Intro to Software Engineering

SE-110

Interface Design

Lecture # 24
08 April

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Today's Outline

- **Interface Design**
- **Introduction**
- **Importance of interface Design**
- **Interface Design Errors**
- **Golden rules for interface design**

Interface Design

- Interface design is not just about the arrangement of media on a screen
- It's designing an entire experience for people, hence a "look and feel"
- **Easy to learn**
- **Easy to use**
- **Easy to understand**

Importance of Interface Design

- System users often judge a system by its interface rather than its functionality
- A poorly designed interface can cause a user to make catastrophic errors.
- Poor user interface design is the reason why so many software systems are never used

Interface Design Errors

- **lack of consistency**
- **too much memorization**
- **no guidance / help**
- **poor response**
- **unfriendly**

Why is good user interface design important?

"Interface inconsistency can cost a big company millions of dollars in lost productivity and increased support costs."

Jesse Berst (1993)

- **Reduced number of users**
- **Company will face financial loss**
- **Designer's profile will be affected**

Golden Rules

- Don't do to others what others have done to you. Remember the things you don't like in software interfaces you use. Then make sure you don't do the same things to users of interfaces you design and develop.
- The three areas of user interface design principles are:

1) **Place the user in control**
2) **Reduce the user's memory load**
3) **Make the interface consistent**

Golden Rule # 1
Place the user in control

1. Interaction Modes

- Define interaction modes in a way that does not force a user into unnecessary or undesired actions
- Two types of interaction modes
- The first type is **application modal**. When in an application mode, users are not allowed to work elsewhere in the program, or it places them in a certain mode for the whole program.
- For example, if working with a database of information and the **viewing mode** is chosen, the program will not allow users to add, delete, or **edit the data** record.
- It is very important that, when designing an interface, you include a visual indicator showing the user the current mode.

Interaction Modes

- The second type of mode is ***system modal***. *While in a* system mode, users are not allowed to work anywhere else on the computer until the mode is ended, or it places them in a certain mode no matter what program they are using.
- The user shall be able to enter and exit a mode with little or no effort
- Example: a document is **printing**, and a message window pops up stating the **printer is out of paper**. Users should not have to get up and put paper in the printer immediately, or even remove the message from the screen. They should be able to continue working. Users might even want to keep the message window on the screen as a reminder to add paper later.

2. Provide Flexible Interaction

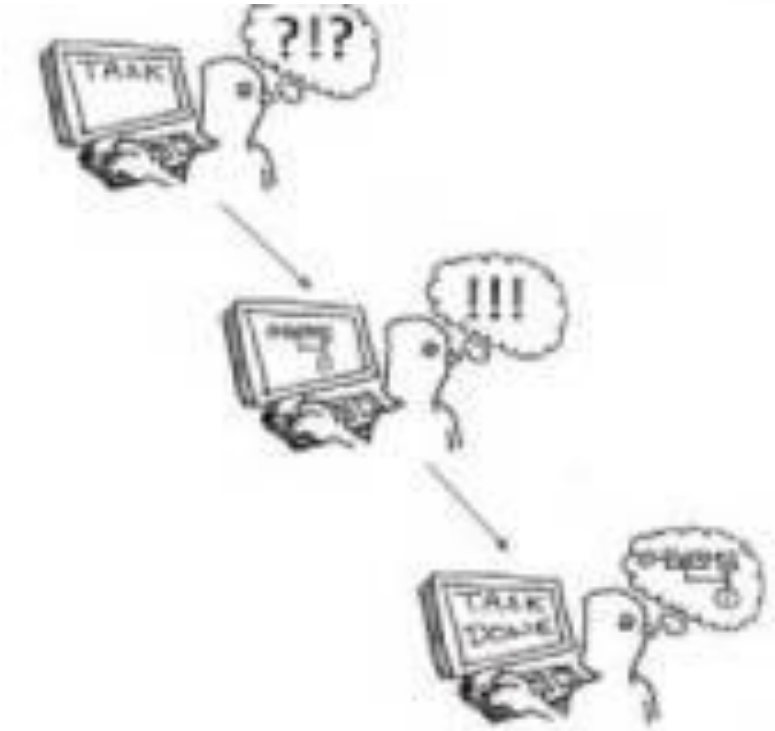
- The user shall be able to perform the same action via keyboard commands, mouse movement, or voice recognition

3. Uninterruptible User Interaction

- **Don't force users to complete predefined sequences. Give them options—to cancel or to save and return to where they left off.**
- **The user shall be able to easily interrupt a sequence of actions to do something else (without losing the work that has been done so far)**
- **"Wizards" are used more and more to lead users through common tasks, but don't lead with an iron hand, let users stay in control while the interface *guides them rather than forces them through steps in a task.***

Wizards

Wizards are designed to help you do tasks you haven't done before, and to do them like an expert.



4. Provide reversible actions, and immediate feedback

- Provide undo actions for users, and hopefully, also redo actions. Inform users if an action cannot be undone and allow them to choose alternative actions, if possible.
- Provide users with some indication that an action has been performed, either by showing them the results of the action, or acknowledging that the action has taken place successfully

5. Provide simple navigation

- **Provide meaningful paths and exits**
- **Users should be able to relax and enjoy exploring the interface of any software product. Users should not be afraid to press a button or navigate to another screen.**
- **navigation is a simple process to learn—basically, navigation is the main interaction technique on the Internet.**

6. Streamlined Interactions

- **Streamline interaction as skill levels advance and allow the interaction to be customized**
- **Don't sacrifice expert users for an easy-to-use interface for casual users. You must provide fast paths for experienced users.**
- **For example, the menu bar and pull-downs of a program can be set up as "standard" or "advanced", depending on user preferences and the types of tasks being performed.**
- **The user shall be able to use a macro mechanism to perform a sequence of repeated interactions and to customize the interface**

7. Hide technical internals from the casual user

- **The user shall not be required to directly use operating system, file management, networking. etc., commands to perform any actions. Instead, these operations shall be hidden from the user and performed "behind the scenes" in the form of a real-world abstraction**
- **The interface can be made transparent by giving users work objects rather than system objects. Trash cans, waste baskets, shredders, and in- and out-baskets all let users focus on the tasks they want to do using these objects, rather than the underlying system functions actually performed by these objects.**

8. Design for Direct Interaction

- **The user shall be able to manipulate objects on the screen in a manner similar to what would occur if the objects were a physical thing (e.g., stretch a rectangle, press a button, move a slider)**
- **Users feel more comfortable and in control of the interface if they can personalize it with their favorite colors, patterns, fonts, and background graphics for their desktop.**

Golden Rule # 2
Reduce the User's Memory Load

1. Relive Short Term Memory

- The system should be able to retrieve the previous information so users don't have to remember and retype the information again.
- For example, when filling in online forms, customer names, addresses, and telephone numbers should be remembered by the system once a user has entered them, or once a customer record has been opened.

2. Rely on recognition, not recall

- People have a limited short-term memory—we can 'instantaneously' remember about seven items of information (Miller, 1957). Therefore, if you present users with too much information at the same time, they may not be able to take it all in.
- Provide lists and menus containing selectable items instead of fields where users must type in information without support from the system.
- E.g. file menu, edit menu

Establish Meaningful Default

- **The system shall provide the user with default values that make sense to the average user but allow the user to change these defaults**
- **The user shall be able to easily reset any value to its original default value**
- **E.g. Reset Option**

Real world metaphor

- **The visual layout of the interface should be based on a real world metaphor**
  - **The screen layout of the user interface shall contain well-understood visual cues that the user can relate to real-world actions**
  - **For example, a bill payment system should use a checkbook and check register metaphor to guide the user through the bill paying process. This enables the user to rely on well-understood cues, rather than memorizing an interaction sequence.**

Define shortcuts

- Define shortcuts that are intuitive
  - The user shall be provided mnemonics (i.e., control or alt combinations) that tie easily to the action in a way that is easy to remember such as the first letter.
  - E.g. Alt P for printing

Provide visual cues (inform)

- Whenever users are in a mode, or are performing actions with the mouse, there should be some visual indication somewhere on the screen that they are in that mode.
- Test a product's visual cues—walk away from the computer in the middle of a task and come back sometime later. Look for cues in the interface that tell you what you are working with, where you are, and what you are doing.
- MS Word, MS Power Point

Disclose information in a progressive fashion

- When interacting with a task, an object or some behavior, the interface shall be organized hierarchically by moving the user progressively in a step-wise fashion from an abstract concept to a concrete action
- e.g., text format options → format dialog box

The more a user has to remember, the more error-prone interaction with the system will be

Golden Rule # 3
Make the Interface Consistent

Maintain Information in Consistent Fashion

- **The interface should present and acquire information in a consistent fashion**
- **Consistency means that users should see information and objects in the same logical, visual, or physical way throughout the product**
  - All visual information shall be organized according to a design standard that is maintained throughout all screen displays
  - Input mechanisms shall be constrained to a limited set that is used consistently throughout the application
  - Mechanisms for navigating from task to task shall be consistently defined and implemented

Meaningful Context

- **Allow the user to put the current task into a meaningful context**
  - The interface shall provide indicators (e.g., window titles, consistent color coding) that enable the user to know the context of the work at hand
  - The user shall be able to determine where he has come from and what alternatives exist for a transition to a new task

Changing Past User Expectations

- **If past interactive models have created user expectations, do not make changes unless there is a compelling reason to do so**.
- Once a particular interactive sequence has become a de facto standard (e.g., alt-S to save a file), the application shall continue this expectation in every part of its functionality

Maintain consistency across a family of applications

- **A set of applications performing complimentary functionality shall all implement the same design rules so that consistency is maintained for all interaction**
- **Learning how to use one program should provide positive transfer when learning how to use another similar program interface.**

That is all