

SUSTAINABILITY DIVISION

Research, Testing and Development

Technical Report

|  |  |  |
| --- | --- | --- |
| Title | : | Git and GitHub |
| Date Compiled | : | 15 December 2016 |
| Report No | : | 1 |
| Revision | : | A |
| Project/Activity/Task No | : | 1 |
| Author | : | Ntuthuko France Sibiya |
| Section | : | Renewables Technologies |
| Functional Area | : | Sustainability |
| Copyright of this report is reserved. No publication or dissemination of its contents is allowed without written permission. | | |
| Keywords: Git and GitHub | | |

|  |  |  |
| --- | --- | --- |
| Submitted by: | Accepted by: | Authorized by: |
| …………………………….. | …………………………….. | …………………………….. |
|  |  |  |
|  |  |  |
| Date: ………………………… | Date: ………………………… | Date: ………………………… |

# **1. Introduction and background**

# **1.1 Version Control System**

A Version Control System allows every user to: revert files back to a previous state, revert the entire project back to a previous state, review changes made over time, see who last modified something that might be causing a problem. Who introduced an issue and when, and more. Using a Version Control System also it means that if you screw things up or lose files, you can generally recovery easily.

**Three functions of a Version Control System are listed below**:

* Allows developers to work simultaneously.
* Does not allow overwriting each other’s changes.
* Maintains a history of every version.

Every Version Control System tool provides a private workplace as a working copy. Developers make changes in their private workplace and after commit, these changes become a part of the repository.

Git takes it one step further by providing them a private copy of the whole repository. Users can perform many operations with this repository such as add file, remove file, rename file, move file, commit changes, and many more.

# **1.2 The difference between Git and GitHub**

# **Git:**

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is mostly based on command line operation.

# **GitHub:**

GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

It provides [access control](https://en.wikipedia.org/wiki/Access_control) and several collaboration features such as [bug tracking](https://en.wikipedia.org/wiki/Bug_tracking_system), [feature requests](https://en.wikipedia.org/wiki/Software_feature), [task management](https://en.wikipedia.org/wiki/Task_management), and [wikis](https://en.wikipedia.org/wiki/Wiki) for every project. GitHub offers both plans for private [repositories](https://en.wikipedia.org/wiki/Repository_(version_control)), and free accountswhich are commonly used to host [open-source](https://en.wikipedia.org/wiki/Open-source) software projects.

# **2. Application for Git and GitHub**

* + It provides a backup of your files.
  + It gives you a visual interface for navigating your repos.
  + It gives other people a way to navigate your repos.
  + It makes repo collaboration easy (e.g., multiple people contributing to the same project).
  + It provides a lightweight issue tracking system

# **3 Important functionality when using the Command line**

**3.1. Blobs**

Blob stands for Binary Large Object. Each version of a file is represented by blob. A blob holds the file data but doesn’t contain any metadata about the file. It is a binary file and in Git database, it is named as SHA1 hash of that file. In Git, files are not addressed by names. Everything is content-addressed.

**3.2. Trees**

Tree is an object, which represents a directory. It holds blobs as well as other sub-directories. A tree is a binary file that stores references to blobs and trees which are also named as SHA1 hash of the tree object.

**3.3. Commits**

Commit holds the current state of the repository. A commit is also named by SHA1 hash. You can consider a commit object as a node of the linked list. Every commit object has a pointer to the parent commit object. From a given commit, you can traverse back by looking at the parent pointer to view the history of the commit. If a commit has multiple parent commits, then that particular commit has been created by merging two branches.

**3.4. Branches**

Branches are used to create another line of development. By default, Git has a master branch, which is same as trunk in Subversion. Usually, a branch is created to work on a new feature. Once the feature is completed, it is merged back with the master branch and we delete the branch. Every branch is GIT 5 referenced by HEAD, which points to the latest commit in the branch. Whenever you make a commit, HEAD is updated with the latest commit.

**3.5. Tags**

Tag assigns a meaningful name with a specific version in the repository. Tags are very similar to branches, but the difference is that tags are immutable. It means, tag is a branch, which nobody intends to modify. Once a tag is created for a particular commit, even if you create a new commit, it will not be updated. Usually, developers create tags for product releases.

**3.6. Clone**

Clone operation creates the instance of the repository. Clone operation not only checks out the working copy, but it also mirrors the complete repository. Users can perform many operations with his local repository. The only time networking gets involved is when the repository instances are being synchronized.

**3.7. Pull**

Pull operation copies the changes from a remote repository instance to a local one. The pull operation is used for synchronization between two repository instances. This is same as the update operation in Subversion.

**3.8. Push**

Push operation copies changes from a local repository instance to a remote one.

This is used to store the changes permanently into the Git repository. This is same as the commit operation in Subversion.