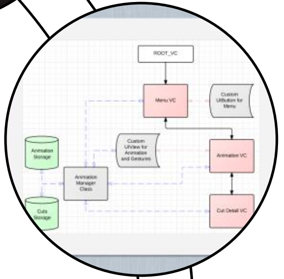
INTRODUCTION TO SOFTWARE DESIGN & DEVELOPMENT

Dr. Issarapong Khuankrue

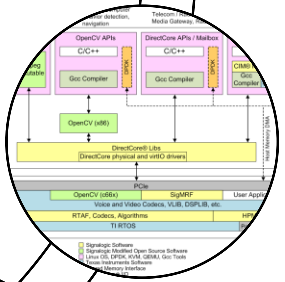
CONTENTS

Keywords

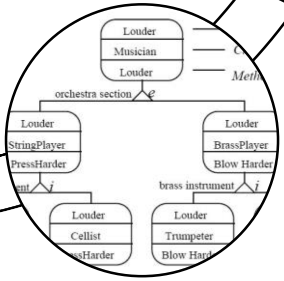
Software Engineering



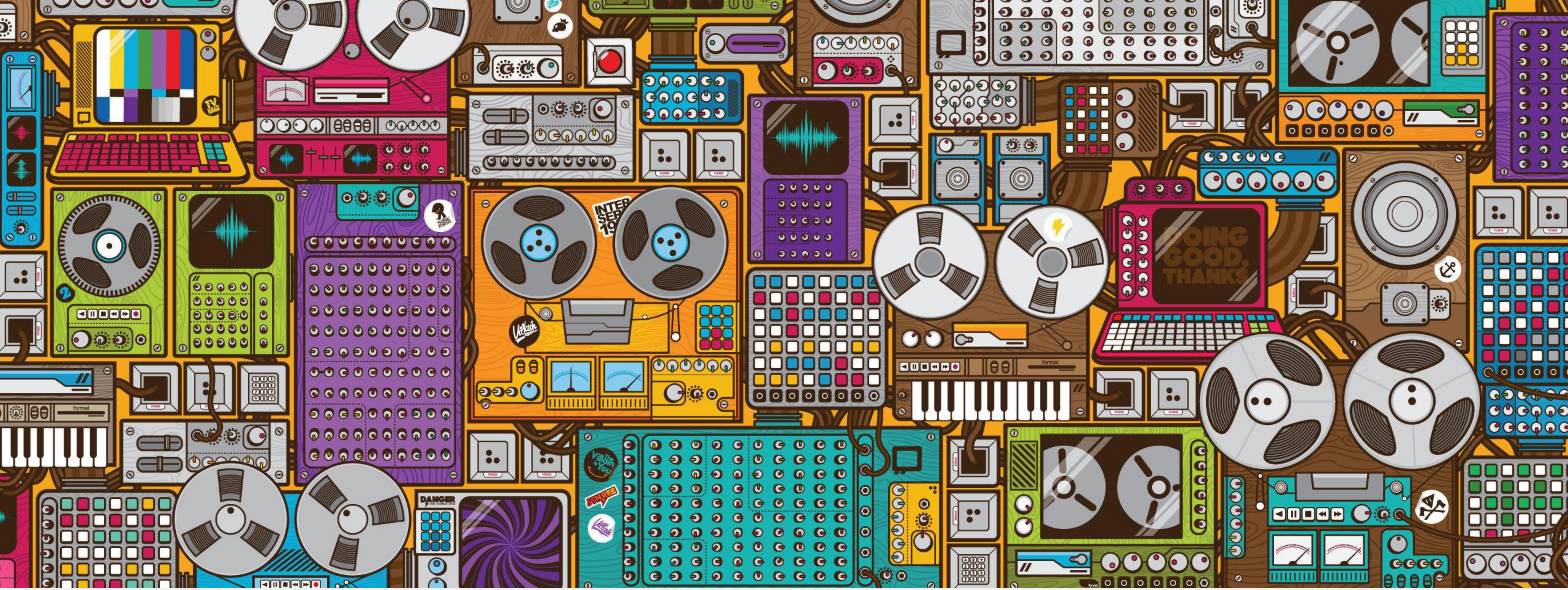
Software Design



Software Architecture

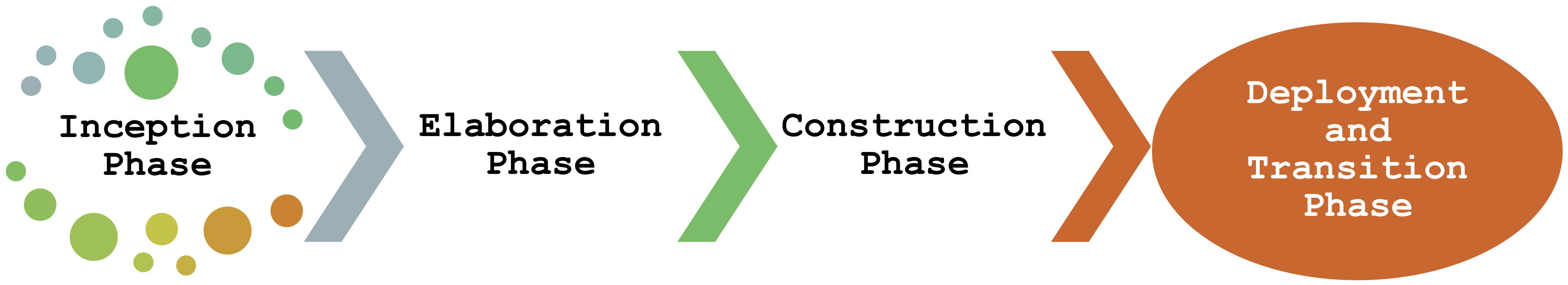


Object-oriented Model

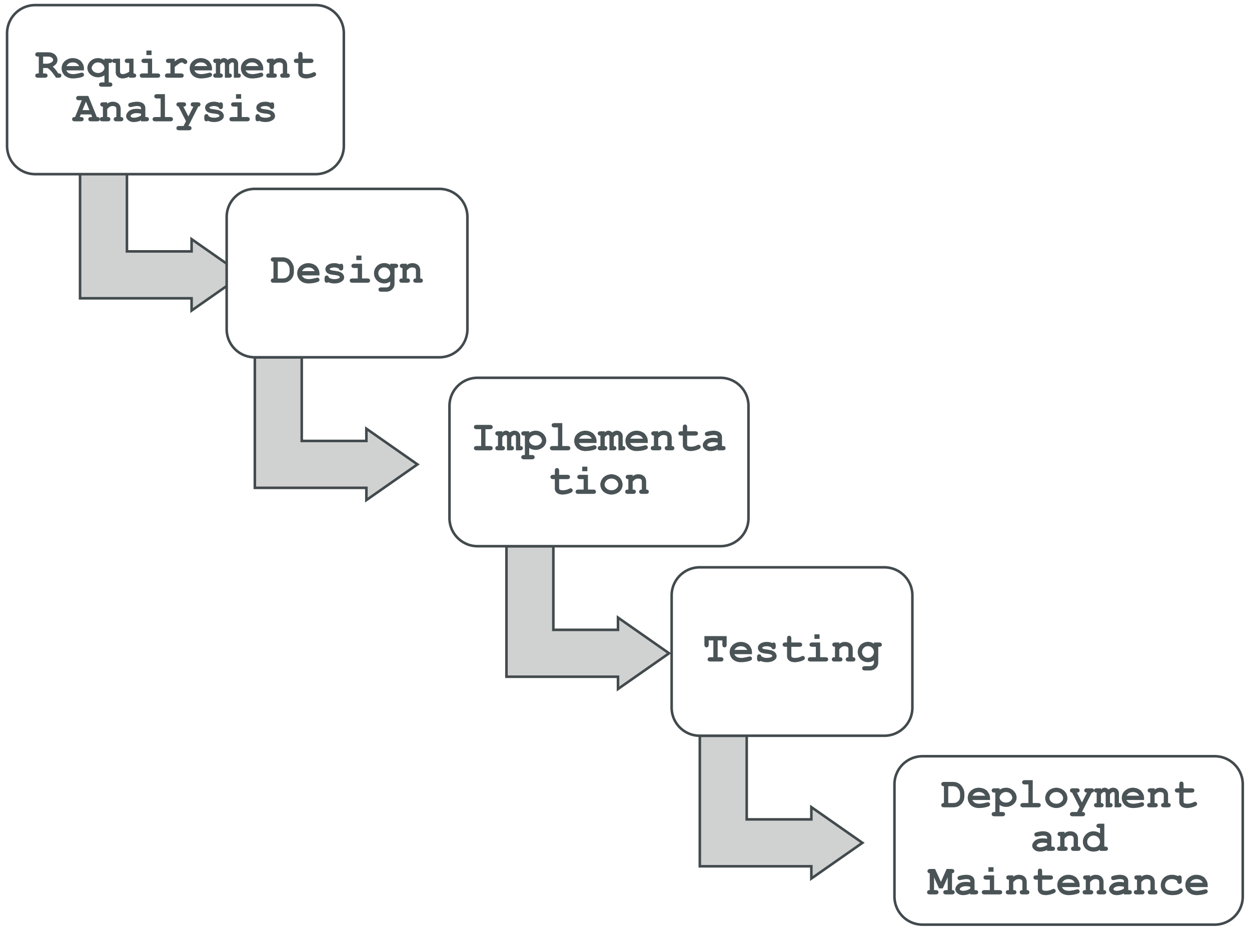


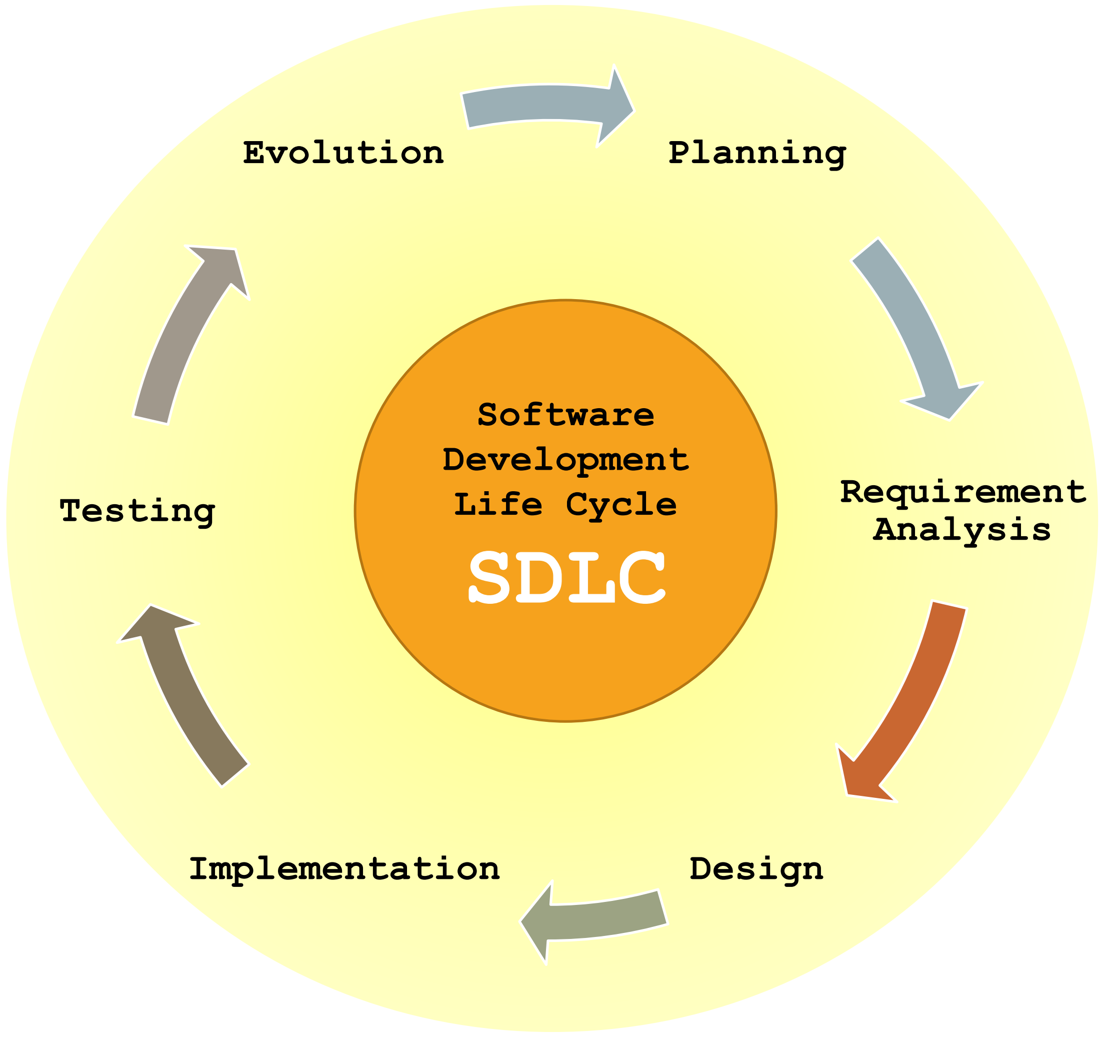
SOFTWARE PRODUCT LIFE CYCLE

SOFTWARE PRODUCT LIFE CYCLE (MANAGEMENT VIEW)

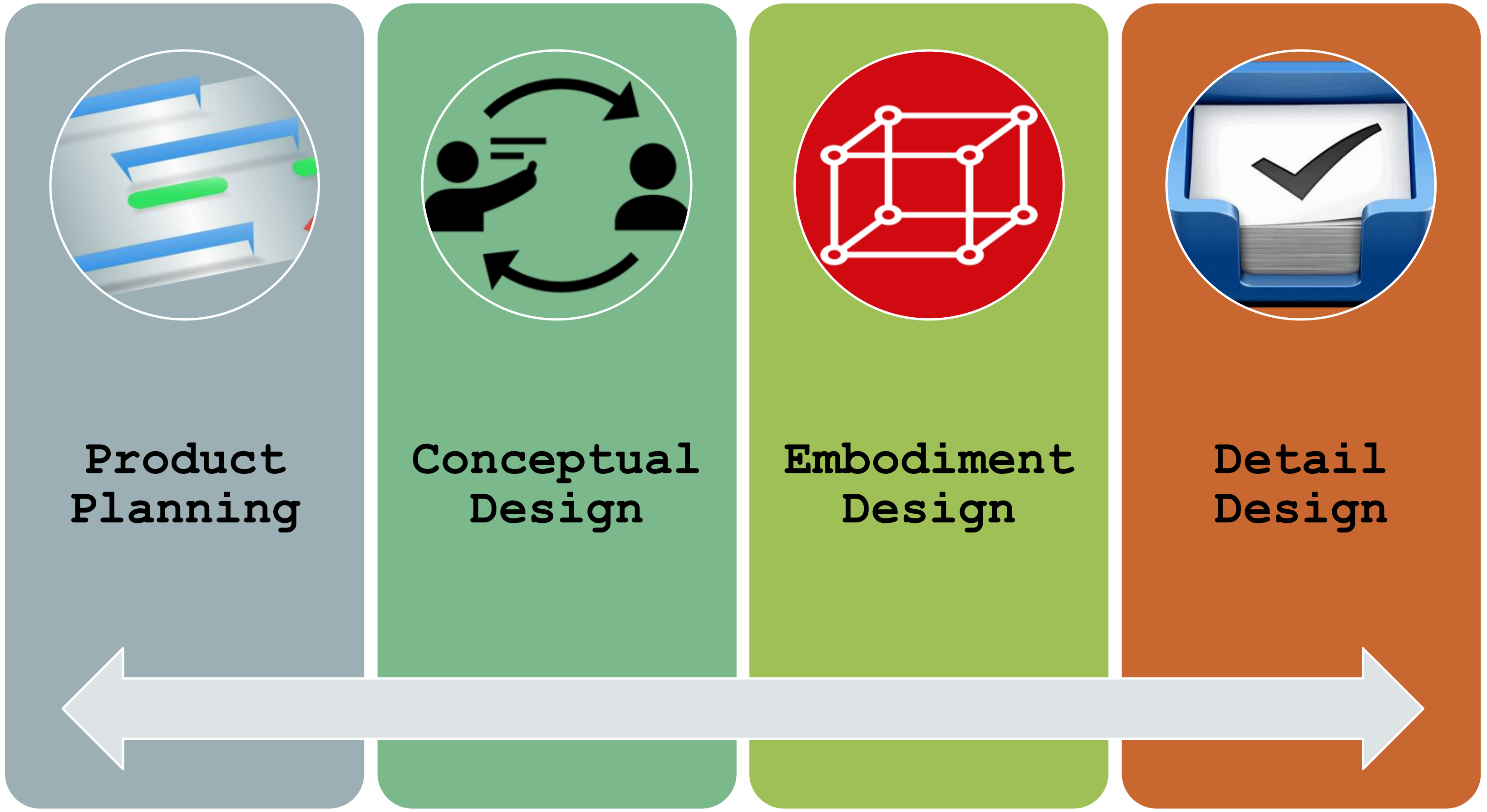


SOFTWARE PRODUCT LIFE CYCLE (ENGINEERING VIEW)

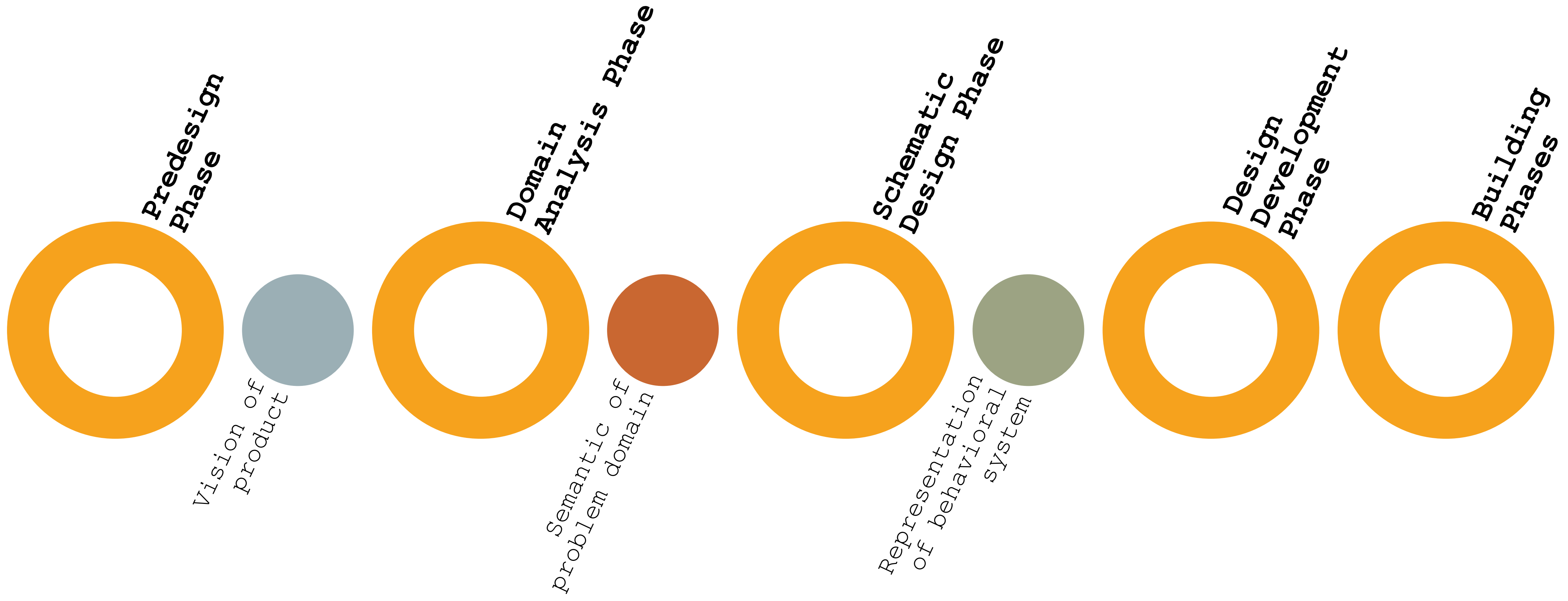




SOFTWARE PRODUCT LIFE CYCLE (ENGINEERING DESIGN VIEW)



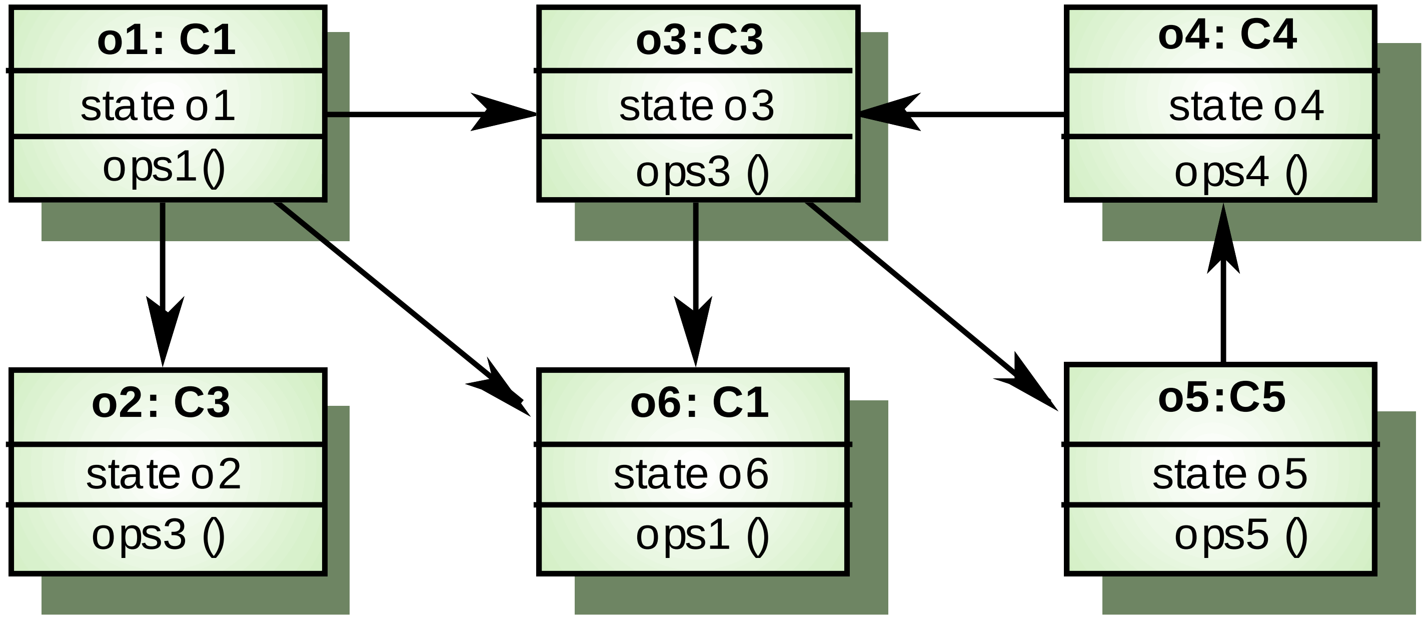
SOFTWARE PRODUCT LIFE CYCLE (ARCHITECTURAL VIEW)



OBJECT-ORIENTED MODEL

- **Characteristics**
  - **Objects** are abstractions of real-world or system entities and manage themselves.
  - Objects are independent and encapsulate state and representation information.
  - System functionality is expressed in terms of object services
  - Shared data areas are eliminated. Objects communicate by message passing
  - Objects may be distributed and may execute sequentially or in parallel

INTERACTING OBJECTS



ADVANTAGES

- **Easier maintenance.**
  - Objects may be understood as stand-alone entities.
- **Objects are appropriate reusable components.**
- For some systems, there may **be an obvious mapping from real world entities to system objects.**

OBJECT-ORIENTED DEVELOPMENT

- Object-oriented analysis, design and programming are related but distinct.
  - **Object-oriented Analysis : OOA** is concerned with developing an object model of the application domain.
  - **Object-oriented Design : OOD** is concerned with developing an object-oriented system model to implement requirements
  - **Object-oriented Programming : OOP** is concerned with realising an OOD using an OO programming language such as Java or C++

OBJECTS & OBJECT CLASSES

- **Objects** are entities in a software system which represent instances of real-world and system entities

- **Object classes** are templates for objects. They may be used to create objects
  - Object classes may inherit attributes and services from other object classes

SOFTWARE DESIGN

FUNDAMENTALS : **GENERAL DESIGN CONCEPT**

- **Definition:** Design is a problem-solving process whose objective is to find and describe a way:
  - To implement the system's **functional requirements.**
  - While respecting the constraints imposed by the **non-functional requirements...** – including the budget
  - General principles of good **quality.**

DESIGN AS A SERIES OF DECISIONS

- A designer is faced with a series of design issues
  - These are **sub-problems of the overall** design problem.
  - Each issue normally **has several alternative** solutions or design options.
  - The designer must make a design decision to resolve each issue.
    - *This process involves choosing the best option from among the alternatives.*
- ***To make each design decision, the software engineer uses knowledge of***

Requirements

Design as created so far

Technology available

Software design principles and 'best practices'

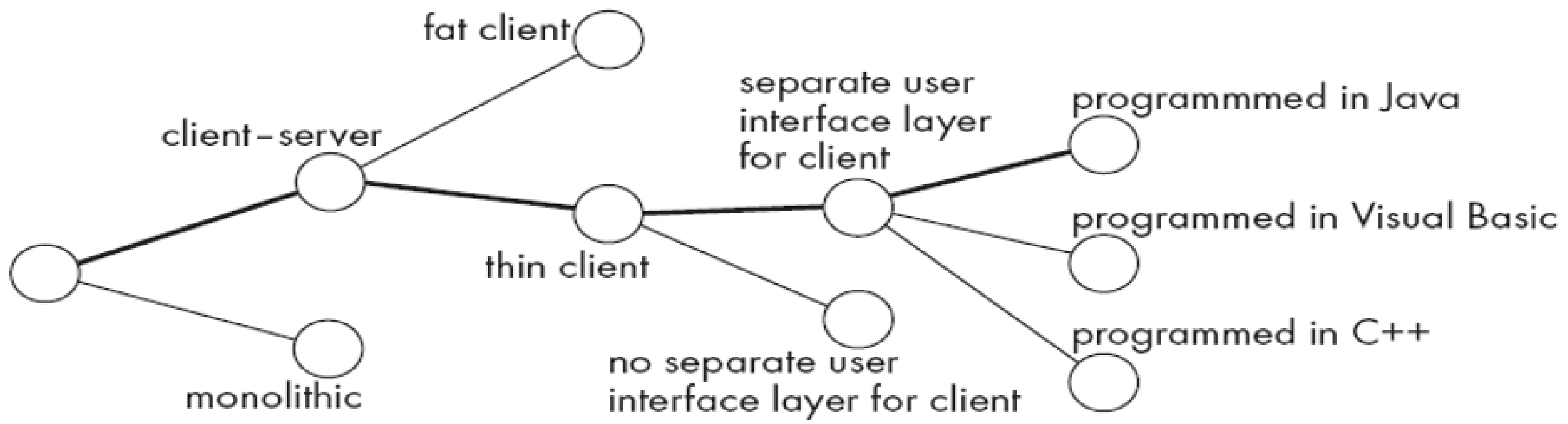
What has worked well

Knowledge

DESIGN SPACE

The space of possible designs that could be achieved by choosing different sets of alternatives is often called the design space.

For example:


fat client
client–server
separate user interface layer for client
programmmed in Java
programmed in Visual Basic
thin client
programmed in C++
no separate user interface layer for client
monolithic

System

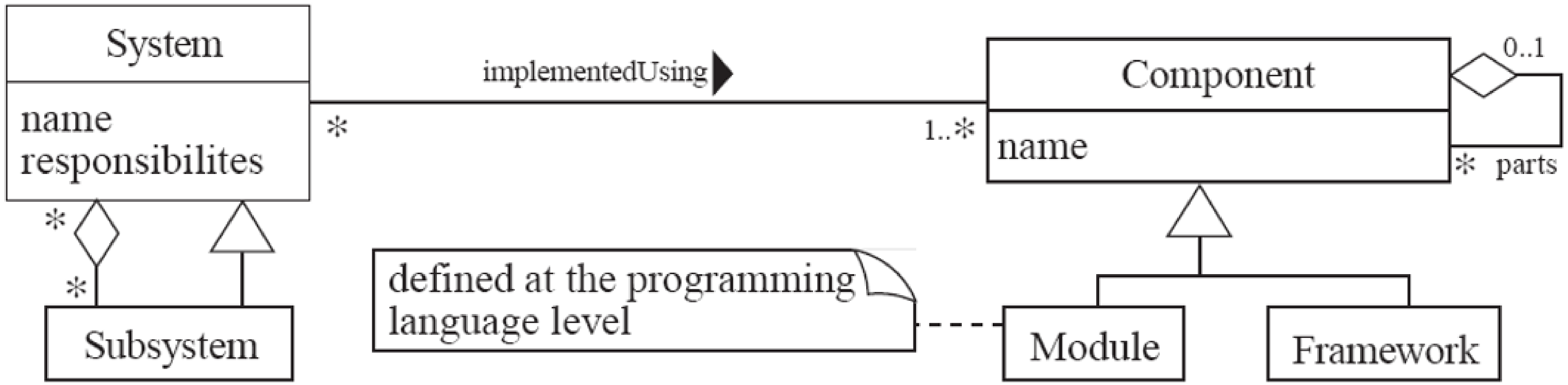
- A logical entity, having a set of definable responsibilities or objectives, and consisting of hardware, software or both.

Module

- A component that is defined at the programming language level.

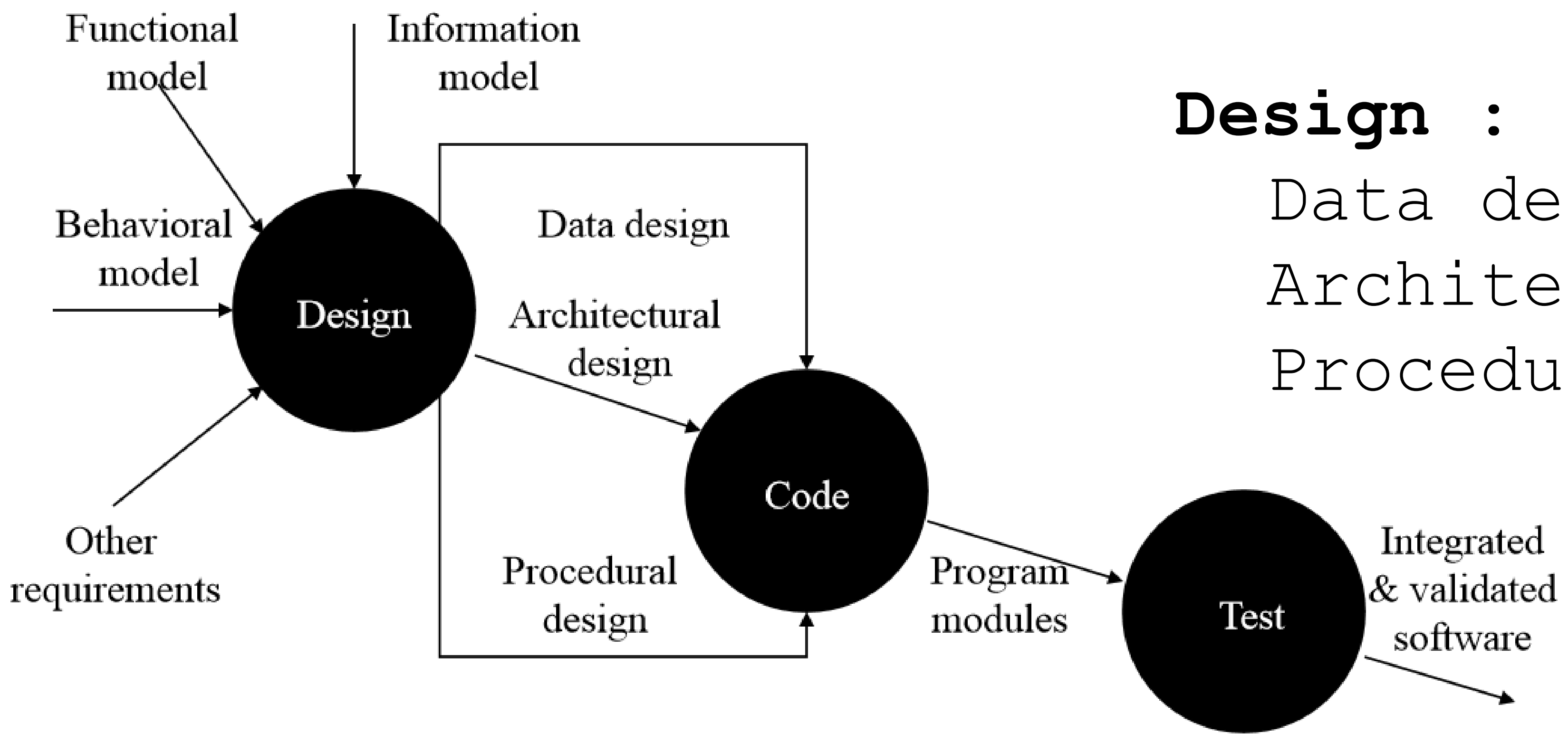
Component

- Any piece of software or hardware that has a clear role.



UML OF SYSTEM PARTS

SOFTWARE DESIGN PROCESS



Design :

Data design

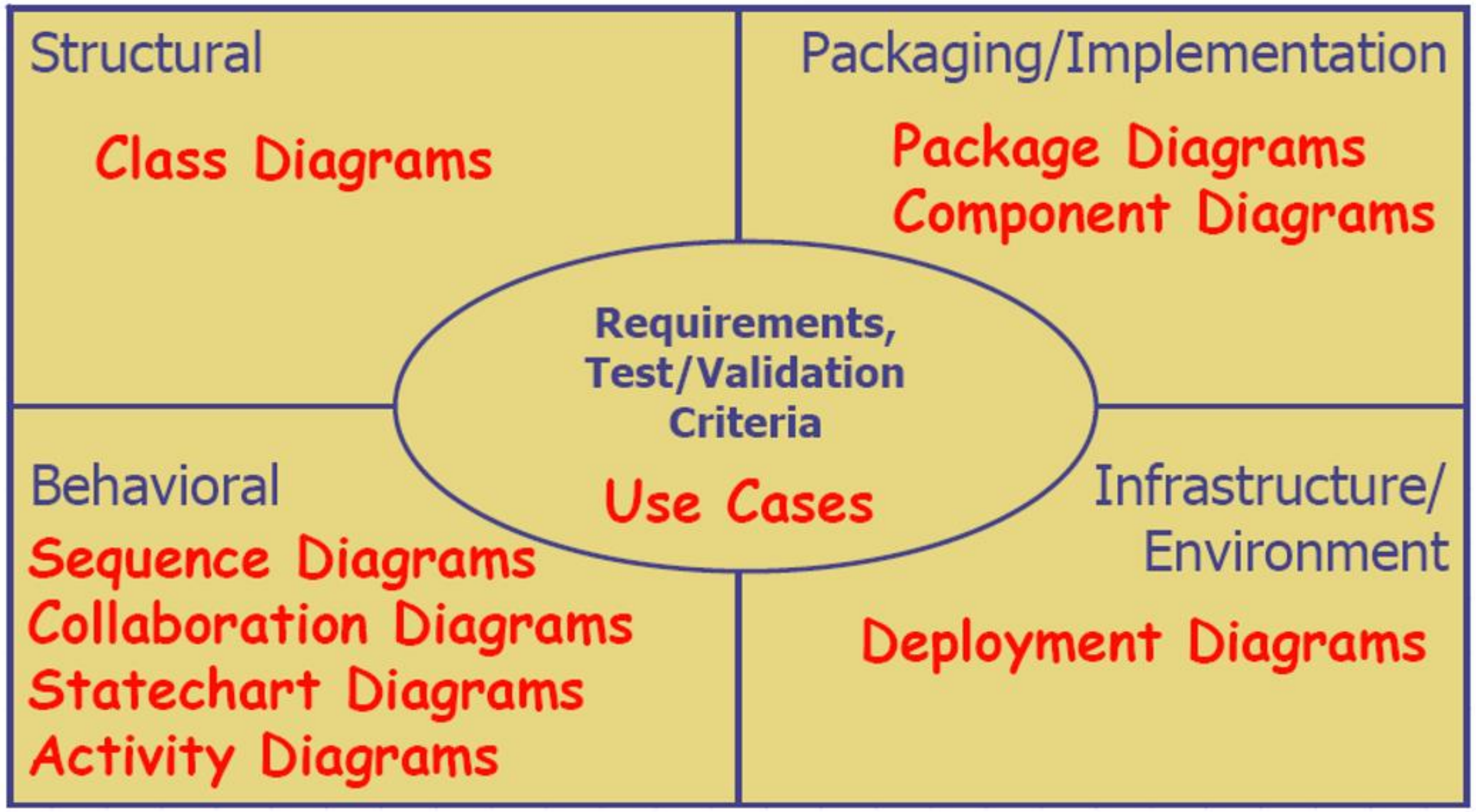
Architectural design

Procedural design

UNIFIED MODEL LANGUAGE (UML)

- A standardized, graphical “modeling language” for communicating software design.
- **UML is a fusion of ideas from several precursor modeling languages.**
- Allows implementation-independent specification of:
  - User/System interactions (required behaviors)
  - Partitioning of responsibility (OO)
  - Integration with larger or existing systems
  - Data flow and dependency
  - Operation orderings (algorithms)
  - Concurrent operations

TYPE OF UML

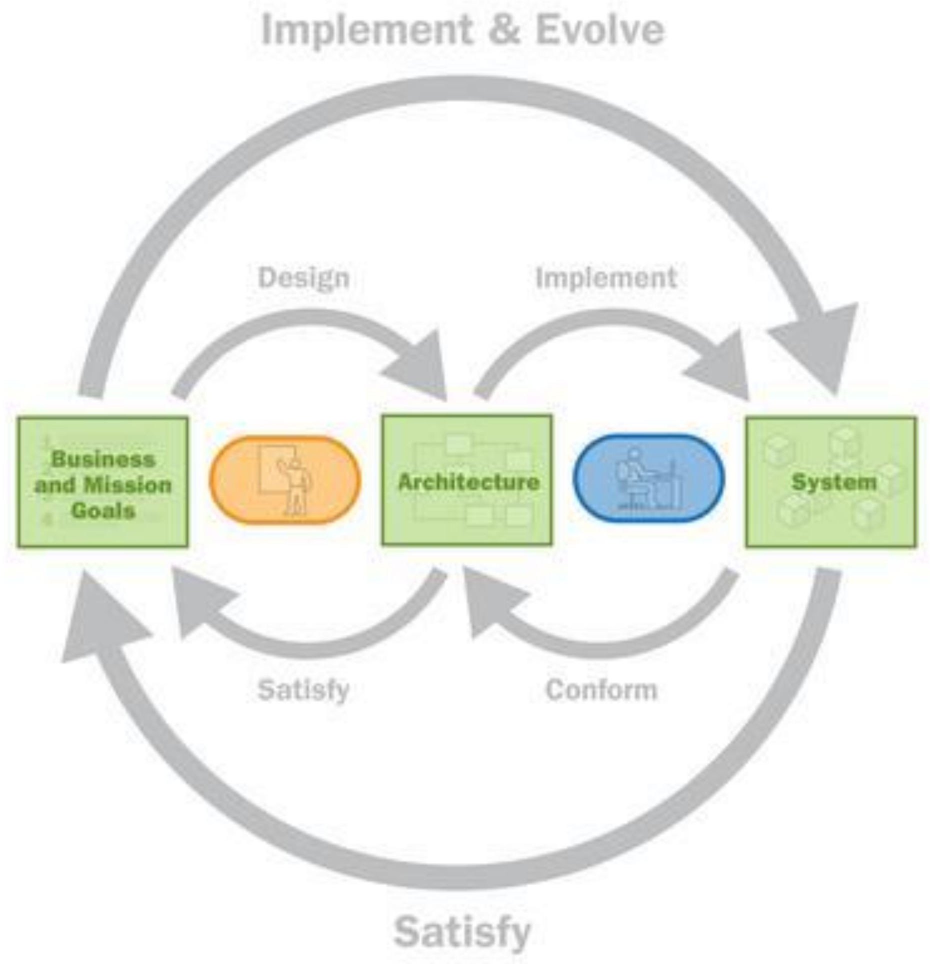

Structural
Class Diagrams
Packaging/Implementation
Package Diagrams
Component Diagrams
Requirements, Test/Validation Criteria
Use Cases
Behavioral
Sequence Diagrams
Collaboration Diagrams
Statechart Diagrams
Activity Diagrams
Infrastructure/ Environment
Deployment Diagrams

SOFTWARE ARCHITECTURE

DEFINITION OF SOFTWARE ARCHITECTURE

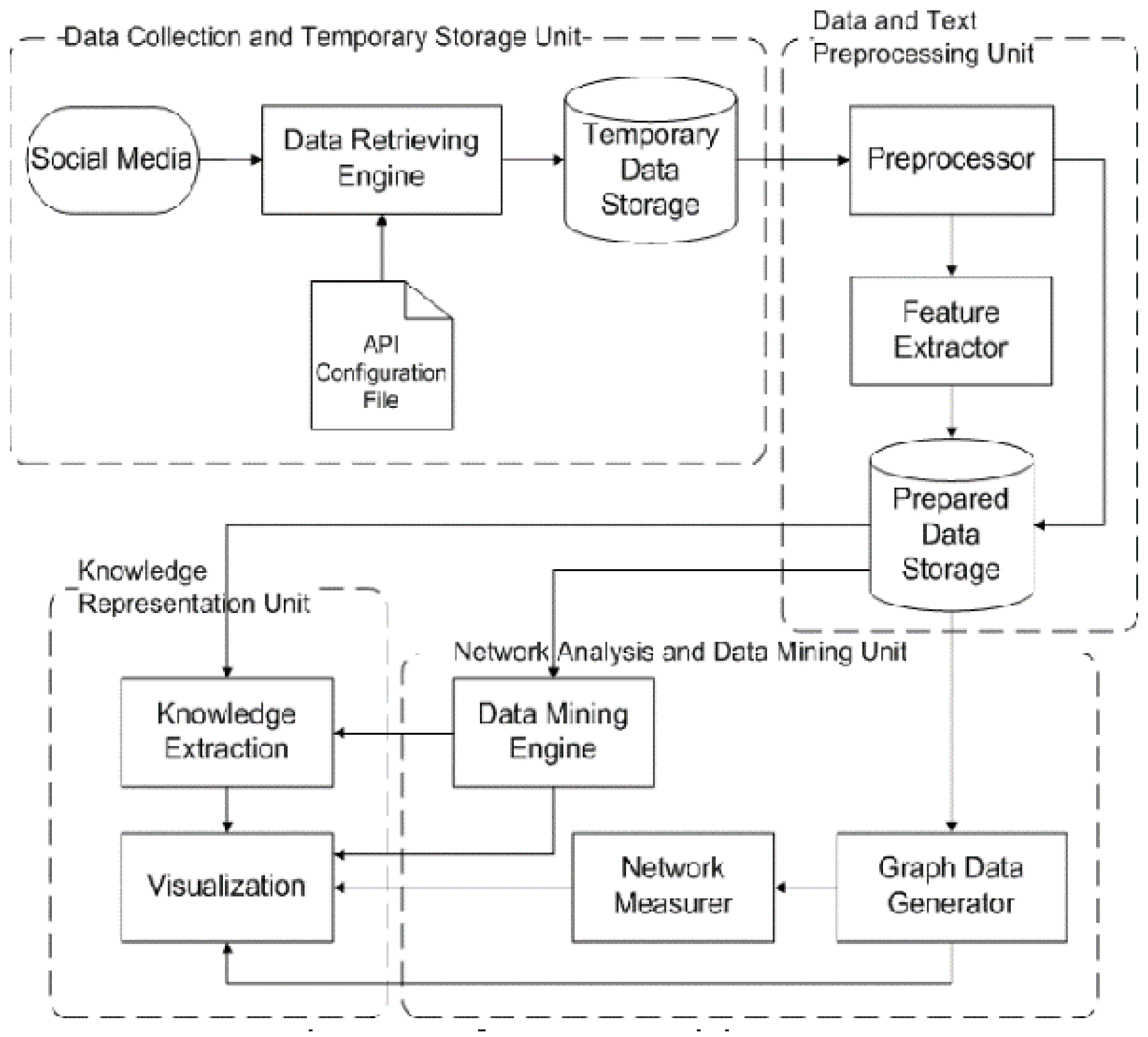
- The Software Architecture of a system consists of a description of the
  - System elements
  - Interactions between the system elements,
  - Patterns that guide the system elements,
  - Constraints on the relationships between system elements.

- Its a more abstract view of the design
- Its helpful for communication and complexity management

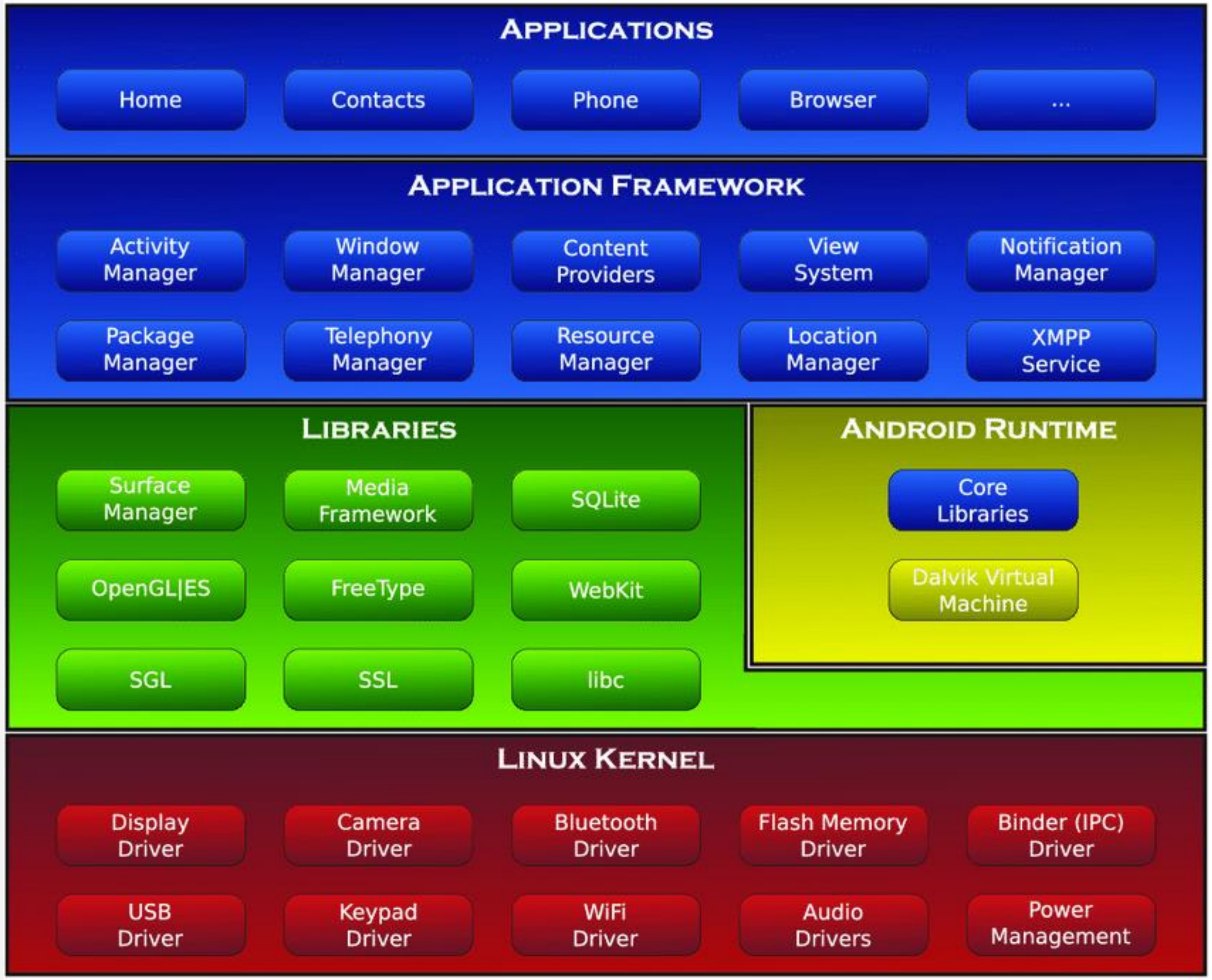


http://www.cs.cmu.edu/afs/cs/project/vit/ftp/pdf/intro_softarch.pdf

SOCIAL NETWORK ARCHITECTURE



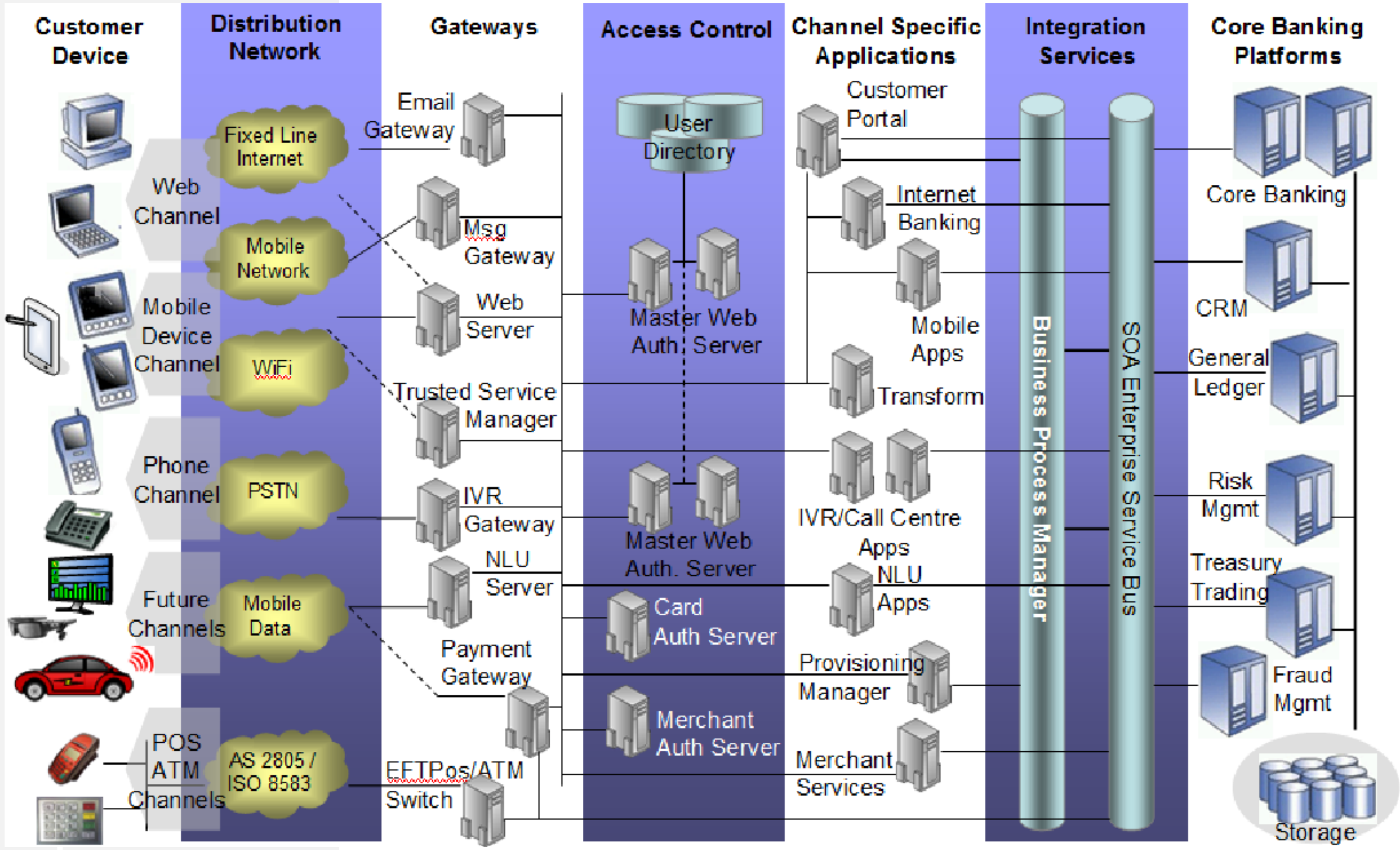
Perwitasari, Anggi et al. "Software architecture for social media data analytics." *2015 International Conference on Data and Software Engineering (ICoDSE)* (2015): 208-213.



ANDROID SYSTEM ARCHITECTURE

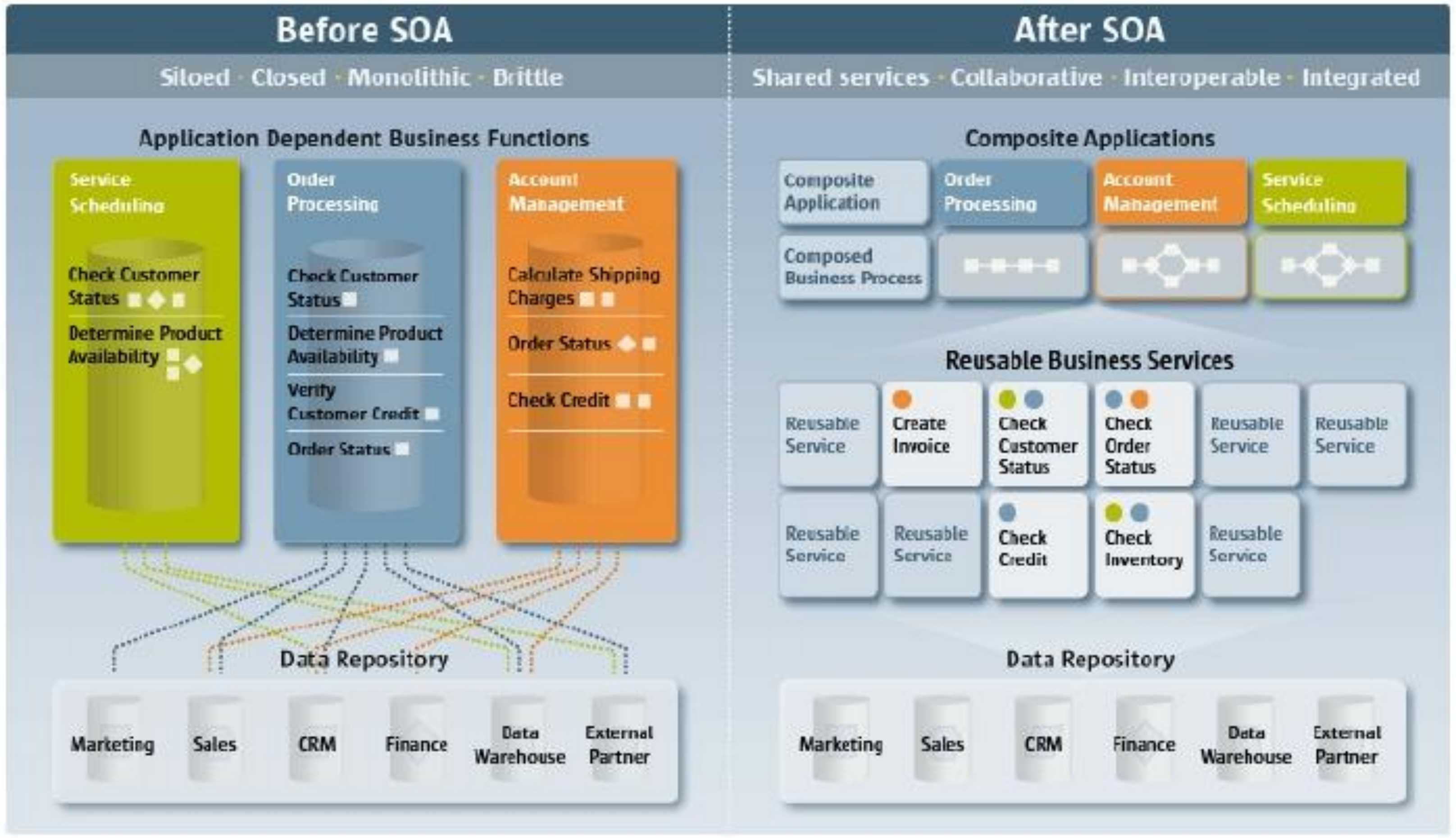
van der Veen, Victor., 2013). Dynamic Analysis of Android Malware. 10.13140/2.1.2373.4080.

CORE BANKING SYSTEM

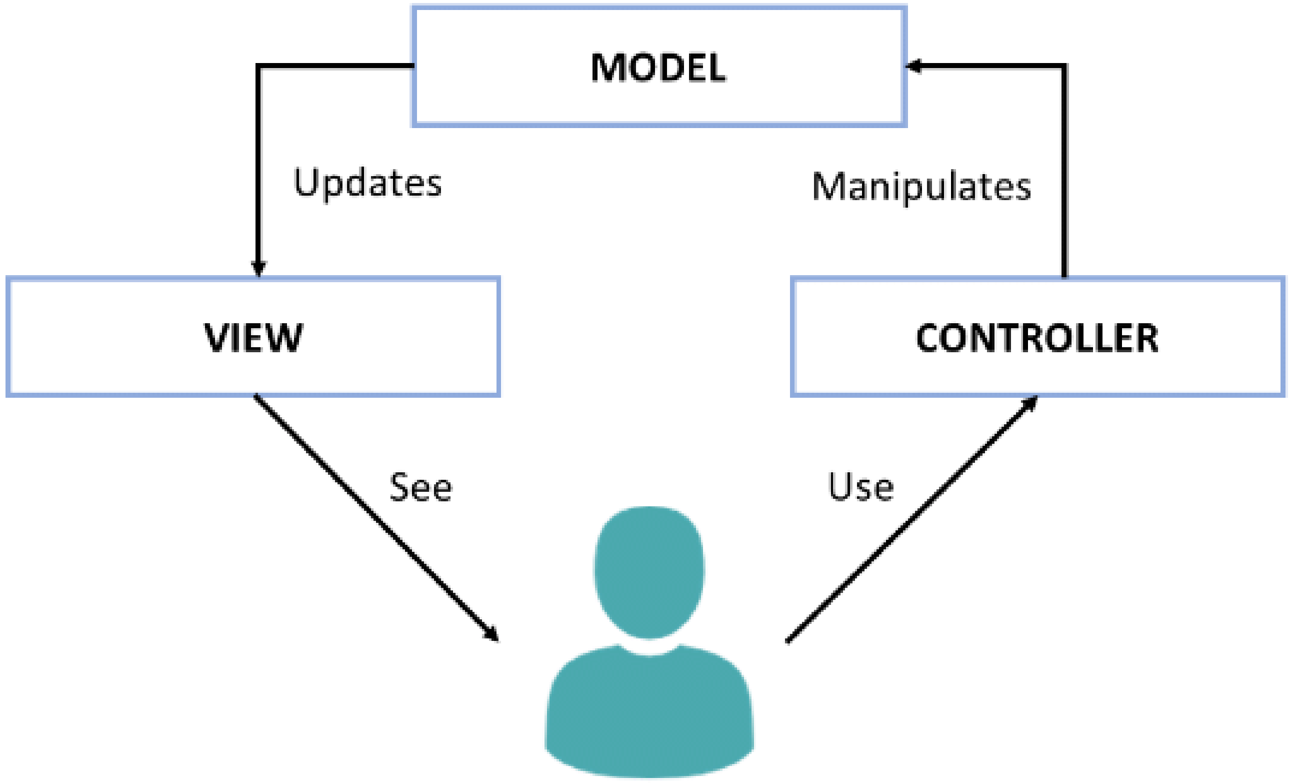


Pavlovski, Chris. "A MULTI-CHANNEL SYSTEM ARCHITECTURE FOR BANKING." (2013). International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.3, No.5 DOI:10.5121/ijcsea.2013.3501

SERVICE-ORIENTED ARCHITECTURE



Selim, Mohamed. (2016). An Empirical Analysis on the Microservices Architecture Pattern and Service-Oriented Architecture.



MODEL-CONTROL-VIEW (MVC)

Mohd Nor, Rizal and others. (2018). Cloudemy: Step into the Cloud. Journal of Telecommunication, Electronic and Computer Engineering Vol. 9 No. 3-5 , 135-139

SUMMARY

- Overview of Software Design and Development
- Software Product Life Cycle / Software Development Life Cycle
- Object-oriented Model (OO)
- Software Design
- Software Architecture