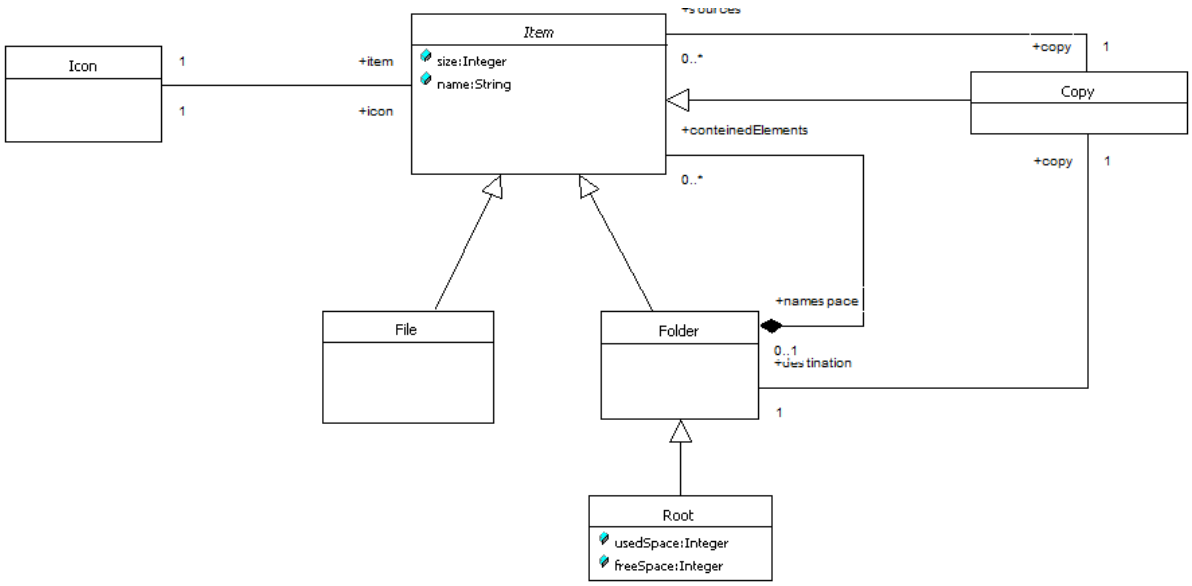
SE – the 5th of June 2019

I. In operating systems, users can copy items (files and/or folders) from a source namespace to a destination namespace. Each item has a non-empty name, a size and an icon associated. The first step in this operation is to select items to be copied. After, the user specifies the destination namespace. If the destination namespace is the source namespace, the name of each item copy must be different from the item name. Among namespaces there is a single namespace named root which is not included in another namespace. Information about used space and free space are stored at the root level.

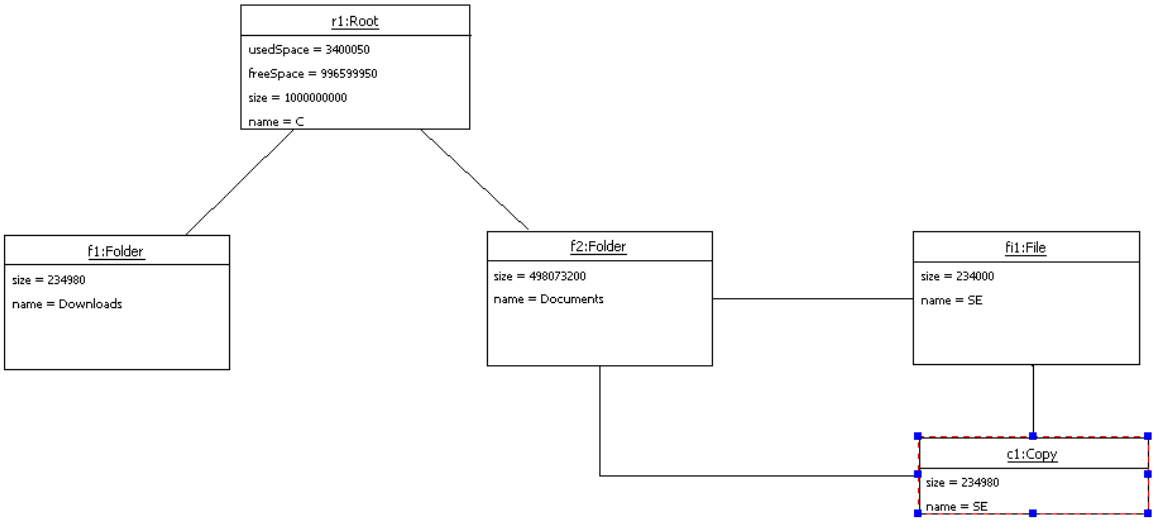
1. Please represent a UML class diagram describing the architecture of an object-oriented application that complies with the above-mentioned requirements. 2 pt
2. Using the OCL, please specify an invariant ensuring that if the source namespace and the destination namespace are the same, the name of the copy is different from the item name.

1 pt

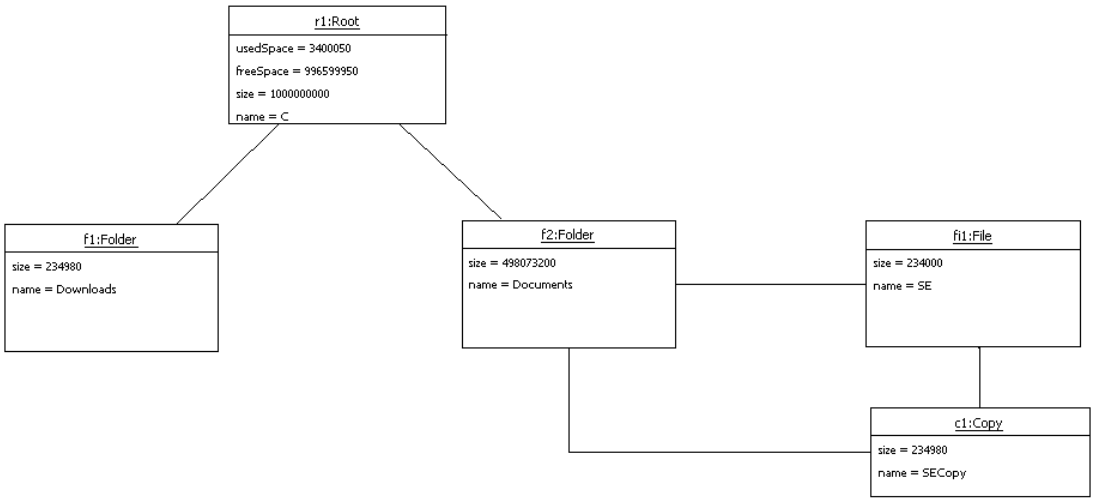
3. Using snapshots, please specify 2 cases: one in which the invariant is satisfied and another in which the invariant is violated. 1 pt




```
model CopyModel

context Copy
    inv sameSourceAndDestination:
        self.destination = self.sources->any(i:Item | true).namespace implies
        self.sources->select(i | i.name = i.copy.name)->isEmpty



endmodel
```



invariant violated



Invariant satisfied

II. Please explain the rationales of including the OCL (Object Constraint Language) in the UML standard. Please enumerate and describe shortly the most important features of the OCL. 1.5 pt

As the language name suggests, OCL was included in the UML in order to support constraints/assertions specification. This because even in case of simplest models, this cannot be specified in a rigorous manner in absence of constraints. However, OCL is needed in order to query/navigate the model. This is mandatory because, at the highest abstraction level, each model is represented as a graph. The graphical concrete syntax cannot express all the important properties and requirements of a system/model. OCL is a complementary declarative language for UML. OCL is not a stand-alone language. Excepting literal expressions, OCL specifications have no sense in absence of models. Declarative or side-free effects language means that after the evaluation of OCL expressions, the state of objects navigated remain unchanged. OCL is a typed language, supporting

first-order algebra. The language has a unique root, named OCLAny, and a common descendent OCLUndefined.

III. Please describe in which manner the UML supports abstraction. 0.5 pt

UML supports abstraction by means of **different views (functional, structural/architectural, behavioral, component, deployment)**, each of them associated with at least a language using less concepts and relationships as the UML language. In fact, the language is considered as being a set of modeling languages.

IV. Please enumerate and characterize shortly the testing activities. What do you mean by regression testing? In the context of testing explain the concepts of oracle, stub, driver and test cases. 2 pt

Testing activities that result in the plan, design, and execution of tests. **Testing activities include**:

- **Usability testing** tries to find faults in the user interface design of the system.
- **Unit testing** tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model.
- **Integration testing** is the activity of finding faults by testing individual components in combination. **Structural testing** is the culmination of integration testing involving all components of the system.
- **System testing** tests all the components together, seen as a single system to identify faults with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
    - **Functional testing** tests the requirements from the RAD and the user manual.
    - **Performance testing** checks the nonfunctional requirements and additional design goals from the SDD.
- **Acceptance testing** and **installation testing** check the system against the project agreement and is done by the client, if necessary, with help by the developers.

Regression testing includes the re-execution of all prior tests after a change. This ensures that functionality which worked before the correction has not been affected.

A **test case** is **a set of input data and expected results** that exercises a component with the purpose of causing failures and detecting faults. Expected results are named **oracle**.

A **test stub** simulates a component behavior that is called by the tested component. The test stub must provide the same API as the method of the simulated component and must return a value compliant with the return result type of the method's type signature.

A **test driver** simulates the part of the system that calls the component under test. A test driver passes the test inputs identified in the test case analysis to the component and displays the results.

V. What do you mean by direct engineering? Which are the advantages of applying this kind of model transformation in software development? Are there also drawbacks or traps in direct engineering? In an affirmative case, please explain them. 1 pt

Direct Engineering (DE) is a synonym used for Forward Engineering (FE), code generation (CG) or M2T Model to Text transformation. DE uses as input an implementation model (a late design model in which there can be off the shelf components (related to a technology)). The advantages of DE are: time-economy, the uniformity of implementation, the diminishing of situations in which errors may be introduce in code. The system architecture including the full association management, and dependency relationships, the code corresponding to constructors, getters and setters, the code corresponding to all kind of assertions and observers an be automatically generated in a percent of 100%.

The preconditions of code generation is to work with a compilable model (a model fully complying with WFRs), and a model tested as regarding the functionality. This is not a trap. Merely it's about a rigorous manner of working with models.

1 pt by default