Customizing the PTP monitoring layout

July 31, 2013 | Carsten Karbach
Monitoring Architecture I
Monitoring Architecture II

- **LML**da gathers status information, calls target system’s remote commands, written in Perl
- **LML** is an XML data format for status information of supercomputers
- LML request: contains table filtering information, visible/hidden columns, machine topology
- LML response: contains the request and status information
- Client stores current layout request for successive Eclipse sessions

**Objective: Customize the LML-Layout**

define machine topology, arrangement of nodes, layout of job lists
Example Default Layout JUROPA

Two-level hierarchy of nodes and cores
Customized Layout JUROPA

- Four-level hierarchy of partitions, racks, nodes and cores
- Images attached to each partition
- Customized background colors
- Disabled columns in job list
Content

1. LML-Layout
2. Define machine topology
3. Setup customized LML-Layout
Part I: LML-Layout

July 31, 2013  |  Carsten Karbach
LML-Layout

- LML-Layout configures the layout of the system monitoring perspective
- Examples for customization:
  - Activate/Deactivate table columns
  - Arrangement of nodes
  - Background colors for nodes
  - Map machine topology, e.g. place nodes into racks
  - Attach images to nodes
- Layout-content is
  - `tablelayout` configures job lists
  - `nodedisplaylayout` defines machine topology, customizes layout of nodes view
Layout Structure

- One tablelayout for each job list
- layout element will be placed into target system configuration (see 3)
## Table Layout

```
<tablelayout id="tl_Run" gid="ActiveJobsView" active="true"
contenthint="jobs">
  <column key="step" cid="1" pos="0" width="0.1" active="true" />
  <column key="owner" cid="2" pos="1" width="0.1" active="true" />
  ...
</tablelayout>
```

- **active** attributes configure visibility of columns
- **key** specifies the attribute shown in the column
- **cid** is the column ID
- **pos** is the position of the column within the table
- **width** is percentaged width of the column
Part II: Define machine topology

July 31, 2013  |  Carsten Karbach
2. Define machine topology

**Nodedisplay Layout**

```xml
<nodedisplaylayout id="nodedisplay" gid="nd_1" active="true">
  <schemehint>
    <el1 tagname="node" min="1" max="206" mask="judge%03d">
      <el2 tagname="core" min="0" max="25" mask="-c%02d" />
    </el1>
  </schemehint>
  <el0 rows="10" cols="21" maxlevel="2" vgap="5"/>
</nodedisplaylayout>
```

- **schemehint** defines the machine topology, here:
  - Two level hierarchy of nodes and cores
  - 206 nodes, each having 26 cores
  - **mask** attribute is used to map the node ID (1..206) to the actual node names (judge001..judge206)

- **el0** contains layout information for the nodes view, here:
  - The nodes are arranged in a grid of 10 rows and 21 columns
  - **maxlevel=2** defines that one rectangle is painted for each core, **maxlevel=1** would draw one rectangle for each node
2. Define machine topology

Layout Result
2. Define machine topology

**Juropa Nodedisplay Layout**

```xml
<nodedisplaylayout id="nodedisplay" gid="nd_1">
  <schemehint>
    <el1 tagname="part" min="0" max="0" map="jj">
      <el2 tagname="rack" min="1" max="25" mask="%02d">
        <el3 tagname="node" min="1" max="96" mask="c%02d">
          <el4 tagname="core" min="0" max="15" mask="-c%02d"/>
        </el3>
      </el2>
    </el1>
    <el1 tagname="part" min="1" max="1" map="jf">
      ...
    </el1>
  </schemehint>
  <el0 vgap="5" maxlevel="3" cols="1" rows="2" background="#aaa">
    <el1 min="0" max="0" showtitle="true" rows="1" cols="25" maxlevel="4">
      <img src="http://www...." align="WEST" width="0.3"/>
    </el1>
    <el2 min="1" max="25" showtitle="true" rows="32" cols="3" maxlevel="4"/>
  </el0>
</nodedisplaylayout>
```
2. Define machine topology

Juropa Layout Result

![Diagram of machine layout]
Juropa Layout Explanation

- **Four-Level** hierarchy, 2 parts, the first with 25 racks, each having 96 nodes
- The concatenated masks need to match the real node names, *map* is used for names without digits
- JUROPA node names match the scheme *jj%02dc%02d*, e.g. *jj09c18*
- Allowed node names defined by the above *schemehint* are *jj01c01..jj01c96..jj02c01..jj25c96*
- The tree within the *el0* element references the tree provided by the schemehint
- *el0* configures the grid for the partitions
- *el1* configures the grid for the racks within the first partition, here it defines: each rack should print its name (showtitle), the racks are arranged in a 1X25 grid (rows,cols)
Part III: Setup customized LML-Layout

July 31, 2013  |  Carsten Karbach
Simple Setup I

- Not yet included in a build, should come soon
- Track the process at bug 360435

Step 1: Create a connection to your target system

Step 2: Let LML\_da create tmp files on your remote machine
3. Setup customized LML-Layout

Simple Setup II

Step 3: Refresh your connection

<table>
<thead>
<tr>
<th>Status</th>
<th>Connection Name</th>
<th>Configuration Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Judge</td>
<td>de.fz-juelich.judge.newlayout.torque.batch</td>
</tr>
<tr>
<td></td>
<td>Juqueen</td>
<td>de.fz-juelich.juqueen.ll_bg</td>
</tr>
</tbody>
</table>

Step 4: Use the layout.xml file as template, copy content

```
karbach@juqueen1:~/.eclipsesettings $ cd tmp_juqueen2.zam.kfa.juelich.de_30815/
karbach@juqueen1:~/.eclipsesettings/tmp_juqueen2.zam.kfa-juelich.de_30815 $ ll
total 3776
-rw-r--r-- 1 karbach zamb 61544 Jul 24 13:25 blocks_LML.xml
-rw-r--r-- 1 karbach zamb 1201838 Jul 24 13:25 datastep_eddcolor.xml
-rw-r--r-- 1 karbach zamb 1183246 Jul 24 13:25 datastep_combineLML.xml
-rw-r--r-- 1 karbach zamb 1225000 Jul 24 13:25 LML_da.log
-rw-r--r-- 1 karbach zamb 21181 Jul 24 13:24 nodes_LML.xml
-rw-r--r-- 1 karbach zamb 6839 Jul 24 13:24 request.xml
-rw-r--r-- 1 karbach zamb 574 Jul 24 13:24 sysinfo_LML.xml
-rw-r--r-- 1 karbach zamb 1708 Jul 24 13:24 workflow.xml
```

July 31, 2013
Carsten Karbach
Simple Setup III

Step 5: Create a customized TSC

Step 6: Import the TSC you want to adjust

see also FAQ for this step
3. Setup customized LML-Layout

Simple Setup IV

Step 7: Place content of layout.xml into TSC and customize it

Note, how the layout element is declared. Insert it as child of monitor-data.

Step 8: Adjust the name of your TSC

```xml
<control-data>
  <monitor-data schedulerType="LL_BG">
    <resource-manager-builder xmlns="http://eclipse.org/ptp/rm"
                               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                               name="de.fz-juelich.juqueen.layout.ll_bg">
      <control-data>
```
Simple Setup V

Step 9: Remove the old connection

<table>
<thead>
<tr>
<th>Status</th>
<th>Connection Name</th>
<th>Configuration Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Judge</td>
<td>de.fz-juelich.judge.newlayout.torque.batch</td>
<td></td>
</tr>
<tr>
<td>Juqueen</td>
<td>de.fz-juelich.juqueen.ll_bg</td>
<td></td>
</tr>
</tbody>
</table>

Step 10: Create a new connection with your new TSC/LML-Layout
Simple Setup – Remarks

- After each change of the LML-Layout you need to repeat step 9-10 to activate the changes.
- The customized LML-Layout is only sent on the first connect.
- The XML schema for the LML-Layout is documented here.
- If your changes are not updated, check if your custom LML-Layout is valid.
- You can easily publish your customized TSC, so that all users of your remote machine can benefit from your layout.
- The adjusted TSC could even be added as default TSC to the list of standard TSCs.
Alternative Setup I

- Already working with latest PTP Kepler build
- Will not work as soon as *Simple Setup* is implemented
- Step 1-4 are identical with the simple setup

**Step 5: Copy layout.xml into samples directory**

```bash
karbach@juqueen1:~/.eclipse_settings $ cd tmp_juqueen2.zam.kfa-juelich.de_21451/
karbach@juqueen1:~/.eclipse_settings/tmp_juqueen2.zam.kfa-juelich.de_21451 $ cp layout.xml ../samples/layout_default_LL_BG.xml
karbach@juqueen1:~/.eclipse_settings/tmp_juqueen2.zam.kfa-juelich.de_21451 $ []
```

Name the file in the samples directory `layout_default_RMS.xml`, where RMS has to be replaced with the remote system name abbreviation. You can find this abbreviation in the `report.log` file within the tmp directory or in the corresponding TSC in the `monitor-data` element stored in the `schedulerType` attribute (e.g. TORQUE, LL_BG, GridEngine, COBALT_BG, SLURM_ALPS, SLURM)

**Step 6: Adjust the layout file layout_default_RMS.xml**
Alternative Setup II

- Run steps 9-10 of the simple setup

Remarks

- This setup is estimated as harder, because you need to detect the RMS abbreviation and you have to change files directly on your remote system
- Disadvantages of this setup:
  - If there is a new version of LML_da, it is automatically copied to your remote machine and might overwrite your custom layout
  - There is no automatic validation of your custom LML-Layout
  - You can only have one LML-Layout per remote machine, while with the *simple setup* you can have multiple
Questions?
Contact

- **E-mail:**
  c.karbach@fz-juelich.de, w.frings@fz-juelich.de

- **LML** → [http://llview.zam.kfa-juelich.de/LML](http://llview.zam.kfa-juelich.de/LML)