# Sequence Diagram Tutorial

From:

*UML Distilled, Third Edition*, Chapter 4
M. Fowler

# Use Cases and Scenarios

- A **use case** is a collection of interactions between external actors and a system
- In UML, a use case is “the specification of a sequence of actions, including variants, that a system (or entity) can perform, interacting with actors of the system.”
- Typically each **use case** includes a primary **scenario** ( or main course of events) and zero or more secondary **scenarios** that are alternative courses of events to the primary **scenario**.
- In RUP (Rational Unified Process), user requirements are captured as **use cases** that are refined into **scenarios**.
- **Then**: A **scenario** is one path or flow through a **use case** that describes a sequence of events that occurs during one particular execution of a system.

# UML Sequence Diagrams

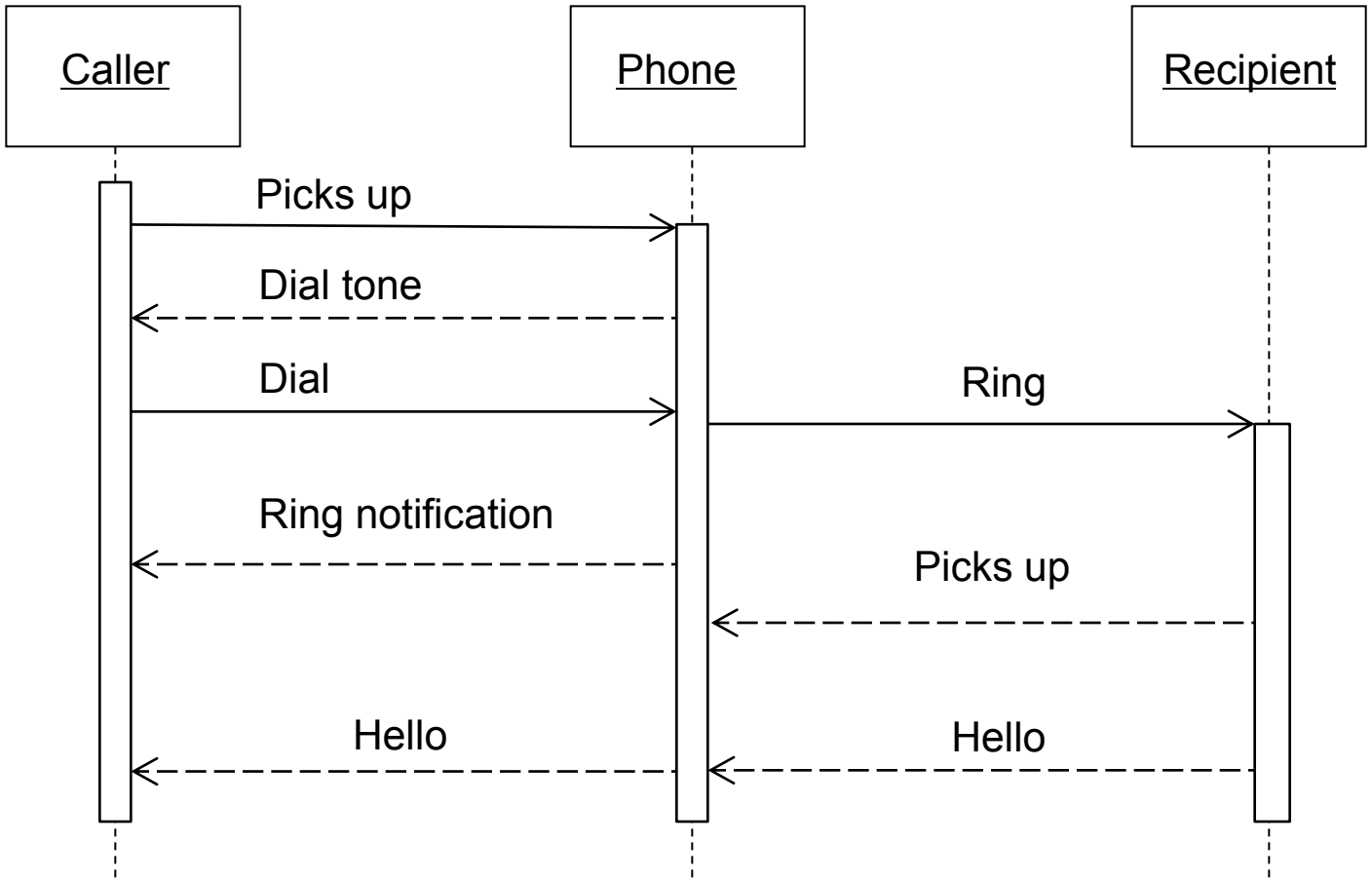
- Describe the flow of messages, events, actions between objects
- Show concurrent processes and activations
- Show time sequences that are not easily depicted in other diagrams
- Typically used during analysis and design to document and understand the logical flow of your system

**Emphasis on time ordering!**

# Sequence Diagram Key Parts

- **participant**: object or entity that acts in the diagram
  - 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)

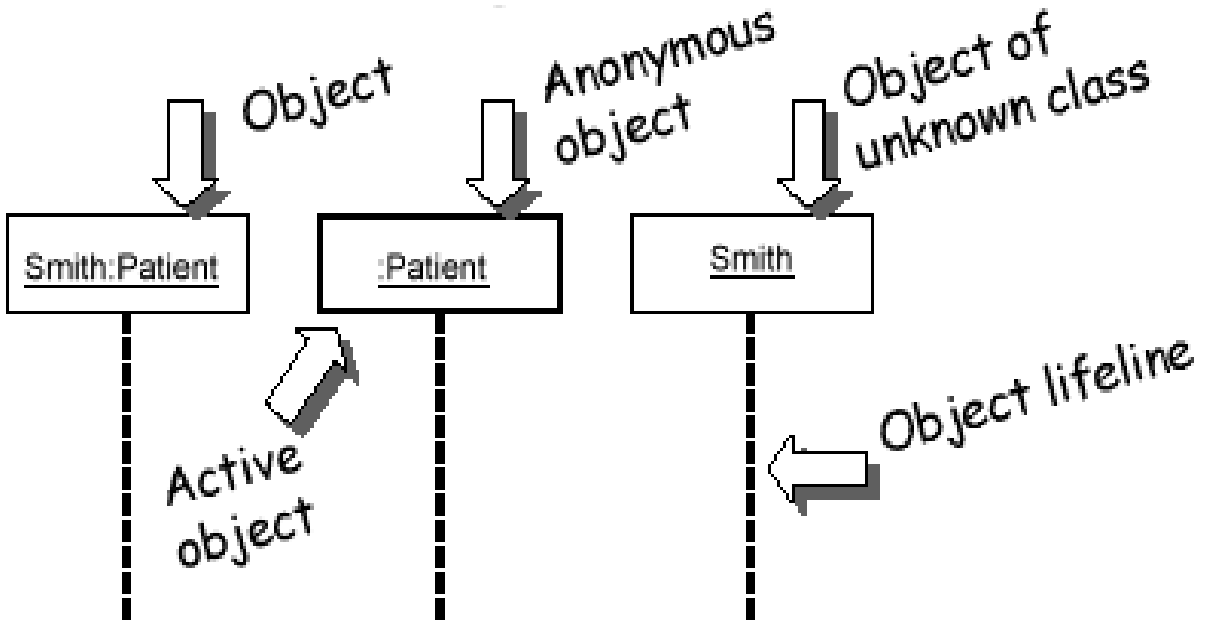
# Sequence Diagram (make a phone call)



# Representing Objects

Squares with object type, optionally preceded by "*name* :"

- write object's name if it clarifies the diagram
- object's "life line" represented by dashed vert. line

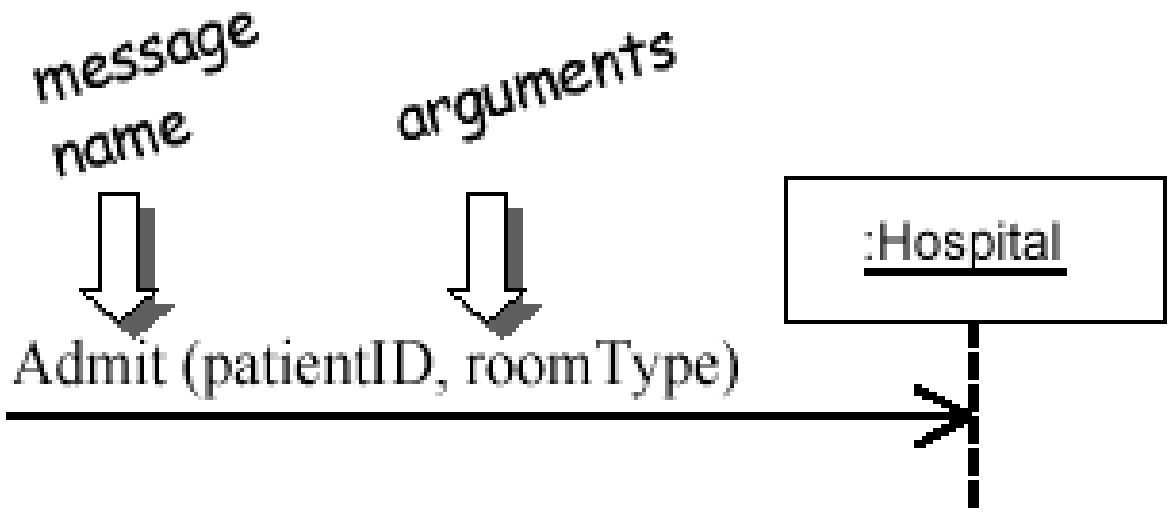



**Name syntax:** <objectname>:<classname>

# Messages Between Objects

**messages** (method calls) indicated by arrow to other object

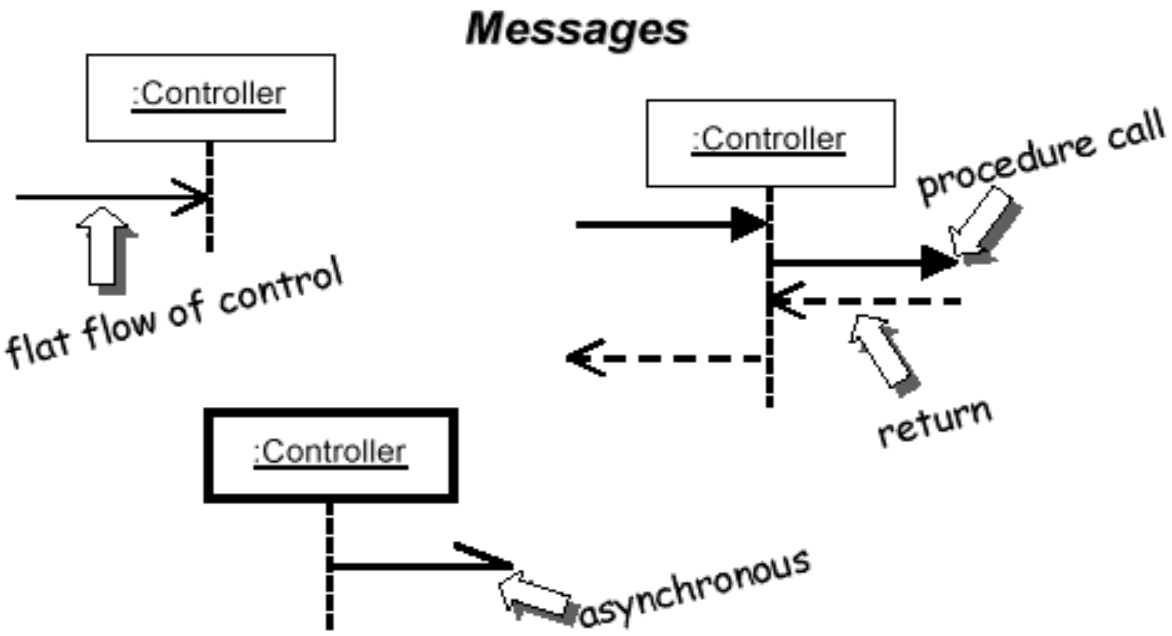
– write message name and arguments above arrow



# Messages, continued

**messages** (method calls) indicated by arrow to other object

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



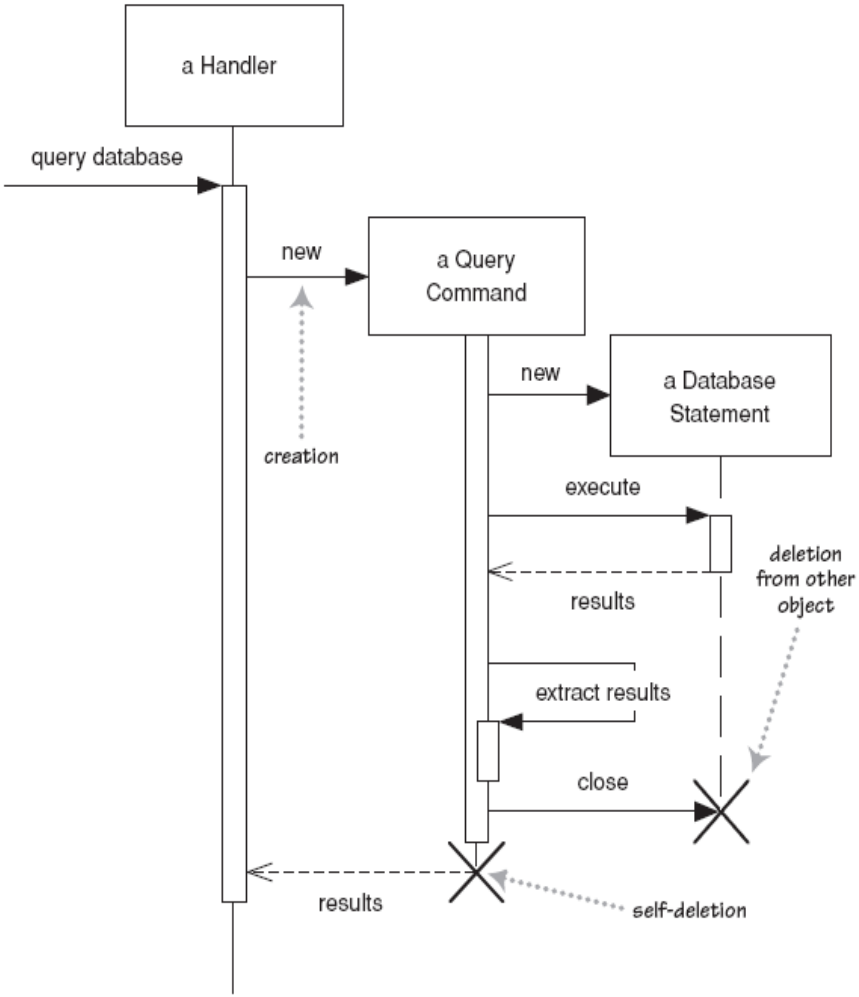
# Lifetime of objects

***creation***: arrow with 'new' written above it

- notice that an object created after the start of the scenario appears lower than the others

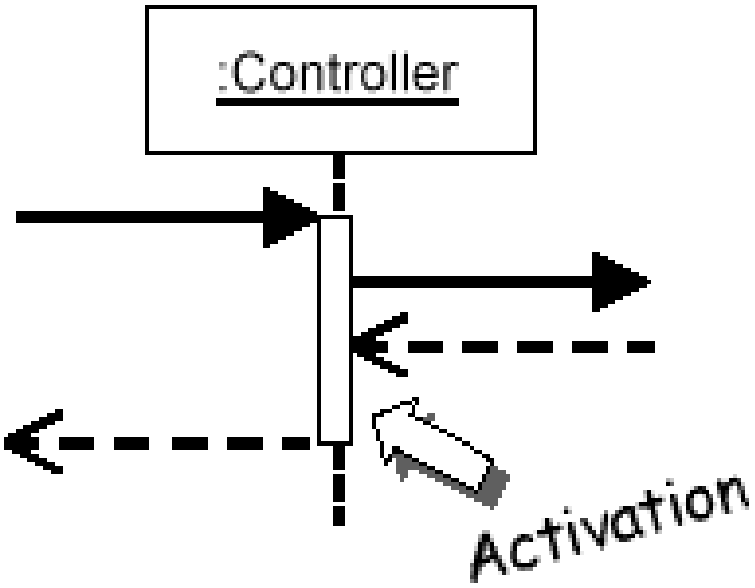
***deletion***: an X at bottom of object's lifeline

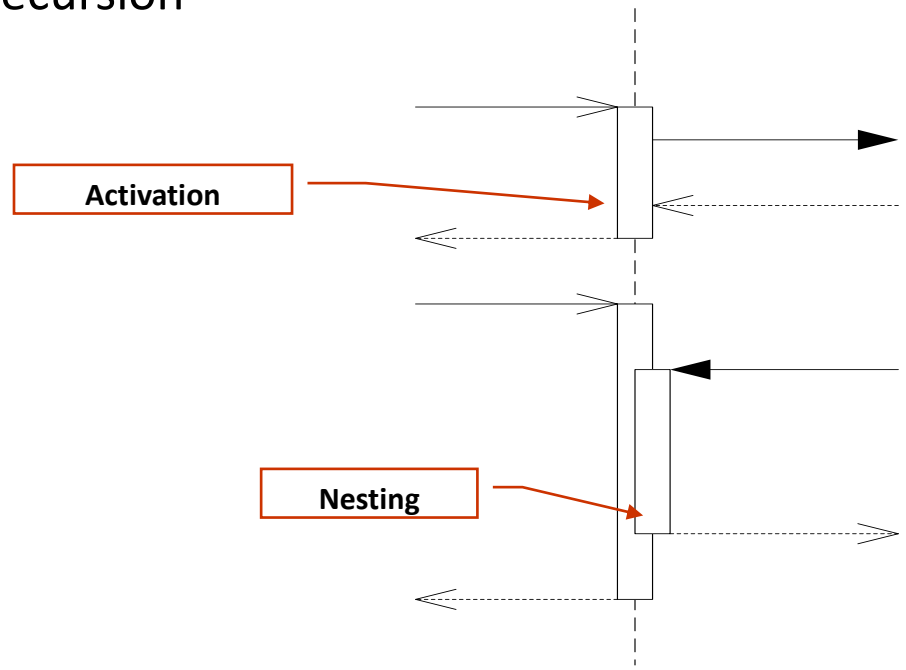
- Java doesn't explicitly delete objects; they fall out of scope and are garbage-collected



# 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 activations to indicate recursion

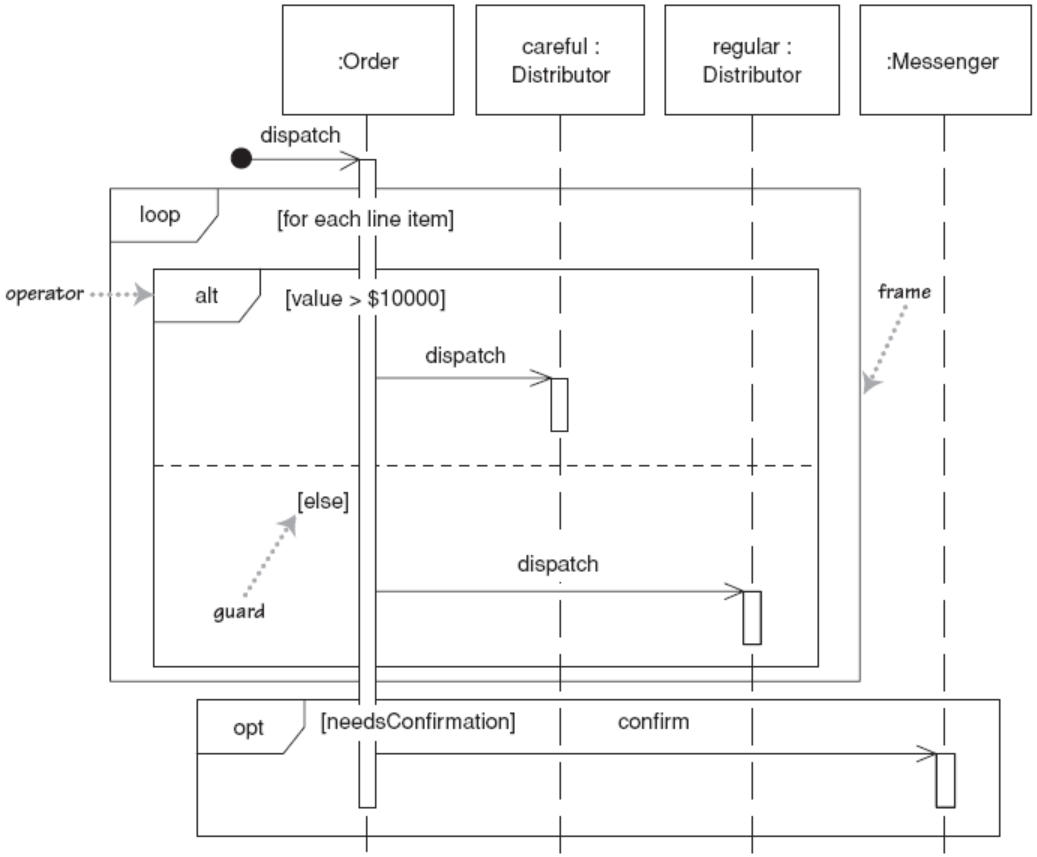


# Selection and loops

**frame**: box around part of diagram to indicate `if` or loop

- `if` -> (opt) [condition]
- `if/else` -> (alt) [condition], separated by horizontal dashed line
- loop -> (loop) [condition or items to loop over]



# Sequence diagram from use case scenario