**User Interface Design**

Table of Contents

[Usability 3](#_Toc83393076)

[Human Factors in Interface Design 4](#_Toc83393077)

[Basic Principles 5](#_Toc83393078)

[User Interface Design Principles 6](#_Toc83393079)

[Design Issues 8](#_Toc83393080)

[Interaction Styles 9](#_Toc83393081)

[Website Usability Problems 10](#_Toc83393082)

[User-Centred Design 11](#_Toc83393083)

[UI Design Teams 12](#_Toc83393084)

[Usability Design Process 13](#_Toc83393085)

[Golden Rules 14](#_Toc83393086)

[User Interface Evaluation 16](#_Toc83393087)

[Evaluation Techniques 16](#_Toc83393088)

The **user interface** is what allows the user to interact with the system. The user can make queries and give inputs and the system can display information in return. This means the way the interface is designed is of utmost importance. No matter how good the actual system is, if the user interface is designed poorly, end users will not use the system. On the flip side, if the user interface is great and end users are comfortable using the interface, then the system will have widespread usage, even if it is not the best system in the market.

For example, consider an e-commerce site that requires their users to register for an account before they can make purchases. This results in users being frustrated on several fronts. Firstly, they may not want to create an account just to make a purchase. Secondly, they may not remember if they already have an account and, if they do, may be forced to go through a password recovery process. All of this is unnecessary and makes the experience painful for the users, which in turn results in lost sales. Simply allowing users to make purchases without creating an account can solve this.

Another example is an e-commerce store where the products are listed in drop-down menus. Replacing this view with a grid with images of the products and some text describing them improves sales significantly.

In the second example, the basic theory is that the customer concentrates on whatever is highlighted for them. If we present a huge amount of text, they will not read it. If we highlight specific parts, such as links or headings, those specific parts will catch their eye.

## Usability

**Usability** is a measure of the **effectiveness**, **efficiency** and **satisfaction** with which users can achieve specific goals in a system.

When discussing **profits**, there are a few things we need to take into account. Firstly, we need to think about the **costs**. Costs can be fixed, for things like warehouse rent, or variable, for things like packaging, which depends on how many products we produce. We also need to think about **revenue**, which depends on the number of products we are selling and the price of the products.

The numbers depend on how **usable** we make our products. If the software we create is very usable, we will have more sales, which increases revenue. It may also reduce variable costs, since we will need to spend less on customer support. However, fixed costs may increase, since we will need to make the product usable in the first place. Thus, usability is very closely related to the profits we make.

Poor usability results in:

* Anger and frustration for users
* Decreased productivity in workplaces
* Higher error rates
* Physical and emotional injury
* Equipment damage
* Loss of customer loyalty

All of this essentially translates, in one way or another, to decreased profits.

## Human Factors in Interface Design

When designing our user interface, we need to take into account several factors about human beings.

Human beings have a **limited short-term memory**. People can instantaneously remember 7 items at most. If we give them more than this, they will make mistakes.

People **make mistakes**. Even in the best designed systems, we cannot ignore this fact. When something goes wrong, we need to handle it properly. Inappropriate alarms or error messages can stress out the user and lead to even more mistakes.

**People are different** and we need to take these differences into account. If we design an application that is meant to be used by elderly people for example, we need to take into account that they are more likely to have poor eyesight. Thus, making the user interface elements larger will help them.

People have **different preferences**. There are cases where we can take different preferences into account to provide a better user experience. For example, most people use a GUI to control their OS, but some prefer using a terminal. Having the terminal available does not cause any issues but does make using the system a better experience for a part of our user base.

## Basic Principles

There are a few basic principles we need to keep in mind when designing a user interface:

* Assume the user has not read the manual
* Assume they have not attended any training sessions
* Assume there is no external help available to the users

These assumptions means that we need to create all the controls in our system in a way such that it is **clear**, **understandable** and **intuitive**. We need to take into account the needs, experience levels and capabilities of most of our users. If the system is aimed at people who are non-technical, we need to make sure the user interface is very easy to understand and use. We also need to take into account mental and physical limitations, as mentioned in the previous section, and keep in mind that a user can always make mistakes.

## User Interface Design Principles

On top of the basic principles, there are some general **user interface design principles** that we should keep in mind. However, not all of these are applicable to every system, so we need to decide which ones will help the system we are creating.

* **User Familiarity** – The interface should contain elements that are familiar to the user. For example, we should use the term ‘folder’ instead of ‘directory’ in a file system, there should be an option to close the window on the top-right corner, etc.
* **Consistency** – The system should be consistent throughout. For example, if we have a menu bar on every page of the system, the menu bar should appear at the same location on every page and should be formatted in the same manner.
* **Minimal Surprise** – This is similar to the point about user familiarity. If we are using a command or feature that is common, it should behave in the same way, so that the user can tell what it does just by looking at it.
* **Recoverability** – The system should be resilient to user errors and should allow the user to recover from mistakes. Some features that support recoverability are the ‘undo’ feature, confirmation before destructive actions, etc.
* **User Guidance** – We should provide some guidance to the users, especially in complicated systems. This includes things like help systems, online manuals, etc.
* **User Diversity** – We should include facilities that allow different types of users to interact with the system. For example, we discussed above how some users may be older and have difficult seeing. For such users, we should provide larger text.

## Design Issues

There are essentially two major issues we are trying to address when creating a user interface:

1. How should information from the user be provided to the system?
2. How should information from the system be provided to the user?

A good user interface answers these questions very well.

## Interaction Styles

|  |  |  |  |
| --- | --- | --- | --- |
| **Interaction Style** | **Main Advantages** | **Main Disadvantages** | **Application Examples** |
| Direct Manipulation | Fast and intuitive interaction | May be hard to implement | Video Games |
| Easy to learn | Only suitable where there is a visual metaphor for tasks and objects | CAD systems |
| Menu Selection | Avoids user error | Slow for experienced users | Most general-purpose systems |
| Little typing required | Can become complex if many menu options |
| Form Fill-In | Simple data entry | Takes up a lot of screen space | Stock control |
| Easy to learn | Causes problems where user options do not match the form fields | Personal loan processing |
| Checkable |
| Command Language | Powerful and flexible | Hard to learn | Operating Systems |
| Poor error management | Command and control systems |
| Natural Language | Accessible to casual users | Requires more typing | Information retrieval systems |
| Easily extended | Natural language understanding systems are unreliable |

## Website Usability Problems

Some typical issues with regards to usability that are common in websites are:

* **Navigation** – It may be difficult to know where in the website we are and to find what we need. For example, if we list all the payment information for students in a university on one page, instead of dividing it into categories for undergraduate, masters and PhD students, people are likely to have difficulty finding the information they need.
* **Structure**
* **Layout** – There should be lots of white space and graphical elements should be large.

The three points above are actually very closely related. They are quite difficult to discuss separately. The point about navigation should sufficiently sum things up.

## User-Centred Design

**User-Centred Design** is a UI design methodology that involves opening up a **dialog** between the customer and the designer. The designer needs to get to know the customer and **understand** them. They need to build an application based on their interaction, following the **usability principles** as discussed above. They also need to **test** their design by **observing** users in a real work setting, with the same environment and work load.

## UI Design Teams

A typical UI design team could include:

* Analysts
* Designers
* Technology Experts
* Graphics Artists
* Social and Behavioural Scientists
* Programmers

Clearly, there are people from a wide range of fields working together.

## Usability Design Process

The process of designing a usable user interface has several steps:

1. **Scenario Development** – The first step is to develop the scenarios under which the system will be used.
2. **Interface Structure Design** – This refers to how the interface will be structured.
3. **Interface Standards Design** – Next, we need to decide on the standards that we will use throughout the system. For example, what will the menu bar on every page look like?
4. **Interface Design Prototyping** – We can create low quality prototypes to begin with, but prototypes are essential to getting feedback from others. The prototype could even just be a drawing on paper.
5. **Interface Evaluation** – We need to gather feedback from other designers and users in order to find out how we can improve the user interface and make it more usable.

We go over these steps repeatedly in cycles until the final user interface has been decided upon.

## Golden Rules

We will now go over some golden rules for user interface design.

1. **Strive for Consistency** – We have previously mentioned the importance of keeping user interface elements consistent. Consistency includes the use of similar colours, typography, terminology, commands and menus. A consistent interface will allow users to complete their tasks easily.
2. **Create Shortcuts** – Frequent users will want to use shortcuts for tasks they perform frequently. For example, every video player has a shortcut for Play/Pause, the spacebar.
3. **Offer Informative Feedback** – At every stage, we need to ensure the user knows what is happening in the system. The information we give needs to be meaningful, relevant, clear and contextual. For example, if the user is uploading a file, we should show the progress in some manner.
4. **Dialogs Should Offer Closure** – Related actions should be performed in a sequence. The end of an action is important. We should provide the user with some informative feedback so that they know the step is complete. We could also provide information about what to do next if that is appropriate. Essentially, we cannot keep the user wondering about what they should do next.
5. **Simple Error Handling** – A good interface should avoid errors as much as possible. However, when an error does occur, we need to make it easy for the user to understand what went wrong and how to solve it. This could be as easy as displaying a clear error message.
6. **Allow Easy Reversals of Actions** – Users will be relieved to find a clear and immediate ‘undo’ option if they perform an action by mistake. For example, Gmail has a feature to undo the sending of an email for a few seconds after the email is initially sent. Users are more likely to explore the system without feeling anxious if they know they can reverse their actions.
7. **Give Users Control** – We need to ensure that users have complete control over the system. We should try to avoid surprises, interruptions or anything at all that the user has not asked for. Users should be the initiators of actions. For example, the user should be asked about what notifications they wish to receive and how frequently they wish to receive them. We should not simply start sending notifications frequently, since that will annoy the users.
8. **Reduce Short-Term Memory Load** – We should not force users to remember lots of information. A simple, consistent UI helps a lot, since users can perform actions nearly on muscle memory. There are also other features we can implement, such as a list of recently viewed or purchased items on an e-commerce website.

## User Interface Evaluation

We should carry out some **evaluation** to assess whether the UI we designed is suitable. Full scale evaluation is expensive and impractical for most systems. Ideally, we should evaluate the UI against some **usability specifications**, but such specifications are rarely created.

Some common criteria we can judge are:

* **Learnability** – How easy is it for a new user to learn how to use the system?
* **Speed of Operations** – How quickly can a user perform operations in the system?
* **Robustness** – How well does the system handle errors?
* **Recoverability** – How good is the system at recovering from errors?
* **Adaptability** – How well does the system adapt to different working environments?

### Evaluation Techniques

Some less expensive techniques to evaluate UI are:

* Questionnaires
* Observing users in their working environment
* Video snapshots of typical users
* Software to gather information about usage of the user interface
* Software for direct feedback from users