

# A SURVEY OF SOURCE CODE MANAGEMENT TOOLS FOR PROGRAMMING COURSES

Delbert Hart  
Computer Science Department  
SUNY Plattsburgh  
Plattsburgh, NY 12901  
1-518-564-2775  
hartdr@plattsburgh.edu

## **TARGET AUDIENCE**

Instructors of courses in which students submit programming assignments, especially those courses with frequent and / or significant amounts of programming. Attendees will gain a greater awareness of the features and limitations of source code management tools. The tutorial will help them identify which tools would be most appropriate for their pedagogical goals. Examples of how the tools can be applied to introductory programming courses as well as software engineering and capstone courses will be discussed. Information presented in the tutorial will be made available for download to conference participants.

## **DESCRIPTION**

In this tutorial I will present an overview of how source code management tools can improve the efficiency, timeliness, and accuracy of providing students feedback on programming assignments. A survey of currently available tools will be presented with a focus on how well the tools support common patterns of interaction between student and instructor, features, ease of use, and ease of deployment. Source code management tools will be compared with other solutions such as courseware software and automatic testing software. The focus will be on software that is widely available and used.

Source code management systems (e.g., subversion and git) are critical tools used in almost all non trivial software projects. These tools support the integration of source code concurrently developed and maintain a history of the changes to the software over time. Source code management in the class used has a number of benefits. The main benefit is that it provides a well understood distribution model for instructors to provide source code to students and for students to submit their work. Detailed information about when submissions were made and what changes were made make it easy for instructors to quickly review resubmissions or corrections. These tools can also provide detailed information regarding individual student contributions to group projects. Because these tools are widely used in industry there is often good support for integration of these tools with other tools such as editors and issue trackers. A byproduct of using these tools is that students gain experience with tools that they are likely to encounter if they work in software development.

## **BIOGRAPHY**

The presenter, Delbert Hart, is an Assistant Professor in the Computer Science Department at SUNY Plattsburgh. He has been teaching software engineering and programming courses for eight years in which he has experimented with the application of a variety of software engineering tools to improving evaluation of student programming artifacts and the ability of students to collaborate in teams. In addition, he manages the department's computer science labs which has provided him with experience in the deployment and managing of these tools.