Software Process Modeling

Software Design

Session Outcomes

- What is Software design?
- Design Types
- Object Oriented Design
  - Understand System and interactions
  - Design System Architecture
  - Identify main classes and objects
  - Develop Design Models
    - UML
    - SysML
  - Specify Interfaces

Story so far …

- Feasibility study
- Requirement phase
  - – Requirements elicitation and analysis
  - – Requirements Specification
    - Use case diagrams
    - Activity Diagrams
  - – Requirement validation
- Today's lecture : Software Design

Programmer's Approach to Software Engineering

Skip requirements engineering and design phases;

start writing code

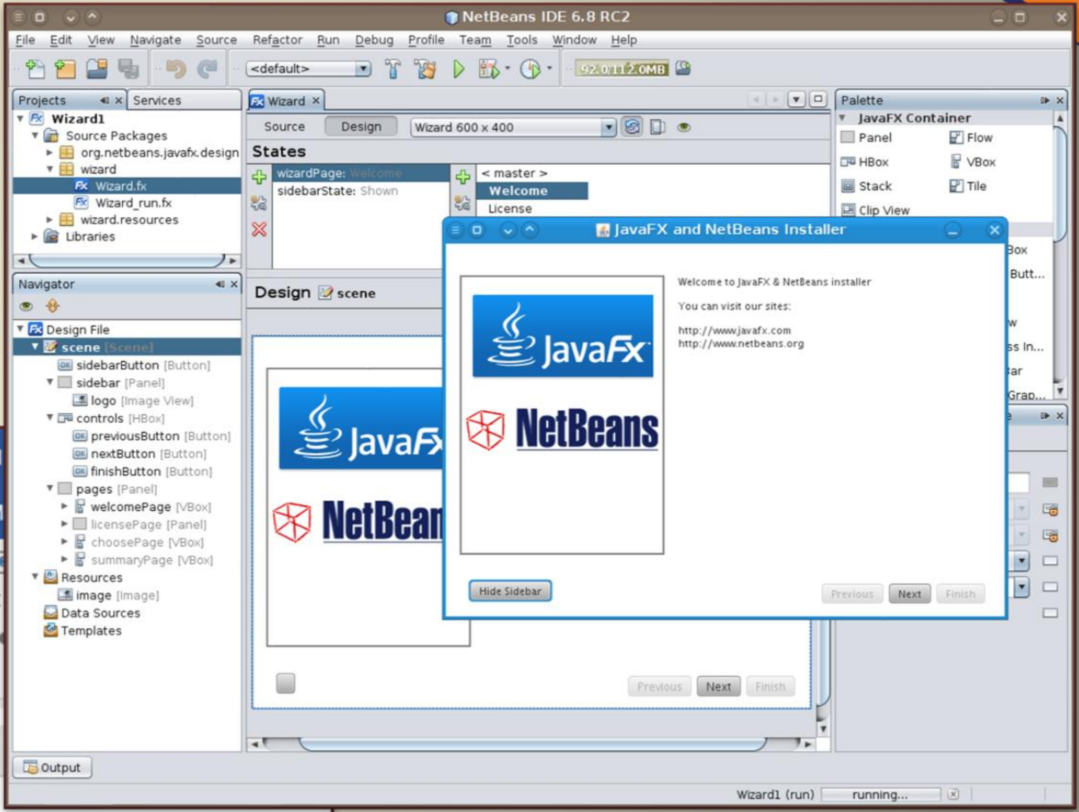
Why this programmer's approach?

- Design is a waste of time
- We need to show something to the customer really quickly.
- We are judged by the amount of LOC/month
- We expect or know that the schedule is too tight

Design of small and large systems

Ref: "Software Architecture and UML" Grady Booch

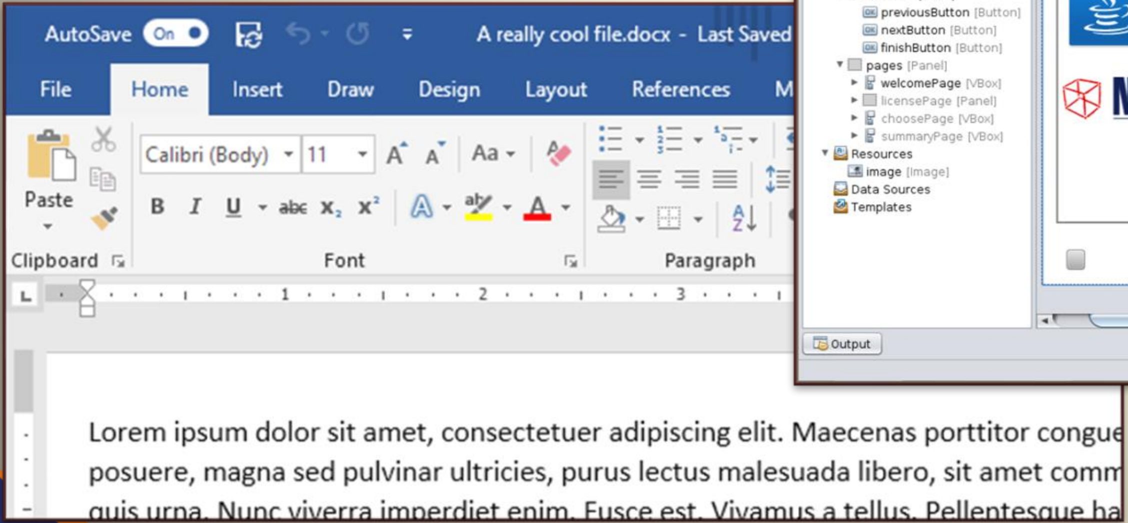
Design of small and large systems



Welcome to JavaFX & NetBeans installer

You can visit our sites:

http://www.javafx.com
http://www.netbeans.org



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue
posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet comm
quis urna. Nunc viverra imperdiet enim. Fusce est. Vivamus a tellus. Pellentesque ha

Importance of design

- Software design is an iterative process through which requirements are translated into a "blueprint" for constructing the software.

Ref: Software Engineering A Practitioner's approach , R.S. Pressman, 7th Edition

- Design is a highly creative stage in software development where the designer plans
  - how the system or program should meet the customer's requirements
  - how to make system effective and efficient.

Ref: Software Engineering, I. Sommerville, 10th Edition

Stages of design

- Understand the problem
  - Look at the problem from different angles to discover the design requirements
- Identify one or more solutions
  - Evaluate possible solutions and choose the most appropriate
- Describe solution abstractions
  - Use graphical, formal or other descriptive notations to describe the components of the design
- Repeat process for each identified abstraction until the design is expressed in primitive terms

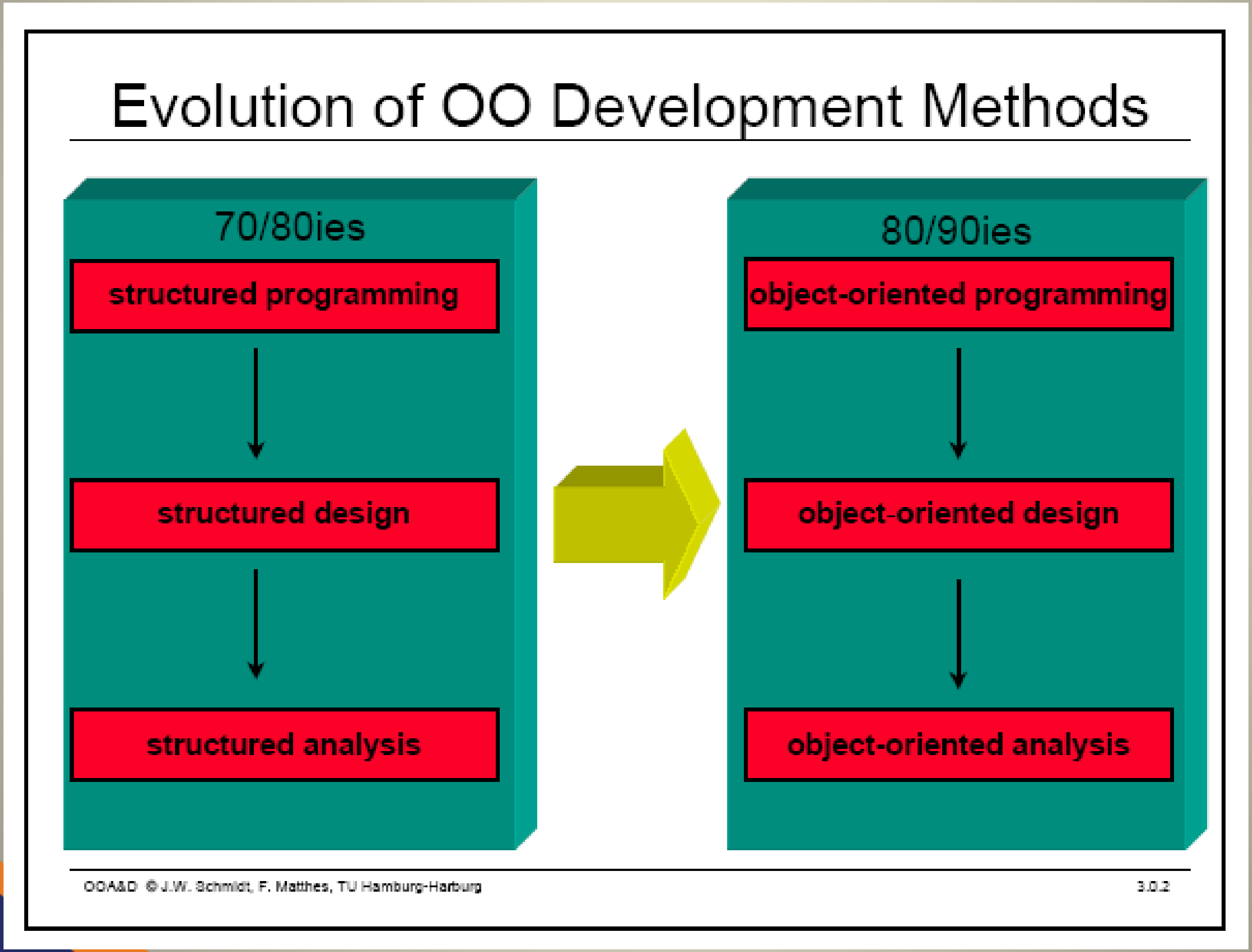
Software Design methods

- Function oriented software design
- Object oriented software design

Software Design

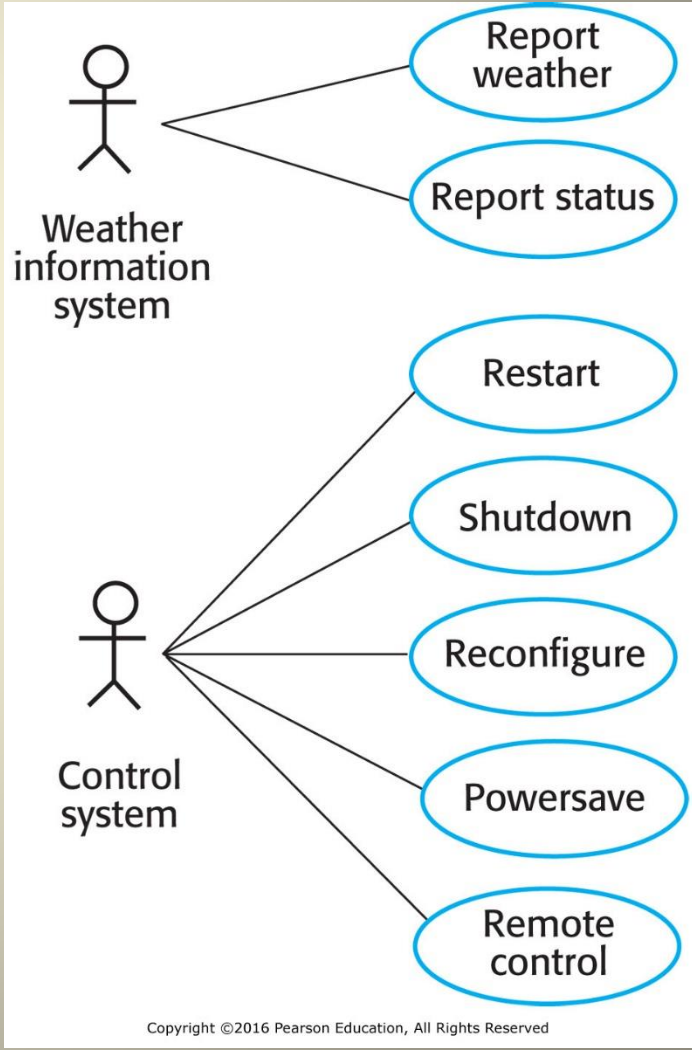
Object Oriented Design

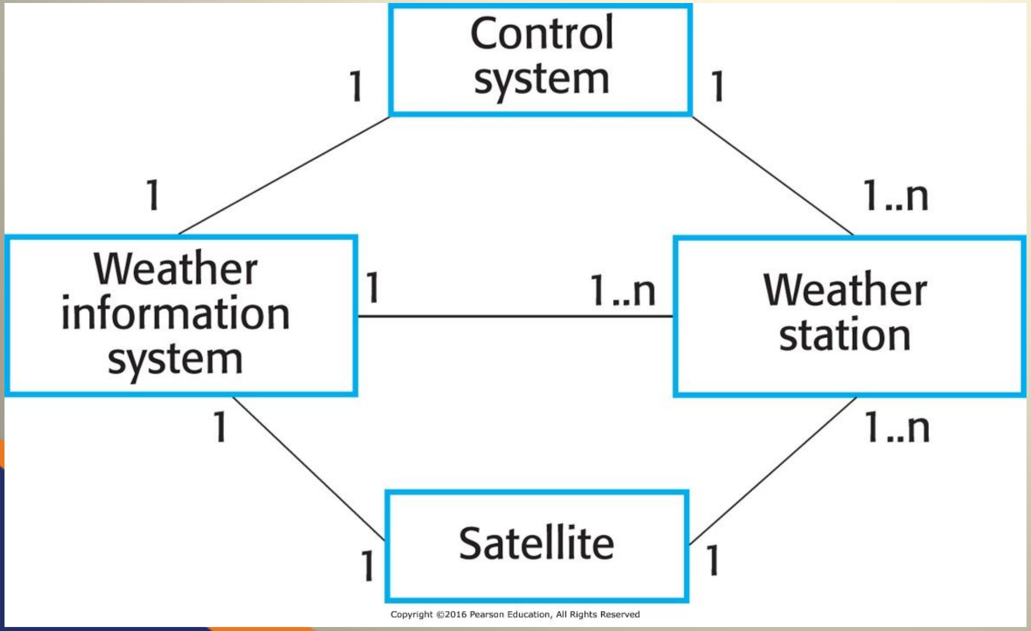
Object oriented software design



Object Oriented Design

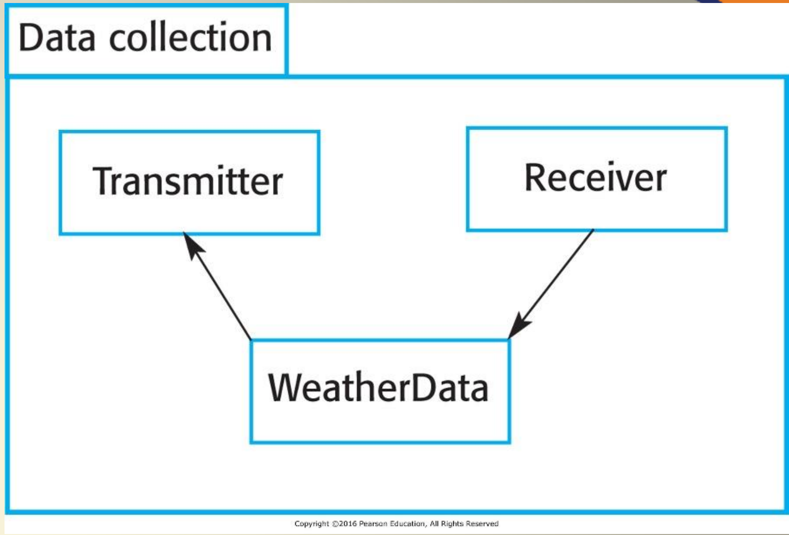
1. Understand System and interactions
   - Use case Diagrams
   - Activity Diagrams
   - Use case Scenarios

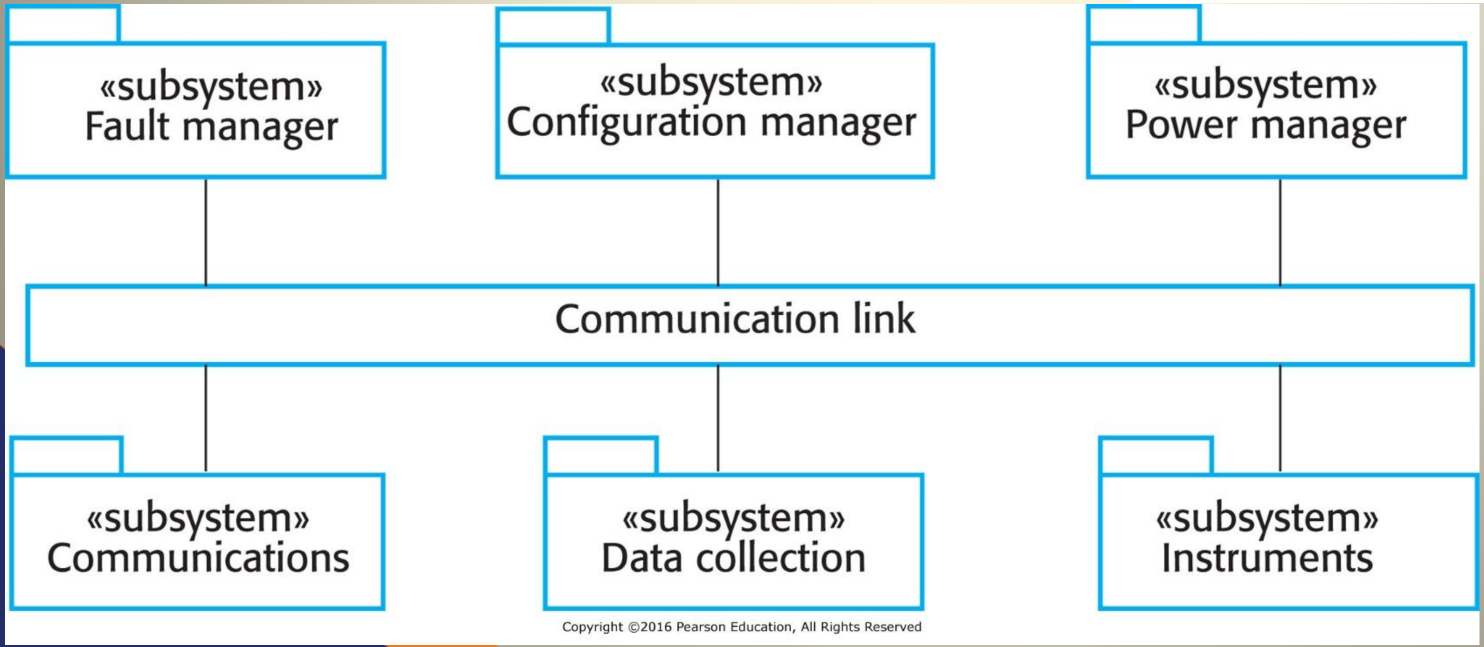




Object Oriented Design

2. Design System Architecture
   – Subsystems and communication between the subsystems.



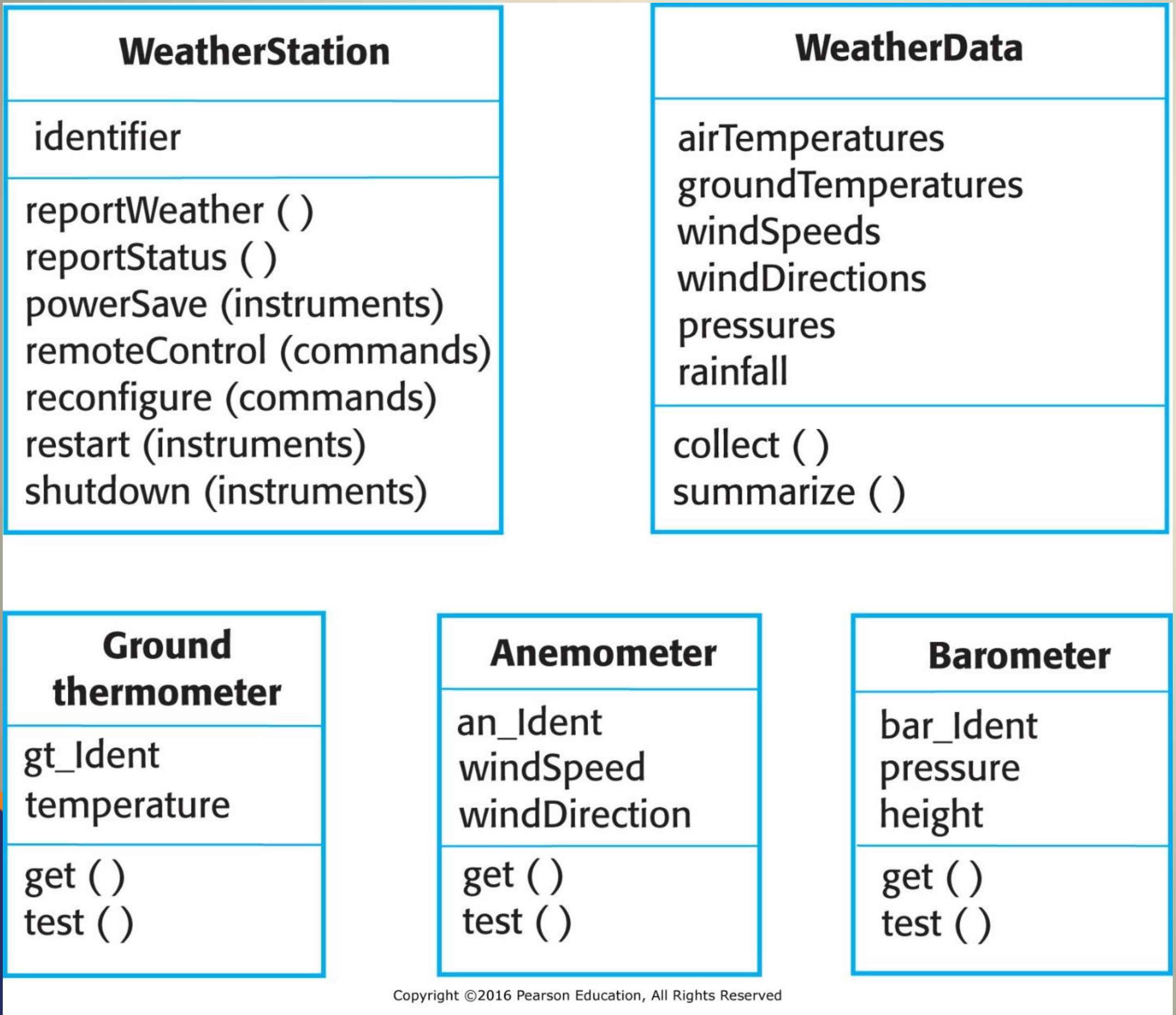

«subsystem» Fault manager
«subsystem» Configuration manager
«subsystem» Power manager
Communication link
«subsystem» Communications
«subsystem» Data collection
«subsystem» Instruments
Copyright ©2016 Pearson Education, All Rights Reserved

Sub system identification

- Different levels
  - Level 1
    - Item Manager
    - User Manager
    - Branch Manager
    - Service Manager
  - Level 2 – Item Manager
    - Periodicals
    - Borrowing Items
    - Soft Items
    - Reference items

Object Oriented Design

3. Identify main classes and objects



Activity

- What are the CRC cards you identified for the Library system?

Answer - CRC Cards

Member	
Add member details Update status Calculate return date Add refund details Add deposit details Search Item	Borrowed Items

Borrowed Item	
Add Borrow details Add return details Calculate Fine Update Status of Borrowing List Borrowed Items	Member Item

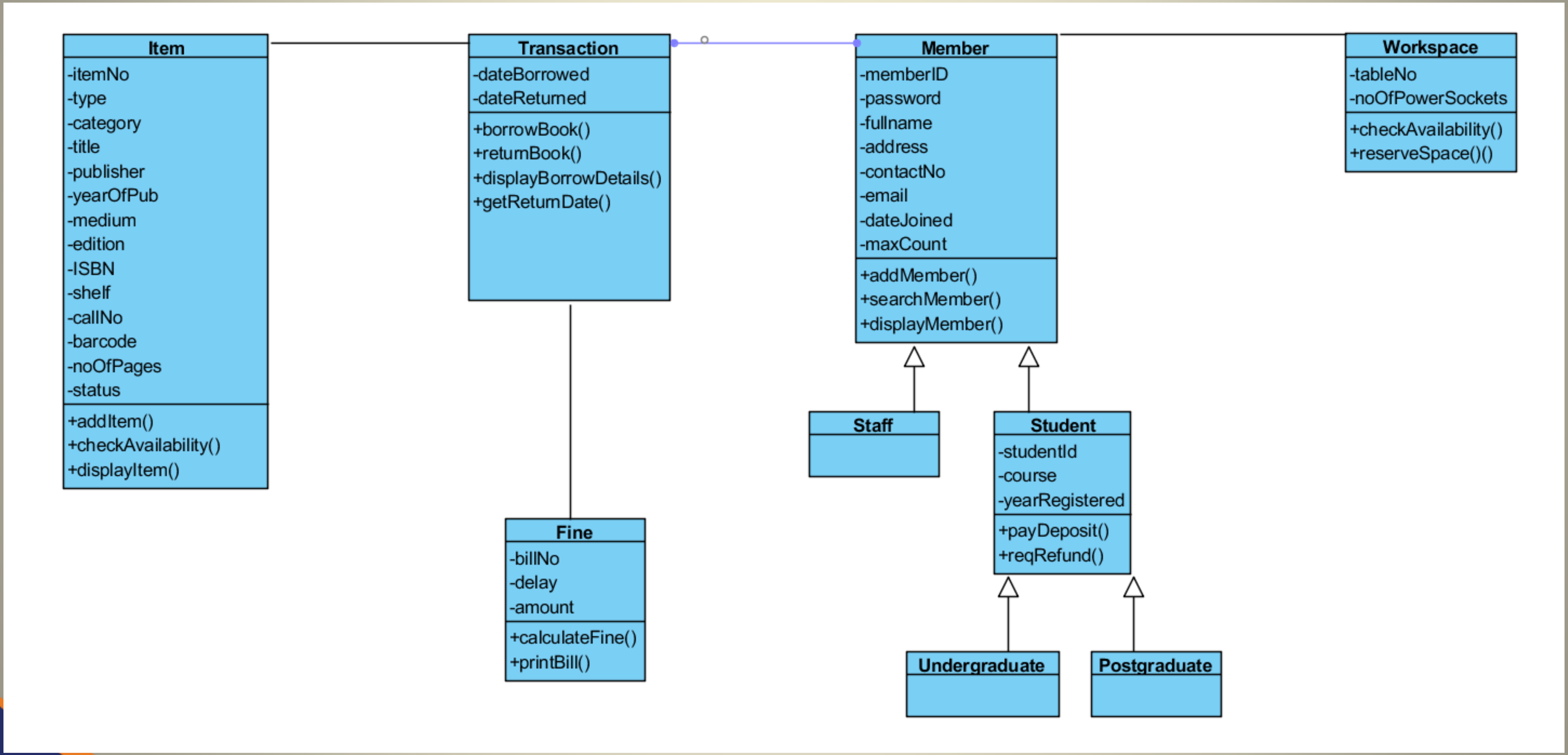
Item	
Add new items Update return details Update Lost Items	Borrowed Items

Reports	
List Items Borrowed List Items Available List Items Overdue List Membership Details	Borrowed Items Item Member

Classes and objects

- Item Manager
  - Borrowing Item
    - Book
    - Magazine
    - CD – Soft Item?

Classes for Library System

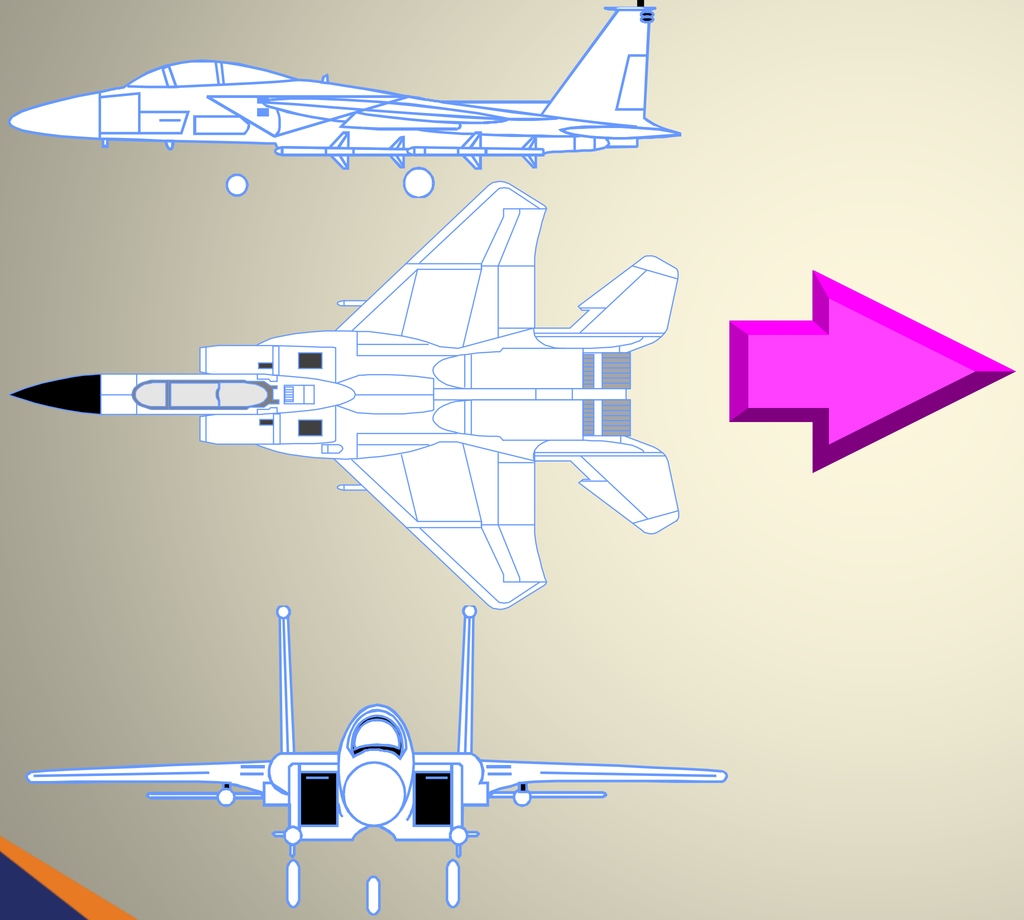


Object Oriented Design

4. Develop Design Models
   - Describing a system at a high level of abstraction
   - Design Model types
     - Structural models
     - Dynamic models
   - Is it necessary to model software systems?

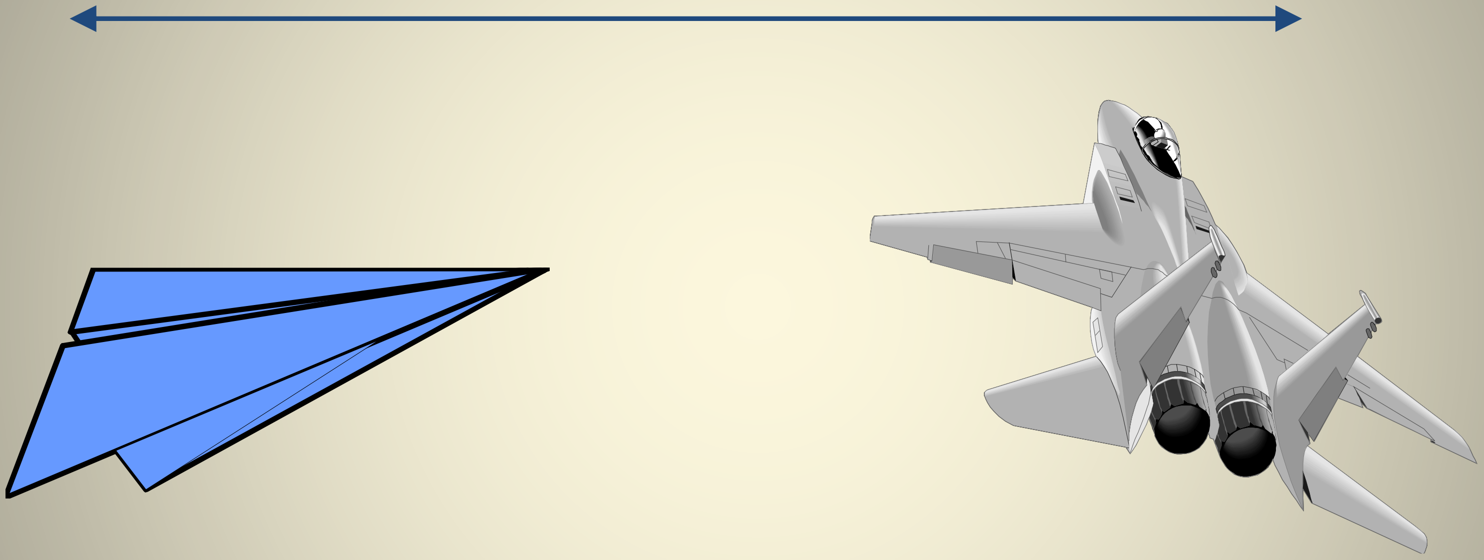
What Is a Model?

- A model is a simplification of reality.



Ref: Fundamentals of Visual Modeling with UML

The Importance of Modeling



Paper Airplane

Fighter Jet

Ref: Fundamentals of Visual Modeling with UML

Software Teams Often Do Not Model

- Many software teams build applications approaching the problem like they were building paper airplanes
  - Start coding from project requirements
  - Work longer hours and create more code
  - Lacks any planned architecture
  - Doomed to failure
- Modeling is a common thread to successful projects.

Ref: Fundamentals of Visual Modeling with UML

Why Do We Model?

- Modeling achieves **four aims**:
  - Helps us to **visualize** a system as we want it to be.
  - Permits us to **specify** the structure or behavior of a system.
  - Gives us a template that guides us in **construct**ing a system.
  - **Document**s the decisions we have made.
- We build models of complex systems because we cannot comprehend such a system in its entirety.
- We build models to better understand the system we are developing.

Ref: Fundamentals of Visual Modeling with UML

Object Oriented Design

4. Develop Design Models
   - Design Model types
     - Structural models
     - Dynamic models
   - Modeling Languages
     - UML
     - SysML
     - Refer https://modeling-languages.com/#

What Is the UML?

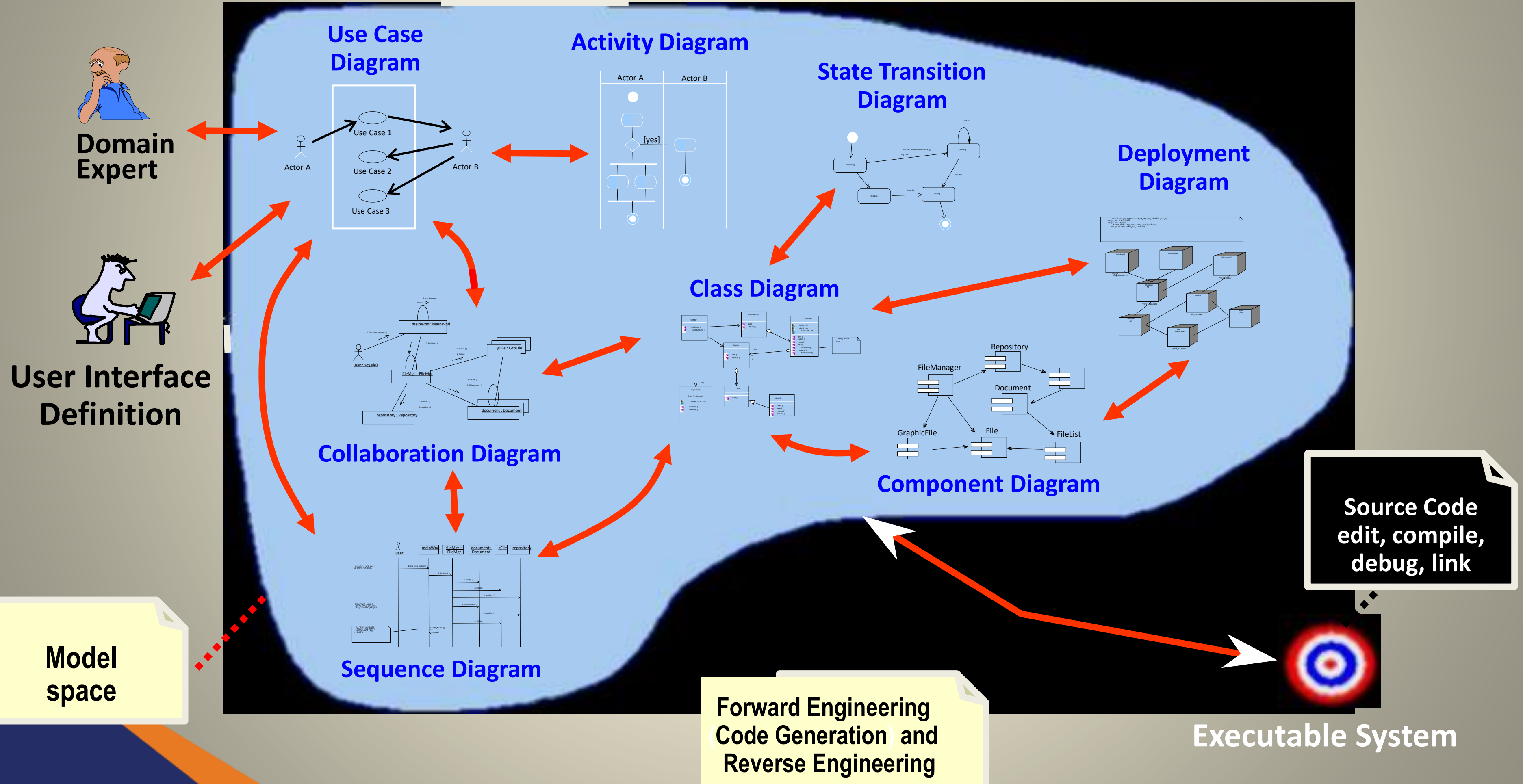
- The UML is a language for
    - Visualizing
    - Specifying
    - Constructing
    - Documenting

  the artifacts of a software-intensive system.

- Out of the above, **SPM** and **SE** modules specially focus on using *UML as a language for specifying and documenting*.

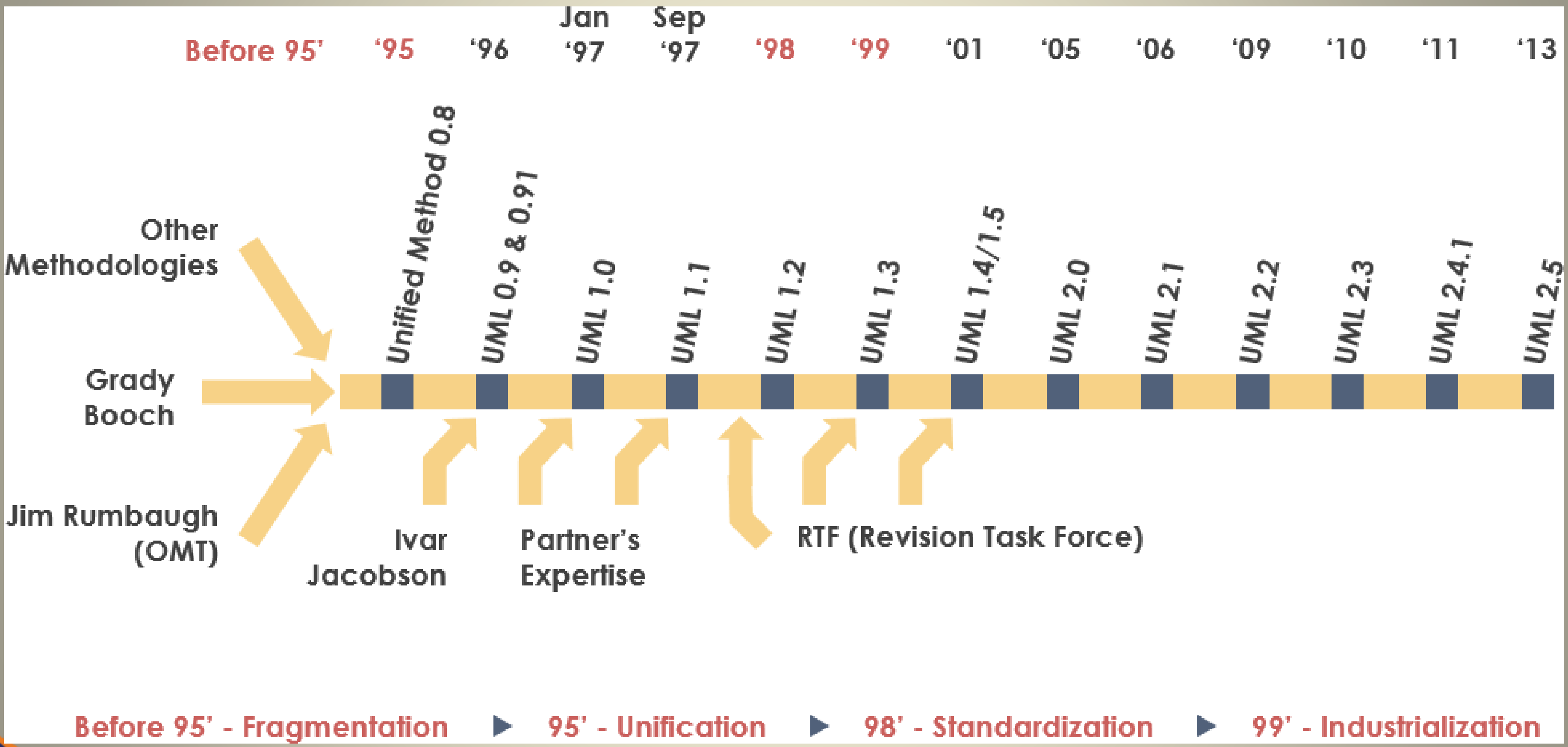
Ref: Fundamentals of Visual Modeling with UML

The UML Is a Language for Specifying and Documenting

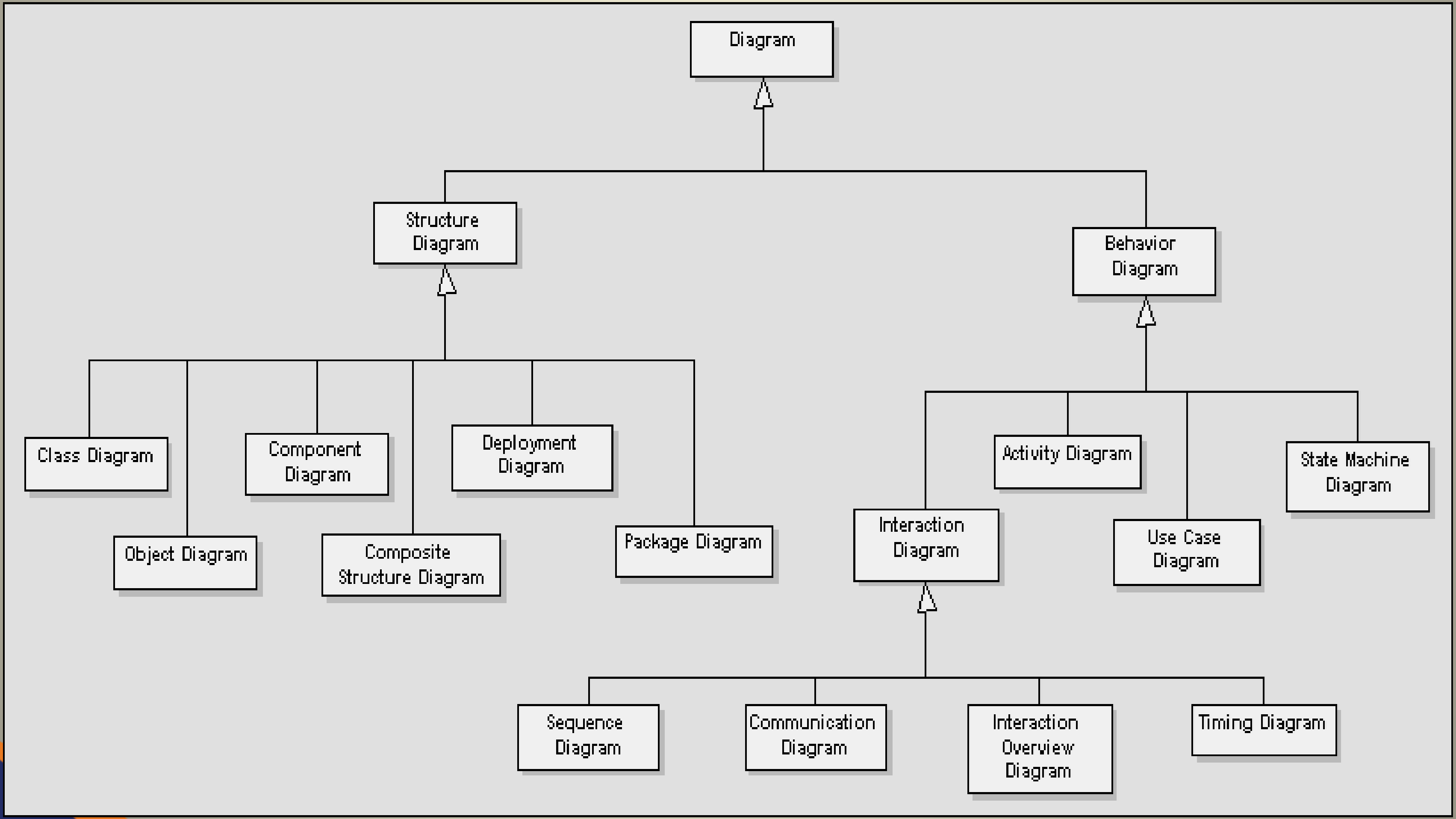


Ref: Fundamentals of Visual Modeling with UML

UML - History


Before 95'
'95
'96
Jan '97
Sep '97
'98
'99
'01
'05
'06
'09
'10
'11
'13
Unified Method 0.8
UML 0.9 & 0.91
UML 1.0
UML 1.1
UML 1.2
UML 1.3
UML 1.4/1.5
UML 2.0
UML 2.1
UML 2.2
UML 2.3
UML 2.4.1
UML 2.5
Other Methodologies
Grady Booch
Jim Rumbaugh (OMT)
Ivar Jacobson
Partner's Expertise
RTF (Revision Task Force)
Before 95' - Fragmentation
95' - Unification
98' - Standardization
99' - Industrialization

UML 2 Structure



Types of UML diagrams

- There are different types of UML diagram, each with slightly different syntax rules:
  - use cases- Covered in RE
  - activity diagrams- Covered in RE
  - class diagrams. – Cover in OOC
  - sequence diagrams.
  - collaboration diagrams.
  - state diagrams
  - component diagrams.
  - deployment diagrams.

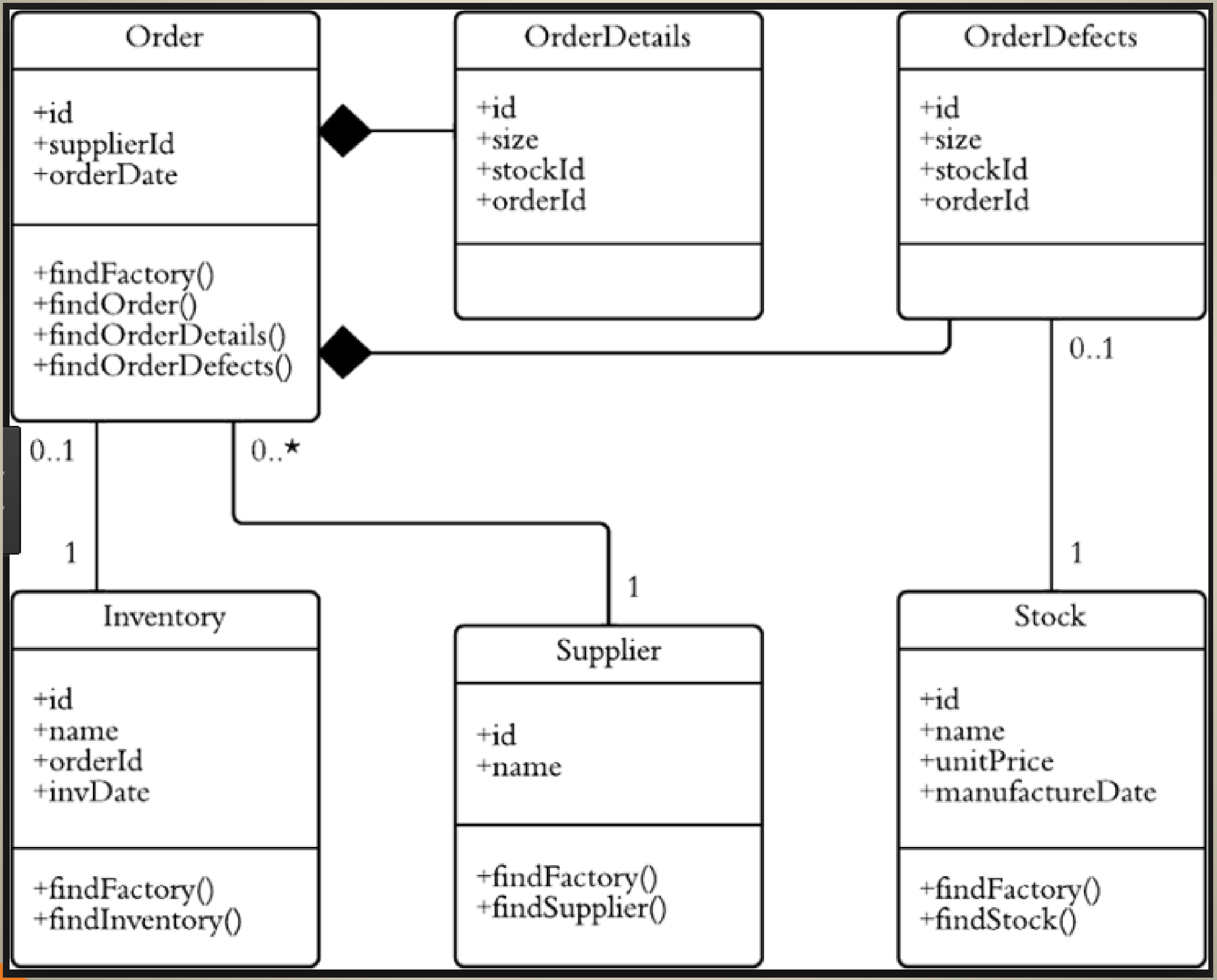
Covers in SE in the next semester

Design

- When you use UML to develop a design, you will normally develop two kinds of design models:
  1. Structural models :
     - describe the static structure of the system using objects, classes and their relationships.
     - Important relationships that may be documented at this stage are generalization (inheritance), aggregation, dependency, and composition relationships. (class diagram relationships in OOC)

Ref: Software Engineering, I. Sommerville, 10th Edition

Structural models Example

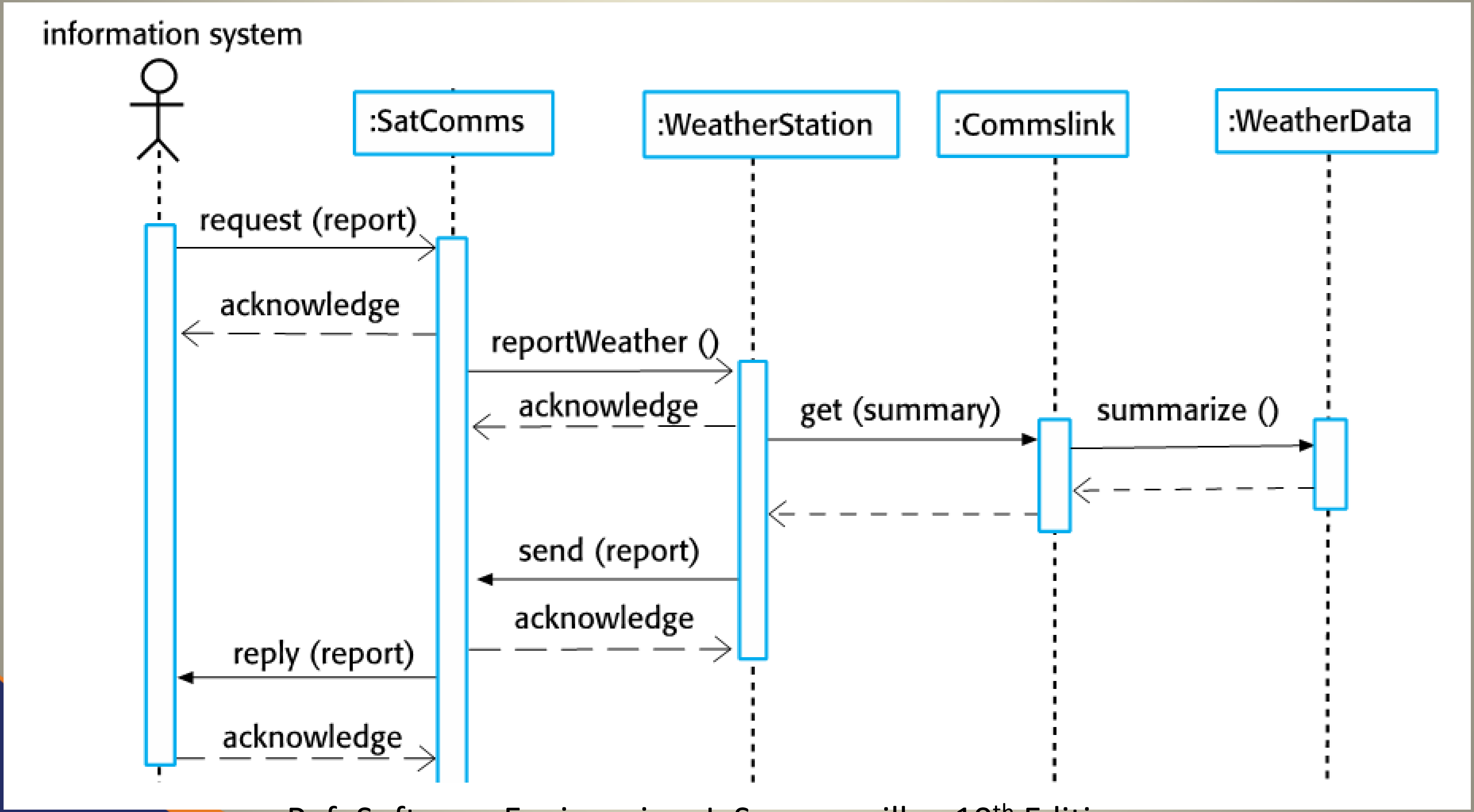


Design

2. Dynamic models :
   - Describes the dynamic structure of the system and shows the interactions between the system objects.
   - Interactions that may be documented include the sequence of service requests made by objects and the state changes that are triggered by these object interactions. (You will learn them in SE next semester)

Ref: Software Engineering, I. Sommerville, 10th Edition

Dynamic models Example

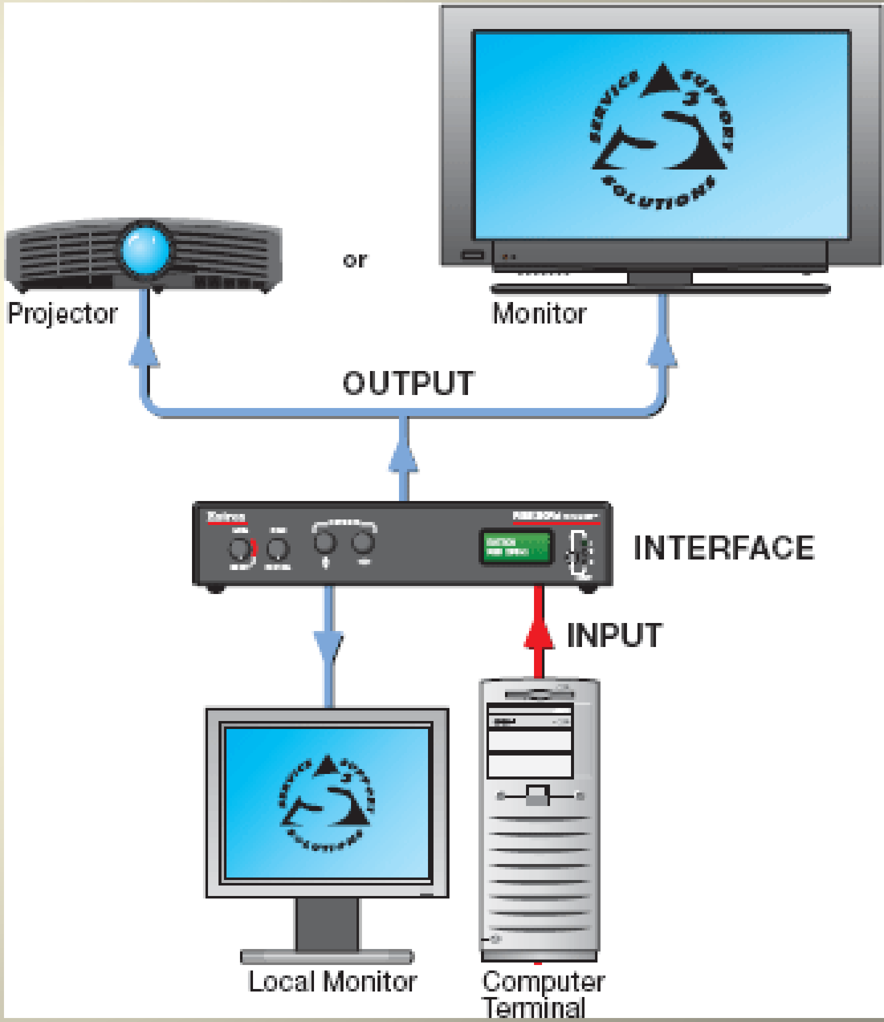


Ref: Software Engineering, I. Sommerville, 10th Edition

Object Oriented Design

5. Specify Interfaces

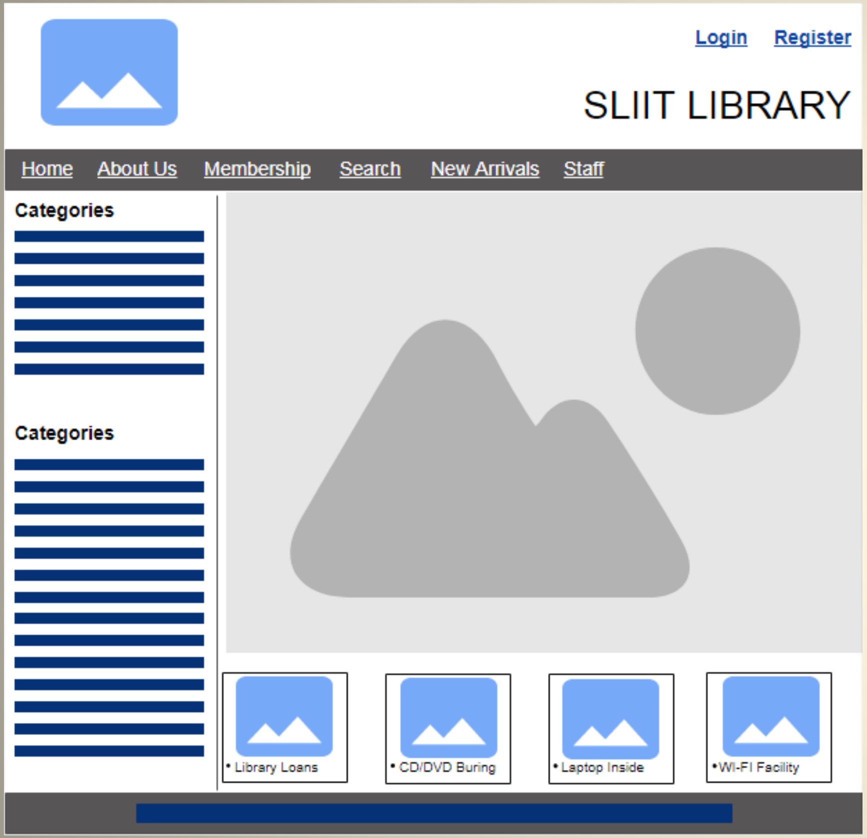
- Interfaces can be
  - devices
  - software
- The collection of all the inputs and outputs of a system defines its *interface*.

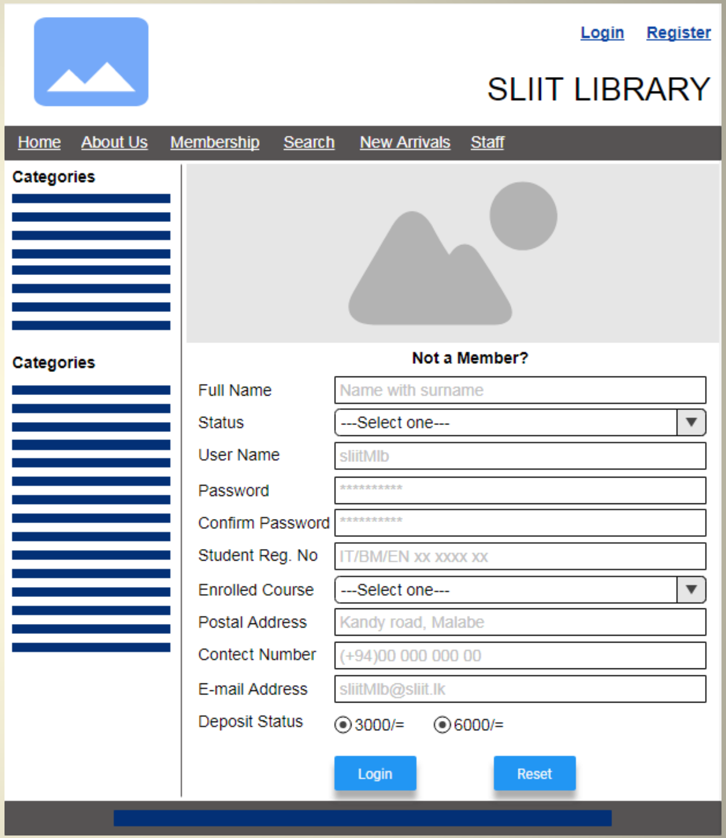


Activity

- Draw a wireframe for the SLIIT Library System

Activity Answer



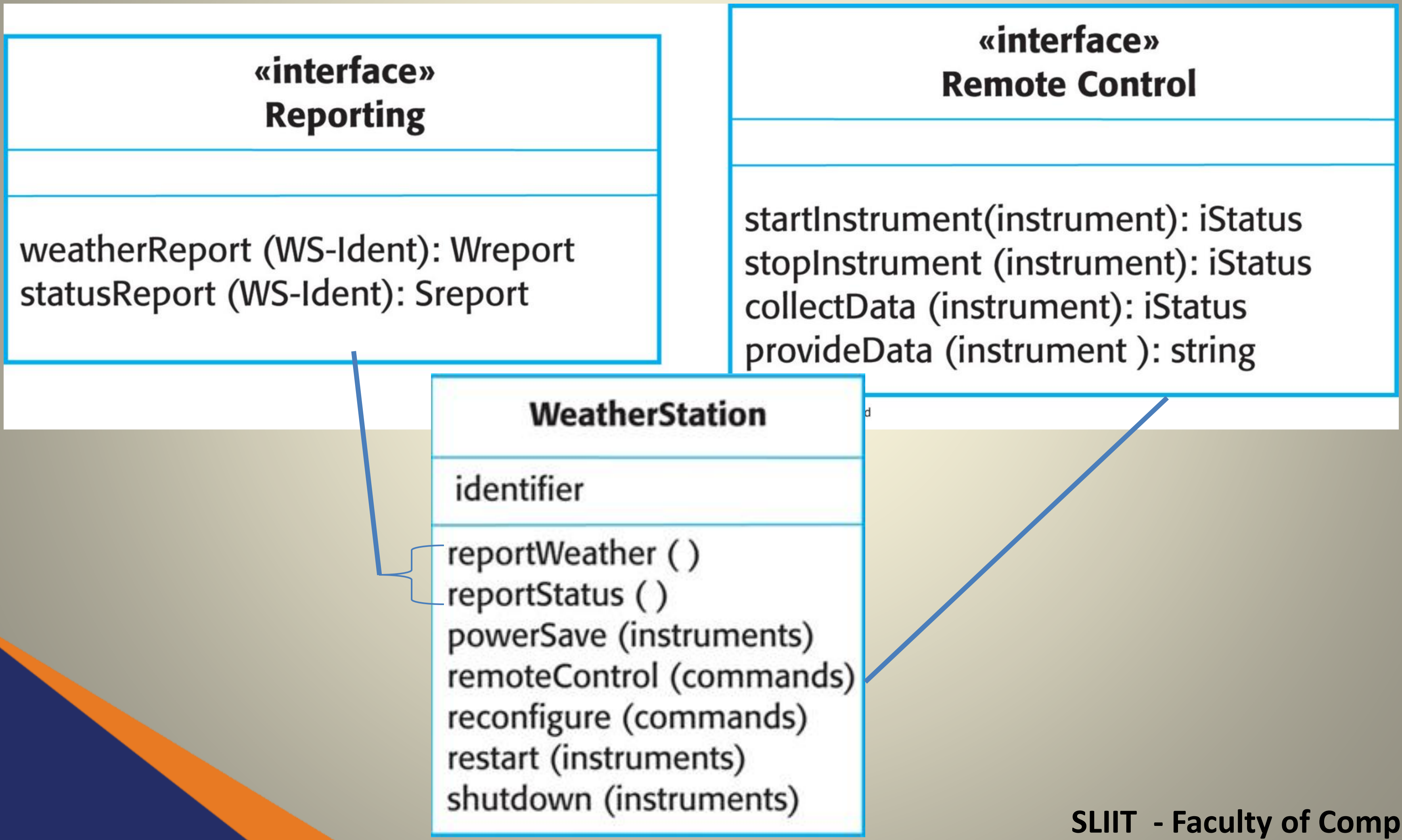


Not a Member?

Full Name	Name with surname
Status	---Select one---
User Name	sliitMlb
Password	**********
Confirm Password	**********
Student Reg. No	IT/BM/EN xx xxxx xx
Enrolled Course	---Select one---
Postal Address	Kandy road, Malabe
Contect Number	(+94)00 000 000 00
E-mail Address	sliitMlb@sliit.lk
Deposit Status	◉ 3000/= ◉ 6000/=

Object Oriented Design

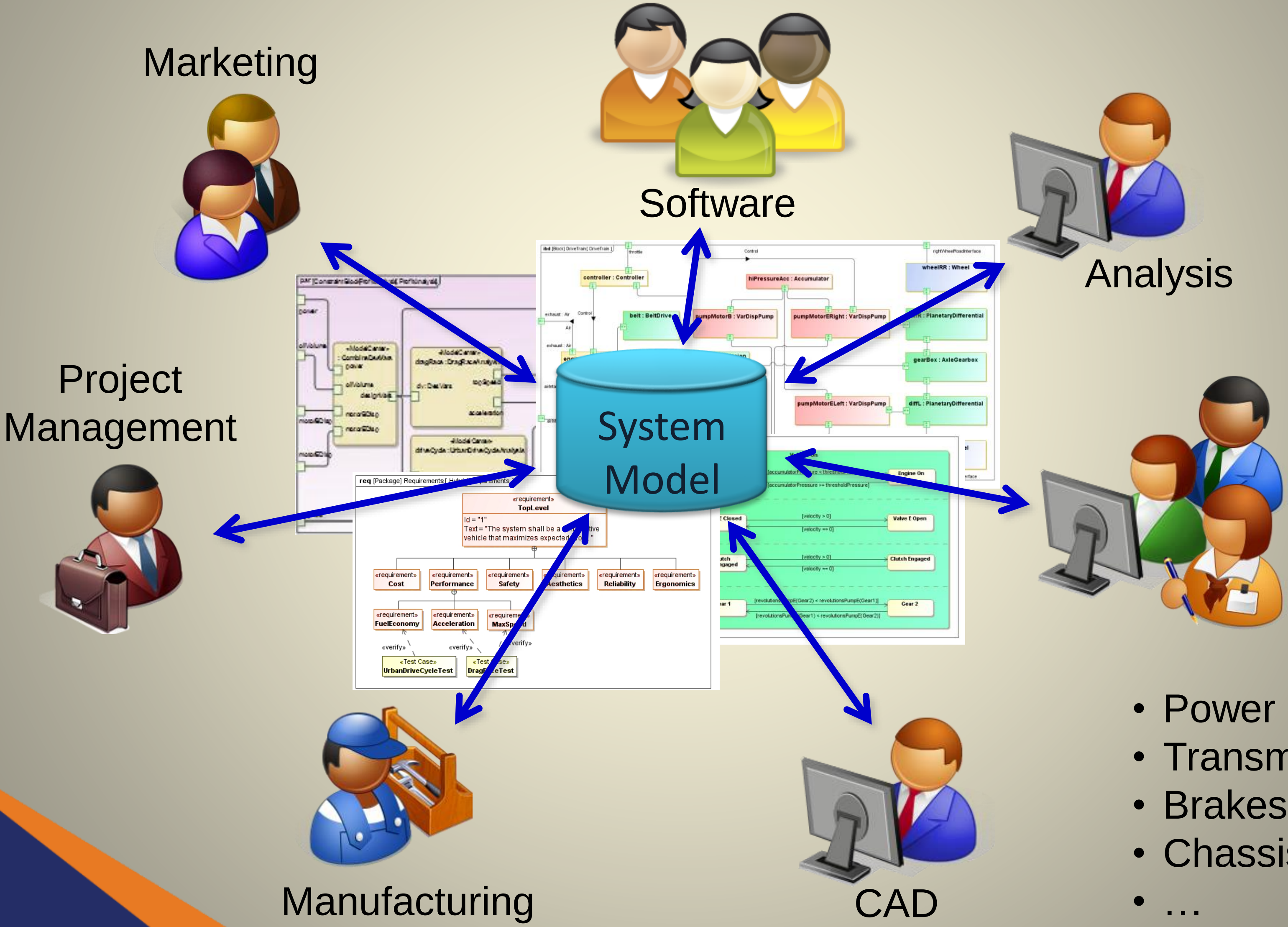
Interfaces



SysML

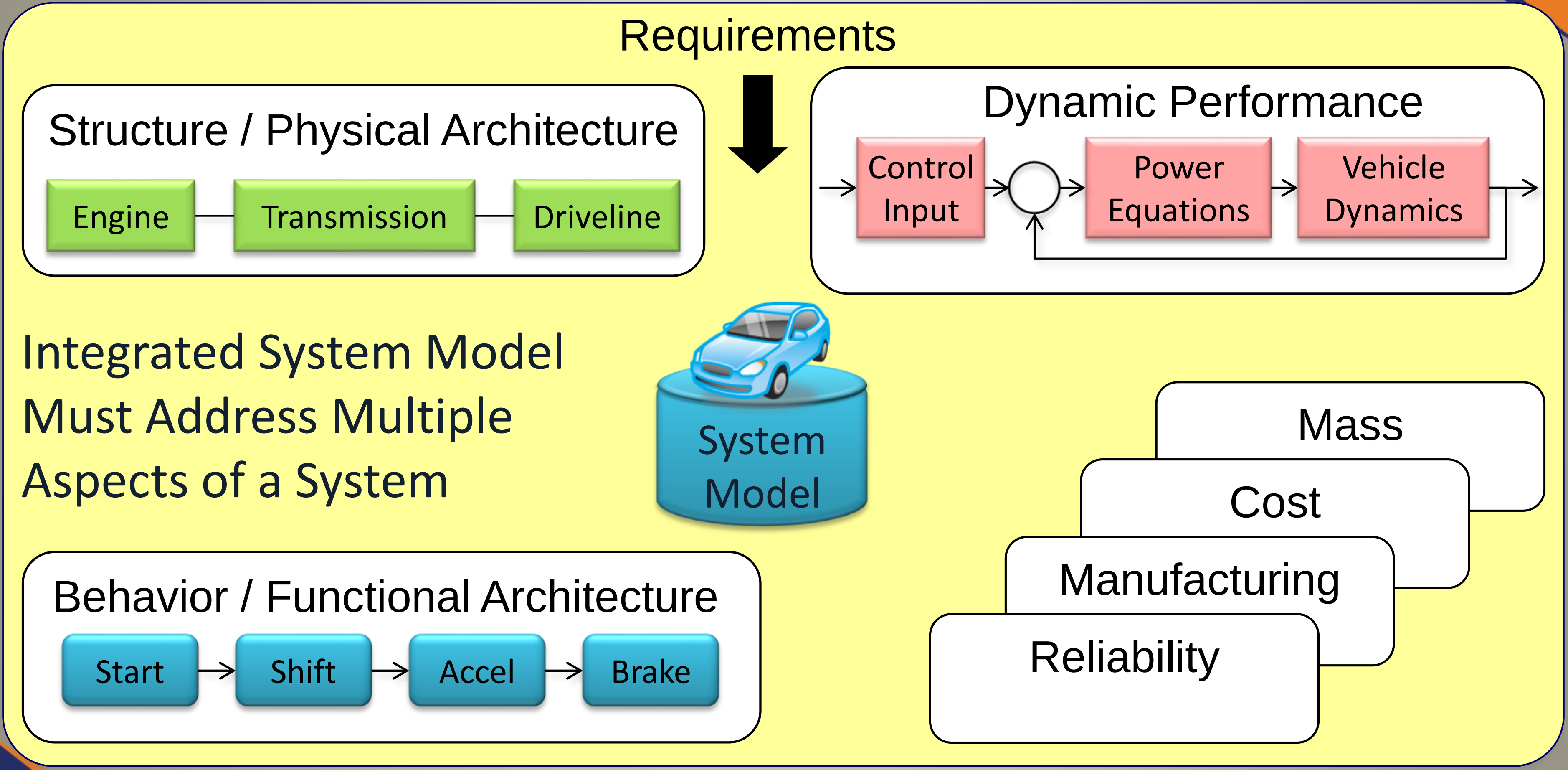
- What is SysML? A graphical modeling language in response to the UML.
  - It is a UML Profile that represents a subset of UML 2 with extensions.
- Supports the specification, analysis, design, verification and validation of systems that include hardware, software, data, personnel, procedures, and facilities.

SysML



- Power plant
- Transmission
- Brakes
- Chassis
- …

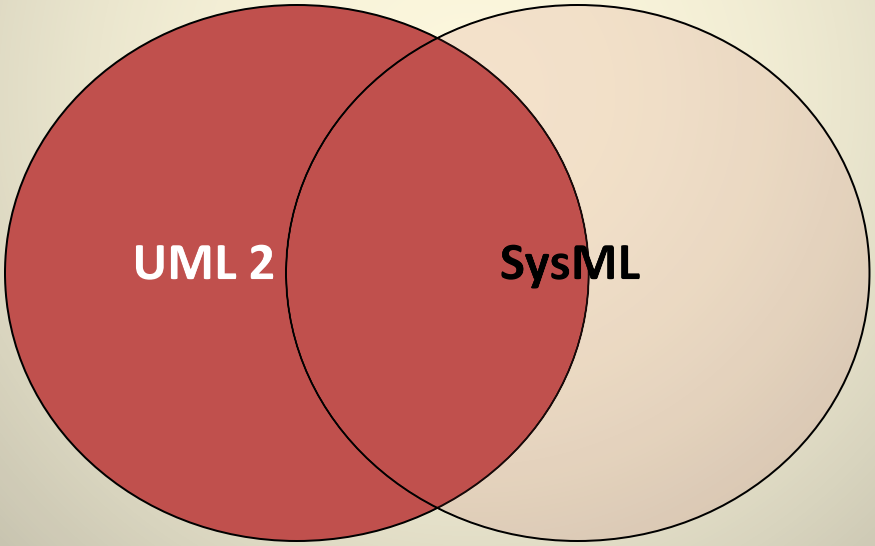
SysML



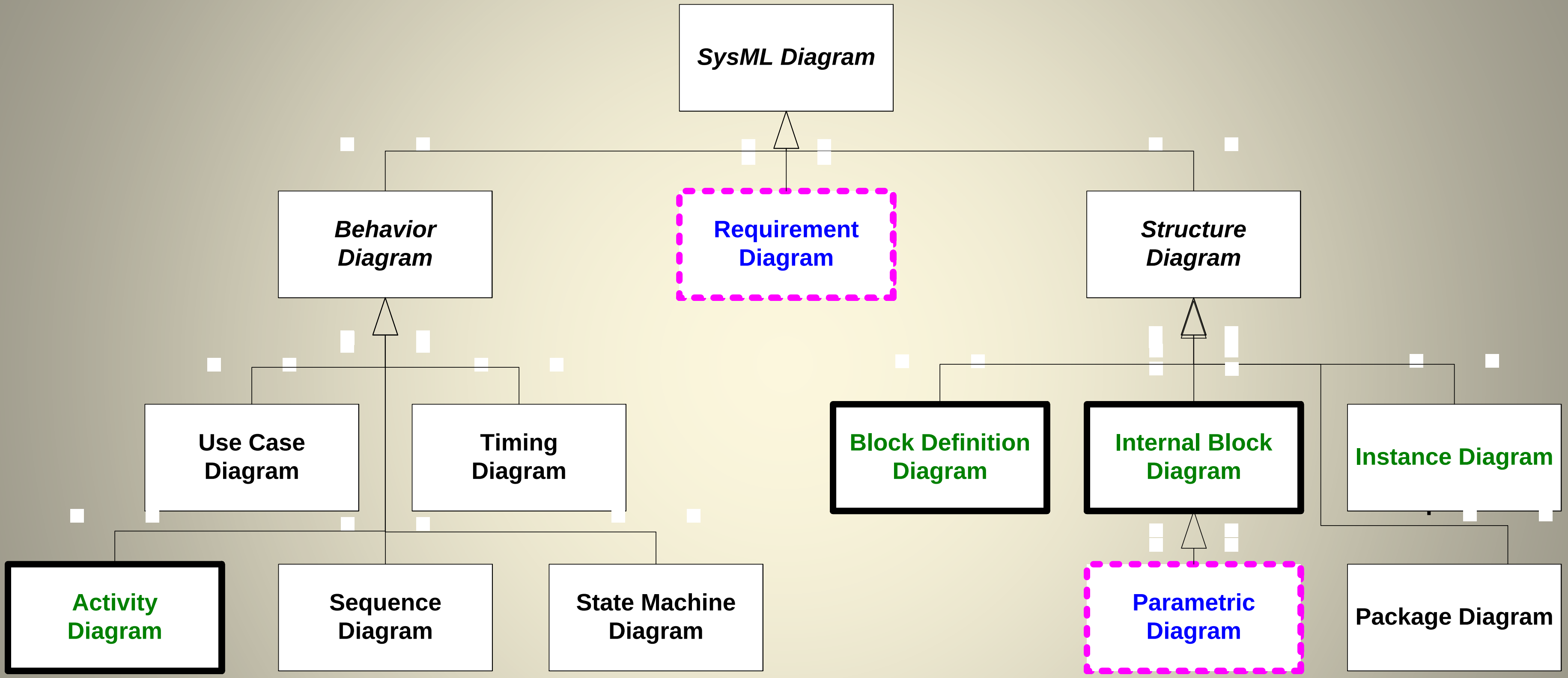
Models are more formal, complete & semantically rich

(Adapted from OMG SysML Tutorial)

Relationship shared by the SysML and UML Standards



- Package hierarchy in SysML



☐	Same as UML 2
■	Modified from UML 2
⬚	New diagram type

References

- Software Engineering – 10th Edition by Ian Sommerville, Chapter 7
- https://modeling-languages.com/#
- http://www.omgsysml.org/
- http://www.omg.org/