A Software Perspective on GNSS Receiver Integration and Operation

J. L. Goodman, United Space Alliance, 600 Gemini Avenue, Houston, Texas, 77058, USA

e-mail: john.l.goodman@usa-spaceops.com

Abstract

The GNSS industry is focusing on potential threats to satellite navigation integrity, such as intentional and unintentional interference, signal-in-space (satellite) and ground support infrastructure anomalies, shared spectrum issues, and multipath. The experience of the International Space Station (ISS) program, the Space Shuttle program, the Crew Return Vehicle (CRV) program and other users of GNSS indicate that navigation outages due to receiver software issues may pose as great a risk, if not more, to the user than threats currently under study.

The improvement in GNSS receiver tracking capability and navigation accuracy has been accompanied by an increase in software quantity and complexity. Current and future GNSS receivers will interface with multiple systems that will further increase software complexity. Rather than viewing GNSS receivers as "plug and play" devices, they should be regarded as complex computers that interface with other complex computers, sometimes in safety critical applications. The high cost of meeting strict software quality standards, and the proprietary nature of GNSS receiver software, makes it more difficult to ensure quality software for safety-critical applications. Lack of integrator and user insight into GNSS software complicates the integration and test process, leading to cost and schedule issues.

1. The Shuttle, ISS and CRV Experience With GPS

Since 1993, NASA has flown a number of GPS receivers on the Space Shuttle in support of scientific payloads and avionics development for the Space Shuttle [1,2], ISS [3] and CRV programs [3]. Since April of 2002, GPS has been operational on the ISS [3]. GPS receiver procurement, pre-flight testing, integration, numerous test flights in space and on aircraft, data analysis, issue resolution, interaction with the GNSS community (vendors, users and satellite operations personnel), and participation in navigation industry conferences has provided Johnson Space Center personnel with insight into the advantages and difficulties posed by GNSS technology [4].

2. Threats to GNSS Integrity

Risks to GNSS integrity include radio-frequency interference (intentional or unintentional), spoofing, ionospheric and solar effects, user errors, multipath, signal obscuration, antenna failure, failures in computers that interact with GNSS receivers, and malfunctioning GNSS satellites or ground

Satellite Navigation Systems: Policy, Commercial and Technical Interaction, International Space University, Strasbourg, France, May 26-28, 2003.

support systems [5]. The effects of on-board satellite equipment failures, or ground control segment induced anomalies, may affect users over a wide area. Spectrum sharing and the effects of new radio-frequency technologies, such as Ultra Wide Band, are important issues that must be resolved at the engineering, government policy and regulatory levels. The navigation industry and government agencies are actively researching the potential of GNSS integrity threats and building systems (such as WAAS, LAAS, EGNOS) and procedures to detect, identify and mitigate them [6]. Much research has been performed on receiver algorithms (such as, Receiver Autonomous Integrity Monitoring, or RAIM) to detect signal-in-space issues [7].

These are not hypothetical threats. On July 28, 2001, the failure of a rubidium clock on GPS satellite PRN 22 was detected by the WAAS, the U.S. Coast Guard and GPS receivers that were equipped with RAIM algorithms [8]. A well-publicized example of unintentional interference was noticed in Moss Landing Harbor, California, in April of 2001. Several months of investigation resulted in the identification of three boat mounted, active UHF/VHF television antennas with preamplifiers, that were the source of the interference [9].

3. Software Quality and GNSS Integrity

Software is at the heart of the GNSS revolution. Receivers, ground monitoring stations, augmentation systems, GNSS satellites and associated constellation ground support equipment are software intensive. The computational capacity and amount of code possessed by GNSS receivers is approaching that of flight management systems and flight control computers. Some current and many future GNSS receivers will interface with multiple systems: GPS, Galileo, GLONASS, WAAS, LAAS, EGNOS and various differential systems. Multiple system interfaces will further increase the software complexity of GNSS receivers. With this revolutionary navigation capability has come increased potential for performance anomalies due to software issues.

Receiver software problems can result in degraded navigation accuracy, or loss of navigation data for a variable amount of time. Most of the threats under examination by the navigation industry and government agencies are external to GNSS receivers. External monitoring, augmentation systems and receiver based RAIM will protect against software failures external to the GNSS receiver, but will not protect a user from a GNSS receiver software problem.

Receiver failures are not always well documented and may not be noticed if a device is not continuously monitored and data is not recorded and analyzed. If a problem is noticed, the reports often tend to be anecdotal in nature. It is difficult to pin down the cause of many GNSS receiver issues (i.e., intentional or unintentional radio-frequency interference, multipath, signal obscuration, antenna failure, hardware failure, receiver software failure, host computer hardware or software issues, operator error). Users are sometimes too quick to blame a receiver problem on a "bad" satellite or receiver software "bug," when the root cause is more likely a user error stemming from inadequate knowledge of receiver operation.

Engineers often underestimate the complexity of software, and overestimate the effectiveness of testing [10]. Tasks in GNSS receivers are started and stopped based on priorities, and the ability to track GNSS satellites. Logic path execution in a receiver is dependant on the radio-frequency environment, introducing an element of randomness into what code is executed, and when. A number of Shuttle receiver problems identified during a GPS receiver code audit [4] were deemed "non-credible" due to the number of conditions that had to occur within a tight timeframe. However, many of these issues later manifested in Shuttle flights, sometimes more than once. Receiver software modifications were made, proven in lab testing and during Shuttle missions, and the Shuttle GPS system was certified for operational use in August of 2002. GNSS satellite signal generators used in lab testing cannot duplicate the exact radio-frequency environment encountered outside the lab. Receiver software anomalies that manifest in flight may not manifest in ground testing. The Shuttle and ISS experience also indicates that changes to receiver software can result in subtle, unintended changes in receiver performance.

Software evolves and changes over time. Many vendors have a library of software modules, many of which are used in multiple applications. Software errors that manifest in a particular application may be deemed to have "no impact" to the user, and are not corrected. This causes software errors to propagate through succeeding product lines, with the potential for affecting future users in different applications. Changes in operating environment that come with a new application may invalidate assumptions made during initial requirements definition, and result in software issues during testing and operation. Software development schedules driven by "time to market" pressures and a desire to lower overhead costs (a small group of requirements developers and programmers, short development and test cycles) negatively effect software quality.

A recent study of stand-alone, TSO C-129 certified GPS receivers, performed in the United Kingdom, found that the probability of a receiver outage (loss of service) due to a software problem was higher than a signal-in-space problem that RAIM is designed to detect and isolate [11]. Data analyzed was collected during a total of 78,384.1 hours of receiver operation. The study concluded that more attention should be paid to characterizing GNSS receiver outage probability and outage modes. NASA's experience with the Shuttle, ISS, and CRV GPS receivers supports these findings.

4. Computer Integration Versus Plug and Play

Many applications of GNSS receivers involve interfacing with other computer systems. The concept of "plug and play" assumes that no software or hardware changes are required during integration. This is not a safe assumption for many applications. Different users may have different data and commanding requirements, and may not be able to economically change the rest of the system to conform to available receiver input and output. Hardware changes for items such as mounting or electrical power may also be required. Interactions of receiver software and software in other parts of a system cannot be assessed without testing and design insight, no matter how many applications already use the receiver in question. A "plug and play" integration assumption, which drives initial budget and schedule planning, can easily result in schedule slips and cost increases due to unanticipated technical problems at the component and system levels.

Successful integration of computers requires knowledge, not just of the interfaces, but also of how the data behind the interfaces are computed and behave under nominal and off-nominal conditions. An integrator may have to negotiate legal agreements concerning access to proprietary documentation with a vendor, so that information needed for integration is available. Integrator verification of interface documentation in a laboratory environment is prudent. System problems may result from the inappropriate interaction between parts of a system, such as computers, rather than individual units. The root causes, manifestations and impacts of system components that behave in a dysfunctional manner are difficult, and sometimes impossible to predict.

Two cases of this, involving GPS receivers, recently occurred on the Space Shuttle and ISS. During the STS-91 (June 1998) flight of Discovery, a GPS software problem interacted in an unanticipated manner with existing Shuttle flight computer software requirements and resulted in a loss of communication with Discovery for over an hour. A second example occurred on the ISS in September of 2002. A GPS software error not recognized in ground testing caused an ISS computer system to go into a diagnostic mode. Shuttle GPS receiver, Shuttle computer and ISS computer software changes were later made. Lab testing and flight experience proved that the changes resolved the software problems.

5. Process is Important

Process entails requirements definition, development of software, manufacture of hardware, verification, validation, integration, testing (at both the component and system level), data analysis, issue resolution, and certification by regulatory agencies. Problems in process can manifest as technical issues at the component or system level, user issues due to lack of understanding proper receiver or integrated system operation, schedule slips and cost overruns. Was the spurious active television antenna output that caused the recent GPS interference incident in California [9] the result of a process problem with design or manufacture? Process problems can delay the fielding of augmentation and improved constellation ground support systems that are needed to enhance navigation capabilities, and ensure integrity [12].

It is difficult to assess the quality of software requirements and code development processes when they are of a proprietary nature. Most software is not written from scratch, but is reused and modified for new applications. Development of legacy code may not have adhered to coding standards as it evolved, or been subjected to a robust process (requirements definition, code reviews, configuration management of code and supporting documentation, unit testing, lab testing of the receiver and integrated system, testing in the operational environment). Processes should be designed to focus resources on areas where the root cause of problems are frequently found. For example, many root causes of software issues occur during requirements definition and at hardware/software interfaces [13].

The requirements definition, verification and validation processes require special attention [14]. Most software problems can be traced back to flaws in specifying requirements, not to coding errors [10, 15]. Flaws in requirements specification result from incorrect assumptions about system operation or unanticipated aspects of the operating environment. Requirements should also specify how a component or system should act under off-nominal conditions.

Proper documentation of requirements, and requirements rationale, is important. Much software in use today is maintained by personnel who did not participate in the original development of the requirements and code [13]. Lack of documentation and corporate knowledge loss can pose technical, cost and schedule risk to projects that reuse existing software [16].

A review [10] of seven recent aerospace accidents identified sixteen common factors: 1) overconfidence and over reliance in digital automation; 2) not understanding the risks associated with software; 3) confusing reliability and safety; 4) over relying on redundancy; 5) assuming that risk decreases over time; 6) ignoring early warning signs; 7) inadequate cognitive engineering; 8) inadequate specifications; 9) flawed review process; 10) inadequate safety engineering; 11) violation of basic safety engineering practices in the digital parts of the system; 12) software reuse without appropriate safety analysis; 13) unnecessary complexity and software functions; 14) operational personnel not understanding automation; 15) test and simulation environments that do not match the original environment; and 16) deficiencies in safety-related information collection and use. Many of these factors may be classified as problems with the processes used to develop, integrate and use devices and systems that contain software, which includes GNSS receivers and associated support systems.

However, robust processes can be expensive [13]. A rigorous and successful process, such as that used for the Space Shuttle flight software, may not be economically viable for a vendor. Controversy over cost has surrounded the Federal Aviation Administration's DO-178B standards for avionics certification. Hiring and retaining skilled personnel to support these processes is challenging.

6. Navigation Conferences

Receiver reliability and robustness is seldom addressed at navigation conferences. Reports on new GNSS applications and integrations rarely highlight the more mundane process problems that were encountered when implementing and using GNSS technology. Problems with hardware and software procurement, software quality, lack of information on receiver design and operation, poor communication between project participants, poor vendor support, cost and schedule problems, test equipment issues, and inaccurate documentation are rarely mentioned in papers or keynote speeches, but are often discussed informally by technical personnel outside conference forums. Vendor feedback to the user community is often missing. The tendency to avoid mentioning problems makes it difficult, if not impossible, for GNSS integrators and users to learn how other users and integrators overcame process challenges to bring a project to a successful conclusion.

7. Meeting the Challenge

Conference agendas should include lessons learned sessions that facilitate discussion among users and integrators. Best practices cannot be identified and communicated if problems and their solutions are not discussed. There should be a willingness to discuss negative aspects of projects, including project failures, while sticking to the facts and not assigning blame to individuals or organizations. A "vendor feedback" session, featuring a panel of vendor representatives, may allow users and integrators to become aware of common problems observed by vendors from having worked with a large customer base. Vendor comments should not be limited to perceived obstacles to selling their products. Tutorials on process engineering (i.e., software development, testing, certification, device selection, integration, project management) as applied to the navigation industry may be helpful to conference participants.

Extensive testing of interim software versions and the integrated system should be performed, both in the lab and the actual operating environment. Computer interfaces, including GNSS receivers, in safety-critical systems should be "bullet proofed" to protect against known and postulated forms of spurious input. Redundancy is effective at mitigating the impact of random hardware failures, and dissimilar redundancy (such as using two GNSS receivers from different manufacturers) may protect against a common mode failure at the receiver level. However, the study of TSO C-129 certified GPS receivers performed in the United Kingdom also noted that all-in-view receivers might be more susceptible to common mode software failures than receivers that track a subset (such as a five channel receiver) of the visible satellites [11]. Redundancy, RAIM, augmentation systems, and ground monitoring systems will not protect against system level problems, receiver software problems, or user errors [10]. While it is a good idea to equip GNSS receivers with an autonomous reset feature ("ctrl-alt-delete"), the existence of such a feature should not be used as an excuse to take shortcuts in software and system development and testing. Widespread acceptance, confidence in and use of GNSS technology should not lull the integration and user community into thinking that problems cannot occur, particularly at the receiver level. Research and testing is needed to characterize receiver failures, and their probability of occurrence.

The process issues encountered by GNSS vendors, integrators, users, and certification authorities are not specific to the navigation industry. The same issues occur throughout the computer industry, and are the subject of research and discussion at computer industry conferences. Interaction between the computer software and the navigation industries may help overcome GNSS software process problems.

References

1. Goodman, J. L.: *Space Shuttle Navigation in the GPS Era*, Proceedings of the National Technical Meeting 2001, Institute of Navigation, Long Beach, CA, January 22-24, 2001, pages 709-724
2. Goodman, J. L.: Parallel Processing: GPS Augments TACAN in the Space Shuttle, *GPS World*, Volume 13, Number 10, October 2002
3. Gomez, S. F.: Flying High – GPS on the International Space Station and Crew Return Vehicle, *GPS World*, Volume 13, Number 6, pages 12-20, June 2002

4. Goodman, J. L.: *The Space Shuttle and GPS – A Safety-Critical Navigation Upgrade*, Springer-Verlag Lecture Notes in Computer Science Volume 2580: Proceedings of the 2nd International Conference on COTS-Based Software Systems, Ottawa, Canada, February 10-12, 2003
5. The Volpe National Transportation Systems Center : *Vulnerability Assessment of the U.S. Transportation Infrastructure Relying on the Global Positioning System*, Final report prepared for the Office of the Secretary of Transportation, August 29, 2001
6. Van Dyke, K., et al.: *GPS Integrity Failure Modes and Effects Analysis*, Institute of Navigation 2003 National Technical Meeting, Anaheim, CA, January 22-24, 2003
7. *RAIM: Requirements, Algorithms, and Performance, Global Positioning System: Papers Published in NAVIGATION*, Volume V, Institute of Navigation, Fairfax, VA, 1998
8. Langer, J. V.: *Near-Term Integrity Improvements for the GPS Operational Control Segment*, Proceedings of Institute of Navigation GPS 2002 Conference, Portland, OR, September 24-27, 2002
9. Clynch, J. R., et al: The Hunt for FRI, *GPS World*, Volume 14, Number 1, pages 16-22, January 2003
10. Leveson, N. G.: *The Role of Software in Recent Aerospace Accidents*, Proceedings of the 2001 International System Safety Conference, Huntsville, AL, September 10-15, 2001
11. Nisner, P. D., and Johannessen, R.: *Ten Million Data Points from TSO Approved Aviation GPS Receivers: Results of Analysis and Applications to Design and Use in Aviation*, Navigation: Journal of the Institute of Navigation, Vol. 47, No. 1, Spring 2000, pages 43-50
12. General Accounting Office: *National Airspace System: Persistent Problems in FAA's New Navigation System Highlight Need for Periodic Reevaluation*, Report to the Chairman, Subcommittee on Transportation, Committee on Appropriations, U.S. Senate, GAO/RCED/AIMD-00-130, June 2000
13. Bhansali, P. V.: Perspectives on Safety-Critical Software, Proceedings of the Australian Software Engineering Conference (ASWEC'97), Sydney, Australia, September 28 – October 3, 1997
14. Rosenberg, Dr. L. H., et al.: *Generating High Quality Requirements*, AIAA Paper 2001-4524, Proceedings of the AIAA Space 2001 Conference and Exposition. American Institute of Aeronautics and Astronautics, Reston, VA, August 28 to 30, 2001
15. Hayhurst, K. J., and Holloway, C. M.: *Challenges in Software Aspects of Aerospace Systems*, 26th Annual NASA Goddard Software Engineering Workshop, Greenbelt, MA, November 27 - 29, 2001
16. Divis, D. A.: Washington View – A New Year's Budget Resolutions, *GPS World*, Volume 14, Number 1, pages 10-12, January 2003