dCache as open-source project showcase for education
Tigran Mkrtchyan for dCache team
CHEP2018, Sofia, 11.07.2018
- 76 installations for WLCG
- ~10 no GRID installations
- ~50% of LHC data
Some numbers

- ~60 developers/contributors in 18 years
  - DESY
  - FNAL
  - NDGF
- 25 modules
- 61 production branches (since 01.11.2007)
- 926 releases (since 01.11.2007)
- ~400K lines of code in ~2600 files
- ~100 external dependencies
- RPM, DEB, TAR, Solaris and Docker packages
Version management
Release Policy

- Yearly long-term branch
  - 2+ years support
- Time-based feature released
  - ~every 3 mounts
  - supported until next LTS release
- Weekly maintenance releases
  - release for all supported branches
The “Release Process” gives a common framework to get stuff done with a predictable and reproducible result.
Code policy

- master/dev
  - ALWAYS compilable
  - ALMOST deployable
- Branch
  - ALWAYS releasable
Course goal

- Data intensive science, Big Data
- Introduction to Open Source project
- Use of standard tools
  - Students should be ready to join any project
  - github
  - Maven
- Developer team interaction
  - Code review
- Release management
Project adjustments

- Standard project structure
  - Convention vs. Configuration
- Standard build environment
  - Make project IDE ready
- Standard workflow
- Simple instruction how to build
  - BUILDING.md
### Development effort guideline

<table>
<thead>
<tr>
<th>Impact</th>
<th>Easy to do</th>
<th>Difficult to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Minor</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

1. Implement immediately
2. Implement immediately
3. Long term plans and development
4. Discarded
# Development effort guideline

<table>
<thead>
<tr>
<th>Impact</th>
<th>Easy to do</th>
<th>Difficult to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Minor</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

- 1. Implement immediately
- 2. Implement immediately
- 3. Long term plans and development
- 4. Discarded

Students work target
Student assignments

- Simple development task
  - Mostly in a single file
  - No project prior knowledge is required
- Collected over the time
  - (easy with minor impact)
- Formed as GitHub issues
  - easypick / education
- Goal: merge of a pull request
Commit/PR workflow

- Full Build/Deploy/Test
- Code analysis
- Unit tests
- Functional tests
- Fail Fast

Aggregated Test Result

<table>
<thead>
<tr>
<th>Test</th>
<th>Fail</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2-SRM-Tests #553</td>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>GridTools-Functional-Tests #848</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>synfe-test-suite #7560</td>
<td>0</td>
<td>866</td>
</tr>
<tr>
<td>Deploy on sisyphus #2681</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>webadmin function test</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

978 tests
Good COMMIT

- Does only one thing
  - easy to review
  - easy to move around
  - easy to revert
- As small as possible
  - “your diff does not fit on my screen”
- Informative log message
  - `git blame` is your friend
Contributor agreement

- External contributors may sneak proprietary code
- Have clear contribution policy
- dCache uses linux kernel model
  - `Signed-off-by:` annotation
- Check with your legal department!
Example:

dcache: added a decorator for RepositoryChannels to get I/O statistics (#3430)

**Motivation:**
Getting a possibility to generate statistics and spot misbehavior in I/O processes of RepositoryChannel.

**Modification:**
Added package statistics, created the class IoStatisticsChannel as decorator for a RepositoryChannel and created the class IoStatistics which provides statistics on number of read and write requests, read and write I/O block sizes and different measures about the read and write I/O speed.

**Result:**
Possibility to add a decorator to a RepositoryChannel which provides statistics about I/O processes.

Signed-off-by: Lotta xxx <xxx@htw-berlin.de>
Goals of code review

- Let others see my code
  - Forces me to publish better code
- Learn best-practice
  - New libraries used
  - New technique used
- Learn about code changes (by others)
  - reduce 'bus factor'
Conclusions

- Use standard tools to attract students
  - make them benefit from new skills
- Make your infrastructure robust for external contributors
  - ensure reliable testing and deployment
- Clear license and contribution policy
- Be open for new contributors
  - simple tasks for newcomers
  - Document build process
Thank You!
Build system setup

- Jenkins master
- Build Jobs
- Testing Jobs
- Deployment Jobs
- Build nodes
- Test nodes
- Deploy nodes
Task: #4

dCache version 2.6 have switch to a new version of Berkeley DB, which was binary incompatible with the used one. A migration script has been added, which is not necessary any more (the task requires git-based archaeology).

Description:
remove file $ conflicts with corresponding
starting point: use 'git log' to find the commit script
added (git/branch/mvc/jre.4.1.2) jar

Task: #9

To spot misbehaving applications and understand data access profile a low level statistics on IO requests is required.

Description:
Create a RepositoryChannel decorator which can provide statistics on number of requests, IO speed and block sizes

Starting point: RepositoryChannel Interface and Decorator design pattern

Task: #12

When inspecting active transfers a 'show transfers' command in the dCache's admin interface is used. On a busy system, the output of the command may return more lines than can be visually inspected.

Description:
Enhancements to NFS doo's admin interface to filter active transfers by client, nfsid and pool.

Starting point: NFSv41Door class and java8 stream API.
Commit message

module: summary line (up to 60 chars)
(empty line here)
Motivation:
(explain why we need this change)
Modification:
(explain what was changed)
Result:
(what will be different now)
Some extra info:
(bug tracker id, branch back-port)
Commit message

ssh: fix handling of idle connection timeout

Motivation:
To preserve idle ssh connection one have to adjust IDLE_TIMEOUT, however sshd has yet an other timeout - READ timeout ....

Modification:
Explicitly set read timeout to prevent it to fire before idle timeout.

Result:
idle ssh session is not closed

Ticket: #9175
Target: master, 3.1