Jef Raskin *The Humane Interface*

Chapter I

Human-computer interactions are not complicated because they can do complicated tasks; simple tasks should always be simple to perform.

Interface design is not limited to the visual/graphical (GUI)—it extends to any kind of interactions the user has with the system.

Design should be based on "universal psychological facts" rather than "industry standards".

Many tools are resistant to change because {Microsoft, Apple, …} won't make major interface changes—they favor flawed but familiar paradigms to major overhauls.

Interface design often included too late in the *design cycle,* at a point when parts of the system that should be changed are already set (due to commitment of resources, dependent components, etc.). Interface design should be included from the beginning and be part of an incremental process. "As far as the customer is concerned, the interface *is* the product"

example: having to tell the computer to save work, losing work, and wasted time

humane interface: An interface is *humane* if it is responsive to human needs and considerate of human frailties.

"Users should set the pace of interaction."

Chapter II

cognetics: the ergonomics of the mind. Just as we have physical limits, we have cognitive limits as well (e.g. multiplying two 30-digit numbers in five seconds).

cognitive conscious and *cognitive unconscious:* (un)conscious mental processes are mental processes that you're (not) aware of as they're going on

Stimuli can pull things into your conscious thought, pushing other things out.

People cannot maintain more than one conscious thought process at a time.

absorption: As a conscious demands more attention, it becomes more difficult to monitor stimuli outside of the attention-demanding task.

locus of attention: Human attention is like a spotlight—it illuminates one thing at a time.

Many stimuli are perceived outside of the locus of attention and ignored.

Humans form habits after repetition; it is our natural tendency to learn tasks to the point where they become automatic, and it is very difficult to focus our attention on preventing habituation.

Multiple conscious tasks performed at the same time interfere with one another, but unconscious activities can be performed simultaneously (e.g. walking and chewing gum). *Automaticity* allows for multi-tasking.

example: Answering "yes" to the prompt for confirming deleting a file becomes a habit, which defeats the purpose. Using a task that prevents habituation (e.g. a task that changes subtly every time) would solve the problem but draw attention away from the user's end task and annoy the user. A better solution would be to allow for easy undos.

Moral of the story: One can take advantage of/pander to the nature of human attention, for instance: making use of time required to swap attention, using stimuli to make delays seem to pass more quickly, and resuming interrupted tasks in the place where the user left off.

Chapter III

Content is what the computer stores that users care about.

gesture: a sequence of human actions completed automatically once set in motion (the result of psychological *chunking* of the separate actions)

Gestures can have different meanings in different *modes*. This can be a source of confusion (e.g. state-based toggles vs. radio buttons). Modality is defined with respect to specific gestures: a system can be modal with respect to one gesture but not another.

Modes can change the effects of habitual actions—they can cause errors or draw away the user's locus of attention from the intended task. Stimuli indicating the mode can help, but they don't come near to solving the problem. Not assigning different meaning to the same gesture in different modes can also help. *Transient modes* (ones that go away after a number of gestures) can cause fewer errors than non-transient modes.

range: the set of states where a gesture has a particular meaning—modes limit range

modal message boxes: message boxes that demand your attention and disable gestures that don't pertain to the message box (i.e. graying out the background windows)

Preferences and customization make for an obtuse way of specifying modes and making mode-changes difficult. The interface designer should design an effective interface, not rely on the inexpert user to pick one. Users often waste time trying to get their customizations right. Users wind up optimizing for likeability, which is different from usability and efficiency.

Sometimes a limited set of gestures makes the use of modes necessary; other times multiple modes are used to cut down on the number of buttons on a device.

quasimodal (quasimode): modes that are maintained kinesthetically, such as the shift key—due to the constant stimulus of physically maintaining the mode, mode errors are few. Sometimes quasimodes create a game of hand-twister on the keyboard. They are most often used for commands controlling the system, rather than content entry, because they take more physical effort, and content entry is presumably the more common task.

Noun-verb interactions are inherently less modal than verb-noun interactions—the user typically chooses the object of the action (i.e. the content) before the action itself, and focus can immediately return to the content after the action takes place. Verb-noun interactions necessitate a "cancel" gesture when the user wants to choose a different action; selecting different nouns, or content, is easy, and removes the need for a "cancel" gesture when the user wants to choose a different noun. (Another way of thinking about this is that the "normal" content-entry mode is the one that allows for selecting the noun.) In some cases, such as palettes in painting tools, the verb-noun paradigm is the accepted norm and has been found to be more effective.

Features of an interface can be *visible* (accessible via human perception) or *invisible;* with invisible features, the user has to memorize where they are or hunt for them.

affordance: The function of a feature can be determined just by looking at it.

monotony: There exists exactly one gesture to achieve a given effect.

Sometimes monotony is broken because of backwards compatibility issues.

Monotony makes it easier for the user to acquire automaticity—choosing methods draws attention away from the task and decreases repetition

In the absence of monotony, users often wind up using only one of the gestures, effectively monotonizing the interface.

beginner-expert dichotomy: Raskin claims it is false.

An adaptive interface is a bad idea—it prevents users from habituating.

Dividing up users into classes often makes sweeping assumptions and removes focus from the individual interactions.