UNIVERSITY OF Southampton
School of Electronics and Computer Science

6. Activity and Sequence Diagrams Revisited

Thai Son Hoang

ECS, University of Southampton, U.K.

COMP1216 - Software Modelling and Design
19th February 2019

Frontmatter

- Most slides are from Bruegge and Dutoit [2014] and Lethbridge and Laganière [2005].
- They do NOT have University of Southampton logo.
- For example the next slide.

Reading for this Lecture

- Chapter 2 and Chapter 5 of Bruegge and Dutoit [2014].
- Chapter 8 of Lethbridge and Laganière [2005].

Object-Oriented Software Engineering

Practical Software Development using UML and Java

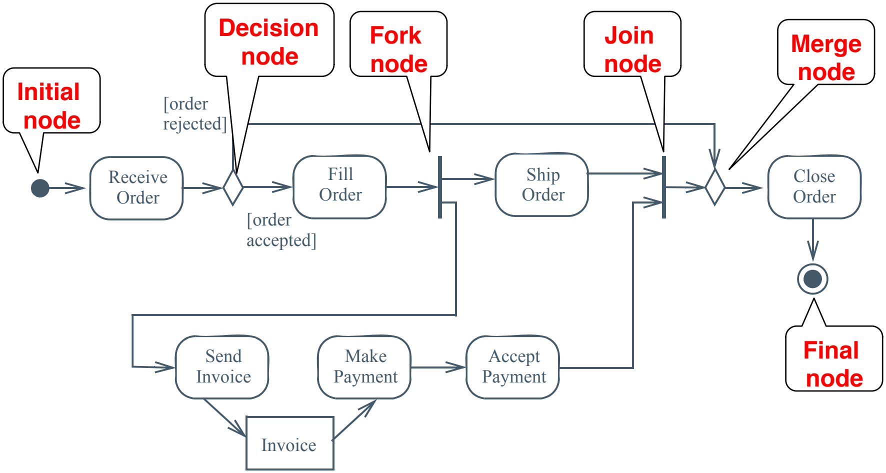
Chapter 8:

Modelling Interactions and Behaviour

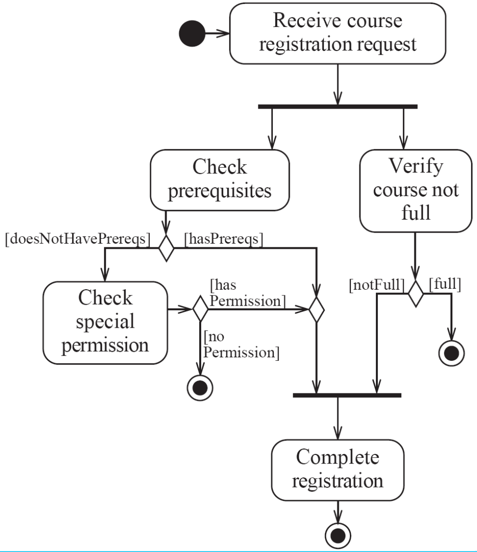
Objectives

- Activity diagrams revisited
- Sequence diagrams revisited

Activity Diagram Example

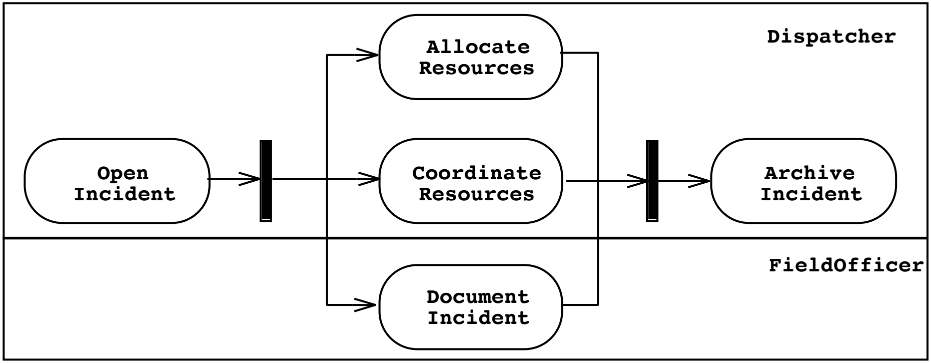


Activity diagrams – an example



Activity Diagrams: Grouping of Activities

- Activities may be grouped into swimlanes to denote the object or subsystem that implements the activities.

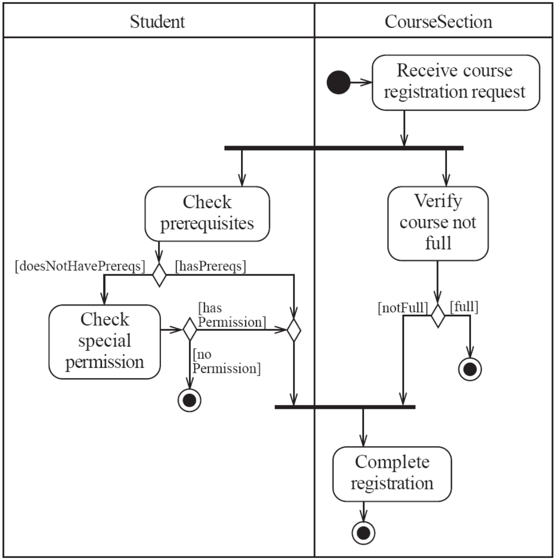


Swimlanes

Activity diagrams are most often associated with several classes.

- The partition of activities among the existing classes can be explicitly shown using *swimlanes*.

Activity diagrams – an example with swimlanes



An Example

- Flow of events in "Get SeatPosition" use case :

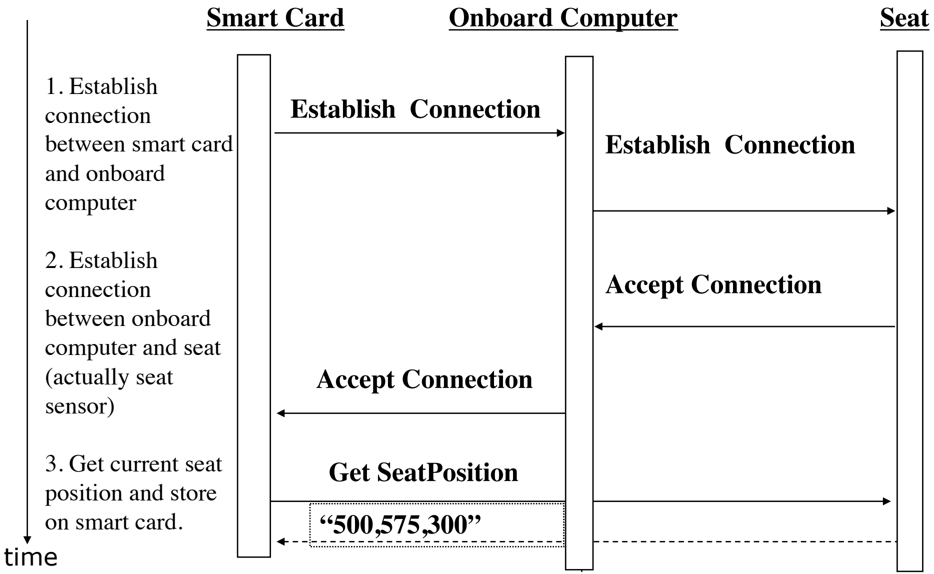
  1. Establish connection between smart card and onboard computer

  2. Establish connection between onboard computer and sensor for seat

  3. Get current seat position and store on smart card

- Where are the objects?

Sequence Diagram for "Get SeatPosition"



1. Establish connection between smart card and onboard computer

2. Establish connection between onboard computer and seat (actually seat sensor)

3. Get current seat position and store on smart card.

There are different types of Objects

- Entity Objects
  - Represent the persistent information tracked by the system (Application domain objects, also called "Business objects")
- Boundary Objects
  - Represent the interaction between the user and the system
- Control Objects
  - Represent the control tasks performed by the system.

Heuristics for Sequence Diagrams

- Layout:
  1st column: Should be the actor of the use case
  2nd column: Should be a boundary object
  3rd column: Should be the control object that manages the rest of the use case

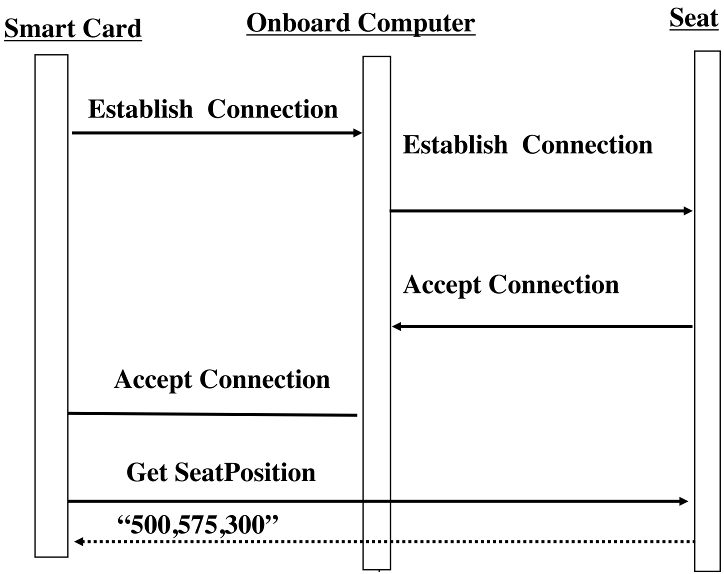
- Creation of objects:
  - Create control objects at beginning of event flow
  - The control objects create the boundary objects
- Access of objects:
  - Entity objects can be accessed by control and boundary objects
  - Entity objects should not access boundary or control objects.

Is this a good Sequence Diagram?

The first column is not an actor

It is not clear where the boundary object is

It is not clear where the control object is

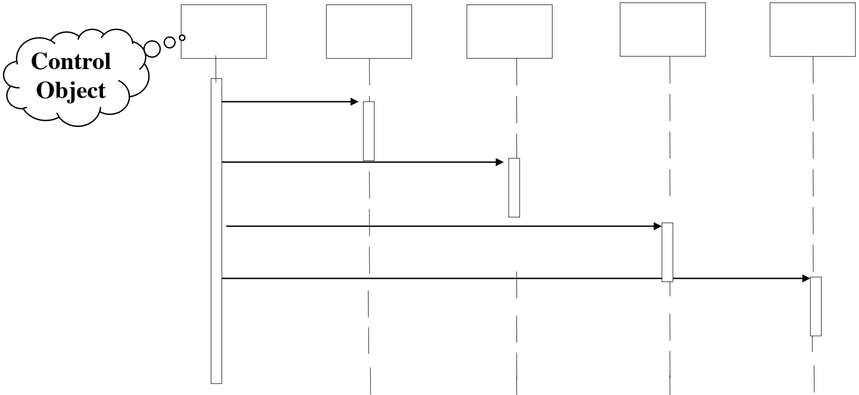


What else can we get out of Sequence Diagrams?

- Sequence diagrams are derived from use cases
- The structure of the sequence diagram helps us to determine how decentralized the system is
- We distinguish two structures for sequence diagrams
  - Fork Diagrams and Stair Diagrams (Ivar Jacobsen)

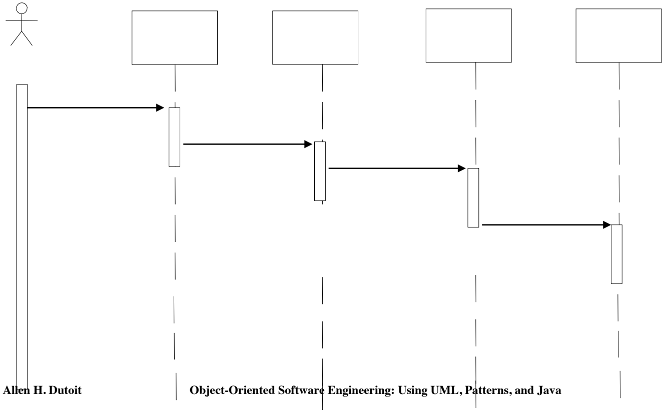
Fork Diagram

- The dynamic behavior is placed in a single object, usually a control object
  - It knows all the other objects and often uses them for direct questions and commands



Stair Diagram

- The dynamic behavior is distributed. Each object delegates responsibility to other objects
    - Each object knows only a few of the other objects and knows which objects can help with a specific behavior



Fork or Stair?

- Object-oriented supporters claim that the stair structure is better
- Modeling Advice:
  - Choose the stair - a decentralized control structure - if
    - The operations have a strong connection
    - The operations will always be performed in the same order
  - Choose the fork - a centralized control structure - if
    - The operations can change order
    - New operations are expected to be added as a result of new requirements.

A 2BWatch

A 2BWatch has the following component

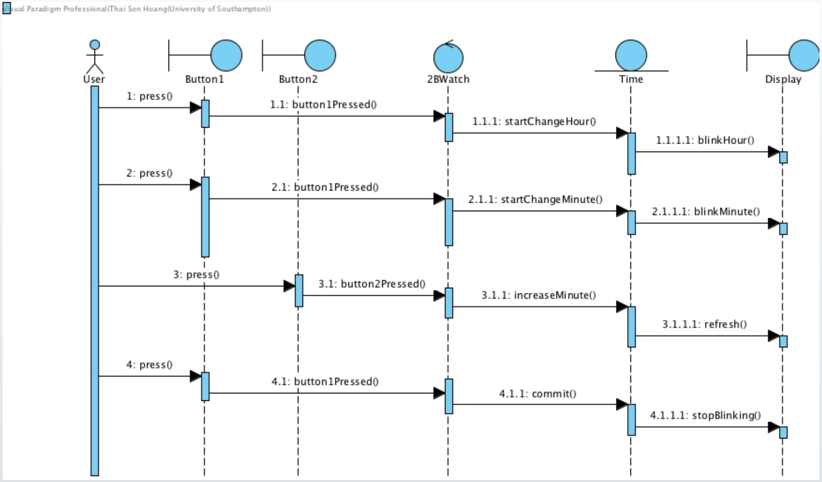
- Two buttons:
  - Button 1: Cycle through the following modes: Normal, Adjust Hour, Adjust Minute
  - Button 2 (During Adjust Hour/Adjust Minute modes) Increase Hour/Minute accordingly.
- A display to show the time. During Adjust Hour/Adjust Minute modes, blink the hour, minute accordingly.

Sequence Diagrams

- Sequence diagram to increase the minute by 1
- Extend the sequence diagram to increase the minute by 5.

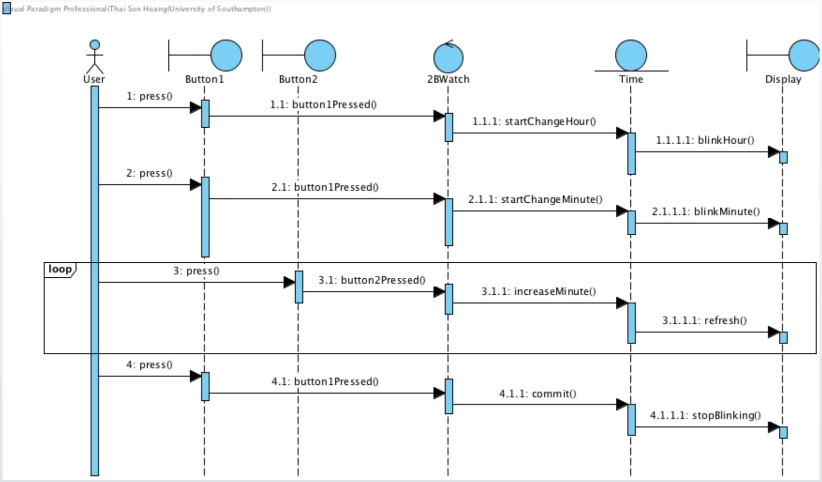
A 2BWatch

Increase the Minute by 1



A 2BWatch

Increase the Minute by 1



Further Reading

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Bernd Bruegge and Allen H. Dutoit. *Object-Oriented Software Engineering - Using UML, Patterns and Java*.
  Pearson, 3rd edition, 2014.
  Pearson International Edition (Chapter 2 and Chapter 5)
- Timothy C. Lethbridge and Robert Laganière. *Object-Oriented Software Engineering: Practical Software Development Using UML and Java*.
  McGraw-Hill, 2nd edition, 2005 (Chapter 8)