The Space Congress® Proceedings

1987 (24th) Space - The Challenge, The Commitment

Apr 1st, 8:00 AM

Space Station Ground Data Management System

Jan Heuser
Chief, Systems Integration Branch Engineering Development Kennedy Space Center- NASA

William Sloan
Systems Integration Branch Engineering Development Kennedy Space Center, NASA

SPACE STATION GROUND DATA MANAGEMENT SYSTEM

Jan Heuser
Chief, Systems Integration Branch
Engineering Development
Kennedy Space Center. NASA

William Sloan
Systems Integration Branch
Engineering Development
Kennedy Space Center, NASA

ABSTRACT

KSC is planning a Space Station Ground Data Management System (GDMS) for support of functional interface verification, integration and test of Space Station modules and elements. This computer system, planned for initial operational support in 1992, currently is entering a definition and prototyping stage. This paper provides an overview of the GDMS system concept. It synopsizes system functional capabilities, and discusses software and hardware architectural approaches currently under evaluation. It identifies programmatic constraints and their influence upon the concept, as well as specific technical issues planned for study or evaluation via prototyping.

INTRODUCTION

The Space Station is highly complex, composed of a variety of elements which will evolve both in number and scope with time. Space Station elements, spanning a number of work packages and customer communities, represent products of multiple NASA Centers, contractors and others. Interfacing elements, designed and developed within different work packages or targeted for launch on different missions, may meet for the first time at the launch site or on-orbit.

To insure a functional Space Station within reasonable cost requires interface verification on the ground, sometimes accomplished via use of simulation or emulation. GDMS is planned for support of those prelaunch through postlanding integration and test activities to be accomplished at the launch sites, KSC and VAFB. It will control and monitor testing of functional interfaces between different work package items, interfaces broken for shipping, interfaces between different launch package items, and interfaces between the launch package and the orbiter.

The GDMS, although targeted primarily for KSC use, will apply software development tools, onboard subsystem simulations and other Program wide capabilities developed elsewhere. Likewise, portions of GDMS software may be applicable for support of integration and test accomplished at a work package Center or contractor site. Where this is true, its use is encouraged.

Although GDMS is targeted for Space Station support, it performs many tasks common to any major realtime command and

control system designed for support of functional integration and test. Key drivers include adaptability and expandability of the hardware architectural and software functional designs for application as a generic checkout system (GCS). Once designed and developed, the GCS could be readily reproduced and augmented to support evolving KSC requirements in areas related to STS launch processing (LPS II), expendable vehicle support or partial payload checkout. The goal is achievement of significant downstream development and operational cost advantages through commonality.

REQUIREMENTS AND CONSTRAINTS

Initial GDMS user requirements have been documented and approved by the KSC Space Station Project. To meet currently baselined processing schedules requires simultaneous support of 8 independent tests coupled with flexible system reconfiguration. Tests may range in scope from standalone support of a single element with limited ground support equipment to a complex end-to-end integrated test. The degree of required flexibility significantly drives hardware quantity as well as software complexity. A large number of element interfaces are specified for simulation, with fidelity ranging from medium to high. Other performance requirements tend to remain undefined or mirror those of the current Shuttle Launch Processing System (LPS).

Requirements emphasize the user's ability to accomplish any combination of the following from his single workstation: test product definition, test execution or performance of post processing of data, and interaction with information management systems. The user interface must support extensive windowing and graphics. Although use of multi-vendor commercial products is emphasized, limited expert systems application is implied. Availability of AI as well as multi-language conventional programming toolsets for future user application, is required.

In addition to user requirements, GDMS design is driven by a number of programmatic standards or constraints. Most notably, Ada has been established as the standard programming language for Space Station funded software such as GDMS. System development is to be accomplished using the Software Support Environment (SSE), with management information capabilities augmented by the Technical Management Information System (TMIS). Unfortunately, detailed definition regarding the SSE, TMIS and related areas is unavailabe at this time, and may significantly impact GDMS software design when baselined.

SYSTEM CONCEPT AND DESIGN GOALS

In response to requirements and constraints, the GDMS is being designed as a highly flexible, distributed and modular system supporting rapid reconfiguration for varying test and

checkout activities. It makes extensive use of commercially available, off-the-shelf hardware, software and network components. This coupled with use of the higher order language Ada for non-commercial software development supports rapid and economical upgrades to the system as new technology is available.

User workstations may be logically configured, based upon user permission level, to support a range of capability from master control of all test activities to read-only permissions for a limited set of data. The goal is to provide, from the user's workstation, common access to capabilities for test definition and development, test execution and post processing of data. interaction with management tracking information. In contrast, this is provided by a variety of KSC systems, each with its own terminal type or offline interface, in today's STS processing environment.

In contrast to earlier KSC systems, particular emphasis is given to the user support environment, including the user interface. GDMS will include a modern user interface providing extensive windowing, Mac-type menus, icon and graphical support, systemwide help, and integration of retrieved test data with user defined graphics. It will provide tools tailored for the integration and test environment, which support a user's definition of display screens and test procedures from graphical and data base inputs rather than conventional code. Many of these techniques have been successfully demonstrated over the past 18 months as a part of KSC's User Support Environment (USE) and expert systems prototyping activities.

A key theme throughout derivation of GDMS concepts has been application of a generic approach wherever possible. Emphasis has been upon definition of a basic hardware architecture and software foundation adaptable to meet the changing technological needs of a long term Space Station as well as near term and future needs of related KSC projects such as LPS II. This influence is seen in the emphasis upon the user environment and toolsets; orientation toward a data base driven system; minimization of custom hardware, microcode and assembly language software; and use of commercial system software.

The challenge now is to properly define and design that generic foundation so that it remains adaptable but efficient and easy to use. It is particularly critical that the degree of complexity and constraint that a user faces be comensurate with the complexity and criticality of the function he is performing; ease of use must be objectively balanced against test integrity. A goal is to minimize the hardware configuration and procedural constraints that an engineer must face to define and execute a small standalone test; while providing more resources but also tighter configuration management and procedural constraints for engineers defining

or executing critical integrated tests. For even the most complex of tests, techniques are planned to greatly improve upon current KSC systems, particularly in the areas of system build and configuration management.

GDMS will be developed utilizing the toolset and standards of Space Station's Software Support Environment (SSE). In addition, KSC plans to utilize other software and hardware elements established as Program wide standards when they are identified; as a minimum this is expected to include software simulators and critical flight type hardware elements (eg., Standard Data Processor or SDP) for which "high fidelity" simulation is required to support launch site activities.

HARDWARE ARCHITECTURE

The hardware architecture currently envisioned for the GDMS consists of a distributed set of micro/miniprocessors networked together, with gateways to the outside. The hardware set is capable of being logically reconfigured to support a multitude of development and test activities. Depending upon the function to be performed, a user may require the minimum configuration of an intelligent display processor up to a maximum configuration of the entire set.

Functional hardware building blocks include the following: Data Acquisition Subsystem, Application Processor, Archival Retrieval Subsystem, Data Base Subsystem and Display Processor. Functionality of the building blocks, when supplemented with software, is briefly synopsized below. Note that the GDMS includes multiple blocks of each type, physically integrated by a commercially available network as shown in Figure 1. Where high fidelity simulation is required, these blocks are supplemented with "flight type" hardware acquired from the work package developers.

The Data Acquisition Subsystem provides the interface to ground support equipment (GSE) and elements under test. Configurable to emulate or communicate in a variety of standard formats, the Data Acquisition Subsystem acquires and preprocesses incoming data during test execution, performing tasks such as limit checking, data compression and linearization. In addition, it routes data to other GDMS elements and if network delays dictate, may perform time critical control logic procedures.

An Application Processor is one of a pool of processors used to execute user application programs and process user commands. This pool provides a multi-user timesharing capability for the GDMS. The processors are capable of being allocated singly or in groups to match processing power to test needs.

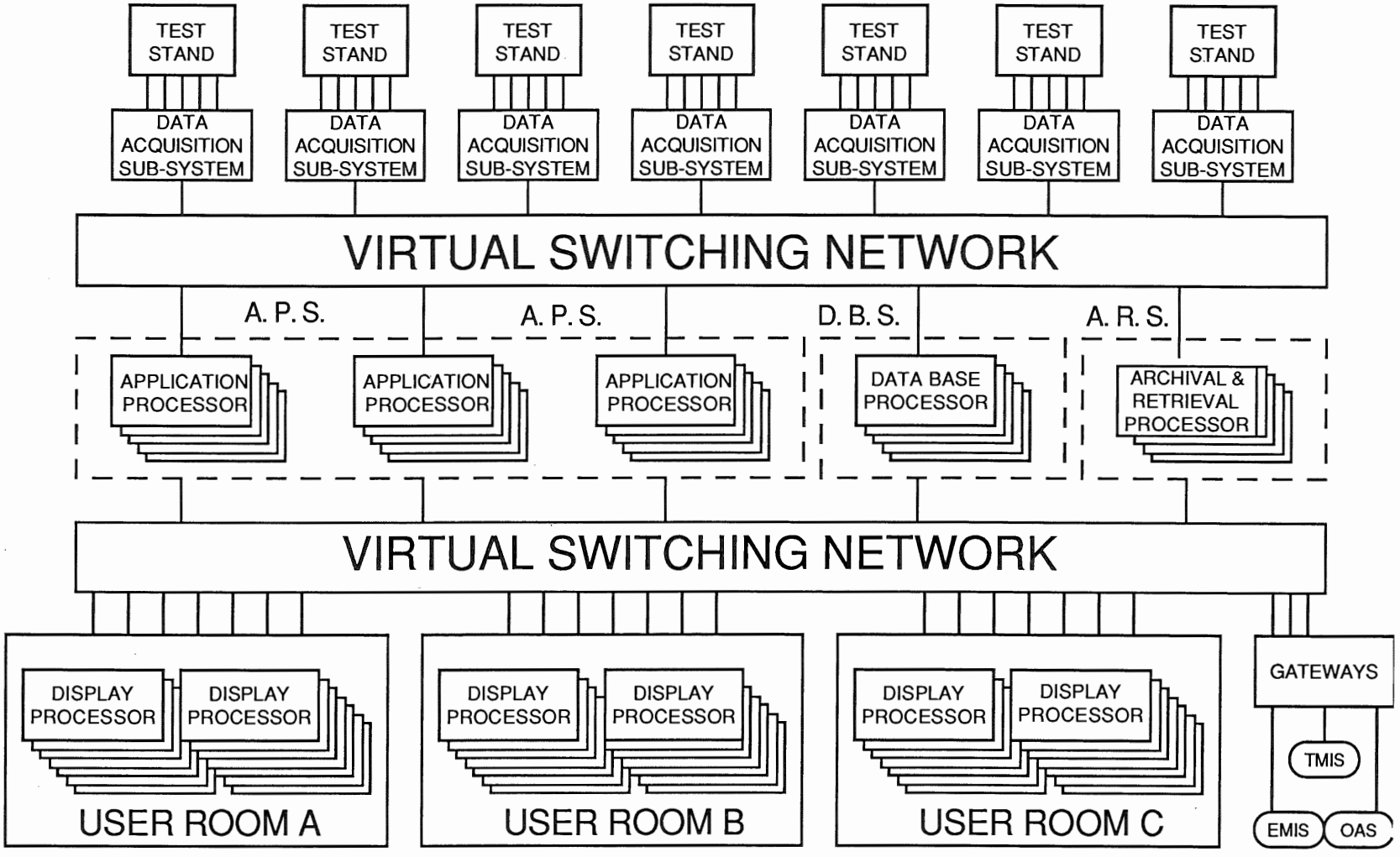

Figure 1. GDMS Hardware Architecture

The Archival and Retrieval Subsystem receives and records data forwarded from the Data Acquisition Subsystem, for near realtime or post test retrieval and processing. The Data Base Subsystem is optimized for performing database management functions as required by GDMS. It is the repository for system configuration information, test element definition data, test definition data, management tracking information and software applications which are data base intensive.

The Display Processor provides the physical interface between the user and the GDMS. It may range from a "dumb terminal" to an engineering workstation. Despite its degree of intelligence, all display processors share a common user interface as discussed earlier.

FUNCTIONAL SOFTWARE ARCHITECTURE

Software design goals include use of commercial off-the-shelf system software and tools with "no" modification to vendor code. Where functional capabilities require augmentation, an independent but interfacing module should be developed. Initial GDMS design will be based upon a largely conventional software approach using Ada, but with prudent application of expert systems. It will stress "generic" software solutions, applicable with data base entry changes to multiple projects. Current evaluation and prototyping activities are testing the validity of these goals and supporting identification of potential areas for application of new technologies such as expert systems.

The GDMS software architecture may be characterized by the following functional components: System Software, Test Definition and Development Software, Test Execution Software, and Management Information Software. Each area is briefly synopsized below.

System Software performs basic GDMS executive and support functions required regardless of the operating mode or required system functionality. It includes a commercial operating system, network operating system, data base management system and utility functions. These will be augmented as required for support of enhanced security, system configuration and redundancy management, system health and diagnostics, external operational interfacing, user interface and system help.

Test Definition Software consists of the tools and utilities required to support definition of test items and test configurations. These include the database and configuration management tools to define the test elements to the system for execution support, as well as to define the system configuration required for test support.

Test Development Software falls into the following two cate-

gories: Software supporting conventional programming by the GDMS user, and higher level tools supporting generation of display and test products with a minimum of conventional coding.

Commercial compilers and associated language environments will be provided for Ada and a variety of the most widely accepted higher order languages, including LISP and PROLOG toolsets for expert system development. In addition, basic utilities and standard libraries developed within GDMS will be made accessible by those languages. (Although Space Station funded projects are required to use Ada, the multi-language support is required to service non-Space Station funded payload users as well as accommodate more extensive use of commercially available non-Ada products.) Simulation capability will be available for user software validation.

To minimize required mission-unique programming, the GDMS emphasizes a rich environment of "generic" definition and development tools, oriented toward the products required for Integration and Test Support. Combining modern data base, graphics and expert systems techniques it will support generation of display skeletons (for control and monitor) and procedures, from graphical and data base screen inputs. Since production of these products currently represents 60%-70% of the manpower effort expended on mission unique programming for checkout, such tools should very positively impact the cost of future operations.

Test Execution Software is that realtime and support software required during the actual performance of a test, the software whose functions were synopsized within the hardware section of this paper. It includes the runtime support libraries and utilities to load and maintain data routing, limit and conversion tables; to perform command processing, interface simulation, data acquisition and preprocessing, exception monitoring and display update, data archival and near realtime retrieval. It supports execution of the user defined test products during a real or simulated test, and provides a command interpreter for direct user interaction with the test element or GSE.

Management Information Software consists of the software and data bases to support management of the integration and test activities for which GDMS is being build. Where such systems already exist at KSC or are planned as a part of TMIS or SSE, this includes only a gateway and augmentation to meet unique data base or reporting requirements. In other instances this includes design and implementation of a new software system. Functions include support for test requirements definition and tracking, project management, work control and scheduling, resource scheduling, configuration management and problem reporting.

FEASIBILITY ISSUES

KSC is currently in a GDMS system definition and prototyping phase. Activities are underway to evaluate GDMS concepts, determining which are feasible and which may require an alternative route, and then refining specifications based upon those conclusions. Within the Generic Checkout System (GCS) activities, currently funded largely by Shuttle, a generic test bed is being assembled which prototypes each of the functional hardware blocks previously discussed; this has a variety of hardware components, all of which share a Unix-like operating system. Within related Space Station funded activities, software prototyping of user support environment concepts similar to the ones discussed in this paper has been underway for several months. An evaluation of Ada also has been initiated.

Key prototyping and evaluation needs at this time lie in areas related to the combined use of a commercially provided operating system (such as Unix or realtime Unix), a commercially available network (such as Ethernet) and associated network operating system, and Ada. Can all these be combined and still meet the needs of a "realtime" command and control system? At this point in time, we have many goals that sound positive but remain to be proven technically feasible. We also are in the initial stages of formulating both GDMS user requirements and system concepts. As both Space Station standards and budgets mature, undoubtedly changes, if not compromises, will be made in both requirements and concepts.