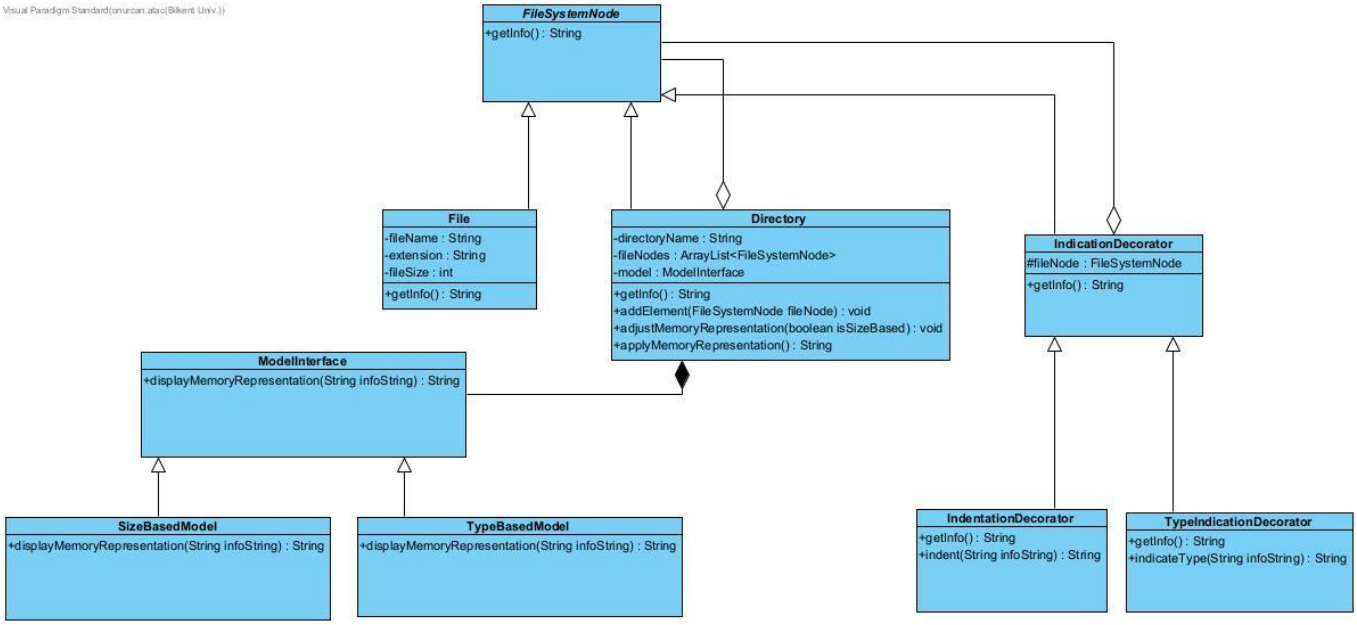
CS319 Design Patterns Homework Report

Onurcan Ataç

22002194

Section 2

Class Diagram and Explanations



For the first part, I used Composite design pattern. It is used on FileSystemNode abstract class, File and Directory classes. I realized that an instance of Directory can also include other instances of Directory, and that reminded me of the example in the design pattern tutorial, how CalculationGroup also became a CalculationComponent. By that way, group of objects can be treated as a single object using FileSystemNode. Because of the abstract class FileSystemNode, File and Directory also had the getInfo() method, which helped with using getInfo() method of File in Directory's getInfo(). Directory has an ArrayList of FileSystemNode's that is contained in that Directory, which can either be File or Directory objects.

For the second part, I used the Decorator design pattern. I decided on Decorator design pattern, because the main concern of the second part is modifying a string in specific ways, also both indentation and type indication can be used at the same time. That is similar to the use of this pattern in the design pattern tutorial in the second example. IndicationDecorator is the super class of IndentationDecorator and TypeIndicationDecorator. Those two classes override the getInfo() method of IndicationDecorator who gets that method from abstract class FileSystemNode. By this way, the formatting of the information string of a FileSystemNode can be changed during runtime, in four different ways depending on the usage of IndentationDecorator and TypeIndicationDecorator.

For the third part, I used the Strategy design pattern. It is used for selecting algorithms in runtime when different algorithms exist for a specific task. Since third part switches between two algorithms for one task, I thought Strategy pattern would be suitable. Directory has a ModelInterface object “model”, which is initialized by adjustMemoryRepresentation() method of Directory differently as SizeBasedModel or TypeBasedModel at runtime. By implementing the interface ModelInterface, SizeBasedModel and TypeBasedModel are contracted to have displayMemoryRepresentation() method.