How to add new functionality to GEMSFIT(GUI)

Starting from GEMSFIT svn revision 72 **by DK, DM & FFH 10.09.2012 (Draft 1)**

1. **Define a development task**

Prepare the algorithms to add to GEMSFIT and/or GEMSFIT-GUI, and additions/modifications to data structures (and perhaps access methods in TNode class in GEMS3K)

Give the development task a short ID (e.g. "devExtFit"). In the following, replace 'DK' with your user login name.

2. **Create development branches in svn repository** (main developer for the task)

2a In the GEMSFIT svn repository, create a new branch "ExtModel" from trunk using the svn copy command (linux):

```
$    svn copy svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/trunk \
          svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/branches/devExtFit \
          -m "DK created a new development branch devExtFit of GEMSFIT trunk"
```

or use the equivalent operation in your svn client of choice.

2b In the GEMS3K svn repository, if appropriate, create a new branch "ExtFit" from trunk using the svn copy command

```
$    svn copy svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/trunk \
          svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/branches/devExtFit \
          -m "DK created a new development branch devExtFit of GEMSTANDALONE trunk"
```

or use the equivalent operation in your svn client.

3. **Checkout new development branches from the svn repository**

3a Every developer involved in this development task must create a local directory for this branch, e.g.

```
$    cd /home/DK/DevGEMSFIT
$    mkdir devExtFit
$    cd devExtFit
```

and checkout new branches to her local PC:

Checkout GEMSFIT:

3b

```
$    mkdir GFIT
$    svn checkout svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/branches/devExtFit GFIT
```

Checkout GEMS3K development branch (if related changes in TNode access methods must be made):

3c

```
$    mkdir standalone
$    svn checkout svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/branches/devExtFit standalone
```

3d Alternatively, checkout GEMS3K trunk (if no changes in GEMS3K are necessary):

```
$    mkdir standalone
$    svn checkout svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/trunk standalone
```

Build and test the GEMSFIT and optionally the GEMSFIT-GUI project. We strongly recommend using QtSDK 1.2.1 (http:\\qt.nokia.com) under LGPL (installed in your home directory). Set there the shadow build in `gemsfit-build` subfolder for the GEMSFIT project; in `gemsfits-build` for the GEMSFITS project (uses pre-compiled libgems3k); and in `gemsfitgui-build` subfolder for the GEMSFT-GUI project. These subfolders (including test data) are provided in the svn repository.

4. **Change the source code in the branch working copy**

Do your changes in this working copy of the devExtFit branches (of both GEMSFIT(GUI) and GEMS3K).

When committing changes to svn repository, mark every commit with your initials and the branch ID, e.g.

```
$    cd GFIT
$    svn commit -m "DK devExtFit: Implemented extension A in the GEMSFIT task data structure."
```

5. **Sync merge the development branch with the trunk**

You know that other people have made relevant bugfixes or extensions to the trunk. To make sure that your development in ExtFit branch(es) are consistent with those changes, first make sure that your working copy has no uncommitted local changes by executing once more the usual svn work cycle:

5a

```
$    cd GFIT
$    svn update
$    svn commit -m "DK devExtFit: Preparing for sync merge with the trunk"
$    svn status
```

5b Then perform the sync merge:

```
$    cd GFIT
$    svn merge svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/trunk
```

5c If there were independent changes in the GEMS3K trunk then perform the same procedure for GEMS3K branch:

```
$    cd ../standalone
$    svn update
$    svn commit -m "DK devExtFit: Preparing for sync merge with the trunk"
$    svn status
$    svn merge svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/trunk
$    cd ../GFIT
```

Build your project(s) and test them to make sure that your changes to the branch work together with the later changes of others to the trunk.

6. **Documentation, benchmarking and testing**

   When the development in the devExtFit task reaches maturity and the extension of the code are up and running, it's time for benchmarking and testing. The initial series of tests in the /tests/cases subdirectory came with the originally copied trunk variants. Run the tests and compare results with the results provided in /tests/results subfolder. Resolve discrepances, if any (this may involve other developers and take time).

   Add test cases related to newly-developed functionality to /tests/cases; run the test cases, verify results, and add them to the /tests /results; make sure that test cases and results are added (recursively) to branch(es) in svn repository.

   Extend documentation files (or add new) to the /docs subdirectory, add new files there to svn repository.
   Add descriptions of new tests to 'Test Book' on GEMS Wiki or on GEMS3K and GEMSFIT web pages (as appropriate).

7. **Re-integrate your extended code functionality into the trunk**

   Now you feel that it is time to add your new functionality, tests and verified results to the main trunk(s).
   For that, first sync merge your development branch with the trunk again, as described in section 5. Build and test the code, update test results files, and commit:

```
$    svn commit -m "DK devExtFit: Final sync merge with the trunk before re-integration"
$    svn status
```

Now you need a clean working copy of the trunk:

```
$    cd /home/DK/DevGEMSFIT
$    mkdir trunk
$    cd trunk
7a $    mkdir GFIT
$    svn checkout svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/trunk GFIT
```

If you also have a development branch of GEMS3K, check-out its trunk:

```
$    mkdir standalone
$    svn checkout svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/trunk standalone
```

7b Now make sure that the trunk working copy is up to date, and re-integrate your development branch into the trunk working copy:

```
$    cd /home/DK/DevGEMSFIT/trunk/GFIT
$    svn update
$    svn merge --reintegrate \
         svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/branches/devExtFit
```

Do also the same with the trunk working copy of GEMS3K, if appropriate:

```
$    cd ../standalone
$    svn update
$    svn merge --reintegrate \
         svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/branches/devExtFit
$    cd ..
```

Build, test, and verify the re-integrated trunk versions of the codes.

7c When everything is ready, commit the new trunk version(s):

```
$    cd GFIT
$    svn commit -m "DK Merged devExtFit branch into trunk"
$    cd ../standalone
$    svn commit -m "DK Merged devExtFit branch into trunk"
$    cd ..
```

Note that, after re-integration, the development branch is not usable anymore; hence, it is recommended to delete it (the svn history will be retained though):

```
$    cd GFIT
$    svn delete svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/branches/devExtFit \
         -m "Removed the devExtFit development branch after re-integration to GEMSFIT trunk"
7d
$    cd ../standalone
$    svn delete svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/branches/devExtFit \
         -m "Removed the devExtFit development branch after re-integration to GEMS3K trunk"
```

If any further development (even in the devExtFit area) needs to be done, a new development branch (even with the same name devExtFit) has to be done as a copy of the trunk, and the whole development cycle repeated, as described in sections 2 to 5.

7e **Inform** (e.g. by e-mail) all co-developers of the entire GEMS project that a new version of the trunk has been made; eventually launch an automated testing sequence. This is especially important when extensions in GEMS3K kernel have been made.

8. **Create release (distribution) tag (admins/leaders only)**

   When the new trunk is ready, you may wish to keep its 'frozen state' that corresponds to a distribution/release code version. This can be achieved by copying the current trunk into the /tags section of the SVN repository. Make sure that all collaborators are aware that any commits into tagged directories are prohibited.

8a To tag the current trunk for a distribution version of GEMSFIT, run

```
$    svn copy svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/trunk \
         svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSFIT/tags/Stable-v-1.1.0 \
         -m "Tagged a 1.1.0 release of GEMSFIT(GUI) code"
```

To tag also the current trunk for a distribution version of GEMS3K, run

```
$    svn copy svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/trunk \
         svn+ssh://DK@savannah01.psi.ch/repos/gems/GEMSTANDALONE/tags/Stable-v-3.2.0 \
         -m "Tagged a 3.2.0 release of GEMS3K code"
```

8b To build and distribute the release, checkout working copies of those tags to your local PC, build the code, and prepare the installer file.

9. **Announcement**

Inform the community that a new release is ready, providing summary of the new functionality e.g. on GEMS wiki or web pages.

10. **Troubleshooting**

In case of svn errors while performing operations described in sections 2 to 8, to resolve the problems, please refer to the svn "red beans" book, http://svnbook.red-bean.com/.