"Good Enough" Reliability for Extreme Programming

Laurie Williams, Lili Wang and Mladen Vouk
Department of Computer Science, North Carolina State University
{lawilli3, lwang6, vouk}@unity.ncsu.edu

Abstract

Extreme Programming (XP), a software development process designed for small to mid-size projects, has strong customer involvement, a simplified requirements gathering and prioritization practice, and an emphasis on testing. These characteristics enable an extension of XP to encompass a measure of reliability. We examine the enhancement of XP practices to include explicit estimation of the probability that the software system performs according to its requirements based on a specified usage profile.

1. XP Practices Related to Reliability

1.1 Customer Involvement

With Extreme Programming (XP) [1], customers incrementally write system requirements via short, natural-language statements called *user stories*. Ongoing customer involvement is necessary for clarification of user stories as development proceeds. Customers also specify black box acceptance test cases. Rather than fulfilling a formal, pre-determined contract, developers show they have implemented a requirement (user story) by demonstrating successful execution of the acceptance test cases related to a particular story.

1.2 Testing

XP has two testing practices: (1) customer-written black-box acceptance test (as described above); and (2) "test-driven development" (TDD) [2]. The customers ultimately run their acceptance test cases via the user interface. However, XP developers often find it beneficial to test the program logic by creating unit test cases that run through the full sequence of program instructions akin to the acceptance test cases (generally not through a GUI). With TDD, software engineers write low-level, automated unit tests every time they create a new class/method, before they write the code. As a result, methods are "testable" (e.g. in the simplest case, at least have return values). These automated unit test cases are often created in test suites using open source tools such as JUnit[1]. Development cannot proceed until all the unit test cases for the new user story pass and all the unit test cases for the entire existing code base pass.

2. Extending XP for Reliability Estimation

We propose the composition of XP acceptance testing and Software Reliability Engineering (SRE) [3] in order to obtain quantifiable measures of reliability.

2.1 Operational Profiles

Operational profiles are at the heart of SRE [3]. Two additional requirements for creating an operational profile are required of the customers. These added steps are necessary for estimation of the reliability range for a system developed using XP:

- The customer must quantify the fraction of usage of each of the "m" user stories, us_i, $0< j \leq m$. It is assumed that the attempted story coverage is 100% from the user perspective.
- For each user story, the customer specifies n_j acceptance test cases that cover this story. Let the fraction of the story that is covered by test-case be at_{ij}, $0 < i \leq n_j$. However, the coverage for all acceptance test cases for a story may not total 100%. Part of it may be due the economics of software testing [5].

2.2 Tool Support

We are developing an open source "Good Enough" Reliability Tool (GERT) to support the composition of XP acceptance testing and SRE. GERT estimates the reliability range for the system. Currently, the tool plugs in with JUnit. An upper bound on reliability is estimated using a Nelson-style model:

$$\hat{R} = \sum_{j=1}^{m} \sum_{i=1}^{n_j} us_j \, at_{ij} \, x_{ij}$$

[1] http://junit.org

where the acceptance test execution score is x_{ij}. If Acceptance Test Case i of User Story j passes, the score is 1, otherwise it is 0.

Proper lower bound is still under investigation. One option is to weight successful test cases proportionally to their number in a particular coverage category, and inversely proportion to the coverage they are intended to offer. We are working on a co-requisite confidence interval model. This model will indicate the upper and lower bounds of our reliability estimate; a prime determinant of the confidence interval model will be the number of test cases written, particularly for critical, high usage user stories. Other issues, such as coverage correlation and sampling issues also need to be taken into account.

2.3 Limitations

There are a number of issues to consider. XP practices assume both mutual independence of user stories and mutual independence of test cases. Also assumed is no overlap in the coverage given by either the user stories or the test cases. Finally, test case selection is not random. XP culture, needing agility, operates under resource constrained conditions [4] using as little as one test case for each space an acceptance scenario covers. That alone creates a reliability over-estimation problem unless it is accounted for. Additionally:

1. The model is highly reliant on good input from a customer regarding the operational profile, and acceptance test cases. Customers must carefully and realistically consider the coverage of their acceptance test cases. The test cases not specified are another problem. A customer should not indicate full coverage by acceptance test cases unless they truly have specified every possible scenario for a specified user story.
2. In XP, the dependencies and overlap between user stories is not identified (e.g. the user stories may not be disjoint). It is possible multiple user stories could have similar functionality, causing a disproportional stressing of some area of the system. Additionally, if the user stories state requirements that are then executed serially, this method would result in an overstatement of reliability.
3. It is very important that customers specify as many acceptance test cases as possible or reliability will be overstated. When validating this model with industrial partners, we will experiment with assessing the completeness of the acceptance test case suites via code coverage analysis.

3. Summary and Future Work

While the initial reliability model has been developed, the research continues to identify and validate this model, its confidence bounds with respect to field operation of XP developed software, the adequacy of the standard XP available production information for realistic estimation of software reliability, and the overhead, if any, SRE imposes on the process. Additionally, we are working on a confidence interval model which will emphasize the importance of writing a thorough set of acceptance test cases.

References

1. Beck, K., *Extreme Programming Explained: Embrace Change*. 2000, Reading, Massachusetts: Addison-Wesley.
2. Beck, K., *Test Driven Development -- by Example*. The Addison Wesley Signature Series. 2003, in press, Boston: Addison Wesley.
3. Musa, J.D., *Software Reliability Engineering*. 1998, New York: McGraw-Hill.
4. Rivers, A.T. and M.A. Vouk. *Resource-Constrained Non-Operational Testing of Software*. in *9th International Symposium on Software Reliability Engineering*. 1998. Paderborn, Germany: IEEE Computer Society Press.
5. Whittaker, J.A., *Certification Practices*, in *Cleanroom Software Engineering Practices*, S.A. Becker and J.A. Whittaker, Editors. 1997, Idea Group Publishing: Harrisburg.