CHAPTER 5

QUALITY USER INTERFACE

5.1 Introduction
Interactive systems can be defined as the class of systems whose operations involve a significant degree of user interaction. The most important aspect of modern interactive computer systems is the level of support they provide for the underlying human activity. This level of support is encompassed in the user interface with which the user interacts with the system. Today, the importance of understanding the intended users when designing interactive computer systems is widely acknowledged, but as recently as 1980, it received little emphasis.

5.1.1 Importance of User Interface
In most major systems development companies, the basic organizational structures and development processes were defined in an earlier era, when the dialogue between computer systems and computer users did not have to be considered. Until 15 years ago, most
computer system users were engineers and programmers. Developers were designing systems for their own use or for other technically proficient users. They felt no need to seek "user participation" in design. Now, however, computer use has spread to workplaces that are very unlike engineering laboratories. To bridge the widening gulf between the developer and user environments requires greater effort. Early proponents of greater user involvement included both human factors specialists and systems developers.

An IBM usability research group stressed "an early focus on users" in the 1970s and in an influential 1983 paper recommended that "typical users (e.g., bank tellers, as opposed to a `group of expert' supervisors, industrial engineers, programmers) become part of the design team from the very outset when their perspectives can have the most influence, rather than using them to `review,' `sign off on,' `agree' to the design before it is coded" [1]. Perhaps coincidentally, this appeared when popular books such as In Search of Excellence were urging industry to cater to "the customer." A user focus was further emphasized in User Centered System Design, an influential 1986 collection of research papers [2]. Also in the 1970s and 1980s, Europeans experimented with greater user involvement in systems design. In particular, the Scandinavian "participatory design" approach, in which users collaborate as full development team members, has recently attracted attention [3].
5.1.2 Effective Interactive Systems

The challenge of designing better interactive systems has not gone unnoticed in software engineering. Boehm observes that the dominant waterfall model of development "does not work well for many classes of software, particularly interactive end-user applications"[4]. His proposal, a "spiral model" of software development, incorporates user involvement, prototyping, and iterative design. Yourdon recently wrote, "The first, and by far the most important, player in the systems game is someone known to systems analysts as a user"[5]. The spiral model is not yet widely used and as Boehm notes, it may be difficult to apply in some contexts. Similarly, Yourdon's observation has not been fully translated into practice.

The software methods that are employed widely today were developed before interactive end-user applications became important. They do not provide for an early and continual focus on users quite the contrary. Traditional structured analysis relegates the task "establish man-machine interface" to one sub-phase of system development [6].

5.2 Factors influencing interactive systems development

- The size of the Development Company or organization.

A start-up or a small product development company may have fewer resources, less division of labor, fewer installed customer base
concerns, and may succeed with far fewer sales than a larger company. These factors permit greater flexibility, more latitude for (inexpensive) innovation, and particularly significant for user involvement, the possibility of focusing on a few potential customers. In these ways, a small product development company takes on some characteristics of in-house or custom development. At the other end of the size continuum, very large companies also present mixed appearances. A large telecommunications company, for example, blends substantial internal development with opportunities for product development provided by its large, identifiable market. Similarly, in a highly divisionalized company, one division may "sell" its product to another or even bid for an internal contract to deliver a system.

- The charter of the company or organization

Organizational charters vary, and a company may work in more than one paradigm. For example, large product development companies pursue government contracts for systems that can be built on or around their products. A separate "Federal Systems Division" may manage these projects, but influences often cross divisional boundaries. Paradigms also blur when a product development company that considers entry into a new market experiments by custom building a system for one customer to obtain domain expertise. The Scandinavian UTOPIA project (an acronym in the Scandinavian languages for "training, technology, and products from the quality of work perspective") did the reverse experiment: Methods from the in-
house/custom development paradigm were adapted to a product development effort [7]. An organizational charter shifts gradually when a company develops a system under contract to one user organization, and then decides to market it more broadly.

- **Organizational structures and procedures**

  Companies that do similar work do not necessarily divide responsibilities and meet obligations in the same way. While certain job descriptions and work procedures are widespread in the industry, companies of similar size and charter organize differently, distribute authority differently, and approach system development differently. Marketing divisions drive some companies, while engineering drives others. Some are hierarchic bureaucracies, others are strongly divisionalized into semiautonomous subunits, and some have experimented with almost ad-hoc approaches. An in-house project in a large, divisionalized company may be managed through a contract competed for by internal development groups. Even where high-level approaches to organizational structure and process are similar, small variations in structure and process can greatly affect individual development projects. Within a single company, such variation results from internal reorganizations.

- **The nature of the system user population**

  The generic term "user" masks a tremendous diversity of computer users and contexts of use. This diversity will continue to
increase even if progress in hardware development stopped today; the current technology would take decades to realize its potential. The number and heterogeneity of users is often a particular concern to in-house development. More generally, some users are "captive audiences," who use systems acquired for them, while others are discretionary users. At the extreme end of discretionary use, those who pay for their own systems become involved through their personal economic stake. The physical separation of developers from some or all users is often critical, as are barriers of class, culture, or language.

- **The degree of design novelty or uncertainty**

  The uncertainty that arises in designing for a new market or in utilizing a new medium increases the motivation for learning about users, but adversely affects the conditions for doing so. In a mature application area, experienced users are more easily found and mediator consultants, trade publications, empirical research are more reliable sources of information about users or about technical alternatives.

- **Commitments and agreements among the groups involved**

  Commitments vary in formality and scope, ranging from informal understandings between individuals within one organization to binding contracts among companies. They also vary in content, focus, and flexibility. The content can be restricted to technical aspects of the system or can include such commitments to the users as
organizational impact statements, installation, or training. Similarly, an agreement can be restricted to the system to be developed or can specify aspects of the development process, including techniques to facilitate user involvement, such as prototyping or scheduled reviews. Contracts vary in flexibility even a formal contract might include a level-of-effort provision or specify times at which its terms can be reconsidered, permitting design changes based on user involvement [8][9].

- **Societal conditions and change over time**

  Projects encounter dynamic influences that the development partners cannot directly control [9]. These influences include aspects of the labor market, economic considerations of supply and competition, available technology, formal standards, and legal requirements for safeguards or restrictions on technology use.

  Finally, the factors described in this section are subject to change within the life of a project; it is a rare project that enjoys static conditions from start to finish.

5.3 **Quality User Interface Design**

Many technological innovations rely upon User Interface Design to elevate their technical complexity to a usable product. Technology alone may not win user acceptance and subsequent marketability. The User Experience, or how the user experiences the end product, is the key to acceptance. And that is where User Interface Design enters the design process. While product
engineers focus on the technology, usability specialists focus on the user interface. For greatest efficiency and cost effectiveness, this working relationship should be maintained from the start of a project to its rollout. When applied to computer software, User Interface Design is also known as Human-Computer Interaction or HCI.

While people often think of Interface Design in terms of computers, it also refers to many products where the user interacts with controls or displays. Military aircraft, vehicles, airports, audio equipment, and computer peripherals, are a few products that extensively apply User Interface Design. Optimized User Interface Design requires a systematic approach to the design process. But, to ensure optimum performance, Usability Testing is required. This empirical testing permits naïve users to provide data about what does work as anticipated and what does not work. Only after the resulting repairs are made can a product be deemed to have a user optimized interface. The importance of good User Interface Design can be the difference between product acceptance and rejection in the market place. If end-users feel it is not easy to learn, not easy to use, or too cumbersome, an otherwise excellent product could fail. Good User Interface Design can make a product easy to understand and use, which results in greater user acceptance.

5.3.1 Quality User Interface

The objective of any User Interface developer should be to design and implement quality user interfaces. “Quality user interface” is defined as any user interface that is intuitive, easy to use and allows the user to
maximize their efficiency and effectiveness when using it. Gravin has given a different approach to quality i.e. user perceived quality which is a combination of product attributes which provide the greatest satisfaction to a specified user [8].

However, if different groups of users have different needs, then they may require different characteristics for a product to have quality, so that assessment of quality becomes dependent on the perception of the user. User-perceived quality is regarded as an intrinsically inaccurate judgment of product quality. For instance, Garvin observes that "Perceptions of quality can be as subjective as assessments of aesthetics". Usability embraces user-perceived quality by relating quality to the needs of the user of an interactive product.

*Quality of use* is the extent to which a product satisfies stated and implied needs when used under stated conditions. This moves the focus of quality from the product in isolation to the particular users of the product, the tasks and the context in which it is used. The quality of use is determined not only by the product, but also by the context in which it is used: the particular users, tasks and environments. The quality of use (measured as effectiveness, efficiency and satisfaction) is a result of the interaction between the user and product while carrying out a task in a technical, physical, social and organizational environment. Measures of quality of use can be used to evaluate the suitability of a product for use in a particular context. However the measures of quality of use also depend on the nature of the user, task and environment - they are a property of the whole "work system". Measures of quality of use can thus also be used to assess the
suitability of any other component of the context. For instance whether a particular user has the necessary training or skill to operate a product, which tasks a product should be used for, or whether changes in the physical environment improve quality of use. A product meets the requirements of the user if it is effective, efficient in use of time and resources, and satisfying, regardless of the specific attributes it possesses.

5.3.2 Human factor in Interface Design

i) **Limited short term memory**: People can instantaneously remember about seven items of information. If you present more than this, they are more liable to make mistakes.

ii) **People make mistakes**: When people make mistakes and systems go wrong, inappropriate alarms and messages can increase stress and hence the likelihood of more mistakes.

iii) **People are different**: People have a wide range of physical capabilities. Designers should not just design for their own capabilities.

iv) **People have different interaction preferences**: Some people like to have pictures and some prefer to have text.
5.4 Usability

Usability is a multidimensional construct that can be examined from various perspectives. It is also an elusive concept and is determined by the tasks, the users, the product, and the environment. We have seen in the literature that the term usability has been used broadly and means different things to different people. Usability is the capacity of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.

Usability is a general term that encompasses everything having to do with "ease of use". That is, how easily people can use any product's controls or displays such as a: tool, computer display, automobile, aircraft, etc. Usability also refers to the study of methods, measurement, and principles of a product's efficiency, elegance, and usefulness. In the computer industry, usability often refers to the ease of use in terms of the human-computer interaction. The clarity, intuitiveness, seamlessness, and elegance of an application or website interface design. The idea behind usability is to design products keeping the user in mind. Putting the user first in the design process results in greater efficiency, learning time, and satisfaction. The goal of optimized usability is to make a product easy to understand, easy to use, and easy to learn. The outcome of good usability is a greater likelihood of user acceptance. User acceptance is often the difference between a product's success or failure in the market place. Users can often reject a well engineered product with great functionality if they are unable to understand, learn, and easily use that product. Similarly the focus of the evaluation may be a complete computer system, the complete software, a specific software component, or a specific aspect of a software component.
Any relevant aspect of software quality may contribute to quality of use, but for interactive software ease of use is often a crucial issue. Quality of use thus provides a means of measuring the usability of a product. It is not meaningful to talk simply about the usability of a product, as usability is a function of the context in which the product is used. The characteristics of the context may be as important in determining usability as the characteristics of the product itself. Changing any relevant aspect of the context of use may change the usability of the product. For instance, the user interface may be improved by conforming to good dialogue design practices, or the fit between the user and the rest of the overall system may be improved through means such as selection and training of users or good task design. A product which is usable by trained users may be unusable by untrained users. Aspects of the working environment such as lighting, noise, or workstation design may also affect usability.

5.4.1 Benefits of Usable Software
Most computer software in use today is unnecessarily difficult to understand, hard to learn, and complicated to use. Difficult to use software wastes the user’s time, causes worry and frustration, and discourages further use of the software. Benefits to the employer include:

- Usable software increases productivity and reduces costs. Difficult to use software is time consuming to use, and not exploited to full advantage as the user may be discouraged from using advanced
features. Difficult to learn software also increases the cost of training and of subsequent support.

- Usable software increases employee satisfaction. Difficult to use software reduces motivation and may increase staff turnover.
- In Europe, employers have an obligation to meet the requirements of the Display Screen Equipment Directive which requires software in new workstations to be "easy to use" and to embody "the principles of software ergonomics" [10].

### 5.4.2 Measures of Usability

ISO 9241-11 explains how usability can be measured in terms of the degree of excellence in use: effectiveness (the extent to which the intended goals of use are achieved), efficiency (the resources that have to be expended to achieve the intended goals), and satisfaction (the extent to which the user finds the use of the product acceptable). ISO 9241-11 also emphasizes that usability is dependent on the context of use and that the level of usability achieved will depend on the specific circumstances in which a product is used.

The context of use consists of the users, tasks, equipment (hardware, software and materials), and the physical and social environments which may influence the usability of a product in a work system. Measures of user performance and satisfaction thus assess the overall work system, and, when a product is the focus of concern, these measures provide information about the usability of that product in the particular context of use provided by the rest of the work system. 
It is important to note that while this definition provides a practical way to measure usability, it is also measuring the consequences of other software quality characteristics such as the functionality, reliability and the efficiency of the computer system. Changes in these characteristics or other components of the work system, such as the amount of user training can also have an impact on user performance and satisfaction.

For this reason, early drafts of ISO 9241-11 also defined a broader concept: quality of use: the extent to which specified goals can be achieved with effectiveness, efficiency and satisfaction in a specified work system.

Figure 5.1: Quality of Use
Usability is the quality of use in context. The effectiveness, efficiency, safety, flexibility and satisfaction are the usability parameters with which specified users can achieve specified goals in specified environments. However the attributes that a product requires for usability depend on the nature of the user, task and environment. The quality of use of an overall system encompasses all factors which may influence use of a product in the real world, including organizational factors such as working practices and the location or appearance of a product, and individual differences between users such as those due to cultural factors and prejudice.

5.4.3 Usability Parameters

- Effectiveness:

Effectiveness refers to correctness, accuracy and completeness. Error-free completion of tasks is important in both business and consumer applications. Effectiveness is the accuracy and completeness with which users achieve certain goals. Indicators of effectiveness include quality of solution and error rates. Measures of effectiveness relate the goals or sub-goals of using the system to the accuracy and completeness with which these goals can be achieved. For example if the desired goal is to transcribe a 2-page document into a specified format, then accuracy could be specified or measured by the number of spelling mistakes and the number of deviations from the specified format, and
completeness by the number of words of the document transcribed divided by the number of words in the source document.

- **Efficiency:**
  Efficiency refers to the measure of resources expended. How quickly a user can perform work is critical for business productivity. *Efficiency* is the relation between (1) the accuracy and completeness with which users achieve certain goals and (2) the resources expended in achieving them. Indicators of efficiency include task completion time and learning time. Measures of efficiency relate the level of effectiveness achieved to the expenditure of resources. The resources may be mental or physical effort, which can be used to give measures of human efficiency, or time, which can be used to give a measure of temporal efficiency, or financial cost, which can be used to give a measure of economic efficiency.

- **Satisfaction:**
  Satisfaction refers to comfort and acceptability of use. Comfort refers to overall physiological and emotional responses to use of the system. Acceptability of use measure overall attitude towards the system or user’s perception of specific aspects. Satisfaction is the extent to which expectations are met. Satisfaction is a success factor for any
products with discretionary use; it’s essential for maintaining workforce motivation. Measures of satisfaction describe the perceived usability of the overall system by its users and the acceptability of the system to the people who use it and to other people affected by its use. Measures of satisfaction may relate to specific aspects of the system or may be measures of satisfaction with the overall system. Measures of satisfaction can provide a useful indication of the user’s perception of usability, even if it is not possible to obtain measures of effectiveness and efficiency. User satisfaction can be measured by the extent to which users have achieved their pragmatic and hedonic goals. ISO/IEC CD 25010.2 suggests the following types of measures:

• **Linkability**: the extent to which the user is satisfied with their perceived achievement of pragmatic goals, including acceptable perceived results of use and consequences of use.

• **Pleasure**: the extent to which the user is satisfied with their perceived achievement of hedonic goals of stimulation, identification, evocation and associated emotional responses [11][12].

• **Comfort**: the extent to which the user is satisfied with physical comfort.

• **Trust**: the extent to which the user is satisfied that the product will behave as intended.

Satisfaction is most often measured using a questionnaire.
• **Safety:**
Safety refers to acceptable levels of risk of harm to people, business, data, software, property or the environment in the intended contexts of use. Safety is concerned with the potential adverse consequences of not meeting the goals. Safety is defined in ISO/IEC CD 25010.3 as the degree of expected impact of harm to people, business, data, software, property or the environment in the intended contexts of use. While effectiveness and efficiency measure the positive benefits of productivity and goal achievement, the term safety is here interpreted in a broad way to measure the potential negative outcomes that could result from incomplete or incorrect output. For a consumer product or game, negative business consequences may not only be associated with poor performance, but also, for example, with a lack of pleasurable emotional reactions or of achievement of other hedonic goals.

• **Flexibility:**
Flexibility is the ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed [15]. It is also defined as the ability of a software application to deal with exceptions to the process model at
the runtime and to cope with periodic changes to the process model. Classic and contemporary literature in software design recognizes the central role of flexibility in software design and implementation. All interactive systems obviously need some type of interaction between the system and the user to achieve something meaningful. This interaction can be considered as a conversation between two persons. One important issue is to decide who has the control or initiative in this conversation. The ideal solution in the point of view of the user is to have freedom from the system imposed constraints. This means that the user must be able to abandon, suspend, or resume any activity at any point. This form of dialog between the system and the user is called user pre-emptive. The dialog is called system pre-emptive if the system has the initiative and asks the user for information, usually in the form of modal dialog boxes. When interactive systems are designed, the programmers have to be careful about choosing which type of dialog is going to be needed in order to achieve maximum flexibility. A good solution must combine both types of dialog since is a bad idea to give the user all the freedom to do whatever he wants to accomplish.
5.5 Framework for Quality User Interface Design

A fundamental reality of application development is that the user interface of any application is the system to the user. What user wants is for developers to build applications that meet their needs and that are easy to use. In software engineering community, the term usability has been more narrowly associated with user interface design. ISO/IEC 9126, developed separately as a software engineering standard, defined usability as one relatively independent contribution to software quality associated with the design and evaluation of the user interface and interaction. The importance of good User Interface Design can be the difference between product acceptance and rejection in the market place. If end-users feel it is not easy to learn, not easy to use, or too cumbersome, an otherwise excellent product could fail. Good User Interface Design can make a product easy to understand and use, which results in greater user acceptance. User Interface design is an iterative process involving close liaisons between users and designers.

The best way to ensure the improved usability is to use an orderly and well defined framework that is specifically geared to produce quality results. The ideal way to specify and measure usability would be to specify the features and attributes required to make a product usable and measure their effect on the implemented product. The problem with usability of user interface is that it is very difficult to specify what these features and attributes should be because the nature of the features and attributes required depends on the context in which the product is used. It is not meaningful to talk about the usability of the product, as usability is a function of the context in which the product is used. The characteristics of the context (users, task and environment) may be as important in determining usability as the
characteristics of the product itself. Changing any relevant aspect of the context of use may change the usability of the product. For instance, the user interface may be improved by conforming the good dialogue design practices or the fit between the user and the rest of the overall system may be improved through means such as selection and training of users or good task design. A product which is usable by trained users may be unusable by untrained users. Aspects of working environment such as lighting, noise, or workstation design may also affect usability.

Figure 5.2: Quality User Interface Design Framework
5.5.1 Analysis of Quality User Interface Design Framework

A new framework is introduced which defines a process for user interface design defining various attributes required for quality user interface which are related to both the user and the product. The sequence in which the activities are performed and the level of effort and detail that is appropriate varies depending on the design environment and the stage of the design process. In the proposed framework, the first step is to plan the process which is human centered which will be helpful to the user while using the interface. Human centered design is an approach to interactive system development that focuses specifically on making systems usable. In user centered design, all development proceeds with the user as the center of focus. In this process, the focus is on usability goals, user characteristics, environment, task and workflow in the design of an interface. After this, the context of use will be specified and the requirements for the application are determined.

Context of use is determined by identifying the people who will use the product, what they will use it for, and under what conditions they will use it. This requires detailed information about the intended users, their tasks, technical and environmental constraints. If the application is to be accepted by the stakeholders, it is imperative that they be involved from the very beginning. They should be sought out and polled as to what they consider their requirements for the application to be. The more broad a representative group of stakeholders is involved; the more broad the acceptance will be when it is delivered. This can be accomplished by holding stakeholders meeting to analyze the content and plan accordingly. The first task is to identify the stakeholders groups and select representatives to
participate in the design team. To gather requirements, stakeholders are interviewed to determine answers to questions like: what the underlying business problems are that this application should address, what benefits the application should provide and what the critical success factors are. Earlier versions and competitor systems are evaluated to identify usability problems and obtain measures of usability as an input to usability requirements.

Any user interface, no matter how well designed, won't be well received if its users feel disenfranchised from the design process. By establishing the scope of the project early, the expectations of the stakeholders will also be established early. Scenarios are one way of describing the requirements where typical episodes of use are described. Scenarios help the users to choose appropriate search terms and carry out search and request copies of relevant material. Scenarios may be related to ‘use cases’, which describe interactions at a technical level. Unlike use cases, however, scenarios can be understood by people who do not have any technical background. They are therefore suitable for use during participatory design activities. Scenarios are appropriate whenever you need to describe a system interaction from the user’s perspective. They are particularly useful when one needs to remove focus from the technology in order to open up design possibilities, or when one needs to ensure that technical or budgetary constraints do not override usability constraints without due consideration. Scenarios can help confine complexity to the technology layer, and prevent it from becoming manifest within the user interface.
After defining the requirements, a design for the application is produced. In this step, a prototype is created. The aim of prototyping is to allow users to gain direct experience with the interface. Without such direct experience, it is impossible to judge the usability of an interface. Prototyping can be done in two stages:

- Early in the process, paper prototypes may be used. Paper prototyping is an effective way of getting user reactions to a design proposal.
- The design is then refined and increasingly sophisticated automated prototypes are then developed.

5.5.2 Quality Parameters for Evaluation
This design solution for the interface has to be evaluated for quality. The overall objective is to achieve quality of use for both end users and the support users. The quality attributes like satisfaction, effectiveness and efficiency when combined together defines the functional feedback of the interface. The framework (Fig. 5.2) represents the parameters like efficiency, effectiveness, safety, satisfaction and flexibility under technical, physical and organizational environment in a context and the dependence of user and product on these parameters resulting in quality of use. Efficiency refers to the resources such as time, money or mental effort that have to be expended to achieve intended goals. Effectiveness is the extent to which the intended goals of use of the overall system are achieved. Satisfaction is the extent to which the user finds the overall system
acceptable. Satisfaction may further be subdivided into further attributes which represents satisfaction related to different goals, physical satisfaction and satisfaction with security. Safety is the acceptable levels of risk or harm to people, data, software, business in the intended context of use.

After the evaluation of design for quality is completed, if the quality achieved is the desired quality, the final quality design is produced. And if the desired quality is not achieved, the requirements are again checked and classified properly from a quality design. The framework defined provides a means of measuring the usability of the product by specifying various parameters of quality of use.

5.6 Empirical Validation

Evaluation is a necessary part of all phases of a development project. Evaluations must be made several times in order to obtain a basis for further development, improvements etc. Evaluation takes time and are resource consuming. This is why they are often not given enough priority, which can lead to usability problems. As evaluations are resource consuming, it is important to choose the correct methods for every specific purpose. Evaluations can be made at different points of time and of different reasons. We can especially differ between evaluations during the development process and evaluations of existing, complete systems.

*Important points to be considered during evaluation:*

- They can easily be traded off in a project, if there are problems with dead-lines etc. This must be avoided.
• There must be a strict plan for evaluations from project start. This plan must include what actions to take, based on the outcome from evaluations.

• Evaluations often involve the users. These must be given sufficient possibilities and resources to participate.

• There are problems to measure some usability parameters. Many usability aspects are of a more qualitative nature, and require appropriate methods.

There exist multiple methods of evaluating usability depending on available resources, evaluator experience, ability and preference, and the stage of development. In broad terms it is worth making the following distinctions between evaluation methods:

• Quantitative Evaluation (Empirical Validation)
• User-based Evaluation
• Expert-bases Evaluation

5.6.1 Quantitative Evaluation

Four different views on human computer interaction to measure interactive qualities currently are [16] [17]:

(1) The interaction-oriented view: usability quality is measured in terms of how the user interacts with the product ("usability testing"). This view is the most common one. All kinds of
usability testing with "real" users are subsumed in this category [18].

(2) The user-oriented view: usability quality is measured in terms of the mental effort and attitude of the user ("questionnaires" and "interviews").

(3) The product-oriented view: usability quality is measured in terms of the ergonomic attributes of the product itself (quantitative measures).

(4) The formal view: usability is formalized and simulated in terms of mental models (formal concepts). Karat describes formal methods in the context of "theory-based" evaluation [19].

The interactive qualities of user interfaces currently are quantified in the context of interaction-oriented view and user-oriented view. The evaluation of user interface quality can be done in a quantitative way by measuring the various quality parameters. Different types of user interfaces can be quantified and distinguished by the general concept of “interaction points”. Regarding to the interactive semantic of "interaction points" (IPs), different types of IPs must be discriminated [20]. An interactive system can be distinguished in a dialog manager and an application manager. So, we distinguish between dialog objects (DO, e.g. "window") and application objects (AO, e.g. "text document"), and dialog functions (DF, e.g. "open window") and application functions (AF, e.g. "insert section mark"). Each function $f \in \text{FS}$, that changes the state of an application object, is an application function. All other functions are dialog functions (e.g., window
operations like move, resize, and close). A dialog context (DC) is defined by all available objects and functions in the actual system state. If the set of available functions changes in the actual DC, then the system changes from one DC to another.

All dialog objects (functions, resp.) in the actual DC are perceptible (PD, PF) or hidden (HD,HF). Each interaction point (IP) is related to at least one interactive function. If both mapping function's $\delta$ and $\alpha$ are of the type 1:m (any), then the user interface is a command interface. If both mapping function's $\delta$ and $\alpha$ are of the type 1:1, then the user interface is a menu or direct manipulative interface where each $f \in FS$ is related to a perceptible structure PF. The perceptual structure (visible, audible, or tactile) of a function (PF) can be, e.g., an icon, menu option, command prompt, or other mouse sensitive areas.

Following are the three ratios to calculate Functional Feedback, Application Flexibility and Dialog Flexibility [21].

**Functional Feedback:** To activate the amount of “feedback” of an interface, a ratio is calculated: number of PFs ($\#PF = \#PDFIP + \#PAFIP$) divided by number of HF$s$ ($\#HF = \#HDFIP + \#HAFIP$) per dialog context. (HDFIP is the hidden function interaction point of a dialog manager. PDFIP is the perceptible representation of a HDFIP. HAFIP is the hidden functional interaction point of the application manager and PAFIP is the perceptible representation of a HAFIP). This ratio quantifies the average “amount of functional feedback” of the functional space. The number of all different dialog contexts is abbreviated as D.
Application Flexibility: To quantify the flexibility of application manager, we calculate the average number of HAFIPs per dialog context.

\[
D
FB = \frac{1}{D} \sum_{d=1}^{D} \left( \frac{\#PF_d}{\#HF_d} \right) \times 100\%
\]

Dialog Flexibility: To quantify the flexibility of the dialog manager, we calculate the average number of HDFIPs per dialog context.

\[
D
DFA = \frac{1}{D} \sum_{d=1}^{D} \left( \#HAFIP_d \right)
\]

\[
D
DFD = \frac{1}{D} \sum_{d=1}^{D} \left( \#HDFIP_d \right)
\]

To quantify the above defined measures, the following values were found from an application having an I/O interface with both dialog manager and application manager. \( D = 20 \), \( P(AFIP) = 360 \), \( P(DFIP) = 520 \), \( H(AFIP,DFIP) = 60 \)

Base on the above values, we calculate the functional feedback, application flexibility and the dialog flexibility.

\[
FB = \frac{1}{20} \left( \frac{360}{60} + \frac{520}{60} \right) \times 100\% = 73.3\%
\]

\[
DFA = \frac{1}{20} (360) = 18
\]

\[
DFA = \frac{1}{20} (520) = 26
\]
We interpret this result with the effect that flexibility must exceed a threshold to be effective (DFD, DFA>15) [22]. These calculated values of the attributes measure the interactive quality of the user interface.

5.6.2 User-based Evaluation

Testing an application with a sample of users performing a set of pre-determined tasks is generally considered to yield the most reliable and valid estimate of an application's usability. Performed either in a usability test laboratory or a field site, the aim of such a test is to examine the extent to which the application supports the intended users in their work. Tightly coupled to the operational approach to usability definition, the user-based approach draws heavily on the experimental design tradition of human factors psychology in employing task analysis, pre-determined dependent variables and, usually, quantitative analysis of performance supplemented with qualitative methods.

In a typical user-based evaluation, test subjects are asked to perform a set of tasks with the technology. Depending on the primary focus of the evaluator, the user’s success at completing the tasks and their speed of performance may be recorded. After the tasks are completed, users are often asked to provide data on likes and dislikes through a survey or interview, or may be asked to view with the evaluator part of their own performance on video and to describe in more detail their performance and perceptions of the
application. In this way, measures of effectiveness, efficiency and satisfaction can be derived, problems can be identified and redesign advice can be determined. In certain situations, concurrent verbal protocols might be solicited to shed light on user’s thought processes while interacting with the tool so that issues of comprehension and user cognition can be addressed. In a usability lab, the complete interaction is normally video recorded for subsequent analysis of transactions, navigation, problem handling etc. However more informal approaches are also possible. Some user-based tests are unstructured, involving the user and the evaluator jointly interacting with the system to gain agreement on what works and what is problematic with the design. Such participative approaches can be very useful for exploring interface options in the early stages of design where formal quantitative assessments might be premature.

5.6.3 Expert-based Evaluation

Expert-based methods refers to any form of usability evaluation which involves an HCI expert examining the application and estimating its likely usability for a given user population. In such cases, users are not employed and the basis for the evaluation lies in the interpretation and judgment of the evaluator. There is considerable interest in this form of evaluation since it can produce results faster and presumably cheaper than user-based tests. In HCI, two common expert-based usability evaluation
methods are Heuristic evaluation (e.g., Nielsen, 1994), and Cognitive Walkthrough. Both methods aim to provide evaluators with a structured method for examining and reporting problems with an interface.

The Heuristic method provides a simple list of design guidelines which the evaluator uses to examine the interface screen by screen and while following a typical path through a given task. The evaluator reports violations of the guidelines as likely user problems. In the Cognitive Walkthrough method, the evaluator first determines the exact sequence of correct task performance, and then estimates, on a screen by screen basis, the likely success or failure of the user in performing such a sequence. In both methods, the expert must make an informed guess of the likely reaction of users and explain why certain interface attributes are likely to cause users difficulties. These methods differ in their precise focus. Heuristic methods are based on design guidelines and ultimately reflect the expert's judgment of how well the interface conforms to good design practice. The Cognitive Walkthrough method concentrates more on the difficulties users may experience in learning to operate an application to perform a given task. In practice, usability evaluators tend to adapt and modify such methods to suit their purpose and many experts who perform such evaluations employ a hybrid form of the published methods.
5.7 Conclusion

The conventional assumption that quality is an attribute of a product is wrong, as the attributes required for quality will depend on how the product is used. The Quality User Interface Design Framework introduced defines a user interface design process defining various attributes required for quality user interface related to both user and the product. The reviews with users of their requirements and the design of the final product in an iterative manner makes the user an active part of the process and results in a higher level of system usability and user satisfaction. Quality is generally treated as a property of a product, thus the product view of quality seeks to identify those attributes which are designed into a product or evaluated to ensure quality. The evaluation process to quantify usability attributes and the interactive quality of user interface enhances the quality of user interface.
5.8 References


