Sequence Diagrams

Curt Clifton
Rose-Hulman Institute of Technology

The Story Thus Far

- Z
  - Mathematical models of key properties
  - For safety critical systems
  - Based on set theory and predicate logic
- Program verification
  - Proving code correctness
  - For subsets of critical applications
  - Based on inference rules derived from program semantics

Focused on specification

Coming Attractions

- Graphical depictions of program properties
- UML
  - Sequence diagrams
  - State charts
  - Activity diagrams
- Java Modeling Lang.

- Graphical notations perhaps better for:
  - Initial design
  - Communicating with non-technical clients and colleagues

Focused on design

Sequence Diagrams

- Captures behavior of single scenario
  - Example objects
  - Messages passed
- Easy to sketch on a whiteboard

Example

- An order for which we want to calculate the price
- Need to:
  - Determine price of each item
  - Sum the prices
  - Calculate customer discount

Cartoon of the Day

Revisiting the Example

- Distributed responsibilities
- Note how clear the differences are vs. previous design
- Sequence diagrams:
  - Good for sketching interactions
  - Not good for detailed algorithms

Which is Better OO Design?

- Is centralized control of first example better?
- Is distributed control of second example better?

Creation & Deletion

Creation – participant box not at top

Deletion – big ol' X

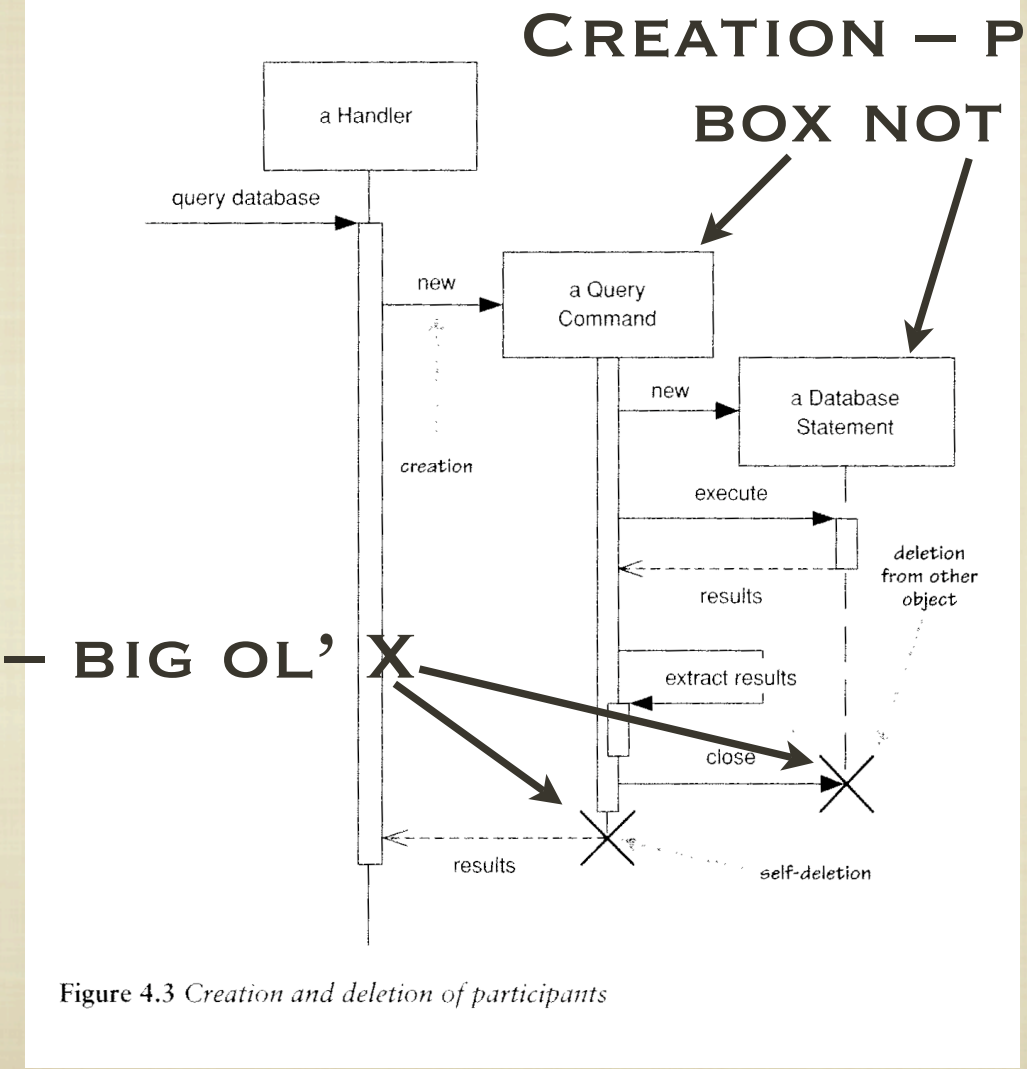

Figure 4.3 *Creation and deletion of participants*

Loops and Other Foo

- Not what sequence diagrams are good for!
- But here's some notation anyway...

Example

```
dispatch() {
  foreach (lineitem)
    if (product.value > $10k)
      careful.deliver(lineitem)
    else
      regular.deliver(lineitem)
    end if
  end for
  if (needsConfirmation())
    msgr.confirm()
}
```

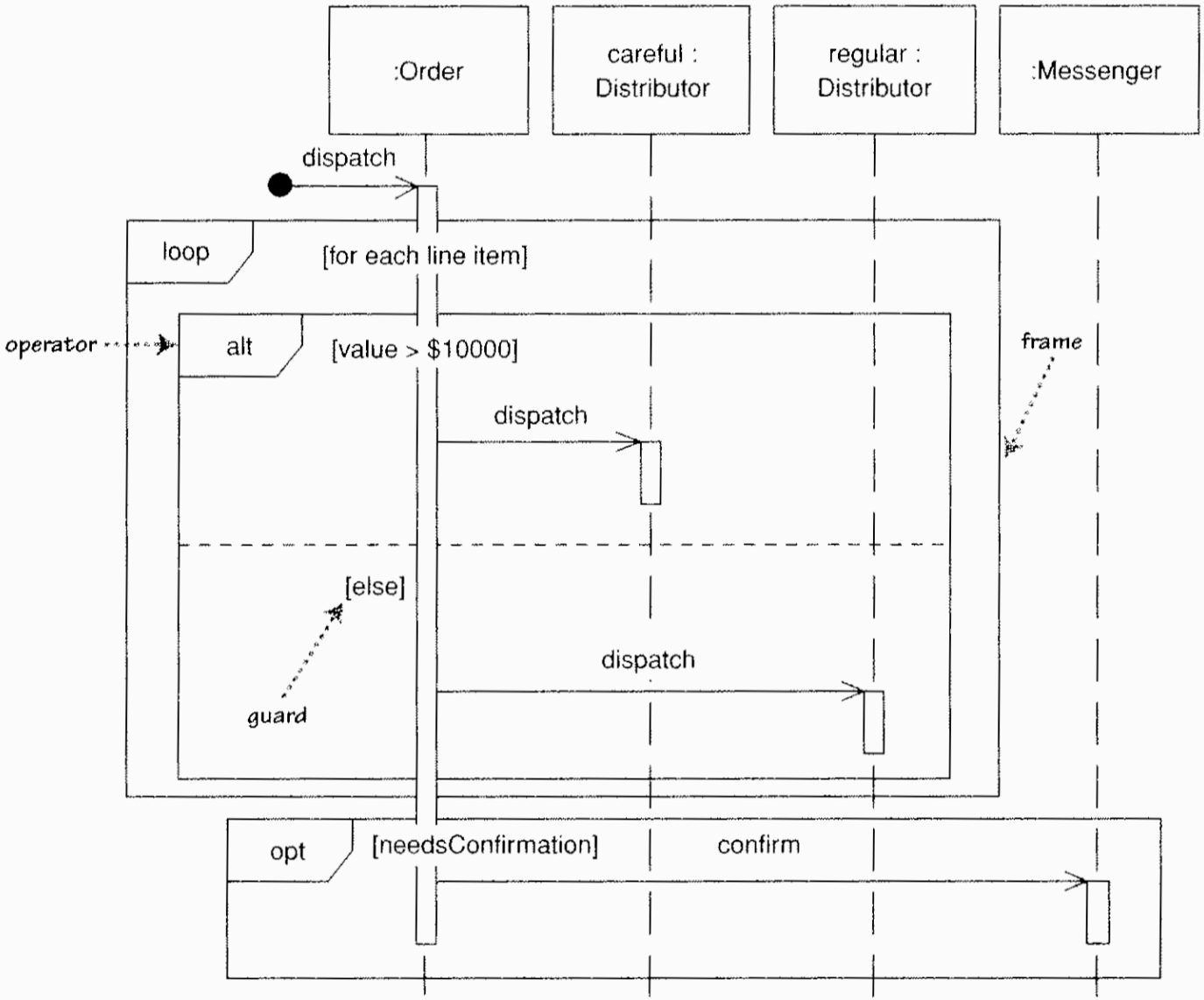

Figure 4.4 *Interaction frames*

Table 4.1 *Common Operators for Interaction Frames*

Operator	Meaning
`alt`	Alternative multiple fragments; only the one whose condition is true will execute (Figure 4.4).
`opt`	Optional; the fragment executes only if the supplied condition is true. Equivalent to an `alt` with only one trace (Figure 4.4).
`par`	Parallel; each fragment is run in parallel.
`loop`	Loop; the fragment may execute multiple times, and the guard indicates the basis of iteration (Figure 4.4).
`region`	Critical region; the fragment can have only one thread executing it at once.
`neg`	Negative; the fragment shows an invalid interaction.
`ref`	Reference; refers to an interaction defined on another diagram. The frame is drawn to cover the lifelines involved in the interaction. You can define parameters and a return value.
`sd`	Sequence diagram; used to surround an entire sequence diagram, if you wish.

Sequence Diagrams

- Good at showing collaborations
- Poor at showing precise definitions

When to Use

- When looking at the behavior of ...
  - Several objects
  - In a single use case
- When your instructor requires you to...

Homework

- Draw sequence diagram using whatever tool but **submit PDF**
- Or draw on paper and scan
- Challenging part of homework is deciphering the specification
  - That's intentional
  - Do your best, state your assumptions