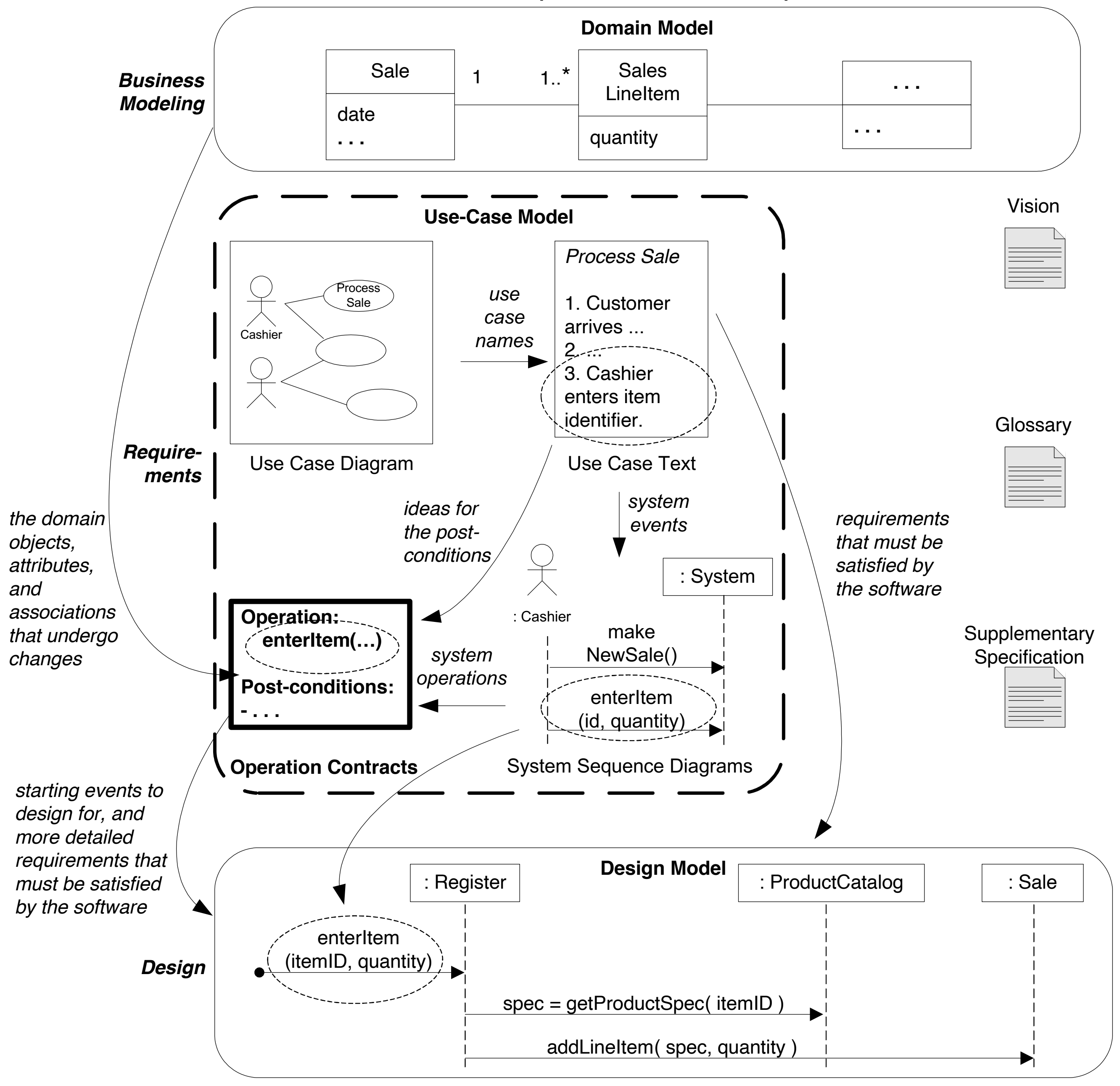
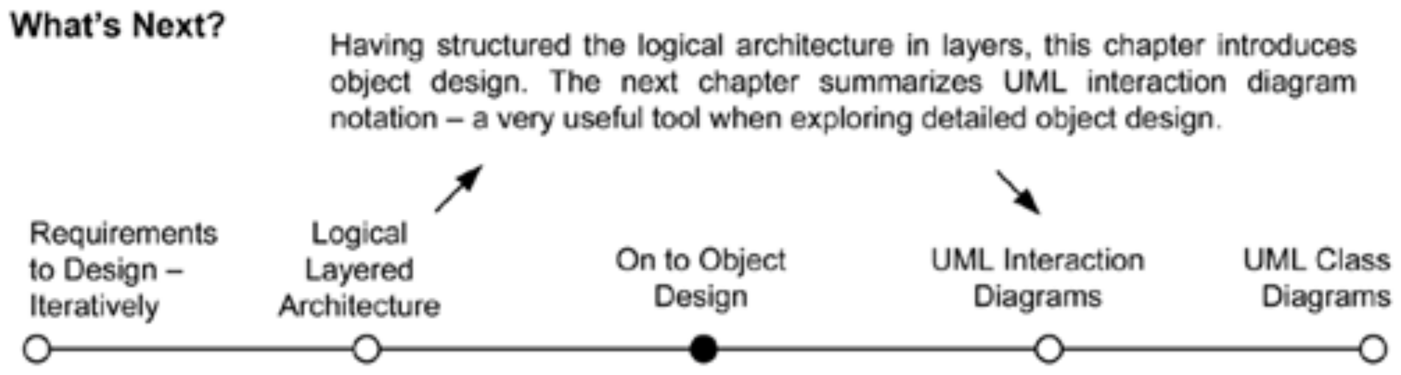
Chapter 14

On To Object Design

Sample UP Artifact Relationships




What's Next?
Having structured the logical architecture in layers, this chapter introduces object design. The next chapter summarizes UML interaction diagram notation – a very useful tool when exploring detailed object design.
Requirements to Design – Iteratively
Logical Layered Architecture
On to Object Design
UML Interaction Diagrams
UML Class Diagrams

- We have some some requirements analysis and domain modeling
  - Use cases, non-functional requirements, system sequence diagrams, operation contracts, domain model…

- Starting Object Design
  - Dynamic and static object design modeling
  - Agile modeling

Starting Object Design

- **Code**: Design-while-coding. From mental model to code
- **Draw, then code:** Drawing some UML on a whiteboard, then coding with a text-strong IDE (such as Eclipse)
- **Only draw:** Somehow, the tool generates everything from diagrams. Many a dead tool vendor has washed onto the shores of this step island. Only draw still involves a text programming language attached to UML elements.

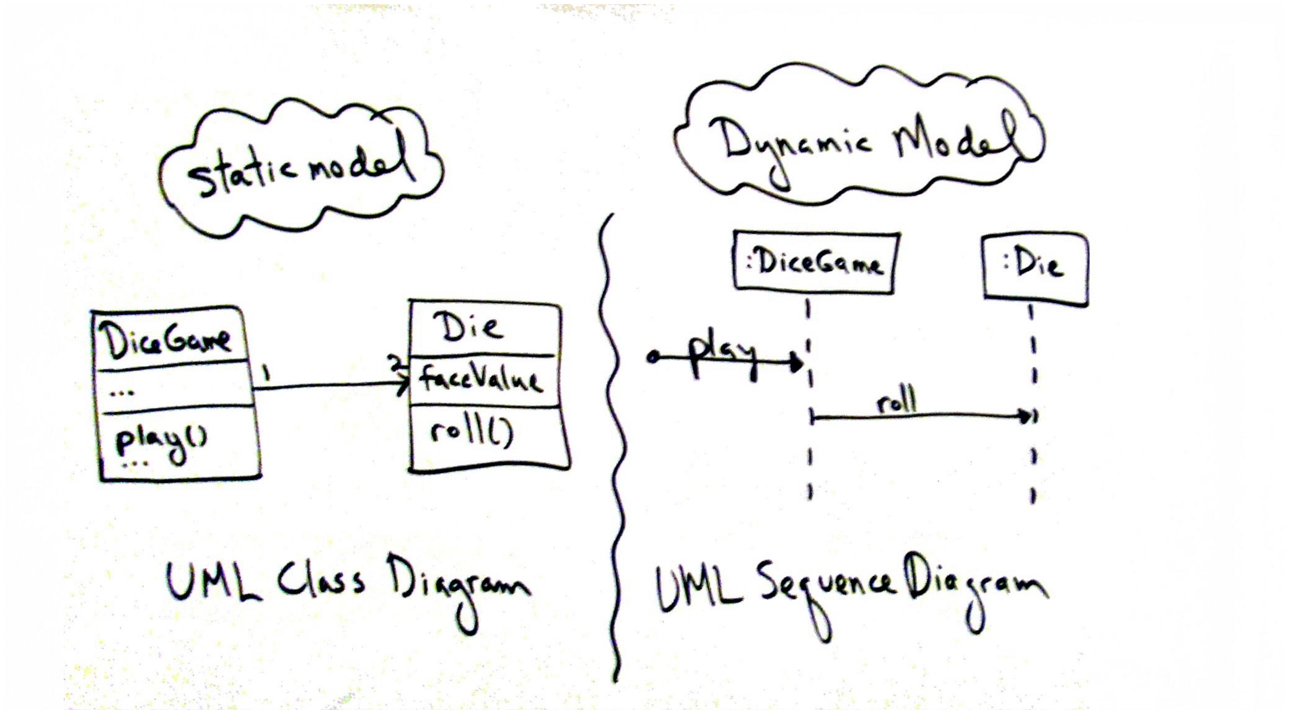
- We emphasize object design and **lightweigth drawing** before coding.

Starting Object Design

- **Agile Modeling**
  - Reduce drawing overhead and model to understand and communicate, rather than to document. Try the simple agile modeling approach. Practices include using lots of whiteboards (ten in a room, not two) using markers, digital cameras, and printers to capture "UML as sketch"one of the three ways to apply UML
  - Modeling with others
  - Creating several models in parallel
- UML CASE Tools:
  - Agile modeling doesn’t mean that UML tools aren’t useful
  - Choose a UML tool that integrates with text-string IDE
  - Many developers find it useful to code awhile their favorite IDE, then press a button, reverse engineer the code, and see a UML big-picture graphical view of their design
  - **Agile modeling on the walls and using a UML CASE tool integrated into a text-strong IDE can be complementary. Try both during different phases of activity.**
  - How much time spent Drawing: spend a few gours at the walls (or UML tool) for the hard parts of the detailed object design, then stop, contrinue with codeing suing diagrams as inspiration. Final design in code cwill diverge and improve.

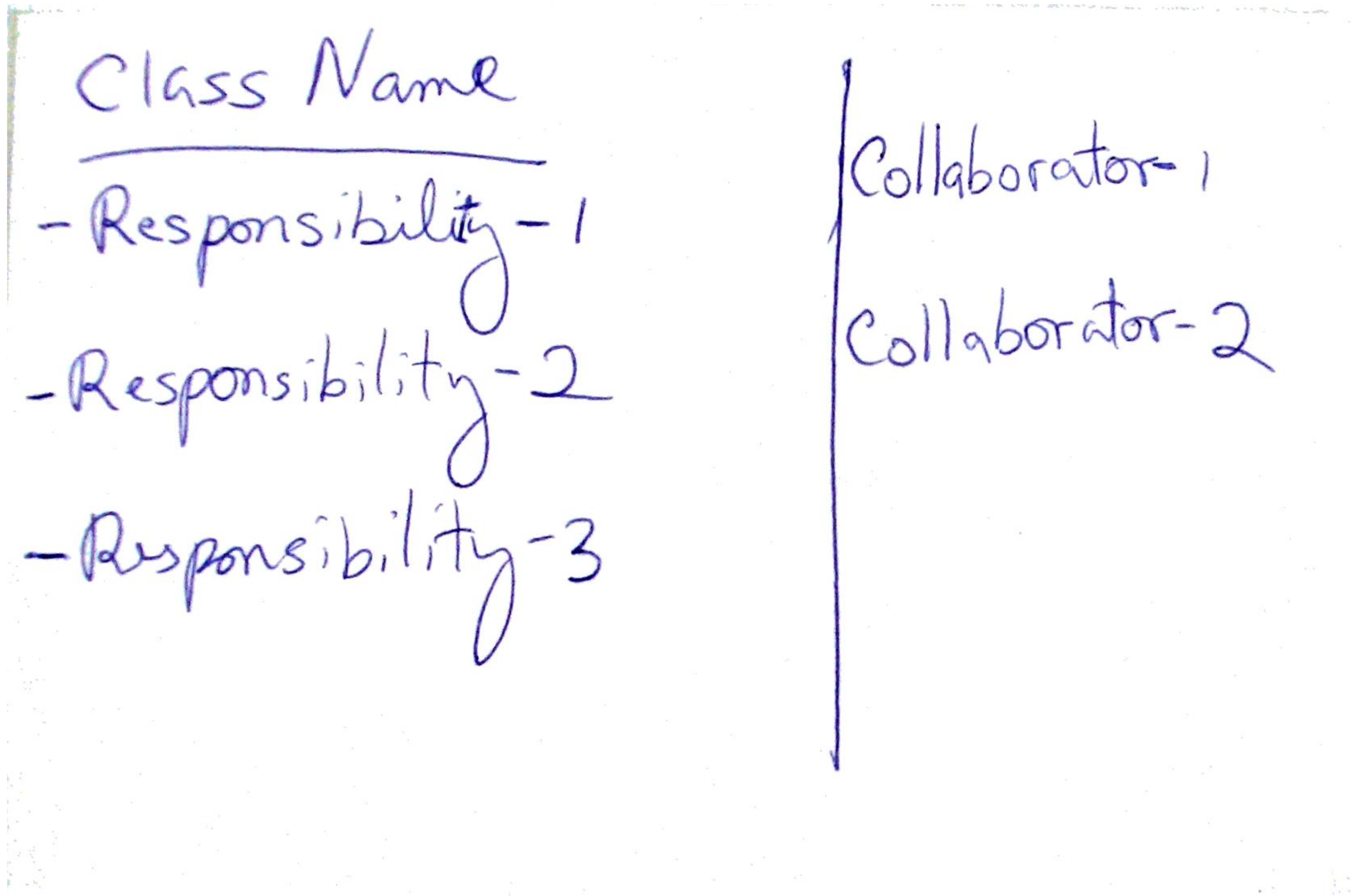
Designing Objecs: Static and Dynamic Modeling

- Two kind of object models:
  - **Dynamic models** as UML interaction diagrams (sequence diagrams, communication diagrams) help design the logic, the behaviour of the code or the method bodies
  - Dynamic models are more interesting and difficult
  - **Static models** as UML class diagrams help design the definition of packages, class names, attributes, method signatures.
  - **Spend time on dynamic models**
    They are the key tool for building good static models

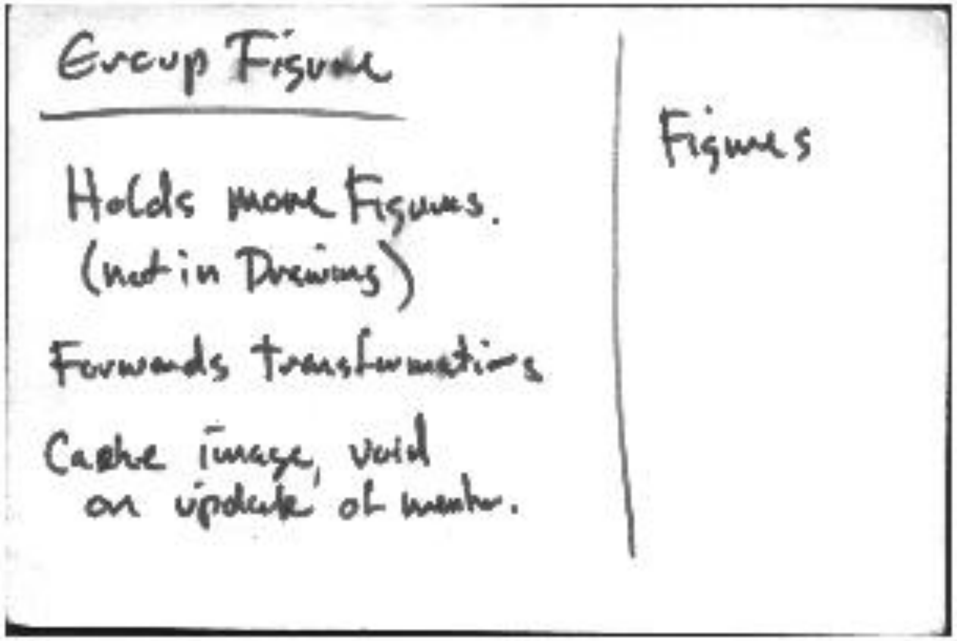


One Object Design Technique: Class Responsibility Collaboration (CRC) cards

- Another popular text-oriented modeling technique is CRC
- Each card represents a class
- A CRC modelign session: a gorup sitting around a table, discussing and writing on the cards as they play 'what if' scenarios with the objects, considering what they do, what other objects collaborate



CRC Card examples



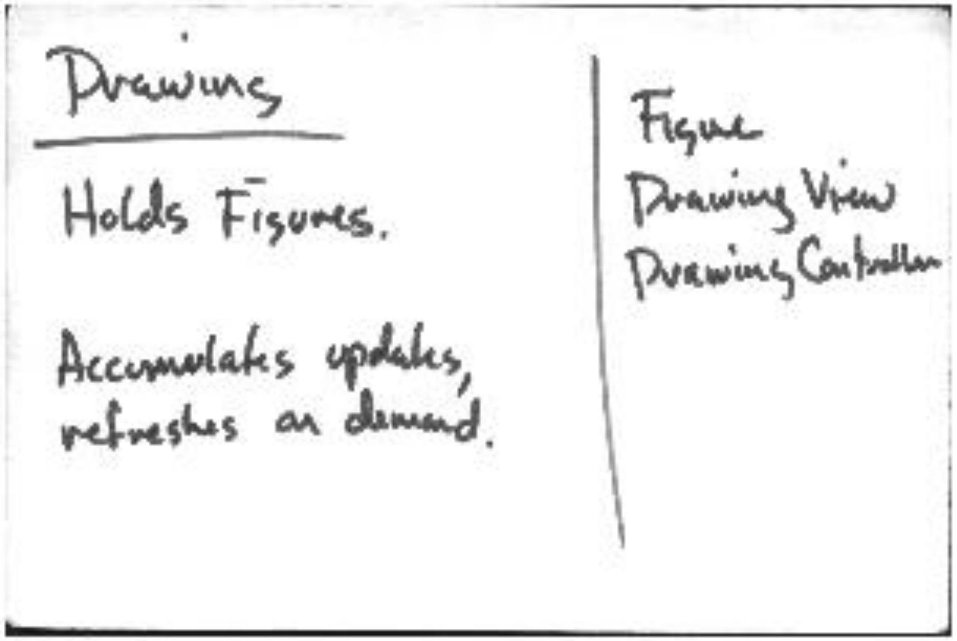
Group Figure

Holds more Figures. (not in Drawing)

Forwards transformations

Cache image, void on update of member.

Figures

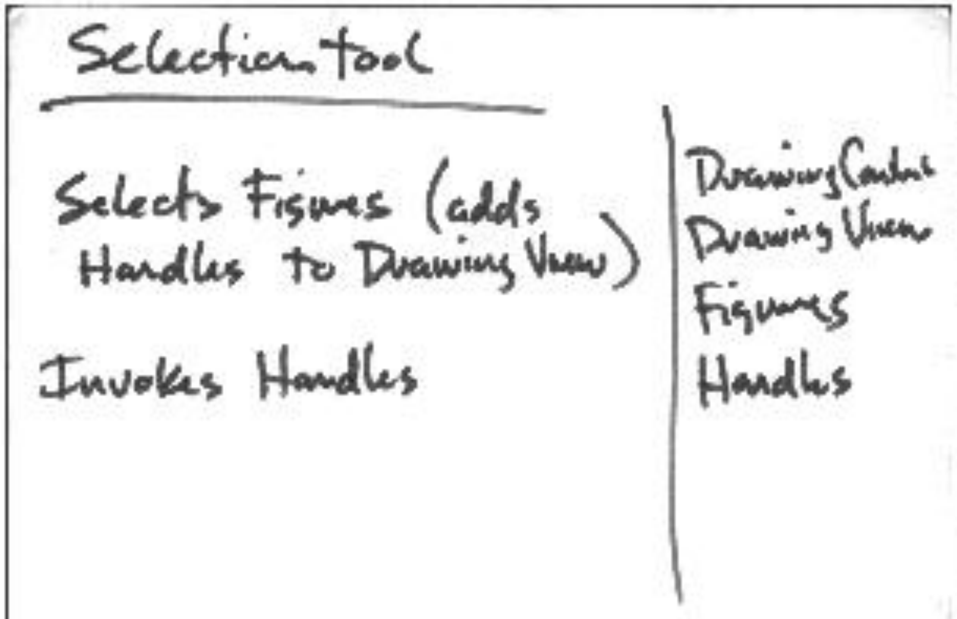


Drawing

Holds Figures.

Accumulates updates, refreshes on demand.

Figure
Drawing View
Drawing Controller

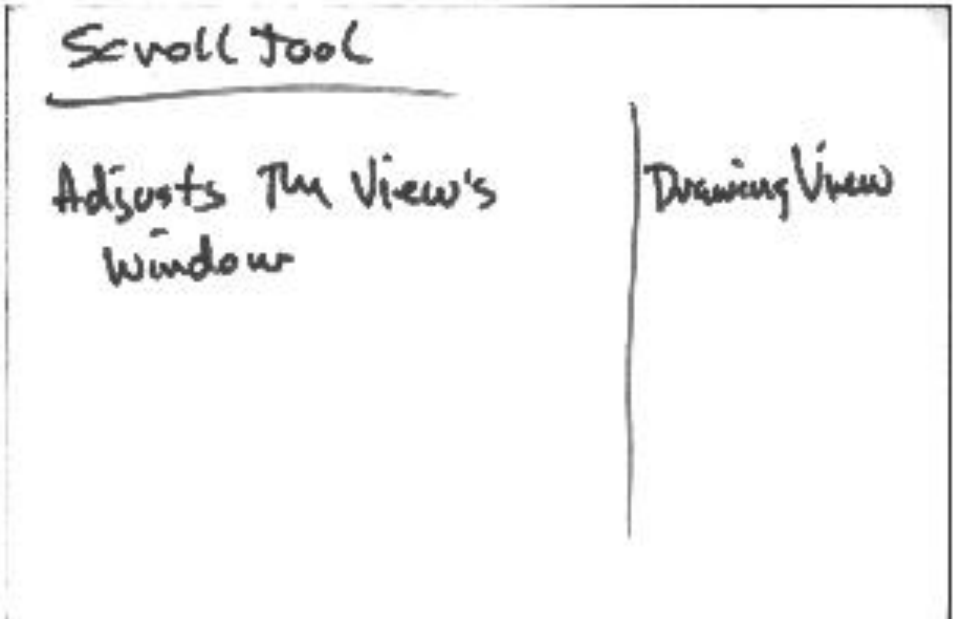


Selection tool

Selects Figures (adds Handles to Drawing View)

Invokes Handles

Drawing Controller
Drawing View
Figures
Handles



Scroll Tool

Adjusts the View's window

Drawing View