Aplicación web para la generación y simulación de eventos domóticos

Realizado por:

Javier González Serrano

Dirigido por:

José Luis Poza Luján

Tabla de contenidos

Resumen

Para la gestión inteligente de la energía en el hogar es necesario probar las diferentes estrategias implementadas por medio de diferentes algoritmos. Para poder comparar las estrategias es necesario realizar pruebas simuladas, generalmente mediante las llamadas cargas.
Las cargas de prueba son secuencias de mensajes que simulan los eventos que se dan en el sistema.
Los eventos son tratados por medio de los algoritmos de gestión energética que proponen gestionar los eventos para minimizar los parámetros de calidad que se hayan definido.
Además debe tenerse en cuenta, que tanto los eventos como la gestión energética debe aplicarse al mismo entorno, para ello se deberá diseñar un generador y diseñador de entorno que permita definir los eventos que se pueden generar y gestionar.
En este proyecto se desarrolla una herramienta que, a partir de los datos almacenados en una base de datos, configura la generación de los eventos de un sistema domótico.
La herramienta se ha desarrollado como aplicación Web para poder ser empleada en sistemas distribuidos a través de conexiones TCP

Introducción

Entorno

La domótica se podría definir como el conjunto de servicios proporcionados por sistemas tecnológicos e informáticos integrados bien en nuestras casas o en otros lugares (oficinas, hoteles, jardines, etc.) que nos ayudan en nuestras tareas diarias y mejoran nuestra calidad de vida.

¿Cuál es el origen de esta palabra? La domótica proviene del latín domus que significa casa, y de robótica, del checo robota (esclavo), uniendo ambas nomenclaturas aparece la "vivienda robotizada o informatizada". Una vivienda al servicio del usuario.

Solo podrá hablarse de domótica si la automatización de sus servicios está integrada en el conjunto de sistemas de una vivienda. Es decir, ha de estar presente tanto en su gestión de energía como en los sistemas de seguridad, comunicación y en diversas áreas que el habitante de la casa utiliza a diario (como por ejemplo, los mecanismos para subir y bajar las persianas, encender y apagar la luz, regular la calefacción, etc.)

Los sistemas pueden estar integrados por medio de redes interiores y/o exteriores de comunicación.

¿Sus ventajas? La domótica ofrece al usuario una mejor calidad de vida, con menor gasto tanto de dinero (al ahorrar energía en la realización de tareas) como de tiempo (ya que su utilización es sencilla y ágil).

Motivación

Para la gestión inteligente de la energía en el hogar es necesario probar las diferentes estrategias implementadas por medio de diferentes algoritmos. Para poder comparar las estrategias es necesario realizar pruebas simuladas, generalmente mediante las llamadas cargas.

Las cargas de prueba son secuencias de mensajes que simulan los eventos que se dan en el sistema.

Los eventos son tratados por medio de los algoritmos de gestión energética que proponen gestionar los eventos para minimizar los parámetros de calidad que se hayan definido.

Además debe tenerse en cuenta, que tanto los eventos como la gestión energética debe aplicarse al mismo entorno, para ello se deberá diseñar un generador y diseñador de entorno que permita definir los eventos que se pueden generar y gestionar.

Objetivos

Los principales objetivos del proyecto son:
Diseñar una aplicación configurable que permita generar eventos que simulen los eventos que se producen en un hogar, para poder aplicarse en los algoritmos de gestión energética.
Desarrollar y documentar una aplicación que permita diseñar entornos y lanzar la generación y la gestión de los eventos para el entorno determinado.

Descripción del documento

En el siguiente capítulo se revisará el entorno de la domótica, presentando las arquitecturas y las herramientas de las que disponemos hoy en día para simular o crear sistemas domóticos virtuales.

Entornos domóticos

Introducción

Una vez introducidos los términos relacionados con la domótica y explicado las motivaciones y objetivos que se pretenden conseguir, ha llegado la hora de ver realmente la viabilidad del proyecto. Esto lo conseguiremos comparando la

funcionalidad que nos ofrecen las aplicaciones de hoy en día con la que queremos desarrollar.
Por último veremos una posible clasificación de los dispositivos existentes en un sistema domótico con sus respectivos eventos asociados.

Sistemas simulación domótica

El objetivo principal de las nuevas aplicaciones de la domótica actual son las prevenciones, la seguridad y la confortabilidad.

A menudo solemos preguntarnos si estamos realmente seguros en nuestro hogar, ante cualquier tipo de riesgos, tanto si nos hallamos dentro de nuestra vivienda como si no.

Todo este tipo de sistemas de **seguridad** se vienen aplicando desde hace algún tiempo, habiendo mejorado considerablemente a medida que la domótica ha ido ampliando sus prestaciones, aplicaciones y dispositivos.

Dentro del campo de las alarmas, podemos encontrar una gran variedad de modelos, los cuales actúan de formas distintas, como son:

• Sistemas de vigilancia

Este sistema nos ofrece una serie de garantías en caso de intrusión o intento de robo. Para ello, pone a nuestro alcance artículos como las cámaras de vigilancia con dispositivo de infrarrojos, que en caso de detección de presencia activan una alarma a la central de **seguridad** o a la de policía.

• Botón del pánico

Este sistema nos permite, en caso de detectar ruidos o una presencia extraña, activar un botón que provocaría que se encendieran todas las luces de la vivienda, pudiéndose también conectar a la central de vigilancia o incluso a la de policía.

• Alarmas

Estos dispositivos están conectados, de la misma manera, a los distintos puntos de seguridad con los que están relacionados, por ejemplo, un sistema de escapes de gas o agua o una alarma de incendios. En este caso, se mandaría un aviso a la central de seguimiento y otro a los Bomberos.

También cabe destacar que en la domótica, dentro del sistema de prevención del hogar, existe lo que se llama sistema de simulación de presencia. Es muy efectivo, ya que activa el encendido y apagado de luces o la bajada y subida de toldos y persianas. Esto da una sensación de que la casa está habitada, aunque sus dueños estén de vacaciones. La activación de este dispositivo se realiza vía telefónica o vía Internet.

También existe software que permiten diseñar y simular entornos domóticos como vamos a ver a continuación.

Sistema: *VISIR* (Simulador de Instalaciones Domóticas)

Es una aplicación para entorno Windows que permite diseñar instalaciones domóticas y efectuar la simulación del comportamiento de las mismas en conexión directa con los autómatas programables de las series *SIMATIC S5* y *SIMATIC S7-200* de *SIEMENS*, encargado del control de la instalación dada.

Consta de dos módulos principales:

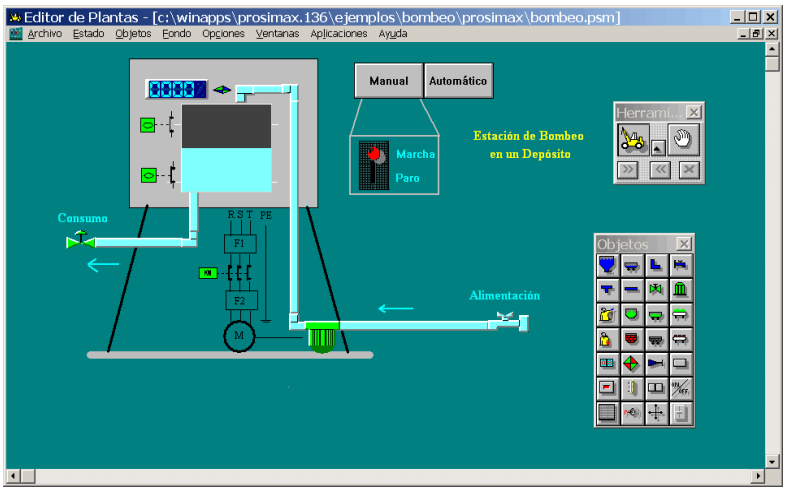
Módulo de Edición: Se diseña la instalación a simular mediante la selección de objetos dinámicos, organizados en grupos: detección de inundaciones, incendio, fugas de gas, seguridad, iluminación, toldos y persianas, control de cargas, etc. Se configuran tamaños y posición de objetos, modos de comportamiento, conexiones y representaciones gráficas, sin necesidad de programación. Adicionalmente se puede incorporar un dibujo de fondo.

Módulo de Simulación: Permite la conexión al autómata a través del cable serie de programación y se pueden comprobar las reacciones de la instalación domótica controlada por el programa de control real en el PLC. Contiene los controladores de comunicación: *S5* y *S7*-200.

Ventajas: Operación muy cercana a la realidad, incluye gran cantidad de objetos, sencilla determinación de errores de programación, flexibilidad, muy económico, rapidez de operación y fácil aprendizaje.

Esta simulación puede ser fácilmente usada por cualquier persona que tenga conocimientos básicos en computación y posee la opción de que el usuario mismo coloque los dispositivos en el lugar preferido que desee.

Sistema: *PROSIMAX* (Simulador de Procesos)



Es una aplicación para entorno Windows que permite diseñar procesos y efectuar la simulación en conexión directa con el autómata programable. *PROSIMAX* no es un paquete de visualización.

Consta de dos módulos principales:

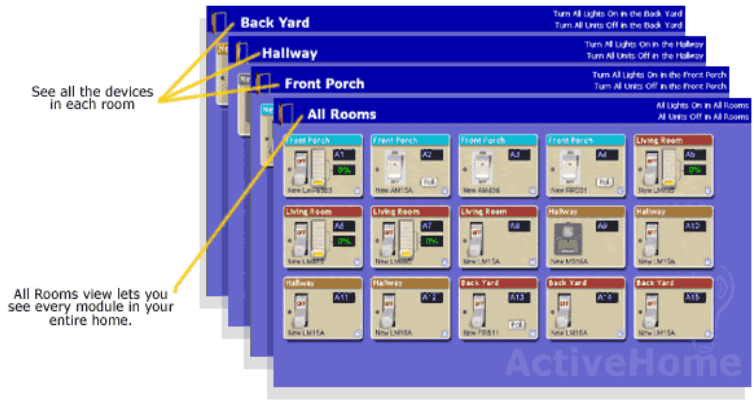
Módulo de Edición: Se diseña la planta a simular mediante la selección de objetos dinámicos. Se configuran comportamientos, conexiones y representaciones gráficas de los objetos de planta sin necesidad de programación. Opcionalmente permite incorporar un dibujo estático de la planta o proceso diseñado.

Módulo de Simulación: Permite la conexión al autómata a través del cable serie de programación y se pueden comprobar las reacciones del proceso guiado por el programa de control real en el PLC. El usuario puede intervenir de igual manera que lo haría en una instalación real. Tiene la posibilidad de comunicación con autómatas de las series *Simatic S5* y *Simatic S7-200.*

Ventajas: Prácticas más reales. Sencilla determinación de errores de programación. Flexibilidad. Economía.

Complemento de las rígidas y costosas maquetas. Rapidez de operación y fácil aprendizaje. Este simulador trabaja principalmente simulando procesos industriales.

Sistema: *ACTIVE HOME* (X-10)



Active Home es un software desarrollado por la empresa *X-10* de Estados Unidos, la cual se enfoca en aplicaciones que tienen que ver con la tecnología *X-10* antes descrita en este capítulo. *Active Home* brinda la seguridad y conveniencia de vivir en un hogar inteligente. Este sistema es de simple uso e

instalación pudiendo ser instalado en minutos por cualquier persona. Este software permite que en pocos minutos uno esté programando las luces y funciones de su casa en una computadora, pudiendo generar macros o rutinas programadas según horarios o días específicos. *Active Home* se comunica con los distintos dispositivos a través del cableado eléctrico existente. Una vez que se hayan programado con el software los eventos que se deben desarrollar se puede apagar la computadora y todo quedará almacenado en la memoria del *Active Home*.

Eventos en domótica

Fuentes de los eventos

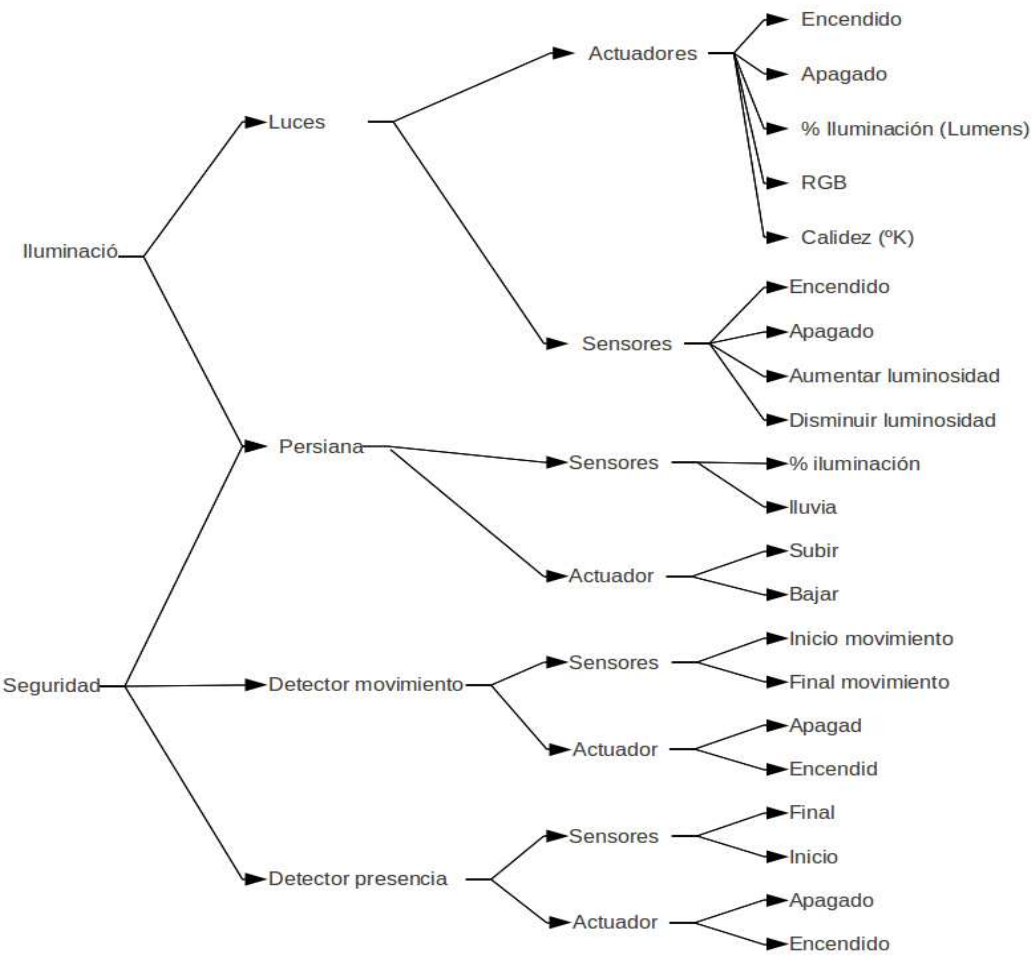
Caracterización: de donde vienen los eventos: personas, fuentes de energía, electrodoméstico.

Los eventos que se pueden generar son los habituales en la actividad diaria de un hogar, que comprenderán los siguientes entornos:

- Ambiente
  - Iluminación
  - Persianas y cortinas
- Electrodomésticos
  - Lavadora
  - Lavavajillas
  - Microondas
  - Etc.
- Electrónica
  - Televisiones
  - Ordenadores
  - Red de datos
  - Etc.
- Climatización

- Seguridad
- Otros aparatos

Organización




Electrodomesticos
Grandes
Línea marrón
Televisión
Reproductor de audio
Reproductor de vídeo
Reproductor de DVD
Cadena de música
Home cinema
Celulares
Sensores (out)
Inicio reproducción
Fin reproducción
Error
Actuador (in)
Eject
Play
Stop
Pause
Apagado
Encendido
Línea blanca
Lavavajillas
Campana extractora
Nevera
Aire acondicionado
Estufa doméstica
Horno
Lavadora
Secadora
Calentador
Calefactor
Congelador
Sensores (out)
Encendido motor
Apagado motor
Apertura entrada de agua
Cierre entrada de agua
Apertura salida de agua
Cierre salida de agua
Apertura detergente
Cierre detergente
Apertura suavizante
Cierre suavizante
Apertura puerta
Cierre puerta
Encendido resistencia
Apagado resistencia
Cambio del filtro
Averia
Exceso de carga
Actuador (in)
Abrir puerta
Play
Stop
Pause
Apagado
Encendido
Pequeños
Mantenimiento
Plancha
Aspiradora
Abrillantadora
Corta césped
Depuradora
Sensores (out)
Inicio depuración
Fin depuración
Cambio de filtros
Bomba averiada
Actuador (in)
Iniciar
Stop
Pause
Apagado
Encendido
Alimentacion
Microondas
Sandwichera
Licuadora
Cafetera
Tostadora
Freidora
Batidora
Robot de cocina
Sensores (out)
Encendido caldera
Apagado caldera
Encendido compresor
Apagado compresor
Inicio colado
Fin colado
Nivel mínimo de agua
Mantenimiento
Actuador (in)
Iniciar
Stop
Pause
Apagado
Encendido
Higiene y belleza
Depiladora
Afeitadora
Secador de pelo
Moldeador
Cepillo eléctrico

Conclusiones

En este capitulo hemos podido comprobar y catalogar la gran diversidad de posibles eventos que pueden generar nuestros electrodomésticos o dispositivos en el hogar.
Ademas se ha realizado un estudio de mercado centrándonos en las herramientas de software dirigidas a la domótica con el fin de hacer una aportación novedosa en dicho ámbito.
En el siguiente capítulo entraremos en detalles técnicos sobre la funcionalidad, objetivos y arquitectura de la aplicación.

Diseño del sistema

Introducción

Una vez que hemos visto lo que podemos encontrar en el mercado y hemos obtenido una visión general de lo que va a ser nuestra aplicación, ha llegado la hora de meterse en profundidad a detallar la funcionalidad del sistema, la arquitectura interna de la aplicación, los posibles requisitos futuros, etc.

Especificación conceptual

Gráficos, diagramas de bloques, esquemas con puntos.

Especificación de requisitos

IEEE 830

Introducción

En la actividad diaria de un hogar se realizan una serie de acciones que consumen energía. Por ejemplo encender una luz, poner la temperatura de

climatización o poner una colada lavadora. Estas acciones se pueden modelar por medio de eventos. Los eventos pueden tener márgenes de actuación, por ejemplo, se puede proponer variar la temperatura del aire acondicionado para evitar un tanto por cien de gasto, se puede programar la lavadora para que realice su tarea en un margen de horas en las que otros electrodomésticos no estén consumiendo energía.

Propósito

Para la gestión inteligente de la energía en el hogar es necesario probar las diferentes estrategias implementadas por medio de diferentes algoritmos. Para poder comparar las estrategias es necesario realizar pruebas simuladas, generalmente mediante cargas.
Las cargas son secuencias de mensajes que simulan los eventos que se dan en el sistema.
Los eventos son tratados por medio de los algoritmos de gestión energética que proponen gestionar los eventos para minimizar los parámetros.
Además debe tenerse en cuenta, que tanto los eventos como la gestión energética debe aplicarse al mismo entorno, para ello se deberá diseñar un generador y diseñador de entorno que permita definir los eventos que se pueden generar y gestionar.

Ámbito del Sistema

El futuro sistema consistirá en:

-Desarrollar una aplicación configurable que permita generar eventos que simulen los eventos que se producen en un hogar, para poder aplicarse en los algoritmos de gestión energética.

-Desarrollar una aplicación que permita diseñar entornos y lanzar la generación de los eventos para el entorno determinado.

-Desarrollar una aplicación cuya interfaz facilite la simulación del envio de eventos, añadiendo la posibilidad de poder registrar patrones de comportamiento o secuencias de eventos.

Definiciones, Acrónimos y Abreviaturas

En esta sección se definirán algunos términos comúnmente usados tanto en el ámbito de los sistemas domóticos como en el de la informática.

Sistema domótico o domótica: El término **Domótica** proviene de la unión de las palabras *domus* (que significa *casa* en latín) y *tica* (de *automática*, palabra en griego, 'que funciona por sí sola'). Se entiende por domótica al conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad, desde dentro y fuera del hogar. Se podría definir como la *integración de la tecnología en el diseño inteligente de un recinto cerrado*.

Evento: Un evento en lo que a la domótica respecta es un mensaje o señal que transmite un actuador o un sensor cuando se produce un determinado suceso, con el fin de informar al sistema gestor.

Actuador: Consiste en un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado. Este recibe la orden de un regulador o controlador y en función a ella genera la orden para activar un elemento final de control como, por ejemplo, una válvula.

Sensor: Dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Las variables de instrumentación pueden ser por ejemplo: temperatura, intensidad lumínica, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, pH, etc.

Ahorro energético: El ahorro energético no es algo tangible, sino un concepto al que se puede llegar de muchas maneras. En muchos casos no es necesario sustituir los aparatos o sistemas del hogar por otros que consuman menos sino una *gestión eficiente* de los mismos.

-Climatización: Programación y zonificación.

-Gestión eléctrica: Racionalización de cargas eléctricas: desconexión de equipos de uso no prioritario en función del consumo eléctrico en un momento dado. Gestión de tarifas, derivando el funcionamiento de algunos aparatos a horas de tarifa reducida.

-Uso de energías renovables.

Confort: Conlleva todas las actuaciones que se puedan llevar a cabo que mejoren el confort en una vivienda. Dichas actuaciones pueden ser de carácter tanto pasivo, como activo o mixtas.

-Iluminación: Apagado general de todas las luces de la vivienda. Automatización del apagado/ encendido en cada punto de luz. Regulación de la iluminación según el nivel de luminosidad ambiente.

-Automatización de todos los distintos sistemas/ instalaciones / equipos dotándolos de control eficiente y de fácil manejo.

-Integración del portero al teléfono, o del videoportero al televisor.

-Control vía Internet.

-Gestión Multimedia y del ocio electrónicos.

-Generación de macros y programas de forma sencilla para el usuario.

Seguridad: Consiste en una red de seguridad encargada de proteger tanto los Bienes Patrimoniales y la seguridad personal.

-Alarmas de intrusión (Antiintrusión): Se utilizan para detectar o prevenir la presencia de personas extrañas en una vivienda o edificio. Detección de un posible intruso (Detectores volumétricos o perimetrales). Cierre de persianas puntual y seguro. Simulación de presencia.

-Alarmas de detección de incendios, fugas de gas, escapes de agua, concentración de monóxido en garajes cuando se usan vehículos de combustión.

-Alerta médica. Teleasistencia.

-Acceso a Cámaras IP.

Comunicaciones: Son los sistemas o infraestructuras de comunicaciones que posee el hogar.

-Ubicuidad en el control tanto externo como interno, control remoto desde Internet, PC, mandos inalámbricos (p.ej. PDA con WiFi), aparellaje eléctrico.

-Tele asistencia

-Tele mantenimiento

-Informes de consumo y costes

-Transmisión de alarmas.

-Intercomunicaciones.

-Accesibilidad: Bajo este epigrafe se incluyen las aplicaciones o instalaciones de control remoto del entorno que favorecen la autonomía personal de personas con limitaciones funcionales, o discapacidad. El concepto "diseño" para todos es un movimiento que pretende crear la sensibilidad necesaria para que al diseñar un producto o servicio se tengan en cuenta las necesidades de todos los posibles usuarios, incluyendo las personas con diferentes capacidades o discapacidades, es decir, favorecer un diseño accesible para la diversidad humana.

La inclusión social y la igualdad son términos o conceptos más generalistas y filosóficos. La domótica aplicada a favorecer la accesibilidad es un reto ético y creativo pero sobre todo es la aplicación de la tecnología en el campo más necesario, para suplir limitaciones funcionales de las personas.

El objetivo no es que las personas con discapacidad puedan acceder a estas tecnologías, porque las tecnologías en si no son un objetivo, sino un medio. El objetivo de estas tecnologías es favorecer la autonomía personal.

Los destinatarios de estas tecnologías son todas las personas, ya que por enfermedad o envejecimiento, todos somos o seremos discapacitados, más pronto o más tarde.

Descripción General

Esta sección nos presenta una descripción general del sistema con el fin de conocer las
funciones que debe soportar, los datos asociados, las restricciones impuestas y cualquier otro factor que pueda influir en la construcción del mismo.

Las funciones que debe realizar el sistema se pueden agrupar de la siguiente manera:

Simulación de eventos: Debe permitir la creación y el envío de eventos emulando un sistema domótico real.

Gestión de eventos: Debe tener la posibilidad añadir un nuevo evento a un dispositivo concreto, así como editar los existentes o borrarlos, realizando las acciones pertinentes para conservar la aplicación en un estado estable.

Gestión de dispositivos: Debe proporcionar la opción de insertar un nuevo dispositivo en el sistema, editarlo y borrarlo sin comprometer la integridad de los datos de la aplicación.

Bus (o canal): Es un sistema digital que transfiere datos entre los componentes de una computadora o entre computadoras. Está formado por

cables o pistas en un circuito impreso, dispositivos como resistores y condensadores además de circuitos integrados.

Dirección IP: Es una etiqueta numérica que identifica, de manera lógica y jerárquica, a un interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP (Internet Protocol), que corresponde al nivel de red del protocolo TCP/IP. Dicho número no se ha de confundir con la dirección MAC que es un identificador de 48bits para identificar de forma única a la tarjeta de red y no depende del protocolo de conexión utilizado ni de la red. La dirección IP puede cambiar muy a menudo por cambios en la red o porque el dispositivo encargado dentro de la red de asignar las direcciones IP, decida asignar otra IP (por ejemplo, con el protocolo DHCP), a esta forma de asignación de dirección IP se denomina dirección IP dinámica (normalmente abreviado como IP dinámica).

Puerto: Es una forma genérica de denominar a una interfaz a través de la cual los diferentes tipos de datos se pueden enviar y recibir. Dicha interfaz puede ser de tipo físico, o puede ser a nivel de software (por ejemplo, los puertos que permiten la transmisión de datos entre diferentes ordenadores) (ver más abajo para más detalles), en cuyo caso se usa frecuentemente el término puerto lógico.

XML: siglas en inglés de eXtensible Markup Language ('lenguaje de marcas extensible'), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML, Android.

Características de los Usuarios

A continuación se van a describir las características generales de los usuario del producto, incluyendo nivel educacional, experiencia y experiencia técnica.

Como ya sabemos, la funcionalidad de dicho producto es simular eventos como si de un ambiente domótico se tratase, con el fin de analizarlos mediante otro modulo o dispositivo por lo que se presupone que el usuario será un usuario con experiencia técnica en el ámbito de la domótica.

Requisitos Futuros

A continuación voy a proponer posibles mejoras que se le podrían añadir a la aplicación en un futuro.

En primer lugar centrándonos en la eficiencia, la mejora mas significativa seria la implementación de varias funcionalidades mediante la tecnología Ajax. Este cambio dotaría a la aplicación de una ejecución mas fluida, además reduciría la carga en la red producida por las diversas consultas que realiza al servidor.

Por otro lado sería una buena opción para hacer mas vistosa la aplicación, dotarla de una interfaz animada en la que el usuario pudiese interactuar con los distintos dispositivos existentes en el entorno domótico configurándolos a su gusto.

Requisitos Específicos

En esta sección se van a describir los requisitos que deberá satisfacer nuestra aplicación cuando este finalizada.

Interfaces Externas

A continuación se van a describir los requisitos que afecten a la interfaz de usuario, interfaz
con otros sistemas (hardware y software) e interfaces de comunicaciones.

Interfaz de usuario debe:
-Disponer de toda la funcionalidad mas utilizada en una misma pantalla.
-Tener toda su funcionalidad accesible.
-Proporcionar una forma eficiente de realizar las funciones para las que ha sido pensada la aplicación.
-Tener una buena usabilidad.
-Ser amigable, con una curva de aprendizaje mínima.
-Poder utilizarse en cualquier tipo de sistema operativo o navegador.

Interfaz con otros sistemas debe:
-Comunicarse mediante un bus con la aplicación gestora.
-Permitir elegir una dirección IP destino y un puerto asociado a esa dirección.
-Poder enviar mensajes mediante el bus a la otra aplicación.

Interfaz de comunicaciones debe:
-Poder lanzar una conexión a un computador remoto.
-Permitir la comunicación unidireccional con un la aplicación gestora.
-Enviar mensajes XML informando a la aplicación gestora de los eventos simulados siguiendo un protocolo determinado.

Funciones

En esta subsección se van a describir todas aquellas acciones (funciones) que deberá llevar a cabo el software.

La aplicación estará constituida por estos objetos:

-Evento
-Dispositivo
-Patrón
-Simulador

Evento: Es un suceso que genera un dispositivo por lo que contendrá un nombre, una descripción, una id para el manejo interno y un dispositivo asociado.

Dispositivo: Como puede ser un electrodoméstico, sensor, persiana...por lo que estará formado por un nombre, una descripción, un id para el manejo interno y un conjunto de eventos asociados, dicho conjunto puede ser vacío.

Patrón: Se trata de una simulación preestablecida que intenta imitar el comportamiento humano o permite hacer pruebas a la aplicación gestora. Dicho patrón se compone de una secuencia de eventos que previamente ha lanzado el usuario y ha decidido guardar en el sistema para su posterior uso. Por tanto estará compuesto por un id para identificarlo, un nombre, una descripción y un conjunto de id's de eventos y dispositivos asociados a ese evento.

Simulador: Este objeto será el encargado de manejar a los citados anteriormente con el fin de producir la funcionalidad con la que la aplicación será creada. Por tanto dicha clase ofrecerá la posibilidad de:
-Listar los dispositivos guardados. El sistema mostrará una lista con todos los dispositivos introducidos en él.

-Insertar un dispositivo. Se permitirá la inserción de nuevos dispositivos en la aplicación.
-Modificar un dispositivo. Una vez listados los dispositivos en el sistema dará opción de editar su nombre, descripción y eventos asociados.
-Borrar un dispositivo lo que borrará sus eventos asociados. En el mismo listado de dispositivos aparecerá la opción de borrar el dispositivo.
-Listar los eventos asociados a un dispositivo. El sistema mostrará una lista de todos los eventos clasificados por el dispositivo asociado disponibles en la aplicación
-Insertar un evento indicando el dispositivo asociado. Se ofertará la posibilidad de insertar un evento asociándolo a un dispositivo existente en el sistema insertado los campos pertinentes.
-Modificar un evento indicando el dispositivo asociado. El sistema permitirá modificar los campos de un evento concreto.
-Borrar un evento indicando el dispositivo asociado. Dentro del listado de eventos asociados a un dispositivo, la aplicación nos dará la posibilidad de borrar un evento determinado.
-Generar un evento, que consistirá en localizar el evento asociado a un determinado dispositivo, generar un xml con los datos obtenidos de dicho evento mas la fecha y la hora a la que se ha generado para posteriormente enviarlo a un computador remoto. Por último se guardará dicho evento.
-Listar patrones. Mostrará todos los patrones guardados en el sistema.
-Enviar patrón. Generará y enviará los eventos tal y como fueron generados en la secuencia guardada.
-Guardar una secuencia de eventos como patrón. El usuario puede generar una secuencia de eventos y guardarla como un patrón indicando su nombre y una descripción.
-Configurar la conexión especificando la ip del sistema gestor y el puerto en el que esta escuchando mediante un archivo de configuración que la aplicación leerá al iniciarse.

Atributos del Sistema

En esta sección se detallarán los atributos de calidad del sistema:

Fiabilidad Se podría clasificar en tres grupos:

-Prevención de fallos: La aplicación deberá desarrollarse siguiendo los estándares de producción de software existentes(ISO) usando la metodología óptima para dicho proceso de producción.

-Detección de fallos: Para ello se desarrollarán pruebas las pruebas necesarias(pruebas unitarias, pruebas de integración...) con el fin de detectar los fallos cometidos en el proceso de implementación.

-Tolerancia a fallos: La aplicación esta obligada a tener una gestión de errores para garantizar un buen funcionamiento de la misma.

Mantenibilidad: La implementación de la aplicación debe permitir que sus componentes puedan ser modificados de forma fácil para corregir fallos, mejorar su funcionamiento u otros atributos o adaptarse a cambios en el entorno.

Portabilidad: Facilidad para transferir la aplicación de un entrono (software o hardware) a otro diferente.

Seguridad: Se deben implementar protocolos y medidas de seguridad necesarias para verificar el normal funcionamiento de la aplicación.

Especificación formal: capa de datos

En esta sección vamos a analizar servicios de persistencia y recuperación de información a las capas superiores, así como la arquitectura elegida para implementar dicha capa.

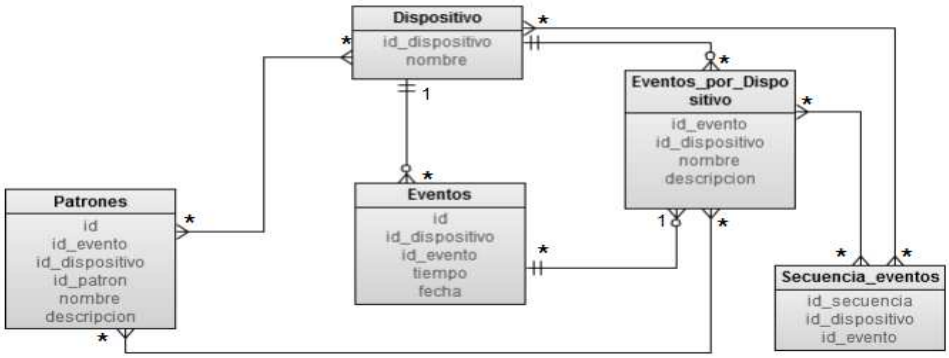
En primer lugar, como gestor de base de datos se eligió PostgreSQL ya que se caracteriza por ser un sistema estable, de alto rendimiento y multiplataforma. Al tratarse de una interfaz web, esto último nos viene muy bien debido al desconocimiento que se tiene sobre la plataforma que va a soportar nuestro sofware.

Con el fin de implementar la capa de persistencia de una forma correcta y eficiente, opté por implementar todos los accesos (o consultas SQL) a la capa de persistencia en archivos individuales agrupados por las diferentes clases existentes en el sistema. Con esto lo que se obtiene una buena organización de los ficheros y una buena tolerancia a futuros cambios. Si deseáramos realizar algún cambio que afectara a la capa de persistencia, la repercusión del mismo solo afectaría a un fichero facilitando en gran medida la tarea.

A continuación vamos a explicar el modelo entidad relación.

El Modelo Entidad Relación (ER) permite desarrollar un diseño de base de datos en un esquema de alto nivel conceptual sin considerar los problemas de bajo nivel como la eficiencia, el modelo implícito del administrador de base de datos o las estructuras físicas de los datos. El Modelo Entidad Relación se hizo muy popular para el diseño de base de datos y es usado extensivamente. Para aumentar su poder de expresión, muchos investigadores han introducido o propuesto ciertas extensiones a este modelo. Algunas de estas extensiones son importantes, mientras que otras agregan poco poder de expresión, pero proveen características auxiliares.

Entity Relationship Diagram (ERD)

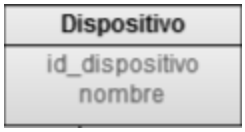


En el podemos encontrar las diferentes tablas y relaciones que modelan nuestra aplicación las cuales vamos a explicar a continuación:

Tablas

Evento:

- **Id:** Identificador de un evento dentro de la tabla.
- **Id_dispositivo:** Identificador del dispositivo asociado al dicho evento.
- **Id_evento:** Identificador único del evento.
- **Tiempo:** Hora a la que se produjo dicho evento.
- **Fecha:** día concreto en el que se produjo el evento.

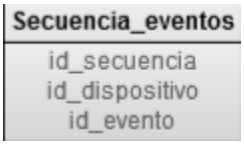


Dispositivo:

- **Id_dispositivo:** Identificador único del dispositivo.
- **Nombre:** Nombre de dicho dispositivo.

Evento_por_dispositivo:

- **Id_evento:** Identificador del evento.
- **Id_dispositivo:** Identificador del dispositivo.
- **Nombre:** Nombre del evento asociado a un dispositivo.
- **Descripción:** Descripción breve del evento asociado a un dispositivo.



Secuencia_de_eventos:

- **Id_secuencia:** Identificador único de una secuencia.
- **Id_dispositivo:** Identificador del dispositivo.
- **Id_evento:** Identificador del evento asociado al dispositivo.

Patrón:

- **Id:** Identificador del patrón dentro de la tabla.
- **Id_evento:** Identificador del evento asociado al dispositivo.
- **Id_dispositivo:** Identificador del dispositivo.
- **Id_patron:** Identificador único del patrón.
- **Nombre:** Nombre asignado al patrón.
- **Descripción:** Breve descripción del comportamiento a imitar por el patrón.

Relaciones

Dispositivo-Evento: Esta relación indica que cada dispositivo puede tener ninguno o varios eventos, por que lo que pueden existir dispositivos dados de alta en el sistema que no contengan ningún evento asociado.

Dispositivo-Evento_por_Dispositivo: Esta relación nos muestra para cada dispositivo, todos los eventos asociados que puede lanzar.

Evento_por_Dispositivo-Evento: En esta relación podemos observar la relación de los eventos con los que están asociados a un dispositivo.

Dipositivo-Patrón: Observando esta relación podemos ver los dispositivos existentes en el sistema que intervienen en un cada patrón.

Evento_por_Dispositivo-Patrón: La siguiente relación nos muestra que eventos asociados a un dispositivo dado intervienen en un patrón.

Dispositivo-Secuencia_de_Eventos: Como el nombre de la relación indica, esta nos visualizará los dispositivos que han generado eventos que a su vez han generado una secuencia de eventos.

Evento_por_Dispositivo-Secuencia_de_Eventos: Esta relación identifica los eventos asociados a un dispositivo que se han lanzado dentro de cada secuencia.

Especificación formal: capa de negocio

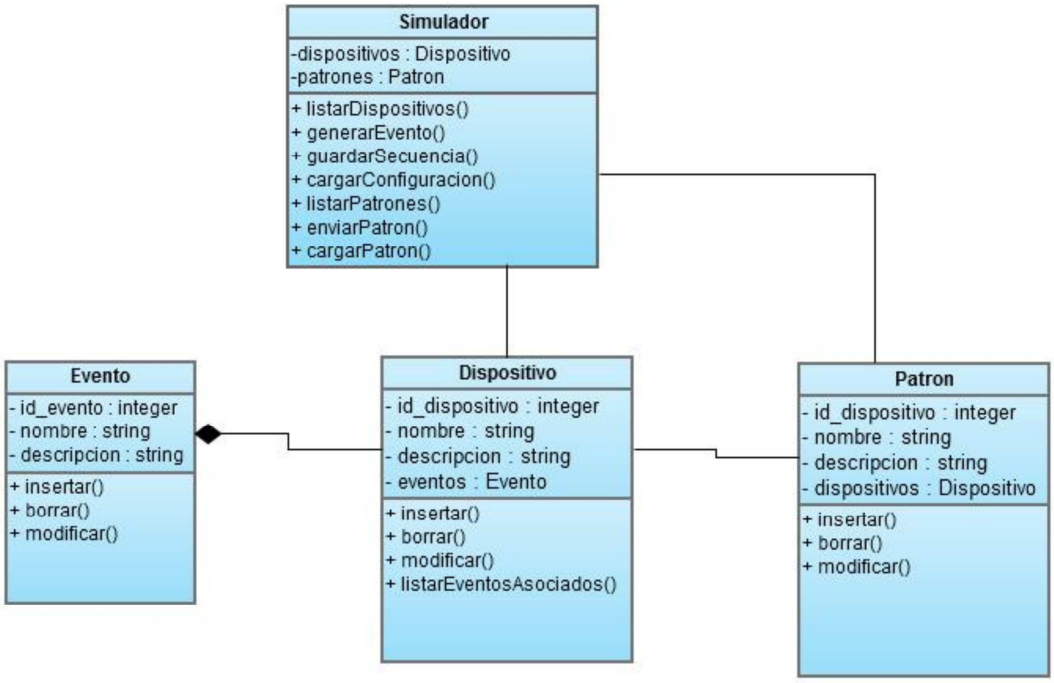
A continuación vamos a ver como esta implementada la capa de negocio o lógica interna de la aplicación mediante el estudio del diagrama de clases.

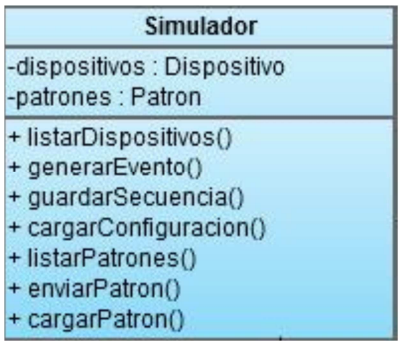
Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.
Representación de:

- Requerimientos en entidades y actuaciones.
- La arquitectura conceptual de un dominio.
- Soluciones de diseño en una arquitectura.
- Componentes de software orientados a objetos.

Definiciones

- **Propiedades** también llamados atributos o características, son valores que corresponden a un objeto, como color, material, cantidad, ubicación. Generalmente se conoce como la información detallada del objeto. Suponiendo que el objeto es una puerta, sus propiedades serían: la marca, tamaño, color y peso.
- **Operaciones** comúnmente llamados métodos, son aquellas actividades o verbos que se pueden realizar con/para este objeto, como por ejemplo abrir, cerrar, buscar, cancelar, acreditar, cargar. De la misma manera que el nombre de un atributo, el nombre de una operación se escribe con minúsculas si consta de una sola palabra. Si el nombre contiene más de una palabra, cada palabra será unida a la anterior y comenzará con una letra mayúscula, a excepción de la primera palabra que comenzará en minúscula. Por ejemplo: abrirPuerta, cerrarPuerta, buscarPuerta, etc.
- **Interfaz** es un conjunto de operaciones que permiten a un objeto comportarse de cierta manera, por lo que define los requerimientos mínimos del objeto. Hace referencia a polimorfismo.
- **Herencia** se define como la reutilización de un objeto padre ya definido para poder extender la funcionalidad en un objeto hijo. Los objetos hijos heredan todas las operaciones y/o propiedades de un objeto padre. Por ejemplo: Una persona puede especializarse en Proveedores, Acreedores, Clientes, Accionistas, Empleados; todos comparten datos básicos como una persona, pero además cada uno tendrá información adicional que depende del tipo de persona, como saldo del cliente, total de inversión del accionista, salario del empleado, etc.





Simulador: Esta clase es la encargada de manejar al resto de clases, a través de ella se puede realizar toda la funcionalidad de la aplicación.

Atributos:

- Dispositivos: Contiene un array de objetos de tipo Dispositivo.
- Patrones: Contiene un array con los objetos Patrón existentes en el sistema.

Métodos:

- ListarDispositivos() Consiste en una función que se encarga de recuperar todos los dispositivos existentes en el sistema y mostrarlos por pantalla en una lista.

- GenerarEvento() Esta es una de las funciones mas importantes de la aplicación ya que es la encargada de generar el evento y enviarlo por red. Su funcionamiento es el siguiente: Lo primero que hace una vez seleccionado el evento asociado a un dispositivo que se desea generar es recuperar dicho objeto. Una vez obtenido, saca parte de sus atributos como el nombre, la id del evento, la id del dispositivo, etc. y con estos datos genera un mensaje XML añadiéndole la hora y la fecha en la que este evento se ha generado. Lo segundo que hace es abrir una conexión tcp con la estación gestora que estará escuchando mediante los parámetros que se han configurado previamente en el archivo de configuración. Una vez esta establecida la comunicación, envía el mensaje XML por la red. Por último guarda las características del evento generado en la tabla secuencia para la posible creación de un patrón.

- GuardarSecuencia() La funcionalidad de este método consiste en una vez generados uno o varios eventos, guardar esa secuencia de generación de eventos como un patrón.

- CargarConfiguracion() Esta función es la encargada de leer el fichero de configuración y configurar la aplicación en torno a los parámetros leídos.

- ListarPatrones() Consiste en una función que se encarga de recuperar todos los patrones existentes en el sistema y mostrarlos por pantalla en una lista.

- EnviarPatron() Dicho método como su nombre indica es el encargado de enviar un patrón previamente seleccionado. Para ello buscara la secuencia de eventos asociada a dicho patrón e irá enviado uno a uno los eventos que aparecen en ella tal y como se enviaron cuando se creó la secuencia.

- CargarPatron() Este método oferta la funcionalidad de cargar un patrón desde un archivo de texto con un formato determinado.

Dispositivo: Esta clase representa los distintos aparatos o dispositivos que podemos contener en nuestro sistema. La misma contiene la funcionalidad asociada a los dispositivos, así como una serie de atributos para identificar y describir cada uno de los objetos almacenados.

Atributos:

- Id_dispositivo: Consiste en un integer que identifica de manera inequívoca a cada dispositivo.

- Nombre: Es la cadena de caracteres correspondiente al nombre del dispositivo.

- Descripción: Se trata de una cadena de caracteres de longitud media que se usa para dar una descripción del dispositivo almacenado.

- Eventos: Consiste en un array que contiene los eventos asociados al dispositivo.

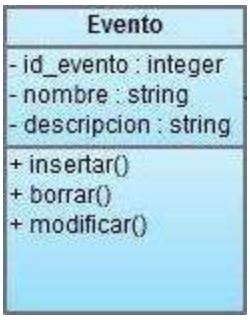
Métodos:

- Insertar() Este método es el que permite dar de alta los dispositivos en nuestra aplicación. Funciona de la siguiente manera, el primer paso es introducir los datos requeridos por la aplicación. Una vez introducidos, la aplicación validará dichos datos y dependiendo de el resultado de la validación almacenará o no el nuevo dispositivo en la base de datos.

- Borrar() Esta función es la encargada de eliminar un dispositivo. Además de eliminar el dispositivo, si este tiene eventos asociados, con el fin de mantener la integridad de los datos y ser fiel al modelo UML descrito, estos eventos se eliminarán también.

- Modificar() Dicho método es el que nos permite editar la información o atributos asociada a un dispositivo. Al igual que en la inserción, cuando se quiera editar información relevante de algún dispositivo, se deberá proceder a la validación de los datos introducidos por lo comentado en la inserción y dependiendo del resultado obtenido se aceptarán los cambios o se denegarán.

- ListarEventosAsociados() La utilidad de esta función como su nombre indica es buscar los eventos asociados a un dispositivo concreto y mostrarlos en un listado.



Evento: Esta clase representa el objeto de tipo Evento. Dicho objeto es una de las partes mas importantes del sistema ya que participa prácticamente en toda la aplicación.

Atributos:

- Id_evento: Consiste en un integer que identifica de manera inequívoca a cada evento.

- Nombre: Es la cadena de caracteres correspondiente al nombre del evento.

- Descripción: Se trata de una cadena de caracteres de longitud media que se usa para dar una descripción del evento almacenado.

Métodos:

- Insertar() Este método es el que permite dar de alta los eventos asociados a un dispositiv en nuestra aplicación. Funciona de la siguiente manera, el primer paso es seleccionar el dispositivo al que se le va a asociar dicho evento, el segundo paso es introducir los datos requeridos por la aplicación. Una vez introducidos, la aplicación validará dichos datos y dependiendo de el resultado de la validación almacenará o no el nuevo evento en la base de datos.

- Borrar() Esta función es la encargada de eliminar un evento de la aplicación. Para ello habrá que seleccionar un dispositivo, listar los eventos asociados y seleccionar el que se desea eliminar.

- Modificar() Dicho método es el que nos permite editar la información o atributos asociada a un evento. Al igual que en la inserción, cuando se quiera editar información relevante de algún evento, se deberá proceder a la validación de los datos introducidos por lo comentado en la inserción y dependiendo del resultado obtenido se aceptarán los cambios o se denegarán.

Patrón: Esta clase es la que se encarga de almacenar secuencias de eventos como puede ser un determinado comportamiento humano o un testeo de la aplicación. Dicha clase nos permite almacenar una secuencia de eventos previamente generados, ofreciendo la posibilidad de volver a generarla cuando el usuario lo deseé.

Atributos:

- Id_dispositivo: Consiste en un integer que identifica de manera inequívoca a cada dispositivo.

- Nombre: Es la cadena de caracteres correspondiente al nombre del patrón.

- Descripción: Se trata de una cadena de caracteres de longitud media que se usa para dar una descripción del patrón almacenado.

- Dispositivos: Consiste en un array que contiene los dispositivos que intervienen en un patrón determinado.

Métodos:

- Insertar() Este método es el que permite guardar una secuencia de eventos como un patrón en nuestra aplicación. Una vez pulsado el botón de guardar secuencia como patrón, la aplicación recuperará la secuencia de eventos y la guardará como un patrón una vez se introduzcan y se validen los datos requeridos por la aplicación. Por ultimo se vaciará la tabla de secuencia de eventos.

- Borrar() Esta función es la encargada de eliminar un patrón de la aplicación. Para ello bastaría con seleccionar uno patrón y darle al botón eliminar.

- Modificar() Dicho método es el que nos permite editar la información o atributos asociada a un patrón. Al igual que en la inserción, cuando se quiera editar información relevante de algún patrón, se deberá proceder a la validación de los datos introducidos por lo comentado en la inserción y dependiendo del resultado obtenido se aceptarán los cambios o se denegarán.

Especificación formal: capa de presentación

Interfaz de usuarios. ☐ UML, bloques, prototipado
En esta sección vamos a ver como se ha diseñado la capa de presentación mediante el diagrama de casos de uso.

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. En el diagrama de casos de uso se representa también el sistema como una caja rectangular con el nombre en su interior. Los casos de uso están en el interior de la caja del sistema, y los

actores fuera, y cada actor está unido a los casos de uso en los que participa mediante una línea. En la Figura 1 se muestra un ejemplo de Diagrama de Casos de Uso para un cajero automático.

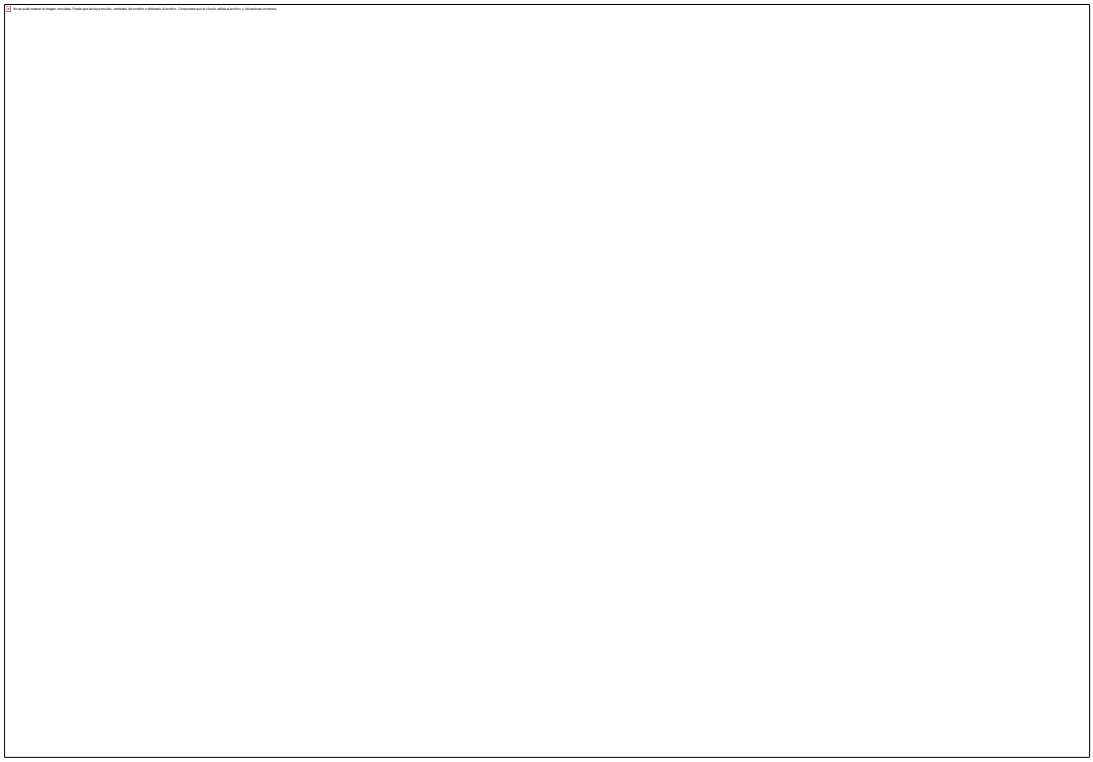


Figura 1 – Ejemplo de diagrama de casos de uso

Elementos

Los elementos que pueden aparecer en un Diagrama de Casos de Uso son: actores, casos de uso y relaciones entre casos de uso.

Actores

Un actor es algo con comportamiento, como una persona (identificada por un rol), un sistema informatizado u organización, y que realiza algún tipo de interacción con el sistema.. Se representa mediante una figura humana dibujada con palotes. Esta representación sirve tanto para actores que son personas como para otro tipo de actores.

Casos de Uso

Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Expresa una unidad coherente de funcionalidad, y se representa en el Diagrama de Casos de Uso mediante una elipse con el nombre del caso de uso en su interior. El nombre del caso de uso debe reflejar la tarea específica que el actor desea llevar a cabo usando el sistema.

Relaciones entre Casos de Uso

Un caso de uso, en principio, debería describir una tarea que tiene un sentido completo para el usuario. Sin embargo, hay ocasiones en las que es útil describir una interacción con un alcance menor como caso de uso. La razón para utilizar estos casos de uso no completos en algunos casos, es mejorar la comunicación en el equipo de desarrollo, el manejo de la documentación de casos de uso. Para el caso de que queramos utilizar estos casos de uso más pequeños, las relaciones entre estos y los casos de uso ordinarios pueden ser de los siguientes tres tipos:

• Incluye (<>): Un caso de uso base incorpora explícitamente a otro caso de uso en un lugar especificado en dicho caso base. Se suele utilizar para encapsular un comportamiento parcial común a varios casos de uso. En la Figura 2 se muestra cómo el caso de uso Realizar Reintegro puede incluir el comportamiento del caso de uso Autorización.

Figura 2 - Ejemplo de Relación <>

• Extiende (<>): Cuando un caso de uso base tiene ciertos puntos (puntos de extensión) en los cuales, dependiendo de ciertos criterios, se va a realizar una interacción adicional. El caso de uso que extiende describe un comportamiento opcional del sistema (a diferencia de la relación incluye que se da siempre que se realiza la interacción descrita) En la Figura 3 se muestra como el caso de uso Comprar Producto permite explícitamente extensiones en el siguiente punto de extensión: info regalo. La interacción correspondiente a establecer los detalles sobre un producto que se envía como regalo están descritos en el caso de uso Detalles Regalo.

Figura 17 - Ejemplo de Relación <>

Ambos tipos de relación se representan como una dependencia etiquetada con el estereotipo correspondiente (<> o <>), de tal forma que la flecha indique el sentido en el que debe leerse la etiqueta. Junto a la etiqueta <> se puede detallar el/los puntos de extensión del caso de uso base en los que se aplica la extensión.

• Generalización ( ): Cuando un caso de uso definido de forma abstracta se particulariza por medio de otro caso de uso más específico. Se representa por una línea continua entre los dos casos de uso, con el triángulo que simboliza generalización en UML (usado también para denotar la herencia entre clases) pegado al extremo del caso de uso más general. Al igual que en la herencia entre clases, el caso de uso hijo hereda las asociaciones y características del caso de uso padre. El caso de uso padre se trata de un caso de uso abstracto, que no está definido completamente. Este tipo de relación se utiliza mucho menos que las dos anteriores.

En nuestro diagrama disponemos de un solo actor y esto se debe a que al tratarse de un simulador de eventos domóticos, el usuario al que va dirigida la aplicación se supone que es un usuario con conocimientos de domótica. Tampoco es necesario que exista un administrador o un gestor de la aplicación para que esta funcione correctamente.

El usuario puede realizar toda la funcionalidad que se muestra en el diagrama de casos de uso.

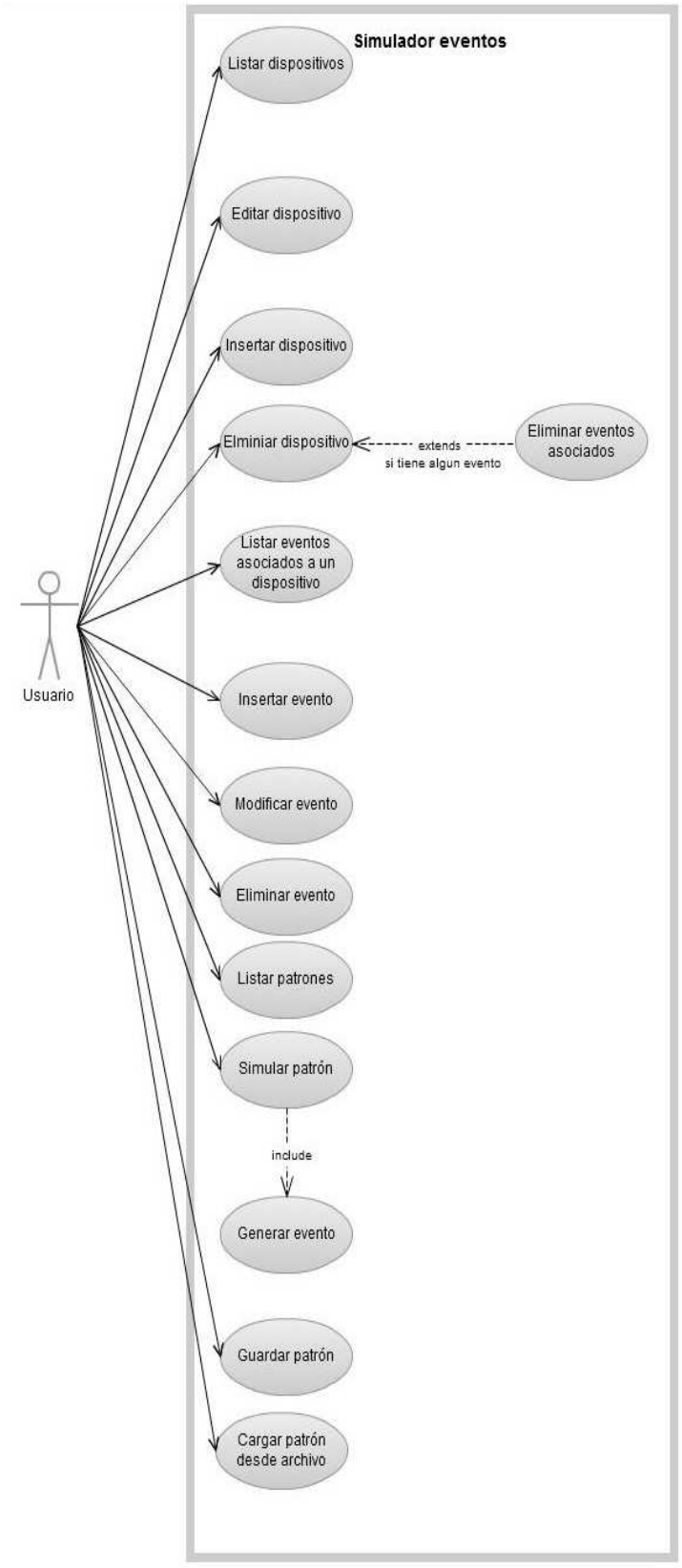
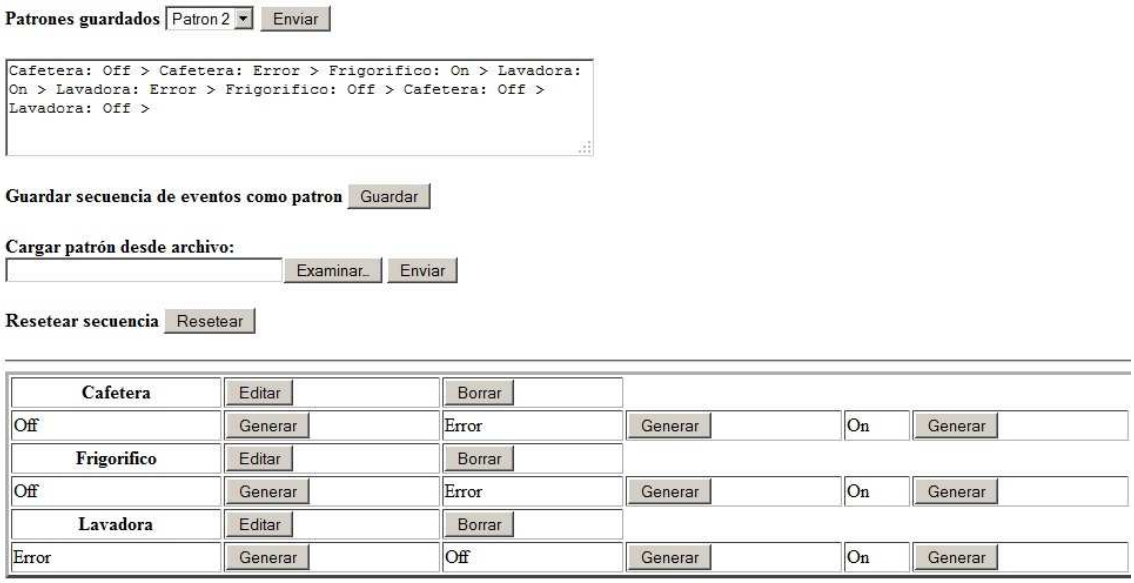


Diagrama de casos de uso.

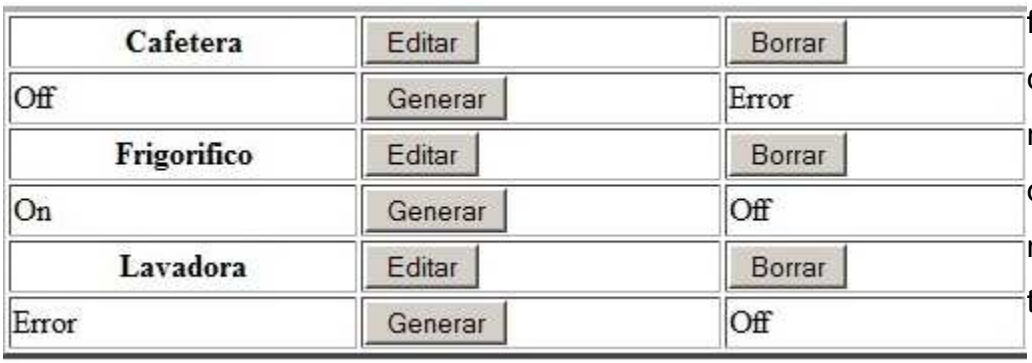
Para facilitar la usabilidad de la aplicación se ha intentado ofrecer al usuario la mayor parte de la funcionalidad ofertada por la aplicación en una misma vista o interfaz.



Interfaz principal

A continuación apoyándonos en la imagen de la interfaz vamos a explicar su funcionalidad.

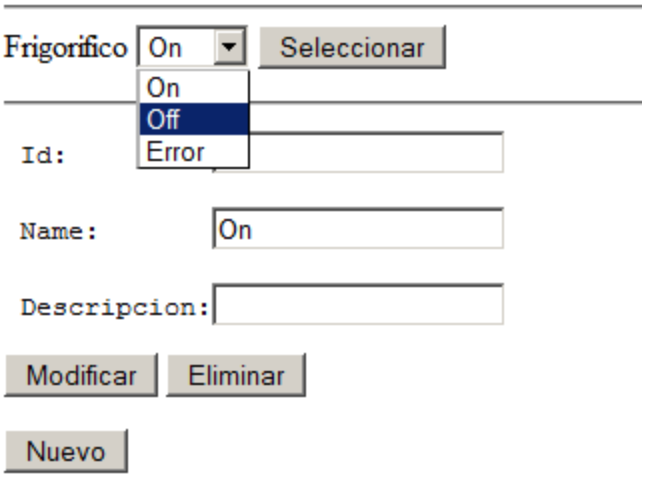
En primer lugar tenemos como se puede apreciar en la figura, toda la



funcionalidad relativa

a la gestión de dispositivos y eventos asociados así como la referente a la generación de los eventos.

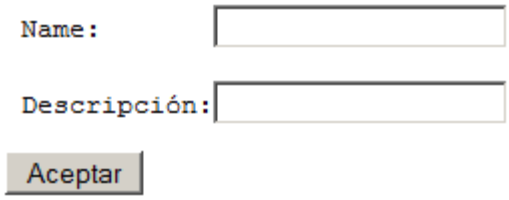
La aplicación nos muestra un listado de todos los dispositivos y sus eventos asociados almacenados en el sistema y nos ofrece la funcionalidad asociada a cada evento o dispositivo. De esta manera podemos acceder de forma fácil a las funciones de la aplicación.
Como vemos en la figura esta interfaz nos ofrece la posibilidad de generar un determinado evento asociado a un dispositivo.
Por otro lado, si pinchamos en el botón editar, la aplicación nos ofrece esta interfaz destinada a la gestión (inserción, modificación y borrado) del dispositivo y sus eventos como vemos en la siguiente imagen.



Interfaz de gestión de dispositivos y eventos asociados.

En primer lugar, la aplicación nos muestra un listado de los eventos asociados al dispositivo elegido. El usuario puede seleccionar un evento y proceder a su modificación o borrado del mismo.

Si lo que se desea es introducir un evento asociado a este dispositivo el usuario deberá pinchar en la opción de nuevo, lo que nos llevaría a la siguiente interfaz.



Interfaz de inserción de eventos asociados a un dispositivo

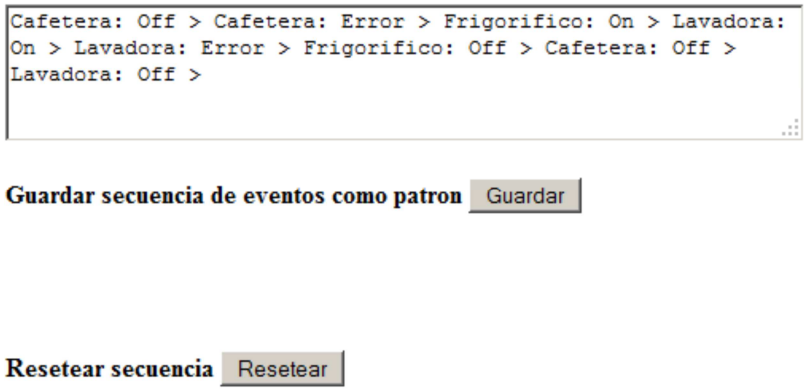
En ella debemos introducir un nombre y una descripción del nuevo evento que queremos ingresar y una vez introducido pulsar el botón Aceptar. Si no hay ningún error en la inserción de datos, la aplicación insertará el nuevo evento y mostrará la anterior interfaz.

Retomando la interfaz principal, nos encontramos con la siguiente figura en la que se muestra la funcionalidad relacionada con el envío de patrones.

Interfaz de envío de patrones.

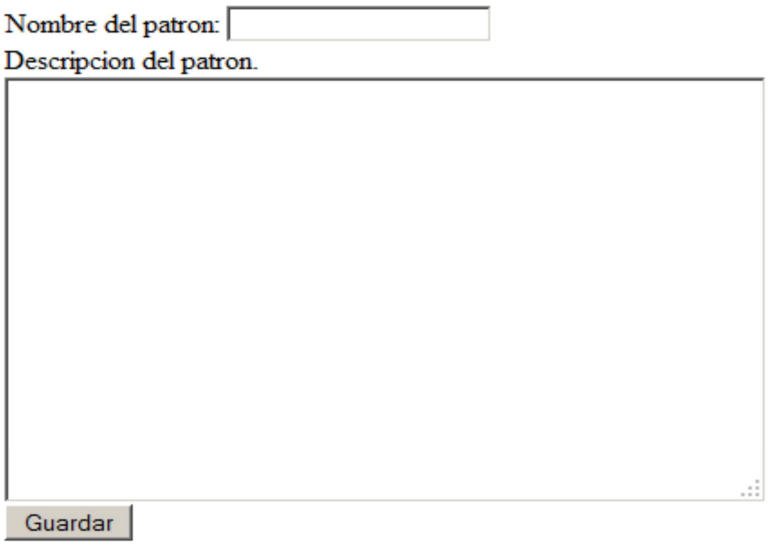
Esta interfaz muestra al usuario el listado de todos los patrones guardados en el sistema. Si se desea enviar un patrón bastaría con seleccionar uno y pulsar el botón enviar.

Por ultimo tenemos la interfaz relativa a la gestión de la secuencia de eventos. En ella como se puede apreciar en la siguiente imagen, se muestra un listado de todos los eventos generados desde el inicio de la aplicación.



Interfaz gestión de secuencias de eventos.

Como podemos observar esta interfaz nos permite ver la secuencia de eventos generada hasta el momento, ofreciendo al usuario la posibilidad de resetearla o bien de guardarla como un patrón lo que nos llevaría a la siguiente interfaz.



Interfaz de inserción de un nuevo patrón.

En ella se nos permite dar un nombre al nuevo patrón que se desea insertar y describir cual va a ser su comportamiento a imitar. Una vez introducidos los datos la aplicación correctamente la aplicación insertará el nuevo patrón y nos mostrará la interfaz principal.

Por último y no por ello menos importante nos queda la interfaz de carga de patrón mediante archivo.

Interfaz de carga de patrón

desde archivo.

Esta nos ofrece la posibilidad de cargar un patrón desde un archivo y si tiene éxito la carga enviarlo.

Conclusiones

Una vez visto toda la arquitectura y el diseño de la aplicación en profundidad ha llegado la hora de pasar a temas de implementación como son los lenguajes de programación a elegir, el hardware sobre el que va a ejecutarse la aplicación, etc.

Desarrollo del sistema

Introducción

A continuación se van a comentar los aspectos técnicos de la aplicación como son los lenguajes de programación usados en la implementación, el IDE utilizado, así como los diversos prototipos que marcaron la evolución de la aplicación.

Aspectos técnicos

Lenguajes de programación (SW)

En este apartado analizamos los lenguajes utilizados argumentando su uso en la implementación de esta aplicación.

En primer lugar, como IDE se ha utilizado Adobe Dreamweaver CS5 ya que da soporte a los lenguajes utilizados, es completo y no tiene a penas curva de aprendizaje.
Como gestor y servidor de base de datos se ha utilizado PostgreSQL 9.0.4, ya que debido al uso que se le va a dar, PostgreSQL no tiene nada que envidiar en comparación con otros gestores como son Oracle o MySQL. También ha influido la destreza al manejar la herramienta obtenida previamente.

Por otro lado, para implementar la lógica de la aplicación se ha utilizado PHP ya que al tratarse de una aplicación web necesitaba un lenguaje de

programación en el servidor capaz de crear paginas web dinámicas y por motivos éticos y funcionales lo he elegido frente a la otra opción que es ASP.

Plataformas (HW)

Esta aplicación esta pensada y diseñada para correr sobre cualquier hardware ya que los requisitos mínimos que necesita la aplicación para ejecutarse son insignificantes a día de hoy.
El único hardware que requiere es una tarjeta de red que pueda enviar paquetes tpc a un destino concreto.

Prototipos

En el desarrollo de la aplicación se ha llevado a cabo la implementación de 3 prototipos no desechables y acumulativos en lo que a funcionalidad se refiere.

Prototipo 1

La primera versión o prototipo 1 abarca únicamente la generación de los eventos con su posterior envió.
En esta versión el esfuerzo se centró en construir el XML, leyendo los datos del evento de la base de datos y añadiendo la fecha y hora a la que se ha generado, para posteriormente enviarlo por red a un servidor tcp.
Dicho servidor mostraba el XML recibido en formato de texto por pantalla.

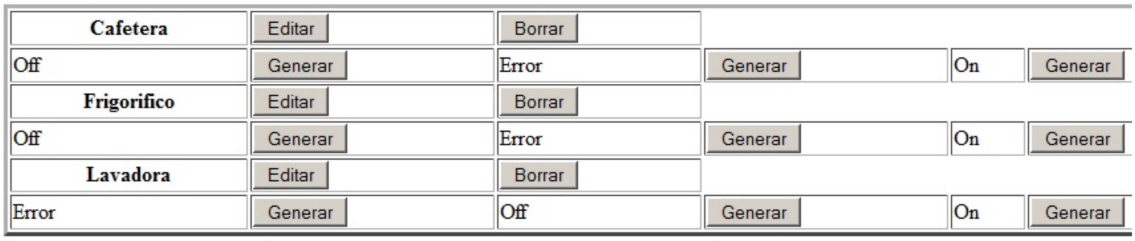
Interfaz del primer prototipo.

Una vez comprobado el buen funcionamiento de lo explicado anteriormente, lo siguiente que se implementó fue parte de la base de datos. Concretamente las tablas de evento, eventos_por_dispositivo y dispositivo.
Después se implemento la funcionalidad de leer la configuración relativa al envío de datos por la red mediante un fichero de configuración.

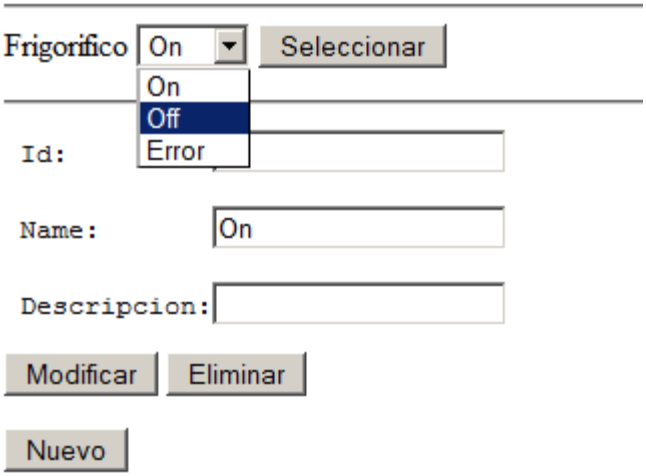
Por ultimo se comprobó la integridad de la base de datos y se procedió a llenarla de datos.

Prototipo 2

La segunda versión de la aplicación se centró básicamente en la gestión de los dispositivos y los eventos asociados.
Se implementó toda la lógica que permite acceder a la base de datos para recuperar información relativa a los dispositivo y eventos asociados y gestionar dispositivo y eventos asociados (insertar, modificar y borrar).
También se desarrolló toda la parte de la interfaz gráfica que permite listar los datos de los dispositivo y eventos asociados y la que da soporte a la lógica mencionada en el párrafo anterior.
La interfaz resultante la podemos ver en la siguientes imágenes.

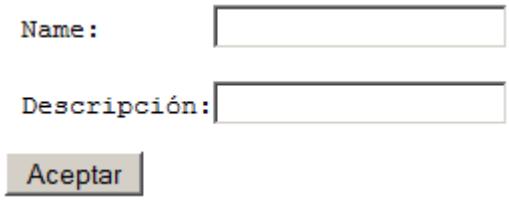


Interfaz del segundo prototipo



Interfaz de gestión de dispositivos

y eventos asociados.



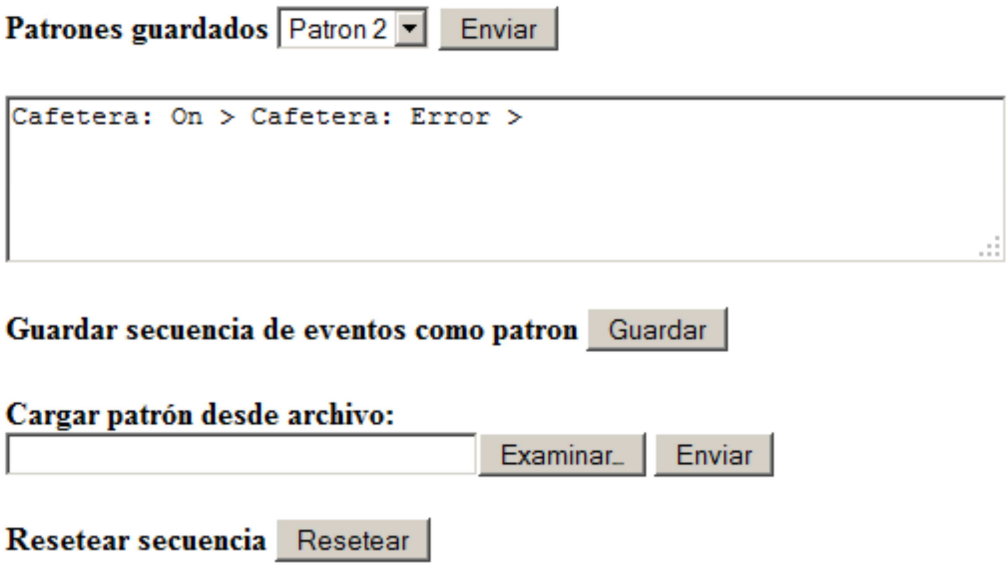
Interfaz de inserción de eventos

asociados a un dispositivo

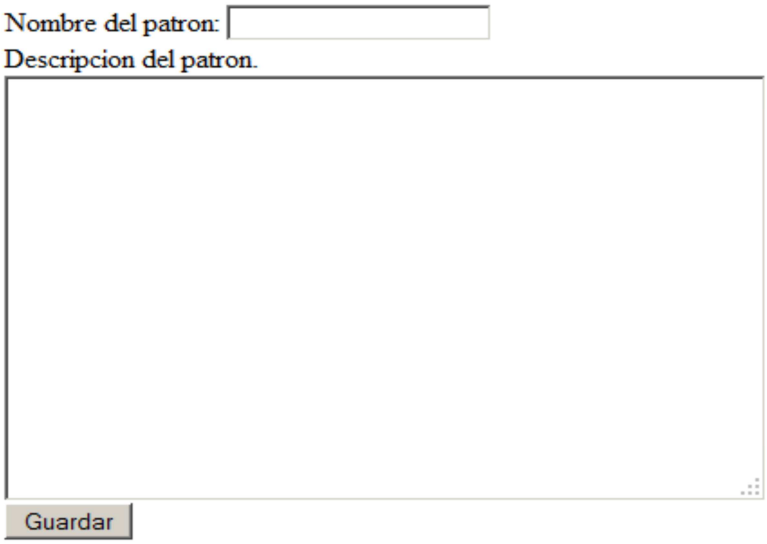
Al igual que en la anterior versión se comprobó que el funcionamiento de la aplicación fuese correcto.

Prototipo 3

En la tercera y ultima versión se desarrolló todo la funcionalidad referente a los patrones y las secuencias de eventos.
En primer lugar se implementó la parte de las secuencias de eventos agregando una nueva tabla a la base de datos para guardarlas y mostrarlas posteriormente.
Después se desarrolló la lógica necesaria para el funcionamiento de los patrones y se insertó una nueva tabla en la base de datos para almacenarlos.
A la par que se implementaba lo citado anteriormente se desarrollaban las interfaces que ofertan dicha funcionalidad.
Dichas interfaces las podemos observar a continuación en las siguientes imágenes.



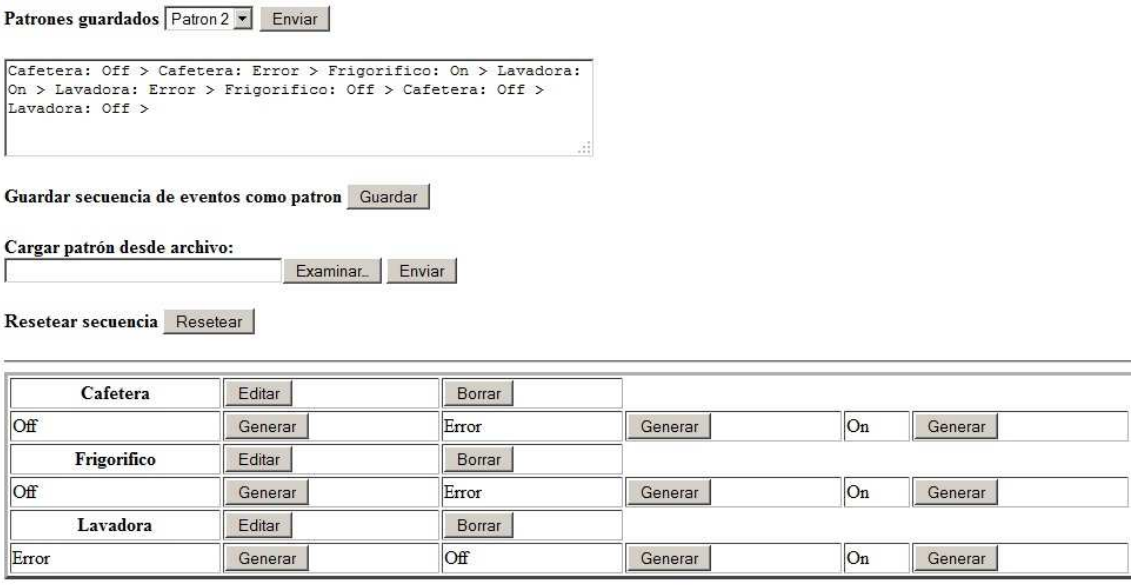
Interfaz de patrones y secuencias de eventos



Interfaz de inserción

de un nuevo patrón

Quedando como interfaz resultante la que se muestra a continuación.



Por ultimo se añadió la funcionalidad de cargar un patrón desde archivo y enviarlo y se comprobó igual que en las anteriores versiones que todo funcionase correctamente.

Pruebas realizadas

En esta sección se comentarán las pruebas a las que se le ha sometido a la aplicación para garantizar un correcto funcionamiento.
Además de las pruebas unitarias, funcionales y de integración, se realizaron pruebas de rendimiento con el fin de determinar la cantidad de usuarios que iba a ser capaz de soportar.

Cargas de estrés

La prueba empieza con 5 usuarios y consistía en que a la vez estos 5 usuarios mandaran el patrón 2.
Se repitió hasta 3 veces incrementado el número de usuario al doble cada vez y la aplicación seguía funcionando. Se apreciaba bastante la ralentización de la aplicación ya que tardaba mas en servir las peticiones. Esto ultimo se debe a que no se disponía de un servidor y el pc en el que se realizó la prueba carecia de grandes recursos.

Resultados

Tiempos y recursos (%procesador, memoria, y otros)
1ª Prueba: 5 Usuarios.
Tiempo en finalizar: 1 segundo.
Procesador 3%.
Memoria 73000 KB.

2ª Prueba: 10 Usuarios.
Tiempo en finalizar: 3 segundo.
Procesador 4%.

Memoria 73000 KB.

3ª Prueba: 20 Usuarios.
Tiempo en finalizar: 8 segundo.
Procesador 10%.
Memoria 75000 KB.

4ª Prueba: 40 Usuarios.
Tiempo en finalizar: 14 segundo.
Procesador 16%.
Memoria 77000 KB.

Conclusiones

Una buena elección del lenguaje seguido de un buen diseño de las interfaces han dado un buen resultado. En lo que a las pruebas se refiere, en mi opinión no son muy relevantes ya que la aplicación no
esta pensada para ser utilizada por mas de 1 o 2 usuarios a la vez.

Documentación

En esta sección vamos a redactar la documentación asociada a la aplicación.

Introducción

En esta sección se van a desarrollar los documentos necesarios para la correcta utilización de la aplicación.

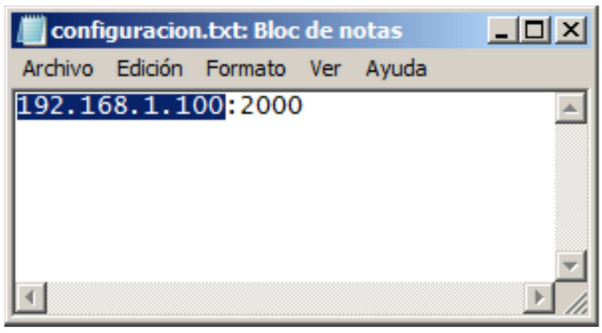
Usuario

Como ya hemos citado anteriormente, la aplicación dispone de un único usuario y se sobreentiende que este tiene unos conocimientos medios en el ámbito de entornos domóticos.
A continuación vamos a redactar el manual de usuario.

Manual de usuario

Configuración inicial

Lo primero que ha de realizar el usuario una vez adquirida la aplicación es montar un servidor web con soporte para PHP.
Una vez este en funcionamiento el servidor web, habrá que depositar la carpeta “Simulador de eventos domóticos” en el lugar determinado. Si estas usando apache el directorio es : ”\htdocs”.
El segundo paso es instalar la base de datos. Para ello debemos instalar el PostgreSQL y una vez instalado restaurar la base de datos mediante el archivo “domotica.backup”.
El tercer paso consiste en editar el fichero de configuración que encontraremos en “/config” con el nombre de “configuracion”. En el nos encontraremos lo siguiente:

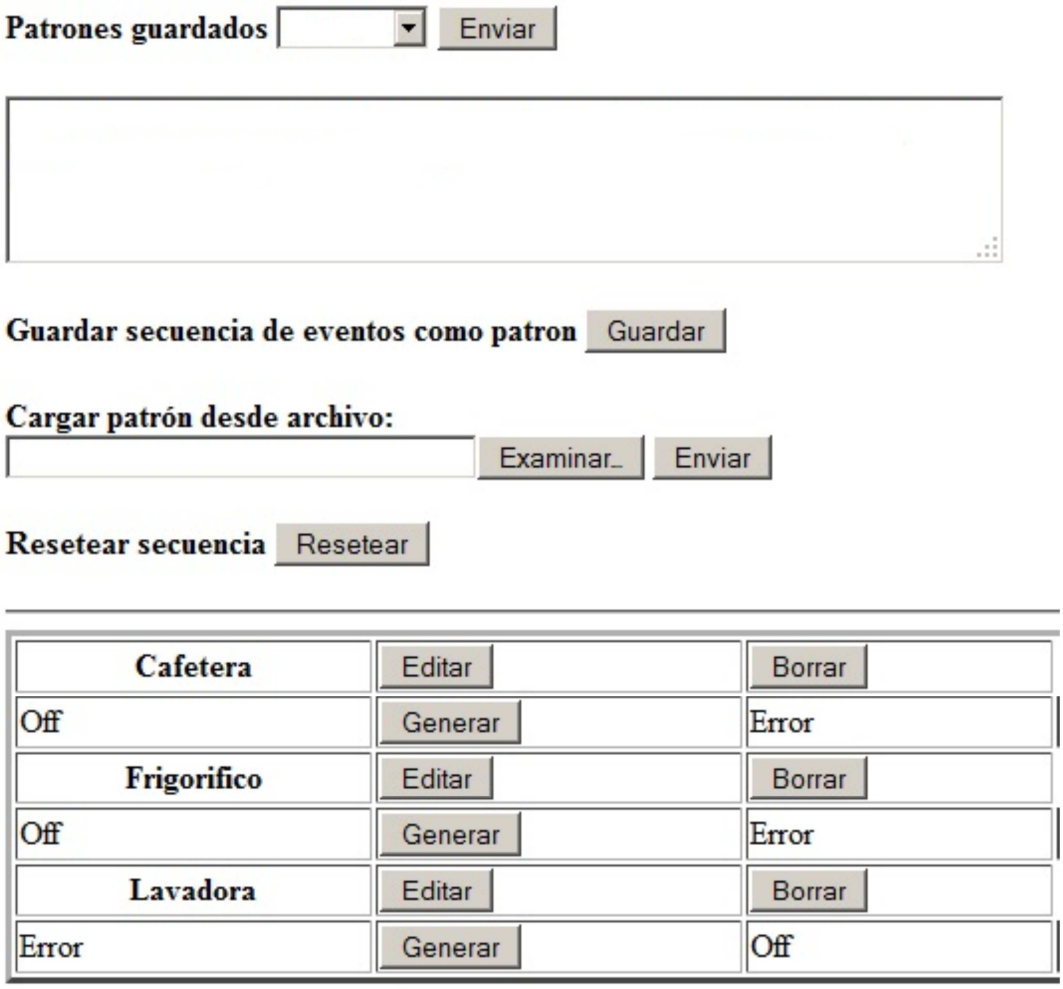


Archivo de configuración de la aplicación.

El campo sombreado corresponde a la dirección IP del pc al que van a ser enviados los mensajes. Esta dirección va seguida de ":" y a continuación el puerto tcp donde va a estar escuchando el receptor de los mensajes.
El usuario si el usuario no desea utilizar los parámetros de configuración que trae por defecto la aplicación, deberá cambiarlos respetando el formato inicial.
Llegados a este punto ya tenemos nuestro entorno perfectamente configurado y listo para lanzar el Simulador de Eventos Domóticos.

Primeros pasos

Una vez realizado con éxito lo anterior, el siguiente paso es ejecutar un navegador web y abrir el archivo "interfaz.php".
El navegador nos mostrará una interfaz como esta.

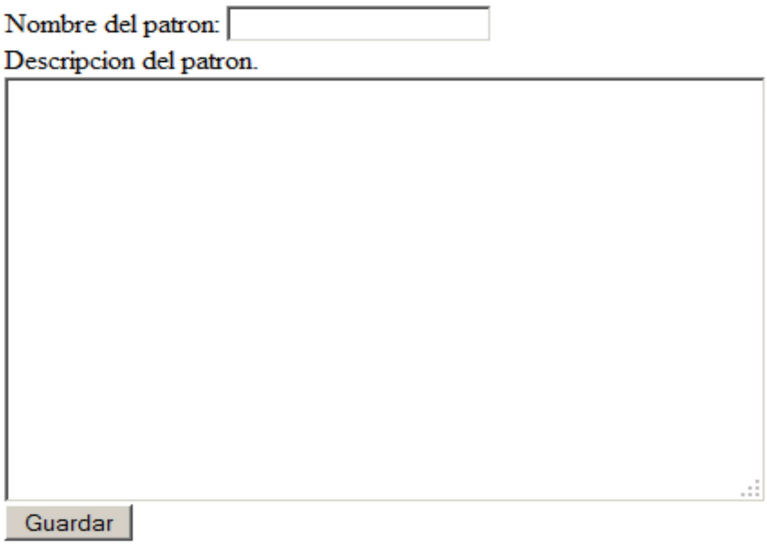


En esta interfaz como podemos ver nos muestra la mayoría de la funcionalidad de la aplicación. Por defecto viene con los datos que se muestran en la imagen a modo de ejemplo.
Si no desea estos datos, el usuario debería modificaros o bien insertar los nuevos datos.

Generando eventos, secuencias y patrones.

El usuario puede empezar a generar eventos pulsando el botón generar. Estos eventos generados se visualizarán como una secuencia en la caja de texto que hay debajo de la parte de patrones guardados.
En el momento se quiera guardar la secuencia existente como un nuevo patrón el usuario deberá pulsar el botón “Guardar” que hay detrás de la frase “Guardar secuencia de eventos como patrón”.

Una vez pulsado la aplicación nos mostrará esta interfaz.



Interfaz de inserción de un nuevo patrón

En esta interfaz el usuario introducirá un nombre al patrón y una descripción del mismo y le dará al botón “Guardar” para su posterior almacenamiento.
Una vez guardado, la aplicación nos mostrará la interfaz principal en la que podremos observar en la parte de “Patrones guardados” aparecerá el nuevo patrón insertado, ofreciéndonos la posibilidad de seleccionar y enviar el nuevo patrón.
Si por el contrario nos hemos equivocado en la generación de la secuencia podemos resetearla y volver a empezar con la generación.

Por ultimo, la aplicación nos ofrece la posibilidad de cargar un patrón desde un archivo de texto mediante el botón “Examinar” en la zona de “Cargar patrón desde archivo”.
Una vez pulsado este botón, la aplicación nos mostrará un cuadro de dialogo en el que podremos navegar por nuestro sistema de archivos hasta que encontremos el archivo que contiene el patrón deseado. Una vez lo seleccionemos y pulsemos el botón aceptar, la aplicación mostrará la ruta absoluta del archivo y ofreciendo la posibilidad de enviarlo.

Conclusiones

Una vez vista la documentación asociada a la aplicación vamos a proceder a comentar los objetivos alcanzados, las aportaciones y los requisitos futuros que se le podrían requerir a la aplicación.

Conclusiones

A continuación vamos a exponer las conclusiones a las que se ha llegado una vez desarrollada la aplicación.

Resumen

A la vista de lo redactado anteriormente, se podría decir que se ha conseguido la mayoría de los objetivos marcados. Se ha conseguido una aplicación capaz de emular a un entorno domótico real. Ademas, se ha logrado desarrollar la parte de los patrones, la cual nos permite simular comportamientos humanos, automatizar baterías de pruebas, etc.
También se ha conseguido la posibilidad de cargar estos patrones desde un archivo de texto, lo que facilita bastante sobre todo a la hora de hacer pruebas.
Todo ello integrado en una interfaz fácil de usar, con una curva de aprendizaje mínima y con la funcionalidad mas utilizada a la vista, evitando engorrosos procedimientos para su uso.
Se ha conseguido una aplicación eficaz y eficiente, a la par que fiable y estable por lo que el grado de satisfacción obtenido por el desarrollador es bastante alto.

Trabajo futuro

Como trabajo futuro se podría ampliar la interfaz existente, adornándola con CSS.
También se podría hacer una nueva interfaz en la que existiese un mapa del entorno domótico con todos los dispositivo existentes y en la que se pudiesen generar los eventos a golpe de ratón observando sus efectos de forma visual.
Otra posible ampliación seria adaptar la actual interfaz para los nuevos dispositivos móviles como son los smartphones o las tablets.

Referencias

Http://es.wikipedia.org/wiki/Domotica

Domótica: Un enfoque sociotécnico. Hugo Martín Domínguez y Fernando Sáez Vacas. Fundación Rogelio Segovia para el Desarrollo de las Telecomunicaciones. 2006.

http://www.domoticausuarios.es/un-hogar-seguro-con-una-buena-tecnologia/2692/

http://www.clikear.com/manuales

http://www.wordreference.com/es/

http://www.diccionarios.com/

http://www.google.es/

http://www.buscamultiple.com/

http://monstercrawler.com/

http://www.ixquick.com/esp/?&cat=web&query=

http://ardilladigital.com/

http://www.discapnet.es

http://www.indracompany.com

Clasificación y proyecto del edificio inteligente

Valencia : Universidad Politécnica de Valencia , 1995

Base de datos domotica

Ibañez Ibañez, Maria-Jose

Valencia : Universidad Politécnica de Valencia , 1998

Aplicación domótica basada en X10 [Recurso electrónico-CD-ROM]

Tárrega Artieda, Asís

Valencia : Universidad Politécnica de Valencia , 2009

DOMOTICA PARA VIVIENDAS Y EDIFICIOS - WERNER HARKE,

MARCOMBO, S.A., 2010