FILS, English Stream, Year 3
Course: Software Development Methods
Prof. Luca Dan Serbanati

"E-Film Hiring" Project

FIFTH MACRO-ACTIVITY: MODELLING BEHAVIOUR WITH STATECHART DIAGRAMS

Deliveries: Statechart Diagrams (belonging to the Design Model)

An **event** is a significant or noteworthy occurrence.
A state is the condition of an object at a moment in time during the time between two significant events.
A transition is a relationship between two states that indicates that when an event occurs, the object moves from the prior state to the subsequent state. A statechart diagram shows the lifecycle of an object: what events it experiences, its transitions, and the states it is in between these events. A statechart diagram may be applied to classes (conceptual or software) and use cases. Since an entire "system" may be represented by a class, it may have its own statechart diagram, too. A statechart diagram that depicts the overall system events and their sequence within a use case is a kind of **use case statechart diagram.**
If, for all events of interest, an object always reacts the same way, it is a **state independent object.** By contrast, **state-dependent objects** react differently to events depending on their state. These are the objects we study with statechart diagrams.
In general, business information systems have a minority of interesting state dependent classes. By contrast, process control and telecommunication domains often have many state-dependent objects.
Here is a list of common objects which are usually state-dependent:

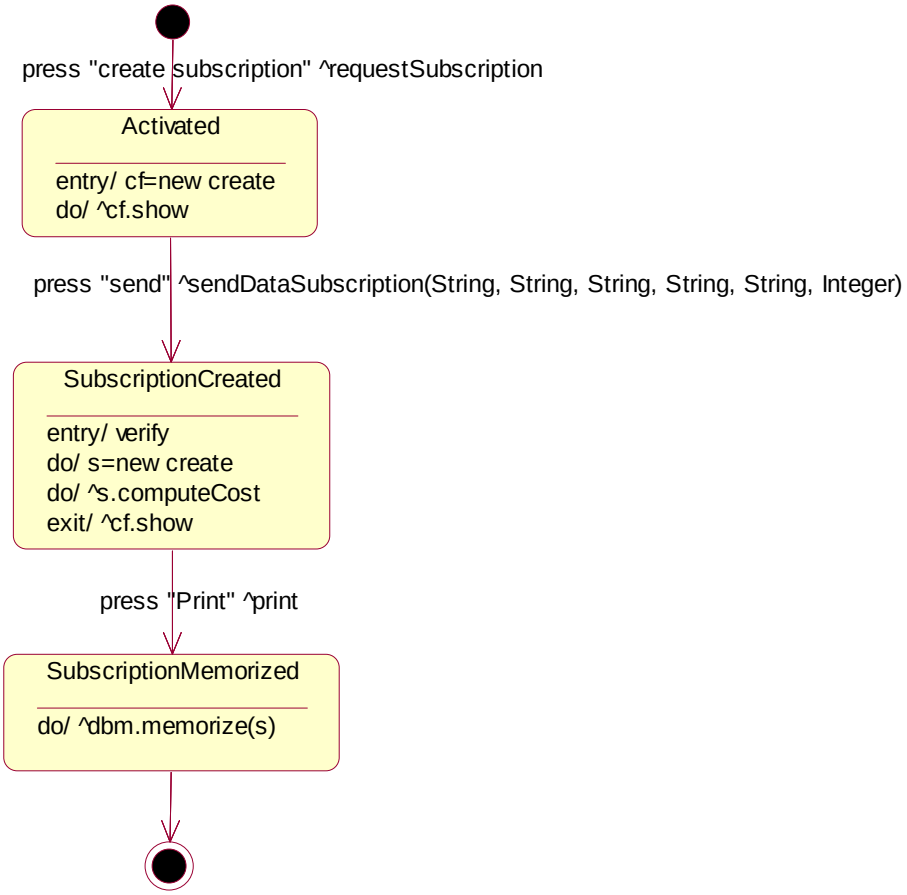
- **Use cases** (often is useful to view a use case implemented as a class)
- **Stateful sessions** (These are server-side software objects representing ongoing sessions or conversations with a client. The software class representing a use case may be also considered a stateful session object).
- **Systems**
- **Windows**
- **Controllers**
- **Transactions**
- **Devices**
- **Classes which change their role** in application during execution (a child Person becomes a young Person, then the later one becomes an adult Person, and so on).

When a statechart diagram model is also needed? When during design and implementation work, it is necessary to create and implement a design that ensures no out-of-sequence events occur. For example, a system which manages sales should not be allowed to receive a payment unless a sale is complete and code must be written to guarantee that. Given a set of use case statechart diagrams, a designer can methodically develop a design that ensures correct system event order.

Procedure

1. Create statecharts for state-dependent objects with complex behavior. Give care to objects with concurrent behaviour: they have orthogonal lifecycles.
2. If complex (i.e. many distinct states), organize hierarchically statechart diagrams.

Objects of ControllerSubscription class



2. Objects of the Hiring class