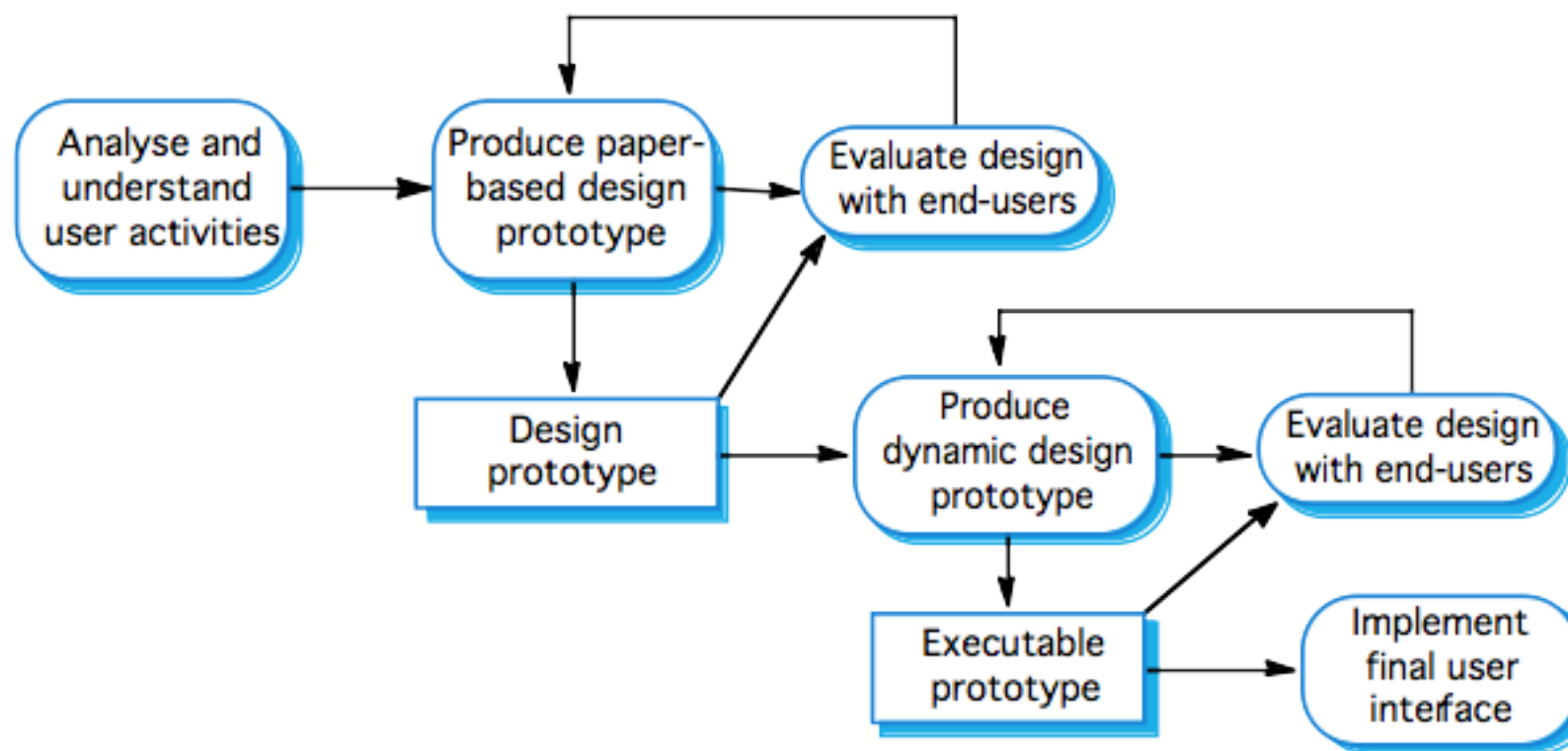# Introduction to Software Engineering

8. User Interface Design

The UI design process

> UI design is an *iterative process* involving close liaisons between users and designers.

> The 3 core activities in this process are:

— *User analysis*. Understand what the users will do with the system;

— *System prototyping*. Develop a series of prototypes for experiment;

— *Interface evaluation*. Experiment with these prototypes with users.

The design process

Interface Design Models

Four different models occur in HCI design:

1. The <u>design model</u> expresses the *software design*.

2. The <u>user model</u> describes the *profile of the end users*. (i.e., novices vs. experts, cultural background, etc.)

3. The <u>user's model</u> is the end users' *perception of the system.*

4. The <u>system image</u> is the *external manifestation* of the system (look and feel + documentation etc.)

User Interface Design Principles

Principle	Description
User familiarity	Use terms and concepts *familiar* to the user.
Consistency	Comparable operations should be activated in the *same way*. Commands and menus should have the same format, etc.
Minimal surprise	If a command operates in a known way, the user should be able to *predict* the operation of comparable commands.
Feedback	Provide the user with visual and auditory feedback, maintaining *two-way communication*.

GUI Characteristics

Characteristic	Description
Windows	Multiple windows allow *different information to be displayed simultaneously* on the user's screen.
Icons	Usually icons represent *files* (including folders and applications), but they may also stand for *processes* (e.g., printer drivers).
Menus	Menus bundle and organize *commands* (eliminating the need for a command language).
Pointing	A pointing device such as a mouse is used for *command choices* from a menu or indicating items of interest in a window.
Graphics	Graphical elements can be *commands* on the same display.

GUIs

Advantages

> They are *easy to learn* and use.

— Users without experience can learn to use the system quickly.

> The user may *switch attention* between tasks and applications.

> *Fast, full-screen interaction* is possible with immediate access to the entire screen

Problems

> A GUI is not automatically a good interface

— Many software systems are *never used* due to poor UI design

— A poorly designed UI can cause a user to make *catastrophic errors*

Direct Manipulation

Advantages

> Users *feel in control* and are less likely to be intimidated by the system

> User *learning time* is relatively short

> Users get *immediate feedback* on their actions

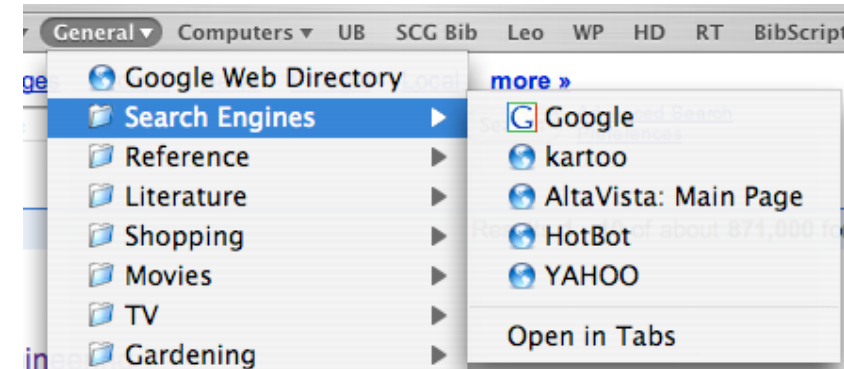> mistakes can be quickly detected and corrected

Problems

> Finding the right user *metaphor* may be difficult

> It can be *hard to navigate* efficiently in a large information space.

> It can be *complex to program* and demanding to execute

Menu Systems

Advantages

> Users don't need to remember command names

> Typing effort is minimal

> User errors are trapped by the interface

> Context-dependent help can be provided (based on the current menu selection)

Problems

> Actions involving *logical conjunction* (and) or disjunction (or) are *awkward* to represent

> If there are many choices, some *menu structuring* facility must be used

> *Experienced users find menus slower* than command language

Menu Structuring

Scrolling menus
> The menu can be scrolled to reveal additional choices
> Not practical if there is a very large number of choices

Hierarchical menus
> Selecting a menu item causes the menu to be replaced by a sub-menu

Walking menus
> A menu selection causes another menu to be revealed

Associated control panels
> When a menu item is selected, a control panel pops-up with further options