The Usability Audit:
A Do-It-Yourself Approach to Improving Usability

Jack Massa

While a useable interface is key to product success, many software development organizations do not have formal processes in place to ensure usability. This session describes a low-cost inspection methodology that can be used to improve the usability of applications and web sites. The technique can be used by various team members—technical communicators, product managers, QA analysts or developers—and can be adapted to any phase of the development cycle. The session includes a case study in which a Usability Audit was designed and conducted for a suite of web applications.

THE BACKSTORY

In April of 2006 I was hired for an unusual assignment. The client is a software and services company that produces web applications for the financial sector. I had been writing technical documentation for their systems for some time, but this assignment came from the Director of Usability. He had been charged with improving product usability by coming up with usability standards.

This task was particularly challenging because the company had arrived at its current form through several mergers and acquisitions. Consequently, the company fielded several different product suites, built on varied platforms by widely-divergent development teams. The Director concluded that in order to produce useful standards, he first needed reliable documentation on the current state of the product interfaces. He proposed to have a *usability audit* conducted and, fortunately, he hired me for the job.

A PRIMER ON USABILITY APPROACHES

In the early stages of the project we discussed various alternate approaches. I was given the opportunity to research a range of usability methods, to better understand how the usability audit would fit into the overall usability approach that was evolving in the company.

A large number of tools and techniques have been devised by usability specialists in industry and academia, and there is a wealth of knowledge about them on the Web. The following table summarizes some of these methods, under a classification scheme used by James Horn.[1]

Inquiry	Inquiry methods involve asking users questions, either about a specific product or about their work. Examples include: **Contextual Inquiry** - A formal observation of how users work in their environment. Focused on the work itself rather than any particular product or tool. **Ethnographic Study / Field Observation** - Similar to Contextual Inquiry, but based on formal methodologies derived from Anthropology. **Interviews and Focus Groups** - Sitting down with users and asking specific questions about a product, either singly or in a group. **Surveys and Questionnaires** - Written lists of questions presented to users.

Inspection	Inspection methods look at a product or interface in some detail and attempt to evaluate its usability. **Heuristic Evaluation** - In this approach, each element of a product is judged against established usability guidelines (heuristics). **Cognitive Walkthrough** - Evaluators construct task scenarios for a product or prototype and then role-play the part of a user performing those tasks. **Formal Usability Inspection** - Adapts the methods of software code inspection to usability. Usually performed on a set of specifications during the design process.
Testing	Testing techniques are experiments conducted with representative users to evaluate a product or prototype. Tests may be recorded in a laboratory setting, or the data may simply be recorded by an observer. Techniques used in testing can include: **Thinking Aloud Protocol** - Users are asked to vocalize their thoughts, feelings or opinions while working on the test. **Question-asking Protocol** - Instead of asking users to vocalize their thoughts, the tester asks specific questions during the test. **Performance Measurement** - The test focuses on quantitative measuring of user performance.

DESIGNING THE USABILITY AUDIT

After surveying the available usability techniques, we decided that the *Heuristic Evaluation* method was the closest match to our requirements. The products already existed, and we needed to determine how closely they followed good usability principles. At the same time, we wanted the evaluation to match users' conditions as closely as possible. We therefore adapted the use of task scenarios from the *Cognitive Walkthrough* method. Our approach, as finally defined, combined these two methods. It differed from a standard Cognitive Walkthrough in that our task scenarios were concise and high-level rather than detailed and rigorous, and that our evaluation focused specifically on a defined set of heuristics.

To perform the audit along those lines, we needed to accomplish several design tasks:

1. Define a set of heuristics to evaluate against
2. Define a set of questions to ask in evaluating each heuristic
3. Define the user task scenarios to test
4. Devise a method for measuring usability problems

As a bonus, these tasks resulted in a set of *reusable tools*, geared toward the company and its products and users, which could be used for future usability evaluations.

Defining the Heuristics

While numerous practitioners have conducted and written about heuristic evaluation, almost invariably the heuristics used have derived from usability expert Jakob Nielsen.[2] The heuristics employed for the Usability Audit are shown in the following table. They were adapted from Nielsen's principles as specifically applied to Internet Banking sites.[3] Additional comments were adapted from Bruce Tognazzini.[4]

Heuristic	Description and Comments
1. Make users feel secure	Users need to feel secure when doing Internet banking. Sites need to be secure, make security measures visible and explain to users how to use sites in the most secure manner, providing appropriate warnings where necessary.
2. Easy navigation	Are there adequate site maps, navigation bars and menus to help users find their way around the site? Are menus broad and shallow? Usable sites do not have deep, narrow and hierarchical menu structures that force users to immerse themselves into the depths of the structure and thus cannot be easily navigated without practice and rote memorization.
3. Visibility of system status	The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. The feedback however must not detract from the perceived or actual security of the web site. Tognazzini – Reduce the user's experience of latency. Acknowledge all button clicks by visual or aural feedback within 50 milliseconds.
4. Match between system and the real world	The system should speak the user's language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. The system should follow real-world conventions, making information display in a natural and logical order.
5. User control and freedom	Users often choose system functions by mistake and need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo. Tognazzini – Make actions reversible; always allow undo.
6. Consistency and standards	Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions – that is to say, do not just make the site internally consistent, but consistent with the majority of other sites. Tognazzini – Consistent interpretation of user actions (such as shortcut keys) is the most important level of consistency. Tognazzini – It is just as important to be visually inconsistent when things must act differently...Avoid uniformity. Make objects consistent with their behavior. Make objects that act differently look different.
7. Error prevention	Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Tognazzini – Protect users' work. Ensure that users never lose their work as a result of error on their part, the vagaries of Internet transmission, or any reason other than the completely unavoidable, such as sudden loss of power to the client computer. Support continuous save.
8. Recognition rather than recall	Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate. For example, provide mouse-over text to explain further where each menu item or link takes you. Tognazzini – Ensure readability of text. Use high contrast and appropriate font sizes.

Heuristic	Description and Comments
9. Flexibility and efficiency of use	The interface should be suitable for novices as well as experienced users. Avoid unnecessary steps toward a user goal, making the process as simple and logical as possible. Convoluted and complex navigation should be avoided, making all parts of the site available from the homepage. Tognazzini – Write help messages tightly and make them responsive to the problem: good writing pays off big in comprehension and efficiency. Tognazzini – Menu and button labels should have the key word(s) first.
10. Aesthetic and minimalist design	Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. Text should be structured by breaking it into separate meaningful chunks to help users scan and locate the information they are seeking.
11. Help users recognize, diagnose, and recover from errors	Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
12. Provide Help	As much as possible, the system should be so intuitive that users do not need to read instructions. However, comprehensive Help is still important, especially for novice users. Help should be easy to search, focused on the user's task, list concrete steps to be carried out, and be written concisely.

Defining the Questions for Each Heuristic

While discovering and describing the list of heuristics was a critical step, we soon realized that by themselves the heuristics would not be enough. To formalize the audit analysis into an understandable and repeatable process, we developed a series of questions to ask in order to judge how well a product adhered to each heuristic.

The heuristics and questions were then formatted into a checklist for evaluation. A portion of the checklist, for Heuristic 2 - Easy Navigation, is shown below:

2. Easy Navigation
Questions to Ask
Do menu choices and links effectively set expectations?
Do menu choices and links match the destination page titles?
Do menus give easy access to any part of the application?
Are procedures that span screens linked by Next and Previous controls if appropriate?
Is the user's place in procedures and in the application plainly indicated?
Are "transition points" to outside vendor sites or functions clearly demarcated and easy to navigate?
Usability Problems/Comments

Defining the User Tasks for Each Product

To simulate real user behavior, a set of tasks were listed for each product. When conducting the audit, the evaluator would attempt to perform each task and then report on how well the product was judged to measure up to the usability heuristics for that task.

The tasks were deliberately kept high-level and generic, and an attempt was made to use tasks that were common across the different products, to facilitate comparison. Some example tasks were:

- Logging in
- Recovering from a forgotten password situation
- Checking an account balance

Devising a Method to Measure Usability Problems

The evaluator would use the Usability Checklist to record usability problems found in the products. In order to quantify these results, we determined that we needed a method to rate the severity of the problems.

The severity scale below resulted from further research. Severity score sheets using this scale were produced and became another of the reusable tools that resulted from the audit project.

Severity Scale	
0	I don't agree this is a usability problem.
1	Slight problem. Correction might be a useful enhancement to the product.
2	Minor problem. Likely to cause users some delay or confusion.
3	Major problem. Likely to cause users significant delay or frustration.
4	Severe problem. Likely to prevent users from completing a task.

CONDUCTING THE USABILITY AUDIT

Conducting the audit then became a process of applying the designed methodology and tools to a set number software products. The evaluator followed these steps for each product:

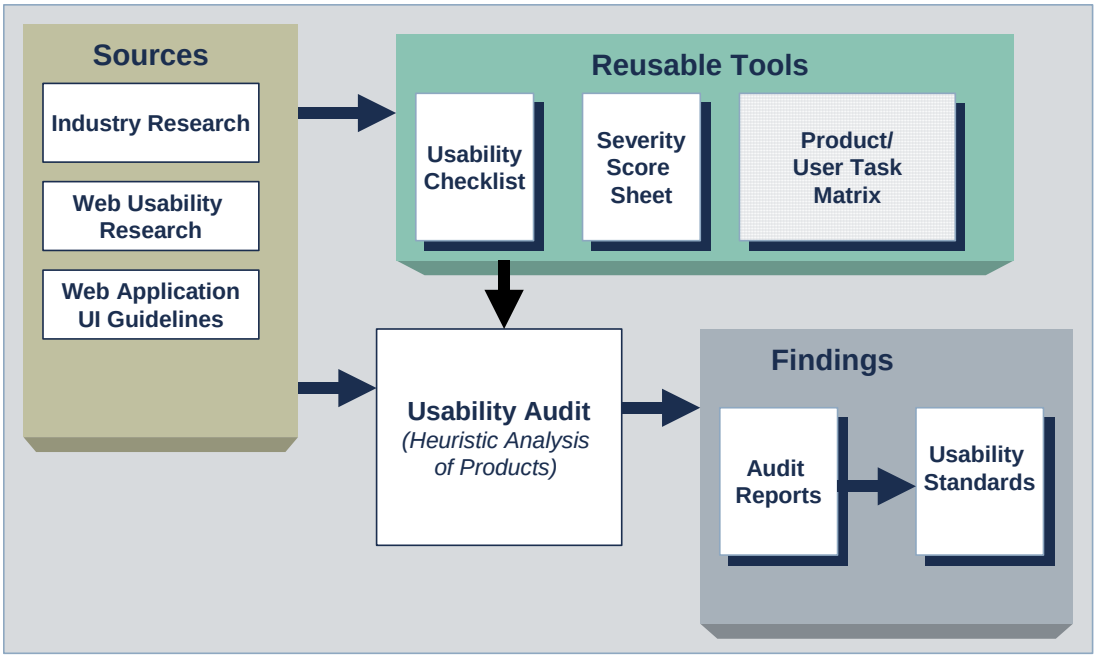
1. Attempted to perform each defined task and observed each user interface component encountered along the way.
2. At each step of each task, evaluated whether the interface satisfied the relevant usability heuristics.
3. Filled in a usability checklist for each task, describing each problem. Particularly usable features and components were also noted as "usability bonuses."
4. After completing all tasks, compiled the checklists for the product, then ranked each problem according to the severity scale.
   Note: The standard procedure in a heuristic evaluation is for all evaluators to meet a day or two after testing to discuss and rank the severity of problems. In our case, resource constraints limited the project to a single evaluator.
5. Produced a report for each product, including a narrative of each task (with screen captures) and the resulting usability checklists and severity score sheets.

RESULTS

As a result of the usability audit project, we were able to provide these benefits to the company:

- A report of usability problems and "plusses" for each evaluated product and task. Product managers received these reports and began to plan fixes for the problems into the development cycle. Many problems were easy to fix, resulting in "quick wins" for the company.
- Reusable tools for conducting future usability evaluations. Usability committees have been established in different product groups and these tools will provide a basis for ongoing usability work. The tools can be used equally well during initial design, prototyping, or for established products.
- At the end of the project, we were able to produce draft *Usability Standards* that focused on the company's users and industry and addressed actual needs in their products. The standards were submitted in draft format to the individual product teams and are being evaluated by the usability committees.

The following figure presents an overview of the usability audit project, including the research and results:



TRYING THIS AT HOME - LESSONS LEARNED

A usability audit is a low-cost and low-resource technique that can produce surprisingly good results. The skills needed—evaluating interfaces, paying careful attention to detail, organizing and documenting results—are well-matched to the skill sets of most technical communicators.

If you are interested in applying this work to your own situation, I recommend you consider a few of the lessons that we learned:

- **Obtain executive buy-in**. It is critical that you have an executive sponsor who understands the potential benefits and has the political clout to obtain the resources and cooperation you will need.
- **Sell early and often**. From the beginning of the project, reach out to all the groups that may be involved or who may benefit from your results. Meet with product management, QA, and especially Development teams to inform them of what you are doing and seek their input. Remember, even if your results are excellent, if these groups do not implement them the effort will be a failure.
- **Focus on actual user tasks**. When conducting an inspection, simulate the actual work of real users as closely as possible. This will ensure that your findings are applicable to the business problems that the products are meant to solve.
- **Create tools for reuse**. Produce reusable tools to allow the company to leverage your efforts. For usability to succeed long-term, it must be part of the ongoing design and development cycle.
- **Present standards and recommendations that are clear and easy to implement**. If you get to the point of recommending standards, make sure they are documented clearly, using concrete examples. Usability standards often fail because developers simply do not understand how to apply them.

Notes

[1] James Horn, *The Usability Methods Toolbox,* at http://jthom.best.vwh.net/usability/index.htm.

[2] Jakob Nielsen, *Heuristic Evaluation*, at http://www.useit.com/papers/heuristic/

[3] David Wenham and Panayiotis Zaphiris ,"User Interface Evaluation Methods for Internet Banking Web Sites: A Review, Evaluation and Case Study," at http://www.soi.city.ac.uk/%7Ezaphiri/Papers/HCII2003/HCII2003-Banks.pdf

[4] Bruce Tognazzini, "First Principles of Interactions Design" at http://www.asktog.com/basics/firstPrinciples.html

Jack Massa is an independent writer and online information developer based in Atlanta. Since 1984, Jack has written and edited marketing copy, training, documentation, and articles covering all kinds of information technology. He's designed and developed web sites, help systems, multimedia projects, and web-based training; taught workshops to corporate clients and at industry conferences; and managed teams for large and small companies.

In 1996, Jack founded Guidance Communications, Inc. (www.guidancecom.com) to provide communication services to technology-industry clients. In addition to STC, Jack is a member of CHI-Atlanta, the Technology Association of Georgia, and the Atlanta Chapter of ASTD. He is also the web master for the Atlanta-based Freelance Forum. Jack is a past vice-president of the STC Atlanta Chapter. In 2004, he was named an Associate Fellow of STC.