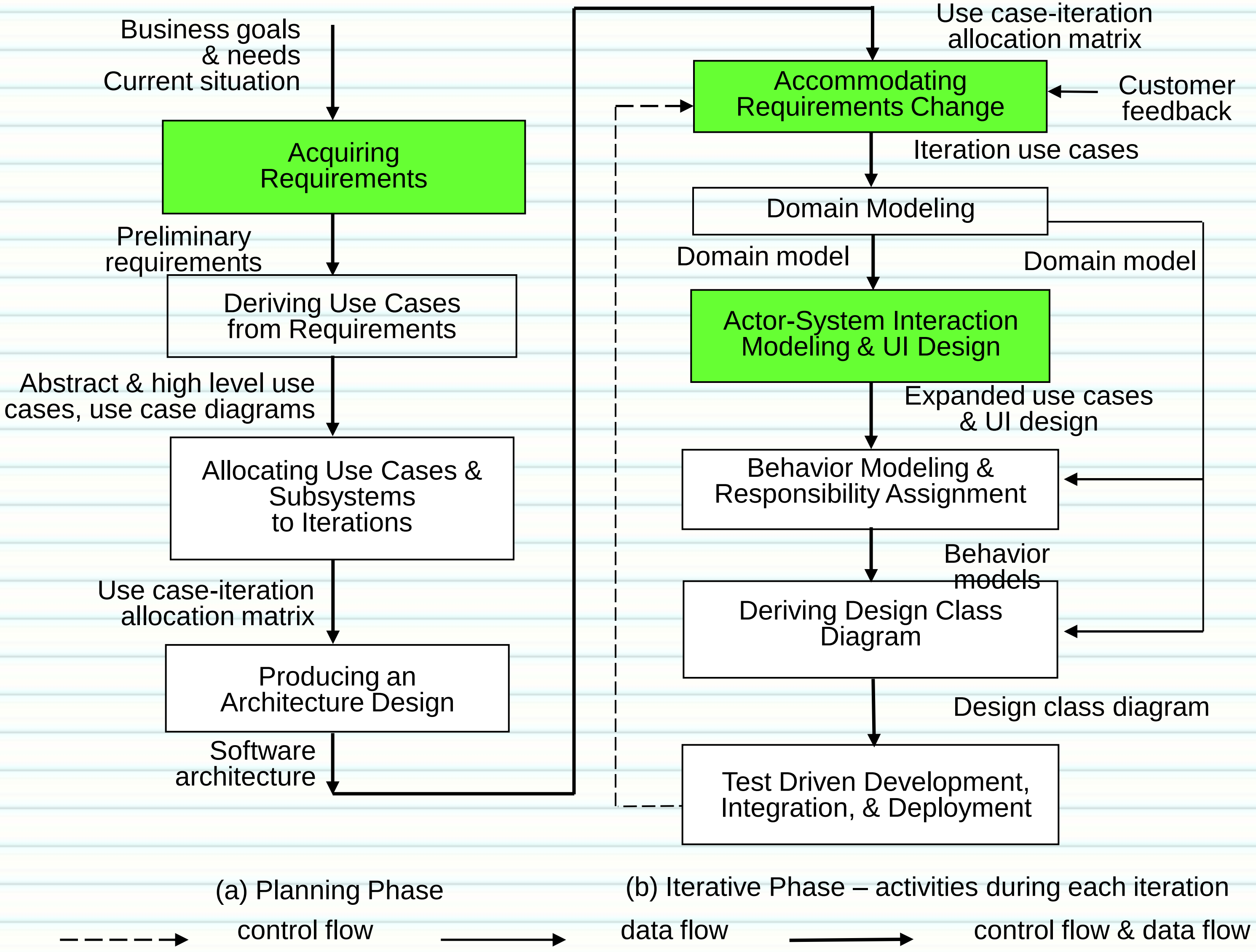
Chapter 12 – User Interface Design

Dr John H Robb, PMP, SEMC
UT Arlington
Computer Science and Engineering

Key Takeaway Points

- User interface design is concerned with the design of the look and feel of the user interfaces.
- The design for change, separation of concerns, information-hiding, high-cohesion, low-coupling, and keep-it-simple-and-stupid software design principles should be applied during user interface design.

UI Design in the Methodology Context



Business goals & needs
Current situation

Acquiring Requirements

Preliminary requirements

Deriving Use Cases from Requirements

Abstract & high level use cases, use case diagrams

Allocating Use Cases & Subsystems to Iterations

Use case-iteration allocation matrix

Producing an Architecture Design

Software architecture

Use case-iteration allocation matrix

Accommodating Requirements Change

Customer feedback

Iteration use cases

Domain Modeling

Domain model

Domain model

Actor-System Interaction Modeling & UI Design

Expanded use cases & UI design

Behavior Modeling & Responsibility Assignment

Behavior models

Deriving Design Class Diagram

Design class diagram

Test Driven Development, Integration, & Deployment

(a) Planning Phase

(b) Iterative Phase – activities during each iteration

- - - - → control flow

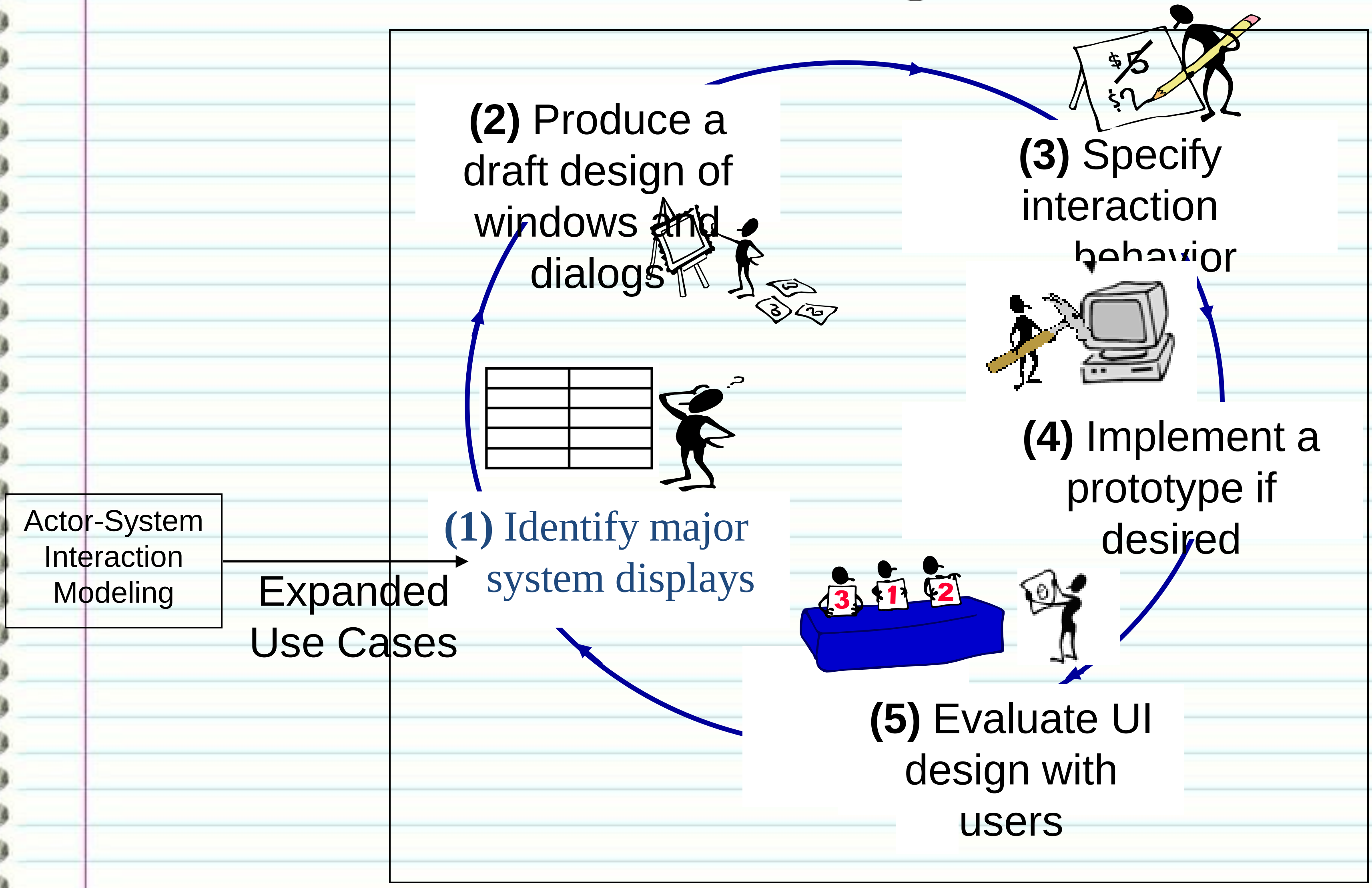
——→ data flow

━━→ control flow & data flow

User Interface Design Activities

- Layout design for windows and dialog boxes.
- Design of interaction behavior.
- Design of information presentation schemes.
- Design of online support.

User Interface Design Process



Actor-System Interaction Modeling

Expanded Use Cases

(1) Identify major system displays

(2) Produce a draft design of windows and dialogs

(3) Specify interaction behavior

(4) Implement a prototype if desired

(5) Evaluate UI design with users

User Interface Design Process (cont.)

- Explanation of steps

1. Identify major system displays.

   The major system displays, user input and user actions are identified from the expanded use cases produced in the current iteration. These form the basis for the design of the look and feel in the next two steps.

2. Produce a draft design of the user interface.

   Develop a draft layout of the user interface (including windows and dialog boxes) corresponding to the major systems displays (previous step) is produced. This step designs the “look” of the user interface.

3. Specify the interaction behavior

   A state diagram is produced to specify the navigation relationships between the user interface items. This step develops the “feel” of the user interface.

4. Construct a user interface prototype

   This step produces a user interface prototype to show the look and feel as designed in the previous two steps.

5. Evaluate the design with users

   In this step, the user interface design, and possibly the prototype, is presented to a group of user representatives to solicit their feedback.

State Diagram Editor Example

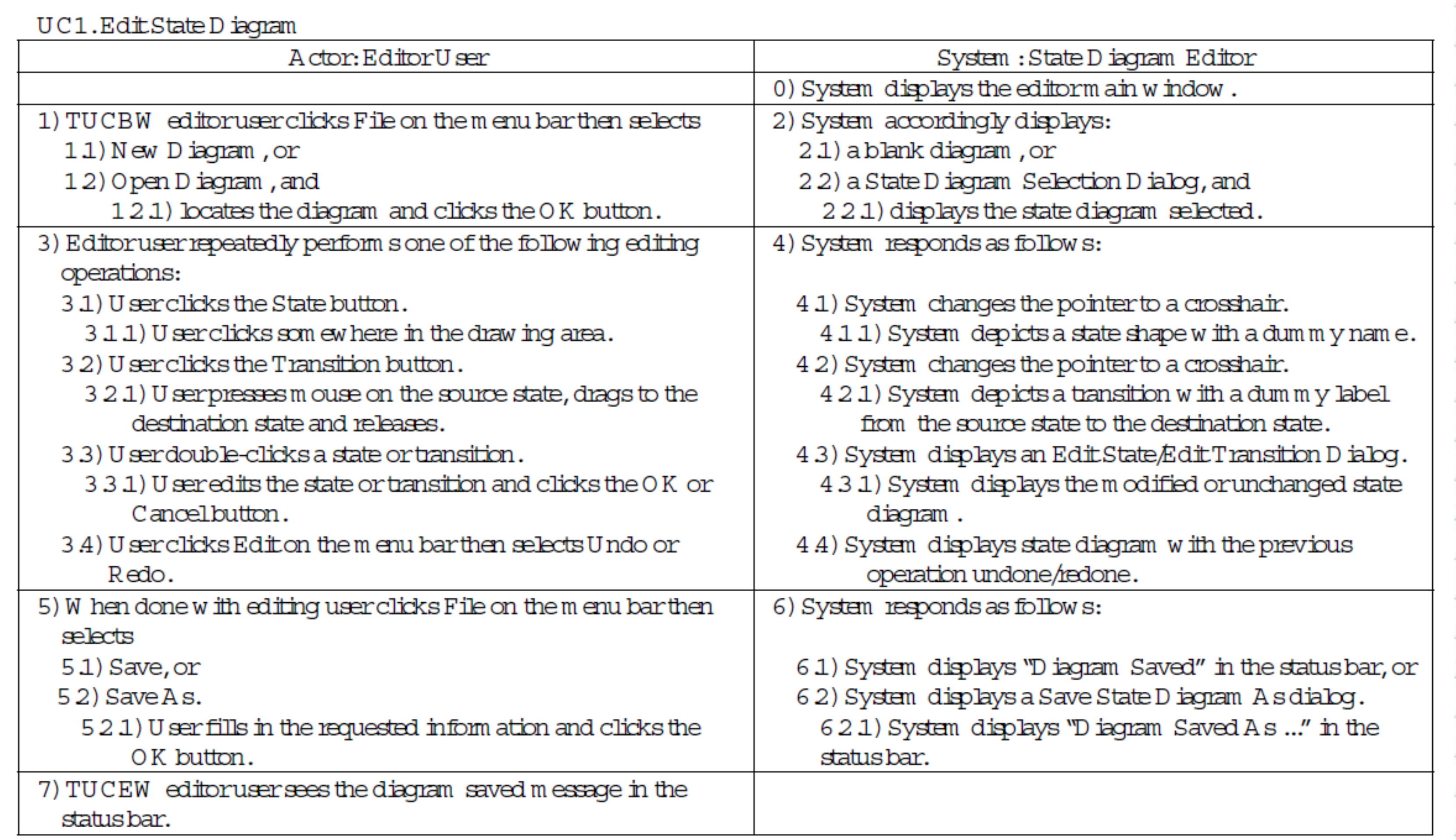
- The state diagram editor is a stand-alone application that allows the user to draw or edit a state diagram.

- For simplicity, only flat state diagrams are considered in the version of the editor – they do not contain other states.

- For the next slide, we are considering the following portion of the requirements
  - "The state diagram editor shall allow a user to edit a new or existing diagram. The editor shall allow a user to add, delete, and edit states and transitions as well as undo and redo these operations. The editor shall allow the user to perform other editing operations including saving a diagram as is."

Edit State Diagram Expanded Use Case

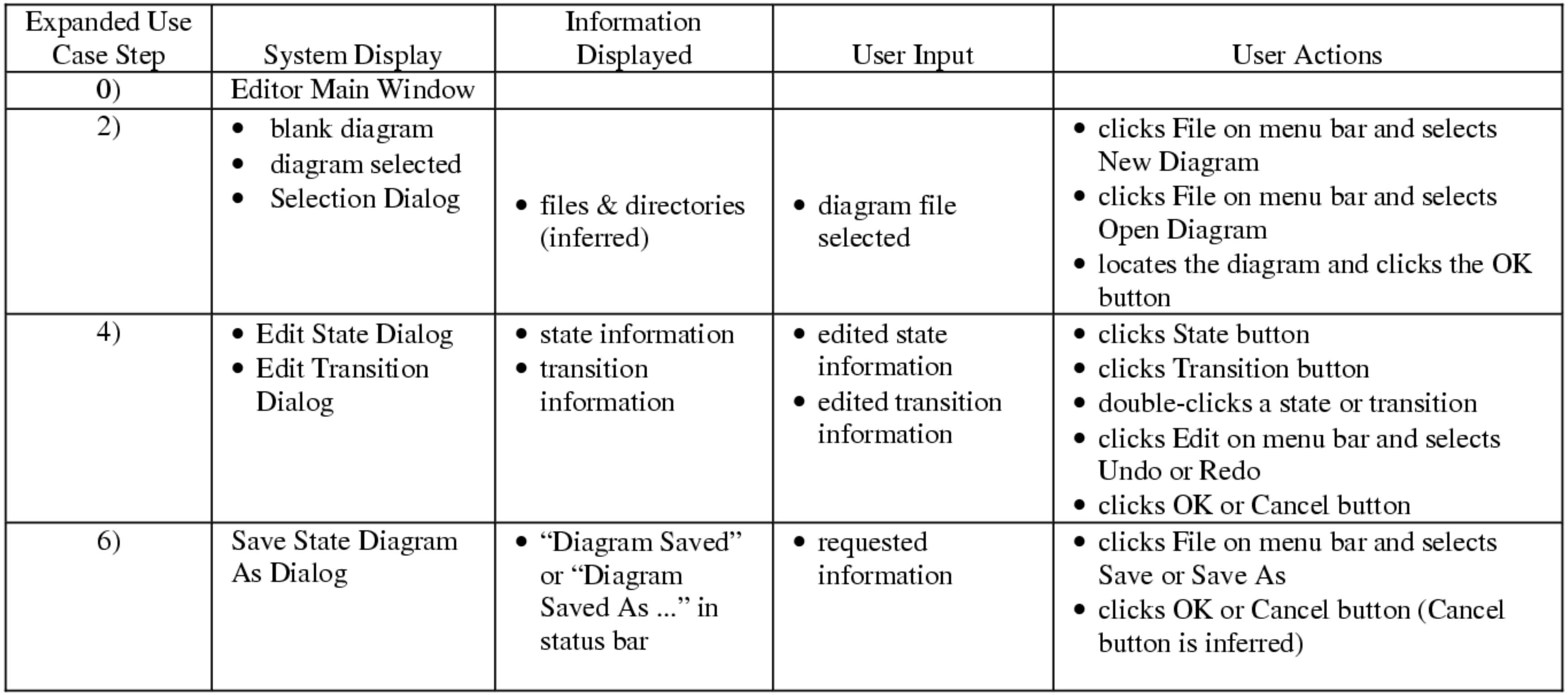
This expanded user case shows is for the state diagram editor. For simplicity, it does not show delete operations.



UC1. Edit State Diagram

Actor: Editor User	System : State Diagram Editor
	0) System displays the editor main window.
1) TUCBW editor user clicks File on the menu bar then selects 1.1) New Diagram, or 1.2) Open Diagram, and 1.2.1) locates the diagram and clicks the OK button.	2) System accordingly displays: 2.1) a blank diagram, or 2.2) a State Diagram Selection Dialog, and 2.2.1) displays the state diagram selected.
3) Editor user repeatedly performs one of the following editing operations: 3.1) User clicks the State button. 3.1.1) User clicks somewhere in the drawing area. 3.2) User clicks the Transition button. 3.2.1) User presses mouse on the source state, drags to the destination state and releases. 3.3) User double-clicks a state or transition. 3.3.1) User edits the state or transition and clicks the OK or Cancel button. 3.4) User clicks Edit on the menu bar then selects Undo or Redo.	4) System responds as follows: 4.1) System changes the pointer to a crosshair. 4.1.1) System depicts a state shape with a dummy name. 4.2) System changes the pointer to a crosshair. 4.2.1) System depicts a transition with a dummy label from the source state to the destination state. 4.3) System displays an Edit State/Edit Transition Dialog. 4.3.1) System displays the modified or unchanged state diagram. 4.4) System displays state diagram with the previous operation undone/redone.
5) When done with editing user clicks File on the menu bar then selects 5.1) Save, or 5.2) Save As. 5.2.1) User fills in the requested information and clicks the OK button.	6) System responds as follows: 6.1) System displays "Diagram Saved" in the status bar, or 6.2) System displays a Save State Diagram As dialog. 6.2.1) System displays "Diagram Saved As ..." in the status bar.
7) TUCEW editor user sees the diagram saved message in the status bar.	

Edit State Diagram System Displays



Expanded Use Case Step	System Display	Information Displayed	User Input	User Actions
0)	Editor Main Window			
2)	• blank diagram • diagram selected • Selection Dialog	• files & directories (inferred)	• diagram file selected	• clicks File on menu bar and selects New Diagram • clicks File on menu bar and selects Open Diagram • locates the diagram and clicks the OK button
4)	• Edit State Dialog • Edit Transition Dialog	• state information • transition information	• edited state information • edited transition information	• clicks State button • clicks Transition button • double-clicks a state or transition • clicks Edit on menu bar and selects Undo or Redo • clicks OK or Cancel button
6)	Save State Diagram As Dialog	• "Diagram Saved" or "Diagram Saved As ..." in status bar	• requested information	• clicks File on menu bar and selects Save or Save As • clicks OK or Cancel button (Cancel button is inferred)

Step 1 – Identify Major System Displays

- A major system display is one that user inputs or displays system processing results. An error message or a confirmation dialog is not a major system display.
- In this step the major system displays, the displayed information, as well as associated user input and user actions are identified from the expanded use cases produced in the current increment.
- The following rules are applied in this step.

1. Examine the steps in the right column of each expanded use case. A major system display is identified if
   a. The step displays system processing results, in this case, the displayed information is also identified, or
   b. The step requests the user to supply input (text, selection, or other type of input)
2. Examine the steps in the left column of each expanded use case to identify user input and user actions associated with each system display

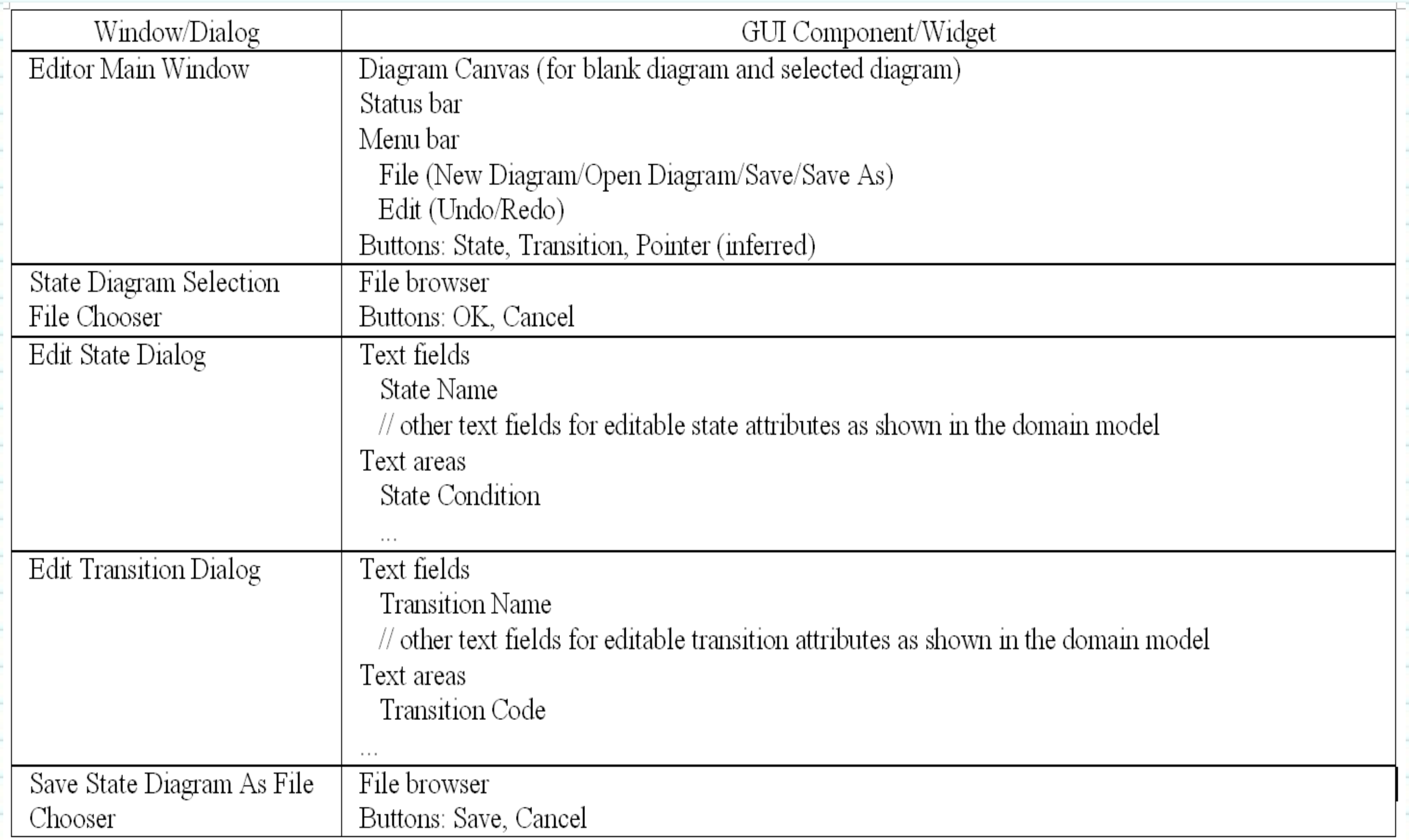
Step 2 – Produce a Draft Layout Design

- In this step 2 activities are performed:
  1. Derive windows and dialog boxes for the system displays identified in the previous step, and
  2. Produce a draft design for these windows and dialog boxes.

- Perform the steps above
  1. The latter part of this presentation deals with design on windows, dialog boxes, etc. – so these are to be used when designing them. From the information displayed, user input and user actions specified in the expanded use cases, the widgets contained in each window or dialog are derived.

- The next slide shows the windows, dialogs, and widgets identified for the state diagram editor

Windows, Dialogs and Widgets



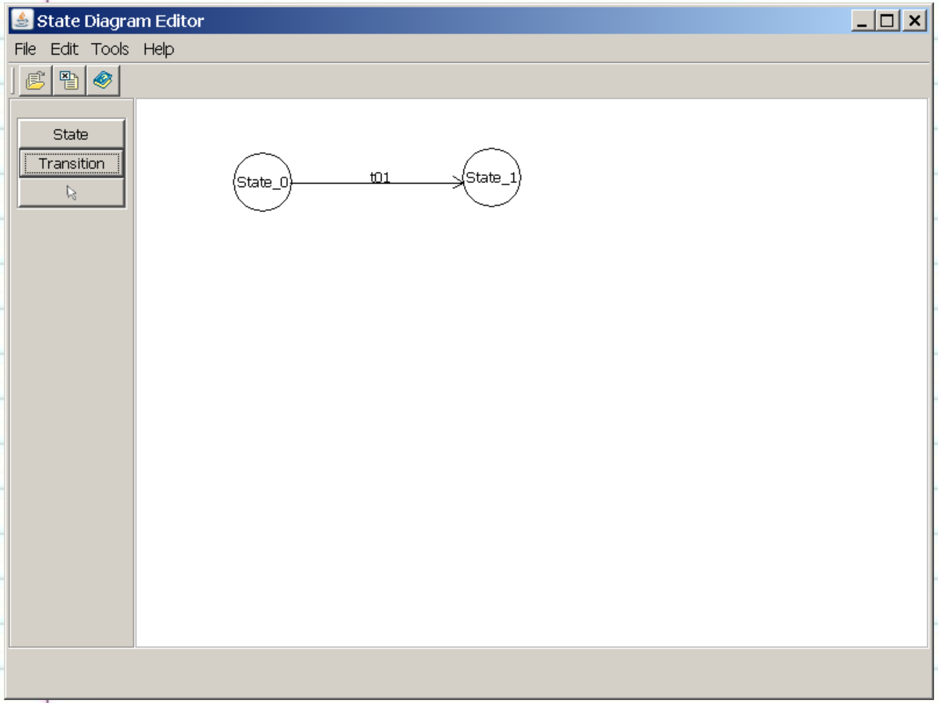
Window/Dialog	GUI Component/Widget
Editor Main Window	Diagram Canvas (for blank diagram and selected diagram) Status bar Menu bar File (New Diagram/Open Diagram/Save/Save As) Edit (Undo/Redo) Buttons: State, Transition, Pointer (inferred)
State Diagram Selection File Chooser	File browser Buttons: OK, Cancel
Edit State Dialog	Text fields State Name // other text fields for editable state attributes as shown in the domain model Text areas State Condition ...
Edit Transition Dialog	Text fields Transition Name // other text fields for editable transition attributes as shown in the domain model Text areas Transition Code ...
Save State Diagram As File Chooser	File browser Buttons: Save, Cancel

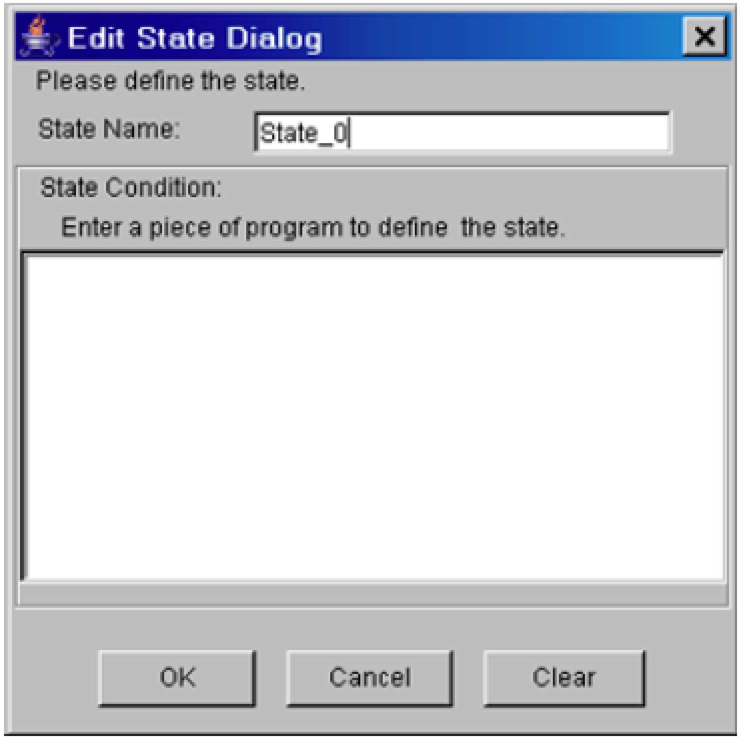
Step 2 – Produce a Draft Layout Design (cont.)

- Perform the steps above (cont.)

  2. The latter part of this presentation deals with design aspects of user interface and should be used when applying a draft layout design. As those slides bring out it is important to stay with existing industry layouts to facilitate user learning curve. For example, there are consensus layout designs for the user interfaces of IDEs and document editors.

Layout Design of State Diagram Editor





Please define the state.

State Name: State_0

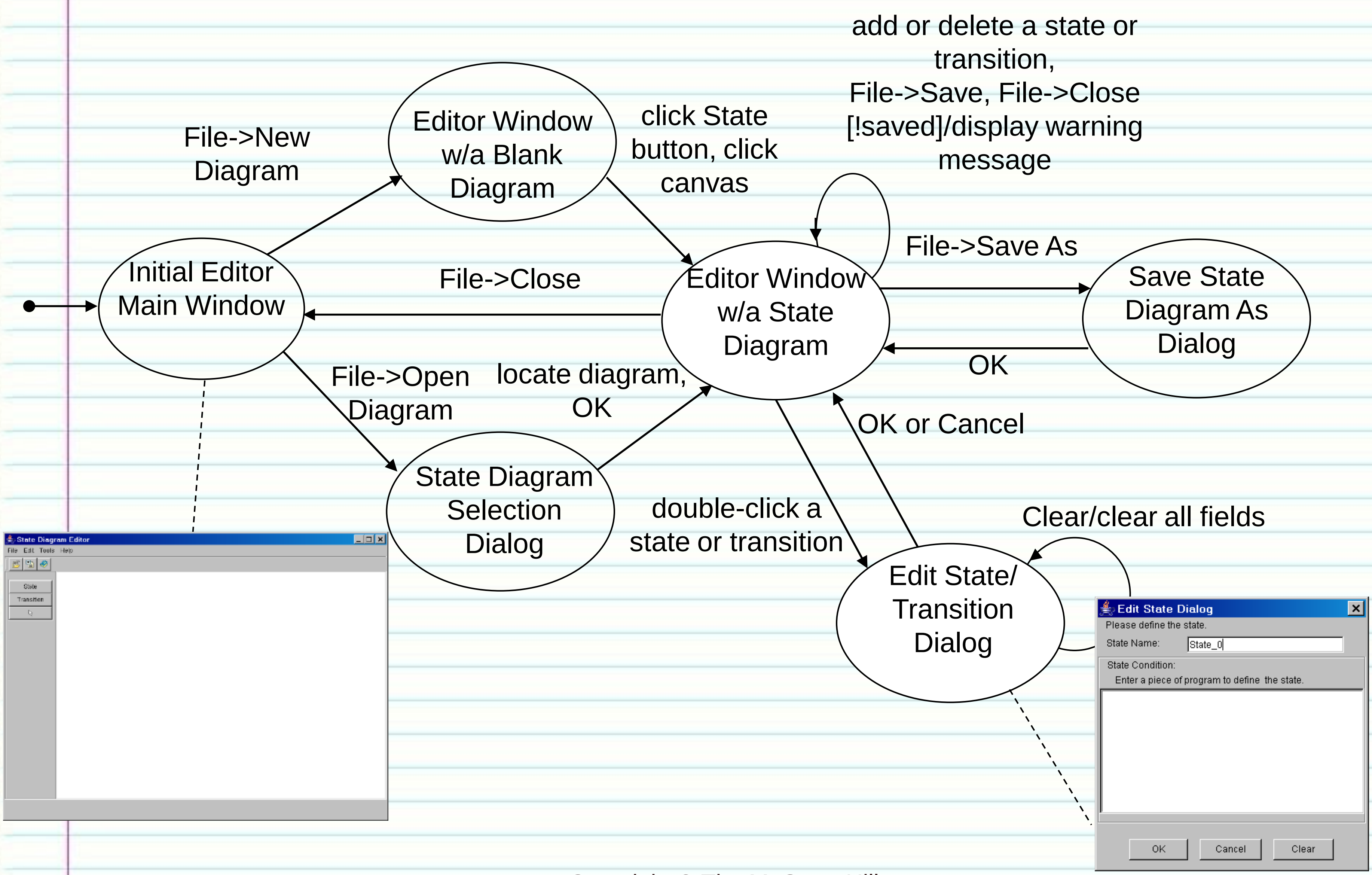
State Condition:

Enter a piece of program to define the state.

Step 3 - Specifying Interaction Behavior

- In this step the interactive behavior of the user interface is designed and specified using a state diagram.
- The states of the diagram represent the windows and dialog boxes and the transitions (events) represent the user input and user actions that cause the change from one window to dialog box to another.
- The states are annotated with the windows and dialog boxes they display, as shown on the next slide.
- This helps the designer visualize the transitions from one display to another.
- This helps to ensure that allowable options from one state to the next are well thought out.

State Diagram Editor Behavior (partial)


add or delete a state or transition,
File->Save, File->Close
[!saved]/display warning message
File->New Diagram
Editor Window w/a Blank Diagram
click State button, click canvas
File->Save As
Initial Editor Main Window
File->Close
Editor Window w/a State Diagram
Save State Diagram As Dialog
OK
File->Open Diagram
locate diagram, OK
OK or Cancel
State Diagram Selection Dialog
double-click a state or transition
Clear/clear all fields
Edit State/ Transition Dialog
State Diagram Editor
File
Edit
Tools
Help
State
Transition
Edit State Dialog
OK
Cancel
Clear

Please define the state.

State Name: State_0

State Condition:

Enter a piece of program to define the state.

Step 4 - Constructing a Prototype

- Types of prototypes
  - Static approaches generate non-executable prototypes.
    - Here the layout design is depicted on paper or in a slide show tool or word processor tool
    - This is effective in showing the look of individual windows and widgets but does not convey the transition between these very well (behavioral)
  - Dynamic approaches generate executable prototypes using a prototyping tool.
    - The prototype can range from simple to fully implemented behavior
    - This is effective in not only showing the look of individual windows and widgets but also the behavior
    - A full behavioral approach can take considerable effort to develop, but if the prototype is usable in the deliverable product then it has many advantages

Step 4 - Constructing a Prototype (cont.)

- Types of prototypes (cont.)
  - Hybrid approaches construct static prototypes during the initial stage of prototype development and switch to dynamic prototyping later after the users are satisfied with the look and feel of the static prototypes.

Step 5 - Evaluating User Interfaces with Users

- User Interface is the first thing that a user sees - it does a lot to sell them on the quality of the software. So it follows that we want to engage with the user as soon as possible to nail down this aspect of the software.
- Here are some of the approaches used to evaluate the interface design with the user
  - User interface presentation. This is performed in the initial stages where the developer shows the windows, widgets, dialog boxes while tracing transitions of the state diagram. This process is iterative and uses a depth-first traversal until all transitions are traversed.
  - User interface demonstration. Static and/or dynamic prototypes are used.
  - User interface experiment. Here the developer demonstrates the prototype and lets the user experiment with it for a short period.
  - User interface review meeting. This approach involves user interface design experts, domain experts, user representatives, and developers.

Step 5 - Evaluating User Interfaces with Users (cont.)

- Here are some of the approaches used to evaluate the interface design with the user (cont.)
  - User interface survey. This approach uses a survey to solicit feedback from the users, user domain experts, and user interface design experts.

User Support Capabilities

- User support capabilities include online documentation, context-dependent help, error messages, and recovery.
- Online help should let the user find the needed information easily.
- Context-dependent help is a user-friendly design technique. Chain of responsibility supports this.
- Error messages should be user-oriented, rather than developer-oriented, and be easy to understand.

User Interface Design Review Checklist

1. Is the overall user interface design consistent with the standard user interface design in each window?
2. Does each window, dialog box, and widget have a simple appearance to allow the user to focus on the main theme of interaction?
3. Is the user interface behavior consistent with the behavior specified in the expanded use case?
4. Does the user interface design make the system easy to learn and use?
5. Are the GUI widgets used correctly and descriptive texts precise and informative from a new user standpoint?
6. Are the labeling of the GUI widgets and descriptive texts precise and informative from a new user standpoint?
7. Are the icons used properly and informative of their function?
8. Is the system output logically and properly structured and organized to facilitate understanding?
9. Are the messages clear and clearly displayed?

User Interface Design Review Checklist (cont.)

10. Are the error messages free of implementation detail, informative, and helpful to the user?
11. Are color, blinking, and other user interface techniques used carefully, properly, and in a restrictive manner to avoid distraction?

Tool Support

- There are many tools available that support user interface design for desktop, web-based and mobile applications.
- Examples for desktop applications include:
  - Adobe Flash Builder
  - Eclipse Window Builder Pro GUI Designer
  - Microsoft Visual Studio Windows Form Designer, and
  - Net-Beans GUI Builder plug-in
- Examples for web applications include:
  - Microsoft Visual Studio Web Designer
  - Adobe Dream Weaver
- Examples for mobile applications include:
  - NetBeans Visual Mobile Designer (Java ME Devices)

QUIZ MATERIAL ENDS HERE

Guidelines For User Interface Design

User Interface Is All Important

- Users first impression of a system is through its interface rather
- A poorly designed interface can cause a user to make catastrophic errors
- Poor user interface design is the reason why so many software systems are never used or never go anywhere
- Most users of business systems interact with these systems through graphical user interfaces (GUIs)
- In some cases, legacy text-based interfaces are still used

What I Really Want

“What I really would like is a system that reads my mind.

It knows what I want to do before I need to do it and makes it very easy for me to get it done."

Golden Rules of User Interface Design

- Strive for consistency
- Enable short-cuts for frequent users
- Don’t assume everyone is a novice user
- Informative feedback
- Design dialogs to yield closure
- Offer simple error handling
- Permit easy reversal of actions
- Support internal locus of control
- Reduce short-term memory load on user
- Don't compel unilateral UI changes on users w/o feedback first

Place User in Control

- Define interaction in such a way that the user is not forced into performing unnecessary or undesired actions
- Provide for flexible interaction (users have varying preferences)
- Allow user interaction to be interruptible and reversible
- Streamline interaction as skill level increases and allow customization of interaction
- Hide technical internals from the casual user
- Design for direct interaction with objects that appear on the screen

Reduce Memory Load

- Reduce demands on user's short-term memory
- Establish meaningful defaults
- Define intuitive short-cuts
- Visual layout of user interface should be based on a familiar real world metaphor
- Disclose information in a progressive fashion

Make Interface Consistent

- Allow user to put the current task into a meaningful context
- Maintain consistency across a family of applications
- If past interaction models have created user expectations, do not make changes unless there is a good reason to do so
- Use the same input techniques across all screens

Ask questions to understand the users

- Trained professionals, technicians, clerical, manufacturing users?
- Level of education
- Learn from written materials or is class needed?
- Expert typists or phobic
- Age range
- One gender predominates?
- Compensation for users for their work is how?
- Use during normal work hours or until job is done?
- Integral part of job or infrequent use?
- What is primary language of users?
- What are consequences of mistakes?
- Are users subject matter experts?
- Knowledge about underlying technology needed?

Data Entry Guidelines

- Consistency
- Minimal user input actions
- Minimal memory load on user
- Compatibility between data entry and data display
- Flexible user control

Getting User’s Attention

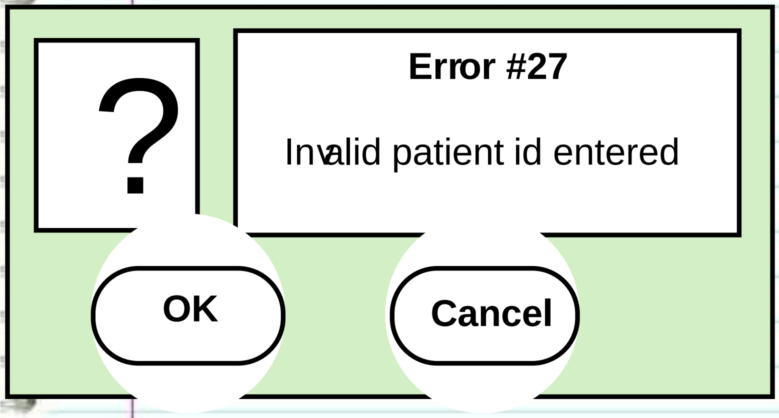
- Remember the direction of human reading (top to bottom, left to right)
- Intensity (2 levels only)
- Marking (e.g. underscore)
- Fonts (up to 3)
- Inverse video
- Blinking (2 to 4 hertz)
- Color (up to 4 standard colors)
- Color blinking
- Audio

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

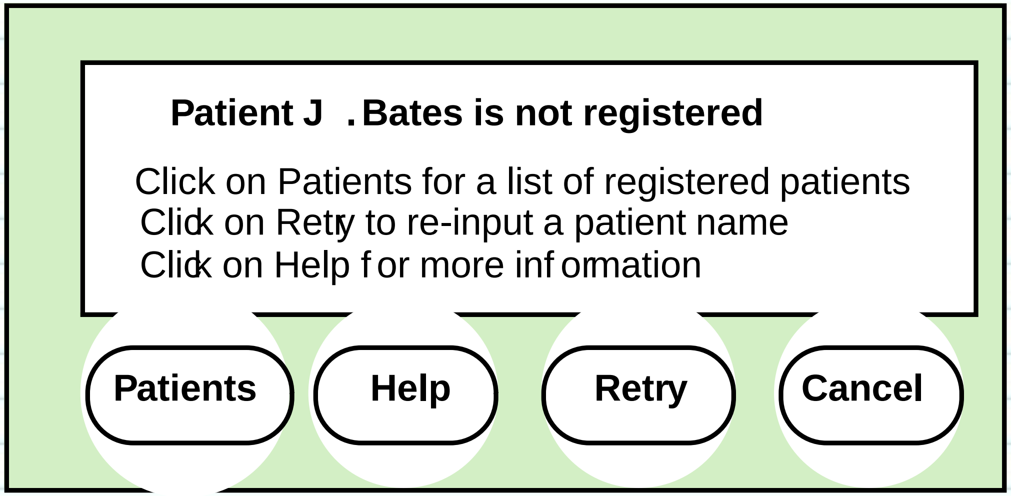
System and user-oriented error messages

System-oriented error message


Error #27

Invalid patient id entered

User-oriented error message


Patient J .Bates is not registered

Click on Patients for a list of registered patients
Click on Retry to re-input a patient name
Click on Help for more information

Bjarne Stronstrup Quote

- “I have always wished that my computer would be as easy to use as my telephone.
- My wish has come true. I no longer know how to use my telephone.”