



"turning data into dollars"

Tom's Ten Data Tips – May 2009

Usability Research

Usability is invisible. As long as things are going well, nobody seems to notice. Hardly anyone ever complains that something is *too easy* to use, right? You *measure* usability by the absence of problems that users experience. Usability becomes ever more important as technology permeates our society and increasingly gets used by people with less technical background than, say, 20 years ago.

Usability pertains to many more areas besides Business Intelligence (BI) or even technology. As products become more complex and are getting used by more diverse sets of users, usability deserves more attention. Valid usability metrics ensure that increasingly sophisticated technology with ever expanding functionality remains easy to use. These are guiding beacons for development.

1. Test As Often As You Can

Usability testing works best in conjunction with iterative or "Agile" software development methods. The more you can test (even tiny and short tests) and feed results back *throughout* the development process, the better chance you have to deliver a highly usable product. Therefore tests should take as little time as possible, while still yielding useful information.

But even if you have less 'agility' in your development process, you still want to test for usability. Better one test than no tests. Focus on what you *can* do, rather than what you *cannot* do. Because without any testing, the *entire product life cycle* becomes your de facto iteration...

2. Test Approaches Change Throughout Your Development Cycle

As you are going through development, the kind of testing you want to do will evolve, and likely entails different methods. In the early stages you'll do exploratory testing (sometimes called "formative studies"), then you move to assessment testing (or: "summative testing"), and towards the end validation tests ("verification testing").

Exploratory studies are in order when you are still defining and designing your product. You'll want to examine the effectiveness of preliminary design concepts. Assessment tests take place halfway into development when sufficient functionality can be tested to see how well users can perform full-blown realistic tasks. Validation tests are done very late in development to actually measure usability against an established benchmark.

3. Usability Is Decomposed Of Satisfaction And Performance

There is a gross distinction between self-reported metrics like satisfaction, and performance data. The former are also referred to as subjective data, or preference data. Performance measures are quantitative, subjective measures are either quantitative or qualitative. Often they are used as counterparts. The best time to collect self-reported data is at the end of a task, or at the end of the session.

Both self-reported and performance data have a role to play in usability. In some studies, despite the appearance of struggling users, their subjective evaluation is surprisingly good. If the experience makes them happy, they will accept when it appears to take them forever to perform a task. You capture self-reported data in one of three ways: verbatim, paper forms, or some on-line tool. A number of well researched, standardized instruments exist out there, like SUS (System Usability Scale – freeware) and WAMMI (Website Analysis and Measurement Inventory – commercial) which is available in multiple languages.

4. Emphasize Positive Findings (Too)

The word “usability issue” tends to have a negative connotation, largely because of our experience with the word “issue” in IT. However, *any* remarkable finding is an issue, positive or negative. An overview of the most important issues found in a usability study needs to display positive as well as negative issues, and not just to be “politically correct”, but more so to position yourself as an ally for developers and marketing.

Although reporting positive issues is highly beneficial in maintaining a good relationship with developers, everybody is, or should be concerned foremost with fact finding anyway. The fundamental reason why you explicitly need to report on positive issues is that you want to make sure that those aspects of the interface remain unchanged and are propagated to subsequent design iterations.

5. Define “Success” Unambiguously

Every task in your test should have Successful Completion Criteria (SCC) so that observers will (hopefully) score all subjects alike. It is often remarkable how much disagreement can arise between jury members observing the same participant. SCC define the boundaries of tasks and clarify scoring. Confusion about SCC tends to indicate confusion about product design and functionality. If only for that reason it's worth documenting SCC's carefully.

Criteria for successful completion can be things like reaching a certain screen, a maximum number of errors or wrong turns before finishing, maximum time to perform a task, etc. Successful performance is a reflection of both correct behavior and timely completion.

6. Profile Target Audiences, And Check With Developers

The first step in usability research is to determine the target audience for your product. “Profiling” here means describing the relevant behavior, skills and knowledge of the user. Of course in an ideal world such a description already existed and was used as input for the initial specifications and development.

In the real world, if you ask a few ‘random’ developers who they are building this product for, you are likely to get many different responses. This lack of clear delineation contributes to usability deficiencies in the first place! Another good reason to start testing usability as early in the development process as possible. Testing brings this ambiguity out in the open.

7. Categorize Participants *Objectively*

There are *two* kinds of characteristics you'll want to use to describe participants: those that all users share, and those that discriminate among user groups. Since participants in usability research represent eventual users, the target group is ‘known’ with marketing and developers (but also see tip# 6). They should provide the information to set criteria. It requires perseverance, discipline and great interviewing skills to surface the qualifiers in objective terms, but you are serving a great cause so persist: how can you determine if the product ‘works’ if you don't know who it should work for?

When you define subgroups as “novice” or “expert”, for instance, be crystal clear about the meaning of such terms. The amount of time a user spends with a product, or frequency of usage are often used but

they do not always describe expertise. Some people may use the web a lot, but predominantly for playing specific games which don't contribute much to their expertise. Use observable facts rather than self-reports, wherever you can. When you describe specifically what it means to be "an experienced user", for instance, this helps to further clarify the user population (see also tip# 6).

8. Learnability Drives "Tourist" Adoption

"Learnability" is one of the determinants of performance (along with task success, time-on-task, number of errors, and efficiency). A pragmatic definition for learnability is how much time and effort are required to become proficient. The way you measure this is often by observing how *quickly* Time-on-Task, Error rates, or Efficiency taper off to the expert level. The usual proxies for experience are elapsed time (in the case of continuous learning) or number of trials (minimum of 3-4 trials, but preferably more).

Learnability is particularly important for tasks which are performed infrequently and which are probably *not* a mandatory part of doing one's job. In his classic segmentation of end-users, Bill Inmon has classified this user group as "Tourists." Adoption of BI applications among this group is living proof of a system's user-friendliness. For incidental users this is largely a function of learnability.

9. Time-On-Task (Usually) Relates Directly To ROI

Time-on-Task or "completion time" is one of the best ways to measure efficiency of (most) products. Exceptions like games, portals or e-learning usually strive to *maximize* the time spent, but typically performance is inversely related to time.

The nice thing about Time-on-Task is that you can relate it directly to productivity: if it takes half the time to complete, knowledge workers can output twice as much, tying this measure neatly to ROI and other business goals you are supporting.

10. Testing The Sum Of Parts Is *Not* Testing The Whole

Usability testing *throughout* the development process helps to create great products. And just like component testing needs to be followed by unit testing, you'll want to test the ultimate product, in its entirety. Merely testing the components that make up the product risks a lack of integration. It's desirable to begin testing as early as possible, but

make sure you explain to management what the limits are of component testing.

When you test all the components together ensure there is (still) enough lead time to make revisions as they follow from your findings. Somehow, sometimes, testing tends to get squeezed out of the planning cycle, in particular when delivery dates are tight. The whole integrated product is what makes or breaks the eventual user experience!