ICT 213
SYSTEM ANALYSIS AND DESIGN (SAD)

HERISON SURBAKTI, PH.D

OBJECT MODELING (INTRODUCTION)

ICT 213 SYSTEMS ANALYSIS AND DESIGN

OBJECTIVES

1. UNIFIED Model Process
2. UML View

INTRODUCTION

- The Object Modeling Technique (OMT) is an object modeling language for software modeling and design.
- Developed by Rumbaugh, Blaha, Premerlani, Eddy, and Lorensen as a method for the development of object-oriented systems and supporting object-oriented programming.

RAUMBAUGH OMT

- The purpose of modeling according to Raumbaugh (1991)
  - Performing physical testing of entities before constructing them (simulation)
  - Communicating with consumers
  - Visualization (an alternative to presenting information)
  - Reducing complexity
- There are 3 main types of models
  1. **Object Model**: The main concept involves classes and associations with attributes and operations. Relationships between classes take the form of aggregation and generalization.
  2. **Dynamic Model**: Represents the state/transition model. The main concepts are state, transitions between states, and events causing transitions. Actions are modeled as events within a state.
  3. **Functional Model**: Deals with processes in the model, linking to data flow diagrams. The main concepts include processes, data stores, data flow, and actors.

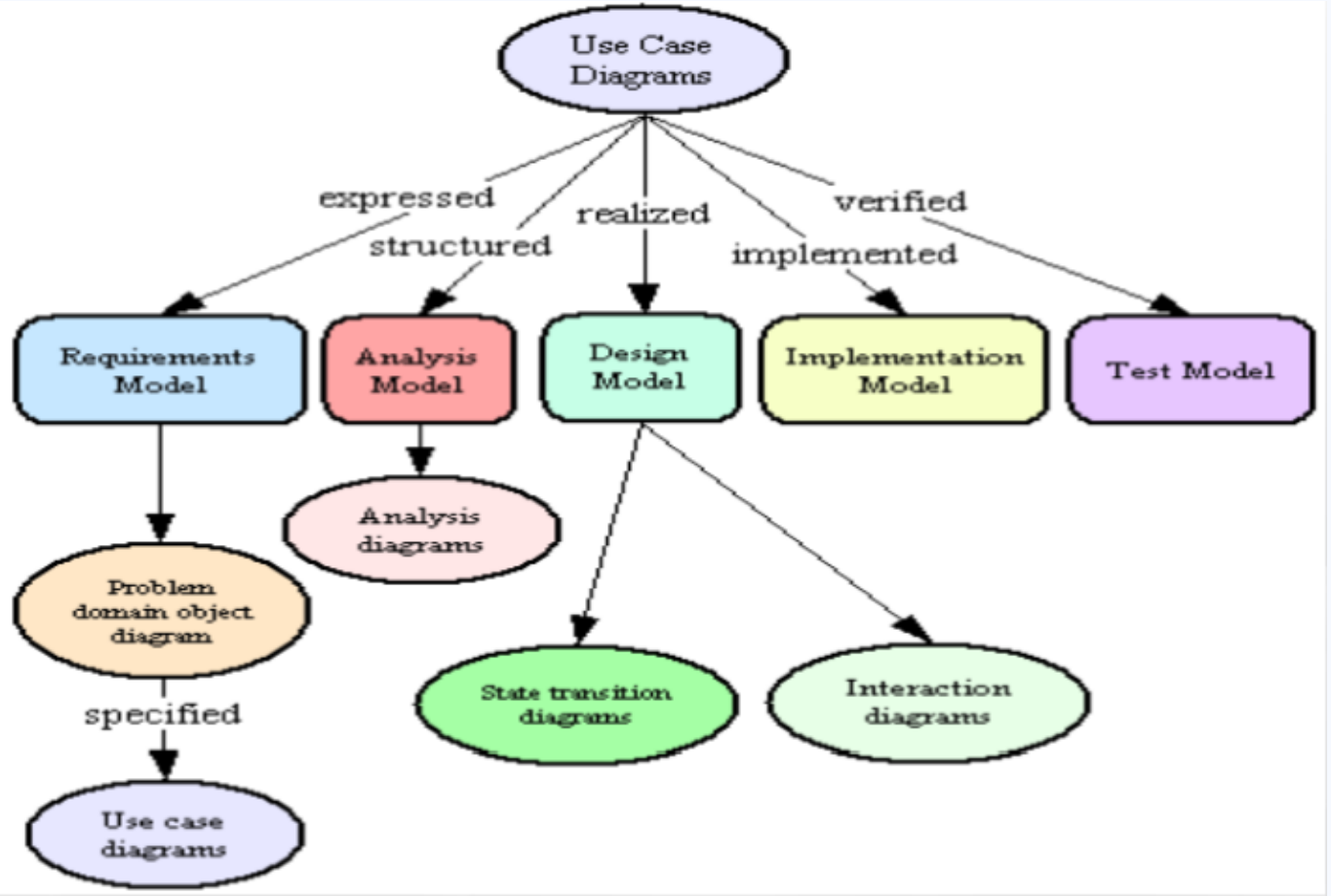
BOOCH OMT

- The analysis phase is divided into several steps
  - Consumer Requirements
  - Domain Analysis
  - Validation Step
  - Requirements, domain analysis, and validation
  - Iterative architectural design
  - Processes, performance, data types, data structures, visibility
  - Logical design, physical design, prototypes, and testing

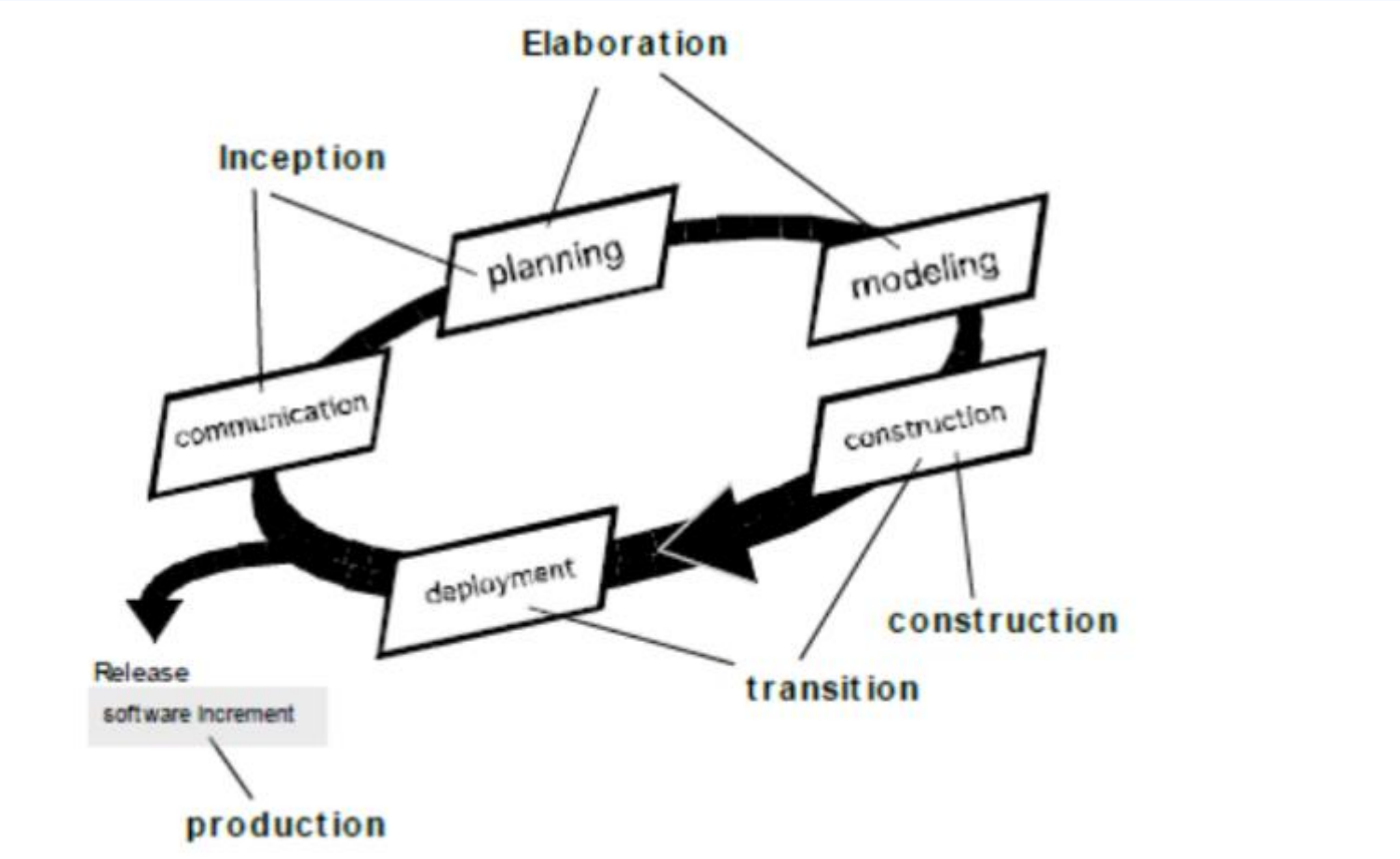
JACOBSON OOSE (OBJECT-ORIENTED SOFTWARE ENGINEERING)

- Object-Oriented Software Engineering (OOSE) is a software design technique used in object-oriented programming.
- OOSE was developed by Ivar Jacobson in 1992.
- OOSE is an object-oriented design methodology that employs use cases in software design.
- Components of OOSE include requirement modeling, analysis, design, implementation, and testing.

OOSE


Use Case Diagrams
expressed
structured
realized
implemented
verified
Requirements Model
Analysis Model
Design Model
Implementation Model
Test Model
Analysis diagrams
Problem domain object diagram
specified
Use case diagrams
State transition diagrams
Interaction diagrams

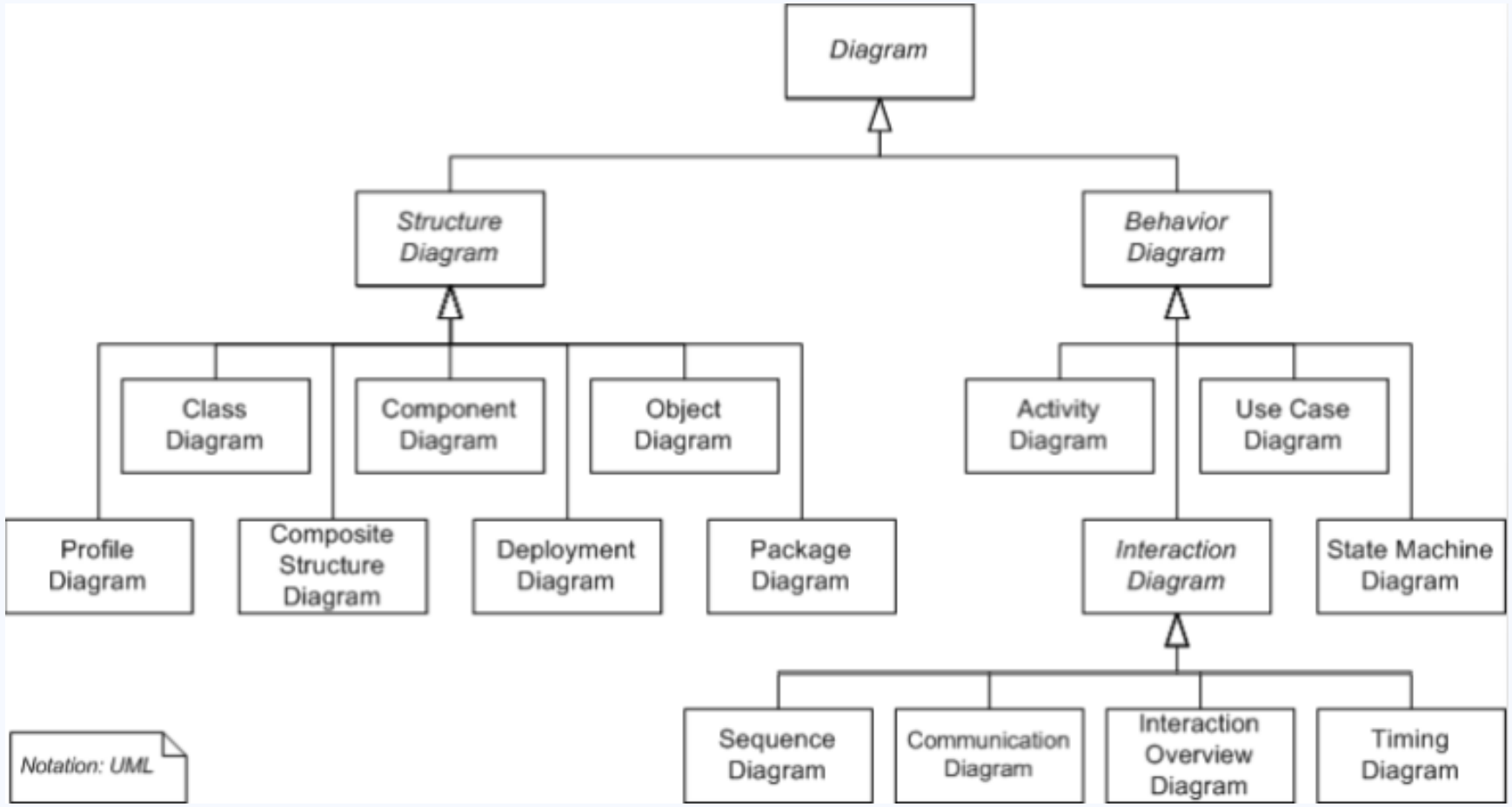
UNIFIED MODEL PROCESS



UNIFIED MODEL PROCESS

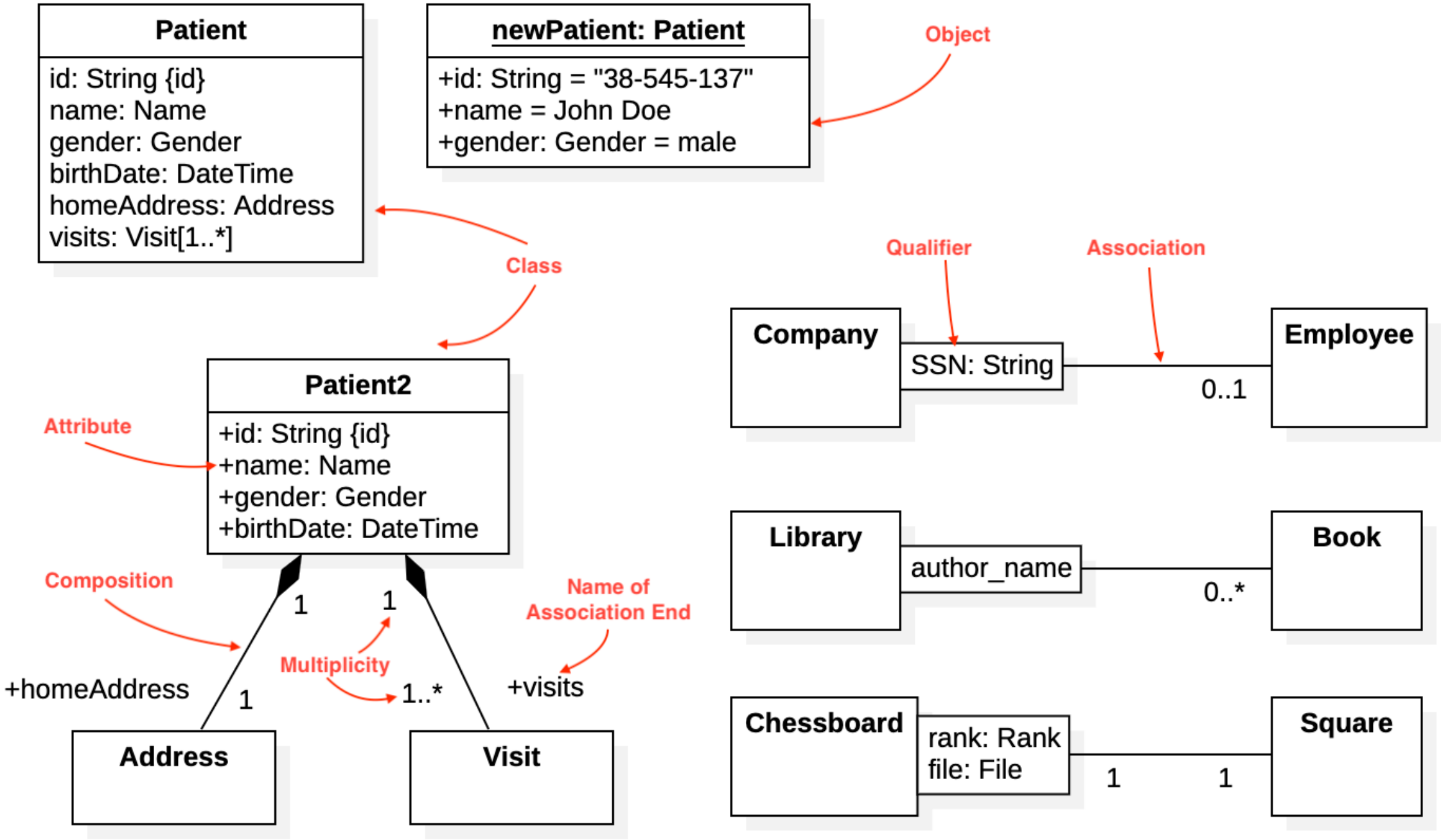
- Iterative and Incremental
- Use Case Driven
- Focus on Risk

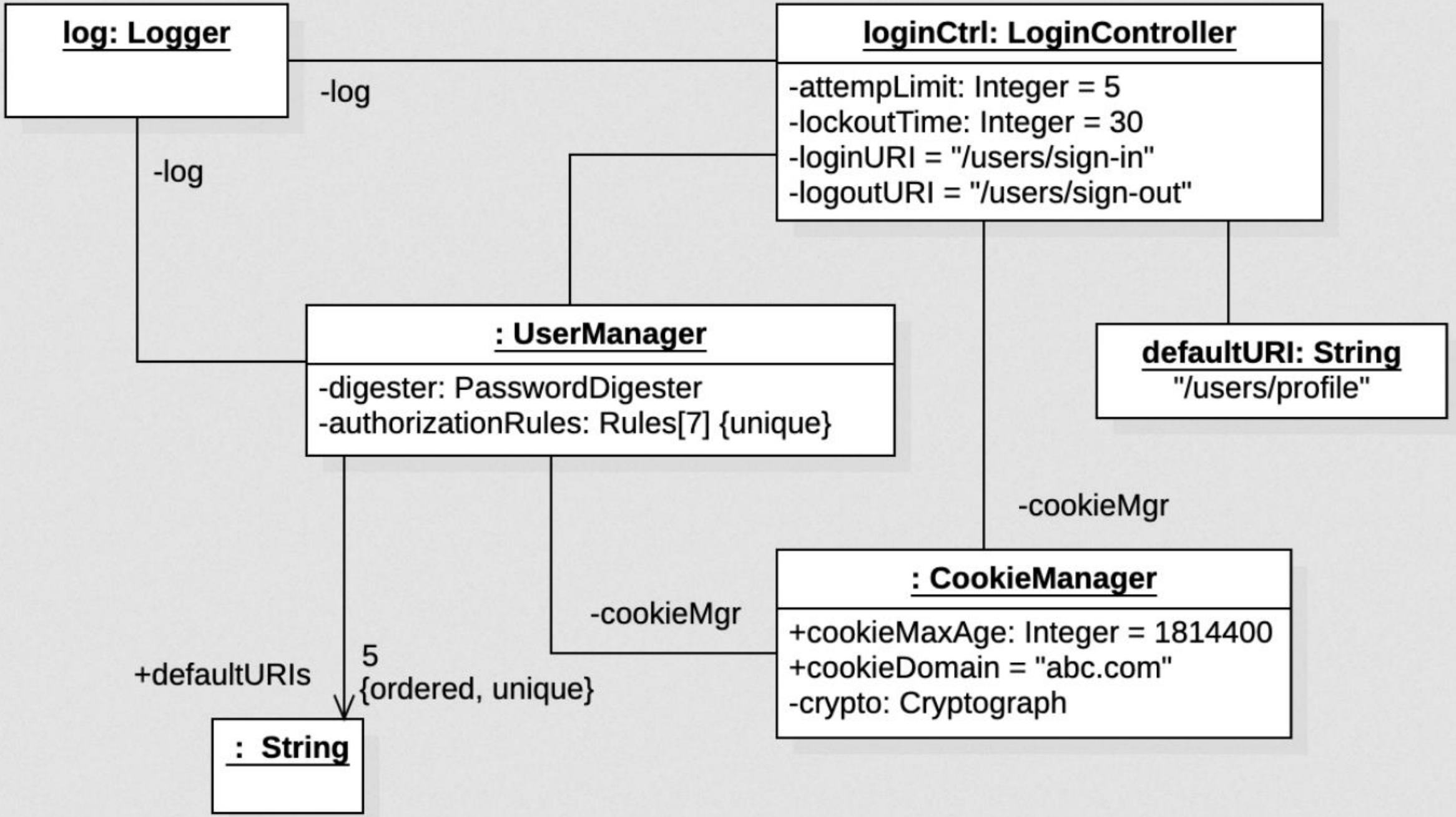
VIEW IN UNIFIED MODELING LANGUAGE (UML)

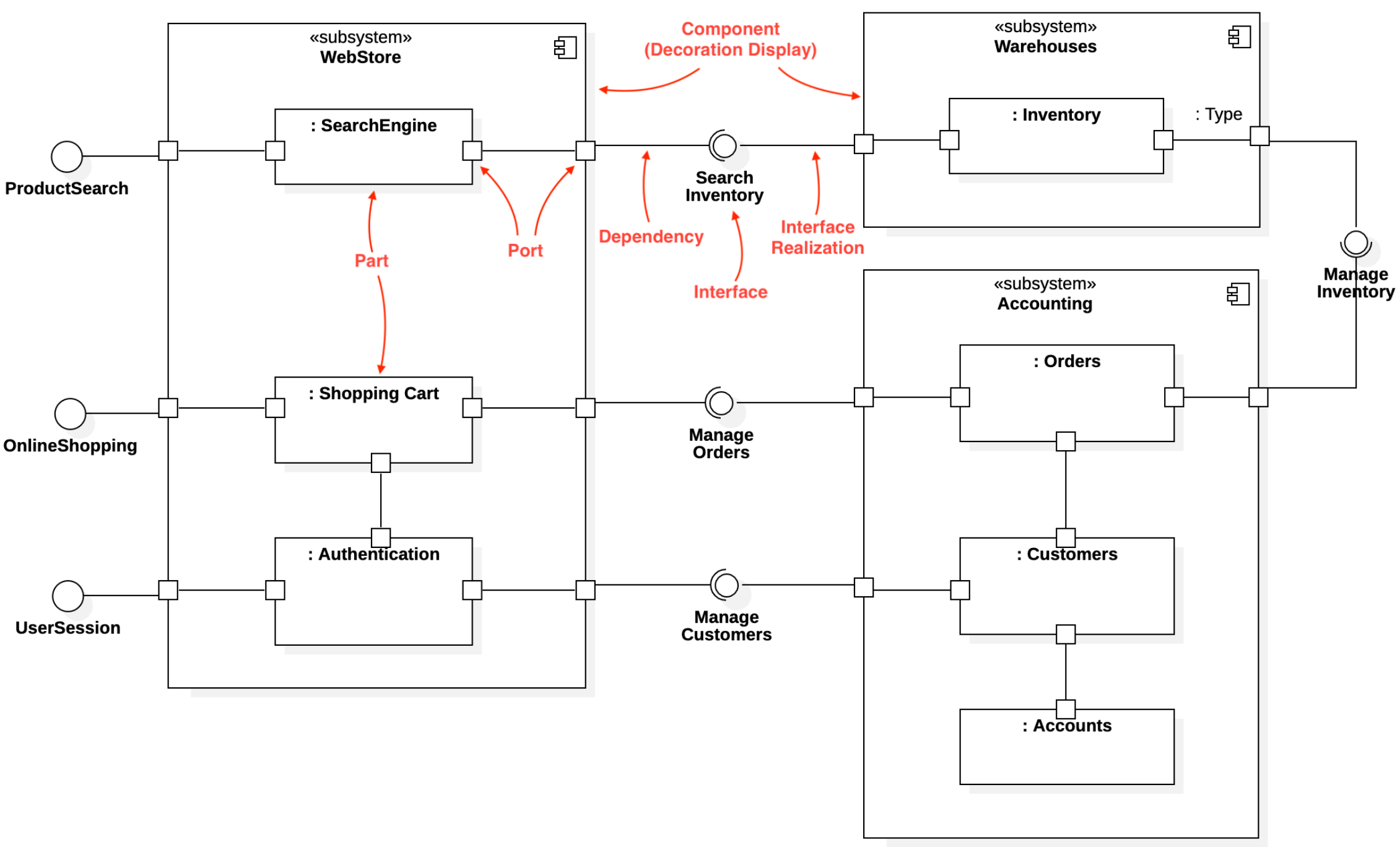


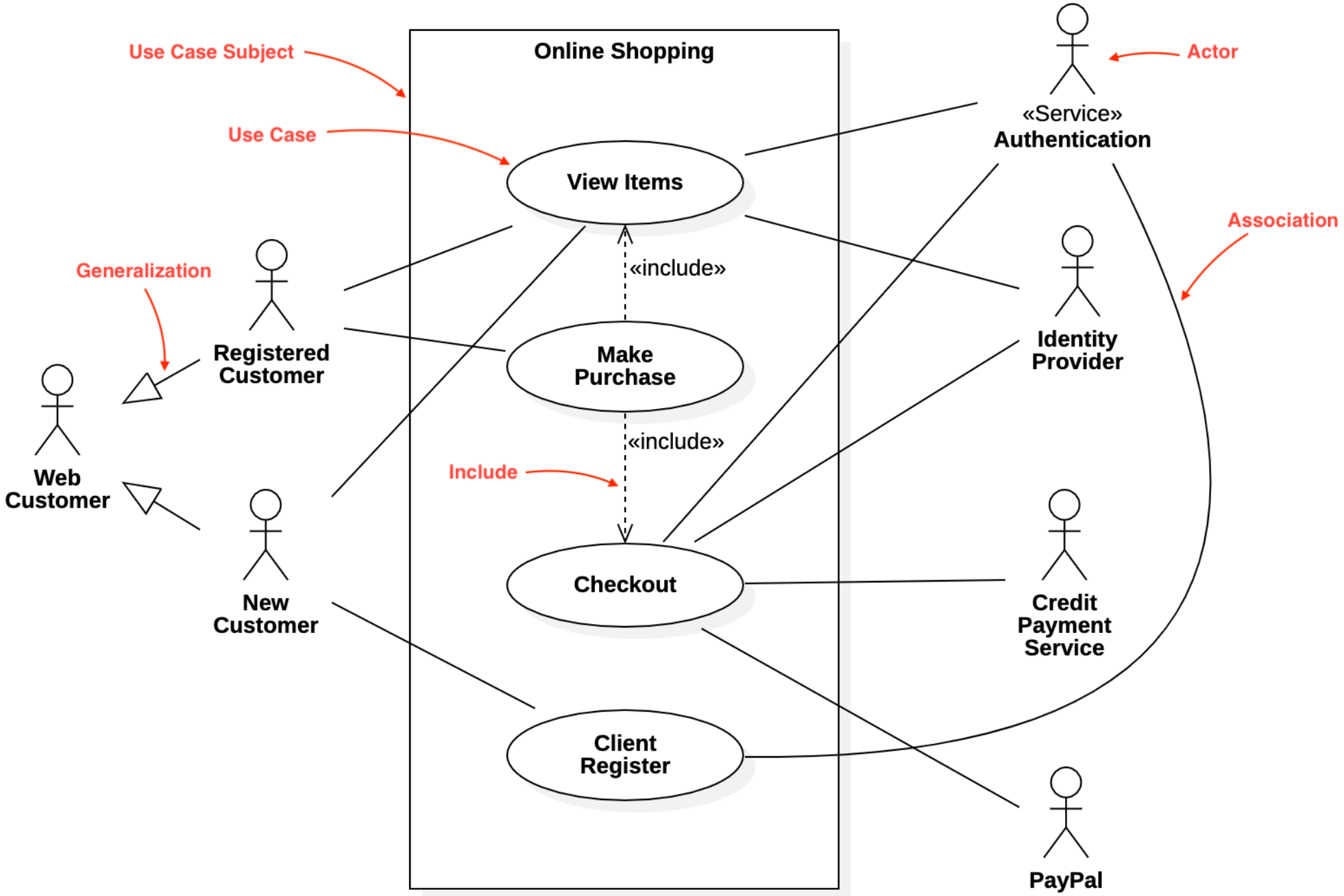
UML DIAGRAM

- Diagrams are the key to UML. These diagrams are categorized into 2 types:
  - ✓ Structural diagrams consist of static diagrams such as class diagrams, object diagrams, etc.
  - ✓ Behavioral diagrams consist of dynamic diagrams such as sequence diagrams, collaboration diagrams, etc.
- The static and dynamic behavior of the system is visualized using these diagrams.


Patient
id: String {id}
name: Name
gender: Gender
birthDate: DateTime
homeAddress: Address
visits: Visit[1..*]
newPatient: Patient
+id: String = "38-545-137"
+name = John Doe
+gender: Gender = male
Object
Class
Patient2
+id: String {id}
+name: Name
+gender: Gender
+birthDate: DateTime
Attribute
Composition
1
1
Multiplicity
1..*
Name of
Association End
+homeAddress
1
+visits
Address
Visit
Qualifier
Association
Company
SSN: String
0..1
Employee
Library
author_name
0..*
Book
Chessboard
rank: Rank
file: File
1
1
Square

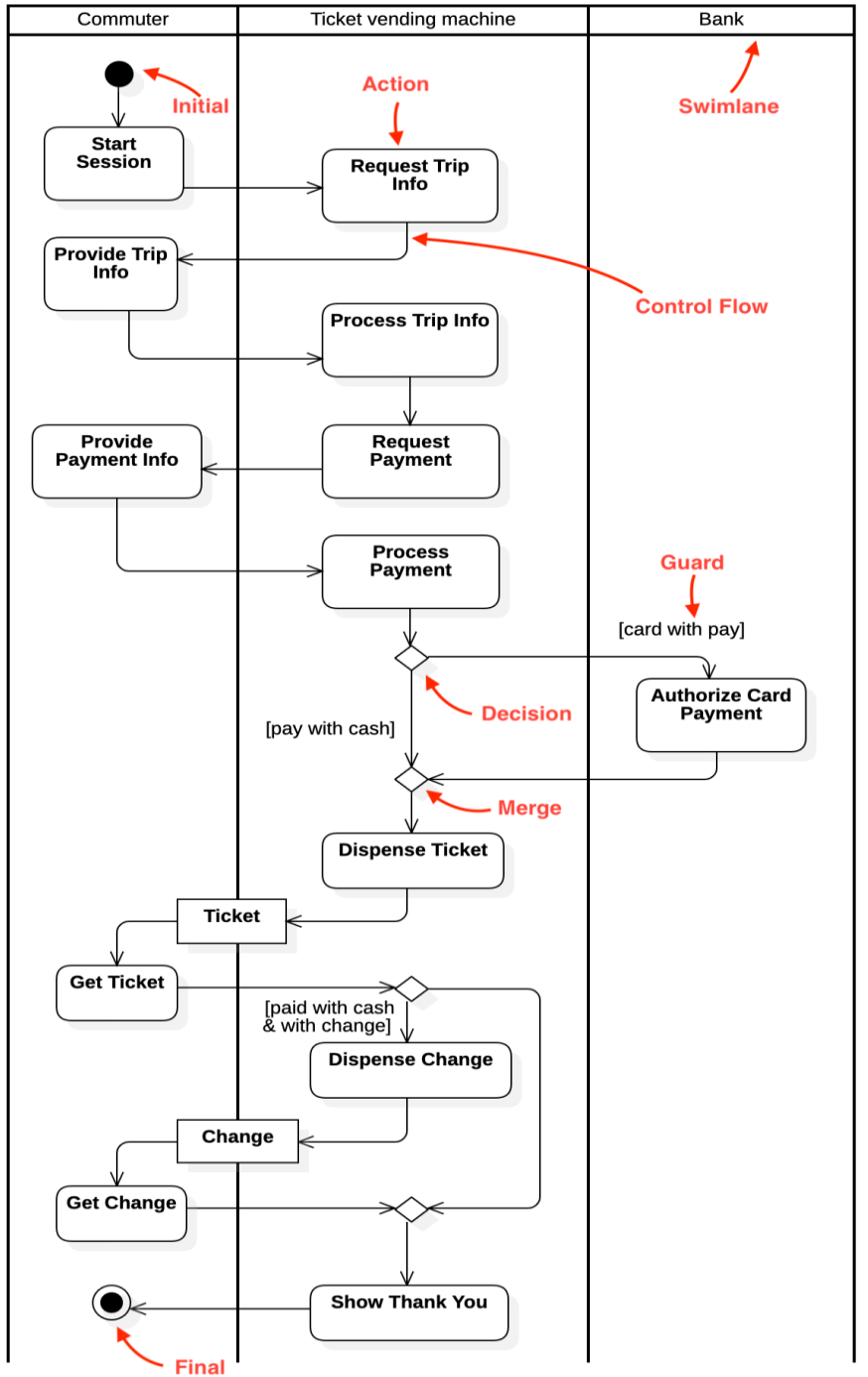

log: Logger
loginCtrl: LoginController
-attempLimit: Integer = 5
-lockoutTime: Integer = 30
-loginURI = "/users/sign-in"
-logoutURI = "/users/sign-out"
-log
-log
: UserManager
-digester: PasswordDigester
-authorizationRules: Rules[7] {unique}
defaultURI: String
"/users/profile"
-cookieMgr
-cookieMgr
: CookieManager
+cookieMaxAge: Integer = 1814400
+cookieDomain = "abc.com"
-crypto: Cryptograph
+defaultURIs
5
{ordered, unique}
: String

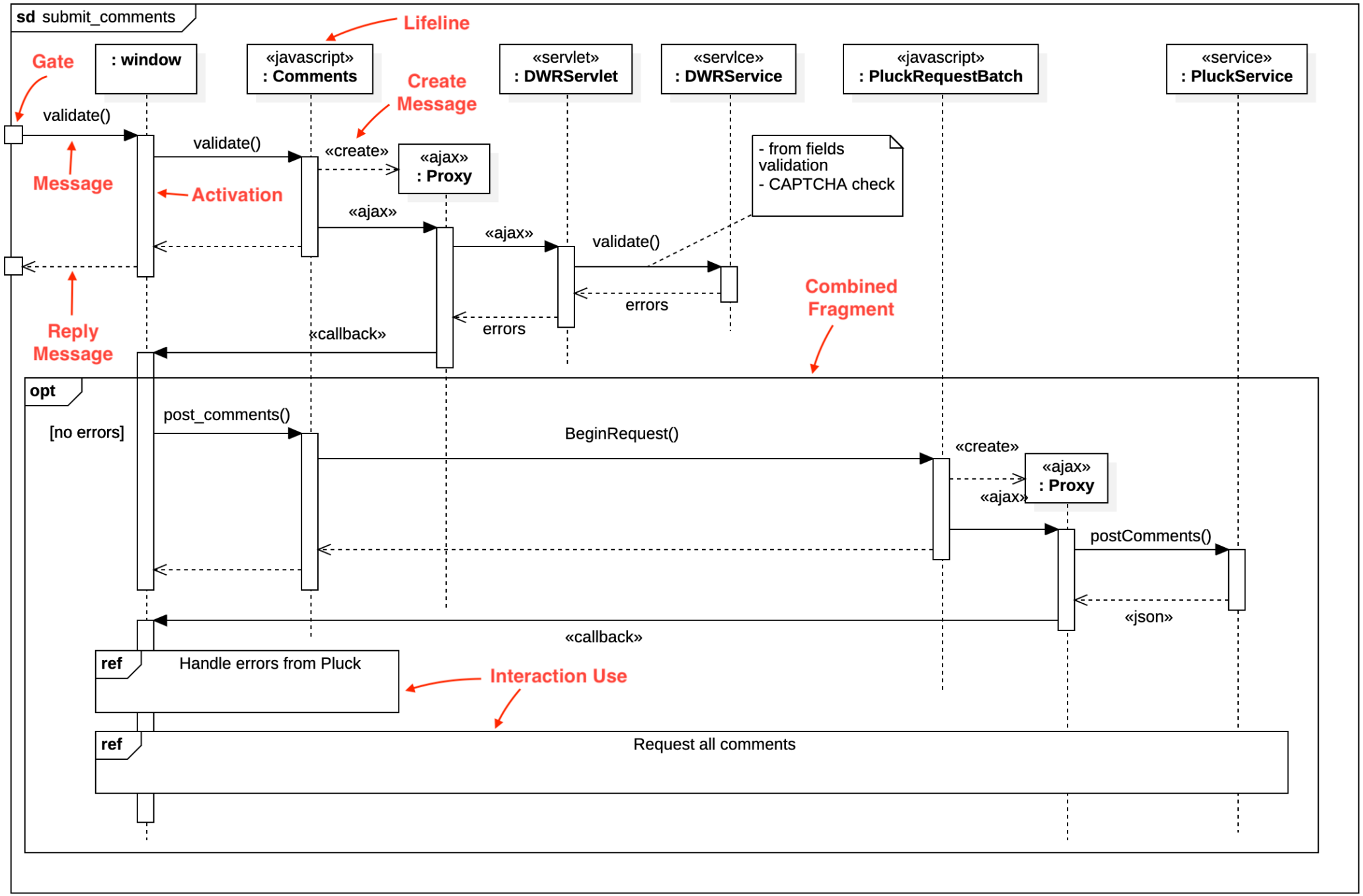

«subsystem»
WebStore
: SearchEngine
ProductSearch
: Shopping Cart
OnlineShopping
: Authentication
UserSession
Part
Port
Component
(Decoration Display)
Dependency
Interface
Search
Inventory
Interface
Realization
Manage
Orders
Manage
Customers
«subsystem»
Warehouses
: Inventory
: Type
Manage
Inventory
«subsystem»
Accounting
: Orders
: Customers
: Accounts


Use Case Subject
Online Shopping
Actor
Use Case
«Service»
Authentication
View Items
Association
Generalization
«include»
Registered
Customer
Identity
Provider
Make
Purchase
Web
Customer
«include»
Include
Checkout
Credit
Payment
Service
New
Customer
Client
Register
PayPal

ACTIVITY DIAGRAM

- Activity diagram is an important diagram for depicting dynamic behavior.
- Activity diagrams consist of activities, links, relationships, etc.
- Model all types of flows, including parallel, single, concurrent, etc.
- Activity diagrams depict the flow control from one activity to another without specific messages.
- This diagram is used to model at a high level of business requirements.




sd submit_comments
Lifeline
Gate
: window
«javascript»
: Comments
Create
Message
«servlet»
: DWRServlet
«servlce»
: DWRService
«javascript»
: PluckRequestBatch
«service»
: PluckService
validate()
validate()
«create»
«ajax»
: Proxy
- from fields validation
- CAPTCHA check
Message
Activation
«ajax»
«ajax»
validate()
errors
errors
Combined Fragment
Reply Message
«callback»
opt
post_comments()
[no errors]
BeginRequest()
«create»
«ajax»
: Proxy
«ajax»
postComments()
«json»
«callback»
ref
Handle errors from Pluck
Interaction Use
ref
Request all comments

Thank you

Dr. Herison Surbakti
Email: herison.s@rsu.ac.th
Department of Information and communication technology, International college