CSC 406 Lecture Series (5)

Design Principles

Conceptual Model

A description of the proposed system in terms of a set of integrated ideas and concepts about what it should do, behave and look like, that will be understandable by the users in the manner intended.

To develop a conceptual model involves envisioning the proposed product, based on the user's needs and other requirements identified. To ensure that it is designed to be understandable in the manner intended requires doing iterative testing of the product as it is developed.

A key aspect of this design process is initially to decide what the user will be doing when carrying out their tasks. For example, will they be primarily searching for information, creating documents, communicating with other users, recording events, or some other activity? At this stage, the interaction mode that would best supports this need to be considered. For example, would allowing the users to browse be appropriate, or would allowing them to ask questions directly to the system in their native language be more affective? Decision about which kind of interaction style use (e.g., whether to use a menu-based system, speech inputs, commands) should be made in relation to the interaction mode. Thus, decision about which mode of interaction to support differ from those made about which style of interaction to have; the former being at a higher level of abstraction. The former are also concerned with determining the nature of the users' activities to support, while the later are concerned with the selection of specific kinds of interface.

Once a set of possible ways of interacting with interactive system has been identified, the design of the conceptual modal then needs to be thought through in term of actual concrete solution. This entail working out the behaviour of the inter face, the particular interaction style that will be used, and the "look and feel" of the interface. At this stage of "fleshing out," it is always a good idea to explore a number of possible designs and to assess the merits and problems of each one.

Design Principles

A number of design principles have been promoted. The best known are concerned with how to determine what users should see and do when carrying out their tasks using an interactive product. Here we briefly describe the most common ones

- Visibility
- Affordance
- Constraints
- Mapping
- Consistency
- Feedback

Visibility

The more visible functions are, the more likely users will be able to know what to do next. In contrast, when functions are "out of sight," it makes them more difficult to find and knows how to use. Donald Norman describes the controls of a car to emphasize this point. The controls for different operations are clearly visible (e.g., indicator, headlights, horn, hazard warning lights), indicating what can be done. The relationship between the way the controls have been positioned in the car and what they do makes it easy for the deriver to find the appropriate control for the task at hand.

Affordance

Affordance is a term used to refer to an attribute of an object that allows people to know how to use it. For example, a mouse button invites pushing by the way it is physically constrained in its plastic shell. At a very simple level, to afford means "to give a clue." When the affordances of a physical object are perceptually obvious it is easy to know how to interact with it. For example, a door handle affords pulling, a cup handle affords grasping, and a mouse button affords pushing. Norman introduced this concept in the late 80s in his discussion of the design of everyday objects. Since then, it has been much popularized, being what can be done to them. For example, graphical elements like button, icon, links, and scroll bars are talked about with respect to how to make it appear obvious how they should be used: icons should be designed to afford clicking, scroll bars to afford moving up and down, buttons to afford pushing.

There are two kind of affordance:

• Perceived

• Real

Real

Physical objects are said to have real affordances, like grasping, that are perceptually obvious and do not have to be learned.

Perceived

User interfaces that are screen-based are virtual and do not make sense to try to design for real affordances at the interface---except when designing physical devices, like control consoles, where affordance like pulling and pressing are helpful in guiding the user to

Constraints

The design concept of constraining refers to determining ways of restricting the kind of user interaction that can take place at a given moment. There are various ways this can be achieved. A common design practice in graphical user interfaces is to deactivate certain menu options by shading them, thereby restricting the user to only actions permissible at that stage of the activity. One of the advantages of this form of constraining is it prevents the user from selecting incorrect options and thereby refuses the chances of making a mistake. The use of different kinds of graphical representations can also constrain a person's interpretation of a problem or information space. For example flow chart diagram show which objects are related to which thereby constraining the way the information can be perceived. Norman classified constraints into three categories: physical, logical, and cultural.

(i)Physical constraints

Physical constraints refer to the way physical objects restrict the movement of things. For example, the way an external disk can be placed into a disk drive is physically constrained by its shape and size, so that it can be inserted in only one way. Likewise, keys on a pad can usually be pressed in only one way.

(ii)Logical constraints

Logical constraints rely on people's understanding of the way the world works. They rely on people's common-sense reasoning about actions and their consequences. Picking up a physical marble and placing it in another location on the phone would be expected by most people to trigger something else to happen. Making actions and their effects obvious enables people to logically deduce what further actions are required. Disabling menu options when not appropriate for the task in hand provides logical constraining. It allows users to reason why (or why not) they have been designed this way and what options are available.

(iii) Culture constraints

Culture constraints rely on learned conventions, like the use of red for warning, the use of certain kinds of signals for danger, and the use of the smiley face to represent happy emotions. Most cultural constraints are arbitrary in the sense that their relationship with what is being represented is abstract, and could have equally evolved to be represented in another form (e.g., the use of yellow instead of red for warning). Accordingly, they have to be learned. Once learned and accepted by a cultural group, they become universally accepted conventions. Two universally accepted interface conventions are the use of windowing for displaying information and the use icons on the desktop to represent operations and documents.

Mapping

This refers to the relationship between controls and their effects in the world. Nearly all artefacts need some kind of mapping between controls and effects, whether it is a flashlight, car, power plant, or cockpit. An example of a good mapping between controls is effect is the up and down arrows used to represent the up and down movement of the cursor, respectively, on a computer keyboard. The mapping of the relative position of controls and their effects is also important. Consider the various musical playing devices. How are the controls of playing rewinding and fast forward mapped onto the desired effects? They usually follow a common convention of providing a sequence of buttons, with the play button in the middle, the rewind button on the left and the fast-forward on the right.

Consistency

This refers to designing interfaces to have similar operations and use similar elements for achieving similar tasks. In particular, a consistent interface is one that follows rules, such as using the same operation to select all objects. For example, a consistent operation is using the same input action to highlight any graphical object at the interfaces, such as always clicking the left mouse button. Inconsistent interfaces, on the other hand, allow exceptions to a rule. An example of this is where certain graphical objects (e.g., email messages presented in a table) can be highlighted using the right mouse button, while all other operations are highlighted using the left button. A problem with this kind of inconsistency is that is quite arbitrary, making it difficult for users to remember and making the users more prone to mistakes.

One of the benefits of consistent interfaces, therefore, is that they are easier to learn and use. Users have to learn only a single mode of operation that is applicable to all objects. This principle worked well for simple interfaces with limited operations, like mini CD player with small number of operations mapped onto separate buttons. Here all the user has to do is learn what each button represents and select accordingly. However, it can be more problematic to apply the concept of consistency to more complex interfaces, especially when many different operations need to be designed for.

For example, consider how to design an interface for an application that offers hundreds of operations. There is simply not enough space for a thousand buttons, each of which maps onto an individual operation. Even if there were, it would be extremely difficult and time consuming for the user to search through them all to find the desired operation. A much more effective design

solution is to create categories of commands that can be mapped into subsets of operations. For the word-processing application, the hundreds of operation available are categorized into subsets of different menus. All commands that are concerned with file operations are placed together in the same file menu.

Feedback

Related to the concept of visibility is feedback. This is best illustrated by an analogy to what everyday life would be like without it. Imagine trying to play a guitar, slice bread using knife, or write a pen if none of the actions produced any effect for several seconds. There would be an unbearable delay before the music was produced, the bread was cut, or the words appeared on the paper, making it almost impossible for the person to continue with the next strum, saw, or stroke.

Feedback is about sending back information about what action has been done and what has been accomplished, allowing the person to continue with the activity. Various kinds of feedback are available for interaction design—audio, tactile, verbal, visual, and combinations of these. Deciding which combinations are appropriate for different kinds of activities and interactivities is central. Using feedback in the right way can also provide the necessary visibility for user interaction.

User-Centered Approach

The user-centered approach means that the real users and their goals, not just technology, should be the driving force behind development of a product. As a consequence, a well designed system should make the most of human skill and judgment, should be directly relevant to the work in hand, and should support rather than constrain the user. This is less technique and more a philosophy.

In 1985, Gould and Lewis laid down three principles they believed would lead to a "useful and easy to use computer system." These are very similar to the three key characteristics of interaction design.

• Early focus on users and tasks: This means first understanding who the users will be by directly studying their cognitive, behavioural, anthropomorphic, and attitudinal characteristics. This required observing users doing their normal tasks, studying the nature of those tasks, and then involving users in the design process.
• Empirical measurement: early in development, the reactions and performance of intended users to printed scenarios, manuals, etc, is observed and measured. Later on, users interact with simulations and prototypes and their performance and reactions are observed, recorded and analyzed.
• Iterative design: when problems are found in user testing, they are fixed and then more tests and observations are carried out to see the effects of the fixes. This means that design and development is iterative, with cycles of "design, test, measure, and redesign" being repeated as often as necessary. Iteration is something, which is emphasized in user-centered design and is now widely accepted that iteration is required.

Applying ethnography in design

Ethnography is a method that comes originally from anthropology and literally means "writing the culture". It has been used in the social sciences to display the social organization of activities, and hence to understand work. It aims to find the order within an activity rather than impose any

framework of interpretation on it. It is a broad-based approach in which users are observed as they go about their normal activities. The observers immerse themselves in the users' environment and participate in their day-today work, joining in conversations, attending meetings, reading documents, and so on. The aim of an ethnographic study is to make the implicit explicit. Those in the situation, the users in this case, are so familiar with their surroundings and their daily tasks that they often don't see the importance of familiar actions or happenings, and hence don't remark upon them in interviews or other data-gathering sessions.
There are different ways in which this method can be associated with design. Beynon-Davies has suggested that ethnography can be associated with the development as "ethnography of", "ethnography for", and "ethnography within." Ethnography of development refers to studies of developers themselves and their workplace, with the aim of understanding the practices of development (e.g. Button and Sharrock). Ethnography for development yields ethnographic studies that can be used as a resource for development, e.g., studies of organizational work. Ethnography within software development is the most common form of study; here the techniques associated with ethnography are integrated into methods and approaches for development.

Ethnography framework

Ethnographic framework has been developed specifically to help structure the presentation of ethnographies in a way that enables designers to user them. This framework has three dimensions

1. Distributed co-ordination
2. Plans and procedures
3. Awareness of work

1. Distributed co-ordination

The distributed co-ordination dimension focuses on the distributed nature of the tasks and activities, and the means and mechanisms by which they are coordinated. This has implications for the kind of automated support required.

2. Plans and procedures

The plans and procedures dimension focuses on the organizational support for the work, such as workflow models and organizational charts, and how these are used to support the work. Understanding this aspect impacts on how the system is designed to utilize this kind of support.

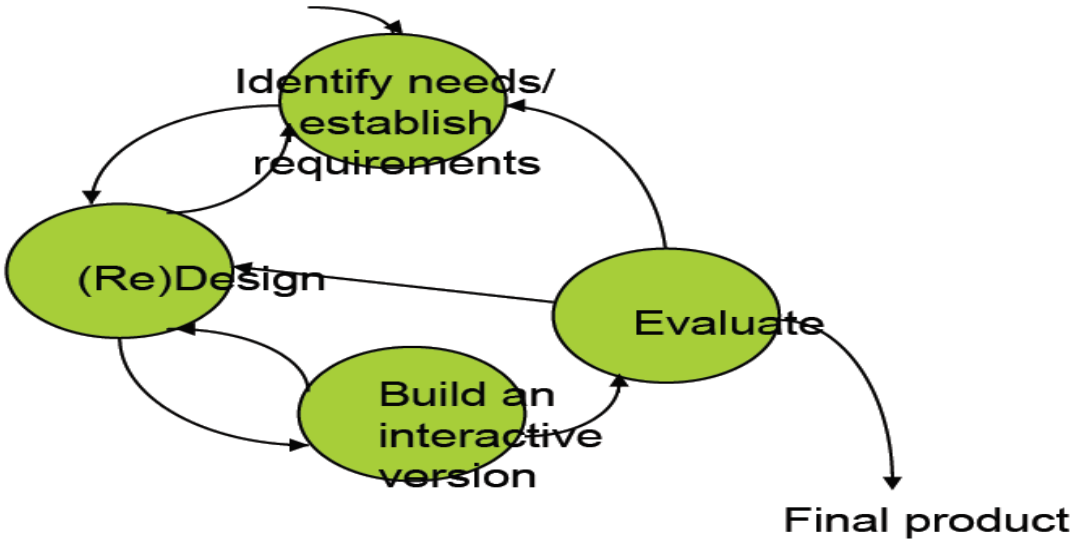
3. Awareness of work

The awareness of work dimension focuses on how people keep themselves aware of others' work. No one works in isolation, and it has been shown that being aware of others' actions and work activities can be a crucial element of doing a good job. In the stock market example this was one aspect that ethnographers identified. Implications here relate to the sharing of information. Rather than taking data from ethnographers and interpreting this in design, an alternative approach is to train developers to collect ethnographic data themselves. This has the advantage of giving the designers first-hand experience of the situation. Telling someone how to perform a task, or explaining what an experience is like is very difficult from showing him or her or even gaining the experience themselves. Finding people with the skills of ethnographers and interaction designers may be difficult, but it is possible to provide notational and procedural mechanisms to allow designers to gain some of the insights first-hand.

User-centred design process

1. Identify needs and establish requirements
2. Generate alternative solutions/designs
3. Build interactive prototypes that can be communicated and assessed

4. Evaluating design



User centred design

Research Techniques Used in HCI

Structured interviews

Most software projects start with a series of meetings in which the system requirements are established. The agenda of these meetings is often concerned with many other matters than the user interface, however. In fact the people who will use the completed system may not even be present. Their requirements are defined by a representative (a system analyst for an internal project or a market researcher for a product) who may not have much experience of design for usability. For this reason, user interface designers often conduct studies specifically to discover the requirements of the system users. One of the cheapest and most straightforward techniques is to conduct ***interviews*** with the users. Interviews must be carefully planned to be effective, however. They are generally more or less ***structured***, encompassing a selected range of users, and taking care to encourage cooperation from users who may feel threatened or anxious.

Observational studies

Observational studies are a less intrusive way of capturing data about users' tasks, and can also be more objective. They involve more intensive work, however. An observational study of tasks that take place in a fixed location can be conducted by making video recordings which are transcribed into a ***video protocol***. This protocol can then be used for detailed analysis of the task - relative amounts of time spent in different sub-tasks, common transitions between different sub-tasks, interruptions of tasks and so on. Audio recordings can

also be used for this purpose in certain domains, but these are less likely to be useful for task analysis than they are in think-aloud studies.

Ethnographic field studies

Ethnographic study methods recognise that the investigator will have to interact directly with the subject, but while taking sufficient care to gain reasonably complete and objective information. An ethnographic study will attempt to observe subjects in a range of contexts, over a substantial period of time, and making a full record using any possible means (photography, video and sound recording as well as note-taking) of both activities and the artefacts that the subject interacts with. Ethnographic methods are becoming increasingly important in HCI, to an extent that many technology companies will now employ an anthropologist as their first social science expert, rather than a psychologist.

Prototyping

Prototyping is becoming increasingly important as a software design method, particularly addressing the problems of developing user interfaces within a strict ***waterfall*** development model. Prototype can be demonstrated to clients and used as a basis for discussion. If a spiral development model is adopted rather than a waterfall, the prototype can be ***refined iteratively*** until the full system functionality is achieved. Incremental prototyping requires that the rapid prototyping tool also meets the engineering requirements of the final system.

Usability evaluation methods

Summative Evaluation techniques

Summative evaluation, often performed under the umbrella of ***usability testing*** is carried out at the end of a project after the system has been built, to assess whether it meets its specification, or whether a project was successful. This is in contrast to ***formative*** evaluation, where the main objective is to contribute to the design of the product, by assessing specifications or prototypes before the system has been built. Formative evaluation is often ***analytic*** (it proceeds by reasoning about the design), while summative evaluation is often ***empirical*** (it proceeds by making observations or measurements).

However, summative evaluation is not so popular in commercial settings as in academic settings. After a system has been built, the creators tend not to be interested in further advice – many small companies consider that releasing a product is so cheap that they might as well release it as testing it. Any usability problems can be resolved in following version, in response to user feedback (***discount*** usability) techniques that are less rigorous than academic studies, but still give more information than crossing your fingers and hoping that users will like it (although that is a surprisingly common approach in small companies). Larger, more established, companies spend more on summative evaluation of new products, because of the danger to their reputation if they were to release a product that was very much inferior. For this reason, companies like Microsoft carry out summative evaluation studies of all products, before they undergo even early (beta) market release. Usability problems can then be tracked and resolved in the same way as other software defects, using the same process as for functional bugs found during system testing.

Controlled experiments

The most common empirical method used in HCI research, derived from its origins in human factors and experimental psychology, is the controlled experiment. An experiment is based on

a number of ***observations*** (measurements made while someone is using an experimental interface).

The sets might be multiple observations of one person performing a task over many ***trials***, or of a range of people (experimental ***participants***) performing the same task under controlled conditions. As with most human performance, the measured results will usually be found to have a ***normal distribution***.

We need to know whether the difference between the averages is the result of ordinary random variation, or the effect of the changes we made to the user interface. This involves a statistical ***significance test*** such as the ***t-test***.

Think aloud studies

Although not really experiments (they are often conducted without a hypothesis, and the data is qualitative rather than quantitative), controlled studies in HCI often use the think aloud technique (described in an earlier lecture) to gain insight into the way the user has interpreted a prototype. When used as a rigorous scientific technique, a great deal of care is taken to ensure that the users vocalizes every thought they are aware of, and the recording is transcribed and analysed in detail for evidence of particular mental processes. However in a commercial context, the think-aloud protocol can seem much more like real-time evaluation feedback, in which users are simply asked to make as many comments as possible on the user interface. This may not provide very much scientific insight, but at least it avoids the problem of users who spend an hour using a new system, then say almost nothing in the way of feedback.

Other empirical techniques

Hypothesis testing is a very useful technique for making quantifiable statements about improvements in a user interface. It also hides a lot of useful information, however. Experimental subjects usually have a lot of useful feedback about the interface that they are trying, but there is no easy way to incorporate this into statistical analyses. Instead, we use a range of other techniques to capture and aggregate interpretative reports from system users.

Surveys

Surveys include a range of techniques for collecting report data from a population. The most familiar types of survey are public opinion polls and market research surveys, but there are a much greater range of survey applications. Surveys are usually composed of a combination of ***closed*** and ***open*** questions. Closed questions require a yes/no answer, or a choice on a ***Likert*** scale - this is the familiar 1 to 5 scale asking respondents to rank the degree to which they agree with a statement. Closed questions are useful for statistical comparisons of different groups of respondents. In open questions the respondent is asked to compose a free response to the question. The latter requires a methodical ***coding*** technique to structure the content of the responses across the population, and is particularly useful for discovering information that the investigator was not expecting.

Questionnaires

Questionnaires are a particular type of survey. (Interview studies of a sample population are also a form of survey). Questionnaires are generally used to gather responses from a larger sample, and can be administered by email as well as on paper.

Field tests

Some very successful software companies have carried out ***field testing*** of their products in addition to field studies at the specification phase. A well-documented example is the "follow-me-home" programme carried out by Intuit Inc. after the release of their Quicken product. Company researchers selected customers at random, when they were buying a shrink-wrapped copy of Quicken in a store. The researcher then went home with the customer in order to observe them as they read the manuals, installed the product, and used it for their home financial management. Intuit directly attribute the impressive success of the product to this type of exercise, and to the observational studies they carried out during initial product planning.

Graphical User Interface

A user interface is a collection of techniques and mechanisms to interact with the system. A Graphical User Interface is a type of user interface which allows people to perform operations and call actions on objects on the interface. The operations include accessing and modifying objects by pointing, selecting, and manipulating. All objects have standard resulting behaviour.

Principles of Visual Interface Design

The human brain is a superb pattern-processing computer, making sense of the dense quantities of visual information that bombard us everywhere we look. Our brains manage this chaotic input by discerning visual patterns and establishing a system of priorities for the things we see which in turn allows us to make conscious sense of the visual world. Visual interface design must take advantage of our innate visual processing capabilities to help programs communicate their behaviour and function to users.

There are some important principles that can help make visual interface as easy and pleasurable to use as possible. Kevin Mullet and Darrell Sano (1995) provide a superb detailed analysis of these principles:

Visual interfaces should:
- Avoid visual noise and clutter
- Use contrast, similarity, and layering to distinguish and organize elements
- Provide visual structure and flow at each level of organization
- Use cohesive, consistent, and contextually appropriate imagery
- Integrate style and function comprehensively and purposefully

Assignment

(i) Differentiate between Web User Interface and Graphical User Interface (GUI)
(ii) Are there any similarities between Web User Interface and GUI ?