AnnoTestWeb/Run: Annotations based Acceptance Testing

David Connolly[1], Frank Keenan[1], and Fergal Mc Caffery[2]

[1] Software Technology Research Centre,
[2] Regulated Software Research Group,
Dundalk Institute of Technology, Dublin Road, Dundalk, Ireland
{david.connolly,fergal.mccaffery,frank.keenan}@dkit.ie

Abstract. Testing is frequently reported as a crucial stage in the software development process. With traditional approaches acceptance testing is the last stage of the process before release. Acceptance Test Driven Development (ATDD) promotes the role of an expert customer in defining tests and uses tool support to automate and execute these tests. This abstract outlines a tool, AnnoTestWeb/Run aimed at expert customers specifying acceptance tests with reuse of existing documentation.

1 Introduction

A large part of software development expenditure is attributed to *testing*. Traditionally, with plan-driven development, acceptance testing, the process of testing functional requirements with "data supplied by the customer" [1] occurs as the final stage of the development process long after the initial investigation has completed [2]. Many reports, however, highlight that costs can be reduced by detecting errors earlier in development [3]. Also supporting this, in many domains (e.g. medical device industry) software is developed in a regulatory environment with a tendency for extensive documentation. In contrast, agile approaches require constant customer collaboration throughout development, with customer provision of acceptance tests being an important part of this role. Often, it is recommended that tests be identified before implementation commences. In eXtreme Programming (XP) [4], for example, acceptance tests are defined as a part of the User Stories practice and, as such, are written before coding of the story begins. The practice of ATDD "allows software development to be driven by the requirements" [5]. A key advantage of ATDD in its wider context is that it leverages existing agile infrastructure including continuous integration and test-first development.

In many organisations business rules are documented in numerous formats. However, ATDD is currently not well supported with tools that enable reusing such existing documents, without rewrites, to create executable tests. A challenge therefore, is to support a suitably informed expert in performing the agile *customer role*, including easily creating tests from existing material. However, successful identification of accurate acceptance tests in this manner is not necessarily straightforward.

2 AnnoTestWeb/Run Tool

AnnoTestWeb/Run is a browser based tool built using the Google Web Toolkit and CouchDB. Creation of tests involves using annotations that describe different elements of an acceptance test. A metadata system provides extra detail to annotations e.g. label, data types. It features a simplified interface and workflow partially because it is aimed at all members of agile teams, including non-developers.

Reporting of test results generated by program execution is also designed to occur in a simplified fashion through a Javascript Object Notation (JSON) API. This means that tests can execute in a wide variety of programming languages and environments. A library to speed up implementation of tests is also provided for Java and C#.

3 Conclusions and Future Work

The AnnoTestWeb/Run tool has been demonstrated to a domain expert and an agile team. Future work will involve empirical investigation of its use with both undergraduate and post-graduate students. A long running project with an agile team adopting the tool is also planned. These evaluations will examine the use of ATDD in an agile setting. Feedback from this work will also be used to inform changes to the tool and prepare for a future open source release.

3.1 Acknowledgments

This research is partially supported by Institutes of Technology, Technological Sector Research Programme, Strand 1 Fund and Science Foundation Ireland through the Stokes Lectureship Programme, grant number 07/SK/I1299.

References

1. Sommerville, I. Software Engineering, 8th edition, Addison-Wesley, pages 80-81, 2007.
2. Pressman, R. S. Software Engineering: A Practitioners Approach, European Adaption, 5th edition. McGraw-Hill, 2000.
3. Tassey, G. The economic impacts of inadequate infrastructure for software testing National Institute of Standards and Technology (NIST), May 2002.
4. Beck, K. and Andres C. Extreme Programming Explained: Embrace Change, 2nd edition, Addison Wesley, Boston, 2005.
5. Park, S.S. and Maurer, F. The benefits and challenges of executable acceptance testing, in APOS 08: Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral, ACM, New York, NY, USA, pages. 19-22, 2008.