PART 2

DEVELOPMENT PROCESSES


Design
users
user
process
development
user-interface
Designers
system
guidelines
usability
software
interfaces
data
systems
Social
developers
requirements
Managers
Participatory
Many
work
information
may
early
Processes
issues
Also
useneeds
methods
contextual
implementation
strategies
must
example
tools
tasks
environment
prototypes
tests
might
methodologies
observation
projects
developing
help
scenarios
Section
successful
interface
technology
people
Statement
privacy
community
participants
chapter
study
increased
devices
changing
customers
become
Ethnographic
However
testing
engineering
important
experience
develop
project
web
building
organizational
organizations
documents
access
provide
review
impact
success
goals
scenario
include
legal
using
One
new
ensure
knowledge
concerns
benefits
ideas
list
end
even
build
application
make
business
mobile
like
Java
helps
et
costs
decisions
every
case
online
high
writing
changes
capture
change
effective
Interactive
control
small
large
vision
products
influence
used
address
clear
creative
interviews
plan
approach
source
video
stakeholders
environments
attention
understanding
key
analysis
four
carefully
expert
self
group
Current
often
protection
user-centered
sequences
team

CHAPTER 3

Managing Design Processes

"Just as we can assert that no product has ever been created in a single moment of inspiration . . . nobody has ever produced a set of requirements for any product in a similarly miraculous manner. These requirements may well begin with an inspirational moment but, almost certainly, the emergent bright idea will be developed by iterative processes of evaluation until it is thought to be worth starting to put pencil to paper. Especially when the product is entirely new, the development of a set of requirements may well depend upon testing initial ideas in some depth."

W. H. Mayall
Principles in Design, 1979

"The Plan is the generator. Without a plan, you have lack of order and willfulness. The Plan holds in itself the essence of sensation."

Le Corbusier
Towards a New Architecture, 1931

Written in collaboration with Steven M. Jacobs

CHAPTER OUTLINE

3.1 Introduction

In the first decades of computer-software development, technically oriented programmers designed text editors, programming languages, and applications for themselves and their peers. The substantial experience and motivation of these users meant that complex interfaces were accepted and even appreciated. Now, the user population for mobile devices, instant messaging, e-business, and digital libraries is so vastly different from the original that programmers' intuitions may be inappropriate. Current users are not dedicated to the technology; their background is more tied to their work needs and the work tasks they perform, while their use of computers for entertainment has increased. Designers who carefully observe current users, refine their prototypes by thoughtful analysis of task frequencies and sequences, and validate through early usability and thorough acceptance tests are likely to produce high-quality interfaces.

In the best organizations, the technocentric style of the past is yielding to a genuine desire to accommodate the users' skills, goals, and preferences. Designers seek direct interaction with users during requirements and feature definition, the design phase, the development process, and throughout the system lifecycle. Iterative design methods that allow early testing of low-fidelity prototypes, revisions based on feedback from users, and incremental refinements suggested by usability-test administrators are catalysts for high-quality systems.

Around the world, *usability engineering* has evolved into a recognized discipline with maturing practices and a growing set of standards. The Usability Professionals Association (UPA) has become a respected community with active participation from large corporations and numerous small design, test, and build firms. The UPA's annual "World Usability Day" sponsors hundreds of lectures plus visits to policy makers and industrial research decision makers. There is a movement to certify usability professionals based on a body of knowledge published by the UPA (Usability Professionals Assn., 2008). Also, usability test reports are becoming standardized (for example, via the Common Industry Format), so that buyers of software can compare products across suppliers.

The variety of design situations precludes a comprehensive strategy. Managers will have to adapt the strategies offered in this chapter (Section 3.2) to suit their organizations, projects, schedules, and budgets. These strategies begin with the organizational design that gives appropriate emphasis to support usability.

As a goal, we should push tools and development capabilities closer towards end users, particularly in the web domain. For examples, check out the tools for building Amazon.com "wish lists," the Google Mashup Editor, or Many Eyes shared visualizations. A willingness to be flexible and open in the development process and to offer some of these tailoring capabilities to the end user can increase the chances for successful user-interface development.

There are four pillars of successful user-interface development: user-interface requirements, guidelines documents and processes, user-interface software tools, and expert reviews and usability testing. These elements are introduced in Section 3.3. Then, in Section 3.4, development methodologies for successful user-interface development are discussed, and contextual inquiry and rapid contextual design are addressed as a framework for user-centered design (Holtzblatt et al., 2005).

Ethnographic observation (Section 3.5) is a proven enabler to the successful development process. Participatory design (Section 3.6) and scenario development (Section 3.7) are also critical to success. Social impact statements should be produced early in the design review (Section 3.8), and legal concerns should be addressed during the design process (Section 3.9).

3.2 Organizational Design to Support Usability

Corporate marketing and customer-assistance departments are becoming more aware of the importance of usability and are a source of constructive encouragement. When competitive products provide similar functionality, usability engineering is vital for product acceptance. Many organizations have created usability laboratories to provide expert reviews and to conduct usability tests of products during development. Outside experts can provide fresh insights, while usability-test subjects perform benchmark tasks in carefully supervised conditions (Rubin and Chisnell, 2008; Dumas and Redish, 1999). These and other evaluation strategies are covered in Chapter 4.

Companies may not yet have chief usability officers (CUOs) or vice presidents for usability, but they often have user-interface architects and usability engineering managers. High-level commitment helps to promote attention at every level. Organizational awareness can be stimulated by Usability Day presentations, internal seminars, newsletters, and awards. However, resistance to new techniques and changing roles for software engineers can cause problems in organizations.

Organizational change is difficult, but creative leaders blend inspiration and provocation. The high road is to appeal to the desire for quality that most professionals share. When they are shown data on shortened learning times, faster performance, or lower error rates on well-designed interfaces, managers are likely to be more sympathetic to applying usability-engineering methods. Even more compelling for e-commerce managers is evidence of higher rates of conversion, enlarged market share, and increased customer retention. For managers of consumer products, the goals include fewer returns/complaints, increased brand loyalty, and more referrals. The low road is to point out the frustration, confusion, and high error rates caused by current complex designs, while citing the successes of competitors who apply usability-engineering methods.

Return on investment (ROI) for usability engineering in major corporations is almost always questioned. However, there have been numerous white papers citing awareness and evidence that usability testing can pay dividends (Nielsen, 2008; Bias and Mayhew, 2005). Most large and many small organizations maintain a centralized human-factors group or usability laboratory as a source of expertise in design and testing techniques (Perfetti, 2006). However, each project should have its own user-interface architect who develops the necessary skills, manages the work of other people, prepares budgets and schedules, and coordinates with internal and external human-factors professionals when further expertise, references to the literature, or usability tests are required. This dual strategy balances the needs for centralized expertise and decentralized application. It enables professional growth in the user-interface area and in the application domain (for example, in geographic information or web-based product catalogs).

Some industries, such as in aerospace, are often required to address Human Systems Integration (HSI) requirements that deal with a combination of human factors, usability, display design, navigation, and so on, while meeting customer requirements for the same (National Research Council, 2007; Defense Acquisition University, 2004).

As the field of user-interface design has matured, projects have grown in complexity, size, and importance. Role specialization is emerging, as it has in fields such as architecture, aerospace, and book design. User-interface design takes on new perspectives when writing web, mobile, or desktop applications, with an emerging discipline in translating the same information across each of these media. Eventually, individuals will become highly skilled in specific problem areas, such as user-interface–building tools, graphic-display strategies, voice and audio tone design, shortcuts, navigation, and online tutorial writing. Consultation with graphic artists, book designers, advertising copywriters, instructional-textbook authors, game designers, or film-animation creators is expected. Perceptive system developers recognize the need to employ psychologists for conducting experimental tests, sociologists for evaluating organizational impact, educational psychologists for refining training procedures, and social workers for guiding customer-service personnel.

As design moves to implementation, the choice of user-interface–building tools is vital to success. These rapidly emerging tools enable designers to build novel systems quickly and support the iterative design/test/refine cycle.

Guidelines documents were originally seen as the answer to usability questions, but they are now appreciated as a broader social process in which the initial compilation is only the first step. Management strategies for the "four Es" presented here—education, enforcement, exemption, and enhancement—are just beginning to emerge and to become institutionalized.

The business case for focusing on usability has been made powerfully and repeatedly (Nielsen 2008; Bias and Mayhew, 2005; Marcus, 2002; Karat, 1994). It apparently needs frequent repetition, because traditional managers and engineers are often resistant to changes that would bring increased attention to the users' needs. Claire-Marie Karat's business-like reports within IBM (Karat, 1994) became influential documents when they were published externally. She reported up to $100 payoffs for each dollar spent on usability, with identifiable benefits in reduced program-development costs, reduced program-maintenance costs, increased revenue due to higher customer satisfaction, and improved user efficiency and productivity. Other economic analyses showed fundamental changes in organizational productivity (with improvements of as much as 720%) when designers kept usability in mind from the beginning of development projects (Landauer, 1995). Even minimal application of usability testing followed by correction of 20 of the easiest-to-repair faults improved user success rates from 19% to as much as 80%.

It is important to note that there are interface-development activities where the ROI for usability analysis during the development cycle is not immediately apparent, but true usability of the delivered system is crucial for success. One familiar example is voting machines. An end result of confused, misinterpreted voting results would be catastrophic and counter to the best interests of the voting population, but the usability analysis and associated development costs should be managable by the government contractor building the electronic voting booth system.

Usability engineers and *user-interface architects*, sometimes called the user experience (UX) team, are gaining experience in managing organizational change. As attention shifts away from software engineering or management-information systems, battles for control and power manifest themselves in budget and personnel allocations. Well-prepared managers who have a concrete organizational plan, defensible cost/benefit analyses, and practical development methodologies are most likely to be winners.

Design is inherently creative and unpredictable. Interactive system designers must blend a thorough knowledge of technical feasibility with a mystical esthetic sense of what attracts users. One method to characterize design (Rosson and Carroll, 2002) is:

- Design is a *process*; it is not a state and it cannot be adequately represented statically.
- The design process is *nonhierarchical*; it is neither strictly bottom-up nor strictly top-down.
- The process is *radically transformational*; it involves the development of partial and interim solutions that may ultimately play no role in the final design.
- Design intrinsically involves the *discovery of new goals*.

These characterizations of design convey the dynamic nature of the process. But in every creative domain, there can also be discipline, refined techniques, wrong and right methods, and measures of success. Once the early data collection is done and preliminary requirements are established, more detailed design and early development can begin. This chapter covers strategies for managing early stages of projects and presents design methodologies. Chapter 4 focuses on evaluation methods.

3.3 The Four Pillars of Design

> "If standardization can be humanized and made flexible in design and the economics brought to the home owner, the greatest service will be rendered to our modern way of life. It may be really born—this democracy, I mean."
>
> **Frank Lloyd Wright**
> *The Natural House*, 1954

The four pillars described in this section can help user-interface architects to turn good ideas into successful systems (Fig. 3.1). They are not guaranteed to work flawlessly, but experience has shown that each pillar can produce an order-of-magnitude speed-up in the process and can facilitate the creation of excellent systems.

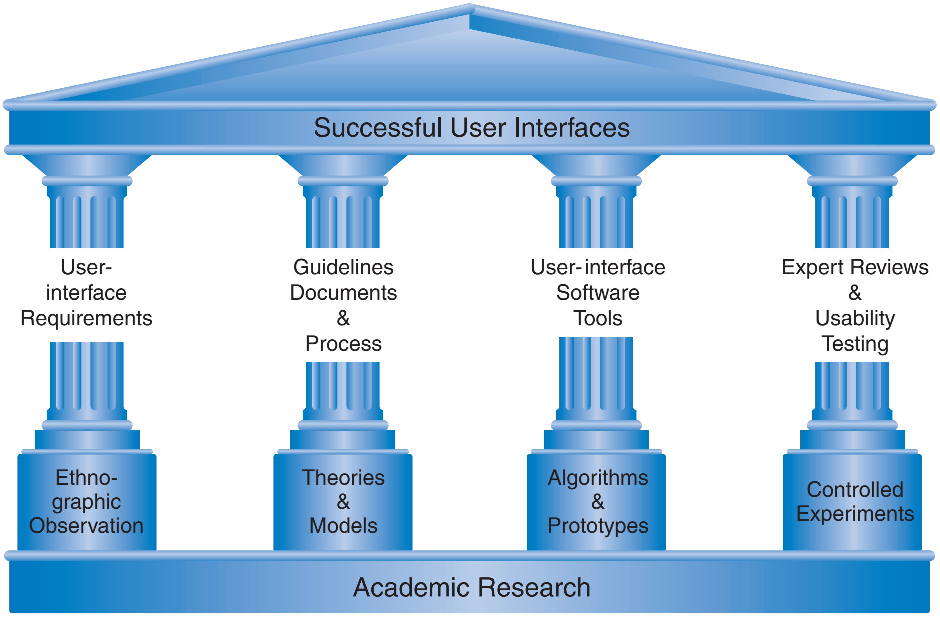

FIGURE 3.1

The four pillars of successful user-interface development.

3.3.1 User interface requirements

Soliciting and clearly specifying user requirements is a major key to success in any development activity (Selby, 2007). Methods to elicit and reach agreement upon user-interface requirements differ across organizations and industries, but the end result is the same: a clear specification of the user community and the tasks the users perform. Laying out the user-interface requirements is part of the overall requirements development and management process; the system requirements (hardware, software, system performance, reliability, etc.) must be clearly stated, and any requirements dealing with the user interface (input/output devices, functionality, interfaces, range of users, etc.) must be specified and agreed upon.

The success or failure of software projects often depends on the precision and completeness of the understanding among all the users and implementers. What happens without adequate requirements definition? You are not sure what problem you are solving, and you do not know when you are done.

Be careful not to impose human operator actions (requirements) onto the user-interface requirements (Box 3.1). For example, *do not* specify a requirement like this: “The user shall decide how much to withdraw from the ATM within

BOX 3.1

Examples of user-interface requirements regarding system behavior.

- Performance requirements:

  "The web site shall give users the ability to update their user profiles, e.g., name, mail address, e-mail address, phone."

  "The system shall permit the ATM customer 15 seconds to make a selection. The customer shall be warned that the session will be ended if no selection is made."

  "The mobile device shall be able to save draft text messages when out of the service area."

- Functional requirements:

  "The system shall ensure that the PIN entered matches the one on file."

  "The web site shall provide other, related purchase options based on past visits to the web site."

  "The credit card transaction must be approved prior to displaying a confirmation number."

- Interface requirements:

  "The web site shall permit ordering stamps online."

  "Kiosk screen styles shall conform to existing print media guidelines."

  "The mobile device shall permit downloading of ring tones."

five seconds." Rather, allocate that same requirement to the computer system: "The ATM shall permit a user five seconds to select a withdrawal amount . . . before prompting for a response."

One successful method for determining user-interface requirements is to use ethnographic observation (discussed in Section 3.5), monitoring the context and environment of real users in action. Tradeoffs between what functions are done best by computers versus humans in human-computer interaction (Section 2.3.6) should also be discussed at this point in the development process.

3.3.2 Guidelines documents and processes

Early in the design process, the user-interface architect should generate, or require other people to generate, a set of working guidelines. Two people might work for one week to produce a 10-page document, or a dozen people might work for two years to produce a 300-page document. One component of Apple's success with the Macintosh was the machine's early and readable guidelines document, which provided a clear set of principles for the many application developers to follow and thus ensured harmony in design across products. Microsoft's *Windows Vista User Experience Guidelines*, which have been refined over the years, also provide a good starting point and an educational experience for many programmers. These and other guidelines documents are referenced and described briefly in the general reference section at the end of Chapter 1.

Each project has different needs, but guidelines should be considered for:

- Words, icons, and graphics
  - Terminology (objects and actions), abbreviations, and capitalization
  - Character set, fonts, font sizes, and styles (bold, italic, underline)
  - Icons, buttons, graphics, and line thickness
  - Use of color, backgrounds, highlighting, and blinking
- Screen-layout issues
  - Menu selection, form fill-in, and dialog-box formats
  - Wording of prompts, feedback, and error messages
  - Justification, whitespace, and margins
  - Data entry and display formats for items and lists
  - Use and contents of headers and footers
  - Strategies for adapting to small and large displays
- Input and output devices
  - Keyboard, display, cursor control, and pointing devices
  - Audible sounds, voice feedback, speech I/O, touch input, and other special input modes or devices
  - Response times for a variety of tasks
  - Alternatives for users with disabilities
- Action sequences
  - Direct-manipulation clicking, dragging, dropping, and gestures
  - Command syntax, semantics, and sequences
  - Shortcuts and programmed function keys
  - Touchscreen navigation for devices such as the Apple iPhone and tabletop systems such as Microsoft Surface™
  - Error handling and recovery procedures
- Training
  - Online help, tutorials, and support groups
  - Training and reference materials

Guidelines creation (Box 3.2) should be a social process within an organization to help it gain visibility and build support. Controversial guidelines (for example,

BOX 3.2

Recommendations for guidelines documents.

- Provides a social process for developers
- Records decisions for all parties to see
- Promotes consistency and completeness
- Facilitates automation of design
- Allows multiple levels:
    - Rigid standards
    - Accepted practices
    - Flexible guidelines
- Announces policies for:
    - Education: How to get it?
    - Enforcement: Who reviews?
    - Exemption: Who decides?
    - Enhancement: How often?

on when to use voice alerts) should be reviewed by colleagues or tested empirically. Procedures should be established to distribute the guidelines, to ensure enforcement, to allow exemptions, and to permit enhancements. Guidelines documents must be living texts that are adapted to changing needs and refined through experience. Acceptance may be increased by a three-level approach of rigid standards, accepted practices, and flexible guidelines. This approach clarifies which items are firmer and which items are susceptible to change.

The creation of a guidelines document at the beginning of an implementation project focuses attention on the interface design and provides an opportunity for discussion of controversial issues. When the development team adopts the guidelines, the implementation proceeds quickly and with few design changes. For large organizations there may be two or more levels of guidelines, to provide organizational identity while allowing projects to have distinctive styles and local control of terminology. Some organizations develop "style guides" to capture this (see, for example, Microsoft, 2008).

The "four Es" provide a basis for creating a living document and a lively process:

- *Education*. Users need training and a chance to discuss the guidelines. Developers must be trained in the resultant guidelines.
- *Enforcement*. A timely and clear process is necessary to verify that an interface adheres to the guidelines.

- *Exemption*. When creative ideas or new technologies are used, a rapid process for gaining exemption is needed.
- *Enhancement*. A predictable process for review, possibly annually, will help keep the guidelines up-to-date.

3.3.3 User-interface software tools

One difficulty in designing interactive systems is that customers and users may not have a clear idea of what the system will look like when it is done. Since interactive systems are novel in many situations, users may not realize the implications of design decisions. Unfortunately, it is difficult, costly, and time-consuming to make major changes to systems once those systems have been implemented.

Although this problem has no complete solution, some of the more serious difficulties can be avoided if, at an early stage, the customers and users can be given a realistic impression of what the final system will look like (Gould and Lewis, 1985). A printed version of the proposed displays is helpful for pilot tests, but an onscreen display with an active keyboard and mouse is more realistic. The prototype of a menu system may have only one or two paths active, instead of the thousands of paths envisioned for the final system. For a form-fill-in system, the prototype may simply show the fields but not actually process them. Prototypes have been developed with simple drawing or word-processing tools or even PowerPoint® presentations of screen drawings manipulated with PowerPoint slide shows and other animation. Flash® and Ajax can also be used. Flash is a multimedia authoring and delivery platform for embedded web content. Building an interface in Flash is comparable to using other tools. Ajax is a combination of technologies for interactive web pages, much closer to a development environment. Other design tools that can be used are Adobe® PageMaker® or Illustrator®.

Development environments such as Microsoft's Visual Basic/C++ are easy to get started with yet have an excellent set of features. Visual Studio®, as well as C# and the .NET Framework, certainly can be evaluated for your user-interface development project. Make sure to evaluate tool capabilities, ease of use, ease to learn, cost, and performance. Tailor your tool choices for the size of the job. Building a software architecture that supports your user-interface development project is just as important as it is for any other (particularly large-scale) software development activity.

Sophisticated tools such as Sun's Java™ provide cross-platform development capabilities and a variety of services. People who want to write their own Java programs can use the Java Development Kit™ (JDK). The *Java Look and Feel Design Guidelines* (Sun Microsystems, 2001) is a terrific reference on the user-interface style for Java developers writing with Java Foundation Classes

(JFC). As an example of how rapidly this field is changing, at the time of this writing, Java released the LightWeight User Interface Toolkit (LWUIT), billed as "a versatile and compact API for creating attractive application mobile user interfaces."

Although they are not the focus of this book, there are many web sites offering insight into current user-interface software tools; one of the authors' favorites is the Web Developers Journal.

3.3.4 Expert reviews and usability testing

Theatrical producers know that extensive rehearsals and previews for critics are necessary to ensure a successful opening night. Early rehearsals may involve only the key performers wearing street clothes, but as opening night approaches, dress rehearsals with the full cast, props, and lighting are required. Aircraft designers carry out wind-tunnel tests, build plywood mock-ups of the cabin layout, construct complete simulations of the cockpit, and thoroughly flight-test the first prototype. Similarly, web-site designers now recognize that they must carry out many small and some large pilot tests of components before release to customers (Dumas and Redish, 1999). In addition to a variety of expert review methods, tests with the intended users, surveys, and automated analysis tools are proving to be valuable. Procedures vary greatly depending on the goals of the usability study, the number of expected users, the danger of errors, and the level of investment. Chapter 4 covers expert reviews, usability testing, and other evaluation methods in depth.

3.4 Development Methodologies

Many software development projects fail to achieve their goals. Some estimates of the failure rate put it as high as 50% (Jones, 2005). Much of this problem can be traced to poor communication between developers and their business clients or between developers and their users.

Successful developers work carefully to understand the business's needs it and refine their skills in eliciting accurate requirements from nontechnical business managers. In addition, since business managers may lack the technical knowledge to understand proposals made by the developers, dialog is necessary to reduce confusion about the organizational implications of design decisions.

Successful developers also know that careful attention to user-centered design issues during the early stages of software development dramatically reduces both development time and cost. User-centered design leads to systems that generate fewer problems during development and have lower maintenance costs over their lifetimes. They are easier to learn, result in faster performance, reduce user errors substantially, and encourage users to explore features that go beyond the minimum required to get by. In addition, user-centered design practices help organizations align system functionality with their business needs and priorities.

Software developers have learned that consistently following established development methodologies can help them meet budgets and schedules (Sommerville, 2006; Pfleeger, 2005). But while software-engineering methodologies are effective in facilitating the software development process, they have not always provided clear processes for studying the users, understanding their needs, and creating usable interfaces. Small consulting firms that specialize in user-centered design have created innovative design methodologies to guide developers, such as rapid contextual design (Holtzblatt et al., 2005), which is based on the approach of contextual inquiry (Beyer and Holtzblatt, 1998). Some large corporations have also integrated user-centered design into their practices; for example, IBM's Ease of Use method fits with its existing corporate methods (Fig. 3.2). Agile technologies and methodologies provide

Role/Phase Matrix	**All Phases**	Business Opportunity	Understanding Users	Initial Design	Development	Deployment	Life Cycle
All Roles							
User Experience Leadership		User Engineering Plan-Initial	User Engineering Plan-Final	Execution of the User Engineering Plan	Satisfaction of Established Metrics	Project Assessment	Satisfaction Survey
Market Planning		Business and Market Requirements	Appropriate User Requirements	Draft Marketing Collateral	Detail Marketing Collateral	Final Marketing Collateral	
User Research			User Requirements	Appropriate Design			
User Experience Design			Design Direction	Conceptual Design, Low-Fidelity Prototypes	Detail Design, High-Fidelity Prototypes	Design Issue Resolution	
Visual & Industrial Design			Appearance Direction	Appearance Guidelines	Appearance Specification		
User Experience Evaluation			Competitive Evaluation	Conceptual Design Evaluation	Detail Design Evaluations	User Feedback and Benchmark	Usage Issue Report

FIGURE 3.2

IBM's Ease of Use development methodology, which specifies activities by roles and phases.

the room to be responsive to user-interface development and usability needs (Boehm and Turner, 2004).

These business-oriented approaches specify detailed deliverables for the various stages of design and incorporate cost/benefit and return-on-investment analyses to facilitate decision making. They may also offer management strategies to keep projects on track and to facilitate effective collaboration among teams that include both business and technical participants. Since user-centered design is only a part of the overall development process, these methodologies must also mesh with the various software-engineering methodologies that are used in industry today.

There are dozens of advertised development methods (such as GUIDE, STUDIO, and OVID), but the focus here is on Holtzblatt et al.'s rapid contextual design, summarized below. There are tools for contextual design and managing the data (Holtzblatt and Beyer, 2008), and there are other recent, excellent sources for usability engineering processes and interaction design (Heim, 2008; Sears and Jacko, 2008; Leventhal and Barnes, 2007; Sharp et al., 2007). The rapid contextual design method (Table 3.1) involves the following steps:

1. *Contextual inquiry*. Plan for, prepare, and then conduct field interviews to observe and understand the work tasks being performed. Review business practices.

2. *Interpretation sessions and work modeling*. Hold team discussions to draw conclusions based on the contextual inquiry, including gaining an understanding of the workflow processes in the organization as well as cultural and policy impacts on work performed. Capture key points (affinity notes).

Contextual inquiry
Interpretation sessions and work modeling
Model consolidation and affinity diagram building
Personas
Visioning
Storyboarding
User environment design
Paper prototypes and mock-up interviews

TABLE 3.1

Rapid contextual design from *Rapid Contextual Design: A How-To Guide to Key Techniques for User-Centered Design*, Morgan Kaufmann, San Francisco, CA (2005).

3. *Model consolidation and affinity diagram building*. Present the data gathered to date from users and the interpretation and work modeling to a larger, targeted population to gain insight and concurrence. Consolidate the work models to illustrate common work patterns and processes and create *affinity diagrams* (hierarchical representations of the issues to address user needs).

4. *Persona development*. Develop personas (fictitious characters) to represent the different user types within a targeted demographic that might use a site or product (Cooper, 2004). This aids the team in communicating the needs of the users and bringing those user needs to fruition. Examples of personas, at a high level, are: 1) 22-year-old male with 5+ years of video game playing experience, or 2) 70-year-old female using computer only for e-mail and digital photo sharing.

5. *Visioning*. Review and "walk" the consolidated data, sharing the personas created. The visioning session helps define how the system will streamline and transform the work of the users. Capture key issues and ideas using flipcharts or any media that will facilitate expressing the vision of the revised business processes.

6. *Storyboarding*. The vision guides the detailed redesign of user tasks using pictures and graphs to describe the initial user-interface concepts, business rules, and automation assumptions. Storyboarding defines and illustrates the "to be built" assumptions.

7. *User environment design*. The single, coherent representation of the users and the work to be performed is expressed in the user environment design (UED). The UED is built from the storyboards.

8. *Interviews and evaluations with paper prototypes and mock-ups*. Conduct interviews and tests with actual users, beginning with paper prototypes and then moving on to higher-fidelity prototypes. Capturing the results of the interviews aids in ensuring that the systems will meet end-user requirements.

3.5 Ethnographic Observation

The early stages of most methodologies include observation of users. Since interface users form a unique culture, ethnographic methods for observing them in the workplace are becoming increasingly important (Fig. 3.3). Ethnographers join

FIGURE 3.3

Preteen researchers with the University of Baltimore's KidsTeam observe children's reading habits in the home (left). Researchers in Paris brainstorm ideas for new family technologies with families from France, Sweden, and the United States (right).

work or home environments to listen and observe carefully, sometimes stepping forward to ask questions and participate in activities (Fetterman, 1998; Harper, 2000; Millen, 2000). As ethnographers, user-interface designers gain insight into individual behavior and the organizational context. However, they differ from traditional ethnographers in that, in addition to seeking understanding of their subjects, user-interface designers focus on interfaces for the purpose of changing and improving those interfaces. Also, whereas traditional ethnographers immerse themselves in cultures for weeks or months, user-interface designers usually need to limit this process to a period of days or even hours to obtain the relevant data needed to influence a redesign (Hughes et al., 1997). Ethnographic methods have been applied to office work (Suchman, 1987), air-traffic control (Bentley et al., 1992), and other domains (Marcus, 2005).

The goal of this observation is to obtain the necessary data to influence interface redesign. Unfortunately, it is easy to misinterpret observations, to disrupt normal practice, and to overlook important information. Following a validated ethnographic process reduces the likelihood of these problems. Examples of ethnographic observation research include: 1) how cultural probes have been adopted and adapted by the HCI community (Boehner et al., 2007), 2) development of an interactive location-based service for supporting distributed mobile collaboration for home healthcare (Christensen et al., 2007), and 3) social dynamics influencing technological solutions in developing regions (Ramachandran et al., 2007). Guidelines for preparing for the evaluation, performing the field study, analyzing the data, and reporting the findings might include the following:

- Preparation
  - Understand policies in work environments and family values in homes.
  - Familiarize yourself with the existing interface and its history.
  - Set initial goals and prepare questions.
  - Gain access and permission to observe or interview.
- Field study
  - Establish a rapport with all users.
  - Observe or interview users in their setting, and collect subjective and objective quantitative and qualitative data.
  - Follow any leads that emerge from the visits.
  - Record your visits.
- Analysis
  - Compile the collected data in numerical, textual, and multimedia databases.
  - Quantify data and compile statistics.
  - Reduce and interpret the data.
  - Refine the goals and the process used.
- Reporting
  - Consider multiple audiences and goals.
  - Prepare a report and present the findings.

These notions seem obvious when stated, but they require interpretation and attention in each situation. For example, understanding the differing perceptions that managers and users have about the efficacy of the current interface will alert you to the varying frustrations of each group. Managers may complain about the unwillingness of staff to update information promptly, but staff may be resistant to using the interface because the log-in process takes six to eight minutes. Respecting the rules of the workplace is important for building rapport: In preparing for one observation, we appreciated that the manager called to warn us that graduate students should not wear jeans because the users were prohibited from doing so. Learning the technical language of the users is also vital for establishing rapport. It is useful to prepare a long list of questions that you can then filter down by focusing on the proposed goals. Awareness of the differences between user communities, such as those mentioned in Section 1.4, will help to make the observation and interview process more effective.

Data collection can include a wide range of subjective impressions that are qualitative or of subjective reactions that are quantitative, such as rating scales or rankings. Objective data can consist of qualitative anecdotes or critical incidents that capture user experiences, or can be quantitative reports about, for example, the number of errors that occur during a one-hour observation of six users. Deciding in advance what to capture is highly beneficial, but remaining alert to unexpected happenings is also valuable. Written report summaries have proved to be valuable, far beyond expectations; in most cases, raw transcripts of every conversation are too voluminous to be useful.

Making the process explicit and planning carefully may seem awkward to many people whose training stems from computing and information technology. However, a thoughtfully applied ethnographic process has proved to have many benefits. It can increase trustworthiness and credibility, since designers learn about the complexities of the intended environment by visits to the workplace, school, home, or other environment where the eventual system will be deployed. Personal presence allows designers to develop working relationships with several end users to discuss ideas, and, most importantly, the users may consent to be active participants in the design of their new interface.

3.6 Participatory Design

Many authors have urged participatory design strategies, but the concept is controversial. *Participatory design* is the direct involvement of people in the collaborative design of the things and technologies they use. The arguments in favor suggest that more user involvement brings more accurate information about tasks and an opportunity for users to influence design decisions.

However, the sense of participation that builds users' ego investment in successful implementation may be the biggest influence on increased user acceptance of the final system (Kujala, 2003; Muller, 2002; Damodaran, 1996).

On the other hand, extensive user involvement may be costly and may lengthen the implementation period. It may also generate antagonism from people who are not involved or whose suggestions are rejected, and potentially force designers to compromise their designs to satisfy incompetent participants (Ives and Olson, 1984).

Participatory design experiences are usually positive, however, and advocates can point to many important contributions that would have been missed without user participation. People who are resistant might appreciate the somewhat formalized multiple-case-studies *plastic interface for collaborative technology initiatives through video exploration* (PICTIVE) approach (Rosson and Carroll, 2006; Muller, 1992). Users sketch interfaces, then use slips of paper, pieces of plastic, and tape to create low-fidelity early prototypes. A scenario walkthrough is then recorded on videotape for presentation to managers, users, or other designers. With the right leadership, the PICTIVE approach can effectively elicit new ideas and be fun for all involved (Muller et al., 1993). Many variations of participatory design have been proposed that engage participants to create dramatic performances, photography exhibits, games, or merely sketches and written scenarios. High-fidelity prototypes and simulations can also be key in eliciting user requirements.

Careful selection of users helps to build a successful participatory design experience. A competitive selection increases participants' sense of importance and emphasizes the seriousness of the project. Participants may be asked to commit to repeated meetings and should be told what to expect about their roles and their influence. They may have to learn about the technology and business plans of the organization and be asked to act as a communication channel to the larger group of users that they represent.

The social and political environment surrounding the implementation of complex interfaces is not amenable to study by rigidly defined methods or controlled experimentation. Social and industrial psychologists are interested in these issues, but dependable research and implementation strategies may never emerge. The sensitive project leader must judge each case on its merits and must decide what is the right level of user involvement. The personalities of the participatory design team members are such critical determinants that experts in group dynamics and social psychology may be useful as consultants. Many questions remain to be studied, such as whether homogeneous or diverse groups are more successful, how to tailor processes for small and large groups, and how to balance decision-making control between typical users and professional designers.

Socio-technical system (STS) developers, who work on complex systems for applications such as transportation security, voting, online auctions, e-learning, and healthcare delivery, are increasingly aware of the value of participatory design. They seek user input from stakeholders at every stage to understand sensitive issues such as privacy protection, damage from errors, costs of delays, and legal constraints, as well as ethical issues, such as bias that favors one user group or exclusion that raises barriers for another user group (Whitworth and De Moore, 2009).

The experienced user-interface architect knows that organizational politics and the preferences of individuals may be more important than technical issues in governing the success of an interactive system. For example, warehouse managers who see their positions threatened by an interactive system that provides senior managers with up-to-date information through desktop displays may try to ensure that the system fails by delaying data entry or by being less than diligent in guaranteeing data accuracy. The interface designer should take into account the system's effect on users and should solicit their participation to ensure that all concerns are made explicit early enough to avoid counterproductive efforts and resistance to change. Novelty is threatening to many people, so clear statements about what to expect can be helpful in reducing anxiety.

Ideas about participatory design are being refined with diverse users, ranging from children to older adults. Arranging for participation is difficult for some users, such as those with cognitive disabilities or those whose time is precious (for example, surgeons). The levels of participation are becoming clearer; one taxonomy describes the roles of children in developing interfaces for children, older adults in developing interfaces whose typical users will be other older adults, and so on, with roles varying from testers to informants to partners (Druin, 2002; Fig. 3.4). Testers are merely observed as they try out novel designs, while informants comment to designers through interviews and focus groups. Design partners are active members of a design team, which in the case of children's software will naturally involve participants of many ages—the intergenerational team.

Further research in this area is published at Participatory Design Conferences (PDCs), held biennially since 1990. The PDC conferences are sponsored by Computer Professionals for Social Responsibility (CPSR). For ethnographic observation and participatory design, developers and project managers who regularly strive to get buy-ins from diverse participants are more likely to succeed.

3.7 Scenario Development

When a current interface is being redesigned or a well-polished manual system is being automated, reliable data about the distribution of task frequencies and

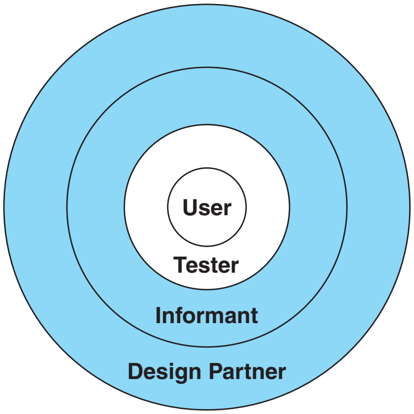

FIGURE 3.4

Druin's model of the four levels of user participation. The blue areas (informant and design partner) represent stages of participatory design.

sequences is an enormous asset. If current data do not exist, usage logs can quickly provide insight.

A table with user communities listed across the top and tasks listed down the side is helpful. Each box can then be filled in with the relative frequency with which each user performs each task. Another representation tool is a table of task sequences, indicating which tasks follow other tasks. Often, a flowchart or transition diagram helps designers to record and convey the sequences of possible actions; the thickness of the connecting lines indicates the frequency of the transitions.

In less well-defined projects, many designers have found day-in-the-life scenarios helpful to characterize what happens when users perform typical tasks. During the early design stages, data about current performance should be collected to provide a baseline. Information about similar systems helps, and interviews can be conducted with stakeholders, such as users and managers (Rosson and Carroll, 2002; Bodker, 2000; Carroll, 2000).

An early and easy way to describe a novel system is to write scenarios of usage and then, if possible, to act them out as a form of theater. This technique can be especially effective when multiple users must cooperate (for example, in control rooms, cockpits, or financial trading rooms) or multiple physical devices are used (for example, at customer-service desks, medical laboratories, or hotel check-in areas). Scenarios can represent common or emergency situations with both novice and expert users. Personas can also be included in scenario generation.

In developing the National Digital Library, the design team began by writing 81 scenarios that portrayed typical needs of potential users. Here is an example:

> *K–16 Users*: A seventh-grade social-studies teacher is teaching a unit on the Industrial Revolution. The teacher wants to make use of primary source material that would illustrate the factors that facilitated industrialization, the manner in which it occurred, and the impact that it had on society and on the built environment. Given the teaching load, the teacher only has about four hours total to locate and package the supplementary material for classroom use.

Other scenarios might describe how users explore a system, such as this optimistic vision, written for the U.S. Holocaust Museum and Education Center:

> A grandmother and her 10- and 12-year old grandsons have visited the museum before. They have returned this time to the Learning Center to explore what life was like in her shtetl in Poland in the 1930s. One grandson eagerly touches the buttons on the welcome screen, and they watch the 45-second video introduction by the museum director. They then select the button on "History before the Holocaust" and choose to view a list of towns. Her small town is not on the list, but she identifies the larger nearby city, and they get a brief textual description, a map of the region, and a photograph of the marketplace. They read about the history of the town and view 15-second videos of the marketplace activity and a Yiddish theater production. They bypass descriptions of key buildings and institutions, choosing instead to read biographies of a famous community leader and a poet. Finally, they select "GuestBook" and add their names to the list of people who have indicated an affiliation with this town. Further up on the list, the grandmother notices the name of a childhood friend from whom she has not heard in 60 years—fortunately, the earlier visitor has left an address.

This scenario was written to give nontechnical museum planners and the Board of Directors an idea of what could be built if funding were provided. Such scenarios are easy for most people to grasp, and they convey design issues such as physical installation (room and seats for three or more patrons with sound isolation) and development requirements (video production for the director's introduction and conversion of archival films to video).

An elaborate scenario development process was also conducted to help U.S. statistical agencies formulate a vision for a Statistical Knowledge Network. Patterns of citizen requests were combined with agency proposals to develop 15 brief scenarios, using the first-person format, such as these two that were the basis of empirical tests of proposed interfaces:

> I'm a social activist in the Raleigh-Durham, North Carolina area and have become increasingly concerned about urban sprawl and the loss of rural

> areas for both farming and recreation. I need statistics to support my claim that significant differences occur when urban development occurs in rural and/or farming areas.
>
> I would like to open a grocery store specializing in organic products in the greater Seattle metropolitan area. What are the trends in production and consumption of organic food products? Would the Seattle area be a good place to locate?

Some scenario writers take a further step and produce videotapes to convey their intentions. There have been several famous future scenarios, including Apple's *Knowledge Navigator* (1987), which produced numerous controversies. It portrayed a professor using voice commands to talk with a bow-tied preppie character on the screen and touch commands to develop ecological simulations. Many viewers enjoyed the tape, but thought that it stepped over the bounds of reality by having the preppie agent recognize the professor's facial expressions, verbal hesitations, and emotional reactions. Another example is Bruce Tognazzini's *Starfire* scenario for Sun Microsystems (1994), which gave his elaborate but realistic impression of a large-screen work environment that supported rich collaborations with remote colleagues.

By 2003, cell-phone developers were producing scenarios about how personal, family, and commercial relationships would change due to mobile video communications—an appealing example is the Japanese NTT DoCoMo's *Vision 2010: Beyond the Mobile Frontier*, which shows how a family can realize its goal of remaining in close contact while children go to study far away from home. NTT DoCoMo has produced several other scenarios worth viewing, including *Mobile Life Story "Concert" version*, *Vision 2010: Old School Friends*, and *The Road to Hokusai's Waterfall*. The DoCoMo videos all portray a wonderful, technology-enhanced future, with easy-to-use interfaces on reliable mobile devices that are accessible to all ages. The video scenarios enforce our need for secure, private data communication to enhance our personal security, health, and safety.

Another scenario of note is Microsoft's *Health Future Vision*, a futuristic look at interconnected mobile communications technology in healthcare. Concepts that are illustrated include: remote transmittal of personal health status information; hospital communications and collaboration tools; advanced, accessible user interfaces using mobile, touchscreen technology; controls for secure patient health and identification information; environments (walls and furniture) that serve seamlessly as input/output devices; and more.

3.8 Social Impact Statement for Early Design Review

Interactive systems often have a dramatic impact on large numbers of users. To minimize risks, a thoughtful statement of anticipated impacts circulated among stakeholders can be a useful process for eliciting productive suggestions early in the development, when changes are easiest.

Governments, utilities, and publicly regulated industries increasingly require information systems to provide services. However, some critics have strong negative attitudes towards modern technologies and see only a hopeless technological determinism: "Technopoly eliminates alternatives to itself. It consists in the deification of technology, which means that the culture seeks its authorization in technology, finds its satisfactions in technology, and takes its orders from technology" (Postman, 1993).

Postman's endless fears do not help us to shape more effective technology or to prevent damage from technology failures. However, constructive criticism and guidelines for design could be helpful in reversing the long history of incorrect credit histories, dislocation through de-skilling or layoffs, and deaths from flawed medical instruments. Current concerns focus on privacy invasion from surveillance systems, government attempts to restrict access to information, and voting fraud because of poor security. While guarantees of perfection are not possible, policies and processes can be developed that will more often than not lead to satisfying outcomes.

A *social impact statement*, similar to an environmental impact statement, might help to promote high-quality systems in government-related applications (reviews for private-sector corporate projects would be optional and self-administered). Early and widespread discussion can uncover concerns and enable stakeholders to state their positions openly. Of course, there is the danger that these discussions will elevate fears or force designers to make unreasonable compromises, but these risks seem reasonable in a well-managed project. An outline for a social impact statement might include these sections (Shneiderman and Rose, 1996):

- Describe the new system and its benefits.
  - Convey the high-level goals of the new system.
  - Identify the stakeholders.
  - Identify specific benefits.
- Address concerns and potential barriers.
  - Anticipate changes in job functions and potential layoffs.
  - Address security and privacy issues.
  - Discuss accountability and responsibility for system misuse and failure.
  - Avoid potential biases.
  - Weigh individual rights versus societal benefits.

  - Assess tradeoffs between centralization and decentralization.
  - Preserve democratic principles.
  - Ensure diverse access.
  - Promote simplicity and preserve what works.
- Outline the development process.
  - Present an estimated project schedule.
  - Propose a process for making decisions.
  - Discuss expectations of how stakeholders will be involved.
  - Recognize needs for more staff, training, and hardware.
  - Propose a plan for backups of data and equipment.
  - Outline a plan for migrating to the new system.
  - Describe a plan for measuring the success of the new system.

A social impact statement should be produced early enough in the development process to influence the project schedule, system requirements, and budget. It can be developed by the system design team, which might include end users, managers, internal or external software developers, and possibly clients. Even for large systems, the social impact statement should be of a size and complexity that make it accessible to users with relevant backgrounds.

After the social impact statement is written, it should be evaluated by the appropriate review panel as well as by managers, other designers, end users, and anyone else who will be affected by the proposed system. Potential review panels include federal government units (for example, the General Accounting Organization or Office of Personnel Management), state legislatures, regulatory agencies (for example, the Securities and Exchange Commission or the Federal Aviation Administration), professional societies, and labor unions. The review panel will receive the written report, hold public hearings, and request modifications. Citizen groups also should be given the opportunity to present their concerns and to suggest alternatives.

Once the social impact statement is adopted, it must be enforced. A social impact statement documents the intentions for the new system, and the stakeholders need to see that those intentions are backed up by actions. Typically, the review panel is the proper authority for enforcement.

The effort, cost, and time involved should be appropriate to the project, while facilitating a thoughtful review. The process can offer large improvements by preventing problems that could be expensive to repair, improving privacy protection, minimizing legal challenges, and creating more satisfying work environments. Information-system designers take no Hippocratic Oath, but pledge themselves to strive for the noble goal of excellence in design can win respect and inspire others.

3.9 Legal Issues

As user interfaces have become more prominent, serious legal issues have emerged. Every developer of software and information should review legal issues that may affect design, implementation, or marketing.

Privacy is always a concern whenever computers are used to store data or to monitor activity. Medical, legal, financial, and other data often have to be protected to prevent unapproved access, illegal tampering, inadvertent loss, or malicious mischief. Recently implemented privacy assurance laws such as those imposed on the medical and financial communities can lead to complicated, hard-to-understand policies and procedures. Physical security measures to prohibit access are fundamental; in addition, privacy protection can involve user-interface mechanisms for controlling password access, identity checking, and data verification. Effective protection provides a high degree of privacy with a minimum of confusion and intrusion into work. Website developers should provide easily accessible and understandable privacy policies.

A second concern encompasses safety and reliability. User interfaces for aircraft, automobiles, medical equipment, military systems, utility control rooms, and the like can affect life-or-death decisions. If air-traffic controllers are confused by the situation display, they can make fatal errors. If the user interface for such a system is demonstrated to be difficult to understand, it could leave the designer, developer, and operator open to a lawsuit alleging improper design. Designers should strive to make high-quality and well-tested interfaces that adhere to state-of-the-art design guidelines. Accurate records documenting testing and usage will protect designers in case problems arise.

A third issue is copyright or patent protection for software (Samuelson, 2007; Lessig, 2006). Software developers who have spent time and money developing a package are frustrated in their attempts to recover their costs and to make a profit if potential users make illegal copies of the package, rather than buying it. Technical schemes have been tried to prevent copying, but clever hackers can usually circumvent the barriers. It is unusual for a company to sue an individual for copying a program, but cases have been brought against corporations and universities. There is also a vocal community of developers, led by the League for Programming Freedom, that opposes software copyright and patents, believing that broad dissemination is the best policy. An innovative legal approach, Creative Commons™, enables authors to specify more liberal terms for others to use their works. The open-source software movement has enlivened these controversies. The Open Source Initiative describes the movement as follows: "When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing." Some open-source products, such as the Linux® operating system and the Apache™ web server, have become successful enough to capture a substantial fraction of the market share.

A fourth concern is with copyright protection for online information, images, or music. If customers access an online resource, do they have the right to store the information electronically for later use? Can the customer send an electronic copy to a colleague or friend? Who owns the "friends" list and other shared data in social networking sites? Do individuals, their employers, or network operators own the information contained in e-mail messages? The expansion of the World Wide Web, with its vast digital libraries, has raised the temperature and pace of copyright discussions. Publishers seek to protect their intellectual assets, while librarians are torn between their desire to serve patrons and their obligations to publishers. If copyrighted works are disseminated freely, what incentives will there be for publishers and authors? If it is illegal to transmit any copyrighted work without permission or payment, science, education, and other fields will suffer. The fair-use doctrine of limited copying for personal and educational purposes helped cope with the questions raised by photocopying technologies. However, the perfect rapid copying and broad dissemination permitted by the Internet demand a thoughtful update (Lessig, 2001; Samuelson, 2003).

A fifth issue is freedom of speech in electronic environments. Do users have a right to make controversial or potentially offensive statements through e-mail or listservers? Are such statements protected by the First Amendment? Are networks like street corners, where freedom of speech is guaranteed, or are networks like television broadcasting, where community standards must be protected? Should network operators be responsible for or prohibited from eliminating offensive or obscene jokes, stories, or images? Controversy has raged over whether Internet service providers have a right to prohibit e-mail messages that are used to organize consumer rebellions against themselves. Another controversy emerged over whether a network operator has a duty to suppress racist e-mail remarks or postings to a bulletin board. If libelous statements are transmitted, can a person sue the network operator as well as the source? Should designers build systems where the default is to "opt out" of lists, and users have to explicity "opt in" by making a selection from a dialog box?

Other legal concerns include adherence to laws requiring equal access for disabled users and attention to changing laws in countries around the world. Do Yahoo! and eBay have to enforce the laws of every country in which they have customers? These and other issues mean that developers of online services must be sure to consider all the legal implications of their design decisions.

NetCoalition is a collective public policy organization that monitors many of the legal issues raised here; its web site is an excellent source for information about privacy legislation and related issues. There are also many other legal issues to be aware of today, including anti-terrorism, counterfeiting, spam, spyware, liability, Internet taxation, and others. These issues certainly require your attention, and legislation may eventually be needed.

Practitioner's Summary

Usability engineering is maturing rapidly, with once-novel ideas becoming standard practices. Usability has increasingly taken center stage in organizational and product planning. Development methodologies such as contextual design help by offering validated processes with predictable schedules and meaningful deliverables. Ethnographic observation can provide information to guide task analysis and to complement carefully supervised participatory design processes. Logs of usage provide valuable data about task frequencies and sequences. Scenario writing helps to bring common understanding of design goals, is useful for managerial and customer presentations, and helps to plan usability tests. For interfaces developed by governments, public utilities, and regulated industries, an early social impact statement can elicit public discussion that is likely to identify problems and produce interfaces that have high overall societal benefits. Designers and managers should obtain legal advice to ensure adherence to laws and protection of intellectual property.

Researcher's Agenda

Human-interface guidelines are often based on best-guess judgments rather than on empirical data. More research could lead to refined standards that are more complete and dependable, and to more precise knowledge of how much improvement can be expected from a design change. Because technology is continually changing, we will never have a stable and complete set of guidelines, but scientific studies will have enormous benefits in terms of reliability and the quality of decision making about user interfaces. Design processes, ethnographic methods, participatory design activities, scenario writing, and social impact statements are evolving. Variations are needed to address international diversity, special populations such as children or older adults, and long-term studies of actual usage. Thoughtful case studies of design processes would lead to their refinement and promote more widespread application. Creative processes are notoriously difficult to study, but well-documented examples of success stories will inform and inspire.

WORLD WIDE WEB RESOURCES

http://www.aw.com/DTUI/

Design processes promoted by companies and professional standards organizations, with information on how to develop style guidelines, are available online. References to guidelines documents are included in Chapter 1.

References

Bentley, R., Hughes, J., Randall, D., Rodden, T., Sawyer, P., Shapiro, D., and Sommerville, I., Ethnographically-informed systems design for air traffic control, *Proc. CSCW '92 Conference: Sharing Perspectives*, ACM Press, New York (1992), 123–129.

Beyer, Hugh and Holtzblatt, Karen, *Contextual Design: Defining Customer-Centered Systems*, Morgan Kaufmann, San Francisco, CA (1998).

Bias, Randolph and Mayhew, Deborah (Editors), *Cost-Justifying Usability: An Update for the Internet Age, Second Edition*, Morgan Kaufmann, San Francisco, CA (2005).

Bodker, Susan, Scenarios in user-centered design – Setting the stage for reflection and action, *Interacting with Computers* 13, 1 (2000), 61–76.

Boehm, Barry and Turner, Richard, *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison-Wesley, Reading, MA (2004).

Boehner, Kirsten, Vertesi, Janet, Sengers, Phoebe, and Dourish, Paul, How HCI interprets the probes, *Proc. Conference on Human Factors in Computing Systems (ACM CHI 2007)*, ACM Press, New York (2007), 1077–1086.

Carroll, John M. (Editor), *Making Use: Scenario-Based Design of Human-Computer Interactions*, MIT Press, Cambridge, MA (2000).

Christensen, Claus M., Kjeldskov, Jesper, and Rasmussen, Klaus K., GeoHealth: A location-based service for nomadic home healthcare workers, *Proc. 2007 Conference of the Computer-Human Interaction Special Interest Group (CHISIG) of Australia on Computer-Human Interaction: Design: Activities, Artifacts and Environments*, ACM Press, New York (2007), 273–281.

Cooper, Alan, *The Inmates are Running the Asylum*, Sams, New York (2004).

Damodaran, Leela, User involvement in the systems design process – A practical guide for users, *Behaviour & Information Technology* 15, 6 (1996), 363–377.

Defense Acquisition University, *Defense Acquisition Guidebook – Chapter 6, Human Systems Integration (HSI)* (2004). Available at https://akss.dau.mil/dag/welcome.asp.

Druin, Allison, The role of children in the design of new technology, *Behaviour & Information Technology* 21, 1 (2002), 1–25.

Dumas, Joseph and Redish, Janice, *A Practical Guide to Usability Testing, Revised Edition*, Intellect Books, Bristol, U.K. (1999).

Fetterman, D. M., *Ethnography: Step by Step, Second Edition*, Sage, Thousand Oaks, CA (1998).

Gould, John D. and Lewis, Clayton, Designing for usability: Key principles and what designers think, *Communications of the ACM* 28, 3 (March 1985), 300–311.

Harper, R., The organization of ethnography, *Proc. CSCW 2000*, ACM Press, New York (2000), 239–264.

Heim, Steven, *The Resonant Interface: HCI Foundations for Interaction Design*, Addison-Wesley, Reading, MA (2008).

Holtzblatt, Karen and Beyer, Hugh, CDTools, InContext Enterprises, Inc., http://www.incontextdesign.com/cdtools/index.html (2008).

Holtzblatt, Karen, Wendell, Jessamyn Burns, and Wood, Shelley, *Rapid Contextual Design: A How-To Guide to Key Techniques for User-Centered Design*, Morgan Kaufmann, San Francisco, CA (2005).

Hughes, J., O'Brien, J., Rodden, T., and Rouncefield, M., Design with ethnography: A presentation framework for design, *Proc. ACM Symposium on Designing Interactive Systems*, ACM Press, New York (1997), 147–159.

Ives, Blake and Olson, Margrethe H., User involvement and MIS success: A review of research, *Management Science* 30, 5 (May 1984), 586–603.

Jones, Capers, A CAI State of the Practice Interview, Computer Aid, Inc., http://web.ecs.baylor.edu/faculty/grabow/Fall2007/COMMON/Secure/Refs/capersjonesinterview1.pdf (July 2005)

Karat, Claire-Marie, A business case approach to usability, in Bias, Randolph and Mayhew, Deborah (Editors), *Cost-Justifying Usability*, Academic Press, New York (1994), 45–70.

Kujala, Sari, User involvement: A review of the benefits and challenges, *Behaviour & Information Technology* 22, 1 (2003), 1–16.

Landauer, Thomas K., *The Trouble with Computers: Usefulness, Usability, and Productivity*, MIT Press, Cambridge, MA (1995).

Lessig, Lawrence, *The Future of Ideas: The Fate of the Commons in a Connected World*, Random House, New York (2001).

Lessig, Lawrence, *Code and Other Laws of Cyberspace, Version 2.0*, Basic Books, New York (2006).

Leventhal, Laura and Barnes, Julie, *Usability Engineering: Process, Products, and Examples*, Prentice Hall, Upper Saddle River, NJ (2007).

Marcus, Aaron, Return on investment for usable user-interface design: Examples and statistics (2002). Available at http://www.amanda.com/resources/ROI/AMA_ROIWhitePaper_28Feb02.pdf.

Marcus, Aaron, A practical set of culture dimensions for global user-interface development (2005). Available at http://www.amanda.com/resources/articles_f.html.

Mayhew, Deborah J., *The Usability Engineering Lifecycle: A Practitioner's Guide to User Interface Design*, Morgan Kaufmann, San Francisco, CA (1999).

Microsoft, Inc., *Microsoft's Windows Vista User Experience Guidelines* (2008). Available at http://www.procontext.com/en/guidelines/style-guides.html#microsoft.

Millen, David, Rapid ethnography: Time deepening strategies for HCI field research, *Proc. ACM Symposium on Designing Interactive Systems*, ACM Press, New York (2000), 280–286.

Muller, Michael, Retrospective on a year of participatory design using the PICTIVE technique, *Proc. CHI '92: Human Factors in Computing Systems*, ACM Press, New York (1992), 455–462.

Muller, Michael, Participatory design, in Jacko, Julie and Sears, Andrew (Editors), *The Human-Computer Interaction Handbook*, Lawrence Erlbaum Associates, Hillsdale, NJ (2003), 1051–1068.

Muller, M., Wildman, D., and White, E., Taxonomy of PD practices: A brief practitioner's guide, *Communications of the ACM* 36, 4 (1993), 26–27.

Myers, Brad, Hudson, Scott E., and Pausch, Randy, Past, present and future of user interface software tools, in Carroll, John M. (Editor), *HCI in the New Millennium*, ACM Press, New York (2001), 213–233.

National Research Council, Committee on Human Factors, Committee on Human-System Design Support for Changing Technology, Pew, Richard W. and Mavor, Anne S. (Editors), *Human-System Integration in the System Development Process: A New Look*, National Academies Press, Washington, DC (2007).

NetCoalition, http://www.NetCoalition.com/ (2008).

Nielsen, Jakob, *Designing Web Usability: The Practice of Simplicity*, New Riders, Indianapolis, IN (1999).

Nielsen, Jakob, Usability ROI declining, but still strong (2008). Available at http://www.useit.com/alertbox/roi.html.

Perfetti, Christine, Building and managing a successful user experience team (2006). Available at http://www.uie.com/events/uiconf/2006/articles/bloomer_ wolfe_interview/.

Pfleeger, Shari Lawrence and Atlee, Joanne, *Software Engineering: Theory and Practice, Fourth Edition*, Prentice-Hall, Englewood Cliffs, NJ (2009).

Postman, Neil, *Technopoly: The Surrender of Culture to Technology*, Vintage Books, New York (1993).

Ramachandran, Divya, Kam, Matthew, Chiu, Jane, Canny, John, and Frankel, James F., Social dynamics of early stage co-design in developing regions, *Proc. Conference on Human Factors in Computing Systems (ACM CHI 2007)*, ACM Press, New York (2007), 1087–1096.

Rose, Anne, Plaisant, Catherine, and Shneiderman, Ben, Using ethnographic methods in user interface re-engineering, *Proc. ACM Symposium on Designing Interactive Systems*, ACM Press, New York (1995), 115–122.

Rosson, Mary Beth, *Usability Engineering: Scenario-Based Development of Human Computer Interaction*, Morgan Kaufmann, San Francisco, CA (2002).

Rosson, Mary Beth and Carroll, John M., *Usability Engineering: Scenario-Based Development of Human Computer Interaction*, Morgan Kaufmann, San Francisco, CA (2002).

Rosson, Mary Beth and Carroll, John M., Dimensions of participation in information system design, in Zhang, Ping and Galletta, Dennis F. (Editors), *Human-computer Interaction and Management Information Systems: Applications*, M.E. Sharpe, Armonk, NY (2006), 337–352.

Rubin, Jeffrey and Chisnell, Dana, *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests, Second Edition*, John Wiley & Sons, New York (2008).

Samuelson, Pamela, Digital rights management {and, or, vs.} the law, *Communications of the ACM* 46, 4 (April 2003), 41–45.

Samuelson, Pamela and Schultz, Jason, Should copyright owners have to give notice about their use of technical protection measures?, *Journal of Telecommunications & High Technology Law* 6 (2007), 41–76. Republication forthcoming in *Digital Rights Management Technologies* (ICFAI 2008).

Samuelson, Pamela, Intellectual property for an information age: How to balance the public interest, traditional legal principles, and the emerging digital reality, *Communications of the ACM* 44, 2 (February 2001), 67–68.

Sears, Andrew and Jacko, Julie A., *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, Second Edition*, CRC Press, Boca Raton, FL (2008).

Selby, Richard W. (Editor), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management, and Research*, John Wiley & Sons, NewYork (2007), 663–685.

Sharp, Helen, Rogers, Yvonne, and Preece, Jenny, *Interaction Design: Beyond Human-Computer Interaction, Second Edition*, John Wiley & Sons, New York (2007).

Shneiderman, Ben and Rose, Anne, Social impact statements: Engaging public participation in information technology design, *Proc. CQL '96: ACM SIGCAS Symposium on Computers and the Quality of Life*, ACM Press, New York (1996), 90–96.

Sommerville, Ian, *Software Engineering, Eighth Edition*, Addison-Wesley, Reading, MA (2006).

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

Sun Microsystems, Inc., *Java Look and Feel Design Guidelines, Second Edition*, http://java.sun.com/products/jlf/ed2/book/ (2001).

Usability Professionals Association, Usability body of knowledge, http://www.usabilitybok.org/ (2008).

Web Developers Journal, http://webdevelopersjournal.com/software/webtools.html (2008).

Whitworth, Brian and De Moore, Aldo (Editors), *Handbook of Research on Socio-Technical Design and Social Networking Systems*, IGI Global, Hershey, PA (2009).