SHNEIDERMAN • PLAISANT • COHEN • JACOBS • ELMQVIST

SIXTH EDITION
GLOBAL EDITION

DESIGNING THE USER INTERFACE

STRATEGIES FOR EFFECTIVE HUMAN-COMPUTER INTERACTION

Pearson

Boston Columbus Indianapolis New York San Francisco Hoboken
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montréal Toronto
Delhi Mexico City São Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

CHAPTER 3

Guidelines, Principles, and Theories

> We want principles, not only developed—the work of the closet—but applied, which is the work of life.

Horace Mann
Thoughts, 1867

> There never comes a point where a theory can be said to be true. The most that anyone can claim for any theory is that it has shared the successes of all its rivals and that it has passed at least one test which they have failed.

A. J. Ayer
Philosophy in the Twentieth Century, 1982

CHAPTER OUTLINE

3.1 Introduction

User-interface designers have accumulated a wealth of experience and researchers have produced a growing body of empirical evidence and theories, all of which can be organized into:

1. *Guidelines*. Low-level focused advice about good practices and cautions against dangers.
2. *Principles*. Middle-level strategies or rules to analyze and compare design alternatives.
3. *Theories*. High-level widely applicable frameworks to draw on during design and evaluation as well as to support communication and teaching. Theories can also be predictive, such as those for pointing times by individuals or posting rates for community discussions.

In many contemporary systems, designers have a grand opportunity to improve the user interface by applying established guidelines to clean up cluttered displays, inconsistent layouts, and unnecessary text. These sources of debilitating stress and frustration can lead to poorer performance, minor slips, and serious errors, all contributing to job dissatisfaction and consumer resistance.

Guidelines, principles, and theories, which offer preventive medicine and remedies for these problems, have matured in recent years (Grudin, 2012). Reliable methods for predicting pointing and input times (Chapter 10), better social persuasion principles (Chapter 11), and helpful cognitive or perceptual theories (Chapter 13) now shape research and guide design. International or national standards, which could be described as commonly accepted and precisely defined so as to be enforceable, are increasingly influential (Carroll, 2014).

This chapter begins with a sampling of guidelines for navigating, organizing displays, getting user attention, and facilitating data entry (Section 3.2). Then Section 3.3 covers some fundamental principles of interface design, such as coping with user skill levels, task profiles, and interaction styles. It presents the Eight Golden Rules of Interface Design, explores ways of preventing user errors, and closes with a section on ensuring human control while increasing automation. Section 3.4 reviews micro-HCI and macro-HCI theories of interface design.

3.2 Guidelines

From the earliest days of computing, interface designers have written down guidelines to record their insights and to try to guide the efforts of future designers. The early Apple and Microsoft guidelines, which were influential for

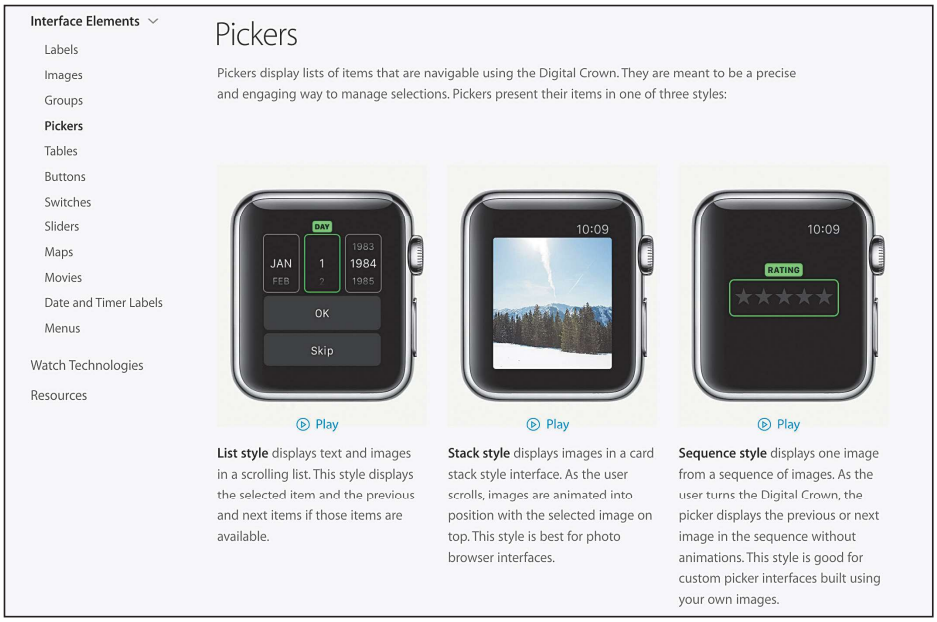


FIGURE 3.1
Example of Apple guidelines for designing menus for the iWatch.

desktop-interface designers, have been followed by dozens of guidelines documents for the web and mobile devices (Fig. 3.1) (see the list at the end of Chapter 1). A guidelines document helps by developing a shared language and then promoting consistency among multiple designers in terminology usage, appearance, and action sequences. It records best practices derived from practical experience or empirical studies, with appropriate examples and counterexamples. The creation of a guidelines document engages the design community in lively discussions about input and output formats, action sequences, terminology, and hardware devices (Lynch and Horton, 2008; Hartson and Pyla, 2012; Johnson, 2014).

Critics complain that guidelines can be too specific, incomplete, hard to apply, and sometimes wrong. Proponents argue that building on experience from design leaders contributes to steady improvements. Both groups recognize the value of lively discussions in promoting awareness.

The following four sections provide examples of guidelines, and Section 4.3 discusses how they can be integrated into the design process. The examples address some key topics, but they merely sample the thousands of guidelines that have been written.

3.2.1 Navigating the interface

Since navigation can be difficult for many users, providing clear rules is helpful. The sample guidelines presented here come from the U.S. government's efforts to

promote the design of informative webpages (National Cancer Institute, 2006), but these guidelines have widespread application. Most are stated positively ("reduce the user's workload"), but some are negative ("do not display unsolicited windows or graphics"). The 388 guidelines, which offer cogent examples and impressive research support, cover the design process, general principles, and specific rules. This sample of the guidelines gives useful advice and a taste of their style:

> *Standardize task sequences.* Allow users to perform tasks in the same sequence and manner across similar conditions.
>
> *Ensure that links are descriptive.* When using links, the link text should accurately describe the link's destination.
>
> *Use unique and descriptive headings.* Use headings that are distinct from one another and conceptually related to the content they describe.
>
> *Use radio buttons for mutually exclusive choices.* Provide a radio button control when users need to choose one response from a list of mutually exclusive options.
>
> *Develop pages that will print properly.* If users are likely to print one or more pages, develop pages with widths that print properly.
>
> *Use thumbnail images to preview larger images.* When viewing full-size images is not critical, first provide a thumbnail of the image.

Guidelines to promote accessibility for users with disabilities were included in the U.S. Rehabilitation Act. Its Section 508, with guidelines for web design, is published by the Access Board (http://www.access-board.gov/508.htm), an independent U.S. government agency devoted to accessibility for people with disabilities. The World Wide Web Consortium (W3C) adapted these guidelines (http://www.w3.org/TR/WCAG20/) and organized them into three priority levels, for which it has provided automated checking tools. A few of the accessibility guidelines are:

> *Text alternatives.* Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols, or simpler language.
>
> *Time-based media.* Provide alternatives for time-based media (e.g., movies or animations). Synchronize equivalent alternatives (such as captions or auditory descriptions of the visual track) with the presentation.
>
> *Distinguishable.* Make it easier for users to see and hear content, including separating foreground from background. Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.
>
> *Predictable.* Make Web pages appear and operate in predictable ways.

The goal of these guidelines is to have webpage designers use features that permit users with disabilities to employ screen readers or other special technologies to give them access to webpage content.

3.2.2 Organizing the display

Display design is a large topic with many special cases. An early influential guidelines document (Smith and Mosier, 1986) offers five high-level goals for data display:

1. *Consistency of data display.* During the design process, the terminology, abbreviations, formats, colors, capitalization, and so on should all be standardized and controlled by use of a dictionary of these items.
2. *Efficient information assimilation by the user.* The format should be familiar to the operator and should be related to the tasks required to be performed with the data. This objective is served by rules for neat columns of data, left justification for alphanumeric data, right justification of integers, lining up of decimal points, proper spacing, use of comprehensible labels, and appropriate measurement units and numbers of decimal digits.
3. *Minimal memory load on the user.* Users should not be required to remember information from one screen for use on another screen. Tasks should be arranged such that completion occurs with few actions, minimizing the chance of forgetting to perform a step. Labels and common formats should be provided for novice or intermittent users.
4. *Compatibility of data display with data entry.* The format of displayed information should be linked clearly to the format of the data entry. Where possible and appropriate, the output fields should also act as editable input fields.
5. *Flexibility for user control of data display.* Users should be able to get the information from the display in the form most convenient for the task on which they are working. For example, the order of columns and sorting of rows should be easily changeable by the users.

This compact set of high-level objectives is a useful starting point, but each project needs to expand these into application-specific and hardware-dependent standards and practices.

3.2.3 Getting the user's attention

Since substantial information may be presented to users, exceptional conditions or time-dependent information must be presented so as to attract attention (Wickens et al., 2012). These guidelines detail several techniques for getting the user's attention:

- *Intensity.* Use two levels only, with limited use of high intensity to draw attention.
- *Marking.* Underline the item, enclose it in a box, point to it with an arrow, or use an indicator such as an asterisk, bullet, dash, plus sign, or X.
- *Size.* Use up to four sizes, with larger sizes attracting more attention.
- *Choice of fonts.* Use up to three fonts.

- *Blinking.* Use blinking displays (2–4 Hz) or blinking color changes with great care and in limited areas, as it is distracting and can trigger seizures.
- *Color.* Use up to four standard colors, with additional colors reserved for occasional use.
- *Audio.* Use soft tones for regular positive feedback and harsh sounds for rare emergency conditions.

A few words of caution are necessary. There is a danger of creating cluttered displays by overusing these techniques. Some web designers use blinking advertisements or animated icons to attract attention, but users almost universally disapprove. Animation is appreciated primarily when it provides meaningful information, such as for a progress indicator or to show movement of files.

Novices need simple, logically organized, and well-labeled displays that guide their actions. Expert users prefer limited labels on fields so data values are easier to extract; subtle highlighting of changed values or positional presentation is sufficient. Display formats must be tested with users for comprehensibility.

Similarly highlighted items will be perceived as being related. Color-coding is especially powerful in linking related items, but this use makes it more difficult to cluster items across color codes (Section 12.5). User control over highlighting is much appreciated, for example, allowing cellphone users to select the color for contacts that are close family members or for meetings that are of high importance.

Audio tones, like the clicks in keyboards or cellphone ring tones, can provide informative feedback about progress. Alarms for emergency conditions do alert users rapidly, but a mechanism to suppress alarms must be provided. If several types of alarms are used, testing is necessary to ensure that users can distinguish between the alarm levels. Prerecorded or synthesized voice messages are a useful alternative, but since they may interfere with communications between operators, they should be used cautiously (Section 9.3).

3.2.4 Facilitating data entry

Data-entry tasks can occupy a substantial fraction of users' time and can be the source of frustrating and potentially dangerous errors. Smith and Mosier (1986) offer five high-level objectives as part of their guidelines for data entry (Courtesy of MITRE Corporate Archives: Bedford, MA):

1. *Consistency of data-entry transactions.* Similar sequences of actions speed learning.
2. *Minimal input actions by user.* Fewer input actions mean greater operator productivity and—usually—fewer chances for error. Making a choice by a single mouse selection or finger press, is preferred over typing in a lengthy string of characters. Selecting from a list of choices eliminates

the need for memorization, structures the decision-making task, and eliminates the possibility of typographic errors.
A second aspect of this guideline is that redundant data entry should be avoided. It is annoying for users to enter the same information in two locations, such as entering the billing and shipping addresses when they are the same. Duplicate entry is perceived as a waste of effort and an opportunity for error.

3. *Minimal memory load on users*. When doing data entry, users should not be required to remember lengthy lists of codes.
4. *Compatibility of data entry with data display*. The format of data-entry information should be linked closely to the format of displayed information, such as dashes in telephone numbers.
5. *Flexibility for user control of data entry*. Experienced users prefer to enter information in a sequence that they can control, such as selecting the color first or size first, when clothes shopping.

Guidelines documents are a wonderful starting point to give designers the benefit of experience (Fig. 3.2), but they will always need processes to facilitate education, enforcement, exemption, and enhancement (Section 4.3).

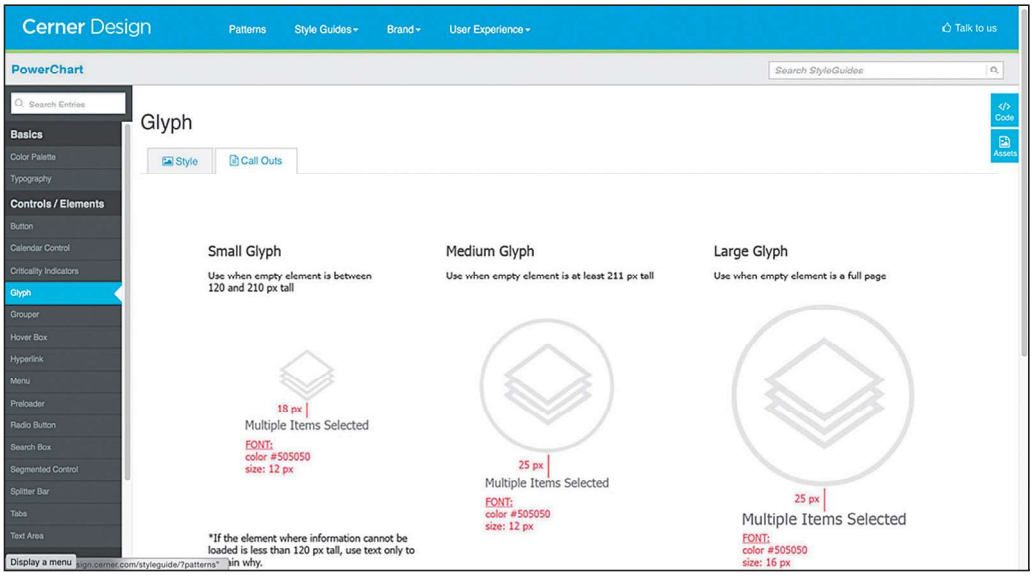


FIGURE 3.2

The guidelines website for Cerner designers and developers. This particular guideline describes the three sizes of icons, or glyphs, that should be used in all electronic health record products (each consisting of hundreds of screens). The 100-plus programmers who develop the user interfaces can also access code samples. Each guideline includes references (http://design.cerner.com).

3.3 Principles

While guidelines are low-level and narrowly focused, principles are more fundamental, widely applicable, and enduring. However, they also tend to need more clarification. For example, the principle of recognizing user diversity makes sense to every designer, but it must be thoughtfully interpreted. A preschooler playing a computer game is a long way from a legal librarian searching for precedents for anxious and hurried lawyers. Similarly, a grandmother sending a text message is a long way from a highly trained and experienced air-traffic controller. These examples highlight the differences in users' background knowledge, frequency of use, and goals as well as in the impact of user errors. Since no single design is ideal for all users and situations, designers who characterize their users and the context of use are more likely to produce successful products.

Chapter 2 introduced the individual differences that designers must address in pursuing universal usability. This section focuses on a few fundamental principles, beginning with accommodating user skill levels and profiling tasks and user needs. It discusses the five primary interaction styles (direct manipulation, menu selection, form fill-in, natural language, and command language) and the Eight Golden Rules of Interface Design, followed by a section on error prevention. Finally, it covers controversial strategies for ensuring human control while increasing automation.

3.3.1 Determine users' skill levels

Learning about the users is a simple idea but a difficult and, unfortunately, often undervalued goal. No one would argue against this principle, but many designers simply assume that they understand the users and the users' tasks. Successful designers are aware that people learn, think, and solve problems in different ways. Some users prefer to deal with tables rather than graphs, with words instead of numbers, or with rigid structures rather than open-ended forms.

All design should begin with an understanding of the intended users, including population profiles that reflect their age, gender, physical and cognitive abilities, education, cultural or ethnic backgrounds, training, motivation, goals, and personality. There are often several communities of users for an interface, especially for web applications and mobile devices, so the design effort is multiplied. Typical user personas—such as nurses, doctors, storekeepers, high school students, or children—can be expected to have various combinations of knowledge and usage patterns. User groups from different countries may each deserve special attention, and regional differences often exist within countries. Other variables that characterize user personas include location (for example, urban versus rural), economic profile, disabilities, and attitudes toward using

technology. Users with poor reading skills, limited education, and low motivation require special attention.

In addition to these personas, an understanding of users' skills with interfaces and with the application domain is important. Users might be tested for their familiarity with interface features, such as traversing hierarchical menus or drawing tools. Other tests might cover domain-specific abilities, such as knowledge of airport city codes, stockbrokerage terminology, or map icons.

The process of getting to know the users is never-ending because there is so much to know and because the users keep changing. However, every step toward understanding the users and recognizing them as individuals with outlooks different from the designer's own is likely to be a step closer to a successful design.

For example, a generic separation into novice or first-time, knowledgeable intermittent, and expert frequent users might lead to these differing design goals:

1. *Novice or first-time users.* True novice users—for example, bank customers making their first cellphone check deposit—are assumed to know little of the task or interface concepts. By contrast, first-time users are often professionals who know the task concepts well but have shallow knowledge of the interface concepts (for example, a business traveler using a new rental car's navigation system). Overcoming their uncertainties, via instructions or dialog boxes, is a serious challenge to interface designers. Restricting vocabulary to a small number of familiar, consistently used concept terms is essential. The number of actions should also be small so that novice and first-time users can carry out simple tasks successfully, which reduces anxiety, builds confidence, and provides positive reinforcement. Informative feedback about the accomplishment of each task is helpful, and constructive, specific error messages should be provided when users make mistakes. Carefully designed video demonstrations and online tutorials may be effective.

2. *Knowledgeable intermittent users.* Many people are knowledgeable but intermittent users of a variety of systems (for example, business travelers filing for travel reimbursements). They have stable task concepts and broad knowledge of interface concepts, but they may have difficulty retaining the structure of menus or the location of features. The burden on their memories will be lightened by orderly structure in the menus, consistent terminology, and interfaces that emphasize recognition rather than recall. These features will also help novices and some experts, but the major beneficiaries are knowledgeable intermittent users. Protection from danger is necessary to support relaxed exploration of features and usage of partially forgotten action sequences. These users will benefit from context-dependent help to fill in missing pieces of task or interface knowledge.

3. *Expert frequent users.* Expert "power" users are thoroughly familiar with the task and interface concepts and seek to get their work done quickly. They demand rapid response times, brief and non-distracting feedback, and the shortcuts to carry out actions with just a few clicks or gestures.

Designing for one class of users is relatively easy; designing for several is much more difficult. When multiple user classes must be accommodated, the basic strategy is to permit a *multi-layer* (sometimes called *level-structured* or *spiral*) approach to learning. Novices can be taught a minimal subset of objects and actions with which to get started. They are most likely to make correct choices when they have only a few options and are protected from making mistakes—that is, when they are given a *training-wheels* interface. After gaining confidence from hands-on experience, these users can choose to progress to ever-greater levels of task concepts and the accompanying interface concepts. The learning plan should be governed by the users' progress through the task concepts, enabling users to take on new interface concepts when they are needed to support more complex tasks. For users with strong knowledge of the task and interface concepts, rapid progress is possible.

For example, novice users of a cellphone can quickly learn to make/receive calls first, then to use the menus to change ring tones, and later to reset the privacy protections. The multi-layer approach helps users with different skill levels and promotes universal usability (Shneiderman, 2003).

Another option for accommodating different user classes is to permit users to personalize the menu contents. A third option is to permit users to control the density of informative feedback that the user interface provides. Novices want more informative feedback to confirm their actions, whereas frequent users want less distracting feedback. Similarly, frequent users like displays to be more densely packed than do novices. Finally, the pace of interaction may be varied from slow for novices to fast for frequent users.

3.3.2 Identify the tasks

After carefully drawing the user profile, the designers identify the tasks to be carried out. Every designer would agree that the tasks must be identified first, but too often, the task analysis is done informally or incompletely. Task analysis has a long history (Hackos and Redish, 1998; Wickens et al., 2012), but successful strategies usually involve long hours of observing and interviewing users. This helps designers to understand task frequencies and sequences and make the tough decisions about what tasks to support. Some implementers prefer to include all possible actions in the hope that some users will find them helpful, but this causes clutter. However, mobile app designers are successful because they ruthlessly limited functionality (for example, calendar, contacts, and to-do list) to guarantee simplicity.

High-level task actions can be decomposed into multiple middle-level task actions, which can be further refined into atomic actions that users execute with a single menu selection or other action. Choosing the most appropriate set of atomic actions is a difficult task. If the atomic actions are too small, the users will become frustrated by the large number of actions necessary to accomplish a higher-level task. If the atomic actions are too large and elaborate, the users will need special options to get what they want from the user interface.

The relative task frequencies are important in shaping a menu tree. Frequent tasks should be near the top and therefore quick to carry out, while rare tasks are deeper down. Relative frequency of use is one of the bases for making architectural design decisions. For example, in a text editor:

- Frequent actions might be performed by pressing special keys, such as the four arrow keys, Insert, and Delete.
- Less frequent actions might be performed by pressing a single letter plus the Ctrl key or by a selection from a pull-down menu—examples include underscore, bold, and save.
- Infrequent actions or complex actions might require going through a sequence of menu selections or form fill-ins—for example, to change the margins or to revise default printers.

Similarly, cellphone users can assign their close friends and family members to speed-dial numbers so that frequent calls can be made easily by pressing a single key

Creating a matrix of users and tasks can help designers sort out these issues (Fig. 3.3). In each box, the designer can put a check mark to indicate that this user carries out this task. A more precise analysis would include frequencies instead of just simple check marks. Such user-needs assessment clarifies what tasks are essential for the design and which ones could be left out to preserve system simplicity and ease of learning.

3.3.3 Choose an interaction style

When the task analysis is complete and the task objects and actions have been identified, the designer can choose from these five primary interaction styles: direct manipulation, menu selection, form fill-in, command language, and natural language (Box 3.1 and Box 3.2). Chapters 7 through 9 explore these styles in detail; this summary gives a brief comparative overview.

Direct manipulation When designers can create a visual representation of the world of action, the users' tasks can be greatly simplified because direct manipulation of familiar objects is possible. Examples of such visual and tangible user interfaces include the desktop metaphor, drawing tools, photo editing, and games. By pointing at visual representations of objects and actions, users can

	TASK				
Job Title	**Query by Patient**	**Update Data**	**Query across Patients**	**Add Relations**	**Evaluate System**
Nurse	**	**			
Physician	**	*			
Supervisor	*	*	**		
Appointment personnel	****				
Medical-record maintainer	**	**	*	*	
Clinical researcher			***		*
Database programmer		*	**	**	*

FIGURE 3.3

Frequency of Task by Job Title

Hypothetical frequency-of-use of data for a medical clinic information system. Answering queries from appointment personnel about individual patients is the highest-frequency task (****), and lower-frequency use is shown with ***, **, or *.

carry out tasks rapidly and can observe the results immediately (for example, dragging and dropping an icon into a trash can). Context-aware, embedded, natural, and wearable user interfaces often extend the capacity of direct manipulation designs by allowing users to gesture, point, move, or even dance to achieve their goals. Direct manipulation is appealing to novices, is easy to remember for intermittent users, and, with careful design, can be rapid for frequent users. Chapter 7 describes direct manipulation and its application.

Navigation and menu selection In navigation and menu-selection systems, users review choices, select the one most appropriate to their task, and observe the effect. If the terminology and the meaning of the items are understandable and distinct, users can accomplish their tasks with little learning or memorization and just a few actions. The greatest benefit may be that there is a clear structure to decision making, since all possible choices are presented at one time. This interaction style is appropriate for novice and intermittent users and can be appealing to frequent users if the display and selection mechanisms are rapid. For designers, menu-selection systems require careful task analysis to ensure that all functions are supported conveniently and that terminology is chosen carefully and used consistently. User-interface–building tools that support menu selection provide an enormous benefit by ensuring consistent screen design, validating completeness, and supporting maintenance. Navigation and menu selection is discussed in Chapter 8.

BOX 3.1

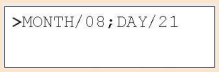
Advantages and disadvantages of the five primary interaction styles.

Advantages	Disadvantages
Direct manipulation	
• Visually presents task concepts • Allows easy learning • Allows easy retention • Allows errors to be avoided • Encourages exploration • Affords high subjective satisfaction	• May be hard to program • Accessibility requires special attention
Navigation and menu selection	
• Shortens learning • Reduces keystrokes • Structures decision making • Permits use of dialog-management tools • Allows easy support of error handling	• Presents danger of many menus • May slow frequent users • Consumes screen space • Requires rapid display rate
Form fill-in	
• Simplifies data entry • Enables convenient assistance • Permits use of form-management tools	• Consumes screen space
Command language	
• Powerful • Allows easy scripting and history keeping	• Requires learning and retention • Error-prone
Natural language	
• Relieves burden of learning syntax	• Requires clarification dialog • May not show context • May require more keystrokes • Unpredictable

BOX 3.2

Spectrum of directness.

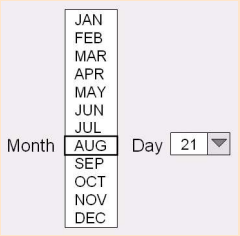
An example of progression toward more direct manipulation: less recall/more recognition, fewer keystrokes/fewer clicks, less capability to make errors, and more visible context.



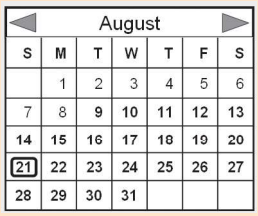
a. Command line

b. Form fill-in to reduce typing

c. Improved form fill-in to clarify and reduce errors



d. Pull-down menus offer meaningful names and eliminate invalid values



e. 2-D menus to provide context, show valid dates, and enable rapid single selection

Form fill-in When data entry is required, menu selection alone usually becomes cumbersome, and form fill-in (also called *fill in the blanks*) is appropriate. Users see a display of related fields, move a cursor among the fields, and enter data where desired. With the form fill-in interaction style, users must understand the field labels, know the permissible values and the data-entry method, and be capable of responding to error messages. Since knowledge of the keyboard, labels, and permissible fields is required, some training may be necessary. This interaction style is most appropriate for knowledgeable intermittent users or frequent users. Chapter 8 provides a thorough treatment of form fill-in.

Command language For frequent users, command languages (discussed in Section 9.4) provide a strong feeling of being in control. Users learn the syntax and can often express complex possibilities rapidly without having to read distracting prompts. However, error rates are typically high, training is necessary, and retention may be poor. Error messages and online assistance are hard to provide because of the diversity of possibilities. Command languages and query or programming languages are the domain of expert frequent users, who often derive great satisfaction from mastering a complex language. Powerful advantages include easy scripting and history keeping.

Natural language Increasingly, user interfaces respond properly to arbitrary spoken (for example, Siri on the Apple iPhone) or typed natural-language statements (for example, web search phrases). Speech recognition can be helpful with familiar phrases such as "Tell Catherine that I'll be there in ten minutes," but with novel situations users may be frustrated with the results (discussed in Chapter 9).

Blending several interaction styles may be appropriate when the required tasks and users are diverse. For example, a form fill-in interface for shopping checkout can include menus for items such as accepted credit cards, and a direct-manipulation environment can allow a right-click that produces a pop-up menu with color choices. Also, keyboard commands can provide shortcuts for experts who seek more rapid performance than mouse selection.

Increasingly, these five interaction styles are complemented by using context, sensors, gestures, spoken commands, and going beyond the screen to include enriched environments that enable users to activate doors, change sound volume, or turn on faucets. These enriched environments, such as those found in automobiles, game arcades, projected displays, wearable interfaces, musical instruments, and sound spaces, go beyond the desktop and mobile devices to produce playful and useful effects. The expansion of user interfaces into clothing, furniture, buildings, implanted medical devices, mobile platforms such as drones, and the Internet of Things enriches traditional strategies and expands the design possibilities.

Chapters 7–9 expand on the constructive guidance for using the different interaction styles outlined here, and Chapter 10 describes how input and output devices influence these interaction styles. Chapter 11 deals with interaction when using collaborative interfaces and participating in social media.

3.3.4 The Eight Golden Rules of Interface Design

This section focuses attention on eight principles, called "golden rules," that are applicable in most interactive systems and enriched environments. These principles, derived from experience and refined over three decades, require validation and tuning for specific design domains. No list such as this can be complete, but it has been well received as a useful guide to students and designers. The Eight Golden Rules are:

1. *Strive for consistency.* Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent color, layout, capitalization, fonts, and so on, should be employed throughout. Exceptions, such as required confirmation of the delete command or no echoing of passwords, should be comprehensible and limited in number.
2. *Seek universal usability.* Recognize the needs of diverse users and design for *plasticity*, facilitating transformation of content. Novice to expert differences,

age ranges, disabilities, international variations, and technological diversity each enrich the spectrum of requirements that guides design. Adding features for novices, such as explanations, and features for experts, such as shortcuts and faster pacing, enriches the interface design and improves perceived quality.

3. *Offer informative feedback.* For every user action, there should be an interface feedback. For frequent and minor actions, the response can be modest, whereas for infrequent and major actions, the response should be more substantial. Visual presentation of the objects of interest provides a convenient environment for showing changes explicitly (see the discussion of direct manipulation in Chapter 7).
4. *Design dialogs to yield closure.* Sequences of actions should be organized into groups with a beginning, middle, and end. Informative feedback at the completion of a group of actions gives users the satisfaction of accomplishment, a sense of relief, a signal to drop contingency plans from their minds, and an indicator to prepare for the next group of actions. For example, e-commerce websites move users from selecting products to the checkout, ending with a clear confirmation page that completes the transaction.
5. *Prevent errors.* As much as possible, design the interface so that users cannot make serious errors; for example, gray out menu items that are not appropriate and do not allow alphabetic characters in numeric entry fields (Section 3.3.5). If users make an error, the interface should offer simple, constructive, and specific instructions for recovery. For example, users should not have to retype an entire name-address form if they enter an invalid zip code but rather should be guided to repair only the faulty part. Erroneous actions should leave the interface state unchanged, or the interface should give instructions about restoring the state.
6. *Permit easy reversal of actions.* As much as possible, actions should be reversible. This feature relieves anxiety, since users know that errors can be undone, and encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data-entry task, or a complete group of actions, such as entry of a name-address block.
7. *Keep users in control.* Experienced users strongly desire the sense that they are in charge of the interface and that the interface responds to their actions. They don't want surprises or changes in familiar behavior, and they are annoyed by tedious data-entry sequences, difficulty in obtaining necessary information, and inability to produce their desired result.
8. *Reduce short-term memory load.* Humans' limited capacity for information processing in short-term memory (the rule of thumb is that people can remember "seven plus or minus two chunks" of information) requires that designers avoid interfaces in which users must remember information from

one display and then use that information on another display. It means that cellphones should not require reentry of phone numbers, website locations should remain visible, and lengthy forms should be compacted to fit a single display.

These underlying principles must be interpreted, refined, and extended for each environment. They have their limitations, but they provide a good starting point for mobile, desktop, and web designers. The principles presented in the ensuing sections focus on increasing users' productivity by providing simplified data-entry procedures, comprehensible displays, and rapid informative feedback to increase feelings of competence, mastery, and control over the system.

3.3.5 Prevent errors

> "There is no medicine against death, and against error no rule has been found."
>
> Sigmund Freud
> (inscription he wrote on his portrait)

The importance of error prevention (the fifth golden rule) is so strong that it deserves its own section. Users of cellphones, e-mail, digital cameras, e-commerce websites, and other interactive systems make mistakes far more frequently than might be expected.

One way to reduce the loss in productivity due to errors is to improve the error messages provided by the interface. Better error messages can raise success rates in repairing the errors, lower future error rates, and increase subjective satisfaction. Superior error messages are more specific, positive in tone, and constructive (telling users what to do rather than merely reporting the problem). Rather than using vague (? or WHAT?) or hostile (ILLEGAL OPERATION or SYNTAX ERROR) messages, designers are encouraged to use informative messages, such as PRINTER IS OFF, PLEASE TURN IT ON or MONTHS RANGE FROM 1 TO 12.

Improved error messages, however, are only helpful medicine. A more effective approach is to prevent the errors from occurring. This goal is more attainable than it may seem in many interfaces.

The first step is to understand the nature of errors. One perspective is that people make mistakes or *slips* (Norman, 1983) that designers can help them to avoid by organizing screens and menus functionally, designing commands and menu choices to be distinctive, and making it difficult for users to take irreversible actions. Norman also offers other guidelines, such as providing feedback about the state of the interface (e.g., changing the cursor to show whether a map interface is in zoom-in or select mode) and designing for consistency of actions (e.g., ensuring that yes/no buttons are always displayed in the same order).

Norman's analysis provides practical examples and a useful theory. Additional design techniques to reduce errors include the following:

Correct actions. Industrial designers recognize that successful products must be safe and must prevent users from dangerously incorrect usage of the products. Airplane engines cannot be put into reverse until the landing gear has touched down, and cars cannot be put into reverse while traveling forward at faster than five miles per hour. Similar principles can be applied to interactive systems—for example, inappropriate menu items can be grayed out so they can't be inadvertently selected, and web users can be allowed to simply click on the date on a calendar instead of having to type in a month and day for a desired airline flight departure. Likewise, instead of having to enter a 10-digit phone number, cellphone users can scroll through a list of frequently or recently dialed numbers and select one with a single button press. A variant idea is to provide users with auto-completion for typing words, selecting from menus, or entering web addresses.

Complete sequences. Sometimes an action requires several steps to reach completion. Since users may forget to complete every step of an action, designers may attempt to offer a sequence of steps as a single action. In automobiles, drivers do not have to set two switches to signal a left turn; a single switch causes both (front and rear) turn-signal lights on the left side of the car to flash. Likewise, when a pilot throws a switch to lower the landing gear, hundreds of mechanical steps and checks are invoked automatically.

As another example, users of a text editor can indicate that all section titles are to be centered, set in uppercase letters, and underlined without having to make a series of selections each time they enter a section title. Then if users want to change the title style—for example, to eliminate underlining—a single change will guarantee that all section titles are revised consistently. As a final example, an air-traffic controller may formulate plans to change the altitude of a plane from 14,000 feet to 18,000 feet in two increments; after raising the plane to 16,000 feet, however, the controller may get distracted and fail to complete the action. The controller should be able to record the plan and then have the computer prompt for completion. The notion of complete sequences of actions may be difficult to implement because users may need to issue atomic actions as well as complete sequences. In this case, users should be allowed to define sequences of their own. Designers can gather information about potential complete sequences by studying sequences of actions that people actually take and the patterns of errors that people actually make.

Thinking about universal usability also contributes to reducing errors—for example, a design with too many small buttons may cause unacceptably high error rates among older users or others with limited motor control, but enlarging the buttons will benefit all users. Section 4.6 addresses the idea of logging user errors so designers can continuously improve designs.

3.3.6 Ensuring human control while increasing automation

The guidelines and principles described in the previous sections are often devoted to simplifying the users' tasks. Users can then avoid routine, tedious, and error-prone actions and can concentrate on making critical decisions, selecting alternatives if the original approach fails, and acting in unanticipated situations. Users can also make subjective value-based judgments, request help from other humans, and develop new solutions (Sanders and McCormick, 1993). (Box 3.3 provides a detailed comparison of human and machine capabilities.)

Computer system designers have generally been increasing the degree of automation over time as procedures become more standardized and the pressure for productivity grows. With routine tasks, automation is desirable, since it reduces the potential for errors and the users' workload (Cummings, 2014). However, even with increased automation, informed designers can still offer the predictable and controllable interfaces that users usually prefer. The human supervisory role needs to be maintained because the real world is an *open system* (that is, a nondenumerable number of unpredictable events and system failures are possible). By contrast, computers constitute a *closed system* (only a denumerable number of normal and failure situations can be accommodated in hardware and software).

BOX 3.3

Relative capabilities of humans and machines.

Humans Generally Better	Machines Generally Better
• Sense-making from hearing, sight, touch, etc. • Detect familiar signals in noisy background • Draw on experience and adapt to situations • Select alternatives if original approach fails • Act in unanticipated situations • Apply principles to solve varied problems • Make subjective value-based judgments • Develop new solutions • Use information from external environment • Request help from other humans	• Sense stimuli outside human's range • Rapid consistent response for expected events • Retrieve detailed information accurately • Process data with anticipated patterns • Perform repetitive actions reliably • Perform several activities simultaneously • Maintain performance over time

For example, in air-traffic control, common actions include changes to a plane's altitude, heading, or speed. These actions are well understood and potentially can be automated by scheduling and route-allocation algorithms, but the human controllers must be present to deal with the highly variable and unpredictable emergency situations. An automated system might deal successfully with high volumes of traffic, but what would happen if the airport manager closed a runway because of turbulent weather? The controllers would have to reroute planes quickly. Now suppose that one pilot requests clearance for an emergency landing because of a failed engine, while another pilot reports a passenger with chest pains who needs prompt medical attention. Value-based judgment, possibly with participation from other controllers, is necessary to decide which plane should land first and how much costly and risky diversion of normal traffic is appropriate. Air-traffic controllers cannot just jump into an emergency; they must be intensely involved in the situation as it develops if they are to make informed and rapid decisions. In short, many real-world situations are so complex that it is impossible to anticipate and program for every contingency; human judgment and values are necessary in the decision-making process.

Another example of the complexity of life-critical situations in air-traffic control was illustrated by an incident on a plane that had a fire on board. The controller cleared other traffic from the flight path and began to guide the plane in for a landing, but the smoke was so thick that the pilot had trouble reading his instruments. Then the onboard transponder burned out, so the air-traffic controller could no longer read the plane's altitude from the situation display. In spite of these multiple failures, the controller and the pilot managed to bring down the plane quickly enough to save the lives of many—but not all—of the passengers. A computer could not have been programmed to deal with this particular unexpected series of events.

A tragic outcome of excess of automation occurred during a flight to Cali, Colombia. The pilots relied on the automatic pilot and failed to realize that the plane was making a wide turn to return to a location that it had already passed. When the ground-collision alarm sounded, the pilots were too disoriented to pull up in time; they crashed 200 feet below a mountain peak, killing all but four people on board.

The goal of design in many applications is to give users sufficient information about current status and activities to ensure that, when intervention is necessary, they have the knowledge and the capacity to perform correctly, even under partial failures (Endsley and Jones, 2004). The U.S. Federal Aviation Agency stresses that designs should place the users in control and automate only to "improve system performance, without reducing human involvement" (U.S. FAA, 2012). These standards also encourage managers to "train users when to question automation."

The entire user interface must be designed and tested not only for normal situations but also for as wide a range of anomalous situations as can be anticipated. An extensive set of test conditions might be included as part of the requirements document. Users need to have enough information that they can take responsibility for their actions. Beyond decision making and handling of failures, the users' role is to improve the interface design.

Advocates of increased autonomy, such as in driverless cars or unmanned aircraft, believe that rapid autonomous responses improve performance and produce fewer errors. However, autonomy has risks for unanticipated situations, such as changing weather or unusual trading activity. In 2015, Toyota shifted its driverless car research from autonomous designs to ones that leave drivers in control. The dangers of unanticipated situations for Unmanned Aerial Vehicles (UAVs) resulted in shifting to Remotely Piloted Vehicles (RPVs) with human control to improve reliability. While autonomy has its benefits, designs that allow human supervisory control, activity logging, and the capacity to review logs after failures appear to improve performance.

In costly business situations, such as high-speed stock market trading, clarifying responsibility for failures could lead to improved designs. Ensuring accountability and liability in advance can encourage designers to think more carefully about potential failures. Advocates of "algorithmic accountability" want developers who implement systems such as Google's search rankings or employee hiring systems to enable open access so as to limit bias and expose errors.

Questions about integrating automation with human control also emerge in consumer product user interfaces. Many designers are eager to create an autonomous agent that knows people's likes and dislikes, makes proper inferences, responds to novel situations, and performs competently with little guidance. They believe that human-human interaction is a good model for human-computer interaction, and they seek to create computer-based partners, assistants, or agents.

By contrast, many designers believe that tool-like interfaces are often more attractive than autonomous, adaptive, or anthropomorphic agents that carry out the users' intentions and anticipate needs. The agent scenarios may show a bow-tied butler-like human, like the helpful young man in Apple Computer's famous 1987 video on the *Knowledge Navigator*. Microsoft's ill-fated 1995 BOB program used cartoon characters, while its much-criticized Clippit, nicknamed Clippy, character was also withdrawn. Human-like bank machines or postal-service stations have largely disappeared, but avatars representing users, not computers, in game-playing and 3-D social environments have remained popular; users appear to enjoy the theatrical experience of creating a new identity, sometimes with colorful hair and clothes (Section 7.6).

The success of Apple's Siri's speech recognition and personality-rich voice response system shows that with careful development, useful tools can be

developed, but there is little evidence of the benefit of a talking face (Moreno and Mayer, 2007). Robot designers have perennially used human and animal forms as an inspiration, encouraging some researchers to pursue human-like robots for care of older adults or as team members in work situations. These designs attract journalists and have entertainment value but have yet to gain widespread acceptance.

A variant of the agent scenario, which does not include an anthropomorphic realization, is that the computer program has a built-in *user model* to guide an adaptive interface. The program keeps track of user performance and adapts the interface to suit the users' needs. For example, when users begin to make menu selections rapidly, indicating proficiency, advanced menu items may appear. Automatic adaptations have been proposed for interface features such as the content of menus, order of menu items, and type of feedback (graphic or tabular). Advocates point to video games that increase the speed or number of dangers as users progress through game levels. However, games are notably different from most work situations, where users bring their goals and motivations to accomplish tasks.

There are opportunities for adaptive user models to tailor designs (such as for e-mail spam filters or search results ranking), but unexpected interface behavior can have negative effects that discourage use. If adaptive systems make surprising changes, such as altering the search results ranking, users may be puzzled about what has happened. Users may become anxious because they cannot predict the next change, interpret what has happened, or return to the previous state. Users may also be annoyed if a one-time purchase of a children's book as a gift leads to repeated promotions of more children's books.

An application of user modeling is *recommender systems* in web applications. In this case, there is no agent or adaptation in the interface, but the program aggregates information from multiple sources in some (often proprietary) way. Such approaches have great practical value such as suggesting movies, books, or music; users are often intrigued to see what suggestions emerge from their purchasing patterns. Amazon.com and other e-commerce companies successfully suggest that "customers who bought X also bought Y."

The philosophical alternative to agents and user modeling is to design comprehensible systems that provide consistent interfaces, user control, and predictable behavior. Designers who emphasize a direct-manipulation style believe that users have a strong desire to be in control and to gain mastery over the system, which allows them to accept responsibility for their actions and derive feelings of accomplishment (Shneiderman, 2007). Historical evidence suggests that users seek comprehensible and predictable systems and shy away from those that are complex or unpredictable; for example, pilots may disengage automatic piloting devices if they perceive that these systems are not performing as they expect.

Agent advocates promote autonomy, but this means they must take on the issue of responsibility for failures. Who is responsible when an agent violates

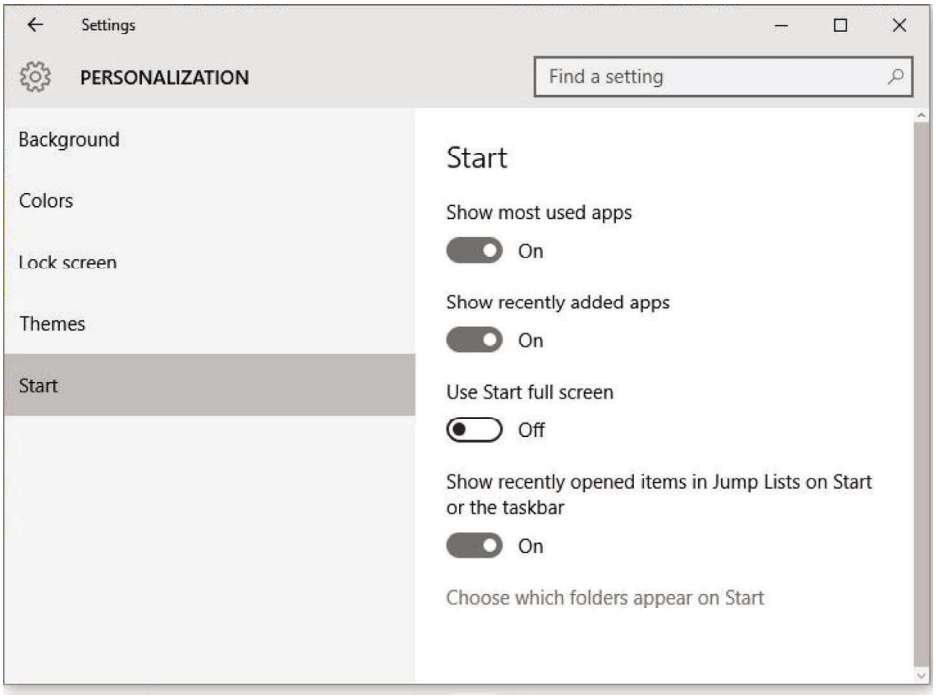


FIGURE 3.4
Windows 10 system preferences include control panels for personalization. Here we see the Start options, which allow users to control what items will display in the Start menu and taskbar.

copyright, invades privacy, or destroys data? Agent designs might be better received if they supported performance monitoring while allowing users to examine and revise the current user model.

An alternative to agents with user models may be to expand the control-panel model. Computer control panels (sometimes called settings, options, or preferences), like automobile cruise-control mechanisms and television remote controls, are designed to convey the sense of control that users seem to expect. Users employ control panels to set physical parameters, such as the cursor blinking speed or speaker volume, and to establish personal preferences such as time/date formats, color schemes, or the content of start menus (Fig. 3.4). Some software packages allow users to set parameters such as the speed of play in games. Users start at layer 1 and can then choose when to progress to higher levels; often they are content to remain experts at layer 1 of a complex interface rather than dealing with the uncertainties of higher layers. More elaborate control panels exist in style sheets of word processors, specification boxes of query facilities, and sliders of information-visualization tools.

3.4 Theories

One goal for the discipline of human-computer interaction is to go beyond the specifics of guidelines and build on the breadth of principles to develop tested, reliable, and broadly useful *theories*. Of course, for a topic as large as user-interface design, many theories are needed (Carroll, 2003; Rogers, 2012; Bødker, 2015).

Some theories are *descriptive*; these are helpful in developing consistent terminology and useful taxonomies, for objects and actions, thereby supporting collaboration and training. Other theories are *explanatory*, describing sequences of events and, where possible, cause and effect, making interventions possible. Still other theories are *prescriptive*, giving designers clear guidance for their choices. Finally, the most precise theories are *predictive*, enabling designers to compare proposed designs for execution time, error rates, conversion rates, or trust levels.

Another way to group theories is according to the types of skills involved, such as *motor* (pointing with a mouse), *perceptual* (finding an item on a display), or *cognitive* (planning the sequence of steps needed to pay a bill) (Norman, 2015). Motor skill performance predictions are well established and accurate for predicting keystroking or pointing times (see Fitts's Law, Section 10.3). Perceptual theories have been successful in predicting reading times for free text, lists, formatted displays, and other visual or auditory tasks. Cognitive theories, involving short-term, working, and long-term memory, are central to problem solving and play a key role in understanding productivity as a function of system response time (Chapter 12). However, predicting performance on complex cognitive tasks (combinations of subtasks) is especially difficult because of the many strategies that might be employed and the many opportunities for going astray. The ratio of times needed to perform complex tasks between novices and experts or between first-time and frequent users can be as high as 100 to 1. Actually, the contrast is even more dramatic because novices and first-time users often make errors and are unable to complete the tasks.

Web designers have emphasized information-architecture theories with navigation as the keys to user success. Web users can be considered as *foraging* for information, and therefore the effectiveness of the *information scent* of links is the issue (Pirolli, 2007). A high-quality link, relative to a specific task, gives users a good scent (or indication) of what is at the destination. For example, if users are trying to find a demonstration of a software package, a link with the text "download demo" has a good scent. The challenge to designers is to understand user tasks well enough to design a large website such that users will be able to find their way successfully from a home page to the right destination, even if it is three or four clicks away. Information-foraging theory attempts to

BOX 3.4

Multiple theory types that researchers and designers consider when evaluating user interfaces.

By Theory Type

- **Descriptive** Describes user interfaces and their uses with consistent terminology and taxonomies
- **Explanatory** Describes sequences of events with causal relationships
- **Prescriptive** Offers guidelines for designers to make decisions
- **Predictive** Enables comparison of design alternatives based on numeric predictions of speed or errors.

By Human Capacity

- **Motor** Skill in pointing, clicking, dragging, or other movements
- **Perceptual** Visual, auditory, tactile, and other human sensory inputs
- **Cognitive** Problem solving with short- and long-term memory

predict user success rates given a set of tasks and a website so as to guide refinements.

Taxonomies can be an important part of descriptive and explanatory theories. A taxonomy imposes order by classifying a complex set of phenomena into understandable categories. For example, a taxonomy might be created for different kinds of input devices: direct versus indirect, linear versus rotary, 1-, 2-, 3- or higher-dimensional, and so on (Card et al., 1990). Other taxonomies might cover tasks (structured versus unstructured, novel versus regular) or user-interface styles (direct manipulation, menus, form fill-in). An important class of taxonomies has to do with the individual differences among users, such as personality styles (convergent versus divergent, field-dependent versus independent), technical aptitudes (spatial visualization, reasoning), and user experience levels (novice, knowledgeable, expert). Taxonomies facilitate useful comparisons, organize topics for newcomers, guide designers, and often indicate opportunities for novel products—for example, a task by type taxonomy organizes the information visualizations in Chapter 16.

Any theory that might help designers to predict performance for even a limited range of users, tasks, or designs is a contribution. At the moment, the field is filled with hundreds of theories competing for attention while being refined by their promoters, extended by critics, and applied by eager and hopeful—but skeptical—designers (Carroll, 2003, 2014; Rogers, 2012). This development is healthy for the growing discipline of human-computer interaction, but it means

that practitioners must keep up with the rapid developments not only in software tools and design guidelines but also in theories. Critics raise two challenges:

1. *Theories should be more central to research and practice*. A good theory should guide researchers in understanding relationships between concepts and generalizing results. It should also guide practitioners when making design tradeoffs for products. The power of theories to shape design is most apparent in focused theories such as Fitts's Law; it is more difficult to demonstrate for explanatory theories, whose main impact may be in educating the next generation of designers.
2. *Theories should lead rather than lag behind practice*. Critics remark that too often a theory is used to explain what has been produced by commercial product designers. A robust theory should predict or at least guide practitioners in designing new products. Effective theories should suggest novel products and services while helping to refine existing ones.

Another direction for theoreticians is to predict the subjective satisfaction or emotional reactions of users. Researchers in media and advertising have recognized the difficulty of predicting emotional reactions, so they complement theoretical predictions with their intuitive judgments and extensive market testing (Nahl and Bilal, 2007).

Broader theories of small-group behavior, organizational dynamics, and sociology are proving to be useful in understanding social media and collaborative interfaces (Chapter 11). Similarly, the methods of anthropology or social psychology may be helpful in understanding technology adoption and overcoming barriers to new technology that cause resistance to change.

There may be "nothing so practical as a good theory," but coming up with an effective theory is often difficult. By definition, a theory, taxonomy, or model is an abstraction of reality and therefore must be incomplete. However, a good theory should be understandable, produce similar conclusions for all who use it, and help to solve design problems. This section reviews a range of descriptive and explanatory theories.

3.4.1 Design-by-levels theories

One approach to developing descriptive theories is to separate concepts according to levels. Such theories have been helpful in software engineering and network design. An appealing and easily comprehensible design-by-levels theory for interfaces is the four-level conceptual, semantic, syntactic, and lexical theory (Foley et al., 1995):

1. The *conceptual level* is the user's "mental model" of the interactive system. Two mental models for image creation are paint programs that manipulate pixels and drawing programs that operate on objects. Users of paint programs think in terms of sequences of actions on pixels and groups of

pixels, while users of drawing programs think in terms of sequences of actions on objects and groups of objects. Decisions about mental models affect each of the lower levels.

2. The *semantic level* describes the meanings conveyed by the user's input and by the computer's output display. For example, deleting an object in a drawing program could be accomplished by undoing a recent action or by invoking a delete-object action. Either action should eliminate only a single object and leave the rest untouched.
3. The *syntactic level* defines how the user actions that convey semantics are assembled into complete sentences to perform certain tasks. For example, the delete-files action could be invoked by dragging an object to a trash can followed by a click in a confirmation dialog box.
4. The *lexical level* deals with device dependencies and with the precise mechanisms by which users specify the syntax (for example, a function key or a mouse double-click within 200 milliseconds).

This four-level theory is convenient for designers because its top-down nature is easy to explain, matches the software architecture, and allows for useful modularity during design. Over the years, the success of graphical direct-manipulation interfaces has shifted attention up to the conceptual level, which is closest to the task domain (Parush, 2015). For example, designers of personal financial interfaces often use direct-manipulation interfaces. These interfaces build on the users' mental model of writing checks by showing the image of a check for users to fill in. The same image of a check serves as the query template so users can specify dates, payees, or amounts.

Increasingly, actions are shown by novel visual representations (for example, a trash can for deletion or a play button to start playing a video). Users have to learn the semantics (e.g., that they can recover a file by opening up the trash can or stop a video by clicking on the pause button), but if the designers choose familiar objects to associate with the actions, users can quickly acquire the correct mental model for operating the user interface. Of course, users also have to learn the syntax of dragging objects or clicking to initiate the actions, but these mechanisms are commonly used and have become well known.

The idea of design-by-levels is successful even in more complex systems with many objects and actions. For example, the human body can be discussed in terms of neural, muscular, skeletal, reproductive, digestive, circulatory, and other subsystems, which in turn might be described in terms of organs, tissues, and cells. Most real-world objects have similar decompositions: buildings into floors, floors into rooms, rooms into doors/walls/windows, and so on. Similarly, movies can be decomposed into scenes, scenes into shots, and shots into dialogue, images, and sounds. Since most objects can be decomposed in many ways, the designer's job is to create comprehensible and memorable levels of objects.

In parallel with the decomposition of objects, designers need to decompose complex actions into several smaller actions. For example, a baseball game has

innings, pitches, runs, and outs, and a building-construction plan can be reduced to a series of steps such as surveying the property, laying the foundation, building the frame, raising the roof, and completing the interior. Most actions can also be decomposed in many ways, so again the designer's job here is to create comprehensible and memorable levels of actions. The goal of simplifying interface concepts while presenting visual representations of the objects and actions involved is at the heart of the direct-manipulation approach to design (Chapter 7).

When a complete user-interface design has been made, the user tasks can be described by a series of actions. These precise descriptions can serve as a basis for predicting the time required to perform tasks by simply counting up the number of milliseconds needed to complete all the steps. For example, resizing a photo may require several mouse drags, selections of menu items, clicks on dialog box buttons, and typing of dimensions, but each of these actions takes a predictable amount of time. Several researchers have successfully predicted the time required for complex tasks by adding up the times required for each component action. This predictive approach, based on *goals*, *operators*, *methods*, and *selection rules* (GOMS), decomposes goals into many operators (actions) and then into methods. Users apply selection rules to choose among alternate methods for achieving goals (Card et al., 1983; Baumeister et al., 2000).

The GOMS approach works best when the users are expert and frequent users who are working on their own, fully focused on the task, and make no mistakes. Advocates of GOMS have developed software tools to simplify and speed up the modeling process in the hope of increasing usage (John, 2011). Critics complain that broader theories are needed to predict novice user behavior, the transition to proficiency, the rate of errors, and retention over time.

Designers have discovered that using design-by-levels theories forces clear definitions of the high-level objects and actions, which are gathered from listening to the language used in the task domain. Music can be thought of as songs, organized by artists, albums, and genres. Users can find a song and then play it or add it to a playlist. The clarity of this conceptual structure earned it a patent and has stimulated multiple commercial successes.

3.4.2 Stages-of-action theories

Another approach to forming explanatory theories is to portray the stages of action that users go through in using interactive products such as information appliances, web interfaces, or mobile devices (e.g., music players). Norman (2013) offers seven stages of action, arranged in a cyclic pattern, as an explanatory theory of human-computer interaction:

1. Forming the goal
2. Forming the intention

3. Specifying the action
4. Executing the action
5. Perceiving the system state
6. Interpreting the system state
7. Evaluating the outcome

Some of Norman's stages correspond roughly to Foley et al.'s (1995) separation of concerns; that is, users form a conceptual intention, reformulate it into the semantics of several commands, construct the required syntax, and eventually produce the lexical token by the action of moving the mouse to select a point on the screen. Norman makes a contribution by placing his stages in the context of *cycles of action* and *evaluation*, which take place over seconds and minutes. This dynamic process of action distinguishes Norman's approach from the other theories, which deal mainly with knowledge that must be in the user's mind. Furthermore, the seven stages of action lead naturally to identification of the *gulf of execution* (the mismatch between the user's intentions and the allowable actions) and the *gulf of evaluation* (the mismatch between the system's representation and the user's expectations).

This theory leads Norman to suggest four principles of good design:

1. The state and the action alternatives should be visible.
2. There should be a good conceptual model with a consistent system image.
3. The interface should include good mappings that reveal the relationships between stages.
4. Users should receive continuous feedback.

Norman places a heavy emphasis on studying errors, describing how errors often occur in moving from goals to intentions to actions and to executions.

The stages-of-action theory helps designers to describe user exploration of an interface (Polson and Lewis, 1990). As users try to accomplish their goals, there are four critical points where user failures can occur: (1) users may form inadequate goals, (2) users might not find the correct interface object because of an incomprehensible label or icon, (3) users may not know how to specify or execute a desired action, and (4) users may receive inappropriate or misleading feedback.

Refinements of the stages-of-action theory have been developed for other domains. For example, information seeking has been characterized by these stages: (1) recognize, (2) accept the information problem, (3) formulate and (4) express the query, then (5) examine the results, (6) reformulate the problem, and (7) use the results (Marchionini and White, 2007). Of course, there are variations with users skipping stages or going back to earlier stages, but the model helps guide designers and users.

Commercial website designers know the benefit of a clear stages-of-action theory in guiding anxious users through a complex process. For example, the Amazon.com website converts the potentially confusing checkout process into a

comprehensible four-stage process: (1) Sign-in, (2) Shipping & Payment, (3) Gift-Wrap, and (4) Place Order. Users can simply move through these four stages or back up to previous stages to make changes. Amazon.com also recognizes the need for a frequent user shortcut, the 1-click purchase, for products such as a Kindle book.

Designers can apply the stages-of-action theory by thinking deeply about the beginning, middle, and end stages to ensure that they cover a wide enough scope of usage. Many new products emerge as a result of adding novel features to what was considered a well-defined process; for example, expanding the music-playing process to include the earlier stages of music purchase or composition and the later stages of music sharing or reviewing/rating.

3.4.3 Consistency theories

An important goal for designers is a *consistent* user interface. The argument for consistency is that if terminology for objects and actions is orderly and describable by a few rules, users will be able to learn and retain them easily. This example illustrates consistency and two kinds of inconsistency (**A** illustrates lack of consistency, and **B** shows consistency except for a single violation):

Consistent	**Inconsistent A**	**Inconsistent B**
delete/insert table	delete/insert table	delete/insert table
delete/insert column	remove/add column	remove/insert column
delete/insert row	destroy/create row	delete/insert row
delete/insert border	erase/draw border	delete/insert border

Each of the actions in the consistent version is the same, whereas the actions vary for inconsistent version A. The inconsistent action verbs are all acceptable, but their variety suggests that they will take longer to learn, will cause more errors, will slow down users, and will be harder for users to remember. Inconsistent version B is somehow more startling, because there is a single unpredictable inconsistency; it stands out so dramatically that this language is likely to be remembered for its peculiar inconsistency.

Consistency for objects and actions (nouns and verbs) is a good starting point, but there are many other forms of consistency that require careful thought by designers. Consistent use of color, layout, icons, fonts, font sizes, button sizes, and much more is vital in giving users a clear understanding of the interface. Inconsistency in elements such as the positioning of buttons or colors will slow users down by 5–10%, while changes to terminology slow users by 20–25%.

Consistency is an important goal, but there may be conflicting forms of consistency, and sometimes inconsistency is a virtue (for example, to draw attention to a dangerous action). Competing forms of consistency require designers to make difficult choices or invent new strategies. For example, while automobile

interface designers have agreed to always place the accelerator pedal to the right of the brake pedal, there's no agreement about whether turn signal controls should be to the right or left of the steering wheel.

Consistency issues are critical in the design of mobile devices. In successful products, users get accustomed to consistent patterns, such as initiating actions with a left-side button while terminating actions with a right-side button. Similarly, up and down scrolling actions should be done consistently using buttons that are vertically aligned. A frequent problem is the inconsistent placement of the Q and Z characters on phone buttons.

Designers can enforce consistency by developing detailed guidelines documents for their designs (Section 4.3) that spell out all of the consistency requirements. Expert reviewers of user interfaces can then verify the consistency of the design. This requires a careful eye and thoughtful attention to how each screen is laid out, each action sequence is carried out, and each sound is played.

3.4.4 Contextual theories

The design-by-levels, stages-of-action, and consistency theories address the specifics of how objects and actions appear on displays and what actions users take to carry out their tasks. These theories and design aspects might be called *micro-HCI*, since they cover measurable performance in terms of speed and errors. Micro-HCI is best studied with the scientific methods of experimental and cognitive psychology using 30- to 120-minute controlled experiments and statistical tests for significant differences between groups working on well-defined tasks

Micro-HCI has been and continues to be a great success story, but there is a growing awareness that tightly controlled laboratory studies of isolated phenomena are only one part of the story. The rise of *macro-HCI*, which emphasizes the user experience, the usage context, and social engagement, has opened up new possibilities for researchers and practitioners. While micro-HCI research is more about laboratory studies to collect clear performance measures for identifiable tasks (e.g., how many seconds to find the last flight on July 4 from Washington, DC, to London), macro-HCI research is more about ethnographic observation of users doing work or play in their familiar context over days or even months. The outcomes of micro-HCI research are statistically significant differences that support or refute a hypothesis, while the outcomes of macro-HCI research are insights about what leads to increased user satisfaction, how the context of use matters, and how new applications could improve education, health, safety, or the environment.

Macro-HCI thinking leads to different kinds of theories that might best be called contextual, since they consider the emotional, physical, and social contexts of use. Happy users will persevere in the face of frustrations, cope with interruptions from neighbors, and ask for help when they need it. In short, the

BOX 3.5

Theory types that organize evaluation of user interfaces and guide design.

Micro-HCI Theories	Focus on measurable performance (such as speed and errors) on multiple standard tasks taking seconds or minutes in laboratory environments
• **Design-by-levels**	Start with high-level design and move to smaller objects and actions
• **Stages-of-action**	Consider user behavior as they form intentions and seek to realize their goals.
• **Consistency**	Strive for consistency in objects and actions, shown by words, icons, colors, shapes, gestures, menu choices
Macro-HCI Theories	Focus on case studies of user experience over weeks and months in realistic usage contexts with rich social engagement
• **Contextual**	Support users who are embedded in emotional, physical, and social environments
• **Dynamic**	Design for evolution of user behavior as users move through levels of mastery, performance, and leadership

physical and social environments are inextricably intertwined with use of information and communications technologies. Design cannot be separated from patterns of use.

Suchman's (1987) analysis in her book *Plans and Situated Action* is often credited with launching this reconsideration of human-computer interaction. She argued that the cognitive model of orderly human plans that were executed when needed was insufficient to describe the richer and livelier world of work or personal usage. She proposed that users' actions were situated in time and place, making user behavior highly responsive to other people and to environmental contingencies. If users got stuck in using an interface, they might ask for help, depending on who was around, or consult a manual, if one were available. If they were pressed for time, they might risk some shortcuts, but if the work was life-critical, they would be extra cautious. Rather than having fixed plans, users constantly changed their plans in response to the circumstances. The argument of distributed cognition is that knowledge is not only in the users' minds but distributed in their environments—knowledge is stored on paper documents, accessible from electronic files, or available from colleagues.

Contextual theories also address the shift from use of a computer to interaction with a device-rich environment filled with sensors, responsive appliances, display walls, and audio generators. Rather than picking up a device, users

activate automatic doors, hand dryers, or light switches. Sometimes users are inside a device such as an automobile, forcing designers to consider the surrounding space and the other people in the car as well as the sounds, vibrations, and forces of acceleration. Contextual theories often emphasize the social environment in which users are engaged with other people who can provide assistance or can be distractions.

Advocates of contextual theories believe that the turbulence of actual usage (as opposed to idealized task specifications) means that users have to be more than test subjects—they have to be participants in design processes. Proponents of contextual theories encourage more ethnographic observation, longitudinal case studies, and action research by participant observers (Boellstorff et al., 2012; Crabtree et al., 2012; Horst and Miller, 2013).

Breakdowns are often seen as sources of insight about design, and users are encouraged to become reflective practitioners who are continuously engaged in the process of design refinement. Understanding the transition from novice to expert and the differences in skill levels has become a focus of attention, further calling into question the utility of hour-long laboratory or half-day usability-testing studies as a guide to the behavior of users after a month or more of experience.

Contextual theories are especially relevant to mobile devices and ubiquitous computing innovations. Such devices are portable or installed in a physical space, and they are often designed specifically to provide place-specific information (for example, a city guide on a portable computer or a museum guide that gives information on a nearby painting). Location information by way of GPS systems enables new services but raises concerns about misuse of tracking information.

Designers can apply contextual theories by observing users in their own environments as they carry out their work, engage socially, or participate in sports or play. A detailed record of how tasks are chosen and carried out, including collaborations with others, internal or external interruptions, and errors that occur, would lay the basis for interface design. Contextual theories are about how people form intentions, how aspirations crystalize, how empathy is encouraged, and how trust shapes behavior; they are also about emotional states of excitement or frustration, the joy of attaining goals, and the disappointment of failure. These strong reactions are hard to capture in predictive mathematical equations, but it is important to study and understand them. To that end, many researchers are shifting their methods from controlled experiments to ethnographic observation, focus group discussions, and long-term case studies. Surveys and interviews can provide quantitative data for much-needed theories of how design variables affect users' levels of satisfaction, fear, trust, and cooperativeness.

While contextual theories emphasize the changes to observation and research, contextual theories can also guide design. If interruptions are an impediment, then users might be given the option of blocking them. If usage outdoors is a requirement, then contrast setting or font sizes should be easily adjustable. If

collaboration with others is a high priority, then easy sharing of screens or texting should be possible.

A taxonomy of mobile device applications could guide innovators:

- *Monitor* blood pressure, stock prices, or air quality and give *alerts* when normal ranges are exceeded.
- *Gather* information from meeting attendees or rescue team members and *spread* the action list or current status to all.
- *Participate* in a large group activity by voting and *relate* to specific individuals by sending private messages.
- *Locate* the nearest restaurant or waterfall and *identify* the details of the current location.
- *Capture* information or photos left by others and *share* yours with future visitors.

These five pairs of actions could be tied to a variety of objects (such as photos, annotations, or documents), suggesting new mobile devices and services. They also suggest that one way of thinking about user interfaces is by way of the objects that users encounter and the actions that they take (Robinson et al., 2015). A more ambitious use of mobile devices is to aggregate information from thousands of cellphones to determine where there is highway congestion or which rides at an amusement park have the longest waiting lines.

3.4.4 Dynamic theories

A key aspect of macro-HCI is how users evolve over weeks and months, especially as they move from novices to experts, from new customers to frequent buyers, or from readers of Wikipedia to active collaborators or administrators. These theories address design for evolutionary development of skills mastery, behavior change, reputation growth, and leadership capacities.

Dynamic theories owe much to the theories of adoption or innovation diffusion (Rogers, 2003), which include five attributes:

1. relative advantage: faster, safer, more error free usage, or cheaper
2. compatibility: fitting for users' need, consistent with existing values
3. trial-ability: availability to experiment with innovation
4. observability: visibility of innovation to others
5. less complexity: ease of learning and use

These attributes lead to macro-HCI design guidelines, such as suggesting specific user-interface features, combining features to make some more visible than others, and providing informative feedback to users about their usage history. Other macro-HCI design guidelines will suggest ways of training users about features (informing them about new features), rewarding them for successes (showing their progress in reading a book or their score in a game), and

sharing their progress with others (notifying friends about an exercise achievement or business associates about a price change).

Dynamic theories deal with long-term (weeks or months) changes in behavior for health (smoking cessation, diet, exercise, or performance in memory games) or education (completing an online course or demonstrating increased familiarity with a body of knowledge). A large category of dynamic theories cover customer loyalty plans that encourage increased commitment, such as awards from restaurants, airlines, or hotels. These carefully designed programs have multiple award levels, such as bronze, silver, gold, and platinum, with carefully chosen benefits to encourage increased activity.

Behavior change by badge awards and loyalty programs will become increasingly important because of the growing data sources about what works and what doesn't. The remarkably focused and personalized ways of persuading users and raising motivation will dramatically increase the possibilities for designers who understand when personal recognition, social rewards, community awareness, and financial compensation are most effective.

Dynamic theories are strong among designers of online communities and user-generated content sites. They know that users often move through stages as they gain confidence and a greater sense of responsibility for quality. There are many paths, but a study of Wikipedia contributors (Bryant et al., 2005) suggests at least these stages: (1) reader of articles related to personal interests, (2) fixer of mistakes and omissions in familiar topics, (3) registered user and caretaker for a collection of articles, (4) author for new articles, (5) participant in community of authors, and (6) administrator who is active in governance and future directions.

Following these results, the Reader-to-Leader Model described how to design user-interface and social engagement features to promote movement through these stages over a period of weeks or months (Preece and Shneiderman, 2009). At early stages, there are user-interface design guidelines, such as highlighting key features and valuable content, and social engagement design guidelines, such as encouragement from friends, family, and respected authorities. At later stages, there are user-interface design guidelines, such as visible recognition for contributions, and social engagement design guidelines, such as promoting empathy, supporting mentoring, raising trust, and facilitating conflict resolution.

Macro-HCI theories also promote the idea that user interfaces have profound societal effects with positive outcomes such as increased social communication, safety, or health awareness and negative outcomes such as undermining concentration, invading privacy, or exposing users to hackers. Visionaries see user interfaces as shaping personal processes of mindfulness, reflection, or empathy and community processes of civic participation, democratic sharing, or conflict resolution (Bell and Dourish, 2011; Nelson and Stolterman, 2012; Calvo and Peters, 2014). At a grander scale, macro-HCI dreamers believe that better user interfaces and user experiences can support international development, improved healthcare, environmental preservation, and peaceful dispute reconciliation.

Practitioner's Summary

Design principles and guidelines are emerging from practical experience and empirical studies. Managers can benefit by reviewing available guidelines documents and then constructing local versions. These documents record organizational policies, support consistency, and record the results of practical and experimental testing. Guidelines documents also stimulate discussion of user-interface issues and help train new designers. More established principles—such as recognizing user diversity, striving for consistency, and preventing errors—have become widely accepted, but they require fresh interpretation as technology and applications evolve. Automation is increasing for many tasks, but preserving human control is still a beneficial goal.

Micro-HCI and macro-HCI theories are being validated and refined to clarify the design implications. For expert users with established sequences of actions, predictive models that guide designers to reduce the time required for each step are valuable. For novel applications and novice users, clarifying task objects and actions (for example, songs and albums that can be played or added to playlists) and promoting consistency can lead to easily learned designs that promote user confidence. For every design, extensive testing and iterative refinement are necessary parts of the development process.

Researcher's Agenda

The central problem for human-computer–interaction researchers is developing adequate micro-HCI and macro-HCI theories. Traditional psychological theories must be extended and refined to accommodate the complex human learning, memory, and problem solving required in user interfaces and user experiences. Useful goals include descriptive taxonomies, explanatory theories, and predictive models. When predictions can be made for learning times, performance speeds, error rates, subjective satisfaction, or human retention over time, designers can more easily choose among competing designs.

Theories in human-computer interaction can be grouped into five families: those that focus on design by levels, stages of action, consistency, contextual awareness, and evolutionary dynamics. Theories can be useful even if they are narrowly focused on a specific task, such as choosing a video from a database of millions of videos. Even more powerful are theories that apply to diverse tasks such as web searching, online reviewing, or encouraging community participation. Applied research problems are suggested by each of the hundreds of design

principles or guidelines that have been proposed. Each validation of these principles and clarification of the breadth of applicability is a small but useful contribution to the emerging mosaic of human performance with interactive systems.

WORLD WIDE WEB RESOURCES

www.pearsonglobaleditions.com/shneiderman

Many websites include guidelines documents for desktop, web, and mobile device interfaces and recommendations for universal usability strategies to accommodate users with disabilities or other special needs. Theories are proliferating, and the web is a good place to keep up with the latest ones from major developers and sources promoting universal usability:

- Apple Human Interface Guidelines: http://developer.apple.com
- Microsoft Windows User Experience Interaction Guidelines: https://msdn.microsoft.com
- World Wide Web Consortium (W3C) guidelines: http://www.w3.org/TR/WCAG20/
- Interaction Design Foundation Encyclopedia covers theories: https://www.interaction-design.org/

Debates over hot topics can be found in relevant blogs and newsgroups, which are searchable from many standard services such as Google or Bing.

Discussion Questions

1. Give a brief explanation of the Eight Golden Rules of Interface Design. State an example you have seen on a device, computer interface, or web site that violates those rules.
2. Don Norman suggests organizing screens and menus functionally, designing commands and menu choices to be distinctive, and making it difficult for users to take irreversible actions. Norman also says to provide feedback about the state of the interface (e.g., changing the cursor to show whether a map interface is in zoom-in or select mode) and designing for consistency of actions (e.g., ensuring that Yes/No buttons are always displayed in the same order). State one example you have seen where you know these rules have been violated. Although this is crucial to a user interface's success, suggest why there may be challenges to implement some of Norman's guidelines.
3. Clarify the difference among guidelines, principles, and theories.

4. What are some of the techniques that can be used to get the user's attention? Why is it important to exercise caution when using these techniques?
5. What are the stages of forming explanatory theories as suggested by Don Norman?

References

Baumeister, L., John, B. E., and Byrne, M., A comparison of tools for building GOMS models, *Proc. CHI 2000 Conference: Human Factors in Computing Systems*, ACM Press, New York (2000), 502–509.

Bell, Genevieve, and Dourish, Paul, *Divining a Digital Future: Mess and Mythology in Ubiquitous Computing*, MIT Press (2011).

Boellstorff, Tom, Nardi, Bonnie, Pearce, Celia, and Taylor, T. L., *Ethnography and Virtual Worlds: A Handbook of Method*, Princeton University Press (2012).

Bødker, Susanne, Third Wave HCI, 10 years later: Participation and sharing, *ACM* Interactions *22*, 5 (Sept.–Oct. 2015), 24.

Bryant, Susan, Forte, Andrea, and Bruckman, Amy, Becoming Wikipedian: Transformation of participation in a collaborative online encyclopedia, *Proc. ACM SIGGROUP International Conference on Supporting Group Work,* ACM Press, New York (2005), 1–10.

Calvo, Rafael A., and Peters, Dorian, *Positive Computing: Technology for Wellbeing and Human Potential*, MIT Press (2014).

Card, Stuart K., Mackinlay, Jock D., and Robertson, George G., The design space of input devices, *Proc. CHI '90 Conference: Human Factors in Computing Systems,* ACM Press, New York (1990), 117–124.

Card, Stuart, Moran, Thomas P., and Newell, Allen, *The Psychology of Human-Computer Interaction,* Lawrence Erlbaum Associates, Hillsdale, NJ (1983).

Carroll, John M. (Editor), *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science,* Morgan Kaufmann, San Francisco, CA (2003).

Carroll, John M., Human computer interaction—brief intro. In Soegaard, Mads, and Dam, Rikke Friis (Editors), *The Encyclopedia of Human-Computer Interaction,* 2nd Edition, The Interaction Design Foundation (2014). Available at https://www.interaction-design.org/encyclopedia/human_computer_interaction_hci.html.

Crabtree, Andrew, Rouncefield, Mark, and Tolmie, Peter, *Doing Design Ethnography,* Springer, London (2012).

Cummings, M. L., Man versus machine or man + machine? *IEEE Intelligent Systems 29,* 5 (2014), 62–69.

Endsley, Mica R., and Jones, Debra G., *Situation Awareness: An Approach to User-Centered Design,* 2nd Edition, CRC Press (2004).

Foley, James D., van Dam, Andries, Feiner, Steven K., and Hughes, John F., *Computer Graphics: Principles and Practice in C,* 2nd Edition, Addison-Wesley, Reading, MA (1995).

Grudin, J., A moving target: The evolution of human-computer interaction. In J. Jacko (Editor), *Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications,* 2nd Edition, Taylor & Francis (2012).

Hackos, JoAnn T., and Redish, Janice C., *User and Task Analysis for Interface Design,* John Wiley & Sons, New York (1998).

Hartson, R., and Pyla, P., *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*, Morgan Kaufmann (2012).

Horst, Heather A., and Miller, Daniel (Editors), *Digital Anthropology*, Bloomsbury (2013).

John, B. E., Using predictive human performance models to inspire and support UI design recommendations, in *Proceedings of the Conference on Human Factors in Computing Systems* (CHI '11), ACM, New York, NY (2011), 983–986.

Johnson, Jeff, *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Rules,* 2nd Edition, Morgan Kaufmann, San Francisco, CA (2014).

Lynch, Patrick J., and Horton, Sarah, *Web Style Guide: Basic Design Principles for Creating Web Sites,* 3rd Edition, Yale University Press, New Haven, CT (2008).

Marchionini, G., and White, R. W., Find what you need, understand what you find, *International Journal of Human-Computer Interaction 23,* 3 (2007), 205–237.

Moreno, R., and Mayer, R. E., Interactive multimodal learning environments, *Educational Psychology Review* 19 (2007), 309–326.

Nahl, Diane, and Bilal, Dania (Editors), *Information and Emotion: The Emergent Affective Paradigm in Information Behavior Research and Theory*, Information Today, Medford, NJ (2007).

National Cancer Institute, *Research-based Web Design and Usability Guidelines*, Dept. of Health & Human Services, National Institutes of Health (2006, updated on the web at http://www.usability.gov).

Nelson, H. G., and Stolterman, E., *The Design Way: Intentional Change in an Unpredictable World,* 2nd Edition, MIT Press, Cambridge, MA (2012).

Norman, Donald A., Design rules based on analyses of human error, *Communications of the ACM* 26, 4 (1983), 254–258.

Norman, Donald A., *Design of Everyday Things,* Revised Edition, Basic Books, New York (2013).

Norman, Kent L., *Cyberpsychology*: *An Introduction to the Psychology of Human-Computer Interaction,* 2nd Edition, Cambridge University Press, New York (2015).

Parush, Avi, *Conceptual Design for Interactive Systems: Designing for Performance and User Experience,* Elsevier/Morgan Kaufmann (2015).

Pirolli, Peter, *Information Foraging: Adaptive Interaction with Information*, Oxford University Press (2007).

Polson, Peter, and Lewis, Clayton, Theory-based design for easily learned interfaces, *Human-Computer Interaction* 5 (1990), 191–220.

Preece, J., and Shneiderman, B., The Reader-to-Leader Framework: Motivating technology-mediated social participation, *AIS Transactions on Human-Computer Interaction 1*, 1 (March 2009), 13–32. Available at http://aisel.aisnet.org/thci/vol1/iss1/5/.

Robinson, Simon, Jones, Matt, and Marsden, Gary, *There's Not an App for That: Mobile User Experience Design for Life*, Morgan Kaufmann (2015).

Rogers, Everett M., *Diffusion of Innovations,* 5th Edition, Free Press, New York (2003).

Rogers, Yvonne, *HCI Theory: Classical, Modern, and Contemporary*, Synthesis Lectures in Human-Centered Informatics (Series Editor John M. Carroll), Morgan & Claypool Publishers (2012).

Sanders, M. S., and McCormick, E. J., *Human Factors in Engineering and Design,* 7th Edition, McGraw-Hill, New York (1993).

Shneiderman, Ben, Promoting universal usability with multi-layer interface design, *ACM Conference on Universal Usability*, ACM Press, New York (2003), 1–8.

Shneiderman, Ben, Human responsibility for autonomous agents, *IEEE Intelligent Systems 22,* 2 (March/April 2007), 60–61.

Smith, Sid L., and Mosier, Jane N., *Guidelines for Designing User Interface Software*, Report ESD-TR–86–278, Electronic Systems Division, MITRE Corporation, Bedford, MA (1986). Available from National Technical Information Service, Springfield, VA.

Suchman, Lucy A., *Plans and Situated Actions: The Problem of Human-Machine Communication*, Cambridge University Press, Cambridge, U.K. (1987).

U.S. Federal Aviation Administration, *The Human Factors Design Standard*, Atlantic City, NJ (updated May 2012). Available at http://hf.tc.faa.gov/hfds/.

Wickens, Christopher D., Hollands, Justin G., Banbury, Simon, and Parasuraman, Raja, *Engineering Psychology and Human Performance,* 4th Edition, Psychology Press (2012).

This page intentionally left blank