Design and performance test of spacecraft test and operation software

Guohua Wang*, Yan Cui, Shuo Wang, Xiaofeng Meng

School of Instrumentation Science & Opto-electronics Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100191, China

Abstract

Main test processor (MTP) software is the key element of Electrical Ground Support Equipment (EGSE) for spacecraft test and operation used in the Chinese Academy of Space Technology (CAST) for years without innovation. With the increasing demand for a more efficient and agile MTP software, the new MTP software was developed. It adopts layered and plug-in based software architecture, whose core runtime server provides message queue management, share memory management and process management services and forms the framework for a configurable and open architecture system. To investigate the MTP software's performance, the test case of network response time, test sequence management capability and data-processing capability was introduced in detail. Test results show that the MTP software is common and has higher performance than the legacy one.

1. Introduction

Since the first artificial satellite was launched in 1970, China has designed, built, operated or contributed to more than one hundred spacecrafts and space missions, and become an emerging key player in space. Especially in recent years, with the increasing demand in telecommunication, navigation, meteorology, earth observation, etc., a dozen of satellites was sent to space every year.

However, the space environment is harsh and each space mission is a great challenge. Many spacecrafts have failed before accomplishing their mission although they are using the more recent technologies [1]. To make sure that the spacecraft works as planned, every single element of the vehicle as well as the completely assembled spacecraft itself must be tested on the ground under conditions simulating those it will face in space. This type of test is very extensive and the test period may vary from months to years, thus, there is a call for automated test equipment that helps to shorten test duration. The equipment that supports spacecraft ground test is electrical ground support equipment (EGSE).

Spacecraft test and operation software is the core of EGSE. However, China's space industry started relatively late compared with the space powers. When developing the first artificial satellite, Dong Fang Hong I (DFH-1), the operation and parameters monitoring of ground test facilities were manual. Although those facilities were low level, they ensured the successful launch of DFH-1 and Shi Jian 1 (SJ-1), and trained the first generation spacecraft test engineers [2]. In order to introduce the international advanced experience, in the early of 1990s, an Overall Checkout Equipment (OCOE) was procured through competition, on the international market to be used in the assembly, integration and test (AIT) program of the China–Brazil earth resources satellite program [3], and was also successfully used for DFH-3 ground test [2]. However, the spacecraft test and operation software running on the OCOE naming European Test Operation Language (ETOL), provide only executable files without detailed design documents, runs on VMS platform and is...
difficult to transport and improve for other satellites [4]. So in 1995, the Chinese Academy of Space Technology (CAST) initiated the spacecraft test and operation software research program, and developed the software naming master test processor (MTP) [5]. This is China’s first self-developed spacecraft test and operation software and widely used in China’s spacecrafts test for years. Unfortunately, over the years, little research has been carried out to improve the MTP. It failed to consider the generality and transportability; thus, it runs on Unix system only and extensive of work should be done when adopted for testing a new spacecraft. Moreover, its performance is never tested.

This paper attempts to develop new MTP software and investigates its performance. The paper starts with a brief overview of EGSE in Section 2. The detailed MTP software architecture is discussed in Section 3 followed by test case in Section 4. Section 5 presents the test result and its analysis.

2. Overview of EGSE

EGSE is the supporting equipment for spacecraft electrical test, whose structure has experienced the evolution from distributed system to centralized system and to hierarchical system. In the distributed system, there are almost no electrical interfaces and information exchange among special checkout equipments (SCOE). Each SCOE executes test dependently and test results are summarized manually after all SCOEs finish their tests. This structure is very popular when computers are not widely used. The centralized system uses standard industry interface, such as general purpose interface bus (GPIB) and VME bus extension for instrumentation (VXI), to integrate all test equipments in one box and is connected to the spacecraft through interface adapter. Computer is used to conduct the test procedures. However, when the spacecraft becomes more complex, the test equipment integration and software development become more and more complicated. Thus, this structure is only feasible for small satellite. The hierarchical system that is widely adopted in modern EGSE is a two level architecture, field level and remote level (see Fig. 1). The field level, i.e. the SCOE, has direct electrical interface with the spacecraft and executes specified test under the control of OCOE. While the remote level, i.e. the OCOE, schedules the SCOEs to test, processes and judges the parameters, provides human–machine interface for test control and parameters monitoring. OCOE and SCOE are connected together through local area network and form a distributed network system. This architecture enables the OCOE general equipment for any spacecraft, and the SCOEs can be reused in the unit level, subsystem level and AIT test.

2.1. SCOE

The SCOE is usually provided for each subsystem of a satellite and can be operable under remote control from OCOE by receiving control instructions as well as in standalone mode via a local control computer user interface. SCOEs’ size, composition and functions are different according to the

![Diagram of EGSE](image-url)
spacecraft; its types are broken down into the following subsystems: power supply subsystem SCOE, attitude and orbit control subsystem SCOE, on-board data handling SCOE, telemetry, tracking and command SCOE, payload module SCOE, etc.

2.2. **OCOE**

OCOE is the remote level of the EGSE, which is a general equipment and can be connected to the SCOEs through local network or fiber-optic cable. Thus, the distance between OCOE and SCOEs is unlimited and the SCOEs can be placed near the spacecraft while the OCOE maybe in the control center far from the launch site. The functions of OCOE include test preparation, test process management, data processing and monitoring, data archiving and replaying, real-time test conducting and graphical parameters display. The equipments of OCOE include MTP, TM front end equipment (TMFEE), TC front end equipment (TCFEE), test conductor console (TCC), graphic display terminal and standard peripheral equipment. In some systems, it is also configured with satellite simulators, SCOE simulators, data servers, etc., for user training, test process debugging and verification.

MTP, which runs the spacecraft test and operation software, is the center of the OCOE. During test, MTP executes test scrip to conduct tests automatically, receives and transmits information between TCC, graphic display terminal, FEE and SCOE, receives and processes telemetry data and SCOE data, archives and replays test process data, logs for errors and debugging. TCC, the client and operation interface of MTP, may vary from 1 to 99, which provides user interface for operator to interfere with the test process by sending telecommand and test sequence execution command. TCFEE, which directly connects to the spacecraft or through TCC&SCOE, controls the spacecraft by receiving the telecommand code from MTP and transmitting to the spacecraft after conversion and modulation. TMFEE, the interface for OCOE to acquire the on-board data, receives the telemetry signal from the spacecraft, and sends to MTP after demodulation and synchronization. Some TMFEE can simulate the spacecraft signal for FEE and MTP testings. Graphic display terminal monitors the spacecraft by collecting and graphically displaying the engineering values broadcast by MTP.

2.3. **MTP**

MTP is the spacecraft test and operation computer system that controls all of the activities of the other EGSE equipment during the AIT and validation campaigns for a spacecraft. It is the central and most crucial component to guarantee the success of a spacecraft mission through comprehensive, traceable and reliable testing. Its main functions include:

2.3.1. **TCC, FEE and SCOE management**

MTP manages up to 99 TCCs, starts a process for each TCC, FEE and SCOE, receives request command from TCCs, sends setting command or telecommand to FEE and SCOE and receives acknowledged message, status data or telemetry data.

2.3.2. **Data processing**

MTP receives telemetry data from TMFEE and test data from SCOES and justifies whether the parameters are within the excepted range after data processing. The data-processing methods are defined in the parameters configuration file with a specified format. The most popular data-processing methods are lookup table, curve fitting and formulation.

2.3.3. **Telecommand management**

Telecommand will directly change the state of spacecraft, thus telecommand management is one of the most important functions of MTP. MTP may authorize, enable, forbid some telecommands or automatically send defined commands in case of emergency.

2.3.4. **Data archiving and replay**

MTP archives all the telemetry data, SCOE test data, keyboard command, telecommand, etc., with time stamp for off-line replay and data analysis. Data replay is a good remedy to avoid some oversight during test especially when the test process is not repeatable.

2.3.5. **Test sequence management**

Spacecraft test procedure can be conducted manually by the operator through typing instructions via a command-line interface or clicking the candidate command from the telecommand list on TCC. However, owing to the importance of repeatability of the test activities, the normal mode of operation for the MTP is via a pre-defined list of control instructions and telecommands, called the "test sequence" or "control file", which can be automatically interpreted and executed by the MTP. The test sequence number can be run, and at the same time it is one of the important characters of MTP.

As mentioned above, the MTP software shoulders heavy missions. Its performances such as expandability, network communication capability, test sequence management capability and data-processing capability have a great impact on spacecraft test efficiency. The world famous MTP software includes EPOCH 2000 of Integral System Company [6], ETOL system and SCOS-2000 of European space agency [7], which leads the trend that MTP is common for test control and flight control [8]. ETOL supports up to 16 test sequence processes [2], SCOS-2000's throughput is more than 100 Mbit/s and can process 100,000 parameters [9] and CAST's traditional MTP can process 2500 parameters every second [10]. In order to improve the MTP's performance, new MTP software is developed, its performance test case and test result are described as follow.

3. **MTP software design**

According to the function requirements, MTP was designed as server software. To meet the real-time and transportability requirement, the new MTP software was realized on a Linux operation system, adopted share memory technology to share data among processes and message queue for process communications. To meet the generality and expandability requirement, MTP adopted a plug-in...
based software architecture, which establishes a plug-in runtime framework to provide a comprehensive solution for MTP function tailoring and expanding. Thus, the MTP software has four levels: system manager, runtime server, function library and information level (see Fig. 2).

3.1. System manager

System manager is the human–machine interface that provides six menus to manage MTP, they are:

- Stop/Start MTP: Stop or start the runtime server and all the plug-ins.
- List MTP info: Show MTP’s process info, such as process name, plug-in name, process ID, plug-in state, etc.
- Load/Unload plug-in: Load or unload a specific plug-in module.
- List/Clear Message queue: Show the message queue or clear the message queue.
- Update Parameters table: Reload the parameter configuration file into the memory.
- Exit MTP: Exit MTP software.

3.2. Runtime server

Runtime server is the core of the MTP software, which provides message queue management, share memory management, etc.
management and process management services and forms the framework for a configurable and open architecture system. The specific functions for SCOE, FEE, TCC, etc., are realized in the plug-ins; each plug-in may be started as one or more instance processes with different names.

1. Message queue manager
Message queue is the main approach for processes communication. Message queue manager establishes a message queue for each process and named by the process name. Each process can be bound to more than one message queue and send message to them, while receive message from the homonymous message queue.
There are two modes for message queue communication: blocking and non-blocking. In blocking mode, the sending message call will be blocked until the message queue is free when the sending target queue is full. In non-blocking mode, the sending message call will return an error immediately when the sending target queue is full.

2. Share memory manager
Share memory is another important approach for large amount of data exchanging in real-time among processes. The share memory is divided into three areas: system state area, command area and test data area. The system states area stores the process state information, message queue and share memory configuration information. The command area stores the telecommand information. Test data area stores the telemetry data from TMFEE and test data from SCOE.

3. Process manager
Process manager initializes the plug-in, forks a new process, and then goes into a message loop by repeatedly calling the plug-in interface’s “get message”, “translate message” and “dispatch message” function. The plug-in execution process is shown as in Fig. 3. The “clear_app” function is called when the process receives the exit message.

All the plug-ins realize the same interface defined in Fig. 4. The meanings of the interface members are shown in Table 1.

![Fig. 3. Plug-in execution process.](https://example.com)

![Fig. 4. Interface of plug-in.](https://example.com)

### Table 1

<table>
<thead>
<tr>
<th>Variables</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>handle</td>
<td>Handle of a plug-in instance</td>
</tr>
<tr>
<td>type</td>
<td>Satellite type</td>
</tr>
<tr>
<td>name</td>
<td>Process name</td>
</tr>
<tr>
<td>ID</td>
<td>Process ID</td>
</tr>
<tr>
<td>log</td>
<td>Pointer to the log information structure</td>
</tr>
<tr>
<td>cef</td>
<td>Pointer to the plug-in configuration information structure</td>
</tr>
<tr>
<td>runtime</td>
<td>Pointer to the runtime server</td>
</tr>
<tr>
<td>threads</td>
<td>Thread table of a plug-in</td>
</tr>
<tr>
<td>handlers</td>
<td>Message handler table of a plug-in</td>
</tr>
<tr>
<td>initialization</td>
<td>Initialize the plug-in</td>
</tr>
<tr>
<td>get_msg</td>
<td>Get message from message sequence</td>
</tr>
<tr>
<td>parse_msg</td>
<td>Parse the message</td>
</tr>
<tr>
<td>dispatch_msg</td>
<td>Dispatch the message to the specific handlers or the message sequence</td>
</tr>
<tr>
<td>clear_app</td>
<td>Clear the plug-in instance and exit</td>
</tr>
</tbody>
</table>

others are stored as ASCII files. Archiving files and log files are generated during runtime, MTP configuration is used for MTP runtime, such as process information, share memory information and message queue information.

4. Performance of MTP

Because of the heavy mission during the spacecraft test, the MTP performance has great influence on spacecraft test effectiveness. According to the description above, the most important MTP performance parameters are network response time, test sequence execution capability and data-processing capability.

- Network Response time
  EGSE is a distributed system, the real time of command and test data transmission among TCC, FEE and SCOEs through MTP is very important. To verify the network performance, the network response time must be tested.

- Test sequence management capability
  Test sequence is a set of commands to conduct a specific test process. The test sequence file is interpreted and executed by the test sequence process to realize automatic test. Since there are maybe up to 99 test conductors, more than one test sequence processes can be run at the same time. The maximum test sequence process number is another important MTP performance measure.

- Data-processing capability
  Data processing and interpretation is the core of spacecraft test. The data to be processed include telemetry data and SCOEs test data. Due to the mass data to be processed, the data process capability per second should be evaluated.

4.1. Test environment setting

To test the MTP performance, the test environment should be setup and should be as similar as the real spacecraft test environment. Our test facilities include MTP, graphic display terminal, satellite simulation software and multi-TCC simulation environment (see Fig. 5). To simulate up to 99 TCCs, we use LoadRunner software record the TCC operation process and generate multiple virtual users. Satellite simulation software is used to simulate the TCFEE, TMFEE and SCOEs, which receive commands from MTP and send simulated telemetry data and SCOEs' test data to MTP.

MTP is a computer with PIV 2.4 GHz CPU, 2 G memory, 320 G SATA hard disk and 100/100 Mbps Ethernet. MTP software was coded with C language, compiled by GCC and executed on Red Hat Enterprise Linux 4 AS.

4.2. Test case

To test the MTP performance, a series of test cases should be defined. A test case is a set of conditions or variables and inputs that is developed for a particular goal or objective to be achieved on a certain application to judge its capabilities or features [11]. We use the LoadRunner software as the test tool to simulate the multi-TCC load and monitor the MTP’s performance parameters. The test process with the LoadRunner is as follow: Planning the test, creating visual user (Vuser) scripts, creating the scenario, running the scenario, monitoring the scenario and analyzing the test result (see Fig. 6).

4.2.1. Network response time

Network response time is the interval between command sending and acknowledgement, which can be classified as the time between TCC and MTP, TCC and FEE, TCC and SCOEs. Their test methods are similar and the test case of network response time between TCC and MTP is as follow:

- Creating Vuser
  Start the TCC software, send network connection request to MTP and record this process by the LoadRunner.

Replace the TCC name in the script with a parameter to avoid runtime error when simulated multi-TCC. Insert a transaction to embody the section from TCC sending command to MTP response, and the execution time of the transaction will be the network response time.

- Creating the scenario
  Create a new scenario, add the new script to the scenario and add 99 loads with different IP addresses with load generator.

- Test process
  Start MTP and run the scenario, record the network response time 100 times and calculate the mean value.

### 4.2.2. Test sequence management capability

For the normal operation of MTP, 15% CPU usage should be reserved for idle, which constrains the maximum test sequence process number. Thus, the test sequence capability test case is as follow:

- Creating Vuser
  Start the TCC software, establish network connection with MTP and then send a test sequence execution request to MTP. Record this process with the LoadRunner and replace the TCC name and test sequence name in the script with a parameter.

- Creating the scenario
  Create a new scenario, add the new script to the scenario and add 99 loads with different IP addresses with load generator.

- Test Process
  Start MTP, modify the scenario to gradually increase the TCC number one by one until the usage of CPU is about 85%, then the TCC number is the maximum test sequence capability.

### 4.2.3. Data-processing capability

Because the data generation speed of spacecraft simulation software is fixed, we use data replay to test MTP's data-processing capability. First get 10 min telemetry data archive with the speed of 128 parameters per second, replay this file and modify the replay speed until the replay time is fixed or drop of data frame. Then divide the parameter number by the replay time, we will get the data-processing capability.

### 4.3. Test result

Run the defined test case, record and analyse the result as follow.

The network response time from TCC to MTP, FEE and SCOE is shown as in Table 2.

From the table we can see that the response times from TCC to FEE to SCOE are similar; this is because their processes are the same. The response time of FEE and SCOE is much higher than that from MTP, this is because there are two network connection processes: from TCC to MPT and from MTP to FEE or SCOE. Moreover, the FEE and SCOE's time calibration process is also included. Furthermore, the command transmission from process to process by message sequence is also time consuming.

The result of test sequence management capability is shown as in Fig. 7.

From the figure we can see that with the current hardware configuration and about 15% CPU idle MTP can manage 25 test sequences at most.

The test result of data-processing capability is shown as in Fig. 8.

From the figure we can see that in current hardware configuration MTP can process 4736 parameters per second at most.

### Table 2

Test result of network response time.

<table>
<thead>
<tr>
<th></th>
<th>TCC to MTP</th>
<th>MTP</th>
<th>FEE</th>
<th>SCOE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response time</td>
<td>21 ms</td>
<td>0.873 s</td>
<td>0.874 s</td>
<td></td>
</tr>
</tbody>
</table>

![Diagram](image_url)  
**Fig. 7.** Test sequence management capability.
5. Conclusion

This paper presented an overview of the MTP software and developed a new one. The software architecture is described. In addition, to investigate the software performance, test cases are designed. Test results show that the MTP has a better performance than the legacy and can meet the requirement of spacecraft test and operation for spacecraft ground test.

Acknowledgements

This project is supported by the Fundamental Research Funds for the Central Universities (no. 201017068).

References