: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> looking up service entry <entry_name>: <error_condition>	Unable to look up port number for service name as specified in a service name property (<i>schReqName</i> , <i>schStatusName</i> , etc.) SNIF will not run without valid values for these properties.
Fatal error in <unit_name> posting/jamming to <queue_name> Queue: Queue full	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> signaling <condition_name> condition: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> unlocking <mutex_name> mutex: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in <unit_name> waiting on <condition_name> condition: <error_condition>	This is a software error that should be reported to SWSI development support.

Table A-1. SNIF Log Messages (4 of 5)

Log Message	Explanation
Fatal error in main doing Isolator UDP socket bind: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error in make_ncc_connection setting <connection_type> socket option: <error_condition>	This is a software error that should be reported to SWSI development support.
Fatal error looking up Isolator readhost: <hostname>	Unable to look up IP address for hostname specified by <i>IsolatorReadHost</i> property. SNIF will not run without a valid value for this property.
Fatal error reading Realtime Connections database table	This is a software error that should be reported to SWSI development support.
Fatal error reading Schedule Connections database table	This is a software error that should be reported to SWSI development support.
Fatal error reading UPD packet descriptions, no entries	This is a software error that should be reported to SWSI development support.
Fatal error retrieving new messageid from Oracle	This is a software error that should be reported to SWSI development support.
Fatal error setting ORACLE_SID environment variable	This is a software error that should be reported to SWSI development support.
Fatal Oracle connect error, missing username and/or password	This is a software error that should be reported to SWSI development support.
Improperly formatted data length: <bad_length>	This is a software error that should be reported to SWSI development support.
Improperly formatted message, possibly a length problem	This is a software error that should be reported to SWSI development support.
Incorrect data length: <length>	This is a software error that should be reported to SWSI development support.
Invalid File Message data length: <length>	This is a software error that should be reported to SWSI development support.
Invalid File Message Type: <file_message_type>	This is a software error that should be reported to SWSI development support.
Invalid Key Message data length: <length>	This is a software error that should be reported to SWSI development support.
Invalid Key Message Index Number: <index_number>	This is a software error that should be reported to SWSI development support.
Invalid Key Message Type: <key_message_type>	This is a software error that should be reported to SWSI development support.
Invalid length for MGCMR parameter <parameter_name>: <bad_length> (should be <correct_length> characters)	This is a software error that should be reported to SWSI development support.
Invalid message type: <bad_type>	This is a software error that should be reported to SWSI development support.
Invalid MGCMR parameter: <parameter_name>	This is a software error that should be reported to SWSI development support.
Invalid NCC Message data length: <length>	This is a software error that should be reported to SWSI development support.
Invalid NCC Message Type: <ncc_message_type>	This is a software error that should be reported to SWSI development support.
Invalid NCCDS system code: <bad_system_code>; should be: <correct_system_code>	This is a software error that should be reported to SWSI development support.
Invalid or missing MGCMR <parameter_name> parameter	This is a software error that should be reported to SWSI development support.

Table A-1. SNIF Log Messages (5 of 5)

Log Message	Explanation
Invalid Performance Data message received, Message ID #<message_id>, Type <bad_type>, Class <bad_class>	An invalid message was received on the <i>pmData</i> connection from NCCDS. SNIF discarded the message without processing it.
Invalid Schedule Result received, Message ID #<message_id>, Type <bad_type>, Class <bad_class>	An invalid message was received on the <i>schStatus</i> connection from NCCDS. SNIF discarded the message without processing it.
Invalid sync pattern: <sync_pattern>	This is a software error that should be reported to SWSI development support.
Isolator alert message too long	This is a software error that should be reported to SWSI development support.
Isolator write message too long	This is a software error that should be reported to SWSI development support.
Message too short, length = <length>	This is a software error that should be reported to SWSI development support.
MGCMR <parameter_name> parameter not required for requested message type	This is a software error that should be reported to SWSI development support.
MGCMR SIC mismatch: SIC = <SIC>, SUPIDEN = <SUPIDEN>	This is a software error that should be reported to SWSI development support.
Schedule Request ID not found: <request_id>	This is a software error that should be reported to SWSI development support.
Schedule Request SIC mismatch: Database = <SIC>, Isolator = <SIC>	This is a software error that should be reported to SWSI development support.
SNIF <version> Launched	SNIF application started. This is the first log message written at startup time.
SNIF Application Exiting	SNIF application is terminating. This is the last log message written before exiting.
State Vector ID not found: <message_id>	This is a software error that should be reported to SWSI development support.
State Vector SIC mismatch: Database = <SIC>, Isolator = <SIC>	This is a software error that should be reported to SWSI development support.
Unable to open <connection_name> <connection_type> connection	Connection attempt with NCCDS failed; e.g., "Unable to open LSAT-7 Realtime reconfig connection"
User GCMR ID not found: <request_id>	This is a software error that should be reported to SWSI development support.
User GCMR SIC mismatch: Database = <SIC>, Isolator = <SIC>	This is a software error that should be reported to SWSI development support.

Appendix B. Abbreviations and Acronyms

AFN	Acquisition Failure Notification
ANCC	Auxiliary Network Control Center
ASAR	Alternate Schedule Add Request
CA	Certificate Authority
CCB	Configuration Control Board
CCR	Configuration Change Request
CCS	Communications and Control Segment
CDE	Common Desktop Environment
CGI	Common Gateway Interface
CNE	Center Network Environment
COTS	Commercial Off-The-Shelf
CSR	Certificate Signing Request
DAS	Demand Access System
DASCON	Demand Access System Controller
DASMAR	Demand Access System Multiple Access Return
DASPBK	Demand Access System Playback
DBA	Database Administrator
DBMS	Database Management System
DCN	Document Change Notice
DN	Distinguished Name
DSMC	Data Services Management Center
EETF	End-to-End Test Forward
EETR	End-to-End Test Return
EIF	Engineering Interface
FDF	Flight Dynamics Facility
FOUO	For Official Use Only

GCMR	Ground Control Message Request
GN	Ground Network
GOTS	Government Off-The-Shelf
GSFC	Goddard Space Flight Center, Greenbelt, MD
GUI	Graphical User Interface
HA	High Availability
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICD	Interface Control Document
IIRV	Improved Interrange Vector
IONET	Internet Protocol Operational Network
IP	Internet Protocol
JAR	Java Archiver
JDK	Java Development Kit
JRE	Java Runtime Environment
JVM	Java Virtual Machine
KaSAF	Ka-Band Single Access Forward
KaSAR	Ka-Band Single Access Return
KaSARWB	Ka-Band Single Access Return Wideband
KSAF	K-Band Single Access Forward
KSAR	K-Band Single Access Return
LOP	Local Operating Procedure
LUN	Logical Unit
MA	Multiple Access
MAF	Multiple Access Forward
MAR	Multiple Access Return
MOC	Mission Operations Center
MSP	Mission Services Program
NCC	Network Control Center

NCCDS	NCC Data System
NISN	NASA Integrated Services Network
NPG	NCCDS Protocol Gateway
NVRAM	Non-Volatile RAM
OPS	Operations
PBKDR	Playback Deletion Request
PBKMR	Playback Modification Request
PBKR	Playback Request
PEM	Privacy Enhanced Mail
PGP	Pretty Good Privacy
PKI	Public Key Infrastructure
RA	Registration Authority
RADR	Resource Allocation Deletion Request
RAID	Redundant Array of Inexpensive Disks
RAM	Random Access Memory
RAMR	Resource Allocation Modification Request
RAR	Resource Allocation Request
RCTD	Return Channel Time Delay
RR	Replace Request
SA	Single Access
SAR	Schedule Add Request
SDIF	SWSI-DAS Interface
SDR	Schedule Delete Request
SIC	Support Identification Code
SMAF	S-Band Multiple Access Forward
SMAR	S-Band Multiple Access Return
SN	Space Network
SNIF	SWSI-NCCDS Interface
SPS	Service Planning Segment

SRM	Schedule Result Message
SSAF	S-Band Single Access Forward
SSAR	S-Band Single Access Return
SSC	Service Specification Code
SSL	Secure Socket Layer
STDN	Spaceflight Tracking and Data Network
STGT	Second TDRSS Ground Terminal
SUPIDEN	Support Identifier
SWSI	SN Web Services Interface
TBD	To Be Determined
TBS	To Be Supplied
TCP	Transmission Control Protocol
TDRS	Tracking and Data Relay Satellite
TDRSS	Tracking and Data Relay Satellite System
TGBFS	Third Generation Beamforming Subsystem
TSW	TDRS Scheduling Window
TTM	Time Transfer Message
TUT	TDRSS Unscheduled Time
UDP	User Datagram Protocol
UPD	User Performance Data
UPDR	User Performance Data Request
UPS	User Planning System
USM	User Schedule Message
UT	Universal Time
UTC	Universal Time Code
WLR	Wait List Request
WSC	White Sands Complex
WSGT	White Sands Ground Terminal
WWW	World Wide Web

452-SOG-SWSI

**Space Network (SN)
Web Services Interface (SWSI)
Server Operator's Guide
Release 04.1**

DCN 1