Software issues
Git migration and JIRA transition

Felix Müller, MPP

DEPFET Lab Meeting

August 10, 2016

SVN

SVN: https://belle2.cc.kek.jp/svn/groups/pxdonline/epics/trunk/pxd-testsetup/system-config

PC: ~/system-config

Content: Setting up PC, install script, system environment, network config, README.md

SVN: https://belle2.cc.kek.jp/svn/groups/pxdonline/epics/trunk/pxd-testsetup/epics-userconfig

PC: ~/epics

Content: EPICS Startup scripts (PS, PS Sequence, DHH, config server)

SVN: https://belle2.cc.kek.jp/svn/groups/pxdonline/epics/trunk/css/analysis

PC: ~/cs-studio/analysis

Content: Measurement and anlyze scripts (pedestals, delays, gated-mode, laser scans, …)

SVN: https://belle2.cc.kek.jp/svn/groups/pxdonline/epics/trunk/css/dhh

PC: ~/cs-studio/dhh

Content: Library for analyze and measurement scripts, CS-Studio Widgets, ini-files

SVN: https://belle2.cc.kek.jp/svn/groups/pxdonline/epics/trunk/dhh/dhh_support_sw

PC: ~/dhh/dhh_support_sw

Content: DHH related software, IPBUS, FrameReceiver, pyDepfetReader

PC Structure

SVN Repositories:

Install scripts, DHH Software, Measurement and Analysis scripts

RPM Packages:

EPICS for DHH, PS

Webpage https://sussrv01.ziti.uni-heidelberg.de/~ritzert/PXD/CSS4/:

CS-Studio Snapshot

Install script took care of almost everything except the svn client in CS-Studio

Git migration

Where: stash.desy.de - Server which hosts all the files from the svn repsitories

Port: **7999** (talk to your IT regarding firewalls)

How:

- With a personal account
  - https://confluence.desy.de/display/BI/Belle+II+Registration+Procedure
    - Register at GridKA (fill out form and sent it online & fill out form, get sign from responsible person at your institute (director, …) and send it by mail (nor email) to GridKa (IMPORTANT: use your personal computer because only with this webbrowser you use to fill out the form, you will be able to get the certificate)
    - Receive email and follow instructions
    - Install GridKa certificate in your webbrowser
    - Get an DESY account by identifying yourself with GridKa
      - https://belle2-request.desy.de/
- With a PC account (for test setup PCs)
  - …

Git migration

- With a PC account (for test setup PCs)

EITHER:

- A common private KEY (which only has read access to 3 git repositories) is located in a git repository (access it with your personal desy account)
  - $ git clone ssh://git@stash.desy.de:7999/b2g/pxd_sc_testsetup.git

  This key is required to run the install script – all the required software is downloaded from the git repositories

  This key has only permissions to read, i.e. committing new code is not allowed

OR:

- Every institute (Bonn, Goettingen, HLL, MPP, …) has a personal key which give read&write access to the 3 repositories (for the testsetup)
- ONLY 1 KEY per institute
- Share the key to all your test setups within the institute
- Create a key and send it (the public key) to me (fmu@mpp.mpg.de)
  - I will grant you access to the 3 repositories

The common key can be exchanged by the private key at any time

How to create the keys?

Go to one of the testsetups:

```
$ ssh-keygen –t rsa
```

Name: “desystash_INSTITUTE”, e.g. “desystash_MPP”

It will create “desystash_<institute>” and “desystash_<institute>.pub” (private and public key)

```
$ vim ~.ssh/config
```

```
Host stash.desy.de
     Hostname stash.desy.de
     Port 7999
     IdentityFile ~/.ssh/stashkey_MPP
```

configures which file/key will be used

$ send the “desystash_<institute>.pub” to fmu@mpp.mpg.de

Having received the confirmation you can test the connection by:

```
$ git ls-remote ssh://git@stash.desy.de:7999/b2g/pxd_sc_css.git
```

SVN <-> git

Major differences for us:

SVN:

Every file within a svn repository could have another revision

`svn commit` commits all changes to svn repository (by default all)

git:

All files/ folders have the same revision

`git add` adds file which should be commited (by default None)

`git commit --author “MYSUERNAME“` commits to the local repository

`git push` commits to desy.stash.de

create a .gitconfig (due to the shared institute key of the PCs, it is impossible to see the author of the commitments if one forgets --author “MYUSERNAME”)

```
$ vim .gitconfig
    [user]
    └──────email = chk@hll.mpg.de
    └──────name = HLL Munich (pxdtest6)
```

New client for CS-Studio (egit)

Read manual: https://stash.desy.de/projects/B2G/repos/pxd_sc_testsetup/browse/system-config

git repositories

ssh://git@stash.desy.de:7999/b2g/pxd_sc_css.git

more or less ~/cs-studio

ssh://git@stash.desy.de:7999/b2g/pxd_sc_dhh.git

more of less ~/dhh

ssh://git@stash.desy.de:7999/b2g/pxd_sc_testsetup.git

~/epics

~/system-config

We cannot check out subfolders:

we checkout `pxd_sc_testsetup.git` and link `~/epics` to `~/pxd_sc_testsetup/epics` and `~/system-config` to `~/ system-config`

is done automatically in the install script

Install script (of course only for SL7):

- `install_S7_git.sh`
- `set_environment_SL7_git.sh`

git repositories

ssh://git@stash.desy.de:7999/b2g/pxd_sc_css.git

ssh://git@stash.desy.de:7999/b2g/pxd_sc_dhh.git

ssh://git@stash.desy.de:7999/b2g/pxd_sc_testsetup.git

In pxd_sc_css there will be a lab_framework

This lab_framework contains the cleanup of the code

- `basics` (plot a frame, no mapping etc)
- `calibrations` (perform all the measurements and analysis like gated-mode, delays, etc)
- `devices` (control commercial devices, like PowerSupplies, PulseGenerator, Chiller, SMU)
- `lib` (libraries like epics_utils, dhp_utils, functions that will be used in multiple measurement and analysis scripts)
- `testbeam_2016_04` (special scripts for the last testbeam, inlcudes DHC)

Philipp Leitl takes care of the code, will operate and commit to desy stash

Redmine to JIRA

Due to the transition from KEK to DESY redmine also needs to move

The new system is called JIRA

Unfortunately, it does not provide subprojects. Hence, there are 3 possibilities for the migration (by Nils Braun)

(1) Create different projects for each sub-projects. To remain the hierarchy, we could invent a sensible naming schema or use project categories for this.

(2) Create one top-level project and import all issues into the same project.

(3) The same as (2), but add a label to each issue from a subprojects, saying from which subproject it came from. Later, it is possible to filter/group/sort by these labels easily, so the hierarchy would remain.

Test the new system by (Nils Braun):

```
ssh  -L 13000:at-stage-05:8080 <user>@pal -o "ProxyCommand ssh
   <user>@bastion.desy.de -W %h:%p"
```

Please fill in <user> with your DESY account. You have to type in your password twice. Then access localhost:13000 in your browser. You can log in with

```
testJIRA
migrationJIRA
```