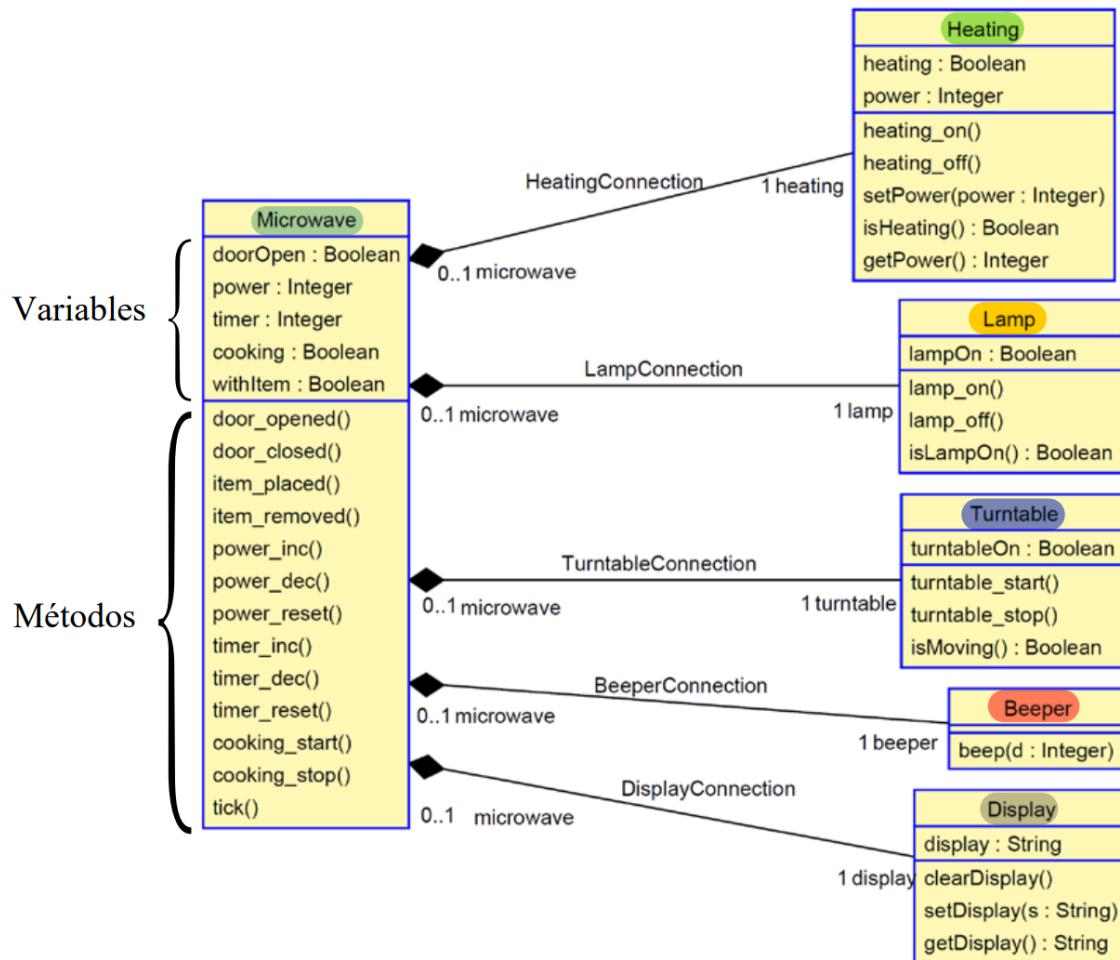


Ingeniería del Software Avanzada

Práctica Final: Horno Microondas



Autor: David Ramírez Arco

Fecha: 10 de junio de 2022

GitHub: https://github.com/Archerd6/Proyecto_Microondas

Índice

Apartados.....	1
Apartado A.....	1
Apartado B.....	1
Apartado C.....	2
Resultados.....	3
Conclusión.....	3

Apartados

Apartado A

Consideremos los siguientes patrones de diseño: Adaptador, Decorador y Representante. Identifique las principales semejanzas y diferencias entre cada dos ellos (no es suficiente con definir, sino describir explícitamente similitudes y semejanzas concretas).

El patrón Adaptador consiste en la implementación de una clase que permita reutilizar clases adaptándolas al funcionamiento otras nuevas.

El patrón Decorador permite extender comportamientos de clases de forma dinámica, para convertir una clase previamente desarrollada en una con un comportamiento diferente, sin la necesidad de crear una clase nueva o subclase de esta.

El patrón Representante proporciona un nivel adicional para proporcionar acceso controlado o inteligente (a través de un proxy).

Por tanto, la principal semejanza es que los tres tienen un propósito estructural, aunque cada uno soluciona un problema distinto, en posibles situaciones distintas.

- Una similitud entre el patrón adaptador y el decorador modifican objetos ya existentes
(El patrón adaptador adapta una interfaz para poder reutilizar una clase existente y el decorador añade funciones a objetos mediante composición)
- Y una diferencia por ejemplo, el patrón representante no busca transformar directamente el comportamiento de ninguna clase.

Apartado B

Consideremos los patrones de diseño de comportamiento Estrategia y Estado. Identifique las principales semejanzas y diferencias entre ellos.

La principal similitud es que los dos modifican el comportamiento de los objetos: Ambos buscan aportar diversas formas de funcionamiento a una clase.

La diferencia entre ellos es que en el patrón estado otorga diferentes funcionalidades dadas las circunstancias en las que se encuentre una clase, mientras que en el patrón estrategia desarrolla nuevas formas de trabajar mediante la elección de las circunstancias.

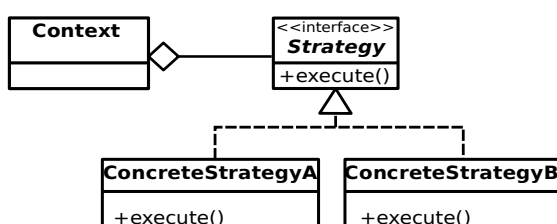


Figura 1: Patrón Estrategia

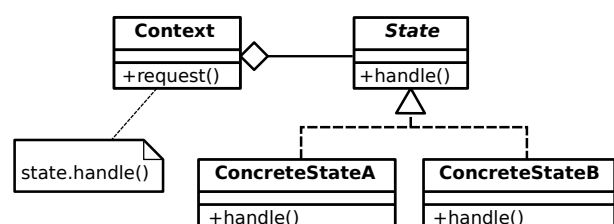


Figura 2: Patrón Estado

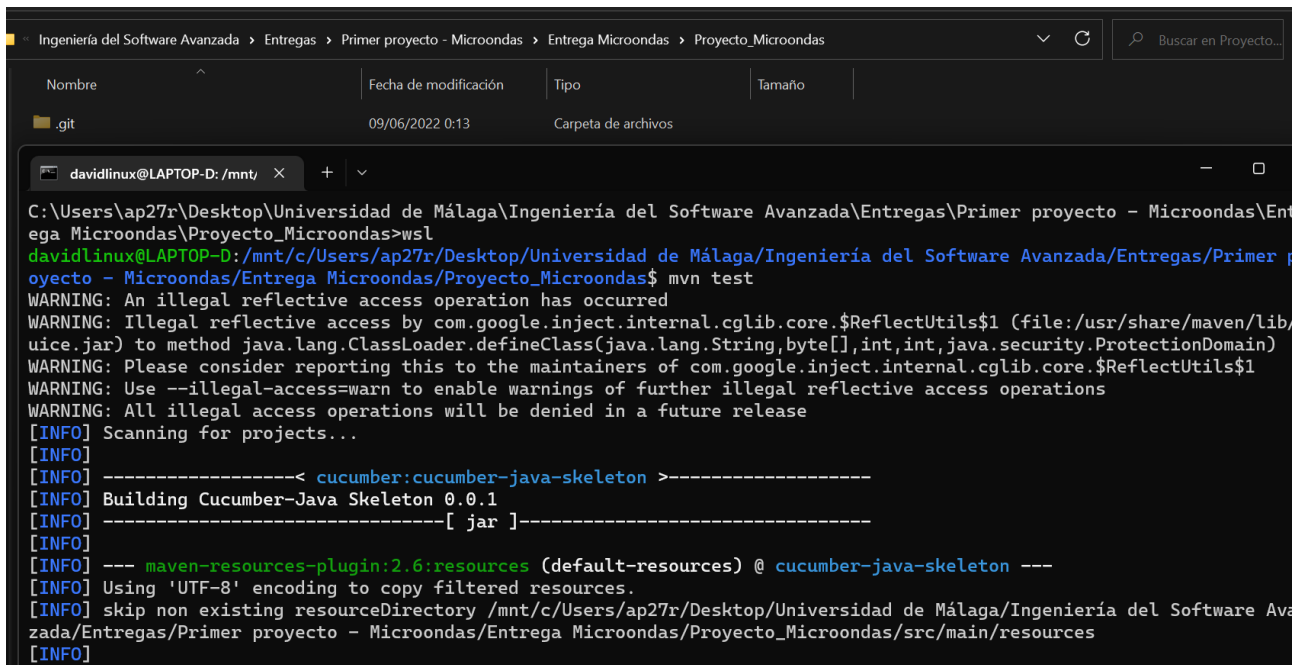
Apartado C

Consideremos los patrones de diseño de comportamiento Mediator y Observador. Identifique las principales semejanzas y diferencias entre ellos.

La principal similitud entre los patrones es que son de comportamiento, al igual que los anteriores modifican las relaciones entre los objetos. Ambos son bastante similares ya que pretenden conectar objetos emisores y receptores de información.

El patrón mediador crea un objeto que define como los objeto interactuaran entre sí. El patrón observador no crea un objeto, si no que define objetos como Observador u Observables

Resultados



```
Ingeniería del Software Avanzada > Entregas > Primer proyecto - Microondas > Entrega Microondas > Proyecto_Microondas
Nombre      Fecha de modificación  Tipo      Tamaño
.git        09/06/2022 0:13      Carpeta de archivos

davidlinux@LAPTOP-D: /mnt/
C:\Users\ap27r\Desktop\Universidad de Málaga\Ingeniería del Software Avanzada\Entregas\Primer proyecto - Microondas\Entrega Microondas\Proyecto_Microondas>wsl
davidlinux@LAPTOP-D:/mnt/c/Users/ap27r/Desktop/Universidad de Málaga/Ingeniería del Software Avanzada/Entregas/Primer proyecto - Microondas/Entrega Microondas/Proyecto_Microondas$ mvn test
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$ReflectUtils$1 (file:/usr/share/maven/lib/uice.jar) to method java.lang.ClassLoader.defineClass(java.lang.String,byte[],int,int,java.security.ProtectionDomain)
WARNING: Please consider reporting this to the maintainers of com.google.inject.internal.cglib.core.$ReflectUtils$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO] -----< cucumber:cucumber-java-skeleton >-----
[INFO] Building Cucumber-Java Skeleton 0.0.1
[INFO] -----[ jar ]-----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ cucumber-java-skeleton ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /mnt/c/Users/ap27r/Desktop/Universidad de Málaga/Ingeniería del Software Avanzada/Entregas/Primer proyecto - Microondas/Entrega Microondas/Proyecto_Microondas/src/main/resources
[INFO]
```

Conclusión

El patrón de diseño Estado se adapta muy bien para este proyecto (ya que el comportamiento del microondas cambia dependiendo de su estado)