Source Control Tools

**GIT**

* Distributed System: Git is a distributed version control system, which means that every user has a complete copy of the entire repository (it can also be used offline to make changes locally)
* Repositories: Git uses repositories to store and manage versions of a project
* Commits: a commit is a specific version of the project
* Branches: Git uses branches to enable parallel development
* Merging and Conflict Resolution: Git provides powerful merging capabilities to combine changes from different branches or repositories

Git has become the standard version control system for many software projects because of its flexibility, speed, and robustness.

**Subversion**

* Centralized System: Subversion follows a centralized model.
* Repository Structure: Subversion organizes data into a hierarchical structure like a filesystem
* Branches and Tags: Subversion supports branching and tagging, allowing you to create independent lines of development or mark specific points in the project's history
* Integration: Subversion integrates well with many development tools and environments

Subversion was popular in the software development community, especially in the early 2000s when control systems like Git weren’t as well known.

**Mercurial**

* Distributed System: Mercurial follows a distributed model, where each user has a complete copy of the repository, including the entire history
* Commits and Changesets: In Mercurial, a commit is known as a "changeset."
* Lightweight and Fast: Mercurial is designed to be lightweight and fast. It performs most operations locally
* Extensions: Mercurial offers an extension system that allows users to customize and extend its functionality

Mercurial has gained a decent following because it is simple, easy to use, and has powerful branching capabilities.

**Perforce**

* Centralized System: Perforce follows a centralized model, where a central server stores the entire repository and manages version control operations
* Workspaces: To work with Perforce, users create local workspaces on their machines. A workspace is a local copy of files from the central depot
* Changelists: In Perforce, a changelist represents a set of changes made to files within a workspace
* Scalability and Performance: Perforce is known for its ability to handle large codebases and large numbers of users efficiently

Perforce has gained popularity in industries that require strict version control and compliance, such as game development, automotive, aerospace, and electronic design.

**Team Foundation Server (TFS)**

* Work Tracking: TFS provides work tracking features to manage and track work items, such as user stories, tasks, bugs, and more
* Build Automation: TFS includes build automation capabilities to define and execute build processes
* Release Management: TFS offers release management features to manage the deployment and release of software applications Reporting and Analytics: TFS includes reporting and analytics capabilities to track project metrics, generate reports, and gain insights into team performance and project health

Team Foundation Server is widely used by organizations, particularly those using Microsoft technologies, for managing software development projects.