CSE443 OBJECT ORIENTED ANALYSIS AND DESIGN HOMEWORK

Class and Sequence Diagrams

Ozan Argıt ÖNÇEKEN

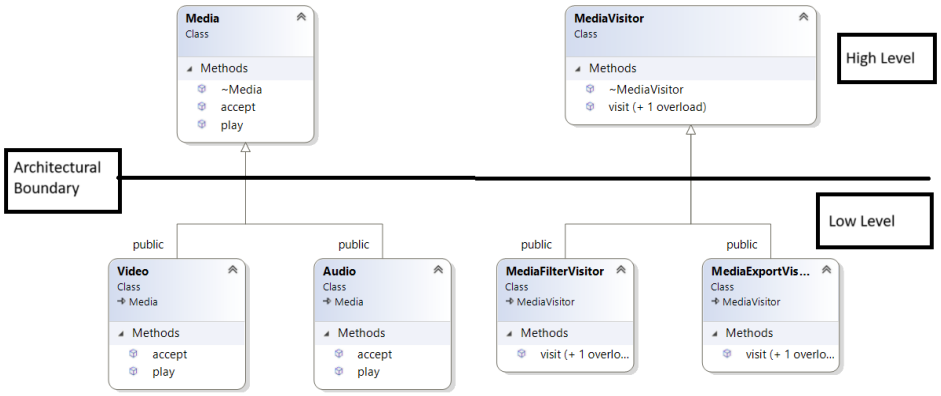
1901042259

Gebze Technical University, Department of Computer Engineering

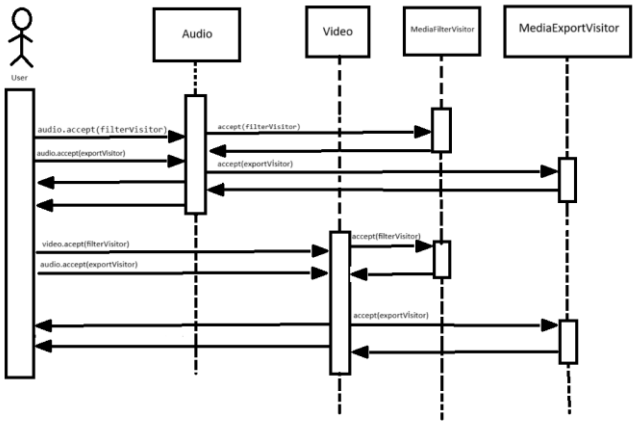
1.1 Introduction:

The existing media file class hierarchy struggles when new operations like `filter()` and `export()` need to be added due to a violation of the open-closed principle. To maintain flexibility and avoid modifying existing classes, the Visitor design pattern is employed. This pattern introduces separate visitor classes responsible for new operations, ensuring extensibility without altering the original class hierarchy. The design fosters a clean separation of concerns, making it easier to add future operations without impacting existing code.

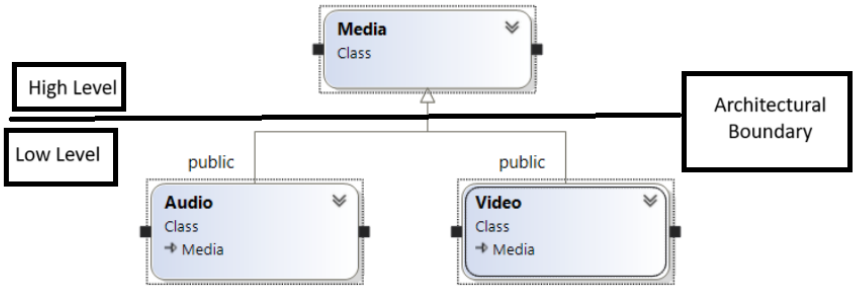
1.a.2 UML Diagram:



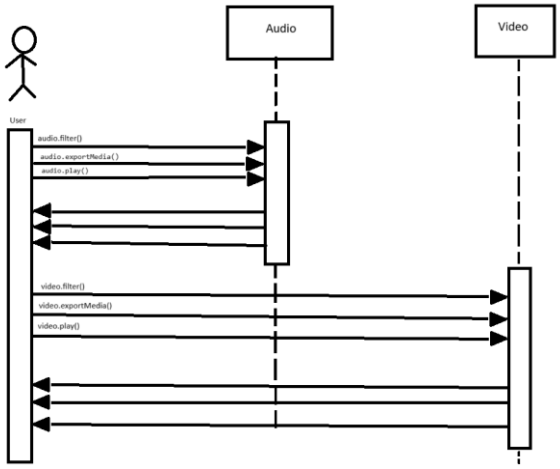
1.a.3 Sequence Diagram:



1.b.2 UML:



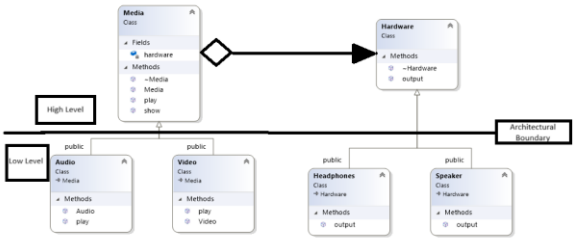
1.b.3 Sequence Diagram:



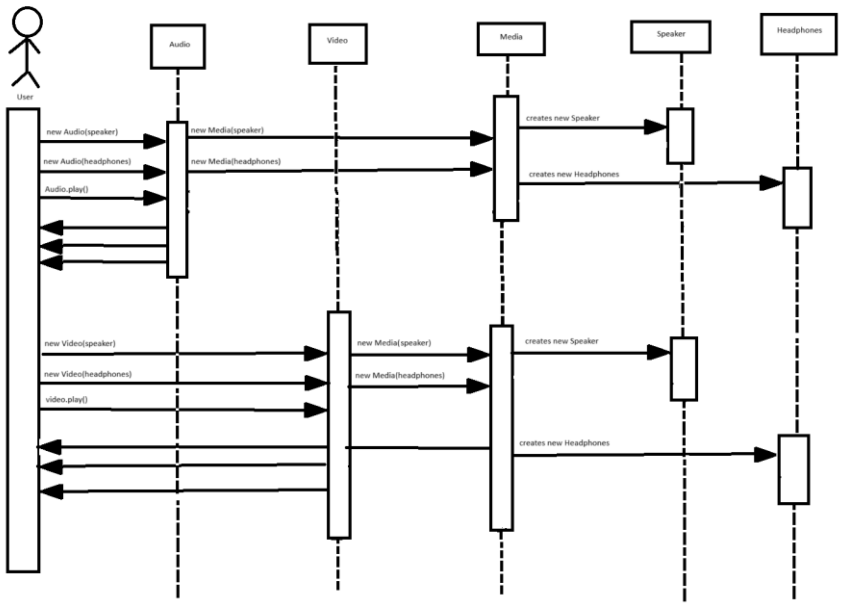
2.1 Introduction:

In the given Media class hierarchy, originally designed for polymorphic media playback, the challenge emerges when tasked with extending the functionality to accommodate new operations such as `filter()` and `export()`. Recognizing the potential for future additions and modifications, the solution lies in applying the Visitor design pattern. This pattern allows the seamless introduction of new operations without altering the existing class hierarchy, promoting code flexibility and adhering to the open-closed principle. By encapsulating the new functionalities within separate visitor classes, the design becomes more modular, facilitating easier maintenance and scalability for future enhancements.

2.2 UML Diagram:



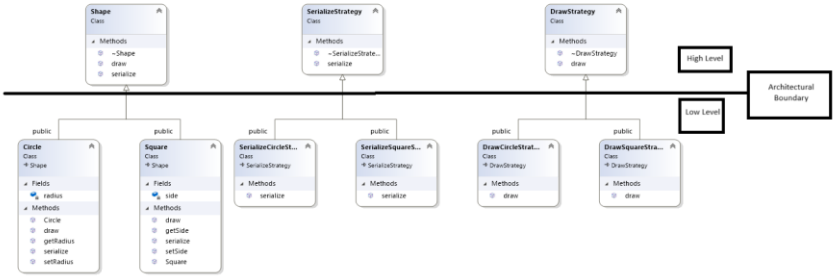
2.3 Sequence Diagram:



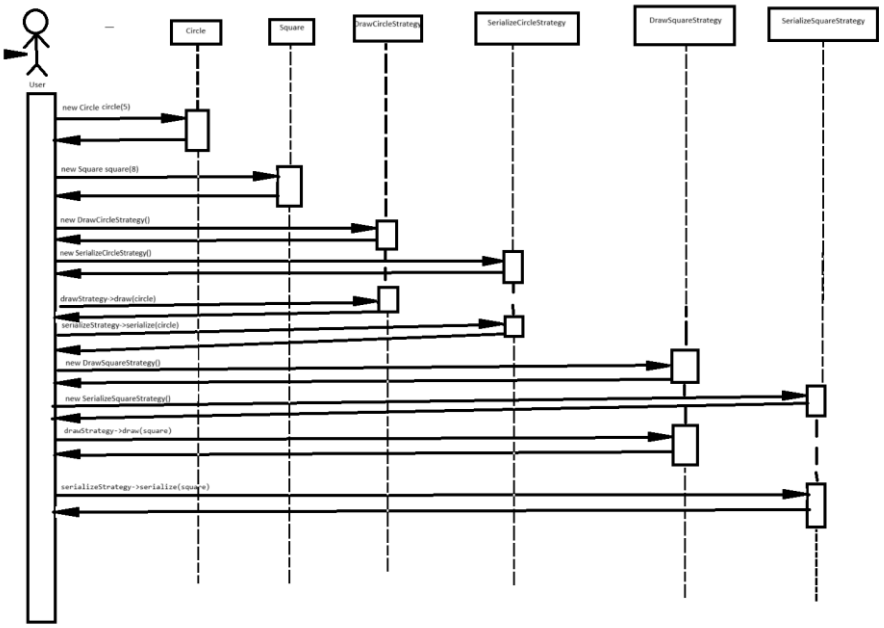
3.1 Introduction:

In addressing the need to implement draw and serialize operations on the existing class library containing "LibCircle" and "LibSquare," the solution involves applying the Strategy design pattern. A custom shape hierarchy is introduced, comprising an abstract base class and a class template derived from it. Each shape class within this hierarchy possesses separate strategies for draw and serialize operations. By encapsulating these strategies, the design achieves flexibility and extensibility, allowing the incorporation of new shapes and operations without modifying existing code. This strategic approach effectively leverages external polymorphism, enabling seamless integration with the provided "LibCircle" and "LibSquare" classes while adhering to the principles of the Strategy design pattern.

3.2 UML Diagram:



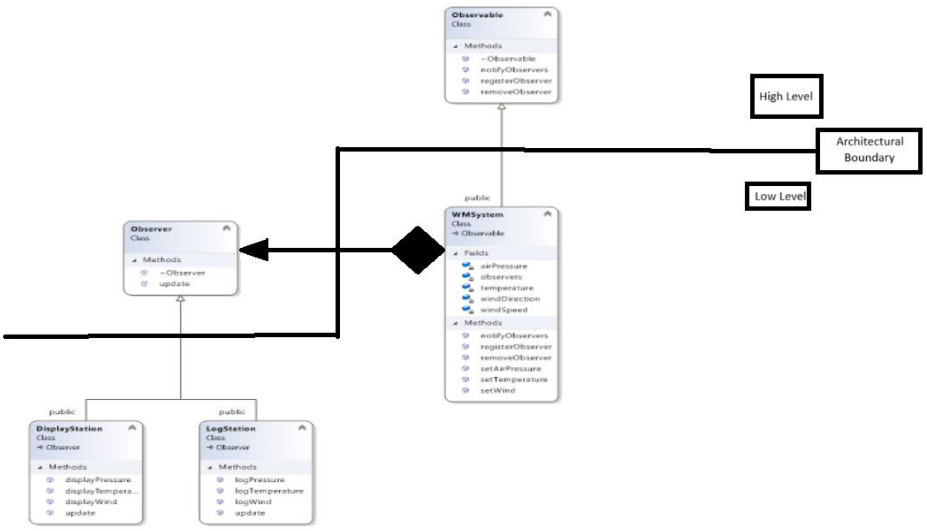
3.3 Sequence Diagram:



4.1 Introduction:

In augmenting the capabilities of the `WMSystem` class to accommodate DisplayStation and LogStation functionalities, with an eye towards future extensibility, the Observer design pattern has been strategically implemented. Transforming `WMSystem` into an Observable subject and introducing the Observer interface enables dynamic communication between the weather monitoring system and various observer stations. With this design, changes in sensor readings prompt notifications to all registered stations, such as DisplayStation for real-time display and LogStation for logging purposes. The Observer pattern not only enhances the modularity and flexibility of the system but also facilitates the seamless addition of new stations without necessitating modifications to the core `WMSystem` class, aligning with the principles of extensibility and maintainability.

4.2 UML Diagram:



4.3 Sequence Diagram: