Strike a balance: Users' expertise on interface

by Mike Padilla

User Experience Design Director
Radian Guaranty

Computers and users process information in distinct ways -- so do individual users. Although it's relatively easy to get a computer to understand input, what with fixed standards and universal APIs, usability with human users is not absolute. User-interface usability is relative to the experience level of individual users. UI designer Mike Padilla provides an overview of UI design for Web-based productivity software with a focus on the broadest range of users, examining what makes an application UI-usable and detailing concepts that can facilitate an efficient, broad-based UI design.

Crafting a well-designed user interface (UI) is not a trivial task; it's the one thin layer that lies between the human user and the not-so-human computer. The fundamental UI design challenge is bridging the communication rift that results from the distinct ways in which the computer and the user process information. The bridge through which this task is done is the relatively narrow informational bandwidth of a user interface.

The often overlooked, underlying challenge when designing a UI for Web-based applications is to design a UI for users with multiple levels of expertise. Just as computers and users process information in distinct ways, specific individual users also process information in distinct ways. Getting the computer to understand input is relatively easy -- it has fixed standards and universally understood APIs.

Not so with users -- with users, usability is not absolute. Interface usability is relative to the individual user. An application may be highly usable by one set of users and completely unusable by another.
The task for UI designers is to design a UI that strikes a balance to best achieve overall system efficacy among all potential users. I start by looking at what makes a UI usable.

**What makes a UI usable?**

Is it just ease of use? What about user efficiency? Does the UI look like something that would be enjoyable to use?

You can judge the quality of a UI with a handful of factors:

- Ease of learning and memorability
- Efficiency of use
- Error frequency, severity, and recovery
- Subjective satisfaction

Now look at an ideal UI. It would:

- Be easy to learn
- Have a high maximum efficiency of use
- Have low error frequency
- Deliver subjective satisfaction to the user

The car you drive every day, especially if it is a Porsche 911 Turbo (talk about subjective satisfaction), is an example of a great UI in the physical world. It would score highly on all four of the above characteristics (although some may argue that the constant mechanic tuning violates the "low error" canon, and that the increased time-in-garage could lower user satisfaction -- see, it's hard to find even a hypothetical ideal analogy).

But UI designers don't have the luxury that automobile designers possess. Designers don't have large volumes of space to craft a near symbiotic, perfectly ergonomic, physical interface between man and machine. Designers are granted a mere 800 by 600 pixels (1280 by 1024 pixels if they're lucky) to bridge a mental and psychological gap between man and computer. The reality is quite a different affair.

**The reality**

If you take a closer look at each of the four factors -- ease-of-learning, maximum efficiency of use, low error frequency, and subjective satisfaction -- the waters begin to muddy because you notice that these factors may conflict with one another.

An application might be dumbed down sufficiently to explicitly explain everything to the user, making it easy to learn but reducing the maximum efficiency of use because of the resulting screen clutter. An application can be stripped to just keystroke commands, making it efficient to use once it
has been mastered, but difficult to learn and prone to user errors. And, an application can be easy to learn and use with little error frequency, yet it may just be a flat experience.

I'm not talking about having clowns walk across your screen, but interacting with a visually well-designed system (such as Mac's Aqua interface, for example) can surely add to your subjective satisfaction while using an application.

![Figure 1: Which UI looks more enjoyable to use?](image)

With the UI design objectives in mind, take a peek at the difficulties that a designer might face when designing a UI.

**Difficulties in designing a UI**

UI design is compounded by three factors:

- Complexity of the information to be communicated
- Informational bandwidth of the interface that the designer has available
- Differential between the two communicating parties

**Expounding on the three factors**

Web application UI designers are faced with the "perfect storm" of these combined factors. With Web applications, designers are charged with communicating complex conditional functionality across a narrow informational pipeline between computer and user. You can think of it as a computer connected to a user by a pipe fitted with a filter through which all information must flow. The following three sections quickly detail the breakdown.

**Complexity**

The complexity of the information is the volume that must be pushed through the pipe. The more volume, the tougher the task. With a Web application, as opposed to a simple brochureware Web site, the user is tasked with loads of information tied into complex functionality. *(A brochureware Web site is basically just an online brochure that offers users one-way interaction -- read only.)*
**Figure 2: The more complex the functionality and data, the tougher the task**

**Bandwidth**

The informational bandwidth is the diameter of the pipe. The narrower the pipe, the tougher the task.

With a Web application, the pipe diameter is limited in all directions. The UI must be designed within the constraints of the webographics of the user population. This includes:

- The computing platform (PC, MAC, or even WebTV!).
- The connection speed (such as 56K modem or DSL).
- The monitor's resolution (640-by-480, 800-by-600, 1280-by-1024).
- The browser and version (for example, Netscape 3.0 or Microsoft Internet Explorer 6.0).
Filter

Finally, the complexity of the filter acts as the differential between the two communicating parties. Humans and computers process information in fundamentally distinct ways. The UI needs to communicate functionality that is cognitively processed by the user in addition to dictating the motor skills (eye movement, mouse and pen tracking, keyboard typing) necessary to interact with the system so that input information can be processed unambiguously by the computer.

Figure 4: The UI forms the differential between users/user-machine interaction

(In Figure 4, "input" on the user side refers to input into the user; "output" refers to output from the user.)

The challenging task is to provide an effective interface to communicate between the computer and the user, two high-powered computational
resources (and some are more high-powered than others), through the relatively narrow bandwidth of the interface that connects the two, the user interface.

The job gets even tougher as the user base diversifies.

**From 1-to-1 to 1-to-many**

The design task gets tougher because the one-to-one relationship between computer and user quickly explodes into a one-to-many relationship when a Web application is deployed to hundreds, sometimes thousands, of users. A single application has to interact with many users, each with a unique level of intelligence, training, and computer savvy.

While designing the UI so that both novice and expert users may use it effectively is important, striking a balance between these two often-conflicting user bases poses the greatest difficulty.

For novice users, clarity is of the utmost importance; for expert users, efficiency. Icons and shortcut keys afford efficiencies for expert users while a fully expounded UI affords clarity for novice users. Unfortunately, satisfying the extremes (as well as the spectrum of users between) is the problem.

The UI elements that afford efficiencies for expert users cause confusion for novice users. The details required of a fully expounded UI that facilitate understanding for novice users clutter the system for expert users. It boils down to this: Clarity comes at a price -- that price is clutter.

![Figure 5: The clutter/simplicity, clarity/efficiency trade-off](image)

By understanding the user population of the application and using a design that *upgrades gracefully* (more on this in the section on useful design...
The challenges that result from UI design for a diverse user base are especially evident when working on a Web application, mostly because of the unique nature of the application.

In an office Web setting, app users are often expected to efficiently process business transactions through the use of the application to contribute to overall productivity. They must use the application extensively in the course of their jobs, making it possible to learn advanced UI interactions. Training and documentation is often available to facilitate the learning of advanced UI interactions. This results in the development of a group of expert users.

As a networked application, it can be quickly and easily deployed across a wide set of users. Expert users will continually develop, but novice users will be introduced to the system due to both personnel turnover and application deployment expansion. These factors contribute to creating an assorted user base.

With such a user base, UI designers are freed of some of the more traditional considerations (and corresponding limitations) of traditional brochureware and basic application Web site design that is usually focused solely on the novice user as the lowest common denominator. (Examples of this type of Web site design include corporate Web sites and online shopping and banking applications in which the software user is only an occasional user. This article is not focusing on that type of application.)

Such tasks as capturing user attention, designing all UI interactions with obvious clarity, and leveraging recognition over recall are no longer top priorities. The user base of a Web application should be considered a trainable corps with the UI design addressing the entire spectrum of the corps, from novice to expert, so that work efficiencies across all users can be optimized.

Neither extreme should be neglected, but remember that most large gains in efficiencies come from the expert users because of their larger work throughput.

**Who are experts and novices?**

Novice users are characterized by their *unfamiliarity* with the application. This unfamiliarity must be overcome by training, a help system, and an explicitly designed UI. Icons are seen as cryptic symbols and keyboard shortcuts are unknown.

Expert users on the other hand, are characterized by their near *symbiotic relationship* with the application. They

---

**Fitts's Law**

Fitts's Law is a robust model of human psychomotor behavior developed in 1954. The model is based on time and distance and lets users predict human motion based on rapid, aimed movement, rather than on drawing or writing efforts.

P.M. Fitts discovered that
have essentially programmed themselves, through training and hours of use, to seamlessly operate within the confines of the application. Explicit instructions and even required mouse interactions are seen as obstacles in their efficient work modes.

For expert users, the motor load is what limits the efficiency of using the application, while with novice users the cognitive load serves as the choking point. After working on the same screen for the hundredth time, the expert user no longer has to fully cognitively process the UI to learn how to work with it. The expert user hardly has to think. But the information still has to be mechanically input from user to computer and here is where the motor load can strangle the usage efficiency of the application.

Such issues as small and poorly arranged click targets (Fitts's Law states that the time to acquire a target is a function of the distance to and size of the target) and requiring the user to frequently switch from keyboard to mouse can severely slow down the use of an application.

While this constraint also applies to novice users, it pales in comparison to the novice's ultimate usage constraint. When a user first begins to use an application, each screen must be visually scanned, read, and understood. This cognitive process is time consuming. Not until the user has understood each screen can the user decide how to interact with the UI.

To help with the cognitive understanding of the screen, the UI needs to be rich in explanatory elements. This goes back to the original dilemma in design for both novice and expert users. As the information density of a screen increases, even expert users may have to re-think each screen to wade through the ensuing clutter. The explanatory value of additional information required by the novice users comes at the expense of the expert users.

Mathematically, Fitts's Law looks like this: \( MT = a + b \log_2(2A/W) \). \( MT \) is the movement time; \( a, b \) are the regression coefficients; \( A \) is the distance of movement from start to target center; and \( W \) is the width of the target.

A disadvantage of Fitts's Law is that it is inherently one-dimensional -- it predicts movement in only one dimension. When using a 2D target-acquisition task, you have to consider a new interpretation of \( W \), the target width.

Fitts's Law also doesn't have a consistent technique for handling errors (a method exists, but it isn't widely used due to its complexity).
**Useful design concepts**

So how can you design a single UI to reconcile the different usage requirements of these polar groups? The answer: Through a combination of transparent functionality that *upgrades gracefully* and good old-fashioned compromise.

Some UI interactions can be available to all users but have limited visibility so that novice users will not be overwhelmed. Good examples of this include keyboard shortcuts, setting the correct tab order across field elements to reflect workflow, and supporting data entry by both keyboard and point-and-click methods. This is what I mean by design that *upgrades gracefully*.

Where such design is not possible, compromise comes into play. Other UI elements, like icons, can be designed to be more easily understood by novice users, yet remain unobtrusive to expert users. For example, a *Comments* icon could show a notepad along with the three small letters, "C-M-T." (See Figure 11.)

The level to which you compromise the UI design (towards the novice or the expert) depends on your users. An application that is used periodically or has a user base with high turnover is best suited by a UI for novice users. Conversely, an application that is used frequently by a stable user base would require a UI designed for expert users.

Look at some of these design elements already mentioned and the concepts behind them.

**Providing subtle shortcuts**

Figure 7 shows a search button that anyone can understand. It is also equipped with a keyboard shortcut (access key = "s") and an unobtrusive visual cue to the shortcut, the underscored "S" in the word "Search." Listing 1 shows the code for this image.
Listed 1: Code for the keyboard shortcut

As the user gains proficiency, he can execute the same action pressing the button as by typing Alt-S.

Tying tab order to workflow

By default, the tab order of form fields in a Web page follow the order in which they are listed in the HTML. To customize the tab order to better reflect work flow, you can explicitly define the tab order by using the tabIndex property.

In the example in Figure 8, the default tab order would be left to right. To set the tab order to work down each column instead, the tabIndex property can be set for each field (see Listing 2).

<table>
<thead>
<tr>
<th></th>
<th>Jason Snyder</th>
<th>Leslie Snyder</th>
<th>Troy Snyder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Overtime:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Bonuses:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Commissions:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Dividend/Interest:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Net Rental Income:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Subject Net Cash Flow:</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Figure 8: Adjusting tab order

Listing 2: Code for tab ordering

Supporting multiple methods of data entry

By allowing both keyboard and point-and-click data entry, all users of the application can work efficiently.

Figure 9 demonstrates how expert users can quickly enter a list of delimited codes to input their data.
For novice users who have yet to memorize the codes, they can click on the lookup book icon and be presented with a full description of each code. They can simply click on the codes they would like to select and the textbox populates with the corresponding codes (Figure 10). Not only does this help the novice do the work; it also facilitates the learning curve.

<table>
<thead>
<tr>
<th>Type</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DT</td>
<td>A Lender Credit has been entered in the Deletion Transaction. Verify that the lender is not an interested party to the transaction. If the lender is with an interesting party to the transaction, lender credits plus the seller contribution do not exceed the limits specified in the Selling Guide. Then the condition is true.</td>
</tr>
<tr>
<td>DT</td>
<td>The following accounts are not listed on the payment. For each liability, provide written verification. [Borrower, Creditor Name, Payment, Balance]</td>
</tr>
<tr>
<td>DT</td>
<td>The credit report does not show 12 months payments. Obtain written verification.</td>
</tr>
<tr>
<td>DT</td>
<td>The following mortgage/home equity liability is not included on the credit report.</td>
</tr>
</tbody>
</table>

Meaning-laden icons

Once a graphic is understood by a user, the user recognizes the graphic faster than he does the text. Unfortunately, because of size limitations, some icons become cryptic and difficult to understand. Does that mean that you should abandon them altogether? Of course not. Instead, an icon can offer the user the explanatory value of an abbreviation in conjunction with the recognizable value of the graphic, all within a small plot of pixels. Figure 11 shows the Comment icon I mentioned at the start of this section.

Advocating for the experts

Creating a well-designed UI that is focused on the needs of expert users has its own set of project-team communication challenges.

As with nearly all Web design initiatives, UI designers often serve as the conciliating hub to multiple, often conflicting, interest groups -- marketing, business experts, business analysts, developers, and the all-important
users. When designing the UI, it is important to solicit feedback from all these groups.

Each group often has different feedback on the UI, but the common aspect is that the user is viewing it as a novice (since the application UI doesn't yet exist). The challenge this presents to the designer is obvious.

A business analyst may argue that an icon is too small and its meaning unclear. Instead, why not use a large icon with a complete textual description? Well, he is exactly right, if you are designing a system solely for novice users. But you're not, so step into the shoes of an expert user who has used the system for 200 hours.

Will this user want to see this big icon on every page? What about the textual description? Would it help to know that this expert doesn't use the icon but instead opts to use a keyboard shortcut?

Since almost all of the interest groups involved in a new Web design project represent the view of the novice user, it is important for the designer to be an advocate for the expert user during the design stage of UI development. The expert's point of view is often neglected because expert users are unavailable and novice users (everyone who has an opinion) are everywhere.

The task of the UI designer is to filter the kernels of valuable feedback at this stage while strongly advocating for the expert users based on existing knowledge about the usage behavior of expert users with other applications.

**Clicking the shut down icon**

Designing an effective UI is quite the challenge. Getting a computer to understand a human user is tough. Getting a computer to understand thousands of individually unique human users, as is the case with a Web application, is exponentially tougher.

From the novice to the expert (and all users in between), you must design the UI so that all users can use it effectively. Unfortunately, certain interactions may benefit one population of users while detracting from another. Providing additional clarity to novice users by designing an explicitly described UI nearly always comes at an expense charged to the expert users. Clarity comes with clutter.

To optimally design a UI that maximizes overall system use efficiency, successful UI designers balance the benefits of the clarity provided to the novice users with the drawbacks of clutter weighed upon the expert users.

---

**Resources**

- In "Emerging frameworks for tangible user interfaces," explore the
philosophy of developing user interfaces at theory level, including the introduction of a model-control-representation, physical-and-digital (MCRpd) model of development, similar to the model-view-controller (MVC) scheme.

- See "Fitts's UI Law Applied to the Web" to apply Fitts's Law (certain characteristics of objects on the screen make the objects easy or hard to click on) can be applied to make Web applications more usable.

- Discover the power of customer guidance when designing a Web application user interface in "Learn from your customers for usable Web apps" by Paul Englefield (developerWorks, June 2003).

- Read "Debunking the myths of UI design" by Paul Smith and blast the myths around user interface design and its costs and benefits (developerWorks, February 2002).

- Filter your way to reduced UI data and UI functions with "Reducing the user interface" by Mark Molander (developerWorks, June 2001).

- Read "Human-facing Web services," a three-part series by Judith Myerson, for insight on user-prompted, Web-application interface design, focusing on the Web Services Experience Language (WSXL) standard (developerWorks, September 2002-January 2003).

- Get another analysis of the Apple Aqua interface (mentioned in this article) in "The cranky user: Drowning in aqua" by Peter Seebach (developerWorks, April 2002).

- Explore User-Centered Design Training at IBM for courses to help you create e-business and other Web applications that are easy for the user to access.

- Concerned with the user experience? Visit the IBM Ease of Use site for perspectives on user-centered design and engineering. The focus is on such design issues as design concepts, Web and OOBE guidelines, OVID methodology, and exploratory user interfaces.

- Check out Human Factors International and find a plethora of downloadable goodies, including articles, booklets, a UI design newsletter, and a Q&A column.

- Try the Usability Special Interest Group newsletter archive, a repository on articles dealing with Web usability issues.

- Get the definiton of Fitts's Law, a robust model of human psychomotor behavior, developed in 1954, based on time and distance, and used to predict human movement and human motion based on rapid, aimed movement, and not on timed drawing or writing efforts.

- Find practical advice about how to explain complex material by visual means in Envisioning Information by Edward R. Tufte, (Graphics Press, 1990) -- still one of the best resources.

NOTE: This article was originally published on IBM developerWorks.
For more information on the products or services discussed in this article, please click here and follow the instructions provided. Thank you!