Improving Quality Together

David Jones
Intelliware Development Inc.
1709 Bloor Street West, Suite 200
Toronto, Ontario M6P 4E5
Canada
+1-416-762-0032
david.jones@intelliware.ca

Gordon Cameron
Intelliware Development Inc.
1709 Bloor Street West, Suite 200
Toronto, Ontario M6P 4E5
Canada
+1-416-762-0032
gordon.cameron@intelliware.ca

Abstract
One recent change in software development is developers starting to take responsibility for the quality of their work by writing and executing automated tests. As with any new activity, there is a wide range of ways to perform this task. DevCreek collects, aggregates, and displays testing activity for individuals, teams, and the developer community as a whole. The goal is to provide individuals with global norms with which they can compare their testing. We currently have twenty person-years worth of data, with the goal of collecting thousands more from a variety of languages, tools, and domains.

Categories and Subject Descriptors
D.2.5 [Software Engineering]: Testing and Debugging – Testing; D.2.8 [Software Engineering]: Metrics - Process metrics

General Terms
Management, Measurement, Documentation

Keywords
Automated tests; metrics; testing visualization; process visualization; test-driven development; team awareness; project rhythms; Eclipse

1. Motivation
Until recently, most software development occurred in a black box that was very difficult to open. A requirements specification was delivered to the development team, time and budget passed, and the following results were delivered to the customer.

- Unpredictable cost – 156% of the original budget
- Often late – delivery date was 184% of the target
- Not what you ordered – only 29% are viewed as successful by the people that pay for them [1]

There wasn’t a simple way to:
- track project metrics
- see inside the development process
- foster continuous improvement within a development team
- allow comparisons with good practices or industry norms

“There is no substitute for knowledge” – W.E. Deming

Without knowledge of what was occurring within the development process, the results presented above were inevitable. DevCreek’s aim is lofty: we want to change the inevitable, one project team at a time. We want to Improve Quality Together.

2. Implementation
DevCreek collects, aggregates, and displays testing activity for individuals, teams, and the developer community as a whole.

Our goal is to provide a comprehensive view of the development process. To accomplish this, we will need to collect project related activity from a variety of sources, such as IDEs, command line tools and web forms. The first supported IDE, Eclipse, captures Java and Ruby testing activity. This data is transmitted to the DevCreek server in an open XML format by HTTP. Real-time feedback is provided in a DevCreek view within Eclipse.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright © 2004 ACM 1-59593-XXX-X/0X/000X...$5.00.

Figure 1. DevCreek System Overview.
3. Aggregate Reporting

The DevCreek service has collected twenty person-years of testing activity to date. The stream of test run executions has supported generation of rich reports; trends in broken test duration, delays propagating new tests amongst the team, changing test results over the course of the project.

The collected data can also reveal unexpected patterns of team behaviour, as displayed in the Hour of Day report. All test run executions from a project team are collected by hour of the day and displayed in a radar chart. Figure 3 depicts a trace of the DevCreek team’s activity in green.

The team starts development from 9am onwards, as seen by the increase in test runs. The quantity of testing takes a sharp drop over lunch, before picking up again at 2pm. The afternoon sees the team focusing on coding, quickly getting into a productive flow, thus generating the bulk of the testing activity. Early evening sees the team winding down, with the occasional hour working from home after dinner. A day-in-the-life of the DevCreek team, depicted by the t-shirt shaped trace.

The yellow trace represents activity from another project at the same company. It is quite different, showing a team that operates a couple of hours earlier in the day. Their activity peaks before a short lunch, often eaten in the project room.

4. Real-time Feedback

DevCreek’s real-time data collection and visual display provides feedback to the entire team on both areas of activity and development idiom. This lets developers react when straying from their desired practices, and supports a self improvement feedback loop. The team becomes more aware of how it is functioning as a whole, with the increased transparency allowing other stakeholders to become more involved.

One way of showing recent testing activity is with the test run ticker, as shown in Figure 4. This compactly shows the result of each test run execution, colour coded by result, with timing and magnitude information. This can reveal basic testing rhythms and help with daily retrospectives.

5. Future Direction

DevCreek evolved from a relatively simple concept; use a background process to capture JUnit test data in Eclipse. The data could be collected with no real effort on the part of the developer, and by analyzing it we’d be able to better understand how we were testing.

Collecting automated testing statistics provided the greatest initial impact for our organization. We are continuing development on DevCreek to broaden the range of collection points for project metrics.

6. References