Expert Heuristics Review Scorecard

Version 1.0

Mark
Bailey

Executive Summary

An Expert Review was conducted to determine the strengths and weaknesses of <Project Name>, user experience (UX), and user interface (UI). The focus of this review was to how employees experience the <Project Name> and the possibly experience the beta currently being developed.

<project name> executive summary

What is an Expert Review?
An Expert Review is a usability inspection method for software that helps to identify usability problems in the user interface (UI) design. It specifically involves evaluators examining the interface and judging its compliance with recognized usability principles (the "heuristics" or “rules of thumb”). Too learn more about UI Heuristics, there are examples in the Appendix section of this document.

Expert Reviews usually are conducted by a small set (one to three) of evaluators. The evaluators independently examine a user interface and judge its compliance against a set of usability principles. The result of this review is a list of potential usability issues or problems.

The usability principles, also referred to as usability heuristics, are taken from published lists. Ideally, each potential usability problem is assigned to one or more heuristics to help facilitate fixing the problem. As more evaluators are involved, more true problems are found.

On the following pages, you will see an Expert Review Scorecard (based on a 100 point scale). The scorecard is placed into context by providing high-level observations – intended to serve as discussion points and focus items for user research and usability testing. The review is specifically base on the user experience – and not reflective of the functionality of the backend processing.

Topic	#	Usability / Design Issue	Points possible	Current Score	Beta Score
1. High-Level Structure and Navigation					
	1	Logical default state of application	3	0	0
	2	Clear, comprehensible navigation structure	3	0	0
	3	Each screen provides sense of place	3	0	0
	4	Primary tasks are promoted, prominent	3	0	0
	5	Supports frequency of use	3	0	0
	6	Primary, secondary interaction is good	3	0	0
			Subtotal	0 of 18	0 of 18
2. Windowing Elements & Design					
	1	Appropriate Application posture	3	0	0
	2	Effective top level menu organization	4	0	0
	3	Effective menu item design and integration	4	0	0
	4	Good toolbar design	3	0	0
	5	Good status bar design	3	0	0
	6	Good dialog box design	4	0	0
			Subtotal	0 of 21	0 of 21
3. Layout & Presentation					
	1	Screen supports task flow	3	0	0
	2	Effective Visual hierarchy	2	0	0
	3	Reduced Visual complexity	2	0	0
	4	Grouping similar functions/elements	2	0	0
	5	Strong sense of underlying grid	2	0	0
	6	Effective use of color and contrast	2	0	0
	7	Effective and consistent visual language	1	0	0
	8	Clear label organization and text presentation	2	0	0
	9	Appropriate wording, terminology level	2	0	0
	10	Functional use of graphics (icons, graphs, etc)	2	0	0
			Subtotal	0 of 20	0 of 20
4. Interaction					
	1	Interaction optimized for input device	4	0	0
	2	Selection of controls supports user task	4	0	0
	3	Conventional control behavior	3	0	0
	4	Appropriate interaction styles supported (e.g. mouse, keyboard, right mouse, direct manipulation)	4	0	0
	5	Speed of interaction and response time	2	0	0
	6	Prevention of errors – format cues, controls, help	2	0	0
	7	Good system feedback approaches	3	0	0
	8	Good error message design	3	0	0
			Subtotal	0 of 25	0 of 25

5. Motivation					
Trust	1	Establishes trustworthy reputation.	3	0	0
	2	Avoids negative language (hard sales, self-interest)	2	0	0
Emotion	3	Evokes visceral reactions.	3	0	0
	4	Optimal level of stimulation and challenge.	3	0	0
Persuasion	5	Assertions made are convincing.	2	0	0
	6	Persuasive statements are used in context.	3	0	0
			Subtotal	0 of 16	0 of 16
			TOTAL	**0 of 100**	**0 of 100**

Observations

What works well?

Redesign Opportunities

Appendix A

List of Heuristics – Rules of Thumb (source: Jakob Nielsen 1994):

1. **Visibility of system status -** The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
2. **Match between system and the real world** - The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
3. **User control and freedom** - Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
4. **Consistency and standards -** Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
5. **Error prevention** - Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
6. **Recognition rather than recall** - Minimize the user's memory load by making 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.
7. **Flexibility and efficiency of use** - Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
8. **Aesthetic and minimalist design** - Dialogues should not contain information which 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.
9. **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.
10. **Help and documentation** - Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.