Human Computer Interaction

UNIT-6
Lecture 5:
Cognitive Architecture

Mr. Nachiket Sainis

B K Birla Institute of Engineering & Technology, Pilani

Object Oriented Programming

Object Oriented Modeling of User Interface Design

Overview

- Interactions
- Interface Objects
- Interface Actions
- Methodology of OOUID
- Object Oriented Modeling using UML
  - Use Cases
  - Analysis Objects
  - Tasks and Scenarios
  - Dialogue transition chart
  - Dialogue component specification
  - Interface prototype
  - Task flow
- Design
  - Model View Controller Architecture
  - Design Classes (M,V,C)

Introduction

- Presently user interfaces are built using rapid UI builders with extensive use of graphical element libraries
- This ease in deploying an UI may lead to haphazard designs
- To design usable interfaces we need to specify interaction goals
- Formal specification of UI leads to robust and effective UI
- Object orientation allows reuse of existing and tested UI elements
- Object oriented specification of UI should be directed towards implementer and the user

Interaction

- End user relies on mental models to predict behavior of the system.
- Mental model is the internal representation of the structure and behavior of the system or reality
- Accurate mental models helps user to understand and operate system efficiently
- Immediate and perceivable visual feedback of every interaction with interface helps end user to tune his mental model
- Perceptions about his interaction can further reinforce mental models towards ideal state.

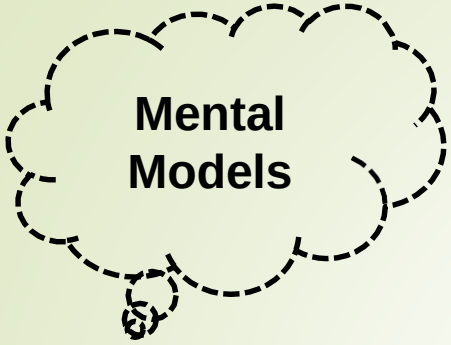
- Thus the visible user interface should reflect user's mental model

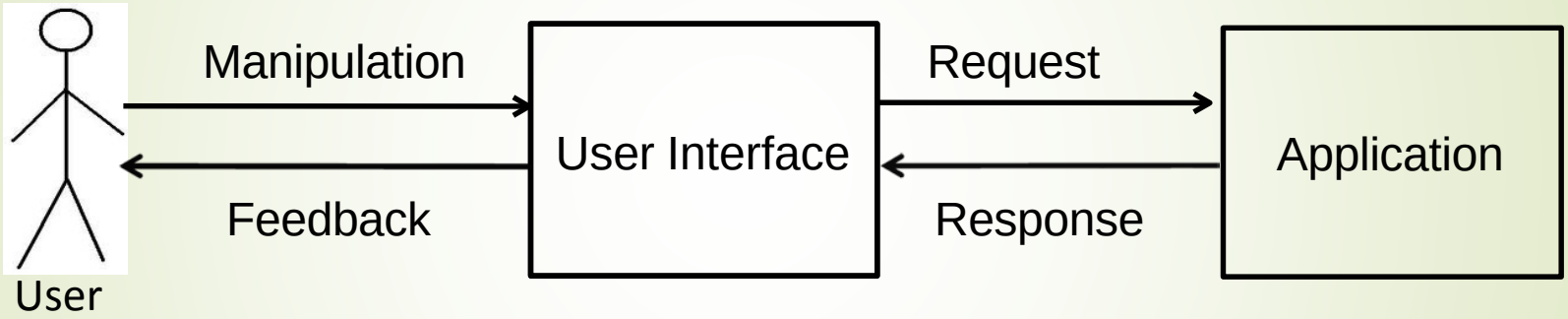
- An object oriented application is a collection of software objects representing the application domain.

- Object –oriented interface therefore connects user with the real world manipulating software objects of the background application.

Refer next figure …

Interaction



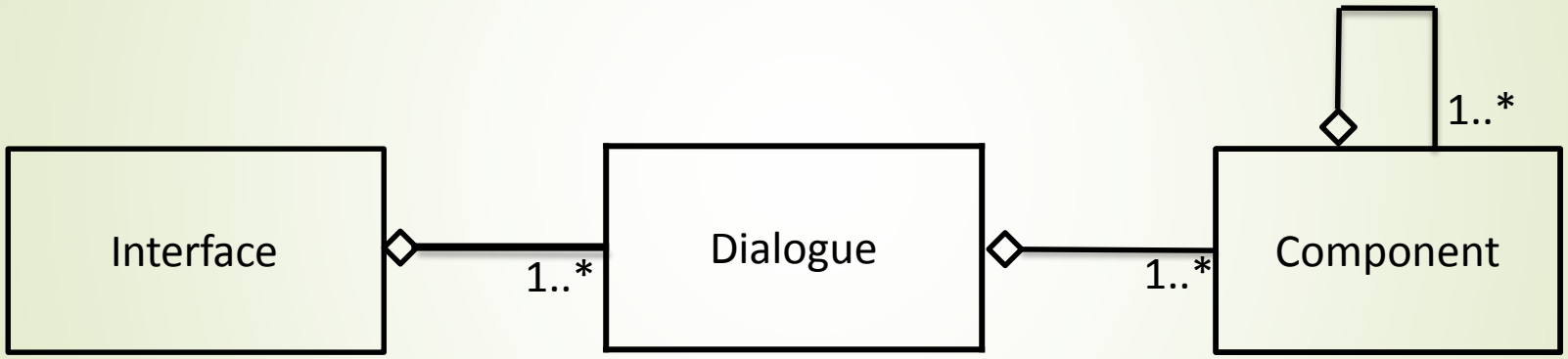


•Interface design should strive to make user successfully accomplish their goals with the help of interaction tasks and manipulation of required interface elements

- Goals → Tasks → Interface elements

Interface Objects

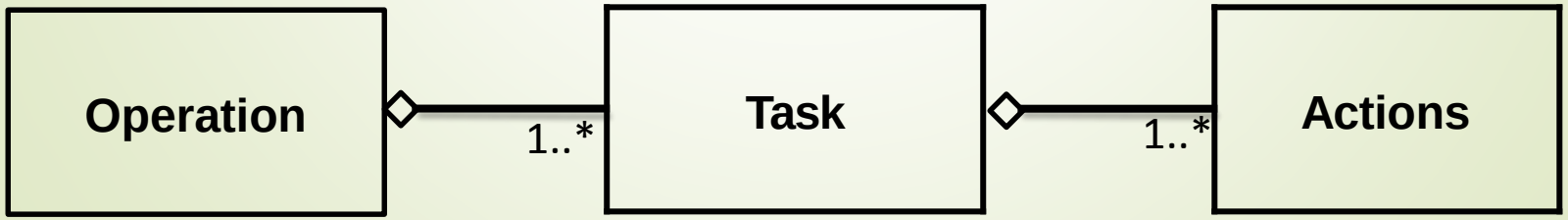
- At top level an interface is a collection dialogues.
- A dialogue is an aggregation of one or more components
- A component is further a collection of window elements that allows user to perform a meaning full action.
- Component is implemented using class having windowing elements or components



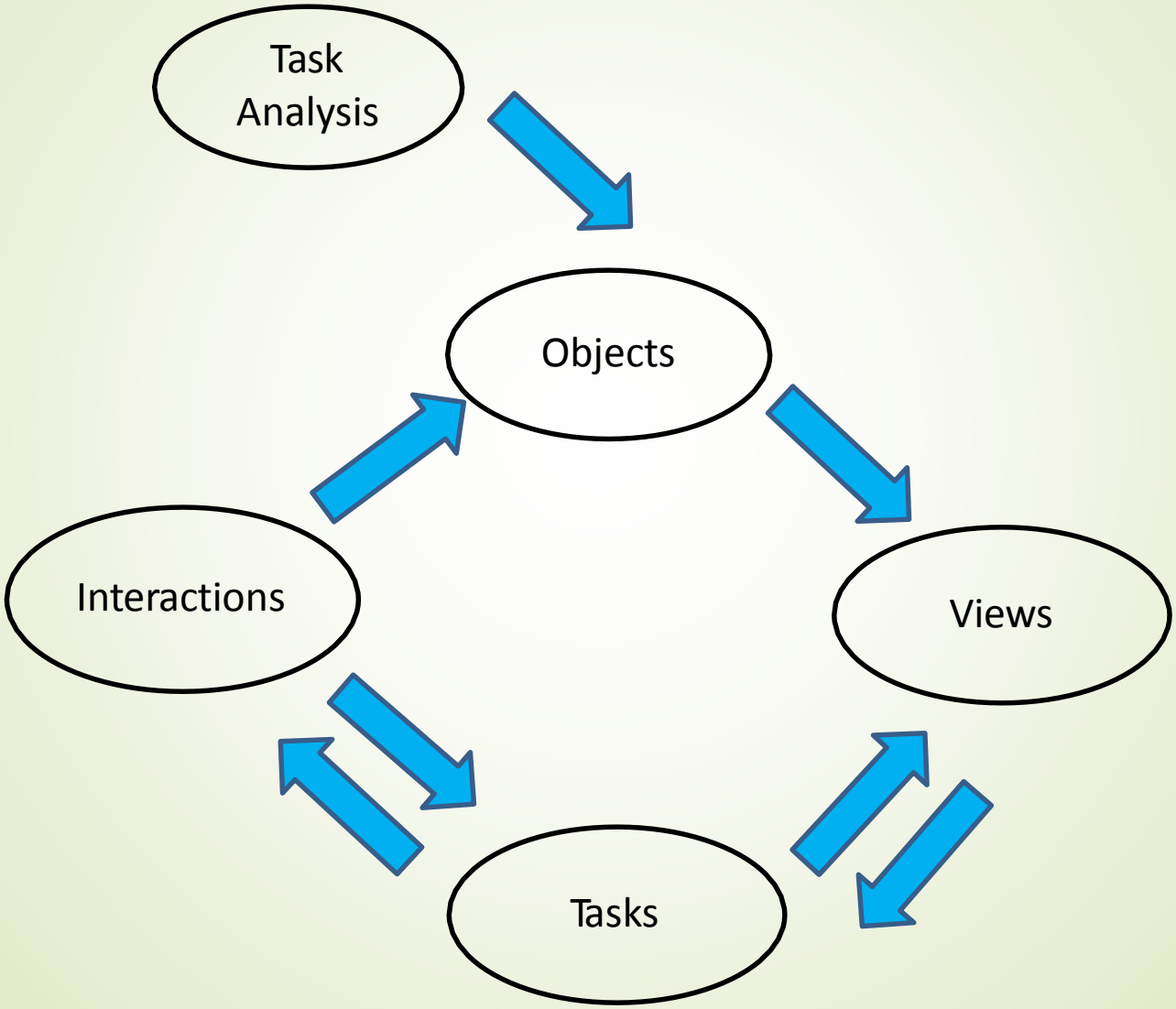
- Above figure shows a UML representation of an object oriented interface
- Filled diamond shows composition while empty diamond shows aggregation
- 1..* represents one or many multiplicity of relationship

Interface Actions

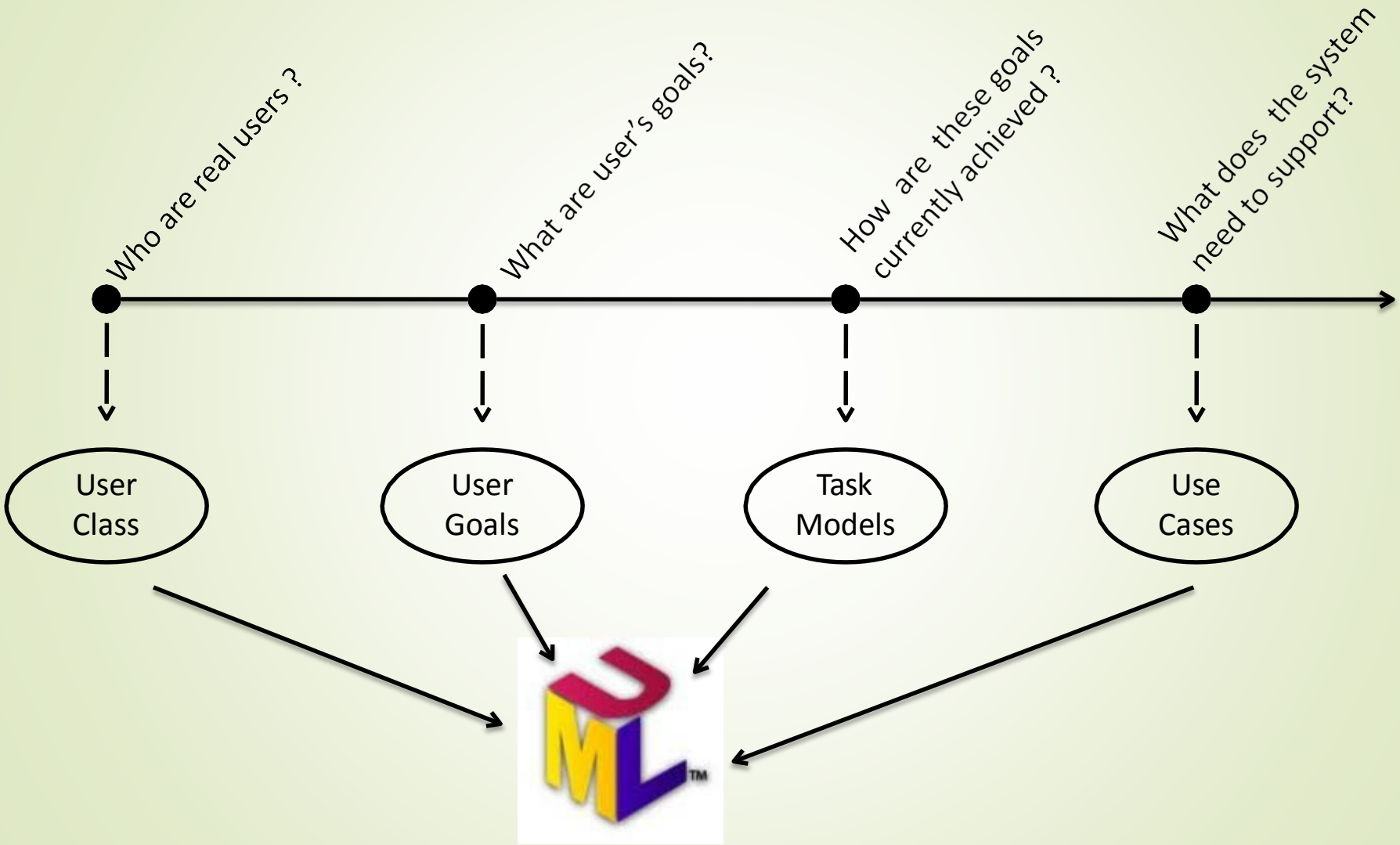
- User performs operation on interface to fulfill his goal by using system function.
- An USE CASE is a specific way of using system's functionality.
- A USE CASE specifies sequence of operations with interface to be performed.
- An operation is an activity for which the system is implemented. e.g. place order , draw picture etc .
- Operation consists of one or more tasks e.g. saving , drawing , deleting etc.
- An action is the smallest activity user performs on the user interface e.g. press button , drag picture etc



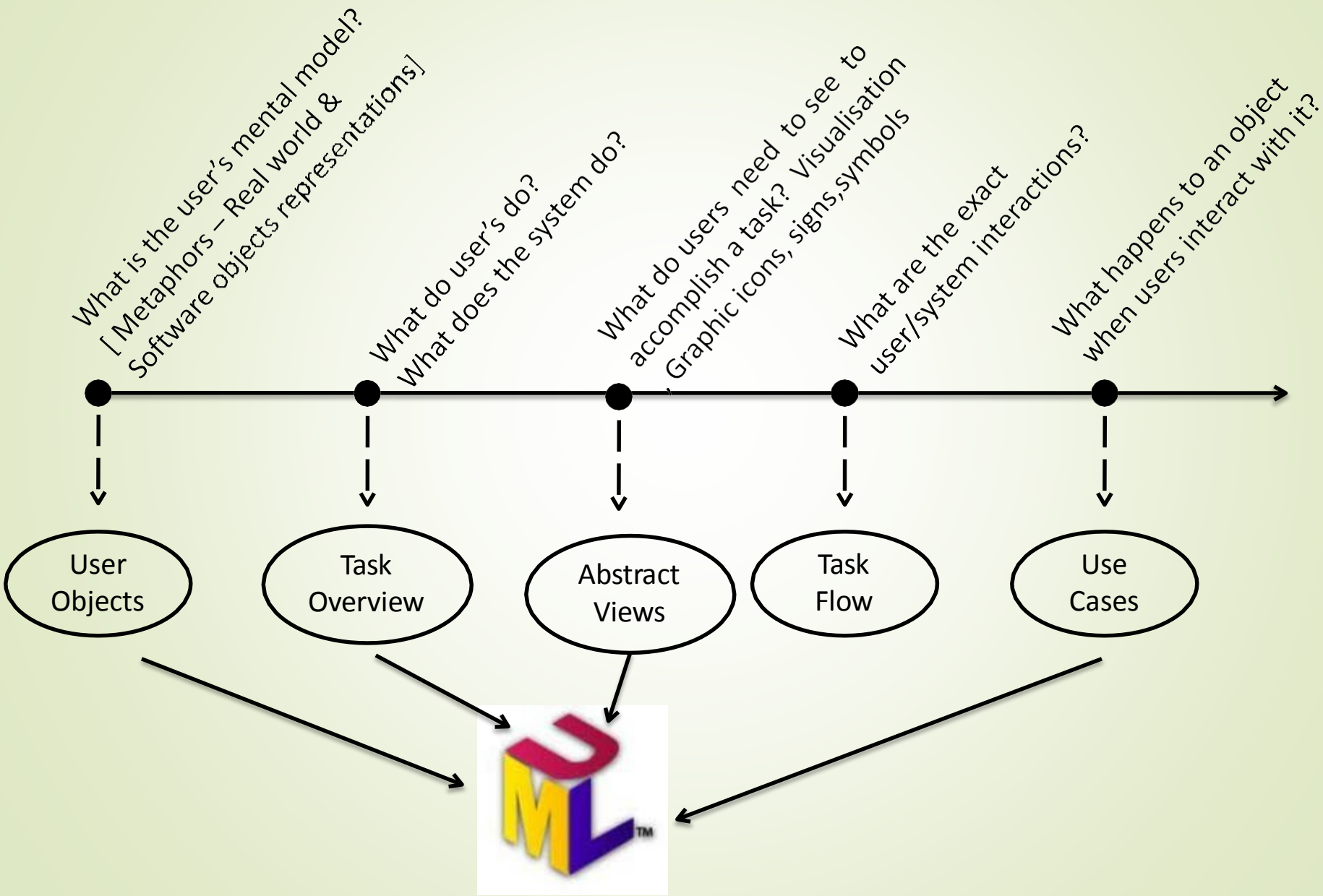
Methodology - Object Oriented UI design



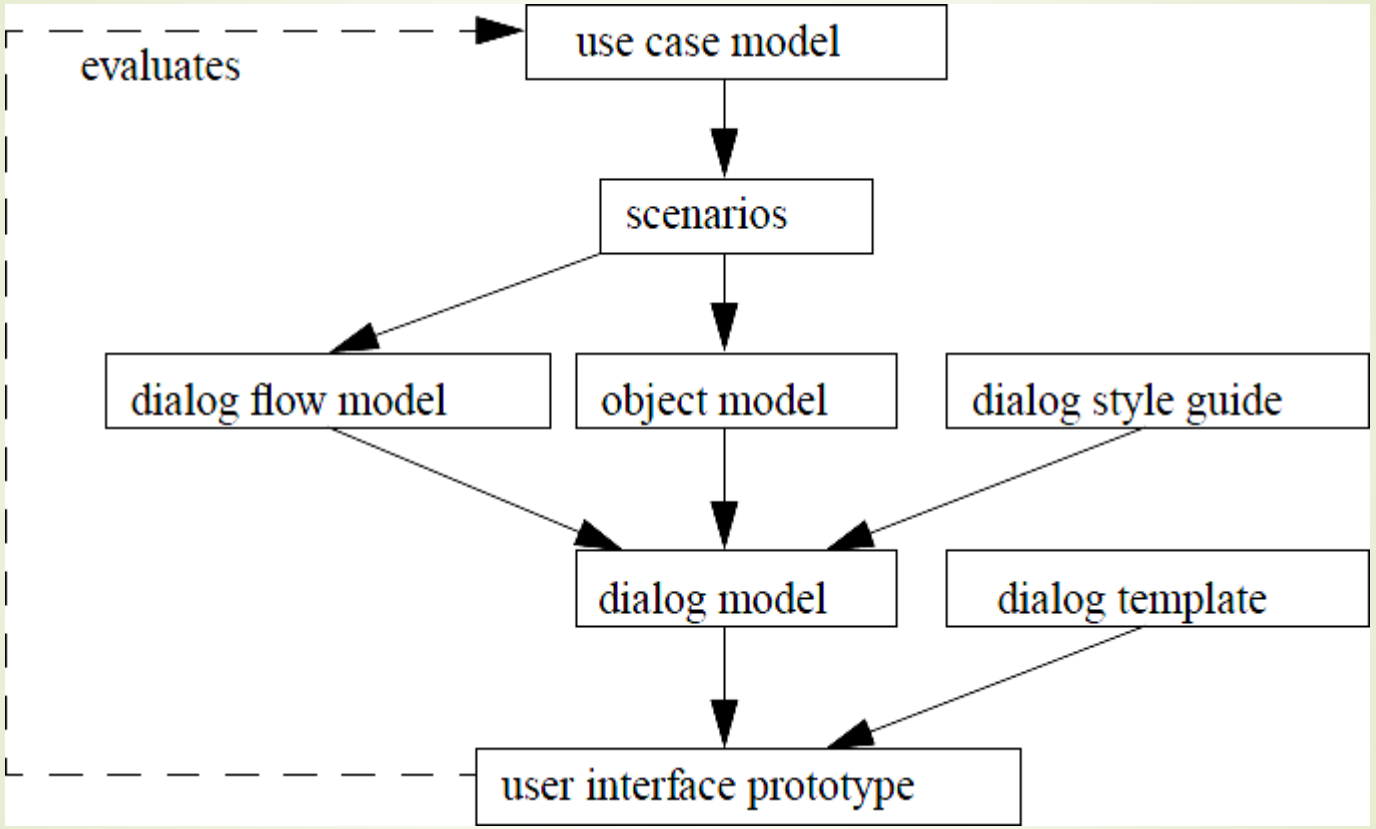
Methodology -Discovery Phase



Methodology - Abstract Design Phase



Methodology - Flow

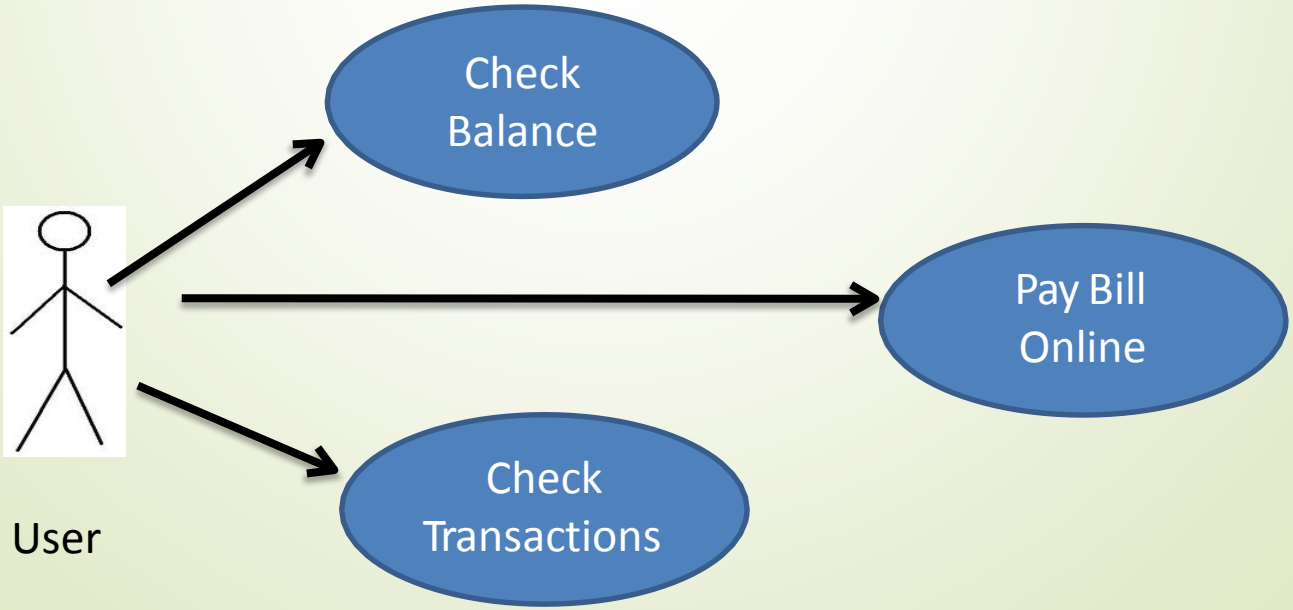


Object-Oriented Modeling of User Interfaces

- First step is analysis of user requirements – identifying USE CASES
- Design specifies the structure and components required each dialogue to accomplish the USE CASE
- Interfaces are then developed iteratively and tested against with USECASE specifications and various criteria viz. performance , usability etc.

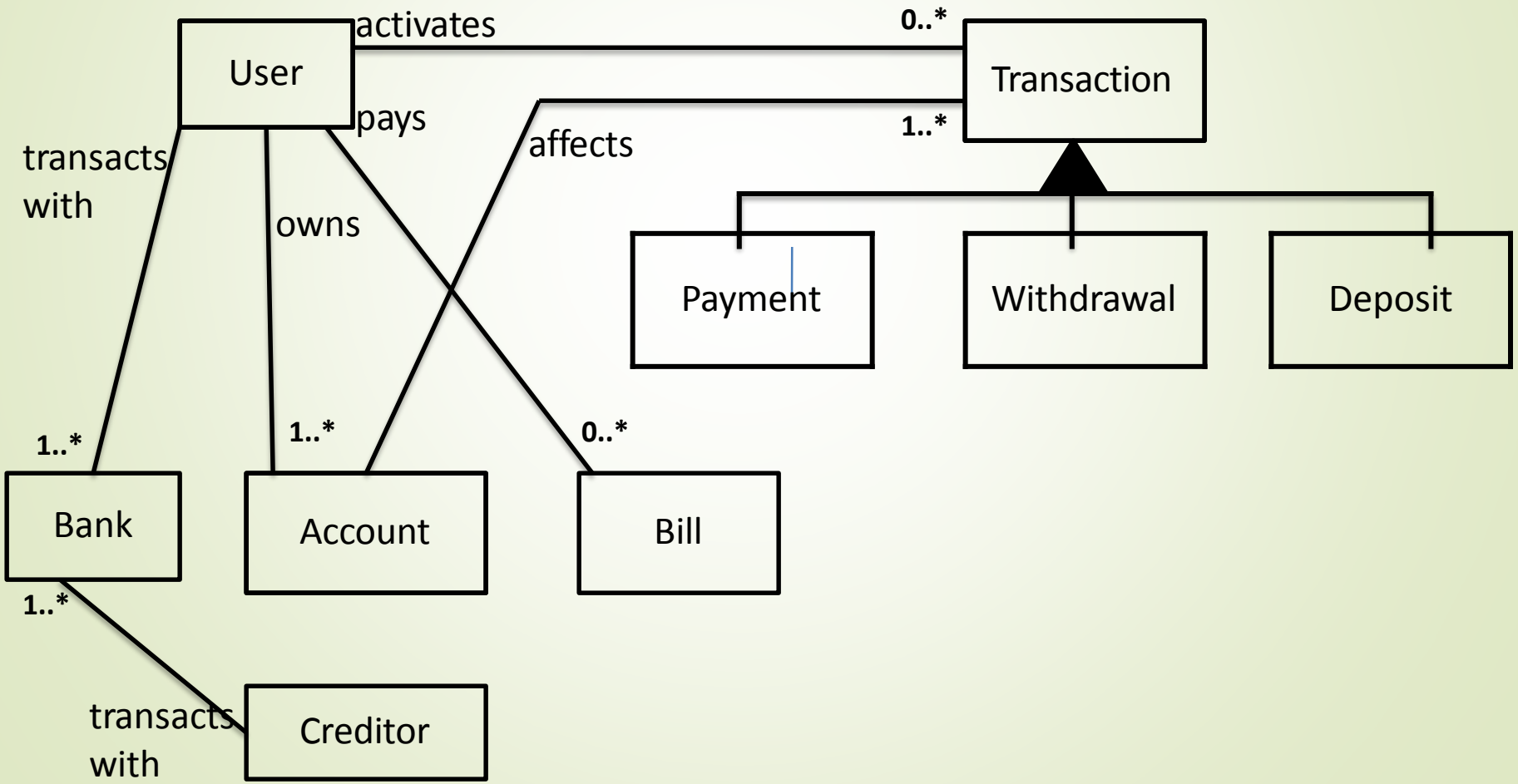
Example : Personal banking application.

- Why would a user interact with such application? What are his goals?
- USE CASES for personal banking application are



Object Analysis Model

- The sequence of operations documented for each USE CASE are analyzed to identified key objects in the application domain resulting into an object model.
- Key objects are called analysis objects and the diagram showing relationships between these objects is called object diagram.



Task (Operation) Analysis

- The sequence of tasks (user interactions and system feedback) are specified for each USE CASE which is called scenario.
- Scenario specifies detailed interaction with interface elements and feedback received.
- Operation or scenarios are analyzed to identify **interaction tasks**.
- Examples

<u>Scenario 1 : USE CASE (Operation) →Check Balance</u>

1. Select account
2. View balance

<u>Scenario 2 : USE CASE (Operation) →Pay Bill Online</u>

1. Select account
2. Update bill
3. Put bill for payment

<u>Scenario 3 : USE CASE (Operation) →Check Transactions</u>

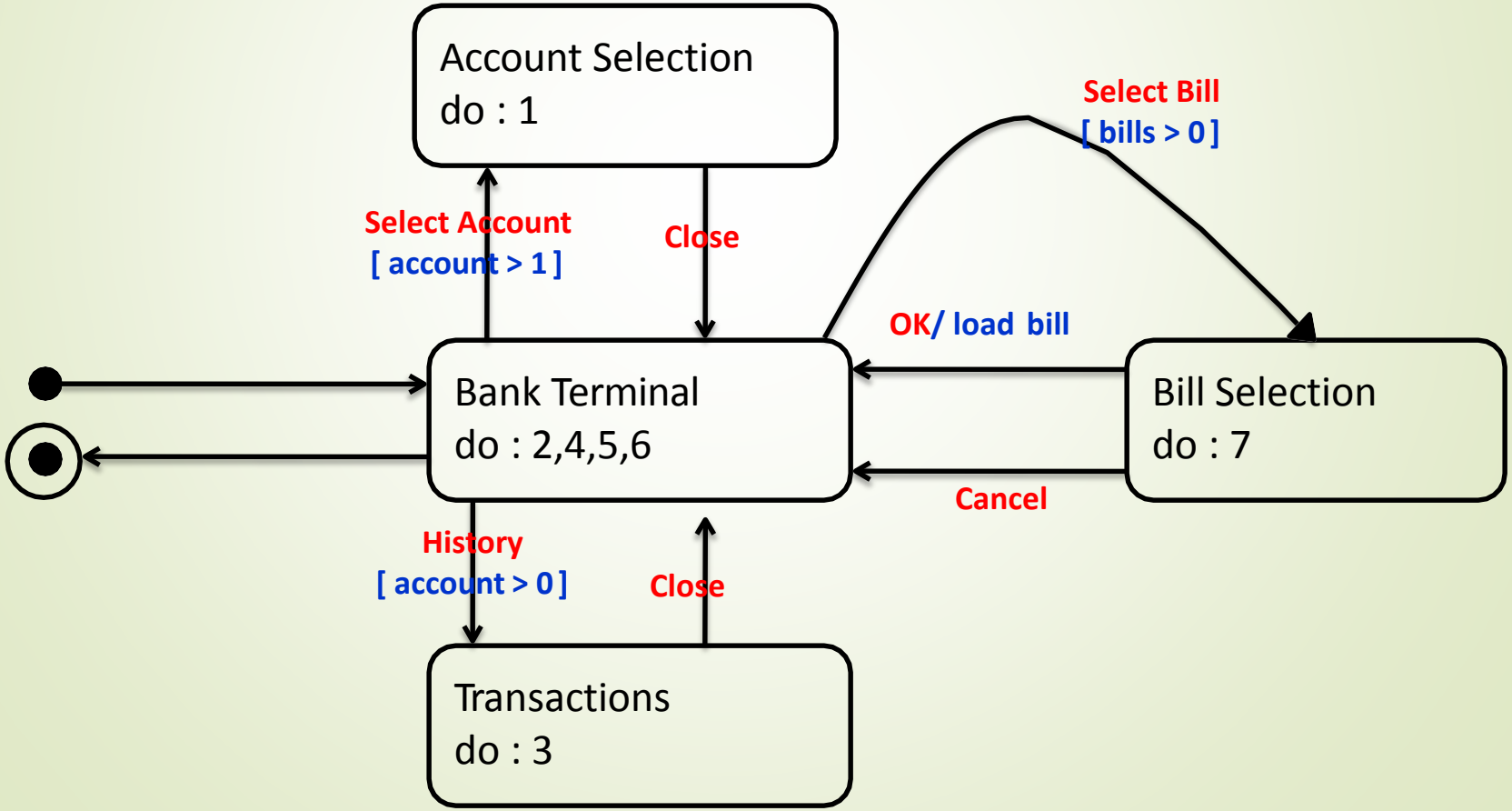
1. Select account
2. View transactions

<u>Task List</u>

- Select account
- View balance
- View transactions
- Update bill
- Put bill for payment

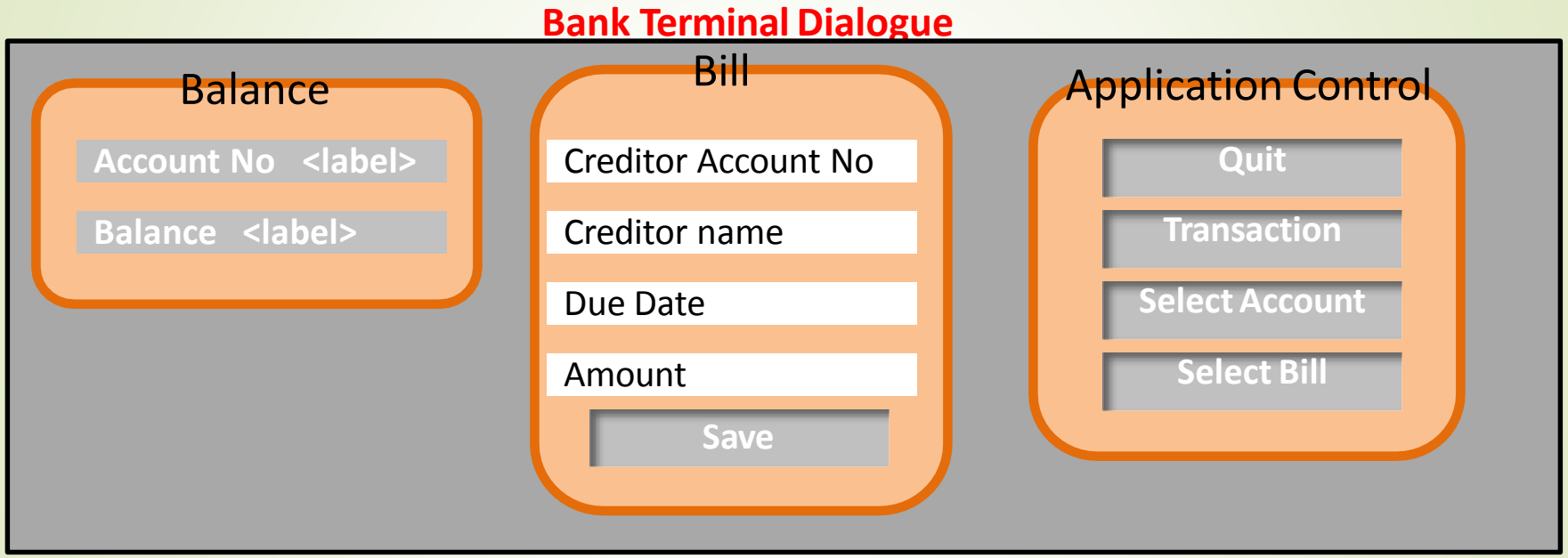
Behavioral Structure Specification - dialogues transitions

- State transition diagram shows relationship between dialogues (windows) of an interface
- Every dialogue is considered as a state of the system.
- Events (user actions) that trigger transition from one dialogue to another are specified
- A do statement within dialogue specifies task number from the scenario .
- Conditions for transition from one dialogue to another are written in square [] brackets
- Operation during transition are written after trigger a preceded with slash /.



Interface Components Specification

- Specifies UI components required to carry out task within each dialogue
- Types of UI components → input (push button , slider) , output (label , image)
- Show below is dialogue : Bank Terminal having instances of 3 components

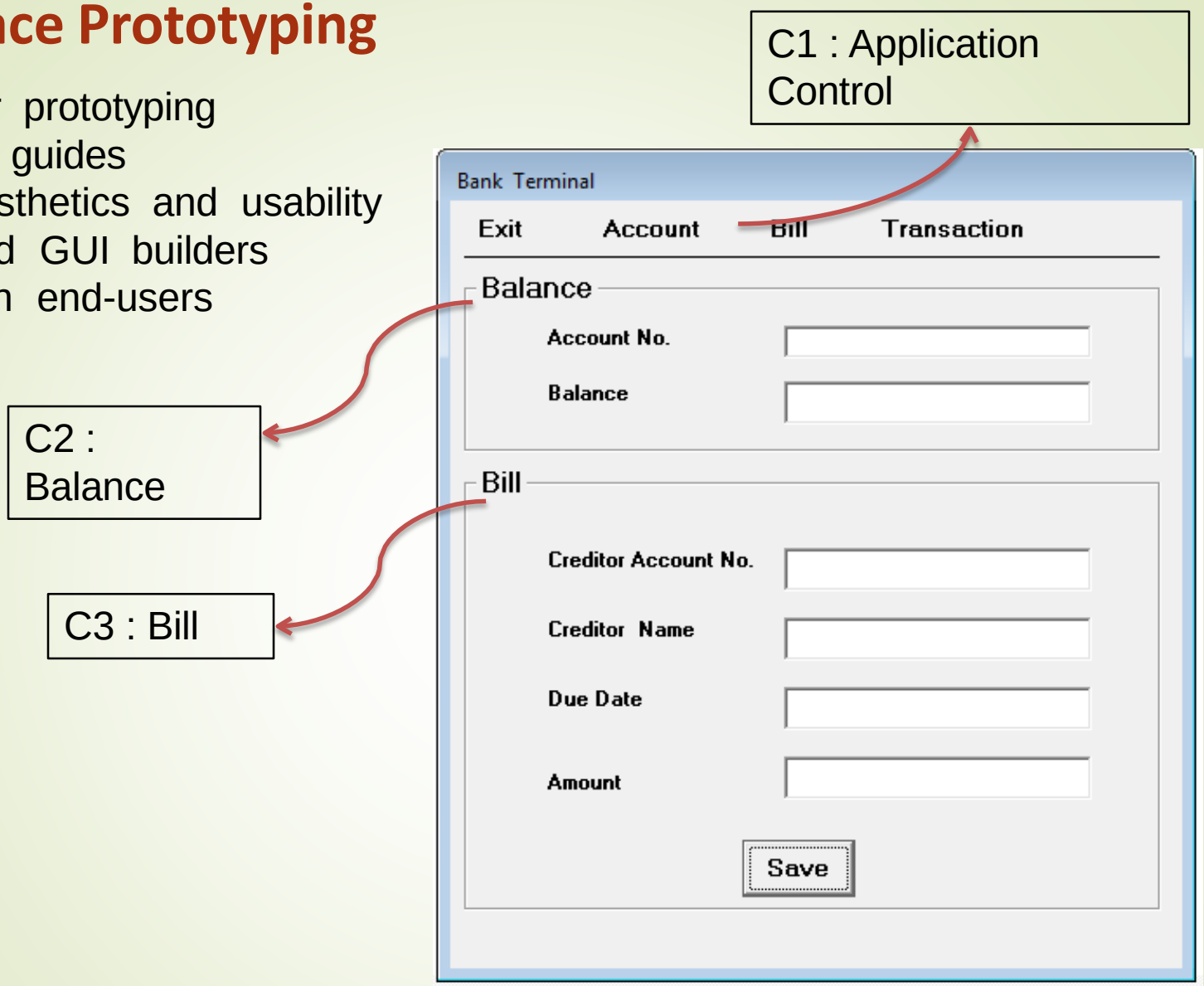


- Structural elements : window , menu , icons , controls(widgets), tabs ….
- Interactional elements : cursor , pointer , selection handle ….
- Component increases reusability and consistency of UI
- A component can be created using elements from scratch or from available standard dialogue classes or components from libraries(packages).

Interface Prototyping

Guidelines for prototyping

- Follow style guides
- Consider aesthetics and usability
- Use standard GUI builders
- Validate with end-users

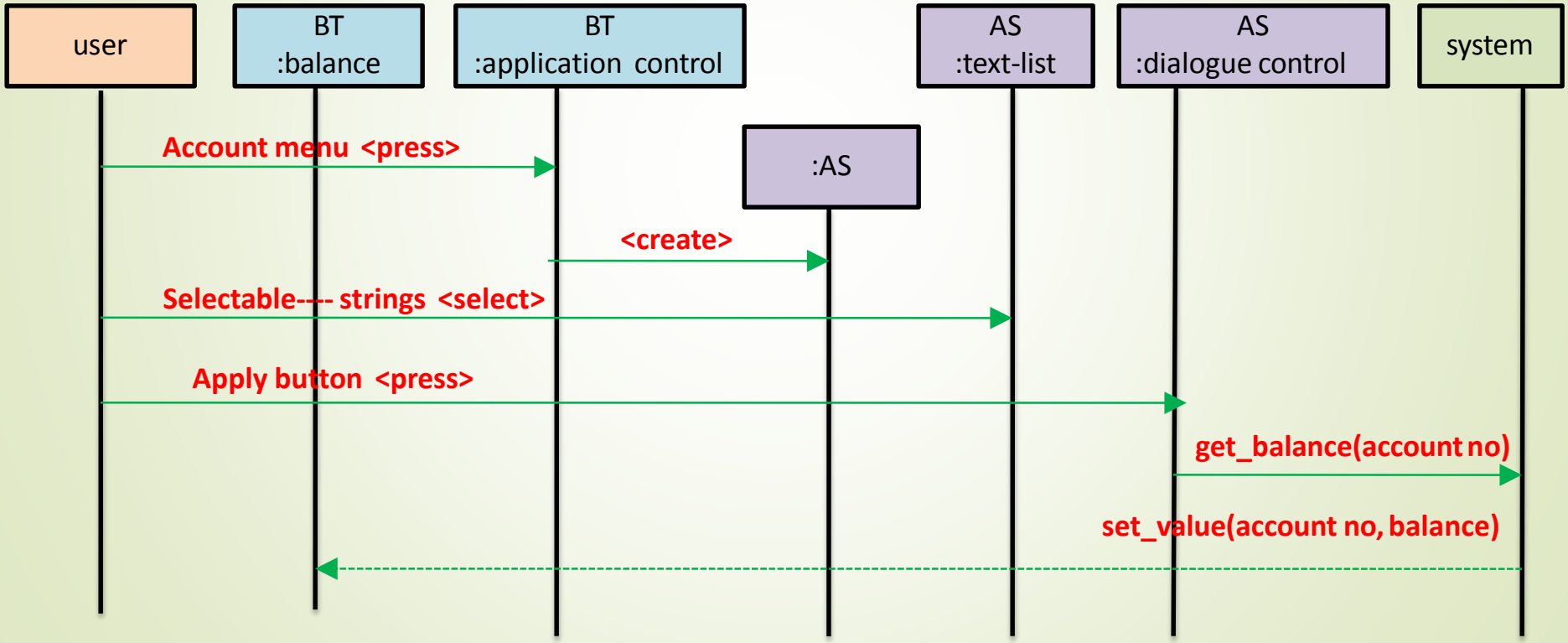


Thorough designing of dialogues and components leads to identification of highly reusable interface elements which finally leads to harmonized user interfaces

Task Specification And Flow

- Specifies sequence of actions to be performed on final user interface for a task.
- Tasks are selected from use case scenarios.
- Designers performs a walk-through with final interface for each task.
- Specification of event trace of interactions among user, interface and system can also be depicted with interaction diagram as below for **"selecting an account".**

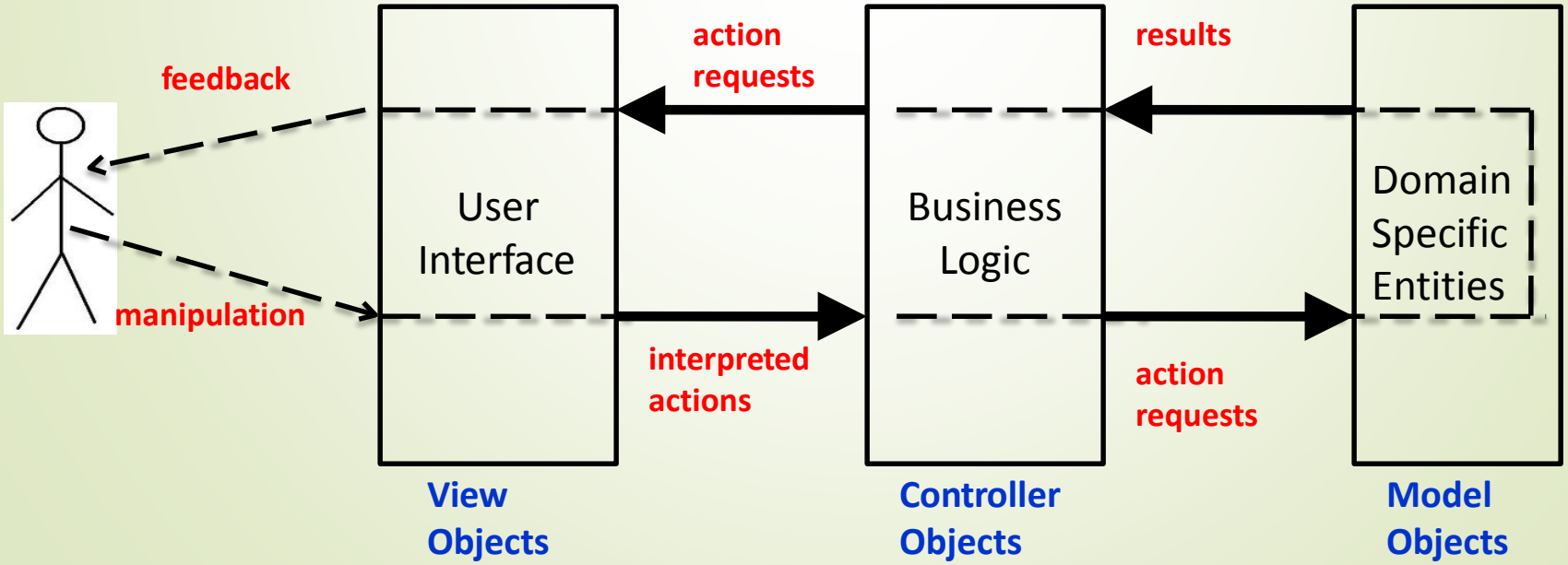
BT= Bank Terminal Dialogue **AS= Account Selection Dialogue**



- **Objects shown as rectangles at top, object interaction shown by arrows and messages**
- **Tail of the arrow shows the invoking object and time axis runs from top to bottom**

Converting UI Specification Into Language Specific Design

- Each application consists of three types of objects
- Domain or **model objects** represent the real world (also called entity objects)
- Model objects are controlled by **controller objects** and are unaware of view objects.
- **View objects** are on boundary of system interacting with end user.
- View objects only knows how to present feedback and accept user inputs. They do not take decisions or process information.
- The controller objects receive the request from view objects which makes application specific decision through model objects. This is called MVC architecture as shown below



Converting UI Specification Into Language Specific Design

- Every dialogue in analysis model forms view object in design model e.g **Balance (View), Bill (View), Application Control (View)**
  - Dialogue components become object members of respective view object.
  - There is a single controller object for every view object
  - Single controller may be related to more that one model object and visa versa.
  - Manipulation(set) and feedback(get) behavior of component is implemented as member functions of the view component class. Query methods are designed for controller object.
  - Application specific c logic is implemented in controller classes.
  - Model classes have domain specific knowledge .
  - Event traces and state machines in analysis phase are used to find member function of design classes.
  - Design model in the next slide depicts all classes of the application

Fig showing all classes in the Design Model