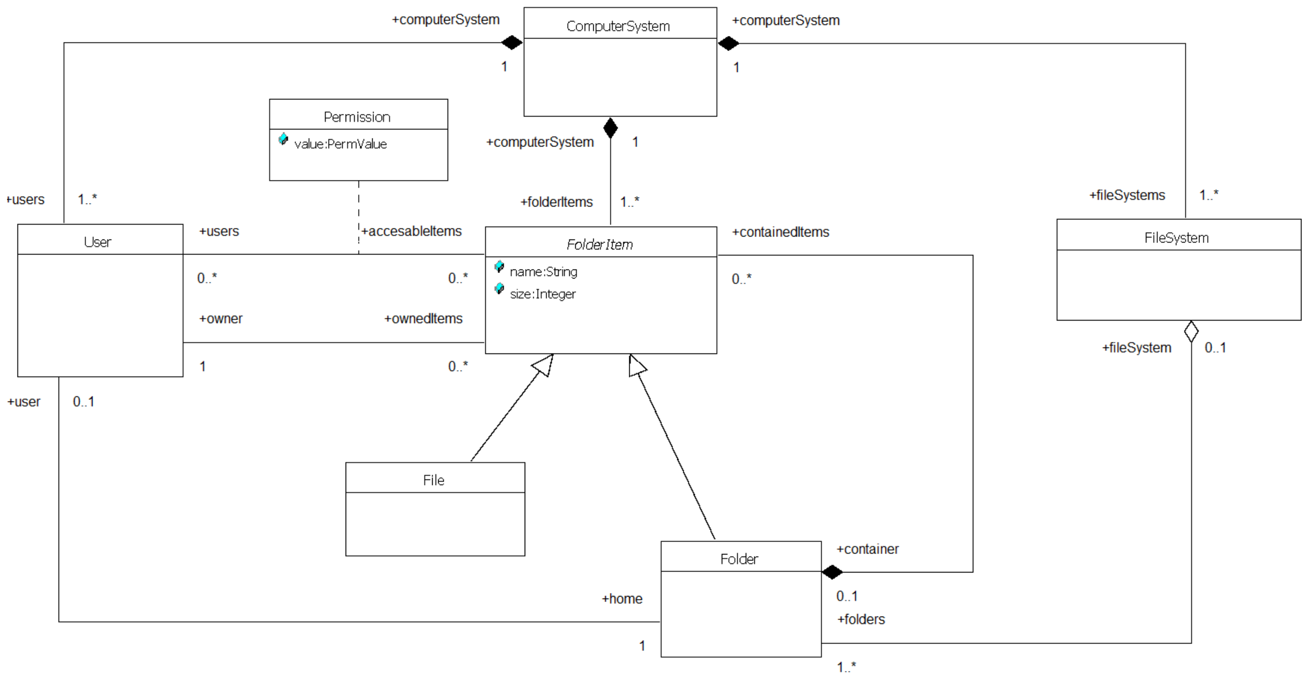
# SE Evaluation 8th of July 2020

I. A computer system contains file systems. A set of different users can access the computer system ressources generically named folderItems. Users may own or only access folderItems (files or folders). Each user has an acces permission for each forderItem. Each folderItem has a name which is unique in its namespace and a size. Let suppose that the permission to access a fileItem can be: write, read and so forth. Each folderItem knows its owner which is a user. Folders and files are structured in a hierarchical manner. Each folder may have attached a file system. Each file system may refer many folders.

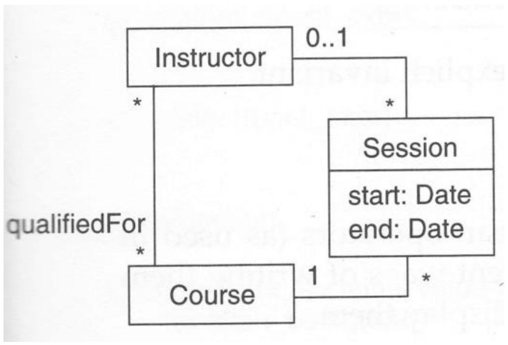
   a. Using the UML, please draw a class diagram representing the structure/architecture of a model complying with the above specification 2 pt
   b. Using the OCL, please specify an invariant requiring that all ownedItems by a user have the permission access write. 1pt
   c. Using the OCL, please specify an observer infering the Set of tuple type of users having the biggest sum of ownedItems size. 1pt



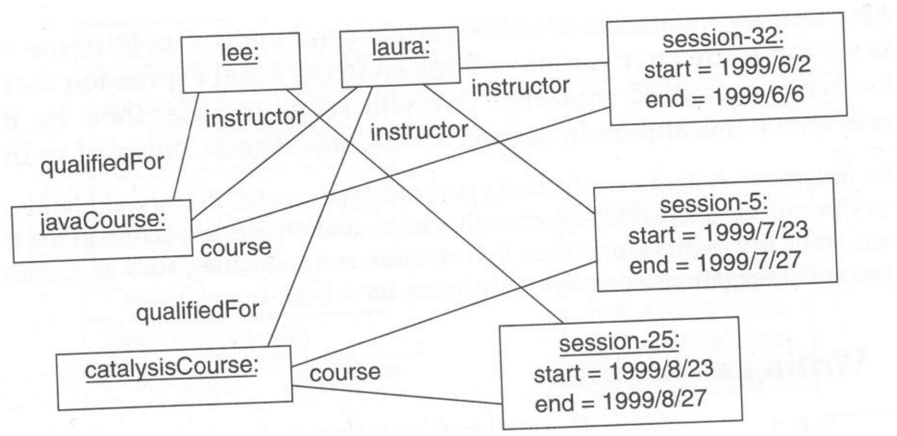
```
context User
    inv ownedItems:
        self.accesableItems->select(fI:FolderItem | fI.permission.value =
PermValue::write)->includesAll(self.ownedItems)
```

```
context ComputerSystem

def impUsers:

        let bigestSize:Integer = self.users->collect(us:User |
Tuple{u=us,ts=us.ownedItems.size->sum})->sortedBy(t | t.ts)->last.ts

        let impUsers:Set(TupleType(u:User,ts:Integer)) = self.users->

collect(us:User | Tuple{u=us,ts=us.ownedItems.size->sum})->select(t | t.ts =
bigestSize)->asSet
```

II. Let's be the following static view described by the class diagram and complemented by the invariant attached. Please argue if the snapshot bellow complies with the model or not. 1pt



```
context Instructor
   inv:
      self.session.course->includesAll(self.qualifiedFor)
```



The snapshot does not comply with the model because the lee: Instructor is not qualified for the catalysisCourse:, programmed in session-25:.

III. Please explain the main rationales of using diagrams in modeling. What's the main advantage of using diagrams to specify models? 1.5pt

When using graphical modeling language, diagrams are used first of all, to specify/construct models. Diagrams are well suitted to suport the communication between different category of developers: analysts, designers, programmers, testers, between different categories involved in software projects: clients, end users, domain experts, developers a.s.o. This is because because an image value more that 1000 words. Moreover, compared with textual specifications, diagrams support a better, easier and finner filtration of information.

IV. Which is the generic name of modeling languages using diagrams? But of modeling languages that can't use diagrams? What's the name of the syntax describing the categories of languages previously mentioned? But of the syntax describing the grammar of the modeling language? 1.5pt

Taking into account nature of the formalism used to specify models, modeling languages are included in one of the two categories/families of modeling languages are known as graphical modeling languages or textual modeling languages. In fact, most of graphical modeling languages also use a textual formalism. In case of the UML, the textual formalism which complements the graphical specification is OCL. So, graphical modeling languages try to combine the advantages of both categories of languages. The concrete syntax can be graphical or textual and describes the category of modeling languages. However the first category is named graphical modeling language due to the fact that mainly, the language is graphical and the textual part only complements the graphical languages. The grammar of modeling languages is specified by the abstract syntax, described by means of the metamodel and of constraints usually referred as Well Formedness Rules(WFRs).

V. Please explain the manner in which using modeling supports the increase of abstraction in specifying problems and solution to different problems. Why increasing the abstraction in specification is the most important vehicle of software development? 1pt

Abstraction mean the use of only the most important/representative information when describe a problem or a solution to a problem. Apart from programming languages, modeling languages split the above mentioned description in many parts, each concerning a separate view:structural/architectural, behavioral, deployment, implementation, testing. In this manner, the abstraction is increased by reducing the number of concepts. The price to pay is the need to check that between different views of the same concept or of related concepts, there are no contradictions. The capacity of people in managing/operating simultaneously with different concepts is restricted at 7+/- 2 conceps. That’s why reducing the number of concepts is crucial when working with complex problems.