Interaction Modelling: Sequence Diagrams

Fabrizio Maria Maggi

Institute of Computer Science

(these slides are derived from the book "Object-oriented modeling and design with UML")

Interaction Modelling

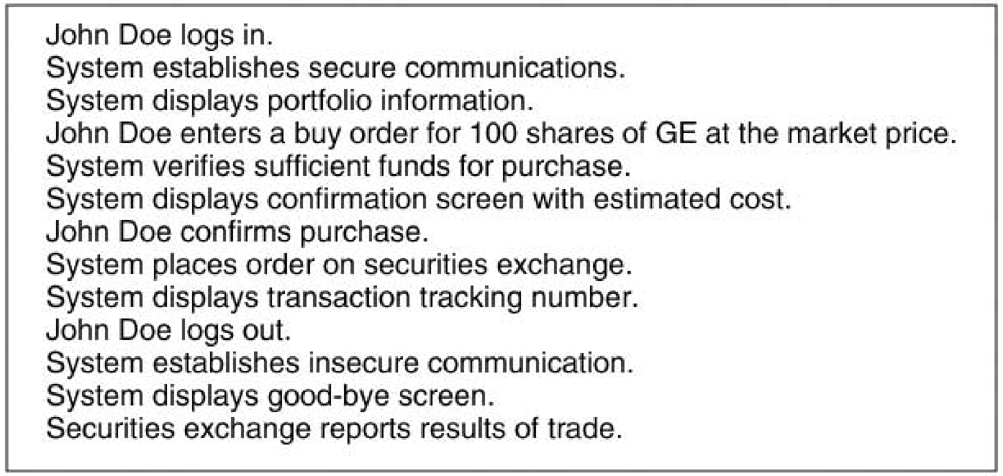
- The class model describes the objects in a system and their relationships
- The interaction model describes how the objects interact to produce useful results
- Interactions can be modeled at different levels of abstraction
  - At a high level use cases describe how a system interacts with outside actors
    - Each use case represents a functionality that a system provides to the user
    - Use cases are helpful for capturing informal requirements
  - Sequence diagrams provide more detail and show the messages exchanged among objects over time
  - Activity diagrams show the steps needed to implement an operation or a business process referenced in a sequence diagram

Scenarios

- A scenario is a sequence of events that occurs during one particular execution of a system
- A scenario highlights the interaction between different objects in a system through a set of steps
  - The steps can be defined at high level at the beginning
  - At later stages the steps can be refined through the specification of messages exchanged between objects and activities performed by the objects within the system
- In a use case description you write one mainstream scenario and one or more exceptional scenarios

Scenarios

- Writing a scenario
  - Identify the objects involved
  - Determine the sender and the receiver of each message
  - Determine the sequence of the messages
  - Add activities



John Doe logs in.
System establishes secure communications.
System displays portfolio information.
John Doe enters a buy order for 100 shares of GE at the market price.
System verifies sufficient funds for purchase.
System displays confirmation screen with estimated cost.
John Doe confirms purchase.
System places order on securities exchange.
System displays transaction tracking number.
John Doe logs out.
System establishes insecure communication.
System displays good-bye screen.
Securities exchange reports results of trade.

Figure 7.4 Scenario for a session with an online stock broker. A scenario is a sequence of events that occurs during one particular execution of a system.

Sequence Diagrams

- A sequence diagram shows the participants in an interaction and the sequence of messages among them
- A sequence diagram shows the interaction of a system with its actors to perform a use case

Sequence Diagrams (Asynchronous)

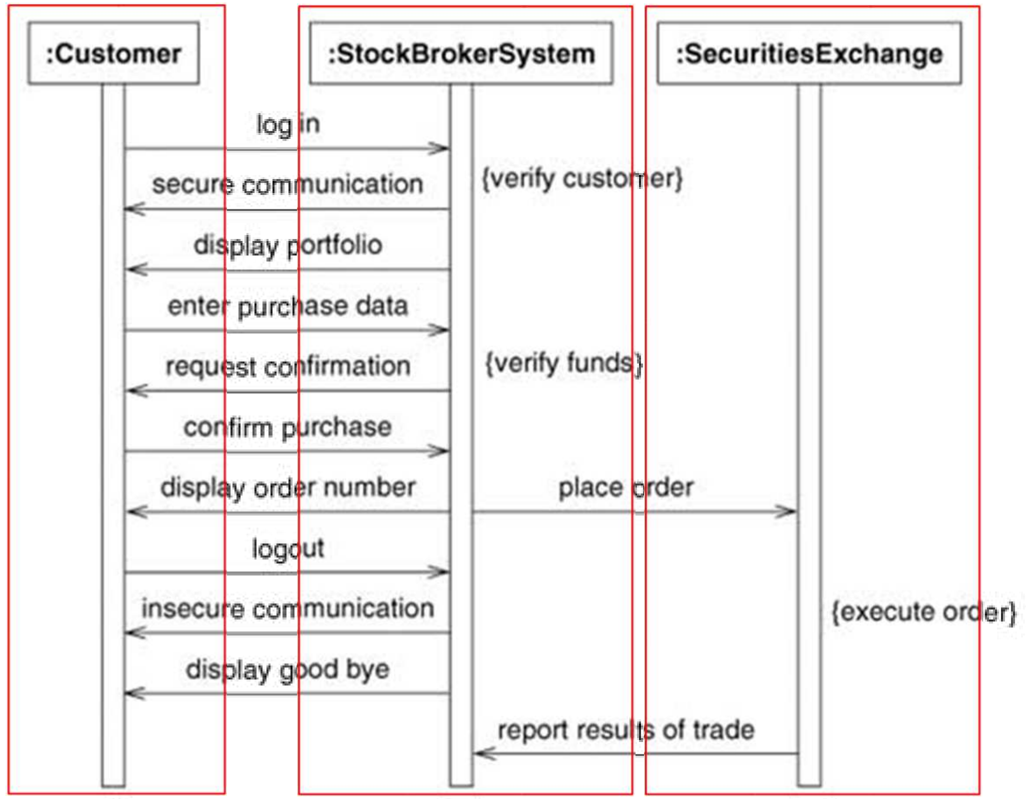


Figure 7.5 Sequence diagram for a session with an online stock broker. A sequence diagram shows the participants in an interaction and the sequence of messages among them.

Object-Oriented Modeling and Design with UML, Second Edition by Michael Blaha and James Rumbaugh. ISBN 0-13-1-015920-4. © 2005 Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Actors as well as the System are represented by a vertical line called a *lifeline*

Sequence Diagrams (Asynchronous)

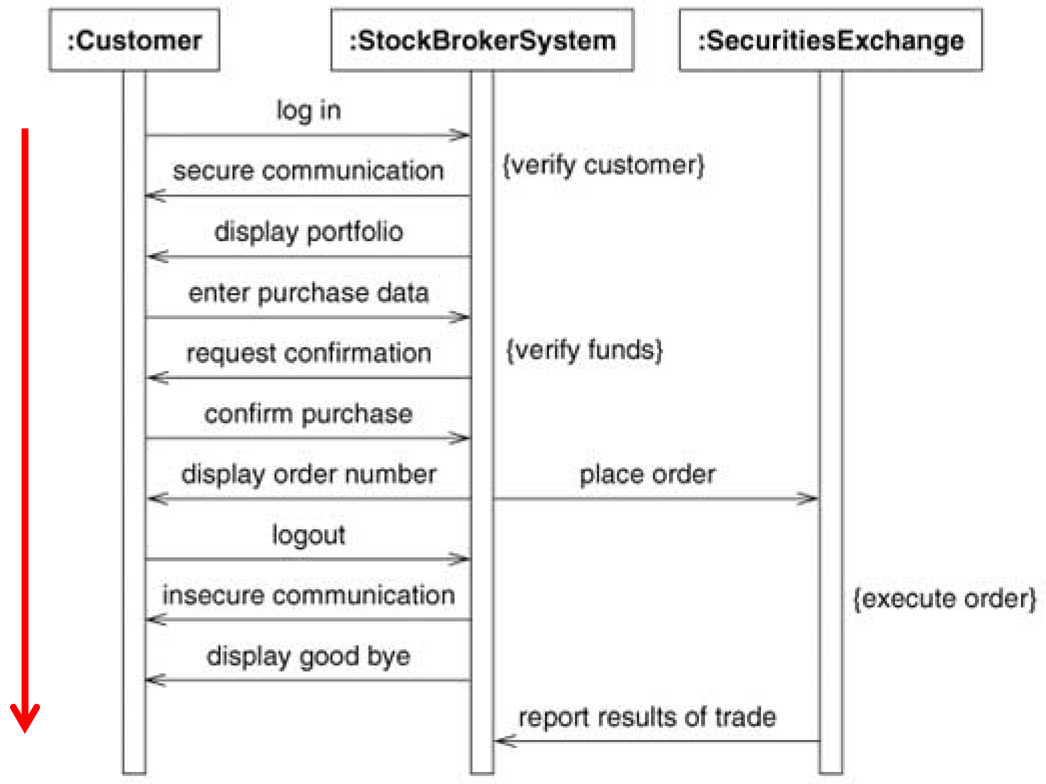


Figure 7.5 Sequence diagram for a session with an online stock broker. A sequence diagram shows the participants in an interaction and the sequence of messages among them.

Object-Oriented Modeling and Design with UML, Second Edition by Michael Blaha and James Rumbaugh. ISBN 0-13-1-015920-4. © 2005 Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Time proceeds from top to bottom but spacing is irrelevant

Sequence Diagrams (Asynchronous)

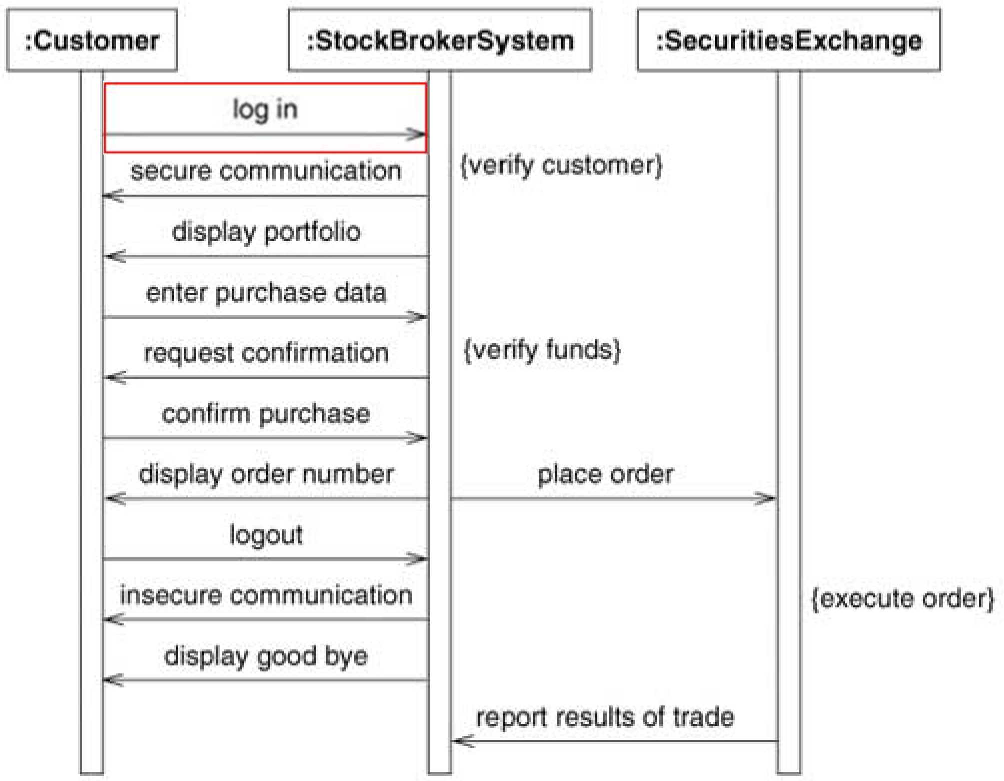


Figure 7.5 Sequence diagram for a session with an online stock broker. A sequence diagram shows the participants in an interaction and the sequence of messages among them.

Object-Oriented Modeling and Design with UML, Second Edition by Michael Blaha and James Rumbaugh. ISBN 0-13-1-015920-4. © 2005 Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

Each message is a horizontal arrow from the sender to the receiver

Sequence Diagrams (Asynchronous)

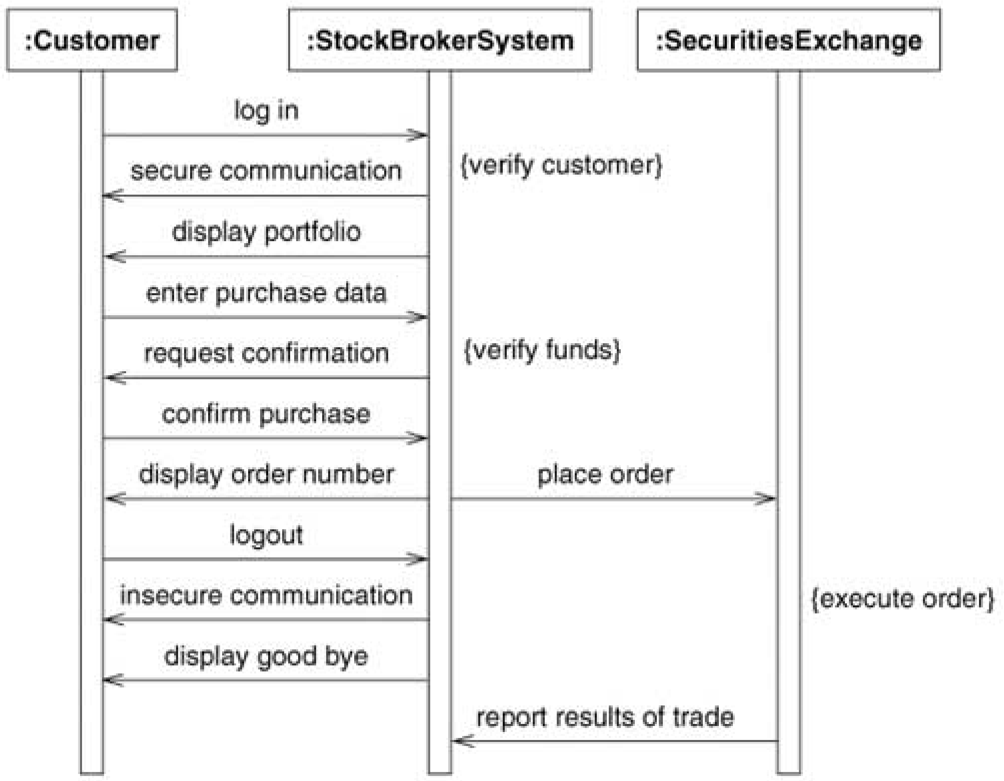
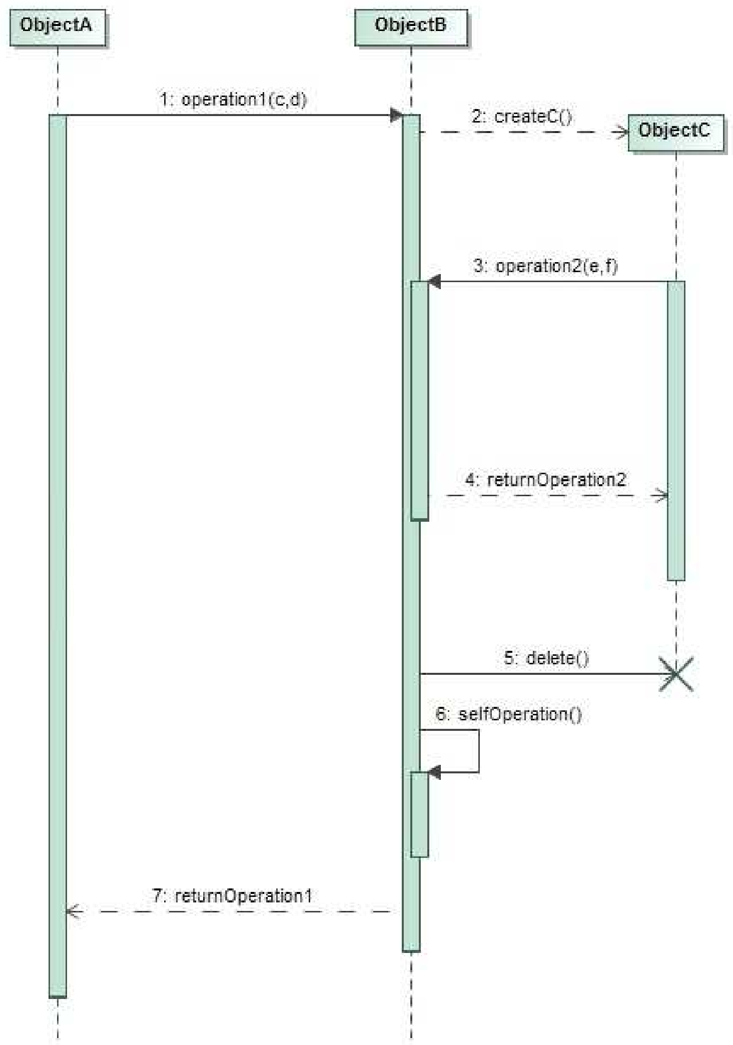


Figure 7.5 Sequence diagram for a session with an online stock broker. A sequence diagram shows the participants in an interaction and the sequence of messages among them.

Object-Oriented Modeling and Design with UML, Second Edition by Michael Blaha and James Rumbaugh. ISBN 0-13-1-015920-4. © 2005 Pearson Education, Inc., Upper Saddle River, NJ. All rights reserved.

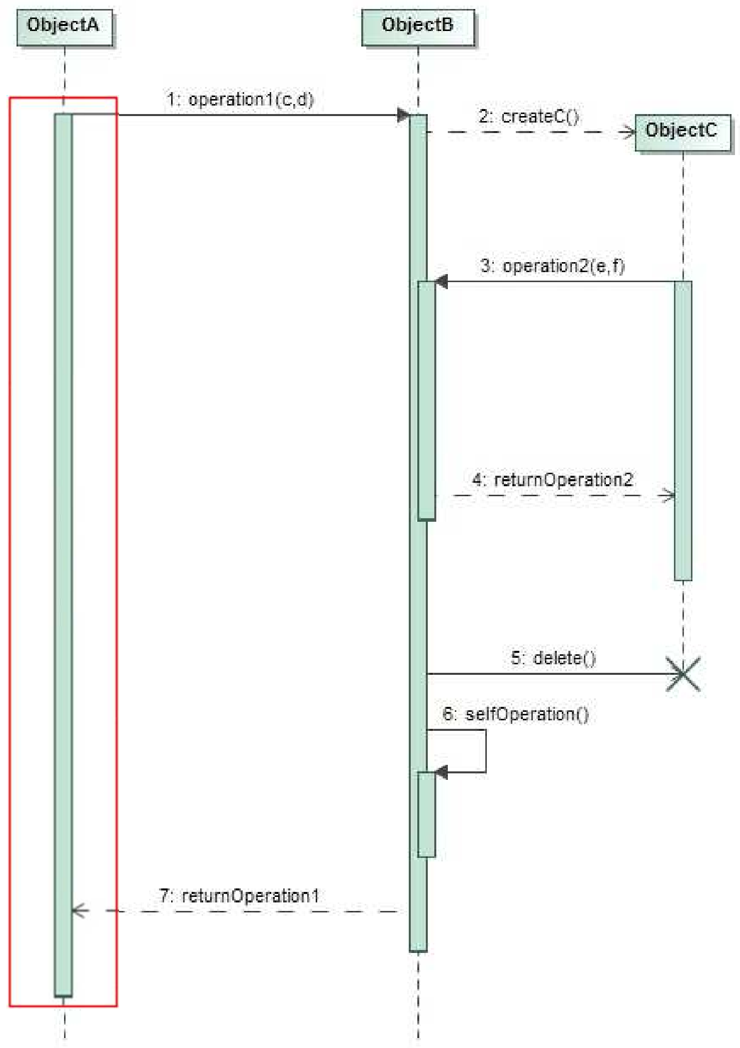
- Are used for high level interactions
- Can contain concurrent signals
  - Stock Broker System sends messages to Customer and Securities Exchange concurrently
- Signals between participants do not need to alternate
  - Stock Broker System sends "secure communication" followed by "display portfolio"

Procedural Sequence Diagrams



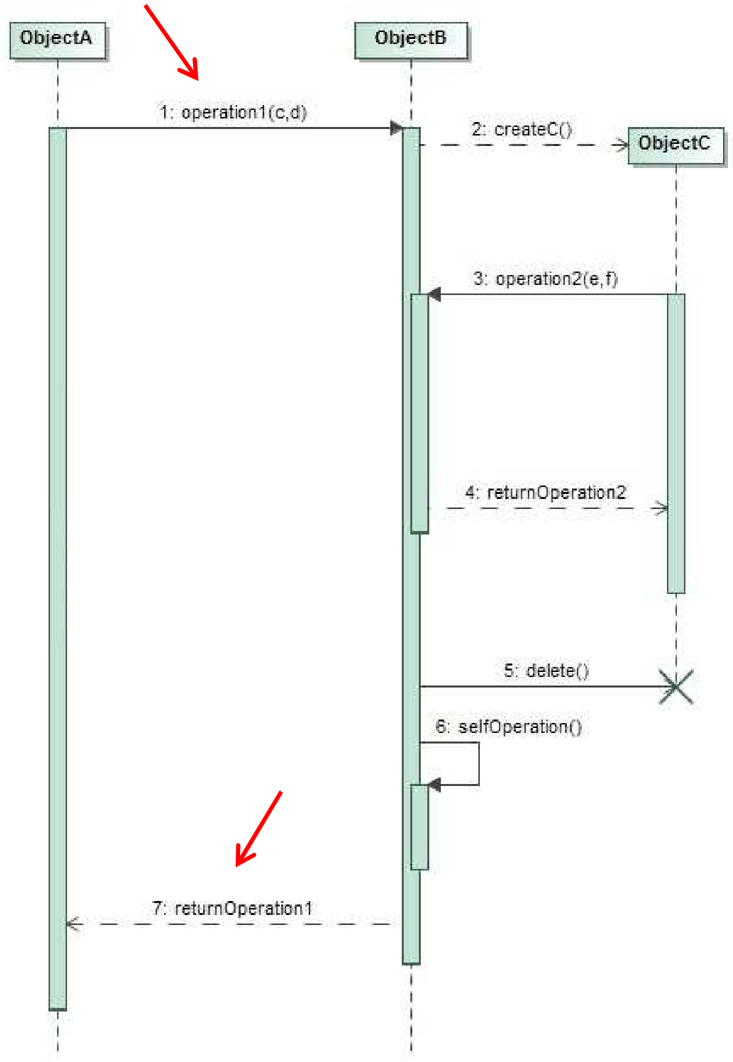
- Most implementation are procedural and limit the number of objects that can execute at a time
- Sequence diagrams can show procedure calls
- An object is activated only when it is called. Once the execution of an operation completes the control returns to the caller and the object becomes inactive

Procedural Sequence Diagrams



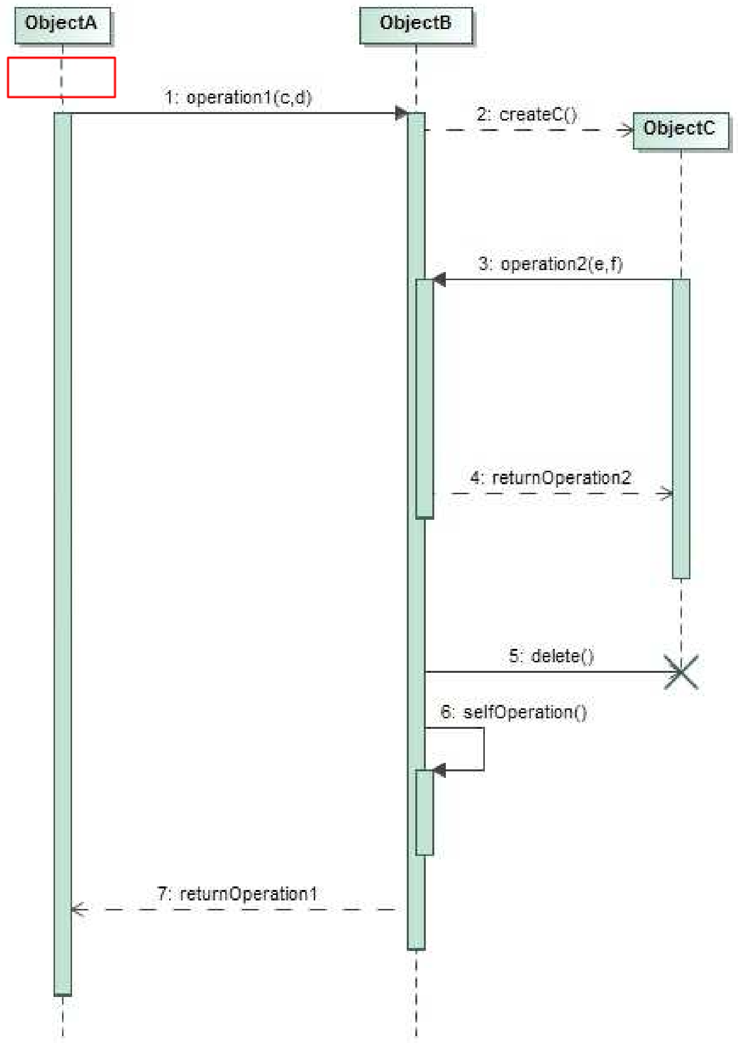
The period of time of an object's execution is a thin rectangle called *activation* or *focus of control*

Procedural Sequence Diagrams



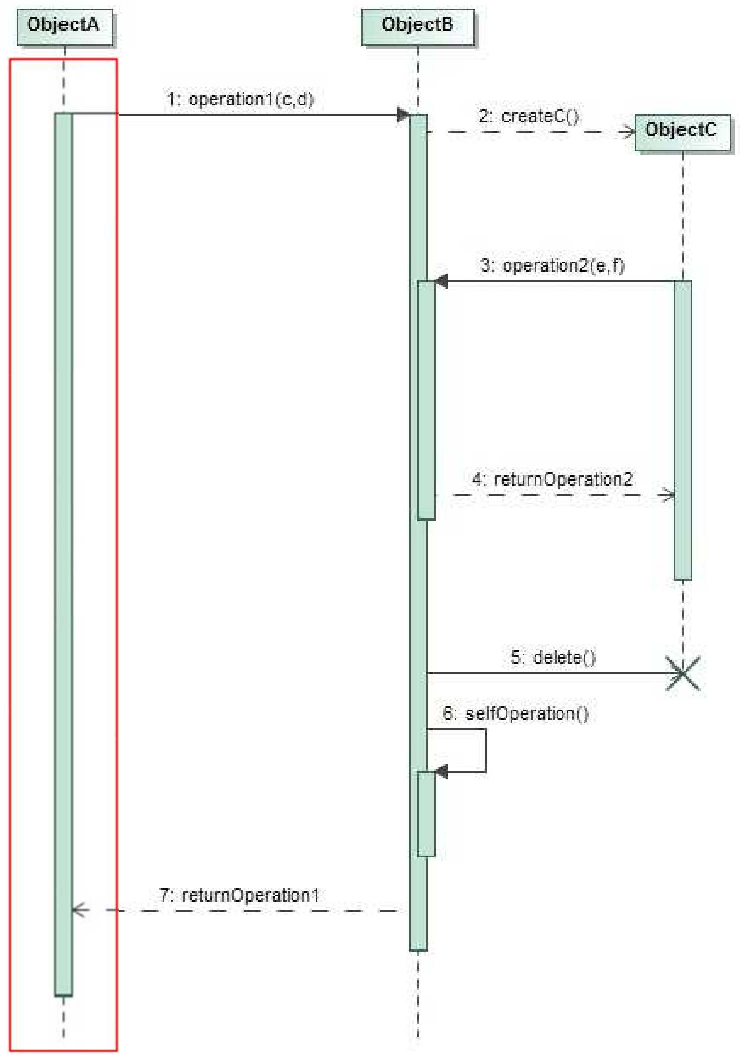
An activation shows the time period during which a call of a method is processed including the time when the called method invoke other methods

Procedural Sequence Diagrams



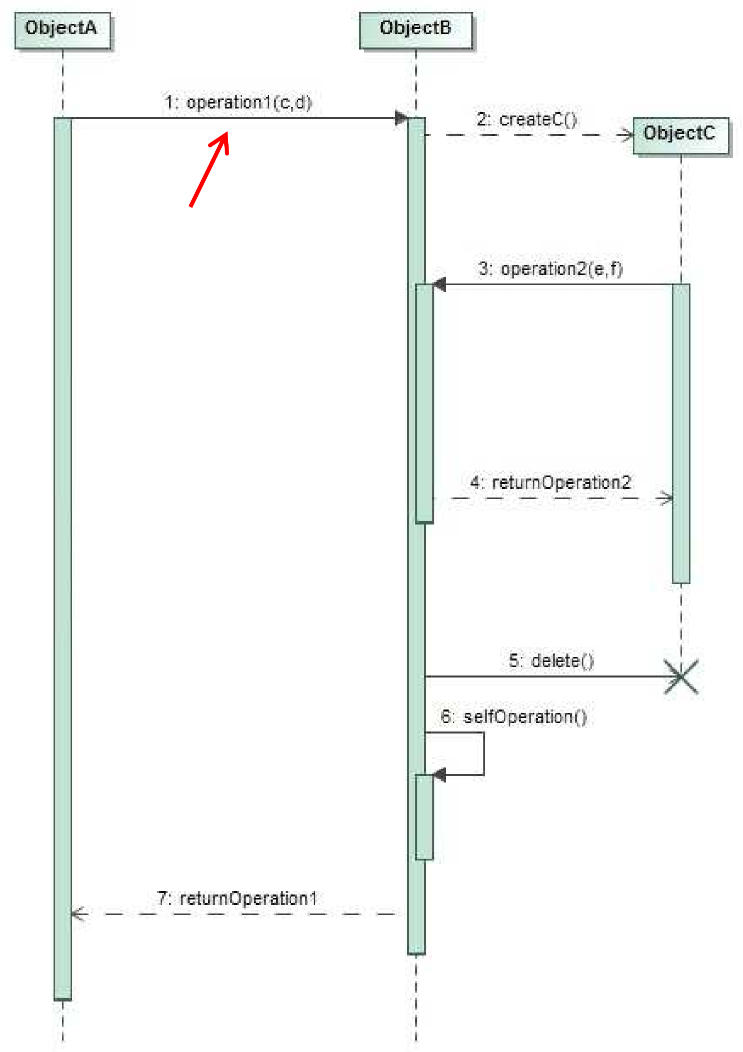
The period of time when an object exists but is not active is shown as a dashed line

Procedural Sequence Diagrams



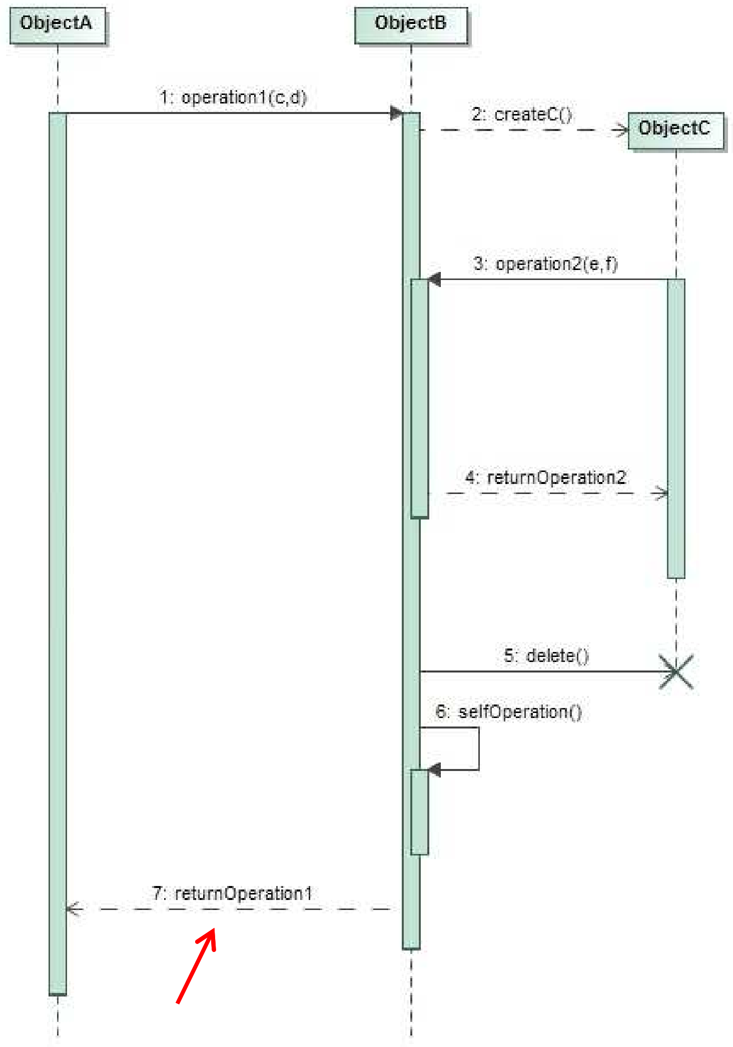
The entire period of time when an object exists is called *lifeline*

Procedural Sequence Diagrams



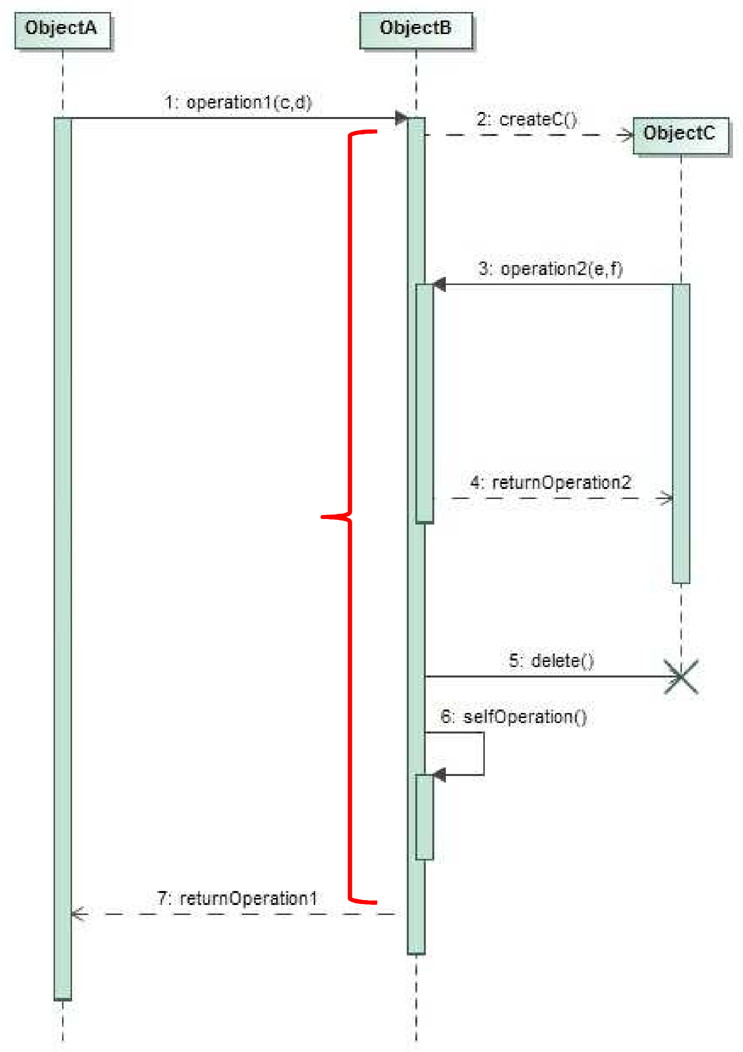
The notation for a call (synchronous message) is an arrow from the calling activation to the activation created by the call. The filled arrowed indicates a call (as opposed to the stick arrowhead for an asynchronous signal)

Procedural Sequence Diagrams



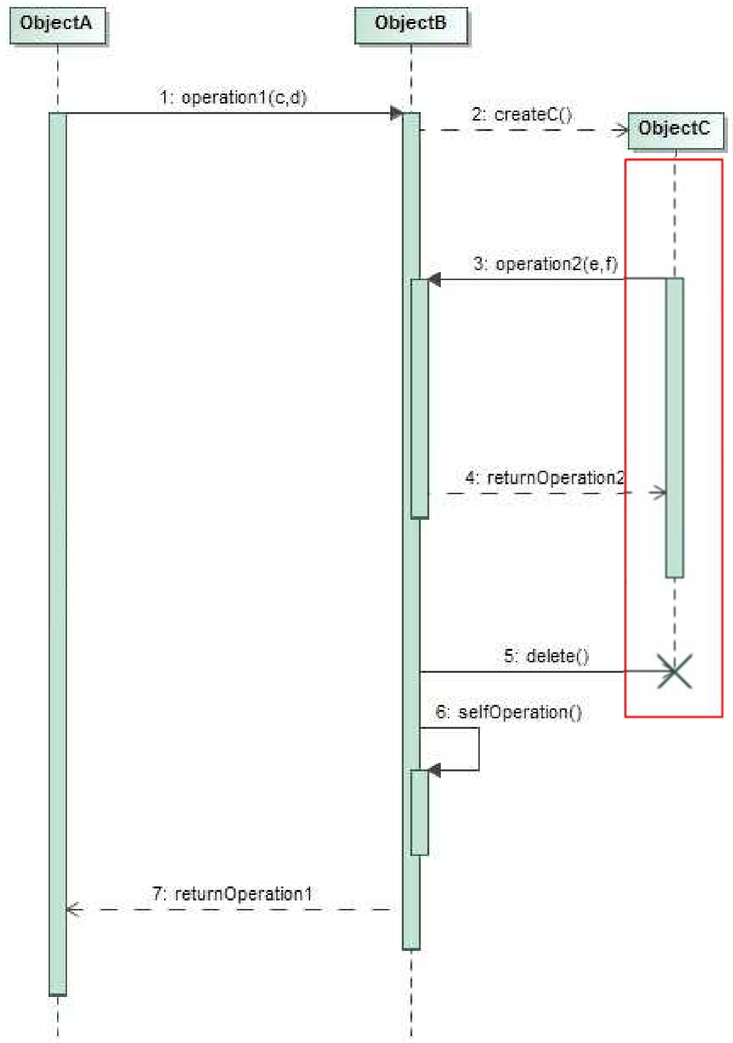
A return of a call is a dashed arrow from the bottom of the called activation to the calling activation

Procedural Sequence Diagrams



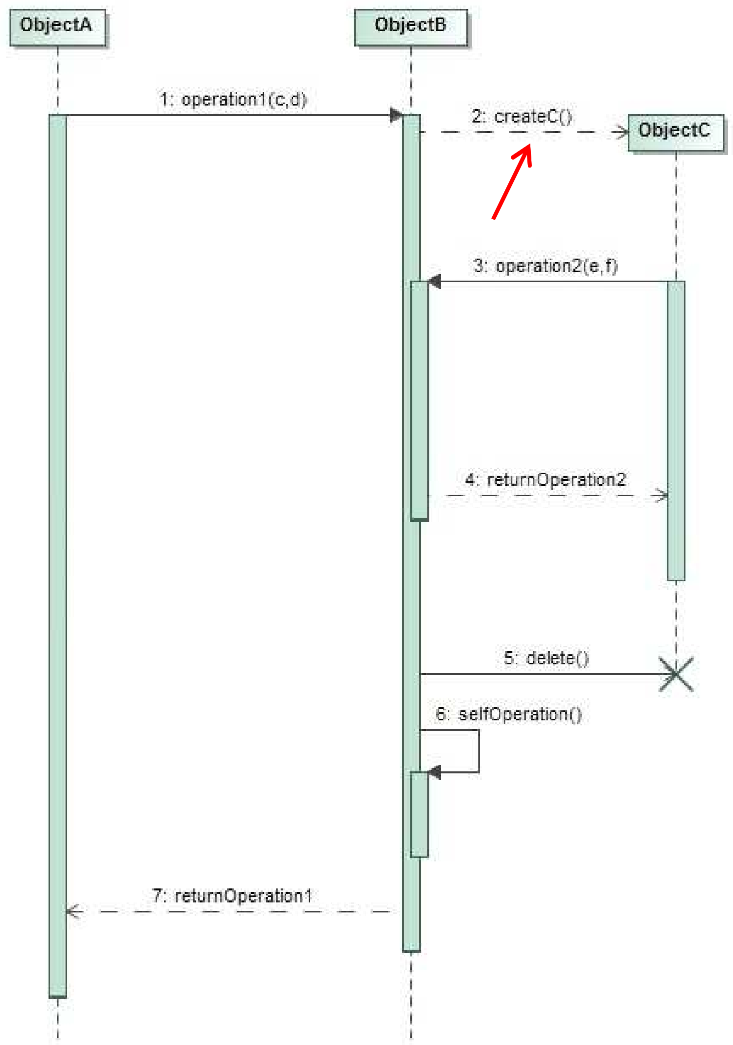
An activation has a call arrow coming into its top and a return arrow leaving its bottom

Procedural Sequence Diagrams



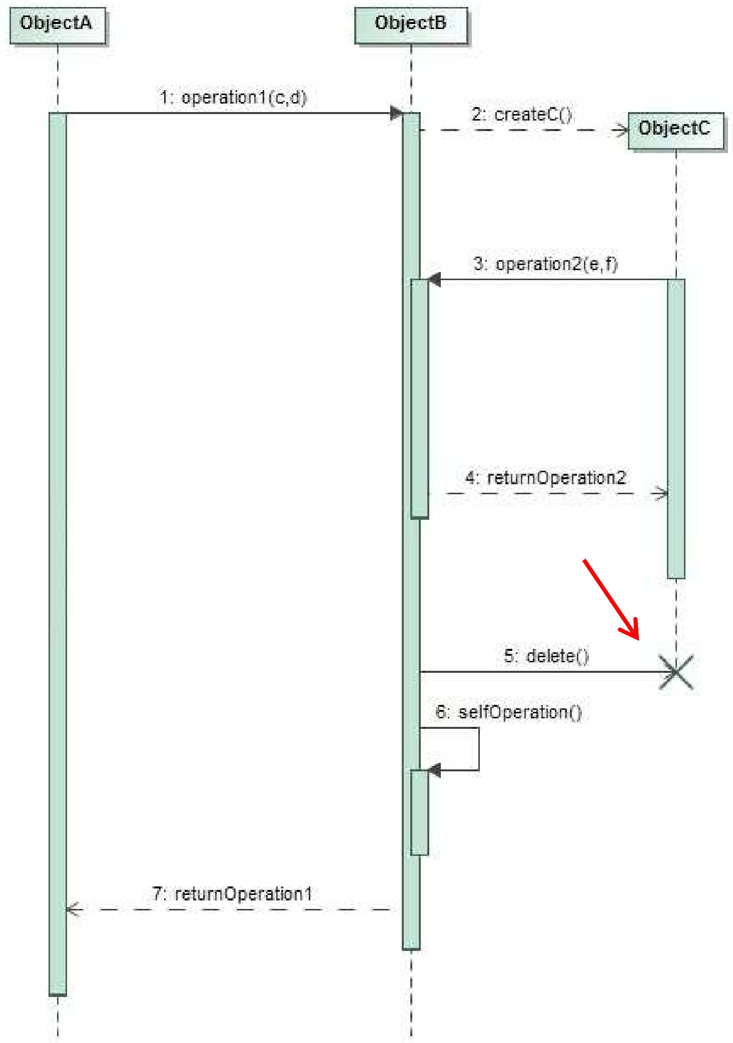
Objects A and B exist during the entire time shown in the diagram, whereas object C is created and destroyed in a smaller period of time. Therefore, its lifetime does not span the whole diagram

Procedural Sequence Diagrams



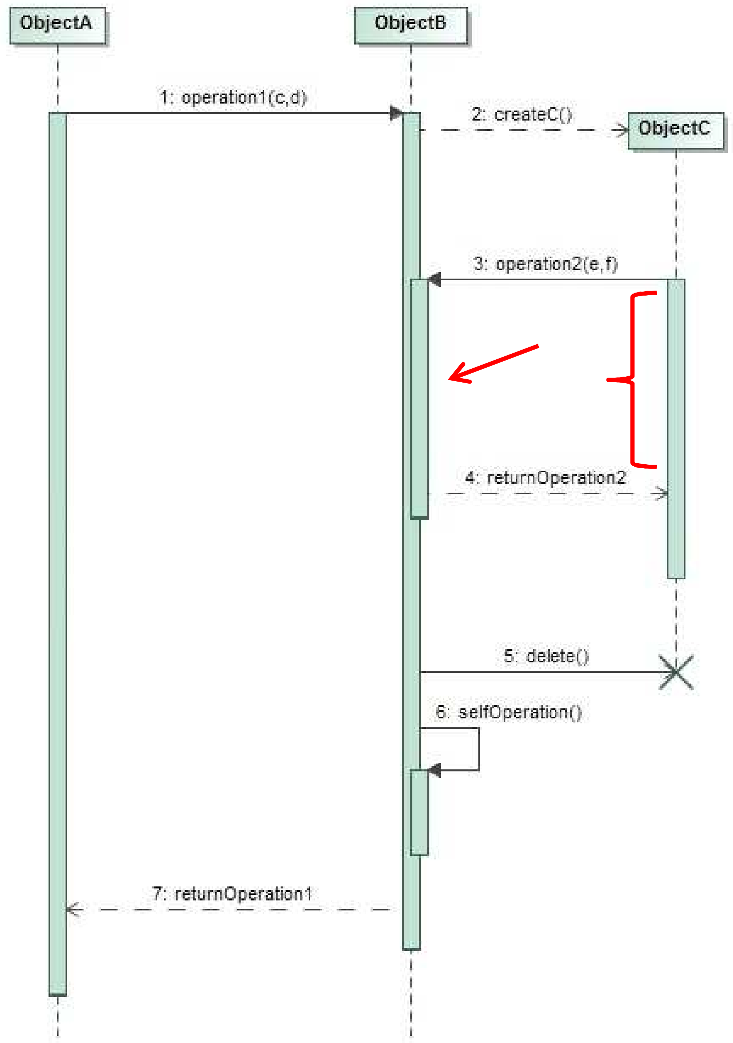
If an object does not exist at the beginning of the sequence diagram, it must be created. UML shows creation by placing the object symbol at the head of the dashed arrow representing the call that creates the object

Procedural Sequence Diagrams



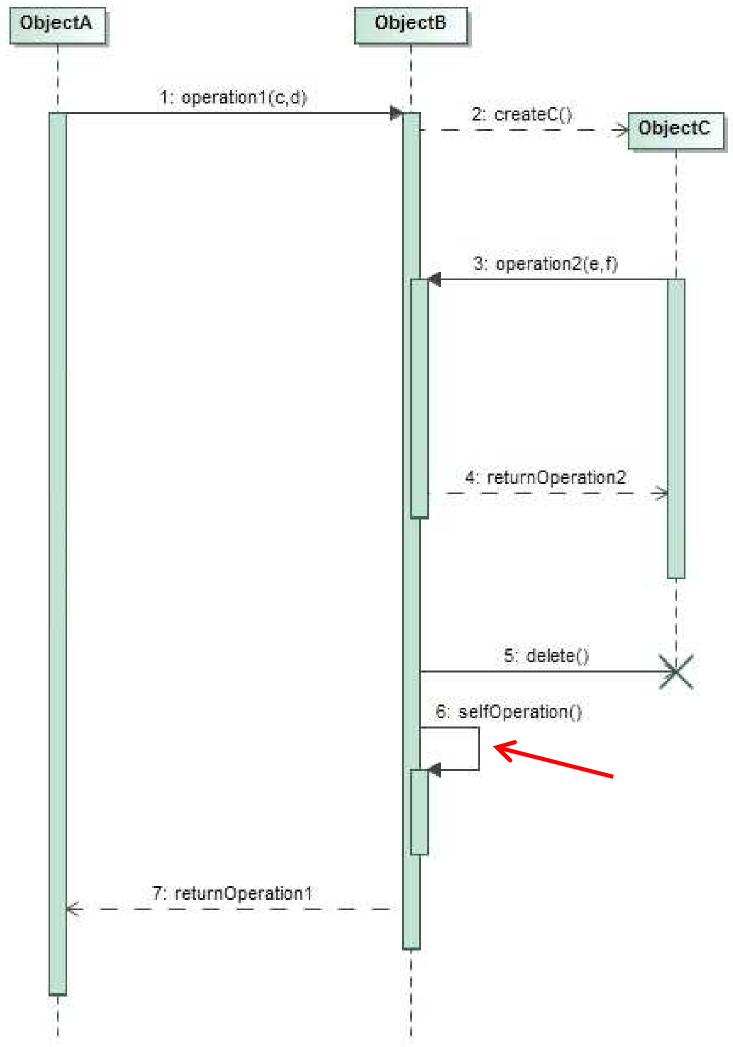
A large 'X' marks the end of the life of an object that is destroyed during the sequence diagram. The 'X' is placed at the head of the call arrow that destroys the object

Procedural Sequence Diagrams



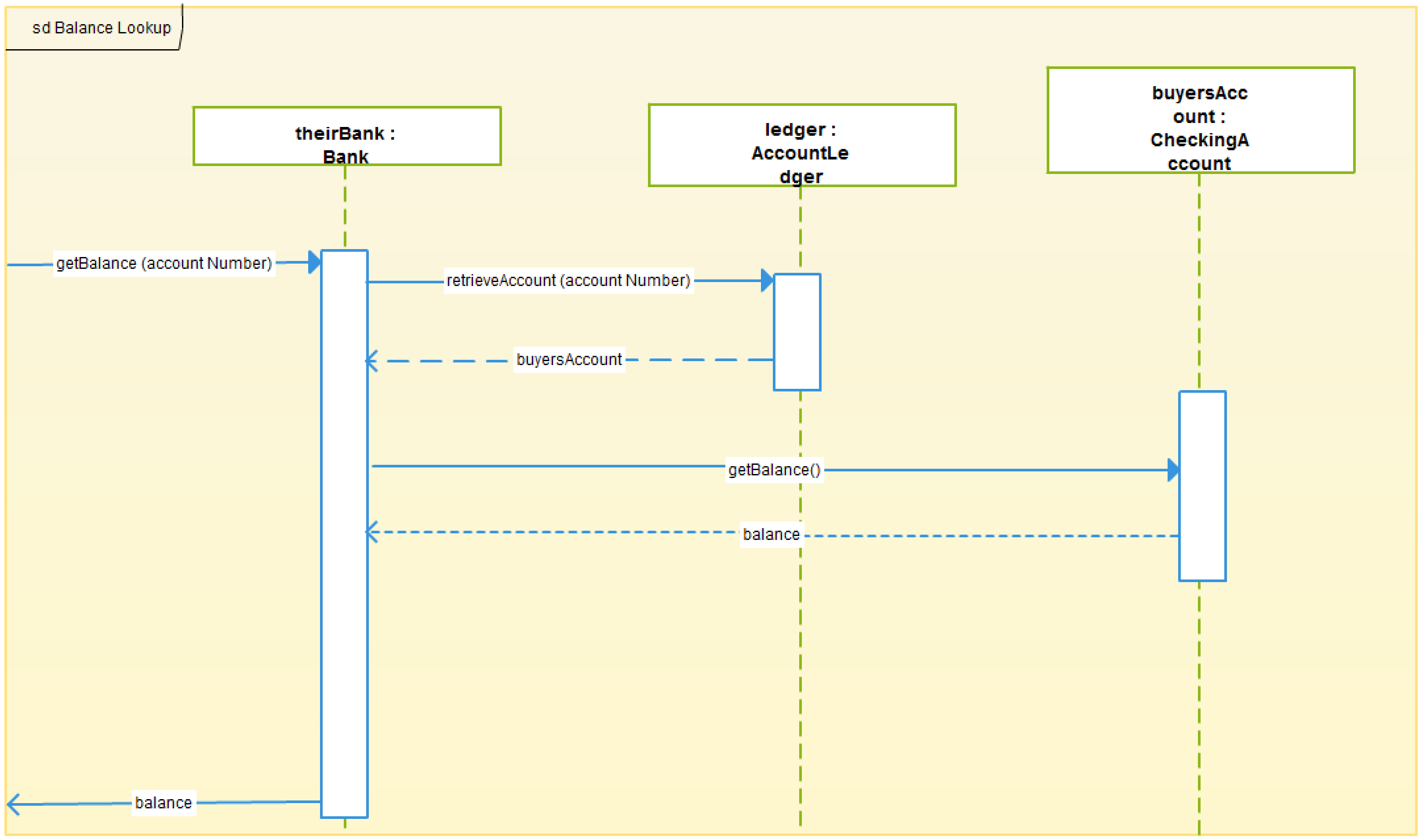
During a call to a method on an object there can be another call to another method on the same object. This call is shown with an arrow from the activation rectangle to the top of an additional rectangle superimposed on the first

Procedural Sequence Diagrams



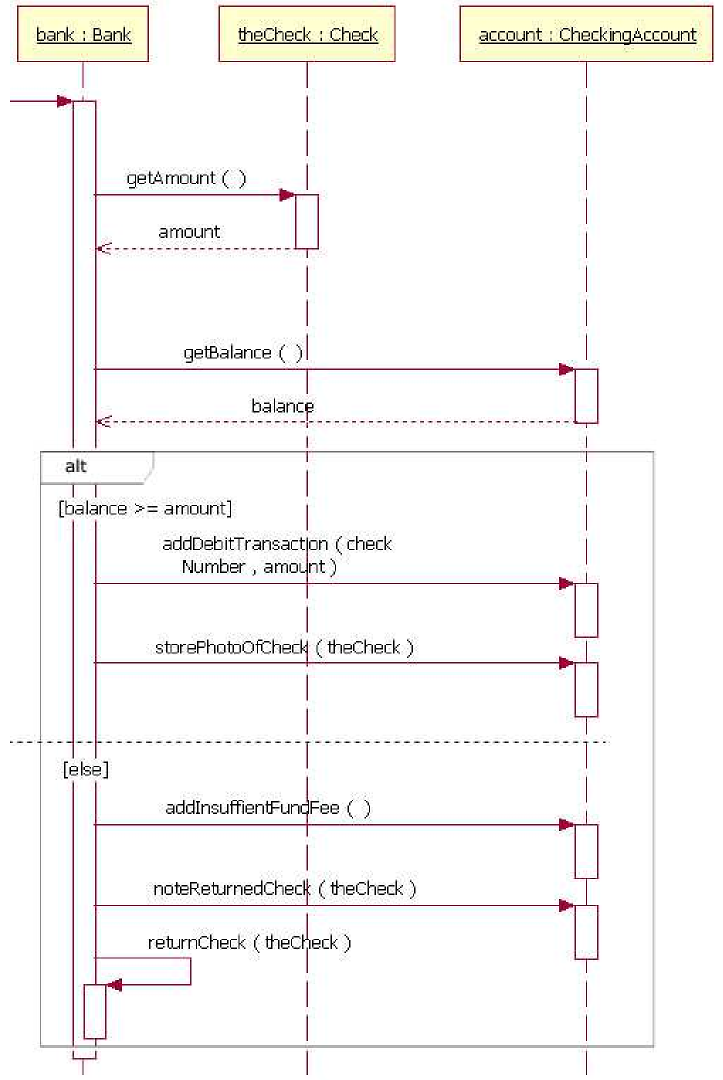
An object can call its own operations (self calls) http://creately.com

A simple example



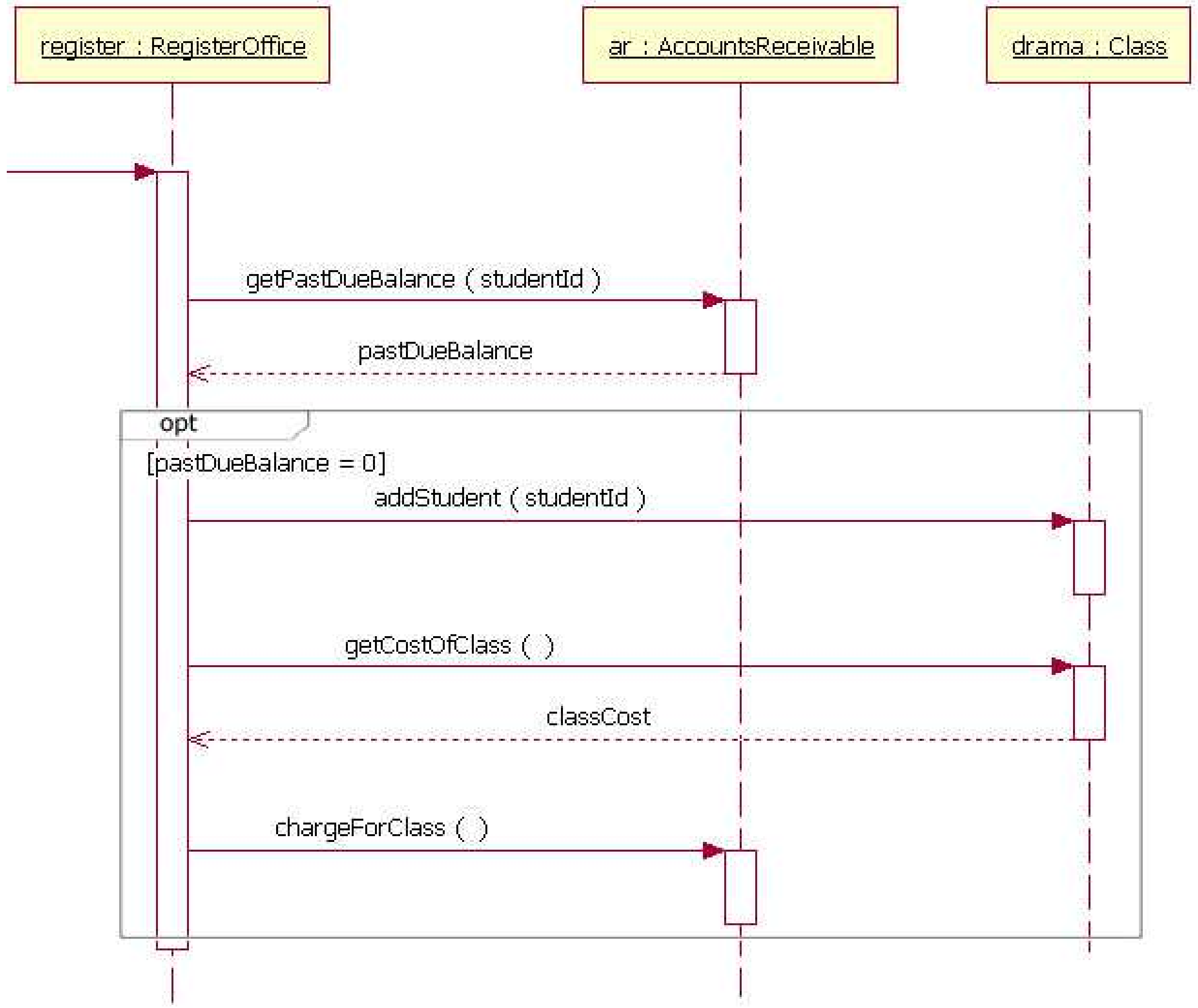
http://www.ibm.com

Advanced Sequence Diagrams: alt



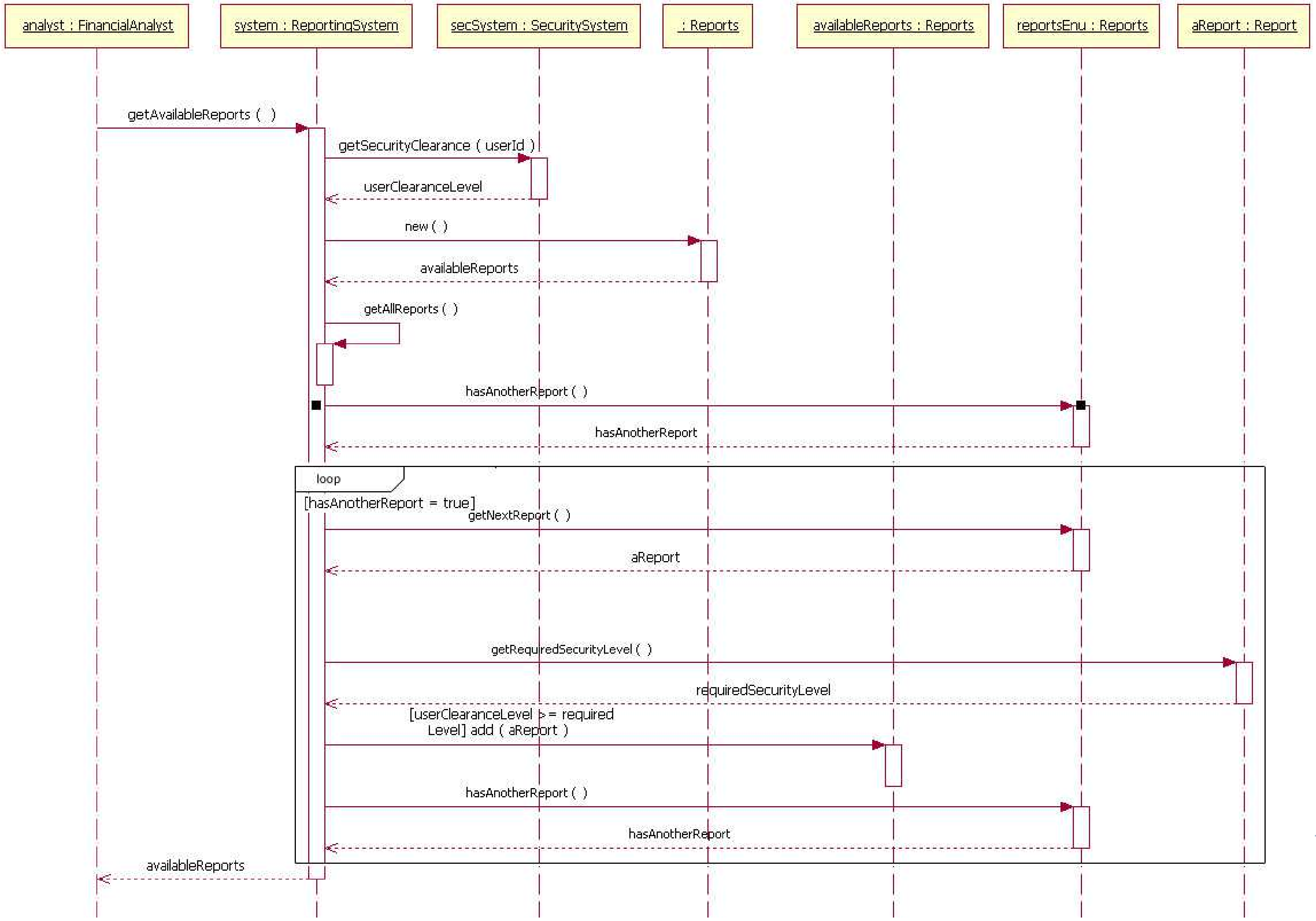
http://www.ibm.com

Advanced Sequence Diagrams: opt



http://www.ibm.com

Advanced Sequence Diagrams: loop



System Conception

- Is the genesis of an application
- A person who understands both business needs and technology thinks of an idea for an application
  - In a later stage developers explore the idea to devise possible solutions
- The purpose of system conception is to defer details and understand the big picture
  - Who is the application for?
  - What problems will it solve?
  - Where will it be used?
  - When is it needed?
  - Why is it needed?
  - How will it work?

Domain Analysis VS Application Analysis

- Domain Analysis
  - Focuses on real-world things whose semantics the application captures
    - Example: a flight is a real-world object that a flight reservation system must represent
  - Domain objects exist independently of any application and are meaningful to business experts
- Application Analysis
  - Addresses the aspects of the application that are visible to users
    - Example: a flight reservation screen is part of a flight reservation system
  - Application objects do not exist in the problem domain and are meaningful only in the context of an application
  - The application model does not prescribe the implementation of an application
    - It describes as a black–box how the application appears from the outside

Domain Analysis

- Domain Class Model
  - Find classes
  - Keep the right classes
  - Find associations
  - Keep the right associations
  - Find attributes
  - Keep the right attributes
  - Grouping classes into packages
- In general, domain analysis does not include interaction models

Application Analysis

- Application Interaction Model
  - Determine the system boundary
  - Find actors
  - Find use cases
  - Prepare normal scenarios
  - Prepare exception scenarios
  - Define sequence diagrams
- Application Class Model
  - Specify user interfaces
    - A user interface is an object that provides the user of a system with a coherent way to access its domain objects, commands and application options
  - Define boundary classes
    - A boundary class is a class that manages communications between a system and an external source
  - Determine controllers
    - A controller is a class that manages control within an application
  - Add operations
    - They are mainly derived from the interaction model