GitHub and the Life Cycle of our Compiler

Joseph DeBruycker

Overview

The process of assembling a large programming project such as a compiler with a team of fellow students over the course of a single semester presents several challenges which closely resemble real-world design constraints such as deadlines and conflicting schedules. Working together as a functional team required careful scheduling and effective communication over the lifetime of the project. To facilitate these goals our team made use of several online resources which allowed us to work on the project individually according to our own schedules and still be able to stay abreast of the most current revisions to the source code. One of these resources was the web based Git repository hosting service, GitHub.

GitHub is a free-to-use, web-based software that offers users a GUI based approach towards the version control and source code management abilities of Git [1]. This tool enables users to create and share a repository of source code and allows the users to merge the changes they have made on their local machine with any changes made by other team members with minimal conflict.

The Importance of Source Code Management

In many modern programming projects, especially those related to object oriented design, the source code quickly becomes a large collection of interdependent files. These files will likely be changed multiple times by multiple team members over the course of developing and debugging, and often chasing a single bug or refining an approach involves changing small parts of several related files. This can quickly become a hassle when several people are attempting to work together to assemble a large project, especially if a team member needs the latest revisions to the code in order to continue testing their own functionality. In recent years several methods of combatting this problem have emerged, the most widely used being the distributed revision control software Git developed by Linus Torvalds [2]. Git was initially developed to be a fast and safe way for people working on the Linux kernel to share their changes with the development community. Users working on a Git repository can make changes on their local machines which can then be uploaded to a central repository to create a new branch of the project. All branches created by all users are saved as a project history, enabling merging of functional code into the main project, branching off to change the project, and the ability to revert to a previous working version if for some reason undesired changes have been made by a contributor. These

tools combined let large distributed teams of coders develop new software rapidly and effectively.

Application to our Compiler Project

Our team saw the addition of several members over the life cycle of the project, eventually doubling its original size to six. Although in the grand scheme of things this is not a large team nor was it a particularly large project, the development cycle we followed was greatly facilitated by using GitHub software. The compiler was developed in three main phases which corresponded to class instruction: a scanner, a parser, and a semantic analyzer. Since all team members were expected to work on each phase of the project together in order to emphasize the lessons taught in class, it was necessary to divide up relatively small objectives into even smaller tasks. This can often be even more challenging to programmers than working on one larger task as it requires careful code integration from several members and can be more difficult to grasp the larger picture of what the code does. The scanner and parser designs both included large sections of code outlined in class that were easy to divide amongst the team, but the semantic analyzer as well final debugging sessions were more challenging to functionally separate.

GitHub became an invaluable tool during this time for several reasons. It was easy to share the entire project with new group members who joined us late in the project. It was also easy for one or two people to work together on a section of code and later merge changes to the project with any work the others had done. This was especially useful over Spring Break as several members worked on the code during an extended period when we couldn't meet as a group. It also made the final weeks of polishing and debugging rather painless as each student could work on small bugs in their free time and quickly upload the changes for the group to see.

Summary

GitHub is a user friendly code management system built on the Git system. It is one of the most successful products of its kind, leveraging the power of today's high speed internet to connect developers in ways that allow them to worry less about the task of distributing source code amongst teammates and focus on writing good functional code. This increases productivity and reduces stress, facilitating the advent of a new era of code sharing and teamwork.

References

[1] http://en.wikipedia.org/wiki/GitHub
[2] http://en.wikipedia.org/wiki/Git_%28software%29