Pass Operations Logging and Anomaly Reporting Interface System (POLARIS) for the Solar and Heliospheric Observatory (SOHO) Detailed Design Specification

October 4, 2006

Version 2

Prepared for:
NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771

Prepared By:
Honeywell Technical Solutions Inc.
Lanham, Maryland 20706

Table of Contents

Table of Figures

Table of Tables

1. Introduction

1.1 Document Purpose and Scope

This document describes the design of the POLARIS for SOHO automation software. It describes the hardware and software components of POLARIS and how they will enable the SOHO project to automate mission operations. The document gives a high-level description of each component, followed by detailed design information such as interface specifications and internal interaction of the software.

1.2 Reference Documents

A list of numbered references is written at the end of this document. Throughout this document, references are given as RD and the number as shown in the list. They are used inline as [RD1], [RD2], etc.

2. Design Overview

2.1 Ground System Description

The current SOHO telemetry and command (T&C) system is the Transportable Payload Operations Control Center (TPOCC). TPOCC is typically configured to run in a way that requires at least two computers. A "Front End Processor" (FEP) is where the majority of the telemetry, command, and report processing occurs. A workstation is used for controlling the software, sending commands, viewing telemetry, viewing reports, and running procedures written in the TPOCC System Test and Operations Language (TSTOL).

To initialize the TPOCC software, a workstation user triggers a startup script called *soho_start_fe* using an icon provided for that purpose. The user picks which FEP to use. The script remotely starts all of the FEP software required, and then starts the local workstation software.

On workstation software initialization, two important processes start: the SOHO TSTOL server and the X-Windows TPOCC display program (xtpdsp). Among other things, xtpdsp provides a command line to execute TSTOL directives. The xtpdsp program connects to the TSTOL server via a socket, over which it sends TSTOL directives typed on the command line. It also receives responses to those directives. For example, if you send the directive "*start o_sohoup*", you will receive all of the procedure output as responses.

When xtpdsp or any other client connects to the TSTOL server, the TSTOL server starts another process called the TSTOL Parser. The parser handles all subsequent communications. The TSTOL Server is then free to accept other connections, which in turn spawn more parsers.

After connecting, xtpdsp directs the TSTOL server to connect via a socket to a FEP process called the State Manager. The State Manager handles communication between TSTOL and all other TPOCC processes.

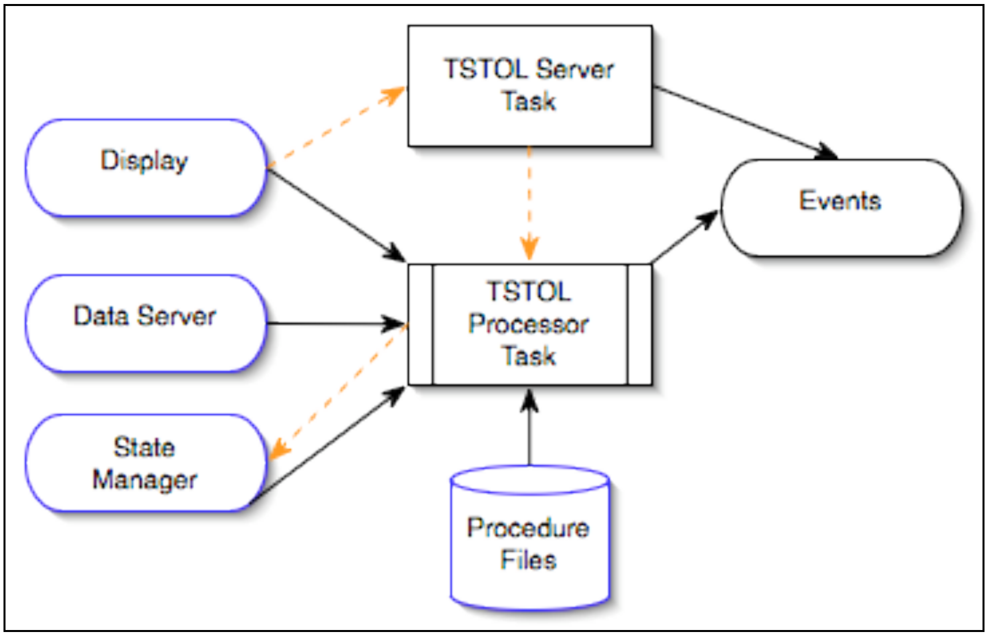

Figure 1: TPOCC Server/Processor Structure (RD 1)

2.2 POLARIS Design Overview

POLARIS will automate the SOHO mission by performing the following tasks (see Figure 3 and Figure 5):

- Retrieving the DSN schedule at periodic intervals
- Generating a pass plan based on the DSN schedule
- Starting the TPOCC software at the scheduled time using *cron*
- Interfacing directly with the TPOCC software via a socket
- Running pass operations at the appropriate times
- Monitoring TPOCC events for anomalies
- Notifying the FOT and/or instrument teams for anomalies
- Killing the TPOCC software after pass completion
- Maintaining logs of all activities

POLARIS will retrieve the current DSN schedule from the CMS. It will then generate an Activity Plan. The user will add or change activities and resubmit the file and generate a Pass Execution Procedure with all necessary checks and pass activities. POLARIS will start the pass procedure via *cron*, which will call the needed programs at the proper times. For example, it can start the TPOCC software, send TSTOL directives such as starting procedures, or run shell scripts.

At the scheduled time, POLARIS will start the TPOCC software using a Perl script. This script will read a resource file, *initialization.rc*, to determine what tasks to perform for initialization. A check will be done by POLARIS to verify that the software is not already running and attempt initialization. POLARIS will verify that all TPOCC software has initialized successfully. If any component fails, POLARIS will terminate all TPOCC software and make another attempt to start it. Any further failures will trigger an automated alert to the FOT. Successes and failures will be written to a log file and sent to the Attention! server.

POLARIS will then start the POLARIS TPOCC interface (PTI). PTI duplicates the functionality of the existing xtpdsp TPOCC process. Like xtpdsp, PTI connects to the TSTOL Server, which spawns a TSTOL parser process, to send directives, which the State Manager will then execute. However, instead of providing a command line on which to type TSTOL directives, PTI accepts TSTOL directives via a socket interface. An external program called *execute_TSTOL_directive* will connect to this socket to send directives specified on its command-line (e.g. *execute_TSTOL_directive "start o_sohoup"*).

POLARIS will then configure the software for the upcoming pass. This includes configuring the station, orbit number, and IRTS monitors. It will then run the custom procedure generated to perform all needed pass activities.

During the pass, TPOCC will log all events to a text file in the */home/soho/ops/output/reports* directory using the "EVTRPT" TSTOL directive (RD5). This includes any TSTOL directives run from the local terminal by a user while POLARIS is running. The file name convention will be: *POLARIS_evtrpt_YYYY_jjj_hhMMss.log* where *YYYY* is the year, *jjj* is the day of year, *hh* is the hours, *MM* is the minutes, and *ss* is the seconds for when the event report was started. See Figure 2.

The Event Monitoring/Notification/Response component of POLARIS will monitor TPOCC events for anomalies such as limit or configuration monitor violations, procedure failures, telemetry dropouts, ranging failures, and other anomalies. It will notify the FOT or perform an automatied response. If the FOT member does not respond, POLARIS will escalate notification to the next person or group in the escalation chain defined in its configuration file.

At the end of each pass, the system will be terminated. A Perl script will be run to perform the system termination. This script will read a user-customized file, *termination.rc*, containing the termination tasks. This includes cleaning up the TPOCC software and generation of pass summary and trending reports. If a termination fails, then it should attempt another termination.

All POLARIS activities are logged in a file in the directory */home/soho/ops/POLARIS/logs*. The format of the filename will be: *POLARIS_system_YYYY_jjj_hhMMss.log*, where *YYYY* is the year, *jjj* is the day of year, *hh* is the hours, *MM* is the minute, and *ss* is the seconds of initialization of POLARIS. See Figure 2.

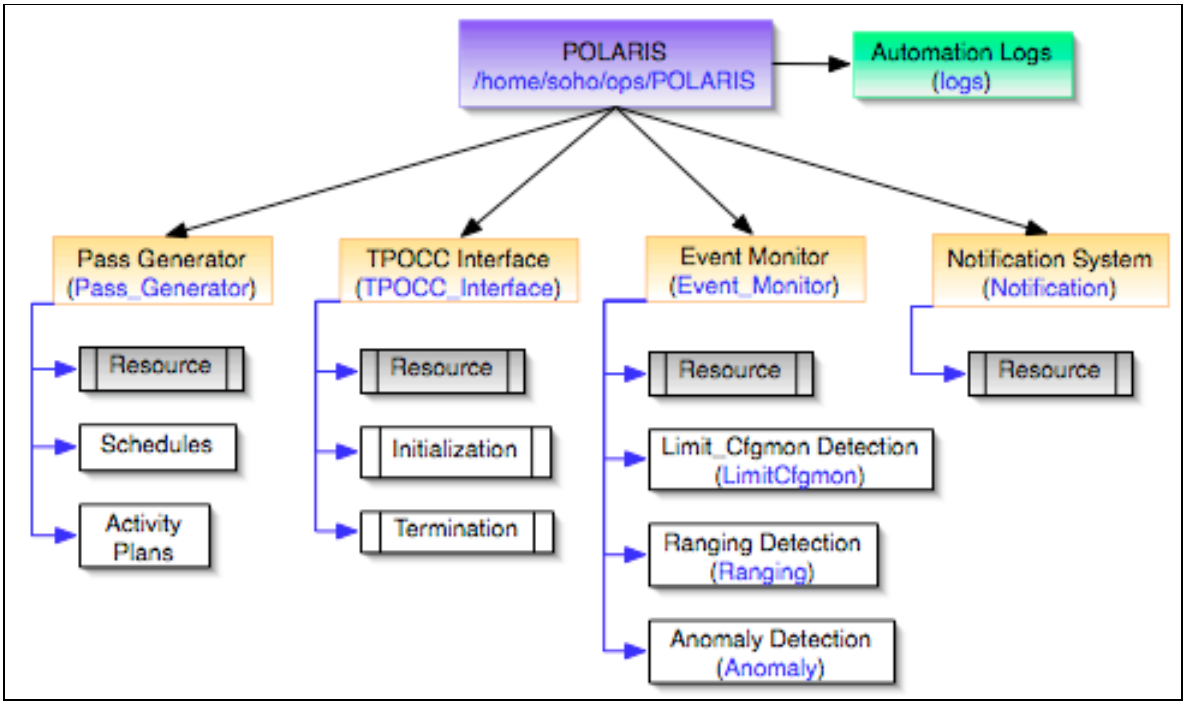

Figure 2: Automation Software Directory Structure

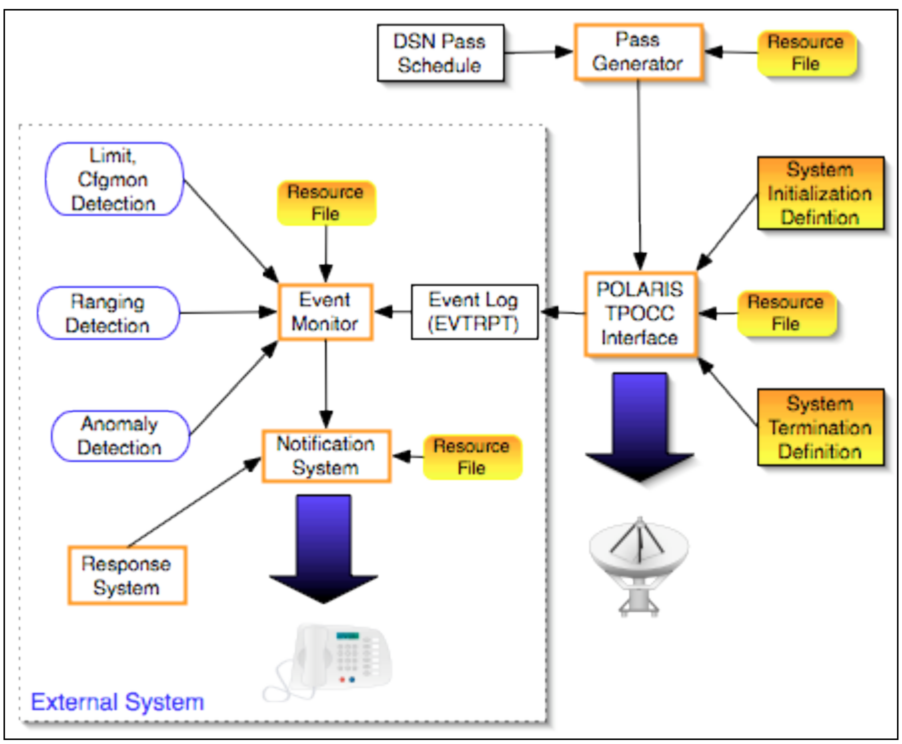

Figure 3: Automation System Interaction

2.2.1 Architecture

The POLARIS software will run on HP-UX 11.11. The system utilizes X-Windows, a graphical user interface (GUI) that helps bridge the gap between the user and the UNIX environment. SOHO uses the Common Desktop Environment (CDE), a cooperative arrangement between HP and Sun. [RD2]

The Event Monitor, Notification, And Response Systems will run on dual servers. These servers will run in hot backup. The SQL database will run on a separate server. A fourth server will have a limited version of Attention! with the sole purpose of monitoring the SQL server.

2.2.2 External Interfaces

POLARIS interfaces with the SOHO ground system, TPOCC software, and other IMOC components. On initialization, POLARIS will connect to the soho_tstol socket interface documented in RD1 to send TSTOL directives. It will also receive directive responses on the same socket.

Instead of a GUI or command-line interface like that provided by the TPOCC software, POLARIS will accept TSTOL and control directives on a socket. This will allow the FOT to schedule events using external programs such as *cron*. The interface specification will be documented in section 3.1.2.

If external host connections are required (between open and restricted IONETs), user authentication will be provided.

2.2.3 Hardware

The automation software will run on an HP B2600 workstation serving as a supplementary operator console. This workstation is a 500 MHz PA-8600 processor with 512 MB of memory, a 73 GB external hard drive, and a DVD-ROM drive. The hostname for this workstation on strings 4, 5, and 6 are *sh4ws2*, *sh5ws2*, and *sh6ws2*, respectively.

An HP J282 workstation serves as a TPOCC front-end processor (FEP) and file server. The front-end performs the main functions of the TPOCC software. Each front-end has 512MB of memory, a 73 GB external hard drive, a CD-ROM drive, and a 4-mm tape drive. The hostnames for the front-end workstations on strings 4, 5, and 6 are sh4fe1, sh5fe1, and sh6fe1, respectively.

The Event Monitor, Notification, And Response Systems will run on dual servers. These servers will run in hot backup. The SQL database will run on a separate server. A fourth server will have a limited version of Attention! with the sole purpose of monitoring the SQL server.

2.2.4 Network Design

POLARIS will run on one of the strings in the IMOC. In Figure 4, the hardware connections between each of the strings are shown. In Figure 5, the network layout of POLARIS itself is shown.

2.2.5 Requirements Met

1, 2, 3, 4, 27, 28, 30, 46, 47, 50, 51

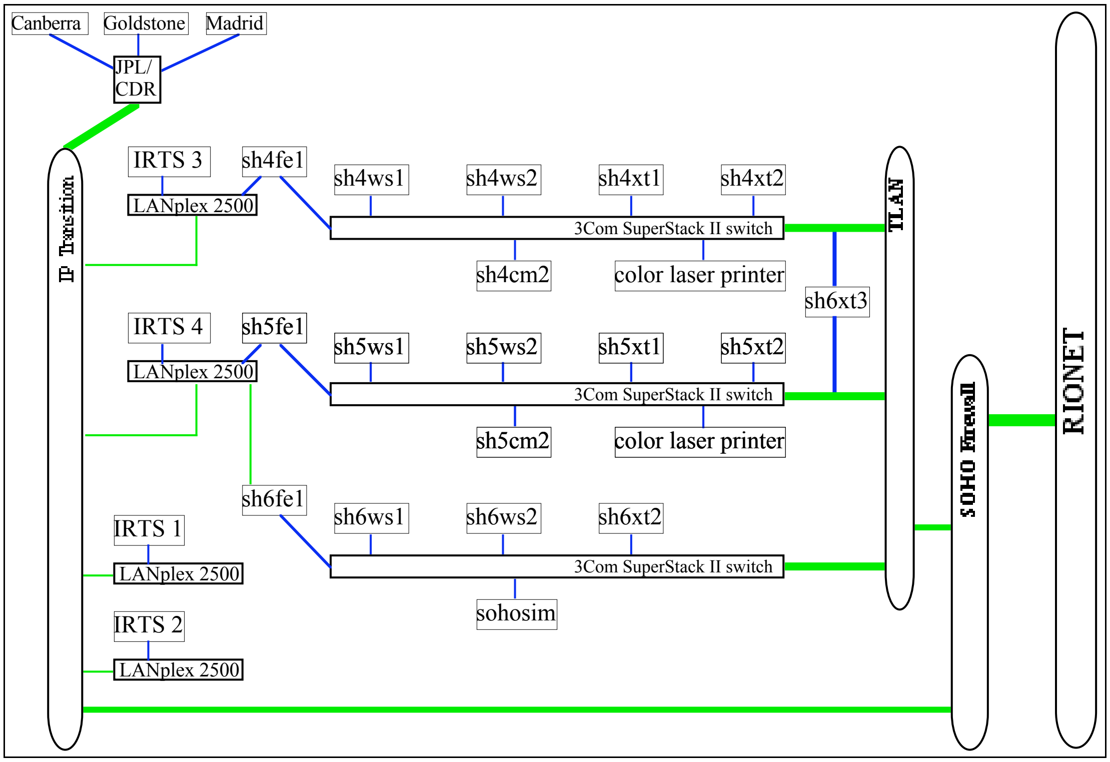

Figure 4: Network Layout of IMOC

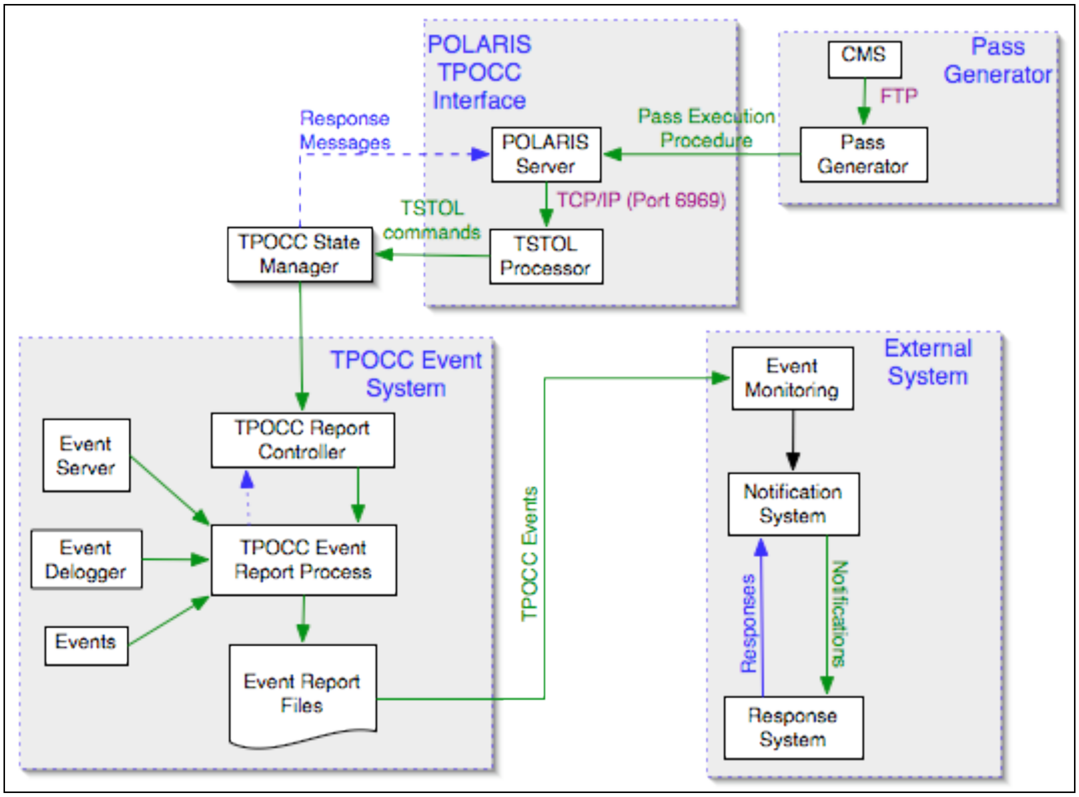

Figure 5: Network Layout of POLARIS (RD1)

3. Design Description

3.1 POLARIS TSTOL Interface

3.1.1 Overview

The POLARIS TSTOL Interface (PTI) is a program that essentially duplicates the functionality of the existing xtpdsp TPOCC process. Like xtpdsp, PTI connects via a socket to the TSTOL server, which spawns a TSTOL parser process. PTI uses the TSTOL parser process to send directives, which the TSTOL State Manager will then execute. However, instead of providing a command line on which to type TSTOL directives, PTI accepts TSTOL directives via a socket interface. An external program called *PTI_client.pl* will connect to this socket to send directives specified on its command-line. (e.g. *PTI_client.pl "start o_sohoup"*)

PTI can run concurrently with xtpdsp, which means the FOT can still type manual directives on the xtpdsp command line.

As noted above, the PTI maintains at least two network communication sockets: one for the TSTOL server, and one or more for external clients. In essence, PTI is a multi-user proxy to the TSTOL server. RD1 specifies that communication with the TSTOL server is *one-way*; i.e. clients should make no assumptions about responses to directives. For example, a single “start procedure” directive can result in multiple responses; the “directive completed” message and the procedure output.

PTI manages and maintains two internal communications queues, implemented as hashes. The hash key is a reference to the associated socket.

Queue name	Source	Destination	Purpose
ready	PTI_client.pl	TSTOL Server	When PTI reads from an PTI_client.pl, it places the data read into this queue. PTI reads the data for internal processing (e.g. checking for special commands to PTI), then places it in the outbuffer to be forwarded to TSTOL Server.
ready	TSTOL Server	internal	When PTI reads from the TSTOL Server, it checks the data for special conditions (see 3.1.2.2), but otherwise ignores it.
outbuffer	(internal processing)	TSTOL server	When the hash key is $tstol_server, this queue holds outgoing directives to the TSTOL server.
outbuffer	(internal processing)	TSTOL server	When the hash key is not the $tstol_server, this queue holds outgoing status (success/failure) messages to PTI_client.pl.

Table 1: PTI Communication Queues

The PTI will maintain internal state information for each directive, consisting of:

Name	Description
directive-text	The actual text received from the external client
receive-time	The time at which the directive was received, with 1-second resolution or greater
sent-to-tstol	True if directive was sent to the TSTOL server
procedure-directive	True if directive was a “start” directive
return-status	Current status of directive, where 0 = complete, 1 = error

Table 2: PTI Internal State Information

3.1.2 External Interfaces

The PTI has several external interfaces, listed in the table below.

Source	Destination	Data Flow	Description	Transfer medium	When	Data volume	Controlling ICD
PTI	TSTOL server	TSTOL directives (XDR)	Directives executed by the TPOCC software	Network	As needed	Max *x* bytes per directive	RD1
TSTOL_ server	PTI	TSTOL responses	Responses to directives	Network	As needed	TBD	RD1
External client	PTI	TSTOL directives	Directives executed by the TPOCC software	Network	As needed	Max *x* bytes per directive	This document
PTI	External client	Status	Whether directive was sent to TSTOL	Network	As needed	1 byte	This document

Table 3: PTI External Interfaces

3.1.2.1 PTI External Client Interface

PTI expects external clients to pass it an XDR-encoded TSTOL directive. PTI performs no validation whatsoever on the directive. It does, however, check for special PTI internal directives (see table below). If the directive is not an internal PTI command, it merely passes the directive to the TSTOL server, and returns success upon doing so. **Note:** PTI does *not* wait for a response from the TSTOL server.

Name	Description
EXITPTI	Exits the PTI software. **Note:** Any queued outgoing messages (for either any connected PTI clients or the TSTOL server) are still sent.
PAUSEPTI	PTI is paused; i.e. it will not send *any* directives to the TSTOL server. Note, however, that any currently running procedures will still run. When PTI is paused, PTI will return a status to PTI_client.pl of "PTI IS PAUSED".
RESUMEPTI	Resume regular operation. Multiple "resume" directives have no ill effect.

Table 4: PTI Special Internal Directives

3.1.2.2 PTI TSTOL Server Interface

PTI processes the output from the TSTOL server for special cases such as, but not limited to:

- Critical command responses (PTI needs to send /ALLOW)
- Sending loads (PTI needs to send /SEND)
- ASK prompts (PTI logs an error)
- Client permission responses (e.g. when requesting CC permissions)

3.1.3 Data flow

Figure 6 shows the program logic for PTI.

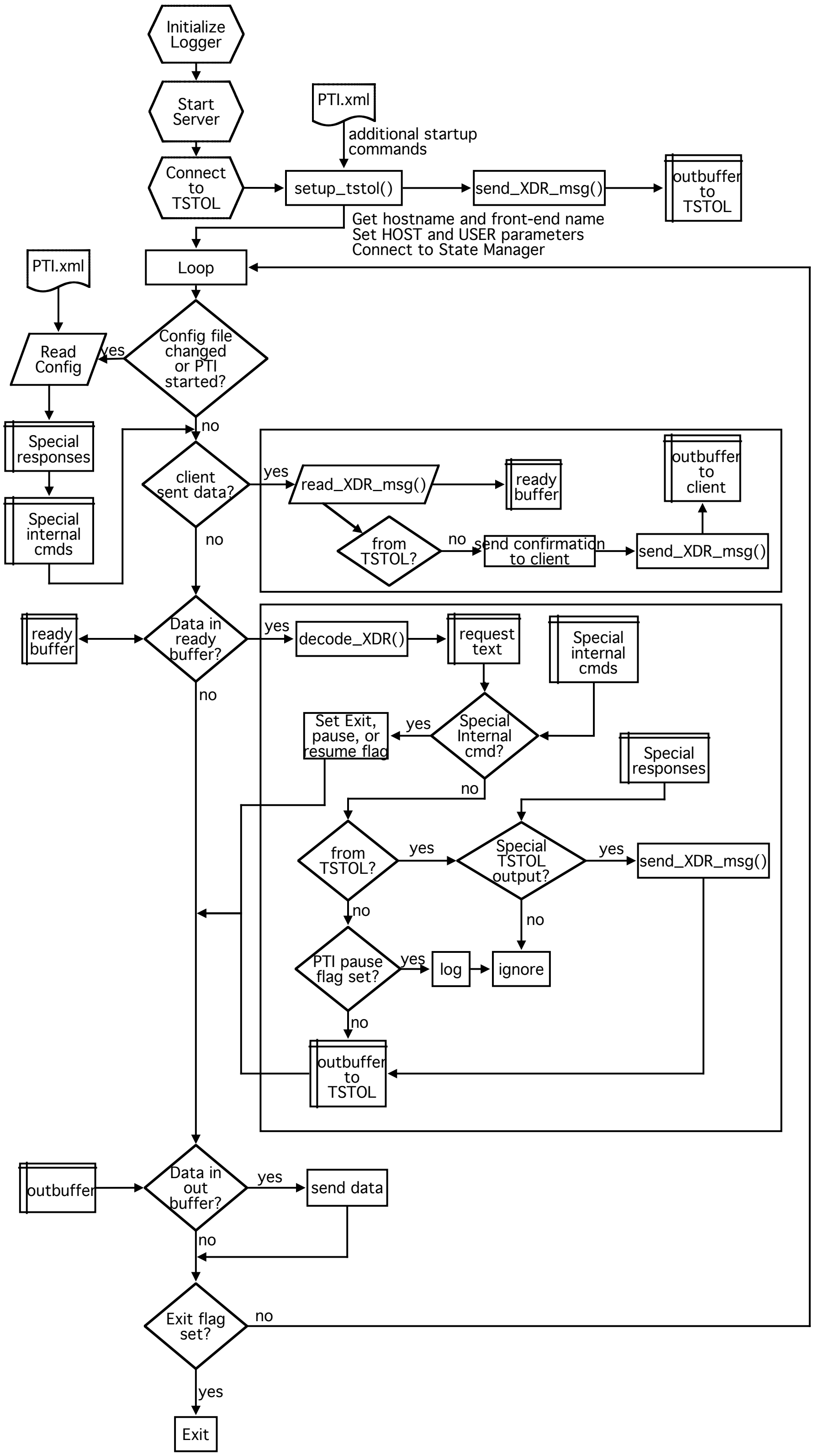

Figure 6: PTI Program Logic

3.1.4 Fault Tolerance

In the event of PTI failure, an external process monitor will restart it. However, some pass configuration information might be lost (TBD). Critical functions such as starting procedures will still work. Restarting PTI after a failure is to be determined.

3.1.5 Requirements Met

5, 16, 17, 18, 26

3.2 GUI

3.2.1 Overview

The automation system does not require a command interface to run, but it is reasonable that at certain times the FOT would need to perform simple tasks with the automation software. These include enabling or disabling POLARIS, displaying the TPOCC privileges assigned to POLARIS, and other tasks to be determined.

The GUI will be written using the Tk extensions to Perl. The modern distributions of Perl already come with Tk extension with ports to both Unix and Windows. In order for the GUI to send commands to TPOCC, it will need to form a connection similar to the TPOCC interface.

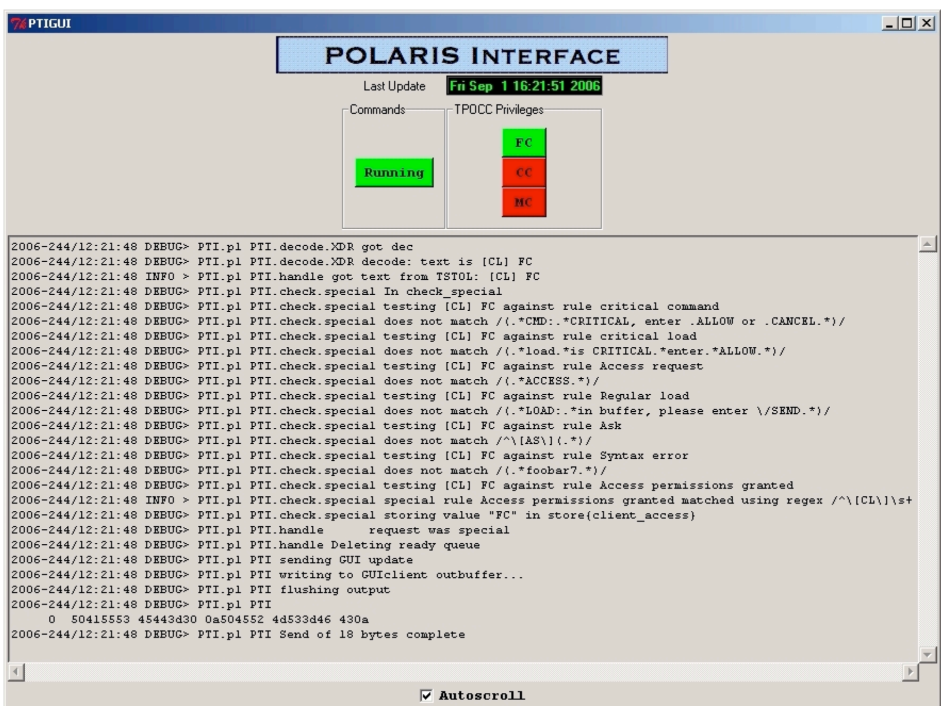

Figure 7: POLARIS GUI

3.2.2 External Interfaces

Because the GUI is written in Perl/Tk, it already has a direct method of interacting with the operating system through callback functions. It is more difficult to interact with TPOCC itself, though.

3.2.3 Data flow

TBD

3.2.4 Fault Tolerance

A failure of the GUI limits the users ability to quickly start or stop POLARIS. The GUI will call Perl scripts that will intercede the Start and Stop issuances. The user may simply run these scripts to change the POLARIS state.

The inability to see POLARIS privileges has no direct effect unless the privileges were incorrectly set in the first place. This is unlikely to happen, as the *o_sohoup* procedure will set the privileges correctly. The *o_sohoup* procedure is used to configure the string for a pass.

The event log is a text file found in */home/soho/ops/output/reports/* (see section 2.2). The user may read any of the POLARIS or TPOCC event logs by manually viewing them.

3.2.5 Requirements Met

6, 19, 20

3.3 Pass Generator

The Pass Generator creates the scheduled activities for SOHO passes. It will also perform a check that activities meet MDI and DHSS transition rules and timing requirements. The ideology behind the Pass Generator is derived from the usage of Perl scripts by the ACE (Advanced Composition Explorer) FOT to create pass procedures. The SOHO FOT takes this one step further by using Perl to help run unattended passes.

3.3.1 Overview

The automation software takes in a raw DSN pass schedule for the next week. The location of the schedule is stated in the Pass Generator's resource file *pass_generator.rc*. The DSN schedule is a text file with station, tracking, and ranging information as seen in Figure 8. The pass generation software can extract the necessary field information and create the Activity Plan.

The Pass Generator will read the DSN schedule from the CMS. The SOHO Scheduler transfers the schedule manually to the CMS. The Pass Generator will create a skeleton set of pass activities called the Activity Plan. A resource file, *procedure_defaults.rc*, is used to determine the default arguments for each procedure. The Activity Plan is based on SOE and time of pass. The Activity Plan will be a text file that is easily imported into a spreadsheet program such as Microsoft Excel. Based on times, the Pass Generator can determine VC2, dump, record, system initialization, and system termination times. This provides the user with a basic framework for pass operations.

```
*                        DSN 7-DAY OPERATIONS SCHEDULE
*                              ACTIVITIES LISTING
*                     WEEK NO. 15 *** 10 APR 06 - 16 APR 06
*--------------------------------------------------------------------------------
*DAY START BOT  EOT  END FACILITY USER   ACTIVITY         PASS  CONFIG/ WRK A C
*                                                        NO.   SOE      CAT R F
*--------------------------------------------------------------------------------
 100 0540 0605 1325 1335  DSS-66  SOHO   VC4 SSR DUMP      151  NONE  B 3C1

 100 1225 1310 2345 0000  DSS-27  SOHO   T/P CONTINUOUS    152  N131  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 100 2245 2325 0545 0555  DSS-46  SOHO   T/P CONTINUOUS    153  NONE  A 3C1

 101 0445 0545 1320 1335  DSS-54  SOHO   T/P CONTINUOUS    154  N083  H 1A1
          STXL,SHMT,UPL,CCP,RNG,NMC,RRPA=2,TLPA=2;
 101 1240 1310 1445 1500  DSS-15  SOHO   TP DL CONTINUOUS  155  N073  K 1A1
          SHMT,NMC,RRPA=2,TLPA=2;
 101 1350 1435 2345 0000  DSS-27  SOHO   T/P CONTINUOUS    156  N131  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 101 2240 2325 0620 0635  DSS-34  SOHO   T/P CONTINUOUS    157  N131  J 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 102 0520 0600 1600 1610  DSS-66  SOHO   VC4 SSR DUMP      158  NONE  A 3C1

 102 1455 1540 2345 0000  DSS-27  SOHO   EUNIS LAUNCH BU   159  N131  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 102 1515 1540 2325 2335  DSS-16  SOHO   EUNIS LAUNCH      160  NONE  B 3C1

 102 2245 2325 0540 0550  DSS-46  SOHO   T/P CONTINUOUS    161  NONE  A 3C1

 103 0535 0600 1555 1605  DSS-66  SOHO   VC4 SSR DUMP      162  NONE  B 3C1

 103 1450 1535 2340 2355  DSS-27  SOHO   T/P CONTINUOUS    163  N131  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 103 2240 2320 0540 0550  DSS-46  SOHO   T/P CONTINUOUS    164  NONE  A 3C1

 104 0530 0555 1555 1605  DSS-66  SOHO   VC4 SSR DUMP      165  NONE  B 3C1

 104 1435 1535 2210 2225  DSS-24  SOHO   T/P CONTINUOUS    166  N113  H 1A1
          STXL,SHMT,UPL,CCP,RNG,NMC,RRPA,RRPB,TLPA=2;
 104 2125 2150 0540 0550  DSS-46  SOHO   T/P CONTINUOUS    167  NONE  B 3C1

 105 0530 0555 1325 1335  DSS-66  SOHO   VC4 SSR DUMP      168  NONE  B 3C1

 105 1220 1305 2330 2345  DSS-27  SOHO   T/P CONTINUOUS    169  N089  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA=2,TLPA=2;
 105 2245 2310 0535 0545  DSS-46  SOHO   T/P CONTINUOUS    170  NONE  B 3C1

 106 0455 0525 0605 0605  DSS-32  SOHO   INTERFACE TEST    171  NONE  - 1A2

 106 0515 0555 1550 1600  DSS-66  SOHO   VC4 SSR DUMP      172  NONE  A 3C1

 106 1445 1530 2330 2345  DSS-27  SOHO   T/P CONTINUOUS    173  N131  V 1A1
          STXL,SHMT,UPL,CCP,NMC,RRPA,RRPB,TLPA=2;
 106 2230 2310 0535 0545  DSS-46  SOHO  T/P CONTINUOUS   174  NONE  A 3C1
```

Figure 8: Sample DSN Pass Schedule

The Pass Generator verifies that these transitions are correct by comparing the initial and final DHSS and MDI modes for each procedure. The rules for these interactions are seen in Table 5 and Table 6, and are found in the *transition_rules.rc* resource file. The Pass Generator will flag to the user any incorrect transitions and allow the user to make the changes.

DHSS Mode	MDI Flags
0: Survival	0: Idle (Coherency Swap)
1: HK	2: VC2
2: Science	3: VC3
3: Idle	4: Full Idle (All flags are set to 0)
4: Record	
5: Playback	
6: MDI-M	
7: MDI-H	

Table 5: DHSS and MDI modes

Procedure	Initial DHSS mode	Final DHSS mode	Initial MDI mode	Final MDI mode
d_ssr_record	3,7	4	0,3,4	0
d_ssr_stp_rc	4	3	4	4
d_ssr_dump	3	5	4	4
d_ssr_stp_dmp	5	3	4	4
k_idtomm	3	6	4	4
k_idtomh	3,4	7	0	3
k_mmtomh	6	7	2	3
k_mhtomm	7	6	3	2
k_mmtoidle	6	3	2	4
k_mhtoidle	7	3	3	4

Table 6: DHSS and MDI transition rules

Once the Activity Plan has been generated, the user can add activities. The Activity Plan is resubmitted to the Pass Generator as seen in Figure 10. The generation, verification, and actions of the Activity Plan are logged in the file *POLARIS_pass_generator_JJJYYYY_HHMMSS.log*.

The completed Activity Plan is then converted to a single TSTOL procedure with individual procedures timed between each other and checks for spacecraft receiver lock status, command status, and telecommand block synchronization. The TSTOL procedure is called the Pass Execution Procedure. A sample set of pass activities for a ranging pass on D46 at 0340 to 0910 is shown in Figure 9.

```
0240 Initialize Automation
WAIT UNTIL 0344
WAIT UNTIL P@MF_QUALITY = GOOD
WAIT UNTIL P@RSLOK1 = LOCK
0346 [RCTX1COH command by SOHO]
WAIT UNTIL P@RSLOK1 = LOCK
WAIT UNTIL P@RSCON1 = AUTH
0350 /DZDUMMY
0351 d_ssr_stp_rc
0353 k_idtomm
0400 k_obt_dist(SWAN,CELIAS)
0410 k_mmtoidle
0414 d_ssr_dump(y)
WAIT UNTIL P@DKSSNMOD = "STAND-BY"
0625 d_ssr_stp_dmp
0630 k_idtomm
0647 k_mmtomh
0900 k_mhtoidle
0905 d_ssr_record(y,y)
0920 Terminate Automation
```

Figure 9: Sample Pass Activities

The OE will check the Activity Plan for accuracy and make edits if necessary. The edited Activity Plan is then sent back to the Pass Generator, which will create a Pass Execution Procedure named *o_yyddd_auto_vxx.prc* in the *~/input/procs/Automation* directory for all pass activities. The Pass Generator will reference a configuration file containing procedure names available for automated execution. If the procedure name is NOT contained in the configuration file, the procedure will NOT be inserted into the pass procedure. It is then distributed to all strings. When a new Pass Execution Procedure is generated, the Pass Generator will clean up the previous versions of the Pass Execution Procedure and Activity Plan on each string. In Table 8, a sample Activity Plan is listed. In Table 7 a description of each section of the Activity Plan is listed.

Time	The UTC time that the activity takes place.
Activity	The TSTOL procedure, command, directive, program, or script set to execute at the given time.
Arguments	Any arguments that should be passed to the listed activity.
Initial DHSS Mode	The DHSS mode prior to the activity. (See Table 5).
Final DHSS Mode	The DHSS mode after the activity. (See Table 5).
Initial MDI Mode	The MDI mode prior to the activity. (See Table 5).
Final MDI Mode	The MDI mode after the activity. (See Table 5).

Table 7: Activity Plan Description

Year	DOY	Time	Activity	Arguments	Initial DHSS Mode	Final DHSS Mode	Initial MDI Mode	Final MDI Mode
2007	288	0240	Initialization	N/A	Record (4)	Record (4)	Idle (4)	Idle (4)
2007	288	0350	/DZDUMMY		Record (4)	Record (4)	Idle (4)	Idle (4)
2007	288	0351	d_ssr_stp_rc		Record (4)	Idle (3)	Idle (4)	Idle (4)
2007	288	0353	k_idtomm		Idle (3)	MDI-M (6)	Idle (4)	VC2 (2)
2007	288	0400	k_obt_dist	SWAN,CELIAS	MDI-M (6)	MDI-M (6)	VC2 (2)	VC2 (2)
2007	288	0410	k_mmtoidle		MDI-M (6)	Idle (3)	VC2 (2)	Idle (4)
2007	288	0414	d_ssr_dump	Y	Idle (3)	Playback (5)	Idle (4)	Idle (4)
2007	288	0625	d_ssr_stp_dmp		Playback (5)	Idle (3)	Idle (4)	Idle (4)
2007	288	0630	k_idtomm		Idle (3)	MDI-M (6)	Idle (4)	VC2 (2)
2007	288	0647	k_mmtomh		MDI-M (6)	MDI-H (7)	VC2 (2)	VC3 (3)
2007	288	0900	k_mhtoidle		MDI-H (7)	Idle (3)	VC3 (3)	Idle (4)
2007	288	0905	d_ssr_record	y,y	Idle (3)	Record (4)	Idle (4)	Idle (4)
2007	288	0920	Termination	N/A	Record (4)	Record (4)	Idle (4)	Idle (4)

Table 8: Sample Activity Plan

The OE is allowed to add any extra activities to the pass operations. Some of these operations may be temporary such as expected limit or cfgmon violations.

3.3.1.1 Special Activities

3.3.1.1.1 Temporary Limit Changes

A temporary limit change may be issued for a pass. The Pass Generator will ask if there are any temporary limits. The OE must enter the original set values, then the new values and the time it becomes active. The original values of the limits are necessary if the limits will revert prior to the end of the pass.

3.3.1.1.2 Special Instrument Requests and SVM activities

During the OE review step in the pass procedure generation process. The OE can add any special activites requested by the instrument teams or additional SVM activities.

3.3.1.1.3 Mid-Pass Activation

The FOT may activate automation mid-pass by manually editing the Pass Execution Procedure, removing the activities that have already occured, and running it directly in TSTOL. The FOT may also start the Pass Execution Procedure at a specific line number when the next activities occur.

3.3.1.2 Pass Execution Procedure Description

The Pass Execution Procedure runs the pass. It has command link checks prior to each procedure and SSR/DHSS/MDI state checks after procedures.

The current set of TSTOL procedures most commonly used is being modified to replace unconditional WAIT statements with telemetry checks and removal of questions. WAIT statements with telemetry checks have a five-minute timeout added. This is done in cases where the spacecraft is not in the correct state:

```
WAIT UNTIL (MKSQRG3 = 0) TIMEOUT $TIMEOUT
IF (MKSQRG3 <> 0) THEN
        WRITE "NOTIFY OE: failed conditional wait for MKSQRG3"
ENDIF
```

If conditions are not met, then the OE is notified. The notification is meant to be as descriptive as possible in the shortest wording. The procedure name, mnemonic, and any error message are included.

A Perl script will verify that the Pass Execution Procedure is on each string by comparing location, version, and file size of the temporary procedure on each string against the local copy. This is done during distribution of the Pass Execution Procedure.

A command link and telemetry check is done before each procedure is run within the Pass Execution Procedure. It will verify lock on the spacecraft receiver (mnemonics RSLOK1 or RSLOK2) and telemetry quality (MF_QUALITY).

```
WHILE (1) DO
WAIT UNTIL (P@MF_QUALITY = "good" AND
           (P@RSLOK1 = "LOCK" OR P@RSLOK2 = "LOCK")) TIMEOUT 60
    BREAK IF (P@MF_QUALITY = "good" AND (P@RSLOK1 = "LOCK" OR
              P@RSLOK2 = "LOCK"))
    IF (P@MF_QUALITY <> "good") THEN
        WRITE "NOTIFY OE: Loss of telemetry"
    ELSEIF (P@RSLOK1 = "UNLOCK" AND P@RSLOK2 = "UNLOCK") THEN
        WRITE "NOTIFY OE: Loss of spacecraft reciever lock"
    ENDIF
    WAIT 300
ENDDO
```

3.3.2 External Interfaces

3.3.2.1 Input

- FTP access is required to either sh4cm2 or sh5cm2 from any string. The DSN schedule will be transferred via ftp from either CMS to sh6fe1 on the local TPOCC network.
- The OE will verify and add additional activities to the Activity Plan.

3.3.2.2 Output

- An Activity Plan is initially generated from the DSN schedule and output for OE review.
- A Pass Execution Procedure, *o_yyddd_auto_vxx.prc*, is created from the Activity Plan. This Pass Execution Procedure is sent to all strings via FTP on the local TPOCC network.

- The Pass Execution Procedure is stored in */home/soho/ops/input/procs/Automation*.
- The generation, verification, and any associated actions of the Activity Plan are logged in the file *POLARIS_pass_generator_JJJYYYY_HHMMSS.log*.

3.3.3 Data flow

In Figure 10, the pass generation process is shown. The Pass Generator will accept a DSN schedule via FTP. The user will have the system create the Activity Plan or the Pass Execution Procedure.

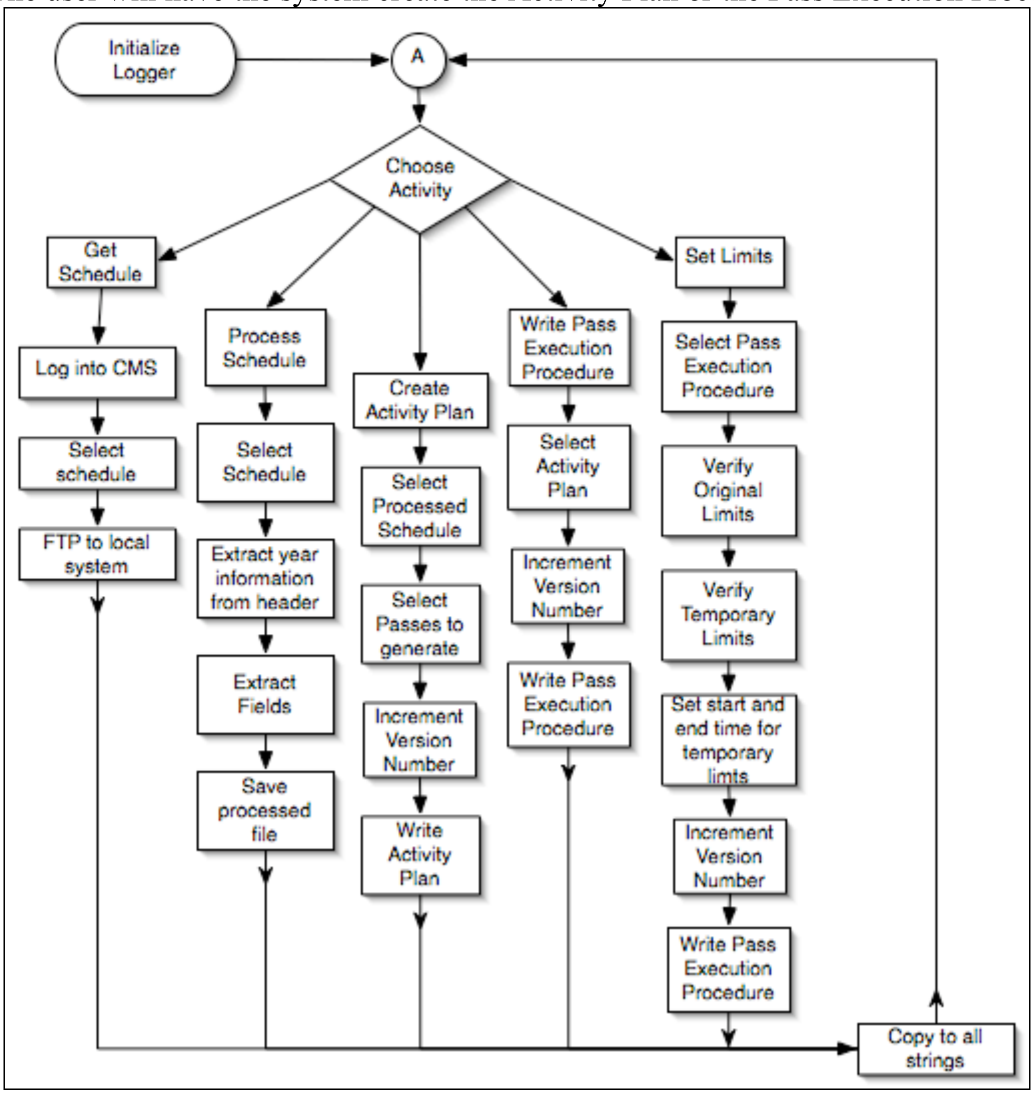

Figure 10: Pass Generator Process

3.3.4 Fault Tolerance

If the Pass Generator fails, an OE can manually create the Pass Execution Procedure. In this case, verification of all activities should be checked by two OEs. A system failure during a pass can be tolerated, as it is only used prior to any passes.

3.3.5 Requirements Met

7, 9, 10, 11, 12, 13, 14, 15, 49

3.4 Event Monitor, Notification, and Response Systems

3.4.1 Overview

The SOHO automation will use three bundled COTS products from Attention!® software to perform the event monitoring, notification, and response tasks. The Attention!® Alarm Manager (AM) provides a central point of alarm configuration, filtering, and reporting. With Attention!® AM, alarms and events are fed into a central database. This centralization provides authorized users with the ability to manage the alarm conditions that affect day-to-day operations. The Attention!® Notification System (NS) delivers real-time alerting of significant events via remote notification technologies such as pagers, cell phones, electronic mail, and other custom notification techniques. Attention!® NS' fault-tolerant design and comprehensive escalation strategy guarantee the appropriate individuals will be notified when the unexpected occurs. Attention!® Contact Center (CC) provides a reliable means to disseminate critical information to individuals and receive responses to alerts.

The POLARIS software will access the TSTOL event report. As new events are added, the POLARIS software will pass these to the Attention!® software using a simple Perl script. The Attention!® software will parse through all events to determine if any alerts are required. The criteria for determining these alerts are contained in a configurable database.

3.4.1.1 Limit Checking

TPOCC limit violation event messages contain the following information:

`DDD-HH/MM/SS.S C` `EVT# mnemonic=value` **`exceeded yellow/red high/low limit`** `(limit set #, limval #)`

where:

DDD-HH/MM/SS.S represents the spacecraft time for the limit violation
C is a critical event flag
EVT# is the TPOCC event number (1601-1607, 1666-1667 are used for limit violation messages)
mnemonic is the telemetry mnemonic violating limits
value is the value of the telemetry mnemonic at the time of the violation.

The Attention!® software will be configured to respond appropriately to limit messages. In general, yellow limit violations will just trigger an email to the OEs and/or instrument team/SOC. Red limit violations will trigger an immediate notification of the on-call OE. The Attention!® software will be configured to handle special cases of limit violations.

Temporary limit changes will be handled in the Pass Generator process. For example, for limits to be ignored, LIMIT OFF directives will be added to the Pass Execution Procedure. For limit sheets that expire during a pass, LIMIT CHG directives will be added to the Pass Execution Procedure at the appropriate time.

3.4.1.2 Cfgmon Checking

TPOCC configuration monitor violation event messages contain the following information:

DDD-HH/MM/SS.S EVT# mnemonic **in config** *cfgmon name* **failed check - value:** tm value **expr:** cfgmon value

where:

DDD-HH/MM/SS.S represents the spacecraft time for the cfgmon violation
EVT# is the TPOCC event number (1508 is used for cfgmon violations)
mnemonic is the telemetry mnemonic violating limits
cfgmon name is the name of the configuration monitor
tm value is the value of the telemetry mnemonic at the time of the violation
cfgmon value is the expected value of the telemetry mnemonic.

The Attention!® software will be configured to respond appropriately to configuration monitor violation messages. For most violations the on-call OE will notified immediately. The Attention!® software will be configured to handle special cases of configuration monitor violations.

3.4.1.3 Command Link Checks

The Pass Execution Procedure will perform checks prior to running procedures to ensure the command link is available. When the checks fail, the Pass Generator will use a TPOCC WRITE statement to create an event to indicate a command link problem. The exact text of this message is TBD. The Attention!® software will be configured to respond to these text messages.

3.4.1.4 Telemetry Status Checks

The Pass Execution Procedure will perform checks prior to running procedures to ensure telemetry is online. When the checks fail, the Pass Execution Procedure will contains a TPOCC WRITE statement to create an event to indicate a telemetry problem. The exact text of message is TBD. The Attention!® software will be configured to respond to these text messages. Additionally, the Attention!® software will monitor events for the output from the LOCKMON watch.

If the Attention!® software detects that the telemetry has switched to low rate, immediate notification of the on-call and lead OE will be made.

3.4.1.5 TSTOL Execution Checks

TSTOL execution problems are manifested in several ways. Event messages containing the following text are indications of a TSTOL execution problem.

- INVALID COMMAND STATE for directive
- Error – stopped at line #: "procedure text"
- No data is available for system variable.

The Attention!® software will be configured to make the appropriate notifications and/or take the appropriate action for these types of problems. The Attention!® software has the capability to initiate Perl scripts to allow an input to the TPOCC interface to remedy these problems without user input.

3.4.1.6 Critical Command/Load Checks

When a critical command has been entered, the following event message occurs:

```
DDD-HH:MM:SS.S  C 6312 /CMD:  mnemonic (command description) CRITICAL, enter /ALLOW or /CANCEL
```

where:

DDD-HH:MM:SS.S represents the ground time
C is a critical event flag
mnemonic is the command mnemonic
command description is text from the database providing a brief functional description.

When a critical load has been entered, NO event message occurs. Therefore, the TPOCC interface will be required to monitor for a response from the TSTOL server indicating a critical load has been entered and /ALLOW or /CANCEL must be entered. When a critical load has been entered, the mnemonic LOAD_NAME#TELECOMMAND is updated with the load name, but the mnemonic CMD_STATE remains "AVAILABLE".

Once /ALLOW has been entered for a critical load, the following event message occurs:

```
DDD-HH:MM:SS.S    6359 /LOAD:  load name in buffer, please enter /SEND or /CLEAR
```

where:

DDD-HH:MM:SS.S represents the ground time
load name is the name of the load.

Whenever either of the two event messages occurs, the Attention!® software will initiate a Perl script to execute the /SEND directive.

3.4.1.7 Software Anomaly Checks (COBS/ACU)

When a COBS software anomaly occurs, two event messages are generated:

9124 software anomaly received:
9125 software anomaly, ID: *XX* , data: *XX, XX, XX, XX, XX.*

If more than one software anomaly occurred during one format, the event message would ONLY show information for the last anomaly. A delta limit violation event message for the mnemonic KKANOCNT will also occur. The Attention!® software will be configured to initiate a Perl script to determine the number of software anomalies and the anomaly ID for each. Appropriate notifications will be made based on the anomaly ID.

When an ACU software anomaly occurs, this event message is generated:

9224 NEW ACU ANOMALY received, see reports/acuanom.log.

A delta limit violation event message will also occur. Appropriate notifications will be made based on the anomaly ID.

When an ACU TC error occurs, an ACU software anomaly may or may not occur. However, a delta limit violation will be generated as an indication of an ACU TC error. The Attention!® software will implement the appropriate response for these errors.

3.4.2 External Interfaces

POLARIS communicates with an Attention!®-provided "agent" which runs on the same computer as the Attention!® software. They also provide a command-line client called *attn*, which runs on the TPOCC system, to interface with the agent. The program *attn* communicates with the Attention server agent via a network socket.

Attention!® AM puts all events in a SQL database. AM provides a web interface to browse these events. In addition, any ODBC-compliant software can access the database. For example, the FOT could create reports using Microsoft Access or Perl scripts which use an ODBC module. The Attention!® software logs its activity to either the database or the NS log.

3.4.3 Data flow

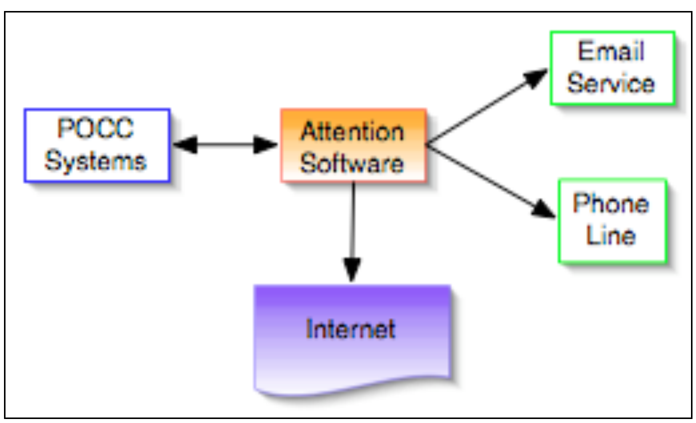

Figure 11: POLARIS/Attention Interaction

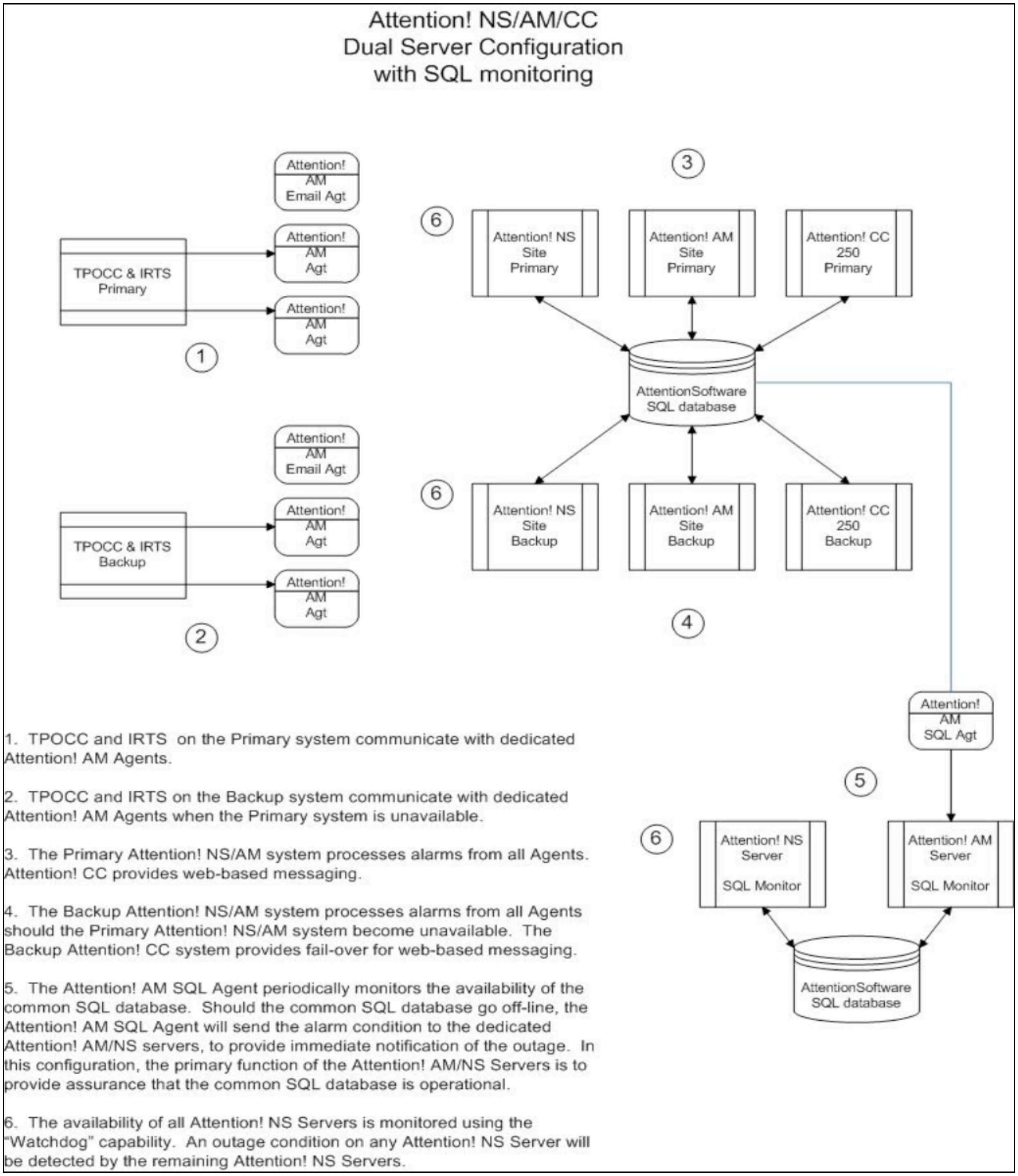

Figure 12: Attention Architecture/Data Flow

3.4.4 Fault Tolerance

Failure of the Attention!® software means that no event monitoring or notifications take place. POLARIS may function without Attention!® running, but any situation which requires user intervention will not be resolved quickly. Refer to figure 12 for Attention!® internal data flow and internal monitoring description.

3.4.5 Requirements Met

21, 22, 23, 24, 25, 26, 29, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44

Appendix A: Requirements Matrix

The following table shows the requirements for the automation software, and the section and/or document, which are designed to meet that requirement.

Requirement	Section
1. The software shall be able to initialize the operational ground software.	2.2
2. The software shall only initialize ground system software if the system is down.	2.2
3. The software shall be able to restart the operational ground system software.	2.2
4. The software shall allow the user to customize the initialization tasks. Specify procedures to run at startup Specify scripts to run at startup Change pass parameters	2.2
5. The software shall shall allow external user interaction while automation is running.	3.1.1
6. The software shall allow user to override automation.	3.2.1
7. The pass generator shall be able to verify pass activities are correct Daily activities Weekly activities Other activities	3.3.1
8. The pass generator shall allow the user to activate automation mid-pass.	3.3.1.1.3
9. The pass generator shall be able to read the DSN schedule from a text file.	3.3.1
10. The pass generator shall be able to generate a pass procedure based on the DSN schedule and user input. User input for temporary limit changes User input for special instrument requests User input for special SVM activities	3.3.1.1
11. The pass generator shall allow scripts to be executed from the pass procedure.	3.3.1
12. The pass generator shall include checks to the pass procedure to verify the completion of all pass activities.	3.3.1
13. The pass procedure shall contain checks to verify command link to the spacecraft.	3.3.1.2
14. The pass procedure shall verify spacecraft ranging configuration. Receiver locked and coherency enabled	3.3.1
15. The pass generator shall log all messages associated with pass plan generation and DSN schedule processing.	3.3.1
16. The software shall send directives to the operational ground system software.	3.1
17. The software shall receive responses from the operational ground system software.	3.1
18. The software shall be able to start procedures at specific times (UTC).	3.1
19. The software shall allow user to switch from manual to automated mode and vice versa.	3.2
20. The software shall be able to determine when operations are running in automated or manual mode.	3.2
21. The software shall monitor the status of the ground system.	3.4
22. The software shall recognize all limit violations, configuration monitor violations, COBS software anomalies, and ACU software anomalies.	3.4.1

Generic notification rules outlined in Chapter 7 of the FOP Immediate notifications for loss of TM, loss of command link, low rate TM, red limit violations, subset of COBS/ACU software anomalies	
23. The software shall recognize telemetry outages and losses of command ability. Monitor events for *lockmon* event messages Monitor receiver lock status for command ability	3.4.1
24. The software shall recognize operational ground system procedure errors.	3.4.1
25. The software shall recognize degraded data.	3.4.1
26. The software shall create log files.	3.1, 3.4.2
27. The software shall log all messages associated with automation initialization.	2.2
28. The software shall log all messages associated with automation termination.	2.2
29. The software shall log all notifications and to whom.	3.4
30. The software shall log any user-generated operations while automation software is running.	2.2
31. The software shall log all error and anomaly identifications.	3.4
32. The software shall log all corrective measures taken and their cause.	3.4
33. The software shall take corrective measures for operational ground system procedure errors.	3.4
34. The software shall be able to execute a corrective measure based on the current spacecraft configuration.	3.4.1.1, 3.4.1.2
35. The software shall notify personnel according to a configuration file.	3.4
36. The user shall be able to configure which personnel are notified according to a particular event or time.	3.4
37. The software shall make notifications when operational ground system procedure problems occur.	3.4
38. The software shall make notifications when system initialization or termination fails.	3.4
39. The software shall accept responses to pages.	3.4
40. If a response is not received within a user-defined period, the software shall escalate notification.	3.4
41. The software notification messages shall contain error identification information.	3.4
42. The software shall deliver notification to e-mail accounts.	3.4
43. The software shall deliver notifications during network outages.	3.4
44. The software shall create a summary for the periods that automation is active.	3.4, TPOCC event delogs
45. The system shall maintain up to 60 days of continuous automation logs.	FOP
46. The software shall require user authentication to sockets, which allow external hosts to connect.	2.2
47. The system shall permit user authentication for localhost connections.	2.2
48. The system shall comply with GSFC IT security regulations.	TSM
49. The software shall be able to terminate pass operations.	3.3
50. The software shall be able to terminate itself.	2.2
51. The software shall allow the user to customize the termination tasks. Specify procedures to run at termination Specify scripts to run at termination	2.2

Change parameters	

Appendix B: Abbreviations and Acronyms

ACU	Attitude Control Unit
AOCS	Attitude and Orbit Control Subsystem
AM	(Attention!®) Alarm Manager
ANTS	ANTenna Subsystem
ASW	Application SoftWare
BARM	Block Acceptance and Reporting Mechanism
BOT	Beginning of Track
CC	(Attention!®) Contact Center
CDMU	Central Data Management Unit
CDS	Coronal Diagnostics Spectrometer
CELIAS	Charge, ELement and Isotope Analysis System
CEPAC	COSTEP-ERNE Particle Analyser Collaboration
CFGMON	Configuration Monitor File
COBS	Central On-Board Software
DSN	Deep Space Network
CRP	Coarse Roll Pointing
DHSS	Data Handling Subsystem - SOHO
EIT	Extreme ultraviolet Imaging Telescope
EOT	End Of Track
EPSS	Electrical Power SubSystem - SOHO
FOP	Flight Operations Plan
GOLF	Global Oscillation at Low Frequencies
HP	Hewlett Packard
LAN	Local Area Network
LASCO	Large Angle and Spectrometric COronograph
MDI	Michelson Doppler Imager
OE	Observatory Engineer
MDI-H	MDI-Helioseismology data
MDI-M	MDI-Magnetogram data
NASA	National Aeronautics and Space Administration
Nascom	NASA Communications Network
NRT	Near RealTime
NS	(Attention!®) Notification System
OBT	On-Board Time
ODBC	Open DataBase Connectivity
POCC	Payload Operations Control Center
POLARIS	Pass Operations Logging and Anomaly Reporting Interface System
PROS	PROpulsion Subsystem
PTCS	PLM Thermal Control Subsystem
PTI	POLARIS TPOCC Interface
RSL	Roll Steering Law
SGSS	Solar Generator Subsystem - SOHO

SHSS	Structure and Harness Subsystem – SOHO
SOHO	SOlar and Heliospheric Observatory
SQL	Structured Query Language
SSR	Solid State Recorder
STCS	SVM Thermal Control Subsystem
SUG	System User’s Guide
SUM	SOHO Users Manual
SUMER	Solar Ultraviolet Measurement of Emitted Radiation
SWAN	Solar Wind ANisotropies
TAI	Temps Atomique International
TLAN	TPOCC Local Area Network
TPOCC	Transportable Payload Operations Control Center
TR or TRC	Tape ReCorder
TR	Tape Recorder
TSTOL	TPOCC System Test and Operations Language
UTC	Universal Time - Coordinated
UVCS	UltraViolet Coronograph Spectrometer
VIRGO	Variability of solar IRradiance and Gravity Oscillations

Appendix C: Glossary

Activity Plan:	Based on the DSN schedule, the Pass Generator will create the Activity Plan. This is a skeleton schedule of pass activities that an OE will check. Upon verification, the OE will submit this file to the Pass Generator, which will create the Pass Execution procedure.
ACU Anomaly:	A message generated by the ACU that states errors with functions or equipment that it monitors.
COBS Software Anomaly:	A message generated by COBS that states errors with functions or equipment that it monitors.
Command Ability:	The process by which a command is transmitted the spacecraft and the ground is able to acknowledge its acceptance.
Configuration Monitor:	Certain spacecraft parameters, especially statuses, are monitored for changes. A configuration monitor will only report the violation and not the subsequent return to a normal value.
Corrective Measure:	An action taken by the automation software to fix a specific error.
Degraded Data:	When a partial data loss occurs in real-time. This is separate from a telemetry outage.
DSN Schedule:	A text file generated by JPL describing the stations and pass times for a given week.
Error Identification:	A message or statement detailing the symptom(s) of an error or anomaly.
Event Monitor	The Event Monitor interprets the event log for any kind of error or anomalous situation. If necessary, it will make notifications. In the early phase, it will be limited to checking for limit and cfgmon violations and ranging. In the later phases, it will be able to determine anomalies and take specific actions.
Keyhole:	A period occurs every three months when the HGA no longer points towards Earth. The FOT switches to the -Z LGA for downlink of all data.
Limit:	Certain spacecraft parameters, especially voltages, currents, and temperatures, are checked to be within tolerable levels. These are divided into red/yellow limits and delta limits. Any violation is reported, but when the parameters return to a normal value that is also reported.
MDI Transition Timing:	Transition between MDI modes is not allowed with -2 or +5 minutes of the generation of an ALT magnetogram.
Message Oriented Middleware:	A software system used to publish and queue messages to a set of subscribers.
Network Outage:	The loss of capability to send or receive email over the network.
Notification System	The Notification System issues messages containing violations and anomalous behavior found by the Event Monitor.
On-call OE:	There are several OEs for SOHO. Each week one is selected to direct actions.
Operational Ground Software:	This software is used to run pass operations. Examples are TPOCC, ITOS, and ASIST.

Pass Execution Procedure:	A sequence of actions taken on a spacecraft pass at specific times. This is based on the auto-generated Activity Plan.
Pass Generator	The Pass Generator creates the schedule activities for SOHO passes. It will also do a check that activities meet MDI and DHSS transition rules and timing requirements.
POLARIS TPOCC Interface:	The TPOCC interface issues the actual commands to TPOCC and check response messages from the TPOCC.
Response System	The Response System relays the acknowledgement of a message by the OE. This function will tell the Notification System that it was successful and to stop paging.
SVM activities	Commanding done by the FOT to maintain the health and safe operation of the spacecraft.
Telemetry Outage:	Loss of SOHO telemetry for at least one minute.
User Authentication:	The process by which a user proves their identity, typically using a username and password.

1. National Aeronautics and Space Administration. February 1994. Transportable Payload Operations Control Center (TPOCC) Detailed Design Specification for Release 10 (511-4DDS/0693). Control Center Systems Branch, Goddard Space Flight Center. Greenbelt, Maryland.
2. National Aeronautics and Space Administration. February 2004. SOHO Flight Operations Plan, DCN 003 (CSOC-GSFC-PLAN-001188: C0932). Goddard Space Flight Center. Greenbelt, Maryland.
3. National Aeronautics and Space Administration. September 1994. Detailed Mission Requirements (DMR) Document for the International Solar-Terrestrial Physics Program (ISTP) Solar and Heliospheric Observatory (SOHO), Issue 2 (501-215/SOHO). Goddard Space Flight Center. Greenbelt, Maryland.
4. Matra Marconi Space. November 1999. SOHO User's Manual, Issue 03. (SH-MAT-UM-13687). Matra Marconi Space. Toulouse, France.
5. National Aeronautics and Space Administration. June 1998. International Solar-Terrestrial Physics (ISTP) Solar and Heliospheric Observatory (SOHO) Payload Operations Control Center (POCC) System User's Guide, Revision 5 (511-4SUG/0693). Goddard Space Flight Center. Greenbelt, Maryland.
6. Walsh, Nancy 1999. Learning Perl/Tk. O'Reilly & Associates, Inc, Sebastopol, California.
7. Attention!® Notification System, Attention!® Control Center, and Attention!® Web Security Package. Attention Software, Inc. <http://www.attentionsoftware.com>.