

Human-Computer Interface Design Principles

Author: Jeff Davis
Date: 09/1/00

Human-Computer Interface Design Principles

This white paper presents a set of principles useful for designing web sites and digital products for Chromosome22 clients. The reason for defining a set of design principles is to help us build a product that meets the standards of Chromosome22. Also, these principles can help clarify what one can do in order to design a product based on what is known about people and how they operate in the world.

One undoubtedly finds out that one can't design in accordance with all of the principles all of the time. In that type of situation, one has to make a decision based on which principle or set of principles is most important in the context of the task one is solving.

Metaphors

One can take advantage of people's knowledge of the world around them by using metaphors to convey concepts and features of your product. Use metaphors involving concrete, familiar ideas and make the metaphors plain, so that users have a set of expectations to apply to computer environments. For example, people often use file folders to store paper documents in their offices. Therefore, it makes sense to people to store computer documents in computer-generated folders that look like file folders. People can organize their hard disks in a way that's analogous to the way they organize their file cabinets.

Menus are an extension of a desktop metaphor. People can connect the idea of making choices from a computer menu with making choices from a restaurant menu. Although people don't keep restaurant menus on the edge of their desks, using the term menu in the computer environment reinforces the idea that people can use computer menus to make choices.

Direct Manipulation

Direct manipulation allows people to feel that they are directly controlling the objects represented by the computer. According to the principle of direct manipulation, an object on the screen remains visible while a user performs physical actions on the object. When the user performs operations on the object, the impact of those operations on the object is immediately visible. For example, a user can move a file by dragging an icon that represents it from one location to another or can position a cursor in a text field by directly clicking the location where the cursor should be placed.

In addition to expecting physical results from their actions, users want their tools to provide feedback. For example, when a drawing tool is moved, a line appears in the document on which the user is working. Users want to see what actions are available at any given moment. If grave consequences might follow from any of those actions, they want to know about those consequences—before any damage is done and while they can still change their minds. They want clues that tell them that a particular command is being carried out, or, if it cannot be carried out, they want to know why not and what they can do instead. Users also want topics of interest to be highlighted.

Animation, when used sparingly, is one of the best ways to show a user that a required action is being carried out. For example, animated pointers reassure the user, during a lengthy process such as downloading a large document to disk, that the computer is completing the task without any problems.

See-and-Point

Within an interface, users perform actions by choosing from alternatives presented on the screen. Users interact directly with the screen, selecting objects and performing activities by using a pointing device, typically a mouse, to point at elements on the desktop.

Chromosome22's user interface designs work according to a fundamental paradigm: that users can see on the screen what they're doing and that users can point at what they see. The paradigm is based on a general form of user action: noun-then-verb. In this paradigm, the user selects an object of interest (the noun) and then chooses the actions to be performed on the object (the verb). All actions available for the selected object are listed in the menus, so users who are unsure of what to do next can refresh their memory by scanning through the menus. At any time, users can choose any available action without having to remember any particular command or name. For example, a user clicks a document icon (the noun) and then prints (the verb) the document by choosing Print from the File menu.

Consistency

Consistency in the interface allows people to transfer their knowledge and skills from one product to any other. Use standard elements within interface design to ensure consistency within your product and to benefit from consistency across products.

Effective products are consistent in a number of different ways. Consistency in the visual interface helps people learn and then easily recognize the graphic language of the interface—for example, once users know what a checkbox looks like, they don't have to learn another symbol for making choices. Consistency in the behavior of the interface means that people have to learn how to do things such as clicking and pointing only once; then they can explore new products or new types of features using skills that they already have. In general, consistency benefits the typical user, who usually divides working time among several products, and it benefits software developers because their users can build on prior experiences with elements in other products when learning how to use a new product.

The following are some questions we can ask you when thinking about consistency in our client's products.

- + is our product consistent?
- + within itself?
- + with earlier versions of our product?
- + with Chromosome22 Interface standards?
- + in its use of metaphors?
- + with users expectations?

Note that the most difficult kind of consistency to achieve is matching people's expectations. Because you often face a wide audience and a range of expertise, it's difficult to meet the expectations of everyone. You can address this problem by carefully weighing the consistency issues in the context of your target audience and their needs.

WYSIWYG (What You See Is What You Get)

Don't hide features in your product by using abstract commands. People should be able to see what they need when they need it. For example, menus present lists of commands so that people can see their choices instead of having to remember and type commands.

People should be able to find all the available features in your product.

If you find a need to initially "hide" features, do it in a way that gives people information about where they can find more choices. A stepped interface, by revealing relevant information to users in steps, shows the choice most users want most of the time while providing a way for the user to get more choices.

Make sure that there is no significant difference between what the user sees on the screen and what the user receives after printing. Let the user be in charge of both the content and the format (spatial layout as well as font choices) of the document. When the user makes changes to the document, quickly and directly display the results; the user shouldn't have to wait for a printout or make mental calculations of how the document shown on the screen will look when it appears on the printed page.

User Control

Allow the user, not the computer, to initiate and control actions. People learn best when they're actively engaged. Too often, however, the computer acts and the user merely reacts within a limited set of options. In other instances, the computer "takes care" of the user, offering only those alternatives that are judged "good" for the user or that "protect" the user from having to make detailed decisions. This approach mistakenly puts the computer, not the user, in control.

The key is to create a balance between providing users with the capabilities they need to get their work done and preventing them from destroying data. For situations in which a user may destroy data accidentally, you can help the user by providing warnings, usually in the form of an alert box, to notify users of a potentially undesirable situation and still allow them to proceed, if they confirm that this is what they want. This approach "protects" users but allows them to remain in control.

Feedback and Dialog

Keep users informed about what's happening with your product. Provide feedback as they do tasks and make that feedback as immediate as possible. When a user initiates an action, provide some indicator, visual or auditory (or both), that your product has received the user's input and is operating on it. Provide as much information as possible about how long operations take. When your product can't respond to user input because it's processing a different task, inform the user of what to expect and describe any delays, why they occur, and how long they'll take. Also, tell the user how to get out of the current situation whenever possible.

Provide direct, simple feedback that people can understand. Most people would not know what to do if they saw this message: "The computer unexpectedly crashed. ID = 13." It would be very helpful if the message spelled out exactly which situation caused the error—for example, not enough memory was available for the computer to complete the task—so that the user could understand how to avoid the situation in the future.

Forgiveness

You can encourage people to explore your product by building in forgiveness. Forgiveness means that actions on the computer are generally reversible. People need to feel that they can try things without damaging the system; create safety nets for people so that they feel comfortable learning and using your product.

Always warn people before they initiate a task that will cause irremediable data loss. Alert boxes are a good way to warn users of this kind of situation. Note, however, that when options are presented clearly and feedback is appropriate and timely, learning how to use a program should be relatively error-free. This means that frequent alert boxes are a good indication that something is wrong with the program design.

Perceived Stability

Sophisticated and powerful applications often introduce a new level of complexity for people. If people are to cope with this complexity, they need some stable reference points. Chromosome22 interface standards are designed to provide users with environments that are understandable, familiar, and predictable.

To give users a visual sense of stability, Chromosome22 interface solutions should a number of consistent graphics elements (menu bar, window border, and so on) to maintain the illusion of stability. Note that it is the perception of stability that you want to preserve, not stability in any strict physical sense. To give users a conceptual sense of stability, the interface provides a clear, finite set of objects and a clear, finite set of actions to perform on those objects. Even when particular actions are unavailable, they are not eliminated from a display but are merely dimmed.

Aesthetic Integrity

Aesthetic integrity means that information is well-organized and consistent with principles of visual design. This means that things look good on the screen and the display technology is of high quality. Since people spend a lot of their time working while looking at the computer screen, design your products to be pleasant to look at on the screen for a long time.

Keep the graphics of the display simple. The number of elements and their behaviors should be limited to enhance the usability of the human-computer interaction, and dialog boxes, and so on—be the basis of effective human-computer interaction and must be designed with that in mind. Don't clutter the screen with too many windows, overload the user with complex icons, or put dozens of buttons in dialog boxes.

Make sure to follow the graphic language of the interface and don't change the meaning of standard items. For example, if you sometimes use checkboxes for multiple choices and other times for exclusive choices, you dilute the meaning of the element. Don't use arbitrary graphic images to represent concepts. When you add nonstandard symbols to menus, dialog boxes, or other elements, the meaning may be clear to you, but to other people the symbols may appear as something different and distracting. If you need symbols other than standard ones, use graphic images that convey meaning through representation, analogy, or metaphor.

Don't use arbitrary graphic elements

In general, match the graphic element with users' expectations of its behavior. Push buttons appear as though they push in rather than slide sideways. Indicators in sliders slide along to change values. These behaviors map to people's expectations of how these elements behave.

Give users some control over the look of their computer environments. This allows them to display their own style and individuality. It also reduces the burden on the designer of trying to create an interface that appeals to every user. When a user sets up his or her computer environment in a certain layout, it should stay that way until the user changes it.

Modelessness

For the most part, try to create modeless features that allow people to do whatever they want when they want to in your product. Avoid using modes in your product because a mode typically restricts the operations that the user can perform while it is in effect. It locks the user into one operation and doesn't allow the user to work on anything else until that operation is completed. In contrast, modelessness allows the user to perform more than one operation at a time and thus gives the user more control over what he or she can do on the computer and in a product. As much as possible, you want to preserve the user's ability to be in control of the task and the order of operations.

This is not to say that you should never use modes in products. Sometimes using a mode is the best way out of a particular problem. Most acceptable modes fall into one of the following categories:

- + Long-term modes, such as doing word processing as opposed to graphics editing. In this sense, each product is a mode.
- + Short-term "spring-loaded" modes, in which the user must constantly do something to maintain the mode. Examples are holding down the mouse button to scroll text or holding down the Shift key to extend a text selection.
- + Alert modes, in which the user must rectify an unusual situation before proceeding. Keep these modes to a minimum.

Other modes are acceptable if they do one of the following:

- + They emulate a familiar real-life situation that is itself modal. For example, choosing different tools in a graphics product resembles the real-life choice of physical drawing tools.
- + They change only the attributes of something, not its behavior. The boldface and underline modes of text entry are examples.
- + They block most other normal operation of the system to emphasize the modality, as in error conditions incurable through the software (for example, a dialog box that disables all menu items except Close).

If a product uses modes, there must be a clear visual indicator of the current mode, and the indicator should be near the object most affected by the mode. It should also be very easy for users to get into or out of the mode (such as by clicking a different palette symbol).

Knowledge of Your Audience

Identifying and understanding your target audience are among the most important first steps when you start designing your product. To create a product that people can and will use, study the people who make up your target audience.

It's useful to create scenarios that describe a typical day in the life of a person you think uses the type of product you're designing. Think about the different workspaces, tools, and constraints and limitations that people deal with. You can also visit actual work places and study how people do their jobs.

Analyze the steps necessary to complete each task you anticipate people wanting to accomplish. Then design your product to facilitate those tasks, using a step-by-step approach by thinking of how a person might get from one place to the next in a logical fashion.

Involve users throughout the design process and observe them working in their environment. Use people who fit your audience description to test your prototypes and development products. Listen to their feedback and try to address their needs in your product. Develop your product with people and their capabilities, not computers and their capabilities, in mind.

Accessibility

The site or product should be accessible to everyone who chooses to use it. There are likely to be members of your target audience who are different from the "average" user that you envision. Users will undoubtedly vary in their ages, styles, and abilities. They may also have physical or cognitive limitations, linguistic differences, or other differences you need to consider. Identify how the individuals in your target audience differ and what special needs they may have.

Make it easy for users to interact with your product using different input devices and output devices. If you develop specialized hardware and software for people with physical limitations, work with product developers so that their software supports your products.

Make your product accessible to people around the world by including support for worldwide capabilities in your designs from the beginning of your development process. Take stock of the cultural and linguistic needs and expectations of your target audiences.

General Design Considerations

These considerations cover three broad areas to consider as you begin your development process:

- + Worldwide compatibility—support for multiple script systems and multicultural sensitivity to your target audience
- + Universal access—provisions for people with disabilities to use your software by means of alternative input devices or output devices.
- + Collaborative computing—support for people working in groups on local networks or people using computers over remote networks

If you are aware of the factors that influence worldwide compatibility, universal access, and collaborative computing, you can plan ahead. By incorporating support for these capabilities from the beginning of your development process, you can save time, money, and development problems and end up with a product that is immediately useful to a wide range of people.

Worldwide Compatibility

239 countries, 6700 languages, 147 currencies, 24 time zones and 10 calendars.

Chromosome22 sites and products are designed to address the complex problems encountered when products are designed to be compatible with regional, linguistic, and writing system differences around the globe.

It's much easier to include worldwide compatibility from the beginning of your development process than to try to incorporate support for script systems after your product is complete. This may mean that you create your product so that it is easy to localize, or adapt for use in a specific area. Localizing software involves translating a product's menus, dialog boxes, alert boxes, and content areas into a language or regional dialect.

Cultural Values

Make sure that visible interface elements can be localized for other regions around the world. Whenever you design a user interface, consider that differences exist in the use of color, graphics, calendars, text, and the representation of time in various regions around the world. For example, different cultures use different objects to store documents. In the United States, file folders are flat and have tabs that can indicate the contents of the folder. In Europe, file folders are more like narrow cardboard boxes. You may want to localize elements of the user interface, such as graphics or the colors of text in versions of your product designed for different regions.

Graphics have the potential to enhance your product, but they can also be offensive. In addition to colors, cultures assign varying values and characteristics to living creatures, plants, and inanimate objects. For example, in the United States the owl is a symbol of wisdom and knowledge, whereas in Central America the owl represents witchcraft and black magic. It's a good idea to avoid the use of seasons, holidays, or calendar events in software that you expect to distribute worldwide. Also avoid using graphics that represent holidays or seasons, such as Christmas trees, pumpkins, or snow—or be sure that the symbols can be localized.

Different calendars are used to mark time around the world. The United States and most of Europe observe time according to the Gregorian calendar. The traditional Arabic calendar, the Jewish calendar, and the Chinese calendar are lunar rather than solar. Often time is marked according to one calendar for business and government purposes and according to a different calendar for religious events. Make your product flexible in handling dates; you also may want to provide the user with a way to change the representation of time. Use the text utilities to handle numbers, dates, and sorting.

Resources

It's essential to store region-dependent information in resources so that text the user sees can be translated during localization without modification of your product's codes. When you create resources, consider text size, location, and direction. Text size varies in different languages. Also, depending on the script system, the direction of text may change. Most Middle Eastern languages read from right to left. Text location within a window should be easy to change.

Language Differences

Translating text is a delicate task and can often cause confusion, so be wary of using colloquial phrases or nonstandard usage and syntax. Carefully choose your words for command names in menus and for messages in dialog boxes, alert boxes, and help balloons. When translated, text can become up to 50 percent larger than U.S. English text. Text needs room to grow up, down, and sideways.

Potential grammar problems may arise with error messages and the user programming structure of phrases such as HyperTalk. Use complete sentences whenever possible. Don't use phrases that you then concatenate to create sentences. The word order of messages may become completely different in translation, rendering such a message nonsensical when translated. For example, word order in German usually places the verb at the end of a sentence.

Text Display and Text Editing

Chromosome22 sites and products should allow users to display different scripts at the same time. A script system consists of resources that support a writing system for a human language. Writing systems may differ in the direction in which their characters and lines flow, the size of the character set used, and whether certain characters are context dependent. Whenever a user installs a non-Roman script system, at least two scripts are available, the Roman script that is present on all Western computers and the non-Roman script. No matter what level of worldwide text support you provide, it's important to avoid four common assumptions:

- + Characters aren't necessarily 1 byte; they can be 2 bytes.
- + Text isn't always left aligned and read from left to right.
- + A person doesn't always read text; it may be spoken through a text-to-speech converter.
- + System and product fonts aren't always Chicago and Geneva or Helvetica and Times.

Default Alignment of Interface Elements

When dialog boxes are localized, the text in the dialog box may vary with localization. For example, Arabic and Hebrew are written from right to left, so the alignment of items in an Arabic or a Hebrew dialog box is generally right to left, just as dialog box items in English or Russian are generally left to right.

When the alignment of items is reversed, it's important that the elements appear vertically aligned. Therefore, when you create dialog box items, make sure that their display rectangles are the same size.

Another common problem occurs when dialog box items are longer than the boundaries of the dialog box. In this case, when the text direction is reversed, the text appears outside of the dialog box and isn't visible on the screen. Be sure to make your dialog box items shorter than the width of the dialog box.

Fonts

When you write software that supports non-Roman scripts, don't make assumptions about font sizes; let the user choose them. For example, system or product fonts may be preset to 12 or 18 points. A font with a resource ID of 0 is not always set to Chicago, nor are system fonts always Chicago and Geneva. Use system and product fonts when the user cannot choose the font, but don't hard code Roman names.

In some scripts and fonts, diacritical marks may extend beyond the ascent line. Other fonts, such as Japanese fonts, contain glyphs that extend to the boundaries of the enclosing rectangle of the font, or to both minimum-y and maximum-y lines. Leave room for space between lines of text and between the top and bottom lines of any enclosing rectangle.

Universal Access

Providing universal access means creating products that all people can use, including people who have a disability. Approximately forty-three million people in the United States alone have some type of disability. Computers hold tremendous promise for people with many kinds of disabilities.

In terms of increasing productivity and mobility, computers can have a far greater impact on people with a disability than on other users.

When one thinks about designing for the wide range of abilities in one's target audience, think about increasing the amount of productivity for the entire audience and be careful not to overcompensate for the special needs of certain members of the group. Don't add features for disability access that get in the way of able users. The features you include should be additional ways to access the hardware or software, not primary ways of input and output that make it more difficult for other, non-disabled users.

Collaborative Computing

Collaborative computing is a shared computing environment or product that facilitates communication and teamwork among groups of people. These people have specific needs that may have ramifications for your software.

When one designs products for use by groups of people, the products should follow all the standard guidelines and principles of the desktop interface. There are several additional concerns to consider when you design collaborative or multi-user interfaces. Our main concern is to ensure a good user experience for groups of people.

Concern for Other Users

When people work in groups, sharing information among computers over networks, usually one person controls some resource that other people are using simultaneously. A shared resource may be a document with data in it, a product, a storage medium, or some other resource. The user in control of the shared resource needs to be aware of the consequences to other current users, particularly in situations in which the user in control wants to cease sharing or disconnect other users from the shared resource. Create a structure that allows clear communication about the shared resource between the users in a group.

User Identification

When people share information, it is important to be able to identify all the people who are participating in the collaboration. You should allow the owner of shared resources to be able to obtain as much information as possible, such as the name, location, and access privileges, about the people who are using the shared resource. Finding out that another person is using a machine as a "guest" provides no useful information and prevents the owner from further communication in case of a problem or situation that requires it.

Access Privileges

Collaborative products typically allow users to share data with other people, providing different access rights to different people. For example, one user may want to allow some people to change a document and allow other people simply to read it. The owner may want to restrict some people from seeing the document at all.

Provide a simple, clear way to assign access privileges to shared information and then clearly display to users what those assigned privileges are. One typical problem with multi-user programs is that they have an interface that makes it easy to provide unintentional access to other users without making this mistake apparent. Try to avoid this compromised security state by making it clear to users what information is shared and available to others.

Passwords

In addition to access privileges, password schemes often protect shared resources such as collective data. When you provide a password system, make the interface to it as clear as possible. Follow these guidelines concerning passwords:

- + Allow passwords to contain both alphabetic and numeric characters.
- + Never display the password on the screen in clear text, not even while the user is typing it. A common method of providing feedback to the user is to display a bullet character for each character that the user types. When the user edits a password, the Delete key erases one character in a system that displays a character for each character typed.
- + Provide a way for the user to verify the password when it is entered or changed. Requiring the user to enter the password two times minimizes the possibility of a typing error.
- + If a person makes a mistake in entering the password but doesn't have to verify it, he or she will then be denied access to the data.

Data Encryption for Security

When people share information with collaborative or networking products, users need to be able to trust that the information is secure from unwanted intruders. If data is stored on a communal file server or if it is sent across network connections, it may easily be inspected by unauthorized users. Therefore, you should encrypt sensitive data, such as passwords, in some way to prevent this intrusion from happening. Make sure you let the user know whether their data has been encrypted or not; they need to know if their data is protected from other users.

Clear Communications

Collaborative computing extends the interface from the realm of human-to-computer communication to human-to-human communication. When people communicate over a network, certain aspects of everyday social interaction, such as voice inflection, gesture, facial expressions, and body language, are lost. You can enhance people's ability to communicate clearly on the computer by supporting the ability to include contextual clues, including font styles and sizes, punctuation, colors, graphics, sounds, and animation.

Displaying the Current State of Data

When many people work on the same data, it must be clear to everyone what is happening at all times. Because some data may be visible to more than one person at a time but editable by only one person, you need to provide a visual clue about the current state of the data. (Make sure that the visual clue is accessible to users with a visual disability, particularly if they require the aid of some kind of special device or software.) The current state of the data may vary from moment to moment or from session to session. If one person is editing the data, other people should not be able to edit it at the same time. It is very important to make clear to all users which information they may change and which they may not. If you can provide more information to users about why some information is presently unavailable for change, it will save them from wasting time trying to change unavailable information.

When there is a change to data that is visible to more than one user at a time, display the change immediately. Such feedback is very important to people who are working in cooperative environments.