CptS 322- Programming Language Design

UML Sequence Diagrams

Instructor: Sakire Arslan Ay

Fall 2021

WASHINGTON STATE UNIVERSITY
World Class. Face to Face.

Outline

- Overview of UML sequence diagrams
- Syntax and semantics
- Examples

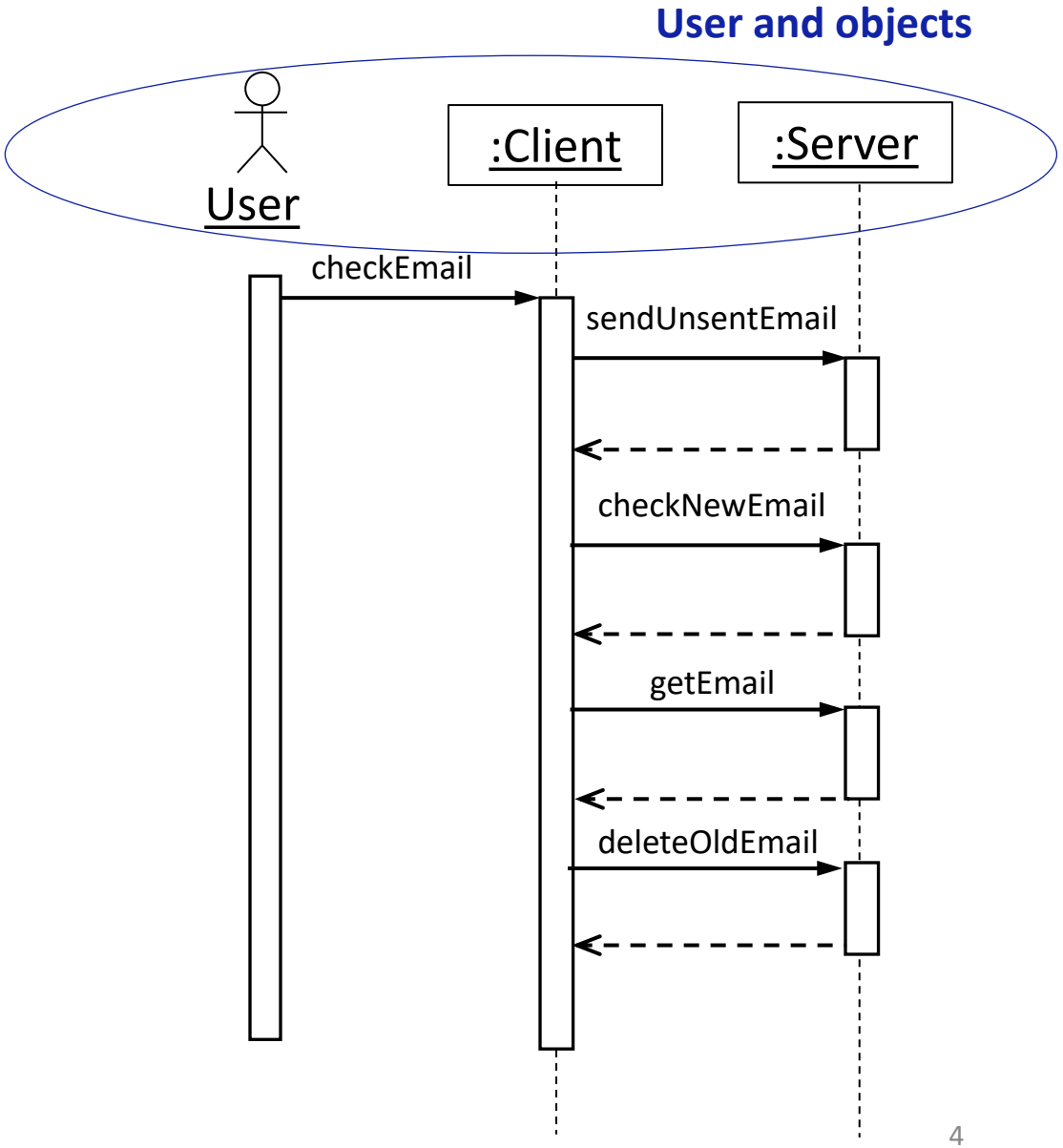
What is a UML sequence diagram?

- Use cases → sequence diagrams

- Sequence diagram:
  - an "interaction diagram" that models a single scenario executing in a system
  - Shows what messages are sent and when
  - 2nd most used UML diagram (after class diagram)

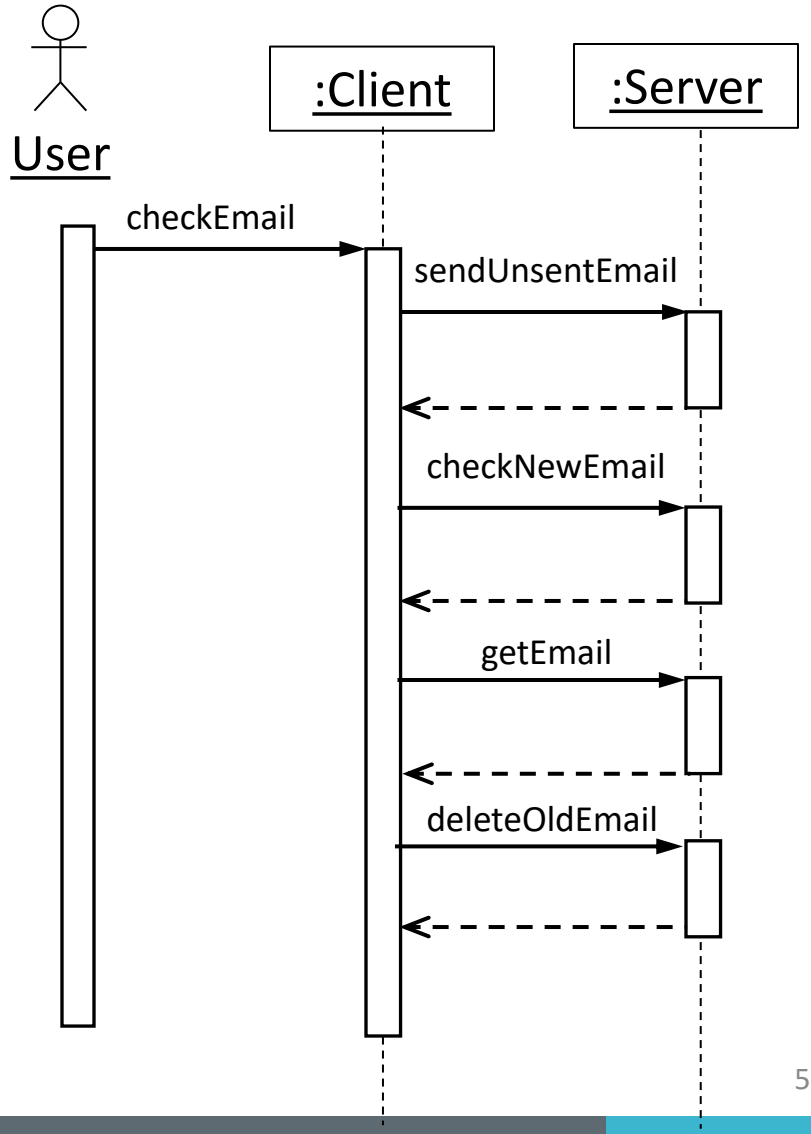
Key Parts of a Sequence Diagram

- **Participant**: User and the classes
  - Sequence diagram starts with the request of the user
- **Message**:
  - Communication between objects
- Axes in a sequence diagram
  - **Horizontal**: which participant is acting
  - **Vertical**: time (forward in time)

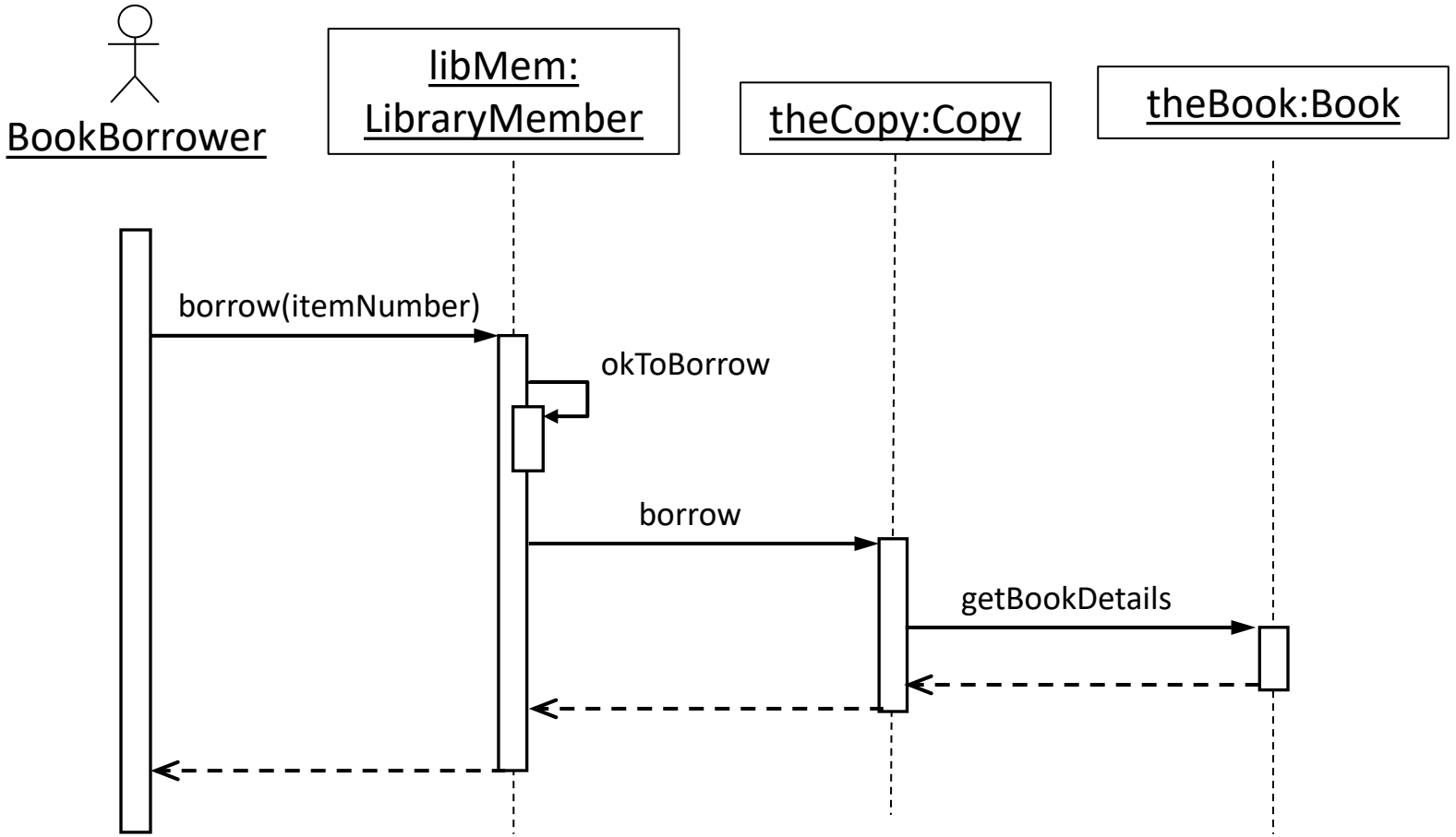


Sequence Diagram From a Use Case: Check Email

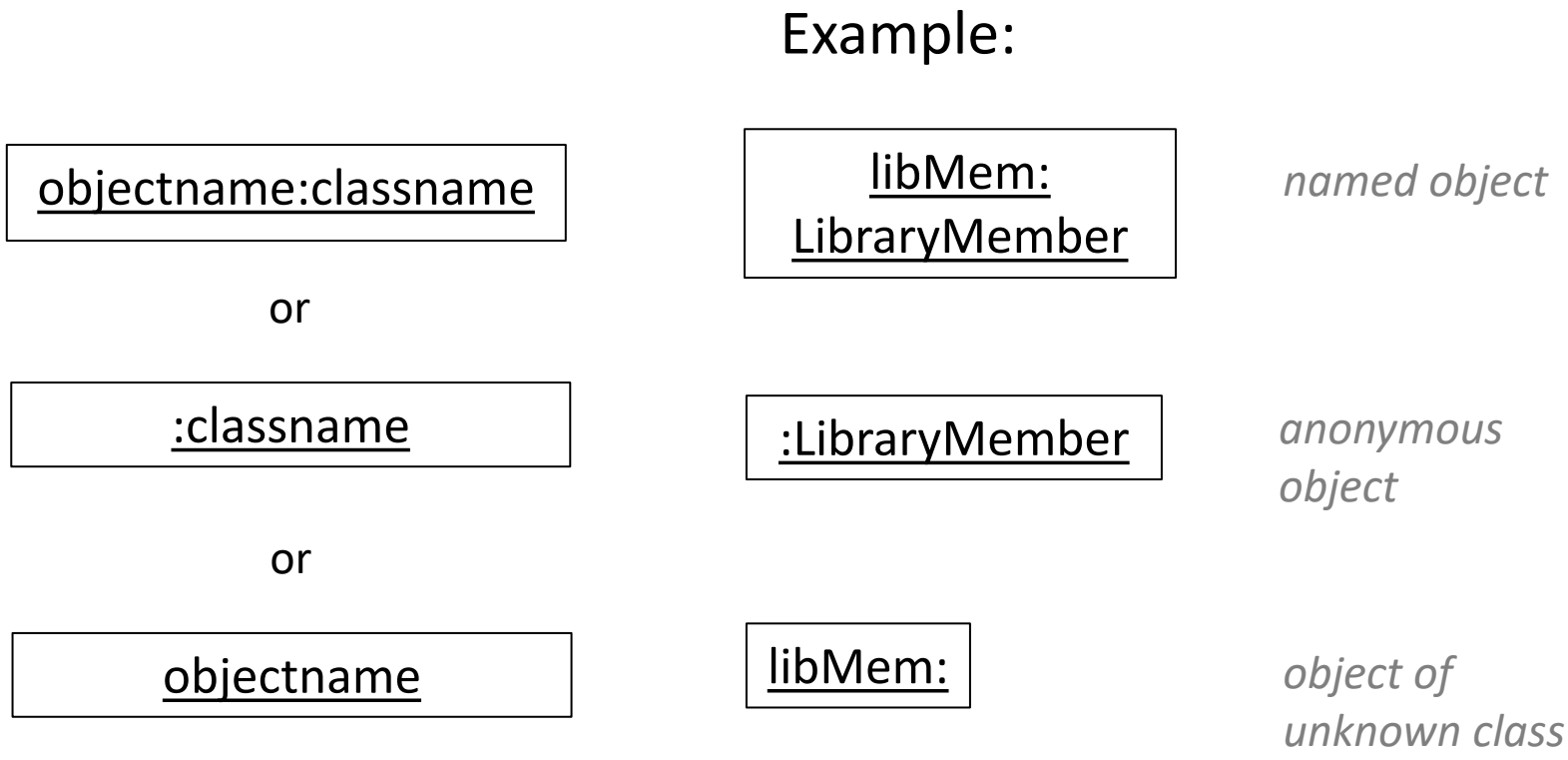
1. The user presses the “check email” button.
2. The client first sends all unsent email to the server.
3. After receiving an acknowledgement, the client asks the server if there is any new email.
4. If so, it downloads the new email.
5. Next, it deletes old thrashed email from the server.



Sequence Diagram From a Use Case: Borrow Copy of a Book



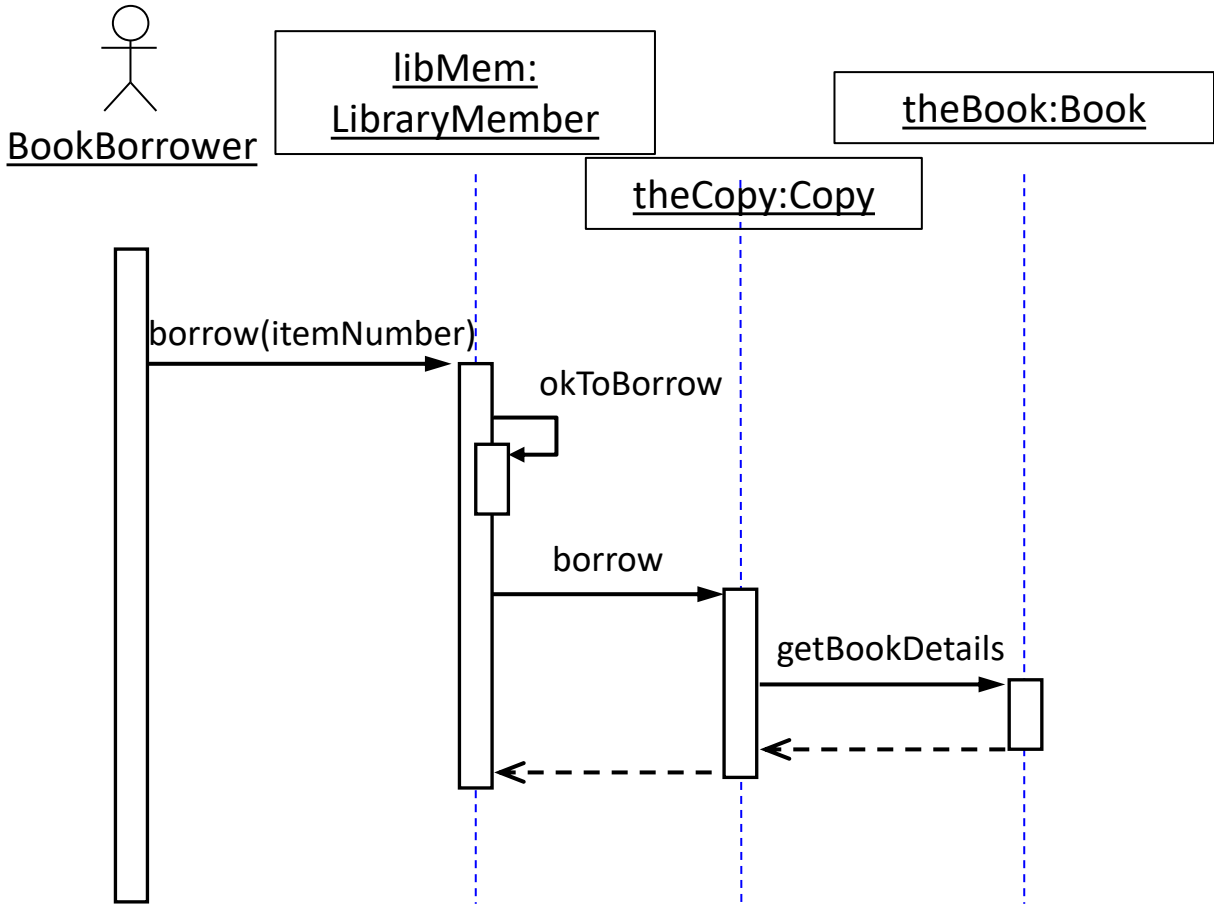
UML Notation: Representing Objects



- **An object**: a box with an underlined label that specifies the object type, and optionally the object name.
  - Write the object's name if it clarifies the diagram.

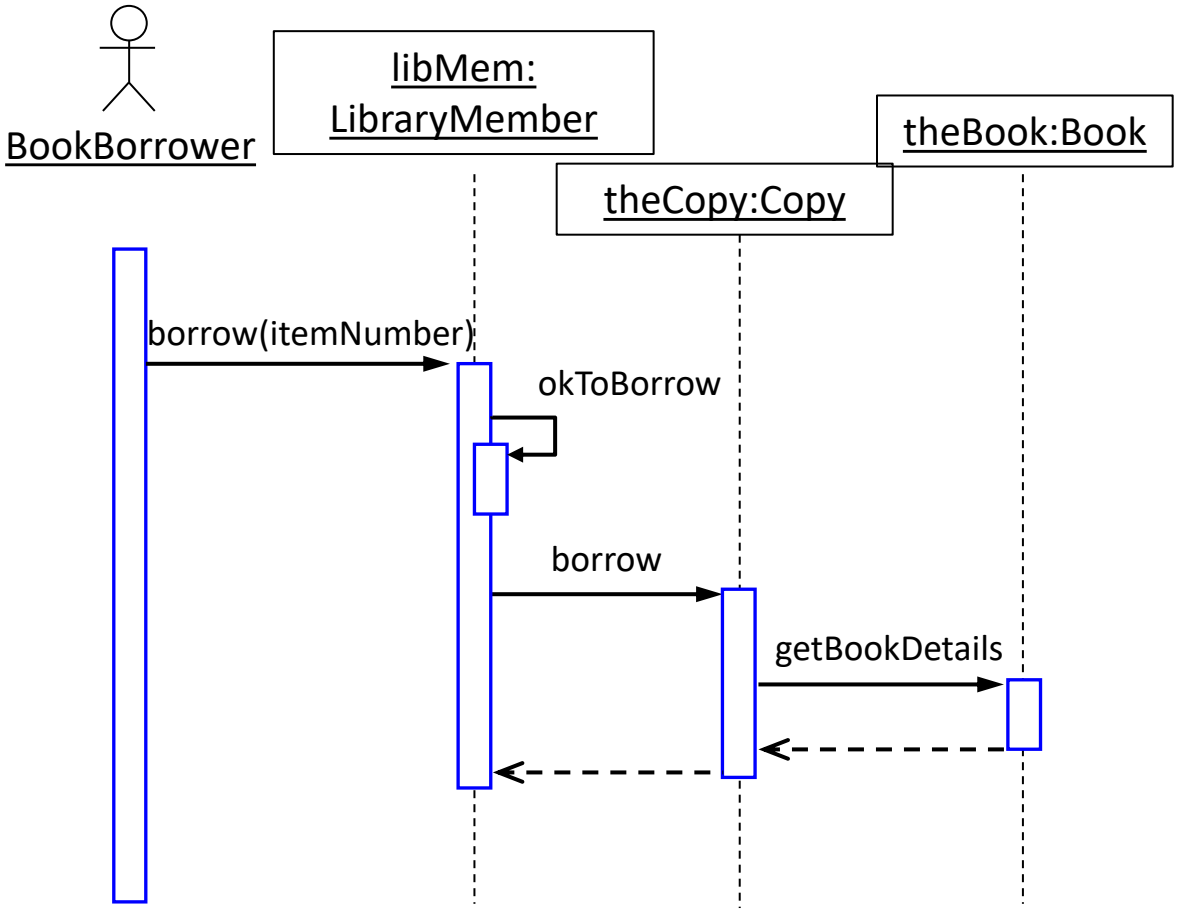
UML Notation: Representing Objects

- An object's **"life line"** is represented by a dashed vertical line.
  - Represents the life span of the object during the scenario being modeled.



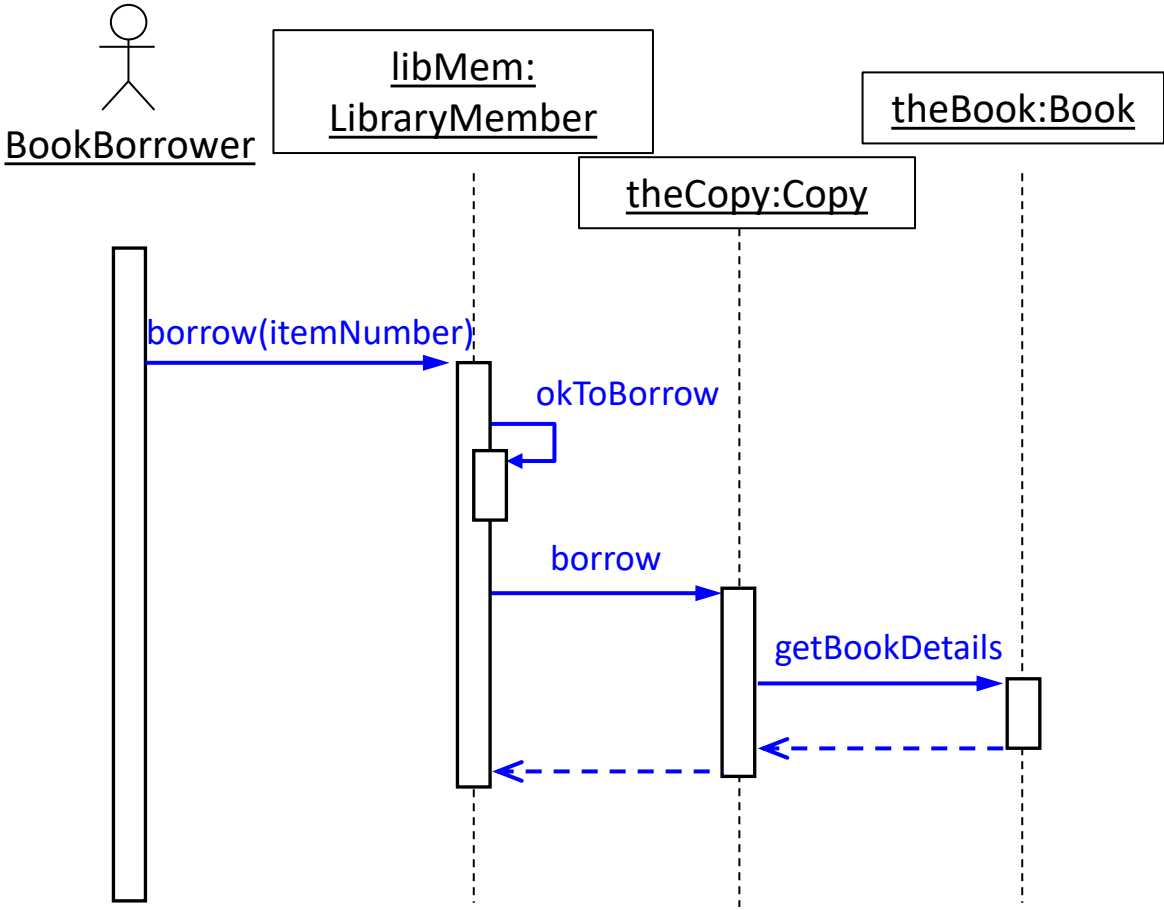
UML Notation: Indicating Method Execution

- Activation: thick box over object's life line, drawn when an 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 an object calling itself.



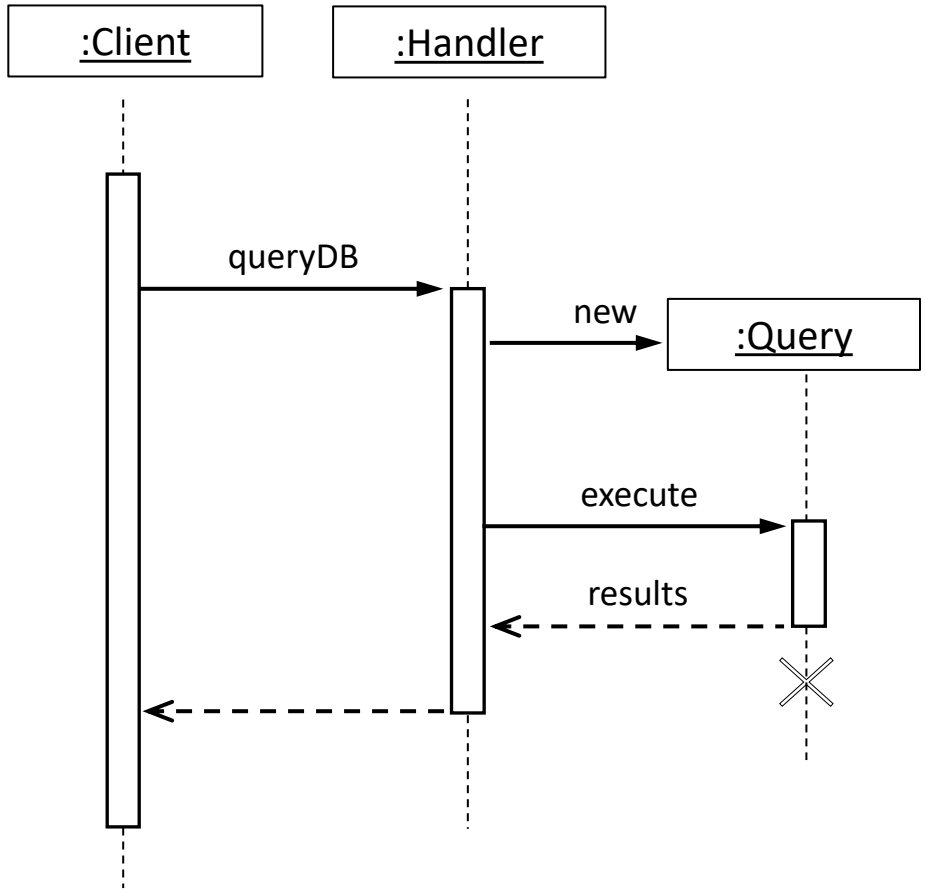
UML Notation: Representing Messages Between Objects

- A message (method call): horizontal arrow to the receiving object.
  - Write message name and arguments above the arrow.
- Type of arrow indicates types of messages:
  - Synchronous message: solid arrow with a solid head.
  - Asynchronous message: solid arrow with a stick head.
  - Return message: dashed arrow with stick head.

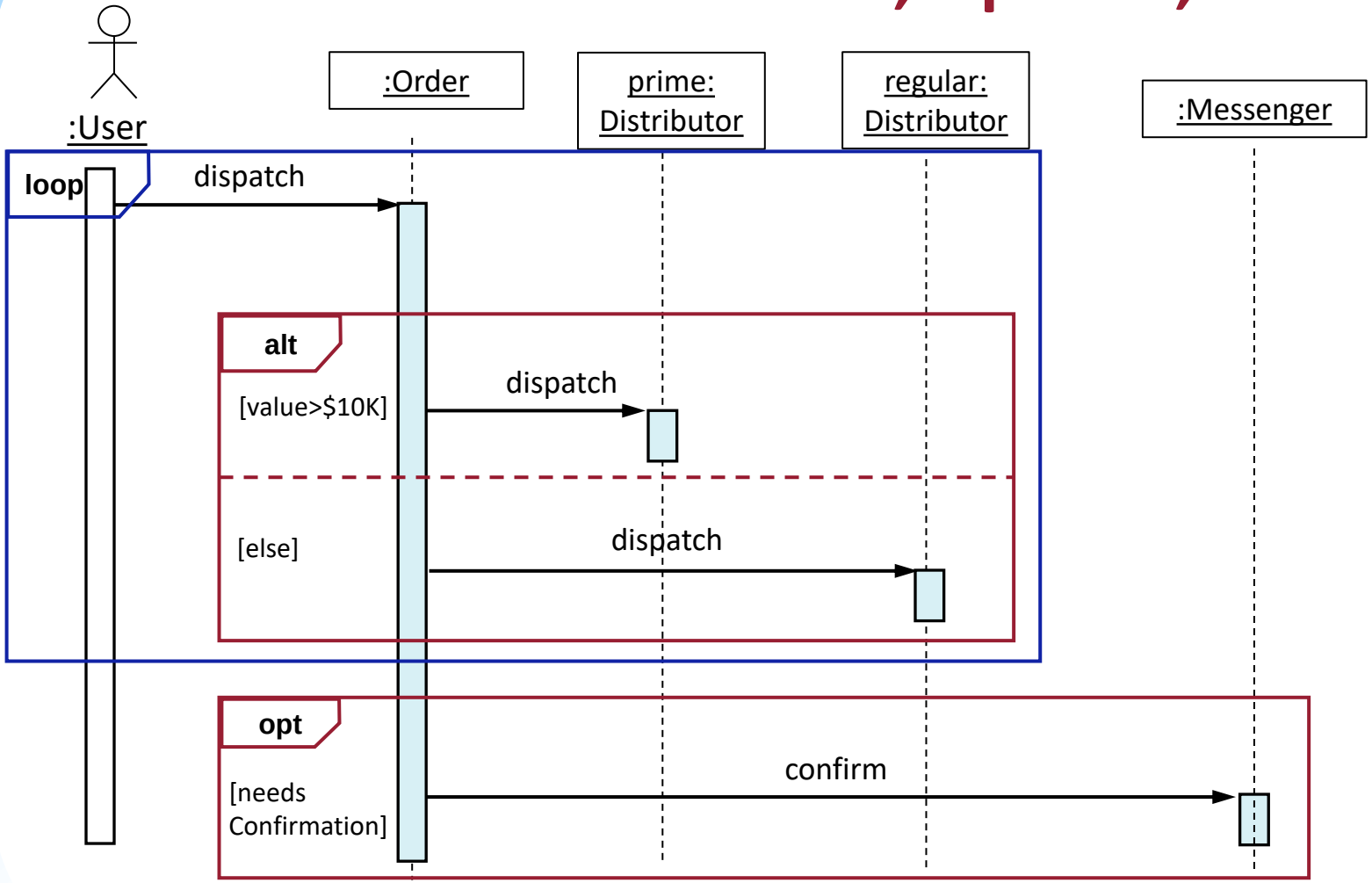


UML Notation: Lifetime of Objects

- Object creation: an arrow with new written above it
  - An object created after the start of the scenario appears lower than the others.
- Object deletion: X at the bottom of object's lifeline
  - In languages with garbage collection (e.g. Java, C#) objects are not explicitly deleted; they fall out of scope and are garbage collected.

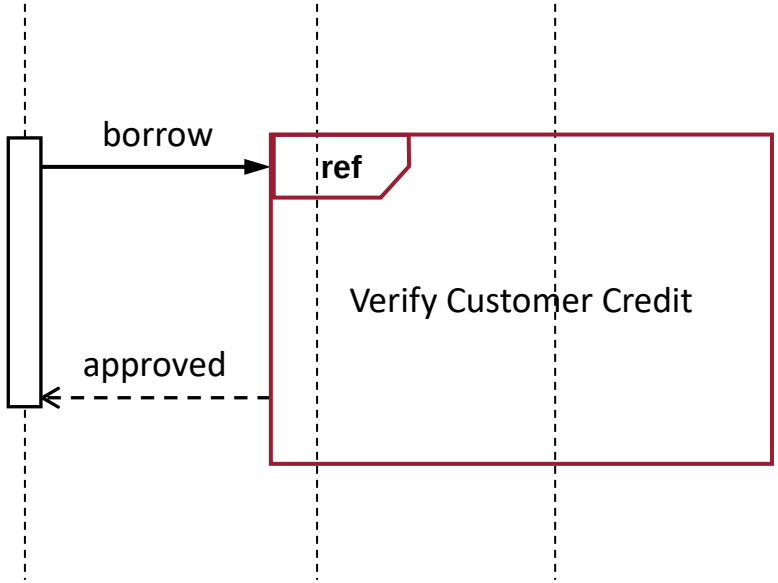


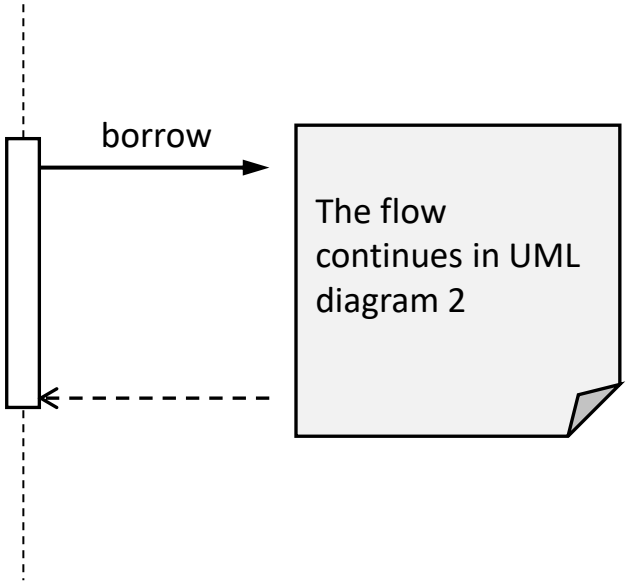
UML Notation: Alternatives, Options, and Loops



- Frame: a box around part of a sequence 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]

Linking Sequence Diagrams

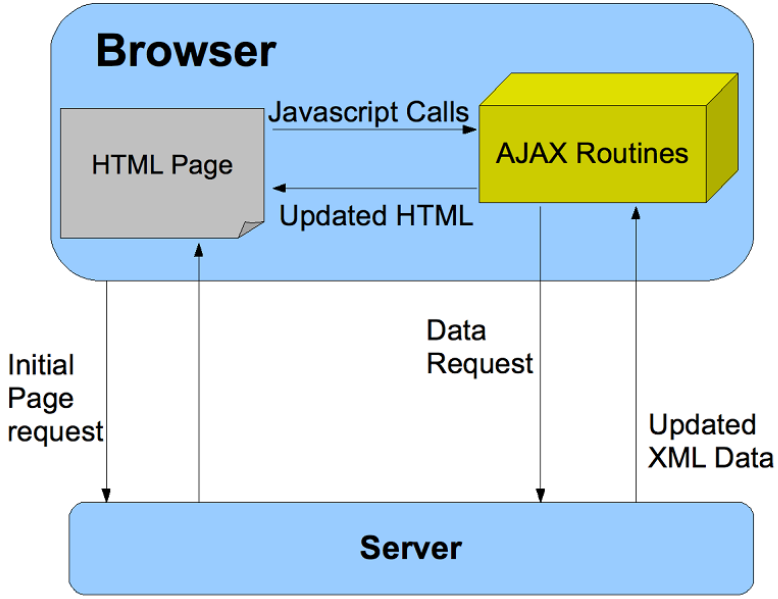


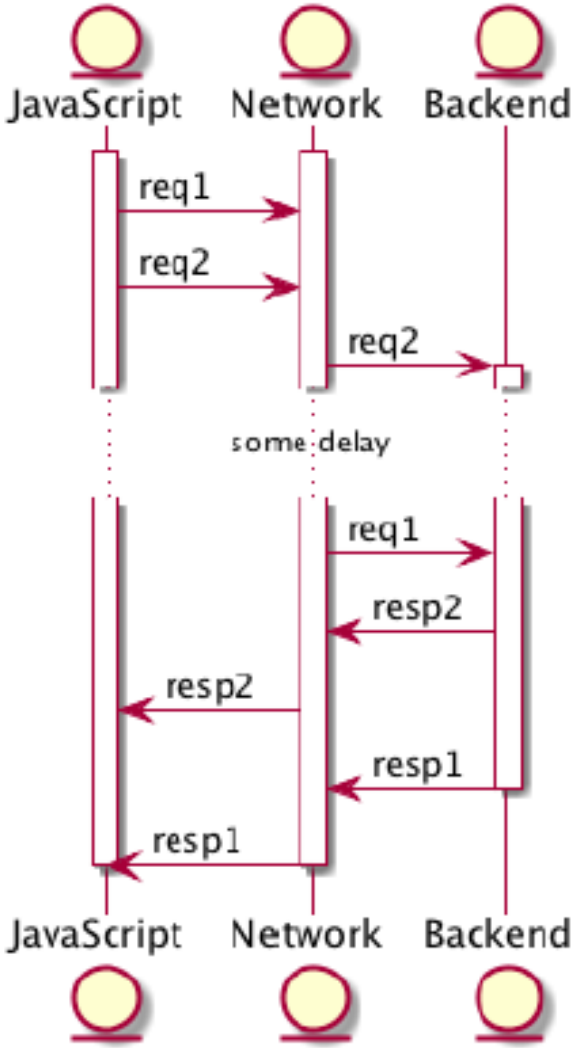


- If one sequence diagram is too large or refers to another diagram:
  - An unfinished arrow and comment.
  - A ref frame that names the other diagram.

Sequence Diagram Example

- Sequence of steps in two AJAX requests





Why use sequence diagrams? Why not just code it?

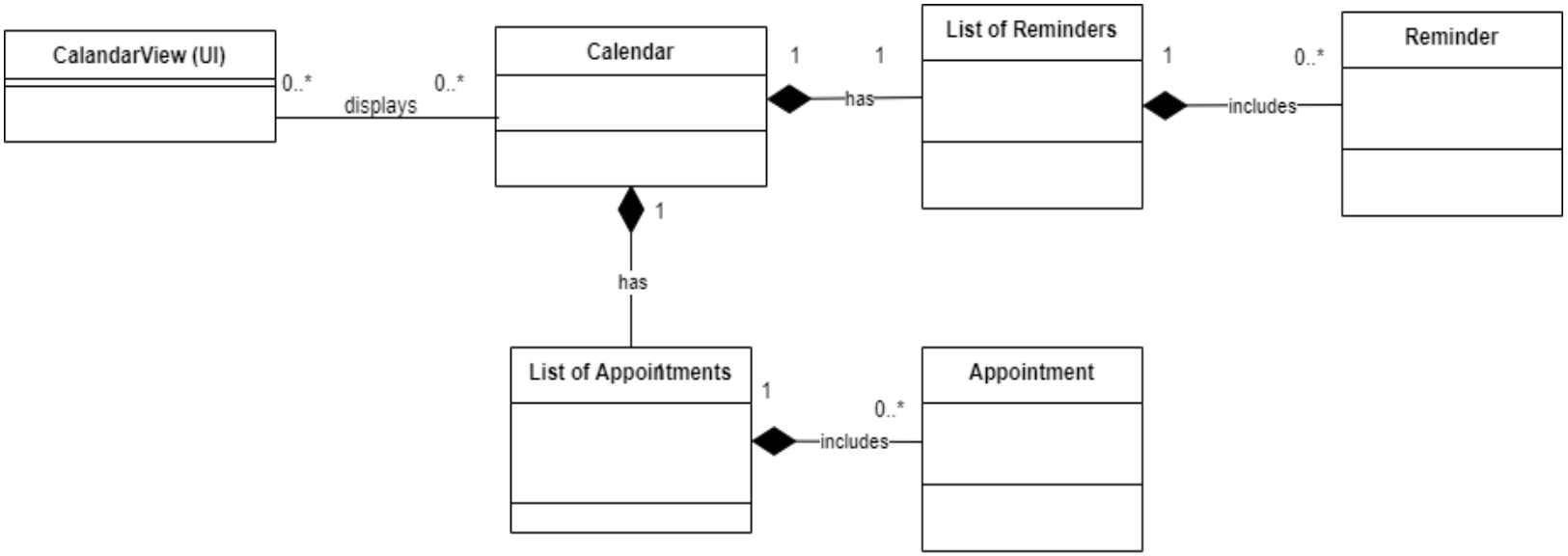
- Sequence diagrams can be somewhat close to the code level. So why not just code up that algorithm rather than drawing it as a sequence diagram?
  - a good sequence diagram is still a bit above the level of the real code (not EVERY line of code is drawn on diagram)
  - sequence diagrams are language-agnostic (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)

Sequence Diagram Exercise

Add new appointment to calendar:

- The scenario begins when the user chooses to add a new appointment in the UI. The UI notices which part of the calendar is active and pops up an Add Appointment window for that date and time.
- The user enters the necessary information about the appointment's name, location, start and end times. The UI will prevent the user from entering an appointment that has invalid information, such as an empty name or negative duration. The calendar records the new appointment in the list of appointments. Any reminder selected by the user is added to the list of reminders.
- If the user already has an appointment at that time, the user is shown a warning message.

Class Diagram for Calendar Example



Class Diagram for Calendar Example

Summary

- A sequence diagram models a single scenario executing in the system.
- Key components include participants and messages.
- Sequence diagrams provide a high-level view of control flow patterns through the system.