Object Oriented Analysis and Design

Sequence Diagram

Syllabus

Preliminary
Purpose
Component
Symbol
Term
SD types
Example
Case Study

Preliminary

Sequence diagrams describe the interactions between objects in and around the system (including users, displays / forms) in the form of messages depicted against time.

Sequence diagrams consist of vertical dimensions (time) and horizontal dimensions (related objects).

Sequence diagrams are commonly used to describe a scenario or series of steps that are carried out in response to an event to produce a certain output . Starting from what triggers these activities, what processes and changes occur internally and what outputs are produced.

This diagram is specifically associated with use case diagrams and shows step by step what should happen to produce something in the use case

Purpose

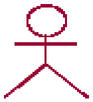
- Used to show interactions between objects in a sequential order.
- The main objective is to define the sequence of events that can produce the desired output
- Similar to an activity diagram
  - Describe the flow of events for an activity
  - More details in describing the flow of data, including data or behavior sent or received
  - However, it is not able to explain the details of an algorithm (loop, branching)

Components

- Actor
- Interface (Boundary)
- Reading process (Control)
- Table name (Entity)

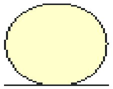
Symbols

a. An Actor



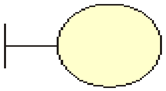
Menggambarkan orang yang sedang berinteraksi dengan sistem

b. Entity Class



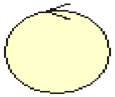
Menggambarkan hubungan kegiatan yang akan dilakukan

c. Boundary Class



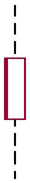
Mengganbarkan sebuah penggambaran dari form

d. Control Class



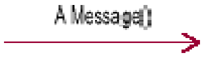
Menggambarkan penghubung antara boundary dengan tabel

e. A focus Of Control & A life line



Menggambarkan tempat mulai dan berakhirnya sebuah massage

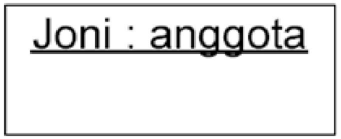
f. A massage



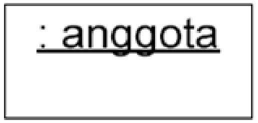
Menggambarkan Pengiriman Pesan

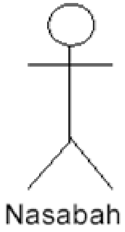
Symbols

- Participant: an object or entity that acts in a sequence diagram
- Message: communication between participant objects
- There are 2 types of lines, namely vertical and horizontal
    - Vertical: time → advances by time
    - Horizontal: which object is in action



- Object / Class name
    - name is optional
    - Object boxes are underlined
    - Unnamed objects are called anonymous objects
    - Boxes in the form of actors can also be drawn with stick figures





Glossary of Terms

- Participant
  - Objects associated with a sequence of processes
- Lifeline
  - Describe the life cycle of an object
- Activation
  - A point in time where an object begins to participate in a sequence
  - Marked by a bar

Time

  - An important element in sequence diagrams
  - The context is **order, not duration**
- Return
  - A result returns an operation
  - The operation returns a result, but may be unwritten

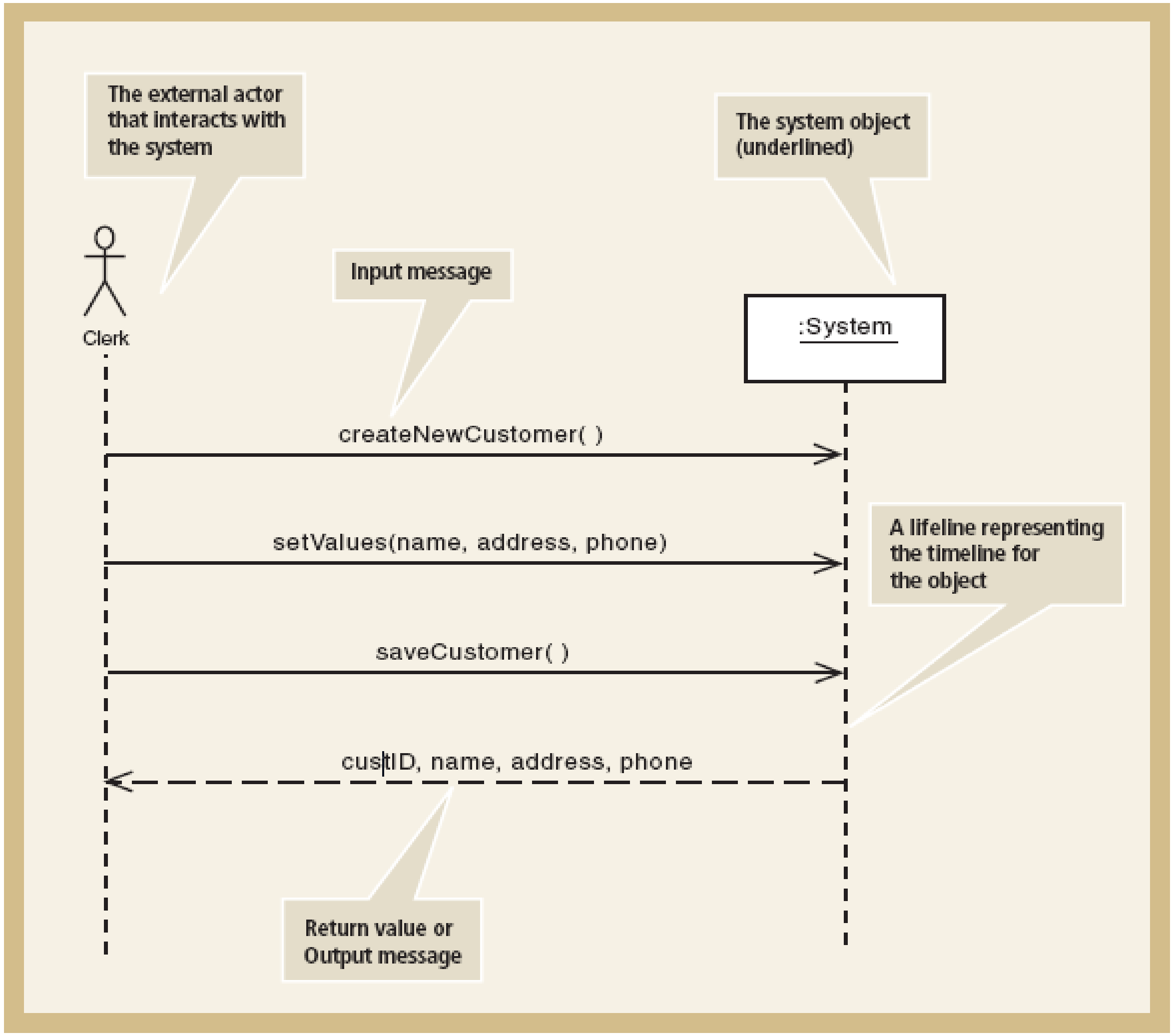
Sequence Diagram

•Sequence Diagrams describe the sequence of interactions between objects when one Use Case is executed, so that the Sequence Diagram (preferably) is as many as the Use Case.

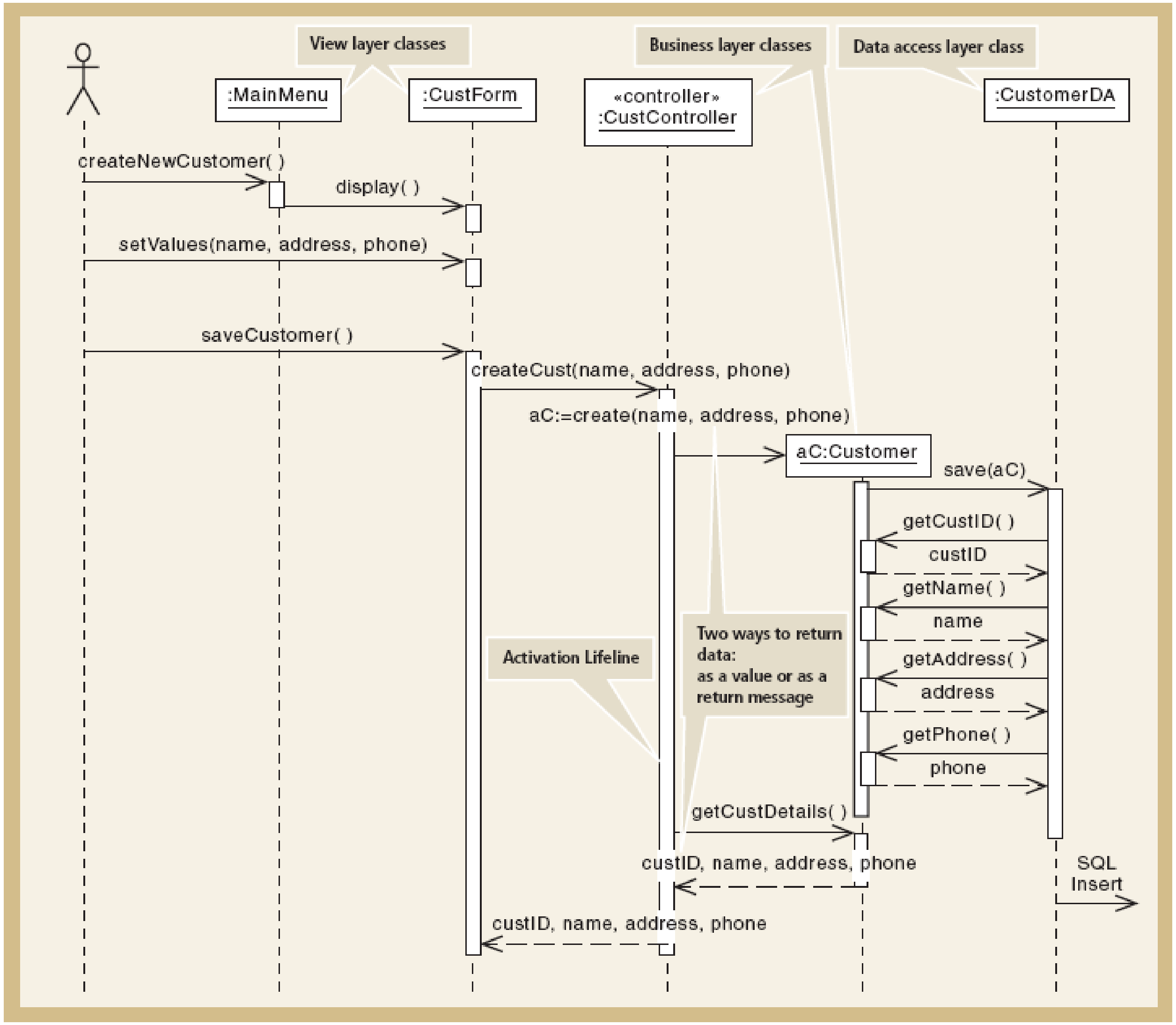
•In making this Sequence Diagram using the MVC (Model-View-Controller) programming method or in other terms Model = Entity, View = Boundary, and Controller = Control.

•View / Boundary is a class that interacts directly with Actor. Controller / Control is an intermediate interaction class between View / Boundary and Mode / Entity. while Model / Entity is a class that stores data.

Example

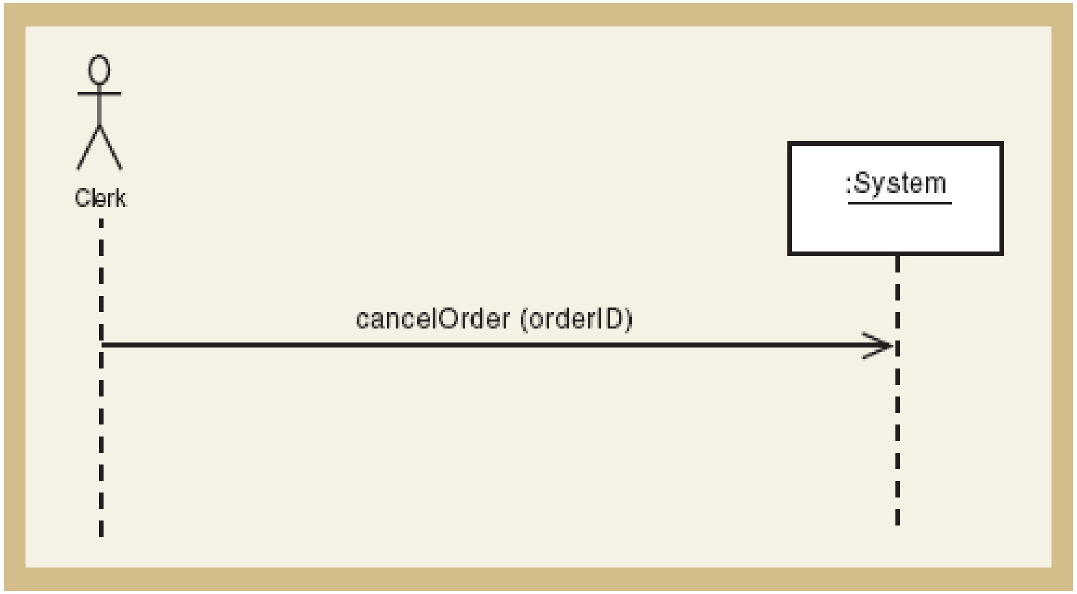

The external actor that interacts with the system
The system object (underlined)
Input message
Clerk
:System
createNewCustomer( )
setValues(name, address, phone)
A lifeline representing the timeline for the object
saveCustomer( )
custID, name, address, phone
Return value or Output message

Example (continued)

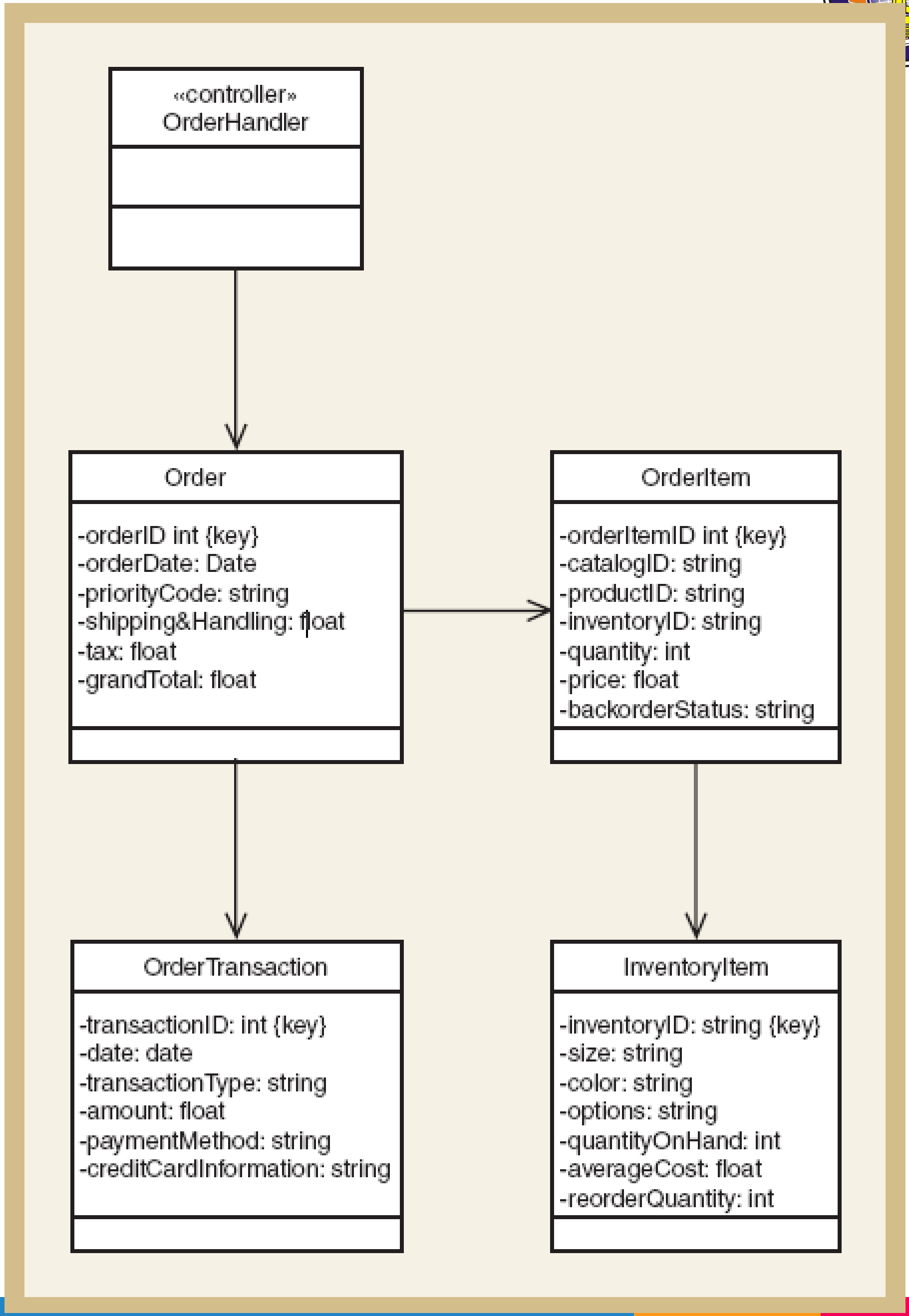


Relationship Between Class Diagram and Sequence Diagram

- Example: Order Cancellation
    - Broadly speaking, order cancellations are performed by staff / clerks, and interact with the system.



- Class Diagram
  - This is a class diagram showing the classes involved in the Cancel Order process.
  - There may be other classes not shown here that are not related to the Cancel Order process.



«controller»
OrderHandler

Order

-orderID int {key}
-orderDate: Date
-priorityCode: string
-shipping&Handling: float
-tax: float
-grandTotal: float

OrderItem

-orderItemID int {key}
-catalogID: string
-productID: string
-inventoryID: string
-quantity: int
-price: float
-backorderStatus: string

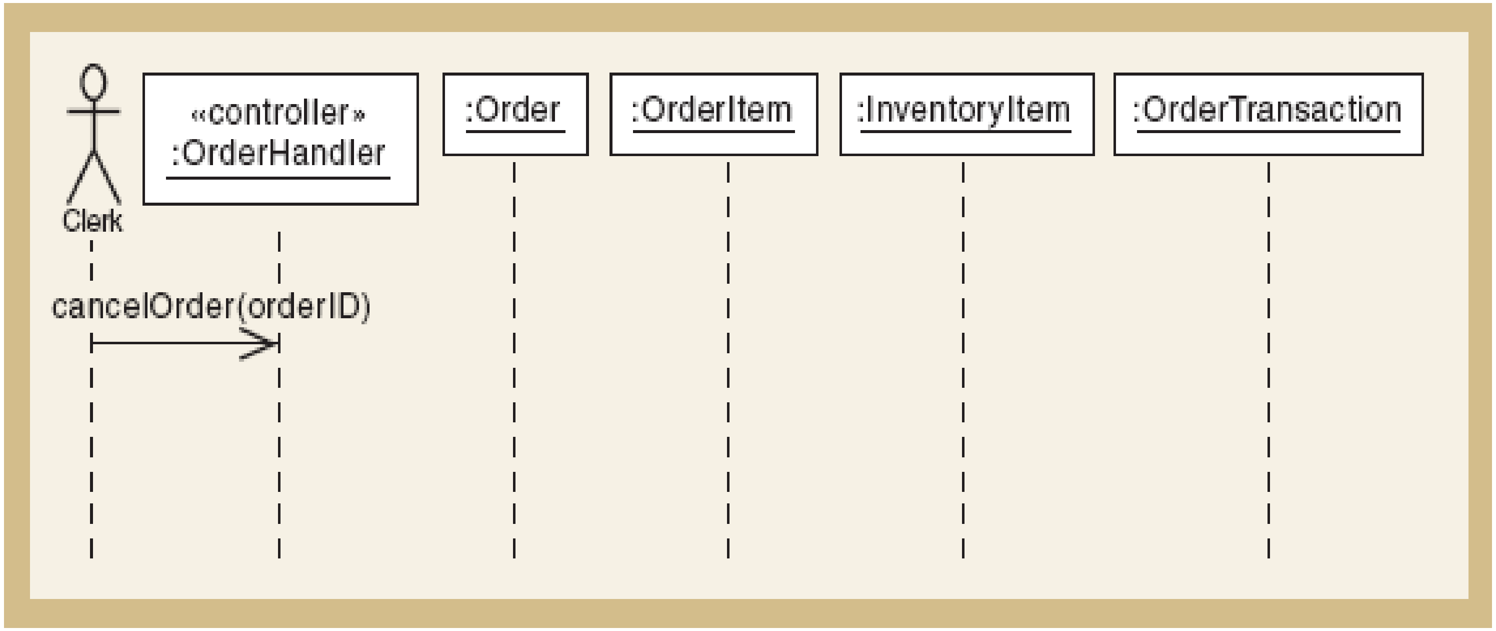
OrderTransaction

-transactionID: int {key}
-date: date
-transactionType: string
-amount: float
-paymentMethod: string
-creditCardInformation: string

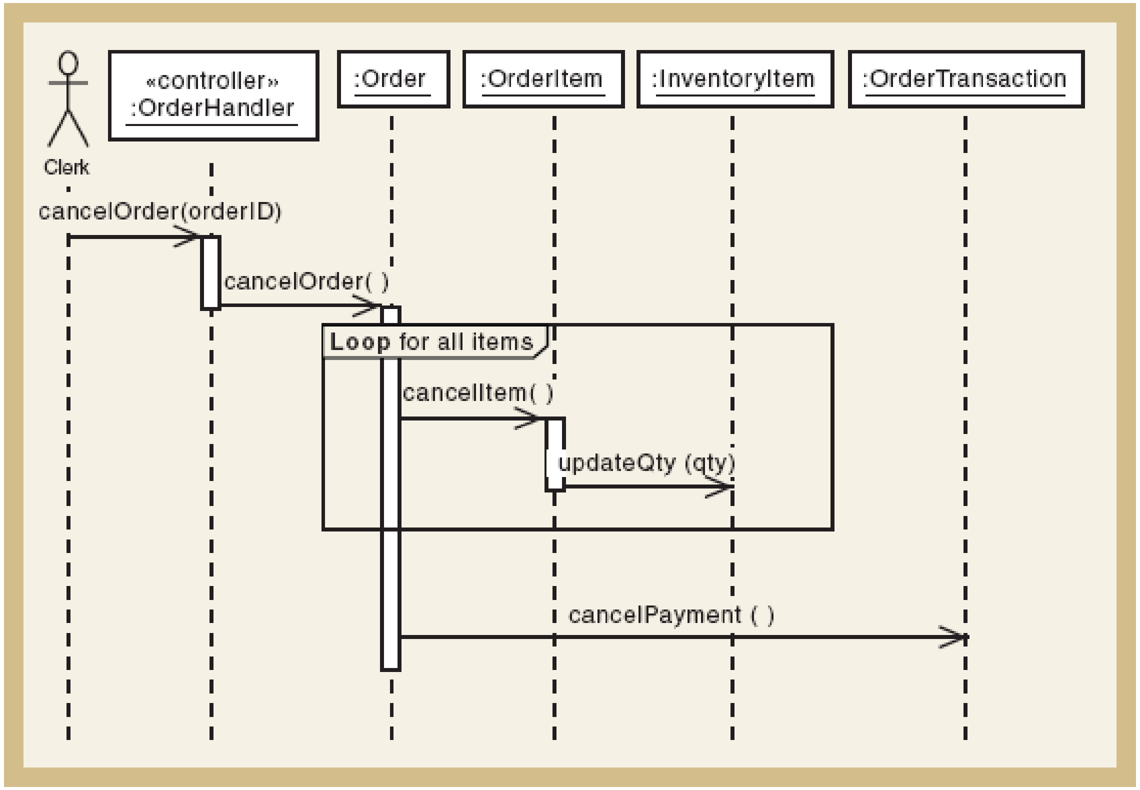
InventoryItem

-inventoryID: string {key}
-size: string
-color: string
-options: string
-quantityOnHand: int
-averageCost: float
-reorderQuantity: int

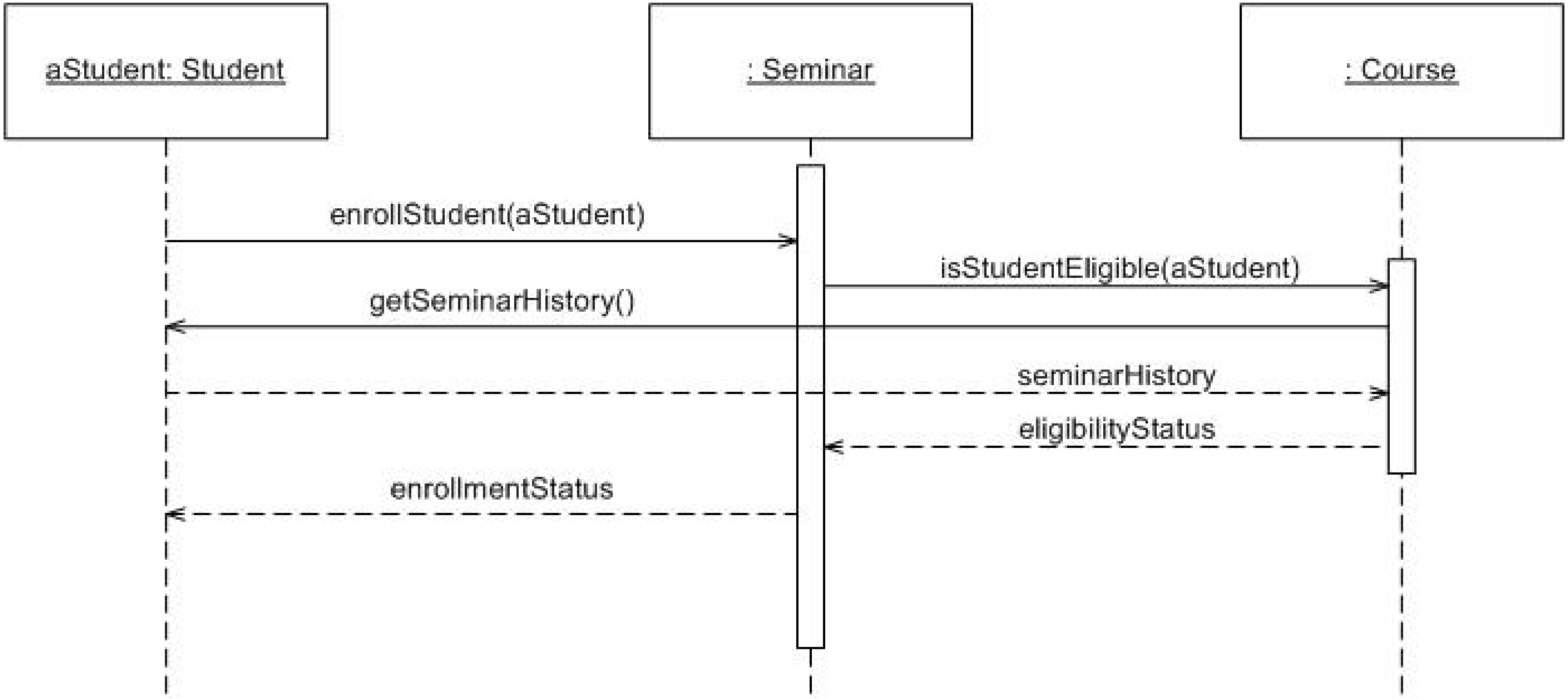
- Potential Objects
  - From the class diagram, we break down to be like this:



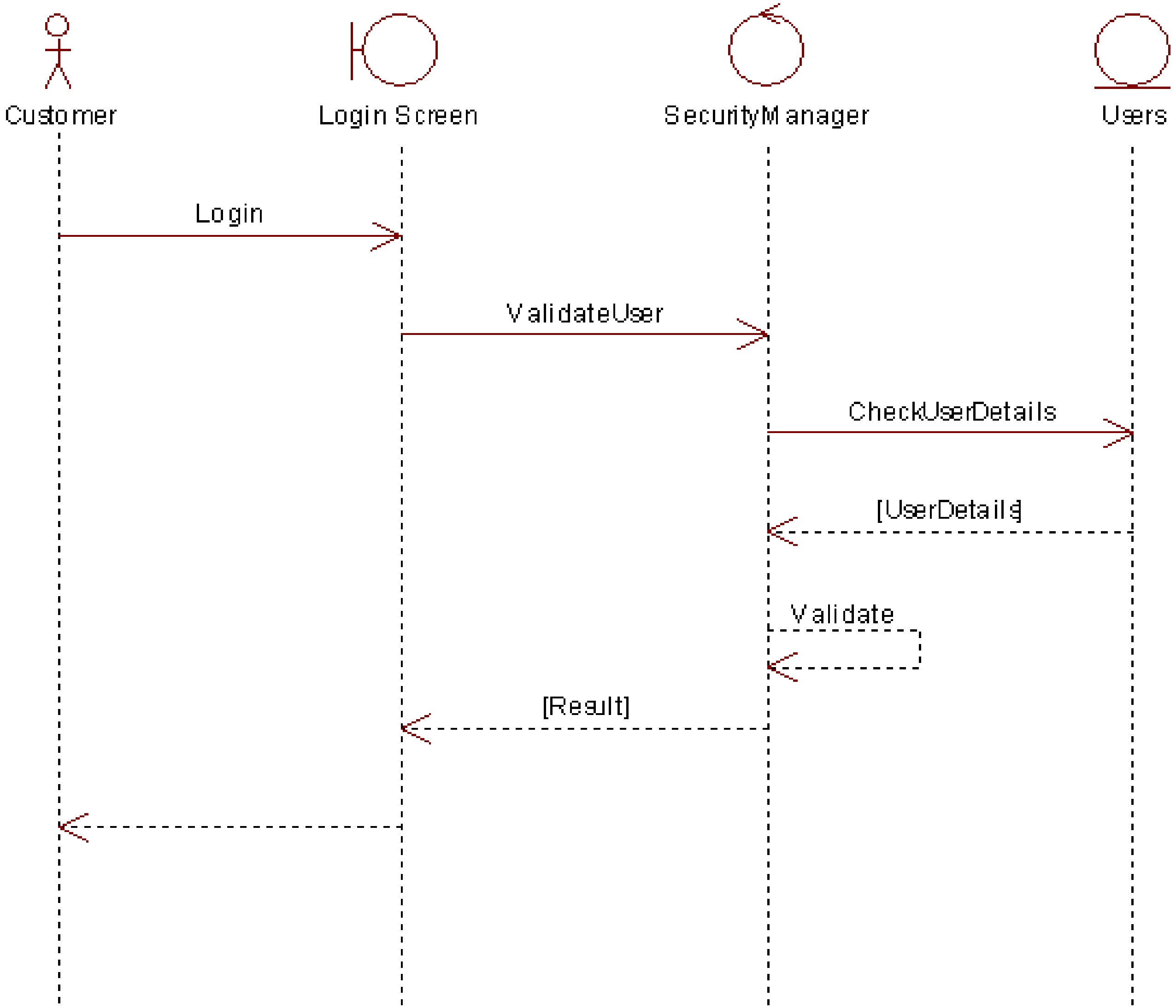
- Then we compile a diagram that illustrates the flow of method calls in each of these classes



Example 1

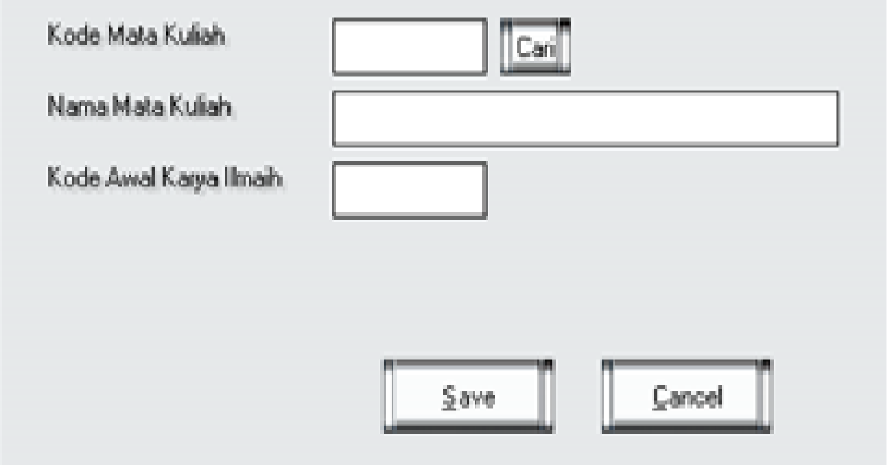

aStudent: Student
: Seminar
: Course
enrollStudent(aStudent)
isStudentEligible(aStudent)
getSeminarHistory()
seminarHistory
eligibilityStatus
enrollmentStatus

Example 2

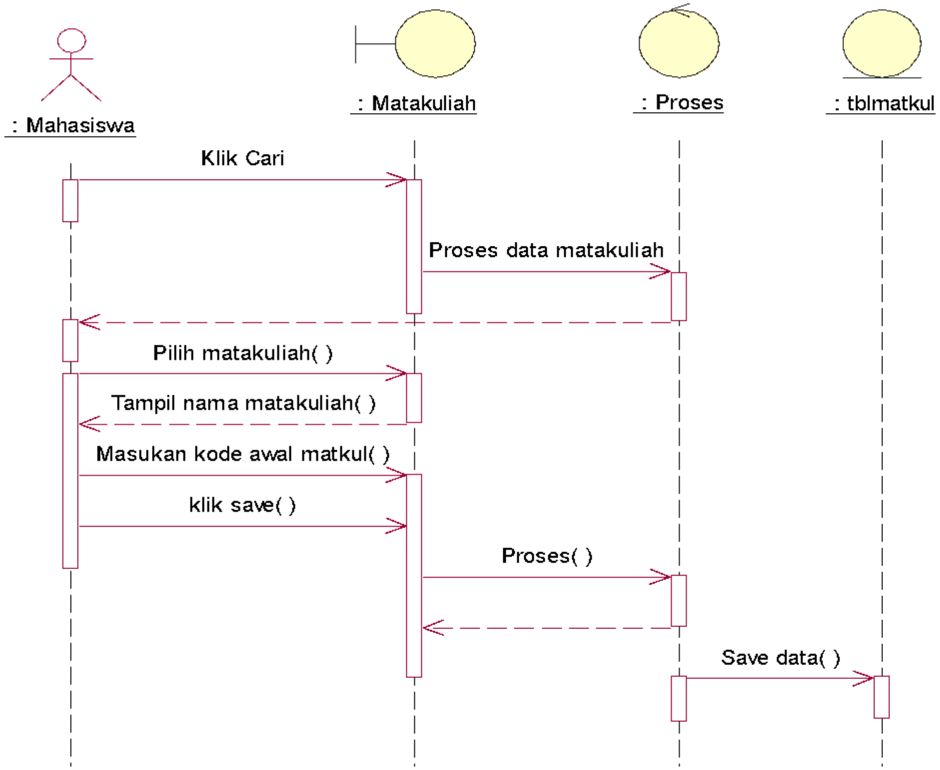

Customer
Login Screen
SecurityManager
Users
Login
ValidateUser
CheckUserDetails
[UserDetails]
Validate
[Result]

Example 3

- Observe the following form. This form has a search feature, update data, and add new data.
- How to draw sequence diagram based on this kind of form?

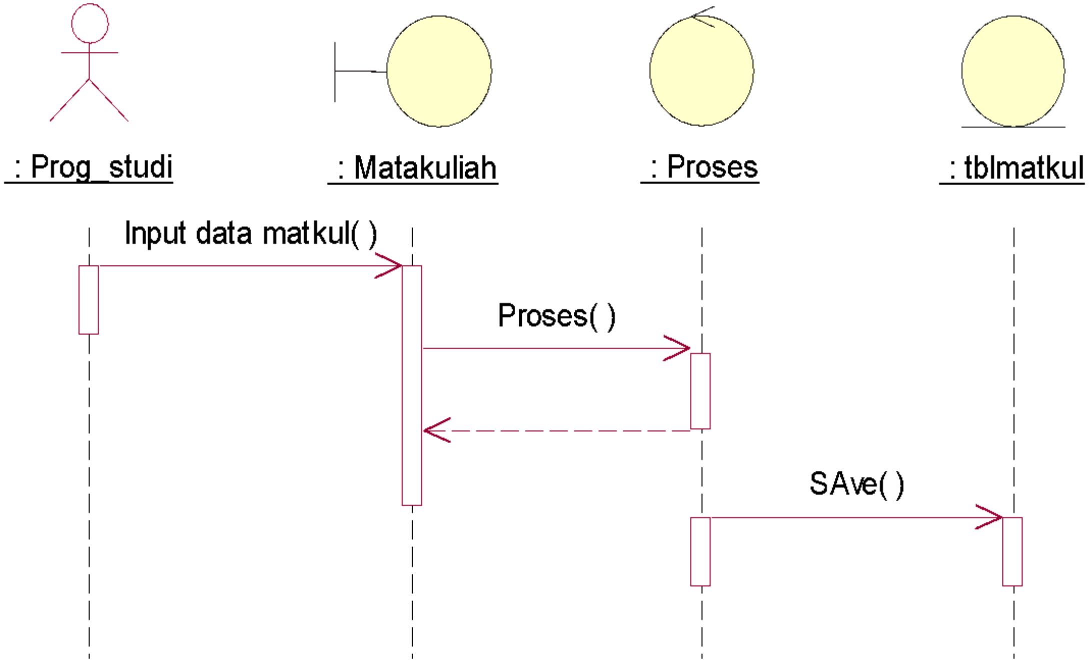


•The following is the Sequence Diagram for the previous form, which is the process of searching for the course code, and updating the data.

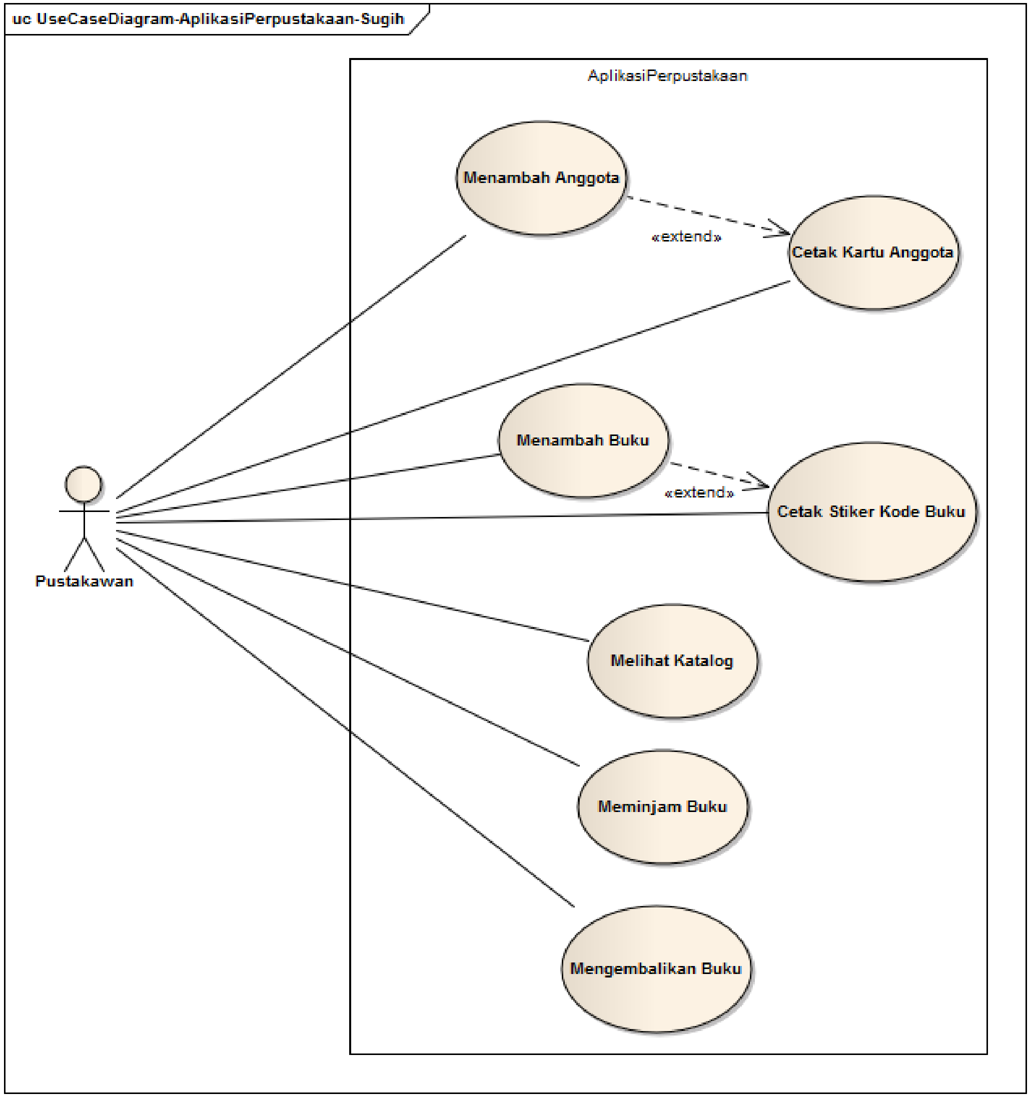


- It is assumed that in this system there are classes:
  - Courses
  - Process
  - tblmatkul

- And this is a Sequence Diagram for the process of adding new MataKuliah data.

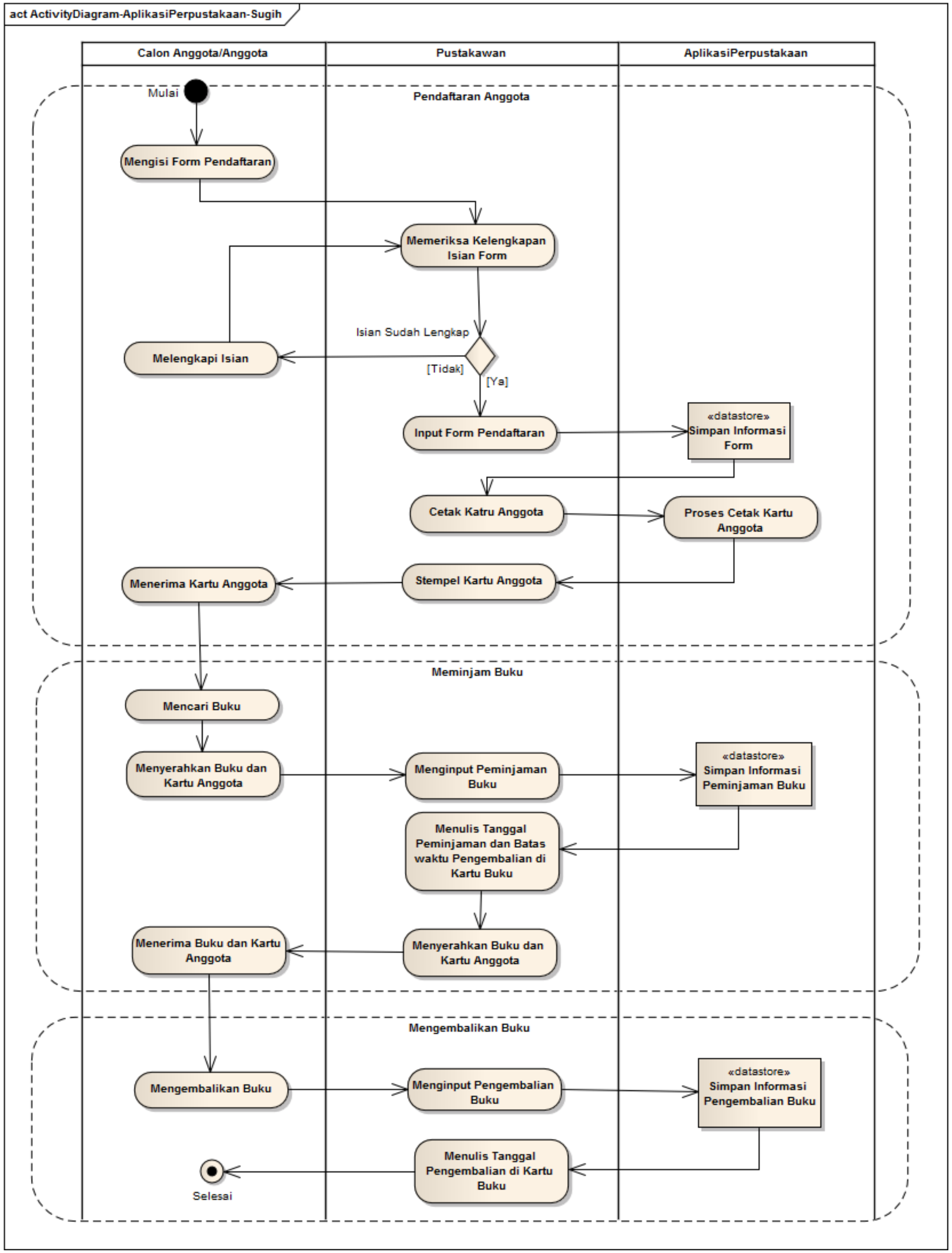


Case Study: Library System



- The picture above illustrates that the application has one actor / user, namely librarian and 7 use cases. This explains that in the application, librarians can Add Members, Print Member Cards, Add Books, Print Book Code Stickers, View Catalogs, Borrow Books, and Return Books.
- There may be confusion as to why it is the librarian who borrows and returns the book, not a library member.
- If we look at the Business Process or Activity Diagram , it can be seen that those who interact directly with the application are Librarians, not members. Members borrow and return books to the librarian, then the librarian who inputs into the application.

The Activity Diagram



act ActivityDiagram-AplikasiPerpustakaan-Sugih

Calon Anggota/Anggota

Pustakawan

AplikasiPerpustakaan

Pendaftaran Anggota

Mulai

Mengisi Form Pendaftaran

Memeriksa Kelengkapan Isian Form

Isian Sudah Lengkap

Melengkapi Isian

[Tidak]

[Ya]

Input Form Pendaftaran

«datastore» Simpan Informasi Form

Cetak Katru Anggota

Proses Cetak Kartu Anggota

Menerima Kartu Anggota

Stempel Kartu Anggota

Meminjam Buku

Mencari Buku

Menyerahkan Buku dan Kartu Anggota

Menginput Peminjaman Buku

«datastore» Simpan Informasi Peminjaman Buku

Menulis Tanggal Peminjaman dan Batas waktu Pengembalian di Kartu Buku

Menerima Buku dan Kartu Anggota

Menyerahkan Buku dan Kartu Anggota

Mengembalikan Buku

Mengembalikan Buku

Menginput Pengembalian Buku

«datastore» Simpan Informasi Pengembalian Buku

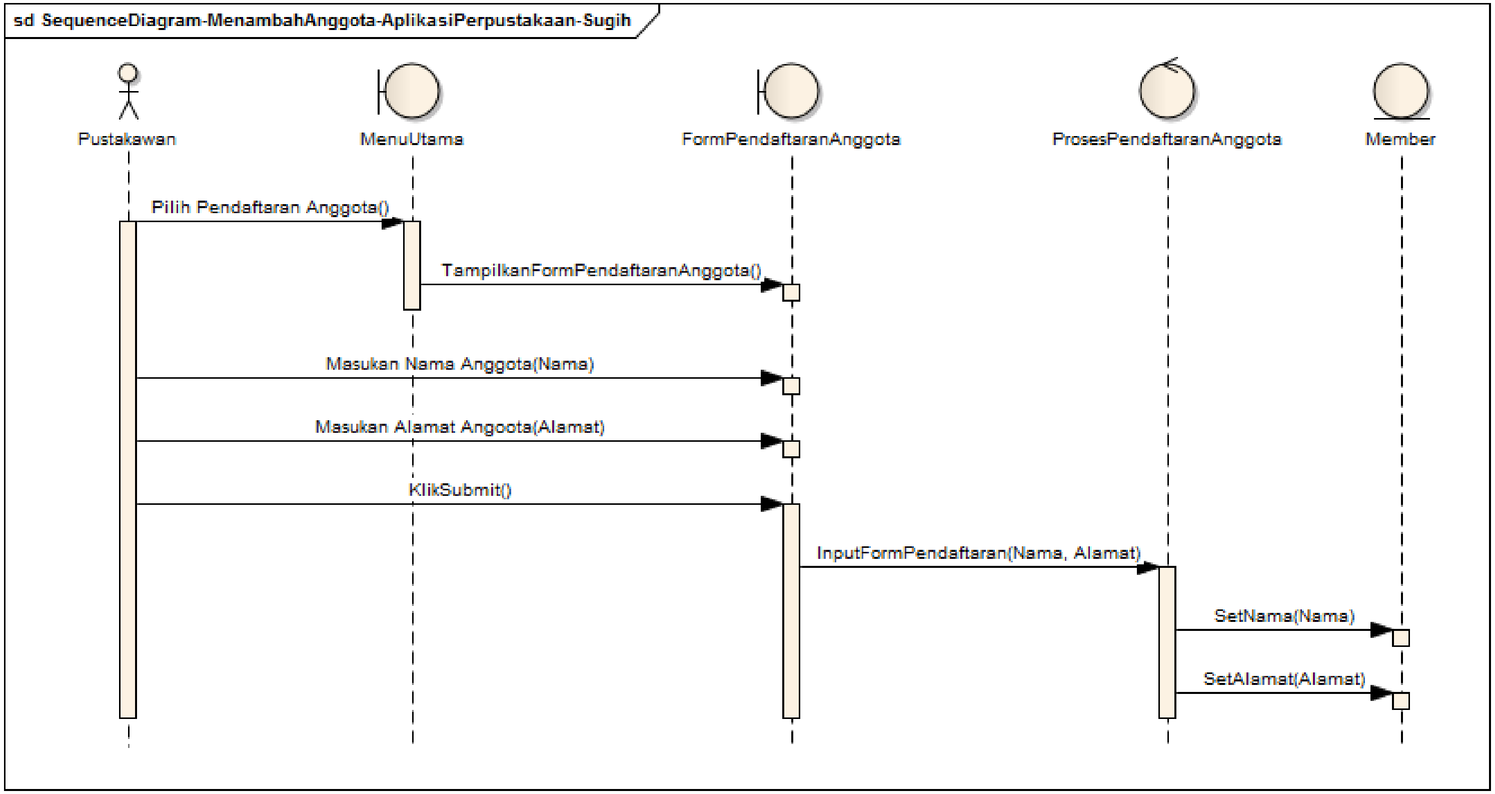
Menulis Tanggal Pengembalian di Kartu Buku

Selesai

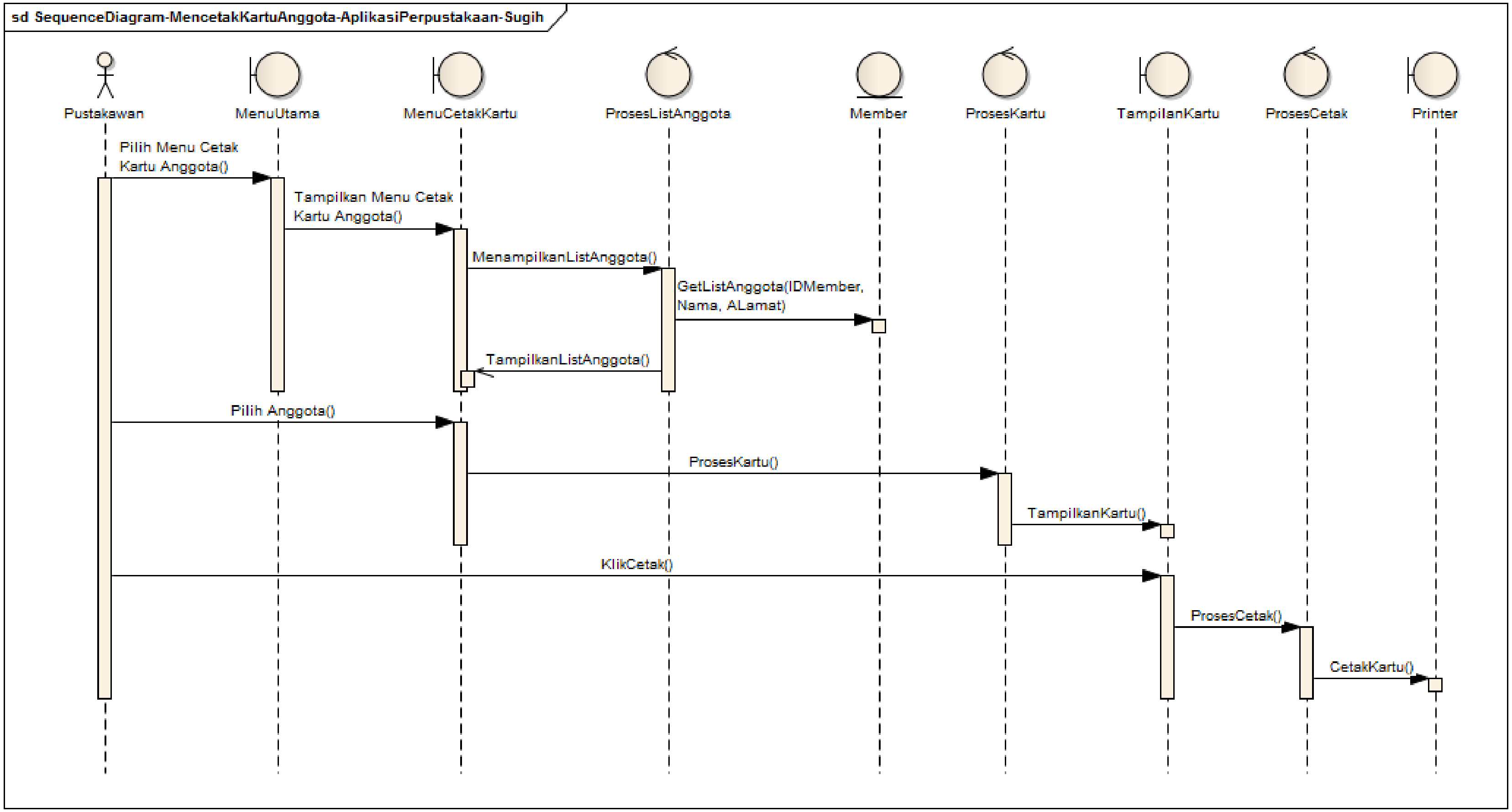
- The diagram above illustrates the 3 main activities in the library, namely:
  - Add members / library members.
  - Members borrow books.
  - Members return books.
- Although there may be many other activities related to libraries, they are not the main business process of libraries.

- Adding members

- There are two boundary classes, namely Main Menu and Member Registration Form, one Control class, namely Member Registration Process, and one Entity class, namely Member.


sd SequenceDiagram-MenambahAnggota-AplikasiPerpustakaan-Sugih
Pustakawan
MenuUtama
FormPendaftaranAnggota
ProsesPendaftaranAnggota
Member
Pilih Pendaftaran Anggota()
TampilkanFormPendaftaranAnggota()
Masukan Nama Anggota(Nama)
Masukan Alamat Angoota(Alamat)
KlikSubmit()
InputFormPendaftaran(Nama, Alamat)
SetNama(Nama)
SetAlamat(Alamat)

- PrintMember Cards



Conclusion

There are many ways to start making Sequence Diagrams. But the most common and easy ones are:

1. Create a Use Case Diagram
2. Create an Activity Diagram
3. Create Class Diagrams
4. Create a Sequence Diagram for each Use Case on your system
    1. Breakdown of the class diagrams involved in the Use Case.
    2. Describe the flow of calling its class methods.

Exercise

- Make a Class Diagram of the system you have created a Use Case and Activity Diagram (from the 3rd meeting).
- Make a Sequence Diagram of the system.