Reducing the Human Cost of Grid Computing with glideinWMS

by Igor Sfiligoi\textsuperscript{1}, F. Würthwein\textsuperscript{1}, J.M. Dost\textsuperscript{1}, I. MacNeill\textsuperscript{1}, B. Holzman\textsuperscript{2}, and P. Mhashilkar\textsuperscript{2}

\textsuperscript{1}UCSD \textsuperscript{2}FNAL
Our environment - Grid computing

- Set of loosely coupled compute clusters (i.e. sites)
- Great for resource providers (i.e. site operators)
  - High autonomy
  - Easy sharing between communities (VOs)
  - High utilization
- Not so great for users
  - Actually, not too bad for users when things work
  - But handling failures extremely time consuming
    - May need to contact multiple site admins
A problem of scale

- O(100) sites
  - Aggregate of O(100k) CPUs
  - At least a few sites have some broken nodes at any point in time
- O(10k) users
  - O(100) users likely hit by those broken nodes every day
  - If each spent even 30 mins debugging
    - O(10) scientific FTEs wasted (and I am being an optimist)
    - Plus drastic reduction in usability (users expect things “to just work”)
The glideinWMS

- The glideinWMS approach to the problem
  - Use the pilot paradigm
  - Split pilot submission from pilot regulation
  - Emphasize sharing of pilot submission service

The glideinWMS is a Grid job scheduler initially developed at FNAL by the CMS experiment
- Based on the CDF glideCAF concept
- With contributions from several other institutes
- Widely used in OSG, with a large instance at UCSD
The pilot paradigm

- Send pilots to Grid sites (never user jobs)
  - Create a dynamic overlay pool of compute resources
  - Handle user jobs within this overlay pool
- A broken node will fail pilot jobs
  - So they will not join the overlay pool
  - No user job ever fails
- Problem moved to the pilot submitter

Pilots not user specific

Overlay pool

One pool user community

Site N

Site 1

Pilot submitter

Pilot

Pilots
Cost reduction

- **Difference in job types**
  - All user jobs are precious => **must recover**
  - Pilot jobs are all the same => pilot failures not critical
    - Failures used to detect broken compute nodes
    - **Diagnose node problem**

- **Fewer humans exposed**
  - Can be **more expert** => **lower cost per event**

### Estimates for a sizable OSG VO

<table>
<thead>
<tr>
<th>Metric</th>
<th>Entities to debug</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(10M) jobs</td>
<td>O(100k)</td>
</tr>
<tr>
<td>O(1k) nodes</td>
<td>O(10)</td>
</tr>
</tbody>
</table>

Assuming 1% error rate

Reduction by several orders of magnitude
Traditional pilots & multiple VOs

- Each user community (VO) wants its own pilot infrastructure
  - To maintain control over scheduling policies

- Many pilot admins, debugging the same sites
Splitting the process

- The glideinWMS separates
  - pilot submission (glidein factory)
  - from pilot regulation (VO frontend)
- Credential owed by VO frontend
  - And delegated to factory as needed
- All scheduling policy implemented in the frontend
The factory can be shared

- Each VO runs **only** its own VO frontend (with the associated overlay pool)
  - While still having full control over policy
- All debugging handled by a single factory team
Risk of common factory?

- A single factory is a single point of failure
  - And possibly a scalability choke point
- The glideinWMS allows for multiple factories
  - For redundancy, scalability, trust, etc.
  - Of course the cost goes up
- How many factories to use is a balance between low cost and low risk
  - Each VO can decide what works best for it
OSG experience

- Operating a multi-VO factory since 2009
  - 12 VOs at the time of writing
- Gliding into ~100 Grid sites
  - We include sites that claim to support the VOs we serve
  - Significant fraction shared
- Weekly statistics

<table>
<thead>
<tr>
<th></th>
<th>All sites</th>
<th>Shared sites</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total glideins</td>
<td>200k</td>
<td>130k</td>
</tr>
<tr>
<td>Failing glideins</td>
<td>25k</td>
<td>22k</td>
</tr>
</tbody>
</table>

One VO much bigger than the other
Effort investment

• About 1 FTE total
  • Only fraction for actual Grid debugging
  • Comparable fraction helping VOs debug problems between Grid nodes and their VO overlay pool
• We also help with know-how in configuring and operating the overlay pool
Savings estimate

- Not counting the consulting services
  - Those tend to be high at start-up and then level off
- For the remainder of the effort:

<table>
<thead>
<tr>
<th></th>
<th>Shared factory</th>
<th>VO provided factory</th>
<th></th>
<th>OSG-wide (12 VOs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Per VO</td>
<td>OSG-wide</td>
<td></td>
</tr>
<tr>
<td>Grid debugging</td>
<td>25%</td>
<td>15%</td>
<td>180%</td>
<td></td>
</tr>
<tr>
<td>Pilot Debugging</td>
<td>28%</td>
<td>15%</td>
<td>180%</td>
<td></td>
</tr>
<tr>
<td>Automation R&amp;D</td>
<td>14%</td>
<td>10%</td>
<td>120%</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>67%</strong></td>
<td><strong>40%</strong></td>
<td><strong>480%</strong></td>
<td></td>
</tr>
</tbody>
</table>

7x cheaper
Conclusions

- Failures in a highly distributed system like the scientific Grids can have a high human cost
- The **pilot paradigm** drastically reduces this by
  - Catching errors during provisioning
  - Debugging by expert staff only
- The **glideinWMS** further reduces the cost by allowing for a **shared pilot factory**
  - Confirmed by the OSG experience
For more information

- The glideinWMS home page
  http://tinyurl.com/glideinWMS

- Relevant papers and supporting material:
  - I. Sfiligoi et al.,
    "The pilot way to grid resources using glideinWMS,"
    CSIE, WRI World Cong. on, vol. 2, pp. 428-432, 2009,
    doi:10.1109/CSIE.2009.950
  - Open Science Grid home page,
    http://www.opensciencegrid.org/
Acknowledgment

This work is partially sponsored by

- US Department of Energy under Grant No. DE-FC02-06ER41436 subcontract No. 647F290 (OSG)
- the US National Science Foundation under Grants No. PHY-0612805 (CMS Maintenance & Operations), and OCI-0943725 (STCI).