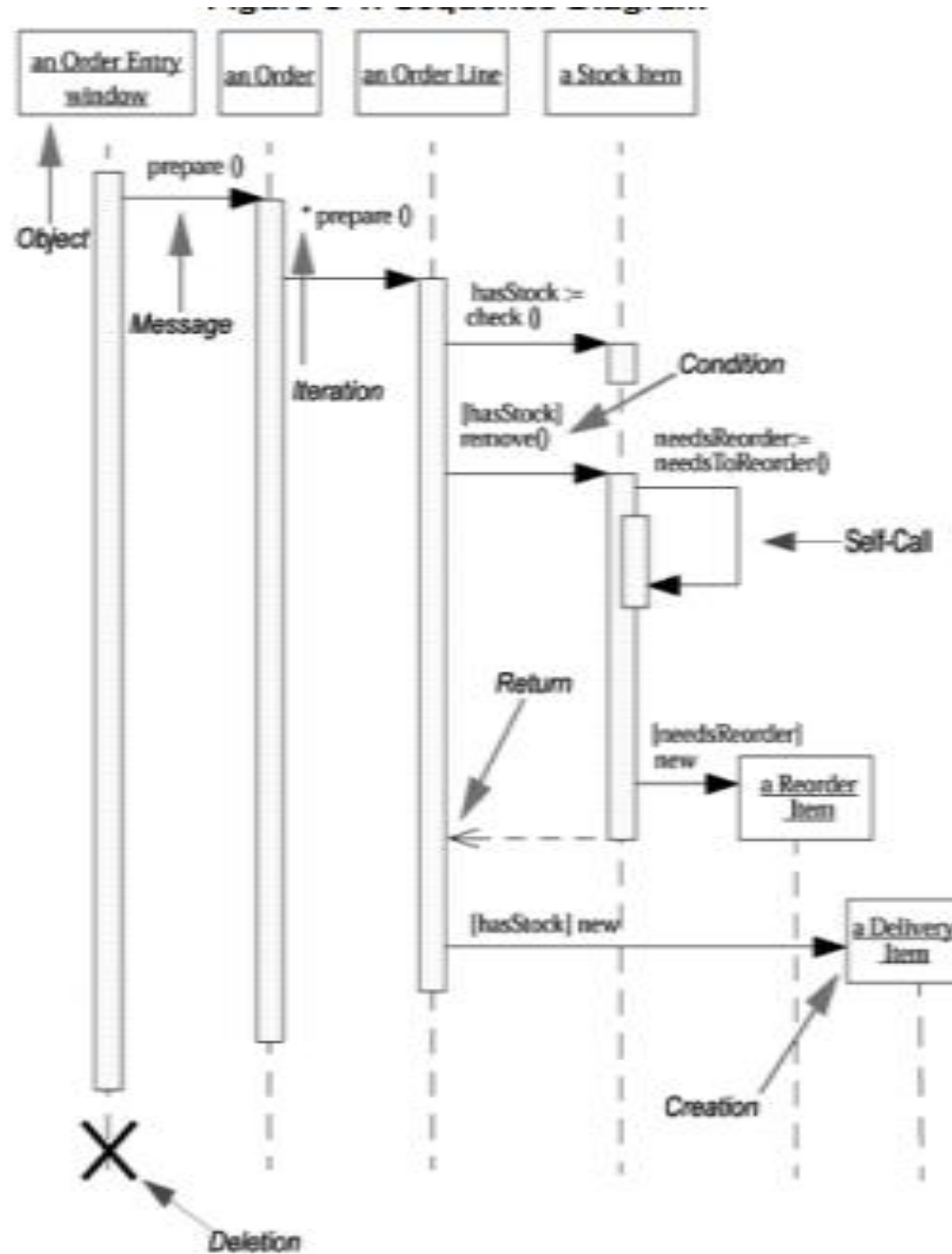


Interaction Diagrams

What is Interaction diagrams

- Describes how groups of objects collaborate in some behavior
- captures the behavior of a single use case
- Shows a number of example objects and the messages that are passed between these objects within the use case
- Order Processing Use Case
 - Order Entry window sends a "prepare" message to an Order
 - Order then sends "prepare" to each Order Line on the Order.
 - Each Order Line checks the given Stock Item
 - If this check returns "true," the Order Line removes the appropriate quantity of Stock Item from stock and creates a delivery item.
 - If the Stock Item has fallen below the reorder level, that Stock Item requests a reorder
- Two types of Interaction Diagram
 - Sequence Diagram
 - Collaboration Diagram

Sequence Diagram

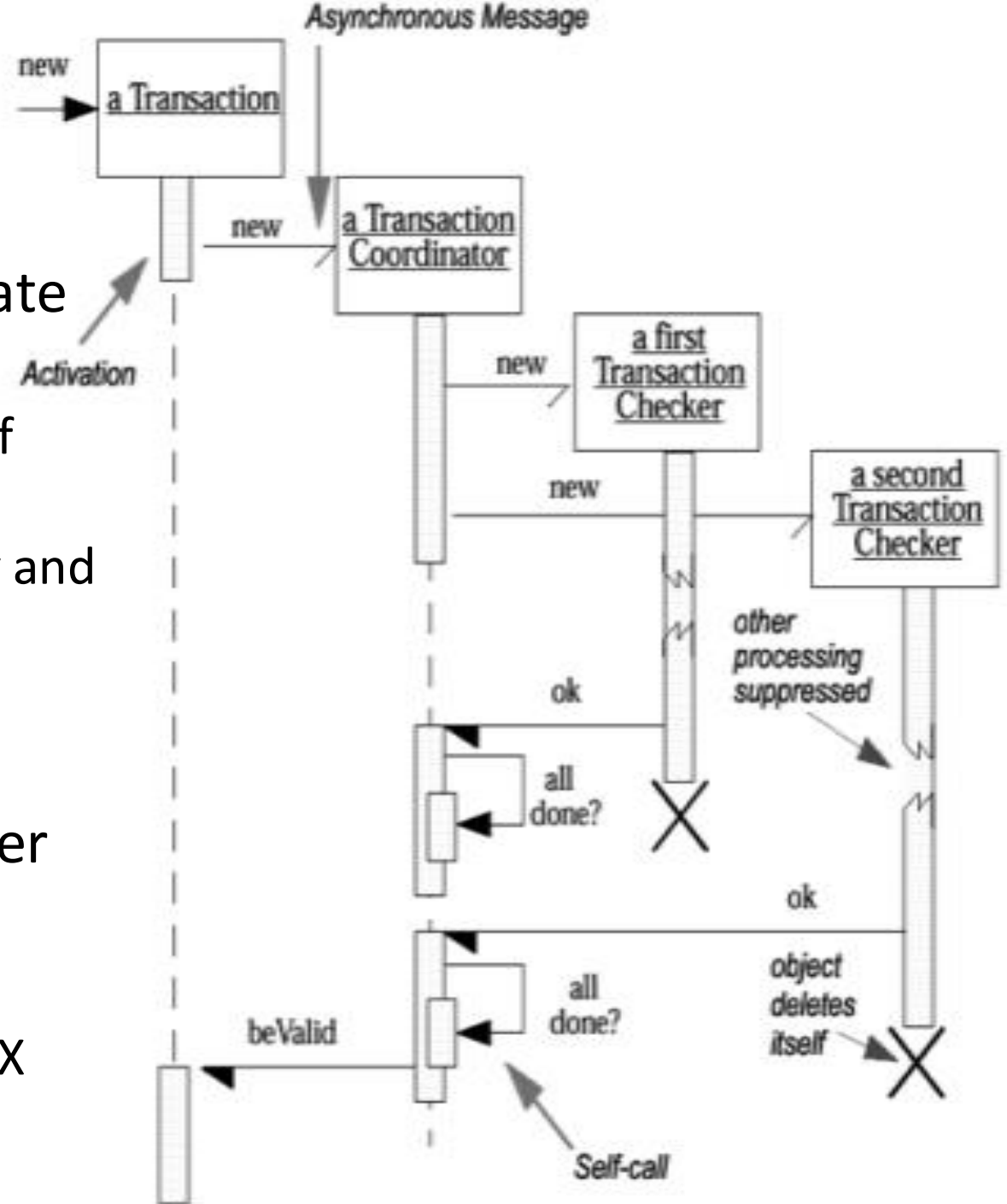


Sequence Diagram Syntax

- object is shown as a box at the top of a dashed vertical line
- The vertical line is called object's lifeline
- Each message is represented by an arrow between the lifelines of two objects
- Message Format: message name; optional arguments and some control information
 - The message is sent only if the condition is true, written in parentheses
 - iteration marker, which shows that a message is sent many times to multiple receiver objects indicated by *
- self-call, a message that an object sends to itself, by sending the message arrow back to the same lifeline.
- return indicates a return from a message
 - Returns differ from the regular messages in that the line is dashed
 - Not always shown
 - Shown only when return is required for clarity

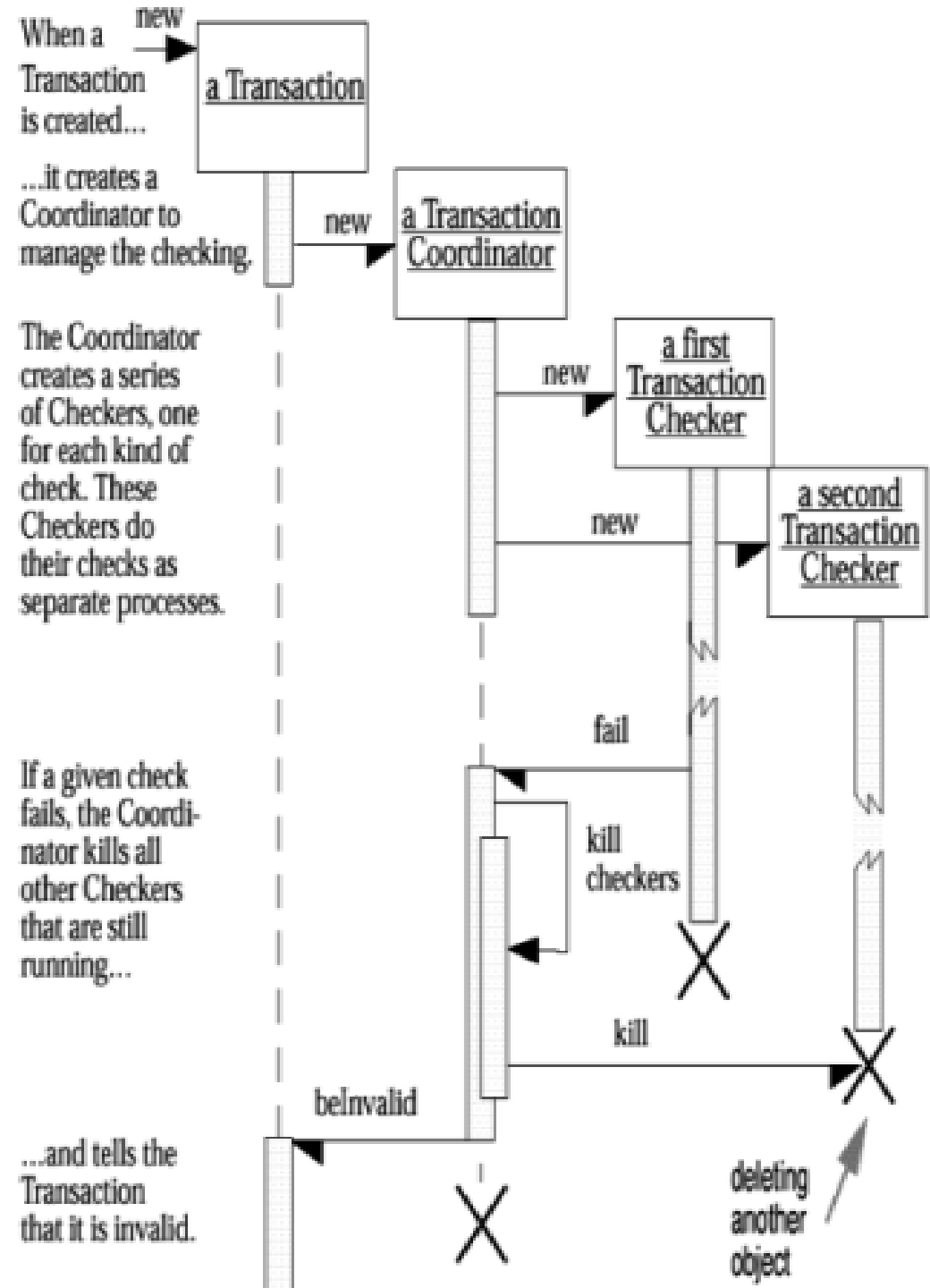
Concurrent Processes in the Sequence Diagram

- Transaction Coordinator to coordinate the checking of the Transaction
 - creates a number (in this case, two) of Transaction Checker objects
 - each checker is called asynchronously and proceeds in parallel
 - The half-arrowheads indicate an asynchronous message.
- The coordinator looks to see whether all the checkers called back
 - Through self call
 - Object deletion is shown with a large X



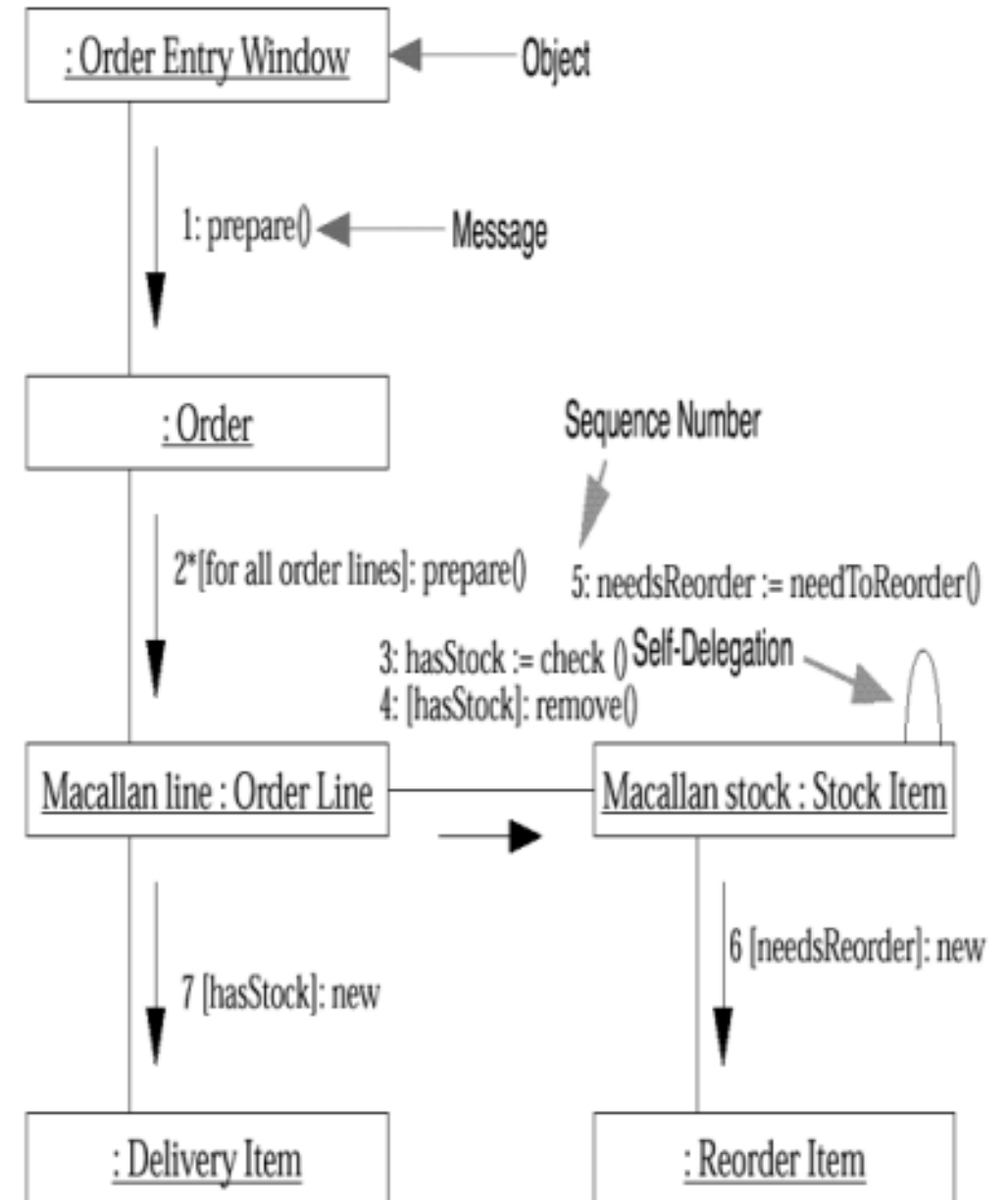
Sequence Diagram: Check Failure

- textual descriptions of what's happening along the left side of the sequence diagram
- Killing threads checking transactions
- A sequence diagram can show two scenarios
 - But not practiced
 - Becomes difficult to understand



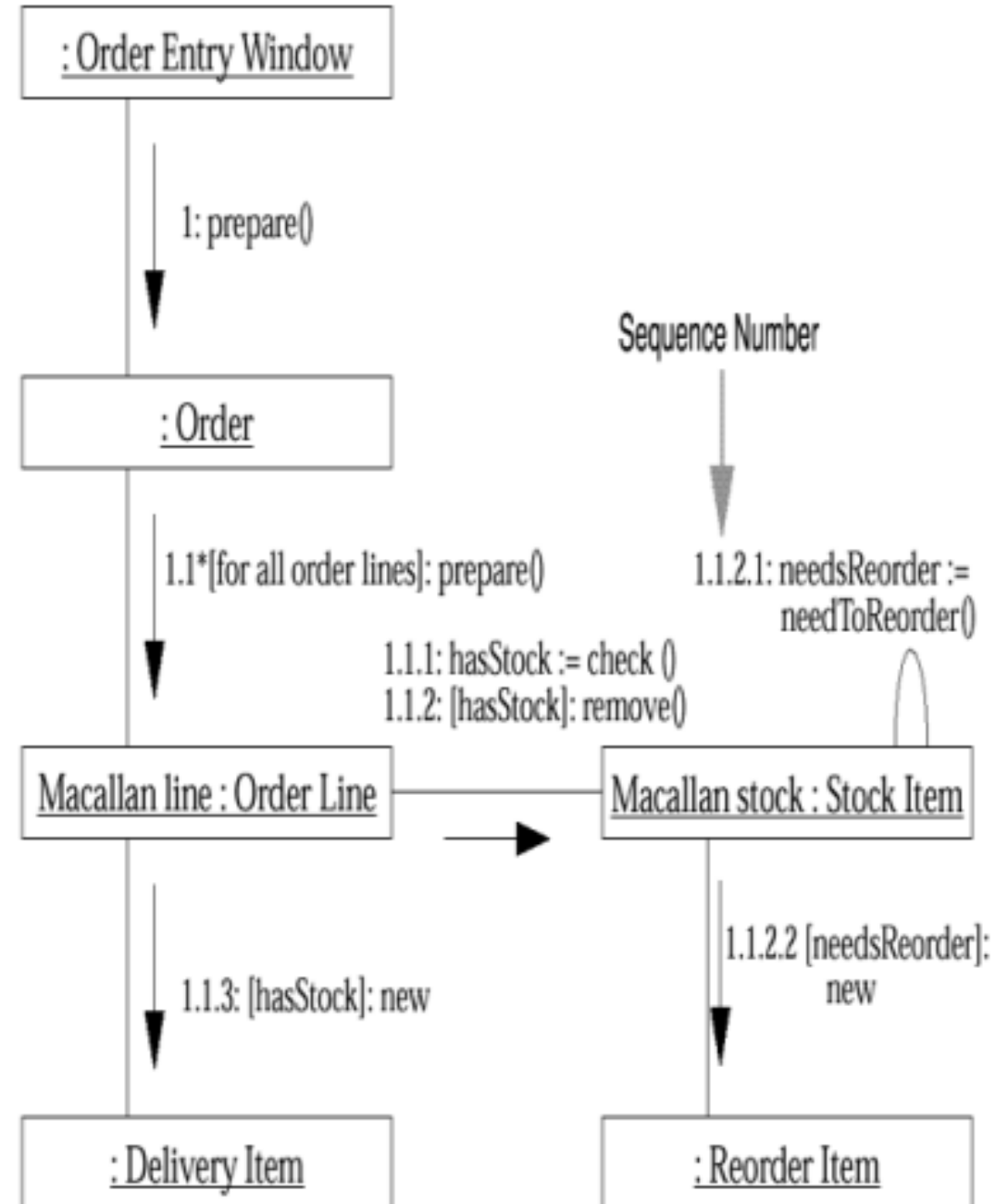
Collaboration Diagrams

- Objects are shown as icons
- the sequence is indicated by numbering the messages
- the spatial layout allows you to show other things more easily
 - how the objects are linked together



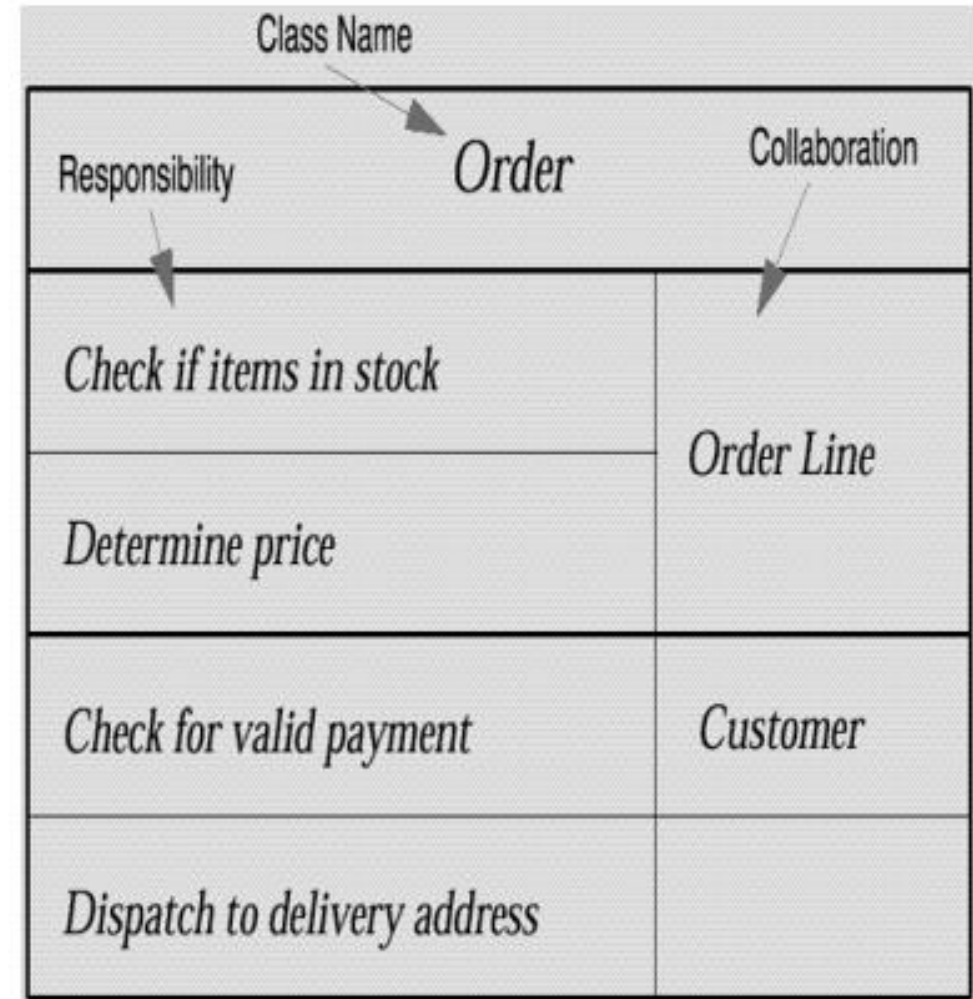
Using Decimal Numbering System

- makes it clear which operation is calling which other operation,
 - makes it harder to see the overall sequence
- Distinguishes original flow and sub-flows of operations
- various forms of the UML's object naming scheme
objectName : ClassName,
 - the object name or the class name may be omitted.
 - Macallan line : Order Line indicates an order line named Macallan



Class-Responsibility-Collaboration (CRC) Cards

- interaction diagrams can be awkward to use when exploring alternatives
- Classes are represented on 4 × 6 index cards
 - Responsibilities are written
 - a high-level description of the purpose of the class
 - capture the purpose of the class in a few sentences



CRC Cards

- In interaction diagram alternative diagrams needs to be drawn for alternative scenarios
 - Takes long time
- In CRC you can simply write them with responsibilities
 - responsibility may correspond to an operation
- long lists of low-level responsibilities is not desired
 - any card with more than three responsibilities
 - Splitting suggested with a class with too many responsibilities

When to Use CRC Cards

- help explore an interaction between classes
- to show how a scenario is implemented
- figure out how the interaction works
 - document it with interaction diagrams
- summarizes the key responsibilities of a class
- useful in highlighting cluttered classes