Byron Laferriere

Professor DiMarzio

Milestone One

**Version Control Summary**

**Benefits**

Version control in software development has been a major benefit to companies as the world transitions through different phases of remote work, as well as project management. Without these teams possessing version control, they would face the issues of duplicity in code written and constantly stepping on each other’s toes. Version control also offers the ability to merge code when another employee has altered the same code you have. The system alerts the users who submitted the change, to the other changes being submitted.

Distributed Version Control Systems allow for the tracking of application builds by tracking versions in the SDLC. It also allows users to identify differences, compare files, and merge chosen changes. Because of these abilities, it allows new users to easily download a selected version and catch up to speed quickly. There is also individual repository options that allow users keep personal development efforts separate, if desired. Finally, one of the finer benefits that version control systems offer a company revolve around its troubleshooting abilities. New releases can produce errors that can be quickly isolated by the system by comparison to previous releases. This is extremely beneficial when debugging large projects and can save a lot of time, which can be invaluable.

**Management Assistance**

Version control helped me work through the calculator application by allowing me to see what was committed separately when needed. Creating the separate branches for each of the new features being added to the user interface, allowed me to isolate the error I had made when following through the tutorials. It also helped me save a project that I accidentally moved the .project file out of. I was trying to clean up my workspace and saw that there was one file in the unstaged changes section of eGit, so I pushed it without investigating and nearly lost the file. This is exactly where version control exemplifies its value to a corporation.

If I had not been using a version control system, both of these mistakes would have resulted in a project restart. These systems help reduce and eliminate the human error that is always bound to happen on projects. Before versioning had been around, I would not have been able to return to a specific point in the project, where the code worked fine, and try to rework it before the error had previously occurred. It is not a perfect system by any means yet, but it is extremely helpful in the management of source code during development cycles.

**II. Development**

**A.**

The functionality provided by a local repository, is limited by its disconnection from a shared repository. Remote repositories provide this functionality and allow for programmers to share code they have worked on and perform many functions when doing so. Examples of expanded functionality that remote repositories provide to a local repo on a user workstation are things like; fetching latest updates on code, viewing updates by you and other users, and then updating the public code in the remote repo.

Adding in configuration and cloning to this remote repository expanded functionality can help by configuring the repository to syncing up automatically, and cloning allows for the local repository user to quickly access these files and create their own working repository they can add. This local repo that they clone to their computer can be individually modified, if chosen, until ready to share. Or it can be shared the whole time, even though there is still a working local directory attached.

**B.**

Using the Branch feature in Git repositories allows for the management of the addition of new features to be developed in an independent manner. In this project, it was used to add both major and minor features, as well as bug fixes to implement missing features. It is useful to do this to create feature isolation from the master branch, to ensure that the working code is not affected by the new code being added. Merging is a helpful feature of Git, especially for teams working remotely, that allows for the changes to be taken and integrated with the target branch. Changes by yourself and others on the team, can be viewed and compared before being committed to the repository. In my project experience, I was able to use compare mode to see the differences in the commits and choose which commit to go with. Conflicts did arise in the project, which only needed to be resolved by editing or reviewing the requested commits. Blaming was a useful feature to enable on the project, as it allowed me to see exactly who made changes to the code and when they did so. Stashing allowed me to save changes that I had made to the project, without committing them to the repository. I was able to review my stashes, and then apply the changes in the stash at a later date. This feature can be very useful when working long hours on a project and wanting to review your work before committing. Rebasing is a very similar feature to Merge, however, rebase helps keep the history log cleaner by creating new commits instead of adding to previous versions.

**C.**

States of a source file were useful to view in the project to help identify the current status of files within the repository. Understanding what these different types of file states meant, allow for the developer to quickly browse and find issues that may arise. Untracked files mean that they have not yet been added to the Git repository, which blocks these files from being tracked. Staged files have been added to the Git, so they can be tracked as updates are added. Staged files have not yet been committed, however. Committed files are staged files that have been fully committed to the repository. When this is true, Git shows that the working directory is clean and there is nothing to commit. Modified files are files that have been previously committed to the repository, that now contain new updates that have not yet been sent. Each of these source file state types have different symbols that allow for them to be identified from the Git Repository window in Eclipse.