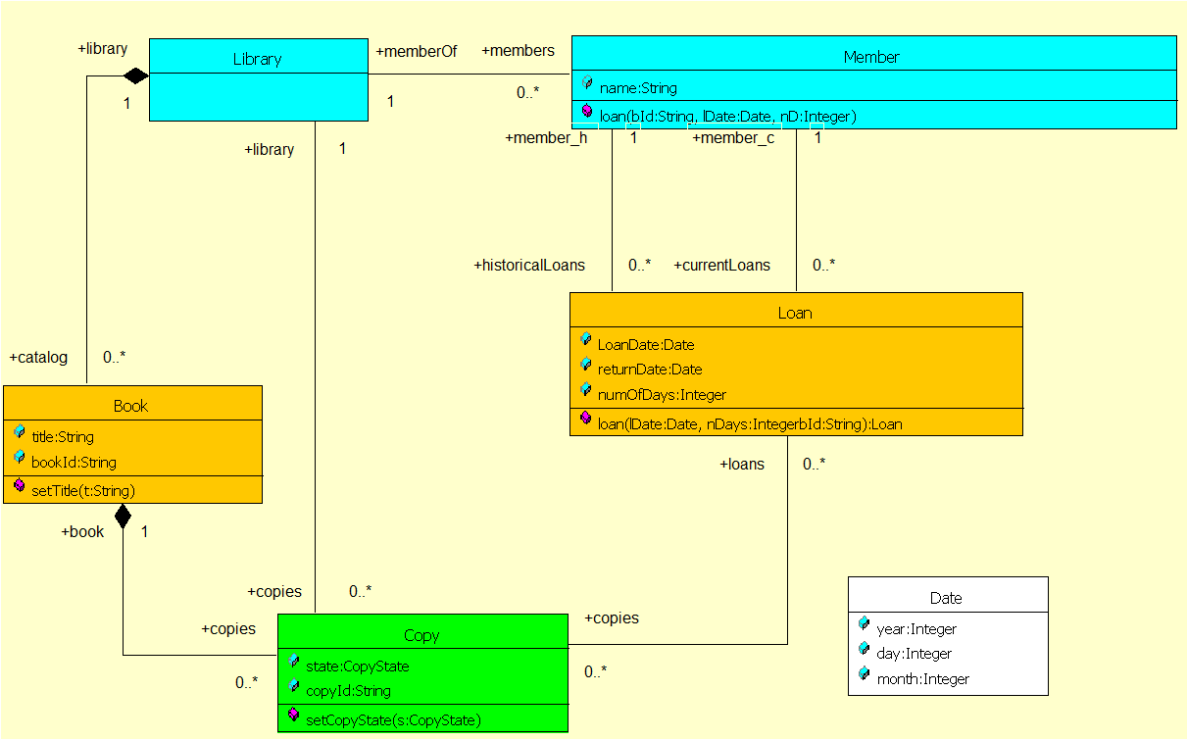
SE evaluation 2023_1

1. Using the UML, please specify be means of a class diagram, the structure of a model complying with the following requirements:

   A library has a catalog of books, each of which has a number of copies (we allow the fact that there might not be any copies of a book), a number of members and a way of keeping track of current and historical loans of those members. At any moment, a member may have at most 3 current loans, each corresponding to a copy. When a loaned copy is not returned in time, an instance of historical loan ponting towards that copy will be created and the corresponding current loan will be deleted. After returning a loaned copy, the state of that copy will be updated at available. Each book has a title and a bookId which is unique. A copy has a copyID, a state, a bReserve and an eReserve attributes. A state can be: available, loaned or reserved. When a currentLoan is created, the state of the associated copy will be modified at loaned. A loan is characterized by the attributes loanDate, returnDate and numberOfDays (representing the number of days for that loan). The value of returnDate will be updated at the currentDay of the returning day, irrespective if it's about a currentLoan or a historicalLoan. 2p



2. Using OCL, please specify:

   a) An invariant requiring that the books associated to the libray have a unique bookId. When the invariant is violted, the specification must support to infer the set of books having the same bookId. 1p

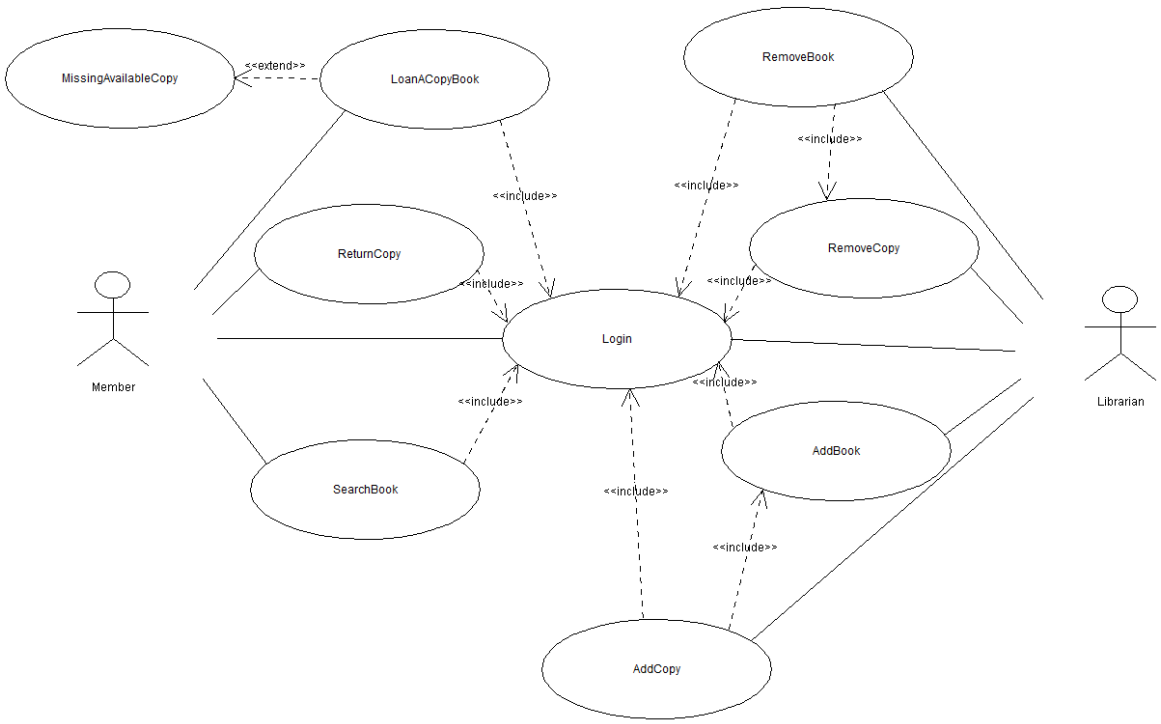
   b) A precondition attached to the Member::loan(bId:String, bDate:Date, nD:Integer), where bId is the bookId of the book associated with the copy we intent to loanm, bDate – the beginData of the loan and nD – the number of days of the loan. The precondition cmust check that there are not historicalLonas unreturned and that the number of currentLoans

are less than 3. Another constraint is that there is at least an available copy of the book we intent to loan. 1p

```
context Member::loan( bId:String, lDate:Date, nD:Integer)
    pre loanIsPossible:
        if self.historicalLoans->size > 0 then not self.historicalLoans->exists(l | l.returnDate.isUndefined) else true endif and
        self.currentLoans->size < 3 and
        self.memberOf.catalog->any(b | b.bookId = bId).copies->exists(c | c.state = CopyState::available)


context Book
    inv uniqueBookId:
      let bookId:Bag(String) = self.library.catalog.bookId in
        if self.library.catalog->reject(b:Book | b.bookId <> self.bookId)->size = 1
            then true
            else false
        endif
```

3. Using UML, please specify a UC diagram in which the actors are a libraryMember and a librarian. The functionalities accesed by the libraryMember are searchBook, loanACopyBook, returnCopy. The librarian must acces addBook, addCopy, removeBook and removeCopy. RemoveBook suposes remove automatically all associated copies. As concerning the loanACopyBook, this can be interrupted due to missing of at least an available copy. Of course, any of the previous functionalities supposes that the actor was previously loged. 1p



Using your own words, please describe shortly the Software Engineering concept. 1.5p

A set of principles, methods, methodologies, techniques and tools supporting the production of quality software:

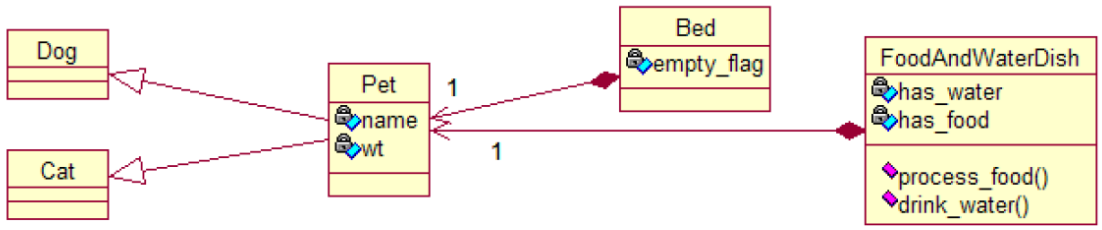
- within a given budget,
- before a given deadline,
- while changes occurs.

4. What it is and what is not software testing? Please explain which is the main mistake of managers when assign team members to software testing activities. 1.5p

Testing support discovering as much as possible bugs in the system in order to fix them. Testing is not to prove that a software has not bugs and runs only as previewed, simply because this is not possible. "Testing can only show the presence of bugs, not their absence" (Dijkstra).

Testing is often viewed as a job that can be done by beginners. Managers would assign the new members to the testing team, because the experienced people detested testing or are needed for the more important jobs of analysis and design. Unfortunately, such an attitude leads to many problems. To test a system effectively, a tester must have a detailed understanding of the whole system, ranging from the requirements to system design decisions and implementation issues. A tester must also be knowledgeable of testing techniques and apply these techniques effectively and efficiently to meet time, budget, and quality constraints.

5. The UML class diagram represented bellow contains some conceptual and some UML specification mistakes. Please identify, justify and mention a correct solution. 1p



The inheritance relationships are from Dog and Cat towards Pet. Pet is an abstract class. It is forbiden as a class (in our case Pet) to be part in more containers. Apart of these the usual relationships between Pet and Bed, Pet and FoodAndWaterDish are simple associations.