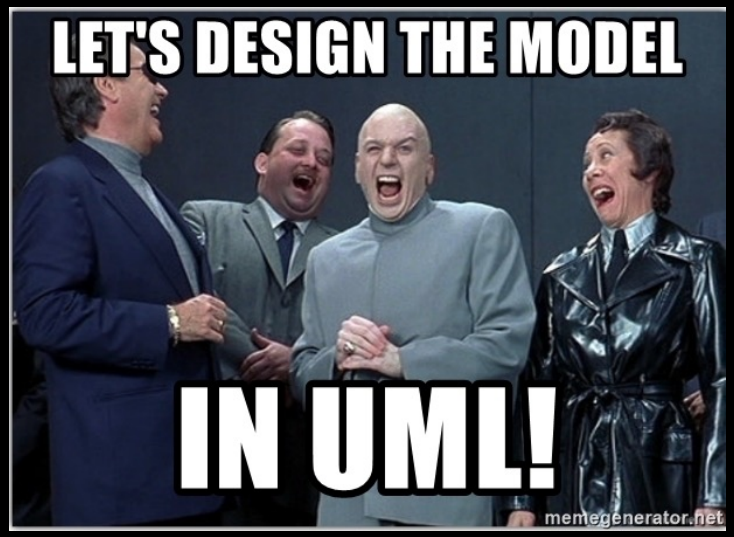
Introduction to Interactions and Interaction Diagrams



OOP

Software Engineering Major

Universidad de las Fuerzas Armadas-ESPE

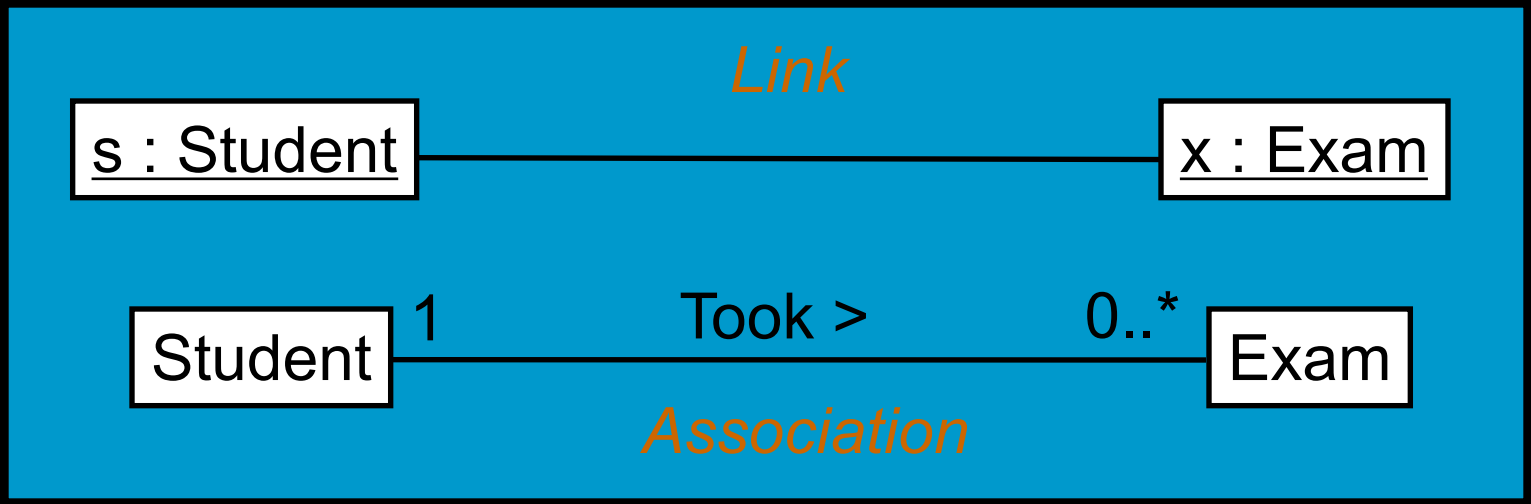
Computer Science Department

Jorge Edison Lascano

Author: Dr. Stephen Clyde

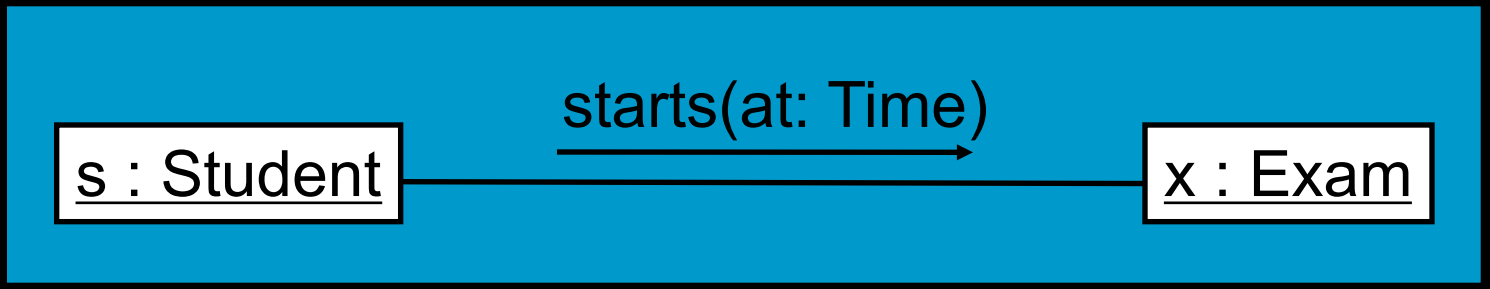
Links

- In general, a link is any kind of semantic connection among objects
  - A link can be an instance of an association, i.e. a persistent link
  - A link can also represent a more transient connection between two objects, e.g. a communication path



Messages

- A link can represent a communication path between two objects
- Object can send a message to another object connected to it via a link path
- A message is represented by a label and an arrow
  - The label may refer to an operation of the target object



Interactions

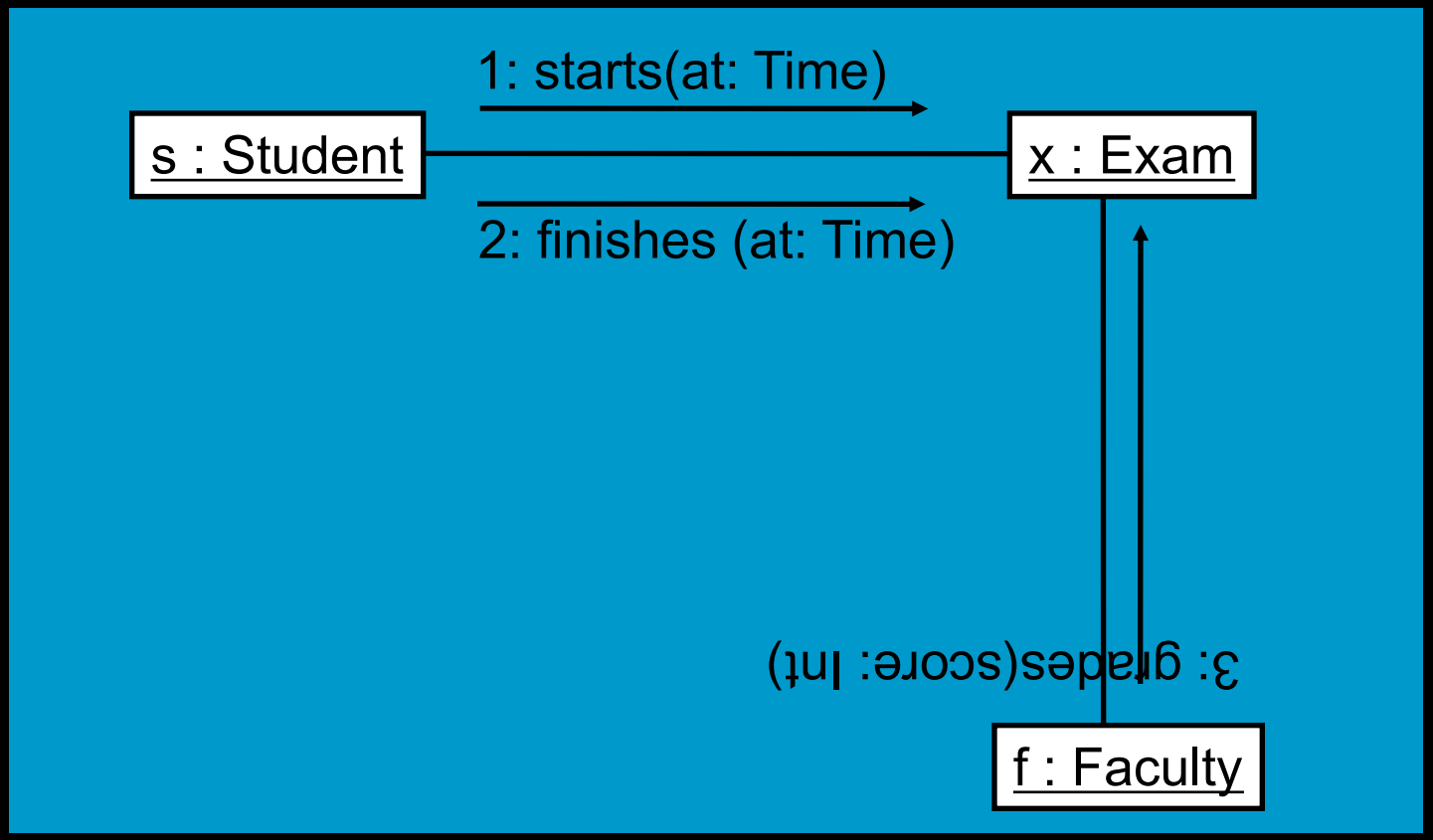
- An interaction is a behavior, comprised of messages exchanged among specific objects within a context to accomplish a purpose
- Interactions describe the dynamic aspect of the collaborations
  - A collaboration fulfills one or more user goals
  - A collaboration can be modeled by describing
    - The objects and their classes – class or object diagrams
    - Object interactions – interaction diagrams
    - Internal object behavior – state charts

Interaction Diagrams

- An *interaction diagram* shows an interaction, which consists
  - of a set of objects
  - messages between those objects
  - their links (communication diagrams only)
- Kinds of interaction diagrams
  - Communication diagrams
  - Sequence diagrams
  - Overview interaction diagrams
  - Timing diagrams

General Syntax for Interactions

- Interaction occur between objects
- Solid lines between objects represent communication pathways, i.e., links
- Arrows represent messages



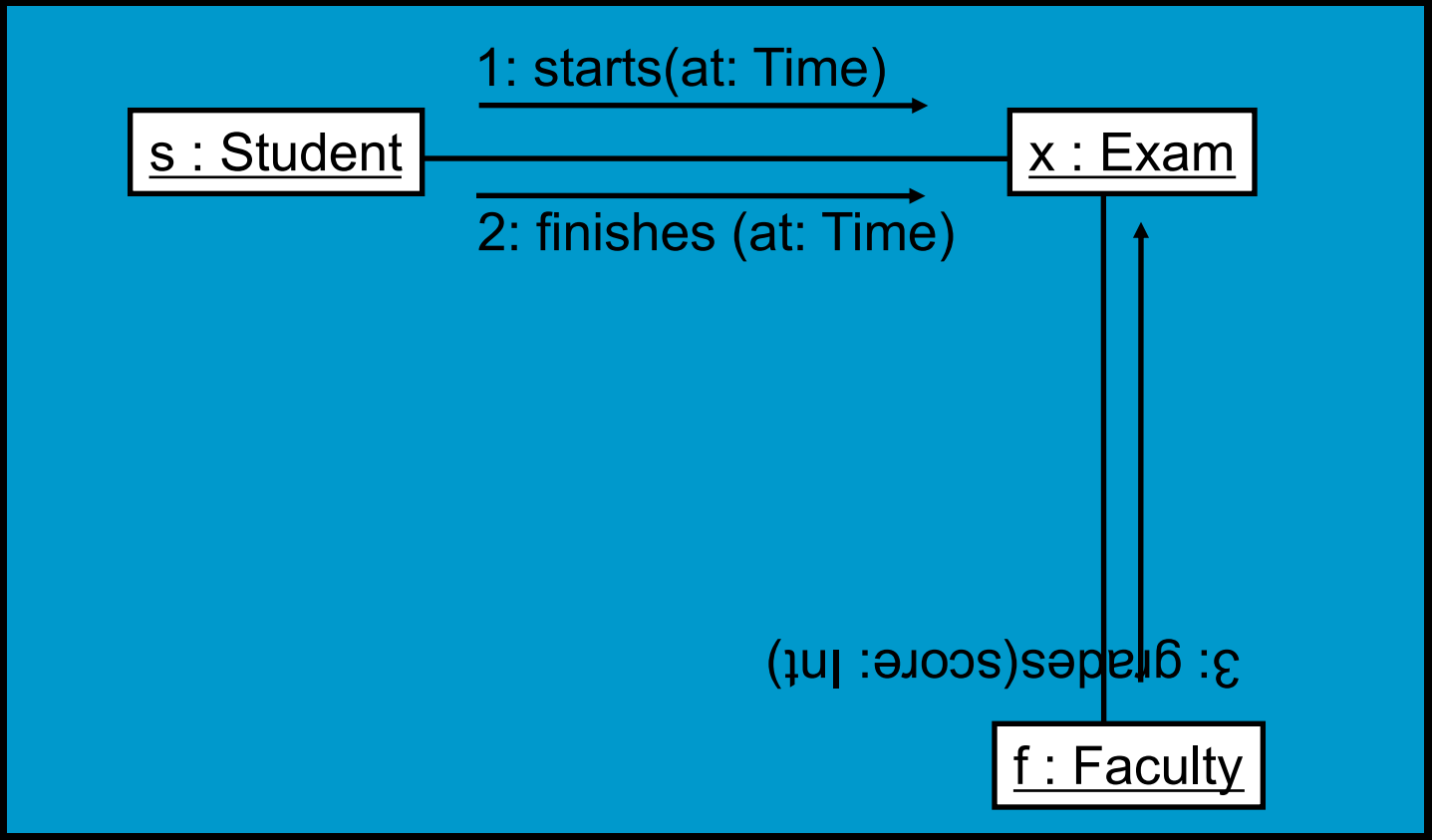
General Syntax for Interactions

- The objects can be concrete or prototypical

- A message is labeled with a signature and an arrow indicating direction of the control flow

Communication Diagrams

- Emphasizes communication pathways between the objects that participate in an interaction

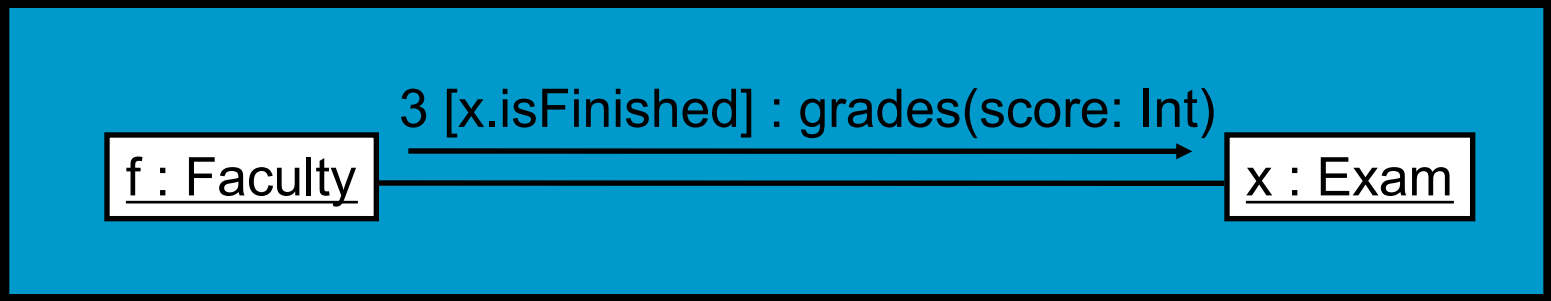


Communication Diagrams - Syntax

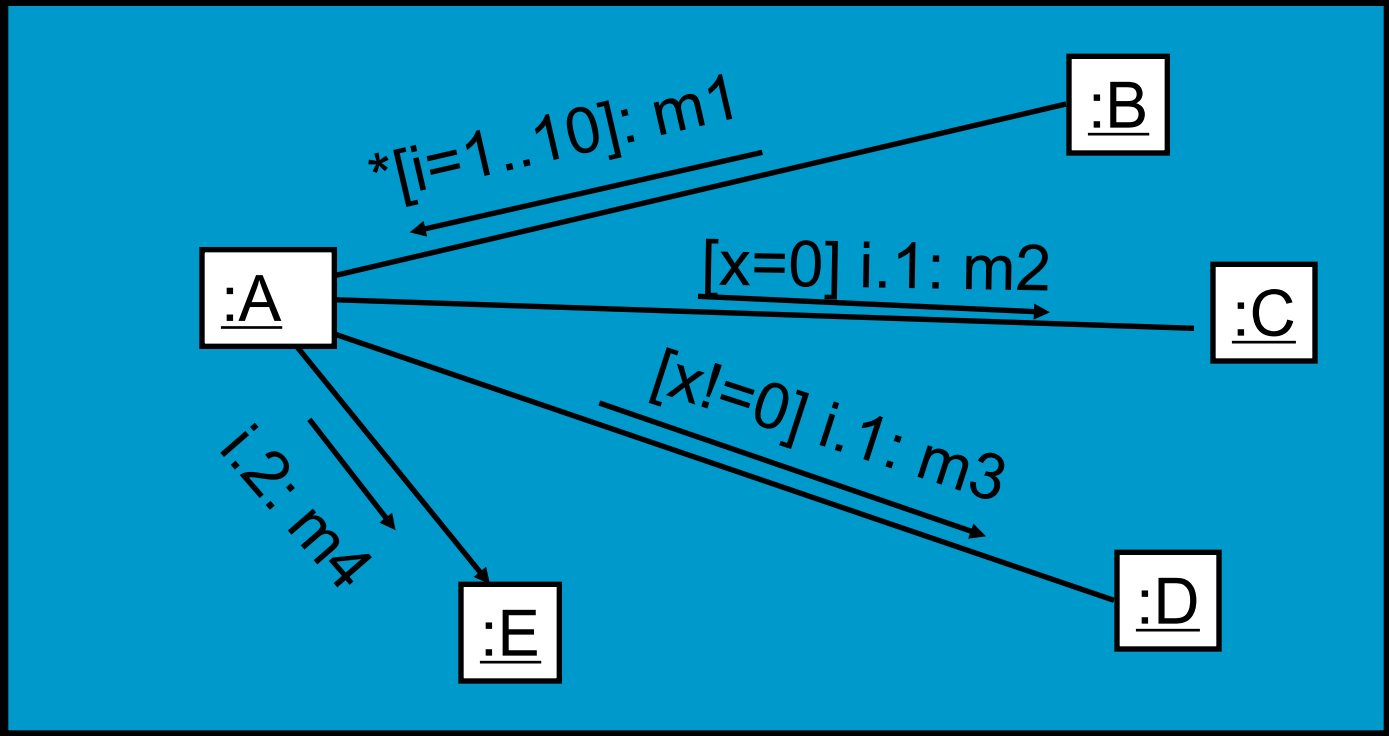
- Lines between objects represent links (specifically, communication paths)
- Links are adorned with messages
- Each messages is proceeded by a sequence number.
  - Sequence numbers can be hierarchical, e.g. 2.1.3
- A sequences number can include a star to indicate that the message can repeat

Communication Diagrams - Syntax

- Messages can be guarded by conditions
  - When you place a guard on a message, it defines a possible ‘branch’
  - The condition is written after the sequence number
  - Give each alternate branch the same sequence number
  - Each alternate branch must have a non-overlapping condition



Communication Diagrams - Syntax



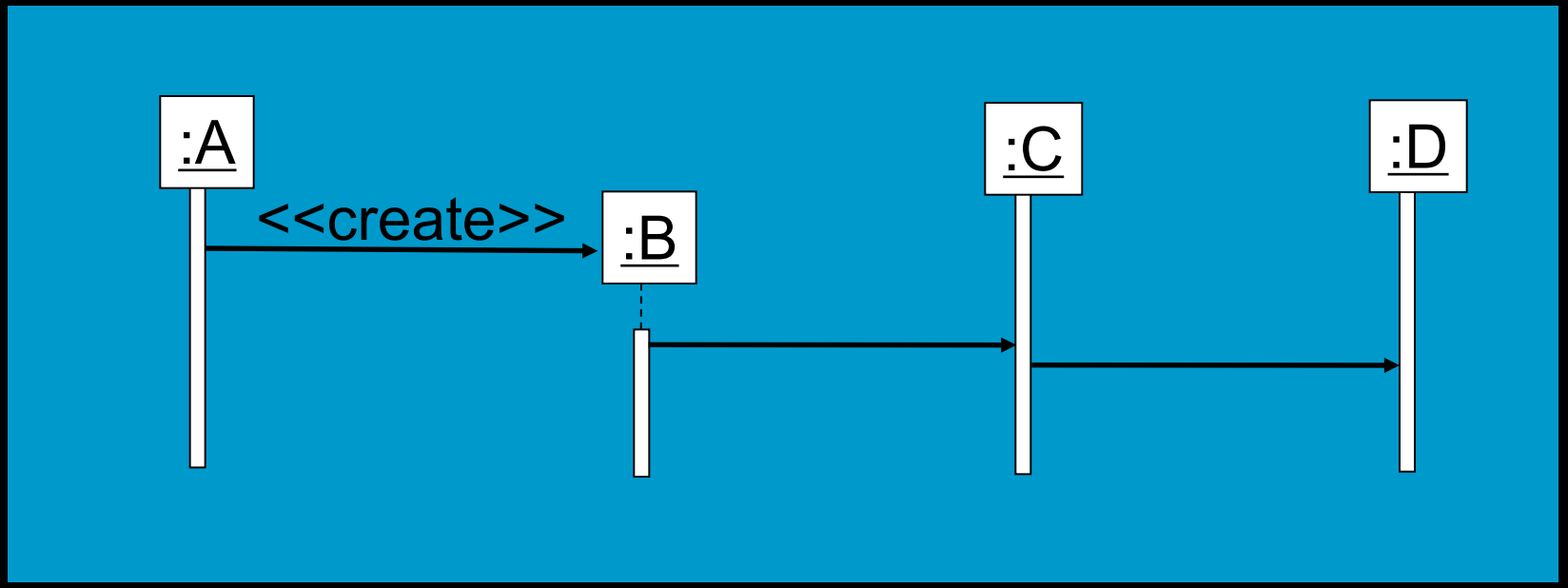
- A *B* object sends 10 *m1* messages to an *A* object
- For each *m1* messages,
  - the *A* object either sends a *m2* message to a *C* object or a *m3* message to a *D* object depending on *x*.
  - The *A* object sends a *m4* to an *E* object

Communication Diagrams - Syntax

- In communication diagrams, the links can be thought of as communication paths
- The end of a link have a stereotype that provides additional semantics
  - *«local»* - Specifies that the corresponding object is visible because it is in a local scope
  - *«parameter»* - Specifies that the corresponding object is visible because it is a parameter
  - *«global»* - Specifies that the corresponding object is visible because it is in an enclosing scope
  - «self» - Specifies that the corresponding object is visible because it is the dispatcher of the message

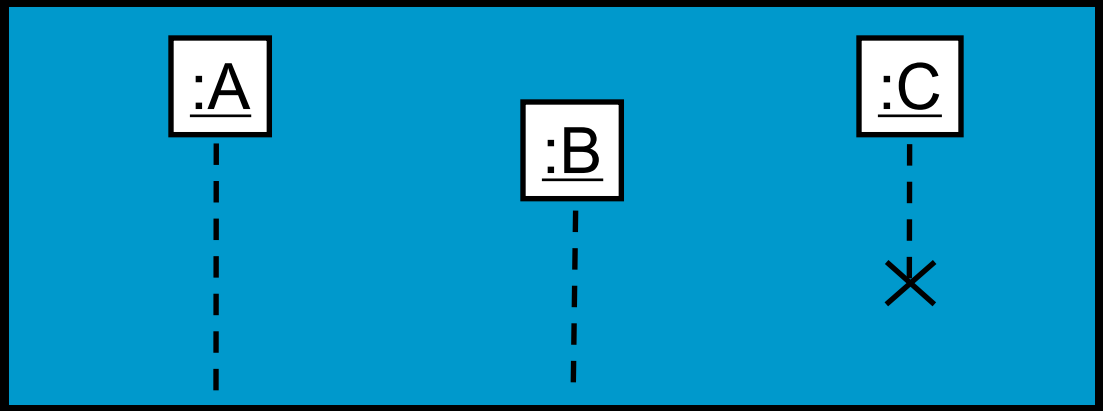
Sequence Diagrams

- Emphasizes the time ordering of messages
- Objects are arranged along the X-axis
- Messages are arranged chronological along the Y-axis
- Communication Links are not shown



Sequence Diagram - Syntax

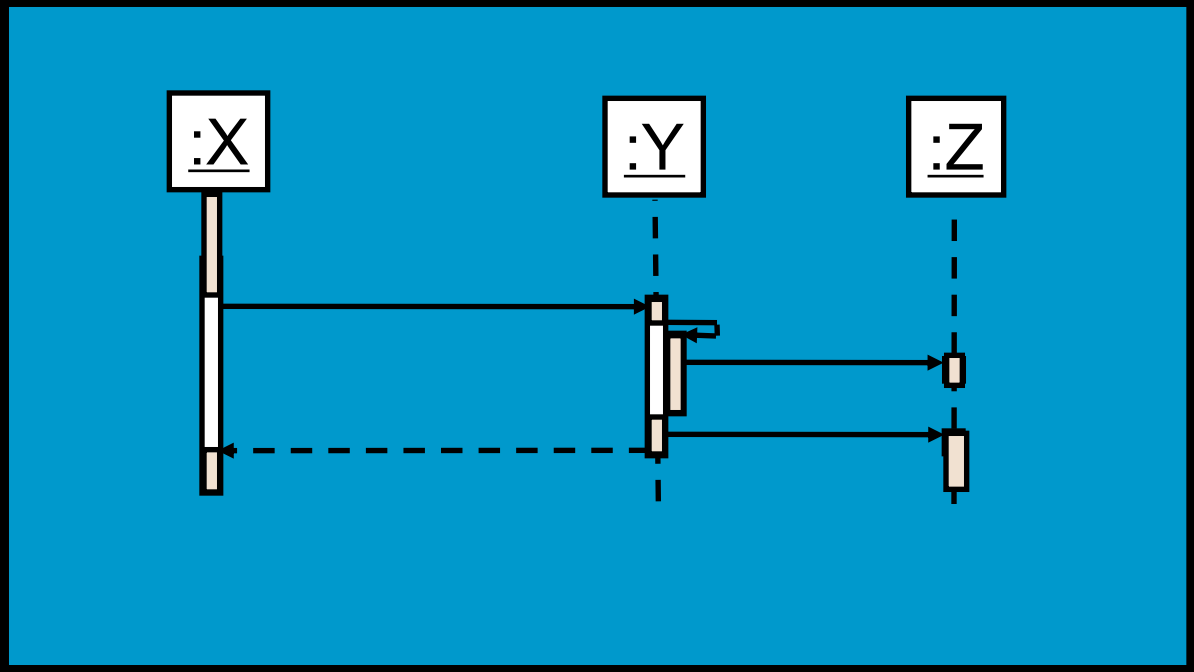
- The position of an object on the Y-axis is relative to its creation
- The dashed-line dropping down from object represents that object's *life line*
- An X at the end of the line indicates that the object is destroyed.



Sequence Diagram - Syntax

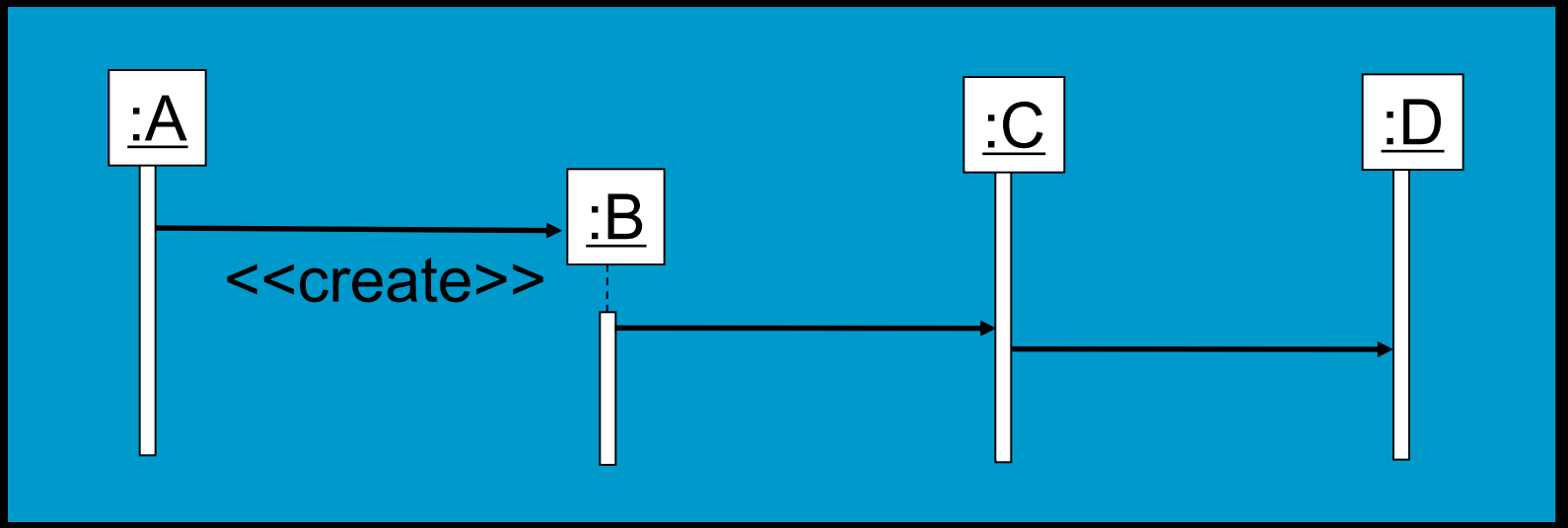
- A thin rectangle on the life-line represents *focus of control*, i.e., the object is performing some action
  - For synchronize method-call type of messages, the focus starts with the arrival of a method-call message and ends with the return message
- Recursion can be indicated by stacking rectangles
- Shading can be used to show actual computing

Sequence Diagram - Syntax



Message Types

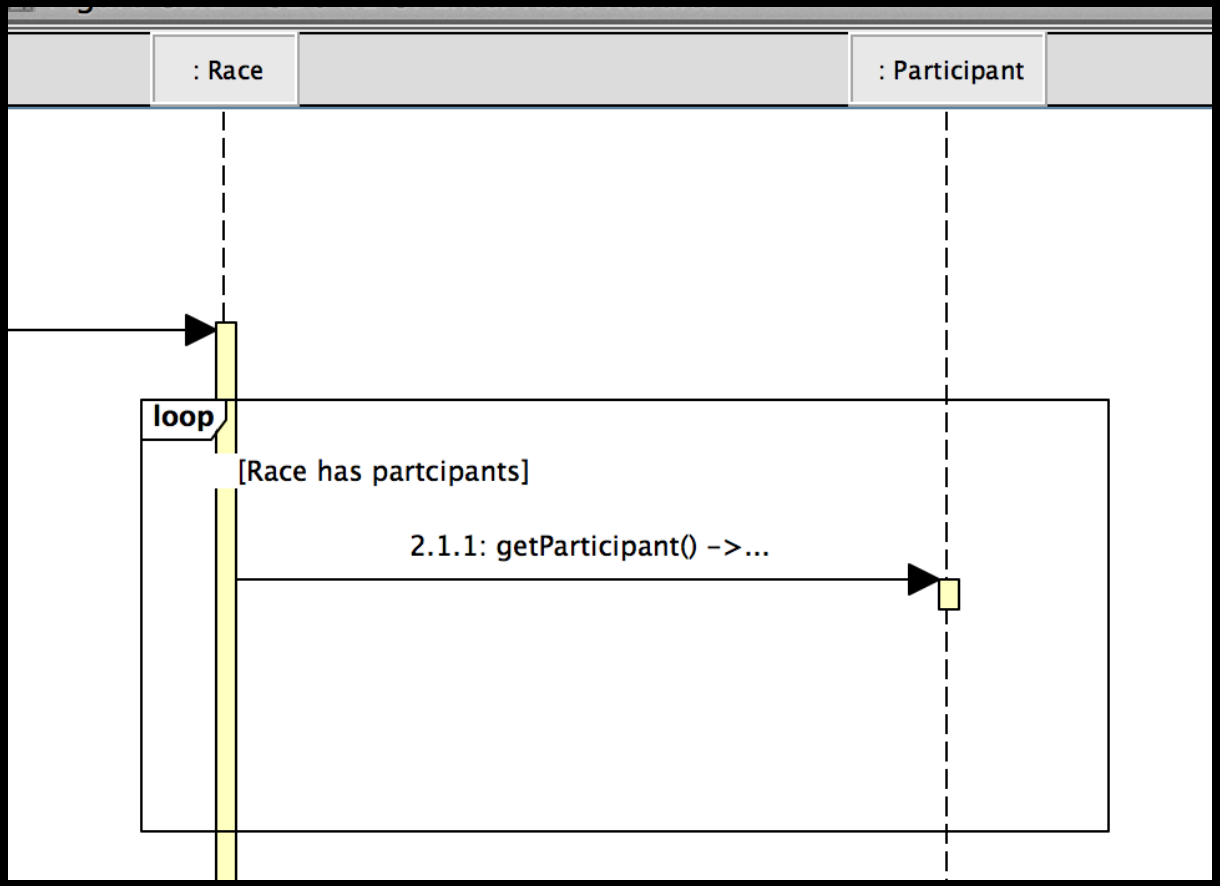
- Synchronous vs. Asynchronous
- Some types of messages flows:
  - Call
  - Send
  - Return



Frames

- Sequence diagrams can include
  - Reference frames
  - Loop frames (or fragments)
  - Alt frames
- Reference blocks allow you to build complex interactions from simpler one
- Loop block allow you to some iteration
- Alt blocks allow you to show conditional messages

Frames


: Race
: Participant
loop
[Race has partcipants]
2.1.1: getParticipant() ->...

Differences between Sequence and Communication Diagrams

- What are the significant difference between communication diagrams and sequence diagrams?
  - Wrt. their syntax
  - Wrt. their semantics
  - Wrt. their use
- Can a sequence diagram be transformed into a communication diagram?
- Can a communication diagram be transformed into a sequence diagrams?
  - Any sticky problems?

Common Uses

- Interaction diagrams model dynamic aspects of a system
  - These aspects may involve interactions of any kind of instance in any view of the system's architecture
  - Interaction diagrams can work within the context of
    - the system as whole
    - a subsystem
    - an operate
    - a class
  - Interactions can also help model instances of use cases → collaborations

Hints and Tips

- No single interaction diagram can capture everything about a system's dynamic aspects
- Don't try to describe all possible scenarios in one diagram; use multiple diagrams
- Keep each diagram focused on a single thought or cohesive set of thoughts
- Use interaction diagrams at different level of abstraction and scope

Hints and Tips

- A well-structured interaction diagram
  - is focused on communicating one aspect of the system's dynamics
  - contains only those elements that are essential to understanding that aspect
  - provides details consistent with its level of abstraction
  - is not so minimalist that it misinforms the reader about semantics that are important

Hints and Tips

- When you draw an interaction diagram
  - Give it a name that communicate its purpose
  - Use a sequence diagram if you want to emphasize time ordering
  - Use a communication diagram if you want to emphasize organization
  - Lay out its elements to minimize line crossings

Hints and Tips

- Use notes and color as visual clues
- Use branching sparingly; its better to use multiple interaction diagrams