Using the cFS Command and Data Dictionary (CCDD) to Automate Software Development on Habulous

Robert Hirsh
Robert.l.Hirsh@nasa.gov
NASA/Johnson Space Center
12/3/2018
• Habulous Background
• CCDD Overview

**CCDD Products used on Habulous**

• C header files that define all software bus commands/telemetry messages
• Generating file defining the Message ID’s used (cfs_msgids.h)
• XML Telemetry and Command Exchange (XTCE) files (displays)
• “Protobetter” code (to manage different endian-ness/architectures)

• Development on Habulous
  • CCSDS_v2 extended headers
  • Extending/customizing SBN to pass messages among computers on multiple networks
  • Using SBN_lib to allow non-cFS node to communicate with cFS nodes

• Next Steps
  • Developing TTE network and schedule tables for all the various CPUs to use
• The Habulous project is an Earth-based testbed (HW/SW)
  • Prototyping future space habitat unit and technologies
  • Representation from various NASA centers and aerospace organizations
    » ARC/JSC/GRC/Goddard/Stennis
  • Distributed nature of the team makes data interfaces especially critical
    » Massively heterogeneous computer architectures and operating systems
      – 32/64-bit, Big/Little Endian, Linux/VxWorks/Windows, x86/PPC/RaspberryPi
• Multiple CPUs use the SBN application to communicate
  » Most CPUs run cFS (use SBN app and Protobetter)
  » Non-cFS CPU (use SBN_lib with Prototbetter)
CCDD Background

• CCDD stands for cFS Command and Data Dictionary
• Goddard’s Core Flight System (cFS) has been, is, and is intended to be used by many projects
  • Examples: Lunar Reconnaissance Orbiter (LRO), Morpheus, Exploration EMU (xEMU) spacesuit, Orion Backup Flight Software (BFS)
  • Success of the cFS concept is shown by the number cFS projects at FSW-2018
• A command and data dictionary (CDD) defines telemetry/command messages
• Each cFS project must select a way to manage their CDD
  • Frequently involves using a spreadsheet, with custom SW to convert into useful files
• cFS Command and Data Dictionary utility (CCDD) was designed as a generic utility to eliminate duplication of effort in order to make CDD management easier
CCDD Goals

- Create a configurable CDD utility that runs on multiple operating systems
  - Written in Java for maximum portability
- Easy creation/modification of CDD information
  - Graphical user interface (GUI) to interact with the database
- Store all CDD information into a standard database (postgresql)
- Bidirectional transfer of information to/from the CCDD
  - Cut-n-paste to Excel, import/export via XTCE/CSV/JSON
- Easy access to CDD information (via scripting languages and web applications)
  - Allows user to code in various languages (ruby/python/js) and access CDD information
    » Create vehicle and ground software products, data summary, etc
    » Generate complicated CFS products: Schedule or network tables, copy table, etc
Data is accessible to scripting languages (JavaScript, Python, etc.)
- Example scripts provided for common products

**Imported/exported via:**
- CSV (comma-separated values)
- JSON (JavaScript Object Notation)
- XML (Extensible Markup Language)
  - EDS (Electronic Data Sheet)
  - XTCE (XML Telemetric and Command Exchange)
- OS clipboard (“cut & paste”)

Web-based dataserver (JSON)
### CCDD Demo

- **Project:** SampleProject
- **Server:** SampleProject
- **Date/Time:** 11/27/2018 12:42:42.376
- **Type:** Success
- **Message:**

  - Project 'SampleProject' locked

- **Index:** 6248
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:42:42.584
- **Type:** Success
- **Message:** Project 'SampleProject' unlocked

- **Index:** 6269
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:42:42.488
- **Type:** Success
- **Message:** Project database 'sampleproject' closed

- **Index:** 6270
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:42.489
- **Type:** Success
- **Message:** Connected to server as user

- **Index:** 6271
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:42.494
- **Type:** Success
- **Message:** Connected to project 'SampleProject' as user

- **Index:** 6272
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:42.494
- **Type:** Status
- **Message:** PostgreSQL: 8.4 *** JDBC: PostgreSQL 9.4.1207.jre7 (type 4)

- **Index:** 6273
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:42.022
- **Type:** Success
- **Message:** Server connection closed

- **Index:** 6278
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:45.060
- **Type:** Success
- **Message:** Connected to project 'SampleProject' as user

- **Index:** 6279
- **Server:** jsc-er-cfs01.jsc.nasa.gov
- **Date/Time:** 11/27/2018 12:43:45.061
- **Type:** Status
- **Message:** PostgreSQL: 8.4 *** JDBC: PostgreSQL 9.4.1207.jre7 (type 4)
• cfs_msgids.h file generation
  • Same file compiled by all CPUs
  • Defines all the MIDs for each cFS message sent/received on any of the various CPUs
• Using CCSDSv2, so each MID is a combination of APID/SystemID/SubSystemID
• Using the CCDD information to automatically generate the C-header files
  • Define the structure for all software bus (SB) commands/telemetry messages
• Generate XML Telemetry and Command Exchange (XTCE) files
  • Used by display team to make displays for any CPU
• Generating “Protobetter” code for communication with other CPUs
  • Manages packing and different endian-ness/architectures
Habulous MID definition

message ID

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>type</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>message type</td>
</tr>
</tbody>
</table>

Telemetry

2018 used by AES

Command

Proposed updates keep 32bit alignment

Telemetry

Command

PSR Template 04/2017
Major Habulous Activity in 2018

- Updating to CCSDS_v2 (and using CPU# as subsystem ID)
  - Running out of room for unique MIDs on all CPUs for the 11-bits of version 1
  - See next slide
- Exporting XTCE files to allow drag-n-drop display development for all CPUs
- Extending/customizing SBN to pass messages to computers
  - Computers with multiple interfaces act as a “bridge” to CPUs that can’t talk directly
  - “Protobetter” developed to manage packing/_endian differences
- Using SBN_lib to allow non-cFS node to communicate with cFS nodes
  - Allows non-cFS nodes to “impersonate” a cFS node and talk to SBN on other CPUs
- Worked to develop the CDD before the SW development was complete
  - Not treat CDD as an “as built” post-development documentation effort
  - Required iterations on data structures and MIDs, but minimized interface issues
• The CCDD tool has successfully been used to automate/autocode a large amount of software used on Habulous
• Working to allow the CCDD to define even more products including
  • Time-triggered Ethernet (TTE) network tables/maps
    » Coordinate message passing between various synchronized machines
  • cFS schedule table (for each CPU)
  • Automated CCDD to SysML export