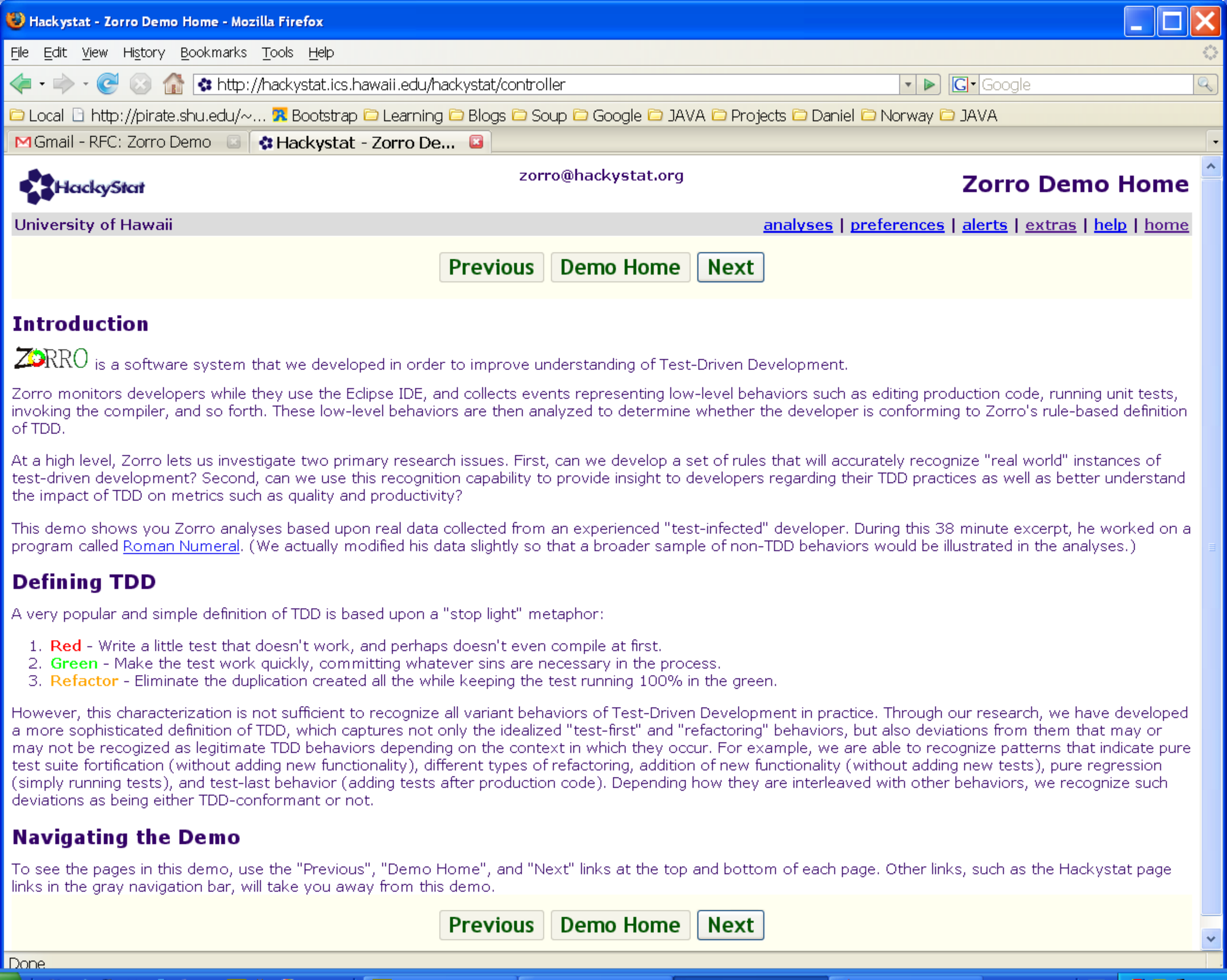
Hackystat - Zorro Demo Home - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://hackystat.ics.hawaii.edu/hackystat/controller
Google
Local http://pirate.shu.edu/~... Bootstrap Learning Blogs Soup Google JAVA Projects Daniel Norway JAVA
Gmail - RFC: Zorro Demo
Hackystat - Zorro De...
HackyStat
Done

zorro@hackystat.org

Zorro Demo Home

University of Hawaii

analyses | preferences | alerts | extras | help | home

Previous Demo Home Next

Introduction

ZORRO is a software system that we developed in order to improve understanding of Test-Driven Development.

Zorro monitors developers while they use the Eclipse IDE, and collects events representing low-level behaviors such as editing production code, running unit tests, invoking the compiler, and so forth. These low-level behaviors are then analyzed to determine whether the developer is conforming to Zorro's rule-based definition of TDD.

At a high level, Zorro lets us investigate two primary research issues. First, can we develop a set of rules that will accurately recognize "real world" instances of test-driven development? Second, can we use this recognition capability to provide insight to developers regarding their TDD practices as well as better understand the impact of TDD on metrics such as quality and productivity?

This demo shows you Zorro analyses based upon real data collected from an experienced "test-infected" developer. During this 38 minute excerpt, he worked on a program called Roman Numeral. (We actually modified his data slightly so that a broader sample of non-TDD behaviors would be illustrated in the analyses.)

Defining TDD

A very popular and simple definition of TDD is based upon a "stop light" metaphor:

1. Red - Write a little test that doesn't work, and perhaps doesn't even compile at first.
2. Green - Make the test work quickly, committing whatever sins are necessary in the process.
3. Refactor - Eliminate the duplication created all the while keeping the test running 100% in the green.

However, this characterization is not sufficient to recognize all variant behaviors of Test-Driven Development in practice. Through our research, we have developed a more sophisticated definition of TDD, which captures not only the idealized "test-first" and "refactoring" behaviors, but also deviations from them that may or may not be recogized as legitimate TDD behaviors depending on the context in which they occur. For example, we are able to recognize patterns that indicate pure test suite fortification (without adding new functionality), different types of refactoring, addition of new functionality (without adding new tests), pure regression (simply running tests), and test-last behavior (adding tests after production code). Depending how they are interleaved with other behaviors, we recognize such deviations as being either TDD-conformant or not.

Navigating the Demo

To see the pages in this demo, use the "Previous", "Demo Home", and "Next" links at the top and bottom of each page. Other links, such as the Hackystat page links in the gray navigation bar, will take you away from this demo.

Previous Demo Home Next