How to Manage and Measure Exploratory Testing

Jon Bach, Quardev, Inc.

Let's get right to it...

The last four pages of this document contain a session report produced from 90 minutes of exploratory testing of a product called DecideRight. It is an artifact of a method to manage and measure exploratory testing effort. The method is called Session-Based Test Management.

You might be thinking that exploratory testing finds good bugs and has good return-on-investment, but it's not measurable or manageable.

While exploratory testing does rely on the skill and freedom of a tester to think of meaningful test ideas and execute them, it is not “random” testing or “thoughtless” testing, though it can seem that way. For certain, it is not unmanageable or unmeasurable.

Brainstorming with a leading test expert (my brother James), we came up with a solution to make exploratory testing both manageable and measurable. We asked ourselves: what if we did manual testing in sessions – blocks of time roughly 90 minutes each? And what if, after that 90 minutes, we delivered our stakeholders a report that told the story of our test effort?

We thought about what we’d need as test managers to tell the story of our exploration in a simple report, and tried a session. It wasn’t long before we had the following structure:

The Charter

A charter is a mission statement consisting of two or three sentences to guide your testing for the next 90 minutes. A charter might say "Analyze the X function. Make note of any risks, claims in the spec, or areas of instability. Be on the lookout for latency when all "Submit" buttons are pressed."

Charters come from a lot of places – meetings, emails, managers, and even yourself. They are meant to be general enough to allow the tester freedom to explore, but specific enough to corral them from testing the whole product.

Duration

As testers begin charters, they make note of the time. With Session-Based Test Management, keeping precise time with a stopwatch is not important, but they’ll need a general sense of how long the session is taking. Sometimes lessons only last 30 minutes, sometimes they can take two hours. Longer than this might mean that the charter is too vague. Shorter than 30 minutes may mean the charter is too specific – that is, it may not have fostered much of an exploration.

Metrics

In the report, the tester estimates their time related to three tasks: Test Execution and Design (T), Session Setup (S) and the time it took to investigate and report any bugs (B). These "gut feeling" estimates or "TBS metrics" are a way to give stakeholders an idea of how the test effort is going.

> **Setup (S):** With charter-in-hand, the tester makes note of the time and starts testing. If they need to print out any docs that help them fulfill their charter, they do it. This is just one of many Setup activities they might need to do depending on their testing style and what's helpful to you. Others may include configuring the machine, installing the build, or changing product settings.
>
> **Test Execution and Design (T):** As they test, they think of ideas and questions to ask the software, just like in manual testing, because this *is* manual testing -- just governed by time and a charter.
>
> **Bug Investigation and Reporting (B):** Bugs found during testing need to be logged into the session report *and* the bug database. I recommend writing up the bug right there in the session report and then cutting and pasting it into the DB after the session is over. This allows the details to remain fresh.

In their best estimation, the tester asks themselves how often they stopped to investigate something weird and take the time to write it up. If they took any time at all, this interrupted testing. This isn't a bad thing in and of itself, but it is meaningful to report because it stopped testing coverage for awhile.

The same is true for setup activities. How much time did they spend on setting up and configuring for their session once it started? Was there any time during the session that they stopped testing to set something up or reconfigure? That time interrupted testing, too.

So, in effect, B and S time during a session is an interrupting to the third metric: T. A manager might look at a session report where a tester reported 50% B time and 30% S time, which means they would have spent 20% on T time. That's important to know because high B and S times may provoke them to harass Programming to give them better builds with less bugs or think of resources to give them so that setup doesn't take so much time.

It all comes down to T. Test Design and Execution time is the amount of time a tester spent covering their charter. T is the progress they made. If T time is high, that may mean the thing they were testing wasn't all that buggy or that setup was minimal or non-existent.

So, T, B, and S together is our best idea to represent what testers actually do when they explore.

Protocol

At the beginning of a project that uses exploratory testing which I want to manage and measure through sessions, I take my team (even if it's only me) and do a session with the charter: "Create some important charters to run first".

In a session like this, my notes will reflect my ideas of where we could spend future sessions that will take about 90 minutes (in our best estimation). That first session is spent looking at the whole product, assessing various risks, and looking for capabilities or vulnerabilities. I may even find a bug or two. But at the end of the session, as with all testing sessions, I have to be ready for scrutiny.

PROOF

At the end of a session, the report is debriefed. PROOF is a mnemonic for Past, Results, Obstacles, Outlook, and Feelings – five areas I want to cover as I talk with the tester about their session report.

It's good to debrief sessions the same day they're created, especially on the first few days of a projects using SBTM. In this first week, you'll be learning more about your testers' ability and they'll be learning about what you will expect from them. One-on-one time like this is valuable because these initial sessions set expectations.

After the first week, testers will get better at knowing what you will ask them in the debrief and will become more prepared. And you will know more about what they are likely to produce. After this "break-in" period, you'll find that testers will need less of your time during the debrief, and you will need less of theirs.

The Tool

There is a free tool that “scans” session reports. Written in Perl, it parses the data contained in the reports and makes other useful reports out of them, including:

Number of sessions completed
Number of problems found
Function areas covered
Percentage of session time spent setting up for testing
Percentage of session time spent testing
Percentage of session time spent investigating problems

The tool (and the instructions to use it) can be found at www.quardev.com/tools/sessions.exe or www.satisfice.com/sbtm/sessions.exe

www.Quardev.com — info@Quardev.com — phone: 206.547.7771 — toll free: 866.305.4843

EXPLORATORY TESTING SESSION REPORT

CHARTER

Using the steps outlined in the manual, create a decision table manually noting any significant differences than when using QuickBuild.

#AREAS
OS | Win98
Build | 1.2
DecideRight | Main Table window
Strategy | Complex | Stress Testing
Strategy | Complex | Function & Data Testing

START

4/17/01 5:30pm

TESTER

Jonathan Bach
Tim Parkman

TASK BREAKDOWN

#DURATION
normal

#TEST DESIGN AND EXECUTION
50

#BUG INVESTIGATION AND REPORTING
30

#SESSION SETUP
20

#CHARTER VS. OPPORTUNITY
90/10

DATA FILES

Thursday.drd

TEST NOTES

Stepped through the steps in the manual, starting on page 5-1 to walk through a new decision table -- ended on page 5-6

* Clicked toolbar buttons (see BUG 1)
* added options and criteria w / weighting (see BUG 8)
* added options that were non-alphanumeric characters
* tested Optional Overview field -- 32000 characters entered
* Edit menu: Add Option (see BUG 2) (via menu and clicking. Tested how many options you can list and alphabetizing (see BUG 3 and 7)

??? Manual says that I can return to the table by "clicking any other table element." Not sure what this means.

* Entered a description for an option
* Changed the name of an existing option
* Added a new option (see BUG 4 and 5)
* Verified a option can be deleted with right-click menu or edit menu

??? Should Undo work after deleting an option? It doesn't. (see ISSUE 2)

* Added 63 columns of criteria -- (see BUG 6 & ISSUE 1)

OPPORTUNITY: tested Find/Replace on option description (DCR about no Replace button -- UI shows a confirm instead) -- spent about 10 minutes testing the max length of the description field

After creating a table using the steps in the manual, I didn't see any important differences from using QuickBuild.

BUGS

#BUG 1
UI: paper clip and pushpin buttons are not disabled, even though they do nothing

Repro:
1 -- create a new decision manually
2 -- click either paper clip or pushpin button (fly out text says "view/edit documents that explain a decision element"

Result: No response.

Expected: Should be grayed out, else to perform the function that the flyout menu claims should be performed.

www.Quardev.com — info@Quardev.com — phone: 206.547.7771 — toll free: 866.305.4843

#BUG 2
No accelerator keys for some menu items

Repro:
The following menu items do not have underbars:

- File | Preferences
- Edit | Add Option
- Edit | Add Criterion
- Edit | Delete Option
- Edit | Delete Criterion
- Edit | Find/Replace
- Edit | Numeric Values
- Edit | Optional Epxlanation
- Edit | Documents
- View | Ratings Graph
- View | Baseline Comparison Graph
- View | Previous Criterion
- View | Next Criterion
- View | Previous Option
- View | Next Option
- Format | Recalc Disable (Minimize Table)

#BUG 3
Decision table lets you have options and criteria that are identically named

Repro:
1 -- Make a decision table manually
2 -- add two criteria and options with the same name

Result: no warning that there is a duplicate
Expected: No duplicates to be allowed, because of potential confusion to user

#BUG 4
When focus is on Option list in table, there is no arrow key support to scroll through list

Repro:
1 -- make a new decision table
2 -- highlight an option
3 -- press the up or down arrows

Result: No response.
Expected: Arrow keys should be active to scroll through the list of options.

#BUG 5
Data can't be entered in entry box for new option when focus is on table

Repro:
1 -- In the option view, click on the table
2 -- click on the insertion point
3 -- type something

Result: There is an insertion-point cursor, but keyboard is unresponsive.

#BUG 6
Crash -- GPF in DECIDER.EXE (crash in GDI.EXE in module 00016:000007f1 when entering over 60 columns of criteria

Repro:
1 -- create a new decision table (manually)
2 -- add criteria by typing in a name and hitting ENTER
3 -- repeat approx. 60 columns

Result: GPF in GDI.EXE. When you try to launch DecideRight again, dialog pops up -- "not enough free memory" even though there is no other app open.

#BUG 7
New option added to existing options does not get sorted alphabetically

Repro:
1 -- create a new decision table
2 -- add 5 or 6 options
3 -- add some criteria
4 -- go back and add another option

Result: The new additional option does not get alphabetically sorted like the others until the table is closed and re-opened.

#BUG 8
Identical options can be entered

ISSUES

#ISSUE 1
Is there a recommended maximum number of criteria? We were getting GPFs with about 60 columns.

#ISSUE 2
Should Undo work after deleting an option? It doesn't.

www.Quardev.com — info@Quardev.com — phone: 206.547.7771 — toll free: 866.305.4843