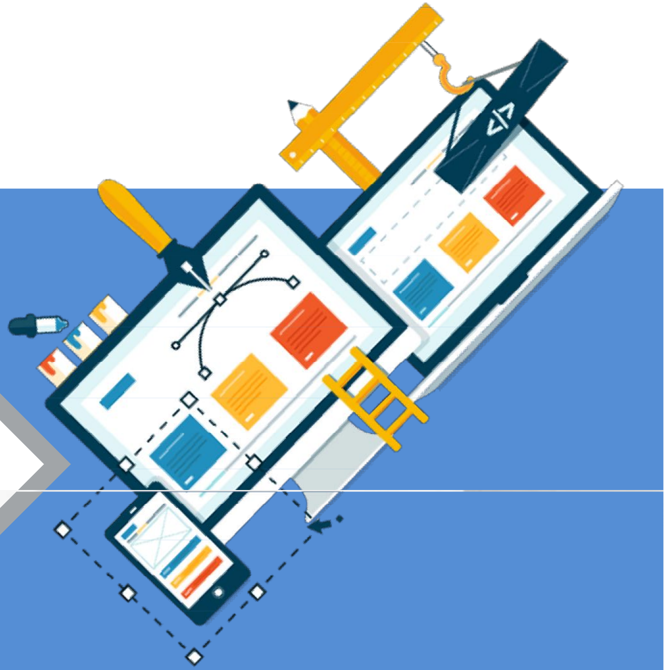
Unit 4.3



User Interface Design

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 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

- 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
  - Definition of major and minor menu items
  - Specification of a real-world metaphor to follow

Interface Objects and Actions - Example

Preliminary use case: User want to gain access to my *SafeHome* system from any remote location via the Internet. Using browser software operating on my notebook computer (while I'm at work or traveling), User can determine the status of the alarm system, arm or disarm the system, reconfigure security zones, and view different rooms within the house via preinstalled video cameras.

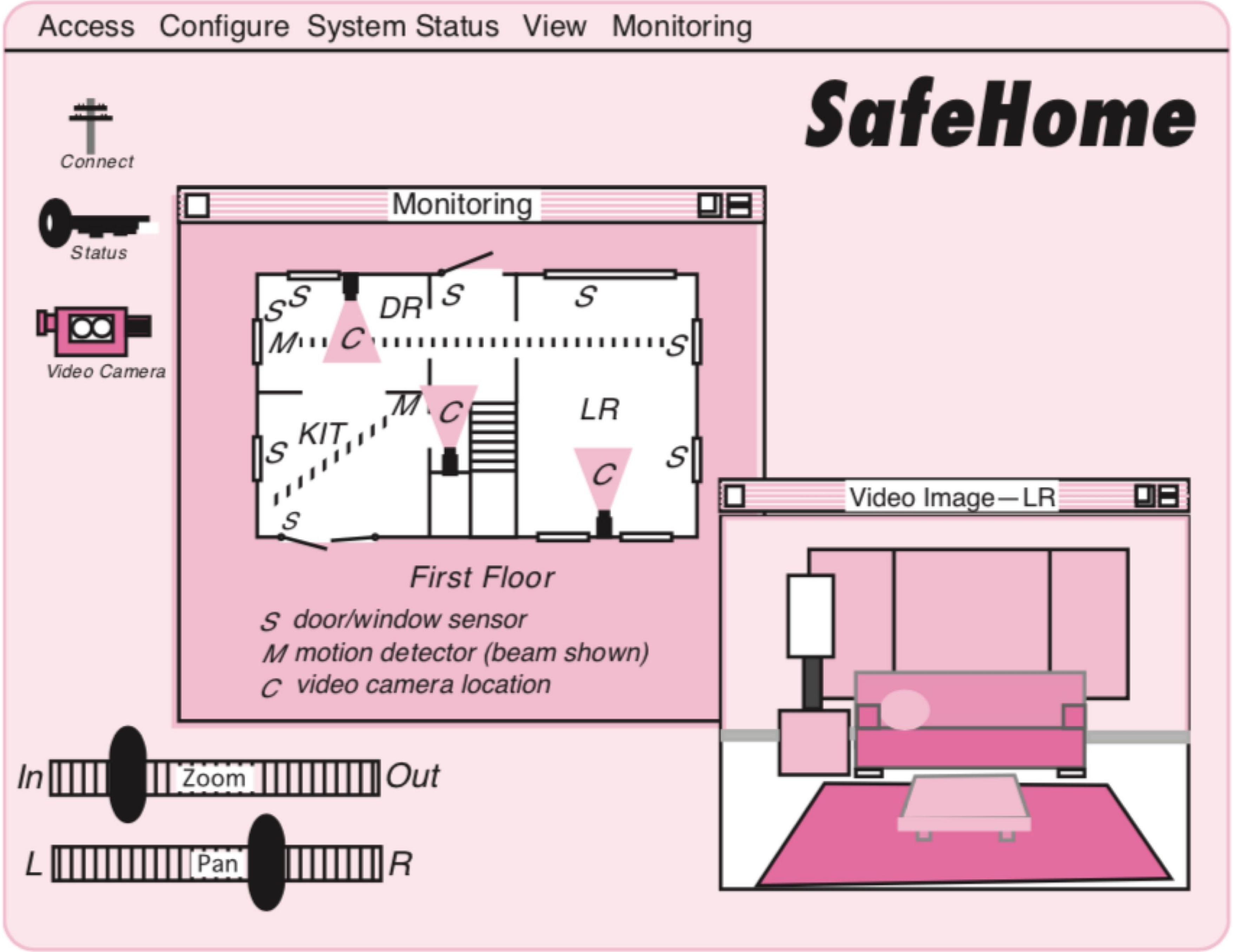
Interface Objects and Actions - Example

Based on this use case, the following homeowner tasks, objects, and data items are identified:

- *accesses* the *SafeHome* system
- *enters* an **ID** and **password** to allow remote access
- *checks* **system status**
- *arms* or *disarms SafeHome* system
- *displays* **floor plan** and **sensor locations**
- *displays* **zones** on floor plan
- *changes* **zones** on floor plan
- *displays* **video camera locations** on floor plan
- *selects* **video camera** for viewing
- *views* **video images** (four frames per second)
- *pans* or *zooms* the **video camera**

Objects (boldface) and actions (italics) are extracted from this list of homeowner tasks. The majority of objects noted are application objects. However, **video camera location** (a source object) is dragged and dropped onto **video camera** (a target object) to create a **video image** (a window with video display).

sketch of the screen layout for video monitoring



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

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 negativec onsequences 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 visualcue 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

User Interface Evaluation

Design and Prototype Evaluation

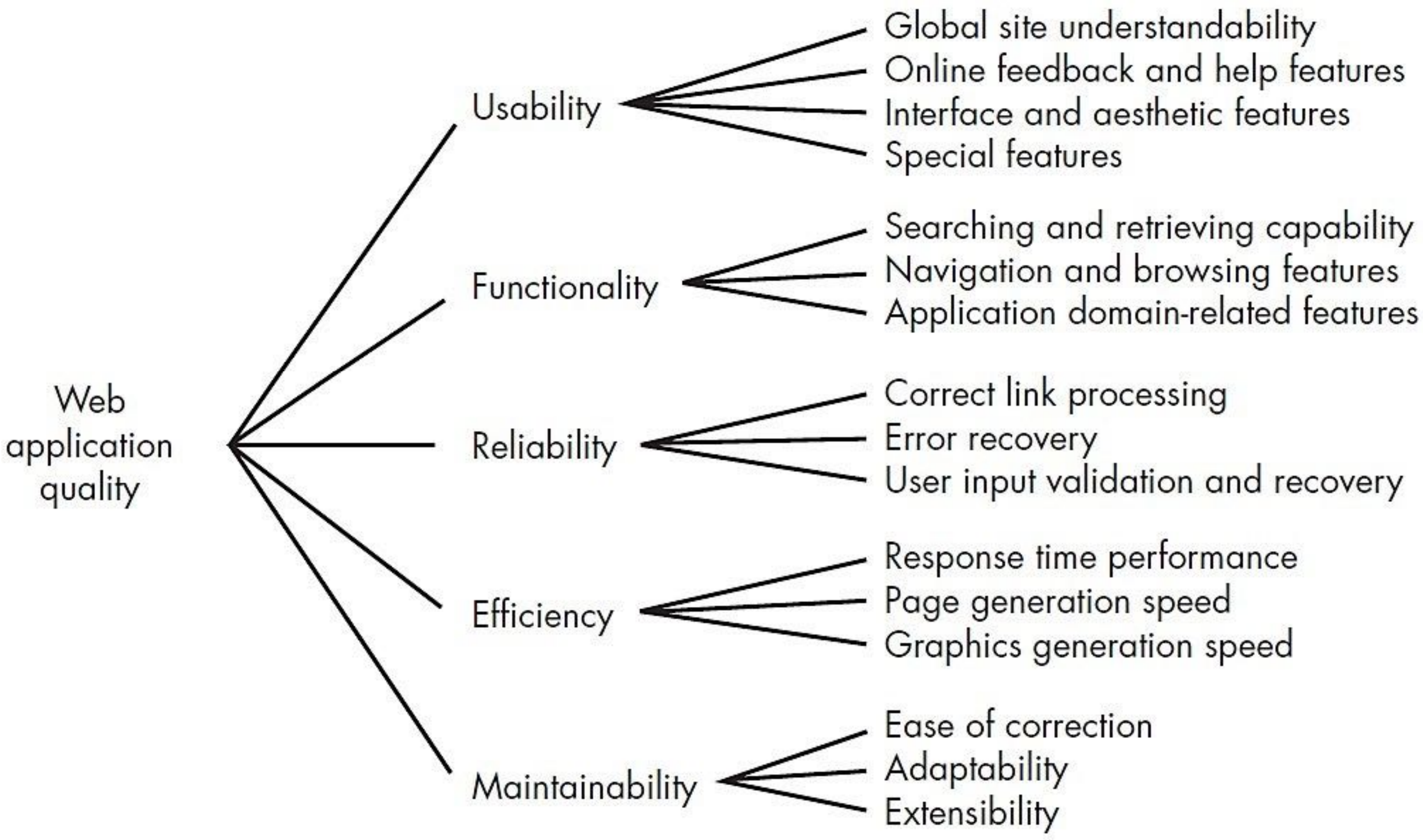
- Before prototyping occurs, a number of evaluation criteria can be applied during design reviews to the design model itself
  - The amount of learning required by the users
    - Derived from the length and complexity of the written specification and its interfaces
  - The interaction time and overall 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

Web Application Design

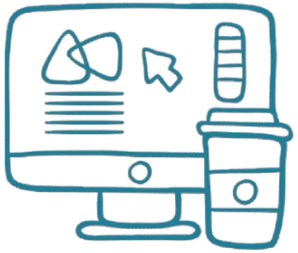
WebApp Design Quality Requirement

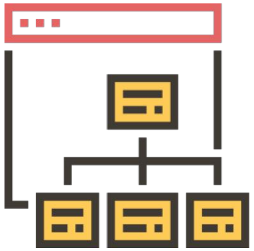

Web application quality
Usability
Global site understandability
Online feedback and help features
Interface and aesthetic features
Special features
Functionality
Searching and retrieving capability
Navigation and browsing features
Application domain-related features
Reliability
Correct link processing
Error recovery
User input validation and recovery
Efficiency
Response time performance
Page generation speed
Graphics generation speed
Maintainability
Ease of correction
Adaptability
Extensibility

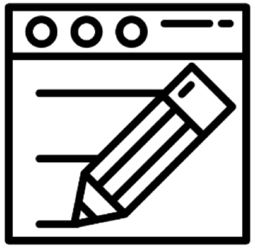
Web Application Design

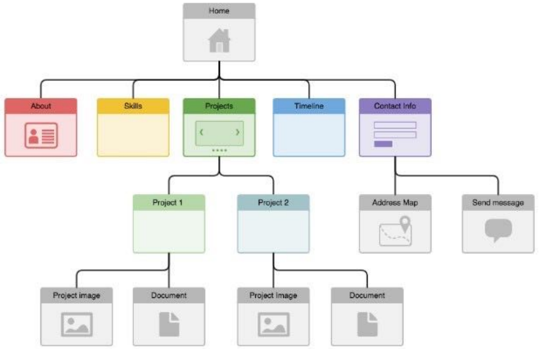
□ Design for WebApp encompasses technical and nontechnical activities that include:

- Establishing the look and feel of the WebApp
- Defining the overall architectural structure
- Developing the content and functionality that reside within the architecture
- Planning the navigation that occurs within the WebApp









Web App Interface Design

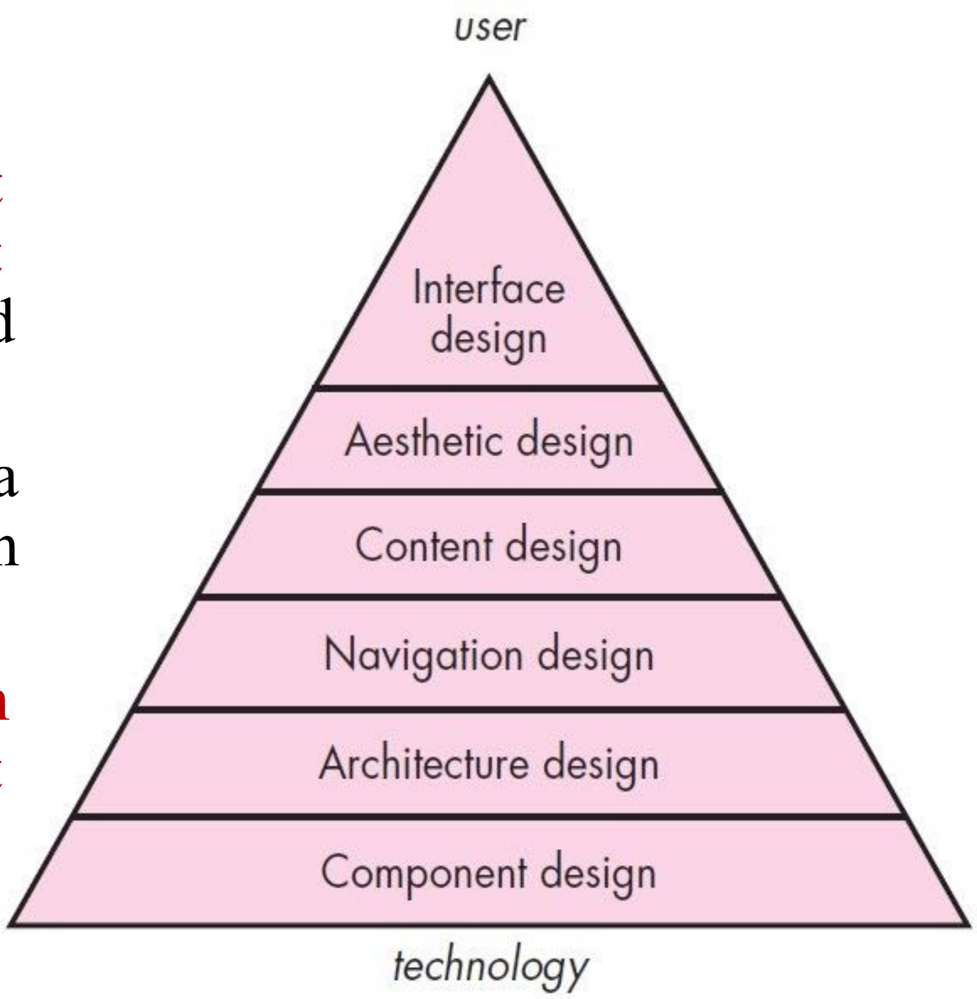
The objectives of a WebApp interface are to:

□ Establish a consistent window into the content and functionality provided by the interface.

□ Guide the user through a series of interactions with the WebApp.

□ Organize the navigation options and content available to the user.

Design pyramid for WebApps



Design pyramid for WebApps Cont.

□ Interface Design

- One of the challenges of interface design for WebApps is the nature of the user's entry point.

□ Aesthetic Design

- Also called graphic design, is an artistic endeavor (offer) that complements the technical aspects of WebApp design.

□ Content Design

- Generate content and design the representation for content to be used within a WebApp.

□ Architecture Design

- It is tied to the goals established for a WebApp,
- the content to be presented, the users who will visit, and the navigation that has been established.

Design pyramid for WebApps Cont.

□ Navigation Design

- Define navigation pathways that enable users to access WebApp content and functions.

□ Component-Level Design

- Modern WebApps deliver increasingly sophisticated processing functions that,
  - Perform localized processing to generate content and navigation capability in a dynamic fashion,
  - Provide computation or data processing capability that are appropriate for the WebApp's business domain.
  - Provide sophisticated database query and access.
  - Establish data interfaces with external corporate systems.

workflow for WebApp interface design:

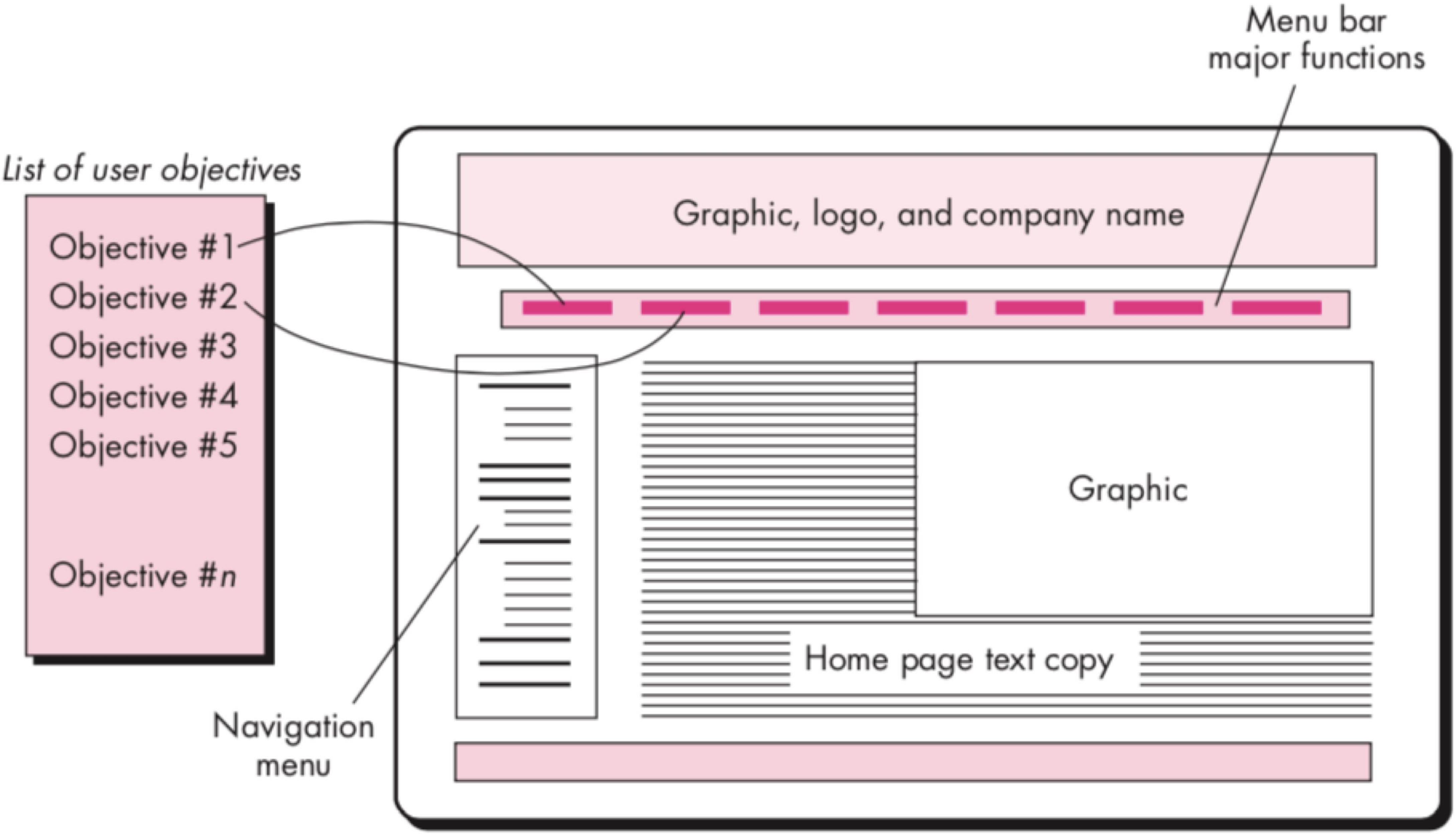
The following tasks represent a rudimentary workflow for WebApp interface design:

1. **Review information contained in the requirements model and refine as required.**
2. **Develop a rough sketch of the WebApp interface layout (See Fig in next few slides).**
3. **Map user objectives into specific interface actions.** Objectives should be mapped into specific interface actions as shown in Figure in next slides
4. **Define a set of user tasks that are associated with each action.** Each interface action (e.g., “buy a product”) is associated with a set of user tasks. These tasks have been identified during requirements modeling.
5. **Storyboard screen images for each interface action.** As each action is considered, a sequence of storyboard images (screen images) should be created to depict how the interface responds to user interaction.

workflow for WebApp interface design:

6. **Refine interface layout and storyboards using input from aesthetic design.**
7. **Identify user interface objects that are required to implement the interface.**
8. **Describe the interface layout for each state.**
9. **Refine and review the interface design model.** Review of the interface should focus on usability.

Mapping user objectives into interface actions


Menu bar
major functions
List of user objectives
Objective #1
Objective #2
Objective #3
Objective #4
Objective #5
Objective #n
Graphic, logo, and company name
Graphic
Home page text copy
Navigation
menu