UML Sequence Diagrams

Hoang Huu Hanh, Hue University
hhhanh@hueuni.edu.vn

UML sequence diagrams

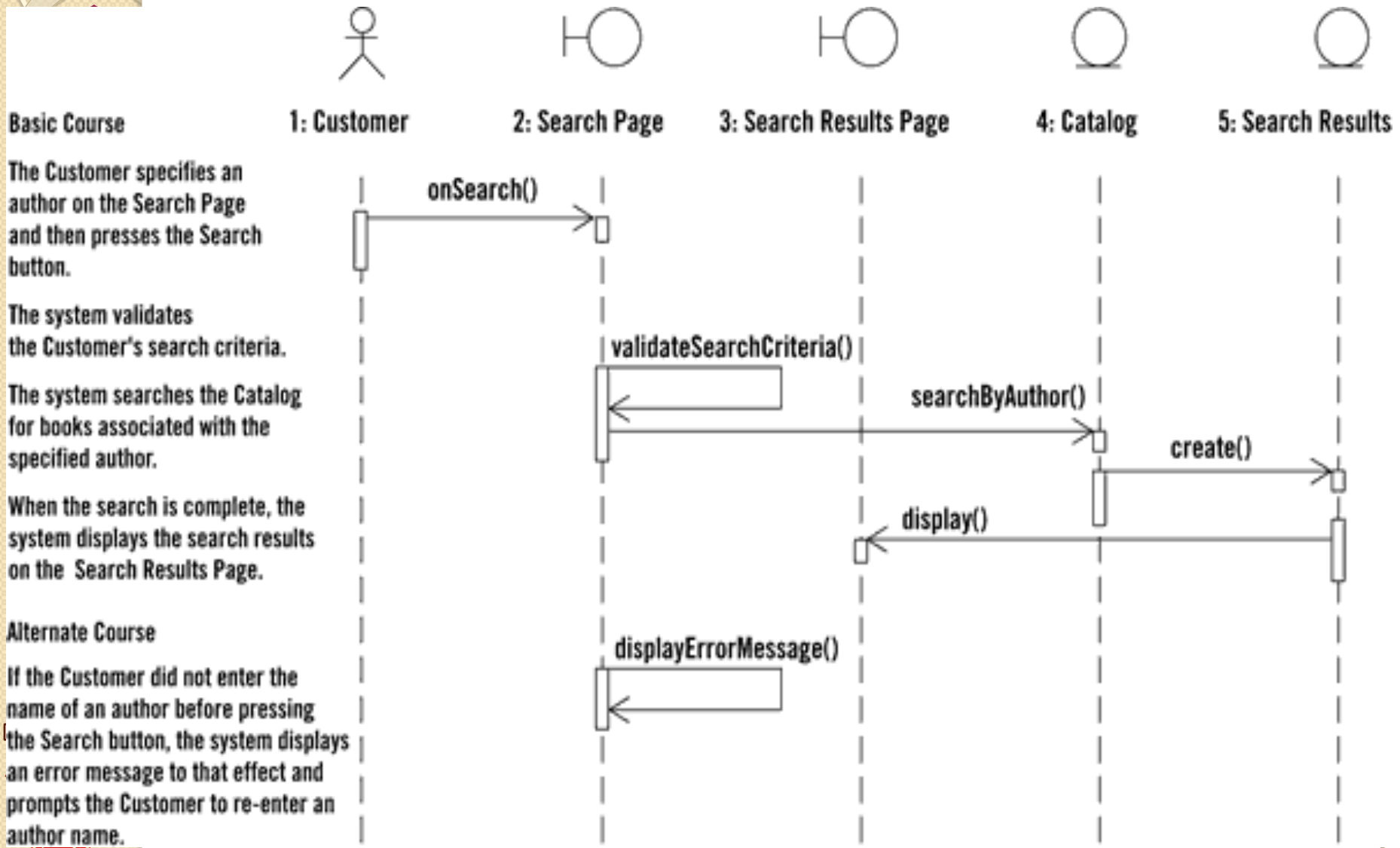
- **sequence diagram**: an "interaction diagram" that models a single scenario executing in the system
  - perhaps 2nd most used UML diagram (behind class diagram)

- relation of UML diagrams to other exercises:
  - CRC cards -> class diagram
  - use cases -> sequence diagrams

Key parts of a sequence diag.

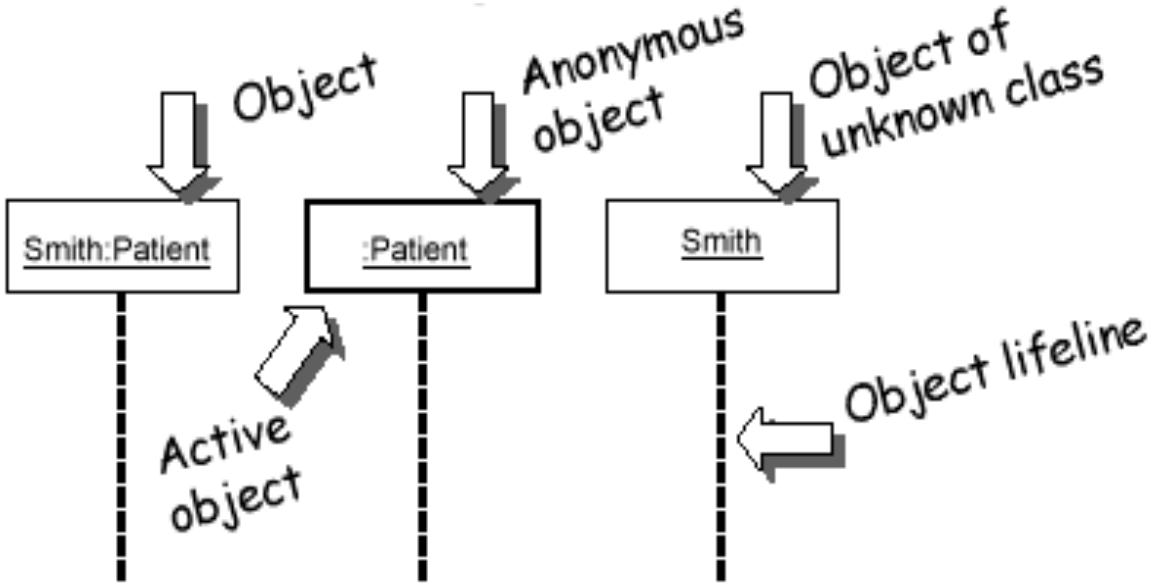
- **participant**: an object or entity that acts in the sequence diagram
  - sequence diagram starts with an unattached "found message" arrow
- **message**: communication between participant objects
- the axes in a sequence diagram:
  - horizontal: which object/participant is acting
  - vertical: time (down -> forward in time)

Sequence diag. from use case



Representing objects

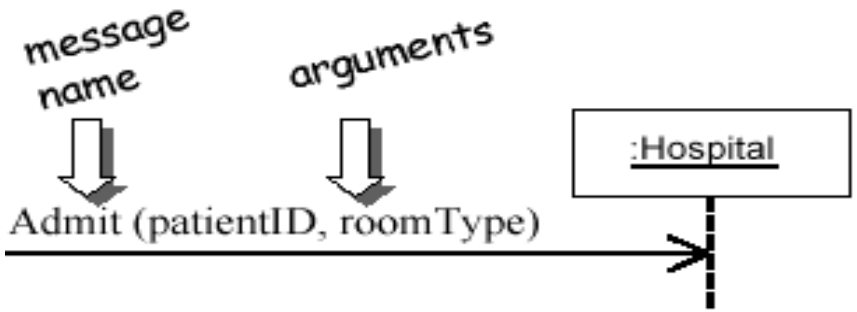
- squares with object type, optionally preceded by object name and colon
  - write object's name if it clarifies the diagram
  - object's "life line" represented by dashed vert.



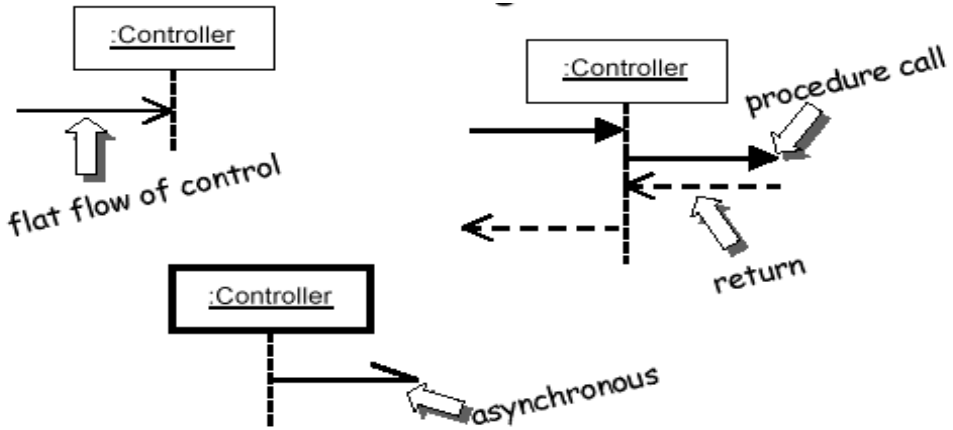
Name syntax: <objectname>:<classname>

Messages between objects

- message (method call) indicated by horizontal arrow to other object
  - write message name and arguments above arrow



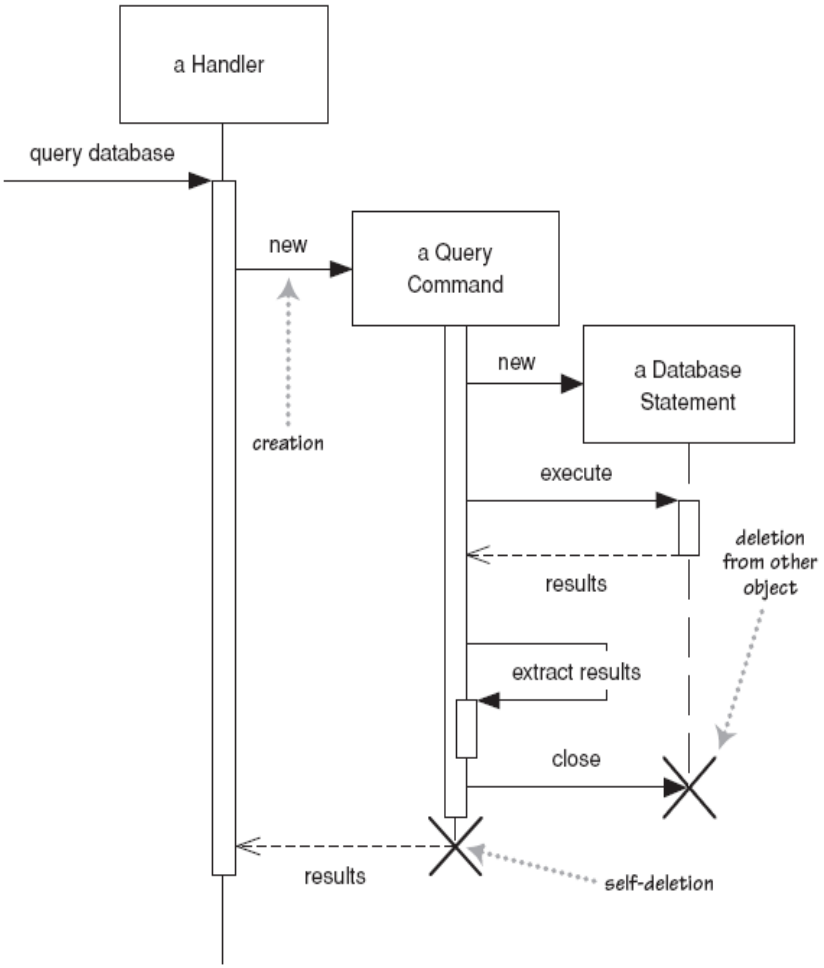
  - dashed arrow back indicates return
  - different arrowheads for normal / concurrent (asynchronous) methods



Hue University

Lifetime of objects

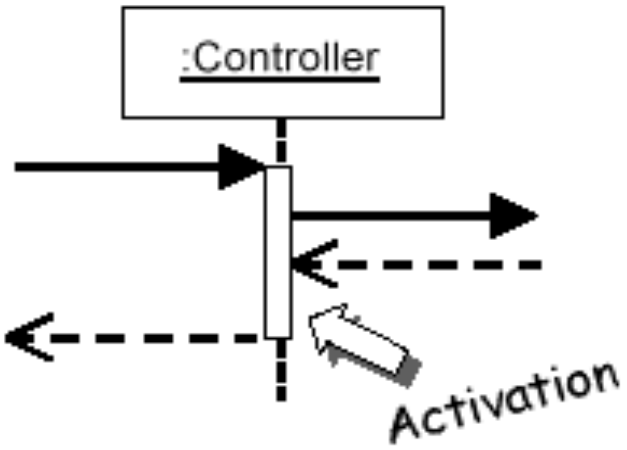
- creation: arrow with 'new' written above it
  - notice that an object created after the start of the scenario appears lower than the others
- deletion: an X at bottom of object's lifeline
  - Java doesn't explicitly delete objects; they fall out of scope and are garbage-collected

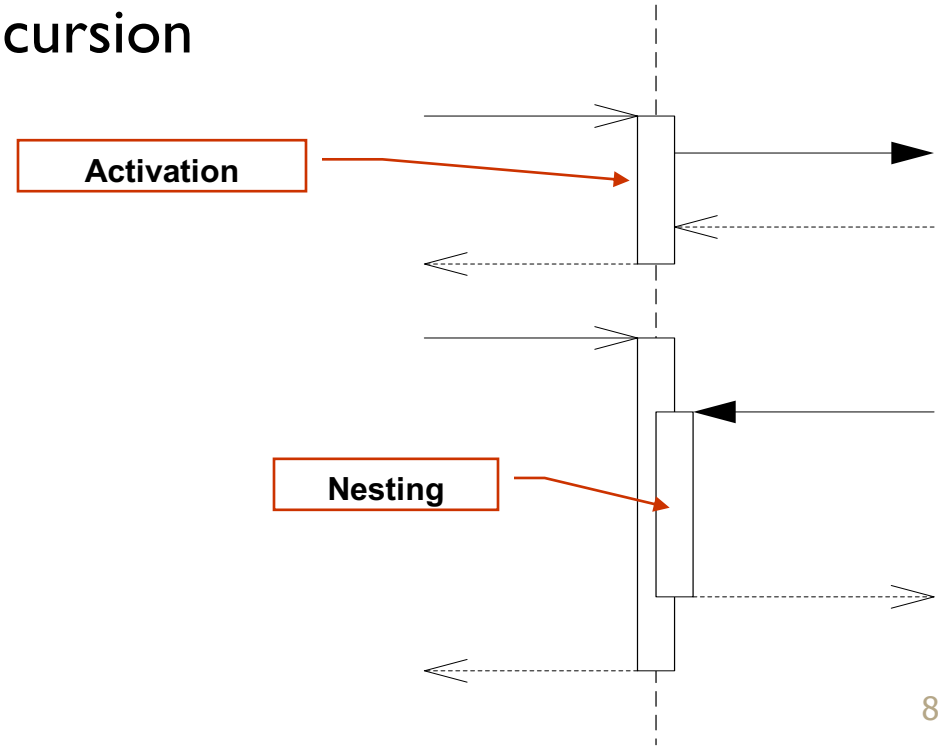


Hue University

Indicating method calls

- **activation**: thick box over object's life line; drawn when object's method is on the stack
  - either that object is running its code, or it is on the stack waiting for another object's method to finish
  - nest to indicate recursion

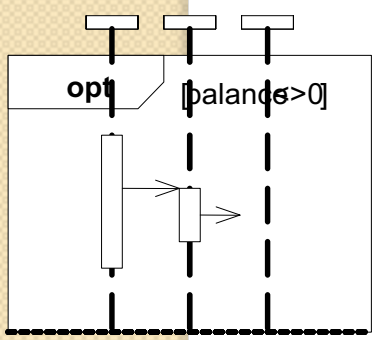


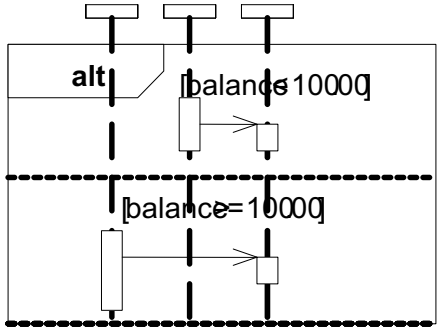


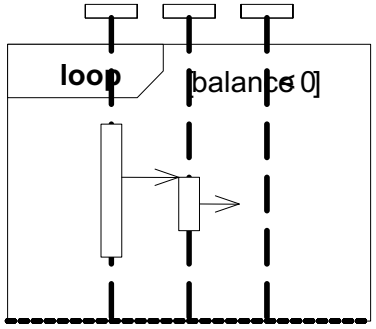
Hue University

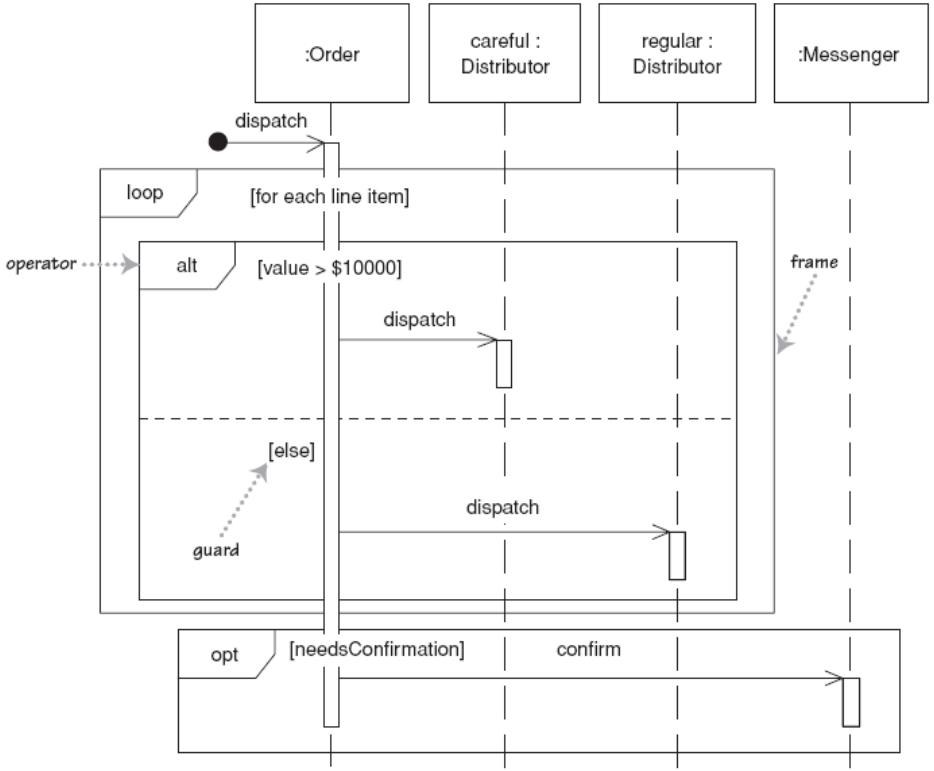
Indicating selection and loops

- **Fragment (frame)**: box around part of a sequence diagram to indicate selection or loop
  - if -> (opt) [condition]
  - if/else -> (alt) [condition], separated by horiz. dashed line
  - loop -> (loop) [condition or items to loop over]

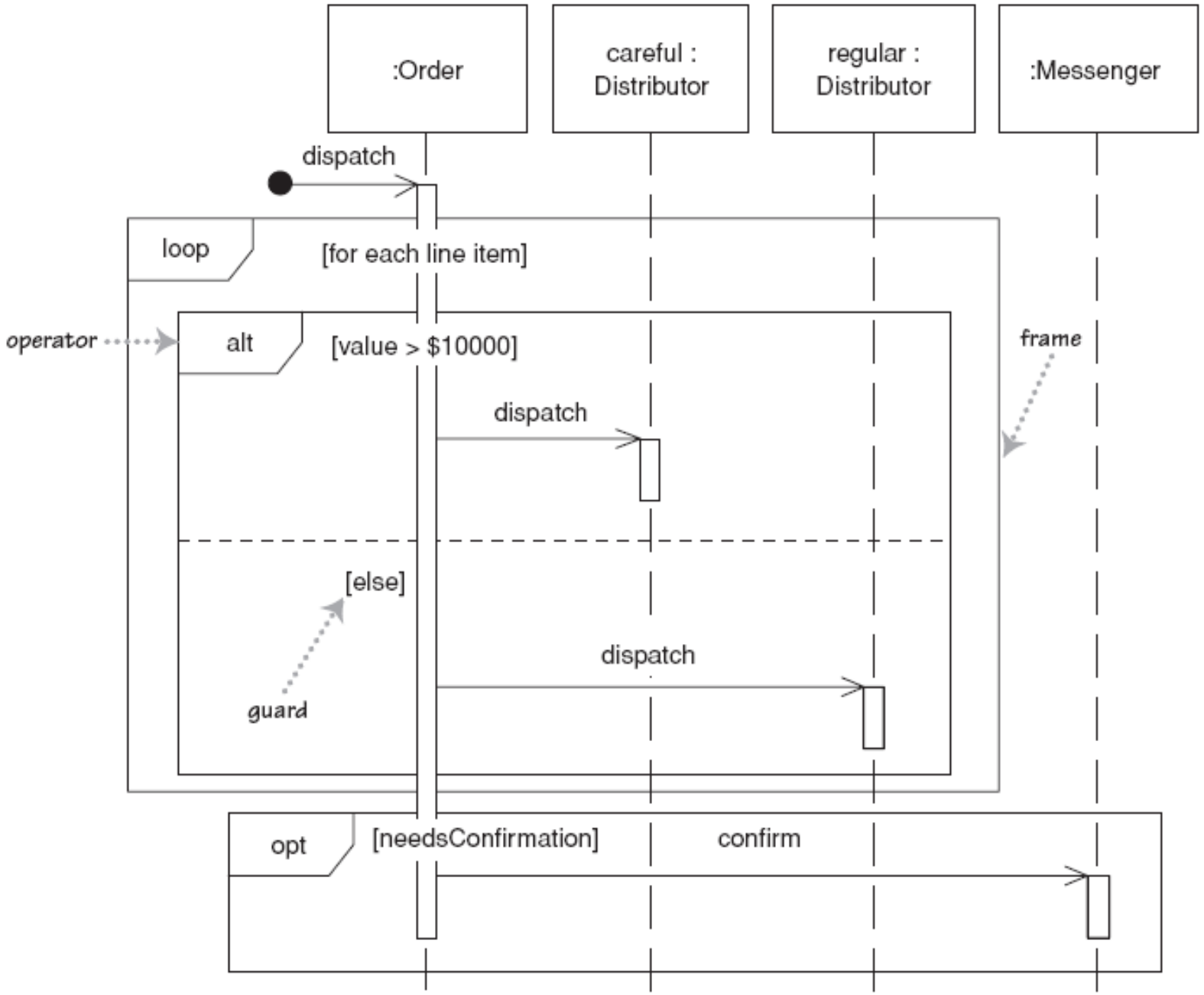








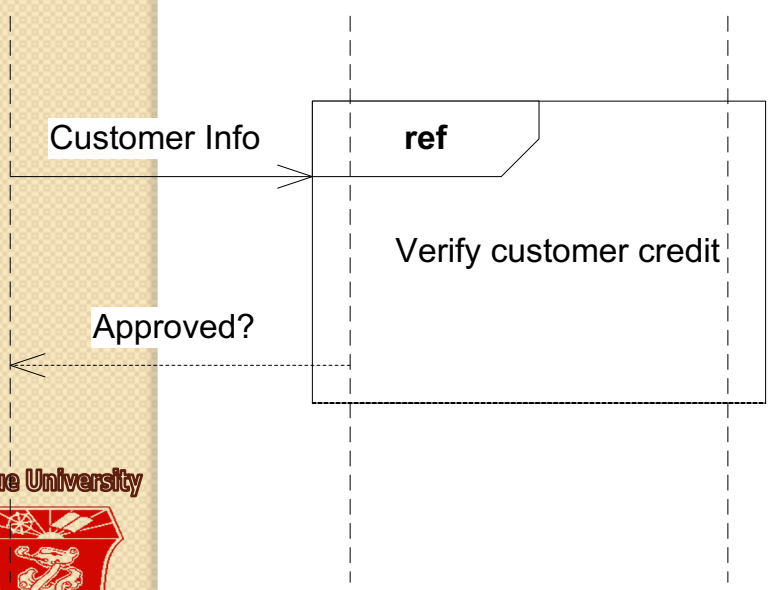
Hue University

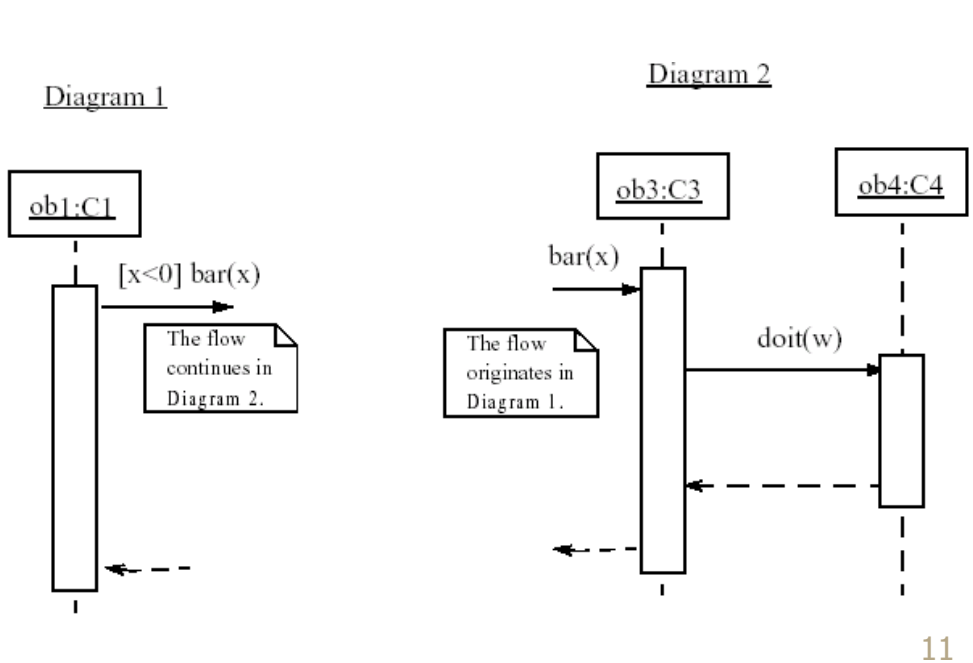


Hue University

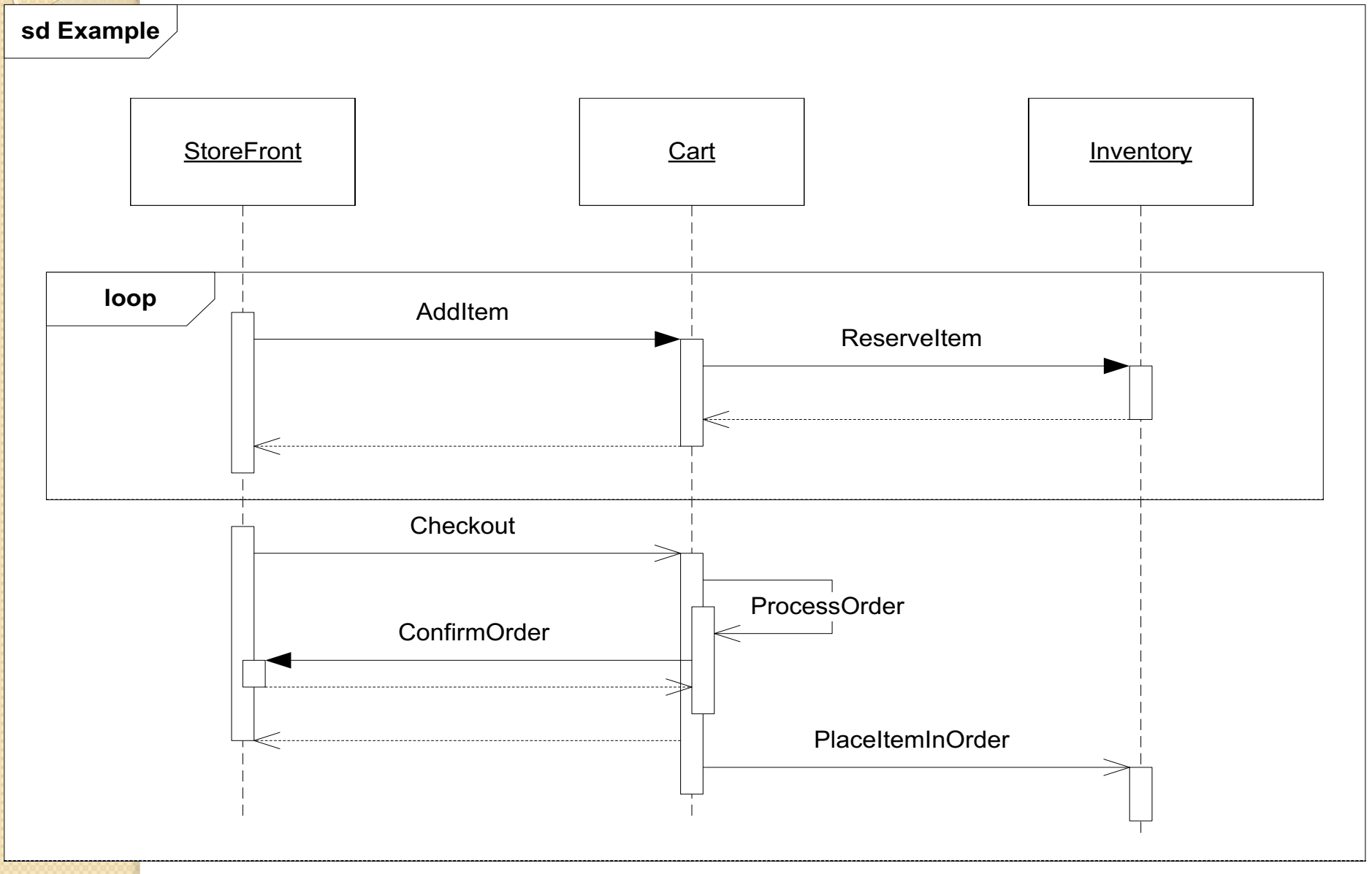
linking sequence diagrams

- if one sequence diagram is too large or refers to another diagram, indicate it with either:
  - an unfinished arrow and comment
  - a "ref" frame that names the other diagram
  - when would this occur in our system?



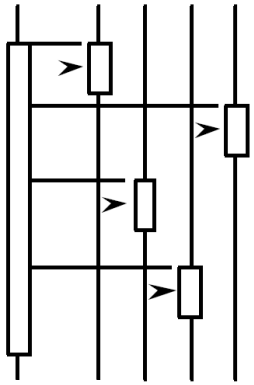


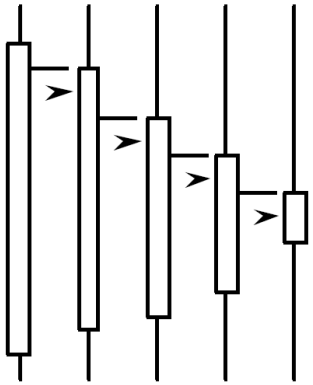
Example sequence diagram

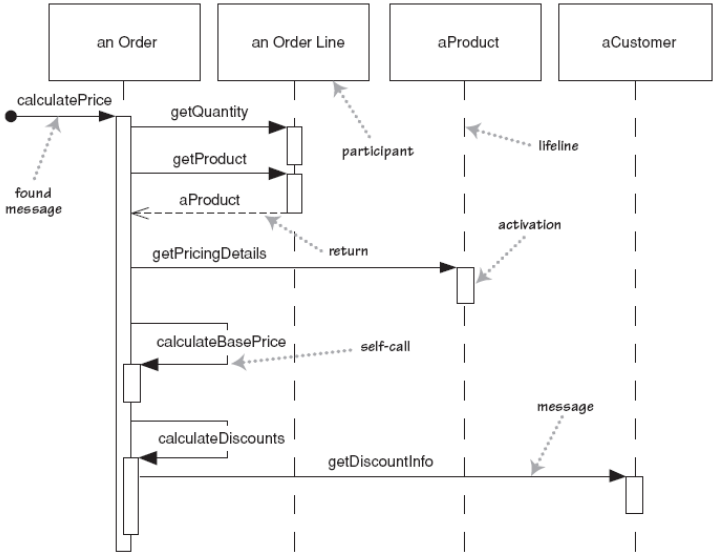


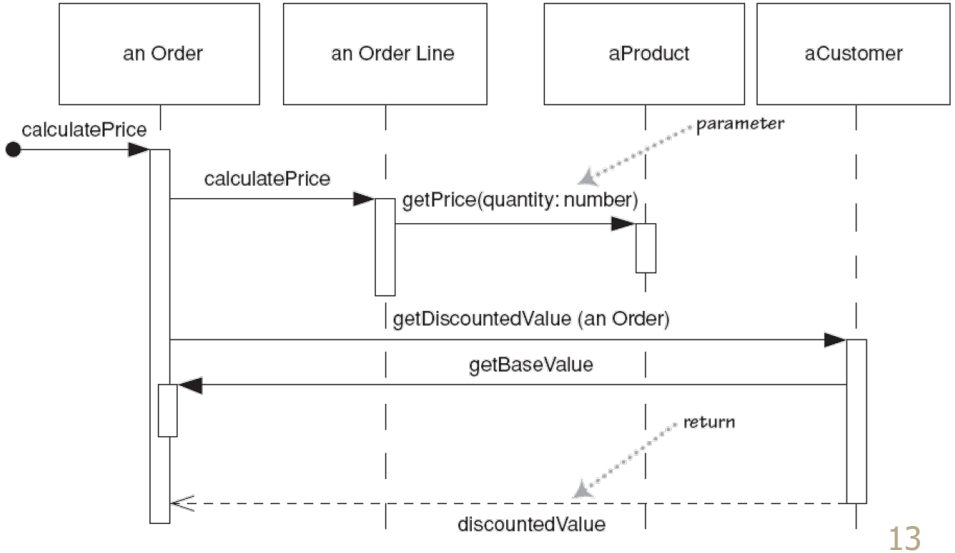
(De)centralized system control

- What can you say about the control flow of each of the following systems?
  - centralized?
  - distributed?









Why not just code it?

- Sequence diagrams can be somewhat close to the code level. So why not just code up that algorithm rather than drawing it as a sequence diagram?
  - a good sequence diagram is still a bit above the level of the real code (not EVERY line of code is drawn on diagram)
  - sequence diagrams are language-agnostic (can be implemented in many different languages
  - non-coders can do sequence diagrams
  - easier to do sequence diagrams as a team
  - can see many objects/classes at a time on same page (visual bandwidth)