

DISEÑO DE SISTEMAS

# Trabajo Práctico Anual

## “Sistema de Gestión Energética”

**Grupo:** 2

**Integrantes:**

- Alejo Scotti - alejoscott@gmail.com - 1528142
- Juan Pablo Ferreira - juanpabloferreira88@gmail.com - 1275902
- Nicolas Hovassapian - Nicohova.95@gmail.com - 1530318
- Ivan Metta
- Sebastián Cairola
- Tomas Villa

**Fecha de entrega:** 22/05/2018

**Profesor:** Martín Agüero

**Ayudante a cargo:** Alejandro Ezequiel Leoz - Nicolas Contreras

**Repositorio:** <https://github.com/tomivilla/DDS-Grupo2>

**Branch:** master

**Commit ID:** ecf3c44cf64d5ed8790916072df6794ea8dc6c3a

## Registro de cambios

Fecha	Modificaciones
12/05/2018	Patron State para el estado de los dispositivos inteligentes
12/05/2018	Patron decorador para adaptadores
13/05/2018	Patron Bridge para los Actuadores
16-05-2018	Diagrama de Clases preliminar con los patrones seleccionados.
11-06-2018	Diagrama de Clases - Se cambió el patro Decorador por un Adapter para los adatadores de dispositivos estandar

## Tabla de Requerimientos no funcionales TP - Entrega 1

Fueron identificados los siguientes requerimientos no funcionales. Se tuvieron en cuenta aquellos que afectaran a la arquitectura del sistema.

Permitir la instalación de sensores.

## Diagrama de Clases TP - Entrega 1



Diagrama de clases - Entrega 1.zip

### Tabla de decisiones de diseño:

Fecha	Decisión	Ventaja	Desventaja	Alternativa
12/05/2018	La idea sería usar un <b>Estate</b> para los <b>Estados</b> de los <b>Dispositivos Inteligentes</b> .	Permitir que un objeto altere su comportamiento cuando su estado interno cambia. Permite modelar las transiciones entre estados.		
12/05/2018	Para los <b>Adaptadores</b> de los <b>Dispositivos Estadar</b> usaríamos un <b>Decorador</b> .	Agregar dinámicamente responsabilidades (funcionalidad) extra a un objeto. Es una forma flexible que sirve de alternativa a subclassing para extender funcionalidad. Mas flexibilidad que la herencia estática.	Un decorador y su componente no son identicos.	Estrategias

13/05/2018	Para el <b>Actuador</b> se usaría el patrón <b>Bridge</b> via la interface <b>Implementador</b> para no depender de implementación que tiene cada electrodomestico según su fabricante como indica el enunciado.	Desacoplar una abstracción de su implementación, de modo que ambas puedan variar de forma independiente.		Composite
13/05/2018	Se agregó el patrón <b>Observer</b> , para avisar cada vez que un <b>Sensor</b> realiza una medición de la magnitud que corresponda	Definir dependencias one-to-many entre objetos, de forma tal que cuando un objeto cambia su estado todos los objetos dependientes son notificados y actualizados inmediatamente		
16/05/2018	Una <b>Regla</b> tiene un listado de Condiciones y Acciones, además conoce a un dispositivo, sobre el cual comprobaría el cumplimiento de las Condiciones, en caso de cumplirse todas las condiciones, ejecutaría las acciones. A su vez el cliente definiría las reglas a aplicar sobre sus dispositivos.			
11-06-2018	Se cambió el patrón Decorador por el patrón Adapter para agregar el Adaptador de dispositivos Estandar.	Permite que dos clases incompatibles puedan funcionar en conjunto.		Decorador