Change-sets

Basavaraj Karadakal (bkrdkl@juniper.net)
Development environment

- Component based development model.
- Developers spread across multiple sites.
- Over 800 components.
- Over 500 repos.
- Repos nested up to 4 levels in the sandbox.
- Development sandboxes managed by in house wrapper scripts.
  - Based on android repo tools.
  - Provides flexible interface to add / remove repos / repo-sets to sandbox.
  - Implements workflow that require users to provide information about the tests they have run and defects they are addressing when uploading changes for review.
- Changes uploaded for reviews often span multiple repos.
  - Developers use wrapper to upload the changes.
What are change-sets?

- Change-set is a set of commits across multiple repos, uploaded together for review, from a single sandbox.
- Meta-data about the change-set is tracked in the user sandbox in a special tracking repository.
- All commits that are part of change-set share same Gerrit “topic”.
  - Makes use of topic submission functionality.
  - All changes part of a change-set are submitted together.
- Change-set is identified by it’s id which corresponds to the common topic of the changes.
- Change-set ids are generated based on user input and are ensured to be unique.
- Change-set is unit of build and test in CICD pipeline.
Why change-sets?

- Enable CICD Pipelines.
  - Provides ability to kick off the pipeline exactly after all the dependent changes across multiple repos are uploaded for review.
  - Works the same for uploading amends to address review comments.
  - In a sense provides atomicity to push the changes through CI pipeline.
- Enable atomicity while pulling the changes across all repos.
- Provide more control over what gets uploaded from the sandbox.
  - Users can create and work on multiple change-sets in a sandbox.
How is a change-set created?

• Explicit way.
  • User creates the change-set first before committing any of the changes to be made part of the change-set.
  • Gives more control over what changes are made part of a change-set.

• Implicit way.
  • Change-set is automatically created for the user.
  • When user uploads changes, a new change-set is created automatically and all the commits in the sandbox are made part of the newly created change-set.
  • What commits will be made part of change-set can be controlled only at repo level. All new commits in a repo are made part of the change-set.
Create a change-set explicitly

```
bkrdkl@bkrdkl-vb:~/workspace/demo/dem01$ sb cset create -p $MYPR -t 'test cset'
Creating new cset...
Created cset test_cset-1102653-10183f17a http://bkrdkl-vb:8080/73
All subsequent commits in the sandbox will be associated with this cset.
To deactivate the cset, run "sb cset deactivate"
```

```
bkrdkl@bkrdkl-vb:~/workspace/demo/dem01$ sb cset list
Csets local to sandbox
  * EMPTY test_cset-1102653-10183f17a bkrdkl
```

```
bkrdkl@bkrdkl-vb:~/workspace/demo/dem01$ sb cset info
Topic: test_cset-1102653-10183f17a
State: EMPTY
Cset URL: http://bkrdkl-vb:8080/73
Sandbox: bkrdkl-vb:/home/bkrdkl/workspace/demo/dem01
```
Explicit workflow

- Create a change-set before committing any changes in sandbox OR activate an existing change-set created before.
  - The explicitly created change-set will be activated in the sandbox.
  - All commits done in the sandbox are associated with the activated change-set.
  - If there is no active change-set then commits are not associated with any change-set.
  - Only a change-set local to the sandbox can be activated.
- Upload the change-set.
  - Individual change-set or change-sets can be uploaded as long as the dependencies are met.
    - All the dependencies of the change-sets should be already uploaded or all dependencies to be specified in the list of change-sets to be uploaded.
    - Users can check the dependencies between change-sets in their sandboxes.
- Submit the change-set.
Change-set state transitions

- **EMPTY**
  - sb cset create
  - sb cset mv

- **LOCAL**
  - git commit

- **UPLOADED**
  - sb cset upload

- **UPLOADED+LOCAL**
  - sb cset upload
  - commit - amend

- **MERGED**
  - submit

- **ABANDONED**
  - abandon
Implementation

• Change-sets functionality is implemented with
  • Client side wrapper scripts.
  • Client side hooks.
  • Change-set reviewer plugin.
• A tracking commit is created per change-set in a special tracking repo on a local topic branch per change-set.
  • The tracking commit is created by committing a unique empty file.
  • All merges of tracking commits are assured to be trivial.
• When changes are uploaded for review the tracking commit is automatically uploaded.
• The tracking commit shares the same topic as the other commits in the change-set.
• The CI pipelines depend on the events happening in the tracking repo / change set repo.
Implementation

- All commits and change-sets data is tracked in notes on a special reference in the tracking / cset repo by client side hooks and wrapper scripts.
- All post hooks catalog the commits against the change-set. Pre-commit hook prevents commits from being associated with a merged change-set.
  - Pre-commit
  - Post-commit
  - Post-rewrite
  - Post-am
Change-set tracking repo

```
bkrdkl@bkrdkl-vb:~/workspace/demo/demo1/csets $ git branch
  __commits_track_branch
  * cset-test_cset-1102653-10183f17a
  cset-test_upload-1102653-9a3413cef
  master
bkrdkl@bkrdkl-vb:~/workspace/demo/demo1/csets $ git show
commit 6b48b2f08d83a20a6270245a6c73fa389d256088
Author: Basavaraj Karadakal <bkrdkl@juniper.net>
Date:   Tue Nov 15 18:00:38 2016 -0800

test_cset-1102653-10183f17a

  Sandbox location: bkrdkl-vb:~/home/bkrdkl/workspace/demo/demo1

  Change-Id: Idfe89cab5a05258d0c28537839553f4bd777a10b

diff --git a/10/183f17a0764e6bf119cd720a3ee0876c279ed4 b/10/183f17a0764e6bf119cd720a3ee0876c279ed4
new file mode 100644
index 0000000..e69de29
bkrdkl@bkrdkl-vb:~/workspace/demo/demo1/csets $  
```
Empty Change-set data structure

```json
"change_id": "Idfe89cab5a05258d0c28537839553f4bd777a10b",
"changes": {},
"created": 1479261640.958241,
"cset_topic_branch": "cset-test_cset-1102653-10183f17a",
"cset_url": "http://bkrdkl-vb:8080/73",
"draft": false,
"origin_site": "bkrdkl",
"owner": "Basavaraj Karadakal",
"owner_email": "bkrdkl@juniper.net",
"prs": [
    "1102653"
],
"sb_location": "bkrdkl-vb:/home/bkrdkl/workspace/demo/demo1",
"sha": "6b48b2f08d83a20a6270245a6c73fa389d256088",
"submitted": null,
"topic": "test_cset-1102653-10183f17a"
```
Change sets

- cset-repo
- repo1
- repo2

Change-set tracking commit

Change commit

Change commit
bkrdkl@bkrdkl-vb:~/workspace/demo/demol/csets $ sb cset list
Csets local to sandbox
  * LOCAL test_cset-1102653-10183f17a bkrdkl

bkrdkl@bkrdkl-vb:~/workspace/demo/demol/csets $ sb cset dump
{
  "change_id": "Idfe89cab5a05258d0c28537839553f4bd777a10b",
  "changes": {
    "test-repo": {
      "I9fbbcc5a17df3922beful6492a389a49c38eca962": {
        "commit_sha": "ec1894033b91fa432491ee0724790c50b36593e7"
      }
    }
  },
  "created": 1479261640.958241,
  "cset_topic_branch": "cset-test_cset-1102653-10183f17a",
  "cset_url": "http://bkrdkl-vb:8080/73",
  "draft": false,
  "origin_site": "bkrdkl",
  "owner": "Basavaraj Karadakal",
  "owner_email": "bkrdkl@juniper.net",
  "prs": [
    "1102653"
  ],
  "sb_location": "bkrdkl-vb:/home/bkrdkl/workspace/demo/demol",
  "sha": "6b48b2f08d83a20a6270245a6c73fa389d256088",
  "submitted": null,
  "topic": "test_cset-1102653-10183f17a"}
bkrdkl@bkrdkl-vb:~/workspace/demo/demol/csets $ sb cset upload
INFO: Uploading change sets
    test_cset-1102653-10183f17a
INFO: Checking cset dependencies
INFO: Uploading ec1894033b91fa432491ee0724790c50b36593e7 --&gt; test-repo
INFO: Successfully uploaded / updated following URLs
INFO: Uploaded : http://bkrdkl-vb:8080/74
INFO: Trying to set default reviewers...
INFO: Setting review group for http://bkrdkl-vb:8080/74 to test-repo-reviewers

bkrdkl@bkrdkl-vb:~/workspace/demo/demol/csets $ sb cset info
Topic : test_cset-1102653-10183f17a
State : UPLOADED
Cset URL : http://bkrdkl-vb:8080/73
Sandbox : bkrdkl-vb:/home/bkrdkl/workspace/demo/demol
PRs: 1102653 HwIds: 00000 VmmIds: 00000
Changes :
    test-repo
        ec1894033b91fa432491ee0724790c50b36593e7
        Test commit
        http://bkrdkl-vb:8080/#/c/74/1
Implementation

• The tracking repo defines following labels.
  • cset-ready
  • verified
  • code-review
• The cset-ready label is toggled every time changes to the cset are uploaded from the sandbox.
• The toggling of the cset-ready label drives the CI-CD pipeline for the change-set.
• The verified label is only defined in the change-set repo and disabled in all others.
• The CI pipeline verifies only the change-set tracking commit.
Change-sets with no dependency
Change sets with simple dependency
Change-sets with cross / circular dependency
Change-set reviewer plugin.

- Abandons all the changes of a change-set when the tracking cset commit is abandoned.
- Restores all the commits associated with the change-set when the change-set commit is restored.
  - Toggles the cset-ready flag to kick off a new CI cycle.
- Toggles cset-ready flag
  - When an individual change of change-set is restored.
    - After timeout
  - When an individual change of change-set is abandoned.
    - After timeout
- Gives a code-review +1 score to cset tracking commit, when all changes in the cset get +1 (We only use -1 and +1 ) and none of them have merge conflicts.
  - This kicks of other PR workflows.
CI pipeline

• Based on open stack zuul.
• Listens to events only on the change-set tracking repo.
• Build jobs create sandbox and cherry-pick the change-set in the sandbox.
• The CI system votes on the change-set tracking commit.
• Only check pipeline is in production.
• Implementation of gate pipeline is in progress.
  • Zuul to be made cset aware.
• Having a cset simplifies the zuul configuration to a great extent as only one repo is being monitored.
Change-set aware pull

- While pulling changes across all 500+ repos we do sometimes pull in partial changes from change-set.
  - Even though topic submission ensures that the changes are committed almost atomically the order in which we pull the repos can lead to pull us partial changes.
- The wrapper script that pulls the changes across all repos will check to see if there are any partially pulled change-sets.
- If there are partially pulled change-sets, the partially pulled change-sets are made consistent.
# change-set commands

<table>
<thead>
<tr>
<th>sb cset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>create</td>
<td>Create a new change-set.</td>
</tr>
<tr>
<td>activate</td>
<td>Activate a change-set in the sandbox.</td>
</tr>
<tr>
<td>deactivate</td>
<td>Deactivate the change-set.</td>
</tr>
<tr>
<td>info</td>
<td>Get information about change-sets in sandbox</td>
</tr>
<tr>
<td>mv</td>
<td>Move a commit to a change-set.</td>
</tr>
<tr>
<td>list</td>
<td>Query change-sets</td>
</tr>
<tr>
<td>dependencies</td>
<td>Show dependencies of change-sets in the sandbox.</td>
</tr>
<tr>
<td>amend</td>
<td>Amend the PR and smoke test ids associated with a change-set upload.</td>
</tr>
<tr>
<td>cherry-pick</td>
<td>Cherry pick a change-set and it’s dependencies to local sandbox.</td>
</tr>
</tbody>
</table>
Implicit workflow, change-set state transitions

Not applicable with implicit workflow

- sb cset create
- sb cset mv
- git commit

.UPLOADED

.UPLOADED+LOCAL

.MERGED

.ABANDONED

Not applicable with implicit workflow
## What gets uploaded?

<table>
<thead>
<tr>
<th>Active change-set in sandbox</th>
<th>No explicit change-sets specified on command line.</th>
<th>All commits part of the active change-set are uploaded as long as the active change-set does not have a dependency on another change-set, which is not already uploaded.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit change-sets are specified on command line.</td>
<td>All commits belonging to the change-sets are uploaded as long as none of the give change-sets have a dependency which is not already uploaded.</td>
</tr>
<tr>
<td></td>
<td>Only explicit components are specified on command line.</td>
<td>Not Supported</td>
</tr>
<tr>
<td></td>
<td>Change-sets and components are specified.</td>
<td>Not supported.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No active change-set in sandbox</th>
<th>No explicit components or change-sets specified on command line.</th>
<th>There are no existing change-sets that are modified.</th>
<th>All commits in the sandbox are made part of the new change-set and uploaded.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One or more modified change-set in the sandbox, with or with out additional commits that are not associated with any change-sets.</td>
<td>All the modified change-sets are uploaded.</td>
<td></td>
</tr>
<tr>
<td>Explicit change-sets are specified on command line.</td>
<td>All commits belonging to the change-sets are uploaded as long as none of the give change-sets have a dependency which is not already uploaded.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Only explicit components are specified on command line.</td>
<td>All commits in the given component repos are made part of the new change-set and uploaded.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change-sets and components are specified</td>
<td>Not supported.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Thank you