# Development of Space Shuttle Telemetry Station Software

by Dr. David K. Mann
*United Space Alliance*

*This paper reports on the space shuttle telemetry ground station development project turnaround brought about through collaborative software development and sustaining engineering. The application of this new project and software development methodology to the development resulted in several positive effects over the standard development method. These include a reduction in the initial software development costs, a reduction in the software time to market, improved marketability of the software technology developed, improved product quality, improved maintainability, and technology transfer. This collaboration between the United Space Alliance (USA) telemetry station development team and APLabs Inc., resulted in software innovations in Frequency Modulation (FM) and Pulse Code Modulation (PCM) processing software, as well as station management software.*

Daniel Golden, NASA Administrator, in his strategic outlook for 1999 [1] provides a statement of strategic intent for the agency. In this statement he outlines a three-part mission in which "Technology Development and Transfer" is a cornerstone of the mission for the agency in 2000. This is consistent with the 1999 external assessment [2] in which the administration places priority on the promotion of "high technology for economic growth through effective partnerships." The prime Space Flight Operations Contract, SFOC (reference: Contract NAS 9-20000) awarded to United Space Alliance, section G-14 and G-15, requires the contractor to provide a portion of the contract funds to small business and to support the "Government's Technology Transfer Program." This new development method is an example of how USA is exploring new ways to increase the marketability of technology developed on the space program while decreasing Space Shuttle Program development and operating costs.

This new collaborative development method involves adopting a software application counterpart available in industry to provide baseline functionality and developing additional capabilities required for the upgrade or replacement system in collaboration with the software vendor. Applying this new development method more effectively leveraged technology and expertise available in industry to reduce initial software acquisition cost and time to market, while providing a superior product to space shuttle operations. Also, the technology developed is built on the state-of-the-art rather than reinventing the state-of-the-art, making the technology developed more valuable to industry and the American public.

The project defined a software development turnaround in terms of key project management metrics (i.e. cost, schedule, and technical). Turnaround was defined as a 50 percent reduction in anticipated software development labor and time to market. The improvement in the technical merit was a little harder to quantify. Two surveys were performed as part of a comparative analysis. Three key technical categories were defined and a turnaround was defined as a marked improvement in two of the three. The categories were marketability, software product quality, and maintainability. Attributes were defined within each category. The first survey was designed to determine the importance of the attributes within each category. The second survey was performed after software development was completed and scored the delivered product against the most likely outcome of continued development on the custom code. The results of these surveys were organized and presented in a Kepner Tregoe decision matrix for comparative analysis [3]. A marked improvement was defined as 100 percent improvement in absolute score within a category.

## Development Method Genesis and Vendor Selection Process

A detailed estimate to complete the project, assuming continued status quo software development was performed, provides a baseline for evaluating improved project performance after the change in direction. This estimate was based on a functional analysis of the custom code under development against the functional requirements for the upgrade.

Functionality was divided into three categories for this analysis: telemetric, station management, and data products and tools. Telemetric functionality was defined as the capabilities to acquire measurement data from the PCM or FM carriers, route this data to a location on the ground station, and capture the data in a file for production of data products. Station management functionality was defined as the capability to set up and manage data acquisition as well as monitor real-time elements status. Data product and tools were defined as the capability to process the raw data acquired into data products.

Figure 1 is a summary of these analyses. The entire pie in each category represents the functionality required for the upgrade in each category. The three slivers contained in the "Functionality Addressed in the Custom Software" regions represent capability outlined in the custom code. The two black slivers in this region represent the capabilities inherent in the first two releases of the custom code. The third shaded slice in this region represents outlined but not functional capabilities. The slices remaining outside the "Functionality Addressed in Custom Software" region represents functionality required by the Shuttle Telemetry Station that was not anticipated during the custom development to date. The analysis revealed that approximately one-third of the telemetric, three-quarters of the station management, and two-thirds of the data products and tools capabilities required new development assuming continued status quo software development. Project leadership and management derived an estimate to complete for the custom software. This estimate was based on a detailed knowledge of the functionality required (function point analysis), performance histories of the developers involved, and developers' estimates. It was estimated that completing the custom code would require an additional 20 man-years of productive software development effort over five years

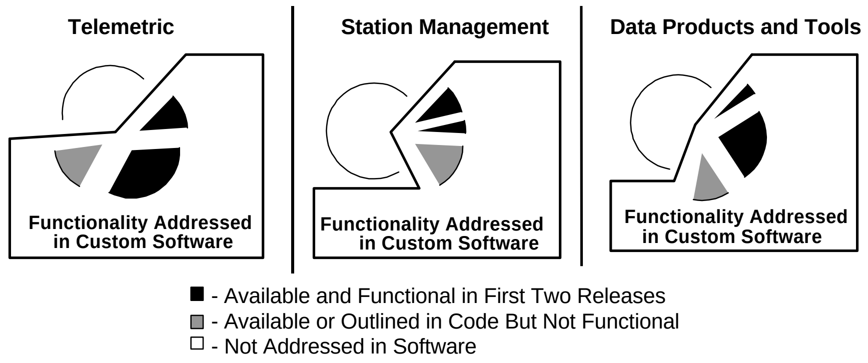


Figure 1. *Evaluation of the Custom Code Revealed Large Gaps in Functionality*

with a probable growth of 20 percent in manpower and schedule.

An industry survey, coupled with a series of product evaluations, was performed to determine if a commercial off-the-shelf (COTS) product that met the functional requirements for the upgrade was available. The survey revealed that no COTS package met the upgrade requirements. There were, however, several close functional counterparts—Veda Systems, Huerikon, Harris, Metraplex, APLabs Inc., Avtec and Acromag. Hardware for the upgrade project had been purchased and major change to the hardware architecture was cost prohibitive. As product and vendor investigations progressed, it became apparent that general-purpose telemetry software packages were coupled to particular hardware architectures. Of course, anyone can do a port, but the challenges associated with developing additional capabilities concurrent with a port to new hardware architecture was an approach deemed to have excessive risk. In addition, the vendor selected had to provide development and licensing flexibility, which will be discussed later. These factors lead to a single viable vendor and product combination. APLabs Inc. wrote VMEwindow to a hardware architecture similar to the one that was in storage for the upgrade and provided the flexibility to complete the project. A functionality assessment, similar to the one performed above, was done in order to scope the software development effort required and justifies the change in strategic direction to management.

## Collaborative Development

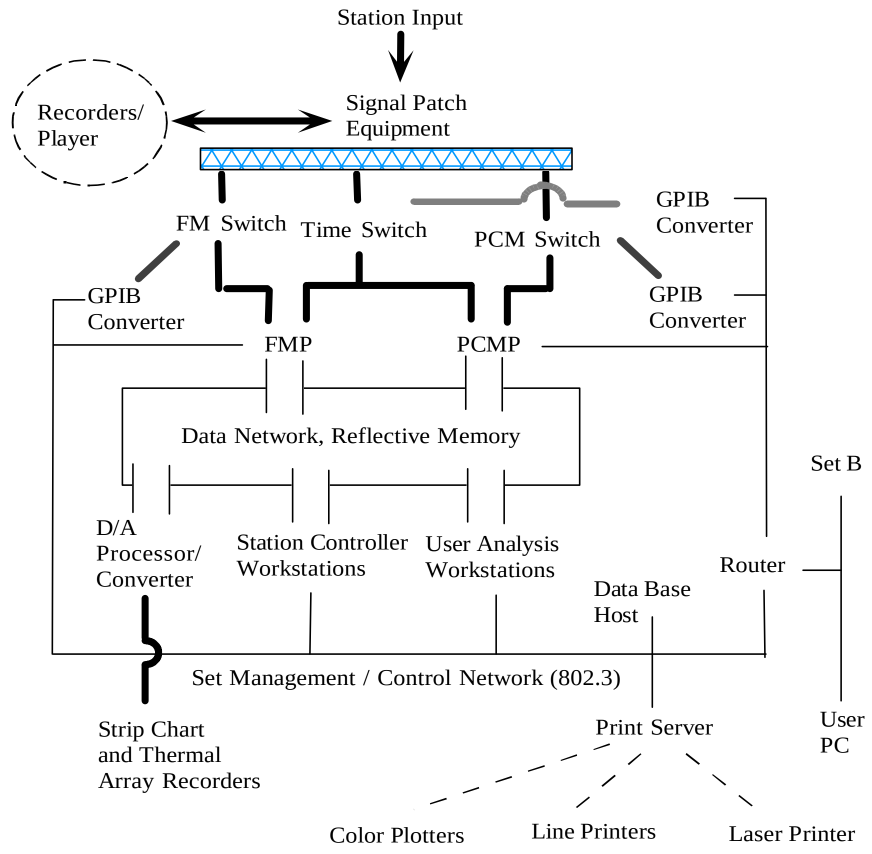
Collaborative software development started the second week of March 1998. The first order of business was to train USA developers in the VMEwindow development environment. Systems engineering was performed to identify the system level requirements and identify gaps in functionality between the baseline VMEwindow product and the required upgrade ground station. Although there were many minor modifications, the collaboration resulted in four software products, with innovators from USA and APLabs Inc.

These were submitted to NASA for Technology Transfer. Software development was completed the second week in November 1998. These software products represent modification to existing technology to improve and expand the capabilities of the baseline product. Figure 2 is a block diagram of the upgrade ground station architecture and is referenced in the following discussions on the software developed. The hardware architecture consists of Sun workstations connected to multiple PCM and FM processing VME chassis that use reflective memory as a real-time data transport mechanism. The chassis use Motorola processors and SBS Berg telemetry System cards. The software architecture consists of the VxWorks operating system, VMEwindow, Matlab, Dataviews, PVWave. The software products developed for the project and submitted to NASA for technology transfer are described below.

## Sbus SCRAMnet Interface for VMEwindow

A reflective memory network shown in the center of Figure 2 is the primary data transport vehicle for raw data from the PCM Processors (PCMPs) to the Digital to Analog Programmable Converters, User and Station Controller Workstations. The VMEwindow ground station telemetry package did not support a SBus Shared Common RAM Network (SCRAMnet) interface required to get data to the workstations. The challenge was to develop an application using the VMEwindow development environment that would provide the capability to

Figure 2. *Ground Station Overview*



acquire and display data at a single SBus workstation from multiple real-time telemetry processors in a deterministic manner with minimal latency. The code needed to be integrated within the VMEwindow environment, providing the operator with a consistent interface. It was decided that a derivative work could be produced from the code already available for interface to the VME version of the SCRAMnet card. It was negotiated that AP Data System would perform code modification and initial development testing and USA would provide the reflective memory topology, memory offset and data requirements in addition to concurrent code review, integration, and testing.

### Stream Definition Flat File Import Capability for VMEwindows

This software provides the capability to automatically load telemetry stream definition. This capability was seen as critical to the design because of the thousands of measurements that must be loaded to support each mission. Space Shuttle PCM downlink, data location, and unit information is contained in a database. Baseline VMEwindow provides a manual utility to load this information into a stream definition but did not provide the capability to automatically load stream definition. The software product developed provides the capability to create an ASCII flat file from a SQL database and import this flat file into the VMEwindows stream definition. This capability not only reduces time required to set up to support a mission, but also improves the reliability of the software load by cutting down on input errors. The user is prompted for parameters to define a PCM stream and accepts lists of measurement names read from an existing file and accesses the database to create an ASCII flat file. The generated file contains all of the information necessary to define PCM stream and populate the stream definition in VMEwindows.

### GPIB Board Setup and Control for FM Snapshots and Calibration

This software provides the capability to set up and control the NI1014 General Purpose Interface Bus (GPIB) board for specialized FM functions. Although a generic GPIB interface capability was available in the baseline VMEwindow environment, it did not meet the requirements of the project. The system uses a Metraplex digital discriminator and a Keithly switcher. The setup and control software allows individual control of both the discriminator and the switch and automated control of report generation. The setup functions were integrated into the ground station software as a newly developed VMEwindow icon. Board level control is provided using existing APLabs Inc. driver software. The software provides a calibration and FM Snapshot capabilities. A FM test signal that shifts the frequency over the bandwidth to represent five levels between -5V to 5V is input to the digital discriminator. The calibration software provides an average of five samples at each of these data levels. Once the set is calibrated, a snapshot can be produced providing the average, minimum, and maximum values for 100 samples of real-time data.

### Datel 622 Digital-to-Analog Driver, Setup, Control

This software product provides the ability to produce thermal array charts of data with loss of signal event indicators. The driver and control software provides the capability to set up and control the digital-to-analog conversion functions of the board. This capability was not available in the baseline VMEwindow environment. The setup functions were integrated into the groundstation software as a newly developed VMEwindow icon. The control software provides the capability to select data channels to be output, select event indicators for edge trace output, scale processing, and calibration functions.

### Collaborative Support, Development, and Licensing Agreement

There are several challenges facing successful application of a collaborative software development on a project in support of shuttle operations. Some challenges stem from the perception that the ability to support operations is somehow compromised. Others are from the notion that derivative or new software technology developed adjunct to a baseline application is not transferable to NASA and industry. Another argument in favor of in-house custom software development is that developers often are most qualified to troubleshoot and repair time-critical bugs during processing. Additionally, if the vendor was to go out of business or drop the product, the space program would run the risk of losing support for operational software. The project had to ensure that each of these issues was addressed and equal or superior capability would be available after the upgrade was complete.

These issues were addressed in several ways. First, in-house software developers were trained and certified as VMEwindow developers. This provided us the capability to collaborate on the development of required new capabilities and sustain the software after the upgrade was complete. Second, we negotiated a "Collaborative Support, Development and Licensing Agreement" with the vendor. This agreement has several clauses designed to address the concerns above. Rights to the source code for the purposes of Research and Professional Services' shuttle processing and technology transfer to NASA were secured in a "Project Buyout License" clause. This eliminated the need to escrow source code with a third party and enabled ad hoc modification of the baseline product during mission support without infringement concerns. Acquiring special user and software developer support, ensuring that immediate attention is paid to software issues during critical processing times in collaboration with in-house developers, addressed operational support concerns. A product upgrade cycle is defined in the agreement where new version software is provided, along with user documentation and installation assistance. This ensures that we have the ability to stay current while minimizing configuration management costs. The software development collaborations were so successful that special provisions were negotiated to ensure future endeavors would adopt a similar course. Subsequently, two new software innovations have been developed and will be written for technology transfer.

## Findings

Software development cost savings are derived by comparing the actual resources expended to the estimated resources required to complete the custom code, assuming continued status quo development. The cost associated with purchasing the

COTS software and development services is conservatively converted to equivalent manpower and added to the actual in-house labor expended to complete the software. The resulting figure is 6.6 man-years; when compared to the estimate to complete the custom code, the figure shows savings of 13.4 to 17.4 man-years. Converting this to equivalent dollars shows more than $800 thousand to $1 million savings.

The software development was complete the second week in November 1998. This meant that the software development took eight months rather than the five to six years estimated to complete the custom code. This represents an 86 to 89 percent reduction in the anticipated software development time to market.

A comparative analysis was performed, designed to determine the relative technical merit of the delivered software to the most probable product assuming continued work on the custom code. Expert engineering judgment is relied upon as the basis for this analysis through two surveys. The first column of Table 1 lists the categories defined to represent technical merit for the purposes of this analysis. Subindentures under each of the three categories (i.e. Marketability, Software Product Quality and Maintainability) are category attributes. All responses were normalized, averaged, and reported on a 1-to-10 scale where 10 was the most important or best. The second column represents the results of the first survey. This survey was designed to determine the relative importance of the attributes within a category from the perspective of the management. The third and fourth columns represent the results from the second survey.

The respondents scored the delivered software and the most probable outcome of continued status quo development. The additional costs associated with COTS software procurement and development was converted to equivalent manpower and the respondents were asked to estimate, assuming these additional resources were brought to bear on the custom software development. The fifth and sixth columns list the absolute scores for the custom and delivered product, respectively. These figures are totaled for each category to obtain an absolute relative score for each of the alternative methods. Comparing absolute scores reveals a three- to four-fold improvement in absolute score in every category.

Table 1. *Method Comparison Matrix*

| | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 1 | Average Normalized Weight | Average Probable Custom Product Score | Average Software Product Delivered Score | Absolute Score Probable Custom Product | Absolute Score of Software Delivered |
| Marketability | | | | | |
| Value to Current Customer | 7.78 | 1.67 | 9.33 | 13 | 73 |
| Value to Other NASA Projects | 4.70 | 2.50 | 8.50 | 12 | 40 |
| Value to Industry | 2.22 | 2.67 | 8.33 | 6 | 18 |
| Value to Future Shuttle Customers | 6.38 | 2.00 | 9.00 | 13 | 57 |
| Value to American Public | 3.92 | 4.17 | 6.83 | 16 | 27 |
| Total = | | | | 60 | 215 |
| Software Product Quality | | | | | |
| Usability | 5.07 | 2.67 | 8.33 | 14 | 42 |
| Reliability | 8.26 | 1.83 | 9.17 | 15 | 76 |
| Functionality and Versatility | 4.57 | 1.83 | 9.17 | 8 | 42 |
| Extensibility | 2.09 | 1.50 | 9.50 | 3 | 20 |
| Total = | | | | 40 | 180 |
| Maintainability | | | | | |
| Training Programs | 5.24 | 2.33 | 8.67 | 12 | 45 |
| Availability of Trained Personnel | 3.69 | 2.83 | 8.17 | 10 | 30 |
| Ability to Enhance Software | 5.55 | 1.67 | 9.33 | 9 | 52 |
| Documentation | 3.42 | 2.00 | 9.00 | 7 | 31 |
| Configuration Management | 7.09 | 3.58 | 7.42 | 25 | 53 |
| Total = | | | | 64 | 211 |

## Conclusions

Comparing the actual project performance and the projected performance, assuming continued development of the custom code, reveals that cost and schedule were reduced. The technical evaluation of the product delivered revealed a marked improvement in every attribute within all three categories evaluated. Several additional benefits included technology transfer, continued collaborative enhancement of the product, and the ability to combat obsolescence.

This methodology represents significant improvement over the status quo and should be evaluated for implementation on future and ongoing NASA software development projects. It is the author's considered opinion that the results warrant application of this methodology where close counterparts are available in industry. Strong close technical counterparts for data warehousing, recording, retrieval, command, control, data monitoring, network traffic generation, and system administration functions can be found in industry.◆

### Acknowledgments

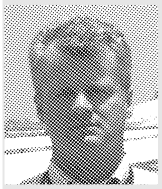
The author gratefully acknowledges Jonathan Morsics, Steven Prenger, Steve Gills, Andy Mason, Richard Price, Tom Perez, Earl Johnson, Bruce Chamberlain, the Integrated Data Systems Management Team and the Record and Playback Subsystem Upgrade development team.

This work was performed under the auspices of the Space Flight Operations Contract (SFOC, NAS 9-2000).

## References

1. Reference: http://hq.nasa.gov/office/nsp/outlook.htm

## About the Author



Dr. David Mann resides on Merritt Island, Fla. with his wife and two daughters. He works for United Space Alliance on space shuttle telemetry and computer systems as a project lead. He has 15 years of systems engineering, design, and project management expertise obtained on NASA and Department of Defense-related programs. Mann graduated with his engineering degree in 1985 and was recently awarded a doctorate in engineering management.

United Space Alliance
Launch Processing System Engineering
8550 Astronaut Blvd., Miss., USK-489
Cape Canaveral, Fla. 32920-4304
Voice: 407-861-7234
Fax: 407-861-7473
E-mail: David.k.mann@usago.ksc.nasa.gov