Bridging the Gap: Web-Based SLE Solution for an Existing MDOS

Deane Sibol
Johns Hopkins University Applied Physics Laboratory (APL)

Josh Choi, Hayk Arutyunyan, Dave Santo, and George Rinker
NASA Jet Propulsion Laboratory (JPL)
California Institute of Technology (Caltech)

© 2017 California Institute of Technology. Government sponsorship acknowledged. Published by The Aerospace Corporation with permission.
AMMOS Mission Data Processing and Control System

• AMPCS is NASA Advanced Multi-Mission Operations System’s (AMMOS) telecommanding, telemetry processing, and user display ground data system
• Users: MSL*, SMAP, InSight, MarCO, INSPIRE, SWOT, NISAR, Europa, and EM-1 CubeSat Missions
• A major product re-architecture project started in FY17 (“Release 8”) and will include new interfaces and capabilities:
  – CCSDS Space Link Extension (SLE) Services
  – CCSDS File Delivery Protocol (CFDP)
  – Target Date: September 20, 2018
• AMMOS collaboration with Johns Hopkins University Applied Physics Laboratory (APL) started in 2015
  – Gained valuable SLE and CFDP domain knowledge and experience

* Using custom MPCS
SLE Needed Sooner

• Exploration Mission 1 (EM-1) CubeSat Missions
  – Lunar Flashlight
  – Lunar IceCube
  – Lunar Polar Hydrogen Mapper (LunaH-Map)
  – Near-Earth Asteroid Scout (NEA Scout)
  – Use NASA’s Deep Space Network (DSN)
  – Launch Date: September 30, 2018

• Requirement: ‘Use SLE’
• SLE capability needed to be delivered in Release 7 to achieve mission risk reduction needs
CCSDS SLE Services
## Choosing the Right Approach

<table>
<thead>
<tr>
<th>Approach</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrate SLE interface into AMPCS</td>
<td>No new application to deploy and manage</td>
<td>AMPCS R7.x’s uplink model conflicts with SLE FCLTU model</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Largest development and test effort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Most difficult to integrate into R8, if desired</td>
</tr>
<tr>
<td>Standalone Java application with built-in GUI</td>
<td>Simple</td>
<td>Difficult to achieve automation objectives</td>
</tr>
<tr>
<td></td>
<td>Matches application architecture and look-and-feel of existing AMPCS applications</td>
<td></td>
</tr>
<tr>
<td>Web application hosted on an Apache Tomcat server</td>
<td>Recently demonstrated by another AMMOS software and proven to work and matches security needs (by using AMMOS Common Access Manager) Enables both web user interface (WUI) and automation</td>
<td>Different UI technology than rest of AMPCS</td>
</tr>
</tbody>
</table>

Best: Use third-party SLE library
- LSE Space’s *SLE User Framework*
- Commercial product: Requires purchase of license
Deep Space Network (DSN), Near Earth Network (NEN), or Space Network (SN)

Concept – Legacy
Concept – SLE

SLE System

MDOS

AMPCS
SLE Proxy
(New)

Existing AMPCS Components

SLE
lib
Applications View

SLE System
- DSN SLE Return
  - All/Channel Frames (RAF/RCF) Provider
- DSN SLE
  - Forward CLTU Provider

Mission GDS (MDOS)
- Scripts
- Apache Tomcat
- chill_sle_proxy
- chill_sle_ui
- Common Access Manager
- TM Processor (chill_down)
- TC Applications (chill_up/send_* )
- Monitor and Query Apps

Transfer Frames
- CLTUs
- TCP: TF + ERT
- TCP: CLTU
- M&C
- Secures

M&C
chill_sle_proxy Features

- SLE service profiles management
  - Profile “dss34-raf”: Service instance ID “sagr=123.spack=ampcs…”, provider host “…”, etc.

- SLE service state management and data transfer operations
  - BIND, UNBIND, START, STOP, PEER-ABORT, THROW-EVENT, GET-PARAMETER, etc.

- Support forward direction and return direction SLE services simultaneously

- Uplink server for AMPCS TC applications

- Client application to the AMPCS TM application (chill_down)

- Support AMPCS TC and TM simultaneously

- Provide configuration and state information, log messages upon query

- RESTful application programming interface (API)

- Real-time monitor data via WebSocket
chill_sle_ui Features

- Platform independent graphical web UI
- Many instances can run simultaneously and receive same updates (benefit of WebSocket)
- Minimize data entry
  - Example: BIND, START, etc. operations are single button clicks
- Correlated information for both sides of interface
  - Can see all uplink parameters/state and all downlink parameters/state
- Provide feedback for SLE operations to same UI (versus legacy/other) that initiated those operations
- Under the hood: HTML5, JavaScript, CSS, Semantic UI, jQuery, DataTables, and WebSocket
chill_sle_ui
Monitor and Control
### Chill_sle_ui

Monitor and Control – Configuration Expanded

<table>
<thead>
<tr>
<th>Service Providers</th>
<th>AMPCS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return</strong></td>
<td></td>
</tr>
<tr>
<td>Profile Name:</td>
<td>uplink-prof</td>
</tr>
<tr>
<td>Host:</td>
<td></td>
</tr>
<tr>
<td>Port:</td>
<td></td>
</tr>
<tr>
<td>Instance ID:</td>
<td></td>
</tr>
<tr>
<td>Provider Auth Mode:</td>
<td></td>
</tr>
<tr>
<td>Provider Name:</td>
<td></td>
</tr>
<tr>
<td>User Auth Mode:</td>
<td></td>
</tr>
<tr>
<td>User Name:</td>
<td></td>
</tr>
<tr>
<td>Return Type:</td>
<td>RETURN_ALL</td>
</tr>
<tr>
<td>Start Time:</td>
<td></td>
</tr>
<tr>
<td>Stop Time:</td>
<td></td>
</tr>
<tr>
<td>Frame Quality:</td>
<td>ALL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AMPCS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Downlink</strong></td>
</tr>
<tr>
<td>Host:</td>
</tr>
<tr>
<td>Port:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Uplink</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Host:</td>
</tr>
<tr>
<td>Port:</td>
</tr>
</tbody>
</table>

[Edit Downlink Config]
# chill_sle_ui

Monitor and Control – State

<table>
<thead>
<tr>
<th>State</th>
<th>Return Provider</th>
<th>Forward Provider</th>
<th>AMPCS Downlink</th>
<th>AMPCS Uplink</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current State:</td>
<td>Active</td>
<td>Active</td>
<td>Connected</td>
<td>Enabled</td>
</tr>
<tr>
<td>Data Count:</td>
<td>691</td>
<td>115</td>
<td>691</td>
<td>115</td>
</tr>
<tr>
<td>Connection Number:</td>
<td>14</td>
<td>3</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
chill_sle_ui
Manage Service Providers

<table>
<thead>
<tr>
<th>Profile Name</th>
<th>Profile Type</th>
<th>Host</th>
<th>Port</th>
<th>Instance ID</th>
<th>Provider Auth Mode</th>
<th>Provider Name</th>
<th>User Auth Mode</th>
<th>User Name</th>
<th>Return Type</th>
<th>Start Time</th>
<th>Stop Time</th>
<th>Frame Version</th>
<th>Frame Quality</th>
<th>SC ID</th>
<th>VC ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>diste-test</td>
<td>RETURN</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_ALL</td>
<td>2016-285700:02:47:00</td>
<td>2016-294700:13:36:00</td>
<td>GOOD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>raf-off1</td>
<td>RETURN</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_ALL</td>
<td>2016-285700:01:36:00</td>
<td>2016-294721:46:36:445</td>
<td>GOOD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>raf-onic1</td>
<td>RETURN</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_ALL</td>
<td>2016-285700:01:36:00</td>
<td>2016-294721:46:36:445</td>
<td>ALL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rcfv2-onit2</td>
<td>RETURN</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_CHANNEL</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>test_profile_1</td>
<td>RETURN</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_CHANNEL</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uplink-prof</td>
<td>FORWARD</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>[redacted]</td>
<td>RETURN_CHANNEL</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Showing 1 to 6 of 6 entries
RESTful Monitor and Control

• Representational state transfer (REST) monitor and control API accommodates both WUI and automation scripts
• Using Restlet, open source REST framework for Java
• Example APIs:
  – GET /sle-proxy/sle-interface/profiles/dss34-raf
  – PUT /sle-proxy/sle-interface/profiles/dss34-fcltu + JSON body of new SLE service profile
  – POST /sle-proxy/chill-interface/config + JSON body 'downlink-host: chilldownhost1'
  – POST /sle-proxy/sle-interface/forward/action/bind?profile=dss34-fcltu
  – POST /sle-proxy/sle-interface/forward/action/start
  – POST /sle-proxy/sle-interface/forward/action/throw?change-rate=125&change-index=930
  – POST /sle-proxy/chill-interface/downlink/action/connect
  – GET /sle-proxy/messages?from-time=2017-065T12:00:00
RESTful Monitor and Control

- Representational state transfer (REST) monitor and control API accommodates both WUI and automation scripts
- Using Restlet, open source REST framework for Java
- Example APIs:
  - GET /sle-proxy/sle-interface/profiles/dss34-raf
  - POST /sle-proxy/chill-interface/config + JSON body 'downlink-host: chilldownhost1'
  - POST /sle-proxy/sle-interface/forward/action/bind?profile=dss34-fcltu
  - POST /sle-proxy/sle-interface/forward/action/start
  - POST /sle-proxy/chill-interface/downlink/action/connect
  - GET /sle-proxy/messages?from-time=2017-065T12:00:00
Automation

- REST APIs naturally become automation hooks

```python
#fcltu-test.py
...
data = json.loads(content)
...
```
Access Control

- AMMOS Common Access Manager (CAM) applies access control on web resources
- Control policies applied on Lightweight Directory Access Protocol (LDAP) users and groups
- Two LDAP groups: chill-sle-proxy.read and chill-sle-proxy.write
  - GET granted to “read” group
  - GET, PUT, POST, and DELETE granted to “write” group
- Gives SLE controllers and control scripts full access, while limiting monitoring users and non-control scripts to viewing only
Planning, Development, and Test

- Weekly WebEx meetings held between JPL and APL
- APL reviewed the requirements and design
- Two JPL engineers working half-time on development
  - Development started late-August 2016, finished on October 31 (10 weeks)
- One JPL test engineer coordinating testing
- Two APL engineers wrote test procedures and test scripts, now performing them
- Two testbeds:
  - LSE Space’s SSPsim (simulated data, our own VMs, outside of Flight Ops network)
  - DSN’s forward and return SLE providers in test environment (real MESSENGER data, providers are identical to production, inside Flight Ops network)
Many benefits to choosing a web-based approach to enable SLE services in an already established GDS
  – Recommend wider use of RESTful APIs and WUIs
  – Free frameworks are available that can significantly minimize development and test

Web apps don’t have to be limited to UI/client apps
  – Telemetry- and telecommand-handling, backend-like apps work well also

Provides early data points for AMPCS R8 design and planning
  – Use of commercially-licensed third-party products

Demonstrated the great synergy that is possible when different organizations working for NASA work together
Acknowledgments

Authors would like to thank Patricia Harrington-Duff\textsuperscript{1}, Arthur “Norm” Tuttle\textsuperscript{1}, Luis Campos\textsuperscript{2}, and Ashley Shamilian\textsuperscript{2} for their testing-related work on this task, which is crucial to the successful delivery of \textit{chill\_sle\_proxy} and \textit{chill\_sle\_ui}.

The work described was carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology (Caltech), under a contract with the National Aeronautics and Space Administration (NASA). The work was funded by the Multimission Ground System and Services (MGSS) Office.

\textsuperscript{1} APL; \textsuperscript{2} JPL/Caltech