Guidelines for HPC-Forge

1. Use capitalized, camel case, decorations in the names of the projects but not in those of the instances of the services. Projects are only for presentation but instances are mapped to URLs and can generate errors.
2. Provide a description for the projects; it's useful to find them.
3. Control the activities of your users: disk space, spam, unauthorized contents, etc.
4. Grant permissions and anonymous access with caution: pay attention to sensitive data.
5. Report malfunctions and request functionality to the support: it improves the quality of the infra-structure.
6. Ask for support in case of troubles before waste too much time or mess something.

Subversion

1. Create a Subversion repository for each project: normally each project is developed separately while the commits are global.
2. Create the standard layout for each project: trunk, branches and tags folders.
3. Branch the overall project.
4. Set fine-grained permissions only if needed to prevent performance degradation and migration issues.
5. Make frequent update and commit to prevent conflicts.
6. Keep the main trunk fully working to don't put other user in trouble, especially the non developers
7. Agree tags with other developers or outsourcers.
8. Don't use subversion as repository data (it's mainly for sources). Use instead the WebDAV repository
9. Qualify the binary data: auto-checking may fail, for examples for data with textual headers as the PDF documents.
10. Use clients with GUI (RapidSVN + kdiff3, or kdesvn, ecc) for the normal usage and commands if needed.
11. Use references to the Trac tickets in commit messages.
12. Delete branches with caution: you can hide some relevant information to the others.
13. Make the releases on the main trunk and copy them to tags. Some bugs can be temporary fixed on the tag.
14. Use the branches to implement the features that can't be straight integrated.

Trac

1. Link Trac to the corresponding SVN repositories. This provides effective control of the repository through the browser, integration with tickets and wiki.
2. Set fine-grained permissions only if needed to prevent migration issues.
3. Create descriptive pages and cross-references with the corresponding tickets.
4. Create links to data on the WebDAV repositories for attachments too much big.
5. Define the milestones.
6. Choose the right optional plug-ins but only if needed to prevent problems with upgrade or migration.
7. Define the components.
8. Define the types of tickets (defect, enhancement, task, extra task, ….) for some particular purpose.

WebDAV

1. Do not enter reserved data in public and images. These folders provide anonymous access.
2. Use the images directory to keep only pictures so browsing can be optimized.
3. Use a consistent naming for any subdirectories (doc, …): the same meaning for different projects.

Hudson

1. Use a private slave for any job with reserved contents.
2. Run periodically, preferably at night, to save the computing resources.
3. Don't build only but also run tests and generate documentation.
4. Test everything: classes, methods, functions, executables …
5. Make your tests produce a Junit XML output to leverage the reporting capabilities of Hudson.
6. Set fine-grained permissions only if needed to prevent migration issues.