Protocol 17. Meeting Di, 13.07.11 14.00 – 15.00

Participants: Ludwig Hechler, Solveigh Matthies, Harald Bräuning, Jutta Fitzek, Raphael Müller, Christoph Handel, Alexander Schwinn(Protokoll)

1 FESA3 SVN Structures

We talked about pro's/ contra's of the current svn-structures v.s. a new suggestion. Finally we came up with the following new structures:

svn:/fesa/framework	/[t/b/t]	/fesa-core			
		/fesa-gsi			
svn:/fesa/device	/driver	/BEL			
		/SD			
		/HF			
		/[GroupName]	/[Drivername]	/[t/b/t]	/*
	/deploy-unit	/[GroupName]	/[DeployUnitName]	/[t/b/t]	/*_prj /*
	/class	/[GroupName]	/[ClassName]	/[t/b/t]	/*_prj /*

[t/b/t] = [trunk/branch/tag]

Reasons for these new SVN - structures:

- [t/b/t] was moved from top level to class/driver/DeployUnit - level in order to prevent tags/branches of the whole structure. Class/driver/deploy-unit are treated as independant artifacts. Each of them can be independently developed and released. So each of them gets an independant [t/b/t]
- [t/b/t] of the framework can stay on top, since fesa-core and fesa-gsi are closely interleaved/depend on each other. They need to be versioned and released at the same time.
- fesa cannot be stored as subfolder of /BEL, since everybody @GSI (with a valid acc-network-login) has to have read + write access for classes, drivers and deployment units.
- "fesa-app" was renamed to "device" since "application" could be missinterpreted as GUI
- BEL/HF/SD/[GroupName] has to stay to provide an easy way to later add write-protection per group.
- It is not necesarry to tag DeployUnit's together with the associated classes, since during compilation and linking of the DeployUnit, the used class-libraries and header files are anyhow obtained from a global folder.
- All these reasons will as well be saved in the wiki in order to remember them in the future.
- If I forgot something, please let me know!

Besides that we decided, to force fesa class developers (In the GSI-Guideline) to use a branch, called "release" for development. This branch could be automatically tagged during class-delivery.

By default every artifact should use a roling release branch. Development happens on trunk (or a user/feature specific branch). Productrive code is merged into the release branch. The release branch will always compile. Releases will happen from this branch and will be tagged from there.

The idea is that in the fesa-class/deploy-unit deploy-procedure, the release branch automatically will be tagged, the tag will be checked out, compiled and only the result of this compilation will be deployed.