**Project 2: Remote Control Revisit**

Chance Chime

California State University, Northridge

COMP 586: Object-Oriented Software Development

Professor Vahé Karamian

October 25, 2024

**Project 2: Remote Control Revisit**

This report will explain my design approach for revisiting the Remote Control project. Using the knowledge gained from Object-Oriented Design (OOD) and Object-Oriented Programming (OOP) and the design patterns introduced in the course, I have restructured the solution to improve its flexibility and maintainability. This redesign focuses on applying three key design patterns, one from each of the following categories:

* Creational: Builder Pattern
* Structural: Façade Pattern
* Behavioral: Command Pattern

### **1. Design Approach**

The redesign of the Remote Control project was a strategic move to enhance code modularity, reuse, and scalability. By incorporating three key design patterns, each chosen to address specific challenges from the original design, we not only solved the immediate issues but also brought long-term benefits to the project. This approach significantly improved the flexibility and maintainability of the solution, ensuring a robust and future-proof system.

* **Builder Pattern (Creational):** I applied the Builder Pattern to construct different types of Screen objects for various TV models. This pattern allowed for a flexible and step-by-step creation process, ensuring that new TV models or features could be added with minimal changes to the existing codebase.
* **Façade Pattern (Structural):** The Façade Pattern was implemented to streamline the interactions between the Remote class and the Screen. It simplifies the interface for controlling the TV while managing the complexity of different subsystems behind the scenes, such as Volume Control, Channel Management, and the Smart Menu.
* **Command Pattern (Behavioral):** The Command Pattern handled user actions like Power On/Off, Volume adjustments, and Channel changes. This pattern allowed me to encapsulate these actions as objects, enabling easy undo/redo functionality and more flexible command management.

### **2. Challenges and Lessons Learned**

During the redesign process, I encountered several challenges:

* **Balancing Flexibility and Complexity:** Incorporating multiple design patterns increased the complexity of the initial design. I had to carefully manage this complexity to avoid overengineering while still gaining the benefits of each pattern.
* **Avoiding Overuse of Patterns:** I had to balance applying design patterns effectively without overcomplicating the code or relying too heavily on creational or structural patterns at the expense of the Command Pattern.
* **Command Pattern Implementation:** One of the biggest challenges was ensuring that the Command Pattern was implemented in a way that could scale across multiple remote actions without redundant code. This led me to explore ways to generalize command handling, ensuring reusability and flexibility and instilling confidence in its implementation.

### 

### **3. Design Patterns Used**

* **Builder Pattern:** The Builder Pattern is crucial in the Remote Control project. It constructs the appropriate Screen instance based on the TV model selected by the user, encapsulating the creation logic. This not only ensures the correct Screen type for each TV model but also paves the way for easier future expansion, such as adding new TV models, thereby enhancing the project's adaptability and longevity.
* **Façade Pattern:** The RemoteControlFacade class is the central interface between the Remote and Screen, hiding the complexity of subsystems like Volume, Channel, and Smart Menu operations.
* **Command Pattern:** Each remote action (Power, Volume Up, Channel Down, etc.) is encapsulated in its Command object. These commands can be executed, undone, and managed independently, allowing for a more flexible control system.

### 

### 

### Image 1: UML Diagram representing Redesigned Remote Control

### **4. UML Diagrams** The UML diagram (*Image 1*) illustrates the revamped system's framework, emphasizing the relationships between the Remote, Screen, RemoteControlFacade, and Command objects. It showcases how implementing design patterns has effectively tackled various challenges, improving code modularity, reusability, and scalability, which has dramatically enhanced the overall architecture of the Remote Control project.