**Synopsis**

This report covers the research done on version control system. The research done have been documented in this report. The realisation is that version control is very important to successful and proficient project documentation management and development. Even if there has been slight improvement in this part for engineers in general, the software advance community has developed several techniques that greatly improve the process of administrating the evolving versions of documents right through their lifecycle. The most common version control systems that I have researched on are Git, and its online management site, GitHub. The VCS knowledge will have benefits in my entire career as an engineering technician.

**Git**

Git is a [version control system](https://en.wikipedia.org/wiki/Version_control_system) (VCS) for tracking changes in [computer files](https://en.wikipedia.org/wiki/Computer_file) and coordinating work on those files among multiple people. The development of software is being done using Git, but it can be used to keep follow up of changes in any files. Git is classified as a [distributed revision control](https://en.wikipedia.org/wiki/Distributed_revision_control) system which allows many software developers to work on a given project without requiring them to share a common network

**GitHub**

GitHub is a web-based [Git](https://en.wikipedia.org/wiki/Git) repository hosting service. A repositoryis an on-disk data structure which is a way of organising data in your computer so that it can be used efficiently to store metadata for a set of files and/or directory structure. Depending on whether the version control system in use is distributed (for instance, Git) or centralized, the whole set of information in the repository may be duplicated on every user's system or may be maintained on a single server. Some of the metadata that a repository contains includes, among other things:

* A historical record of changes in the repository.
* A set of commit objects.
* A set of references to commit objects, called heads.

It offers all of the distributed version control of [Git](https://en.wikipedia.org/wiki/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 which is a software application that keeps track of reported software bugs in software development projects, [feature requests](https://en.wikipedia.org/wiki/Software_feature) and [task management](https://en.wikipedia.org/wiki/Task_management).

**The difference between Git and GitHub**

The difference between Git and GitHub is that Git is the distributed version control system. Git is responsible for keeping track of changes to content (usually source code files), and it provides mechanisms for sharing that content with others.GitHub is a company that provides Git repository hosting. If your team has a shared repository on GitHub, you could conceivably use GitHub without ever looking at its website. But, the website provides a lot of value on top of the core Git repository.

**Version control systems**

Version control has been closely studied and understood in the software engineering community for a long time. The solutions are stable, robust and well-supported. There are various systems suitable for small local teams and for large distributed teams, making them ideal for coordinating software development, and for mitigating differences in culture and time zone.

Version control is provided at sites such as Github, Source Forge and Google Code. These sites typically build a suite of services around version control: archiving, release downloads, mailing lists, bug trackers, web hosting and build farms. This range of functionality makes them particularly attractive for those projects that do not have the resources to maintain their own server for version control.

CVS used to be the most widely used open source version control system but these days Subversion and Git have overtaken it are commonly used in open source projects. The basic capabilities of these systems are very similar, but they offer different security, networking and abstraction functionality, and different licences. There are also many proprietary solutions available from a range of suppliers.

Many activities in business are accompanied by a responsibility to perform ‘due diligence’ checks. Precisely what these checks entail will depend on the business activity in question, but with regard to intellectual property one important ‘due diligence’ activity is the tracking of the ownership of its constituent parts. So for example, if someone creates a piece of software and wishes her organisation to release it, her organisation will almost certainly want to check the provenance of all the code within the software. This process is facilitated by the ability to track who made which changes to the code, and when they were made. A version control system enables a list of contributors to be compiled and the dates of their contributions to be ascertained. Such a list can be easily cross-checked with a list of IP contracts.

Open development involves contributors making small regular changes to resources. A version control system provides a means for monitoring those changes as they occur. Automated systems will notify those responsible for managing the IP in project outputs. These notifications, coupled with the logs provided for each individual modification, allow project managers to monitor and trace all contributions.

Open development demands care concerning the provenance of the contributions. Open development projects need to follow best practice in this area. If an IP infringement is found to have occurred, the version control system can be used to determine the extent of the contamination (which files were affected by the problematic change), who performed the change and when they performed it. A version control system can even be used to recover the last uncontaminated version of the software.

Version control systems can also be used to establish precedence, when there is a dispute regarding the ownership of code or ideas.