Level 1 Processing Pipeline

Daniel Flath

For

Daniel Flath (Stanford),
Alex Schlessinger (SLAC)
Overview

- Requirements
  - Processing
  - Cataloguing
- How it works
- OPUS
  - Introduction & Desirable features
  - LAT-Specific Additions
- Implementation Status
  - OPUS
  - Add-Ons
- Goals
  - DC1
  - EM
Requirements

• Processing
  – Conversion of L0 (near-raw) to L1 data upon receipt from MOC
    • Xfer of HSK data to OF
    • Reconstruction & Digitization
    • Must be complete before next downlink
    • Xfer of diagnostics to OF
  – Reprocessing of L1 data as needed
  – Production of simulated data

• Data Cataloguing
  – Classification and storage of L1 data after processing
    • Summary information on each dataset
  – Tracking status and schedule of reprocessing
How it Works

• Pushes data through a sequence of processing steps
• Monitors status of a dataset as it proceeds through processing
• Provides notification upon failure of a processing stage
• Catalogues the result of processing for each dataset
OPUS – Introduction & Features

- Developed by AURA for the Space Telescope
- In use by several NASA experiments
- Does almost everything we need
  - Will run any *program* that can be “wrapped” by a shell script
  - Provides hooks for trapping processing errors
  - Distributes processing over a network of machines
  - Will run multiple processing sequences simultaneously
- Supports Extension of Functionality
  - Provides a C++ API to develop “OPUS-Aware” applications that have access to OPUS state information
- Displays live processing status (see next slide)
  - User can modify or override individual jobs & statuses
OPUS in Action (PMG)

(Image Courtesy of OPUS manual)
## OPUS in Action (OMG)

![Image of OPUS Observation Manager: bab5](Image Courtesy of OPUS manual)

### Table: OPUS Observation Manager: bab5

<table>
<thead>
<tr>
<th>OSFs</th>
<th>File</th>
<th>Edit</th>
<th>Manage</th>
<th>View</th>
<th>Tools</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>datas</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data_id</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dcf_num</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>o4r103wbc</td>
<td>04:52:14</td>
<td>sti</td>
<td>061</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r103w7e4</td>
<td>04:51:56</td>
<td>sti</td>
<td>060</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r103010</td>
<td>04:53:01</td>
<td>sas</td>
<td>000</td>
<td>_</td>
<td>_</td>
<td>_</td>
</tr>
<tr>
<td>o4r101vnq</td>
<td>2000 09/20 13:51:14</td>
<td>sti</td>
<td>054</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r101vkq</td>
<td>2000 09/20 13:51:14</td>
<td>sti</td>
<td>054</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r101vhhq</td>
<td>2000 09/20 13:51:12</td>
<td>sti</td>
<td>054</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r101vldq</td>
<td>2000 09/20 13:51:02</td>
<td>sti</td>
<td>053</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4r101010</td>
<td>2000 09/20 13:51:59</td>
<td>sas</td>
<td>000</td>
<td>_</td>
<td>_</td>
<td>_</td>
</tr>
<tr>
<td>o4rj05b3q</td>
<td>2000 09/20 13:49:37</td>
<td>sti</td>
<td>005</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4rh05dyq</td>
<td>2000 09/20 13:49:44</td>
<td>sti</td>
<td>018</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4lm52fdq</td>
<td>2000 09/20 13:50:46</td>
<td>sti</td>
<td>026</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
<tr>
<td>o4lm52f6q</td>
<td>2000 09/20 13:50:37</td>
<td>sti</td>
<td>025</td>
<td>c</td>
<td>n</td>
<td>c</td>
</tr>
</tbody>
</table>

(Image Courtesy of OPUS manual)
LAT-Specific Additions

• Support for the SLAC LSF Batch Processing Farm (Alex)
  – OPUS supports processing over multiple nodes using R/SSH
  – Alex is developing an interface layer that will extend this to the Batch Farm at SLAC

• Support for the ORACLE processing DB
  – OPUS saves log files containing the status of each processing stage
  – Dan is developing a set of scripts to wrap the processing database
  – At completion of processing for a particular dataset, these will catalog the status, location, and a summary of the dataset
Implementation Status

• **OPUS**
  – Sample pipeline has been developed
    • Runs GLEAM (Sim/Digi/Recon)
    • Runs ROOT macros to verify the output
  – Support garnered from development team

• **Database & LSF**
  – Database routines about 50% complete
  – LSF layer is researched and ready to be developed
  – Awaiting OPUS source code & documentation
    • Upon receipt, can develop the LSF layer and the Cataloguing program (that will use the DB routines)
GOALS

• DC1
  – Stress-test Pipeline components (OPUS, LSF-Layer, DB-Layer) by running GLEAM on many machines
    • Full implementation contingent on getting source code from STSC
    • Kludge implementation possible failing this in short order

• EM
  – Test a toy version of what will be the L1P using data as it comes off the instrument