Decisiones 27/04/2018

1. En Cliente, en el tipo de documento se decide implementar un enum. Una alternativa al enum es usar un string para diferenciar el tipo de documento. Se utiliza un enum al darnos simplicidad al código ya que sólo se le asigna un número a cada tipo de documento. Nos permite agregar un nuevo tipo de documento en un futuro. Tiene la desventaja de ser poco flexible, si en algún día se le quiere dar comportamiento al tipo de documento, se tiene que cambiar de técnica.
2. En Dispositivo, para saber si un dispositivo está encendido se decide usar un booleano. Una alternativa al booleano es un Strategy. No se usa la alternativa porque no tiene comportamiento los estados encendido y apagado, no se cree que los vaya a tener un comportamiento los estados. El boolean tiene la cualidad de ser simple, lo cual nos da la desventaja de tener un solución menos extensible; si se quiere agregar otro estado en un futuro, se tendrá que modificar el código. No se considera que algún día el dispositivo tenga otro estado del que ya tiene, por lo tanto se elije un booleano.
3. Para modelar las categorías se decide usar un repositorio. En un momento se pensó usar un Strategy, pero se descarta esa alternativa porque no hay categorias con comportamiento distinto, sino con instancias distintas. El repositorio tiene la característica de ser flexible al poder dejarte agregar una categoría con poco esfuerzo.

Decisiones 18/05/2018

1. Para modelar el estado de un dispositivo utilizamos el patrón State. Su motivo es que cada estado del dispositivo tiene comportamiento propio. Consideramos que no es responsabilidad del objeto cambiarse. Además, agrega simplicidad el hecho de que el estado cambie entre Apagado, encendido y Ahorro de energía. Otra opción es el patrón Strategy, no se utiliza por que este comportamiento es quien tiene la responsabilidad de cambiar los estados. Nos da más extensibilidad que el Strategy dado que da facilidad para agregar funcionalidad a estos estados.

2. Hemos corregido, los repositorios con los consejos de la entrega anterior, un Repositorio general padre, que hereda a los repositorios que utilizamos para levantar los Json de tipo Singleton.

3. La forma elegida de implementar los actuadores, sensores, y reglas parte de la idea de un observer. La clase regla tiene una lista de actuadores esta pendiente de que ocurra un evento, y para que ocurra todos los sensores de la lista de regla detecten que tienen la condición que tienen como atributo, y se utiliza ocurrir evento ocurrirEvento para verificarlo mapeando y si da true, se le informa a los actuadores que envíen el mensaje a los dispositivos para realizar la acción.