UML
(Unified Modeling Language)

CITRA N., MT

Definition and function

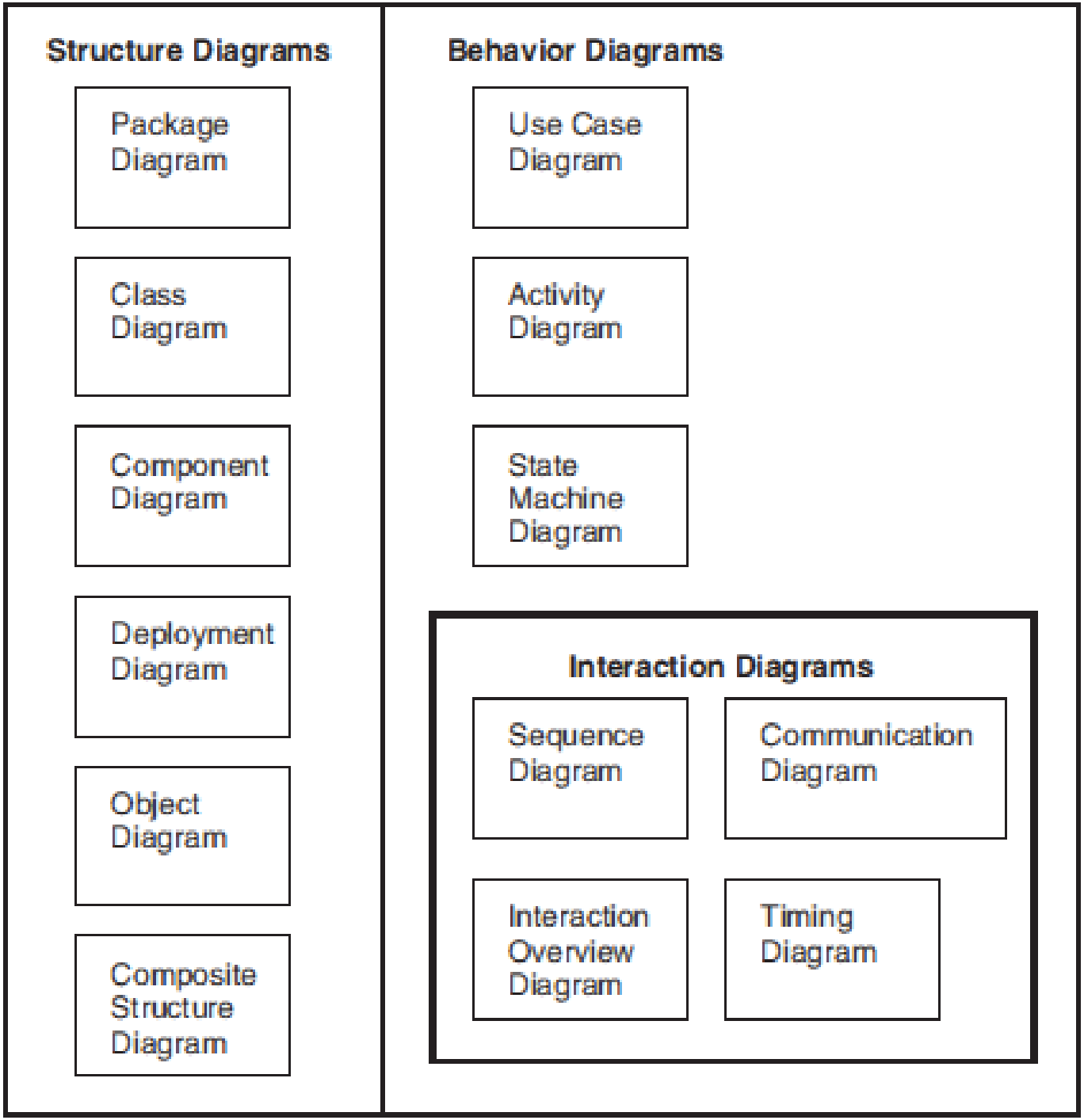
- The UML defines a diagrammatic notation for describing the artifacts of an OOAD.
- UML can visualize, specify, construct and document our software application. As our software system become ever larger and ever more complex we need to manage that complexity in a sense, simplify it so we have a better undertanding,
- By inspecting our models we can identify deficiencies in our designs as well as opportunity to enhance them. The UML acts as a specification language in which we can precisely and unambiguously capture our design decisions.

Conceptual, Logical and Physical Models

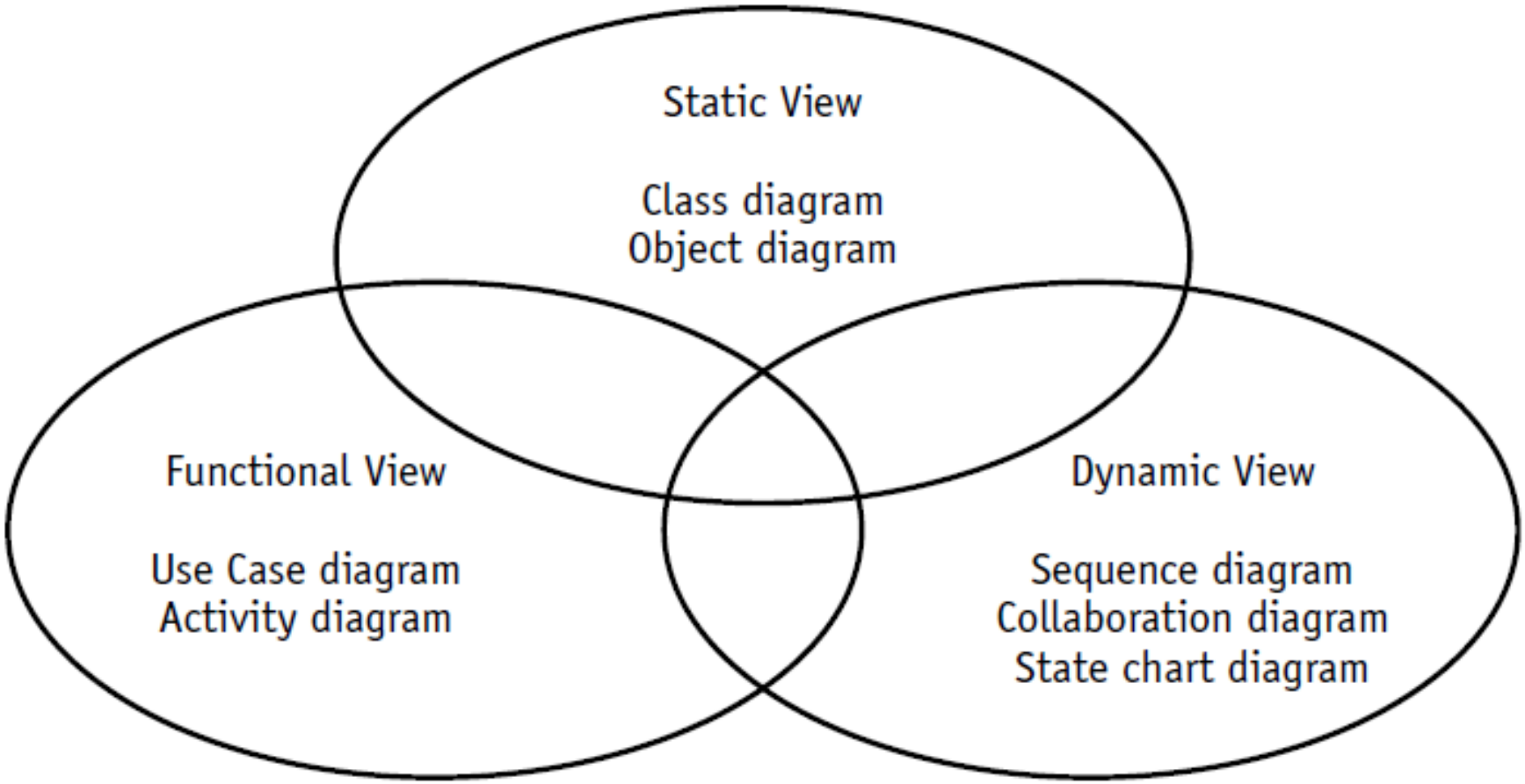
- The models of your system may present various levels of detail as your system development progresses and matures over time.
- The conceptual model captures the system in terms of the domain entities that exist (or will exist) and their association with other such entities of your system. The conceptual level of modeling is performed using the terminology of your business domain and should be technology-agnostic.
- The logical view of a system takes the concepts created in the conceptual model and establishes the existence and meaning of the key abstractions and mechanisms that will determine the system's architecture and overall design.
- The physical model of a system describes the concrete software and hardware composition of the system's implementation. Obviously, the physical model is technology-specific.

- We can built UML as :
  - Diagrams
  - Views

UML Diagrams

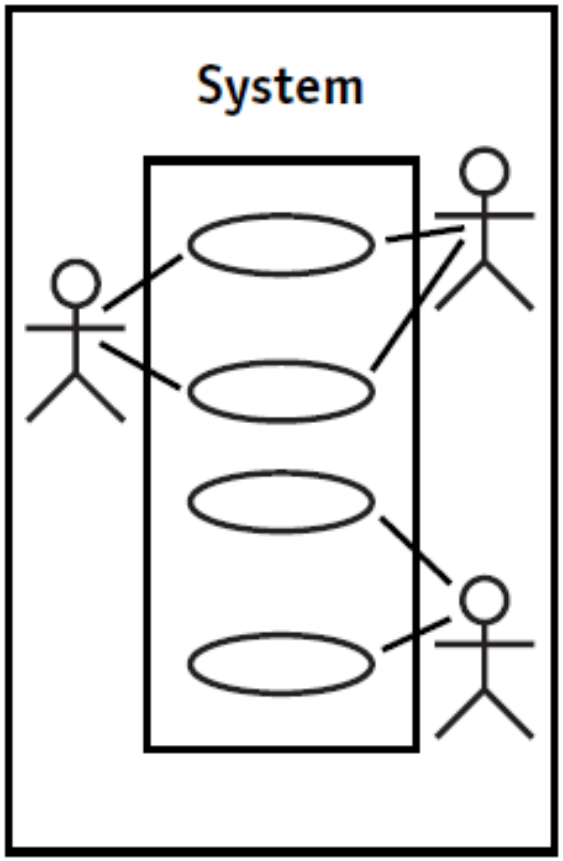


- One way to organize the UML diagrams is by using views.
- A View is acollection of diagrams that describe a similiar aspect of the project.
- We very often use a set of a three distinct complementary views that are called the Static View, Dinamyc view and Functional view,


Static View
Class diagram
Object diagram
Functional View
Use Case diagram
Activity diagram
Dynamic View
Sequence diagram
Collaboration diagram
State chart diagram

Functional View

- In the functional view, we include both the use case diagram and the activity diagram. We keep them together because they are used together so often and they both model how the system supposed to work,
- Use case diagram defines the functions that the system must provide, the functions are expressed first as goals, but then the goals are fleshed out in anarrative to describe what each use case is expected to do to achieve each goal.



Use Case Diagram

Name

Assumptions

Pre-conditions

Dialog

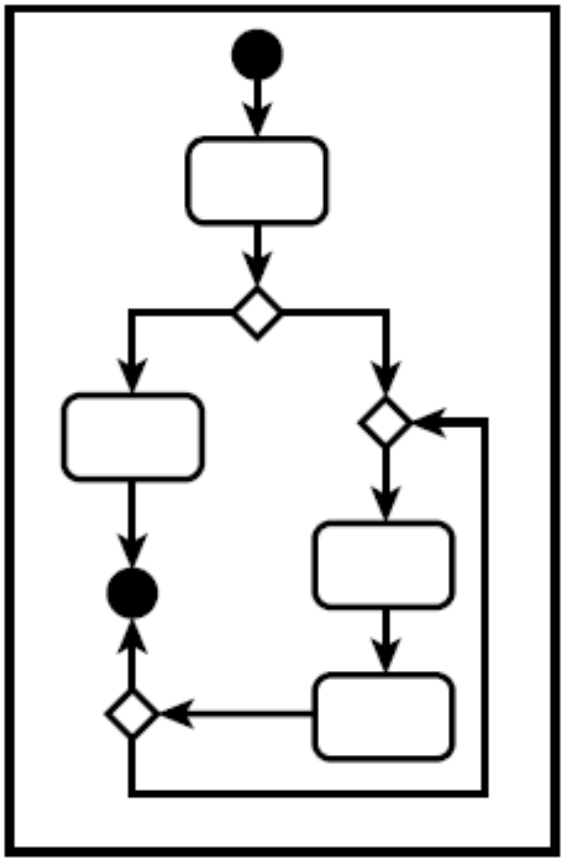
Post-conditions

Exceptions

Future Enhancements

Open Issues

Use Case Narrative



Activity Diagram

Static View

- Static view include those diagram that provide a snapshot of the elements of the system but don't tell you how the elements will behave. It is like blueprint.
- The class diagram is the primary tool of the static view, it provides a fixed look at every resource and its features.

Dynamic View

- The dynamic view includes the diagrams that reveal how objects interact with one another in response to the environment.
- It includes the sequence and collaboration diagrams, which collectively are referred to as interaction diagrams. They are specifically designed to describe how objects talk each other. It also includes the Statechart diagram which shows how and why an object changes over time in response to the environment.