SALOME Maintenance Procedure

Frédéric Pons (Open Cascade)
Roman Nikolaev (Open Cascade)
Back Office

- Back Office Organization

- Back Office Tasks
  - Continuous integration
  - Production and Qualification of Released versions
  - Maintenance
Back Office Organization

Back Office
Nijni Novgorod (Russia)

CEA Project Manager

Open CASCADE Project Manager

Back Office Project Manager

Open CASCADE QA Manager

Production Team

EDF Front Office

CEA Front Office
Back Office Team

- 1 Software Developer Manager
- 4 developers full time
- Pool of 15-20 developers working with SALOME
  - Improvements projects
  - Applications based on SALOME
Where is Nijni?
Back Office

- Back Office Organization
- Back Office Tasks
  - Continuous integration
  - Production and Qualification of Released versions
  - Maintenance
Continuous Integration

- Building SALOME
  - Sources & Prerequisites
  - Unit tests

- Platforms
  - 5 official platforms (4 Linux + Windows)
  - 3-4 additional Linux platforms

- SALOME modules
  - 57 repositories
  - ~30 modules maintained
  - > 60 configurations
Infrastructure

A pool of virtual workstations is used to build and test SALOME application on all officially supported platforms.

<table>
<thead>
<tr>
<th>Server</th>
<th>CentOS 6.4</th>
<th>Debian 8.0</th>
<th>Fedora 24</th>
<th>Ubuntu 16.04</th>
</tr>
</thead>
<tbody>
<tr>
<td>vms</td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
</tr>
<tr>
<td></td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
</tr>
<tr>
<td></td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
</tr>
<tr>
<td></td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
<td>vm</td>
</tr>
</tbody>
</table>
Version Control System

- Private repositories managed in CEA’s Forge
- Public repositories synchronized every day

GIT Web interface also is available: [http://git.salome-platform.org/gitweb](http://git.salome-platform.org/gitweb)
Build third-party prerequisites

- All SALOME third-party prerequisites are built from scratch on each platform
- Shell script is used to build third-party prerequisites
- Sometimes patching of the source code of the third-party prerequisite’s is required
Jenkins CIS

- Jenkins is an open source automation server for continuous integrations

- It is used to fully automated regular building and testing of SALOME application

- The workflow is split to a set of sub-tasks ("jobs" in the terms of Jenkins), which are executed one by one
Jenkins: Dashboard
Jenkins: View build and test results

- Jenkins stores logs which are used to analyze reason of job’s failure
Check nightly build

- Night build starts automatically (scheduled run):
  - Download latest SALOME sources from GIT repositories
  - Build all SALOME modules on all platforms
  - Run automatic test for all SALOME modules on all platforms (unitary tests: make test)
  - Run non-regression tests for whole SALOME application

- Jenkins allows to control that:
  - All SALOME modules are correctly built
  - All unitary tests pass
  - All non-regression tests pass
Quick testing

- Quick testing is performed via `make test` command for each SALOME module.
- Usually testing procedure consists of C++ and/or Python unitary tests.
- This task usually takes about 2 hours.

<table>
<thead>
<tr>
<th>SALOME Module</th>
<th>Number of tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry</td>
<td>112</td>
</tr>
<tr>
<td>Mesh</td>
<td>126</td>
</tr>
<tr>
<td>YACS</td>
<td>12</td>
</tr>
<tr>
<td>MED, MEDCoupling</td>
<td>25</td>
</tr>
</tbody>
</table>
Non regression testing

- SALOME non regression test base is implemented in Python
- The test base includes ~2500 test cases for KERNEL, Geometry and Mesh modules
- New tests are created for new features or bugs
- This task usually takes about 4-5 hours
Back Office

- Back Office Organization

- Back Office Tasks
  - Continuous integration
  - Production and Qualification of Released versions
  - Maintenance
Production and Qualification of Released versions

- Production of new version
  - ~2 months to produce a version
  - 2 official versions per year

- Workflow
  - Alpha version: stabilization of prerequisites
  - Beta version: integration of developments
  - Release Candidate: version tested internally CEA/EDF
  - Final version
TestLink is a web-based test management system that facilitates software quality assurance. The tool offers support for test cases, test suites, test plans, test projects and user management, as well as various reports and statistics.
Test link: Test suites, Test cases, Test plans
GUI testing

- GUI testing is performed within version release procedure
- Test campaign is performed by QA engineers
- TestLink and MS Excel test cases (test scenarios) are used
- Mantis bug-tracker is used to register bugs found during test campaign
- All failed scenarios are re-tested after resolving of all related problems
- For each release a Test Report is created

<table>
<thead>
<tr>
<th>SALOME Module</th>
<th>Number of tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry</td>
<td>43</td>
</tr>
<tr>
<td>Mesh</td>
<td>74</td>
</tr>
<tr>
<td>YACS</td>
<td>3</td>
</tr>
<tr>
<td>Other GUI functiona</td>
<td>22</td>
</tr>
</tbody>
</table>
Example of the test case (test scenario)

- Each SALOME test case (or test scenario) represents a set of actions which may be accompanied with images (snapshots). Test case is written in form of MS Excel document.

<table>
<thead>
<tr>
<th>Scenario: MESH-017</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1_2-9a: Multiple elements modification: smoothing, united with Mesh-016 (free edges)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case</th>
<th>Description</th>
<th>Notes on description</th>
<th>Result</th>
<th>Notes on result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Application start</td>
<td>Remove file, config/salome/SalomeApp/etc 7.7.0 file in home directory, if it exists</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>Launch SALOME</td>
<td>OK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Menu File \ New</td>
<td>OK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>New study appears, including Object Browser and Python windows</td>
<td>OK</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name: Generating mesh</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Click on Mesh Components toolbar button</td>
</tr>
<tr>
<td>2.2 &quot;Mesh&quot; module is activated, new window with VTK Viewer appears</td>
</tr>
<tr>
<td>2.3 Type &quot;import SMESH; test1&quot; in the python console and press &lt;Enter&gt;</td>
</tr>
<tr>
<td>2.4 &quot;Mesh/...&quot; and &quot;Geometry/...&quot; data trees appear in the Object Browser</td>
</tr>
<tr>
<td>2.5 Select &quot;Mesh/Meshbox&quot; object in the Object Browser, call popup - Compute</td>
</tr>
<tr>
<td>2.6 Close in Mesh computation succeeded window.</td>
</tr>
</tbody>
</table>

Computed mesh appears in the 3D viewer. | OK |
Back Office

- Back Office Organization

- Back Office Tasks
  - Continuous integration
  - Production and Qualification of Released versions
  - Maintenance
Maintenance

- Since 2005
- More than ~2200 issues since 2008
Maintenance – Bug Tracker

Mantis is used for tracking of all SALOME issues: [http://salome.mantis.opencascade.com](http://salome.mantis.opencascade.com)

Access is restricted via IP filter & login/password
Maintenance – workflow of SALOMÉ issue

- status for clarifying of requirements, solutions between FO and BO
Thank you for attention