DRAFT STAGE. Subject to discussion and change

Tango svn to git migration

Authors: cpascual@cells.es, gjover@cells.es, gcuni@cells.es, coutinho@esrf.fr, reynald.bourtembourg@esrf.fr, antonio.milan_otero@maxiv.lu.se, vincent.hardion@maxiv.lu.se

This document is a proposal to migrate the Tango control system and device servers from the source forge SVN repositories to a git based system.

Motivation

A quick survey of the web will show find extensive literature explaining why git is better than SVN. But why should Tango migrate to Git? Would Tango be able to benefit from the advantages provided by git at all? The authors of this proposal are firmly convinced that the Tango project will profit from a git based version control system.

Currently, contributing to any of the Tango core projects is difficult and error prone. Since it is virtually impossible to manage all possible contributors as members of the tango-cs source forge (SF) project, in general, a contribution implies attaching a patch file to a ticket which needs to be downloaded and verified by an integrator, which will commit the change with his identity to SVN. This last step poses a problem because the project just lost track of who actually did the modification.

With git and friends, the repository can be forked by the contributer and a pull request (PR) can be made. After being accepted by the integrator, the information about both the contributer(s) and integrator will be stored as well in git. Another advantage is that the PR can be discussed openly and developers can even comment on specific lines of code, making code reviews a natural part of the development process.

As part of their development strategy, many of the institutes using tango prefer to stabilize their tango major version for relatively long periods. Tango bug fixes are usually applied on the current development branch. The development cost of backporting the same fix to older tango versions is forbiddenly high with SVN. As a consequence, institutes that are not able to follow the latest development version of Tango cannot profit possibly important bug fixes that could have been easily made with a git cherry-pick.

Platform

A Tango organization in github will be used to host all the Tango-related repositories.

Each different svn-subproject (a subdirectory in the svn repository that has trunk/tags directories) will be migrated to an independent git repository in the tango-controls organization.

Each project (aka github repository) will have a separate lifecycle: git repository, issues, users and

teams, releases. Below you will find more detailed proposals for the organization of the projects.

Some projects may choose to be hosted elsewhere (although this is discouraged), in which case they should register their repository in the Tango organization so that they can be linked (and possibly even mirrored) from the Tango organization.

Workflow

Each project within the Tango organization may choose its own workflow (adapted to the reality of its developer group, release policies, etc), but the Tango organization makes some recommendations in order to encourage the participation and promote quality.

These guidelines should be applicable even for the smallest projects (even with a single developer and no formal release policy), and their optional parts may be adopted (if required) by larger projects (e.g., with several developers, spread amongst different locations, and supporting multiple releases simultaneously).

1. In its most basic form, we adopt the githubflow : the default branch is `master`, which is always in a deployable state (i.e. must never be broken); work on new features is done using temporary feature branches (which may live in the same repo or in a fork)
2. Reporting issues: use github issues
3. Code contributions: use Pull requests. Pull requests can be associated with issues. Trivial fixes can even be done from the web (a temporary branch is created automatically to use the Pull request infrastructure)
4. Code review: most projects (all except single-developer ones) **should** only allow commits to the master branch after peer review. This can be enforced by the convention that all commits to master must be done via a Pull request and the pull request approved by a person different from the author of the commit.
5. Releases (named versions):
    1. Simple projects may be ok with just tagging certain (ideally, all) commits done to the master branch with a version number. With this system, only one release is actively maintained simultaneously.
    2. More complex projects may require to simultaneously maintain more than one release (e.g. Tango may choose to support bugfixes in Tango9 even after Tango10 is released). In this case, releases may be done on release branches starting from master (see APPENDIX I for an example)
6. Semantic versioning is recommended.
7. **Public** automatic testing/continuous integration (e.g., via Travis) is recommended
8. The main development **should** be done on the tango-controls hosted project (as opposed to using a private organization project and just pushing to the tango-controls repo from time to time). This allows for public visibility of the latest development and issues and encourages sharing and reuse. If a given organization needs special tweaks or has particular release/testing cycles, the recommendation is that the organization forks from the "canonical" repo

Tickets

Tickets only exist in the tango-cs Sourceforge project. There is no ticket in the tango-ds project.

A slightly modified version of the following script can be used to migrate automatically tickets from

a Sourceforge project to a github repository:https://github.com/cmungall/gosf2github.

This script has been modified in order to leave the unassigned tickets unassigned after the migration. It is using the GitHub API for importing issues.

The main difficulty of the migration will be to split the Sourceforge tickets and assign them to the different newly created Github repositories.

The ticket migration will be done project per project (from a tango-cs subsystem to a corresponding Github repository).

The requirements for the tickets migration of a specific subsystem from tango-cs are the following: * The GitHub repository for the project must have been created * The list of tickets to be migrated must be communicated to the person in charge of the migration. This can be specified using complex expressions like “All the tickets belonging to Sourceforge ticket category Astor and Starter and all the tickets having labels Astor and starter device server or Starter and tickets #123 and #345, except all the tickets having label Astor” for instance. * The persons currently having tickets from the previous ticket list assigned to them on Sourceforge must have a GitHub account and be registered as collaborator of the newly created Github project with admin permissions before the migration. This will ensure the newly created GitHub issues will be automatically assigned to the correct person during the migration. Some tools have been developed to get the list of assignees from a list of tickets. * A generic github account like the following will be used to migrate the tickets: https://github.com/tango-tickets-migrator. This account will need to be registered as collaborator of each target repository with admin rights.

Limitations

All issues and comments will appear to originate from the tango-tickets-migrator user but the original author will be specified in the comments and issue description.

If the original assignee does not have a github account or is not registered as collaborator on the github repository, the ticket will be assigned to the tango-ticket-migrator user.

The tickets assigned to people no longer involved in the Tango community will be assigned to the tango-tickets-migrator user. A link to the original ticket is automatically added in the description of each migrated github issue.

The migrating script does not support attachments. It might be possible to add a link to the original attachment but this will require to hack the migrating script. Since there is already a link to the original ticket in the description of the imported issues, the steering committee should decide whether having this link would be sufficient or we should put more effort to find a way to import the attachments too.

Example

The results of the first tests are available on the following link:

https://github.com/tango-controls-migration-tests/sf2github/issues

In this first test, all the tickets from tango-cs (older than 25th May 2016) have been migrated to this sf2github single repository. Only Emmanuel Taurel, Tiago Coutinho and Reynald Bourtembourg were configured as collaborators of this repository when the script was executed. This is why all the Sourceforge tickets which are assigned to someone different than Emmanuel, Tiago or Reynald have been assigned to the tango-tickets-migrator user.

Todo

pyGitHub.py script should be improved to actually do the real migration from svn to git for a given project using the GitHub source import API.

Risks

Only the subversion directories respecting the TTB pattern (Containing trunk, tags and branches directories) can be migrated smoothly to git with the available tools. This means that the files in subversion which do not have a directory named trunk, tags or branches in their parent directories (for instance: files in archiving/hdb++ directory) will need a special treatment to ensure they are also migrated. This treatment might vary from one project to another. For some projects, it might be necessary to restructure the subversion directories before the migration. For some other projects, it might be necessary to create a new separate git repository for these directories, which we might need to merge to another git repository afterwards.

The list of files from tango-ds which are not under the TTB pattern (which do not have a directory named trunk, branches or tags in their parent directories) are listed in the following gist:

https://gist.github.com/bourtemb/49feb682806877177eaff7953869b213.

The ones from tango-cs are listed in the following gist:

https://gist.github.com/bourtemb/c3d3c89d5b6a41ac6a1b475d759c9b6d

Another risk is linked to the fact that the external definitions feature from subversion was used in some directories under the distrib directory. We need to find the right way to handle this special case.

Roadmap

The following roadmap applies from Day0 (day from which the green-light is given). Note that it can be applied first to a subgroup of projects and then repeated for a different subgroup. We propose to start by applying this to tango-cs (it could even be split in subgroups inside tango-cs, but that would complicate the management of the lock on commits and tickets in SF.net).

- Day 0: Compile the list of svn subprojects (using svn_find_subrepos.sh). For example(List for tango-cs and List for tango-ds)
- Day 0: Create map(s) from the above paths to proposed github repo names and metadata (by

default, the github name can be the basename of each svn-subrepo path and the rest can be used as additional info for the description)
- Day 0 to Day 14: Circulation of the map(s) among Tango developers for manual tweaking. The changes can be managed as pull requests. Identify projects "with special needs" (e.g. git repos that should merge more than one svn subrepo) and remove them from the auto-migration list.
- Day 0 to Day 15: Identification of the set of tickets to be migrated for each project.
- Day 15: Lock commits and ticket creation on SF.net
- Day 15: Automatic bulk-migration of repos using pyGitHub.py
- Day 16: Semi-Automatic migration of tickets of auto-created repos
- Day 16 to ??: Manual migration of "special needs" projects

Questions to be answered by the Tango Steering Committee:

- Who are the nominated members from each institute to be appointed as "admins" for the core repos (at least 2/institute)?
- Is the Roadmap accepted?
  - Which are the subgroups for implementation of the roadmap?
  - Which is the "Day0" for each subgroup?

References

APPENDIX I: Workflow example maintaining multiple releases

Let's consider the Tango C++ core:

- Let's assume that the latest commit in the master branch is tagged, for example, as "9.2.1".
- Now someone starts working on Tango10 and starts a feature branch (in the same repo or in a personal fork).
- Meanwhile some critical bug is discovered and fixed for Tango9, which implies a pull request and a new commit to master tagged as "9.2.2".
- When the work on the Tango 10 feature branch is finished, a pull request is submitted and eventually accepted. Now the latest commit in master will be tagged "10.0.0".
- After that, a new critical bug affecting **both** Tango9 and Tango10 is discovered and a patch provided (for simplicity let's assume that the patch works on both). The Tango team decides that it is worth supporting the Tango9 users, so:
  - a new branch called "9.x" is branched off the "9.2.2" commit, the patch is committed to it (via a pull request) and the commit tagged as "9.2.3"
  - The patch is also applied on master (via a pull request) and the latest master commit is then tagged "10.0.1"

APPENDIX II: Tickets migration technical details

Tickets only exist in the tango-cs Sourceforge project. There is no ticket in the tango-ds project.

The following script can be used to migrate automatically tickets from a Sourceforge project to a github repository: https://github.com/cmungall/gosf2github.

In order to work as expected, the script requires the following:

- The tickets must have been exported from Sourceforge using the admin page: https://sourceforge.net/p/tango-cs/admin/export.

In the exported files, 2 files are important:

- feature-requests.json containing all the tango-cs feature requests tickets
- bugs.json containing all the tango-cs bugs tickets

A new json file containing only the bugs tickets relevant to the new github repository will have to be created from the bugs.json file.

Similarly a new json file containing only the feature requests tickets relevant to the new github repository will have to be created from the feature-request.json file.

- A JSON file containing the mapping of Sourceforge usernames to GitHub usernames. So we might need to ensure that all the important users have created a Github account.
- An OAuth token has to be given to the script. It is possible to get one here:

https://github.com/settings/tokens

Note that all tickets and issues will appear to originate from the user that generates the token.

Important: make sure the token has the public_repo scope.

- A JSON file containing the list of collaborators for this repository.

Please note that this is required in order to be able to automatically assign github issues to the correct person.

This implies that the list of collaborators for the github repository should be as complete as possible before the tickets migration and should ideally contain all the persons being assigned to the sourceforge tickets we want to migrate.

This file can be generated with the following command:

curl -H "Authorization: token *TOKEN*" https://api.github.com/repos/tango-controls/*theRepository* /collaborators > sf-test-collab.json

- A generic github account like the following will be used to migrate the tickets: https://github.com/tango-tickets-migrator

This generic user must be part of the list of collaborators for each github target repository with admin rights.

Note that all issues and comments will appear to originate from the tango-tickets-migrator user but the original author will be specified in the comments and issue description.

If the original assignee does not have a github account or is not registered as collaborator on the github repository, the ticket will be assigned to the tango-ticket-migrator user.

Please note that the script in its original version assigns tickets currently not assigned on Sourceforge to the tango-ticket-migrator user. The script has been improved (locally for the moment) so that it keeps unassigned SF tickets unassigned in github.

A link to the original ticket is automatically added in the description of each migrated github issue.

The migrating script does not support attachments. It might be possible to add a link to the original attachment but this will require to hack the migrating script.

The results of the first tests are available on the following link:

https://github.com/tango-controls-migration-tests/sf2github/issues

In this first test, all the tickets from tango-cs have been migrated to this sf2github single repository. Only Emmanuel Taurel, Tiago Coutinho and Reynald Bourtembourg were configured as collaborators of this repository when the script was executed. This is why all the Sourceforge tickets which are assigned to someone different than Emmanuel, Tiago or Reynald have been assigned to the tango-tickets-migrator user.

Here is the list of all the persons having a tango-cs sourceforge ticket assigned to them (as of 25/05/2016):

Sourceforge login	Github login	Full name	e-mail address
pascal_verdier	Pascal-Verdier	Pascal Verdier	verdier@esrf.fr
nleclercq		Nicolas Leclercq	nicolas.leclercq@synchrotron-soleil.fr
taurel	taurel	Emmanuel Taurel	taurel@esrf.fr
ingvord		Igor Khorkhriakov	Igor.Khokhriakov@hzg.de
trogucki	nobody	Thomas Rogucki	No longer in Tango community
poncet		Faranguiss Poncet	poncet@esrf.fr
klenov		Ludmila Klenov	ludmila.klenov@synchrotron-soleil.fr
bourtemb	bourtemb	Reynald Bourtembourg	bourtemb@esrf.fr
candelrg		Raphaël Girardot	raphael.girardot@synchrotron-soleil.fr
andy_gotz	andygotz	Andy Gotz	andy.gotz@esrf.fr
abeilleg		Gwenaelle Abeille	gwenaelle.abeille@synchrotron-soleil.fr
zreszela	reszelaz	Zbigniew Reszela	zreszela@cells.es
piccaf		Frédéric Picca	frederic-emmanuel.picca@synchrotron-soleil.fr
tiagocoutinho	tiagocoutinho	Tiago Coutinho	tiago.coutinho@esrf.fr
jlpons		Jean-Luc Pons	pons@esrf.fr
ollupac	nobody	ollupaC De La Pradera?	No longer in Tango community
cpascual	cpascual	Carlos Pascual	cpascual@cells.es

The tickets assigned to people no longer involved in the Tango community will be assigned to the tango-tickets-migrator user.

Tickets labels

The migrating script will associate the following labels to the migrated tickets:

- sourceforge
- Auto-migrated

The *bug* label will be associated to tickets migrated from sourceforge bug tickets.

The *enhancement* label will be associated to tickets migrated from sourceforge feature request tickets.

The script will also set the "high priority" label for all tickets having a sourceforge priority > 5 and the "low priority" label to tickets having a sourceforge priority < 5.

The main difficulty of the migration will be to split the sourceforge tickets and assign them to the different newly created github repositories.

One way to do that is to split them using the labels or categories. But first we need to know what will be the names of the newly created repositories.

Here is the list of sourceforge labels:

Sourceforge label	GitHub repository	Number of open tickets	Number of closed tickets	Other related labels
Archiving		0	3	
Astor and starter device server		0	16	
Astor		1	0	JTangoServer
ATK toolkit		3	51	
ATK		10	4	
Atkpanel application		1	19	
Attribut alias order		0	1	(Jive)
C++ api and idl files		3	141	
C++ API		0	1	
Centos		0	1	PyTango 8.1.6/ PyTango Trunk/ Tango9.1.0
common_target.opt		0	1	Tango9.1.0
Database server		3	19	
db schema		1	0	mariadb mysql
debian		1	0	
Device Property		0	1	HLAPI
Device tree application		0	13	
Device		0	1	HLAPI

Sourceforge label	GitHub repository	Number of open tickets	Number of closed tickets	Other related labels
DeviceProxy		1	0	PyTango lock segfault
Event API		1	0	TangORB
event		1	0	segfault (PyTango)
Group		0	1	Python binding
HLAPI		0	1	
igor-pro-bindings		1	0	
Incompatibility between ZMQ releases between client and server		0	1	(C++ API)
Interface Improvements (example)		2	4	
Java api		1	12	
Jive		2	35	
JTangoServer		6	2	
jzmq		0	1	
Kernel feature		4	1	Several different projects
lock		1	0	PyTango
Log for Tango		0	2	
Log viewer application		1	2	
Makefile		1	0	(Pogo)
mariadb		1	0	db schema mysql
mysql		1	0	db schema mariadb
Packaging		0	6	
patch		2	0	(qgraphicsplot) igor-pro-bindings
pipes		0	1	JTangoServer
Pogo		0	20	
push asynch		1	0	(PyTango, PyTango 8.16)
PyTango array		1	0	
PyTango Trunk		0	1	Centos PyTango 8.1.6PyTango TrunkTango 9.1.0
PyTango		1	0	
PyTango 8.0.4		0	1	
PyTango 8.1.6		0	1	
Python binding		0	109	
python_binding		1	26	
Sardana		4	42	
segfault		2	0	(PyTango)
Source distribution		1	25	(Packaging)
Starter		1	0	Astor JTangoServer
tango.opt		0	1	Tango 9.1.0
Tango 9.1.0		0	2	

Sourceforge label	GitHub repository	Number of open tickets	Number of closed tickets	Other related labels
Tango9.1.0		0	1	common_target.opt
Tango9.2.0		1	0	(common_target.opt)
TangORB		4	0	
Tau		0	21	(Taurus)
taurus		5	42	
zmq events		0	2	(C++ API)
zmq		1	0	TangORB

The best would be to ask the people from each project to give us a list of the tickets they want to transfer for their project.

For example:

"For Taurus, we want all the tickets having the labels *Tau* or *Taurus* in it + tickets belonging to *Taurus* category + tickets #4367 and #54389".

A python script which can extract all the tickets having specific labels and/or belonging to a specific category and/or having specific ticket numbers has been developed. This script is able to produce a new JSON file containing only the tickets having specific labels and/or specific ticket numbers and/or tickets belonging to a certain category. This newly created JSON file can then be given as parameter to the ticket migration script.

A complementary python script has been developed in order to be able to generate new json files containing all the tickets listed in another json file except the tickets having specified labels, categories and/or ticket numbers. Using these 2 python scripts, it will be possible to generate json files based on requests like the following for instance: *All the tickets having "Kernel feature" label, except the tickets belonging to "PyTango" category and excluding ticket #1234.*

Tickets categories

Here is the list of tango-cs Sourceforge ticket categories:

Sourceforge category	Github repository
Archiving	
Astor and Starter	
Atkpanel	
ATK toolkit	
C++ API	
C++_API	
Database server	
Java API	
Jive	
Log4Tango	

Sourceforge category	Github repository
Miscellaneous	Tickets #756,751,722,716,689,670,448,447 & 437
Pogo	
PyTango	
Sardana	
Source distribution	
Taurus	

Please note that some tickets haven't been assigned to any category. Some tickets have labels assigned but no category and vice versa.