MODELING - BEHAVIOR

SU:L6:E16

AGENDA

- **Elevator example: Structure**
- **Problem domain analysis: Behavior**

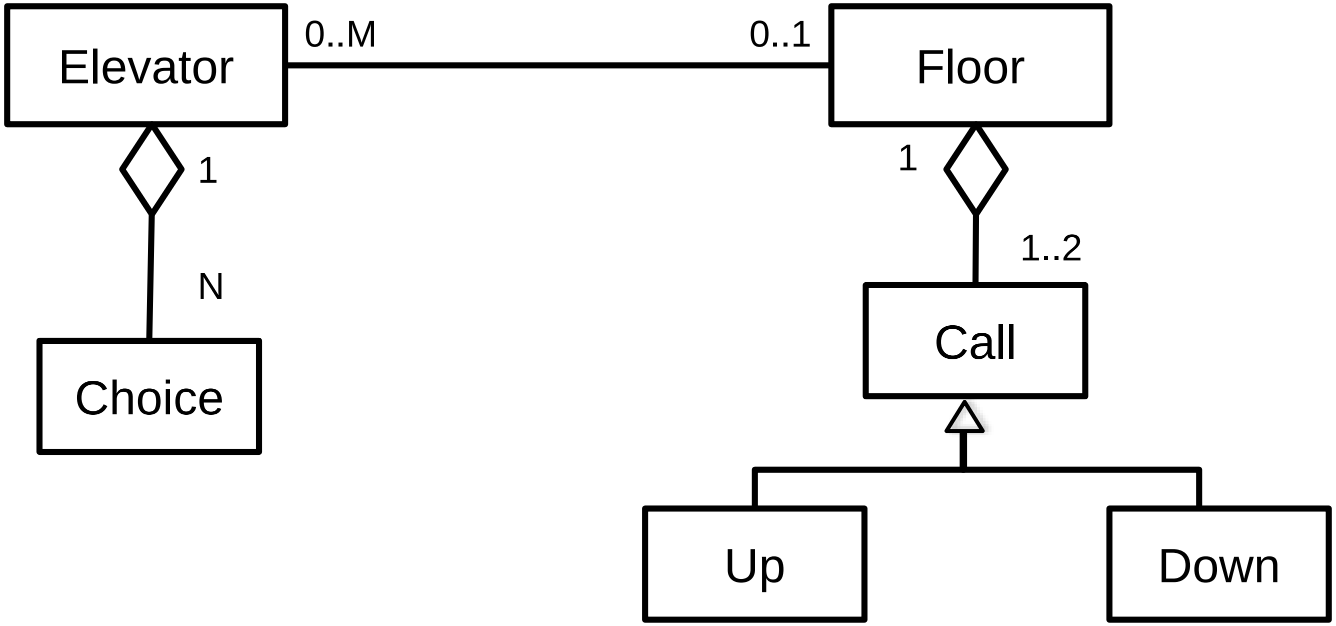
ELEVATOR – EVENT TABLE

- **Context:**
  - The system is used to decide where an elevator must stop based on the requests of users; e.g. “go to floor 3”. Other systems control the movements of the elevator, its speed, direction and position between floors.

	Elevator	Floor	Choice	Call down	Call up
Left down	X	X		X	
Left up	X	X			X
Arrived	X	X	X		
Called down		X		X	
Called up		X			X
Floor chosen			X		

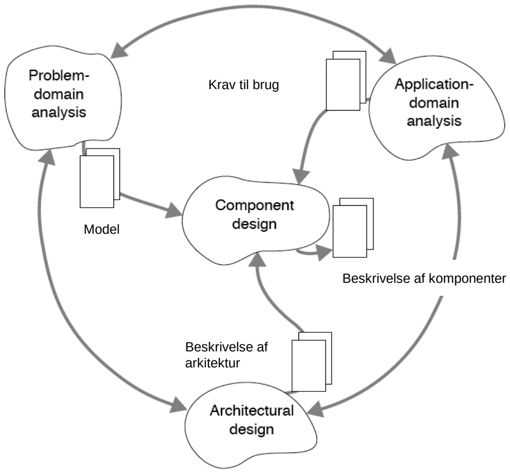
ELEVATOR – CLASS DIAGRAM

- **Generalization of the buttons (calls) at the floors**



ANALYSIS OF BEHAVIOR

ACTIVITIES IN 'PROBLEM DOMAIN ANALYSIS '


Problem-domain analysis
Krav til brug
Application-domain analysis
Model
Component design
Beskrivelse af komponenter
Beskrivelse af arkitektur
Architectural design

Classes

- Which objects and events are part of the problem domain?
- Class, object, and event.

Structure

- How are classes and objects conceptually tied together?
- Generalization, aggregation, association, and cluster.

Behavior

- What are the dynamic properties of objects?
- Event trace, behavioral pattern, and attribute.

An instantaneous incident involving one or more objects

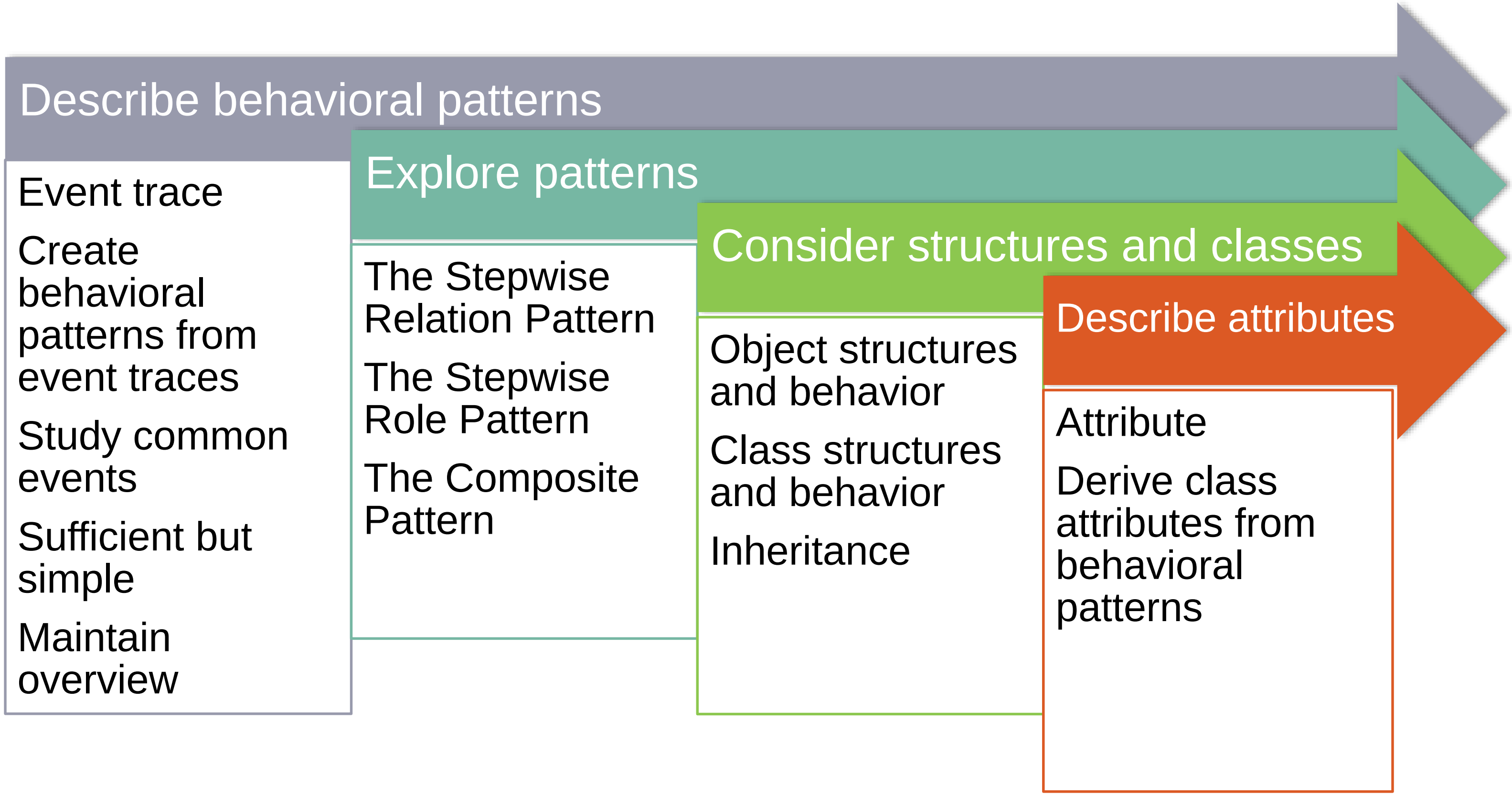
Event trace: sequence of events involving a specific object

Behavioral pattern: a description of possible event traces for all objects in a class

Attribute: a descriptive property of a class or an event

ACTIVITIES IN ‘BEHAVIOR’

Behavior: unordered set of events that involve an object



Describe behavioral patterns

- Event trace
- Create behavioral patterns from event traces
- Study common events
- Sufficient but simple
- Maintain overview

Explore patterns

- The Stepwise Relation Pattern
- The Stepwise Role Pattern
- The Composite Pattern

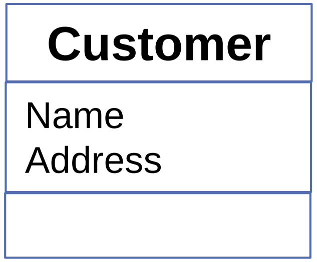
Consider structures and classes

- Object structures and behavior
- Class structures and behavior
- Inheritance

Describe attributes

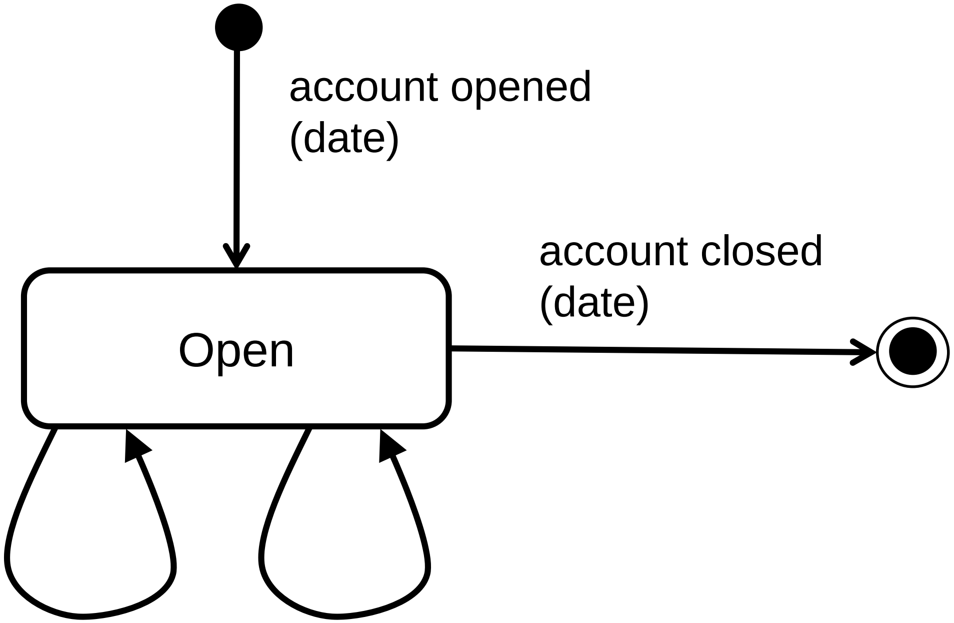
- Attribute
- Derive class attributes from behavioral patterns

RESULT (1)



Customer

Name
Address



account opened
(date)

Open

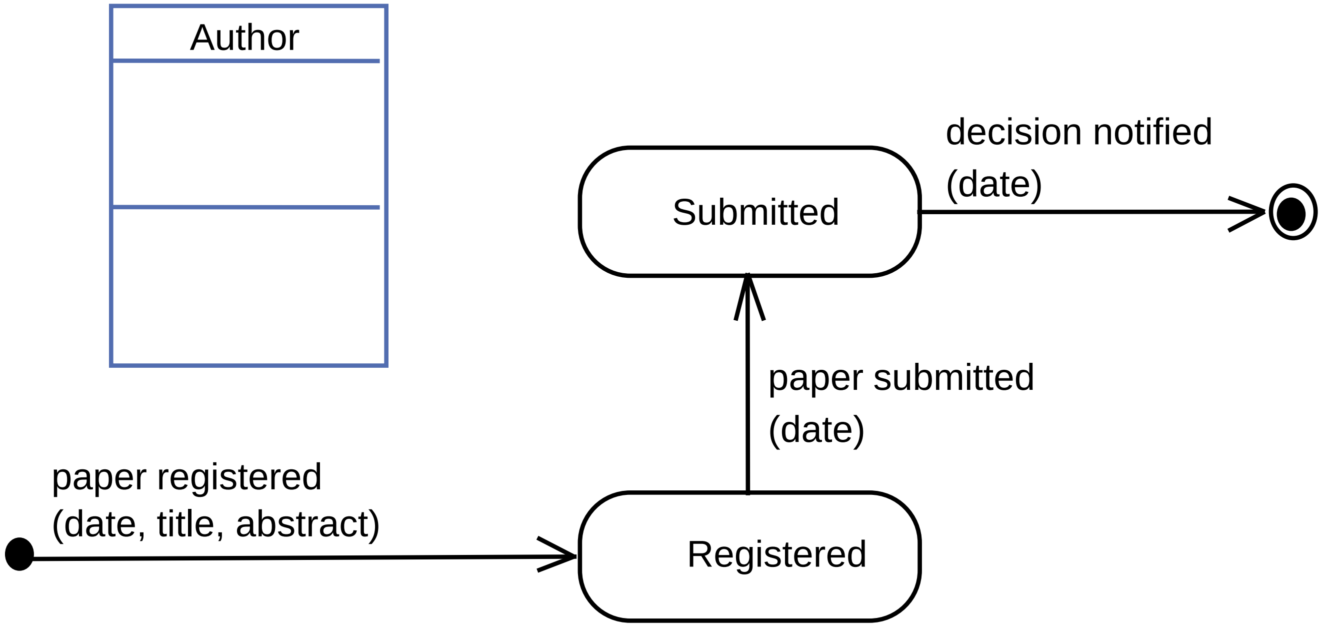
account closed
(date)

amount
deposited
(date, amount)

amount
withdrawn
(date, amount)

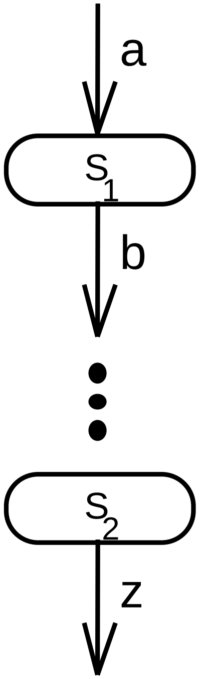
October 10, 2016

RESULT (2)

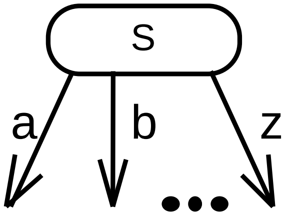

Author
decision notified
(date)
Submitted
paper submitted
(date)
paper registered
(date, title, abstract)
Registered

NOTATION: STATECHART DIAGRAMS

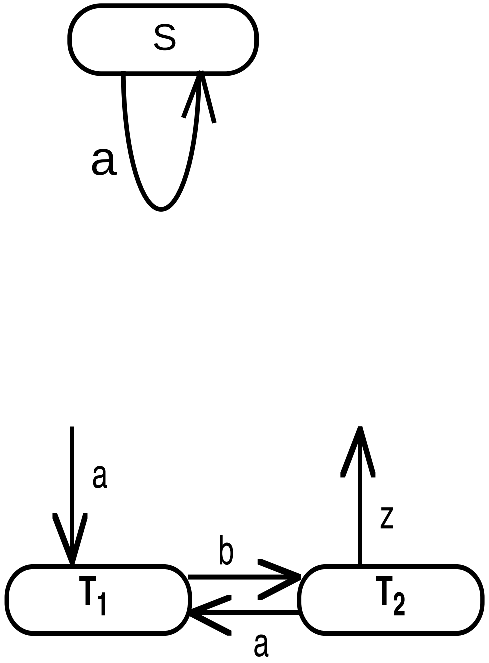
Sequence



Selection



Iteration



FIND BEHAVIORAL PATTERNS FROM EVENT TRACES

For each class we ask:

- Which events cause the creation of a problem-domain object?

  These events are grouped as selections that can cause the birth of an object.
- Which events cause the ‘death’ of a problem-domain object?

  These events are grouped as selection(s) that may lead to the object’s end-state.

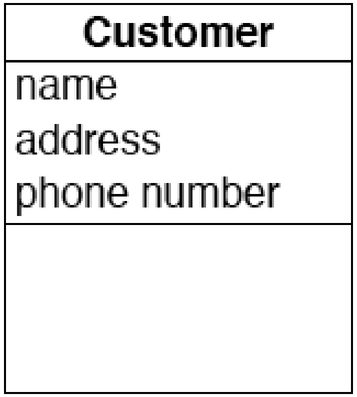
Typical event traces:

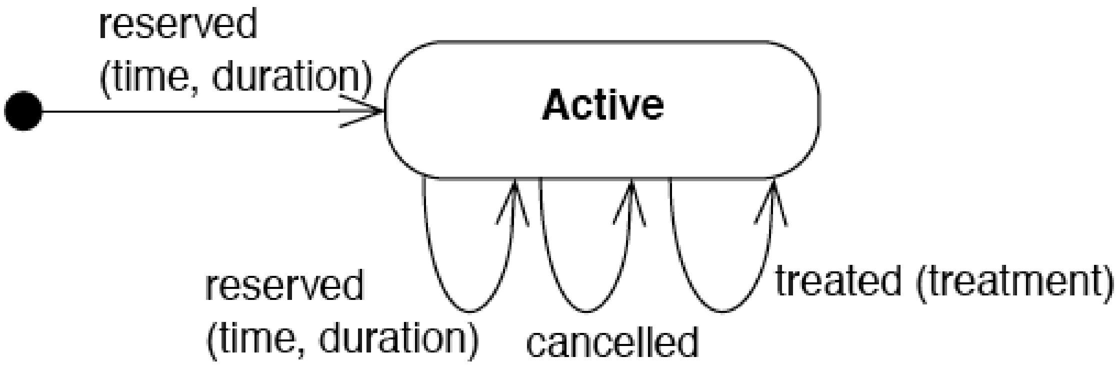
- Is the overall form structured or unstructured?
- Which events occur together in a sequence?
- Are there any alternative events?
- Can a given event occur more than once?

The death of a problem-domain object does not mean that the object ceases to exist. It only implies that the object can no longer be involved in events.

CUSTOMER

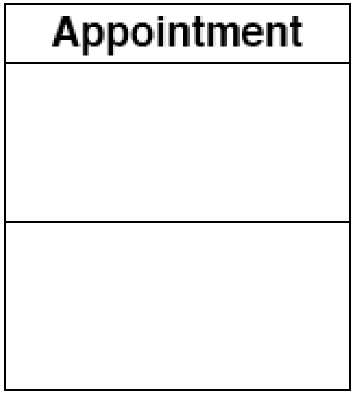
	Customer	Assistant	Apprentice	Appointment	Plan
reserved	X	X		X	X
cancelled	X	X		X	
treated	X			X	
employed		X	X		
resigned		X	X		
graduated			X		
agreed		X	X		X

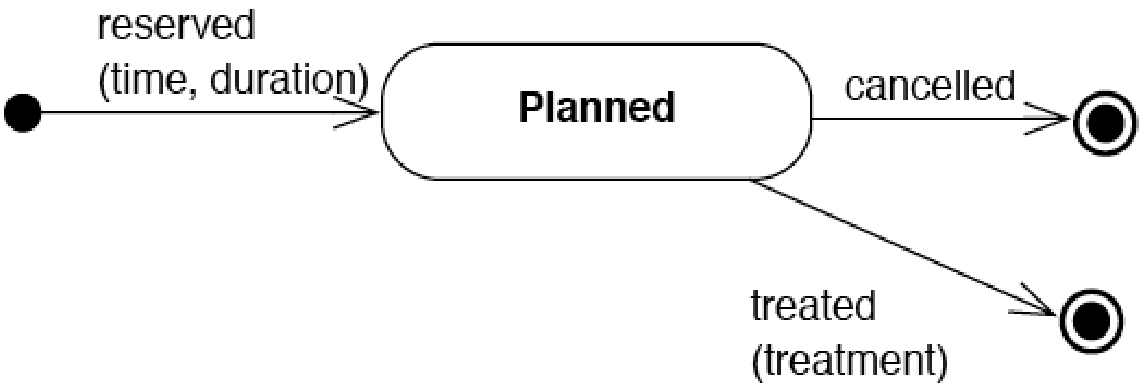

Customer
name
address
phone number


reserved (time, duration)
Active
reserved (time, duration)
cancelled
treated (treatment)

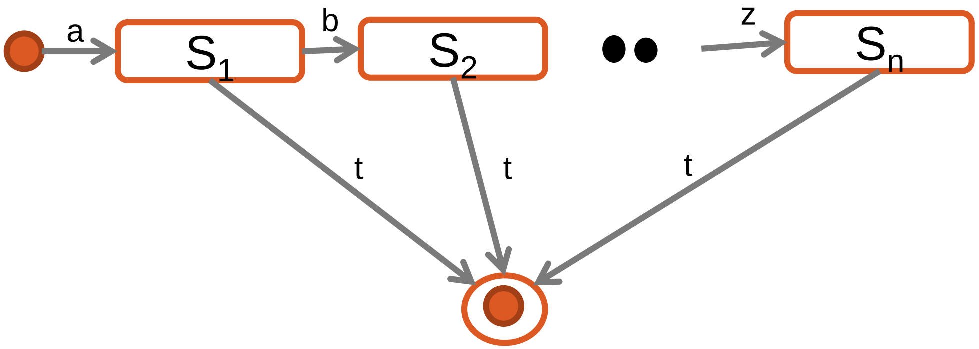
APPOINTMENT

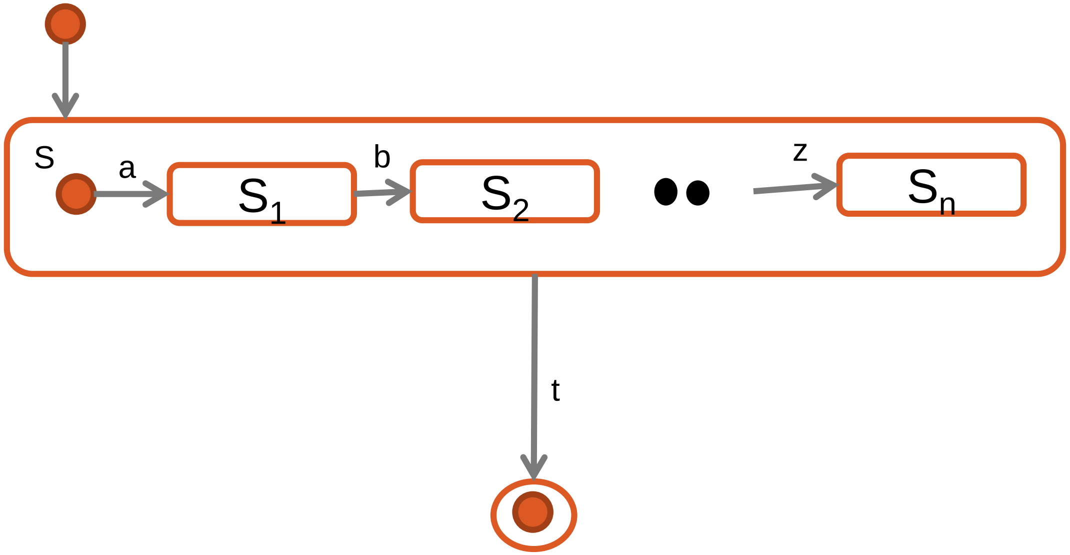
	Customer	Assistant	Apprentice	Appointment	Plan
reserved	X	X		X	X
cancelled	X	X		X	
treated	X			X	
employed		X	X		
resigned		X	X		
graduated			X		
agreed		X	X		X





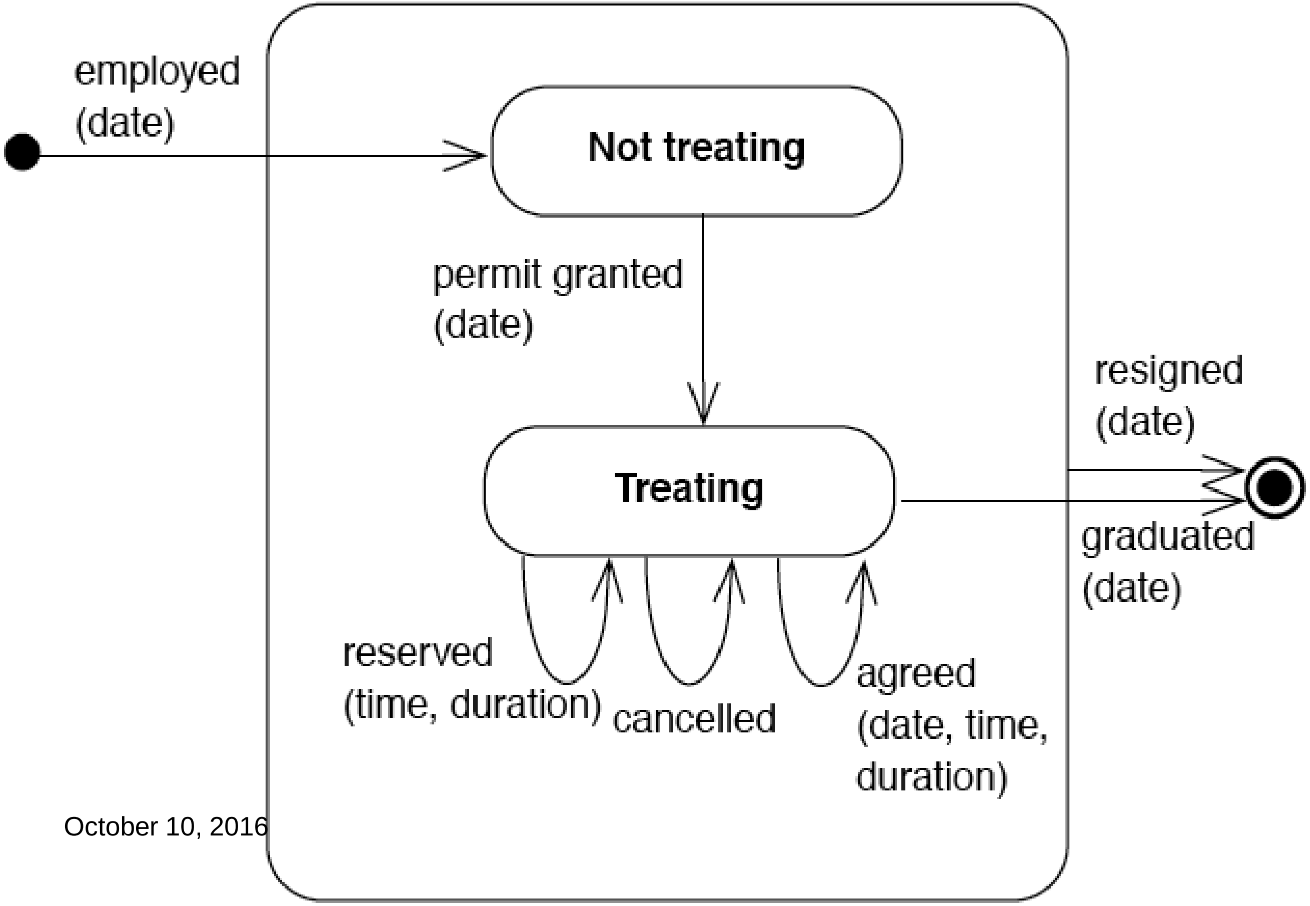
HIERARCHICAL STATES

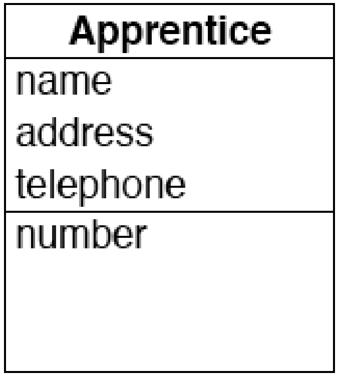




APPRENTICE

	Customer	Assistant	Apprentice	Appointment	Plan
reserved	X	X		X	X
cancelled	X	X		X	
treated	X			X	
employed		X	X		
resigned		X	X		
graduated			X		
agreed		X	X		X



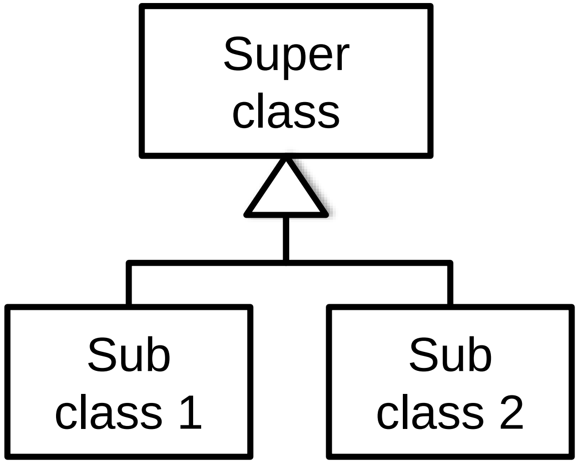


Apprentice

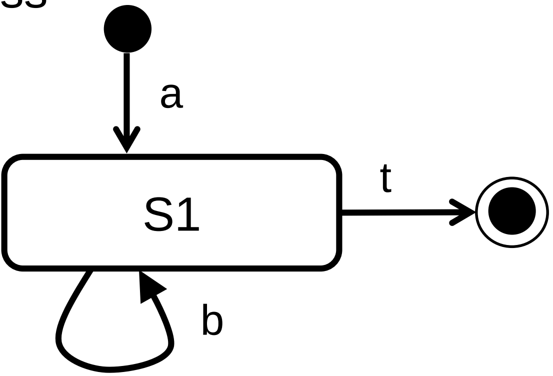
name
address
telephone

number

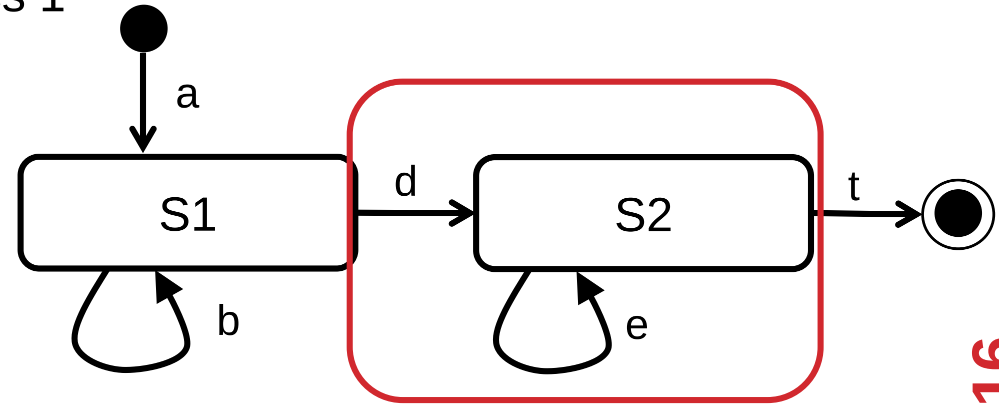
INHERITANCE OF BEHAVIORAL PATTERNS



Super class

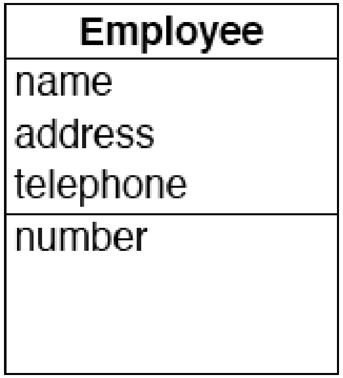


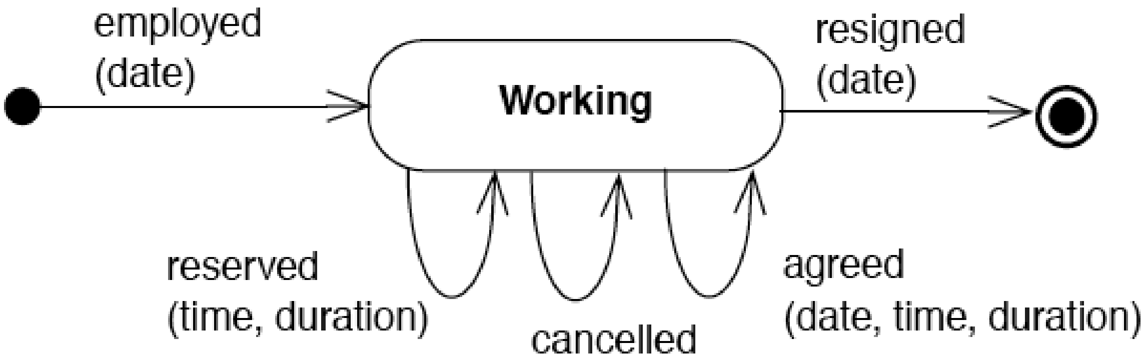
Sub class 1



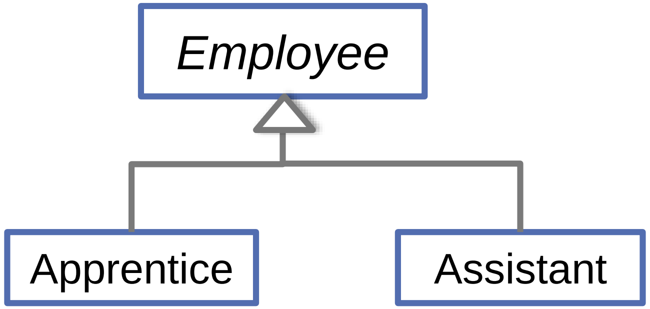
EMPLOYEE (ASSISTANT)

	Customer	Assistant	Apprentice	Appointment	Plan
reserved	X	X		X	X
cancelled	X	X		X	
treated	X			X	
employed		X	X		
resigned		X	X		
graduated			X		
agreed		X	X		X





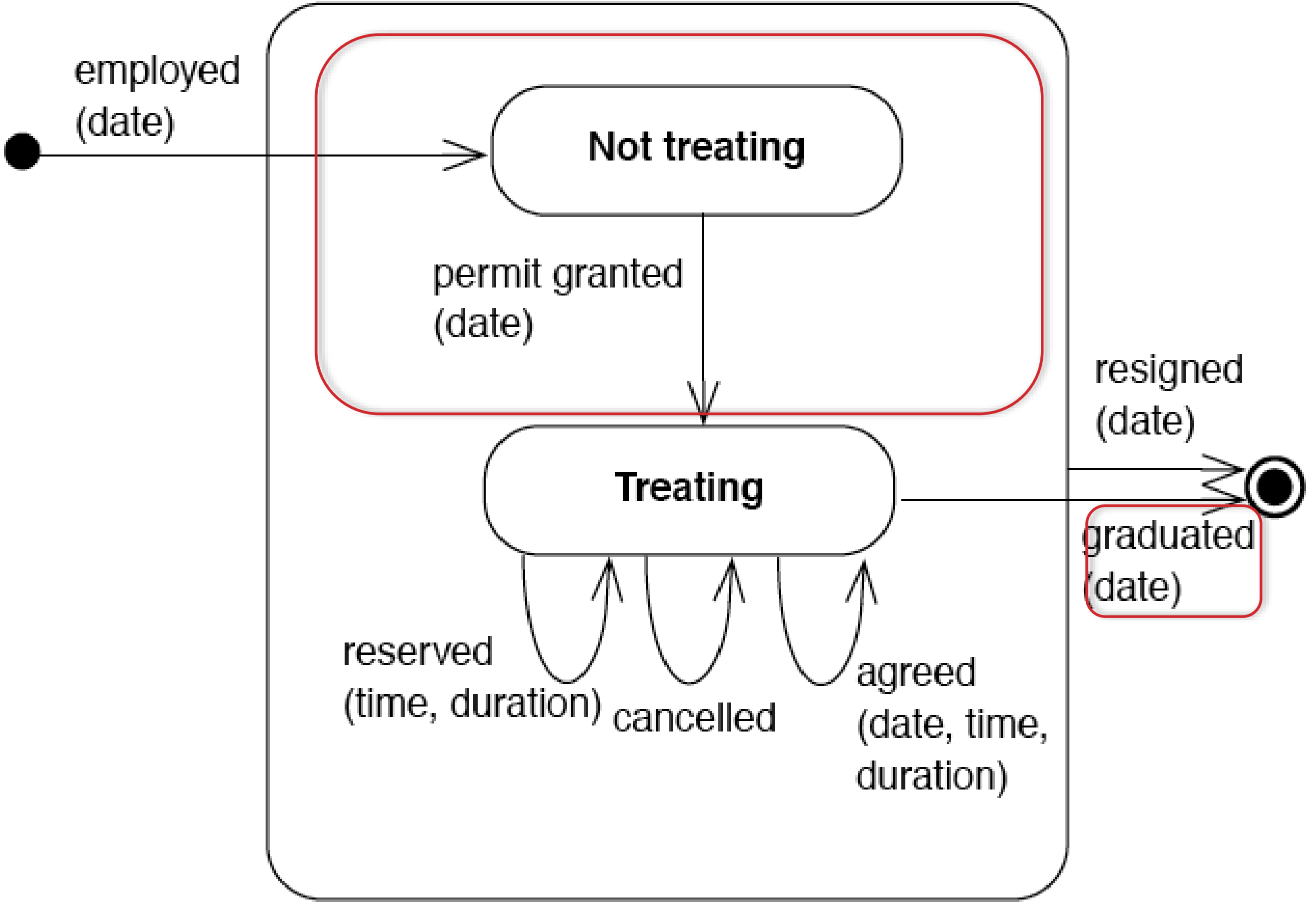
INHERITANCE – *EMPLOYEE* AND APPRENTICE



Employee

Apprentice

Assistant



employed (date)

Not treating

permit granted (date)

Treating

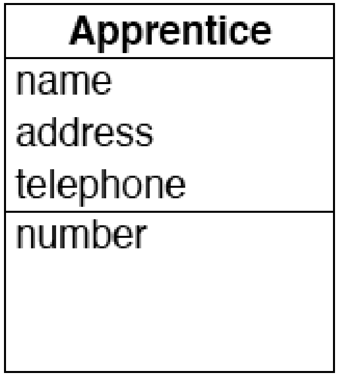
resigned (date)

graduated (date)

reserved (time, duration)

cancelled

agreed (date, time, duration)



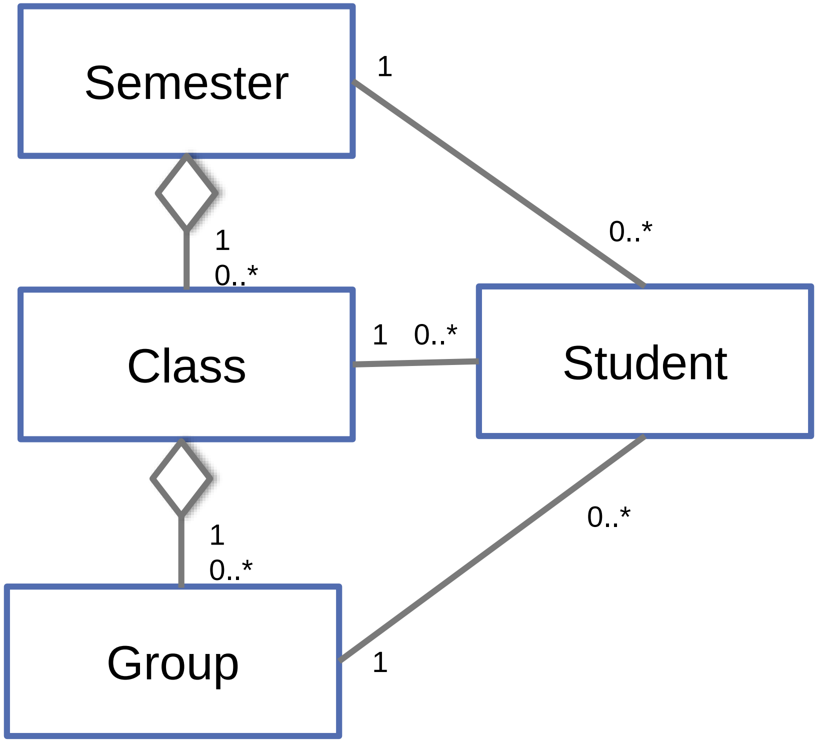
Apprentice

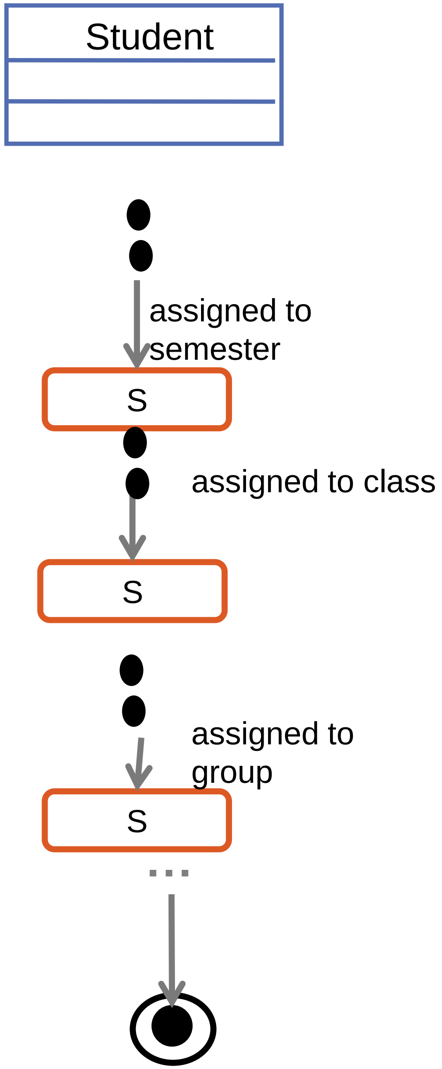
name
address
telephone

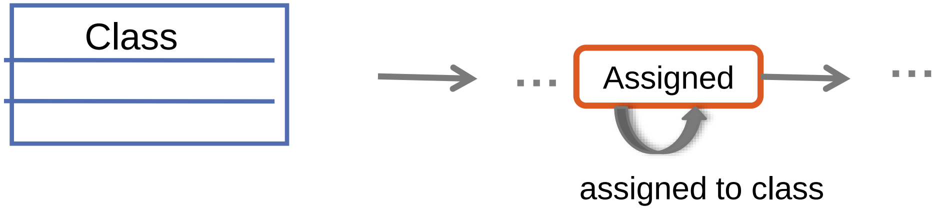
number

EXPLORE PATTERNS: STEPWISE RELATION

Describe behavior when objects are related to the elements of a hierarchy in a sequential manner

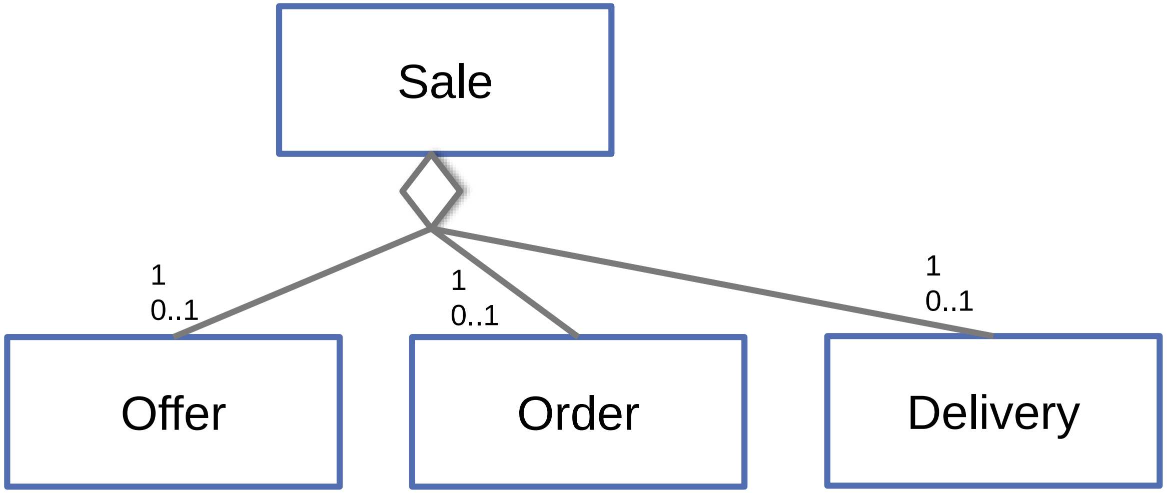


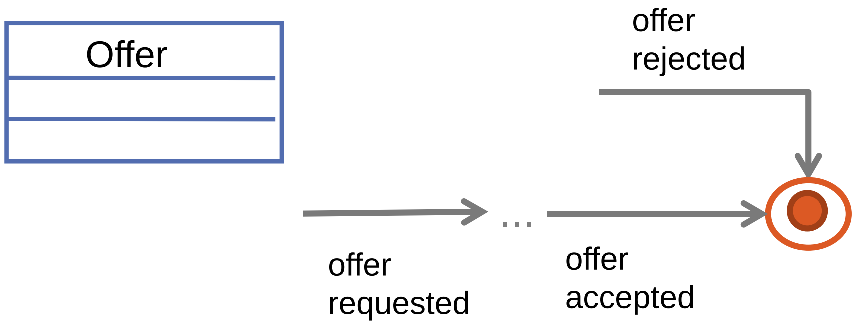


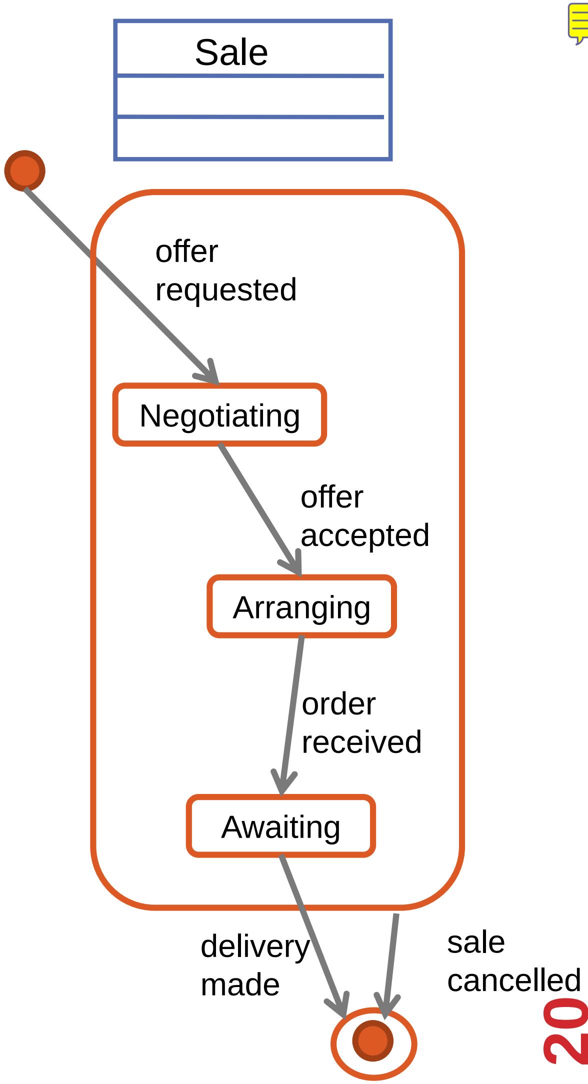


EXPLORE PATTERNS: STEPWISE ROLE

Describes how the behaviour of a whole changes as its parts become active

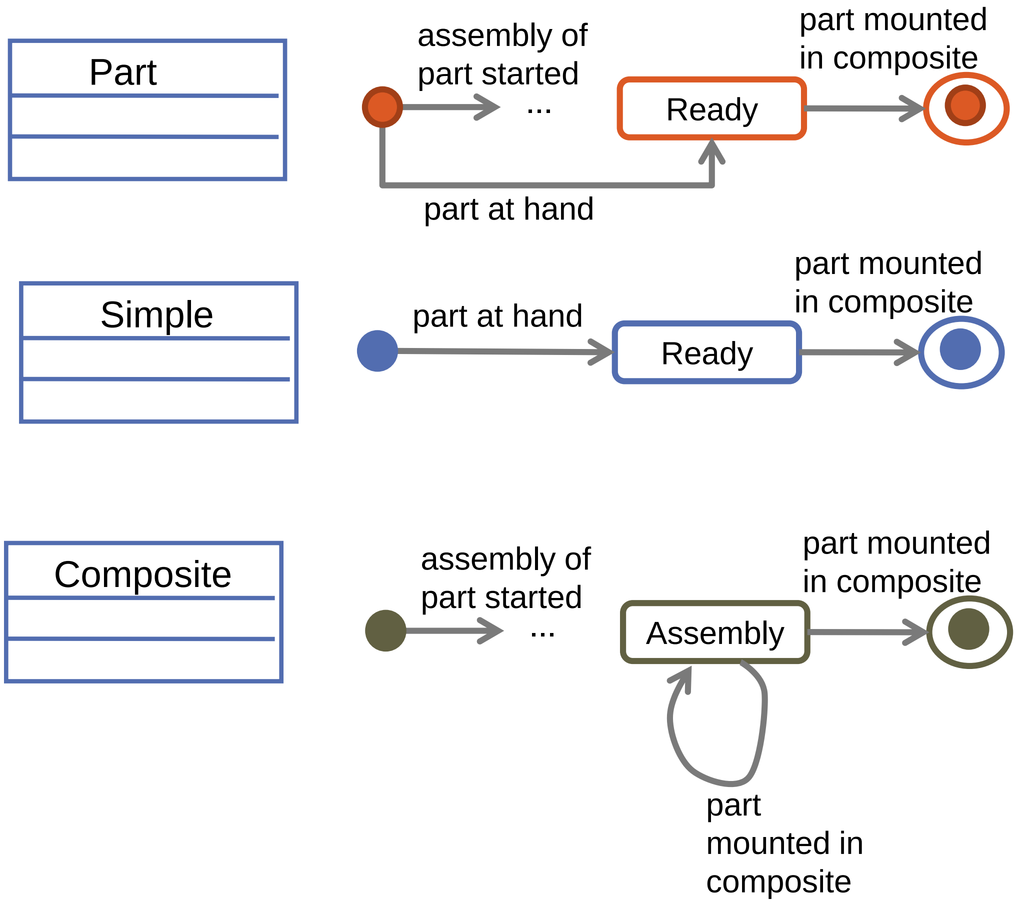


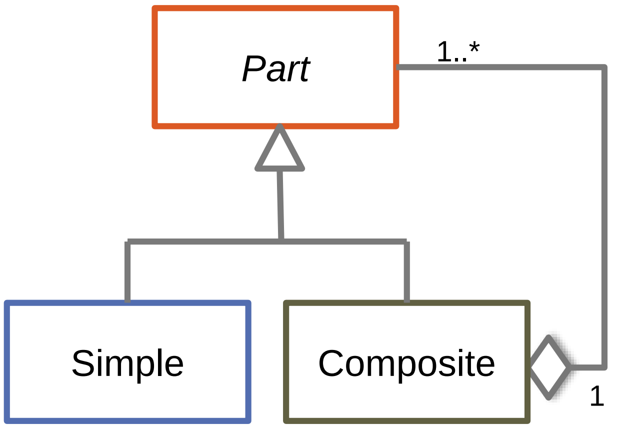




EXPLORE PATTERNS: COMPOSITE PATTERN

describe the creation or destruction of a hierarchy that is unknown at model-development time





EVALUATE CLASSES AND STRUCTURE

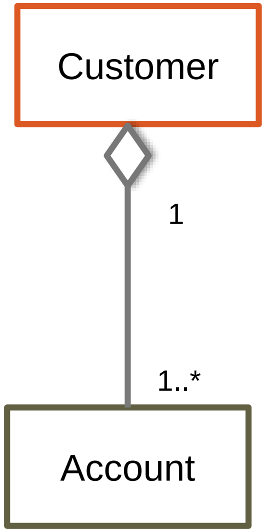
Generalization

- If the **same event is tied to two classes**, consider whether one class is a generalization of the other.
- If **two classes have many events with the same name**, consider whether they are different specializations of a third class.

Aggregation and association

- If two or more objects have **common events**, consider adding an aggregation or association structure between them.
- If two classes are related by an aggregation or association structure, at least one **common event** should be considered.

CONSIDER CLASSES

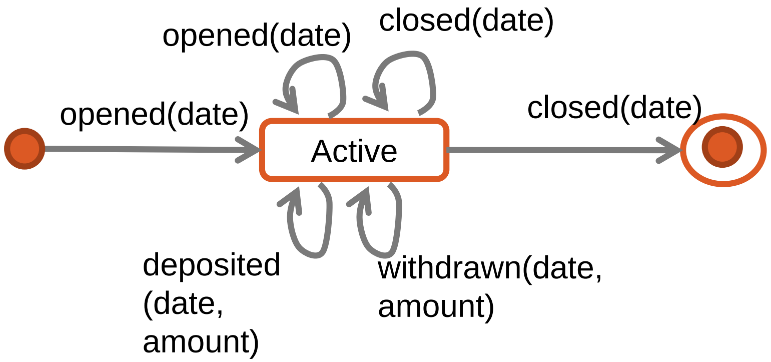


Customer

1

1..*

Account



opened(date)

closed(date)

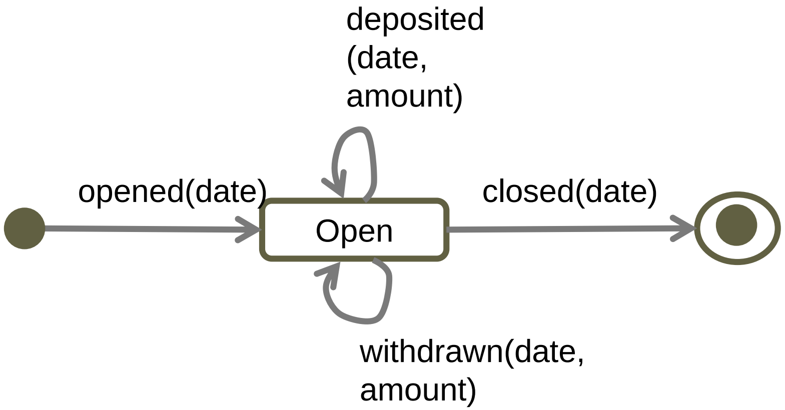
opened(date)

Active

closed(date)

deposited
(date,
amount)

withdrawn(date,
amount)



deposited
(date,
amount)

opened(date)

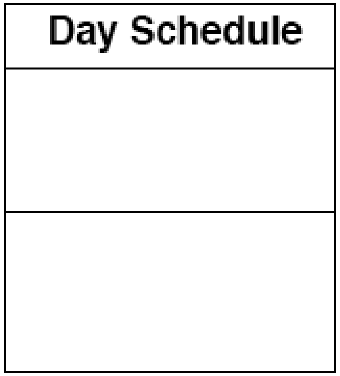
Open

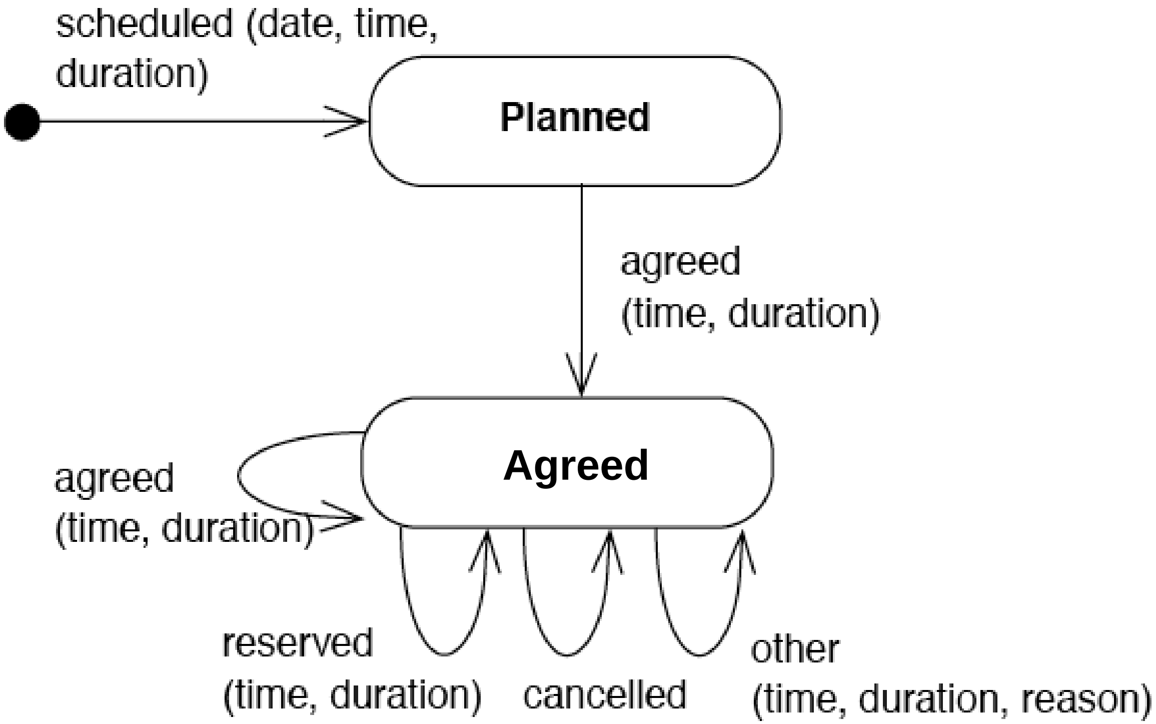
closed(date)

withdrawn(date,
amount)

PLAN

	Customer	Assistant	Apprentice	Appointment	Day schedule
reserved	X	X		X	X
cancelled	X	X		X	X
treated	X			X	
employed		X	X		
resigned		X	X		
graduated			X		
agreed		X	X		X




scheduled (date, time, duration)
Planned
agreed (time, duration)
Agreed
agreed (time, duration)
reserved (time, duration)
cancelled
other (time, duration, reason)

UPDATE THE EVENT TABLE

Event	Class				
	Customer	Assistant	Apprentice	Appointment	Day schedule
reserved	*	*		+	*
cancelled	*	*		+	*
treated	*			+	
employed		+	+		
resigned		+	+		
graduated			+		
agreed		*	*		*
scheduled		*	*		+
other		*	*		*
permit granted			+		

October 10, 2016

DESCRIBE ATTRIBUTES

- **Attribute**: a descriptive property of a class or an event

Three types:

1) Attributes connected to events (date, amount)
2) General information (name, address)
3) Derived from other attributes (account balance)

For classes

- What are the general characteristics of the class?
- How is the class described in the problem domain?
- What basic data must be captured about objects from this class?
- What results from an event trace must be captured?

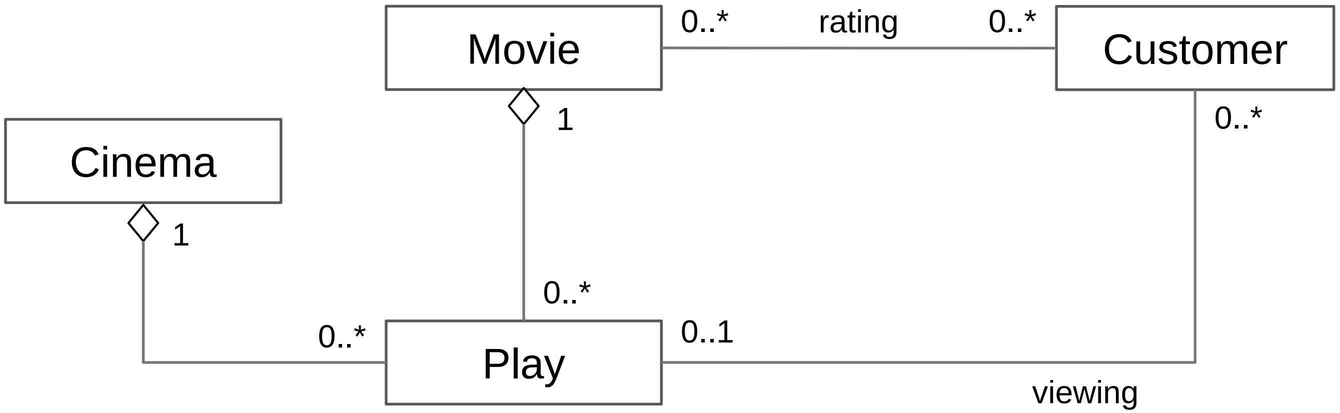
For events

- What time did the event occur?
- Which amount did it concern?

GROUP DISCUSSION

System for rating movie shows in a cinema. Registered customers can rate movies after viewing them in a theater.

	Movie	Customer	Play	Cinema
Movie opened	X			
Movie closed	X			
Movie played	X		X	x
Play planned	X		X	X
Customer closed		X		
Customer opened		X		
Movie viewed		X	X	
Movie rated	X	X		
Cinema opened				X
Cinema closed				X



October 10, 2016

Complete the activity ’behavior’ and show the results in state chart diagrams

OVERVIEW OF ‘BEHAVIOR’

Purpose

- To model the dynamics of a problem domain

Concepts

- Event trace: A sequence of events involving a specific object.
- Behavioral pattern: A description of possible event traces for all objects in a class
- Attribute: A descriptive property of a class or an event.

Principles

- Create behavioral patterns from event traces
- Study common events.
- Derive class attributes from behavioral patterns.

Result

- A behavioral pattern with attributes for every class in a class diagram.