# The Usability Design Process

## The Usability Design Process

There are many approaches to interaction design, and no one method guarantees good results.

Therefore a mix of approaches is advisable, for example:

* Find out who the users are, what they want to do with the system, etc.
* Design to suit, taking into account:
  + Interaction styles and devices
  + Guidelines (where available)
  + Heuristics and modelling techniques
* Evaluate and revise as necessary

Such approaches are often collectively termed *user-centered design*.

## Who are the users?

It is imperative to know who the system is being designed for.

Therefore the designer should start by identifying the target users.

This can be done by drawing-up a 'profile' which includes factors such as:

|  |  |
| --- | --- |
| Age | Sex |
|  | |
| Physical abilities and disabilities | Educational background |
|  | |
| Culture | Attitude |
|  | |
| Computing/IT knowledge experience | |

It's very difficult to design for a large, loosely-defined group.

Studies suggest such that applications designed in this way often fail to meet the needs of any of their users.

To overcome this, many HCI specialists recommend designing for small, tightly-defined groups.

One approach is to develop a small number of profiles of imaginary users. These are called *personas*.

A persona should cover all the factors listed above, but should also include other details, such as likes and dislikes, habits, etc..

The persona can be a composite, taking characteristics from a number of real people, but should be consistent and realistic.

The persona should read as the description of the real person.

A useful approach is to develop several personas, each representing a group of users.

For example, the personas for a particular project might include:

* A young person who uses IT regularly for entertainment and education purposes.
* A middle-aged person who uses IT regularly at work.
* A middle-aged person who does not use computers at work but has a home-computer and uses it for email, web-surfing and in connection with their hobby.
* A retired person who has little knowledge of computers, and who uses them only when absolutely necessary.

The use of personas is relatively new in computing but is widely used in other industries and is very successful.

It has been found that a product developed to meet the specific needs of a particular group often meets the needs of a much wider group.

## What do the users want to do?

In identifying needs we must distinguish between:

* Needs identified by professional designers/developers. These are often referred to as **normative needs**
* The needs of the end-user. These can be difficult to determine. It often helps to think in terms of:
  + **Expressed needs** - what end-users SAY they want.
  + **Felt needs** - what end-users ACTUALLY want (or would like) from the system.   
    In many cases, users lack the technical vocabulary or understanding to express such needs, but feel them nonetheless.   
    Felt needs are often expressed as a general dissatisfaction with a system.

There may be considerable discrepancies between normative and end-user needs, and possibly conflicts.

Some end-users needs may be excessive and impractical. However, this does not mean they can be ignored.

Such needs often reflect a deeper dissatisfaction with the system which cannot be overcome but can be ameliorated through reasonable compromise.

The traditional methods for determining user needs are:

* direct observation (where possible)
* questionnaires
* interviews.

## Conceptual Design

Having identified the users and determined what they want to do with the system, the next stage is to develop a conceptual design.

The conceptual design should describe what appears on the screen (or is presented through other modalities) at each stage in the interaction.

The design should take relevant guidelines into account, and may also be based on formal modelling, etc..

However, guidelines and models are largely *analytical* rather than *generative* tools, and the designer has to use insight and imagination to create an interface.

The designer should also ask:

* Is it necessary to re-engineer the task?
* Should a metaphor be employed?

### Is it necessary to re-engineer the task?

Wherever possible, interface designers try to accommodate the users' conceptualisation of a task.

To do this, they first attempt to identify how users conceptualise the task, then design an interface that allows users to carry out the task in accordance with this concept.

However, sometimes this is impractical because, for example:

* the move from (e.g.) a paper-based system to a computer-based system introduces new possibilities which do not fit within the existing system concept.
* there are conflicting requirements.

In these cases it may be necessary to force users to adopt a new way of conceptualising the task.

This is known as *re-engineering* the task.

This often involves the creation of a metaphor (see below) to enable users to more easily comprehend the possibilities offered by the new system.

### Should you employ a metaphor?

Webster's dictionary defines 'metaphor' as follows:

A figure of speech in which a word or phrase denoting one kind of object or action is used in place of another to suggest a likeness or analogy between them.

It can be argued that everything which takes place within a computer uses metaphors, since computing has borrowed words, phrases and ideas from the outside world.

For example, the grouping of particular items of data into a 'file' can be thought of as a metaphor.

It is more helpful to the user to view the data as a single file than as (e.g.) many pieces of data fragmented across a disk.

Moreover, some metaphors have already been defined for us, such as the 'desktop' metaphor used in many GUIs.

However, it may be appropriate to adopt a metaphor for a specific task. Many interfaces use metaphors in this way, for example:

* Calculator utilities are often designed to look like physical calculators.
* E-commerce websites often use a 'shopping-basket & checkout' metaphor.
* Audio and video applications often mimic the control-functions and layout on physical audio/video devices such as CD players, DVD-players, etc..

A metaphor is simply a way of helping users develop a suitable *mental model* of a system.

Metaphors allow the user to employ previously-acquired skills and knowledge when using an application rather than having to learn new skills and/or acquire new knowledge.

However, metaphors must be chosen and used with care.

A metaphor carries with it a particular set of ideas, and users will assume that if you have chosen to use a particular metaphor it is because you wish to exploit these ideas.

Therefore, there is an expectation that the interface will conform to the metaphor. If this expectation is not fully met, the interface may prove difficult and frustrating to use.

Many psychologists believe we carry around a model of the world, and of each of the systems we encounter within it.

We build this collection of models throughout our lives, constantly refining them as new experiences reveal inconsistencies or inadequacies in the models.

We deal with new situations by identifying the most relevant of our existing models and employing it.

* If a new situation is similar to one we have previously encountered, we will have few problems coping with it.
* If a new situation is quite unlike anything we have previously encountered, we may have difficulty coping with it.

Two main types of mental model have been identified: Structural and Functional.

* Structural Models

A structural model represents the structure of some system or device.

For example, we may have a structural model of the national road network and be able to use to identify the best route to take when travelling to particular destination.

A London taxi-driver probably has a very detailed structural model of London's road network.

* Functional Models

A functional model represents procedural knowledge about how to use a system or device.

It is based on past experience of using a particular system or device, but - unlike a structural model - is not based on knowledge of how the device works.

For example, someone who uses the ticket-machines at an underground railway station may have a detailed functional model of their use, but have no knowledge of the internal structure of the machine and its software.

The difference between the two types of model can be summarised by saying that:

* a structural model is a model of how something works
* a functional model is a model of how to use something