USABILITY ENGINEERING METHODS FOR SOFTWARE DEVELOPERS

The human–computer interaction community aims to increase the awareness and acceptance of established methods among software practitioners. Indeed, awareness of the basic usability methods will drive an Information Society for all.

By Andreas Holzinger

Usability is most often defined as the ease of use and acceptability of a system for a particular class of users carrying out specific tasks in a specific environment. Ease of use affects the users' performance and their satisfaction, while acceptability affects whether the product is used [1]. Thus, it is of great importance that every software practitioner not only be aware of various usability methods,

but be able to quickly determine which method is best suited to every situation in a software project.

One of the basic lessons we have learned in human-computer interaction (HCI) is that usability must be considered before prototyping takes place. There are techniques (such as usability context analysis) intended to facilitate such early focus and commitment [11]. When usability inspection, or testing, is first carried out at the end of the design cycle, changes to the interface can be costly and difficult to implement, which in turn leads to usability recommendations. These are often ignored by developers who feel, "We don't have usability problems." The earlier critical design flaws are detected, the more likely they can be corrected. Thus, user interface design should more properly be called user interface development, analogous to software development, since design usually focuses on the synthesis stages, and user interface components include metaphors, mental models, navigation, interaction, appearance, and usability [6].

It is generally accepted that the following five essential usability characteristics should be part of any software project: *learnability,* so the user can rapidly begin working with the system; *efficiency,* enabling a user who has learned the system to attain a high level of productivity; *memorability,* allowing the casual user to return to the system after a period of non-use without having to relearn everything; *low error rate,* so users make fewer and easily rectifiable errors while using the system, and no catastrophic errors occur; and *satisfaction,* making the system pleasant to use. There are trade-offs among these criteria, and some are more important than others, although this ranking depends on the situation. For example, long-term efficiency may be sufficiently important for developers to be willing to sacrifice rapid learnability [10].

To ensure a software project has these essential usability characteristics, we use methods we divide into *inspection methods* (without end users) and *test methods* (with end users). The accompanying figure details these characteristics.

Usability Inspection Methods
This is a set of methods for identifying usability problems and improving the usability of an interface design by checking it against established standards.

	Inspection Methods			Test Methods		
	Heuristic Evaluation	Cognitive Walkthrough	Action Analysis	Thinking Aloud	Field Observation	Questionnaires
Applicably in Phase	all	all	design	design	final testing	all
Required Time	low	medium	high	high	medium	low
Needed Users	none	none	none	3+	20+	30+
Required Evaluators	3+	3+	1-2	1	1+	1
Required Equipment	low	low	low	high	medium	low
Required Expertise	medium	high	high	medium	high	low
Intrusive	no	no	no	yes	yes	no
Comparison of Usability Evaluation Techniques						

Comparison of usability evaluation techniques.

These methods include heuristic evaluation, cognitive walkthroughs, and action analysis.

Heuristic evaluation (HE) is the most common informal method. It involves having usability specialists judge whether each dialogue or other interactive element follows established usability principles [8]. The original approach is for each individual evaluator to inspect the interface alone. Only after all the evaluations have been completed are the evaluators allowed to communicate and aggregate their findings. This restriction is important in order to ensure independent and unbiased evaluations. During a single evaluation session, the evaluator goes through the interface several times, inspects the various interactive elements, and compares them with a list of recognized usability principles (for example, Nielsen's Usability Heuristics [7]). There are different versions of HE currently available; for example, some have a cooperative character. The heuristics must be carefully selected so they reflect the specific system being inspected, especially for Web-based services where additional heuristics become increasingly important. Usually 3–5 expert evaluators are necessary (increasing the cost of this technique); less-experienced people can perform an HE, but the results are not as good. However, HE using non-experts is appropriate at times, depending on who is available to participate.

Advantages include the application of recognized and accepted principles; intuitiveness; usability early in the development process; effective identification of major and minor problems; rapidity; and usability throughout the development process.

Disadvantages include separation from end users; inability to identify or allow for unknown users' needs; and unreliable domain-specific problem iden-

tification. Also, HE does not necessarily evaluate the complete design, since there is no mechanism to ensure the entire design is explored, and evaluators can focus too much on one section; and the validity of Nielsen's guidelines has been questioned [9].

A cognitive walkthrough (CW) is a task-oriented method by which the analyst explores the system's functionalities; that is, CW simulates step-by-step user behavior for a given task. CW emphasizes cognitive issues, such as learnability, by analyzing the mental processes required of the users. This can be achieved during the design by making the repertory of available actions salient, providing an obvious way to undo actions, and offering limited alternatives [5]. The background is derived from exploratory learning principles. Several versions of CW exist, including pluralistic walkthroughs wherein end users, software developers, and usability experts go through the system, discussing every single dialogue element.

Advantages include independence from end users and a fully functioning prototype, helping designers to take on a potential user's perspective, effective identification of problems arising from interaction with the system, and the ability to help to define users' goals and assumptions.

Disadvantages of CW include possible tediousness and the danger of an inherent bias due to improper task selection, emphasis on low-level details, and non-involvement of the end user.

The action analysis method is divided into formal and back-of-the-envelope action analysis; in both, the emphasis is more on what the practitioners do than on what they *say* they do. The formal method requires close inspection of the action sequences a user performs to complete a task. This is also called keystroke-level analysis [2]. It involves breaking the task into individual actions such as move-mouse-to-menu or type-on-the-keyboard and calculating the times needed to perform the actions. Back-of-the-envelope analysis is less detailed and gives less precise results, but it can be performed much faster. This involves a similar walkthrough of the actions a user will perform with regard to physical, cognitive, and perceptual loading. To understand this thoroughly we must keep in mind that goals are external, and we *achieve* goals. Tasks are those processes applied through some device in order to achieve the goals, and we *perform* tasks. Actions are tasks with no problem-solving and no internal control structure. We *do* actions. The main problem of task analysis [3] is the difficulty in accommodating complicated tasks completed by more than one individual. Furthermore, the representation of a task analysis is complex, even when a simple task is studied, and tends to become very unwieldy very

rapidly. Such representations can often only be interpreted by those who conducted the analysis.

Advantages include precise prediction of how long a task will take, and a deep insight into users' behavior.

Disadvantages of action analysis include it is very time-consuming and requires high expertise.

Usability Test Methods

Testing with end users is the most fundamental usability method and is in some sense indispensable. It provides direct information about how people use our systems and their exact problems with a specific interface. There are several methods for testing usability, the most common being thinking aloud, field observation, and questionnaires.

Thinking aloud (THA) [7] may be the single most valuable usability engineering method. It involves having an end user continuously thinking out loud while using the system. By verbalizing their thoughts, the test users enable us to understand how they view the system, which makes it easier to identify the end users' major misconceptions. By showing how users interpret each individual interface item, THA facilitates a direct understanding of which parts of the dialogue cause the most problems. In THA the time is very important, since the contents of the users' working memory contents are desired. Retrospective reports are much less useful, since they rely on the users' memory of what they had been thinking some time ago. A variant of THA called *constructive interaction* involves having two test users use a system together (co-discovery learning). The main advantage is that the test situation is much more natural than standard THA with single users working alone, since people are used to verbalizing their thoughts when trying to solve a problem together. Therefore, users may make more comments when engaged in constructive interaction than when simply thinking aloud for the benefit of an experimenter.

Advantages of THA include revealing *why* users do something; providing a close approximation to how individuals use the system in practice; provision of a wealth of data, which can be collected from a fairly small number of users; user comments of often contain vivid and explicit quotes; preference and performance information can be collected simultaneously; THA helps some users to focus and concentrate; and early clues can help to anticipate and trace the source of problems to avoid later misconceptions and confusion in the early stage of design.

Disadvantages include a failure to lend itself well to most types of performance measurement; the different learning style is often perceived as unnatural, dis-

tracting, and strenuous by the users; nonanalytical learners generally feel inhibited; and this method is time-consuming since briefing the end users is a necessary part of the preparation.

Causing users to focus and concentrate is both an advantage and a disadvantage since it results in less-than-natural interactions at times and causes THA to be faster due to the users' focus.

Field observation is the simplest of all methods. It involves visiting one or more users in their workplaces. Notes must be taken as unobtrusively as possible to avoid interfering with their work. Noise and disturbance can also lead to false results. Ideally, the observer should be virtually invisible to ensure normal working conditions. Sometimes video is used to make the observation process less obtrusive, but it is rarely necessary. Observation focuses on major usability catastrophes that tend to be so glaring they are obvious the first time they are observed and thus do not require repeated perusal of a recorded test session. Considering the time needed to analyze a videotape is approximately 10 times that of a user test, the time is better spent testing more subjects or testing more iterations of the design. Video is, however, appropriate in some situations. For example, a complete record of a series of user tests can be used to perform formal impact analysis of usability problems [4].

Another means of electronic observation is data logging, which involves statistics about the detailed use of a system. Data logging can provide extensive timing data, which is generally important in HCI and usability. Normally, logging is used to collect information about the field use of a system after release, but it can also be used as a supplementary method of collecting more detailed data during user testing. Typically, an interface log will contain statistics about the frequency with which each user has used each feature in the program and the frequency with which various events of interest (such as error messages) have occurred.

Many aspects of usability can best be studied by querying the users. This is especially true for issues related to the subjective satisfaction of the users and their possible anxieties, which are difficult to measure objectively. Questionnaires are useful for studying how end users use the system and their preferred features, but need some experience to design. They are an indirect method, since this technique does not study the actual user interface: it only collects the opinions of the users about the interface. One cannot always take user statements at face value. Data about people's actual behavior should have precedence over people's claims of what they think they do.

A simpler form of questionnaire is the interview.

The form of the interview can be adjusted to respond to the user and encourage elaboration.

Advantages include that subjective user preferences, satisfaction, and possible anxieties can be easily identified; and questionnaires can be used to compile statistics.

Disadvantages include that indirect methods result in low validity (discrepancies between subjective and objective user reactions must be taken into account); this method needs sufficient responses to be significant (we are of the opinion that 30 users is the lower limit for a study); and it identifies fewer problems than the other methods.

Usability inspection needs to be combined with usability test methods. For example, a cognitive walkthrough can be supplemented with a task-independent method, such as heuristic evaluation. Indirect usability tests, such as questionnaires or interviews, must be supplemented with direct usability tests; thinking aloud or observation would be suitable. An absolute must is understanding the user's task, culture, and capabilities; involving the users in the design early on; and testing and iterating, with or without users. **c**

REFERENCES
1. Bevan, N. Measuring usability as quality of use. *Softw. Quality J.* 4 (1995), 115–130.
2. Card, S.K., Moran, T.P. and Newell, A. *The Psychology of Human-Computer Interaction.* Erlbaum, Hillsdale, NJ, 1983.
3. Carroll, J.M. Making use is more than a matter of task analysis. *Interact. Comput. 14,* 5 (2002), 619–627.
4. Holzinger, A. Application of rapid prototyping to the user interface development for a virtual medical campus. *IEEE Softw. 21,* 1 (Jan. 2004).
5. Lewis, C. and Wharton, C. *Cognitive Walkthroughs. Handbook of Human-Computer Interaction, 2nd ed.* M. Helander, Ed. Elsevier, Amsterdam, 1997, 717–732.
6. Marcus, A. Dare we define user-interface design? *interactions 9,* 5 (2002), 19–24.
7. Nielsen, J. *Usability Engineering.* Morgan Kaufmann, San Francisco, 1994.
8. Nielsen, J. and Mack, R.L. *Usability Inspection Methods.* Wiley, New York, 1994.
9. Sears, A.L. Heuristic walkthroughs: Finding problems without the noise. *Int. J. Human-Comput. Interact. 9,* 3 (1997), 213–234.
10. Shneiderman, B. *Designing the User Interface, 3rd ed..* Addison-Wesley, Reading, MA, 1997.
11. Thomas, C. and Bevan, N. (Eds.). *Usability Context Analysis: A Practical Guide.* National Physical Laboratory, Teddington, UK, 1996.

ANDREAS HOLZINGER (andreas.holzinger@uni-graz.at) is an associate professor of information processing at Graz University, Austria, and is a member of the European Research Consortium for Informatics and Mathematics (ERCIM) Working Group "User Interfaces for All" (UI4ALL).