IEPM SVN Repository

Background

Projects within the IEPM group are version controlled using SubVersioN.

A nice online manual of SVN usage and theory is located here.

SVN is a version control system that enables groups of developers to code on the same software project and or files with little to no worries in terms of conflict of code. More importantly, it allows the branching and tagging of projects. Finally it provides a central repository whereby developers can quickly and easily obtain the latest (or not so latest) versions of project code.

SVN Access

In order to access the SVN, you must have access to the SLAC machines (instructions for creating an account here). Users must also be part of the 'iepm' group.

The path for the SVN repository is

```
file:///afs/slac.stanford.edu/g/scs/net/netmon/repo/svn/
```

SVN Command Primer

SVN works similarly to CVS. In as much as many of the commands are the same.

The basic concept is that the SVN Repositry holds all of the versions of the different code that all developers use.

Users/developers have to work on 'checked out' copies of the code from this central repository. Developers should ONLY make changes to this 'checked out' version; when they have tested and are happy with their changes, only then should they 'commit' their changes back into the repositoty for other developers.

Other developers must issue an 'update' command in order to get the changes from other developers (once they have committed them)

SVN Help

Use the svn help command. if you use man help this will point you to the svn help command):

```
$ svn helpusage: svn <subcommand> [options] [args]
Subversion command-line client, version 1.4.3.
Type 'svn help <subcommand>' for help on a specific subcommand.
...
```

SVN Checkout

In order to work on SVN projects, it is necessary to 'checkout' the project files. Given a project, you are expected to work on edits on a local copy from which editing and testing is to occur before the changes are 'checked in' back into the SVN repository.

You can check-out a project via:

```
$ cd /tmp/
$ svn co  file:///afs/slac.stanford.edu/g/scs/net/netmon/repo/svn/<project-name> my_project
```

This will retrieve the latest versions of the files contained within the SVN project <project_name> and place it into the directory `/tmp/my_project`. Note that this location is arbitary.

Checking-In Edits

Once a SVN project has been checked out, files can be edited. Code should be tested to work on the local copy before committing the changes back into the SVN repository. The processes of committing the edits is called 'checking in' your files.

For example, should a file in your project have been edited, you can commit the changes back by:

```
$ cd /tmp/my_project
$ # edit, say, file /tmp/my_project/test
$ svn commmit
```

Unless the options `--message` or `-m} are supplied, the editor specified in your {{EDITOR` environment will open to prompt for some text input. This text will form the description of your edits and will be stored in the SVN log for the file(s).
Please put something meaningful such as 'Fixed bug which caused text to display incorrectly' rather than 'Fixed bug'.

The processing of checking in edits will also scan for all files in the project (and recursively do so).
If you wish to only commit changes to one or a select number of files, use the `--file` or {-F} options.

Adding Files to the Project

In order to add additional files after the initial import into a project, you must manually add files using the command:

```
$ cd /tmp/my_project
$ touch new_file.pl
$ svn add new_file.pl
$ svn commit
```

Once you have run the command, you must `commit` the action in order to place the newly created file into the repository (again it will prompt for some text describing the changes).

Updating files

To simply update your copy of the file to the latest version use the update command:

```
$ cd /tmp/my_project
$ svn update
```

Restoring a particular version of code

If you wish to discard your changes to a file, or you wish to 'roll back' to a version of the file/project then you can issue a `update` with the `-r` option to revert to the version specified.

```
$ svn update -r version_no my_project
 $ svn update -r 300 My.pm
```

Administrative Details

SVN Administrative Details.

IEPM SVN Projects

What follows are the current SVN Projects that have been imported. Please keep this list upto date.

Note that the relative location is that of

`/afs/slac.stanford.edu/g/scs/net/netmon/repo/svn/` + `<Relative Location>`

Project Name	Relative Location	Description
IEPM Base Libraries	iepm	This project contains the base/standard perl libaries that other project may depend upon. This should contain fundamental base classes that may or may not be extended/inherited by project specific modules.
IEPM-BW	iepm-bw	This contains the iepm-bw related modules and scripts such as that to query for the IEPM-BW database, maintaince scripts etc.
Netflow Analysis	netflow	For the Terapaths Project, this package contains the libraries, scripts and installation scripts required for the Netflow analysis backend and frontend

PingER	pinger	PingER related scripts and modules
Topology Analysis	topology	Topology analysis tools such as `traceanal`, graphviz visualisation etc.
Event Diagnosis	event_diagnosis	This project contains scripts related to event diagnosis. Two types of scripts are present in distributions. Those to be deployed on Central Node and those which should be deployed at Monitoring Node.
Pinger-Db	pinger-db	This contains the pinger-db related modules and scripts to fetch data from pinger database located at pinger.slac.stanford.edu