Conceptual design for an energy consumption meter simulator (workshop 3)

Students:

- Andres Jeronimo Ramirez
- David Santiago Ibañez
- Johan Danilo Trujillo

Course: Object-Oriented Programming — Semester 2025-II

Methodology and Deliverables:

1. SOLID Principles Analysis:

- **SRP - Single Responsibility Principle (S):** A class should have one, and only one, reason to change. This means that a class should have a single responsibility; this reduces complexity and makes the class easier to understand and modify, since changes to one function do not affect another.

  Application in our simulator:

  Before: The DeviceManager class handled device registration, consumption calculations, and high-usage detection.

  After: We separated responsibilities into:

  DeviceRegistry: Manages device registration and storage

  EnergyCalculator: Handles all energy-related calculations

  ConsumptionAnalyzer: Detects anomalies and high usage patterns

- **OCP - Open/Closed Principle (O):** Software entities should be open for extension but closed for modification.

Application in our simulator:

Created an abstract EnergySensor class that can be extended for different sensor types without modifying existing code

Notification system designed to accept new notification types without changing core logic

- **LSP - Liskov Substitution Principle (L):** Subclasses (derived classes) must be replaceable by their base classes without altering the correctness of the program; this ensures that inheritance hierarchies work correctly with polymorphism.

- **ISP - Interface Segregation Principle (I):** Clients (classes that use interfaces) should not be forced to depend on interfaces they do not use; prevent classes from implementing irrelevant methods, keeping interfaces small and cohesive.

- **DIP - Dependency Inversion Principle (D):** High-level modules should not depend on low-level modules. Both should depend on abstractions. Abstracx|tions should not depend on details; details should depend on abstractions.

2. Updated UML and CRC Cards:

• Revise your class diagrams and CRC cards to show how SOLID principles have influenced your design.

1.1

Class: Device	
Responsibilities: ● Store device data (name, power, usage time). ● Calculate individual energy consumption (kWh). ● Provide consumption history. ● Send data to the consumption manager.	**Collaborators:** ● Device manager ● Energy calculator ● UI

1.2. Class: Device manager

Class: Device manager	
Responsibilities: ● Register devices. ● Delete and edit connected devices. ● Maintain a list of all devices. ● Request individual usage calculations. ● Detects devices with high usage.	**Collaborators:** ● Device ● Energy calculator ● UI

1.3. Class: Energy calculator

Class: Energy calculator (Circuit)	
Responsibilities: ● Calculate consumption in kWh (energy = power × time). ● Convert kWh to Colombian pesos. ● Calculate total monthly consumption. ● Generate statistics (averages, peaks, trends).	**Collaborators:** ● Device ● Device manager

1.4. Class: Consumption history

Class: Consumption history	
Responsibilities: ● Save daily/weekly/monthly usage. ● Allow historical analysis.	**Collaborators:** ● Device ● Device manager

• Export data for reporting. • Notify of usage changes	• UI

1.5. Class: Concept info

Class: Concept info	
Responsibilities: • Store educational definitions (voltage, current, power, resistance). • Show real-life examples. • Provide optional diagrams or images. • Act as a study guide.	**Collaborators:** • UI • Settings

1.6. Class: User settings

Class: User settings	
Responsibilities: • Save user preferences. • Configure notifications (consumption limits). • Change currency and units (optional). • Control language and interface.	**Collaborators:** • UI

1.7. Class: Notification manager

Class: Notification manager

Responsibilities: ● Send alerts when a device consumes too much. ● Notify of monthly increases. ● Show savings tips.	**Collaborators:** ● Device manager ● Consumption history ● UI

1.8. Class: UI (user interface)

Class: Notification manager	
Responsibilities: ● Display monthly usage. ● Display monthly cost in pesos. ● Render list of devices. ● Pressing a device → opens details. ● Display educational information (Concept Info).	**Collaborators:** ● Device manager ● Energy calculator ● Concept info ● Notification manager

• Highlight new interfaces, abstract classes, or refactored responsibilities.

3. Python Code Snippets:

• Provide short Python code examples (class definitions, method overrides, interface usage, etc.) that demonstrate your application of SOLID.

• Clearly comment each snippet to explain which principle is being illustrated.

4. Reflection:

• Write a brief reflection (half a page) on the process of applying SOLID principles as a second-semester engineering student.

• Discuss any difficulties, trade-offs, or insights gained.