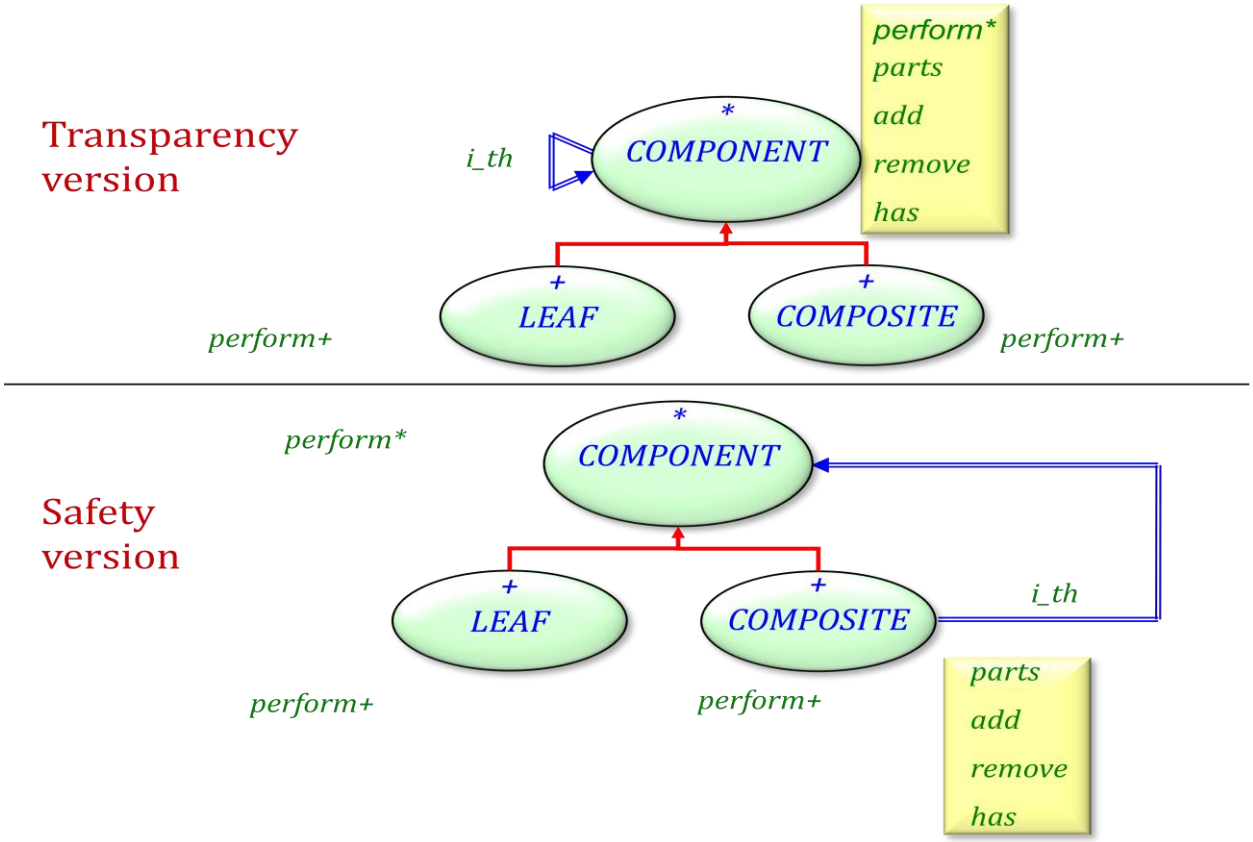
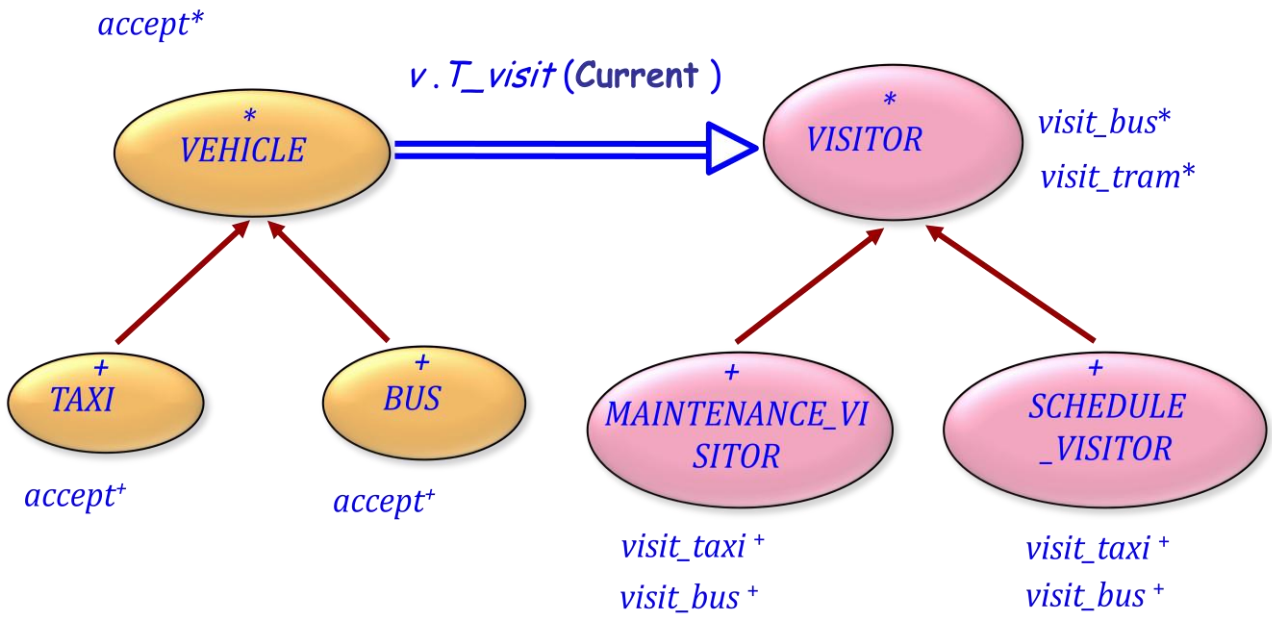
Software Architecture Exercise: Design Patterns

1. Write classes that represent a file system. It should at least contain two classes: *FOLDER* and *FILE*. Both classes define queries *name* and *size*. The size of objects of type *FILE* should store its size as an attribute, while the size of objects of type *FOLDER* calculate it as the sum of all recursively included file sizes. A folder may contain other folders or files. Your implementation should make sure the folder hierarchy is a tree structure (no cycles and every element is contained by at most one folder) and should use one of the variants of the **composite pattern**.



2. Add an attribute *is_text_file*: BOOLEAN to the FILE class. Use a **visitor pattern** to count the number of text files in your hierarchy. (You may need to adapt the FOLDER and FILE classes).



3. Support both NTFS and EXT files and folders. Use the **abstract factory pattern** to make clients' creation of files and folders independent of the file system type. Let NTFS_FACTORY create NTFS files and folders, and EXT_FACTORY create EXT files and folders. Every factory creates products (files and folders) of one family (NTFS or EXT), so mixing products of incompatible families will not happen frequently.