**Week 4 Discussion 2: GitHub**

Shaun Hoadley

CST499: Capstone for Computer Software Technology

Professor Amr Elchouemi

February 3, 2022

**GitHub**

**Search GitHub for any open project that piques your interest.**

A project that I found interesting on GitHub is 3D City by [lo-th (https://github.com/lo-th/3d.city)](https://github.com/lo-th/3d.city), a freelance WebGL and 3d modeler. I am a fan of city builder games and would like to create one sometime in the future.

* Start date – June 6, 2014
* Number of active members – 79 watchers and 227 forks
* Date of latest update – January 20, 2022
* Key description of the project – 3D City builder game written in Javascript
* Activity level – Moderate to High (recently)
* Language used – Javascript 99.8% and HTML 0.2%
* Number of contributions that can be made and accepted – As far as I can tell, contributions are not accepted on this project.
* Additional information that you deem is valuable

**Compare the different version control systems. Include technologies like Git, SVN, and CVS.**

There are three types of version control systems or VCS. The first type of VCS is a local version control system. Local version control systems are databases stored on the local machine in which each file changed is stored as patches. Only the changes made between versions are stored in these patches, and to see what files look like at specific times requires adding up all of the relevant patches prior to and including the latest to the time being examined. The most significant drawback to local version control systems is that they are stored locally, so all patches would be lost if anything happens to the database.

The second type of VCS is a centralized version control system. A centralized version control system stores all of the file versions in one place, usually a server, so that multiple users may access it simultaneously. Because it is stored in one place, it suffers from the same drawback as a local VCS, where if something happens to the database, you could lose everything. Two examples of centralized version control systems are CVS, which is open source under the GNU general public license (GPL) version 1.0, and SVN (Subversion), an open-source solution under the Apache license.

Lastly, there is a distributed version control system. Distributed version control systems mirror the entire repository (including its history) from the host server(s) to all clients who access the project. What this means is that every client has a complete copy, so if any single copy is corrupted or lost, it can be restored by any of the other mirrors. The most widely known and used distributed version control system is Git. Git is free and open-source under the GNU GPL version 2.0.

**Discuss the key features provided via any version control system and how the different options you picked will support these features.** **Develop an objective way to select the option you feel is the best.**

A local VCS is going to be the most secure since only those with access to the local machine can access the project, but there are ways to add security to centralized and distributed version control systems. Distributed version control systems offer the greatest data integrity because it is stored in multiple places. The version control system that I feel is the best is Git for the following reasons (Git, n.d.):

* Git is easy to learn.
* Git is free and open-source under the GNU GPL version 2 license.
* Git is distributed, providing multiple backups and supporting a number of workflows.
* Git provides cryptographic integrity adding checksums to every file and commit submitted.
* Git is small and fast.
* Git allows and encourages the use of multiple branches that operate independently from each other to be worked on and can then be merged back into the “working” branch. This allows one to test alternative methods of implementation before incorporating them.
* Git has an intermediate staging area for formatting and reviewing commits before submitting them.

**References**

Git (n.d.). <https://git-scm.com/about>. Retrieved February 3, 2022

Tsui, F., Karam, O., & Bernal, B. (2018). Essentials of software engineering (4th ed.). Jones & Bartlett Learning.