Human Factors Engineering

James R. Lewis
IBM Software Group, Boca Raton, Florida, U.S.A.

Abstract
Human factors engineering (also known as ergonomics) is a body of knowledge and a collection of analytical and empirical methods used to guide the development of systems to ensure suitability for human use. Human factors engineering is a component of user-centered design, and encompasses the disciplines of human–computer interaction and usability engineering. The focus of human factors engineering is on understanding the capabilities and limitations of human abilities, and applying that understanding to the design of human–machine systems. This entry introduces the general topics of human factors engineering and usability testing.

INTRODUCTION

Much of the code written for software systems involves the design of fully automated interactions among modules, with no human–computer interaction. For many projects, however, there are points at which humans will interact with the code, either controlling the system with inputs or interpreting system outputs. Thus, producing good software designs for these points of human–computer interaction requires an understanding of the relative strengths and weaknesses of humans and computers. Failing to design for human use can cause a variety of problems, from user frustration and reduced system performance[1] to far more serious consequences, including death.[2]

This entry introduces human factors engineering, also referred to as ergonomics, or HF/E—the discipline that focuses on understanding the capabilities and limitations of human abilities, and applying that understanding to the design of human–machine systems. As a discipline, HF/E lies within the higher-level discipline of user-centered design (UCD) and encompasses the lower-level disciplines of human–computer interaction (HCI) and usability engineering (UE) (see Fig. 1). For definitions and descriptions of UCD and HCI, see the entries on these topics in this encyclopedia (see *Human-Centered Design*, p. 395 and *User-Centered Design*, p. 1308).

The main topics discussed in this entry are the following:

- What human factors engineering/ergonomics is
- The history of human factors engineering
- Fundamentals of human factors engineering
  - — Anthropometry
  - — Human motor control and input devices
  - — Human perception and output devices
  - — Information processing psychology and interaction design
  - — Usability testing
- Conclusion

WHAT IS HUMAN FACTORS ENGINEERING/ ERGONOMICS?

The definition of HF/E, developed by the International Ergonomics Association (IEA, http://www.iea.cc) and adopted by the Human Factors and Ergonomics Society (HFES, http://www.hfes.org), is:

> Ergonomics (or human factors) is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design to optimize human well-being and overall system performance. Ergonomists (or human factors engineers) contribute to the design and evaluation of tasks, jobs, products, environments and systems to make them compatible with the needs, abilities and limitations of people.[3]

There are many other definitions of HF/E, but this definition most appropriately conveys the breadth of the field and its systems approach, especially its scientific nature and the inclusion of both research (theory) and design (application).

The term *ergonomics* comes from the Greek *ergon* for "work" and *nomos* for "laws"—thus, the science of work. Originally, it primarily connoted physical considerations for the design of equipment and tasks. *Engineering psychology* was most associated with mental considerations,

Encyclopedia of Software Engineering DOI: 10.1081/E-ESE-120044161

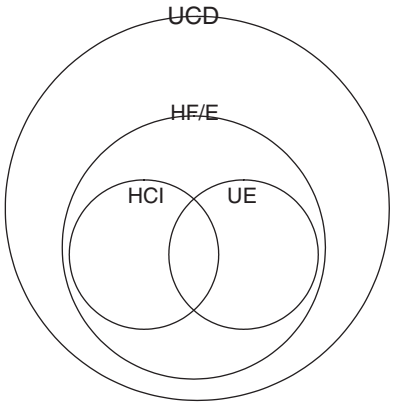

Fig. 1 Relationships among UCD, HF/E, HCI, and UE.

and is the term still used by the American Psychological Association for their Division 21 (see http://www.apa.org/divisions/div21/). *Human factors*, from the beginning, included both physical and mental aspects of the design of equipment and systems that interface with humans, and for decades was the preferred term in the United States, with *ergonomics* being preferred in Europe. The IEA has broadened the use of the term *ergonomics* beyond the traditional *physical ergonomics* to include *cognitive ergonomics* and *organizational ergonomics* (IEA, http://www.iea.cc). In recognition of the increasing international use of the term *ergonomics*, the Human Factors Society changed its name to the Human Factors and Ergonomics Society in 1993.

HISTORY OF HUMAN FACTORS ENGINEERING

Stone Age implements are the earliest examples of ergonomics or human factors applied to the design of artifacts. One of the earliest publications cited as an example of human factors design is Hippocrates' description of the design of a surgeon's workplace in the fifth century B.C.E.[4] Other early roots of human factors include the "scientific management" of Fredrick Taylor and the time-and-motion studies of Frank and Lillian Gilbreth in the late nineteenth and early twentieth centuries. Human factors engineering also owes many of its empirical methods to those developed by experimental psychologists and applied statisticians in the late nineteenth and early twentieth centuries, with contributions from those disciplines continuing today.

Most histories of modern human factors engineering trace its origin to World War II.[5,6] The new, complex machinery of the war placed significant demands on operators, who had to deal with not only new machinery, but also inconsistent designs among similar machines, such as the controls in aircraft cockpits. One of the most famous examples of human factors engineering in World War II was the identification of a problem in the shape of the control handles in airplanes.

A common cause of P-47, B-17, and B-25 airplane crashes was due to the "pilot error" associated with retracting the wheels instead of the flaps during landing. Alphonse Chapanis, one of the fathers of human factors engineering, noted that in these aircraft the wheels and flaps operations used the same type of control—either a toggle switch or a lever—placed right next to one another. In another type of aircraft, the C-47, these controls were different and not in close proximity, and the wheels-up landing error never happened. Chapanis realized that the "pilot error" in the P-47, B-17, and B-25 aircraft was actually a design problem. The solution was to put shape-coded rubber fittings over the controls, with a wheel shape for the wheels and a wedge shape for the flaps—a coding strategy that has persisted into modern aircraft.[6]

The success of human factors engineering during World War II led to its continuation in the military, industry, and academia following the war, with increased emphasis on systems design. Application areas included transportation, consumer products, energy systems (especially, the design of nuclear power plant controls following the Three Mile Island incident), medical devices, manufacturing, office automation, organizational design and management, aging, farming, sports and recreation, mining, forensics, education, and, of particular interest to software engineers, the design of computer hardware and software.

Before the mid-1970s, only experts used the very expensive computers of the time. In the late 1970s and early 1980s, the introduction of the personal computer brought computer usage to the masses, with a corresponding explosion in the number of human factors engineers in industry and academia focused on improving the usability of computer systems, both hardware and software. It was during this time that the disciplines of HCI (see *Human-Centered Design*, p. 395), UCD (see *User-Centered Design*, p. 1308), and UE[7,8] emerged.

FUNDAMENTALS OF HUMAN FACTORS ENGINEERING

One of the fundamental tenets of human factors engineering is to design equipment and systems around human capabilities, both physical and cognitive. To do that, it is necessary to have an understanding of the operating characteristics of humans. Thus, human factors engineering draws upon such diverse disciplines as anthropometrics, biomechanics, industrial engineering, experimental psychology, cognitive psychology, social psychology, and organizational psychology, and applies them to the physical design of equipment, such as input and output devices, and to interaction design. Human factors engineering can

play a role in software engineering from the definition of requirements through the assessment of a deployed system.

The following sections provide a very high-level discussion of anthropometry, input methods, output methods, interaction design, and usability testing for software systems. For more details, refer to the "Bibliography" section.

Anthropometry

One of the first steps in the development of any equipment is to ensure that its physical design matches well with the design of the human body. Anthropometry is the branch of anthropology that deals with comparative measurements of the human body, used in engineering to achieve the maximum benefit and capability of products intended for human use.[9] Although the gross dimensions of the human body have not changed in thousands of years, designers must consider the specific population of intended users. For example, the anthropometry of small children is different from that of adults, and men differ from women. Even within specific populations, it is important to take into account biological variability. It is not sufficient to design for the average person. The smallest and largest users in the target population must be able to grasp, view, and manipulate the equipment. Typically, designs should accommodate the 5th percentile female and the 95th percentile male. Practitioners can find much design-relevant information in published tables of anthropometric dimensions.

Human Motor Control and Input Devices

To design input devices, it is important to understand human capabilities of vision, touch, proprioception, motor control, and hearing.[10] For example, in keyboard design, the lettering on keys and associated controls must be legible, the keys must provide appropriate touch and proprioceptive feedback, and if any auditory feedback occurs, the user must be able to hear it. In modern computer systems, the most common types of input devices are keyboards, buttons, mice, touchpads, touchscreens, pointing sticks, handwriting recognition, and speech recognition. Less common input devices are for eye tracking, detection of tilt and acceleration, and the detection of emotion, for example, the detection of galvanic skin response through sensors on a mouse. Software engineers need to know how different input devices will interact with software for different conditions of human use. For example, although the mouse is a widely used pointing device, it is unsuitable for use in environments such as on a shipboard or in space, or where there will be insufficient space for its use, such as when flying on an airplane, especially in coach.

One of the most widely applied laws from engineering psychology with regard to the design and evaluation of pointing devices is Fitts' law: $MT = a + b \log_2(ID)$, where MT is "movement time," ID is the "index of difficulty" (which is twice the movement amplitude—the distance from the starting point to the center of a target—divided by the target width), and a and b are constants (typically estimated from data using linear regression).[10] The equation says that the farther the distance to a target and the smaller it is, the more time it will take to acquire it. This is consistent with common sense, but Fitts' law goes beyond common sense by showing that the relationship movement time and the index of difficulty is a log-linear function. Studies based on Fitts' law systematically vary the distance to and the size of targets that users must select with a pointing device. These studies are part of the empirical foundation indicating that the mouse is a nearly optimal pointing device because, relative to other pointing devices, it tends to have a lower value for the slope (b). The typical interpretation of this finding is that it indicates better overall efficiency for target selection over a range of target distances and sizes.

Paul M. Fitts, the developer of Fitts' law, was the head of the Psychology Branch of the U.S. Army Airforce Aeromedical Laboratory at its founding in July 1945 until 1949, when he established an engineering psychology laboratory at Ohio State University. In addition to his influential law, he also investigated the principle of stimulus–response compatibility. Some stimulus–response mappings are more natural than others for humans. It would not affect the computer to have lateral movement of the mouse mapped to vertical movement of the pointer, but users would definitely care. On the other hand, users seem to have no trouble with the vertical mapping of the pointer to forward and backward mouse movements.[10]

Another famous law from the engineering psychology of the 1950s is the Hick–Hyman law: $RT = a + bH_T$ (where RT is reaction time, a is a constant for the basic processing time associated with any decision, b is the amount of increase in RT as the number of choices increases—H_T is $\log_2 N$ for N equally likely alternatives).[10] Like Fitts' law, there is an impressive body of research supporting the Hick–Hyman law. Stimulus–response compatibility can affect the slope of the function, with greater compatibility leading to a shallower slope, which also decreases as a function of practice. For modeling the effect of practice, the power function $RT = A + BN^{-\beta}$ works well for mean data from groups of subjects (where RT is reaction time, A is the asymptotic RT, B the performance time on the first trial, N the number of trials, and β the learning rate). For individual data sets, the exponential function $RT = A + Be^{-\alpha N}$ provides a better fit (where α is the rate parameter).[10]

Fitts' law and the Hick–Hyman law have their origin in the 1950s, but have continuing applicability to certain aspects of software engineering. For example, Soukoreff and MacKenzie[11] published a very influential paper on the theoretical upper and lower bounds on typing speed using a stylus and a soft keyboard. Their model used the Hick–Hyman law for choice reaction time to model novice input, Fitts' law for rapid aimed movements to

model expert input, and linguistic tables of English digrams (letter pairs), including the space bar, to model basic text entry. Their model predicted a soft QWERTY typing rate of 8.9 wpm for novices and 30.1 wpm for experts, a finding consistent with the observation that typing with soft keyboards is slower than typing with a standard keyboard, even when the soft keyboard is the same size as a standard physical keyboard. This model has enabled researchers to investigate non-QWERTY layouts for stylus typing, and in some cases has been included in programs designed to search through key layout design spaces in search of optimal key arrangements for stylus typing.

Human Perception and Output Devices

Fundamentally, output devices in a system provide inputs to humans in the system through human sensory systems. The standard five human senses are vision, hearing, touch, smell, and taste. In addition to these, body awareness senses such as the sense of balance and awareness of limb location are also important. Each of these sensory systems has associated receptors that are sensitive to different aspects of the environment. For an output device to be effective, it must interface effectively with one or more of the human senses.[12]

The scientific psychological study of human senses, known as psychophysics, began in the mid-1800s with the work of Gustav Fechner, the father of classical threshold methods. These methods assume the existence of sensory thresholds—a minimum amount of stimulation required for human perception. In the pursuit of the measurement of thresholds, Fechner developed a number of psychophysical methods, but none of the methods succeeded in pinpointing precisely measured thresholds. Consequently, the concept of a threshold became a statistical concept—the intensity at which observers would detect the stimulus 50% of the time. Furthermore, Fechner found that factors that affected observer motivation had an effect on the values of the thresholds.[12]

Although classical threshold methods still have some utility in human factors engineering, the psychophysical methods of signal detection theory have largely replaced them.[12] A key advantage of signal detection studies is that the method produces two independent outcomes, a measure of observer sensitivity (d') and a measure of observer bias (β). In a signal detection study, a stimulus either is or is not present, and on each trial, the observer states that the stimulus either is or is not present. Thus, there are two ways for the observer to be right: when the stimulus is present and the observer says it is (i.e., a hit) and when the stimulus is not present and the observer says it is not (i.e., a correct rejection). There are two ways to be incorrect: the stimulus is present but the observer says it is not (i.e., a miss), or the stimulus is not present but the observer says that it is (i.e., a false alarm). With values for the hit rate and the false alarm rate converted to standard normal Z scores, it is possible to compute both d' and β.

Another goal of psychophysics is to describe mathematically the relation between the physical intensity of a stimulus and the intensity, or magnitude, of its perception. The most common approaches are methods based on difference thresholds, paired comparisons, and magnitude estimation.[12] For example, the power function $B = aI^{0.33}$ models the relationship between judgments of brightness and the physical intensity of light (where B is brightness, a is a constant, and I is the physical intensity). There are some situations in which other factors play a role, for example, exposures to a stimulus that last less than 100 m s^{-1}, or very small stimuli, but for most visual stimuli, the model works very well.

The most common type of output device in computer systems is the visual display, with increasingly greater use of liquid crystal display (LCD) over Cathode ray tube (CRT) technology. Clearly, the design and evaluation of these types of devices require a deep understanding of the human visual system. The next most common type of output is auditory, either multimedia sounds or speech, the design of which requires an understanding of human hearing and, in the case of speech, spoken language, especially when the voice is synthetic rather than a recorded human voice. The third type of output is tactile, for example, Braille displays, or vibration settings on mobile phones.

For humans, the visual modality is the sensory channel with the highest bandwidth. The receptors in the human eye are sensitive to light frequencies from 370 to 730 nanometers (nm). The cornea and lens focus light entering the eye. The cornea has fixed optical power; the flexible lens fine-tunes the focus. Beyond 6 m, the lens is fairly flat, but to focus on objects closer than that, muscles attached to the lens change its sphericity. As humans age, the lens becomes less flexible, making it difficult to focus on near objects. This is why the use of reading glasses becomes more prevalent after the 40th birthday, a phenomenon known as loss of accommodation, or presbyopia.[12]

The retina of the normal human eye contains two types of receptors: the rods, which are not sensitive to light frequencies, and the cones, of which there are three subtypes, each sensitive to a different range of frequencies, the basis for human color vision. Consequently, normal human color vision is trichromatic—"any spectral hue can be matched by a combination of three primary colors, one each from the short-, middle-, and long-wavelength positions of the spectrum" (Proctor and Proctor,[12] p. 68). Standardized color specification systems include the CIE (Commission Internationale de l'Éclairage) tristimulus coordinate system and the Munsell Book of Colors.

In computer displays, one use of color is to present a realistic representation of an image. Another use of color is to highlight, emphasize, and code information. When using colors for coding, it is important to keep in mind that about 10% of males have red-green color blindness—the

inability to distinguish between red and green due to the absence of one of the three types of cones. The prevalence of other types of color blindness, including total absence of color vision, is much less, about 1% of males. Even so, one consequence of the possibility of color blindness is that displays using color coding should avoid contrasting use of red and green, and should also use some redundant coding scheme, for example, shape coding.[12]

Visual displays can range from simple indicator lights, such as a low battery indicator, to large flat panel displays. They can use reflective or emissive technologies. Regardless of the technology, for displays of text, the text must be legible. Legibility is a function of several factors, especially text size, font design, and contrast ratio. Software designers need to know the size and resolution of the displays of the devices on which their software will run, especially for non-standard platforms such as PDAs and cell phones.

Human hearing is sensitive to sound waves, the changes in ambient air pressure. The most common measure of sound intensity is the decibel (dB). The power function $L = aI^{0.6}$ models the relation between the intensity of a sound and its perceived loudness (where L is loudness, a is a constant, and I is the physical intensity of the sound). The human ear is most sensitive to frequencies between 500 and 3000 Hz (Hertz, or cycles per second). For users to hear an auditory signal, it must be 40–50 dB above ambient background noise; and, if less than 300 ms in duration, it may have to be even louder.[12]

An advantage of an auditory signal is that it provides an eyes-free alert to users that something, for example, an incoming call, requires their attention. Simple tones do not provide much information, but complex tones, such as ring tones assigned to specific callers, can provide a significant amount of information. For an audio display to be effective, users must be able to hear and discriminate among the different audio signals presented by a device, and they must be able to disable audio output easily.

The least common type of display is the tactile display, although there is continuing interest in the research and design of tactile displays.[13] As with auditory displays, tactile displays are good for eyes-free alerts (e.g., for use in place of an auditory ringer on a cell phone) and can vary from very simple (e.g., vibrating or not vibrating) to somewhat more complex (e.g., distinctly different vibratory patterns). The human ability to detect subtle tactile differences is, however, less refined than the ability to detect subtle visual or auditory differences, especially the ability to reliably distinguish different frequencies of vibration. Some touch receptors in the skin are rapid adapting (sensitive to the start and ending of a touch) and others are slow adapting (sensitive for the duration of a touch). Different body sites have different sensitivities, ranging from 0.4 to 1000 Hz. Except for very small contact areas, the optimal sensitivity for vibration at all body sites is about 150–300 Hz. At both lower and higher frequencies, it is necessary to increase the amplitude of the vibration to ensure detection. User interface designers have used patterns of frequency, duration, intensity, and location to create complex patterns of vibration for tactile displays. The most recent review of research in this area indicates that coding based on location and duration is more effective than coding that uses frequency and intensity.

Information Processing Psychology and Interaction Design

Interaction design requires knowledge of human capabilities in input and output activities plus the tasks that humans will do with the system and the environments in which humans will use the system. Every designed artifact, such as a software program, implies a set of beliefs, even if unintentional, about human factors. Those artifacts that are most consistent with true human capabilities will be the most usable.

One of the most important human factors activities in support of interaction design is task analysis. To perform task analysis, it is necessary to specify who will carry out the task, what the contents of the task are, and why the user must perform the task. Task analysis has historical roots in the time-and-motion studies of the early twentieth century,[5,6] but for decades has also included analysis of the cognitive aspects of tasks, such as identification of the points in task performance that threaten to overload human working memory capabilities.

Information processing psychology is the study of aspects of human cognition such as attention and memory.[14] The findings of information processing psychology are of particular interest to researchers in and practitioners of the design of human–computer interaction. When humans work with systems, "transformations must be made on the information as it flows through the human operator. These transformations take time and may be the source of error. Understanding their nature, their time demands, and the kinds of errors that result from their operation is critical to predicting and modeling human–system interaction" (Wickens and Carswell,[14] p. 111).

The primary topics of information processing psychology are action selection, perception, attention, memory and cognition, and multiple task performance.[14] In information processing frameworks, humans first sense and attend to incoming stimuli, transformations in the sensory system result in perception, which then connects with information in human long-term memory, followed by action selection or the temporary storage of information in working memory, eventually resulting in decision and response selection, then response execution. For information about action selection (motor control), see the "Human Motor Control and Input Devices" section, and for the topic of perception, see the "Human Perception and Output Devices" section.

Attention can be selective, focused, or divided. Four factors influence selective attention: salience, expectancy,

value, and effort. Salient events are those that tend to capture attention. For example, auditory events are more salient than visual events. This is one reason that sounds are so useful for alerts but, if misused, can negatively affect human performance. Human knowledge about the likely time and location of events drives expectancy, and humans are faster to respond to expected events. Different events have different values, and humans consider this when allocating selective attention. When monitoring the environment for events, humans are more likely to engage in low-effort than in high-effort activities. When people make an effort to keep their attention on a desired source of information, they engage in focused attention. Alternatively, when they must attend to more than one source of information, the task requires divided attention. As you can imagine, divided attention places the greatest demand on human cognition. One application of the research in human attention capabilities is to try to make valuable information sources as salient as possible and to minimize the effort required to use valuable and expected sources.[14]

One of the best-known limitations of human ability is that of working memory, sometimes referred to as short-term memory or the short-term store.[14] Perhaps the most famous "law" of cognitive psychology applied to human factors is "the magic number seven plus or minus two." The psychologist George Miller coined the phrase in a paper (published in 1956) in which he pointed out that a variety of experimental results indicated that humans have trouble holding more than five to nine items (chunks) simultaneously in working memory. To hold information in working memory for more than a few seconds, humans must rehearse (repeat) the information, usually doing so silently or subvocally. This has been an extremely valuable observation with wide applicability to the cognitive analysis of tasks, but it has also seen wide misuse. For a brief time in the 1980s, creators of visual menus limited their designs in accordance with the "law," despite the fact that visual search places very little demand on working memory. More recently, research has shown that the application of the "law" to the design of auditory menus in interactive voice response systems is also misguided.[15]

Some appropriate applications based on the current understanding of the limitations of working memory are the following:[14]

- Do not require users to hold more information in working memory than necessary, and then only for as short a period of time as possible
- Parse longer alphanumeric strings into three- or four-item units to enhance chunking of information
- Do not require users to perform several actions before allowing them to dump the contents of working memory
- Do not require users to update working memory too rapidly
- Design tasks so they avoid interference with working memory, keeping in mind that interference is greatest when the interfering material is similar in sight or sound to the information currently in working memory

Many jobs require the allocation of attention to more than one source of information and the performance of more than one task at a time. Analysis of this situation has revealed different modes of multiple-task behavior, one of strict serial processing (task performance is not concurrent), and two types of concurrent—perfect parallel processing (concurrent performance is equal to separate performance) and degraded concurrent processing (performance on one or both tasks is poorer when done concurrently).[14]

In strict serial processing, it is not possible to do the tasks at the same time, so the user must switch attention from one task to the other. One challenge in this mode is for the user to spend the optimal amount of time in each of the tasks. Relying on human memory to manage this typically results in suboptimal allocation, so some systems provide reminders to switch tasks. It is important, however, to design reminders to switch to a secondary task so the reminders are not likely to interfere with performance on the primary task (see the last point in the above list for applications of research on working memory).[14]

For concurrent processing, the goal is to achieve perfect parallel processing. Success in concurrent processing is a function of task similarity, task demand, and task structure. The more similar the tasks are with respect to perceptual signals, items held in working memory, or similarity of controls, the more difficult it will be to perform them concurrently. Other aspects of similarity, however, can make the tasks easier to perform concurrently, such as the use of similar rules to map events to actions. Users will be more successful with concurrent performance of easy rather than difficult tasks. In general, easier tasks are those with which the user has significant practice or experience—in other words, with which the user has attained automaticity, attention-free performance. Tasks that promote ease of learning have a consistent relationship among displayed elements, cognitive operations, and responses. Tasks that place their demands on different cognitive resources are easier to perform concurrently. The multiple resources are "processing code (verbal or linguistic vs. spatial), processing stage (perceptual–cognitive operations vs. response operations), perceptual modality (auditory vs. visual), and visual subsystems (focal vision, required for object recognition, vs. ambient vision, required for orientation and locomotion)" (Wickens and Carswell,[14] p. 141).

Usability Testing

Because human factors engineering is an empirical as well as a theoretical discipline, it is not enough to use human factors principles in design—it is also necessary to engage

in test and evaluation of the system. Drawing from the discipline of experimental psychology, human factors tests must address three elements that affect the generalizability of the test: the representativeness of human participants, the representativeness of variables, and the representativeness of settings and environments. Human factors methods for test and evaluation include designed experiments, observational methods, surveys, questionnaires, and audits. Each method has its strengths and weaknesses, so it is common to take a multimethod approach to human factors test and evaluation. By far, the most common type of human factors testing in software engineering is usability testing.[8]

What is usability testing?

For software development, one of the most important tests and evaluation innovations of the past 30 years is usability testing.[8] The term "usability" first appeared in 1979, and came into general use in the early 1980s. Related terms from that time were "user friendliness" and "ease of use," which "usability" has since displaced in professional and technical writing on the topic. Recent surveys of usability practitioners indicate that it is a frequent activity in commercial software development, second only to iterative design, with which it has a close relationship.

As a concept, usability is not a property of a person or thing. No instrument exists that provides an absolute measurement of the usability of a product. Usability is an emergent property that depends on the interactions among users, products, tasks, and environments. It is important to keep in mind that practitioners use the term "usability" in two related but distinct ways. In one usage, the primary focus of usability is on measurements related to the accomplishment of global task goals, known as summative, or measurement-based, evaluation. In the other usage, known as formative, or diagnostic, evaluation, the focus is on the detection and elimination of usability problems.[8]

During a usability test, one or more observers watch one or more participants perform specified tasks with the product in a specified test environment. Thus, usability testing is different from other UCD methods. People who are being interviewed or are participating in a focus group do not perform work-like tasks. Expert evaluations, heuristic evaluations, and other usability inspection methods do not include the observation of users or potential users performing work-like tasks, which is also true of techniques such as surveys and card sorting. Field studies, which include ethnographic and contextual inquiry studies, can involve the observation of users performing work-related tasks in target environments, but restrict the control that practitioners have over the target tasks and environments. This is not necessarily a bad thing, but is a defining difference between usability testing and field studies. For a recent survey of the different usability evaluation methods, see Dumas and Salzman.[16]

Within the bounds of usability testing, the tests can vary widely in formality. Observers and participants might sit next to one another, be in a laboratory separated by a one-way glass, or be half a world apart using Internet technologies to enable the observation. Usability tests can be think-aloud tests, in which observers train participants to talk about what they're doing at each step of a task with prompting as required to continue talking, or simply observational tests. A test might require participants to work alone, in pairs, or in teams. The software under test might be a low-fidelity prototype, a high-fidelity prototype, a product under development, a predecessor product, or a competitive product.[8]

History of usability testing

The roots of usability testing lie firmly in the experimental methods of cognitive and applied psychology and human factors engineering, with strong ties to the concept of iterative design. Traditional experimentation requires a plan that includes the exact number of participants that the experimenter will expose to different experimental treatments. For results to be generalizable, participants must be representative members of the population under study. Before the experiment begins, the experimenter provides instructions, and later debriefs the participant, but at no time during an experimental session does the experimenter interact with the participant, unless such interaction is part of the experimental treatment. Summative (measurement-based) usability studies are usually indistinguishable from standard psychological experiments. Formative (diagnostic) usability studies are much less like a traditional experiment, although the requirements for sampling from a legitimate population of users, tasks, and environments still apply. Many of the other principles of psychological experimentation that protect experimenters from threats to reliability and validity, such as the control of demand characteristics, are also part of any usability test.[8]

Just as Alphonse Chapanis was one of the fathers of human factors engineering, he was also one of the fathers of usability testing, both on his own and through his students (e.g., Janan Al-Awar and Jeff Kelley), with influential publications in the early 1980s. At the same time, usability testing began to take place in software companies such as IBM (with influential publications by John Gould and his associates at the IBM T.J. Watson Research Center), Xerox, and Apple. In particular, the 1982 ACM Computer–Human Interaction (CHI) conference was a key event in the history of usability testing.[8]

In contrast to the goal of scientific psychology to develop and test competing theoretical hypotheses, iterative usability testing satisfies the need to modify early product designs as quickly as possible, using the results

of usability tests to guide the redesign. When it becomes apparent that users are encountering difficulties during the early stages of iterative design as revealed by usability testing, there is little point in asking participants to continue performing tasks. Concerns about ethical treatment of human participants and economic concerns argue against putting participants into situations where they will encounter known error-producing situations. Furthermore, delays in updating the product delays the potential discovery of problems associated with the update or problems previously blocked by the presence of the known flaws. All of these considerations support the principle of rapid iterative test and redesign, especially early in the development cycle.[8]

Fundamentals of usability testing

Although there are many potential measurements associated with usability tests, in practice, and as recommended in the 2001 ANSI Common Industry Format for Usability Test Reports, the fundamental global measurements for usability tasks are as follows:

- Successful task completion rates for a measure of effectiveness
- Mean task completion times for a measure of efficiency
- Mean participant satisfaction ratings, collected on a task-by-task basis, at the end of a test session, or both

For summative usability tests, the goals should have an objective basis and there should be shared acceptance across the various stakeholders. Data from previous comparable usability studies or pilot tests provide the best objective basis for measurement goals: similar types of participants completing the same tasks under the same conditions. If this information is not available, an alternative is to recommend objective goals based on general experience and to negotiate with the other stakeholders to arrive at a set of shared goals.[8]

After establishing usability goals, the next step is to collect the data as representative participants perform the target tasks in the specified environment. Even when the focus of the test is on measurement, test observers typically record the details about the specific usability problems they observe. The usability test team provides information about goal achievement and prioritized problems to the development team, and as data come in, the team determines whether the product has met its objectives. The ideal stopping rule for measurement-based iterations is to continue testing until the product has met its goals, although realistic constraints of time and money may also affect the stopping rule.[8]

With regard to the appropriate number of iterations, it is better to run one usability test than not to run any at all. On the other hand, the real power of usability testing requires an iterative product development process. Ideally, usability testing should begin early and occur repeatedly throughout the development cycle. It is a common practice to run at least three usability studies during a project: 1) an exploratory usability test on prototypes at the beginning of a project, 2) a usability test on an early version of the product during the later part of functional testing, and 3) a usability test during system testing. Once the final version of the product is available, it can be useful to run an additional usability test focused on the measurement of usability performance benchmarks—not to change the final version of the product, but as early input to a planned follow-on product.[8]

After completing the test, the next step is to analyze data and report the results. Summative studies typically report the standard task-level measurements such as completion rates, completion times, and satisfaction, and usually also include a list of the observed problems. Formative reports normally provide only a list of problems.[8]

The best way to describe usability problems depends on the purpose of the descriptions. For usability practitioners working in a software engineering environment, the goal should be to describe problems in ways that logically lead to one or more potential interventions, expressed as recommendations. Ideally, the problem description should also include some indication of the importance of fixing the problem, most often referred to as problem severity. There are varieties of problem classification schemes that researchers have developed, but software developers rarely care about these levels of description (e.g., slips vs. mistakes; or skill- vs., rule- vs., knowledge-based errors). They just want to know what they need to do to make things better while also managing the cost (both monetary and time) of interventions against their likely benefits.[8]

There is no rote procedure for the development of recommendations from problem descriptions—it is more of a craft. Good problem descriptions will often strongly imply one or more interventions. For each problem, it is reasonable to start, if possible, with interventions that would eliminate the problem. If problem-eliminating interventions are expensive enough to make them difficult to implement, the report should include other less costly interventions that would reduce the problem's severity. When different interventions involve different trade-offs, it is important for the recommendations to communicate this clearly.[8]

Because usability tests can reveal more problems than available resources can address, it is important to prioritize problems. The data-driven approach to prioritization takes into account problem-level data such as frequency of occurrence, impact on the participant, and the likelihood of using the portion of the product that was in use when the problem occurred. The usual method for measuring the frequency of occurrence of a problem is to divide the number of occurrences within participants by the number of participants. A common impact assessment method is to

assign impact scores according to whether the problem: 1) prevents task completion, 2) causes a significant delay or frustration, 3) has a relatively minor effect on task performance, or 4) is a suggestion. The combination of the frequency and impact data produces an overall severity measurement for each problem.[8]

The most common use of quantitative measurements is the characterization of performance and preference variables by computing means, standard deviations, and, ideally, confidence intervals for the comparison of observed to target measurements when targets are available. When targets are not available, the results can still be informative, for example, for use as future target measurements or as relatively gross diagnostic indicators.[8]

The failure to meet targets is an obvious diagnostic cue. A less obvious cue is an unusually large standard deviation. Because the means and standard deviations of time scores tend to correlate, one way to detect an unusually large variance is to compute the coefficient of variance by dividing the standard deviation by the mean, or the normalized performance ratio by dividing the mean by the standard deviation. Large coefficients of variation or, correspondingly, small normalized performance ratios, can indicate the presence of usability problems.[8]

Sample size estimation for usability tests

When the cost of a test sample is low, sample size estimation has little importance. In usability testing, however, the cost of a sample, specifically, the cost of running an additional participant through the tasks, is relatively high, so sample size estimation is an important usability test activity.[8]

When the focus of a usability test is on task-level measurements, sample size estimation is relatively straightforward, using statistical techniques that have been available since the early twentieth century and, in some cases, even earlier. These methods require an estimate of the variance of the dependent measure(s) of interest and an idea of how precise the measurement must be. Precision is a function of the magnitude of the desired minimum critical difference and statistical confidence level. There are numerous sources for information on standard sample size estimation,[17] or for methods applied to usability testing.[8] The past 15 years have seen a growing understanding of appropriate sample size estimation methods for problem-discovery (formative) testing.

Probably the best-known formula for modeling the discovery of usability problems is $P = 1 - (1 - p)^n$, where P is the likelihood of a problem occurring at least once given an overall probability of occurrence of p and a sample size of n. The derivation of this formula follows from the binomial probability formula, in which $(1 - p)^n$ is the probability that a problem that has p probability of occurrence will *not* occur given a sample size of n, so $1 - (1 - p)^n$ is the probability of the problem occurring at least once. For formative usability tests, the goal is to discover problems that can occur, so the criterion of occurring at least once is the most liberal possible criterion and, in practice, seems to be the best for sample size estimation. Several independent Monte Carlo studies have shown that $P = 1 - (1 - p)^n$ provides an excellent model of problem discovery.[8]

To convert this formula to a form more suitable for sample size estimation, it is possible to redefine P as *Goal* (where *Goal* is the target proportion of problem discovery) and solve for n, which results in $n = \log(1 - Goal)/\log(1 - p)$.

In this version of the formula, the value of p should be an estimate of the mean value of p across all of the discoverable problems in the study, which is a function of the specific types of participants, specific tasks, and specific environments used in the test. For standard formative usability tests, the literature contains large-sample examples that show p ranging from 0.16 to 0.42. For planning purposes, estimates of p can come from previous similar tests or from a pilot test. Regardless of the source, recent Monte Carlo studies have shown that small-sample estimates of p have a large built-in inflation bias. Fortunately, it is possible to deflate small-sample estimates of p using the formula $p_{adj} = ½[(p_{est} - 1/n)(1 - 1/n)] + ½[p_{est}/(1 + GT_{adj})]$, where GT_{adj} is the Good–Turing[18] adjustment to probability space (which is the proportion of the number of problems that occurred once divided by the total number of different problems). The $p_{est}/(1 + GT_{adj})$ component in the equation produces the Good–Turing adjusted estimate of p by dividing the observed unadjusted estimate of p (p_{est}) by the Good–Turing adjustment to probability space. The $(p_{est} - 1/n)(1 - 1/n)$ component in the equation produces the normalized estimate of p from the observed unadjusted estimate of p and n (the sample size used to estimate p). The reason for averaging these two different estimates is that the Good–Turing estimator tends to overestimate the true value of p, and the normalization tends to underestimate it. For more information and a full computational example, see Lewis.[8] For an online calculator, see http://www.measuringusability.com/problem_ discovery.php.

Another approach to sample size estimation for formative usability tests is to use $n = \log(1 - Goal)/\log(1 - p)$ to create a table in the format of Table 1.

To use Table 1 to estimate a required sample size, it is necessary to balance problem occurrence probability, the cumulative likelihood of detecting the problem at least once, and the available resources.[8] Smaller target problem occurrence probabilities require larger sample sizes, for example, to have a 50% chance of seeing problems that will occur only 1% of the time would require a sample size of 68 participants. Larger problem discovery likelihoods require larger sample sizes, for example, to have a 99% chance of seeing problems that will occur 25% of the time requires a sample size of 15 participants.

Table 1 Sample size requirements for problem discovery (formative) studies.

Problem occurrence probability	Cumulative likelihood of detecting the problem at least once					
	0.50	0.75	0.85	0.90	0.95	0.99
0.01	68	136	186	225	289	418
0.05	14	27	37	44	57	82
0.10	7	13	18	22	28	40
0.15	5	9	12	14	18	26
0.25	3	5	7	8	11	15
0.50	1	2	3	4	5	7
0.90	1	1	1	1	2	2

Table 1 clearly shows why it is unrealistic to design a usability test with the goal of having a 99% chance of seeing problems that will occur only 1% of the time, as that would require a sample size of 418 participants. A more realistic use of the table would be to establish a goal of having a 90% chance of seeing problems that will occur 25% of the time, which requires a sample size of eight participants. Another consideration is whether there will be only one usability study, in which case the goals should be more ambitious and the sample correspondingly larger, or whether there will be a series of usability tests as part of iterative development, in which case the goals for each individual study should be consistent with smaller sample sizes.

An alternative use of the table is to evaluate what a usability study is capable of detecting given a specific sample size. For example, suppose a practitioner has limited resources and can only run six participants through a usability study. Examination of Table 1 indicates that even though a sample size this small has its limitations, it is worth running because:

- The study will almost certainly detect problems that have a 90% likelihood of occurrence (the cumulative likelihood is 99% with only two participants).
- There is a high probability (between 95% and 99%) of detecting problems that have a 50% likelihood of occurrence.
- There is a reasonable chance (about 80% likely) of detecting problems that have a 25% likelihood of occurrence.
- The odds are about even for detecting problems with a 15% likelihood of occurrence (the required sample size at 50% is five).
- The odds are a little less than even for detecting problems with a 10% likelihood of occurrence (the required sample size at 50% is seven).
- The odds are low that the study will detect problems that have a likelihood of occurrence of 5% or 1% (the required sample sizes at 50% are 14 and 68, respectively).

CONCLUSION

Any artifact, including software artifacts, intended for human use benefits from the application of human factors engineering and usability testing, which are components of user-centered design methodologies in a software engineering context. This entry has provided an introduction to human factors engineering and usability testing, but it is not a substitute for relevant academic training, HF/E experience, and certification. In addition to providing a high-level background in these topics, this entry demonstrates the need for trained human factors engineers and usability specialists on development teams that produce software with which humans will interact.

REFERENCES

1. Norman, D.A. *The Design of Everyday Things*; Basic Books: New York, NY, 2002.
2. Casey, S.M. *Set Phasers on Stun*; Aegean: Santa Barbara, CA, 1998.
3. Human Factors and Ergonomics Society, Definitions of HF/E, http://www.hfes.org/Web/EducationalResources/HFEdefinitionsmain.html (accessed April 2008).
4. Marmaras, N.; Poulakakis, G.; Papakostopoulos, V. Ergonomic design in ancient Greece. Appl. Ergon. **1999**, *30* (4), 361–368.
5. Human Factors and Ergonomics Society, HFES History, http://www.hfes.org/web/AboutHFES/history.html (accessed April 2008).
6. Roscoe, S. N. The adolescence of engineering psychology. In *The Human Factors History Monograph Series*; Casey, S. M., Ed.; Human Factors and Ergonomics Society: Santa Monica, CA, 1997; Vol. 1, 1–9, http://www.hfes.org/Web/PubPages/adolescencehtml.html.
7. Nielsen, J. *Usability Engineering*; Academic Press: Boston, MA, 1993.
8. Lewis, J. R. Usability testing. In *Handbook of Human Factors and Ergonomics*; G. Salvendy, Ed.; John Wiley: Hoboken, NJ, 2006; 1275–1316.
9. Robinette, K. M.; Hudson, J. A. Anthropometry. In *Handbook of Human Factors and Ergonomics*; G. Salvendy, Ed.; John Wiley: Hoboken, NJ, 2006; 322–339.

10. Proctor, R. W.; Vu, K-P. L. Selection and control of action. In *Handbook of Human Factors and Ergonomics*; G. Salvendy, Ed.; John Wiley: Hoboken, NJ, 2006; 89–110.
11. Soukoreff, R. W.; MacKenzie, I. S. Theoretical upper and lower bounds on typing speed using a stylus and soft keyboard. Behav. Inf. Technol. **1995**, *14*, 370–379.
12. Proctor, R. W.; Proctor, J. D. Sensation and perception. In *Handbook of Human Factors and Ergonomics*; G. Salvendy, Ed.; John Wiley: Hoboken, NJ, 2006; 53–88.
13. Jones, L. A.; Sarter, N. B. Tactile displays: Guidance for their design and application. Hum. Factors **2008**, *50* (1), 90–111.
14. Wickens, C. D.; Carswell, C. M. Information processing. In *Handbook of Human Factors and Ergonomics*; G. Salvendy, Ed.; John Wiley: Hoboken, NJ, 2006; 111–149.
15. Commarford, P. M.; Lewis, J. R.; Smither, J. A.; Gentzler, M. D. A comparison of broad versus deep auditory menu structures. Hum. Factors **2008**, *50* (1), 77–89.
16. Dumas, J. S.; Salzman, M. C. Usability assessment methods. In *Reviews of Human Factors and Ergonomics, Vol. 2*; R. C. Williges, Ed.; Human Factors and Ergonomics Society: Santa Monica, CA, 2006; 109–140.
17. Diamond, W. J. *Practical Experiment Designs for Engineers and Scientists*; Lifetime Learning Publications: Belmont, CA, 1981.
18. Manning, C. D.; Schutze, H. *Foundations of Statistical Natural Language Processing*. MIT Press: Cambridge, MA, 1999.

BIBLIOGRAPHY

In addition to the related entries in this encyclopedia, other sources for information on human factors and usability are books, scientific journals, trade journals, professional conferences, and standards.

Books

According to a survey conducted by the Human Factors and Ergonomics Society (Human Factors and Ergonomics Society, Recommended Textbooks, http://www.hfes.org/Web/Educational Resources/textbooksmain. html (accessed April 1, 2008)), the most frequently used books for college courses in introductory human factors and ergonomics, in reported frequency order, are:

1. Wickens, C. D.; Lee, J.; Liu, Y. D.; Gordon-Becker, S. *An Introduction to Human Factors Engineering*, 2nd Ed.; Prentice Hall: Upper Saddle River, NJ, 2004.
2. Sanders, M. S.; McCormick, E. J. *Human Factors in Engineering and Design*, 7th Ed.; McGraw-Hill: New York, NY, 1993.
3. Casey, S. M. *Set Phasers on Stun*; Aegean: Santa Barbara, CA, 1998.
4. Chaffin, D. B.; Andersson, G. B. J.; Martin, B. J. *Occupational Biomechanics*, 4th Ed.; Wiley-Intersciences: New York, NY, 2006.
5. Stanton, N.; Hedge, A.; Brookhuis, K.; Salas, E.; Eds. *Handbook of Human Factors and Ergonomics Methods*; CRC Press: Boca Raton, FL, 2004.
6. Wilson, J. R.; Corlett, E. N.; Eds. *Evaluation of Human Work: A Practical Ergonomics Methodology*, 3rd Ed.; Taylor & Francis: Philadelphia, PA, 2005.
7. Norman, D. A. *The Design of Everyday Things*; Basic Books: New York, NY, 2002.

Other valuable books and encyclopedias, some current, others of historical interest, are listed below:

1. Bailey, R. W. *Human Performance Engineering: Using Human Factors/Ergonomics to Achieve Computer System Usability*; Prentice Hall: Englewood Cliffs, NJ, 1989.
2. Bias, R. G.; Mayhew, D. J. *Cost-Justifying Usability: An Update for the Internet Age*; Morgan Kaufmann: San Francisco, CA, 2005.
3. Dumas, J.; Redish, J. C. *A Practical Guide to Usability Testing*; Intellect: Portland, OR, 1999.
4. Helander, M. G.; Landauer, T. K.; Prabhu, P. V.; Eds. *Handbook of Human-Computer Interaction*, 2nd Ed.; North-Holland: Amsterdam, 1997.
5. Karwowski, W.; Ed. *International Encyclopedia of Ergonomics and Human Factors*, 2nd Ed.; CRC Press: Boca Raton, FL, 2006.
6. Mayhew, D. J. *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design*; Academic Press: San Diego, CA, 1999.
7. Norman, D.; Draper, S.; Eds. *User Centered System Design: New Perspectives on Human-Computer Interaction*; Lawrence Erlbaum: Hillsdale, NJ, 1986.
8. Rasmussen, J. *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*; Elsevier: New York, NY, 1986.
9. Rubin, J. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*; John Wiley: New York, NY, 1994.
10. Salvendy, G.; Ed. *Handbook of Human Factors and Ergonomics*; 3rd Ed.; John Wiley: Hoboken, NJ, 2006.
11. Sears, A.; Jacko, J. A.; Eds. *The Human-Computer Interaction Handbook*, 2nd Ed.; CRC Press: Boca Raton, FL, 2007.
12. Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*; Addison-Wesley: Reading, MA, 1987.
13. Wickens, C. D.; Hollands, J. G. *Engineering Psychology and Human Performance*, 3rd Ed.; Prentice-Hall: Upper Saddle River, NJ, 1999.

Scientific Journals

The major scientific journals that publish information on human factors engineering, including those with an emphasis on software engineering and human–computer interaction, are listed below:

ACM SIGCHI Bulletin
ACM Transactions on Computer-Human Interaction
ACM Transactions on Information Systems
Applied Ergonomics
Behaviour & Information Technology
Behavior Research Methods, Instruments, & Computers
Communications of the ACM
Computers in Human Behavior

Ergonomics
Human Factors
Human–Computer Interaction
IBM Systems Journal
IEEE Transactions on Professional Communication
Interacting with Computers
International Journal of Human–Computer Interaction
International Journal of Human–Computer Studies
International Journal of Industrial Ergonomics
International Journal of Man–Machine Studies
International Journal of Speech Technology
Journal of Applied Psychology
Journal of Biomechanics
Journal of Cognitive Engineering and Decision Making
Journal of Usability Testing
Personal and Ubiquitous Computing

Trade Magazines

The major professional societies publish trade magazines periodically. The major trade magazines and their associated organizations are listed below:

Ergonomics in Design (*HFES*)
Interactions (*ACM*)
User Experience (*UPA*)

Professional Conferences

The major conferences that have human factors and usability evaluation as a significant portion of their content are listed below:

ACM Special Interest Group in Computer–Human Interaction (see http://www.acm.org/sigchi/)

Human–Computer Interaction International (see http://www.hc-international.org/)

Human Factors and Ergonomics Society Annual Meeting (see http://hfes.org/)

INTERACT (held every 2 years, for example, see http://www.interact2005.org/)

Usability Professionals Association (see http://www.upassoc.org/)

Standards

1. ANSI. *Common Industry Format for Usability Test Reports* (ANSI-NCITS 354-2001); Washington, DC, 2001.
2. ANSI. *Human Factors Engineering of Computer Workstations*: (ANSI/HFES 100-2007); Human Factors and Ergonomics Society: Santa Monica, CA, 2007.
3. ISO. *Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)* (ISO 9241:1998); Author: Geneva, Switzerland, 1998.