Identifying Objects

A Case Study and Class Exercise

Ward Cunningham
Computer Research Laboratory
Tektronix, Inc

- Outline

I. Division of Responsibility
   Waveform Example

II. A Method of Design
   Drawing Editor Example

III. Class Exercise
   Bank Machine Problem

IV. A Design Tool
   HyperCard Stack

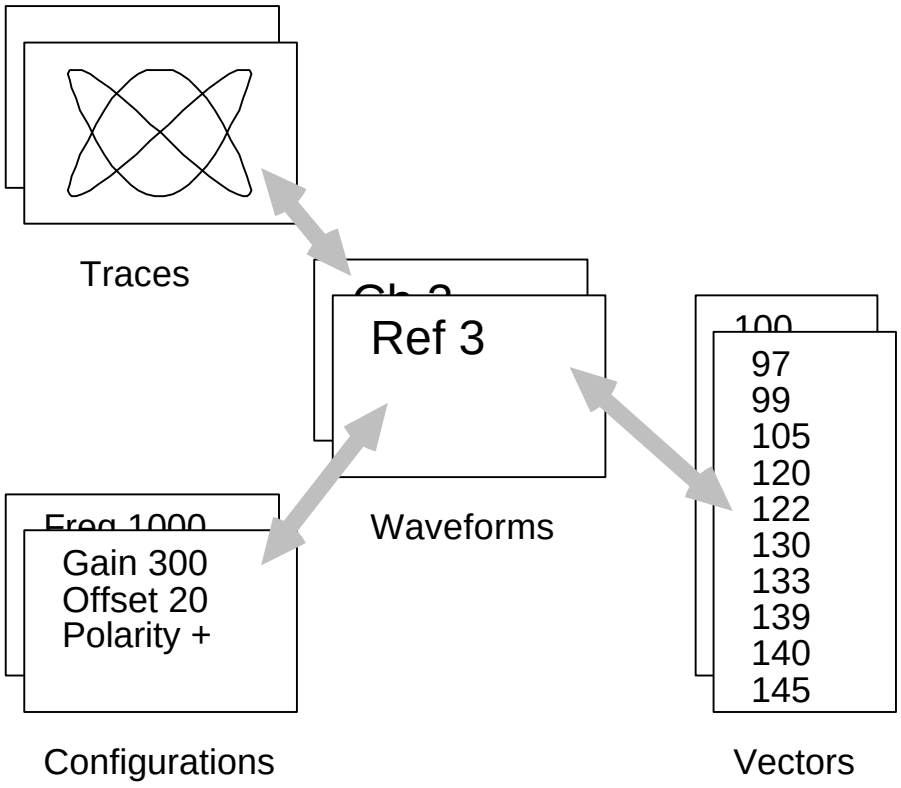
- Object-Oriented Programming

1. Identify Objects
2. Design Protocol
3. Factor Hierarchy
4. Implement Methods

increasing difficulty

- Identifying Objects
  - Model Computation as an activity of collaborating agents (objects).
  - Define the classes to which objects will belong.
  - Distribute responsibility (to meet requirements) over classes.

- Division of Responsibility Example



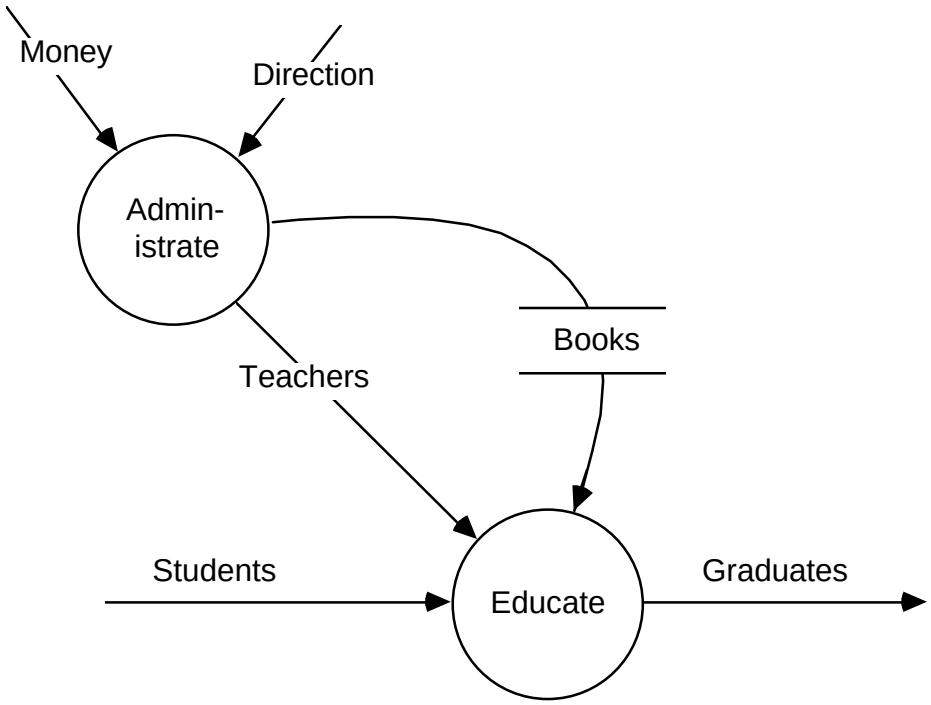
- Trace Responsibilities
    - Render waveforms as visible image. Record display position, scale and emphasis.
- Waveform Responsibilities
    - Interpret samples as a representation of a real signal. Manage signal processing.
- Vector Responsibilities
    - Store sample data. Perform numerical computations. Access signal processor.
- Configuration Responsibilities
    - Configure acquisition circuitry. Record acquisition circumstances.

- Questions
  - What objects need be duplicated when a signal is displayed both normal and magnified?
  - What objects must (should) change when a new trigger occurs?
  - Exactly when does a configuration update after a user-requested change in acquisition parameters?

- Grade School Example
  - Object-oriented description

	Responsiblity	Collaborators
Teacher	Teaches Lessons Evaluates Students	Secretary Student Principal
Student	Learns Lessons	Teacher Principal
Principal	Administers Funds Diciplines Students Hires Staff	Teacher Secretary Student
Nurse	Gives First Aid Gives Vacinations	Students Teachers
Secretary	Answers Phone Prints Handouts	Teacher Principal
Janitor	Cleans Building Fixes Equipment	Teacher Secretary
Cook	Prepares Meals	Janitor

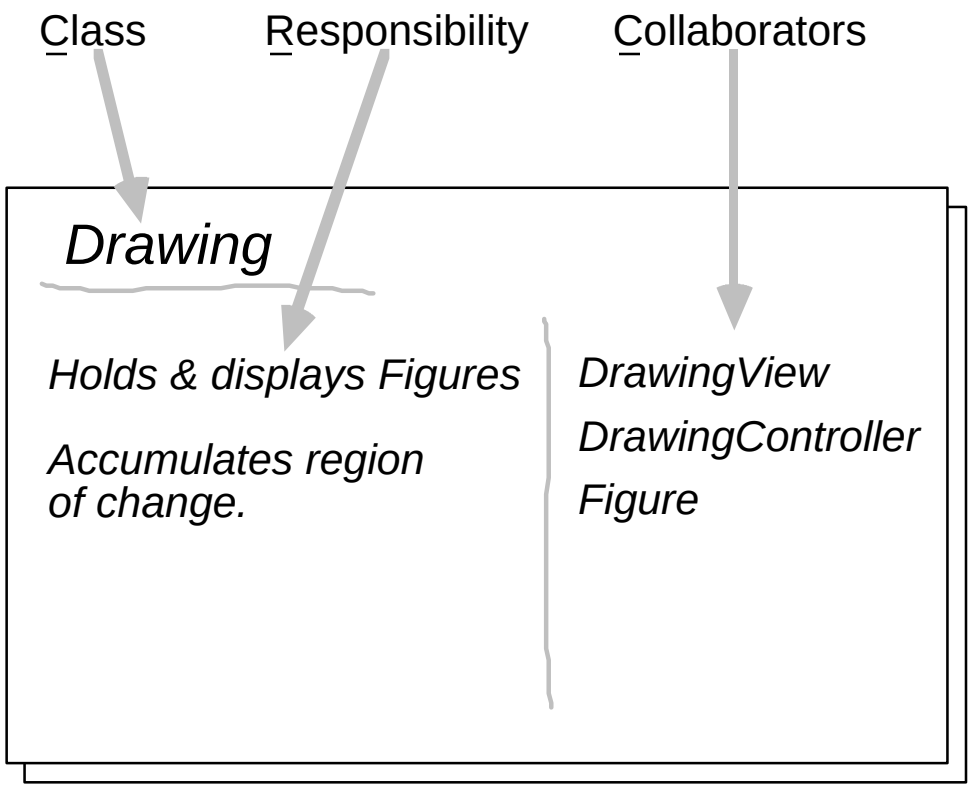
- Grade School Example
  - Process oriented description



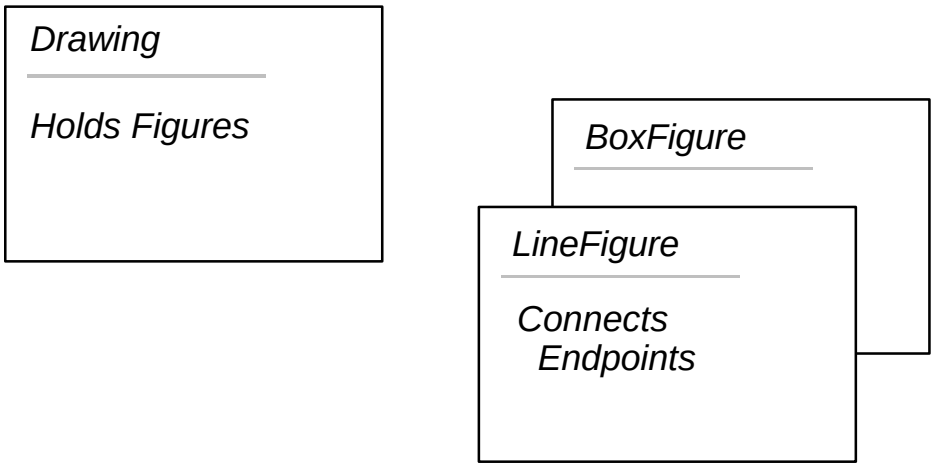
- A Method of Design
  - Make decisions based on wisdom and experience.
  - Record decisions in a structured design document.
  - Test design for adequacy and completeness.
  - Maintain and refer to the design throughout implementation.

- Design Representation
    - Enumerates all (new) classes.
    - Defines responsibilities assumed by members of each class.
    - Describes collaborations through which responsibilities are discharged.

- Introducing CRC Cards

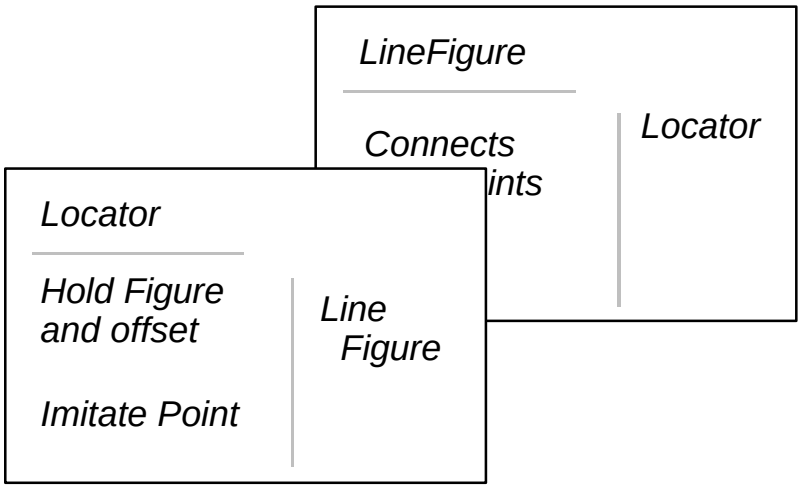


Step 1: Start With Knowns



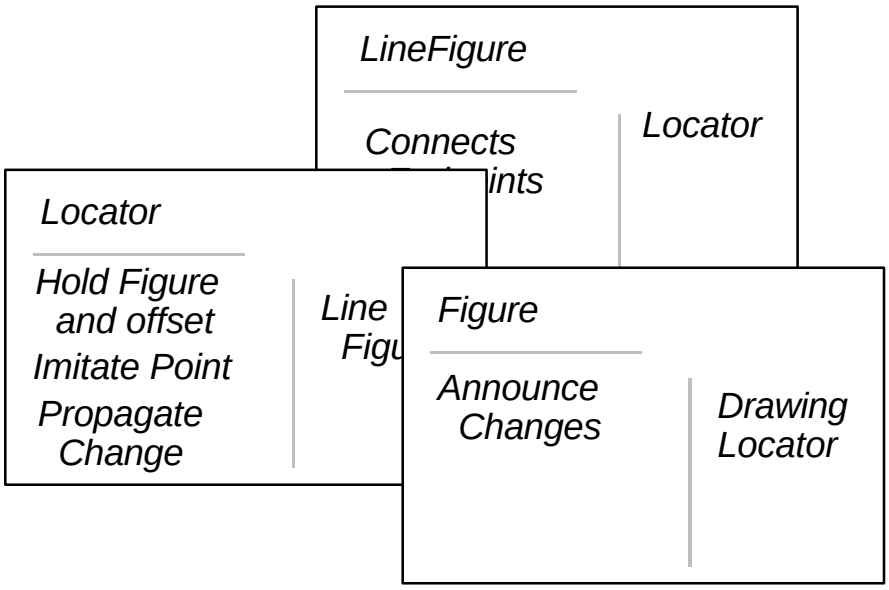
- A Drawing is composed of Figures
- Figures come in several kinds

Step 2: Hypothesize Support



- A line may connect to other figures
- A "smart" point does the work

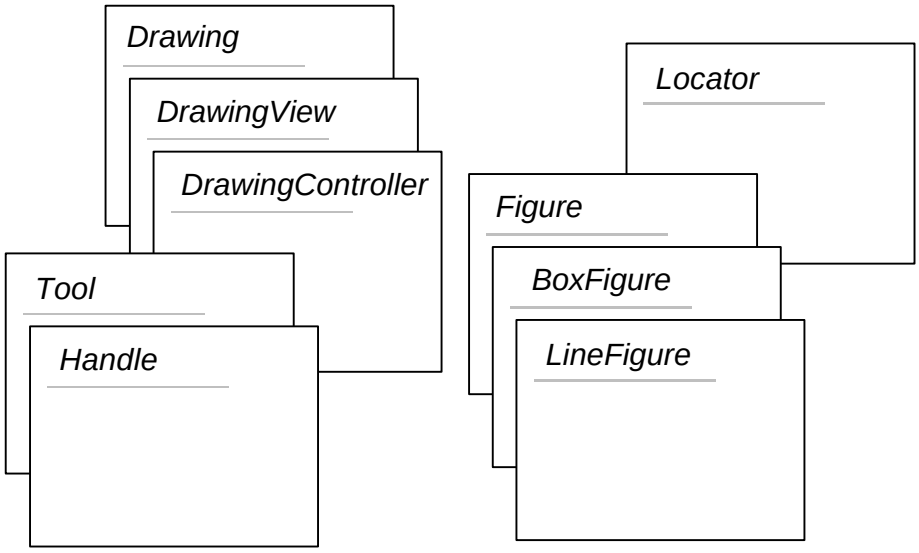
Step 3: Test with Scenarios



- Figures notify the Drawing and dependent Locators when moved.

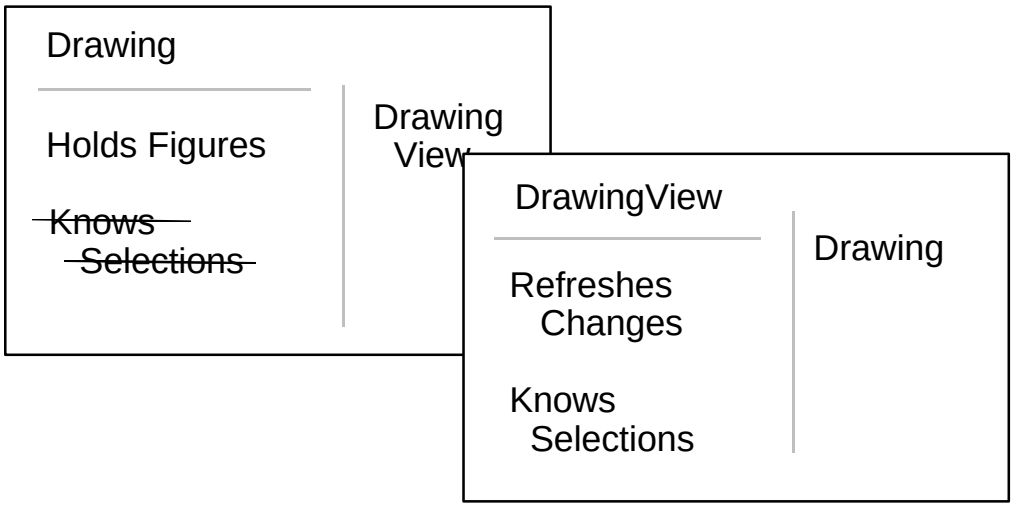
- Change propagates through Locators

Step 4: Try Various Groupings



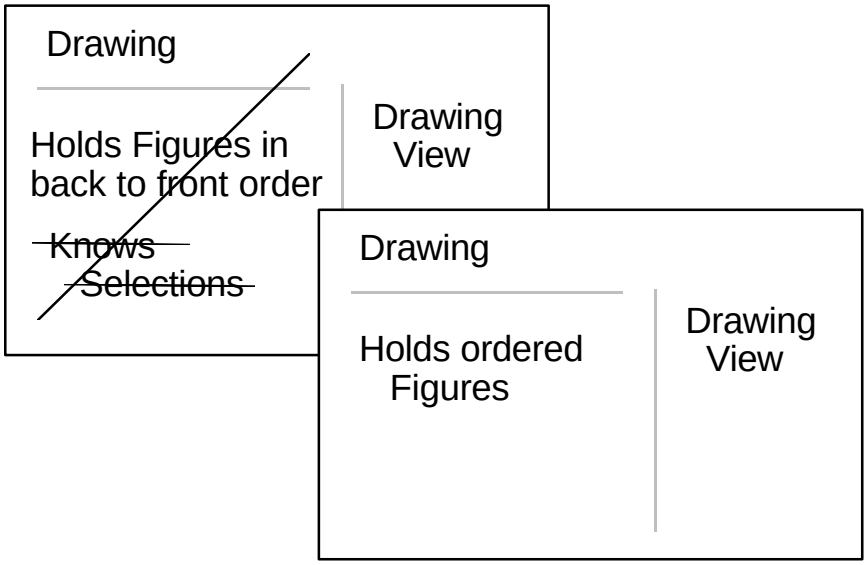
- A Handle is like a Tool ...
- Locators are quite unique ...

Step 5: Redistribute Responsibilities



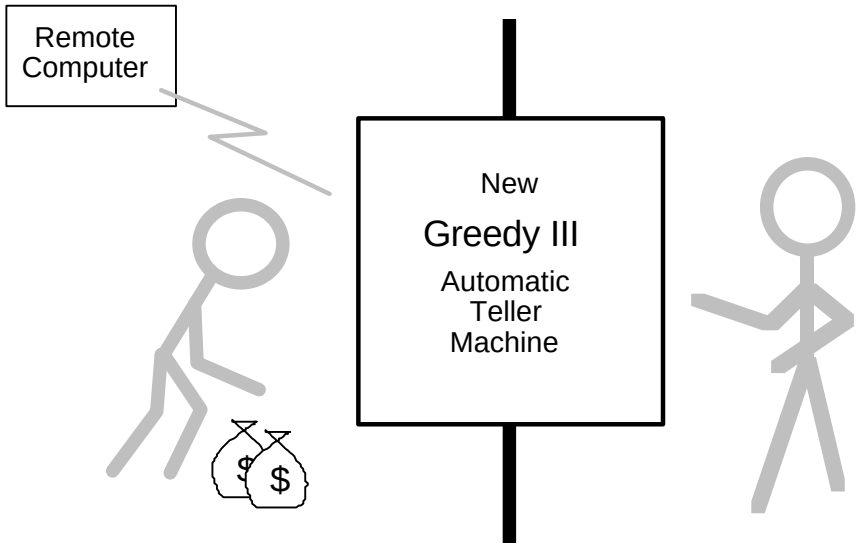
- Selections are kept in the View
- Selections won't be saved with a Drawing

Step 6: Rewrite for Clarity



- It is important that Figures are ordered

- Design Exercise (Requirements)



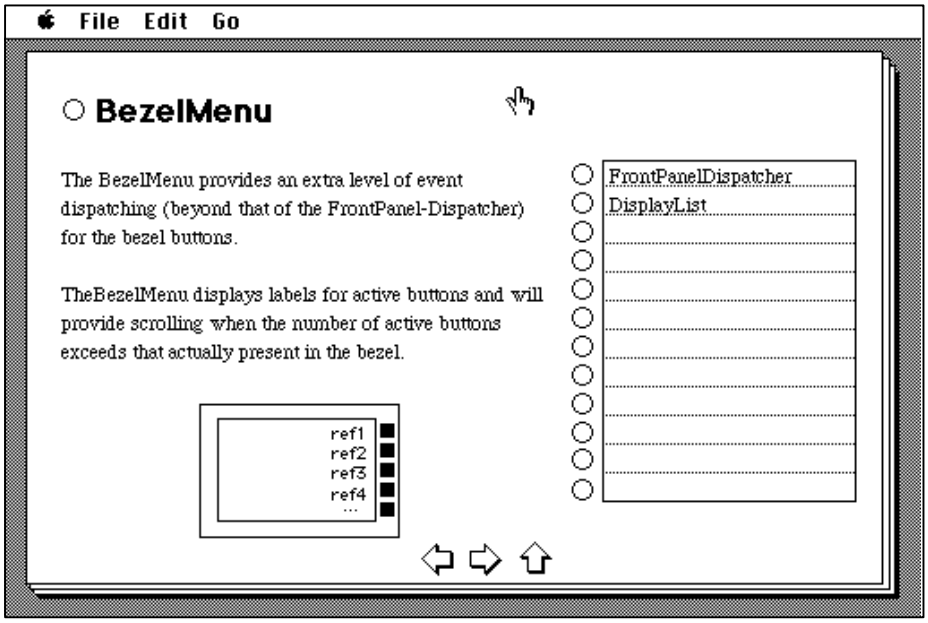
- Design Exercise (Methodology)

    1. Start with Knowns
    2. Hypothesize Support
    3. Test with Scenarios
    4. Try Various Groupings
    5. Redistribute Responsibility
    6. Rewrite for Clarity

- Design Exercise (Schedule)

• Find partner and design system	1 Hr.
• Find new partner and review design	15 Min.

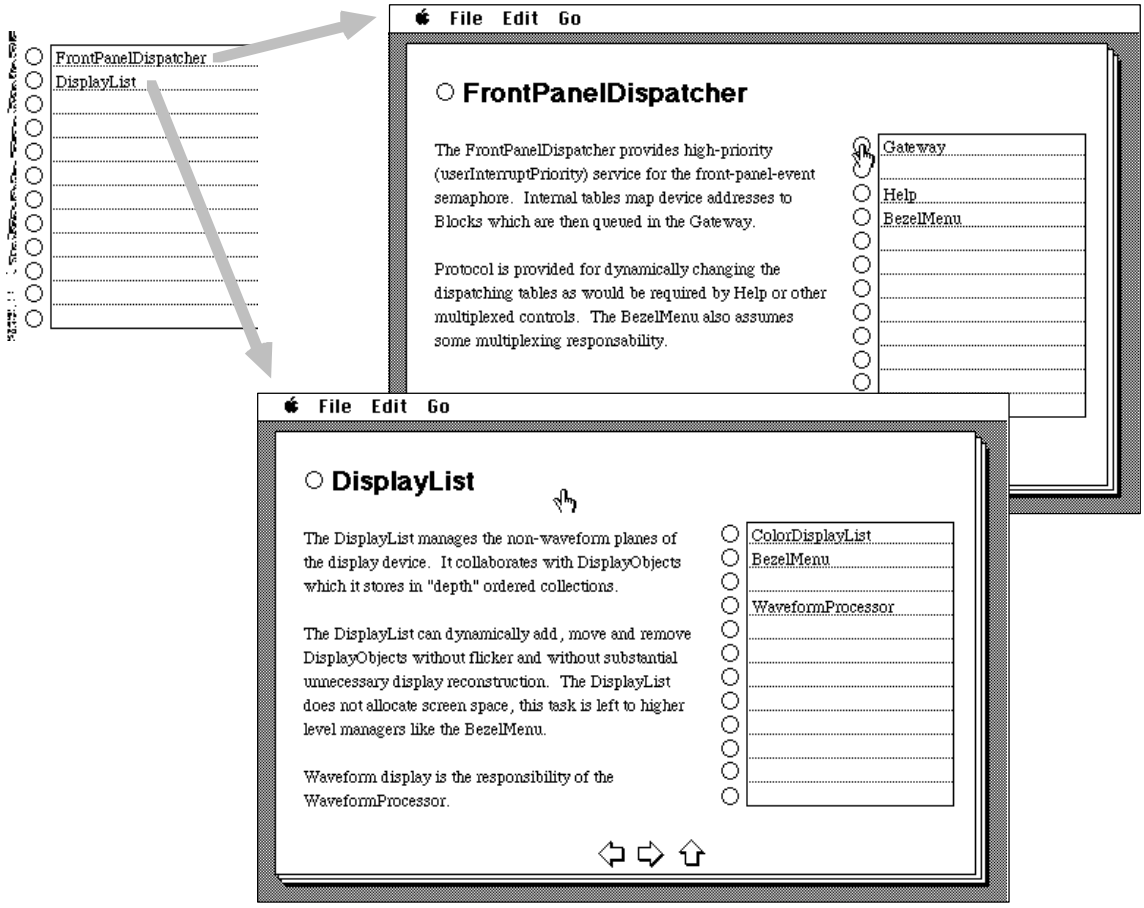
- CRC Stack



- A HyperCard stack for browsing and editing an object-oriented design

- A Machine-readable design aids distribution, maintenance and validation.

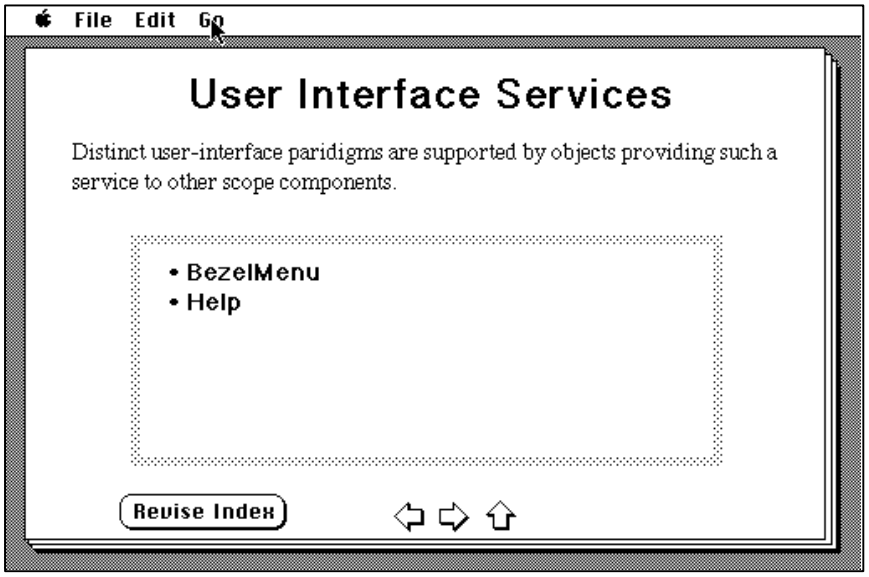
- Browsing Collaborators

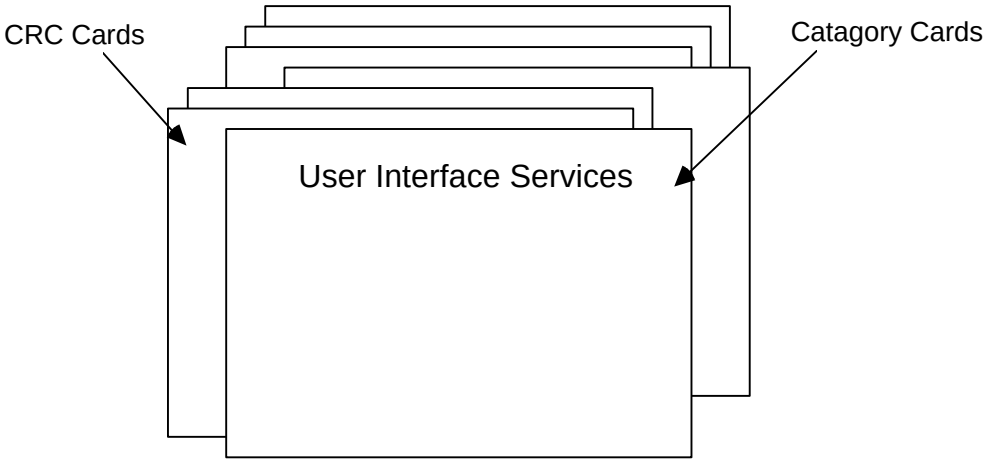


- Click to browse a collaborator, press and hold to create and link a new collaborator card

- Use HyperCard browsing (Find, Recent, etc)

- Category Cards





- Categories organize a completed design as an aid to learning and understanding