Operational State and Servo Instability DMT Software

Keith Riles
University of Michigan

LIGO Scientific Collaboration Meeting
LIGO Hanford Observatory
August 16, 2000

Outline:

• Operational State Condition
  – Motivation
  – Definition of interface
  – Status

• Servo Instability Detection
  – What we want to detect
  – Definition of interface
  – Status
Before undertaking an analysis, one often needs to specify required machined conditions (Operational State).

Examples:

- Before seeking instability in Recycling Mirror (RM) servo:
  - Require lock of RM and Beam Splitter (BS) servos
  - Require wave front sensing engaged (or not engaged)
- Require laser intensity above certain threshold
- Require seismic RMS below ceiling (or above threshold)

Want to specify Boolean combinations of conditions:

Cond1 = “RM servo locked”
Cond2 = “BS servo locked”
Cond3 = Cond1 & Cond2

Want convenience of standard conditions, i.e., “Full Lock” or “IFO_quiet”

Also want run-time convenience of ascii configuration file
Operational State Condition Software

Sample Configuration File (April engineering run)
(To define quiet seismic motion at Hanford LVEA and locked single 2-km arm)

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>definition</th>
<th>double parms</th>
<th>int parms</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_quiet</td>
<td>rmsrange</td>
<td>&quot;H0:PEM-LVEA_SEISX&quot;</td>
<td>0.0</td>
<td>150.0</td>
</tr>
<tr>
<td>y_quiet</td>
<td>rmsrange</td>
<td>&quot;H0:PEM-LVEA_SEISY&quot;</td>
<td>0.0</td>
<td>150.0</td>
</tr>
<tr>
<td>z_quiet</td>
<td>rmsrange</td>
<td>&quot;H0:PEM-LVEA_SEISZ&quot;</td>
<td>0.0</td>
<td>300.0</td>
</tr>
<tr>
<td>all_quiet</td>
<td>boolean</td>
<td>&quot;x_quiet &amp; y_quiet &amp; z_quiet&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>arm_locked</td>
<td>valueabove</td>
<td>&quot;H2:ASC-QPDX_DC&quot;</td>
<td>10000.0</td>
<td></td>
</tr>
<tr>
<td>stable</td>
<td>boolean</td>
<td>&quot;all_quiet &amp; arm_locked&quot;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Condition types supported in initial release:

- **boolean**  | Boolean condition
- **valueabove** | Any value in time interval above threshold
- **valuebelow** | Any value in time interval below ceiling
- **valuerange** | Any value in time interval in range
- **meanabove** | Mean value in time interval above threshold
- **meanbelow** | Mean value in time interval below ceiling
- **meanrange** | Mean value in time interval in range
- **rmsabove** | RMS value in time interval above threshold
- **rmsbelow** | RMS value in time interval below ceiling
- **rmsrange** | RMS value in time interval in range
More types supported in initial release:

- **bitandany**: All bits in mask present in at least one value
- **bitnandany**: Not all bits in mask present in at least one value
- **bitorany**: At least one bit in mask present in at least one value
- **bitnorany**: No bits in mask present in at least one value
- **bitandall**: All bits in mask present in all values
- **bitnandall**: Not all bits in mask present in all values
- **bitorall**: At least one bit in mask present in all values
- **bitnorall**: No bits in mask present in all values
Internally, conditions defined by OperStateCond class with methods not (normally) accessed by users.

User interacts via a linked list (class OperStateCondList) of related condition objects

Example

Initialization:

```java
OperStateCondList osclist;

osclist.readConfig("myconfig.file");
```

When processing frames:

```java
if (osclist.satisfied("all_quiet") ) {
    do analysis
}
```

(Actual monitor should be derived from DatEnv class and pass data accessor object to osclist at initialization – see sample program `osc_sample.cc` for guidance.)
Operational State Condition Software

Status:

- Initial package integrated into production DMT
- Source code, sample program, makefile & documentation in \texttt{~dmt/cvs/dmt/src/dmtlib/osc/} on sand
- New condition types for servo instability detection defined in development version in \texttt{~keithr/osc/} on sand (next part of talk)
Servo Instability Detection Software

Servo “instability” refers here not only to real instability (runaway behavior), but also to any servo state with too high gain, giving excess noise just below the unity gain frequency.

Another worry is excitation of out-of-band resonances, \textit{i.e.}, internal test mass normal modes.

Want an early-warning system on all vulnerable servos to inform operator of impending problems.

Signatures:

- Rapidly increasing band-limited rms in servo channel
  (broad band for gain peaking, narrow band for resonance excitation)

- Significant deviation from nominal spectral shape

Want flexible ascii config file to allow tuning of spectral parameters and derivatives that define instability.

Want to combine servo instability conditions with other operational state conditions.

Natural solution: define new operational state condition types.
Servo Instability Detection Software

Sample Configuration File

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>channel</th>
<th>double parms</th>
<th>int parms</th>
</tr>
</thead>
<tbody>
<tr>
<td>butterfly</td>
<td>poweravemag</td>
<td>&quot;H2:LSC-GW_T0&quot;</td>
<td>6700</td>
<td>6800</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.0</td>
<td>10</td>
</tr>
</tbody>
</table>

Condition above satisfied if power in 6.7-6.8 kHz band increases by ≥ factor of two in 10 time intervals

⇒ Looking for excitation of “butterfly” test mass resonance

Eventually hope to have standard set of config file conditions to monitor all longitudinal and orientation servos with spectral and derivative parameters tuned for high sensitivity under nominally stable conditions

In the meantime, though, must deal with unstable absolute power levels and even unstable spectral shapes

⇒ Provide flexibility for development

⇒ Many possible servo instability conditions defined
Types supported in current development version:

- `abspowerabove`  Power in band above threshold
- `abspowerbelow`  Power in band below ceiling
- `fractpowerabove`  Fractional power in band above threshold
- `fractpowerbelow`  Fractional power in band below ceiling
- `abspoweranyrise`  Power in band rises fast in any interval
- `abspoweranyfall`  Power in band falls fast in any interval
- `fractpoweranyrise`  Fractional power rises fast in any interval
- `fractpoweranyfall`  Fractional power falls fast in any interval
- `abspoweraverise`  Power in band rises fast over N intervals
- `abspoweravefall`  Power in band falls fast in over N intervals
- `fractpoweraverise`  Fractional power rises fast over N intervals
- `fractpoweravefall`  Fractional power falls fast over N intervals
- `poweranymag`  Power in band magnified fast in any interval
- `poweranydemag`  Power in band demagnified fast in any interval
- `poweravemag`  Power in band magnified fast over N intervals
- `poweravedemag`  Power in band demagnified fast over N intervals
New operational state conditions provide tools, but also need a dedicated background monitor to apply those tools.

Monitor should read osc config file and monitor control file

Sample monitor control file

```
interval 2.0
logfile /export/home/keithr/sample.log
trigger butterfly warn epics 60. meta 3600.
```

Above config file requests:

- Monitoring at 2-second time intervals
  (default = 1.0 second, minimum=\(\frac{1}{16}\) second)
- Logging of all output to the file
  `/export/home/keithr/sample.log`
  (default = console)
- Triggering if “butterfly” condition (defined in osc config file) is satisfied. The trigger is given a warning priority, sets an EPICS alarm (but no more frequently than every 60 seconds), and generates a meta-database entry (but no more frequently than every 3600 seconds).

The epics and/or meta keywords can be omitted. Default disposition is merely the logging of triggers.
Status:

- Development version of revised osc package available in
  \texttt{\sim keithr/osc/} on sand
  (includes private enhancements to dmt Sine class and temporary kludge to handle 1st-bin inconsistency in dmt F Spectrum class)

- Directory also contains sample program with makefile and osc config file
  (but without EPICS or meta-database triggers – output logging only)

- Will deliver background monitor program with EPICS / meta-database triggers by September 15

- Will deliver revised monitor with support for graphical display of defined conditions, run-time spectra and thresholds by November 15, along with config and monitor control files tuned to key 2-km servos. (Will work with Dick Gustafson on tuning.)