THE MISSION EVENTS GRAPHIC GENERATOR SOFTWARE: A SMALL TOOL WITH BIG RESULTS

Mark Lupisella, Jack Leibee, Charles Scaffidi

NASA/Goddard Space Flight Center
Greenbelt, Maryland USA

ABSTRACT

Utilization of graphics has long been a useful methodology for many aspects of spacecraft operations. This paper presents a personal computer based software tool that implements straight-forward graphics and greatly enhances spacecraft operations. This unique software tool is the Mission Events Graphic Generator (MEGG) software which is used in support of the Hubble Space Telescope (HST) Project. MEGG reads the HST mission schedule and generates a graphical timeline.

Key Words: mission operations, mission schedule, graphics, automation

1. INTRODUCTION

The general purpose of this paper is to introduce the MEGG software at a conceptual level. If additional technical information is desired, the author would be happy to provide it.

Section 2 presents an overview of the HST Ground System and the MEGG software. Section 3 discusses the operational needs and design features incorporated in MEGG. Section 4 addresses lessons learned. Section 5 discusses future plans for the software. Section 6 summarizes. Section 7 provides references.

2. OVERVIEW

2.1 HST Ground System Overview

This paper begins with a high level introduction to the HST Ground System in order to provide a general context in which the MEGG software role can be understood.

The HST Ground System consists of science systems and control center systems. The science systems are located at the Space Telescope Science Institute (ST ScI) in Baltimore, Maryland. The control center systems are located at the Goddard Space Flight Center in Greenbelt, Maryland. Figure 2.1 shows a high level functional diagram of these systems. The primary process begins with a proposal being submitted by the scientific community to the ST ScI. Once approved, it is incorporated into a Science Mission Schedule (SMS), generated by the ST ScI, which is a detailed science schedule spanning one week.

The SMS is transmitted to the control center where it processed with orbit data and Tracking and Data Relay Satellite System (TDRSS) schedule data to generate the mission schedule, TDRSS file, and command loads for one week. The command loads are sent to the on-line operations system and then uplinked to the spacecraft via the NASA Communications (NASCOM) Space Network.

On the return link, the spacecraft sends science and engineering telemetry to the on-line operations system. Astrometry and engineering data is processed and sent to the ST ScI. The spacecraft also returns 1 megabit science data directly to a dedicated data capture facility where it is then sent to the ST ScI. The ST ScI processes and archives the data for use by the scientific community.

2.2 MEGG Overview

2.2.1 Operational Function

MEGG's primary operational function is to translate the HST mission schedule file and the TDRSS schedule file into a graphic timeline representing the spacecraft activities for one week. Both files are ASCI reports. In general, the mission schedule file is 8,000 pages long and takes 8 megabytes of disk space. The TDRSS file is usually 1.2 megabytes in size. MEGG is an

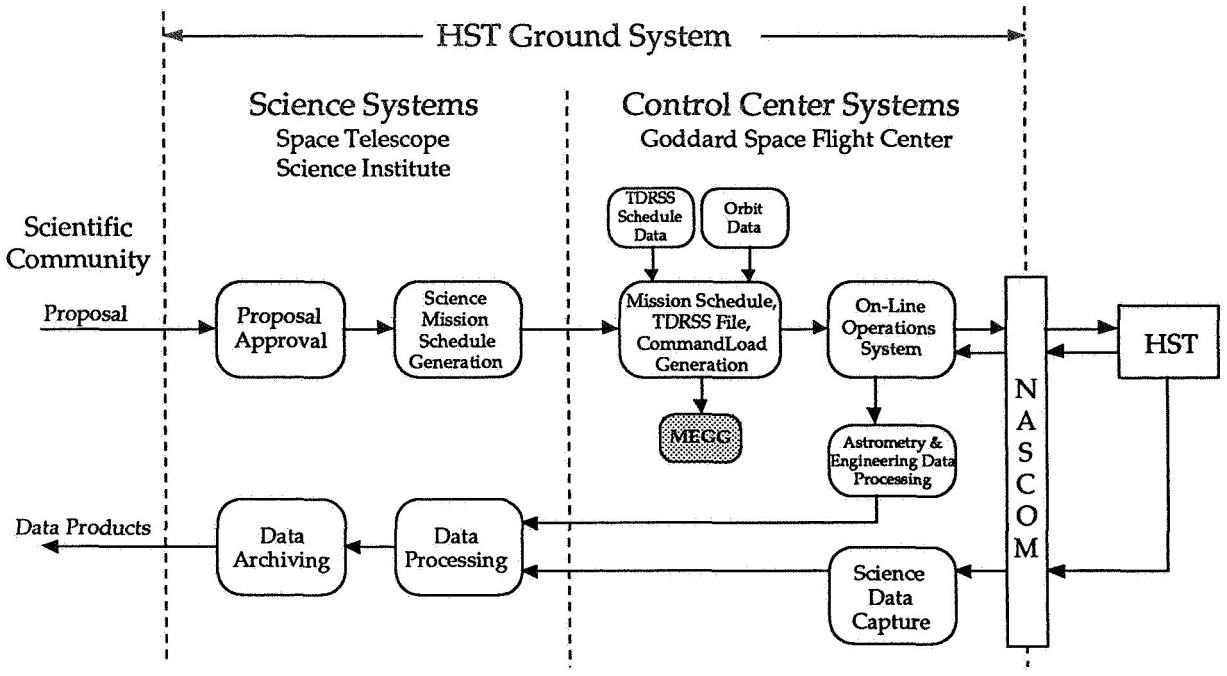

Figure 2.1: HST Ground System Functional Overview

automated system that generates a product used by the full spectrum of individuals ranging from operations engineers to scientists. Because of the sophisticated and complicated nature of HST, the need for this kind of tool emerged early in the pre-launch testing phase. Initially, the need was so great, the graphic timeline was generated manually. However, it became imperative to automate this system and increase its operational functionality. Computer Technology Associates (CTA) originally developed the software and it has since been enhanced and maintained as a NASA in-house software product.

2.2.2 Hardware and External Interfaces

Figure 2.2 shows the MEGG hardware components and its interface to the HST Ground System. As mentioned, the ST ScI generates the SMS and transmits it directly to one of the HST control center mainframe computers. The Flight Dynamics Facility sends HST orbit data to this mainframe and the Network Control Center, which is responsible for scheduling and operations of the TDRSS satellites, transmits the TDRSS schedule data to the mainframe. Here, command files, the mission schedule, and the TDRSS file are generated by the HST off-line support software. The command files and TDRSS file are used to generate the command load which is sent to the on-line operations system. The mission schedule and TDRSS file are sent to the MEGG system via an ethernet connection. With the mission schedule and TDRSS file both residing on the personal computer, MEGG is ready to be executed and produce a PostScript file which the printer converts into graphics. The final 85 page hard copy of the timeline can then be distributed accordingly.

2.2.3 Software Overview

MEGG uses three programming languages: C, AWK, and PostScript. The user interface and controlling code module is written in C. The user interface is menu driven. This module also controls the flow of the software during a run.

AWK is a pattern recognition language which closely resembles C but has its own interpreter. The code developed in AWK does most of the processing during a MEGG run. The modules written in AWK read the input files (the mission schedule file and TDRSS file) and create the

event list of recognized activities in these input files. The main modules of code written in AWK are the Spacecraft Support Module (SSM), the Science Instrument Module (SIM), and the TDRSS File Module (TFM). The SSM and SIM both read the mission schedule separately and generate their corresponding events for the event list. The TFM reads the TDRSS file and generates events for the same event list.

Additional AWK modules, along with the Postscript kernel module and graphic-defining module, also produces the final Postscript file which is sent to a laser printer. PostScript is an advanced graphics language used by some laser printers to produce sophisticated graphics.

3. OPERATIONAL NEEDS AND DESIGN FEATURES

3.1 Automation

As mentioned previously, graphic representations of mission schedules were generated manually during the pre-launch testing phase. It was soon recognized that the process needed to be automated in order to be incorporated as an efficient operational system.

The primary design features that accomplish automation are automatic input of ASCI report files, mentioned in Section 2.2.1, and the use of different programming languages for different tasks. The main tasks performed by MEGG are pattern recognition and graphics generation. The graphical user interface is written in C.

Pattern recognition is accomplished using the programming language, AWK. It closely resembles C but is easier to use for pattern recognition. AWK was designed primarily for high speed pattern recognition.

Graphics are generated using the PostScript graphics language. PostScript is versatile and compatible with most printers and allows for detailed graphic representation.

3.2 Wide Range of Applicability

A tool was needed that could be used by the full spectrum of individuals and serve a wide

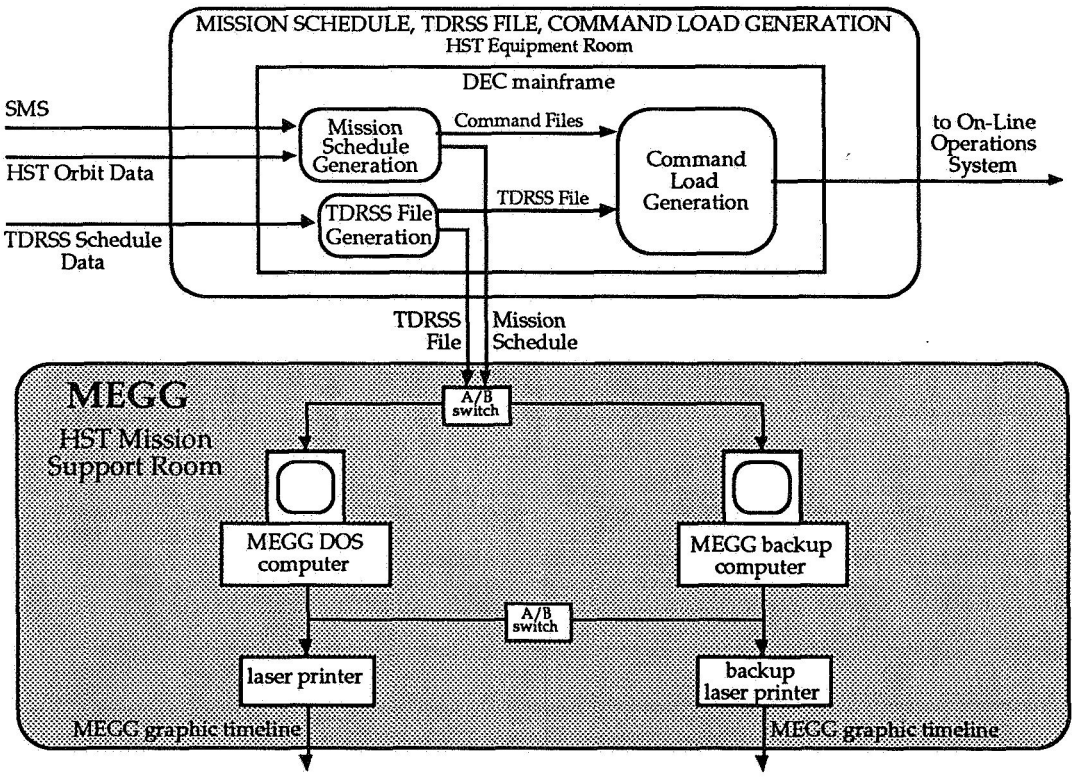

Figure 2.2: MEGG Hardware/Interface Overview

range of functionality. Scientists, engineers, console operators, and managers have to be able to reference the same source of schedule information in a simple and efficient manner. MEGG serves this purpose. It is used for schedule validation and review, real-time operations, and off-line analysis. The design features used to satisfy the need for a wide range of applicability are comprehensive and tailored graphics.

The MEGG timeline is comprehensive because it displays information about all the major elements of the spacecraft's operation. It has functionality for all users. Figure 3.1, the MEGG key, gives an idea of what the timeline looks like and shows the range of information covered by the timeline. A more typical page would have less information and would show correlations between different lines of the timeline.

In addition, the timeline's graphics are tailored to meet specific functional needs of people interested in the mission schedule. Again, this can be seen in Figure 3.1. This allows for schedule review of specific events, visual access to the schedule for console engineers, and off-line analysis for specific events and time periods. Efficient schedule review is facilitated by grouping related information together to provide the most efficient visual representation. In addition, MEGG has often alerted engineers to potential difficulties in the schedule. These difficulties can then be resolved before they become a problem for on-line operations. Using the MEGG output, console engineers are able to more closely follow operations events associated with the spacecraft, resulting in a better overall awareness. MEGG also serves as a valuable off-line analysis tool since it can be referenced to understand what specific events were happening at specific times such as during anomalies.

3.3 Simple Interactive Use

It was desirable to design software that was not difficult to learn or use. This allows for efficient operations and training.

The user interface was designed to be straight-forward and easy to use. It is menu driven and has on-line help. If the operator makes an error, notification and an explanation is provided. Options for execution of the software are limited to reduce possibility of operator error.

Also, MEGG is personal computer based and therefore is inherently easier to use and understand compared to the usual mainframe computer or workstation.

3.4 Ease of Enhancement and Maintenance

It was desirable to have a software tool that was very easy to enhance and maintain. This would allow for timely deliveries when needed.

The primary design feature that allows for easy enhancement and maintenance is the use of modular coding. There are eight modules of developed code in the MEGG software. Conceptually, this makes the code easy to understand and hence modify. Also, modular coding allows for efficient troubleshooting since problem areas can easily be isolated.

Again, the fact that the software runs on a personal computer makes it easy to maintain and develop enhancements. A personal computer can be used as an isolated system, thereby relieving resource conflicts. Also, personal computers are relatively inexpensive making transfer of the software to other groups less of a financial impact.

4. LESSONS LEARNED

4.1 Replace AWK With C

It is likely that a better choice of language for the pattern recognition processing is C. While AWK specializes in high speed pattern recognition, and was a good choice for that reason, C could do the same job and offers more versatility and easier portability. It is a language that most programmers are familiar with. In addition, using C would alleviate the need for using and obtaining AWK interpreters for different operating systems. This is important when considering implementation into other systems. The cost effectiveness and performance impacts of this change require further investigation.

4.2 Perform One Read Pass Of Mission Schedule

MEGG presently takes two passes to read the mission schedule. The first pass reads for

engineering (health and safety) events, and the second pass reads for science instrument events. This is useful for development and conceptual understanding of the software, however, it is CPU intensive. This could be reduced if only one pass was executed to read for all events. This would be an easy modification and would reduce the complexity of the software as well as the overall run-time.

5. FUTURE PLANS

5.1 Implement Lessons Learned

The AWK subroutines will be re-coded in C for increased versatility and portability. In addition, the code will be restructured to make only one read pass of the mission schedule.

5.2 Servicing Mission

In late 1993, NASA will launch a mission to service the HST. A new Wide Field Planetary Camera (WFPC II) and the corrective optics (COSTAR) will be installed. Additional servicing tasks will be executed but only the above mentioned affect the MEGG software.

The MEGG software needs to be modified to incorporate the two new instruments. New command structures for WFPC II and relevant operational states for COSTAR need to be investigated and implemented so that all relevant information is accurately represented.

5.3 Incorporate MEGG Into Other Systems

The system wide uses for the MEGG software are numerous. The HST Flight Software group has expressed interest in implementing the software into their system. Also, MEGG would be very useful in the HST trend analysis system since engineering information could readily be referenced against a comprehensive visual schedule of spacecraft events. The Telemetry Analysis for Operation Support (TALOS), a supplement system to the core operations software, has incorporated the MEGG display into its video graphics.

Ultimately, it is desirable to incorporate MEGG into the core operational software allowing console engineers instant access to a chosen aspect of the graphical representation of the mission schedule. This could ultimately lead to artificial intelligent based systems which alert operators to various events and required procedures.

6. SUMMARY

MEGG is a software tool that greatly enhances HST operations by efficiently and graphically representing the mission schedule. It is a small tool that provides an important function.

The operational needs of automation, wide range of applicability, simple interactive use, and easy maintenance are satisfied by hardware components and software design.

Future plans for MEGG involve implementing lessons learned, preparing for the HST servicing mission, and incorporation into other systems.

7. REFERENCES

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Final Report*, prepared for NASA/Goddard Space Flight Center

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Developer's Guide*, prepared for NASA/Goddard Space Flight Center

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Sourcecode Handbook*, prepared for NASA/Goddard Space Flight Center

Lupisella, M. (1992), *Hubble Space Telescope Mission Events Graphic Generator Requirements Document*, NASA/GSFC Mission Operations Division, Control Center Systems Branch

NASA/Goddard Space Flight Center, Mission Operations Flight Software Systems Branch and Control Center Systems Branch, (1992), *Hubble Space Telescope Mission Events Graphic Generator User's Guide*