DISPLAY OPERATION APPLICATION

Technical Report 95-10

Robert A. Brown, Jr.

Department of Computer Science and Engineering
Auburn University
Auburn University, Alabama 36849

October 20, 1995

CSE TR 95-10.

DISPLAY OPERATION APPLICATION

Robert A. Brown, Jr.

Department of Computer Science and Engineering

Auburn University

Auburn University, Alabama 36849-5347

October 20, 1995

Table of Contents

1. Introduction

During the history of the National Aeronautics and Space Administration (NASA), the ability to send and receive telemetry and command data to and from spacecraft has been an important factor in the successful execution of its mission. Over time, the complexity and volume of data transmitted and received has increased in direct proportion to the increased sophistication of NASA projects, experiments and the increasingly complex telemetry and computer hardware/software.

Because of increasing complexity, the current system utilized by NASA's Marshall Space Flight Center (MSFC) to perform these functions is quickly becoming insufficient in providing the necessary capabilities. The current system at MSFC's Huntsville Operations Support Center (HOSC) utilizes a character based user interface (UI) to DEC 8350s and MicroVAX's. One of the limitations of this system includes the lack of a multi-windowed environment that allows the user to perform and monitor multiple tasks on the same workstation. Also, data displays used to view telemetry data are limited graphically, and generally do not provide the functionality needed by the user. This report discusses a solution to the current system's data display limitations and provides information about the implementation of NASA's solution to the problem: *Display Operation.*

One solution to the current system's data display weaknesses is the Display Operation (DO) application. DO is a real-time X Window/Motif application, and is a component of a larger system called the Enhanced HOSC System (EHS). The basic role of the EHS is to provide mission support to EHS users. For the purposes of Display

Operation this can be viewed as providing the capability to display telemetry data and issue command data to the spacecraft. Figure 1-1 provides a graphical representation of these basic capabilities.

As shown in the figure, the spacecraft transmits telemetry data to a receiving station which in turn forwards this data to the EHS. The Data Distribution System within the EHS distributes the data to the many EHS user workstations, each of which has the capability to execute the DO application. Commands can be issued by the user to the spacecraft via the DO application. These commands are processed through the Command Processor and then sent to the spacecraft.

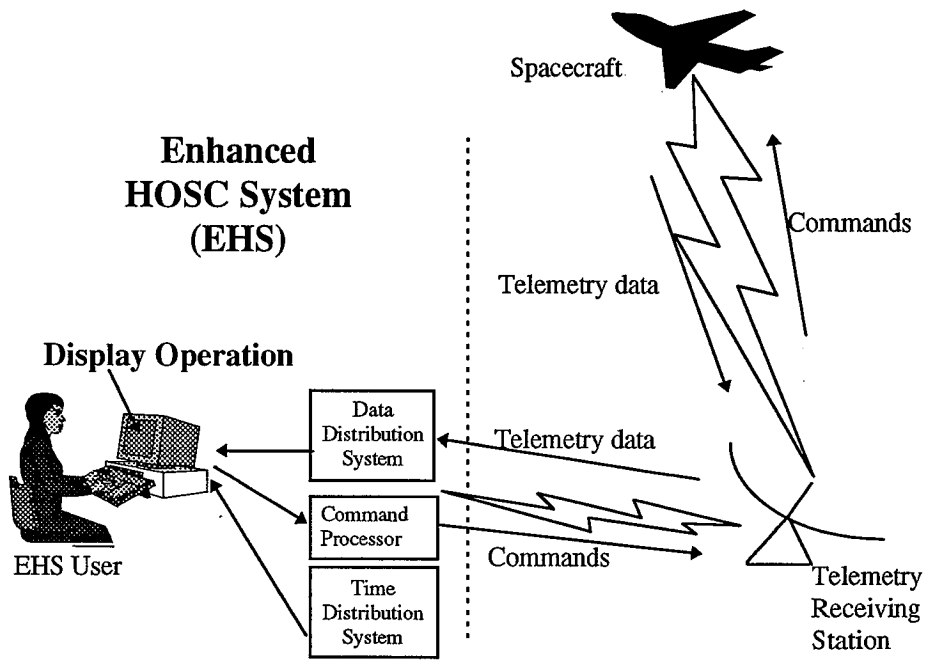

Figure 1-1 EHS Data and Command Processing

The DO application provides users with the means of viewing telemetry data and perform other functions (e.g., issue commands) by use of a data display that was

created/generated through the use of another EHS application called Display Generation (DG). The data can be viewed graphically or textually in any layout the user desires. The layout (position of graphical and textual objects in the display) is set up by the user in the DG application. When the data display is operated in the DO application it has the exact layout as defined in the DG application. The difference is that telemetry data is used to update the display from within the DO application. This allows the user to monitor real-time (or recorded) data values in the output method of choice.

It may be helpful to envision a hospital monitoring device that monitors a patient's vital signs. The device displays the data on a monitor and updates the data continuously. The DO application provides similar capabilities, but is used to view telemetry data on a *customized* display. It also provides the capability to send command data to spacecraft in support of various NASA projects and experiments. A detailed discussion of these and other capabilities of the DO application is presented later in the report.

As mentioned, the benefits of a more 'user friendly' window based UI has become evident in recent years with the popularity of window based applications in use on personal computers. The DO application will replace the current application used to view data. It will provide a graphical user interface (GUI) and enhanced capabilities to the user by utilizing the X Window System on newer, faster workstations (RISC).

The remainder of this chapter presents a more detailed overview of the EHS and the Display Operation application. These sections are intended to provide background information that will aid the reader in understanding the more detailed project description found in later sections.

1.1 Enhanced HOSC System (EHS) Overview

The Enhanced HOSC System (EHS) is a general purpose, distributed-CPU system for processing and display of real-time spacecraft telemetry and command data. It is planned to be used to support numerous projects located at the Marshall Space Flight Center in Huntsville, AL [7]. The HOSC mission is to provide real-time and near real-time telemetry processing, and command processing. The EHS provides this functionality for multiple projects simultaneously. These projects include the Space Station Freedom, Space Transportation System (Space Shuttle), Advanced X-Ray Astrophysics Facility (AXAF), and Space Lab.

Users of the EHS can analyze and display telemetry data and uplink commands through User-generated Data Elements (UDEs). UDEs are console tools created by utilizing various EHS applications known as generation tools. These UDEs are then operated by the user by using EHS applications known as operation tools. A display is an example of a UDE, and Display Operation is an example of an operation tool used to operate a display UDE. Figure 1-2 represents the relationship between the DO application and the Display Generation application and is used to show the general relationship between the EHS *generation applications* and *operation applications*.

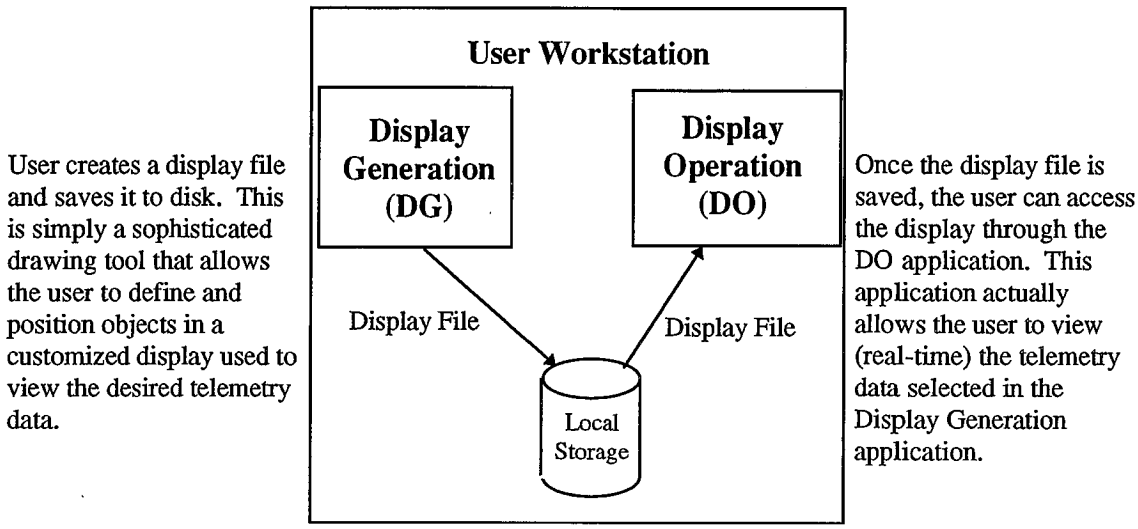

Figure 1-2 DO/DG Relationship

As discussed previously, numerous projects can be supported simultaneously by the EHS. Each of these projects is supported by a Local Area Network (LAN) which hosts user workstations and servers to perform the telemetry and command processing for a project's mission. Figure 1-3 is a top-level representation of the EHS architecture including a Project LAN, an Operations LAN, and a Mission Support Services LAN that support the mission.

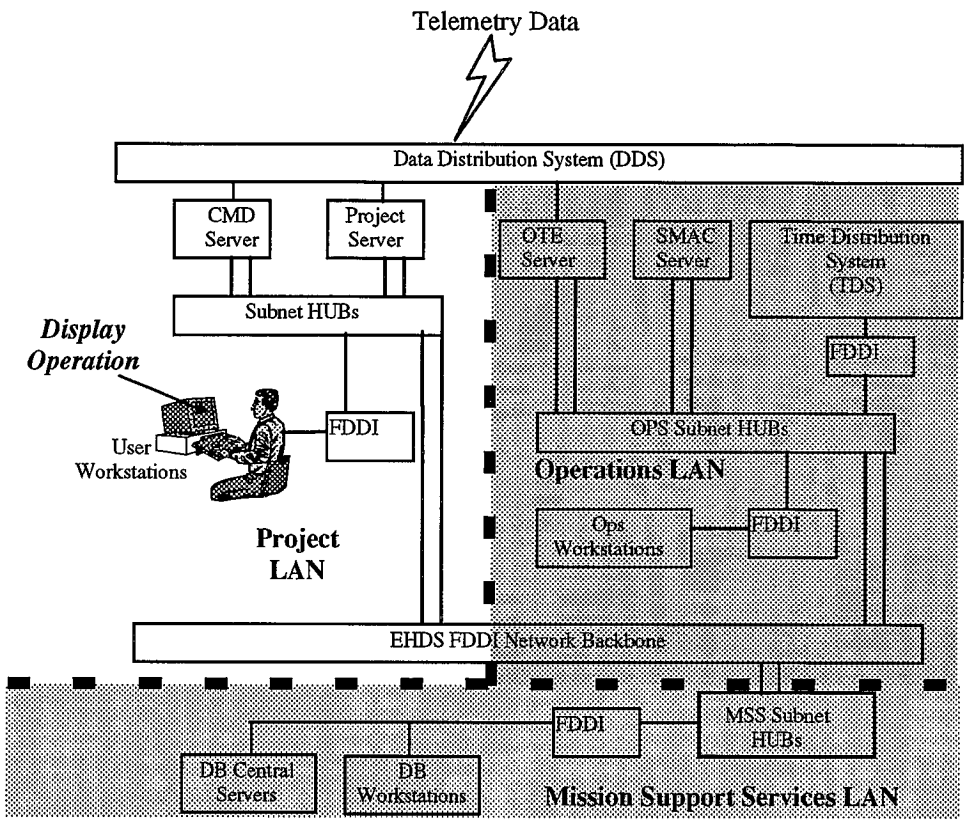

Figure 1-3 EHS Architecture for a Project

The *User Workstations* indicated in the *Project LAN* are the means by which users access the numerous EHS applications available to include the DO application. The workstations utilize the X Windows protocol and provide a multi-application window environment. Each user has certain privileges which are contained in a data structure known as the user profile which controls access to the different applications available on the workstation. The *Project Server* provides three major functions: perform front-end processing for telemetry, log telemetry, and run applications required by multiple users. The *Command (CMD) Server* provides a central point where real-time commands are built and uplinked [4].

The *Operational Test Equipment (OTE) Server* indicated in the *Operations LAN* is used to perform functions similar to the Project Server. The *System Monitor and Control (SMAC) Server* is responsible for maintaining the system configuration files for all projects which are available to a project LAN. The *Operations (Ops) Workstations* are basically identical to the User Workstations and are used by operations personnel.

The *DB Central servers* indicated in the *Mission Support Services (MSS) LAN* maintain databases for all projects. The workstations in the MSS Net are used by database personnel to update and maintain the various EHS mission and system databases.

As depicted in Figure 1-3, telemetry data is distributed to the various LANs by the *Data Distribution System (DDS).* Likewise, internal EHS time is distributed by the *Time Distribution System (TDS).*

1.1.1 Telemetry Processing

Telemetry is defined as the transmission of data from remote sources (e.g., spacecraft) to a receiving station for recording, analysis, or display. For receiving systems to be able to process the telemetry, the structure of the data must be defined. Each telemetry stream or data packet received by the EHS is defined in the telemetry database. Each of the parameters in a data stream is assigned a measurement/stimulus identifier (MSID). MSIDs are used to provide a unique identification key for each of the parameters that are to be processed for a given project and mission.

Again, it may be convenient to use the hospital monitoring device as an example. Each of the items of information about the patient (e.g., blood pressure, pulse, temperature, etc.) would have an MSID assigned to it. The display system would

recognize the MSIDs and display the appropriate data. Similarly, when the user generates the display in the DG application, he selects from a database the MSIDs he desires to monitor. The DO application uses these MSIDs to retrieve the appropriate data and display it to the user.

The database mentioned above, defines for each MSID the location in the telemetry stream, calibration data used to calibrate the parameter to engineering units, and data to determine whether the parameter's value is within limits or in its expected state. Limit ranges and expected states are used to notify the user when parameters are out of their nominal range or state [4]. For example, if the following ranges are defined in the database for an MSID

0 - Warning Low

6 - Caution Low

21 - Caution High

31 - Warning High

any value less than or equal to 0 would result in a 'warning low' condition indicating that the value is out of the limit ranges specified in the database. Any value in the range 7-20 is normal. A value less than or equal to 6 but greater than 0 would result in a 'Caution Low' condition, and so on. Each MSID has a set of limits defined for it in the database. These limit ranges and expected states are generally used to determine when there may be a malfunction or undesirable event occurring on the spacecraft.

Once again, during operation of a display, telemetry parameters are requested by the DO application through the use of MSIDs. The telemetry packets containing the

requested MSIDs are received from the network, and can then be calibrated and/or compared against limits or expected states and then displayed to the user.

1.1.2 Command Processing

A command (as used in this report) is defined as a collection of data words that can be uplinked from the EHS to the spacecraft. These commands can specify data or an action to be executed onboard the receiving spacecraft.

Command processing is similar to telemetry processing in that it is database driven. The project command database defines for each command a mnemonic, data fields, and whether the command is critical (could cause damage to payload or spacecraft and impair the mission) or hazardous (could pose a threat to human life or the entire mission). The command mnemonic uniquely identifies each command and is used to issue commands. Commands are issued through the Display Operation application via the use of input objects. A command mnemonic is assigned to the input object (e.g., a pushbutton) when the display is generated.

During operation of the display, when the input object is invoked, the DO application issues the command using the assigned command mnemonic. The command mnemonic is converted to the actual command data by the command processor (see Figure 1-3) and then forwarded for transmission to the spacecraft.

1.2 Display Operation Application Overview

As stated previously, the DO application is a real-time X Window/Motif application. The DO application is also categorized as an event-driven program utilizing

the client-server model. An event-driven program creates an initial top level window or set of windows and then enters a continuous event loop. As events occur, either through system signals or user inputs, the program performs actions based on the event type. An example of an event is the selection of a menu item from a pulldown menu on a window based application. This event triggers a call to a callback function that performs specific actions relative to the item on the menu that was selected (e.g., opens and reads a file). Events are queued in the X event queue and when one event is handled the next is retrieved from the queue and dispatched to the appropriate event handler or callback routine. This process continues until the user initiates an event that triggers a callback to exit/terminate the application or until a terminate/kill signal is received from the operating system. An advantage of event-driven programming is the versatility it provides the user in performing tasks. In general, tasks can be performed in any sequence desired by the user, thus alleviating the step by step approach of traditional sequential processes.

The Display Generation (DG) application provides the user with the capability to create a customized display that can be executed in the Display Operation (DO) application. The user creates a display that, when invoked in DO, allows the user to view the values for selected telemetry parameters through objects known as output objects. The user can also issue commands to the spacecraft using from the display through the use of objects known as input objects. Numerical values can be plotted on various graph types (line, bar, scatter, etc.) or on various scale types (radial meter, output slider, etc.). Discrete values can be represented by use of toggle buttons and graphic objects.

Input objects (pushbutton, input text field, etc.) are the objects the user utilizes to issue commands to the spacecraft. Input objects can also perform functions other than issuing commands. These functions (to include uplinking commands) are performed by using something known in the EHS as a scratchpad line directive. Each input object has a scratchpad line directive associated with it. A scratchpad line directive is an English-like, interpreted instruction that allows the user to provide parameters used in the execution of the directive. These input objects and their associated scratchpad line directives provide the user with the capability to start and stop *scripts*, *displays*, or *computations*; update or generate *pseudo telemetry*, and of course, issue *commands* [4]. Section 2 provides a definition for each of these items.

A list of all display objects available to the user to create displays follows. Any number and combination of these objects may be used when generating the display.

Basic Objects:

- Text
- Circles
- Ellipses
- Quarter Arcs
- Arcs
- Pixmaps
- Lines
- Rectangles
- Irregular Polygons (closed and open)

- Regular Polygons (consisting of 3 to 8 sides)

Output Objects:

- MSID Text Fields
- Time Plots
- Scatter Plots
- XY Plots
- Bar Charts
- Pie Charts
- Radial Meters
- Output Sliders
- Output Toggle buttons
- Output Radio Boxes

Input Objects:

- Push buttons
- Input Scratchpad Line Fields
- Input Data Fields
- Input Slider Fields

Once again, the Display Operation (DO) application provides the user with the capability of viewing specified command, telemetry and processed data by operating a display previously created with the Display Generation (DG) application (see Figure 1-2). These displays are initiated upon startup of DO or can be opened during the execution of DO by selecting 'Open' from the 'File' menu item on the DO main window menu bar.

Upon selection of this menu item, a file selection dialog box appears with which the user selects one of the displays residing locally on the workstation. The DO application allows the user to view telemetry values and graphical displays of data via use of output objects (bar charts, plots, text fields, etc.). The user is also allowed to issue commands; start and stop displays, computations, and scripts; and update pseudo telemetry (user generated) through use of Scratchpad Line (SPL) directives associated with input objects in the display (pushbuttons, input text fields, etc.). An example display is shown in Figure 1-4.

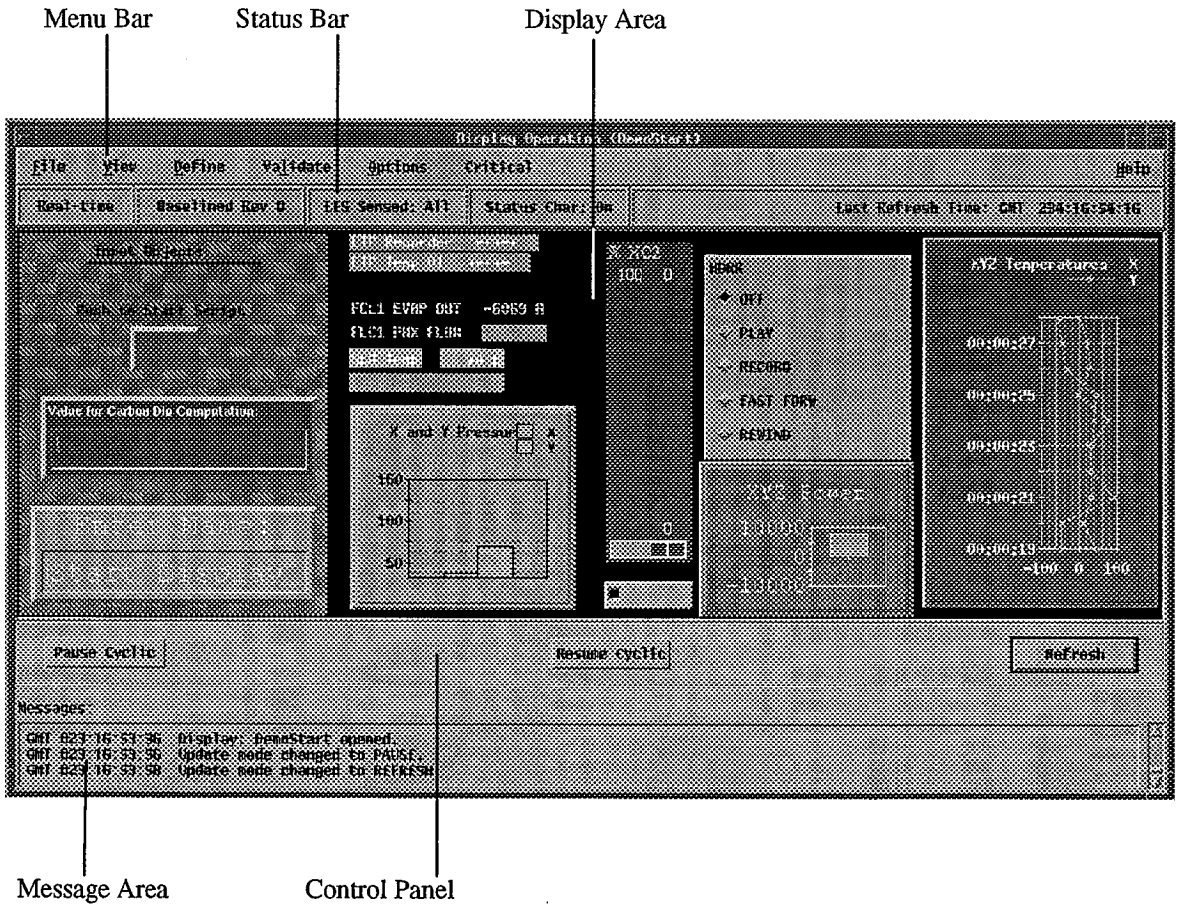

Figure 1-4 Sample Display in Display Operation

2. Terminologies

This section presents some definitions and terminologies that will be used in further discussion about the Display Operation application and other EHS applications/services.

COMMAND - a collection of data words that can be uplinked from a ground system to a spacecraft.

COMMON CONFIGURATION INFORMATION - data structure containing information used to configure the application to include: project name, mission ID, database versions, etc.

COMPUTATION - a user defined 'C' or 'FORTRAN' routine used to transform input MSIDs into output pseudo MSIDs.

DARL - Database Access Routine Library

DATA MODE - there are three modes in which data will be received and processed, including: real-time, playback, and dump.

DATA STATUS - a status returned by the Telemetry and Network Services (TNS) routine that decommutates data. This status indicates whether the data is good (acquisition of signal), bad (corrupt data), or nonexistent (loss of signal).

DISPLAY - a user generated tool used to view telemetry data and initiate command updates and command uplinks. Displays are generated in the Display Generation (DG) application and operated in the Display Operation (DO) application.

LES - limit/expected state sensing. MSIDs can have limit ranges (e.g., 0-20 warning, 21-25 caution, 26-50 OK, 51-55 caution, 56+ warning) or expected states (e.g., 1 - OK, 0 - NOT OK) assigned to them. LES is the process of determining whether an MSID value is within limits or out of state.

MSID - measurement/stimulus identifier. A unique key defined for each parameter in the telemetry streams to be processed for a specific project and mission.

PSEUDO TELEMETRY - values that are generated by performing calculations on telemetry parameters. These values are distributed either on the project network (external pseudo MSIDs) or locally on the user's workstation (internal pseudo MSIDs).

SCRIPT - a set of high-level, English-like, interpreted instructions called directives. Scripting directives support the monitoring of telemetry, updating and uplinking commands, and executing computations and displays.

TELEMETRY - refers to the transmitting of data collected from a source in space to a ground system for analysis.

UDE - user defined data element. A display is an example of a UDE.

USER PROFILE - data structure containing information about the user to include: user ID, user privileges, etc.

VALIDATION - the process of determining whether certain information items associated with a display exist and are available for use by the user.

3. Display Operation System Engineering

Specific development efforts included designing, coding, testing, and documenting the Display Operations application. Requirements for the DO application were determined and published by NASA at Marshall Space Flight Center and are later in this report.

The DO application was developed and operates on a Silicon Graphics Indy workstation running the IRIX 5.2 operating system. DO is an X Windows application that is Motif and POSIX compliant. The graphical user interface (GUI) builder/designer tool, TeleUSE v3.0, was utilized in the design of the UI. The application code was written in both 'C' and 'D' (a TeleUSE dialog language which is interpreted and converted to 'C' by the TeleUSE 'D' compiler). Motif widgets and K.L. Group's XRT/graph widgets are used for the objects in the displays. The primitive graphics (circles, lines, polygons, etc.) are drawn and managed by utilizing Xt and Xlib functions.

The design and development of the Display Operation application component of the EHS followed the classic software development waterfall and structured software development paradigms. The EHS development process began in early 1991 and continues as of the date of this document. Five (5) lifecycle phases have been identified which together comprise the EHS system lifecycle. These five phases include:

1. The Definition Phase
2. The Analysis Phase
3. The Design Phase
4. The Program/Code and Unit/Integration Test Phase
5. The System Test and Validation Phase

The Definition phase was used to develop system requirements and to divide these requirements among subsystems. The EHS Definition phase concluded with a System Requirements Review (SRR) in January 1992.

The Analysis phase was used to develop software requirements and allocate the requirements to Computer Software Configuration Items (CSCIs). This phase concluded with a Preliminary Design Review (PDR) in August 1992.

The Analysis phase was followed by the Design phase. Activities in this phase include developing detailed software design from the requirements developed in the Analysis phase. The Design phase for Display operation concluded with a Critical Design Review (CDR) in May 1995.

Following the Design phase, the lifecycle entered the Program/Code and Unit/Integration Test phase. This phase included transforming the detailed design into source code modules. Unit testing was conducted on these modules in preparation for integration with other CSCIs. Integration testing will conclude this phase and is scheduled for November 1995.

The final phase of the EHS development lifecycle is the System Test and Validation phase. this phase will include formal system testing followed by system validation.

3.1 Requirements

As mentioned previously, the DO application provides the user with the capability to operate the displays generated in the DG application. By operating a display, the user can view telemetry values and graphical representations of data. The input objects defined

on a display are used to activate the scratchpad line directives associated with the object. The DO application allows the user to select the data mode, the update mode, the update rate, and to turn LES on and off for the objects in the display.

All of these capabilities are specified in formal requirements developed by personnel at NASA's Marshall Space Flight Center. The following table summarizes these requirements. These requirements were extracted from two Marshall Space Flight Center Documents: 1) MSFC-RQMT-1440B, Generic Requirements for the Enhanced HOSC System [7]; 2) HOSC-SRS-027 R1, Software Requirements Specification for the User Telemetry Applications of the EHS [6]. The list is not all inclusive but provides a good overview of the general requirements of the application.

REQUIREMENT	DESCRIPTION
View data	Each display shall be able to have up to 100 cyclic measurements with a one second update rate.
Transmit commands	The user shall have the capability to initiate transmission of a command or chain of commands.
Execute script	The user shall have the capability to execute a script.
Execute a user Computation	The user shall have the capability to execute a user Computation.
Select data mode	The user shall have the capability to select the data mode (i.e., real-time, playback, dump).
Update computation constant	The user shall have the capability to update a computation constant.
View data status	Data status information shall be provided for each MSID (loss of signal, stale or old, etc.).
Invoke Display Operation	The user shall have the capability to start another Display Operation application from the current display.
Pause/restart cyclic	The user shall have the capability to pause and restart

updating of display	the cyclic updating of parameters on the display.
Help	The user shall be able to view help information on the application by selecting a menu item from the main menu bar.
Resize a display window	The user shall be able to resize the display window.
Toggle LES on/off	The user shall be able to turn LES on and off for each output object in the display.
Print a display	The user shall be able to print a screen dump of the display during operation.
Data status	The DO application shall present data status information.
Messages	The DO application shall present messages to the user in a message area on the main application window.
Open	The user shall be able to open a display.
Exit	The user shall be able to exit the display at any time during operation.

Table 3-1 Display Operation Requirements

3.2 Detailed Design

The approach in designing the Display Operation application was to divide the application into six major functional areas which are depicted in Figure 3-2. This diagram was used as the foundation for the detailed design of the application. The external block labeled 'TNS' represents the Telemetry and Network Services Library used to retrieve specified data from the telemetry stream. The external block labeled 'Validate Display' represents the process that is initiated to perform the validation of a display. The disk symbol labeled 'Files/Pixmaps' indicates that Display Operation retrieves these files from

local storage. The following paragraphs describe the processing performed by each of the six internal functional blocks presented in Figure 3-2.

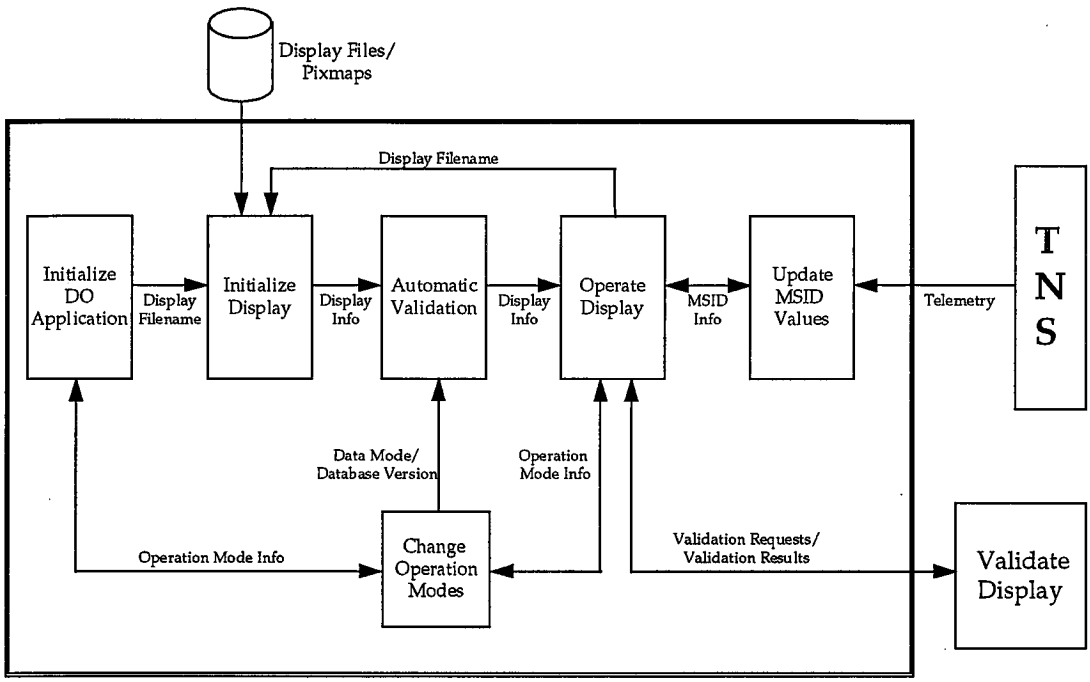

Figure 3-2 Display Operation Data Flow

3.2.1 Initialize DO Application

The first functional area is the 'Initialize DO Application' block. This block represents the processing that occurs during the initialization of the application. Some of the processing that occurs within this functional block includes:

- Retrieves the user profile and common configuration information
- Registers the application with the Workstation Command Processor
- Initializes the receipt of time data from Telemetry and Network Services
- Initializes display settings to a default value:

- ◊ Time reference: GMT
- ◊ Data mode: Real-time
- ◊ Database Version: Baselined databases
- ◊ Update mode: Cyclic
- ◊ Update rate: 1 second

All of these actions occur when the application first becomes active and prior to a display file being opened.

3.2.2 Initialize Display

The second functional area is the 'Initialize Display' block, and represents the processing that occurs when a file is opened by the application. The line leading from the disk symbol labeled 'Display Files/Pixmaps' represents the process of the application retrieving the display file from local storage. The line leading from the 'Operate Display' functional block indicates that the user can open a display file any time during operation of display. The processing that occurs during the initialization of a display includes:

- Retrieves display file from local storage
- Opens and reads the display file
- Allocates memory for display object/widgets and associated data structures
- Populates data structures with information from display file
- Issues directives specified to be issued in the operation startup list

3.2.3 Automatic Validation

The third functional area is the 'Automatic Validation' block. This block represents the validation process that occurs each time a display file is opened for operation. This validation also occurs when the user changes the database version number during operation (represented by the line drawn from the 'Change Operation Modes' block). This occurs because each validation is performed against the selected database and when the database changes, the validation is no longer accurate. The processing that occurs during the automatic validation of a display file includes:

- Checks validation information within display file to determine if the display has been validated against the selected database version (user can validate a display file within the Display Generation application and this validation information is saved with the file)
- Utilizes information from the telemetry database to perform the validation
- Checks all MSIDs referenced in the display file against the telemetry database to ensure that the MSIDs exist, are available to the user (permissions), and that a valid data representation is chosen
- Results of the validation are displayed in the 'View Auto-Validation Report' dialog box

3.2.4 Operate Display

The 'Operate Display' functional block is the 'heart' of the application, and represents the processing performed to actually operate the display file. The line drawn

from this block and the external block entitled 'Validate Display' indicates that the user can choose to perform a 'Check for UDEs' validation during the operation of the display. This validation process verifies that any UDEs that are referenced within scripting directives (e.g., a pushbutton has a directive: 'start display XY_EXPERIMENT', the display XY_EXPERIMENT is the referenced UDE) exist and are available on the workstation. If not, the user is informed. The line drawn to and from the 'Change Operation Modes' block indicates that the user can change the way the display operates by setting items such as: update rate, data mode, and database version. The processing that occurs within this block includes:

- Updates an active display with MSID telemetry data in the update mode and update rate specified
- Displays Limit/Expected State sensing information if requested
- Displays data status information if requested
- Issues scratchpad line directives initiated by the user via use of pushbuttons, input text fields, input sliders, scratchpad line fields, or user defined pulldown menus
- Processes user initiated changes in the data mode
- Processes user initiated changes in the update rate
- Processes user initiated changes in the update mode
- Performs a 'Check for UDEs' validation when requested by the user
- Performs graceful exit in response to user input or termination signal

3.2.5 Update MSID Values

The 'Update MSID Values' functional block represents the processing that occurs each time new values are requested for the display. The processing occurring here is limited and includes:

- Requests new data sample for each MSID referenced in the display
- Requests other processing to be conducted on the data:
    - ◊ Calibration
    - ◊ Limit/Expected State sensing
    - ◊ Conversion

3.2.6 Change Operation Modes

The final functional area is the 'Change Operation Modes' block. This block represents the processing that occurs when the user changes one of the operation modes. The user can change these items at any time during the operation of the application which is indicated by the line drawn to the 'Initialize DO Application' block (before a display is opened) and the line to the 'Operate Display' block (during operation of a display). The processing that occurs includes processing changes in:

- Data mode
    - ◊ Real-time
    - ◊ Playback
    - ◊ Dump
- Database version

- ◊ Performs auto-validation

- Update mode
  - ◊ Cyclic
  - ◊ Refresh
  - ◊ Pause
- Update Rate (1-999 seconds)

4. Application Description

The EHS performs three major functions in support of the projects mentioned previously. These include:

1. Acquisition of telemetry data for use by the EHS community for display and computation purposes.

2. Provide capabilities for users to generate and transmit commands to spacecraft supported by the EHS.

3. Provide Mission Support Services (MSS) required to provide process flow control of mission execution and status information.

These three functions can be further partitioned into twelve service areas. These service areas include:

1. Data Acquisition and Distribution Services
2. Telemetry and Network Services
3. Near Real-Time Services
4. Command System Services
5. Mission Support Services
6. Database Services
7. System Monitor and Control
8. Scripting Services
9. Computation Services

10. Display Services

11. General Purpose Utilities

12. Strip Chart Recording Services

The DO application is a component of the Display Services functional area from the list above and utilizes aspects of some of the different services. Each of these service areas provide common library routines for use by other applications in the EHS to perform specific services/functions. DO utilizes functions from the following service areas:

1. Database Services
2. Command System Services
3. Telemetry and Network Services
4. System Services
5. System Monitor and Control
6. Scripting Services
7. General Purpose Utilities

Figure 4-1 depicts the relationship between the DO application and these service areas.

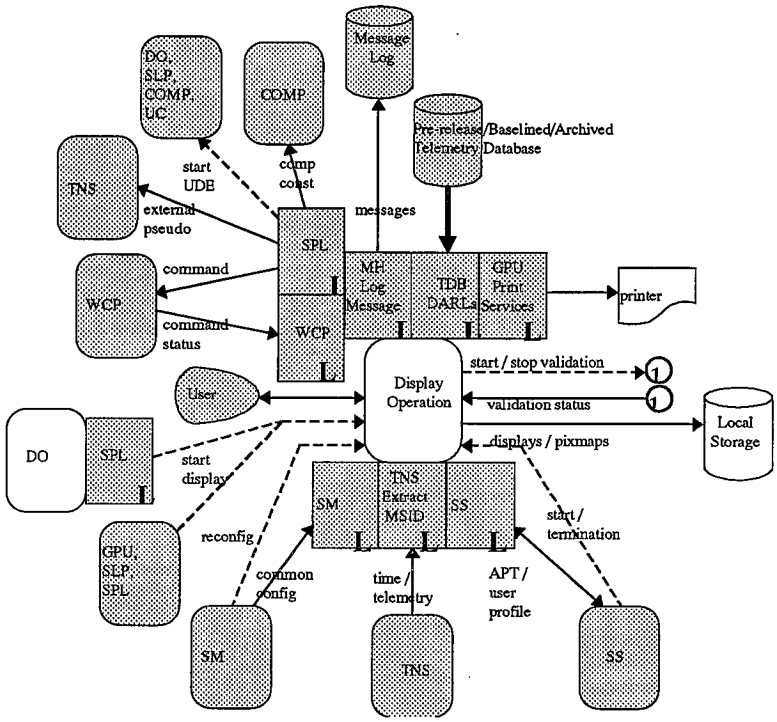

Figure 4-1 Display Operation Architecture Drawing

The following paragraphs provide a brief overview of this diagram and how DO performs its functions. The purpose of this information is to provide a better understanding of the relationship between the different service areas of the EHS system and provide a top-level view of the internal design of the application.

Upon execution of the DO application the user profile information is obtained from System Services (SS) by calling a System Services supplied library function. This function returns information about the user's account to include a user ID and a group ID. DO then utilizes a function supplied by System Monitor and Control (SM) to retrieve common configuration information for the user. This information includes current database version numbers, mission and support information, and project information. Following this, DO

calls a Telemetry and Network Services (TNS) function to initiate the reception of the time information distributed on the network by the Time Distribution System (TDS).

To digress somewhat, after the creation of a display in the Display Generation application, the user saves the newly generated display in a file called a display file. This file contains information about the display to include specific information about the objects and associated MSIDs if applicable. When the user selects a display to operate in DO, the application opens the selected display file and parses its contents. Data structures are created with all the information necessary to create the widgets and draw the graphics objects. These data structures are used throughout the operation of the display to manage the objects and communicate with the X Window System. Xlib, Xt, Motif, and TeleUSE functions are used to manage the GUI and display objects.

DO receives telemetry from the network through use of routines provided by Telemetry and Network Services (TNS). When the user selects a display to operate, DO will send a list of valid MSIDs used in the display to Telemetry and Network Services (TNS) and also specify to TNS the data mode and database version in which the display will operate. TNS initializes the workstation to retrieve the appropriate data packets containing the desired MSIDs. The data is retrieved from the network and stored in a buffer on the user's workstation by TNS. At this point DO retrieves MSID values as needed. The DO application once again uses a TNS function to retrieve data from the buffer and to update MSID values with the newest samples of the requested MSID. DO uses these values to update the output objects in the display.

When the user decides to close a display or exit the DO application, certain procedures are followed to ensure graceful termination of the display. A TNS library function is utilized to cancel the telemetry extraction process and deallocate associated memory. The objects in the display are then closed and all of the memory associated with the object data structures is deallocated. If the user is exiting the DO application then the application exits gracefully.

4.1 Display Operation Detailed Description

To review what has been discussed in previous sections, the DO application provides the user with the capability of viewing specified command, telemetry, and processed data by operating a display previously created using DG. These displays can either be initiated upon the startup of DO or can be opened at anytime during the execution of DO. The DO application allows the user to view telemetry values and graphical displays of data via use of output objects (bar charts, plots, text fields, etc.). The user can also uplink and update commands; start and stop displays, computations, and scripts; and update external pseudo telemetry and computation constants through the use of Scratchpad Line (SPL) directives associated with input objects in the display (pushbuttons, input text fields, etc.). Other capabilities provided to the user by DO are discussed later

4.1.1 Display Operation Initialization

Upon execution of the Display Operation application the user profile information is obtained from System Services (SS) by calling a SSUP_UPAR supplied library function.

This function returns information about the user's account. Display Operation then utilizes a function from System Monitor and Control's SM library to retrieve common configuration information. This information includes current database version numbers, mission, operation support mode, and project information. This information is used by DO in the validation and operation of displays.

Following this, DO calls a function from the TNS_Extract_MSID library to initiate the reception of the time information distributed on the network by the Time Distribution System (TDS). Display Operation then checks to see if command line arguments are provided in the command that started the execution of DO. The command line arguments that can be utilized include the following:

- Display Filename
- Data Mode
- Telemetry Database Version
- X/Y Coordinates

If no arguments are present then the application defaults to the real-time data mode and baselined Telemetry Database. Once the application has started, the user can select the data mode and database version from within the application prior to opening the initial display. If arguments are present then DO automatically opens the named display for operation, and the display operates in the data mode and database version provided in the arguments. The X/Y coordinates are optional, and are used to position the application main window on the monitor. If the X/Y coordinates are not provided then DO utilizes default coordinates to position the main window.

Once the Display Operation application has been initialized the user can select displays to operate. If the user selects a display to be opened and operated, and there is a display currently being operated, then the old display is replaced by the newly selected display. If the user desires to have multiple displays operating simultaneously then a separate DO application must be running for each individual display. However, there cannot be multiple instances of the same display running with the same data mode and database versions.

4.1.2 Automatic Display Validation

When a display file is opened, DO checks to see if this display has been previously validated and, if so, for which database version numbers. If the display has not been validated against the selected database in which the display will operate, then Display Operation automatically performs the validation of the display prior to execution. This automatic validation is a top-level validation and checks the selected Telemetry Database to see if the MSIDs associated with the display exist and are available to the user and that the MSIDs have a valid data representation. Display Operation utilizes the TDB_DARL library to access the Telemetry Database during the automatic validation. Any invalid MSIDs found in the display are indicated to the user in the form of an error message in the View Auto-Validation Report dialog window.

Invalid MSIDs do not affect the rest of the objects in the display, and therefore, the display is still operable and only those objects with associated invalid MSIDs are inoperable. Scratchpad line directive syntax, UDEs, and command mnemonics are not validated by DO during the automatic validation process.

4.1.3 Operation Startup Lists

Each display file can have a list of displays and/or computations that are to be started when the display is opened. This startup list is defined by the user in the Display Generation application. DO reads the items in the operation startup list and performs the necessary actions to open the displays and/or start the computations. This is done by using library functions provided by Scripting Services. Each item in the startup list is converted to a Scratchpad Line (SPL) directive and is then validated and executed by use of a routine from the SLP library. A separate process is started for each UDE, and each process is initiated using the parent application's current data mode and database version. For more details on Scratchpad Line directives, see section 4.1.7, Scratchpad Line Directives.

The computations and displays specified in the operation startup list can also be flagged by the user, so that it can be terminated upon shutdown of the initiating display (see section 4.1.9, Display Shutdown). The user flags these UDEs when the startup list is generated in the Display Generation application.

4.1.4 Data Acquisition and Execution

Display Operation receives telemetry from the network through use of routines from the TNS_Extract_MSID library. When the user selects a display to operate, the DO application checks to see if an automatic validation is needed and does so if necessary (see section 4.1.2, Automatic Display Validation). Upon completion of the validation (if performed) and after initiation of the UDEs in the operation startup list, DO uses a

function from the System Services Process Manager (SSPM) library to verify that the same display is not currently being executed in the same data mode and database versions in another DO application. If it is determined that one is being executed then the display is not opened and a message is displayed to the user. In the case that one is not being executed then DO sends the following information to TNS:

- Valid MSIDs
- Parameter to request only the most recent sample
- Parameter to request types of processing (unprocessed, converted, calibrated, and/or LES)
- User ID (global variable)
- Database Version Number (global variable)
- Data Mode (global variable)

A display can operate in only one data mode and one database version at any given time. If the user desires to concurrently operate a display in different data modes and/or database versions then he/she must execute separate Display Operation applications for each data mode. DO utilizes a library function from the TNS_Extract_MSID library to provide the required information to TNS. TNS then initializes the workstation to receive the appropriate data packets containing the desired MSIDs.

Data is received from the network and stored in a buffer on the user's workstation by TNS. At this point DO retrieves MSID values as needed. The DO application once again uses a TNS_Extract_MSID library function to retrieve data from the buffer (at the update rate interval) and to update MSID values with the newest samples of the requested

MSID. DO uses these values to update the output objects in the display. When a loss of signal or stale/old data status is returned, the object is updated with the latest value received for the MSID. Values are not retrieved for any MSIDs that were determined to be invalid during the validation of the display. The objects associated with the invalid MSIDs are handled differently according to the type of object. Table 4-1 provides a consolidated list of how DO indicates an invalid MSID status for each type of object used in displays.

Object	Invalid MSID Indicator
Line	Reserved pattern
Primitive Graphic (circle, square, polygon, etc.)	Reserved Pattern
Text	Reserved Pattern
MSID Text Object	Value field filled with asterisks followed by the invalid MSID data status character.
Plot/Chart	Value not plotted
Pixmaps	Reserved pattern
Radial Meter (180/360)	Needle not shown. Value filled with asterisks.
Toggle Button/Radio Box	Buttons unset. Labels filled with asterisks.
Output Slider	Value defaults to minimum value. Label filled with asterisks.
Input Slider	When the user attempts to operate the object, a message is presented notifying the user that the associated MSID is invalid.
Input Data Field	When the user attempts to operate the object, a message is presented notifying the user that the associated MSID is invalid.
Pushbutton	When the user attempts to operate the object, a message is presented notifying the user that the associated MSID is invalid.

Table 4-1 Invalid MSID Indicators

The TNS_Extract_MSID library function used to retrieve data also returns the status of Limit/Expected State Sensing (LES) (if requested) and the status of any processing (e.g., calibration) that was requested to be performed. This information is used

to inform the user of the status of the data. Limit sensing information is included in this status and is presented on the display only if requested by the user. Section 4.1.5, User Controlled Capabilities, discusses this user controlled option.

If an MSID's value goes into caution or warning limit ranges currently specified for that MSID, the object associated with that MSID indicates the LES status to the user. This is done by the use of colors or user selected pixmaps representing the limit ranges and expected states. The colors yellow and red are reserved for representing 'caution' and 'warning/out of state' conditions respectively. Table 4-2 provides a consolidated list of the colors used by DO when presenting LES information to the user for MSIDs with discrete values (group and bit). Table 4-3 provides a consolidated list of the colors used by DO when presenting LES information to the user for MSIDs with analog values (LES ranges).

OBJECT	EXPECTED STATE COLORS		NOTES
	In State	Out of State	
Line	User defined color other than yellow or red	Red	The line changes colors based on the state code of the MSID.
Primitive Graphic (square, circle, etc.)	User defined border color other than yellow or red User defined fill color for the state code	Red border User defined fill color	Border colors and fill colors change based on the state code of the MSID. The user can assign up to 32 different colors for the fill color to an MSID and any color can be chosen.
Text/MSID Text	User defined color for the text other than yellow or red	Red text	The text changes color based on the state code of the MSID. The user can assign up to 32 different colors to an MSID.
Pixmap	User defined pixmap with any colors No border	User defined pixmap with any colors Red border	The pixmap changes based on the state code of the MSID. The user can assign up to 32 different pixmaps to an MSID.
Toggle Button	No border	Red border	The object border and the button label turns red when out of state.
Radio Box	No border	Red border	The object border turns red when out of state.

Table 4-2 Discrete MSID LES Colors

OBJECT	LIMIT RANGE COLORS			NOTES
	Nominal	Caution	Warning	
Line	User defined color other than yellow or red	Yellow	Red	The line changes colors based on the value of the MSID.
Primitive Graphic (square, circle, etc.)	User defined border color other than yellow or red User defined fill color	Yellow border color User defined fill color	Red border color User defined fill color	The graphic's border colors and fill colors change based on the value of the MSID.
Text/MSID Text	User defined color for the text other than yellow or red	Yellow	Red	The text changes color based on the value of the MSID.
Plots/Chart	User defined color other than yellow or red	Yellow	Red	The lines, points, bars, or pieces of the plot/chart change colors based on the value of the MSID. Line styles or fill patterns are used to differentiate between MSIDs.
Pixmap	User defined pixmap with any colors No border	User defined pixmap with any colors Yellow border	User defined pixmap with any colors Red border	The pixmap changes based on the value of the MSID.
Output Slider	User defined color other than yellow or red	Yellow	Red	The slider value changes color based on the value of the MSID.
Radial Meter (180 & 360)	User defined color other than yellow or red	Yellow	Red	The needle changes color based on the value of the MSID.

Table 4-3 Analog MSID LES Colors

The data status information returned by TNS is also used to present data status on the display via the use of status characters. Status characters are only used when displaying data in a MSID text object. These status characters represent a variety of information about the data to include loss of signal (N), old data (S), invalid MSID (U), etc. This status character is one character and is displayed in the last position of the MSID value string within the output text field. The user has the capability to turn the status characters on or off at any time during execution of a display. The action of turning the status characters on or off affects all objects in the display that utilize the status characters. The Display Operation status bar provides a visual indication to the user whether the status characters are on or off for the display. Table 4-4 contains all status

characters to be used and their meaning. The status characters are listed in descending order of precedence.

Status Character	Definition
'U'	Unrecognized/Invalid MSID
'p'	Private Measurement - No Privilege to Access
'M'	Invalid Format for Data Type
'n'	Packet not Defined in the Routing Table
'$'	Invalid Status Received from TNS (Old data used)
'S'	Data Condition - Old/Stale Data
'N'	Source Status - Loss of Signal (No data returned - old data used)
'R'	Source Status - Source Initialized (Data Stream Replacement - old data returned)
'~'	Data Quality - No Data (No data returned - old data used)
'F'	Data Quality -DQ Failed w/ Override (New data is used - bad data)
'?'	Data Quality - DQ Suspect w/ Override (New data is used - suspect data)
'f'	Data Quality - DQ Failed (No data returned - old data is used)
'x'	Data Quality - Data is Suspect (No data returned - old data is used)
'D'	Decom/Conv/Cal Status - Decom Error
'C'	Decom/Conv/Cal Status - Conversion Error
'c'	Decom/Conv/Cal Status - Calibration Error
'l'	LES Status - LES Error
'H'	LES Status - Warning High
'^'	LES Status - Caution High
'v'	LES Status - Caution Low
'L'	LES Status - Warning Low
'E'	LES Status - Out of Expected State
'#'	Data Format Field Overflow
' ' (space)	Source Status - Acquisition of Signal (Parameter is okay - new data is returned)

Table 4-4 Status Character Definitions

4.1.5 User Controlled Capabilities

The Display Operation application provides the user with a number of functions and settings that can be performed, selected, and/or configured. This section describes these capabilities and their affect on the operation of a display.

4.1.5.1 Data Modes and Database Versions

The user can change the data mode or database version in which the DO application is running at any time during execution of the application. The data modes the user can select include real-time, dump, and playback. There are thirteen playback channels from which the user can choose. Database version types include baselined, and archived. The default data mode is real-time, and the default database version is baselined. The data mode and/or database version can be changed by the user at any time during operation of a display.

Before DO changes the data mode and/or database version, a confirmation dialog box appears on the monitor prompting the user for confirmation of the action about to take place. DO then confirms that a valid database is selected for the data mode. If the user has selected the data mode and database version in which the display is currently running, then no changes are made. When the user selects a different data mode or database version for a display, DO again utilizes a function from the System Services SSPM library to insure that the same display is not currently being executed in the same data mode and database versions in another DO application (see section 4.1.4, Data Acquisition and Execution).

If it is determined that there is another display being executed as described, then the user is notified that the desired changes cannot be implemented. Otherwise, the DO application clears the display of all previous data and then utilizes another TNS_Extract_MSID library function which terminates telemetry extraction for the MSIDs specified and frees allocated memory used by TNS for the return of samples and status information. Upon successful completion of this decom cleanup process, DO uses a TNS_Extract_MSID library function to initialize decom with the new data mode and database version information provided. The procedures duplicate the tasks performed at the initial startup of a display. A TNS_Extract_MSID function is again used to retrieve MSID values from the buffer and update the display objects as described in the previous section.

4.1.5.2 Update Rates

During operation of a display the user can also select the method by which the display is updated. The user can select from one of the following update modes: cyclic, refresh, and pause. During the cyclic update mode the display is updated at intervals selected by the user (e.g., updated once every five (5) seconds). The display can be updated as often as once every second which is the default. The user can change the update rate (in seconds) for the display at any time during operation of a display while in the cyclic update mode.

Although many DO applications can be executed simultaneously, the minimum requirement for DO is to operate two displays, each containing one hundred MSIDs with a two second update rate per display. The ability of a DO application to update its display

at the requested update rate may be adversely affected by the number of applications running concurrently with the DO application. When in pause or refresh modes the capability to set/change the update rate is not needed and is therefore not available to the user. When the user selects the refresh mode for updating the display, the display is updated only upon the user's request and the display is updated with the most recent data available. The pause mode actually pauses the updating of the display until the user selects either the refresh or cyclic methods of update. The update mode can be changed at any time during the operation of a display.

The update mode and update rate (if cyclic update rate is used) of a display when opened by the user is contained in the display file and is specified during the creation of the display using the DG application. If the user did not explicitly specify the update mode during generation of the display then the update mode defaults to cyclic with an update rate of one second. Update modes do not affect the TNS telemetry extraction process. Display Operation simply does not make the necessary function calls to get new data until needed.

4.1.5.3 Configuring LES Processing

Display Operation provides the user with the capability to request that limit/expected state sensing (LES) be performed on the data being extracted by TNS for the display. This capability is performed on a per object basis, and can be turned on or off at any time during the operation of a display. Projects can configure the application to provide special LES sensing indicators in two ways.

The first of these configurable items is to have a reserved color for the application to indicate when LES sensing is turned off for each object. The second of these configurable items is to use the reserved color to indicate when limits are not defined in the local table for each MSID. DO can be configured to utilize none, one, or both of these methods.

The user profile obtained from System Services (see section 4.1.1, Display Operation Initialization) will contain the information used by DO to configure the application in one of these three ways. Table 4-5 provides a description of how the reserved color is used when either one of these configurable items is utilized. When a display is opened, all applicable objects default to 'LES On' and the user must purposely turn LES off for each object that LES information is not desired.

Object	LES Sensing Off and Limits not Defined Indicator
Line	Reserved color used for line only when the line is used to represent limits or state violations. If the line is used to represent state codes then LES sensing is not applicable and therefore no indicator is used.
Primitive Graphic (circle, square, polygon, etc.)	Reserved color used for border.
Text	Reserved color used for text only when the text is used to represent limits or state violations. If the text is used to represent state codes then LES sensing is not applicable and therefore no indicator is used.
MSID Text Object	Reserved color used for text.
Plot/Chart	Reserved color used for individual lines/pie pieces/bars in the plot/chart.
Pixmaps	Reserved color used for border
Radial Meter (180/360)	Reserved color used for needle color.
Toggle Button/Radio Box	Reserved color used for border.
Output Slider	Reserved color used for slider value.

Table 4-5 LES Sensing Off and Limits not Defined Indicators

4.1.5.4 Other Capabilities

As mentioned previously, a display that the user selects to operate is validated automatically only if the display is not valid for the selected Telemetry Database. Automatic validation is a top-level validation against the selected Telemetry Database. Display Operation also provides the user with the option to perform a 'Check for UDEs' validation on a display after it has been opened. The 'Check for UDEs' option checks all referenced UDEs in the display that is currently being operated to ensure that they are resident on the user's workstation and that the user has privilege to access them. The 'Check for UDEs' option does not perform a validation on the referenced UDEs.

The capability of selecting a plot within the display and zooming in this plot to allow greater resolution is also provided to the user by Display Operation. The user can select the desired plot to zoom and then unzoom at anytime. A bounding box of the area of the plot to be zoomed is drawn by the user using the mouse cursor. This area is enlarged to the size of the plot boundary and the axes are changed accordingly. The actual plot size does not change, only the resolution of the X axis and Y axis. When the user selects a plot to unzoom, the plot returns to its original state (i.e., the state before any zooming was performed). The zooming of a plot does not affect the updating of the plot nor the rest of the objects in the display.

Each object in a display can have text information associated with it that was entered by the user during creation of the display in the Display Generation application. This text is known as 'recall text' and can be viewed by the user during the operation of a display. If the user did not enter any recall text for the object then the recall text will be blank. The user also has the capability to print a snapshot of the display at any time during the operation of a display. DO utilizes a GPU_Print_Services library function to perform this action.

4.1.6 Messages

The Message Handler (MH) interface provides Display Operation with the capability of logging messages that are generated during the operation of a display. The Display Operation application logs messages through a function from the MH_Log_Message library. Messages are also displayed to the user in the message area portion of the Display Operation main application window. However, only those

messages pertaining to the DO application are displayed in this area. These messages remain in the message area until the application is exited or until the user voluntarily clears the message area.

4.1.7 Scratchpad Line Directives

Scratchpad Line (SPL) Directives are processed by the Scripting Language Processor (SLP) via a routine from the SLP library. SPL directives provide the user with the capability to start and stop displays, computations, and scripts. A user can also uplink and modify commands, and update external pseudo MSIDs and computation constants through the use of SPL directives. SPL directives are issued by the DO application by use of input objects (pushbuttons, input text fields, and input slider scales), user defined pulldown menus, and/or operation startup lists (see section 4.1.3).

When the user executes an action on an object in the display (e.g., pressing a pushbutton) that has an SPL directive associated with it, the DO application requests confirmation of this action prior to actual execution. If the user confirms that the action is to occur then DO issues the SPL directive to the Scripting Language Processor. A function from the SLP library is used to issue the directive. Once again, as for all actions, a message notifying the user of the action just initiated is displayed to the user in the application message area and is also sent to the Message Handler for logging. Invalid directives are not issued and a message is displayed to the user.

The user also receives messages on the status of the directive that was issued (e.g., status of command) as these become available. When a command is issued the Workstation Command Processor (WCP) Control_Uplink process writes status

information about the command to the DO application's command response queue. DO utilizes a function from the WCP library to retrieve the information from the queue when the command responses are written. If the display is operating in the playback data mode then commands and command chains cannot be uplinked. If the user tries to issue a directive to uplink a command while operating in the playback data mode, the directive is sent to the SLP for processing. A status is returned from the SLP library function indicating that commands cannot be uplinked from applications operating in the playback data mode. A message notifying the user of this status is then displayed.

4.1.8 Reconfiguration

Upon receipt of a reconfiguration signal the DO application retrieves the new common configuration information, determines the nature of the reconfiguration and performs actions as necessary. The new common configuration information is obtained by calling a function from System Monitor and Control's SM library. DO uses this new information to determine what has been reconfigured.

If there is a reconfiguration in the database version(s) and there is a display currently being operated that is using one of the reconfigured databases, then DO shuts down the display by using the shutdown procedures outlined in section 4.1.9, Display Shutdown. In the event of a reconfiguration in the Mission, Operational Support Mode, or Project (MOP), DO shuts down the display, if one is being operated, and then terminates utilizing the procedures outlined in section 4.1.9, Display Shutdown. If DO is in the process of an automatic validation and a database version reconfiguration is received, then DO disconnects from the database, retrieves the new common configuration information,

and informs the user that a reconfiguration has occurred and that the display must be reopened.

4.1.9 Display Shutdown

Displays can be shutdown in a number of ways during their operation. A display and/or its parent application can be closed or shutdown as a result of any of the following actions:

- The user opens another display and replaces the one currently operating (display shuts down)
- The user exits the application (display shuts down and DO terminates)
- The action of an SPL directive (display shuts down and DO terminates)
- The receipt of a termination signal from the System Services Process Manager (display shuts down and DO terminates)
- The receipt of a reconfiguration signal from SM as the result of a change in the database versions (display shuts down if utilizing a database that has changed)
- The receipt of a reconfiguration signal from SM as the result of a change in the Mission, Operational Support Mode, and Project (MOP) (display shuts down and DO terminates)
- The receipt of a termination signal for any reason (display shuts down and DO terminates)

When any of these events occur, certain procedures are followed to ensure graceful termination of the display and/or application. A TNS_Extract_MSID library

function is utilized to cancel the telemetry extraction process and deallocate associated memory. If Display Operation is performing an automatic validation then DO disconnects from the database and stops the automatic validation. If the application is in the process of performing a 'Check for UDEs' validation then the validation process is terminated. The objects in the display are then closed and all of the memory associated with the object data structures is deallocated. Also, any computations or displays that were initiated by the parent display in its operation startup list that were flagged to be terminated upon the display's shutdown are shutdown.

Termination of these UDEs is accomplished by use of SPL directives. The parent DO application issues the SPL directive to terminate the flagged UDEs using the filename that was used when the UDE execution directive was issued. If the UDE cannot be terminated for any reason, the user is notified in the form of a message in the application message area. Upon termination of all the flagged UDEs in the operation startup list, if the application itself is terminating either by receipt of a termination signal, MOP reconfiguration, or by choice of the user, then the Display Operation application process exits gracefully.

During operation of a display the user may have changed certain characteristics of how the DO application and display are configured (e.g., message area setup, data mode, update rate, etc.). When the user opens a new display to replace the current one, some of these changes are no longer in effect for the new display. Table 4-6 summarizes the user settings and how they are affected when a display is closed.

User Settings	Persistence
Message area setup	Effective until the application is exited.
Data mode	Effective until the application is exited.
Update rate	Effective for that display only.
Update mode	Effective for that display only.
Printer setup	Effective until the application is exited.
Zoom	Effective for that display only.
LES on/off	Effective for that display only.
Database version	Effective until the application is exited.

Table 4-6 User Preference Items and Their Persistence

5. Solution Approach

5.1 Code

The Display Operation application was written utilizing the C language and the TeleUSE D language which is discussed in section 5.1.1. The application code is a combination of original software, C libraries developed 'in-house' and used across all EHS applications, code re-used from prototypes developed previously, and COTS libraries.

The code followed the design layout described in section 3.2, Detailed Design. Each of the functional blocks was further divided into more specific tasks and coded. The following is an approximate break down of the number of lines of code written for the application:

- C code specifically for Display Operation: ~9000 LOC
- D code specifically for Display Operation: ~2500 LOC
- C code used across applications (generic code): ~7000 LOC

The generic code specified above is shared between the Display Operation and Display Generation applications. This code included functions to open and read the

display file; allocate memory for colors, fonts and objects; and draw the display in the application window.

All code was required to adhere to coding and naming standards specified in the Enhanced HOSC System Coding Standard and the Enhanced HOSC System Naming Standard documents.

5.1.1 TeleUSE

The user interface for the Display Operation application was developed utilizing a GUI development tool called TeleUSE 3.0. TeleUSE 3.0 provides the developer with the capability to generate Motif user-interface components by dragging and creating interface widgets from a palette containing the Motif widget set.

GUI development tools can be broadly categorized into two types. Interface Development Tools (IDTs) are used strictly for building the interface and nothing else. User Interface Management Systems (UIMSs) provide additional functionality to include application development or scripting tools. TeleUSE 3.0 is an example of a UIMS. TeleUSE 3.0 is a complete application development tool and has its own development language called the Dialog Scripting Language also known as D. The D language allows a developer to display the UI, manipulate the UI as needed, obtain information about the state of the UI (e.g., data values entered into text fields by the user) and make calls to C routines with UI data supplied as arguments. The D language was used to write the code that actually managed the application main window and enter the X event loop.

D events (similar to C functions) were 'attached' to each of the user interface components that accept user input or that the user can interact with (e.g., pulldown menu

options, input text fields, etc.). These D events are activated when the user utilizes the respective UI component. The D event normally calls a C function and passes to this function any data associated with the UI that is pertinent to processing that is performed by the called function. So it is clear that most of the user interface code (code used to manage the windows, etc.) was written in D while all of the application code (code used to update display objects, draw primitive graphics, etc.) was written strictly in C.

5.1.2 XRT/Graph

XRT/Graph is a COTS product developed and distributed by the K.L. Group located in Ontario, Canada. The XRT/Graph widget is used by the Display Operation application to display telemetry data in various graphical forms to include scrolling time plots, scatter plots, bar charts, and pie charts. XRT/Graph basically adds a new widget to the Motif widget set.

The graph widget displays data graphically in a window and has resources that determine how the graph looks and behaves. Some of these resources are set by the user upon generation of the display, and other resources are managed by Display Operation during run-time operation of the display. The graph widget has many resources which allows flexibility in the look and feel of the graph. Some of these resources include:

- Graph type (plot, pie, bar, etc.)
- Header and footer positioning, border style, text, font, and color
- Line patterns, fill colors and patterns, point style, size and colors
- Legend positioning, orientation, font and color
- Axis labeling, minimum and maximum values, and tick marks

5.2 Testing

Formal testing of the application was performed using a number of test and debugging tools. The two that were most often used are discussed in the following sections. The application was tested as a unit and specific goals were desired. These goals included:

- 100% line coverage
- Execution of critical values (when relevant):
    1. Nominal
    2. Lower Bound
    3. Upper Bound
    4. Lower + 1
    5. Lower - 1
    6. Upper + 1
    7. Upper - 1
    8. Zero

Test coverage is used to show how many lines of code, functions, and files have been executed. The resulting data can be used to provide some idea of the overall quality of the testing. Coverage analysis of the Display Operation application was performed by use of a test tool called Insight.

The source code was compiled with the 'insight' command instead of the normal C compiler. This 'instrumented' the code with information to include checks for coverage analysis and then utilized the normal compiler to create instrumented object code. When

the program is executed, a database is created by Insight containing information about coverage. This information can then be analyzed by utilizing the 'tca' utility provided by Insight. The test coverage data provides several levels of summary:

- Overall
    - * Shows the percentage of coverage at the application level
- Function
    - * Shows coverage of each individual function in the application
- Block
    - * Shows coverage broken down by individual program statement blocks

5.2.1 Workshop Debugger

The Workshop Debugger is a debugging tool provided in the Workshop suite of tools distributed by Silicon Graphics, Inc. with their workstations. Workshop Debugger is a UNIX source-level debugging tool that provides a windows environment for performing various debugging functions to include viewing program data and execution status. It is a very versatile and useful tool that aided greatly in the development and testing of the Display Operation application. The tool allows users to perform numerous actions to include:

- Set traps
- Inspect Debugger data
- View and change source code

Setting traps is a major part of the debugging process. This allows the developer to halt the program execution at a specified point in the source code and inspect data values at these points. The debugger allows the developer to set traps:

- at a line in a file
- at an instruction address
- on entry into/exit from a function
- when a signal is received
- when a variable or address is written to, read from, or executed

When the process stops, the user can inspect many items of interest to include:

- call stack at the breakpoint
- value of a selected expression
- value, type, or address of variables
- data structures
- value in memory location

This tool was used extensively during the development process and proved to be invaluable. Run-time errors that caused segmentation violations, bus errors, etc. were detected and corrected much more rapidly than would have been possible without the aid of the tool.

5.2.2 Software Test Works' CAPBAK/X

Software Test Works (STW) is a suite of tools designed to provide an automated testing environment for applications that utilize graphical user interfaces. CAPBAK/X is a test capture/playback tool for the X Windows System and is one of the many tools

provided in the STW suite. CAPBAK/X was utilized in the testing of the Display Operation application.

Traditional software testing involves manual testing or a few written scripts stored in batch files. This is generally sufficient to fully test the product. However, today's applications are much more complex and a product will likely have to undergo testing numerous times. In this case the traditional methods of testing can be very time consuming. Automated testing is the solution to this problem.

CAPBAK/X provides a means of automating software testing. A test session is created during which the tester provides mouse and keyboard event inputs to the product being tested. During the session the user can capture images off the screen which serve as baseline files and will be compared with re-runs of the recorded session. Future tests simply entail playing back the recorded test session.

CAPBAK/X records all the X Window activities (including all keyboard and mouse events). During the recording session CAPBAK/X automatically generates input statements that represent the user input. These statements are saved in a test script file called a 'keysave file'. This keysave file is utilized to playback the test session. When a test is played back, the same input statements are sent to the program which executes these inputs exactly as before. Reliable playbacks are ensured by means of built-in and user-defined synchronization points. using this method, automatic test cases can be re-run as many times as needed providing a much more efficient and time saving test method for the application.

6. Application Examples

The following scenarios provide 'real-world' examples of the Display Operation application in use. They describe how the user utilizes the DO application to view telemetry data, uplink commands, and various other capabilities. The scenarios are intended to provide a better understanding of the 'real-world' employment of the DO application at NASA's Marshall Space Flight Center in Huntsville, Alabama.

6.1 Scenario 1

The user initiates the Display Operation application which causes the main application window to appear on the monitor. The main window has a menu bar, a status bar, a display area, a control bar, and a message area (see Figure 1-2). Since a display file has not been opened, the display area is empty.

The user decides to run a display file entitled 'DISP_1' that was created in the Display Generation application. The user selects 'Open...' from the 'File' pulldown menu option on the menu bar. A file selection dialog appears in which a list of available display files appears. The user selects DISP_1 from the list and the display is opened. The user receives a message in the message area stating that the display DISP_1 was opened. When the display is opened, the status bar indicates that the datamode is real-time, the status characters are on, LES is turned on for all objects, and the time reference is GMT. The display area contains the textual and graphical objects for the display DISP_1 (see Figure 6-1). The display is updating in the cyclic mode once every two seconds, therefore each object is updated with the most recent telemetry data at this rate. The user decides

that the display needs to be updated once every second. The user selects 'Set Cyclic Update Rate...' from the 'Define' pulldown menu option on the menu bar. The 'Set Cyclic Update Rate' dialog appears and the user enters a '1' in the text field and presses the 'Set' button. The display now updates with the most recent data every second.

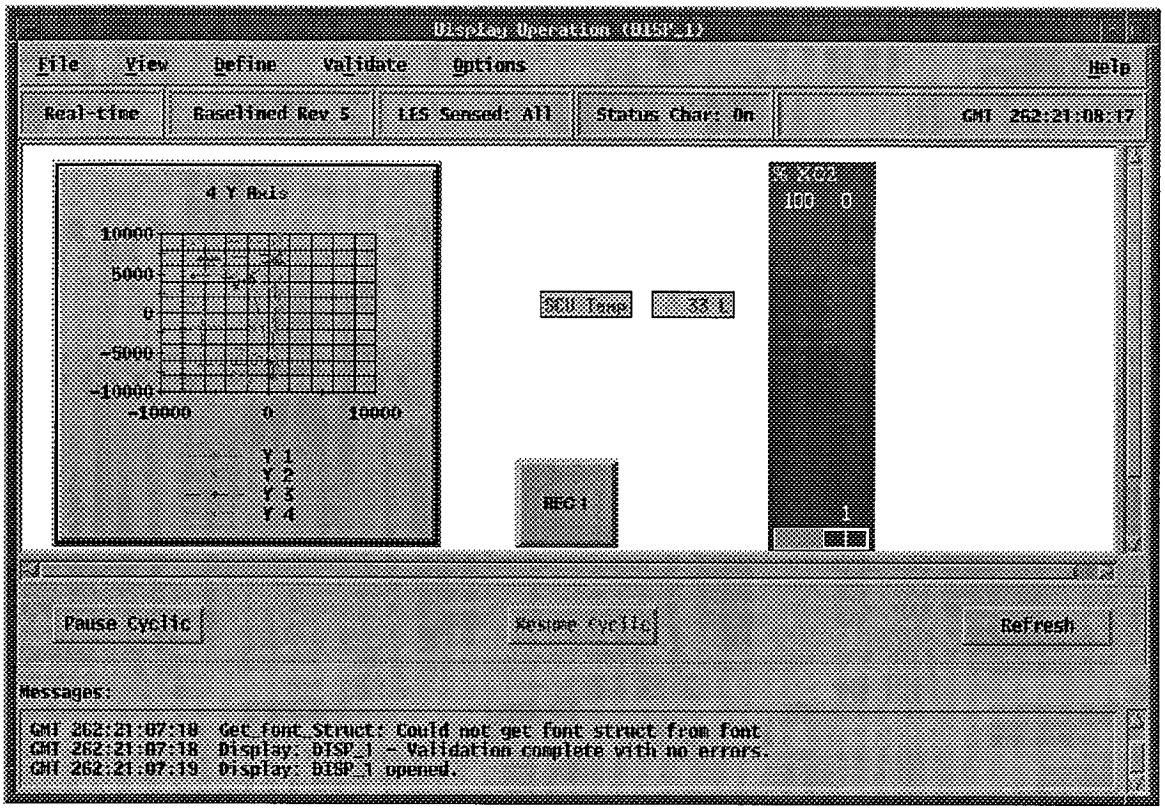

Figure 6-1 Display 'DISP_1' Opened in DO

One of the objects in the display is a pushbutton (input object) labeled 'REC 1' which has a scratchpad line directive associated with it. This was done at the time the display was created in the Display Generation application. The user wants to check the recall text associated with the object because he knows that the scratchpad line directive was also entered as recall text for informational purposes at the time of generation. The user places the mouse pointer over the object and presses the right mouse button which

causes a pop-up menu to appear. One of the items on the pop-up menu is 'View Recall Text'. The user selects this option and a dialog box appears containing the recall text associated with that object. This object's recall text states: "This pushbutton initiates the SPL directive 'start command form recorder_1' " indicating that the directive is a command to start recorder number 1 (see Figure 6-2).

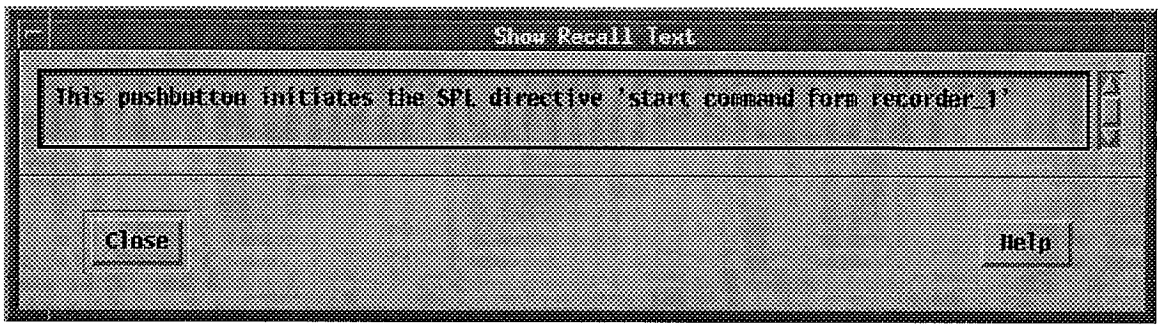

Figure 6-2 Recall Text for Pushbutton 'REC 1'

After closing the recall text window, the user wants to issue the scratchpad line directive and presses the pushbutton. A dialog box appears requesting confirmation of the directive (see Figure 6-3).

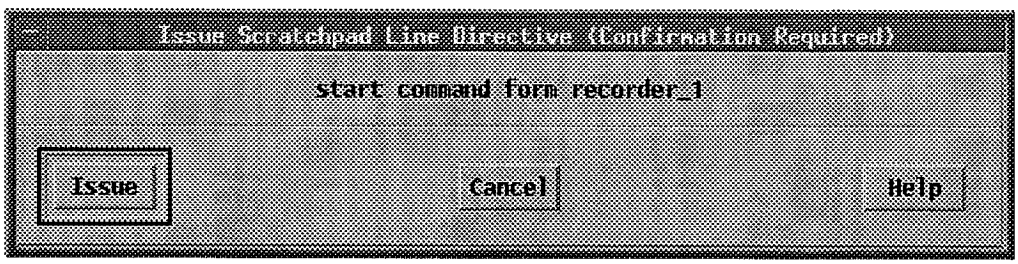

Figure 6-3 Confirmation Dialog for SPL Directive

The confirmation is required because the user may have pressed the button accidentally and did not mean to issue the directive. The user presses the 'Issue' button on the dialog. The dialog box disappears and the Display Operation application issues the

directive. The user is finished with the display and selects 'Exit...' from the 'File' pulldown menu. A confirmation dialog box appears requesting confirmation before exiting. The user presses the 'Yes' button and the application terminates.

6.2 Scenario 2

The user initiates the Display Operation application which causes the main application window to appear on the monitor. The main window has a menu bar, a status bar, a display area, a control bar, and a message area. Since a display file has not been opened the display area is empty (see Figure 1-2).

The user wants to operate the display file entitled 'SS_View_Data' so he opens the desired file using the procedures described in Scenario 1. The status bar indicates that the current data mode is real-time, the data status characters are on, LES is turned on for all objects, and the time reference is GMT. The display is updating at a rate of once every five seconds.

One of the objects in the display is a scrolling time plot that is plotting temperature data from one of the instruments on the spacecraft. Three minutes after the display was opened, the user notices that the line representing the temperature on the plot object has risen drastically (see Figure 6-4). The color of the line on the plot has changed to the color red to indicate that the value of the MSID has gone into a warning range (in this case - warning high).

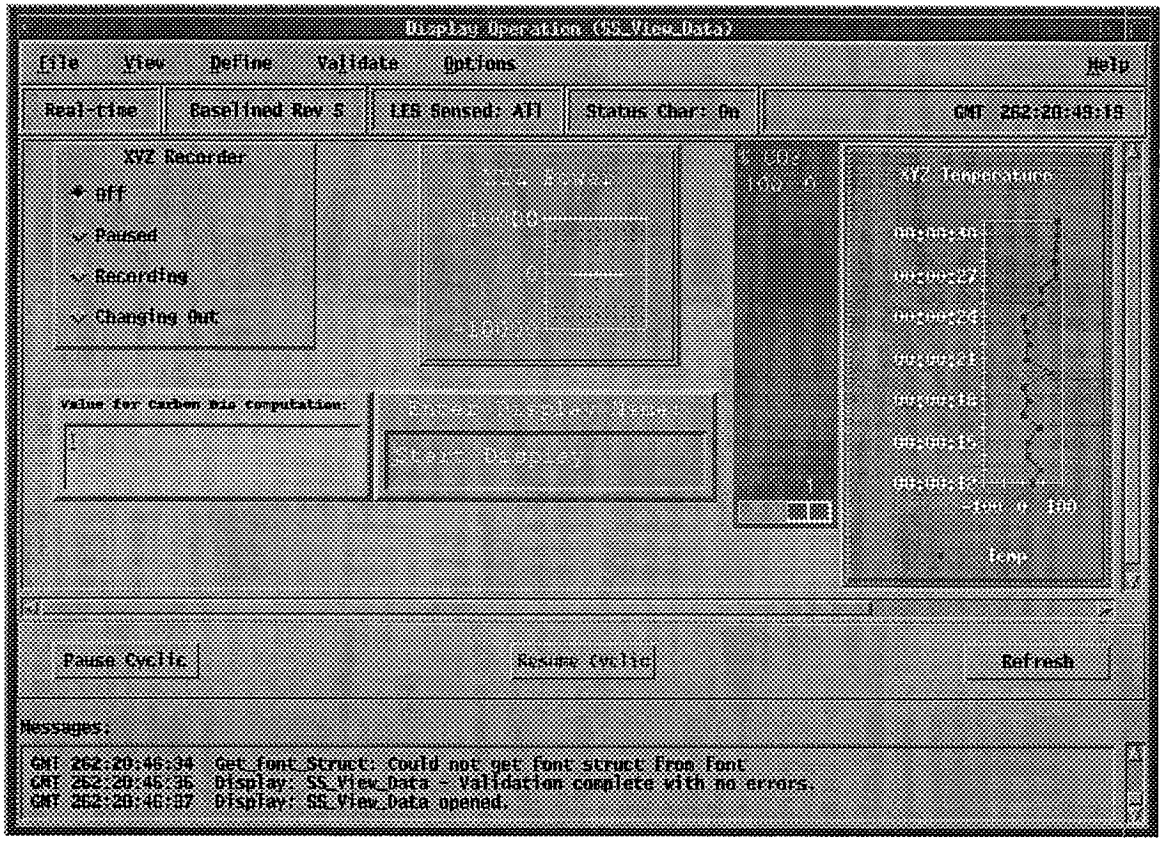

Figure 6-4 Display 'SS_View_Data' Opened in DO

At this point the user wants to change the time reference on the status bar to Mission Elapsed Time (MET). The user selects 'Set Time Reference...' from the 'View' pulldown menu. A dialog box appears that allows the user to select the desired time reference (GMT or MET) (see Figure 6-5).

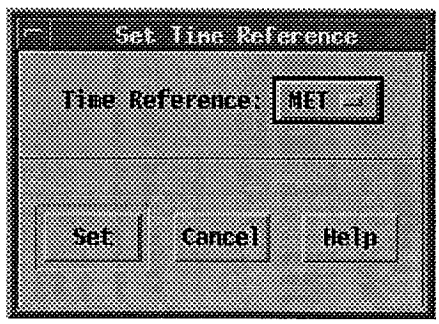

Figure 6-5 Set Time Reference Dialog

The user selects MET and presses the 'Set' button on the dialog box which causes the dialog box to disappear and the time reference to be changed to MET. A message appears in the message area stating that the time reference has been changed to MET. The user now needs to study the display without further updates so he presses the 'Pause Cyclic' button on the control panel (see Figure 6-6).

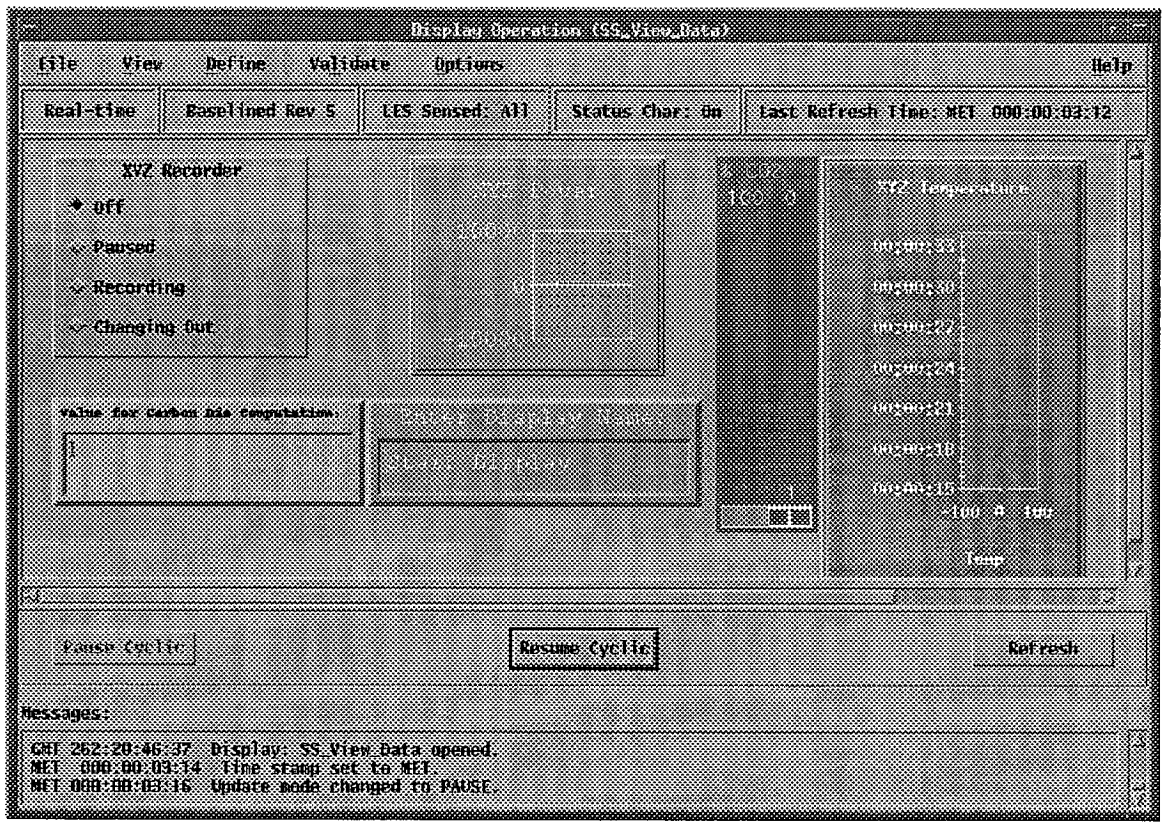

Figure 6-6 Paused Display

This pauses the updating of the display until the user selects either the 'Refresh' or 'Resume Cyclic' buttons. The user documents the information from the display and selects 'Print Window' from the 'File' pulldown menu to get a screen dump of the display.

The user then presses the 'Resume Cyclic' button on the control panel to continue cyclic updates of the display.

7. Conclusion

With the increasing popularity of windows based applications and graphical user interfaces (GUIs), the need for a more 'user friendly' and state of the art system for displaying telemetry data has become more evident at the Huntsville Operations Support Center at NASA's Marshall Space Flight Center. GUIs provide the user with an intuitive and easy way to use application tool that shelters the user from the more complicated underlying commands. The use of standards (e.g., OSF Motif, etc.) provide the user with an across the board common look and feel for every application running on the user's workstation. This provides familiarity with basic functions common to many applications and gives the user a higher level of confidence when learning a new application.

The Display Operation application provides the Enhanced HOSC System user with all of these features. The current system being operated at Marshall Space Flight Center's HOSC is a text based interface and an upgrade is overdue. State of the art technology to include graphics workstations and the X Windows System were used in the development of the Display Operation application, thus providing the answer to the demands of today's HOSC users.

8. References

[1] Asente, P. J., and R. R. Swick, "X Window System Toolkit", Digital Press, 1990.

[2] Flanagan, D. (editor), " X Toolkit Intrinsics Reference Manual", O'Reilly and Associates, Inc., 1992.

[3] Heller, D., "Motif Programming Manual", O'Reilly and Associates, Inc., 1991.

[4] HOSC-EHS-065 R1, Enhanced HOSC System Concepts and Scenarios Document, October, 1994.

[5] HOSC-EHS-076, Enhanced HOSC System Process Definition Document, August, 1994.

[6] HOSC-SRS-027 R1, Software Requirements Specification for the User Telemetry Applications of the EHS, 1994.

[7] MSFC-RQMT-1440B, Generic Requirements for the Enhanced HOSC System, January, 1994.

[8] MSFC-STD-1956, HOSC Common User Interface Standard, October 1994.

[9] Nye, A., "Xlib Programming Manual", O'Reilly and Associates, Inc., 1992.

[10] Nye, A., "Xlib Reference Manual", O'Reilly and Associates, Inc., 1992.

[11] Nye, A., and T. O'Reilly, "X Toolkit Intrinsics Programming Manual", O'Reilly and Associates, Inc., 1992.

[12] Pressman, R. S., "Software Engineering, A Practitioner's Approach", McGraw-Hill, Inc., 1992.

[13] Roetzheim, W. H., "Developing Software to Government Standards", Prentice-Hall, Inc., 1991.

[14] Scheifler, R. W. and J. Gettys, "X Window System", Digital Press, 1992.

[15] Young, D. A., "The X Window System, Programming and Applications with Xt", Prentice-Hall, Inc., 1994.

[16] Young, D. A., "Motif Debugging and Performance Tuning", Prentice-Hall, Inc., 1995.