Software Requirements Management

Space Shuttle Lessons Learned #3377

1/27/2011

Hal Turner
NASA
Kennedy Space Center
NE-C1
Application and Simulation Software Engineering

Harold.H.Turner@nasa.gov
(321) 861-3789
Launch Processing System (LPS) Application Software is used in the Launch Control Center Firing Room to perform Vehicle & Ground Support Equipment (GSE) testing and launch of the Space Shuttle.

GOAL: Ground Operation Aerospace Language
Developed (1977) as a very high order, English-like language

Approximately 2500 programs, 2 Million lines of code (Firing Room/Launch Configuration)

> 1000 GOAL Displays, > 760 PCGOAL Displays

> 50,000 Command and Measurement parameters:
30,000 Vehicle / 20,000 GSE (Function Designators/FDs, Measurement & Stimuli IDs/MSIDs)
Firing Room 4 Layout
<table>
<thead>
<tr>
<th>ASWT</th>
<th>ASWT DESCRIPTION</th>
<th>ASSOCIATED CSCIs*</th>
</tr>
</thead>
<tbody>
<tr>
<td>APU</td>
<td>Orbiter/SRB Auxiliary Power Unit and SRB Hydraulics</td>
<td>APU, HPU/HYD</td>
</tr>
<tr>
<td>COMM/NAV</td>
<td>Communications and Navigation</td>
<td>COMM/NAV</td>
</tr>
<tr>
<td>DPS</td>
<td>Data Processing System</td>
<td>DPS, OS</td>
</tr>
<tr>
<td>DPSME</td>
<td>Space Shuttle Main Engine Controllers</td>
<td>DPSME</td>
</tr>
<tr>
<td>ECLSS</td>
<td>Environmental Control and Life Support System</td>
<td>ECLSS</td>
</tr>
<tr>
<td>ECS/PVD</td>
<td>Environment Control System and Purge, Vent and Drain</td>
<td>ECS/LPS, ECS/HIM</td>
</tr>
<tr>
<td>EPDC</td>
<td>Electrical Power Distribution and Control</td>
<td>EPDC</td>
</tr>
<tr>
<td>GLS</td>
<td>Ground Launch Sequencer</td>
<td>GLS</td>
</tr>
<tr>
<td>GNC</td>
<td>Guidance, Navigation and Control</td>
<td>GNC</td>
</tr>
<tr>
<td>HGDS/HWS</td>
<td>Hazardous Gas Detection System/Hazardous Warning System</td>
<td>HGDS/HWS</td>
</tr>
<tr>
<td>HYD</td>
<td>Orbiter Hydraulics</td>
<td>HYD</td>
</tr>
<tr>
<td>INST</td>
<td>Instrumentation</td>
<td>INST</td>
</tr>
<tr>
<td>INTG</td>
<td>Integration</td>
<td>INTG</td>
</tr>
<tr>
<td>LO2/LH2/MPS/SSME</td>
<td>Liquid Oxygen/Liquid Hydrogen/Main Propulsion System/Space Shuttle Main Engine</td>
<td>LO2, LH2, MPS/SSME</td>
</tr>
<tr>
<td>MASTER</td>
<td>Master</td>
<td>MASTER</td>
</tr>
<tr>
<td>MECH</td>
<td>Mechanisms</td>
<td>MECH</td>
</tr>
<tr>
<td>OMS/RCS</td>
<td>Orbital Maneuvering System/Reaction Control System</td>
<td>OMS/RCS, HMF</td>
</tr>
<tr>
<td>PAYLOAD</td>
<td>Shuttle Payload Operations</td>
<td>PLDTEST, RMS</td>
</tr>
<tr>
<td>PRSD/FC</td>
<td>Power Reactant Storage and Distribution – Fuel Cells</td>
<td>PRSD/FC</td>
</tr>
<tr>
<td>SRSS</td>
<td>Shuttle Range Safety System</td>
<td>SRSS</td>
</tr>
<tr>
<td>SWING ARMS</td>
<td>Swing Arms</td>
<td>ARMS</td>
</tr>
</tbody>
</table>
The ASWT is the primary point of contact and responsible for all aspects of Checkout, Control and Monitor Subsystem (CCMS) Application Software development.

Each ASWT is responsible for the development and maintenance of the Application Software required to support checkout and launch activities for the associated vehicle/ground support equipment (GSE) systems. Each ASWT can be responsible for one or more Computer Software Configuration Items (CSCI), which equates to an Application Software set.

* 21 ASWTs
• Shuttle/System Engineering
  – Determination of requirements needed for checkout and launch
  – Allocation of requirements to hardware and software implementations
  – Development of software requirements
  – Technical support to software engineering
  – Validation of application software

• Software Engineering
  – Assessment of requirements
  – Code development and verification
  – Maintenance of design documentation
  – Assist Shuttle/System engineering
These (ASWT) functions include:

A. **Requirements Identification**
   The team is responsible for identifying the software requirements necessary to implement all program directed changes, all Operational Maintenance Requirements Specification (OMRS) and Launch Commit Criteria (LCC) requirements and all operational enhancement changes necessary to improve the safety and efficiency of checkout and launch activities. The specifics of how to perform this identification are defined in USA004715 and USA004732.

B. **Application Software Development Planning**
   The team shall be responsible for the detailed planning of all software development activities related to the system. This includes maintaining a prioritized schedule of all mandatory and non-mandatory requirements that must be implemented. These plans and schedules shall be maintained by the team for tracking, statusing and presentation to management upon request.

C. **Application Software Development**
   The team is responsible for development, testing and verification of all Application Software for the associated vehicle/GSE system. The specifics of the development activities are defined in USA004715 and USA004732.

D. **Action Item Tracking**
   The team is responsible for tracking and responding to all external action items assigned (e.g., by TRP, CCB) and all internal action items (e.g., Shuttle Configuration Management System (SCMS) or Open Item Status Report (OISR) items within the allotted time frames. These action items shall be tracked in the ASWT minutes, with responsibility for the action assigned to specific individuals and projected completion dates defined. The team is responsible for implementing any tasks resulting from the assigned action items.

E. **Issue Resolution**
   The team is responsible for resolving all issues associated with the Application Software of all associated CSCIs. If an issue cannot be resolved at the ASWT level, the ASWT Lead and CSCI Lead may elevate the problem to their First Line Managers and/or the Application Software Technical Review Panel (TRP) for assistance in reaching resolution.

Refer to IDS-SEPG-058, LPS Defined Software Process, for the issue escalation process to use in these instances.
Requirements Processing and Control Boards

Data Change Request (DCR) → Shuttle Recon Coordination (SRC) Level II → Shuttle Data Control Board (SDCB) Level II → SASCB Pre-board → Shuttle Avionics Software Control Board (SASCB) Level II

Shuttle SW Change Request (SSCR), Discrepancy Report (DR), Flight SW Change Proposal (FCP), Software Authorization Notice (SAN), Initial-Load DCR (I-Load DCR)

Space Shuttle Main Engine Requirements Change Notice (SSME RCN), Engineering Change Proposal (ECP)

Requirements Change Notice (RCN) (with software changes)

Launch Commit Criteria (LCC) Change Notice (LCN)

Operations & Maintenance Requirements Specification (OMRS) Test Requirements Approval Process

Program Requirements Control Board (PRCB) Level II

Integrated Data Systems Configuration Control Board (IDS CCB) (Local KSC Board)

Engineering Support Request (ESR)

CSP Change Screening Panel

Shuttle Engine Software Review Group (SRG)
Requirements Processing and Control Boards

BOEING, H.B.
- Data Change (DC), DCR
- SSCR, FCP
- DR
- ECP, SAN, RCN, DCR’s

JSC
- NASA – Control boards (SASCB, PRCB)
- USA – Data RECON, Flight SW (FSW)

KSC Payloads
- Data Change (DC), DCR (STAR)
- SSCR, FCP
- DR

KSC Ground Ops
- DC, DCR (MAST)
- SSCR, FCP (FSW, Multi-function Electronic Display System (MEDS))
- DR (FSW)
- ECP, SAN, RCN (engine)
- RCN (Requirements)
- LCN (LCC)
- Master Change Request (MCR)
- PRCBD (directives)
- ESR (ground S/W)

Marshall Space Flight Center
- Shuttle Engine or engine controller related
  - ECP, SAN, RCN, DCR’s
**Software Engineering Life Cycle**

**PHASES:**

1. Project/Task Initiation
2. Requirements Analysis and Definition
3. Software Implementation
4. Acceptance Test
5. Release and Delivery

A succession of phases in a software development life cycle are formalized in a software process model.
Capability Maturity Model Integration® (CMMI) developed by Software Engineering Institute (SEI), Carnegie Mellon University

CMMI provides guidance for improving an organization’s processes and the ability to manage the development, acquisition, and maintenance of products or services.

United Space Alliance (USA) Integrated Data Systems
Space Flight Operations Contract (SFOC):
• CMM Level 3, Dec. 2004
  (Advent of LPS Software Requirements “Tagging” and “Tracking”)
• CMMI Level 3, Mar. 2006
• Backup
Abstract: The ability to manage and trace software requirements is critical to achieve success in any software project and to produce software products in a cost-effective and timely fashion. Conversely, incomplete, incorrect, or changing software requirements result in cost and schedule impacts that increase the later they occur (or are discovered) in the software life cycle. Current software technology, processes, and tools provide innovative, automated methods to facilitate optimum management of software requirements. Additionally, a collaborative relationship between the customer, or user, of the software and the developer of the software is essential to the success of the software project. That is, the user/customer requirements must be accurately communicated and understood to be correctly implemented in the software in order to meet end-users needs.

Description of Driving Event: The legacy (manual) methods for managing the implementation of software requirements for the Space Shuttle Program have had a major impact on cost and schedule over the entire software development life cycle.

Lesson(s) Learned: Manual methods for management of software requirements are ineffective and inefficient, contributing to excessive costs as well as schedule delays. Aspects of the management of software requirements include the elicitation/specification, analysis, development, tracking, and changing of software requirements used during the implementation and sustaining phases of the software life cycle. Management and traceability of software requirements are critical to the success of producing reliable, high-quality, and safe software products that meet end-users requirements and needs in a cost-effective and timely fashion. Cost and schedule impacts that result from incomplete, incorrect, or changing software requirements increase the later they occur in the software life cycle. Current software technology, processes, and tools provide innovative automated methods to facilitate optimum management of software requirements (e.g., IBM Rational DOORS, IBM Rational RequisitePro, Cradle requirements management software). Additionally, a collaborative relationship between the customer using the software and the developer providing the software is paramount to the success of the software project. More specifically, the users/customers must effectively define and accurately communicate their requirements to the developer. For example, the user’s defined requirements should be clearly stated and unambiguous, concise, complete, autonomous, able to be implemented, and testable.

Recommendation(s): Adopt and use current, state-of-the-art software processes and tools to manage requirements for software development. Value and foster the collaborative relationship between the customer who uses the software and the developer providing the software to ensure the success of the software project.
Launch Processing System (LPS)

- **VAB** (Vehicle Assembly Building)
- **RPS** (Record & Playback Subsystem)
- **OPF** (Orbiter Processing Facility)
- **CCMS** (Checkout, Control, & Monitor Subsystem)
- **RSI** (Real-time Simulation Interface)
- **KATS** (Kennedy Avionics Test Set)
- **LON** (LPS Operations Network)
- **SDC** (Shuttle Data Center)

**Key Components:**
- **Ground Data Bus**
- **Bus Monitor Data / PC-GOAL Link**
- **50K BPS Modem / TCD Updates**
- **9.6K BPS / Modem Link**
- **Raw Data**
- **Processed Data Recorder**
- **Shared Peripheral Area**
- **Strip Charts**
- **LPS Operations Network**
- **Orbiter / LPS Signal Adaptor**

**Linked Systems:**
- **CCMS**
- **RSI**
- **KATS**
- **LON**
- **SDC**
- **VAB**

**Important Notes:**
- PCGOAL Data (Shuttle Data Stream)
- PCGOAL Workstations
- Video & Data Assembly (V&DA)
- Hardware Interface Module (HIM)
- Shuttle Data Stream