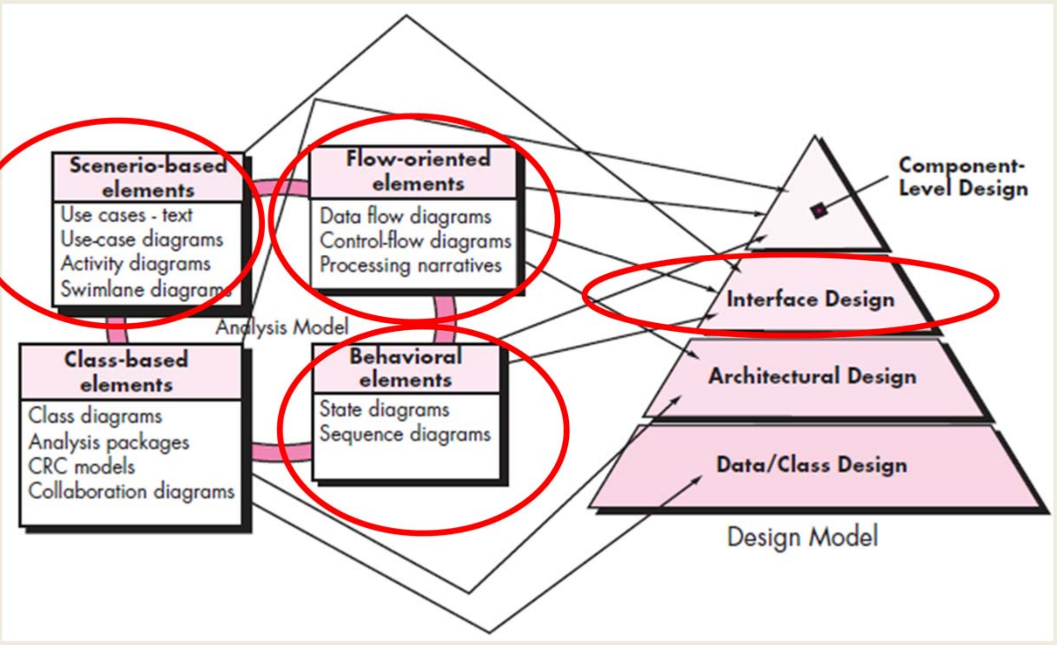
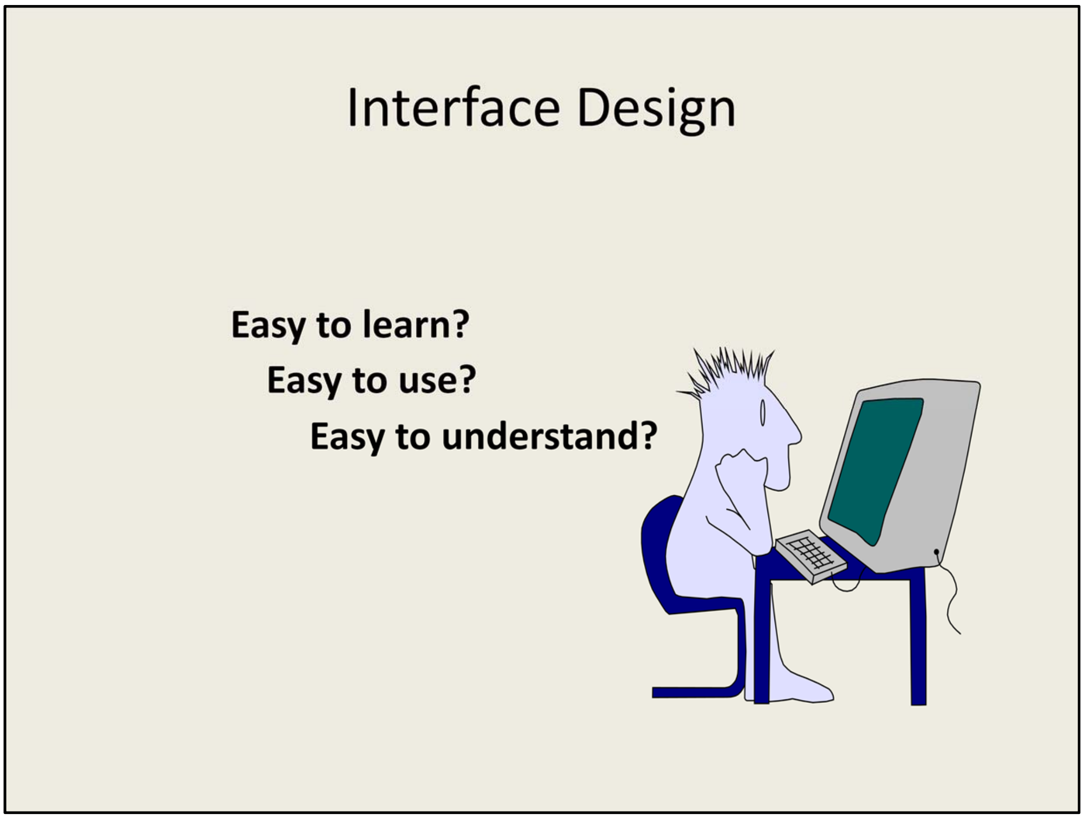
User interface design

Translation: analysis to design





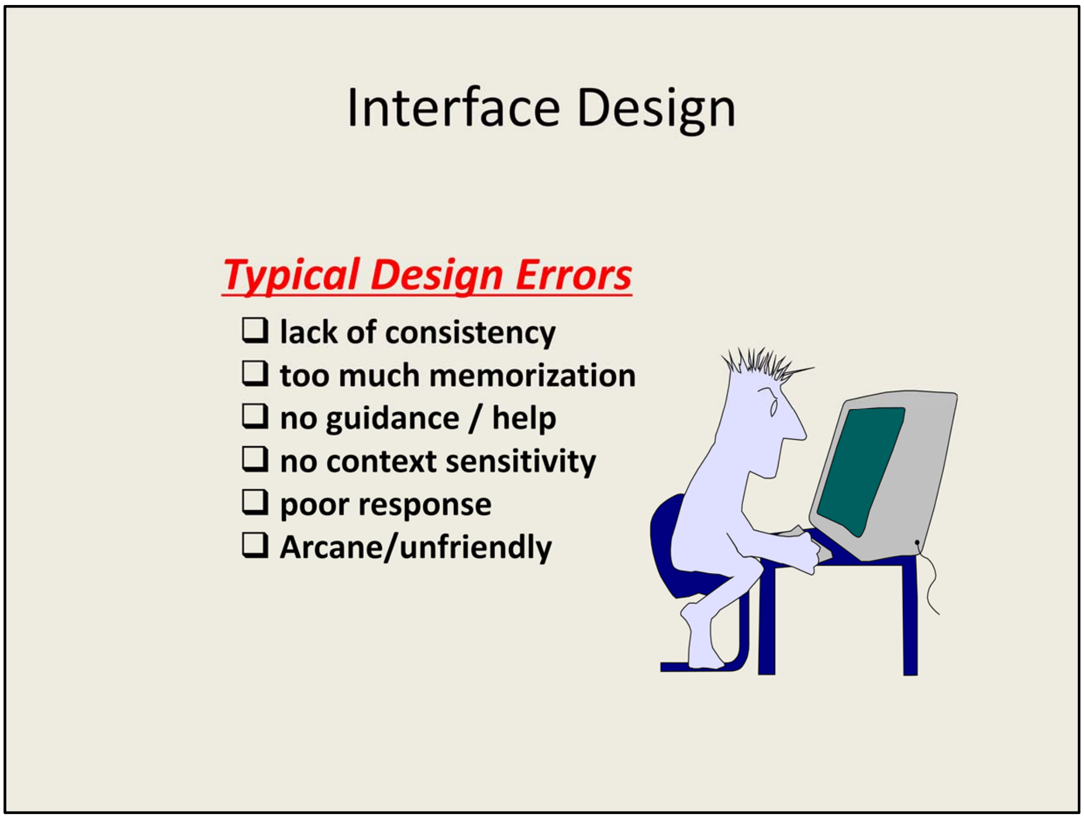
Interface Design

Easy to learn?

Easy to use?

Easy to understand?

The goal of designing user interfaces / questions to ask when doing user interface design



Interface Design

Typical Design Errors

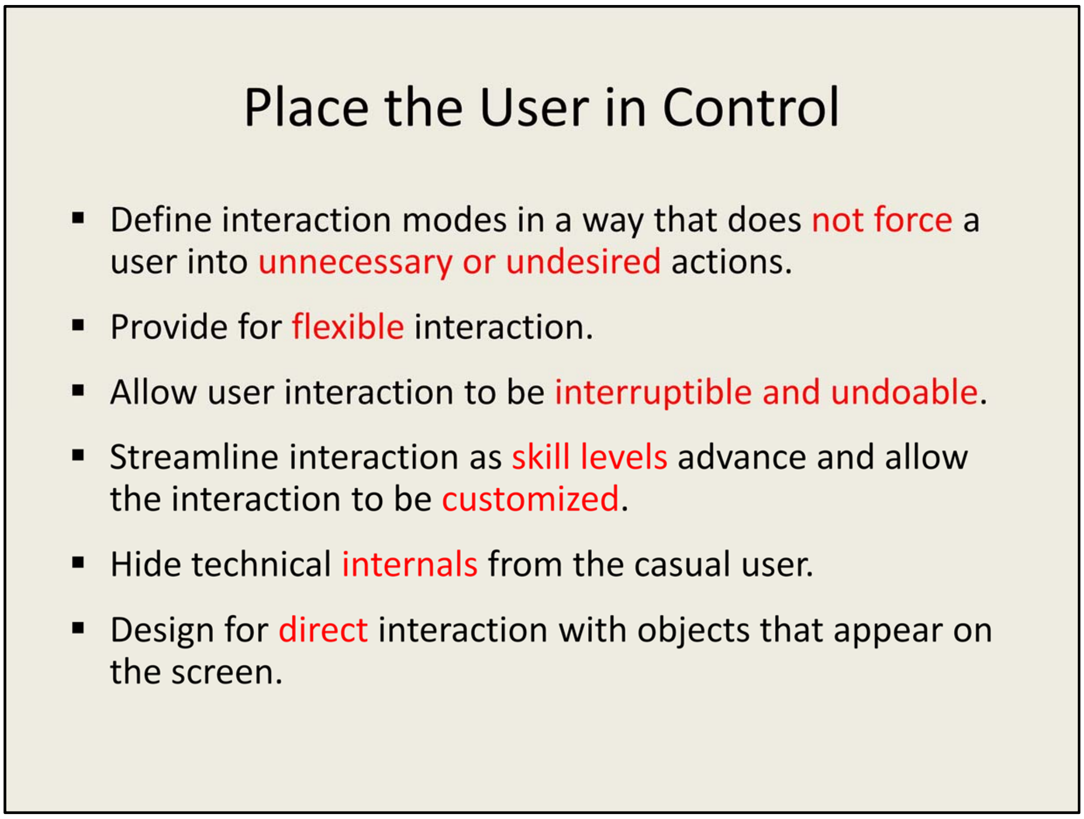
- ❑ lack of consistency
- ❑ too much memorization
- ❑ no guidance / help
- ❑ no context sensitivity
- ❑ poor response
- ❑ Arcane/unfriendly

Lack of consistency: use red color on an OK button in sphere shape in one dialog and blue color in rectangular shape in another dialog

Unfriendly: hard to navigate

Golden Rules

- Place the user in control
- Reduce the user’s memory load
- Make the interface consistent



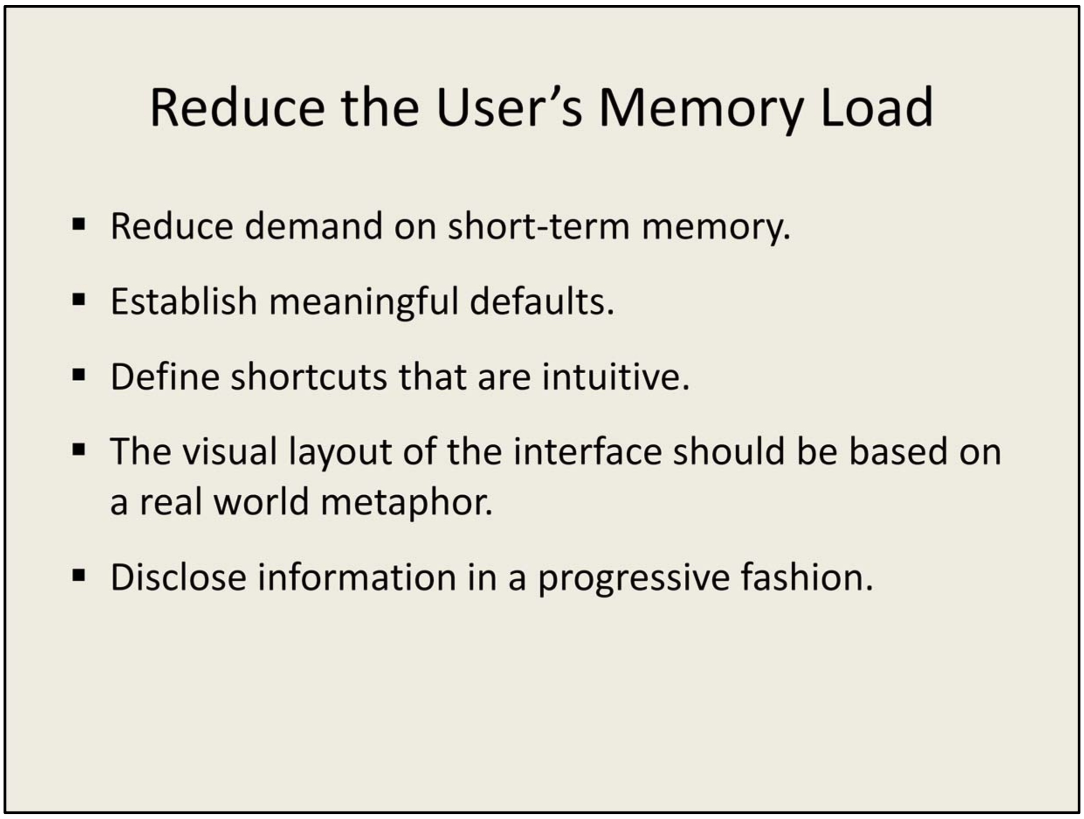
Place the User in Control

- Define interaction modes in a way that does not force a user into unnecessary or undesired actions.
- Provide for flexible interaction.
- Allow user interaction to be interruptible and undoable.
- Streamline interaction as skill levels advance and allow the interaction to be customized.
- Hide technical internals from the casual user.
- Design for direct interaction with objects that appear on the screen.

a number of design principles that allow the user to maintain control / how can we design interface in a way that places the user in control

Examples:

1. User should be able to enter or exit a mode easily (e.g., switch easily)
2. Multiple choices for text input (handwrite, type, voice, etc.)
3. Typing and then switching to a game app, when getting back, the words typed before should not be lost
4. Define a macro for advanced users to perform a frequently used sequence of actions
5. High level operations instead of low level internal commands
6. If you show a ball on the screen, you should allow users to manipulate the ball (rather than a placeholder or static object)



Reduce the User’s Memory Load

- Reduce demand on short-term memory.
- Establish meaningful defaults.
- Define shortcuts that are intuitive.
- The visual layout of the interface should be based on a real world metaphor.
- Disclose information in a progressive fashion.

interface should be designed to reduce the requirement to remember past actions, inputs, and results

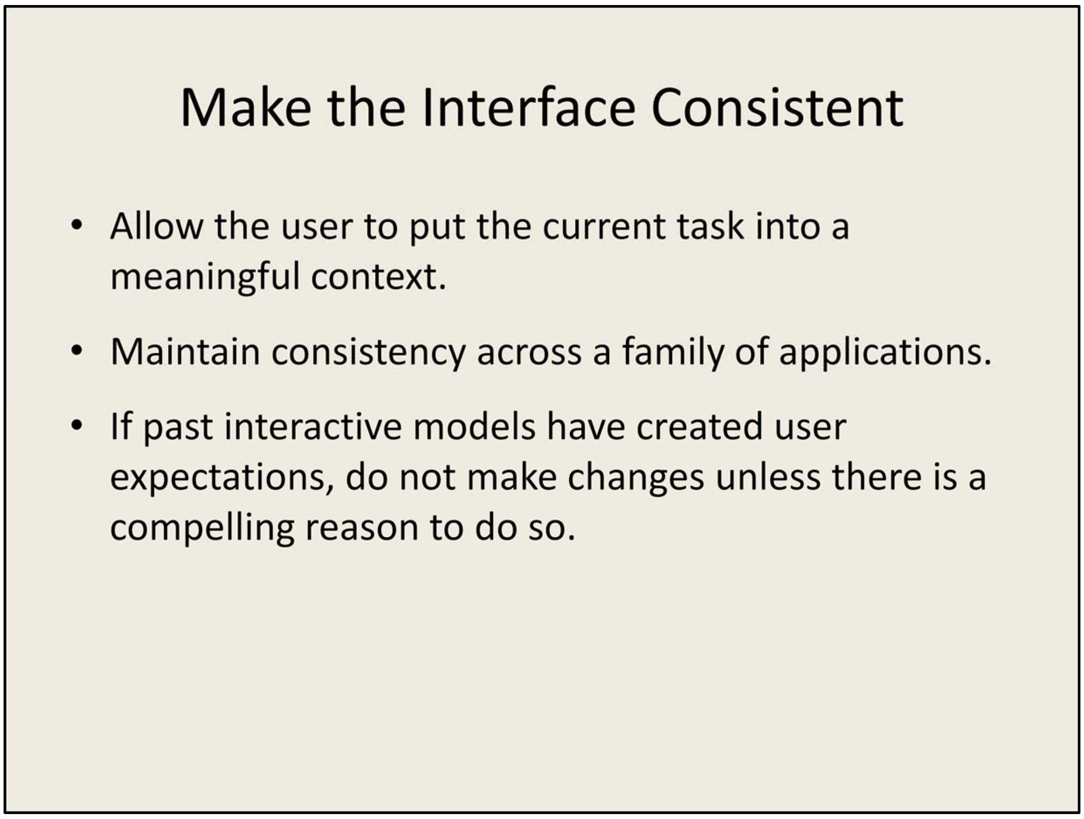
The initial set of defaults should make sense for the average user, but a user should be able to specify individual preferences. However,

a “reset” option should be available

Use Ctrl^S for saving and Ctrl^O for opening a file, rather than the other way around

An online form should look like an actual paper form which users have been familiar with.

Show text formatting options only when users select a sequence of words



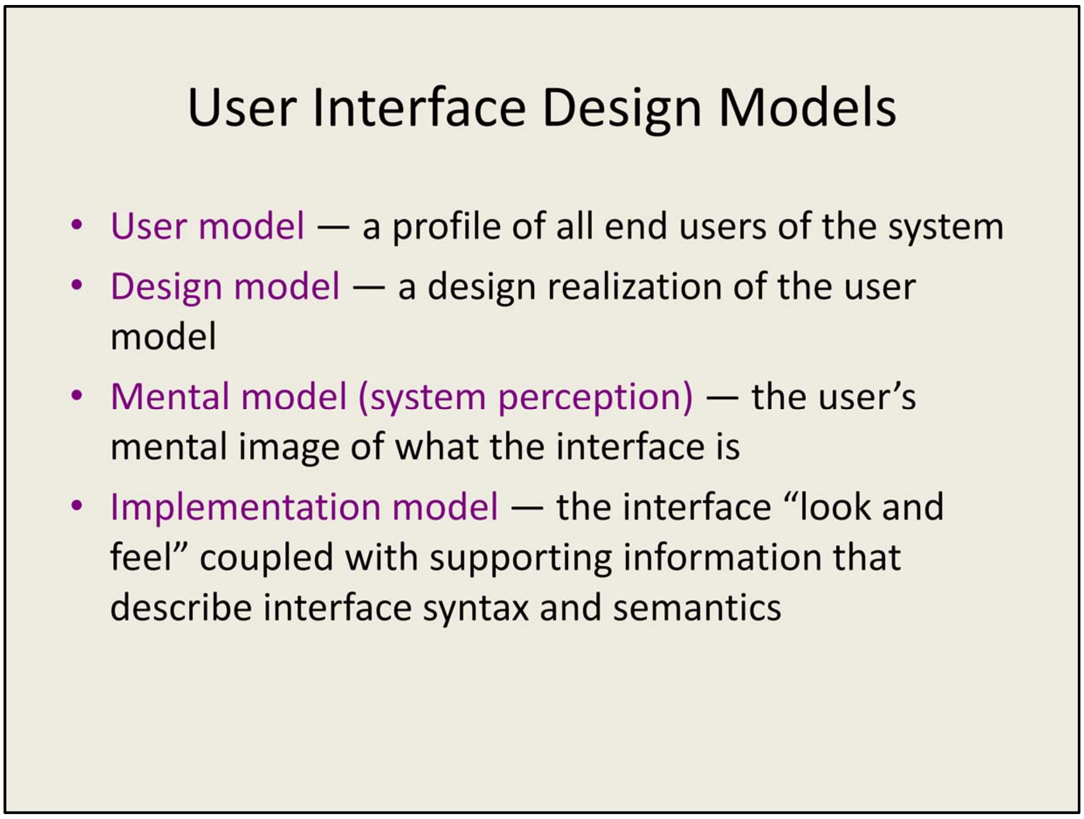
Make the Interface Consistent

- Allow the user to put the current task into a meaningful context.
- Maintain consistency across a family of applications.
- If past interactive models have created user expectations, do not make changes unless there is a compelling reason to do so.

Show a meaningful dialog title / hint the section of a long document the user is currently editing at the footnote

Same interaction design should be followed in a set of similar apps to be consistent.

Allowing user to delete a line by pressing the ‘Del’ key at a selected line, rather than changing it to a different bahavoir with this key or a different key for the same behavior, because users have been used to it



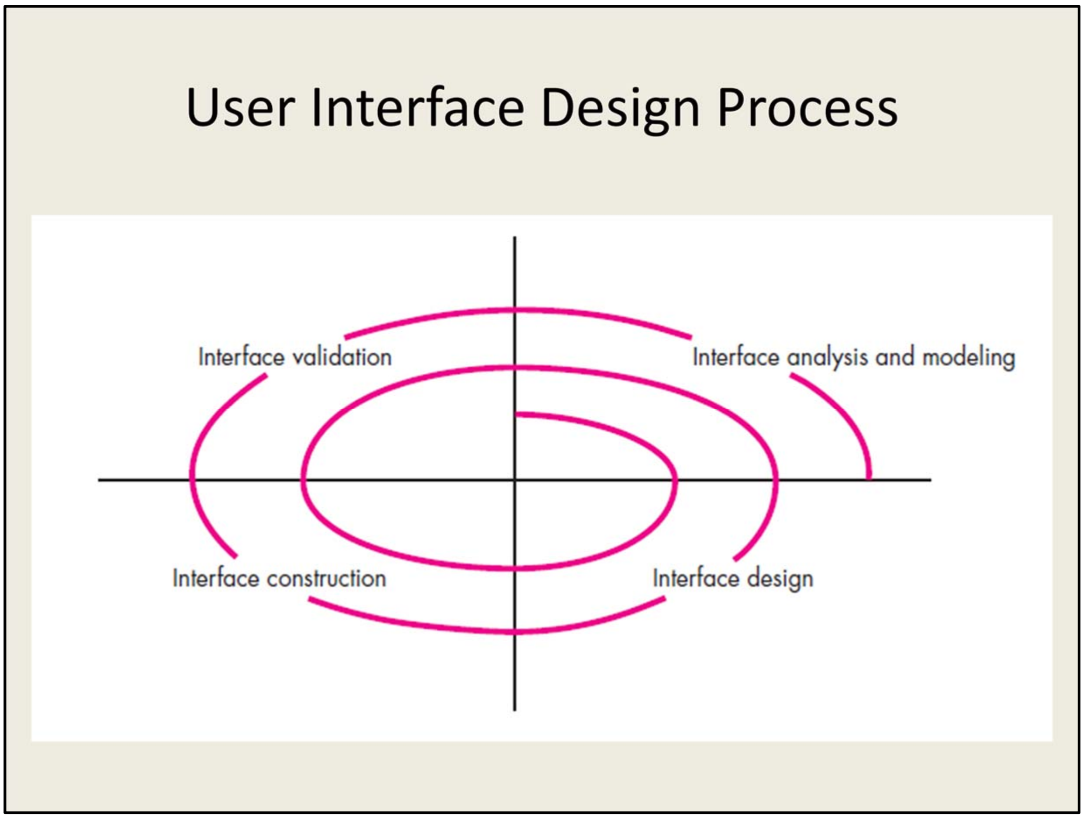
User Interface Design Models

- User model — a profile of all end users of the system
- Design model — a design realization of the user model
- Mental model (system perception) — the user's mental image of what the interface is
- Implementation model — the interface "look and feel" coupled with supporting information that describe interface syntax and semantics

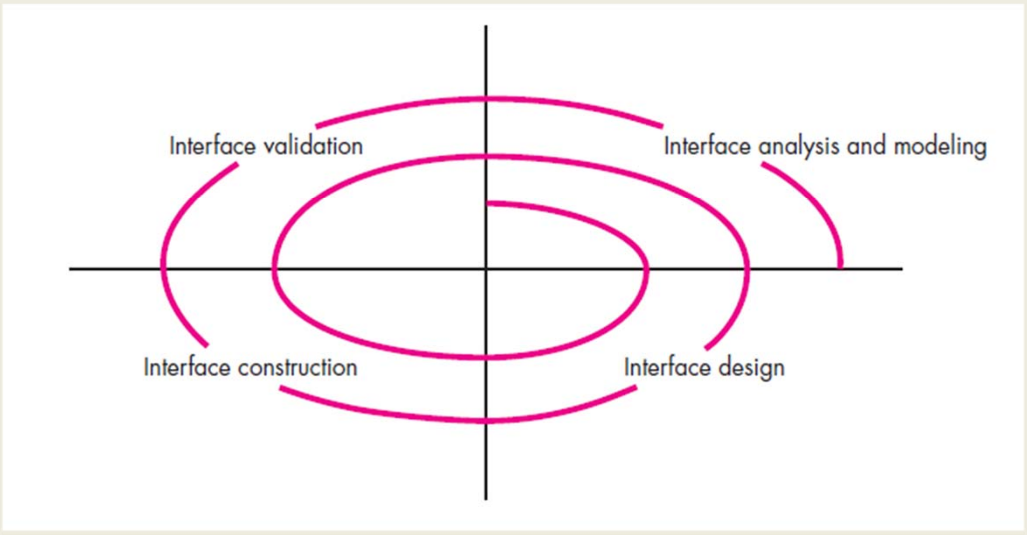
A human engineer (or the software engineer) establishes a *user model,* the
software engineer creates a *design model,* the end user develops a mental image that
is often called the user's *mental model* or the *system perception,* and the implementers of the system create an *implementation model*

Unfortunately, each of these models
may differ significantly. Your role, as an interface designer, is to reconcile these differences
and derive a consistent representation of the interface.

To build an effective user interface, "all design should begin with an understanding
of the intended users, including profiles of their age, gender, physical abilities, education,
cultural or ethnic background, motivation, goals and personality

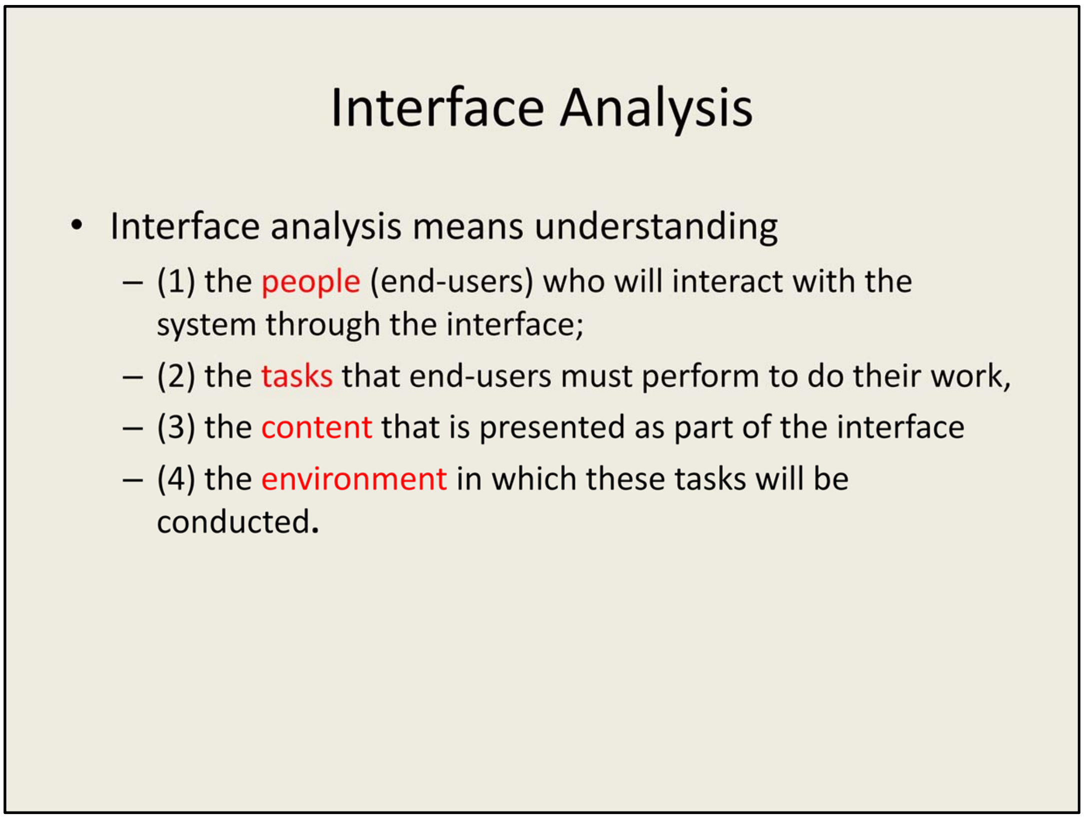


User Interface Design Process



the user interface analysis and design process begins at the interior of
the spiral and encompasses four distinct **framework** activities (1) interface
analysis and modeling, (2) interface design, (3) interface construction, and (4)
interface
validation. The spiral shown in Figure 11.1 implies that each of these tasks will
occur more than once, with each pass around the spiral representing additional
elaboration of requirements and the resultant design.

Interface construction normally begins with the creation of a prototype


Interface Analysis

- Interface analysis means understanding
  - (1) the people (end-users) who will interact with the system through the interface;
  - (2) the tasks that end-users must perform to do their work,
  - (3) the content that is presented as part of the interface
  - (4) the environment in which these tasks will be conducted**.**

A key tenet of all software engineering process models is this: *understand the problem*

before you attempt to design a solution. In the case of user interface design, understanding

the problem means understanding

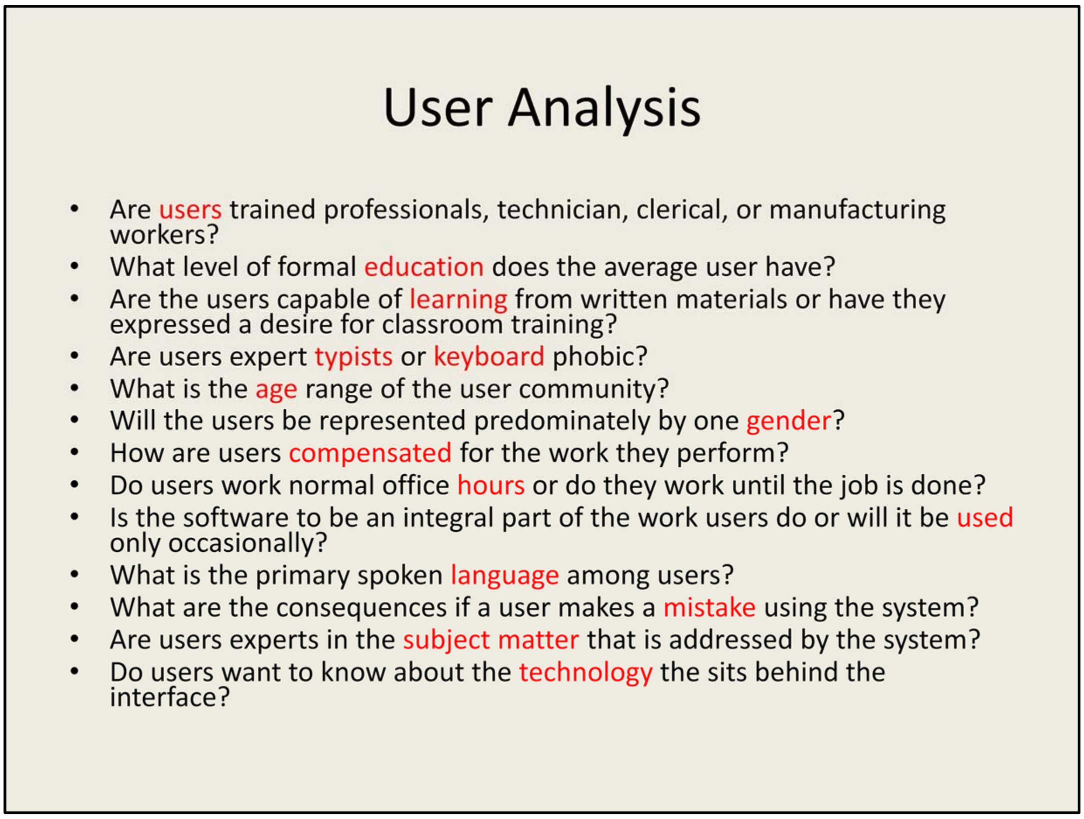
We will talk more about the first three.

For (4): In some applications the user interface for a computer-based system is placed in a

“user-friendly location” (e.g., proper lighting, good display height, easy keyboard

access), but in others (e.g., a factory floor or an airplane cockpit), lighting may be

suboptimal, noise may be a factor, a keyboard or mouse may not be an option.



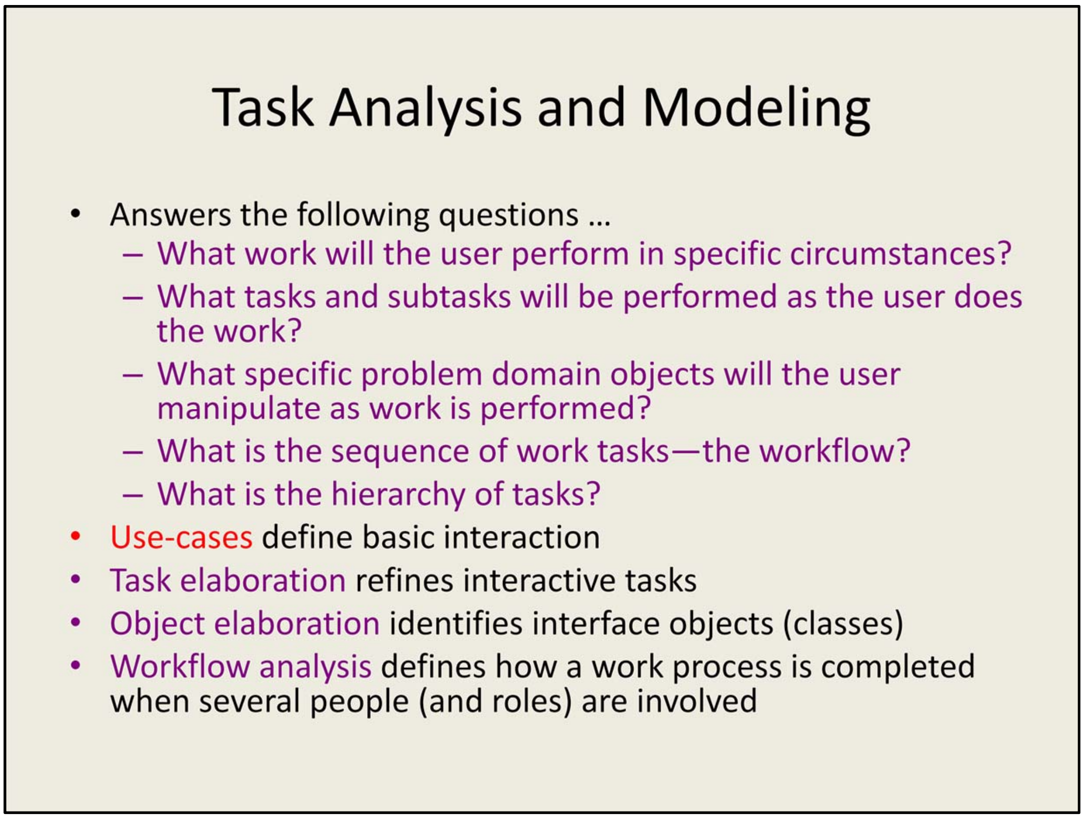
User Analysis

- Are users trained professionals, technician, clerical, or manufacturing workers?
- What level of formal education does the average user have?
- Are the users capable of learning from written materials or have they expressed a desire for classroom training?
- Are users expert typists or keyboard phobic?
- What is the age range of the user community?
- Will the users be represented predominately by one gender?
- How are users compensated for the work they perform?
- Do users work normal office hours or do they work until the job is done?
- Is the software to be an integral part of the work users do or will it be used only occasionally?
- What is the primary spoken language among users?
- What are the consequences if a user makes a mistake using the system?
- Are users experts in the subject matter that is addressed by the system?
- Do users want to know about the technology the sits behind the interface?

The only way that you can get the mental image and the design model to converge is to work to understand

the users themselves as well as how these people will use the system. Information

from a broad array of sources can be used to accomplish this: **User Interviews / Sales input / Marketing input/ Support input**



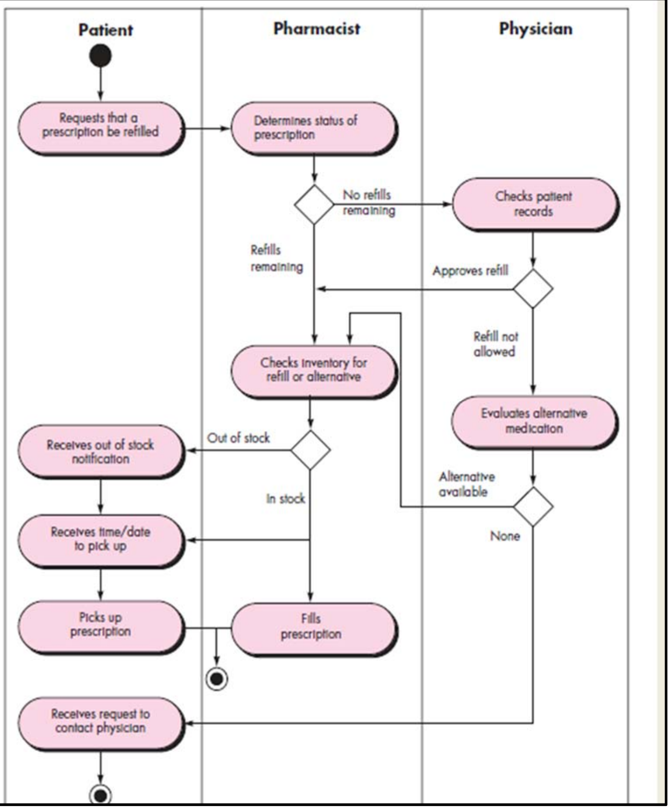
Task Analysis and Modeling

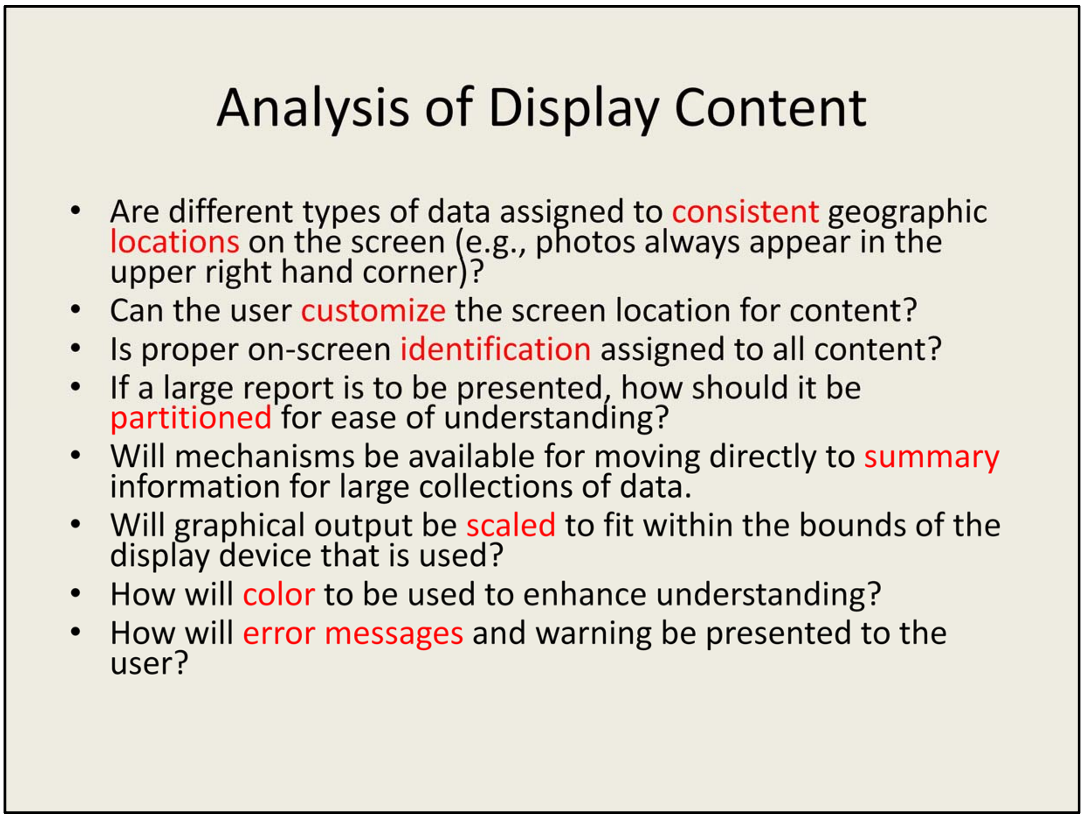
- Answers the following questions …
  - What work will the user perform in specific circumstances?
  - What tasks and subtasks will be performed as the user does the work?
  - What specific problem domain objects will the user manipulate as work is performed?
  - What is the sequence of work tasks—the workflow?
  - What is the hierarchy of tasks?
- Use-cases define basic interaction
- Task elaboration refines interactive tasks
- Object elaboration identifies interface objects (classes)
- Workflow analysis defines how a work process is completed when several people (and roles) are involved

To answer these questions, you must draw upon techniques that I have discussed earlier in this book, but in this instance, these techniques are applied to the user interface.

We consider only a small part of the work process: the situation that occurs when
a patient asks for a refill. Figure 11.2 presents a swimlane diagram that indicates the
tasks and decisions for each of the three roles noted earlier

- Swimlane Diagram


Patient
Pharmacist
Physician
Requests that a prescription be refilled
Determines status of prescription
No refills remaining
Checks patient records
Refills remaining
Approves refill
Refill not allowed
Checks inventory for refill or alternative
Evaluates alternative medication
Out of stock
Receives out of stock notification
Alternative available
In stock
None
Receives time/date to pick up
Picks up prescription
Fills prescription
Receives request to contact physician

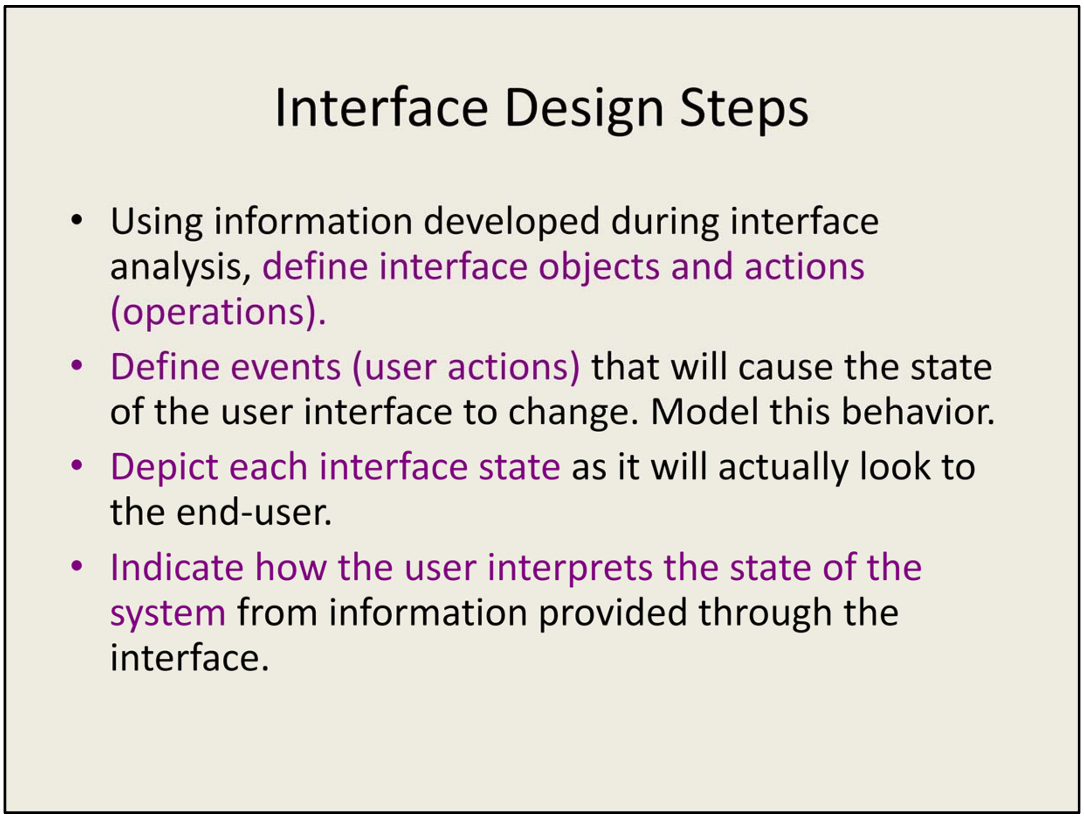


Analysis of Display Content

- Are different types of data assigned to consistent geographic locations on the screen (e.g., photos always appear in the upper right hand corner)?
- Can the user customize the screen location for content?
- Is proper on-screen identification assigned to all content?
- If a large report is to be presented, how should it be partitioned for ease of understanding?
- Will mechanisms be available for moving directly to summary information for large collections of data.
- Will graphical output be scaled to fit within the bounds of the display device that is used?
- How will color to be used to enhance understanding?
- How will error messages and warning be presented to the user?

The answers to these (and other) questions will help you to establish requirements

for content presentation.

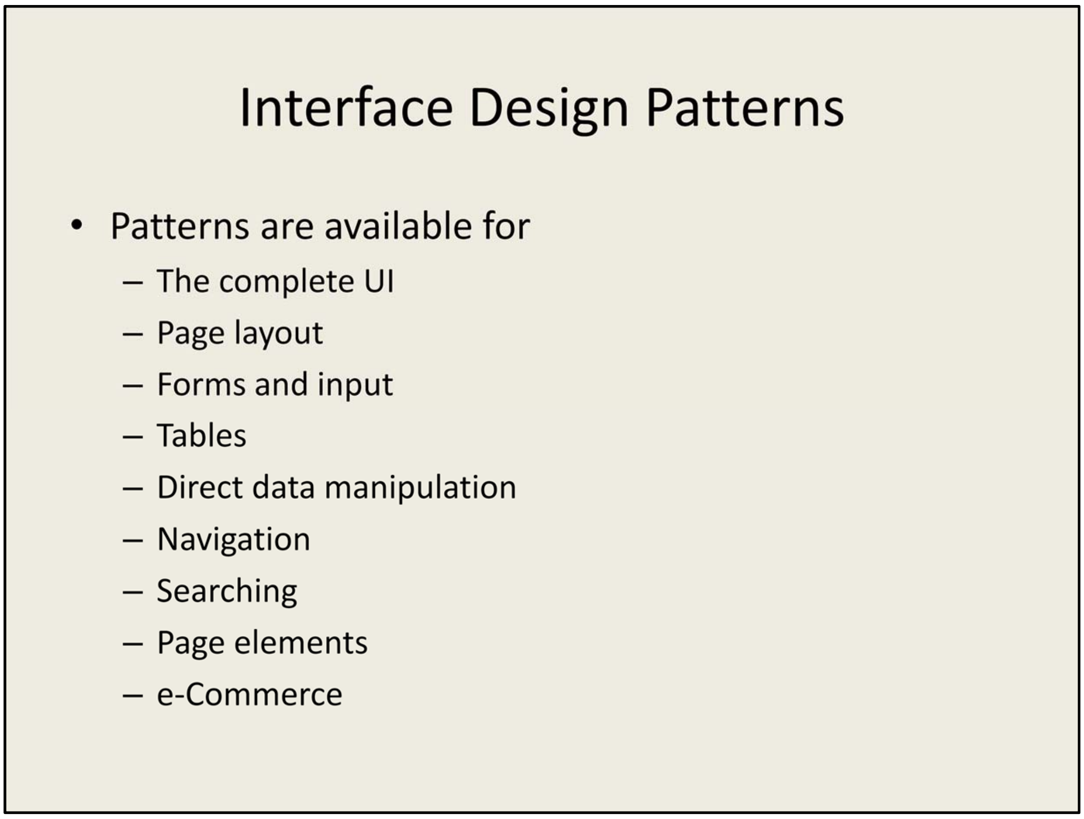


Interface Design Steps

- Using information developed during interface analysis, define interface objects and actions (operations).
- Define events (user actions) that will cause the state of the user interface to change. Model this behavior.
- Depict each interface state as it will actually look to the end-user.
- Indicate how the user interprets the state of the system from information provided through the interface.

Although many different user interface design models (e.g., [Nor86], [Nie00]) have

been proposed, all suggest some combination of the following steps:


Interface Design Patterns

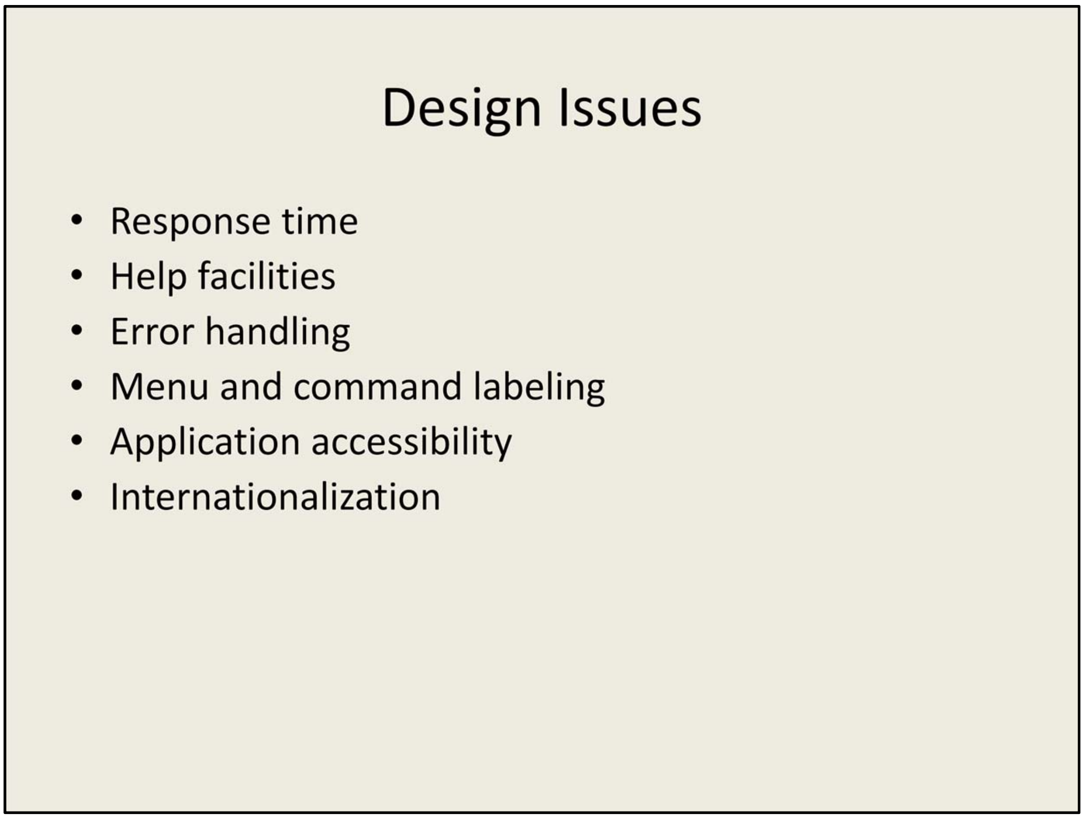
- Patterns are available for
  - The complete UI
  - Page layout
  - Forms and input
  - Tables
  - Direct data manipulation
  - Navigation
  - Searching
  - Page elements
  - e-Commerce

Graphical user interfaces have become so common that a wide variety of user interface

design patterns has emerged

As I noted earlier in this book, a design pattern is an abstraction that prescribes a design solution to a specific, well-bounded design

problem.



Design Issues

- Response time
- Help facilities
- Error handling
- Menu and command labeling
- Application accessibility
- Internationalization

As the design of a user interface evolves, four common design issues almost always
surface:

System response time is the primary complaint for many interactive
applications.

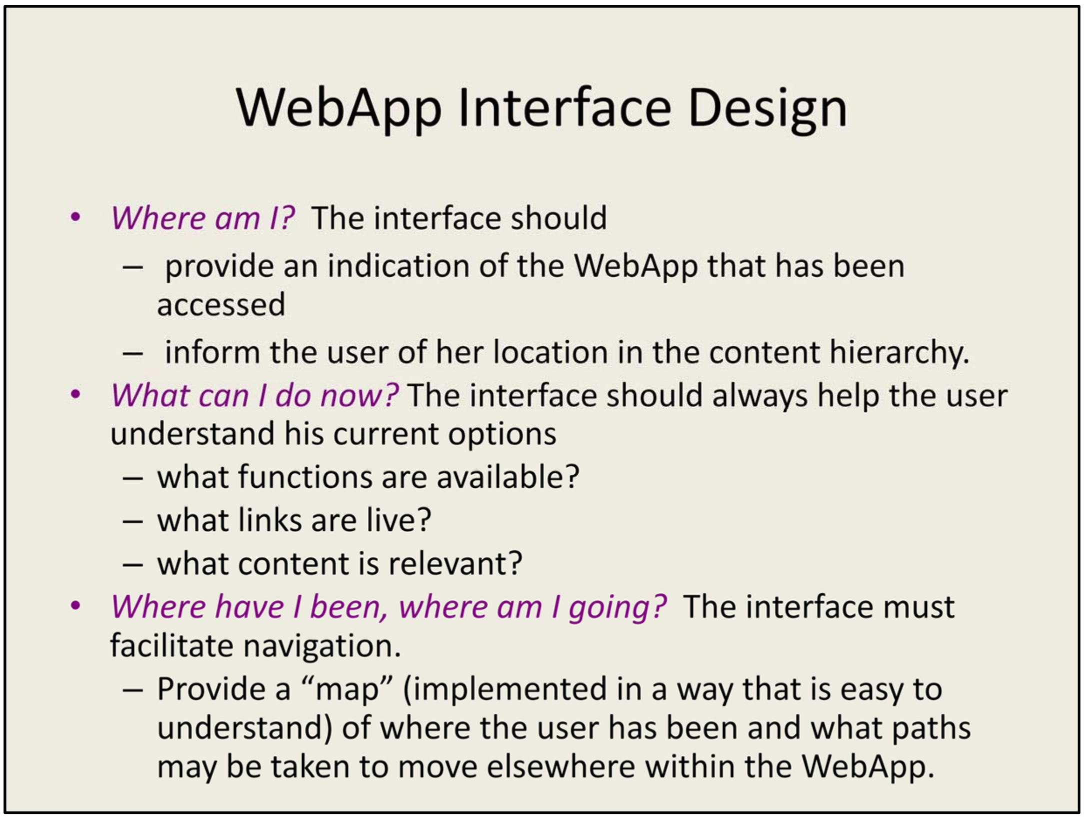
Almost every user of an interactive, computer-based system requires
help now and then

Error messages and warnings are “bad news” delivered to users
of interactive systems when something has gone awry

the use of window-oriented, point-andpick
interfaces has reduced reliance on typed commands, but some power-users
continue to prefer a command-oriented mode of interaction

Software engineers must ensure that interface design encompasses mechanisms that
enable easy access for those with special needs. *Accessibility* for users (and software
engineers) who may be physically challenged is an imperative for ethical, legal, and
business reasons.

Software engineers and their managers invariably underestimate
the effort and skills required to create user interfaces that accommodate the
needs of different locales and languages

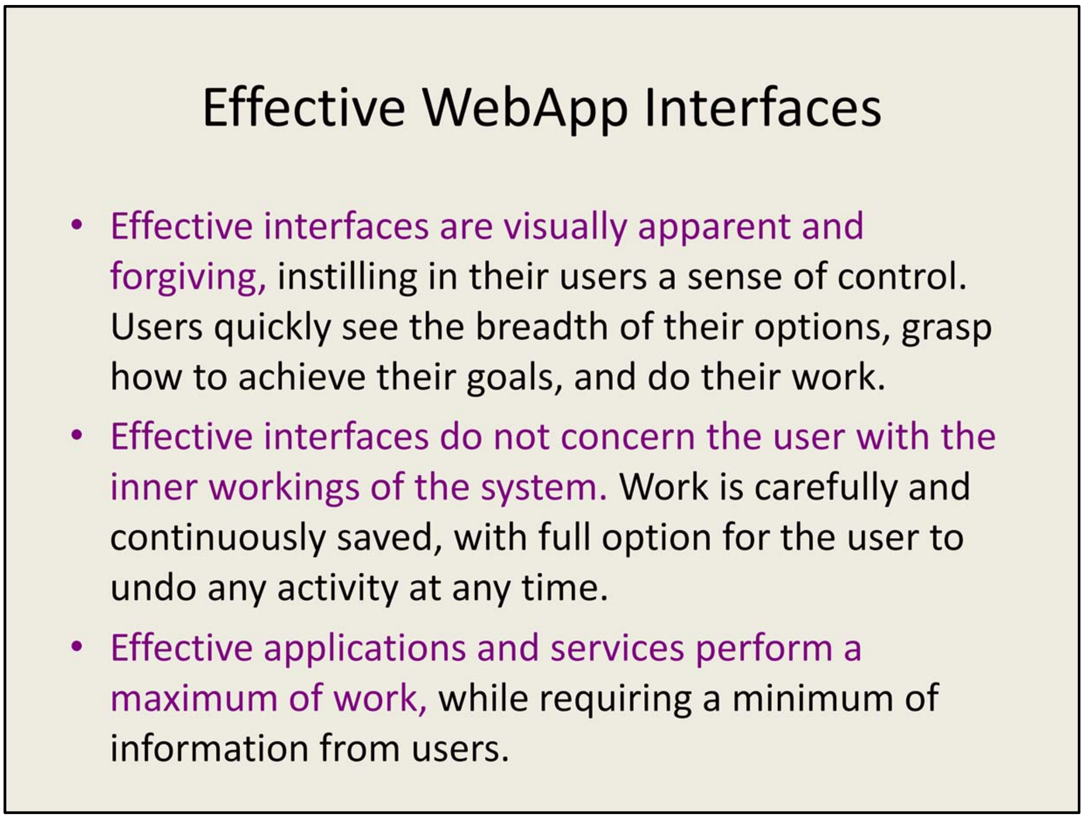


WebApp Interface Design

- *Where am I?* The interface should
  - provide an indication of the WebApp that has been accessed
  - inform the user of her location in the content hierarchy.
- *What can I do now?* The interface should always help the user understand his current options
  - what functions are available?
  - what links are live?
  - what content is relevant?
- *Where have I been, where am I going?* The interface must facilitate navigation.
  - Provide a “map” (implemented in a way that is easy to understand) of where the user has been and what paths may be taken to move elsewhere within the WebApp.

you should design a WebApp interface so that it answers three primary questions for

the end user:



Effective WebApp Interfaces

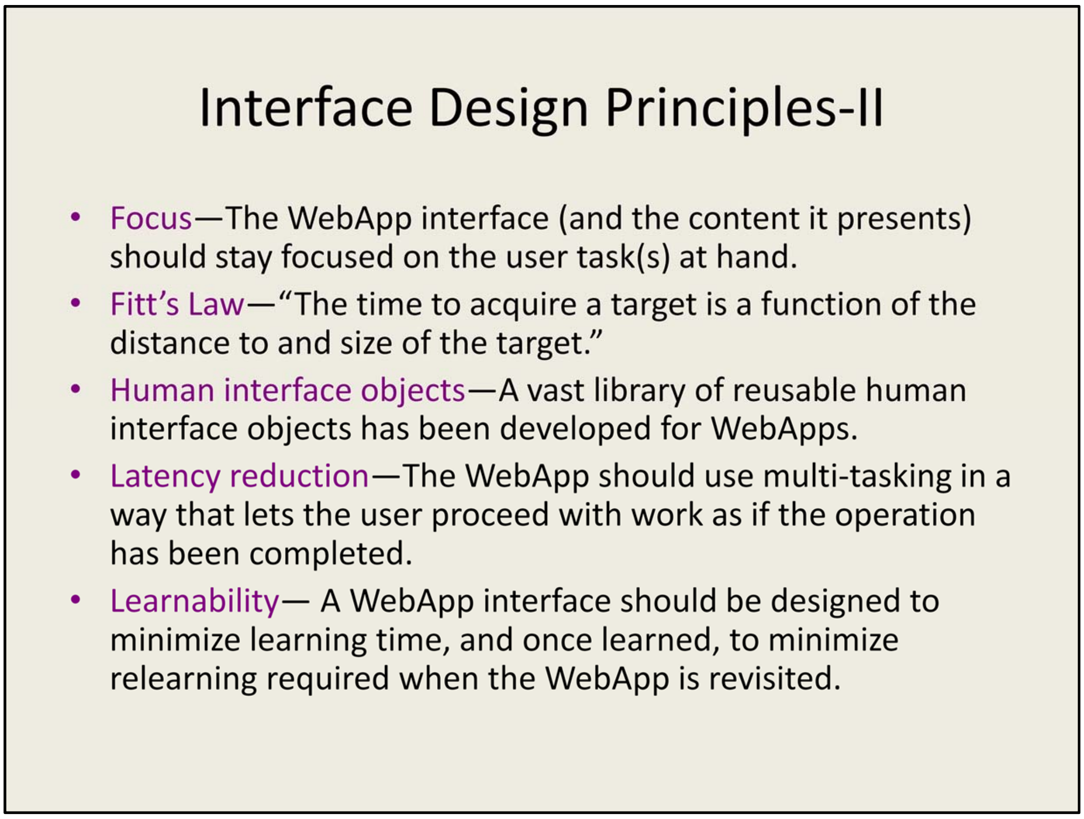
- Effective interfaces are visually apparent and forgiving, instilling in their users a sense of control. Users quickly see the breadth of their options, grasp how to achieve their goals, and do their work.
- Effective interfaces do not concern the user with the inner workings of the system. Work is carefully and continuously saved, with full option for the user to undo any activity at any time.
- Effective applications and services perform a maximum of work, while requiring a minimum of information from users.

a set of fundamental characteristics that all interfaces should exhibit and in doing so, establishes a philosophy that should be

followed by every WebApp interface designer:

Interface Design Principles-I

- Anticipation—A WebApp should be designed so that it anticipates the use’s next move.
- Communication—The interface should communicate the status of any activity initiated by the user
- Consistency—The use of navigation controls, menus, icons, and aesthetics (e.g., color, shape, layout)
- Controlled autonomy—The interface should facilitate user movement throughout the WebApp, but it should do so in a manner that enforces navigation conventions that have been established for the application.
- Efficiency—The design of the WebApp and its interface should optimize the user’s work efficiency, not the efficiency of the Web engineer who designs and builds it or the client-server environment that executes it.

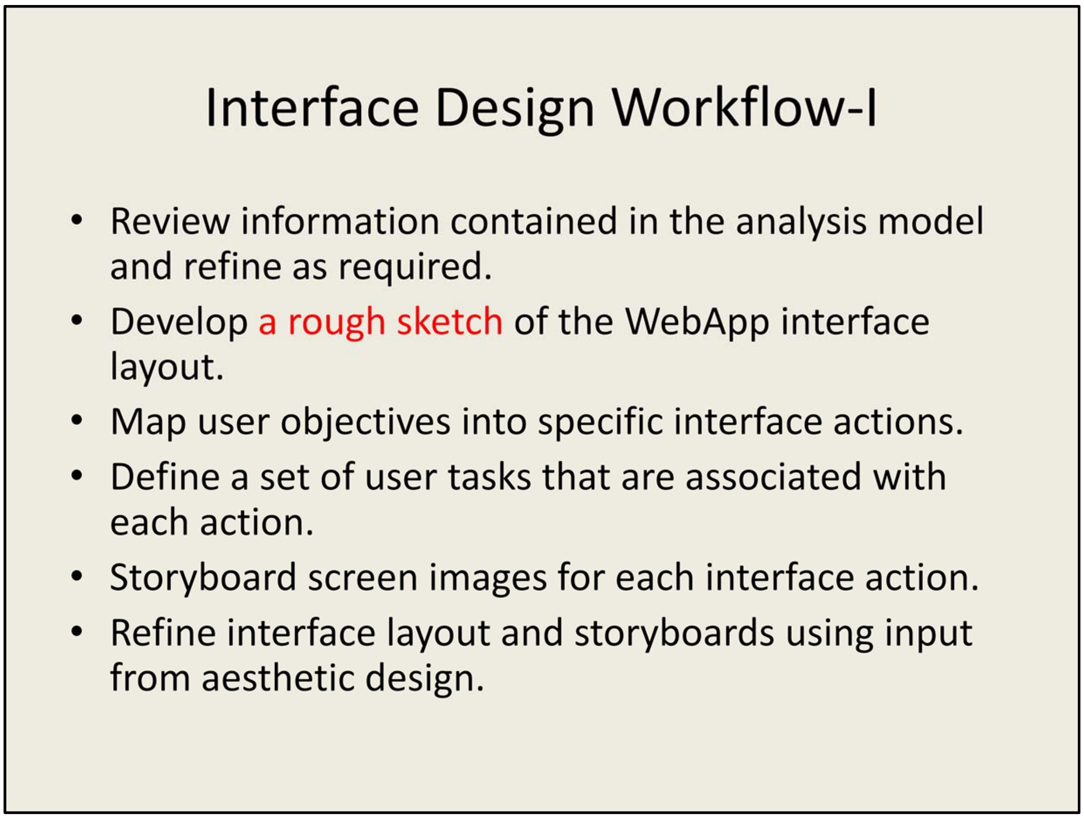


Interface Design Principles-II

- Focus—The WebApp interface (and the content it presents) should stay focused on the user task(s) at hand.
- Fitt's Law—"The time to acquire a target is a function of the distance to and size of the target."
- Human interface objects—A vast library of reusable human interface objects has been developed for WebApps.
- Latency reduction—The WebApp should use multi-tasking in a way that lets the user proceed with work as if the operation has been completed.
- Learnability— A WebApp interface should be designed to minimize learning time, and once learned, to minimize relearning required when the WebApp is revisited.

Interface Design Principles-III

- Maintain work product integrity—A work product (e.g., a form completed by the user, a user specified list) must be automatically saved so that it will not be lost if an error occurs.
- Readability—All information presented through the interface should be readable by young and old.
- Track state—When appropriate, the state of the user interaction should be tracked and stored so that a user can logoff and return later to pick up where she left off.
- Visible navigation—A well-designed WebApp interface provides “the illusion that users are in the same place, with the work brought to them.”

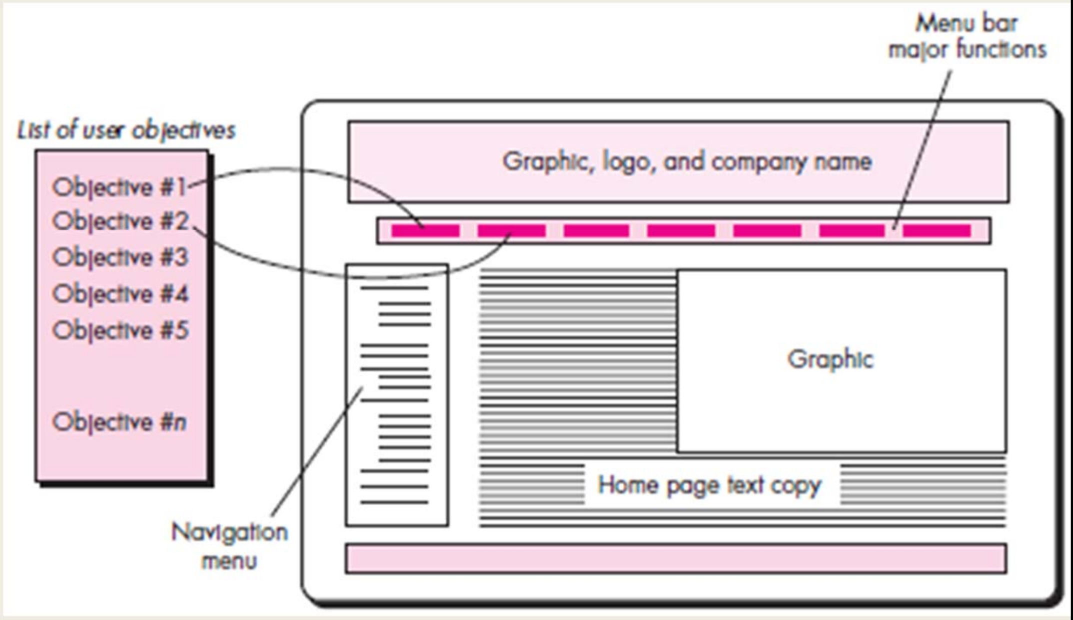


Interface Design Workflow-I

- Review information contained in the analysis model and refine as required.
- Develop a rough sketch of the WebApp interface layout.
- Map user objectives into specific interface actions.
- Define a set of user tasks that are associated with each action.
- Storyboard screen images for each interface action.
- Refine interface layout and storyboards using input from aesthetic design.

A rought sketch is exemplified in the next slide

Mapping User Objectives



An interface
prototype (including the layout) may have been developed as part of the
requirements modeling activity. If the layout already exists, it should be
reviewed and refined as required. If the interface layout has not been developed,
you should work with stakeholders to develop it at this time. A
schematic first-cut layout sketch is shown in this Figure

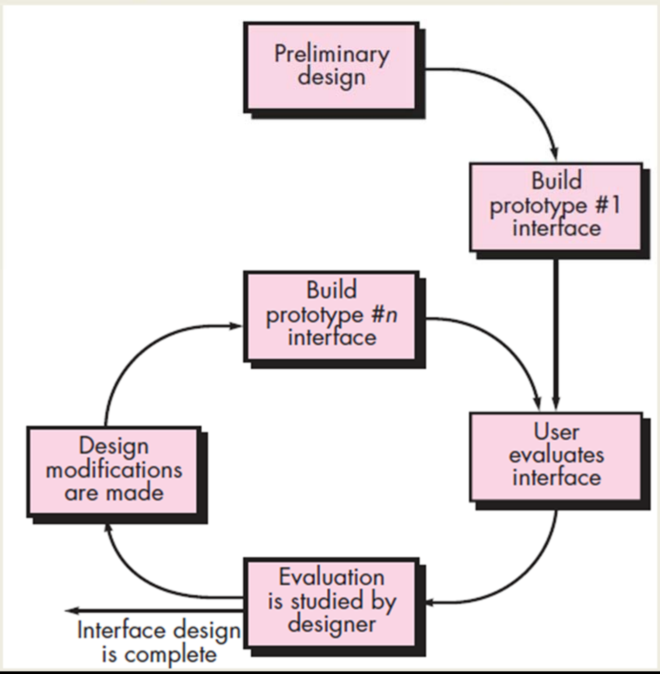
Interface Design Workflow-II

- Identify user interface objects that are required to implement the interface.
- Develop a procedural representation of the user’s interaction with the interface.
- Develop a behavioral representation of the interface.
- Describe the interface layout for each state.
- Refine and review the interface design model.

Aesthetic Design

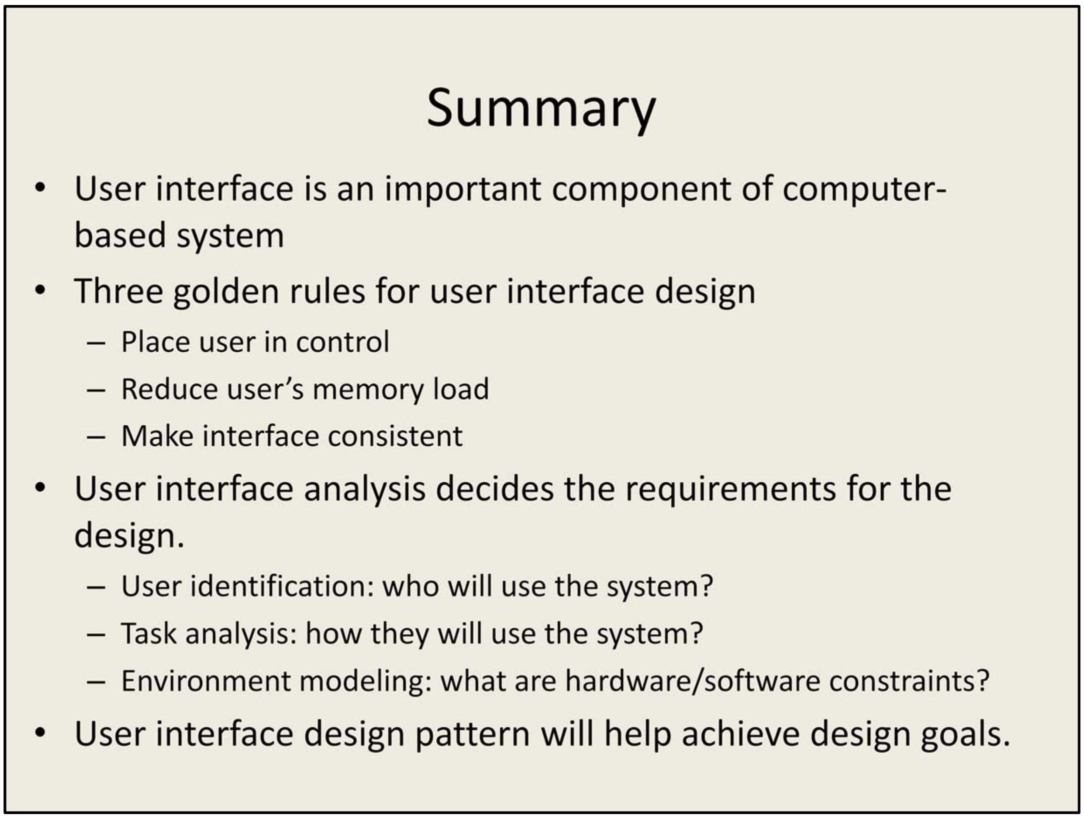
- Don’t be afraid of white space.
- Emphasize content.
- Organize layout elements from top-left to bottom right.
- Group navigation, content, and function geographically within the page.
- Don’t extend your real estate with the scrolling bar.
- Consider resolution and browser window size when designing layout.

Design Evaluation Cycle


Preliminary design
Build prototype #1 interface
Build prototype #n interface
User evaluates interface
Design modifications are made
Evaluation is studied by designer
Interface design is complete

Once you create an operational user interface prototype, it must be evaluated
to determine whether it meets the needs of the user

The user interface evaluation cycle takes the form shown in Figure 11.5. After the
design model has been completed, a first-level prototype is created. The
prototype is
evaluated by the user,11 who provides you with direct comments about the
efficacy
of the interface. In addition, if formal evaluation techniques are used (e.g.,
questionnaires,
rating sheets), you can extract information from these data (e.g.,
80 percent of all users did not like the mechanism for saving data files). Design
modifications
are made based on user input, and the next level prototype is created. The
evaluation cycle continues until no further modifications to the interface design
are
necessary.



Summary

- User interface is an important component of computer-based system
- Three golden rules for user interface design
  - Place user in control
  - Reduce user’s memory load
  - Make interface consistent
- User interface analysis decides the requirements for the design.
  - User identification: who will use the system?
  - Task analysis: how they will use the system?
  - Environment modeling: what are hardware/software constraints?
- User interface design pattern will help achieve design goals.