Administrative Stuff

- We are now in week 10
- No class on Thursday
- User and Maintenance Manuals
  - Individual teams meetings to discuss
Design (and Build) with Reuse
Origins and Scope

- Long history in engineering
  - E.g. automotive (Plymouth Voyager, VW Jetta, etc.)
  - Appliances, even computers
- Levels
  - Application
  - Component
  - Function
Benefits of Reuse

- Increased reliability
- Reduced process risk
- Effective use of specialists
- Standards compliance
- Accelerated development
Requirements for Design w/ Reuse

- **Accessible**
  - It must be possible to find appropriate reusable components

- **Reliable**
  - The reuser of the component must be confident that the components will be reliable and will behave as specified

- **Understandable**
  - The components must be documented so that they can be understood and, where appropriate, modified
Main Problems in Practice

- “Not-invented-here” syndrome
- Maintaining a component library
- Finding and adapting reusable components
Generator-Based Reuse

- Generally limited to small domains
  - Graphical program generators
    - Visual UI tools, etc.
  - Parsing
    - lex, yacc, etc.
Component-Based Development

- Component-Based Software Engineering (CBSE)
- OOD failure to realize re-use
  - Single-object classes are too detailed
- “Components” are more abstract
  - Stand-alone services
Component Interfaces

- **Two types**
  - "Provides" interface
    - Services provided to other components
  - "Requires" interface
    - Services that must be made available for the component to execute as specified

- **Publication via the interface**
  - Interface "discovery"
  - AppleScript dictionary
Building Reuse Into the Process

1. Design system architecture
2. Specify components
3. Search for reusable components
4. Incorporate discovered components

Outline system requirements

- Search for reusable components
- Modify requirements according to discovered components

Architectural design

- Search for reusable components
- Specify system components based on reusable components
Problems with CBSE

- Component incompatibilities
- Finding and understanding components
  - Experience will help
- Managing evolution of design
  - May be impossible to change component behavior
COTS Product Reuse

- COTS = Commercial Off-The-Shelf systems
- Complete application systems w/ API
- Building large systems by integrating COTS systems is now a viable development strategy for some types of system such as E-commerce systems
COTS Problems

- Little control over function and performance
- Problems with COTS system interoperability
- No control over system evolution
  - COTS vendors control evolution
- Support from COTS vendors
Component Development for Reuse

- Generalize existing components
- Component reusability
  - Stable domain abstractions
  - Hide state representation
  - As independent as possible
  - Publish exceptions through component interface
- Reusability and usability trade-off
User Interface Design
What and Why

- System users often judge a system by its interface rather than its functionality.
- A poorly designed interface can cause a user to make catastrophic errors.
  - Therac-25 incident
- Poor user interface design is a reason why many software systems are never used.
- Graphical User Interface (GUI)
  - 2D versus 3D
<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>Multiple windows allow different information to be displayed simultaneously on the user’s screen.</td>
</tr>
<tr>
<td>Icons</td>
<td>Icons different types of information. On some systems, icons represent files; on others, icons represent processes.</td>
</tr>
<tr>
<td>Menus</td>
<td>Commands are selected from a menu rather than typed in a command language.</td>
</tr>
<tr>
<td>Pointing</td>
<td>A pointing device such as a mouse is used for selecting choices from a menu or indicating items of interest in a window.</td>
</tr>
<tr>
<td>Graphics</td>
<td>Graphical elements can be mixed with text on the same display.</td>
</tr>
</tbody>
</table>
UI Design Process

1. Analyse and understand user activities
2. Produce paper-based design prototype
3. Design prototype
4. Evaluate design with end-users
5. Produce dynamic design prototype
6. Evaluate design with end-users
7. Executable prototype
8. Implement final user interface
Design Principles

- User familiarity
- Consistency
- Minimal surprise
- Recoverability
- Guidance
- Diversity
Interaction Styles

- Direct manipulation
- Menu selection
- Form fill-in
- Command language
- Natural language
Information Display Factors

- Precise information or data relationships?
- How quickly do information values change? Must the change be indicated immediately?
- Must the user take some action in response to a change?
- Is there a direct manipulation interface?
- Is the information textual or numeric? Are relative values important?
Color Use Guidelines

- Don't use too many colors
- Use color coding to support user tasks
- Allow users to control color coding
- Design for monochrome then add color
- Use color coding consistently
- Avoid color pairings which clash
- Use color change to show status change
- Be aware that color displays are usually lower resolution
- 10% of male population is color blind!
User Support

- System facilities to support users
  - on-line help, error messages, manuals etc.
- The user guidance system should be integrated with the user interface
  - Help users when they need information about the system or when they make some kind of error
- Apple guides
- The help and error message system should, if possible, be integrated
Error Messages

- Error message design is critically important.
  - Poor error messages can mean that a user rejects rather than accepts a system
- Messages should be polite, concise, consistent and constructive
- The background and experience of users should be the determining factor in message design
Help System

- **Multiple entry points** should be provided so that the user can get into the help system from different places.
- Some indication of where the user is **positioned** in the help system is valuable.
- Facilities should be provided to allow the user to **navigate and traverse** the help system.
See on-line (Notes) links