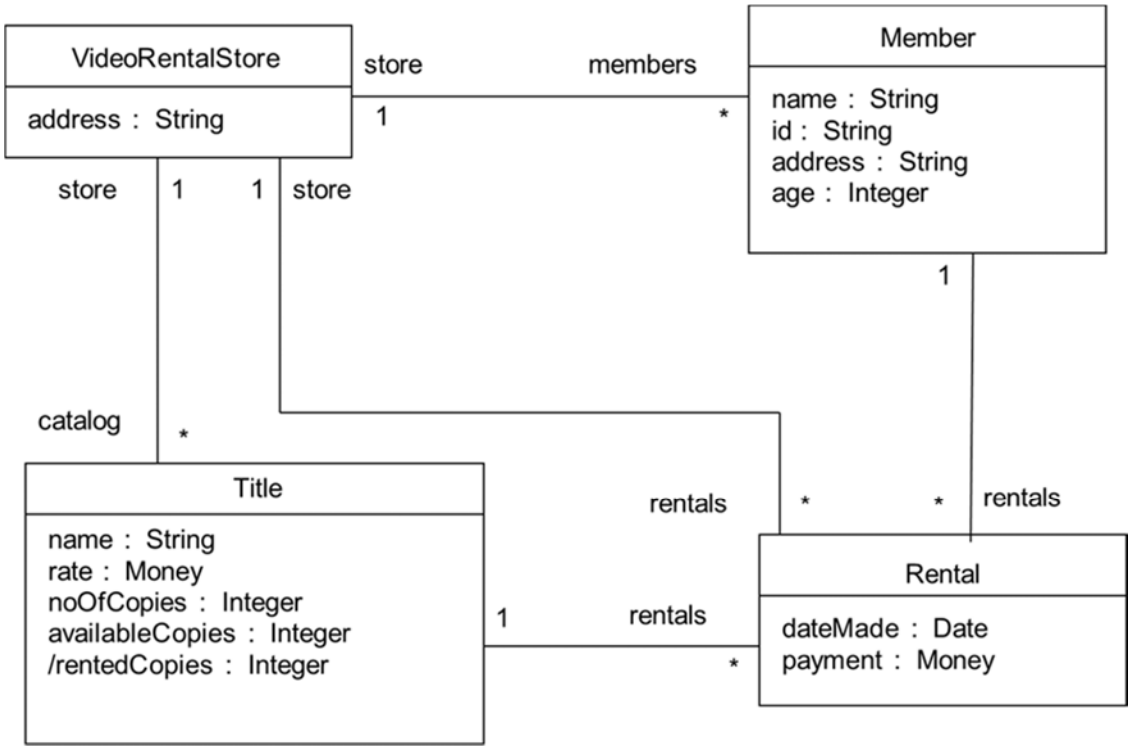
UML/OCL Exercises

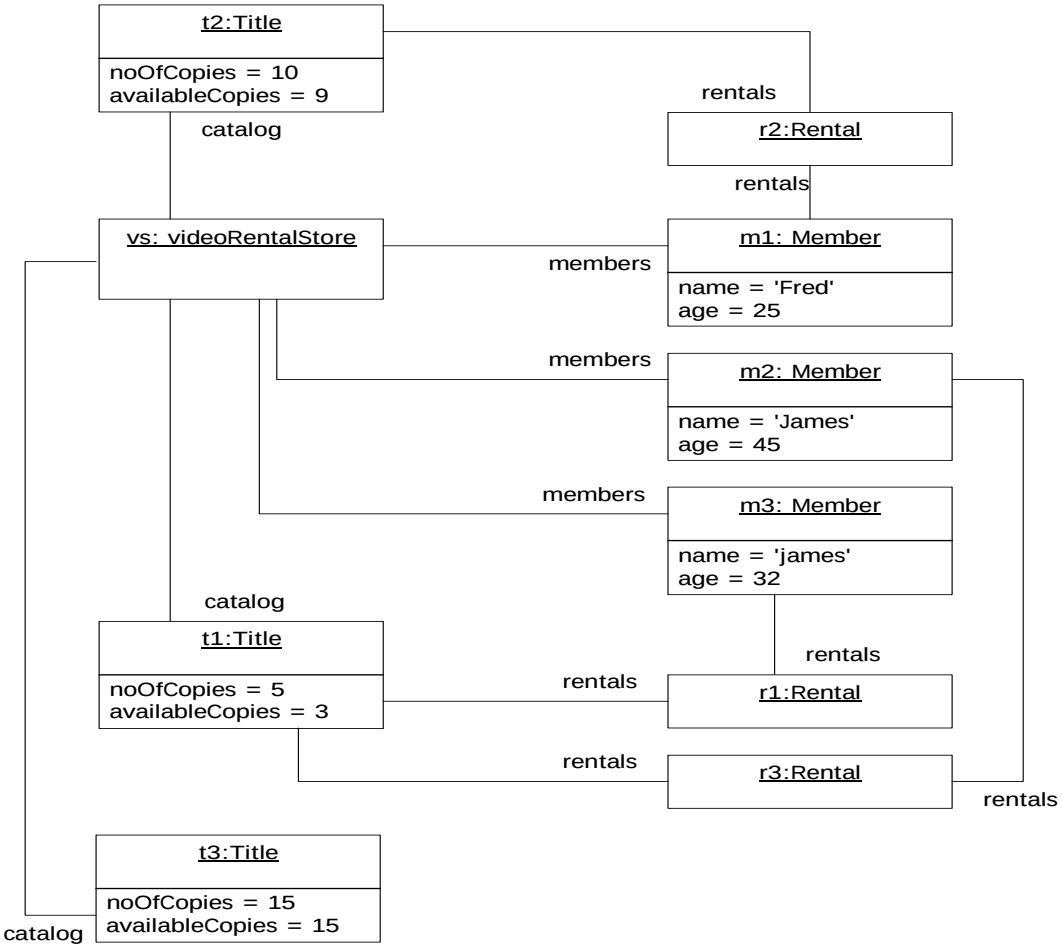
1. The following is a partial class model for the video rental store.



a) Extend the class diagram by adding a class `Copy` to represent copies of a title. The extension may involve adding suitable attributes and associations between classes. This should cover the requirement that each title can have zero or more copies and a copy is associated with exactly one title.

b) How would you extend the above class model to deal with reservations in the video rental store?

2. The following is an object diagram for the video rental store.



Evaluate the following expressions with respect to the above diagram

a) `vs.catalog`

b) `vs.members.name`

c) `vs.members -> size()`

d) `vs.members -> select(m : Member | m.age  >  30)`

e) `vs.members -> select(m:Member | m.rentals -> size()>= 1)`

f) `vs.catalog -> select(t : Title |  t.rentals -> isEmpty())`

g)
```
vs.catalog -> forAll(t:Title |
                        t.noOfCopies >= t.availableCopies)
```

h) `vs.catalog -> exists(t : Title | t.rentals -> isEmpty())`

i) `vs.catalog.noOfCopies->sum()`

j) `vs.rentals.Member.age -> size()`

3. Based on the class diagram for the video rental store given in the lecture, express the following informal queries as formal expressions in OCL.

   a) The members of a video rental store

   b) The members of a video rental store that has more than 10 rentals

   c) All titles with some rented copies of a video rental store

   d) All the members of a video rental store renting a particular title t

   e) All the titles with copies rented by a particular member m

   f) The total number of copies in the video store

   g) The total number of available copies in the video rental store

   h) The members of all the rentals of the video rental store

4. Formalize each of the following informal requirements of the video rental store as invariants in the OCL.

   a) The number of available copies for a given title is greater or equal to zero and less than or equal to the number of copies of that title.

   b) Each rental is uniquely associated with one title and one member.

   c) The name of each title is not the empty string

   d) The name of each title is unique.

   e) The number of rentals for a member cannot exceed 15.

   f) The total number of copies of a video store is less than 50.

5. The following exercises are based on the class diagram of the video rental store given in the lecture.

   a) The operation `enrollMember` creates a new member object with appropriate properties and adds it to the video rental store. Its signature is given as follows.

```
context  VideoRentalStore::
   enrolMember(name:String, address:String, age :Integer)
```

   Provide a formal specification for `enrollMember` using OCL.

b) The operation `getNoCopies` returns the number of video copies currently in the video rental store. Its signature is given as follows.

```
context VideoRentalStore::
                getNoCopies() : Integer
```

Provide a formal specification for `getNoCopies` using OCL.

c) The operation `addCopies` adds a number of copies for a given title. Its signature is given as follows.

```
context VideoRentalStore::
              addCopies(t:Title, n:Integer)
```

The parameter `n` is the number of copies to be added for title `t`. Provide a formal specification for `addCopies` using OCL.

d) The operation `deleteMember` removes a member from the video store. Its signature is given as follows.

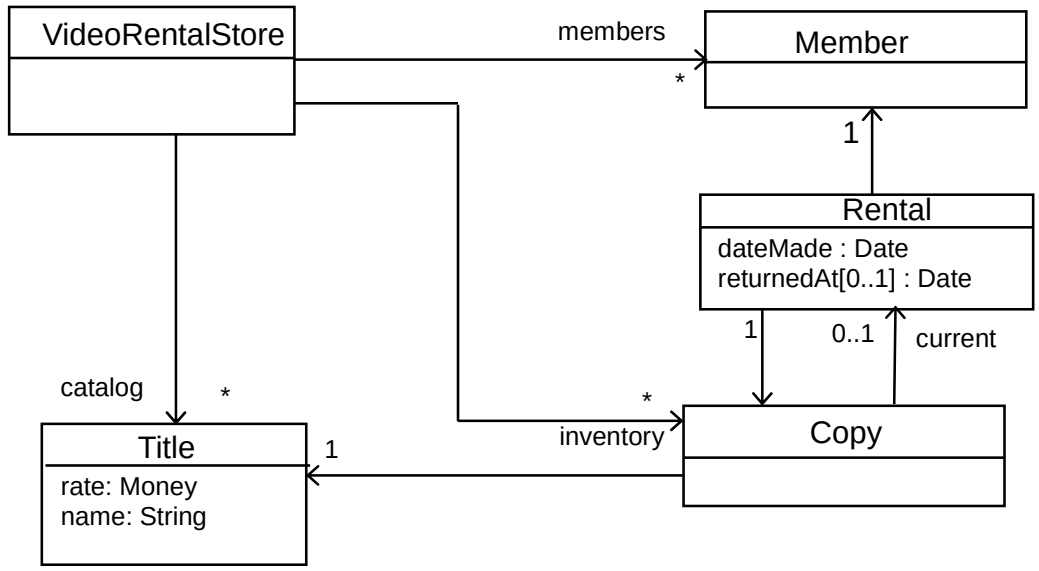
```
context VideoRentalStore::
                    deleteMember(m : Member)
```

Provide a formal specification for `deleteMember` using OCL.

6. USE (UML based Specification Environment) is a system for the specification of information systems. Among other offered functionalities, USE allows to validate UML and OCL models by constructing snapshots representing system states at a particular point in time with objects, attribute values, and links. Such snapshots are represented as object diagrams.

   Follow the following link http://sourceforge.net/apps/mediawiki/useocl/ and look at the section for validating pre- and postconditions using USE. The tool can also be freely downloaded for that website.

7. Here is a partial design class diagram for the video rental store.



A copy can be in two states `OnShelf` and `Rented` defined as:

```
context  Copy
inv: self.oclInState(OnShelf) = self.current->isEmpty
inv: self.oclInState(Rented) =  self.current->notEmpty
```

Consider the following specification of the operation `return`:

```
context VideoRentalStore::return(c : Copy, in : Date)
--returns copy c to the store shelf and set the returned at date of the rental
-- to in
 pre: self.inventory->includes(c) and
                                  c.oclInState(Rented)
post: c.current@pre.returnedAt = in   and
                                 c.oclInState(OnShelf)
```

Provide a design for the operation return that would satisfy the above specification and illustrate your solution using a UML sequence diagram.