Slide 3

background on interface design:

Interface Design Focuses on Three Main Areas:

1. **Interfaces Between Software Components**:
   - Designing how different parts of the software communicate with each other.
2. **Interfaces Between Software and Nonhuman Systems**:
   - Designing how the software interacts with other systems that produce or consume information (e.g., databases, sensors).
3. **Interfaces Between Humans and Computers**:
   - Designing how humans interact with the software, typically through graphical user interfaces (GUIs).

Graphical User Interfaces (GUIs):

- GUIs have improved the user experience by making interfaces more visually intuitive.
- Despite these improvements, some GUIs are still:
  - Difficult to learn
  - Hard to use
  - Confusing
  - Counterintuitive
  - Unforgiving
  - Frustrating

User Interface Analysis and Design:

- This involves studying how people interact with technology.
- The goal is to create interfaces that are easy to use and understand.

==

Slide 4

Spiral Process for User Interface Development:

1. **Interface Analysis**:
   - **User Analysis**: Understand the profile of the users who will use the system.
   - **Task Analysis**: Identify what tasks users need to perform.
   - **Environment Analysis**: Study the work environment where the system will be used.
   - **System Function Models**: Look at how the system functions from the user's perspective.
   - **Human- and Computer-Oriented Tasks**: Determine the tasks required by both the user and the computer to achieve the system's goals.
2. **Interface Design**:
   - **Interface Objects and Actions**: Define the elements (buttons, menus, etc.) and actions that users will interact with.
   - **Screen Representations**: Design how these elements will appear on the screen.
   - **Usability Goals**: Ensure the design meets all usability objectives for the system.
3. **Interface Construction**:
   - **Prototype Creation**: Start with a prototype to test how users interact with the interface.
   - **Development Tools**: Use tools to build the complete interface.
4. **Interface Validation**:
   - **Task Implementation**: Check that the interface correctly supports every user task.
   - **Task Variations**: Ensure the interface can handle different ways users might perform tasks.
   - **User Requirements**: Verify the interface meets all general user needs.
   - **Ease of Use and Learning**: Assess how easy the interface is to use and learn.
   - **User Acceptance**: Determine if users find the interface helpful in their work.

==

Slide 5,6,7

Golden Rules of User Interface Design simplified:

1. Place the User in Control

- **Flexible Interaction Modes**:
  - Design interactions so users are not forced into unnecessary actions.
  - Allow users to switch between tasks easily (e.g., spell check, edit text).
- **Flexible Interaction Methods**:
  - Enable actions to be performed using different methods (keyboard, mouse, voice).
- **Interruptible and Undoable Actions**:
  - Allow users to pause tasks without losing progress.
  - Provide an "undo" function for reversing actions.

2. Streamline Interaction and Customization

- **Skill Level Adaptation**:
  - Allow advanced users to speed up interactions using macros.
  - Enable users to customize the interface.
- **Hide Technical Details**:
  - Users should not need to handle operating system commands or technical operations.
  - These operations should be managed in the background.
- **Direct Interaction with On-Screen Objects**:
  - Make on-screen objects interactive in a way that feels natural (e.g., stretching a rectangle, pressing a button).

Summary

These rules aim to make user interfaces intuitive and user-friendly by giving users control, providing flexibility, hiding technical complexity, and enabling direct interaction with on-screen elements.

==

Slide 8,9

Reduce the User's Memory Load

1. **Reduce Demand on Short-Term Memory**:
   - Provide visual cues to help users remember past actions and results, reducing the need to rely on memory.
2. **Establish Meaningful Defaults**:
   - Use sensible default values that suit most users but allow for customization.
   - Allow users to easily reset any value to its default.
3. **Define Intuitive Shortcuts**:
   - Offer shortcuts (like control or alt combinations) that are easy to remember, often using the first letter of the action (e.g., Ctrl + C for copy).
4. **Use Real-World Metaphors for Visual Layout**:
   - Design the screen layout to include visual elements that users can relate to real-world actions (e.g., a trash can icon for deleting files).
5. **Disclose Information Progressively**:
   - Organize the interface in a hierarchical manner, guiding users step-by-step from general concepts to specific actions (e.g., selecting text format options leads to a detailed format dialog box).

Summary

To reduce the user's memory load, user interfaces should provide visual cues, sensible defaults, intuitive shortcuts, real-world metaphors, and hierarchical information disclosure. This makes the interface easier to use and understand without overloading the user's memory.

==

Slide 10,11

Make the Interface Consistent

1. **Consistent Presentation and Acquisition of Information**:
   - Organize all visual information according to a standard design that is used across all screens.
   - Use a limited and consistent set of input methods throughout the application.
   - Define and implement consistent navigation methods for moving from one task to another.
2. **Provide Context for the Current Task**:
   - Use indicators like window titles and color coding to help users know what they are working on.
   - Enable users to understand their current location, where they came from, and what options they have for moving to a new task.
3. **Maintain Consistency Across Applications**:
   - Ensure that all applications with complementary functions follow the same design rules.
   - Do not change established user interaction models unless absolutely necessary.
   - Continue using widely accepted interactive sequences (e.g., Alt + S to save a file) consistently throughout the application.

Summary

To make an interface consistent, maintain uniform design standards, use consistent input methods, and ensure navigation is predictable. Provide context for users and maintain consistency across similar applications and established interaction models. This helps users understand and use the interface more easily.

==

Slide 12,13

Reconciling Four Different Models

1. **Introduction**:
   - When designing a user interface, four different models need to be considered and aligned.
2. **The Four Models**:
   - **User Profile Model**:
     - Created by a human engineer or software engineer.
     - Represents the characteristics and needs of the users.
   - **Design Model**:
     - Created by a software engineer.
     - Represents how the software is designed to function.
   - **Implementation Model**:
     - Created by the software implementers.
     - Represents how the software is actually built and coded.
   - **User's Mental Model**:
     - Developed by the user through their experience with the application.
     - Represents how the user thinks the application works.
3. **Role of the Interface Designer**:
   - The interface designer's job is to align these models.
   - Create a consistent and coherent representation of the interface that meets the needs of all models.

==

Slide 14

User Profile Model

1. **Purpose**:
   - Establishes the characteristics of the end-users of the system.
2. **User Characteristics**:
   - **Age, Gender, Physical Abilities**: Differences in user demographics.
   - **Education, Cultural/Ethnic Background**: Variations in user backgrounds.
   - **Motivation, Goals, Personality**: Different user intentions and traits.
3. **Types of Knowledge**:
   - **Syntactic Knowledge**:
     - Mechanics of using the interface (e.g., how to navigate and interact).
   - **Semantic Knowledge**:
     - Understanding the application's functions, the meaning of inputs and outputs, and overall objectives.
4. **User Categories**:
   - **Novices**:
     - Little to no knowledge of the system or application.
     - Basic general computer skills.
   - **Knowledgeable, Intermittent Users**:
     - Good understanding of the system's functions.
     - Low recall of how to use the interface.
   - **Knowledgeable, Frequent Users**:
     - Deep understanding of both the system's functions and interface mechanics (power users).
     - Prefer shortcuts and efficient ways to operate.

==

Slide 15

Design Model

1. **Source**:
   - **Derived from the Analysis Model**:
     - It comes from analyzing the requirements of the system.
2. **Components**:
   - **Data Representation**:
     - How data is organized and managed in the system.
   - **Architectural Representation**:
     - The overall structure and organization of the system.
   - **Interface Representation**:
     - How users will interact with the system.
   - **Procedural Representation**:
     - The processes and workflows within the system.
3. **Constraints**:
   - **Requirements Specification**:
     - The design model must follow the requirements that describe who will use the system and how it should work.
4. **Importance**:
   - **Incidental Yet Crucial**:
     - While it may seem secondary to other parts of the design model, the interface design can be just as important as data, architecture, and procedures.

==

Slide 16

Implementation Model

1. **Components**:
   - **Look and Feel of the Interface**:

        - The visual appearance and interaction style of the interface.
    - **Supporting Information**:
        - Includes resources like books, videos, and help files that explain how to use the system.
2. **Objective**:
    - **Align with User's Mental Model**:
        - Make the interface feel intuitive and familiar to users, so they are comfortable and can use it effectively.
3. **Function**:
    - **Translation of the Design Model**:
        - Realizes the design model by incorporating details from the user profile model and the user's mental model.

Summary

The Implementation Model combines the visual and interactive aspects of the interface with supporting resources. It aims to match the user's expectations and understanding, making the software easy and comfortable to use. It translates the design into a practical, user-friendly interface.

==

Slide 17

User's Mental Model

1. **Definition**:
    - **User's System Perception**:
        - How users think and perceive the system in their minds.
2. **Components**:
    - **Image of the System**:
        - The understanding and expectations users have about how the system works.
3. **Factors Affecting Accuracy**:
    - **User's Profile**:
        - The user's background, knowledge, and experience with similar software.
    - **Familiarity with the Application**:
        - How well users know the specific software and its functions.

Summary

The User's Mental Model is the perception and understanding that users have about a system. It is influenced by their background, knowledge, and familiarity with the software, and it affects how intuitively they can use the system.

==

Slide 19

Elements of the User Interface

To effectively analyze a user interface, you need to study and understand four key elements:

1. **Users**:
    - **Who They Are**:
        - The people who will interact with the system through the interface.
2. **Tasks**:
    - **What They Do**:
        - The activities and tasks that users need to perform to accomplish their work.
3. **Content**:
    - **What They See**:
        - The information and elements that are displayed on the interface.

4. **Work Environment**:
    - **Where They Work**:
        - The physical and social environment in which users perform their tasks.

Summary

In user interface analysis, focus on understanding the users, their tasks, the content presented, and the work environment to design a user-friendly interface.

==

Slide 20

User Analysis

1. **Objective**:
    - Align the user's mental model with the design model by understanding:
        - **Who the users are**.
        - **How they use the system**.
2. **Sources of Information**:
    - **User Interviews**:
        - Direct conversations with end users to understand their needs and experiences.
    - **Sales Input**:
        - Feedback from salespeople who interact with customers and users regularly.
    - **Marketing Input**:
        - Insights from market analysis to understand usage patterns across different population segments.
    - **Support Input**:
        - Feedback from support staff about what works well, common issues, user preferences, and features that generate questions.
3. **Key Questions to Answer**:
    - Gather detailed information about users and their interactions with the system.

Summary

User analysis involves understanding who the users are and how they use the system by gathering information from various sources like user interviews, sales, marketing, and support staff. The goal is to align the user's mental model with the design model to create an intuitive and effective interface.

==

Slide 21

User Analysis Questions

1. **User Background**:
    - **Profession**:
        - Are users trained professionals, technicians, clerical, or manufacturing workers?
    - **Education**:
        - What is the average level of formal education of the users?
2. **Learning and Training**:
    - **Self-Learning**:
        - Can users learn on their own from written materials, or do they prefer classroom training?
    - **Typing Skills**:
        - Are users expert typists or do they avoid using the keyboard?
3. **Demographics**:
    - **Age Range**:
        - What is the age range of the user community?
    - **Gender**:

    - Will the user base be predominantly one gender?

4. **Work and Usage Patterns**:
    - **Compensation**:
        - Are users paid for their work or are they volunteers?
    - **Work Hours**:
        - Do users work regular office hours or at varying times as needed?
    - **Frequency of Use**:
        - Is the software a key part of their job or used occasionally?
5. **Language and Mistakes**:
    - **Primary Language**:
        - What is the main language spoken by users?
    - **Mistake Consequences**:
        - What happens if a user makes a mistake using the system?
6. **Expertise and Interest**:
    - **Subject Matter Expertise**:
        - Are users experts in the topic the system addresses?
    - **Interest in Technology**:
        - Do users want to understand the technology behind the interface?

Summary

These questions help understand the users' background, learning preferences, demographics, work patterns, language, potential consequences of mistakes, and their expertise and interest in technology. This information is crucial for designing a user-friendly interface tailored to the users' needs and characteristics.

===

Slide 22

Task Analysis and Modeling

1. **Objective**:
    - **Understand User's Work**:
        - Know what work users do in specific situations.
    - **Identify Tasks and Subtasks**:
        - Determine all the tasks and subtasks users perform.
2. **Key Elements**:
    - **Problem Domain Objects**:
        - Identify the specific items users interact with while working.
    - **Workflow**:
        - Understand the sequence of tasks (how tasks flow from one to another).
    - **Task Hierarchy**:
        - Recognize how tasks are organized, from main tasks to subtasks.
3. **Use Cases**:
    - **Purpose**:
        - Show how users perform specific tasks related to their work.
    - **Benefits**:
        - Help software engineers understand tasks, objects, and workflows.
        - Identify additional features that might be helpful for users.

Summary

Task analysis and modeling involve understanding the tasks users perform, the objects they interact with, the workflow of these tasks, and how tasks are organized. Use cases are used to illustrate these tasks and interactions, helping software engineers design more user-friendly and efficient systems.

===

Slide 23

Content Analysis

1. **Types of Display Content**:
   - **Variety**:
     - Content can include text reports, graphics, or multimedia information.
2. **Content Sources**:
   - **Internal Generation**:
     - Content created by other parts of the application.
   - **Database**:
     - Content fetched from a database accessible by the application.
   - **External Systems**:
     - Content received from systems outside the application.
3. **Display Format and Aesthetics**:
   - **Considerations**:
     - How the content looks and is formatted when displayed in the interface.
4. **Questions for Content Analysis**:
   - **Purpose**:
     - To ensure the content is presented in a clear, attractive, and useful way.

Summary

Content analysis involves understanding the types of content displayed, where it comes from, and how it should be formatted and presented in the user interface. This ensures that the content is clear, visually appealing, and useful to the user.

==

Slide 24

Content Analysis Guidelines

1. **Consistent Placement**:
   - **Data Locations**:
     - Are different types of data always in the same place on the screen? (e.g., photos always in the upper right corner)
2. **Customization**:
   - **User Preferences**:
     - Can users change where the content appears on the screen?
3. **Identification**:
   - **Labeling**:
     - Is all content clearly labeled on the screen?
4. **Report Handling**:
   - **Large Reports**:
     - Can big reports be split up to make them easier to understand?
5. **Summary Information**:
   - **Quick Access**:
     - Are there ways to quickly get to summary information in large data sets?
6. **Graphical Output**:
   - **Scaling**:
     - Are graphics resized to fit the display device properly?
7. **Color Usage**:
   - **Enhancement**:
     - Is color used to make the content easier to understand?
8. **Error Messages**:
   - **Visibility**:
     - Are error messages and warnings easy to see and understand?

Summary

These guidelines ensure that content is consistently placed, customizable, clearly labeled, easy to navigate, properly sized, visually enhanced with color, and that error messages are prominent and understandable. This helps create a user-friendly interface.

==

Slide 25

Work Environment Analysis

1. **Purpose**:
   - **Fit with Work Environment**:
     - Ensure software is designed to suit the user's work environment to avoid frustration.
2. **Factors to Consider**:
   - **Lighting**:
     - Type of lighting in the workspace.
   - **Display**:
     - Size and height of the computer screen.
   - **Keyboard**:
     - Size, height, and ease of use of the keyboard.
   - **Mouse**:
     - Type and ease of use of the mouse.
   - **Noise**:
     - Surrounding noise levels.
   - **Space**:
     - Space available for the computer and user.
   - **Weather and Atmosphere**:
     - Weather conditions and atmospheric factors like temperature and pressure.
   - **Time Restrictions**:
     - When the software will be used, how fast it needs to operate, and how long it will be used.

Summary

Work environment analysis ensures that software is compatible with the physical and environmental conditions of the user's workspace. This includes factors like lighting, equipment ergonomics, noise levels, space, and time constraints, making the software more user-friendly and efficient.

==

Slide 27

User Interface Design Process

1. **Iterative Process:**
   - User interface design is done in steps, with each step building on the previous one.
2. **General Steps:**
   - **Define Interface Elements:**
     - Use information from analysis to determine user interface objects (like buttons) and actions (like clicking).
   - **Define Events:**
     - Identify user actions that will change the interface state (e.g., clicking a button changes the screen).
   - **Depict Interface States:**
     - Show how each state of the interface will look to the user.
   - **Interpret State Information:**
     - Explain how users will understand the current state of the system from the interface.
3. **Designer Considerations:**
   - **Follow the Three Golden Rules:**
     - Place the user in control.

        - Reduce the user's memory load.
        - Make the interface consistent.
    - **Implementation Modeling:**
        - Plan how the interface will be created.
    - **Consider Computing Environment:**
        - Think about the technology (like display type, operating system, and development tools) that will be used.

==

Slide 28

Interface Objects and Actions

1. **Finding Interface Objects and Actions**
    - Use use cases and problem statements to identify interface objects.
    - Categorize these objects as source, target, or application objects.
2. **Types of Interface Objects**
    - **Source Object:** An item that can be moved, such as dragging a report to a printer icon to print it.
    - **Target Object:** The destination where a source object is dropped.
    - **Application Object:** Data related to the application that isn't directly manipulated on the screen, like a list of items.
3. **Screen Layout Tasks**
    - Design and place icons.
    - Write descriptive text for the screen.
    - Title windows and define their purpose.
    - Organize major and minor menu items.
    - Use real-world metaphors to make the interface intuitive (e.g., a trash can icon for deleting files).

Images Overview

1. **Interface Objects and Actions Diagram**
    - The diagram depicts the categorization of interface objects and examples of how they interact.
2. **Screen Layout Elements**
    - This image likely shows the placement of icons, descriptive text, window titles, menu items, and real-world metaphors in a user interface.

The content of the images provides visual guidance for understanding the categorization and interaction of interface objects, as well as how to lay out the user interface elements effectively.

==

Slide 29

Design Issues to Consider

When designing a user interface, four key issues commonly arise:

1. **System Response Time**
    - How long the system takes to respond.
    - Consistency in response time (avoiding high variability).
2. **User Help Facilities**
    - Availability: When and where help can be accessed.
    - Access: How users can reach help (e.g., help button, F1 key).
    - Representation: How help is presented (text, video, interactive).
    - Structure: How help content is organized.

  - Exit: What happens when the user exits the help system.
3. **Error Information Handling**
   - Meaningfulness: How useful the error messages are to the user.
   - Descriptiveness: How clearly the error messages explain the problem.
4. **Menu and Command Labeling**
   - Consistency: Labels should be consistent throughout the interface.
   - Ease of Learning: Labels should be intuitive and easy to understand.
   - Accessibility: Menus and commands should be accessible to all users, including those with disabilities.
   - Internationalization: Labels should be adaptable to different languages and cultures.

Pitfalls in Addressing Design Issues

- Many software engineers leave these considerations until late in the design or construction process, which can lead to:
  - Unnecessary iterations.
  - Project delays.
  - Customer frustration.

Simplifying these design issues early in the process can help prevent these problems and lead to a more efficient and user-friendly interface.

==

Slide 30,31

Guidelines for Error Messages

1. **Plain Language**
   - Use simple, clear language that any user can understand.
2. **Constructive Advice**
   - Provide helpful steps to recover from the error.
3. **Negative Consequences**
   - Inform the user of any potential issues caused by the error (e.g., data corruption) so they can check or fix them.
4. **Audible/Visual Cues**
   - Use beeps, flashing, or special colors to draw attention to the error message.
5. **Non-Judgmental Tone**
   - Avoid blaming the user. Keep the tone neutral and supportive.

By following these guidelines, error messages can be more helpful, less frustrating, and easier for users to understand and act upon.

==

Slide 32

Questions for Menu Labeling and Typed Commands

1. **Menu and Command Correspondence**
   - Will every menu option have a matching command?
2. **Command Form**
   - What will commands look like? Will they be control sequences, function keys, or typed words?
3. **Learning Difficulty**
   - How hard will it be to learn and remember these commands?
4. **Forgotten Commands**
   - What can users do if they forget a command?
5. **Customization**
   - Can users customize or abbreviate commands?

6. **Self-Explanatory Labels**
    - Are menu labels clear and easy to understand within the interface context?
7. **Consistency of Submenus**
    - Do submenus logically relate to the master menu item?

By considering these questions, you can create a user-friendly interface that is easy to navigate and use.

==

Slide 34,35

Design and Prototype Evaluation

Design Review Criteria

Before creating a prototype, evaluate the design model with these criteria:

1. **Learning Required by Users**
    - How much do users need to learn? This depends on the complexity of the specifications and interfaces.
2. **Interaction Time and Efficiency**
    - How long and efficient is the interaction? This is based on the number of user tasks and the average actions per task.
3. **Memory Load on Users**
    - How much memory load is required from users? This is derived from the number of actions, tasks, and system states.
4. **Interface Complexity and User Acceptance**
    - How complex is the interface and will users accept it? This is influenced by the interface style, help facilities, and error handling procedures.

Prototype Evaluation

Prototype evaluation can range from informal tests to formal studies using statistical methods and questionnaires. The evaluation process involves:

1. **Prototype Creation**
    - Create the prototype based on the design.
2. **User Evaluation**
    - Have users test the prototype and provide feedback.
3. **Prototype Modification**
    - Modify the prototype based on user feedback and repeat the cycle until all issues are resolved.

Evaluation Goals

Evaluate the prototype for:

1. **User Requirements Satisfaction**
    - Does the prototype meet the user's needs?
2. **Conformance to UI Design Rules**
    - Does it follow the three golden rules of user interface design?
3. **Reconciliation of User Interface Models**
    - Does it align with the four models of a user interface?

By following these steps, you ensure that the design and prototype are user-friendly and meet all necessary requirements.