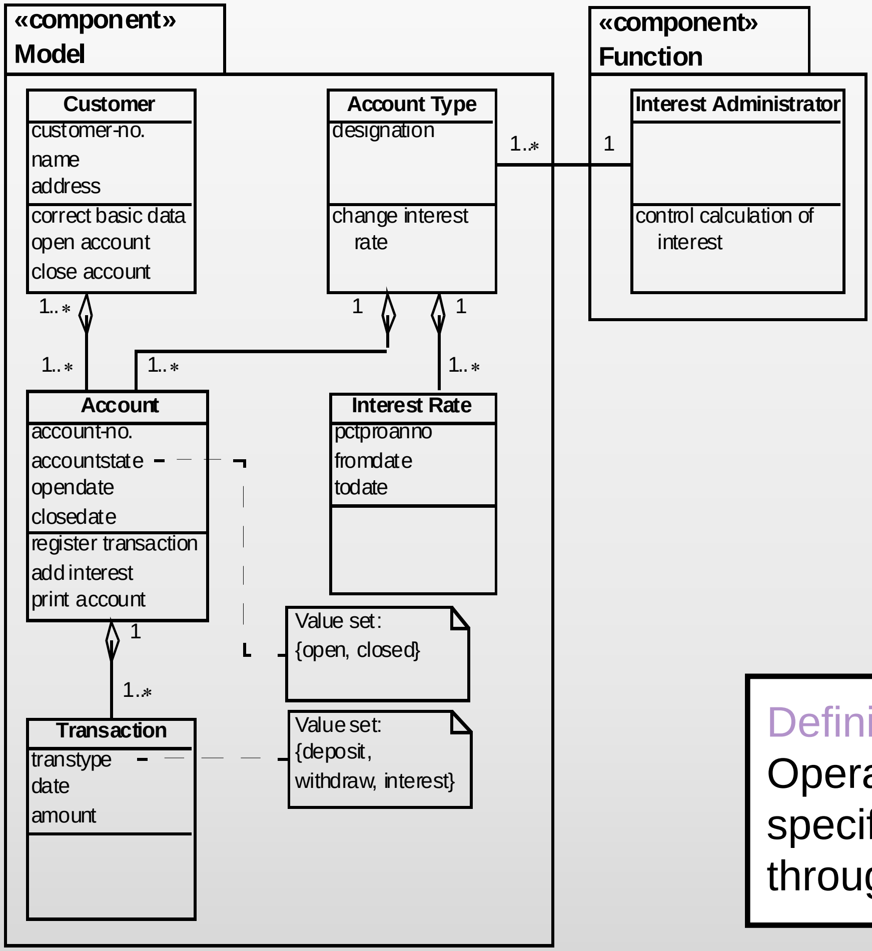
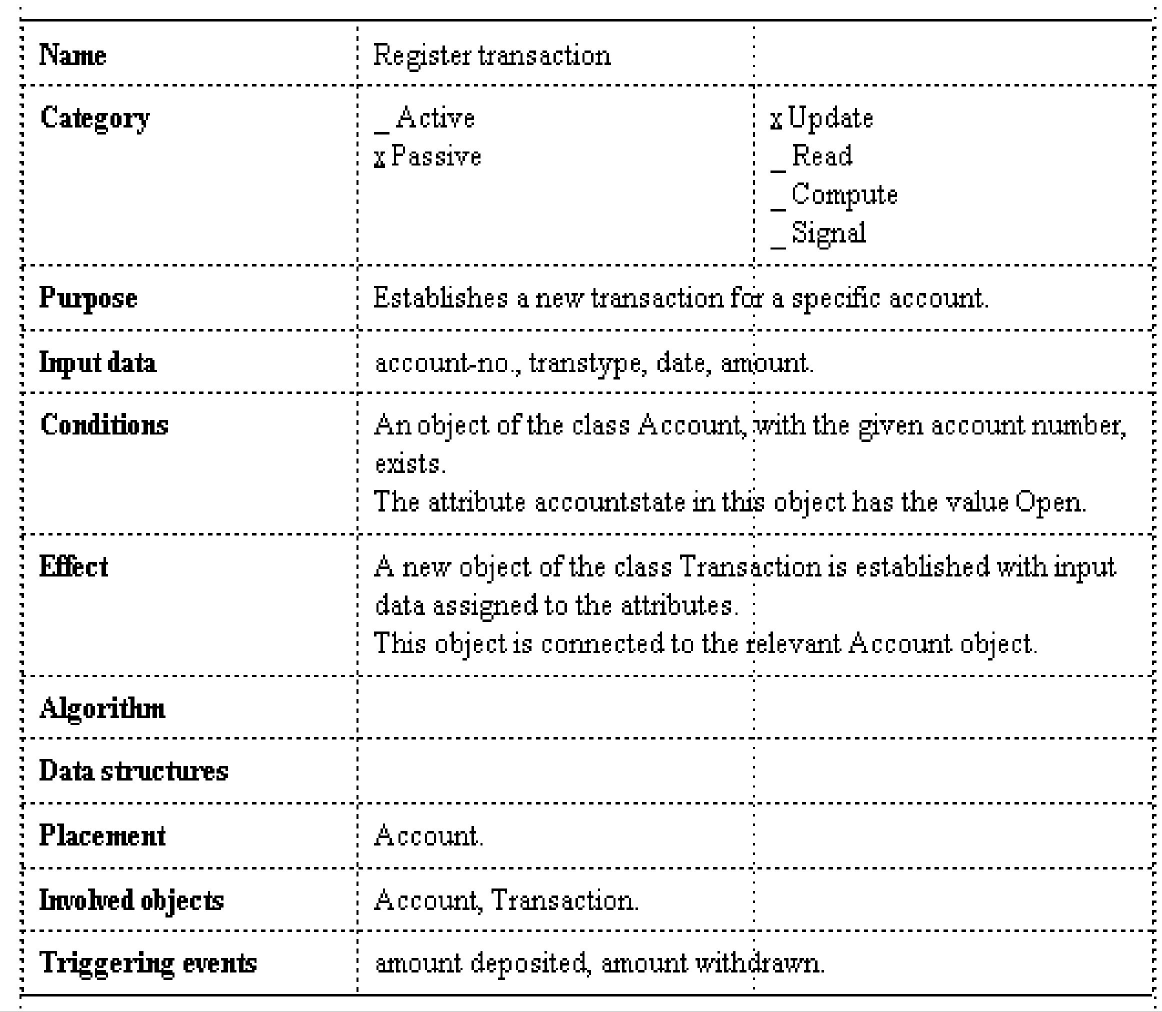
Function Component: Results



- Continue on the class diagram from design of the model component
- Extend with operations realizing requirements to functions from the analysis of the application domain

Definition:
Operation: A process property specified in a class and activated through the class' object

Secondary Result: Specification

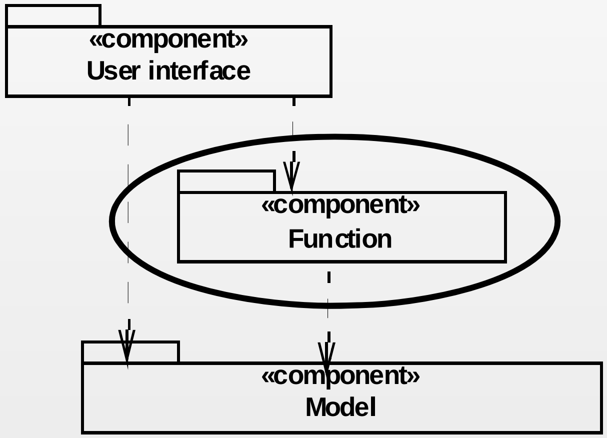


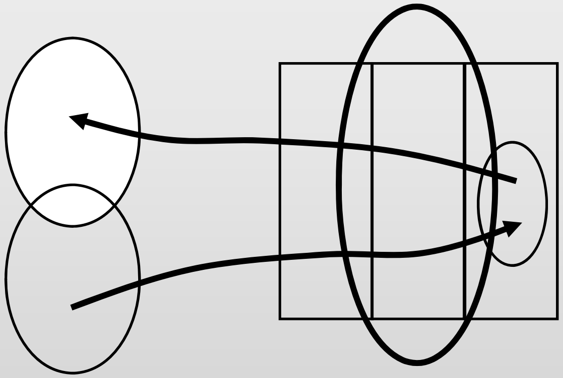
Name	Register transaction	
Category	_ Active x Passive	x Update _ Read _ Compute _ Signal
Purpose	Establishes a new transaction for a specific account.	
Input data	account-no., transtype, date, amount.	
Conditions	An object of the class Account, with the given account number, exists. The attribute accountstate in this object has the value Open.	
Effect	A new object of the class Transaction is established with input data assigned to the attributes. This object is connected to the relevant Account object.	
Algorithm		
Data structures		
Placement	Account.	
Involved objects	Account, Transaction.	
Triggering events	amount deposited, amount withdrawn.	

Function Component: Responsibility

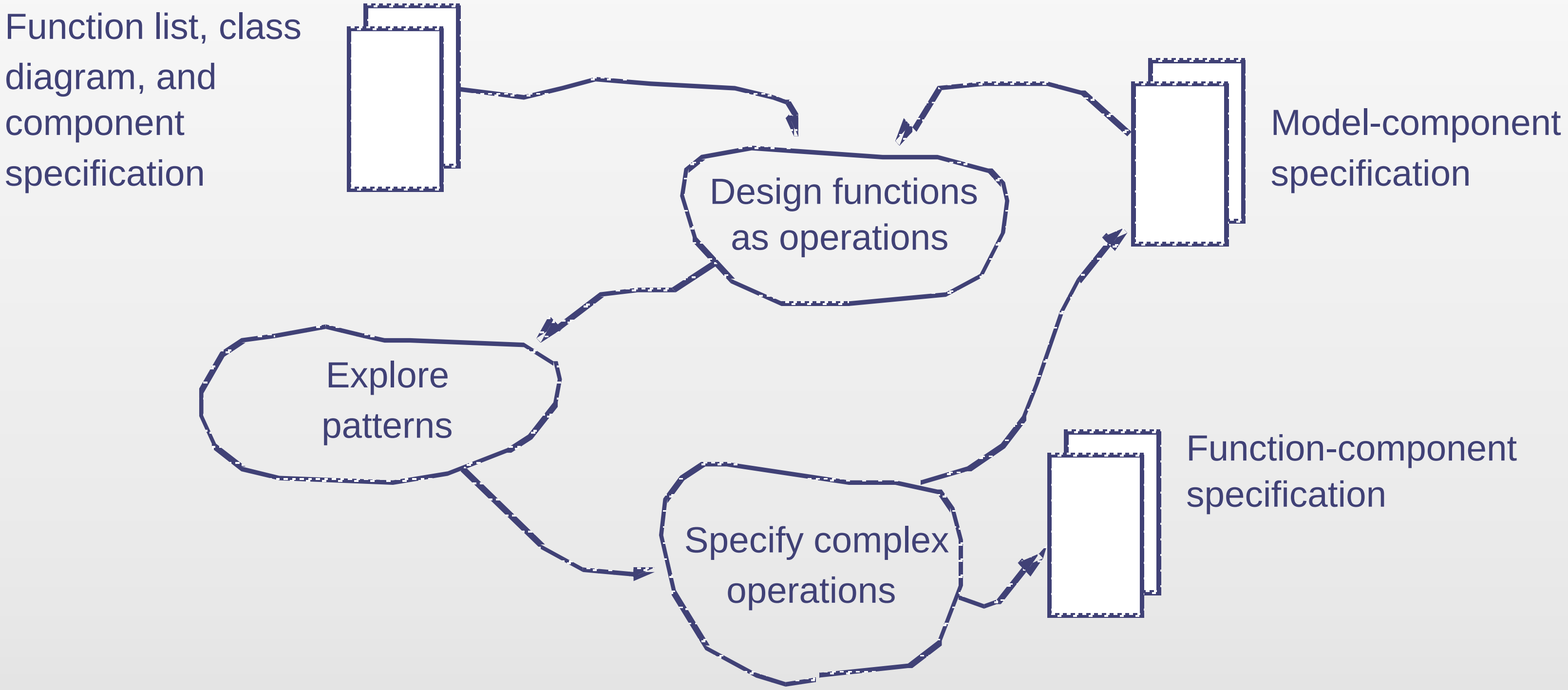
Component:
A collection of program parts that constitutes a whole and has well-defined responsibilities

Responsibility of the function component:
Make the model component available as a ressource to actors

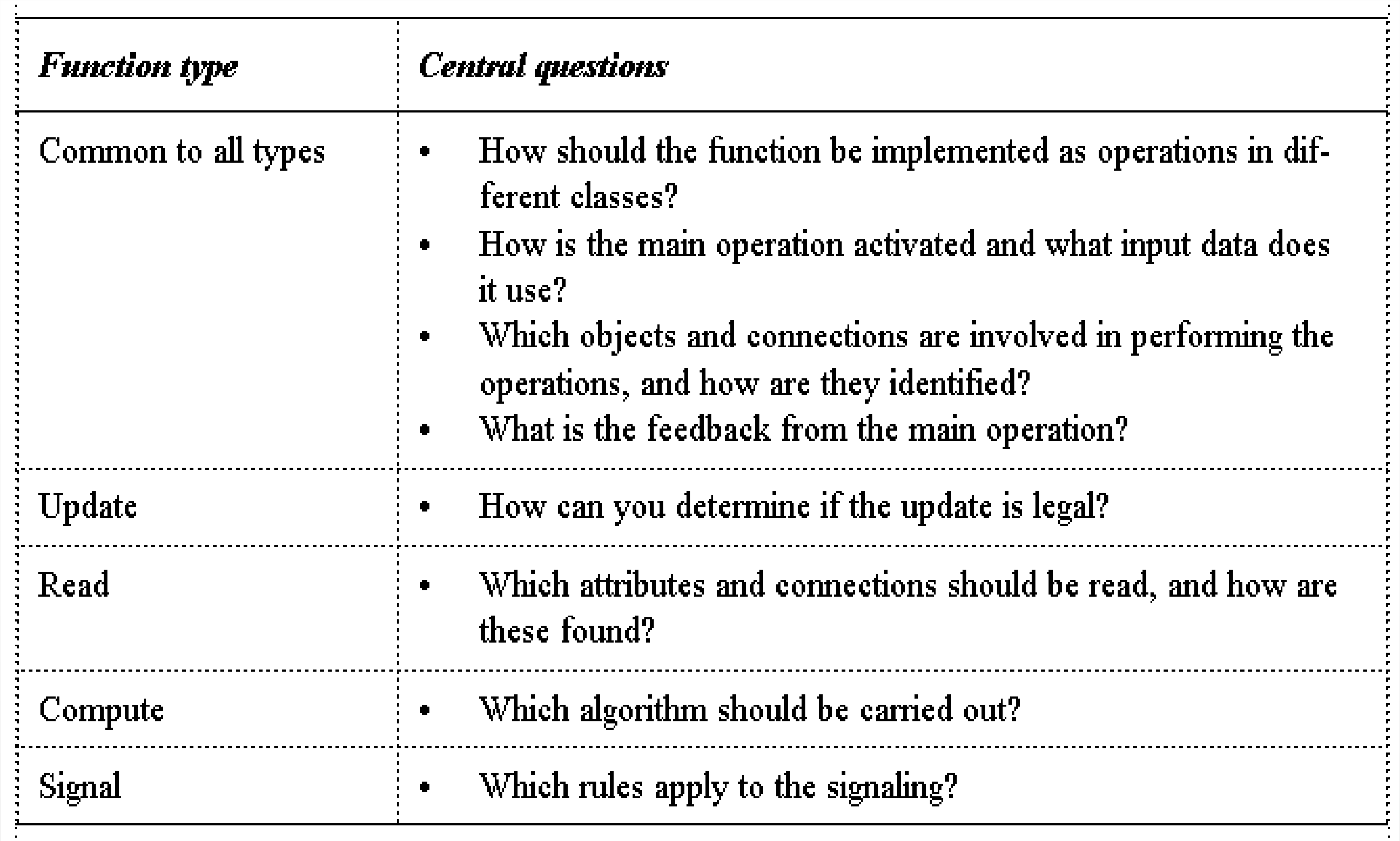




Function Component: Activities


Function list, class diagram, and component specification
Design functions as operations
Model-component specification
Explore patterns
Specify complex operations
Function-component specification

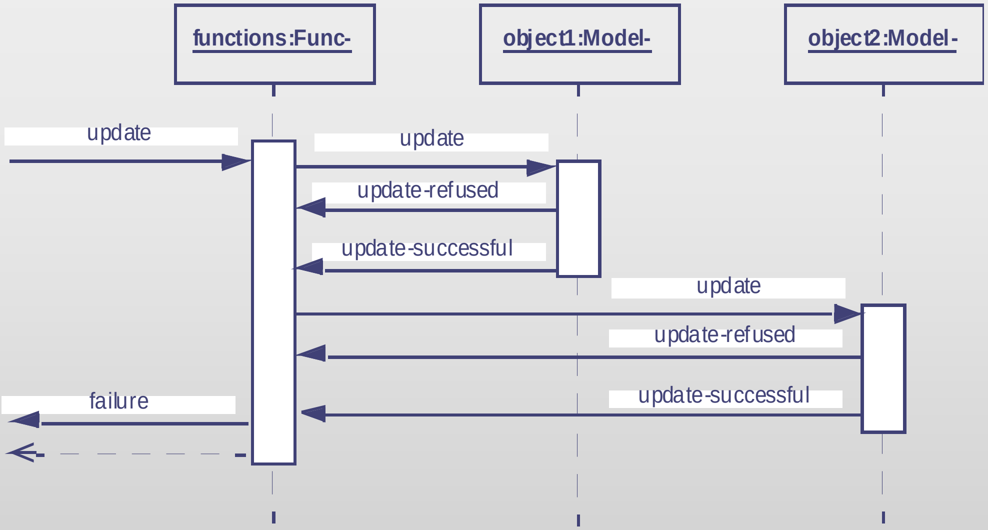
Design Functions as Operations



Function type	*Central questions*
Common to all types	• How should the function be implemented as operations in different classes? • How is the main operation activated and what input data does it use? • Which objects and connections are involved in performing the operations, and how are they identified? • What is the feedback from the main operation?
Update	• How can you determine if the update is legal?
Read	• Which attributes and connections should be read, and how are these found?
Compute	• Which algorithm should be carried out?
Signal	• Which rules apply to the signaling?

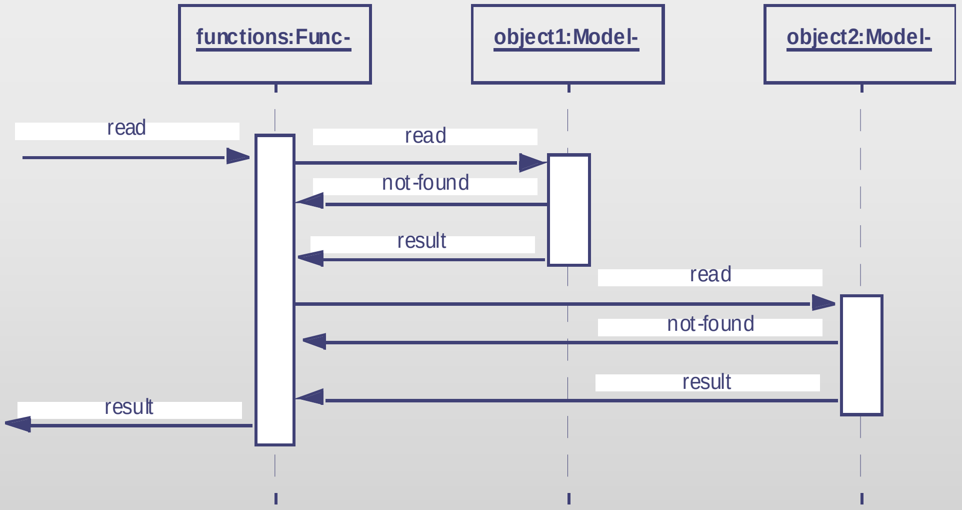
Update

- A relevant event in the problem domain must leave a trace in the model.
- Recieves input describing the event.
- Primary effect is an updated model.
- Identify relevant objects and connections by means of the events.
- The update operation should check that the event is legal.



Read

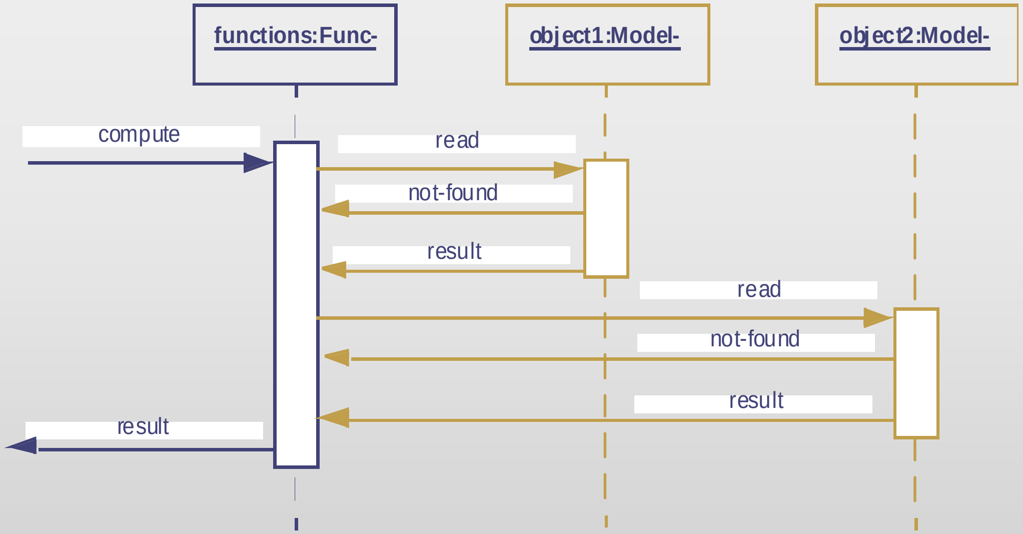
- An actor needs information about the model.
- Input describes the wanted reading and seection of objects.
- Describe the necessary operations.
- Can be done through precedence analaysis backward from expected output.



Compute

- An actor needs a computation performed.
- Can also encompass reading and updating

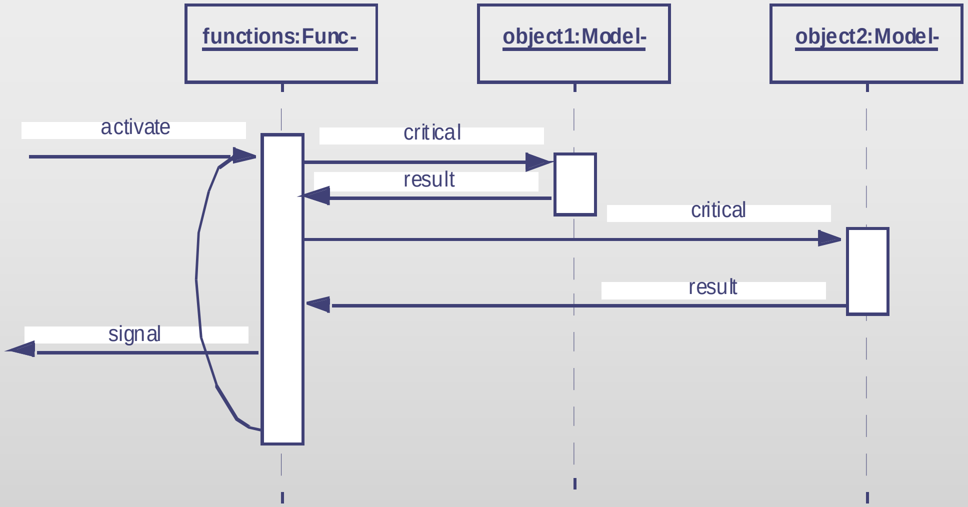
- Describe input and readings of the model.
- Describe algorithm possibly through decomposition.



Signal

- An actor needs to monitor or control a part of the problem domain.
- The critical state is read from the model.

- Typically few inputs.
- Identify the state transitions that might require a signal.
- Determine how to signal.
- Is the signaling active or passive?



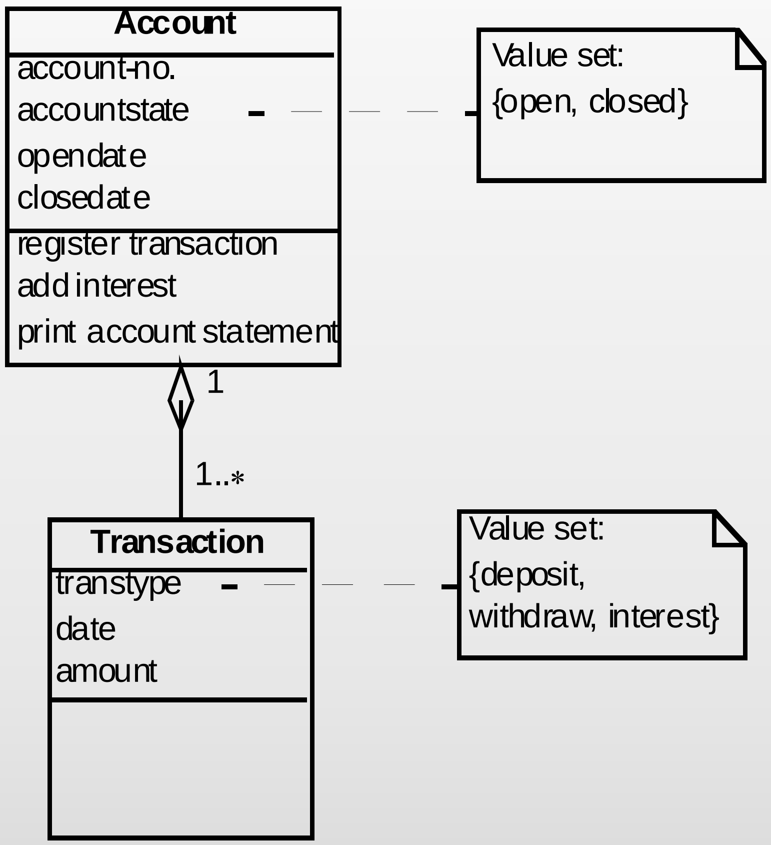
Explore Patterns

The patterns specify how functions can be realized as a set of operations:

- Model-Class Placement
- Function-Class Placement
- Strategy
- Active Function

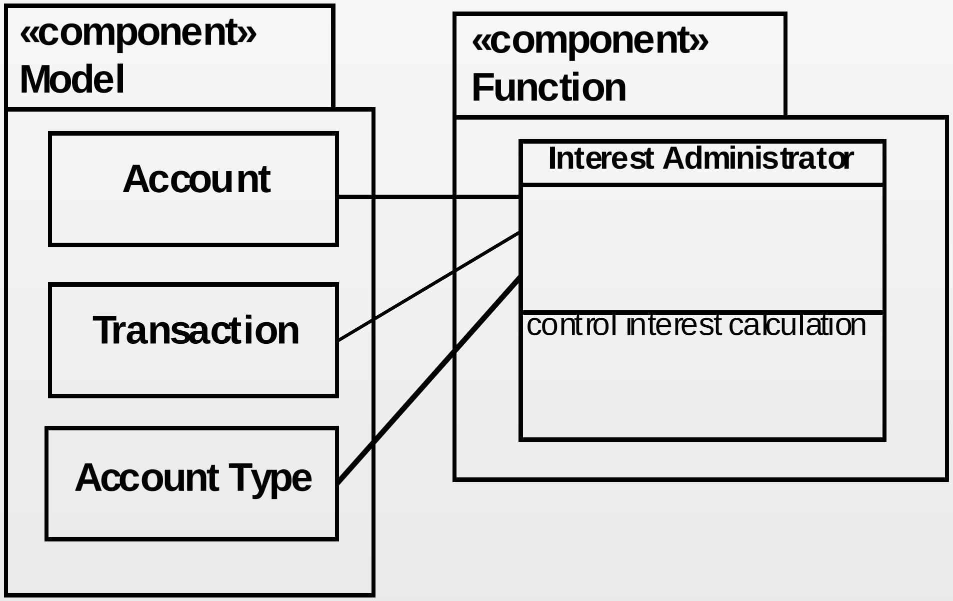
Pattern: Model-Class Placement

- A number of operations are specified on class Account. That again is realized through several operations:
  - Transaction registration (update)
  - Calculate interests (compute) and deposit interests (update)
  - Print account statement (read)



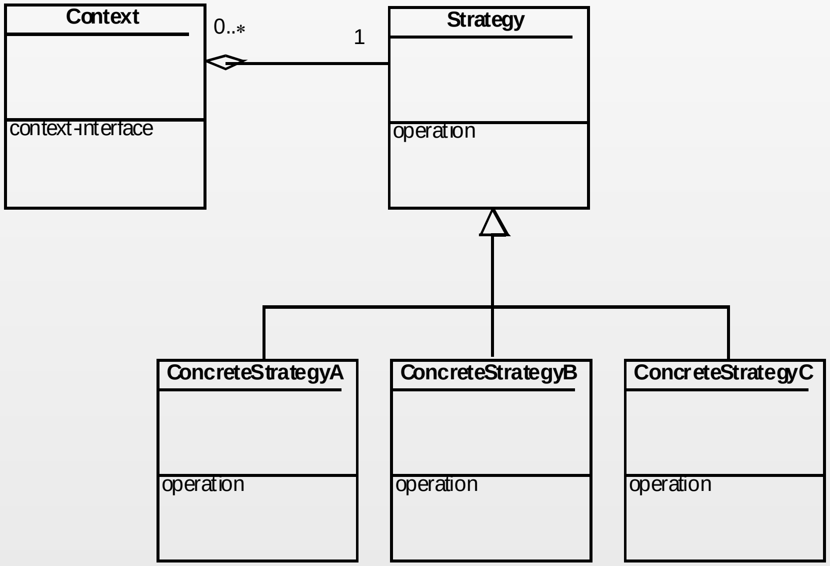
Pattern: Function-Class Placement

- Some operations cannot be placed on a class in the model.
- Typically functions that operate on several objects.
- A new class is then designed in the function component. That class contains the operation that realizes the function.



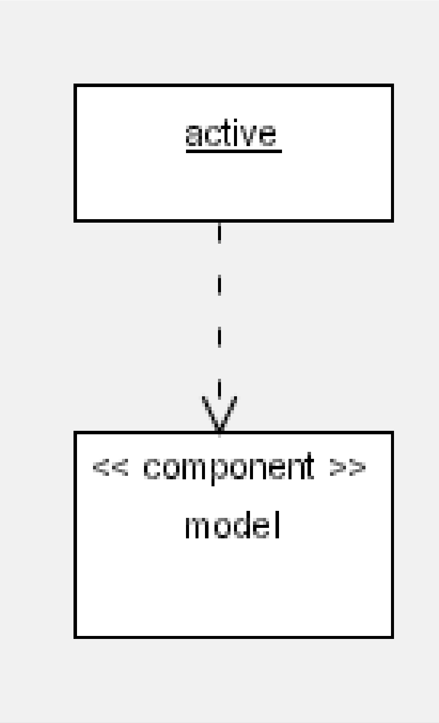
Pattern: Strategy

- If a class has several specializations and a function is performed different dependent on each specialization.
- The Stategy Pattern defines a general operation that is then described in detail in each specialization.



Pattern: Active Function

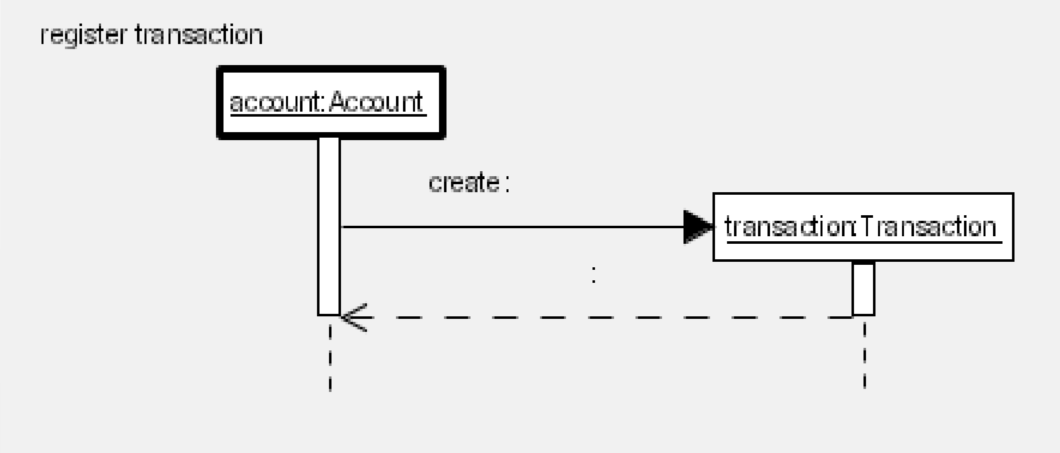
- A signal function can be active or passive.
- An active function can be realized in an active object.
- The function is then realized with its own control.

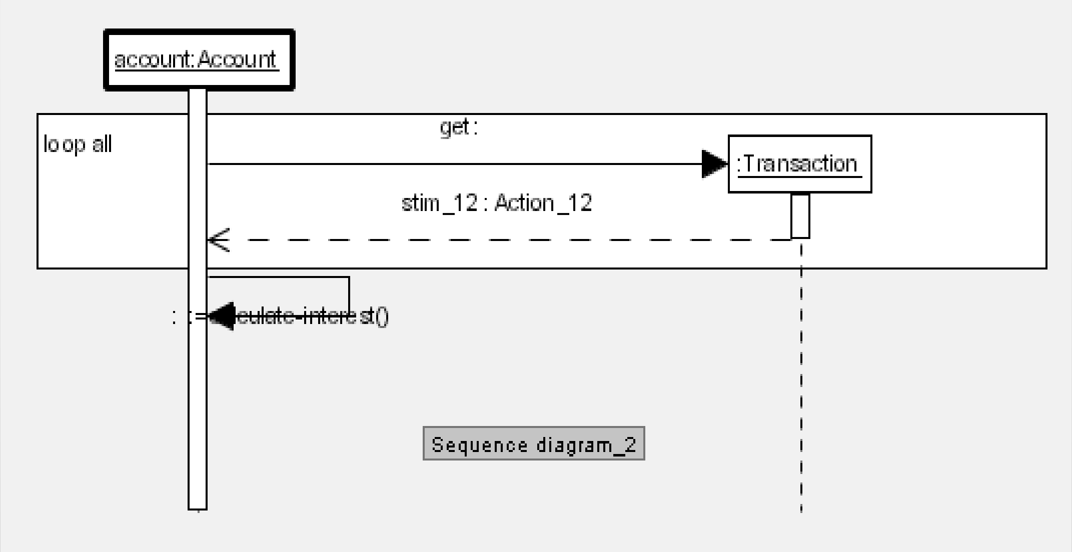


Specify Complex Operations

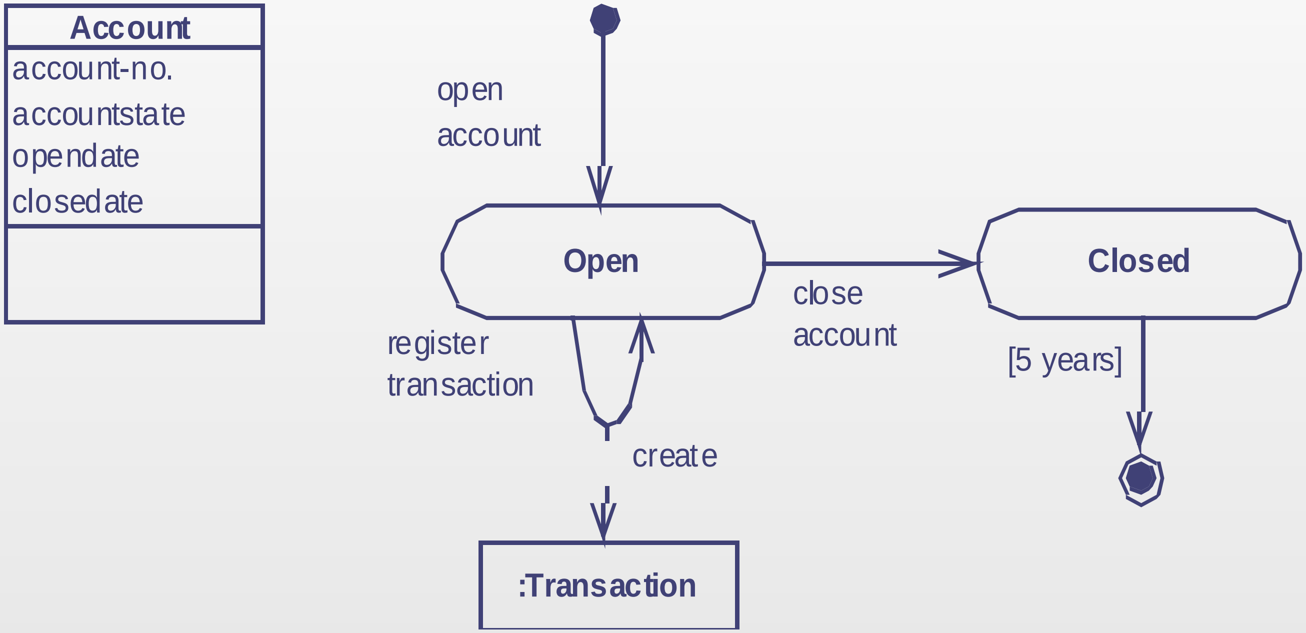
- Implicit specification by an attribute in the class
- Naming the operation in the class
- Operation Specification
- Sequence Diagram
- Statechart for a class
- Statechart for a system

Sequence Diagram

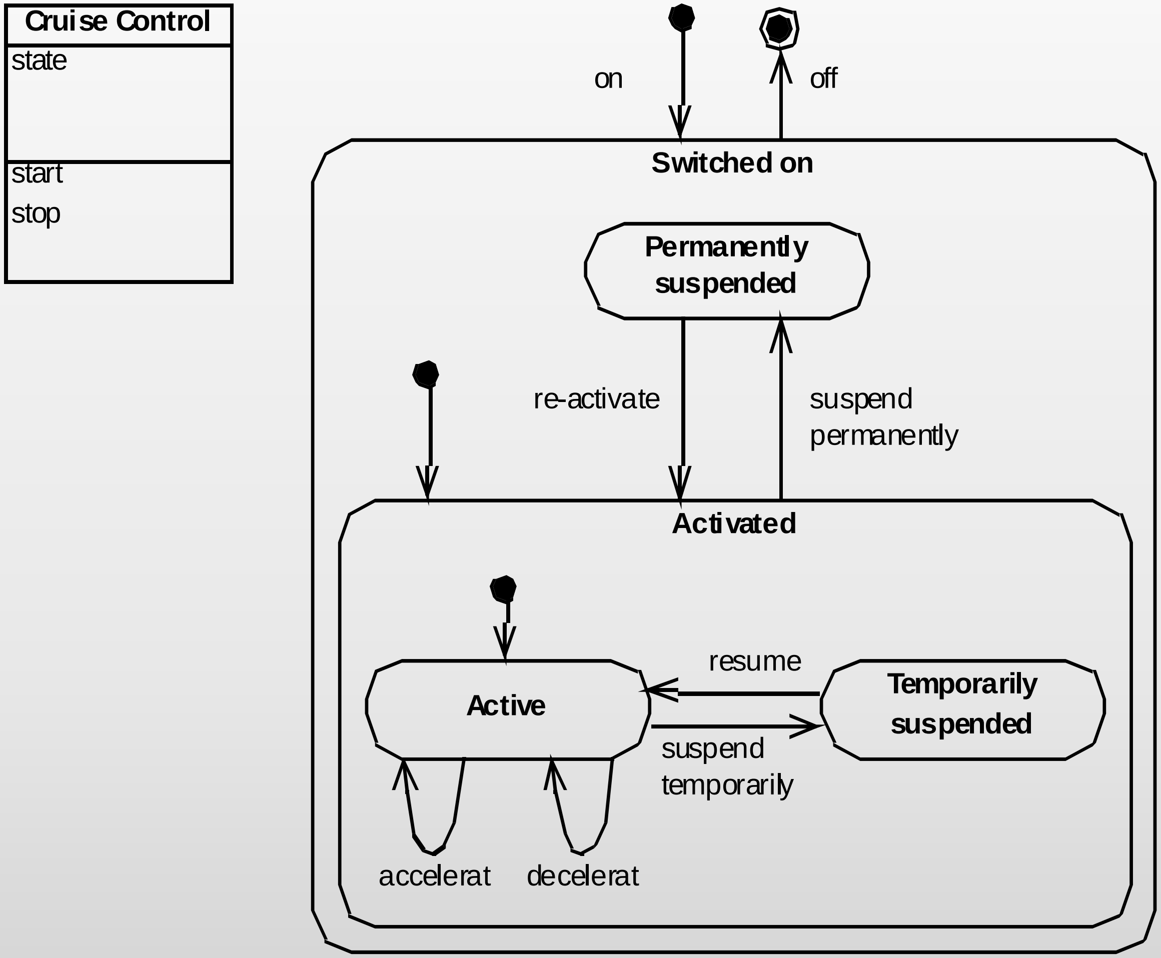




Statechart for a Class



Statechart for the System's Total Behavior



Function Component: Summary

Purpose	• To determine the implementation of functions.
Concepts	• Function component: A part of a system that implements functional requirements. • Operation: A process property specified in a class and activated through the class' objects.
Principles	• Base the design on function types. • Specify complex operations.
Results	• A class diagram with operations and specifications of complex operations.