**INTRODUCCIÓN**

En esta memoria vamos a explicar como hemos realizado mediante el lenguaje de programación de C++ el proyecto de una casa domótica. Para ello nos hemos ayudado del programa DIA para la realización del diagrama UML, del programa QT Creator para la realización del código del programa y también hemos usado un repositorio en Github para poder programar de manera más sencilla el código entre los dos miembros del equipo de forma paralela.

La casa domótica necesitará un sistema de comunicaciones que conectará con los distintos nodos, para así poder establecer que valores se leen de los sensores y cuales son los que deben cambiar y enviarle cuales serán los cambios.

Una vez establecido este sistema, procederemos a crear un PC Central que controlará toda la casa, este será el que tendrá los nodos y dentro de los nodos, los actuadores, los sensores y el controlador.

También hemos tenido en cuenta que tanto los sensores como los actuadores, dependiendo de la función que tengan, podrán ser discretos o analógicos, esto será añadido a ambas clases con el objetivo de que el programa pueda tener todas las opciones disponibles.

**DIAGRAMA UML**

Para la realización del diagrama de UML hemos utilizado la aplicación ya mencionada DIA, basándonos en la introducción nos hemos dispuesto a realizarlo.

Para comenzar, hemos valorado que el programa tiene que girar en torno a una clase de Comunicaciones, esta debe de tener las identificaciones ID pertinente para después poder establecer los lazos con otra clase que estará dentro de PC Central.

La Clase Comunicaciones deberá tener un atributo al que nosotros hemos llamado Mensaje y será de tipo String, este mensaje será el que contenga la ID del sensor, la ID del actuador, la acción que debe realizar y el valor que tiene que tener.

Para ello hemos creado un método dentro de esta clase llamado createMessage (ID1, ID2, acción y valor) será de tipo string. Este método se encargará de crear el mensaje que después mandaremos a Nodo, que estará agregado a PC Central.

Dentro de esta clase haremos una agregación para que la comunicación se pueda hacer a través de ficheros que será una nueva clase, o a través de Ethernet, esta última será un trabajo futuro, mientras que la clase de Ficheros tendrá un atributo de tipo string al que hemos denominado fichero. También contará con 2 métodos que completarán lo que se agrega de su clase Padre, sendMessage(mensaje: string), este método se encargará de mandar el atributo mensaje y receiveMessage(mensaje: String) que se encargará de leerlo.

Una vez realizada la columna vertebral del programa que es la Clase Comunicaciones, nos disponemos a realizar el UML de la parte Principal del programa, a esta le llamaremos PC Central que tendrá como atributo un vector nodo[] del tipo nodo que será una clase agregada a esta posteriormente.

PC Central tendrá un método que será básico para el programa, que será la de añadir un nuevo nodo, cada vez que queramos agregar algún elemento a nuestra casa. También tendrá como método el estado de los actuadores que será el que lee cuando el valor que le tiene que dar al actuador. Mientras que el estado valoresSensores será de tipo float y leerá el valor que tiene el sensor.

Agregado a PC Central nos encontraremos con la Clase Nodo, esta tendrá como atributo a comunicación que es del tipo comunicaciones, y nos servirá para enlazar el Pc Central con comunicaciones. También tendrá el atributo ID que le dará a los objetos de las clases herederas su identificación para cada sensor y para cada actuador.

Dentro de Nodo encontraremos los Métodos añadirsensor y añadiractuador, los dos harán la misma función pero cada uno refiriéndose a una clase distinta heredera de Nodo.

Esta clase será la clase Padre de las clases controlador, actuador, y sensor, de la que heredarán sus métodos y sus atributos. De las dos últimas también habrá una nueva herencia para distinguir si los actuadores son discretos o analógicos.

La clase controlador tendrá un atributo al que hemos denominado ref que será el setPoint del regulador que establezcamos de tipo float,.También tendrá como atributos las IDs de los sensores y de los actuadores. Estos tres atributos se unirán en un método al que hemos denominado controlActuador que se encargara de las funciones de un regulador de tipo P.

Dentro de la clase Actuador hemos hecho herencia de esta clase dando lugar a las clases Discreto y Analógico, la primera con dos métodos que serán abrir y cerrar, mientras el que analógico sólo tendrá un método que devolverá los valores de la ID de tipo int y el porcentaje abierto de tipo float.

La clase Sensor será la clase Padre de analógico que tendrá como atributo valor que será del tipo float y un método que leera el valor, getvalor(), el discreto tendrá un atributo valor de tipo bool y tendrá como método de nuevo getvalor().

Con esto se concluye la explicación del diagrama UML que hemos establecido para la realización de este proyecto, ahora debemos implementar el UML en el código ayudándonos del programa QT Creator.

**EXPLICACIÓN DEL PROGRAMA**

**CONCLUSIONES**

**TRABAJOS FUTUROS**

* **Implementar la clase comunicaciones con los reguladores**
* **Añadir tipo de los sensores y actuadores (método enumerado)**
* **Implementar usuarios.**