

# Chapter 12

## User Interface Analysis and Design

- Introduction
- Golden rules of user interface design
- Reconciling four different models
- User interface analysis
- User interface design
- User interface evaluation
- Example user interfaces

# Introduction

# Background

- Interface design focuses on the following
  - The design of interfaces between software components
  - The design of interfaces between the software and other nonhuman producers and consumers of information
  - The design of the interface between a human and the computer
- Graphical user interfaces (GUIs) have helped to eliminate many of the most horrific interface problems
- However, some are still difficult to learn, hard to use, confusing, counterintuitive, unforgiving, and frustrating
- User interface analysis and design has to do with the study of people and how they relate to technology

# A Spiral Process

- User interface development follows a spiral process
  - Interface analysis (user, task, and environment analysis)
    - Focuses on the profile of the users who will interact with the system
    - Concentrates on users, tasks, content and work environment
    - Studies different models of system function (as perceived from the outside)
    - Delineates the human- and computer-oriented tasks that are required to achieve system function
  - Interface design
    - Defines a set of interface objects and actions (and their screen representations) that enable a user to perform all defined tasks in a manner that meets every usability goal defined for the system
  - Interface construction
    - Begins with a prototype that enables usage scenarios to be evaluated
    - Continues with development tools to complete the construction
  - Interface validation, focuses on
    - The ability of the interface to implement every user task correctly, to accommodate all task variations, and to achieve all general user requirements
    - The degree to which the interface is easy to use and easy to learn
    - The users' acceptance of the interface as a useful tool in their work

# The Golden Rules of User Interface Design

# Place the User in Control

- Define interaction modes in a way that does not force a user into unnecessary or undesired actions
  - The user shall be able to enter and exit a mode with little or no effort (e.g., spell check → edit text → spell check)
- Provide for flexible interaction
  - The user shall be able to perform the same action via keyboard commands, mouse movement, or voice recognition
- Allow user interaction to be interruptible and "undo"able
  - 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)
  - The user shall be able to "undo" any action

# Place the User in Control (continued)

- Streamline interaction as skill levels advance and allow the interaction to be customized
  - The user shall be able to use a macro mechanism to perform a sequence of repeated interactions and to customize the interface
- 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
- Design for direct interaction with objects that appear on the screen
  - The user shall be able to manipulate objects on the screen in a manner similar to what would occur if the object were a physical thing (e.g., stretch a rectangle, press a button, move a slider)

# Reduce the User's Memory Load

- Reduce demand on short-term memory
  - The interface shall reduce the user's requirement to remember past actions and results by providing visual cues of such actions
- Establish meaningful defaults
  - 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
- 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



# Reduce the User's Memory Load (continued)

- 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
- 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

# Make the Interface Consistent

- The interface should present and acquire information in a consistent fashion
  - 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
- 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

# Make the Interface Consistent (continued)

- 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
- 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

# Reconciling Four Different Models

# Introduction

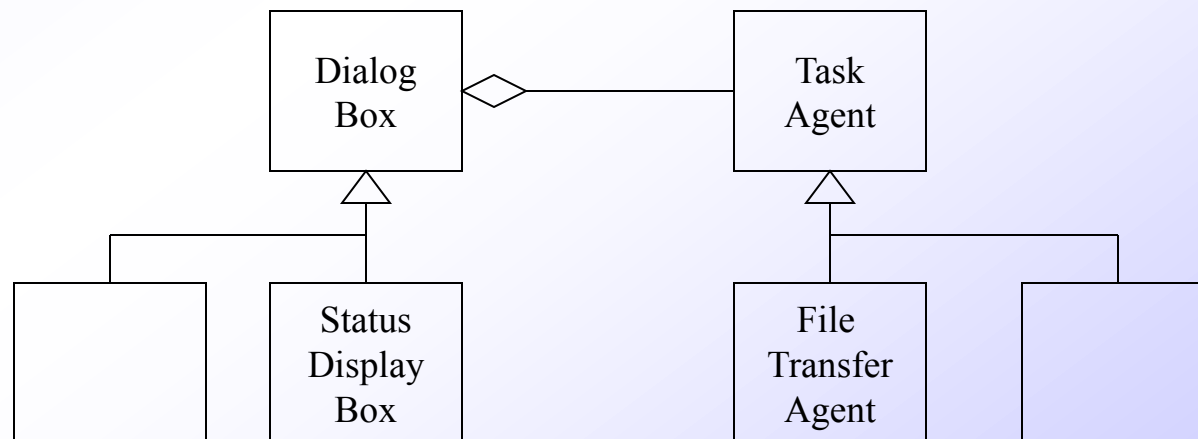
- Four different models come into play when a user interface is analyzed and designed
  - User profile model – Established by a human engineer or software engineer
  - Design model – Created by a software engineer
  - Implementation model – Created by the software implementers
  - User's mental model – Developed by the user when interacting with the application
- The role of the interface designer is to reconcile these differences and derive a consistent representation of the interface

# User Profile Model

- Establishes the profile of the end-users of the system
  - Based on age, gender, physical abilities, education, cultural or ethnic background, motivation, goals, and personality
- Considers syntactic knowledge of the user
  - The mechanics of interaction that are required to use the interface effectively
- Considers semantic knowledge of the user
  - The underlying sense of the application; an understanding of the functions that are performed, the meaning of input and output, and the objectives of the system
- Categorizes users as
  - Novices
    - No syntactic knowledge of the system, little semantic knowledge of the application, only general computer usage
  - Knowledgeable, intermittent users
    - Reasonable semantic knowledge of the system, low recall of syntactic information to use the interface
  - Knowledgeable, frequent users
    - Good semantic and syntactic knowledge (i.e., power user), look for shortcuts and abbreviated modes of operation

# Design Model

- Derived from the analysis model of the requirements
- Incorporates data, architectural, interface, and procedural representations of the software
- Constrained by information in the requirements specification that helps define the user of the system
- Normally is incidental to other parts of the design model
  - But in many cases it is as important as the other parts



# Implementation Model

- Consists of the look and feel of the interface combined with all supporting information (books, videos, help files) that describe system syntax and semantics
- Strives to agree with the user's mental model; users then feel comfortable with the software and use it effectively
- Serves as a translation of the design model by providing a realization of the information contained in the user profile model and the user's mental model



# User's Mental Model

- Often called the user's system perception
- Consists of the image of the system that users carry in their heads
- Accuracy of the description depends upon the user's profile and overall familiarity with the software in the application domain

# User Interface Analysis

# Elements of the User Interface

- To perform user interface analysis, the practitioner needs to study and understand four elements
  - The users who will interact with the system through the interface
  - The tasks that end users must perform to do their work
  - The content that is presented as part of the interface
  - The work environment in which these tasks will be conducted

# User Analysis

- The analyst strives to get the end user's mental model and the design model to converge by understanding
  - The users themselves
  - How these people use the system
- Information can be obtained from
  - User interviews with the end users
  - Sales input from the sales people who interact with customers and users on a regular basis
  - Marketing input based on a market analysis to understand how different population segments might use the software
  - Support input from the support staff who are aware of what works and what doesn't, what users like and dislike, what features generate questions, and what features are easy to use
- A set of questions should be answered during user analysis (see next slide)

# User Analysis Questions

- 1) Are the users trained professionals, technicians, clerical or manufacturing workers?
- 2) What level of formal education does the average user have?
- 3) Are the users capable of learning on their own from written materials or have they expressed a desire for classroom training?
- 4) Are the users expert typists or are they keyboard phobic?
- 5) What is the age range of the user community?
- 6) Will the users be represented predominately by one gender?
- 7) How are users compensated for the work they perform or are they volunteers?
- 8) Do users work normal office hours, or do they work whenever the job is required?
- 9) Is the software to be an integral part of the work users do, or will it be used only occasionally?
- 10) What is the primary spoken language among users?
- 11) What are the consequences if a user makes a mistake using the system?
- 12) Are users experts in the subject matter that is addressed by the system?
- 13) Do users want to know about the technology that sits behind the interface?

# Task Analysis and Modeling

- Task analysis strives to know and understand
  - The work the user performs in specific circumstances
  - The tasks and subtasks that will be performed as the user does the work
  - The specific problem domain objects that the user manipulates as work is performed
  - The sequence of work tasks (i.e., the workflow)
  - The hierarchy of tasks
- Use cases
  - Show how an end user performs some specific work-related task
  - Enable the software engineer to extract tasks, objects, and overall workflow of the interaction
  - Helps the software engineer to identify additional helpful features

# Content Analysis

- The display content may range from character-based reports, to graphical displays, to multimedia information
- Display content may be
  - Generated by components in other parts of the application
  - Acquired from data stored in a database that is accessible from the application
  - Transmitted from systems external to the application in question
- The format and aesthetics of the content (as it is displayed by the interface) needs to be considered
- A set of questions should be answered during content analysis (see next slide)

# Content Analysis Guidelines

- 1) Are various types of data assigned to consistent locations on the screen (e.g., photos always in upper right corner)?
- 2) Are users able to customize the screen location for content?
- 3) Is proper on-screen identification assigned to all content?
- 4) Can large reports be partitioned for ease of understanding?
- 5) Are mechanisms available for moving directly to summary information for large collections of data?
- 6) Is graphical output scaled to fit within the bounds of the display device that is used?
- 7) How is color used to enhance understanding?
- 8) How are error messages and warnings presented in order to make them quick and easy to see and understand?



# Work Environment Analysis

- Software products need to be designed to fit into the work environment, otherwise they may be difficult or frustrating to use
- Factors to consider include
  - Type of lighting
  - Display size and height
  - Keyboard size, height and ease of use
  - Mouse type and ease of use
  - Surrounding noise
  - Space limitations for computer and/or user
  - Weather or other atmospheric conditions
  - Temperature or pressure restrictions
  - Time restrictions (when, how fast, and for how long)

# User Interface Design

# Introduction

- User interface design is an iterative process, where each iteration elaborate and refines the information developed in the preceding step
- General steps for user interface design
  - 1) Using information developed during user interface analysis, define user interface objects and actions (operations)
  - 2) Define events (user actions) that will cause the state of the user interface to change; model this behavior
  - 3) Depict each interface state as it will actually look to the end user
  - 4) Indicate how the user interprets the state of the system from information provided through the interface
- During all of these steps, the designer must
  - Always follow the three golden rules of user interfaces
  - Model how the interface will be implemented
  - Consider the computing environment (e.g., display technology, operating system, development tools) that will be used

# Interface Objects and Actions

- Interface objects and actions are obtained from a grammatical parse of the use cases and the software problem statement
- Interface objects are categorized into types: source, target, and application
  - A source object is dragged and dropped into a target object such as to create a hardcopy of a report
  - An application object represents application-specific data that are not directly manipulated as part of screen interaction such as a list
- After identifying objects and their actions, an interface designer performs screen layout which involves
  - Graphical design and placement of icons
  - Definition of descriptive screen text
  - Specification and titling for windows
  - Definition of major and minor menu items
  - Specification of a real-world metaphor to follow

# Design Issues to Consider

- Four common design issues usually surface in any user interface
  - System response time (both length and variability)
  - User help facilities
    - When is it available, how is it accessed, how is it represented to the user, how is it structured, what happens when help is exited
  - Error information handling (more on next slide)
    - How meaningful to the user, how descriptive of the problem
  - Menu and command labeling (more on upcoming slide)
    - Consistent, easy to learn, accessibility, internationalization
- Many software engineers do not address these issues until late in the design or construction process
  - This results in unnecessary iteration, project delays, and customer frustration

# Guidelines for Error Messages

- The message should describe the problem in plain language that a typical user can understand
- The message should provide constructive advice for recovering from the error
- The message should indicate any negative consequences of the error (e.g., potentially corrupted data files) so that the user can check to ensure that they have not occurred (or correct them if they have)
- The message should be accompanied by an audible or visual cue such as a beep, momentary flashing, or a special error color
- The message should be non-judgmental
  - The message should never place blame on the user

An effective error message philosophy can do much to improve the quality of an interactive system and will significantly reduce user frustration when problems do occur

# Guidelines for Error Messages

- The message should describe the problem in plain language that a typical user can understand
- The message should provide constructive advice for recovering from the error
- The message should indicate any negative consequences of the error (e.g., potentially corrupted data files) so that the user can check to ensure that they have not occurred (or correct them if they have)
- The message should be accompanied by an audible or visual cue such as a beep, momentary flashing, or a special error color
- The message should be non-judgmental
  - The message should never place blame on the user

An effective error message philosophy can do much to improve the quality of an interactive system and will significantly reduce user frustration when problems do occur

# Questions for Menu Labeling and Typed Commands

- Will every menu option have a corresponding command?
- What form will a command take? A control sequence? A function key? A typed word?
- How difficult will it be to learn and remember the commands?
- What can be done if a command is forgotten?
- Can commands be customized or abbreviated by the user?
- Are menu labels self-explanatory within the context of the interface?
- Are submenus consistent with the function implied by a master menu item?



# User Interface Evaluation

# Design and Prototype Evaluation

- Before prototyping occurs, a number of evaluation criteria can be applied during design reviews to the design model itself
  - The amount of learning required by the users
    - Derived from the length and complexity of the written specification and its interfaces
  - The interaction time and overall efficiency
    - Derived from the number of user tasks specified and the average number of actions per task
  - The memory load on users
    - Derived from the number of actions, tasks, and system states
  - The complexity of the interface and the degree to which it will be accepted by the user
    - Derived from the interface style, help facilities, and error handling procedures

# Design and Prototype Evaluation (continued)

- Prototype evaluation can range from an informal test drive to a formally designed study using statistical methods and questionnaires
- The prototype evaluation cycle consists of prototype creation followed by user evaluation and back to prototype modification until all user issues are resolved
- The prototype is evaluated for
  - Satisfaction of user requirements
  - Conformance to the three golden rules of user interface design
  - Reconciliation of the four models of a user interface

# Example User Interfaces

# Summary

- Introduction
- Golden rules of user interface design
- Reconciling four different models
- User interface analysis
- User interface design
- User interface evaluation