Software Design

Year 1 Semester 2

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

Design of small and large systems

Ref: "Software Architecture and UML" Grady Booch
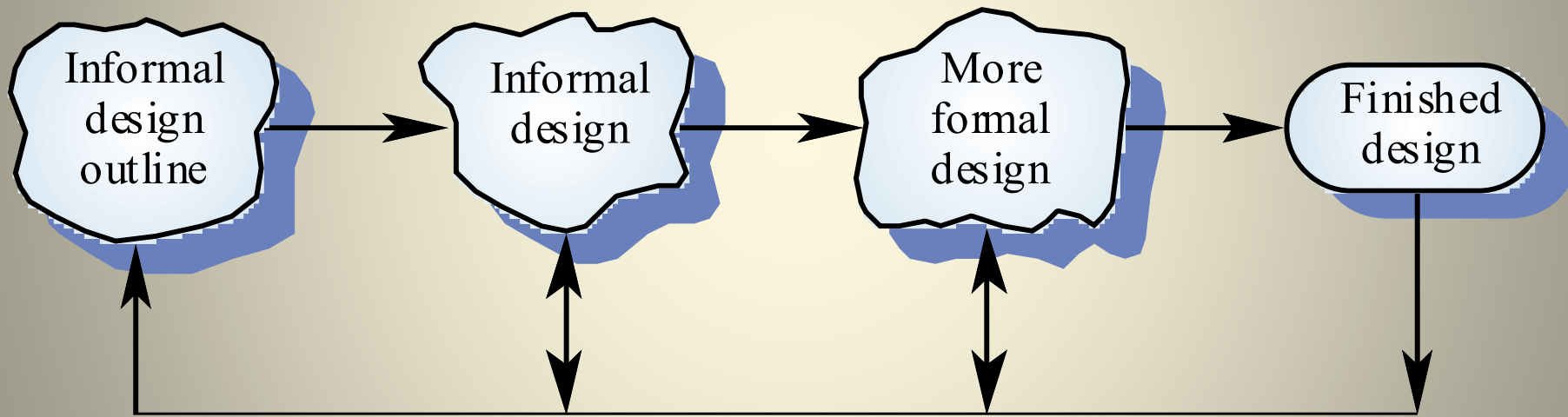
Importance of design

- 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

From informal to formal design

```
Informal
design     →    Informal    →    More          →    Finished
outline         design            formal             design
                                  design
```

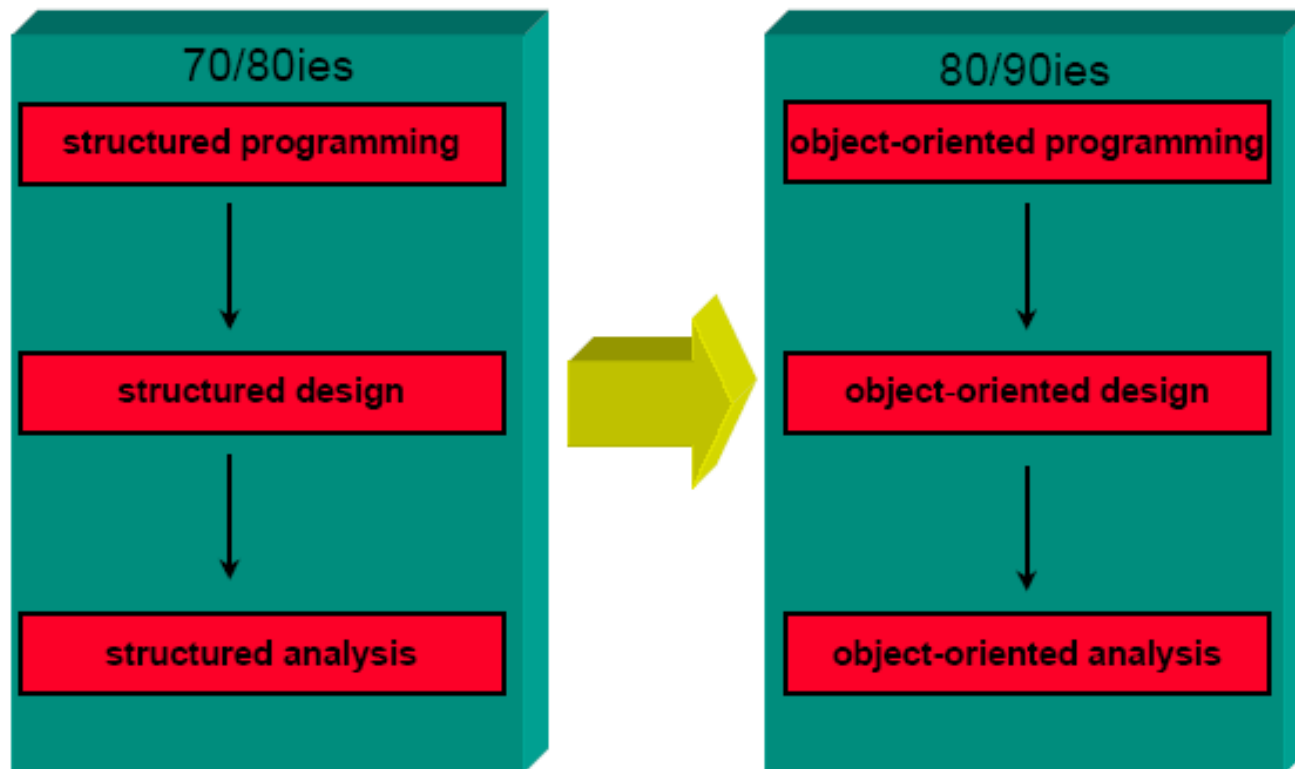Ref: Software Engineering, I. Sommerville

Software Design methods

- Function oriented software design
- Object oriented software design

Software Design
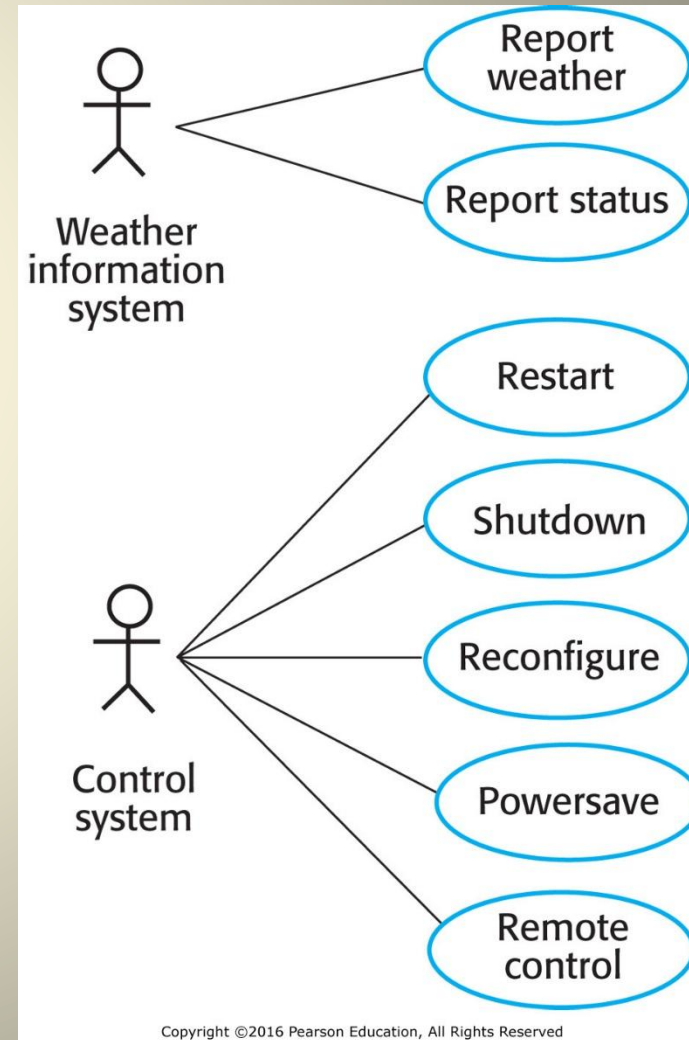
Object Oriented Design

Object oriented software design

Evolution of OO Development Methods

70/80ies	80/90ies
structured programming	object-oriented programming
↓	↓
structured design	object-oriented design
↓	↓
structured analysis	object-oriented analysis

OOA&D © J.W. Schmidt, F. Matthes, TU Hamburg-Harburg

3.0.2

Object Oriented Design

1. Understand System and interactions
   - Use case Diagrams
   - Activity Diagrams
   - Use case Scenarios
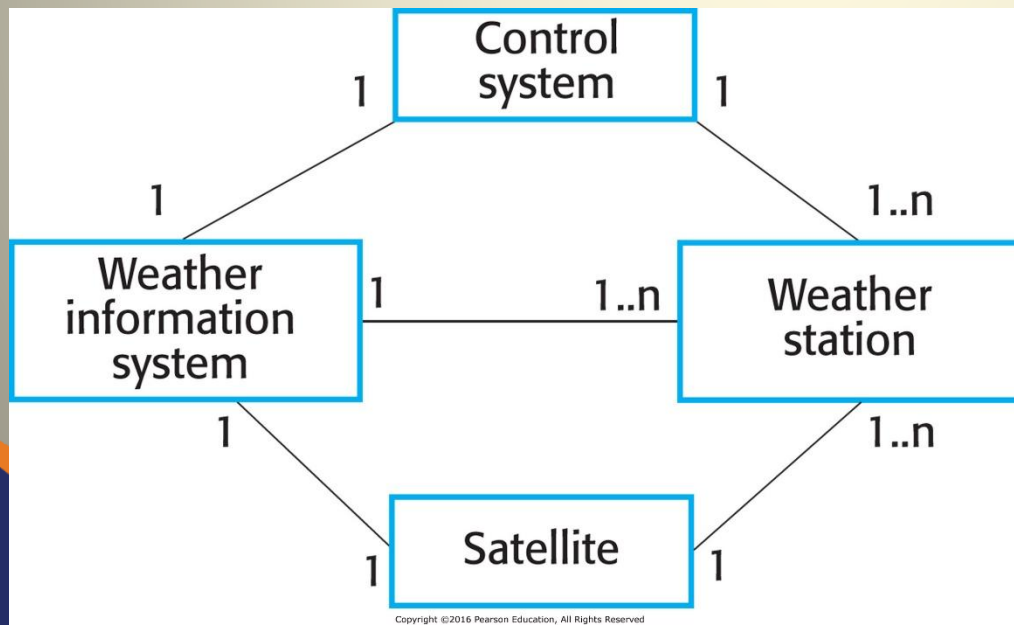
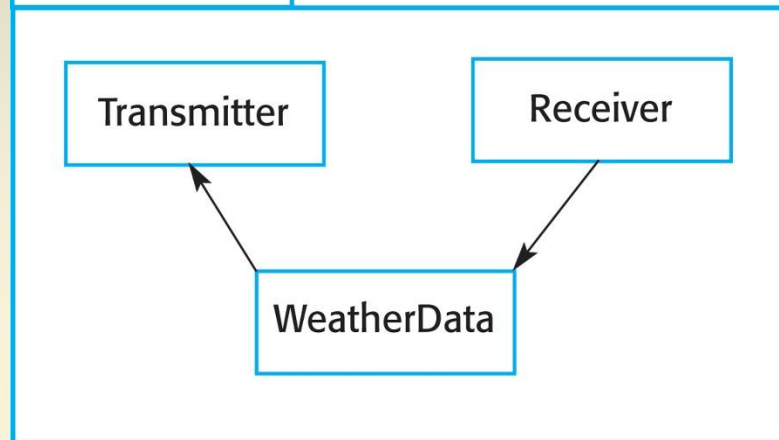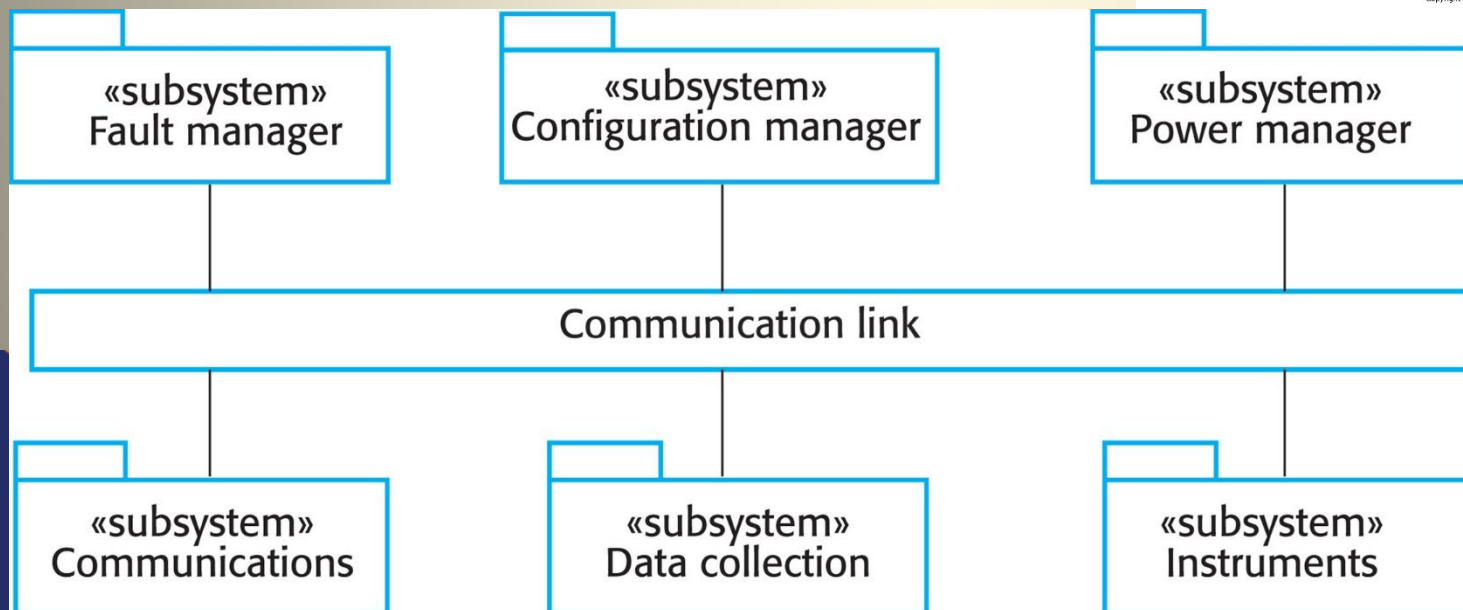# Object Oriented Design

2. Design System Architecture

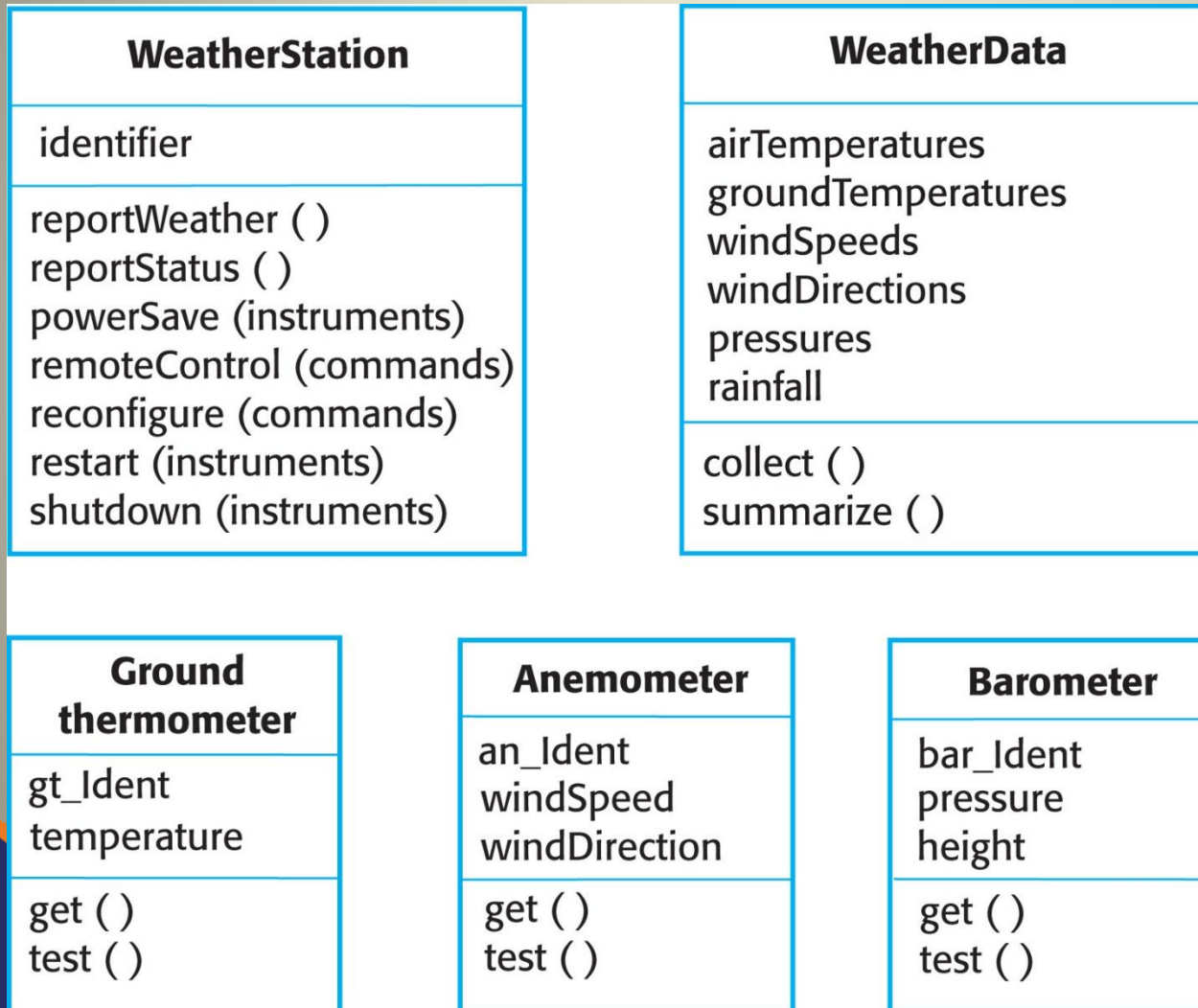– Subsystems and communication between the subsystems.

Data collection

Transmitter   Receiver

WeatherData

«subsystem» Fault manager

«subsystem» Configuration manager

«subsystem» Power manager

Communication link

«subsystem» Communications

«subsystem» Data collection

«subsystem» Instruments

Object Oriented Design

3. Identify main classes and objects

WeatherStation
identifier
reportWeather ( ) reportStatus ( ) powerSave (instruments) remoteControl (commands) reconfigure (commands) restart (instruments) shutdown (instruments)

WeatherData
airTemperatures groundTemperatures windSpeeds windDirections pressures rainfall
collect ( ) summarize ( )

Ground thermometer
gt_Ident temperature
get ( ) test ( )

Anemometer
an_Ident windSpeed windDirection
get ( ) test ( )

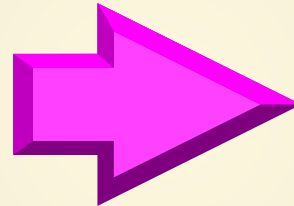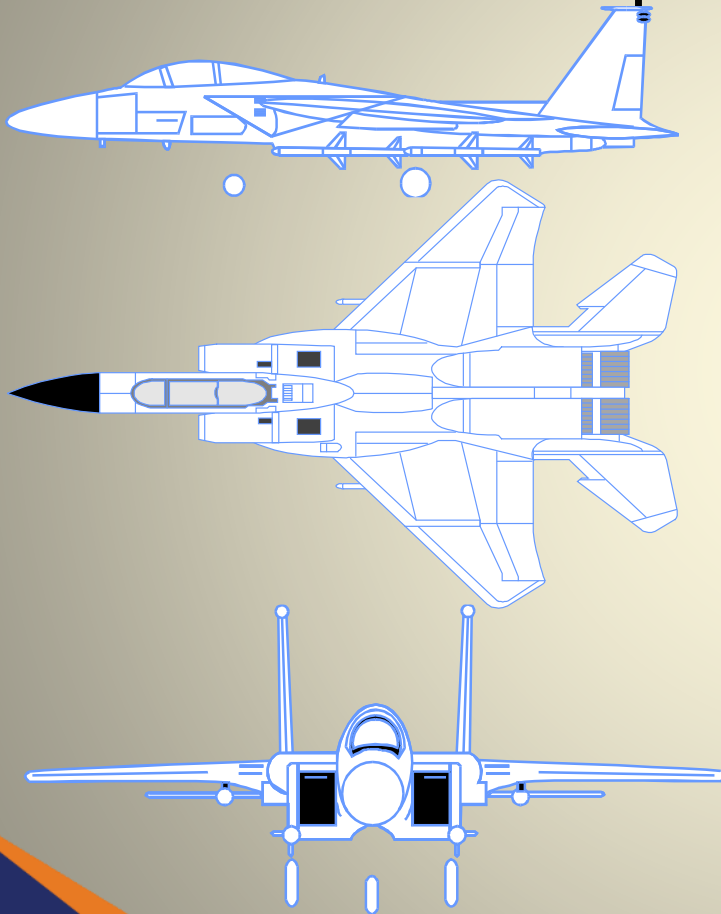| Barometer |
|---|
| bar_Ident
pressure
height |
| get ( )
test ( ) |

SLIIT - Faculty of Computing

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?
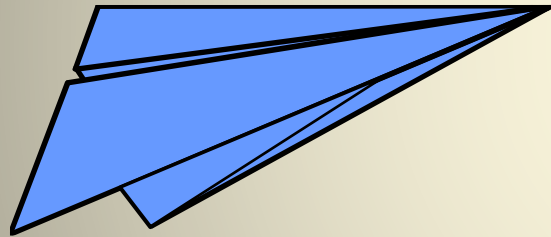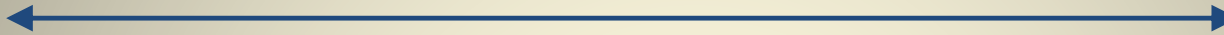
- A model is a simplification of reality.

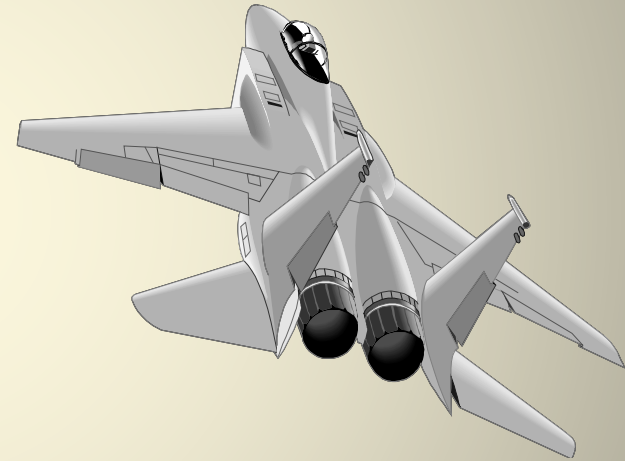Ref: Fundamentals of Visual Modeling with UML

The Importance of Modeling

Less Important   More Important

$\longleftrightarrow$

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/#
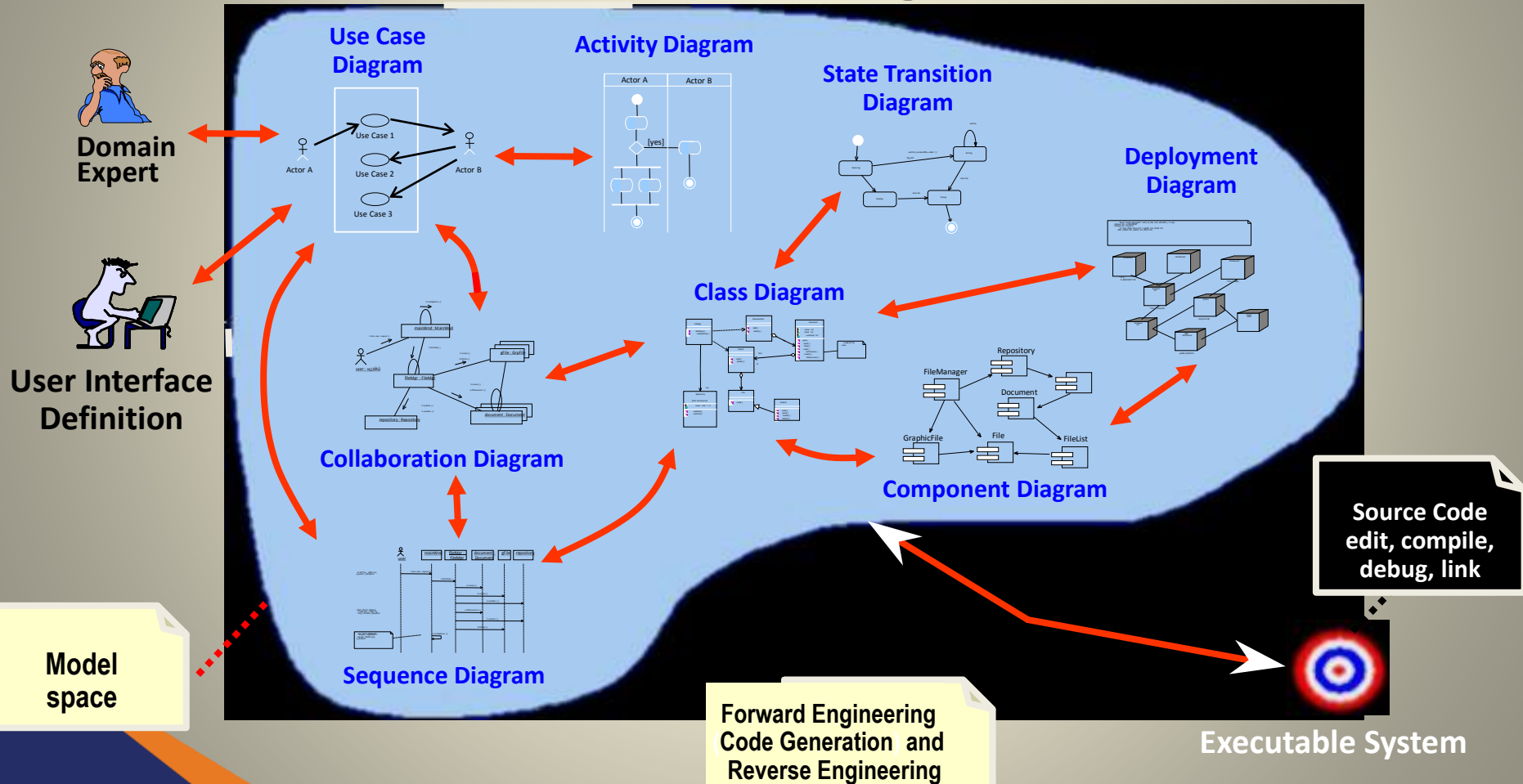
What Is the UML?

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

  the artifacts of a software-intensive system.

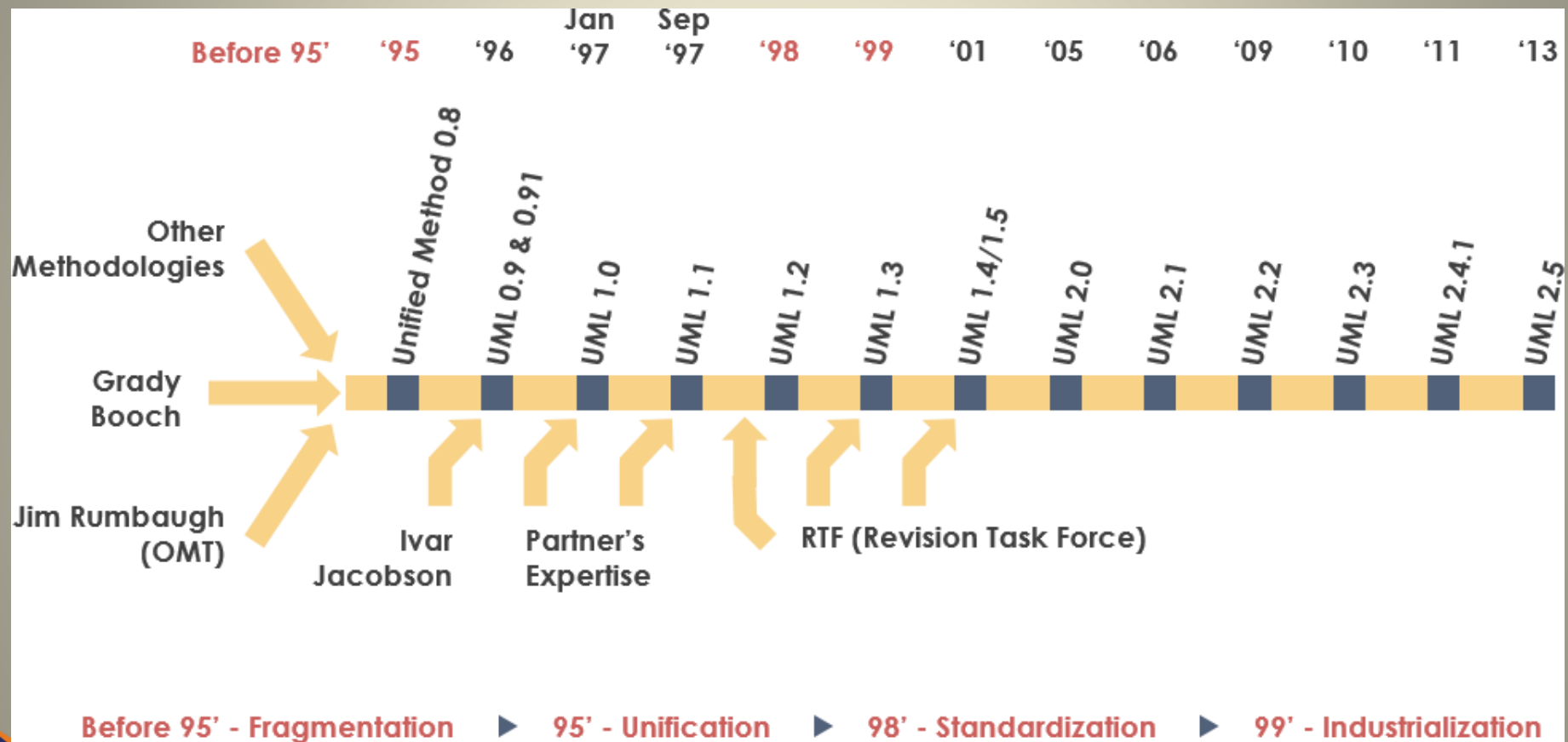- Out of the above, **SPM** and **SE** modules specially focus on using *UML as a language for documenting*.
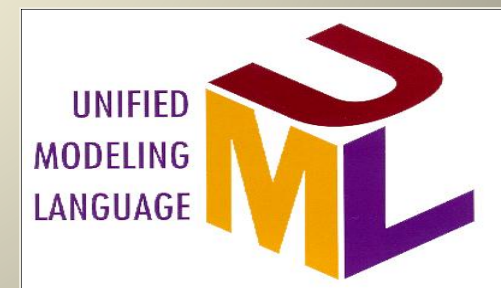
Ref: Fundamentals of Visual Modeling with UML
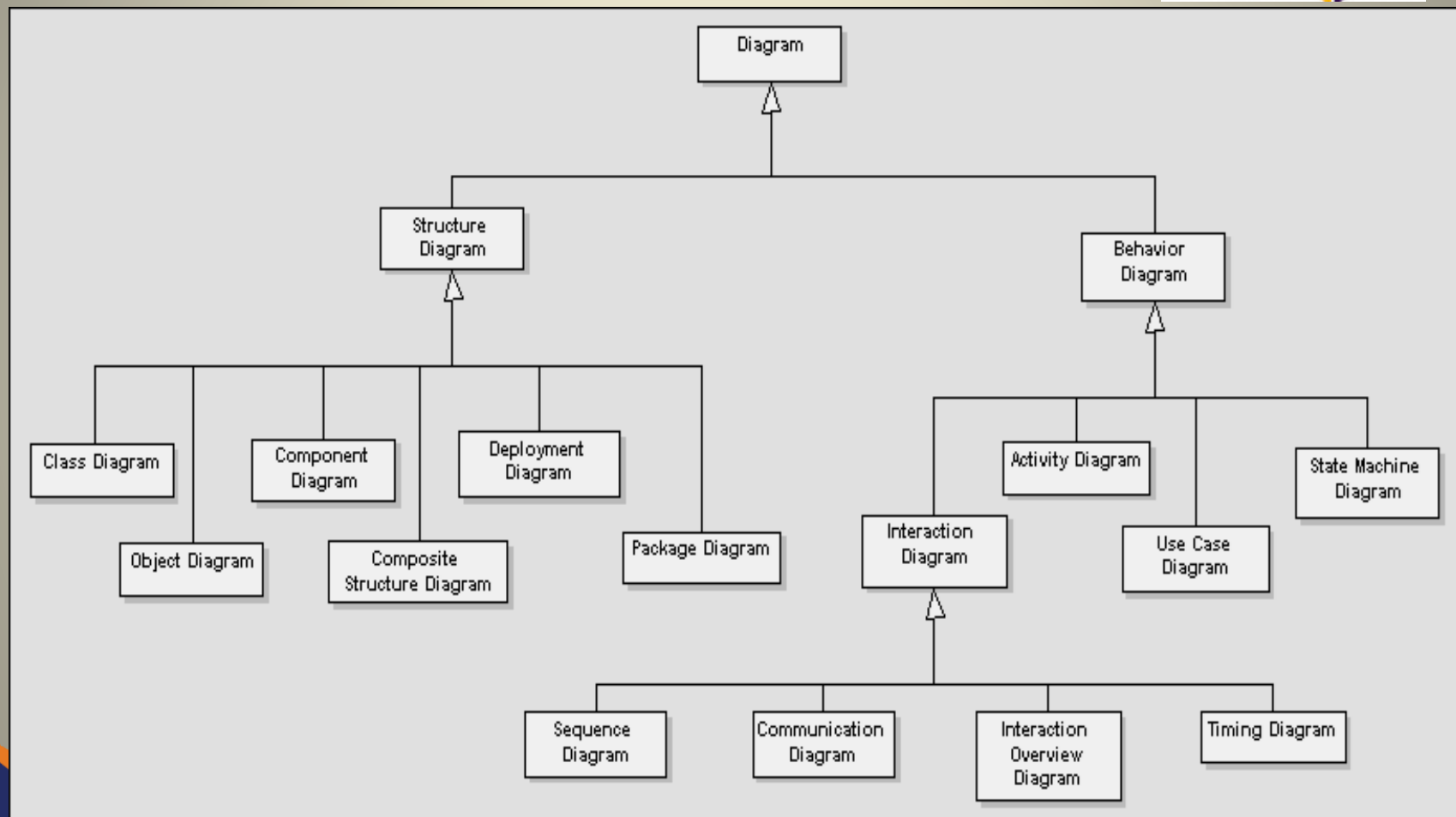
The UML Is a Language for Documenting

Ref: Fundamentals of Visual Modeling with UML

UML - History

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
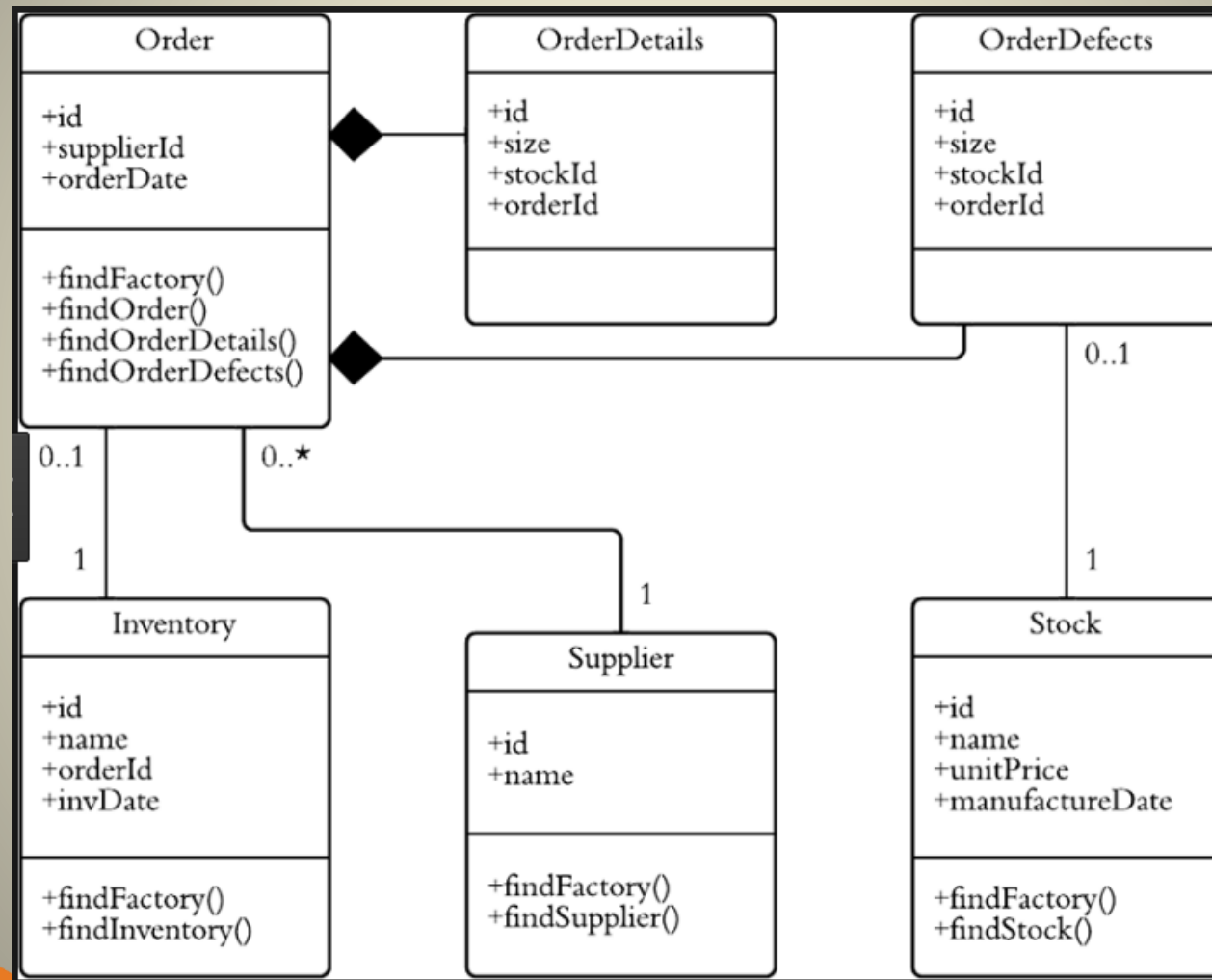
Activity

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

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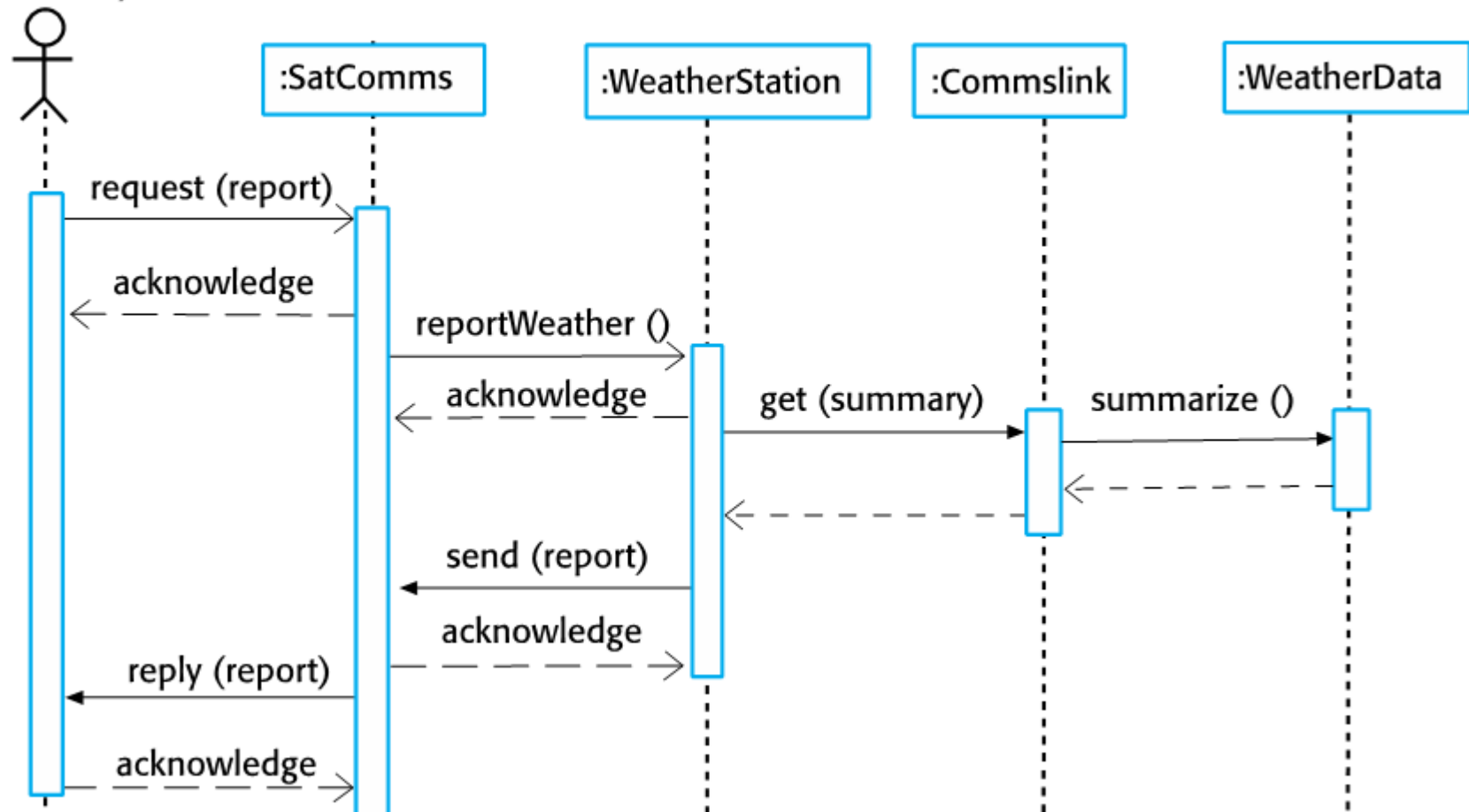
Ref: Software Engineering, I. Sommerville, 10th Edition

Object Oriented Design

5. Specify Interfaces

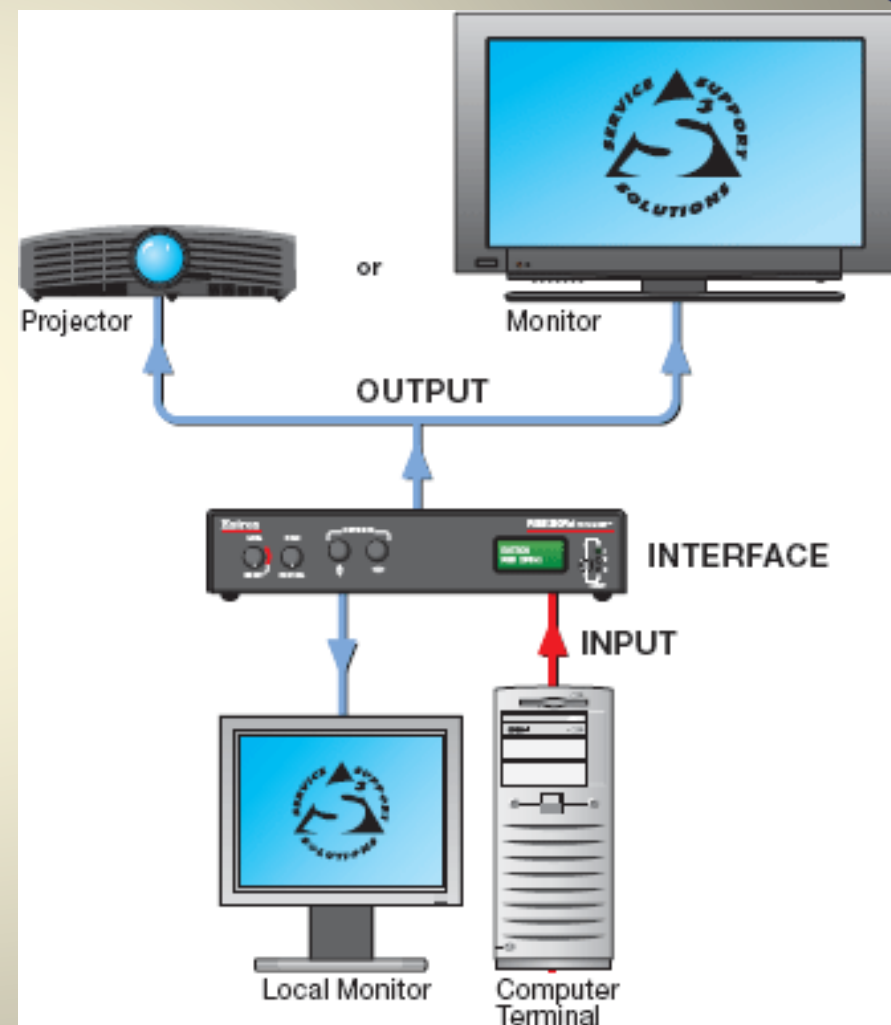- Interfaces can be
  - devices
  - software

- The collection of all the inputs and outputs of a system defines its *interface*.

Object Oriented Design

3. Specify Interfaces

«interface»
Reporting

weatherReport (WS-Ident): Wreport
statusReport (WS-Ident): Sreport

«interface»
Remote Control

startInstrument(instrument): iStatus
stopInstrument (instrument): iStatus
collectData (instrument): iStatus
provideData (instrument): string

WeatherStation

identifier

reportWeather ( )
reportStatus ( )
powerSave (instruments)
remoteControl (commands)
reconfigure (commands)
restart (instruments)
shutdown (instruments)

SysML

- What is SysML? A graphical modeling language in response to the UML.

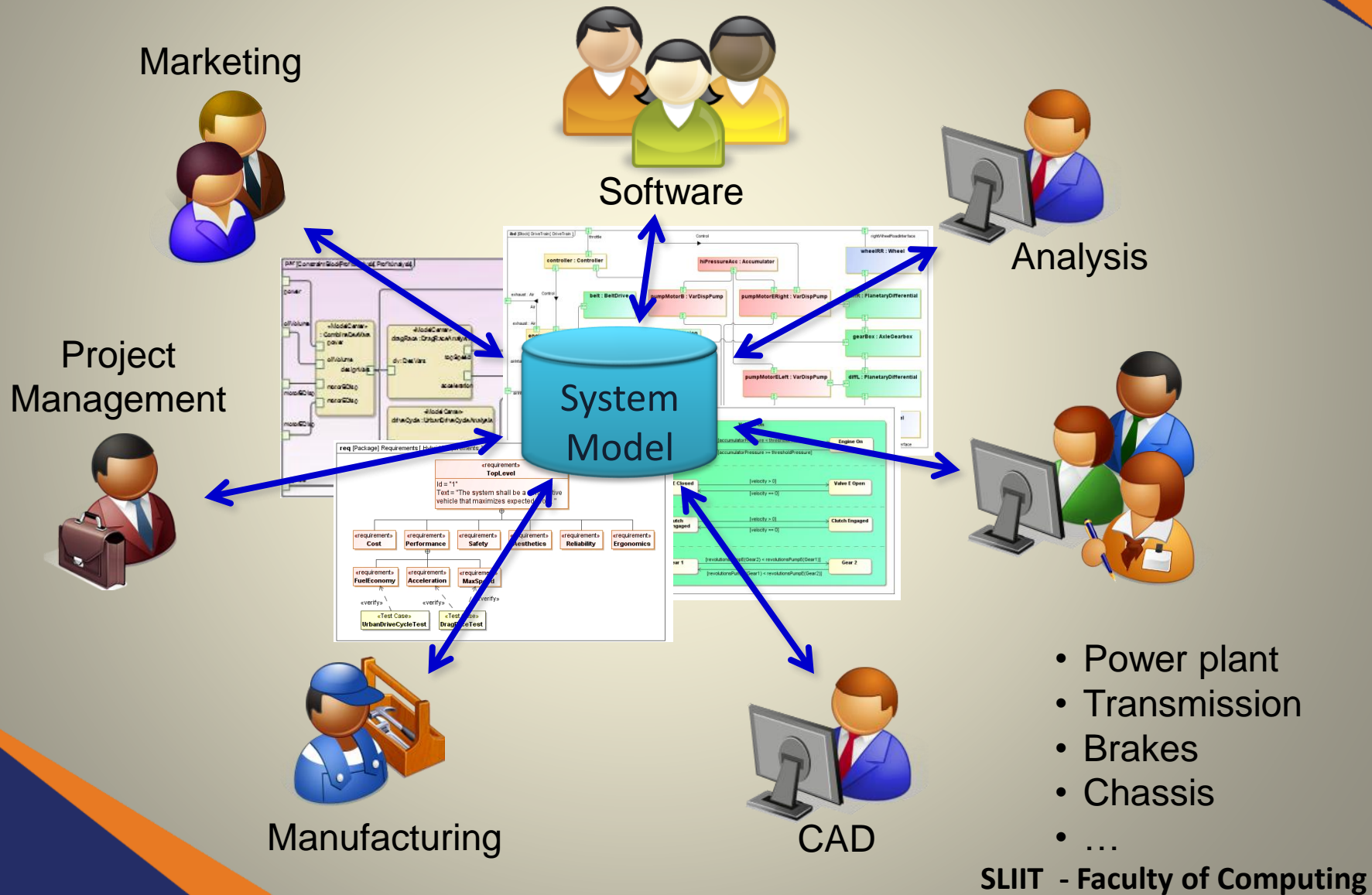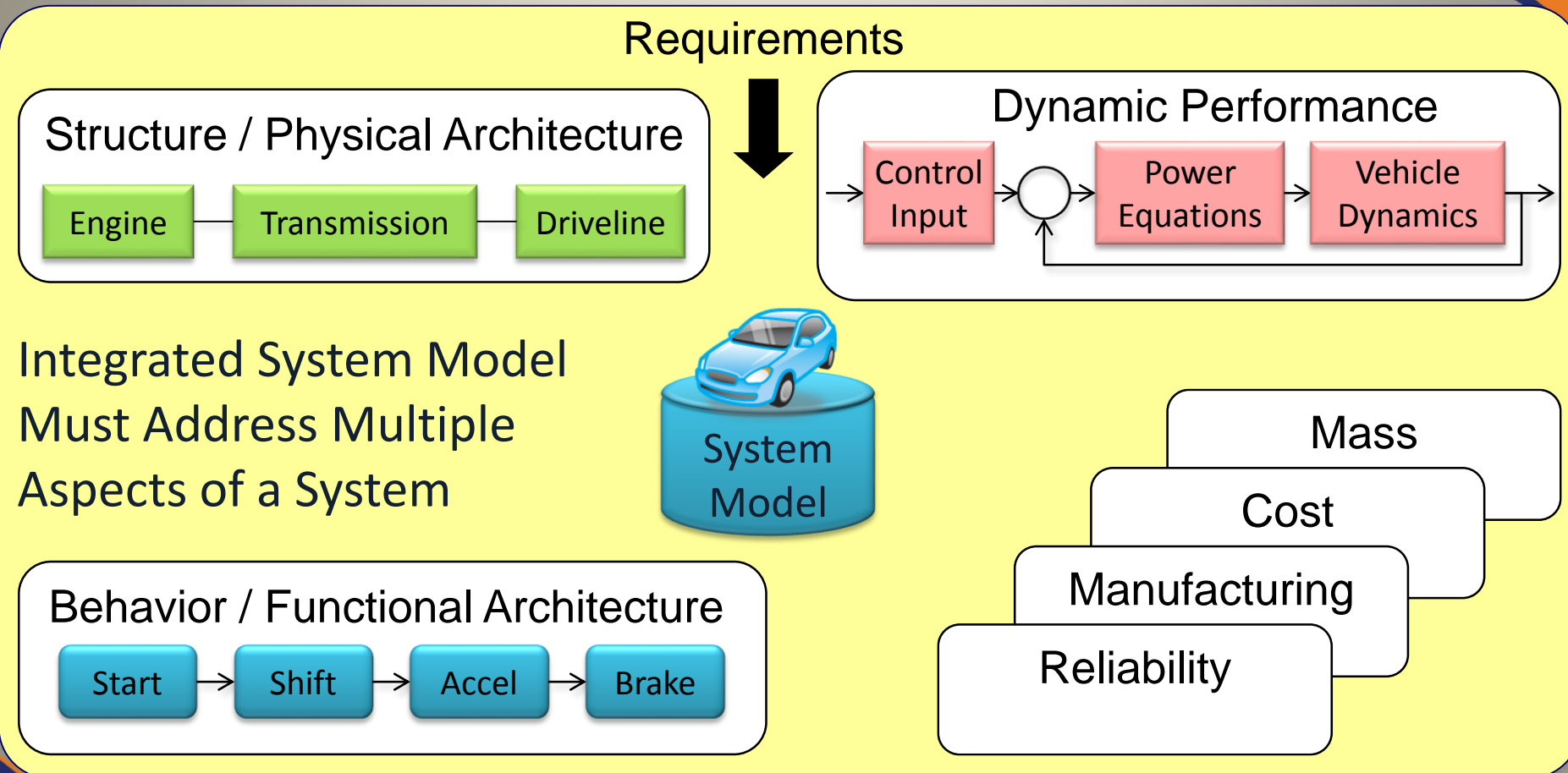 - It is a UML Profile that represents a subset of UML 2 with extensions.

- Supports the specification, analysis, design, verification and validation of systems that include hardware, software, data, personnel, procedures, and facilities.

SysML

SysML

Requirements

Structure / Physical Architecture

Engine	Transmission	Driveline

Dynamic Performance

Control Input	Power Equations	Vehicle Dynamics

Integrated System Model
Must Address Multiple
Aspects of a System

System Model

Behavior / Functional Architecture

Start → Shift → Accel → Brake

Mass

Cost

Manufacturing

Reliability

Models are more formal, complete & semantically rich

(Adapted from OMG SysML Tutorial)

Relationship shared by the SysML and UML Standards
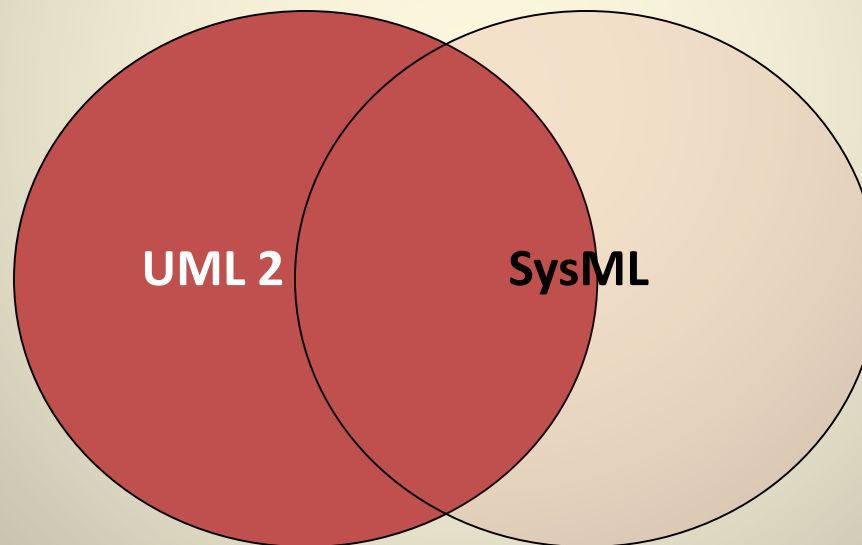
- # Package hierarchy in SysML


```
                        ┌─────────────────┐
                        │  SysML Diagram  │
                        └────────┬────────┘
```

SysML Diagram

Behavior Diagram

Requirement Diagram

Structure Diagram

Use Case Diagram

Timing Diagram

Block Definition Diagram

Internal Block Diagram

Instance Diagram

Activity Diagram

Sequence Diagram

State Machine Diagram

Parametric Diagram

Package Diagram
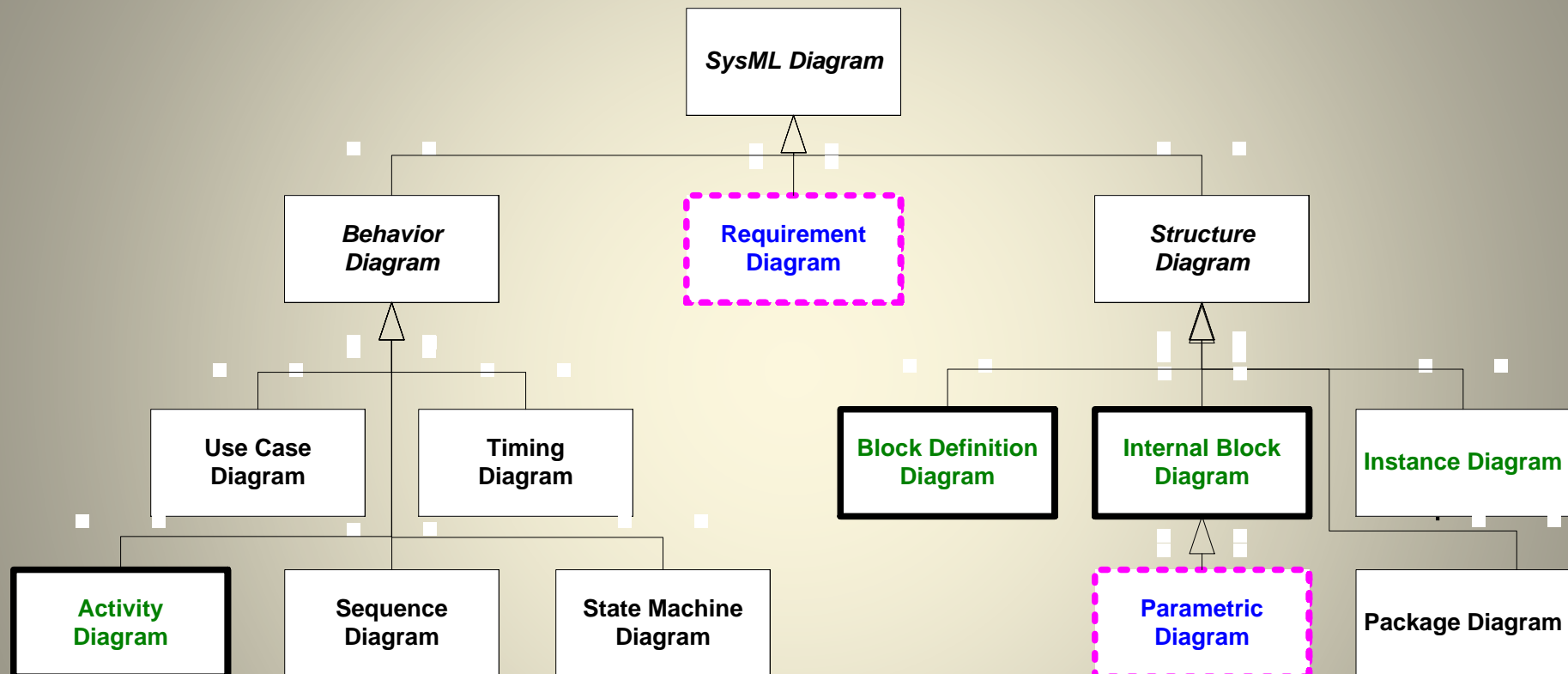
Same as UML 2

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/