

SPRINT 2 – Mytestbed portal implementation notes

Sprint 2 changes have been deployed to pdev and ptst environments. It is preferable to use ptst environment for user acceptance testing as the data in that environment is clean. We have not yet deployed these changes to the production environment and will not do so until requested.

Test environment

<https://ptst.mytestbed.net> (login: rodney.berriman@nicta.com.au/rodney)

We cleaned out ptst projects and issues due to conflict in old data so I have set up one new project:

name: Incontext test id: test1

Functionality implemented this Sprint

1. Enhancements to Sprint 1

1. All 3 environments are now using SSL self-signed certificates and should redirect from http to https. CD was emailed to Rodney with details of certificate authorities.
2. Backups put in place for all 3 environments with info on InContext wiki:
<https://pm.incontext.com.au/wiki/nicta/Backups>
3. Sub tabs “Issues” and “New issues” renamed as “Experiments” and “New experiment”
4. Log text from XML results appears as “more” link for Script runs. This expands into scrolling text box.
5. Attribute text from XML results has been formatted as table.

2. Integration between Git and Redmine

1. When setting up a new project or updating the project settings, there is a check box “Create Git repository”, if this option is ticked, a Git repository will automatically be set up for this project. After creation, you will see a field with the repository path for that project.
2. For a project that has been created with this option, when you create a new tracker type of “script” in Redmine, it will automatically create an empty file in GIT and generate the name and path.
3. The identifier and path for the script are generated automatically from the tracker number which is unique in the Redmine portal. The identifier and script path cannot be edited by the user.

3. Accessing and editing script source in Redmine

1. To access editing of source code, from a “script” tracker in Redmine click on the link in the Script path field.
2. This should open new page with Ruby aware text editor. It will be empty initially.
3. Type in your script or changes to existing script; type in Commit message and hit Commit button. This should increment the number of commits by 1; and add a new version to the drop list of versions.
4. You can navigate to different version by selecting the version from drop list and hitting Load button. Version drop-lists show incremental version number, commit message, user name and time stamp of commit.
5. You can edit any version of the script and commit it and this will create a new version as the latest version.
6. From “Script run” trackers in Redmine you will get a link to the particular version of the script that was run which will open in the same script editor.
7. When manually creating a new “script run” in Redmine you can select the version of the script to be run from the drop list.
8. An error will be thrown if there are no changes to script and user attempts to commit (although note that if original version was committed by XML post and then user attempts to re-commit from Windows environment with no changes to script, this will be committed as Windows inserts end of line character Ctrl M in script).

4. Modifications to posting of XML results:

1. The Curl command has changed slightly with the move to https. You may post to Redmine using following command:

```
curl --insecure -i -X POST -d "`cat path_to_xml_file`" -u uname:password https://redmine_url/results  
eg. curl --insecure -i -X POST -d "`cat test.xml`" -u admin:eegh9ieK https://ptst.mytestbed.net/results
```

2. The load process checks the <project> name and attempts to match to an existing project id in the Redmine portal based on the project identifier field. If it does not match then the load fails.
3. If there is a match on project, the load process then checks for a matching <id> against the new “Identifier” field in Redmine for that project.
 - If match exists, then the <source> text in XML is diffed against the latest version of the source script in Git, if it is not the latest version then a new version of the script will be added to that script in Git and the script run results linked to the new version.
 - If a match does not exist for <id> in that project, a new script tracker will be added to the project and the run results will be added to this new script as a script run. The <source> will be added to Redmine as the initial version of the script in Git, the file name and path for Git will use the <id> from the XML file.
4. With successful load of results to Redmine, the status of the Script run is set to “Done”.
5. The commit message for script versions added by the system are generated with date and time from the http post.

5. Constraint on editing script run results

1. Once the status of a script run is set to “Done” (eg. After receiving results); the results fields can no longer be edited by user, however the subject, description and status fields may still be changed.

6. Configuration options

1. There is a new permissions option for roles that governs whether or not different roles can access script editing. This can be configured via the Administration/Roles and Permissions page and is called “Access experiment scripts”. We have configured the test environment so that role of “Developer” has access to scripts but this is configurable by NICTA.
2. We have created a new status of “Failed” and configured the status work-flow so that issues can change from “Done” to “Failed” only. This configuration is handled through Administration/Workflow page

7. Admin access to Git

1. NICTA administrative users may be set up to access the Git repositories via SSH. If access is required, please contact InContext for user accounts.

Limits on functionality

1. Functionality to allow linking through to common library scripts was put out of scope for this Sprint by agreement as NICTA are in the process of determining how best to handle this.
2. Customised functionality to copy initial script from repository has not been implemented, as user has ability to copy and paste from existing script in Redmine.
3. Likewise customised functionality to select project when creating a new script has not been implemented, as Redmine natively works by selecting project first and then creating a new issue.
4. The inbuilt Redmine functionality configured in Sprint 1 that allows browsing the Git repository conflicts with new Git integration and cannot currently be used.