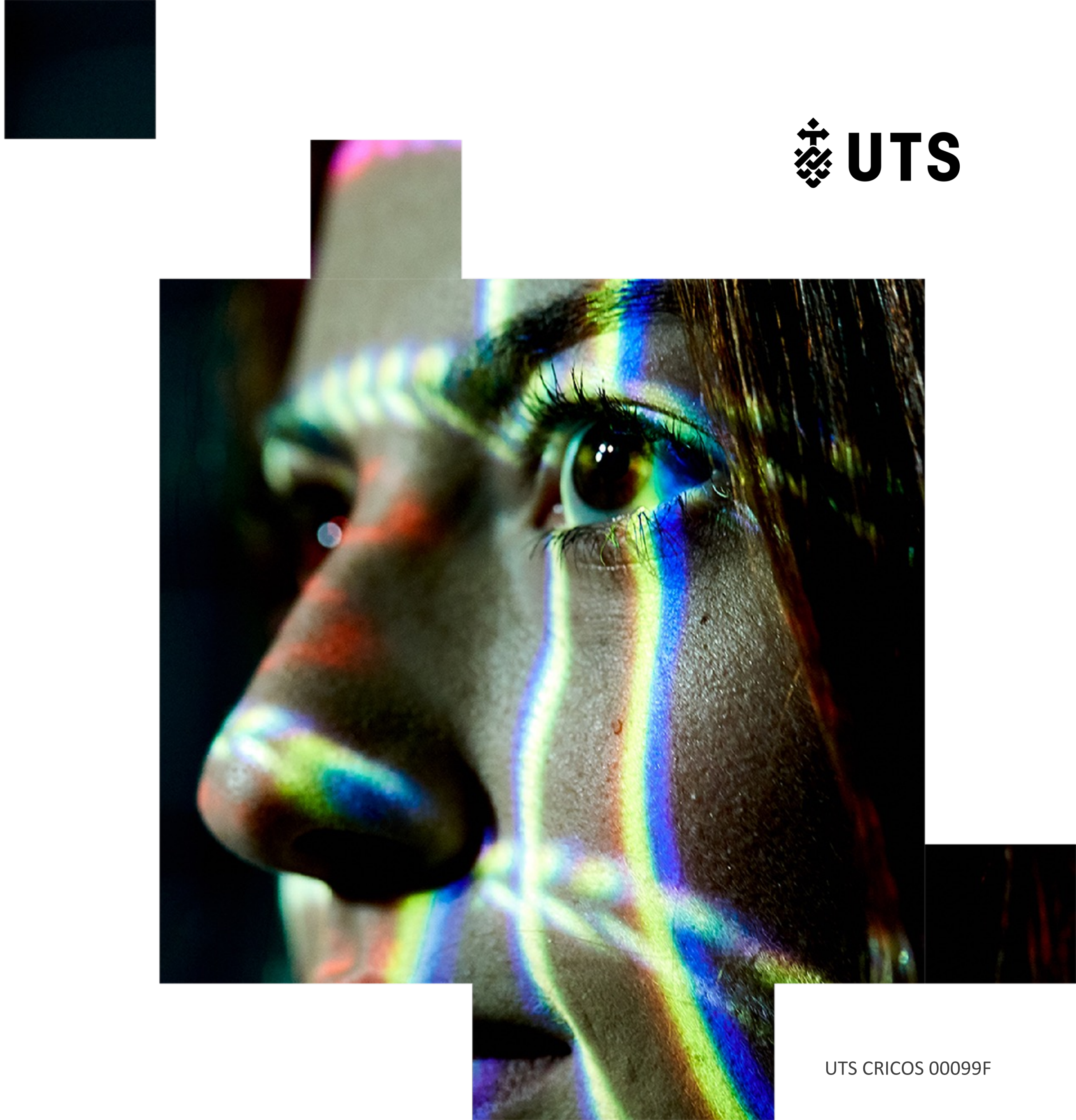
Module 4: Sequence Diagram

SSTC 2022
Module 4 – Lecture 2

Dr. Salvatore Flavio Pileggi

SalvatoreFlavio.Pileggi@uts.edu.au
https://www.uts.edu.au/staff/salvatoreflavio.pileggi

School of Computer Science, Faculty of Engineering and IT
University of Technology Sydney (Australia)



Sequence Diagram

Interaction Modelling

Interaction diagrams

According to the OO approach, a system is modelled as a set of **interacting** objects.

How do objects (or actors) interact with each other (or external systems of interest) to execute the system functionality?

- Interaction Diagrams aim to illustrate **how objects interact**
- They model **how multiple objects collaborate to perform some behavior**.
- UML includes interaction diagrams (Sequence Diagram and Collaboration Diagram)

Interaction Modelling

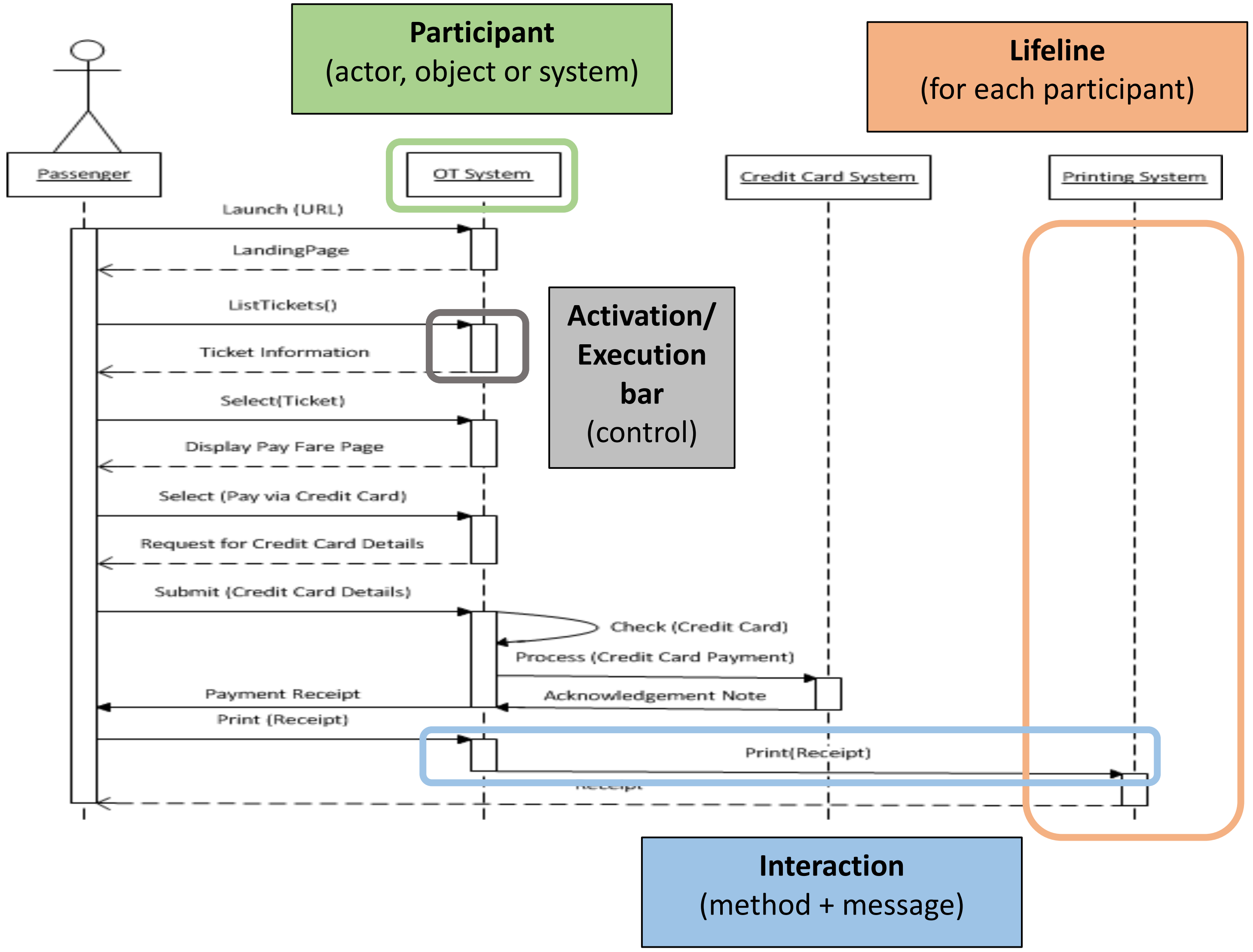
Sequence Diagram

A graphical/visual representation whose main structure is based on:

- **"Lifeline"** (for each **participant** – i.e. involved actor, object or system)
- **Interaction (method + message)** that involves a source and a target object/system

Additionally, the **activation/execution bar** shows the period of time during which an operation is executed along the lifeline

We are able to model a functionality as a controlled sequence of operations involving multiple objects



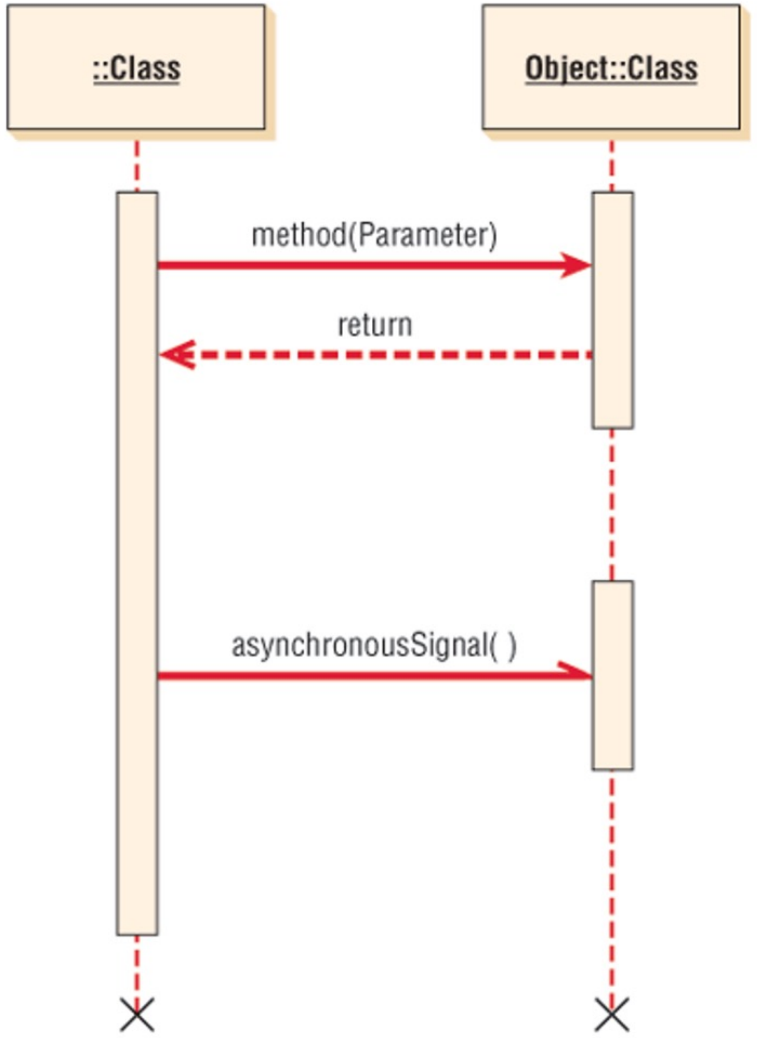
Synchronous and Asynchronous communication

Sequence Diagram

Synchronous vs Asynchronous call

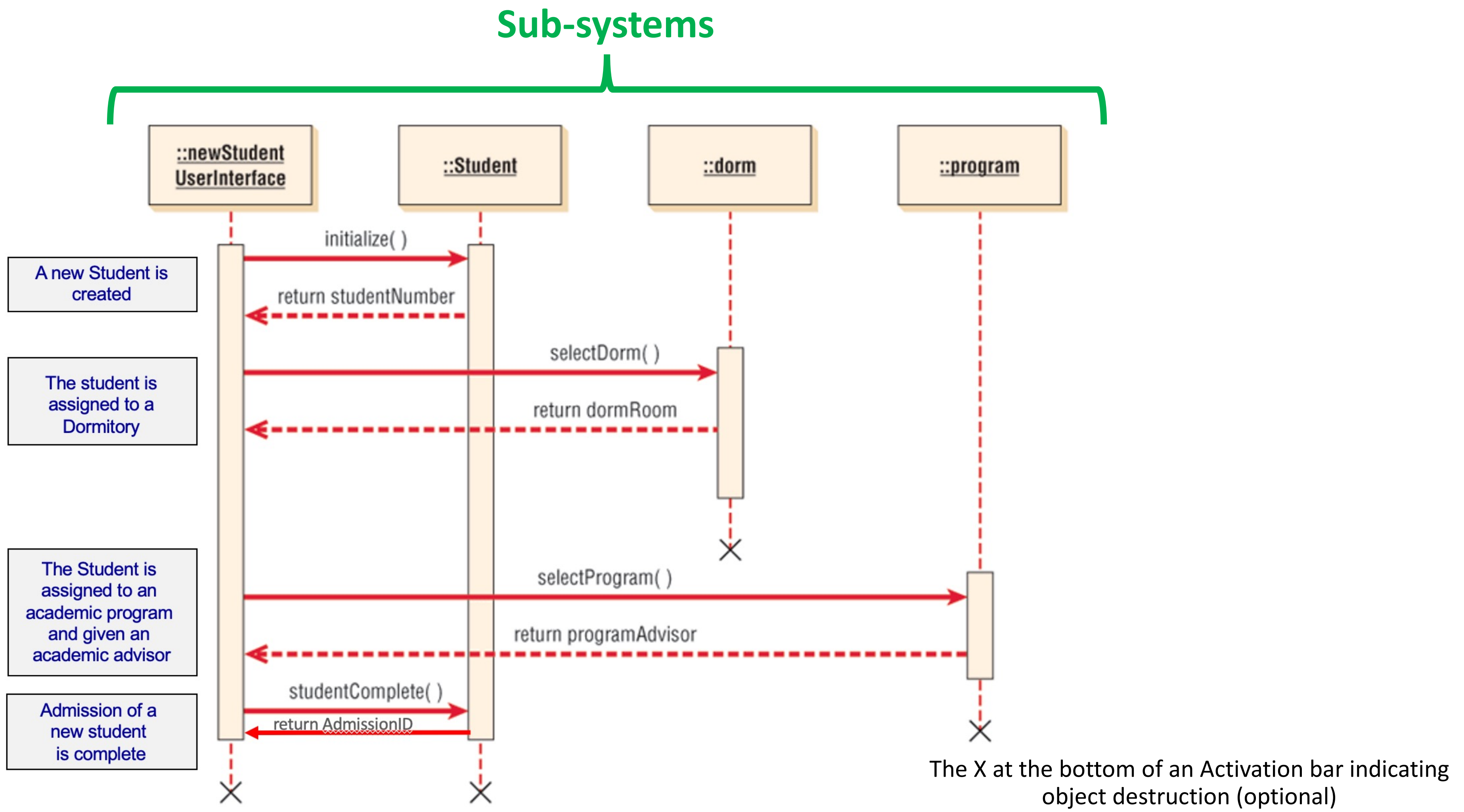
Synchronous vs Asynchronous communication:

- Solid full arrowheads represent **synchronous calls** (the sending class waits for a response). Synchronous calls are normally understood like **functions** (value returning methods).
- Half arrowheads represent **asynchronous calls** (sent without waiting for a returning signal). Asynchronous calls are more like **procedures** (void methods).



Sequence Diagram

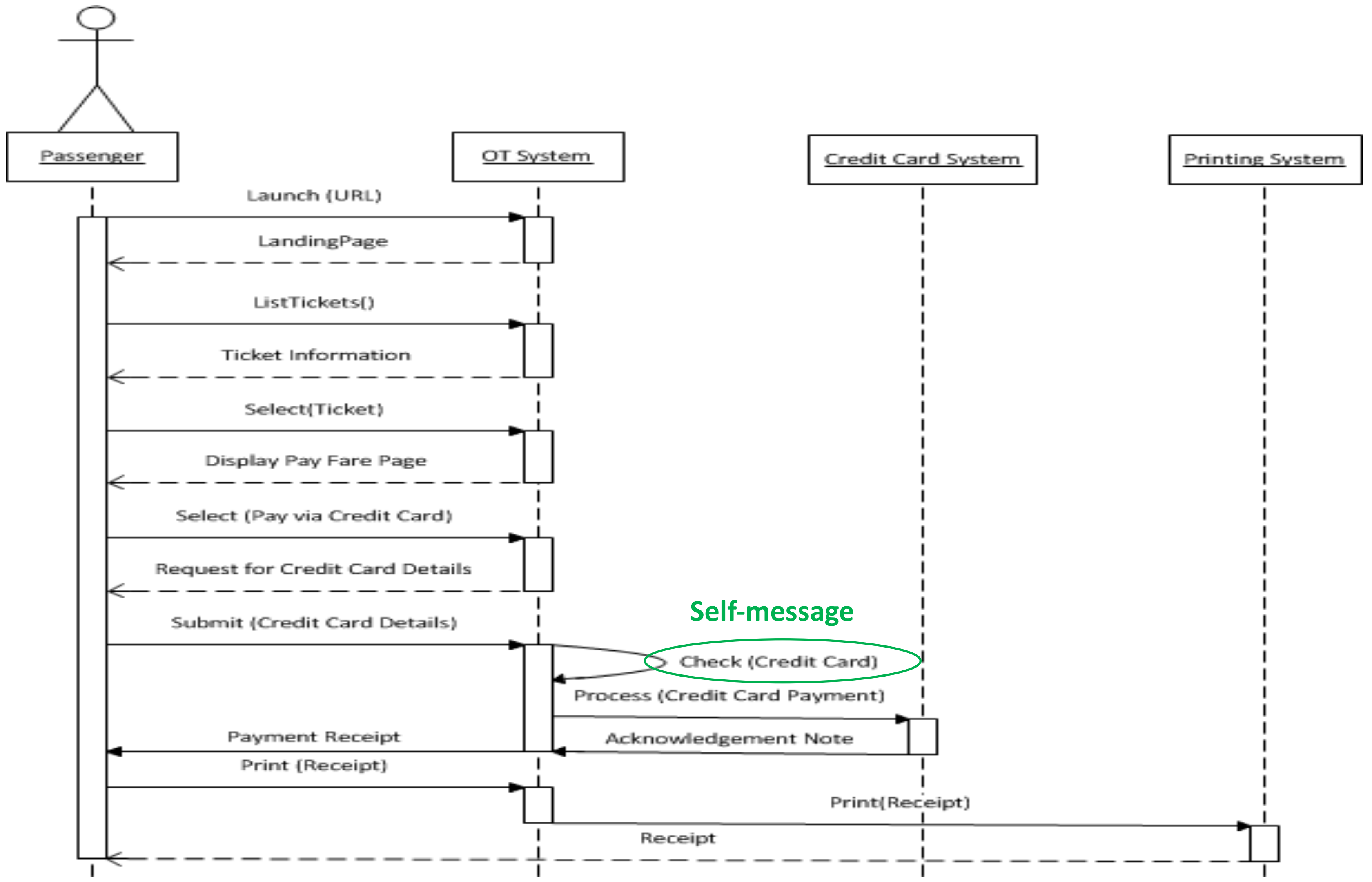
Example



Sub-systems
::newStudent UserInterface
::Student
::dorm
::program
initialize( )
return studentNumber
selectDorm( )
return dormRoom
selectProgram( )
return programAdvisor
studentComplete( )
return AdmissionID
A new Student is created
The student is assigned to a Dormitory
The Student is assigned to an academic program and given an academic advisor
Admission of a new student is complete
The X at the bottom of an Activation bar indicating object destruction (optional)

Sequence Diagram

Self-messaging



Conditional and looping constructs

Sequence Diagram

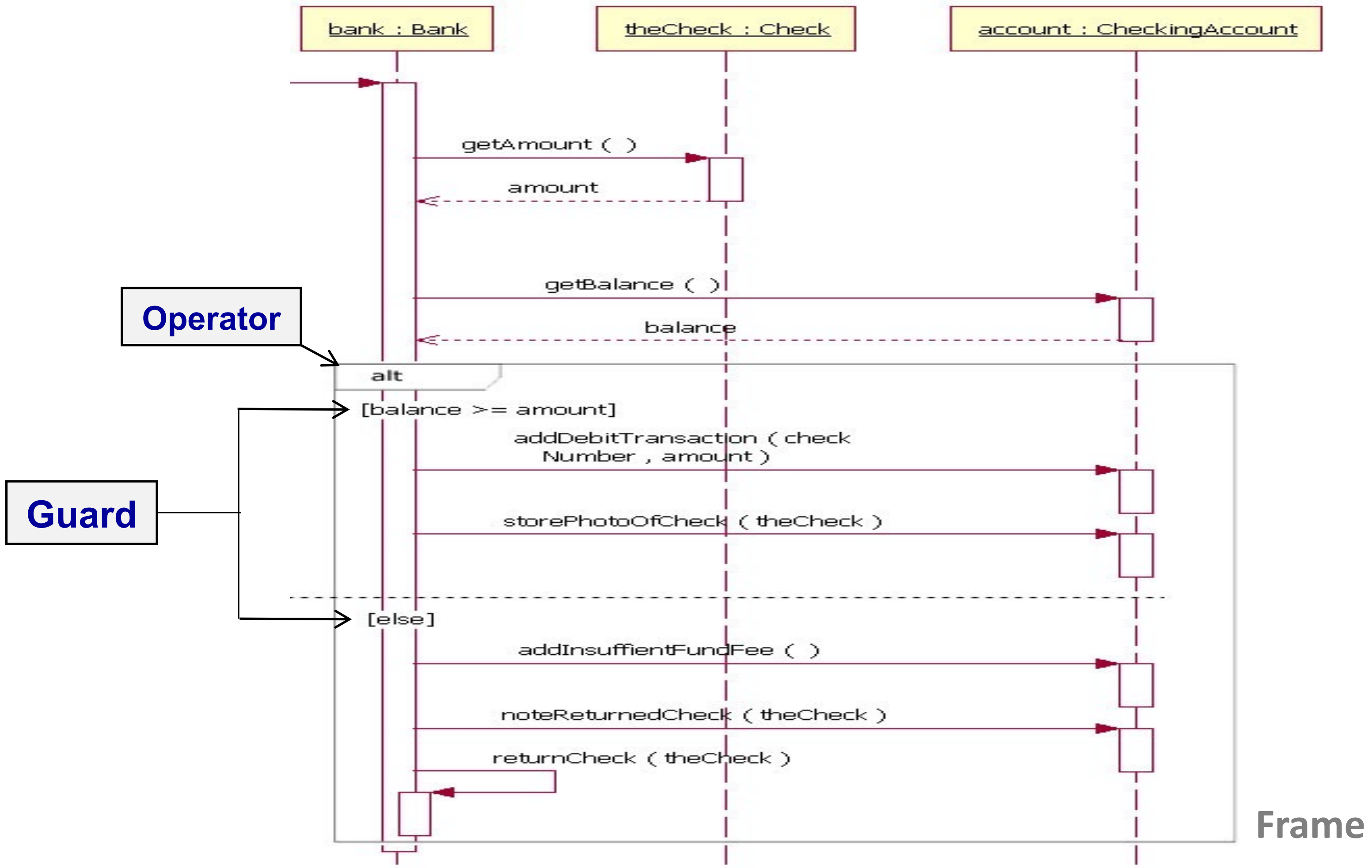
Frames

- To support **conditional and looping constructs**, in Sequence Diagrams we can add **frames.**
- Frames are **regions or fragments of the diagrams**; they have an **operator or label** (such as loop) and a **guard** (conditional clause).
- Common frame operators:

alt: for conditional messages (for example, if condition)
loop: to implement a looping or iterative message
opt: optional fragment that executes if guard is true

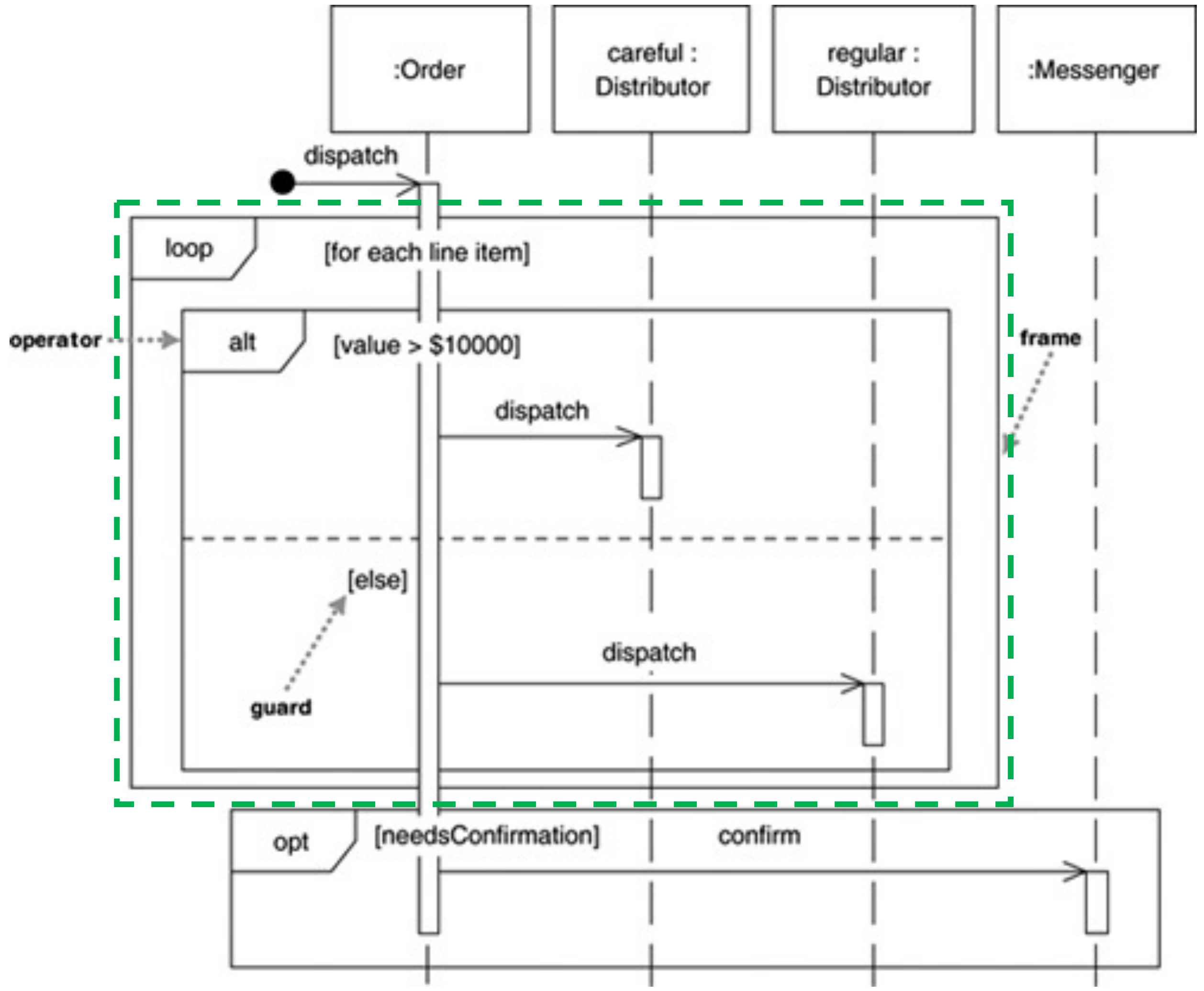
Sequence Diagram

Conditional Message (alt)



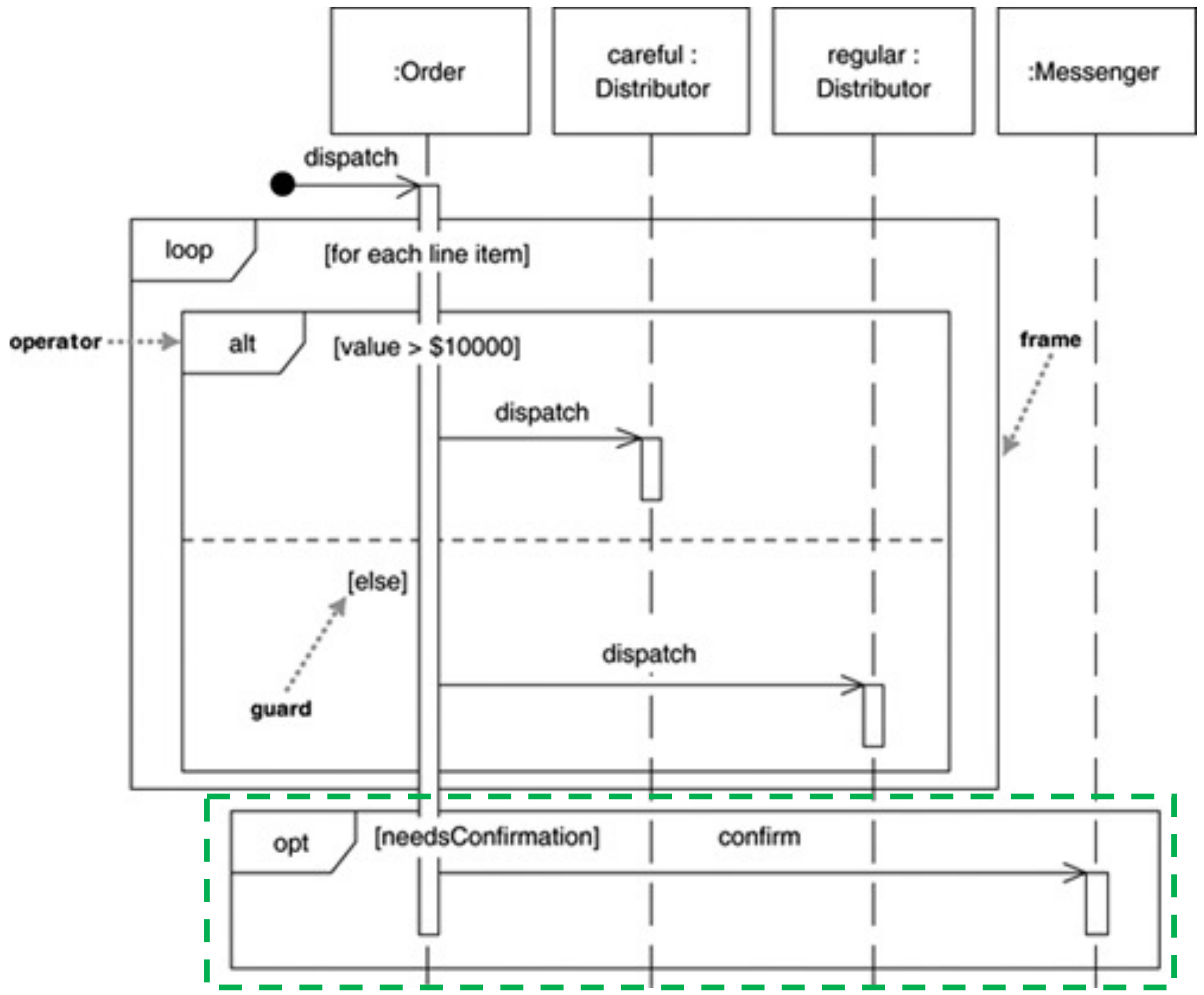
Sequence Diagram

Loops (loop)



Sequence Diagram

Optional Message (opt)



Thank You!