Open-Book Testing: a Method to Teach, Guide, and Evaluate Testers

Jon Bach

OVERVIEW

One technique for teaching and evaluating students in an academic setting is to give them an exam. One method is a "closed-book" exam, where students are tested on what they know and are not allowed to consult resources. The instructor lists questions they want the student to answer, and either the student answers them correctly or not. If a student brings a resource and consults it during the exam, it is often grounds for dismissal and expulsion.

In an "open-book" exam, however, students can bring and consult references. Unlike closed-book exams, open-book exams do not just test knowledge, but also the student's ability to acquire the knowledge on demand and apply it to the question.

As curriculum content was reviewed at Singapore's Nanyang Technological University in 1998, a survey of students showed that in order to develop creative and independent thinkers, "more open-ended tasks that reflect real-life situations – questions involving problem-solving should be incorporated in examinations, especially for higher level education." (Han C, 1998).

Another study said that open-book "removes much of the fear and emotional block encountered by students during examination, while, at the same time, it emphasizes practical problems and reasoning rather than recall of facts." (Theophilides & Dionysiou - 1996)

In my ten years as a tester and test manager, I can attest to the same experience as I experiment with open-book techniques in my role as Managing Test Lead for a software testing lab specializing in rapid and heuristic exploratory test techniques

This paper is about how instructors and test managers can use an Open-Book Testing paradigm on software projects to:

1) Teach: Acquaint testers with the main activity of testing – asking questions of the software

2) Guide: Give testers a context or a mission to frame their questions and get them familiar with the product quickly

3) Evaluate: Assess testers' questioning and critical thinking skills

ORIGIN

Open-Book Testing (OBT) as a paradigm for teaching software testers started in 2002 when I was a tester on Microsoft Flight Simulator 2004.

There were many pilots on that team, and one of the resources offered to the testing staff was a series of Ground School classes taught by one of the test managers who was also a certified flight instructor. Ground School is training that all student pilots in the United States must take in pursuit of a pilot's license granted by the Federal Aviation Administration (FAA). Topics include aerodynamics, weather, navigation, and FAA regulations.

It was not required that Microsoft Flight Simulator testers be actual pilots-in-training, but they were encouraged to attend Ground School because of past testimony from testers who said it enhanced their knowledge of Flight Simulator.

When the 8-week series of weekly classes ended, there was an open-book final exam which allowed many resources to be used to answer the questions, the most valuable being Microsoft Flight Simulator 2004 software itself, which features accurate flight model characteristics of several popular general aviation aircraft, as well as accurate weather depictions, terrain maps, navigational charts, and flight tutorials.

This is an example of one of the open-book exam questions:

A pilot is planning a trip from Singapore to Los Angeles (LAX) and has Samoa as one of the waypoints programmed into the GPS. Just in case he encounters bad weather approaching Samoa, he wants to plan a landing at a nearby airport. What airports will the GPS list as options for emergency airports and what is his best option if the storm is right in his path? (Aircraft is a DC-3 at 20,000 feet with 3 hours worth of fuel.)

In researching the answer, I was exploring features that weren't in my assigned feature area to test. But outside of the exam, I could see how the features I was responsible for could interact with others I had previously not thought about. I found myself more engaged, alert, and remembering more about what I saw when I explored. I was immersed in the product and was learning more effectively in the mission of answering the question than I had been while listening to each lecture or exploring the software on my own. I attribute this to the fact that I had defined frameworks, or missions, to answer all kinds of questions; that is to say, I had new contexts to run all kinds of *tests*.

It left me with a better model of the product, but more importantly, it created memories of my travels, leaving me with more ideas of how the software might be used. It also made me more conscious of opportunities to find failures that a wider range of customers might find because of the varied scenarios and contexts that arose in my exploration in pursuit of the answer.

PROCESS

Struck by my experience with the exam, I set out to define a procedure where testers and test managers could use an open-book framework to guide their testing.

If testing is *questioning*, then Open-Book Testing is a test-idea factory, creating and driving all kinds of question-driven contexts to reveal problems in the software.

Here is a suggested process:

1) Interrogate: The test manager or lead develops a list of questions for the testers to answer.

2) Manipulate: The testers execute actions to answer the question.

3) Observe: Testers take notes on what they find.

4) Plan: Testers determine any follow-up questions (tests) that occur to them, in preparation to debrief their results.

5) Evaluate: Testers and test manager meet to compare answers (test results).

6) Negotiate: After the debrief, testers and test managers talk about the appropriate next steps in mission or coverage

APPLICATIONS TO SOFTWARE PROJECTS

I. Teaching Testers

If an entry-level tester (or an employee transferred into the QA department) is given the question "What are the ways you can launch a setup executable in Windows XP", some of their answers may include:

- Double-click the exe
- Double-click a shortcut to the exe
- Press the Enter key on setup.exe when it has focus
- Start menu / Run / type the path to the exe
- Open a command shell and navigate to the exe

But there's also some slightly less common ways:

- Open Task Manager / New Task / type the command
- Write a batch file that calls the exe
- Control Panel / Add Remove Programs

Since the question is open-ended, it can show how the tester attains and reports new knowledge when the tester has many avenues in which to find the answer, such as:

- Intuition
- Experience
- Expertise
- Online help
- Collaboration
- Written resources outside the software

It is an effective teaching technique because it creates context-driven goals that lead to memories. It also creates opportunity – even room for misunderstanding and misinterpretation of the question could find bugs.

Paired testing is encouraged here as well because "it can result in high productivity and high creativity, serving as an effective training technique" (Kaner & Bach, 2001). Two testers working on the same list of questions can combine the avenues above, collaborating to share expertise.

Confidence can easily be built by choosing open-ended questions that require less exploration, or questions with one correct answer but that have more than one way of getting to that answer.

Here's another example in more detail. It describes the thinking behind an Open-Book question:

> *Using Flight Simulator's GPS feature, what are the 10 airports nearest to Samoa?*

What this scenario is meant to uncover:

1) Is the algorithm for the Nearest (airport) function working?

2) Did the tester navigate to Samoa using:
    - Slew
    - Initialize position
    - Flight Planner – Samoa as init point
    - Direct to (Samoa as a destination)
    - Samoa as a waypoint
    - Airports – typing it in flight sim
    - Typing it into GPS

3) Did they use the "Nearest NDB" feature?

4) Did they use the "Nearest VOR" feature?

5) Did they zoom in or out using the GPS Map?

6) Did they "cheat" and not use GPS, but Map View?

7) Did they use the "Choose country" feature and choose "Samoa" as the end point?

II. Guiding Testers

Whether a new tester is brought into an existing project or a seasoned tester is starting a new project, each has to develop a model of the software. OBT provides a mechanism to get them immersed quickly. It gives them a mission, a goal, a charter and a context for exploration.

When a test manager is handed a group of experienced testers, the manager can use OBT to task them, solving the problem that new teams often have – no destination or course to guide them on the first day.

OBT is guided exploration similar to the concept of charters in Session-Based exploratory testing (Bach & Bach, 2000). In SBTM, charters are mission statements to host time-boxed exploratory "sessions". The goal in SBTM is for the test leads and managers to collaborate with testers to write charters that are not so specific that it only takes a few minutes to accomplish the mission.

With OBT, the goal is to get more specific. Lists of questions are designed so that they can be answered in the same time as one session charter (anywhere from 1 – 3 hours). In fact, the questions can almost read like a closed book exam, with one right answer in mind, except for the fact that testers can consult resources.

Here are a few examples of Open-Book questions for exploring Microsoft Flight Simulator:

1) What are the ways to shut off the engine on the DC3?
2) List all the objects that can result in collision crashes.
3) How many approach types are there for Los Angeles International (LAX)?
4) Can the user specify snowstorm weather for Miami International airport?
5) Using the GPS, how many nautical miles is it on a direct route from KSAN to KSFO?

The test manager has to design enough questions and use their judgment in assigning how long it may take for testers to find the answers. The same questions can be given to multiple testers, or different questions, but the aim is to get them tasked and building a model of the product.

III. Evaluating Testers

Just as there is a student/instructor relationship, there is a manager/tester relationship where testers are evaluated on their agility, resourcefulness, and time management.

As the answers and notes are debriefed after a series of Open-Book questions are answered, there is a coaching and evaluation opportunity for the test lead. It's not just a way to see how much of the product has been explored and how, but a way to engage the tester and evaluate their skill, energy, and technique. It also gives a student a chance to assess themselves.

Some criteria to consider:

- Did they get the answer that was expected?
- Were they stuck?
- Did their confusion stop them or did they consult a resource to work around it?
- Did their confusion lead to an interesting misunderstanding of the question, and did that misunderstanding lead to an interesting new test idea?
- Did the tester answer the question in a way that leads to good coverage and effective test design?
- Did it lead to them being able to demonstrate a better understanding of the product?
- Did they have fun?
- Did they remember what they did?
- Did the questions inspire them to come up with their own open-book questions?
- Did Open-Book team debriefings of the answers seem to lead to higher morale?
- Did it promote speed and agility?
- Did they pushback with questions when they are confused, or did they need more context?
- Did they consult a variety of resources?

APPLICATIONS TO ACADEMIC SETTINGS

I don’t work in a university setting, but part of my responsibility as Managing Test Lead for Quardev Laboratories is to teach testing classes. As I prepare to present OBT at the 4th Annual Workshop on the Teaching of Software Testing, I know that the audience is comprised of university-level instructors, and I have thought of some practical applications with OBT that may work for them.

These will be discussed more in my presentation, so I will summarize my thoughts here as bullet points in the three categories I have targeted for this paper as useful applications of OBT.

I. Teaching Testers

- Where do questions come from?
- How to pay attention to the questions you have, second by second
- Questioning as a way to learn product modeling
- Paired testing exercises: teach collaboration and test technique
- Class-wide debriefing: teaches testers what test managers expect
- How to “charter” your own Open-Book testing
- Students writing open-book exams for each other to take

II. Guiding Testers

- Types of Open-Book tests – using several types of questions to demonstrate different paths or contexts through a product
- OBT as an exercise in critical thinking when a question is vague or has several answers
- Using personas as a frame for OBT
- Acquainting students with both an intellectual “workspace” where certain answers are expected, and a “playspace” where initiative, creativity, and exploration is encouraged
- OBT as a way to orient students with a piece of software used in class

III. Evaluating Testers

- How do they approach the questions?
- How detailed are their answers?
- What initiative have they taken?
- What energy do they bring to the tasks?
- What kinds of abilities are emerging for them?
- What resourcefulness is demonstrated (i.e. what kinds of literal resources are they consulting)?
- Are their notes and narratives sufficient to convey the answer(s)?
- What kinds of Open-Book tests are they writing for others to answer?

LAB RESEARCH

Over the past 10 years, I have led many testing projects with new testers and veteran testers. But in experimenting with OBT recently, I have noticed some interesting and profound differences in team dynamics than when not using OBT.

In early December 2004, I decided to a 2-day experiment with OBT at Quardev Laboratories in Seattle where I work as a test manager.

On the first day, I assembled 5 testers with various skill sets and experience and told them to explore the Microsoft MSDN and TechNet websites. I gave them little other instructions than that. Their only mission was that in preparing to test the site's function and content, they were to "become familiar" with it in any way they thought useful.

I checked in on each of them at various times of the day, but only very informally to see if they had any questions. There were a few minor clarifying questions as to the scope of the exploration, but on the whole, everyone seemed quiet and did their best to do what they were told.

A one-hour debrief at the end of the day told a different story. I learned that some testers had been bored, others uninspired, and two even overwhelmed with the site's rich content. None of them had any context of what a user was except to see themselves as users, and the different skill sets of the testers yielded different results – some finding minor content and function bugs, others finding no problems and saying everything was "fine" and "straightforward".

The next day, I hosted two Open-Book Testing exercises. In the first hour of the four-hour experiment, they were given a set of 9 questions to answer, and in the second hour, we debriefed them.

The debrief was held in our conference room with a projector and a laptop as we each debriefed our answers to the first set of questions. Right away I saw that they were more engaged then the day before. They seemed more energetic and enthusiastic. The mission, they said, was the key. It helped them focus, they engaged in paired testing on their own, they felt more effective and inspired.

The debriefing went so well that it inspired the need for another level of focus – the addition of fictional personas (user profiles) who may use the site in different ways. As a group, we built 7 different personalities for them to focus on. Each tester took a different persona to think about and a new set of questions.

While they explored answers to the questions, the lab seemed more active with their energy as they conjectured some ways the questions may have different answers. The confidence of the newest tester on the team seemed higher as I checked in with him. He proudly demonstrated his resourcefulness in pursuit of the questions, and I agreed it was a marked difference than the previous day.

During the debriefing in the fourth hour, the success was more profound. The team was talking much more and each of the testers agreed that today's testing using OBT was much more effective toward acclimating them with the site’s contents. They seemed to be getting the idea of how to ask interesting questions despite the site being so complex.

Below are the results:

On the left column is the Open-Book question, on the right, a compilation of their answers. On the following page, I have listed the rough outline of fictional personas used to help give them context for the questions.

OBT question	**Answer compilation**
1) List the major MS products or apps that can be downloaded from the MSDN site.	Security Tools, Drivers, Service Packs DirectX 9, SQL Server 2005 Express Edition, Tablet PC 2005 SDK, Virtual Server 2005, SQL Server 2005 Beta 2, Server 2003 SP1 beta, Longhorn Alpha, Visual Studio 2005 Beta, Exchange Server, Windows scripts (?), SDK (general)
2) What's the difference between TechNet and MSDN?	One definition: TechNet geared toward IT (business, admins) -- MSDN toward Developers (has beta testing component)
3) For what purpose do you think the MSDN and TechNet sites were built? (1 paragraph)	Self-help -- somewhere to go to get all their answers; marketing for Microsoft, generating buzz, portal for dev, official word from MS (fixes and patches, beta releases, control content), chat function, newsgroup, wiki -- way to mitigate customer support calls and frustration -- helps tech support and build community. TECHNET -- more of a knowledge base and downloads.
4) What are the top 5 downloads?	TechNet: 1. DirectX 9.0 End User Runtime 2. IE 6.0 sp 1 3. Windows media player 10 4. Windows XP SP2 5. .NET Framework 1.1 MSDN: 1. DirectX 9.0 End User Runtime 2. XML parser SP4 3. Windows scripts 5.6 (admin) 4. Direct 9.0 SDK update 5. MS XML 4.0 sp2
5) Of the top 5 hits when you do a search for "Testing" on the MSDN site, which was of the most interest to you?	Tester #1: Community area: how people are related, how people use the site, how information travels, dev to MS and vice versa, Tester #2: Ladybug: bug reporting tool -- comparison and interop with to Product Studio Tester #3: Communication and support -- documentation in general, troubleshooting capability Tester #4: DevCenter -- Longhorn -- probably going to be a big area Tester #5: Longhorn -- XML -- code samples in VB (not in C#, but would be nice), backend database

6) What is Longhorn? (one sentence)	Longhorn is the next operating system (2006) -- 32- and 64-bit (?) -- 2 .NET Frameworks -- new file storage system -- WinFS -- going to incorporate a database -- designed for security -- TCI -- Avalon - presentation aspect to it -- do away with GDI and do away with DirectX, Indigo -- focus on communications and networking, shift from OO to service-oriented, using that to push themselves away from J2EE.
7) What is Whidbey? (one sentence)	Whidbey is Visual Studio Team System 2005 -- enterprise system -- incorporates Dev, Test, and management -- integrated into Longhorn more than previous versions of VS -- is there standalone capability?
8) What is Ladybug? (one sentence)	Ladybug is the public bug-tracking app for beta users of all the beta versions on the MSDN site (uses Passport).
9) Where can I find the top 10 MSDN Code Samples? (URL)	http://www.msdn.microsoft.com/code
OBT, part II (with personas)	
10) MSDN: What are the steps to download the SOAP Toolkit 3.0?	Downloads, Click here to see the next 40, #22 http://tinyurl.com/6ac6k
11) How would I find the system requirements for Visual Basic 6.0 Upgrade Samples?	Downloads, Click here to see the next 40, #39, see reqs http://tinyurl.com/4cj47
12) Emma found a bluescreen while ejecting a DVD from her laptop. Using MSDN, how would she find a workaround for this problem?	Knowledge base, search for "bluescreen", third link down http://tinyurl.com/3qe7s
13) What operating system was she running when she found this problem?	Windows ME
14) What type of DVD drive did she likely have?	TORiSAN DRD-U424
15) What 3 other support options does she have?	Contact Microsoft, Customer Service, Newsgroups
16) Mike Kinsman is a real PM on the Subscriptions team at Microsoft. According to his blog, what are the 4 things a user can do to reduce the size of their kit?	If you're receiving CDs, switch to a DVD Subscription, Go through your kit and remove redundant/irrelevant discs, Use MSDN Subscriber Downloads for some Archiving, Order fewer languages to receive on Disc.

17) What are the top 5 new things in subscriber downloads?	Systems Management Server 2003 with Service Pack 1 (French) - (November 5) , Office Publisher 2003 (Chinese-Hong Kong SAR) - (November 5), SharePoint Portal Server 2003 Service Pack 1 (Arabic) - (November 5) , Windows Server 2003 Service Pack 1 Beta - Build 1247 (English) - (November 4) , SharePoint Portal Server 2003 Service Pack 1 (English) - (October 27)
18) How do I find the benefits of being a subscriber?	Click the "Not a subscriber?" link on the subscriptions page -- http://www.msdn.microsoft.com/howtobuy/subscribers/
19) How do I find Chris Sells' blog on Longhorn issues?	DevCenter, Longhorn, Editor's blog
20) How can I read the current issue of the MSDN Flash newsletter?	Dev Center, Longhorn, MSDN Flash link to the right, read current issues
21) Where can I get the latest PowerPoint developer information?	DevCenter, PowerPoint, Technical Articles
22) Where can I find code samples to use to spice up my PowerPoint presentations?	DevCenter, PowerPoint, Code Samples and Tools http://www.msdn.microsoft.com/office/understanding/powerpoint/default.aspx
23) How do I register and get started with .NET Developer training sessions?	http://msdn.microsoft.com/vstudio/using/training/seminarlabs/default.aspx
24) How many international languages does the MSDN site offer?	14

Persona #1 -- Ed
Ed is Brazilian and a technophile. He works for his own software development consulting company, and creates web applications for a few select clients. He has his own website. He wants to know as much as possible about Microsoft technology, goes frequently to slashdot, subscribes newsgroups, used to be a white-hat security consultant. He is self-educated, college dropout (but was president of his high school computer club). Uses DevCenter, Downloads, Library, Code Center, Search Engine. (i.e. search for "buffer overrun") -- uses TechNet as well because he IS the only network admin.

Persona #2 – Xavier
Xavier is a Ukrainian developer. He creates applications for a development company (more specialized). He doesn't care about latest and greatest apps, but has a job to do so needs focused solutions. The meter is always running, so he needs relevant answers quickly. Uses Library, Downloads, Code Center, DevCenter. One problem he needs to solve is implementing a thermometer for his company's fund-raising website and implement rotated text -- see http://blogs.duncanmackenzie.net/duncanma/archive/2004/12/02/913.aspx. He participates in chat rooms and newsgroups for only 30 minutes a day. He's an MSDN subscriber. He creates applications for a development company (more specialized). He doesn't care about latest and greatest apps, but has a job to do so needs focused solutions. The meter is always running, so he needs relevant answers quickly. Uses Library, Downloads, Code Center, DevCenter. One problem he needs to solve is implementing a thermometer for his company's fund-raising website and implement rotated text -- see

Persona #3 – Peter
Peter is a Network Admin -- Mixture of hardware and software expertise, focused on tailoring an existing app to meet specific needs (security or productivity). He needs fixes and patches and a deployment strategy (to combat viruses and worms) -- participates in pilot programs. He has to deal with co-workers who do things they're not supposed to do with their PCs, like opening mail attachments and doing SP updates without permission. He administers new user accounts, email settings, firewall settings, new network topologies, scalability, RAS, compatibility. He's not locked down in terms of network privileges on the server. Uses Exchange, SMS, Server 2003, SharePoint for security, deployment, rollout, management, upgrades, productivity and managing downtime of company staff. He's looking for best practices up on TechNet to stay on top of things like SPs, KB articles. He's on a tight budget to acquire new tools and software -- deals with OS configurations, NetMeetings, setting up accounts, VPN -- subscribes to TechNet Flash Newsletter.

Persona #4 -- Lisa
Lisa is a Program Manager -- Focused on managing application development process, manages by overseeing their work, i.e. -- decision-maker who may need to refer to things like SDLC, whitepapers. Needs to keep up with latest technologies that may be discussed at the triage meeting. High-level overview concept only, can let herself get bewildered by the process or overwhelmed with technical jargon if she's not diligent and careful. Goes to MSDN site a few times per week, linked to TechNet through bookmark -- how often finds only some of what she's looking for on the site.

Persona # 5 -- Oscar
Oscar is a college student interested in training and certification so he uses TechNet and MSDN -- unemployed -- IT (CS) -- limited budget -- looking for free stuff, no previous knowledge with programming. Not a programmer, but looking for understanding and expertise about buzzwords (Longhorn, Whidbey, etc). -- non-professional, reads whitepapers, infrequently accesses the site -- found Via a non-Microsoft search engine (e.g. Google, Yahoo!, Altavista, etc.) -- doing research for a term paper -- code samples for school assignments -- wants to learn about C# -- people tell him it is good to know -- wants to be on the cutting edge of technology to increase chances of being hired -- uses library for research.

Persona #6 -- Rachel

Test Lead / IT rollout "goddess" who specifically works with her company's (Safeco) IT dept to do rollouts for new software, including their own as well as MS patches and betas and service packs -- beta user -- Ladybug entry -- wants to find the newest, coolest app - works in support and needs to know how it interact with Longhorn, is responsible for making sure her company's existing software works with the newest MS stuff so she can deliver a story to her customers about interoperability. Subscriber to MSDN quarterly, Early Adopter -- wants to be part of the MS tech community -- very interested in blogs and user groups, newsletter info, looking for opinions and tech support -- they have bought the subscription, MSDN CDs as well as every access capability there is on the sites.

Persona #7 -- Emma

Sales for firewall application software – product interaction with MS software (firewall interacting with ICF; middleware) – hunter/gatherer of info -- needs to interpret technical stuff for herself and clients (sales pitch) Participates in newsgroups/Usergroups for customer responses/issues, find how her answers their needs – Newsletters/Mags – Technet, rather than MSDN – Less detailed tech info, more general. –- Dev center for quick tips/enhancements that can benefit her customers. Will be selling via presentations…gatherer of info from any and all sources (whitepapers, blogs, etc). Needs to be confident of her sources. Security – Consumer level user of techie info.

REFERENCES

Bach, Jonathan and Bach, James (2000), How to Measure Ad Hoc Testing, *Software Testing and Quality Engineering* magazine

Han, Christine (1998) Singapore: Review of Educational Events in 1997, *Asia Pacific Journal of Education* 18(1), p. 88 – 96.

Kaner, Cem and Bach, James (2001), Exploratory Testing in Pairs, *STARWest 2001 Conference*, slide 4

Loi, Loi Soh and Yuan, Wu (1998), Open Book Examinations, p 3. http://www.ntu.edu.sg/nbs/sabre/working_papers/10-98.pdf

Theophilides, Christos and Dionysiou, Omiros (1996), The major functions of the open-book examination at the university level: A factor analytic study. *Studies in Educational Evaluation* 22(2), 157 - 170.

Tussing, L. (1951), A consideration of the open book examination, *Educational and Psychological Measurement 11*, p 597 – 602.