Informe Práctica Diseño:

Alejandro Peral Taboada

José Benjamín González Doval

Entre los principios de diseño hemos usado :

Principio de Responsabilidad Única, hemos empleado este principio en la mayor parte de las clases, por ejemplo la clase registro. Otras como por ejemplo la clase piscina, si que realizan múltiples acciones, sin embargo nos hemos encargado de que la piscina se encargue de ser una interfaz que conecte clases delegando muchas de sus funciones en las clases Registro y Estado.

Principio de sustitución Liskov: Lo podemos apreciar en las clases de los empleados que utilizan el método avisar que se encuentra en la clase empleado y todas las distintas subclases se encargan de realizar ese trabajo de manera transparente para la plantilla.

Principio de la inversión de la dependencia. Vemos que la piscina trabaja con interfaces y clases abstractas, siendo estas las que la construyen.

El principio de segregación de interfaces está presente en toda la práctica, un ejemplo pueden ser los distintos sucesos o los empleados.

Para la realización de esta práctica hemos optado por varios patrones.

Para permitir que una piscina tuviera múltiples estados y que añadir uno nuevo no resultara demasiado costoso, hemos utilizado el patrón Estado. La piscina juega el rol del contexto, el interfaz EstadosPiscina juega el rol de Estado y las clases Activa, Mantenimiento, Evacuación y Cerrada. Son cada uno de los estados concretos.

Para elaborar el registro hemos utilizado el patrón composición de tal forma que cuando cambiemos un parámetro lancemos un nuevo suceso de ese tipo y que en caso que añadamos piscinas con parámetros a mayores, solo tendremos que añadir otro tipo de suceso por cada parámetro. El interfaz suceso es el contexto, el registro tiene el rol de composición y cada uno de los diferentes sucesos (SucesoNivelPH, SucesoPersonas, SucesoTemperatura…) son los componentes concretos.

Para implementar los sensores hemos utilizado el patrón estrategia, puesto que una piscina tendrá unos sensores específicos hemos hecho que todos los sensores tengan una y solo una piscina. No poseen métodos públicos puesto que los sensores dependerán de los valores que modifiquen, para emular esos cambios hemos añadido un método medir. Así tenemos que todos los sensores implementarán sensor y cada uno se encargará de su parte. La piscina es el contexto, el interfaz sensor es la estrategia, y cada uno de los distintos tipos de sensores (SensorCloro, SensorSales …) son las estrategias concretas.

Para todo el personal hemos optado por dos tipos de patrones, en primer lugar para implementar los distintos tipos de empleados hemos optado por estrategia para poder añadir nuevos tipos de empleados de manera sencilla. Donde la plantilla es el contexto clase abstracta empleado es la estrategia y los Encargados Gestores Socorristas Marketing y Mantenimiento son las estrategias concretas. En segundo lugar hemos hecho que los equipos utilicen el patrón composición para que posean a su vez empleados y otros equipos, por ello el interfaz es la plantilla, empleado es el componente concreto y equipo es el que tiene el rol de composición.

Finalmente para las alarmas hemos también mezclado 2 patrones, hemos utilizado el observador para que la piscina no sepa quien la está observando y que a su vez no le interese. Donde la piscina es el sujeto concreto, su superclase observable es el sujeto, el interfaz observer es el observador, y la clase alarma es el observador concreto. Para luego implementar múltiples alarmas hemos hecho que la clase alarma sea abstracta y que juegue el papel de estrategia, haciendo que las alarmas la implementen y para de este modo poder vincularlas a la piscina de manera sencilla. Cada una de las alarmas que implementemos sean las alarmas concretas y que la clase observer sea el contexto donde actual las alarmas.