The Truth About Exploratory Testing

by Dion Johnson

Dion Johnson says exploratory testing and scripted testing can coexist.

Exploratory testing is an approach that all true test engineers use, but its exploitation by those who wish to escape accountability and forgo up-front planning is disturbing. If these misguided individuals would more closely study the detailed works of exploratory testing experts (those in the industry who have led the effort to perform and teach exploratory testing as an official approach), they would discover what exploratory testing really is—a way to enhance scripted testing.

Only a surface-level view of the approach is obtained from abbreviated write-ups. In some cases even the experts incorrectly suggest that scripted testing conflicts with meaningful exploratory testing and hurts the quality of applications. This sends the wrong message about scripted testing and about testing processes in general.

The term "scripted testing" often encompasses test case and procedure design, documentation, and execution. Test design is the mental conception or planning of the test. Documentation is formally writing or printing the test design information, and execution is the actual test case implementation. Test scripting creates traceability, provides information about expected application functionality, identifies test scenarios more completely than could often be done during real-time testing, and provides information for properly setting up the environment and establishing test data.

Exploratory testing—simultaneous learning, test design, and test execution—allows the tester to control the design of tests as he performs them and to use information gained while testing to design new, more effective tests. Experts indicate that the term "exploratory testing" refers to a thought process that exists on a continuum between purely scripted and purely exploratory. Purely scripted means the tester only performs tests based on instructions provided by the script. Purely exploratory means none of the tester's activities are pre-specified, and the tester is not required to generate any test documentation beyond bug reports. Using this definition, we can clearly see that it's not really a question of whether you perform exploratory testing or scripted testing, but rather to what degree you perform each. Exploratory testing can be executed during up-front script design and test execution whether or not a script is used. But the ability to effectively implement the analytical behavior associated with exploratory testing separates skilled software test engineers from "test executors," who merely follow steps created by someone else (see the Sticky Notes for a link to my article on this topic).

When experts speak of a conflict between scripted testing and exploratory testing, they're referring to people who follow pre-documented information to avoid having to make real-time decisions for better testing of the application. This is a problem, but not due to the development of scripts or a conflict between scripted and exploratory testing. The problem lies with the test implementation, and this may be better understood by comparing test scripts to the test plan as a whole. The test plan is a guide for managing the entire testing effort, but during implementation there are things in the plan that must change based on what's happening in real time. The fact that you need to make changes, however, doesn't mean that writing and using the test plan is bad. It only means that the test manager must be analytical and flexible enough to make reality-based adjustments to the plan. The plan is still necessary, because without it testing would be terribly inefficient. The plan is not meant to limit you, but rather to aid you, and the same applies to scripts.

Test scripts provide a springboard for executing tests in the application. In some respects, the concept of a continuum of scripted and exploratory testing oversimplifies the relationship, because it suggests that the more scripted the test approach, the less exploratory it is. This is not necessarily true, especially since the term "scripted" also addresses script documentation. You can have a well documented test case and a scarcely documented test case that are equally as exploratory in their implementations.

(Continued on page 42)

(Continued from page 44)

There may be a time when you need to reduce the amount of documentation produced (specifically the procedural portion of the test case) in order to focus on development or execution of more tests. But that's not a conflict between scripted and exploratory testing—simply a need for prioritization and test management.

Suggesting that scripted testing conflicts with exploratory testing opens the floodgates for an assault on any pre-planning and documentation activities. People will reference exploratory testing doctrine in an attempt to justify their resistance to improving poor requirements analysis and documentation practices, or to prove it's unnecessary to allocate adequate time for up-front test planning and documentation. No amount of exploratory testing will serve as a substitute for good processes, and given the opportunity, we must all work to improve our testing processes. Suggesting that scripted testing is somehow the cause of poor quality is terribly irresponsible.

When a detailed study of the body of work on exploratory testing is performed, there's not much confusion about where it fits into our quality practices. But not everyone sees the value of scripted testing. Testers and testing experts must do a better job of qualifying their statements to reveal that scripted testing does not conflict with exploratory testing. Such statements are made to the detriment of software projects, and any perceived conflict is really a problem with prioritization, test management, or the analytical skills of the tester. **{end}**

As a senior test consultant and managing partner for both DiJohn IC, Inc. and Achieve QA, Inc., Dion Johnson provides IT consulting services that focus on the overall system development lifecycle, with particular focus on the quality assurance, quality control, and requirements analysis elements. Dion has presented at numerous SQE conferences and contributed to StickyMinds.com and Better Software *magazine. Email Dion at dionjohnson@dijohn-ic.com or dionjohnson@achieveqa.com.*

STAR EAST SPEAKER

Sticky Notes

For more on the following topic, go to www.StickyMinds.com/bettersoftware

- "Executor or Engineer"

Have the Last Word!

We are looking for insightful, thought-provoking commentary for possible use as a Last Word column.

Please send your submission to editors@bettersoftware.com.

You will be notified if you are being considered for publication.

Index to Advertisers

Akimbi Systems	www.akimbi.com	Opposite 17
Agile	www.agile2006.org	43
AutomatedQA	www.automatedqa.com	Inside Back Cover
Better Software Conference & EXPO	www.sqe.com/bettersoftwareconf	13
Empirix	www.empirix.com	31
IBM	ibm.com/takebackcontrol/flexible	6
iTKO	www.iTKO.com	11
LogiGear	www.logigear.com	14
Parasoft	www.parasoft.com/insure	5
PNSQC	www.pnsqc.org	40
Rally	www.rallydev.com/bsm	19
Seapine Software	www.seapine.com	2
Segue	www.segue.com	Back Cover
Software Quality Engineering Training	www.sqe.com	39
Software Quality Solutions (SQS)	www.sqs.com	Inside Front Cover
SPI Dynamics	www.spidynamics.com	9
StickyMinds.com	www.StickyMinds.com	41
STAR*WEST* 2006	www.sqe.com/starwest	23
TechExcel, Inc.	www.techexcel.com	15
Worksoft	www.worksoft.com	1

Display Advertising

Shae Young syoung@sqe.com

All Other Inquiries

info@bettersoftware.com

Better Software (USPS: 019-578, ISSN: 1532-3579) is published eleven times per year. Subscription rate is US $75 per year. A US $35 shipping charge is incurred for all non-US addresses. Payments to Software Quality Engineering must be made in US funds drawn from a US bank. For more information, contact info@bettersoftware.com or call (800) 450-7854. Back issues may be purchased for $15 per issue (plus shipping). Volume discounts available.

Entire contents © 2006 by Software Quality Engineering (330 Corporate Way, Suite 300, Orange Park, FL 32073), unless otherwise noted on specific articles. The opinions expressed within the articles and contents herein do not necessarily express those of the publisher (Software Quality Engineering). All rights reserved. No material in this publication may be reproduced in any form without permission. Reprints of individual articles available. Call for details.

Periodicals Postage paid in Orange Park, FL, and other mailing offices. POSTMASTER: Send address changes to *Better Software*, 330 Corporate Way, Suite 300, Orange Park, FL 32073, info@bettersoftware.com.