Opstate Solutions Comparison

Rob Wilton
IETF 94 – Yokohama, NETMOD WG
High level summary of Requirements

The core “opstate requirements” basically come down to:

• To recognize that some systems are configured asynchronously
• To determine what configuration a system is actually running. I.e. programmed everywhere it needs to be
• To be able to determine when some intended configuration change has actually been applied to a system
Proposed Solutions

Three solutions have been presented as drafts that each solve the core problem in different ways:

1. draft-openconfig-netmod-opstate
   This solution is based around the **structure of the model**

2. draft-kwatsen-netmod-opstate
   This solution is based around **multiple datastores**

5. draft-wilton-netmod-opstate-yang
   This solution is based around **schema encoding**
<table>
<thead>
<tr>
<th>Solution</th>
<th>Main benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Model structure</td>
<td>No enhancement to NETCONF or RESTCONF needed. Does not require datastores.</td>
</tr>
<tr>
<td>2. Multiple datastores</td>
<td>No change required to YANG data models. Minimal NETCONF protocol impact.</td>
</tr>
<tr>
<td>3. Request encoding</td>
<td>No change required to YANG data models. Does not require datastores.</td>
</tr>
</tbody>
</table>