SAICOH

Sistema de Automatización Inteligente para Control de Oficinas y Hogares

Natalia Botto – Claudia Cardozo – Favio Sapelli – Fernando Severo

Setiembre 2008
Montevideo - Uruguay

Índice

Tabla de contenido

Datos del Proyecto

Nombre del Proyecto:

SAICOH

Integrantes del Grupo:

Nombre	C.I.	E-Mail
NATALIA BOTTO	**3.661.804-1**	**NATALIA_BOTTO@HOTMAIL.COM**
CLAUDIA CARDOZO	**4.135.622-4**	**CLAUCLERMONT@GMAIL.COM**
FAVIO SAPELLI	**2.972.381-9**	**FSAPELLI@GMAIL.COM**
FERNANDO SEVERO	**4.068.983-8**	fersevero@gmail.com

Cliente:

INTEGRADORES DE SERVICIOS

USUARIOS FINALES DE SISTEMAS DOMÓTICOS/INMÓTICOS

ARQUITECTOS Y PROMOTORES INMOBILIARIOS

Tutor:

ING. JOSÉ ACUÑA

Presentación del Problema

El objetivo de este capítulo es introducir las ideas que motivaron la realización de este proyecto, cuáles fueron los objetivos planteados inicialmente y el alcance esperado por el proyecto.

Motivación

Para poder describir el siguiente proyecto, debemos explicar primero en qué consiste un sistema domótico[1] y los beneficios que este supone para la vida cotidiana.

Se entiende por domótica al conjunto de sistemas capaces de automatizar una vivienda o edificio de oficinas, aportando servicios de gestión para el manejo de iluminación, climatización, seguridad, etc. con el fin de incrementar el bienestar y el confort de los usuarios. Usualmente, estos sistemas se integran por medio de redes cableadas o inalámbricas de alcance local o global lo que facilita el control y monitoreo desde dentro y fuera de la edificación.

Las aplicaciones de la domótica en el ámbito doméstico son bastas. No obstante, las áreas en las que se han dedicado mayores esfuerzos son las relativas a la seguridad, al incremento del confort y la gestión de la energía. En relación a las personas, es importante considerar el aumento de la calidad de vida que los sistemas domóticos pueden aportar, más aún en el caso de personas mayores o discapacitadas. En estas circunstancias algunas medidas de confort se convierten en necesidades vitales y los mecanismos de seguridad cobran un interés específico evidente.

Un sistema domótico dispone de una o varias redes de comunicación que permiten la interconexión de equipos a fin de obtener información sobre el entorno, para basándose en ésta, realizar acciones de control.

Un sistema tradicional de control consta de una unidad central que gestiona distintos elementos periféricos. El funcionamiento del sistema recae totalmente

[1] Ver Anexo I.

en la lógica de control programada en la central. En cambio, en los sistemas descentralizados, cada elemento posee capacidad de procesamiento propia lo que permite distribuir la carga sobre el sistema.

En la mayoría de los sistemas, los sensores transmiten las señales captadas del medio a una unidad de control central. Esta unidad se encarga de procesar los datos recibidos y de ser necesario realizar y/u ordenar acciones de control.

El correcto funcionamiento de estos sistemas se encuentra íntimamente relacionado con los procedimientos de transmisión de información que posibilitan el diálogo entre dichos periféricos y la unidad central. Usualmente, los terminales son equipos con capacidad de comunicación a través de una o varias interfaces.

Generalmente, en los esquemas centralizados, frente a la necesidad de controlar un entorno doméstico o de oficinas, se opta por topologías donde el sistema “orientado al confort” (luces, calefacción, cortinas, etc.) se separa del sistema “orientado a la seguridad” (incendio, gas, intrusión, etc.). Esta topología tiene la ventaja de que independiza los sistemas por lo que un fallo en uno no afecta el resto (efecto altamente deseado). Por otro lado el aumento tanto a nivel de cableado como sensores agrega costo extra a la solución haciéndola muchas veces irrealizable.

Imaginemos por un momento que podemos crear una red de control integrada donde cada dispositivo tiene, además de la capacidad de adquirir datos, la de procesarlos. De esta manera los nodos de control podrían interactuar entre ellos sin necesidad de una central, inclusive en situaciones donde la comunicación con la estación central de monitoreo u otros nodos fallara. Sin lugar a dudas, la optimización de recursos se maximiza en condiciones normales de operación, donde con una sola red de sensores podemos relevar la realidad y actuar sobre ella sin necesidad de depender de centrales de control. Esto favorece la disminución del tiempo de parada del sistema, abarata costos (tanto en sensores como en cableado) e incrementa la robustez.

La integración y optimización de todos los servicios se hace posible gracias a la utilización de un único medio de comunicaciones que los conecte a todos, un bus de campo.

Objetivos

El objetivo de este proyecto pretende realizar un sistema domótico que permita el monitoreo y el control remoto de una muestra descriptiva de dispositivos eléctricos y electrónicos que puedan encontrarse en el entorno de un hogar u oficina. A su vez, podrá integrar soluciones existentes de distintos proveedores en un único sistema optimizando la funcionalidad, el costo y el beneficio.

En particular, se busca ofrecer funciones, servicios e información que facilite la gestión y el mantenimiento de un hogar u oficina. Se buscará también aumentar la seguridad de los bienes y las personas, incrementar el confort, racionalizar el consumo de energía y el ahorro de tiempo mediante automatización de tareas. Otro punto a tener en cuenta es que tener un sistema con todos los dispositivos integrados, permite disminuir los costos en sistemas de control y monitoreo. Es de considerar que este tipo de sistemas permita incrementar la calidad de vida de usuarios con ciertas discapacidades.

Para considerar el proyecto exitoso existen ciertas prioridades a tener en cuenta, como ser que los fallos del sistema no pongan en riesgo a bienes y personas, poder realizar un adecuado control de usuario para las conexiones remotas así como también en el acceso físico al edificio, poder contar con una interfaz de usuario amigable, que el sistema sea robusto, tratar de minimizar el pago de licencias en software, que sea de fácil implantación y mantenimiento y la escalabilidad en la cantidad de dispositivos y funciones de control.

Tampoco se pueden dejar de considerar cuales serán los criterios de éxito y aceptación. En este punto tenemos que considerar una serie de pruebas sobre el sistema que, de realizarse de forma exitosa, concluirán exitosamente el proyecto. Entre las pruebas se considerará el control de una red de dispositivos LonWorks mediante un cliente local conectado a la red y también sobre un cliente que acceda de forma remota a través de Internet. Se probará la obtención de información vía Internet y su posterior incorporación al sistema, en particular se obtendrá el estado del tiempo. Otra prueba a considerar es la comunicación del sistema con el exterior, esto se hará a través del envío de SMS y E-Mail. Una parte importante del funcionamiento del sistema, es su

robustez. Se realizarán una cierta cantidad de pruebas que nos permitan asegurarla. Se consideran importantes para este punto los fallos de alimentación, la caída del sistema informático de monitoreo (PC), falla en alguno de los nodos LonWorks y falla en las comunicaciones. No podemos dejar de considerar que para llegar a un resultado exitoso, debemos probar la estabilidad del sistema en el tiempo.

Alcance

Se planteó el desafío de realizar una aplicación que permita realizar el monitoreo y control tanto local como remoto de una red de dispositivos inteligentes que puedan encontrarse en una edificación.

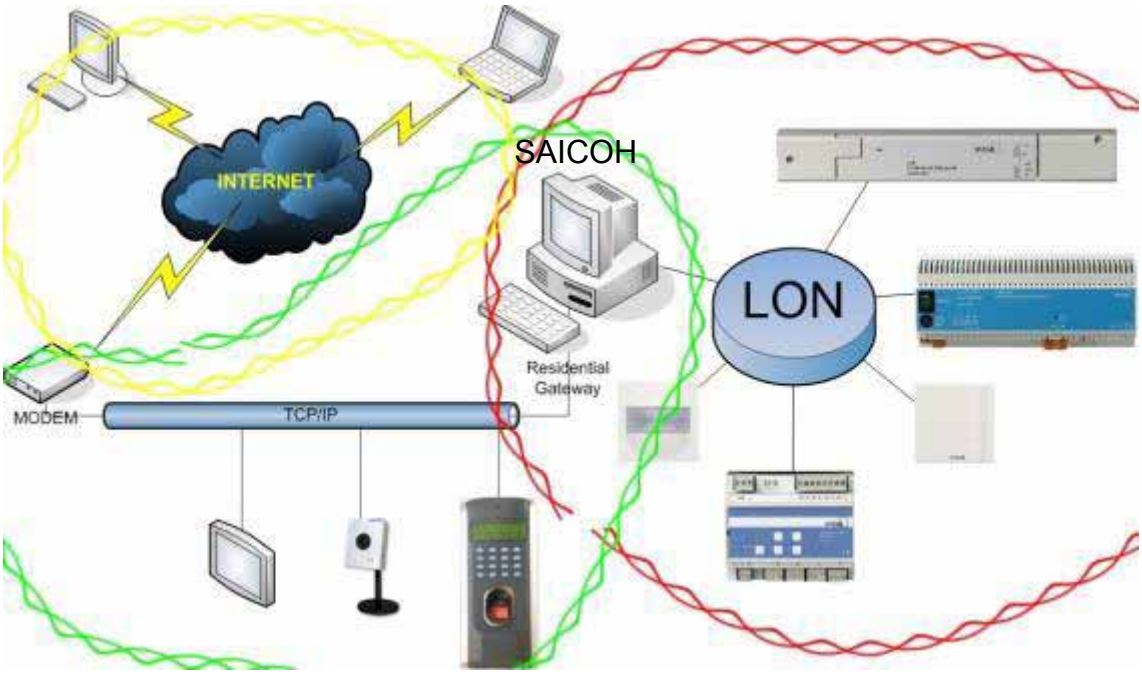
El fin de la aplicación es integrar los dispositivos para unificar y facilitar el manejo de iluminación, climatización, seguridad, así como otras áreas de interés en instalaciones domésticas o de oficinas. La integración provee también a la aplicación de una cantidad múltiple de fuentes de información facilitando la operativa global del sistema.

Descripción del sistema

En este capítulo explicaremos en qué consiste Saicoh e introduciremos algunos conceptos utilizados durante el desarrollo del mismo.

Saicoh

Saicoh es una pasarela residencial, un software que hace de interfaz entre los dispositivos que se encuentran en el hogar, el usuario y a su vez brinda posibilidades de interconexión con internet u otros medios. En este caso se utiliza el hardware domótico LonWorks.



En el funcionamiento del sistema, este posee la posibilidad, frente a una serie de acontecimientos predeterminados de realizar avisos a los usuarios a través del envío de correo electrónico o de mensajería instantánea a los celulares seleccionados para este fin. Definimos que estos acontecimientos son aquellos donde la integridad física de las personas puede verse afectada, como ser frente a avisos de alarmas o sensores.

Otra posibilidad es que se pueda programar un evento a ser ejecutado en un instante de tiempo predeterminado o la de definir inicio y fin de una serie de eventos.

Concepto de Política

Llamamos políticas a los procedimientos que se ejecutarán automáticamente frente a la ocurrencia de determinados incidentes.

Dentro de las acciones a tomar frente a la activación de alguna de estas políticas, se encuentra el envío de SMS y mails a usuarios, la activación de alarmas o realizar avisos a diferentes autoridades, según corresponda. Todas las acciones son logueadas en un archivo del sistema y presentadas en la interfaz grafica a su vez.

Las políticas implementadas por el sistema son: Falla de Alimentación, Incendio, Perdida de Gas, Inundación, Meteorológica, Pánico, Robo, Seguridad de Personas, y AC. Las mismas se describirán con más detalle en los aspectos previos del desarrollo de software.

Concepto de Escena

Una escena es una modificación de uno o más de un elemento del sistema. Esta modificación puede tener fecha y hora de inicio o fin y el valor a tomar el elemento. Este será digital (on/off) o un valor analógico dependiendo del elemento.

Las escenas son básicamente la programación del estado que deben tener los dispositivos en el hogar en un momento determinado en el tiempo. Esta programación puede ser para un momento fijo o repetitivo, como por ejemplo todas las mañanas de lunes a viernes encender una cafetera.

Concepto de Película

Una película es la ejecución en una secuencia predeterminada de escenas.

La película permite seleccionar escenas predefinidas o establecer directamente el estado de un dispositivo en el tiempo.

Un ejemplo claro de la utilización de políticas es el establecimiento de Simulación de Presencia en el Hogar. Otro es la utilización del sistema en salones de fiestas donde cada película establece la secuencia de eventos en una fiesta temática.

Fundamentos Teóricos

Antes de comenzar a analizar el sistema implementado, se deben estudiar algunos fundamentos teóricos necesarios para su correcta comprensión.

Estos conceptos van a permitir entender cuál es el producto esperado, la tecnología domótica utilizada y los fundamentos de la arquitectura del software.

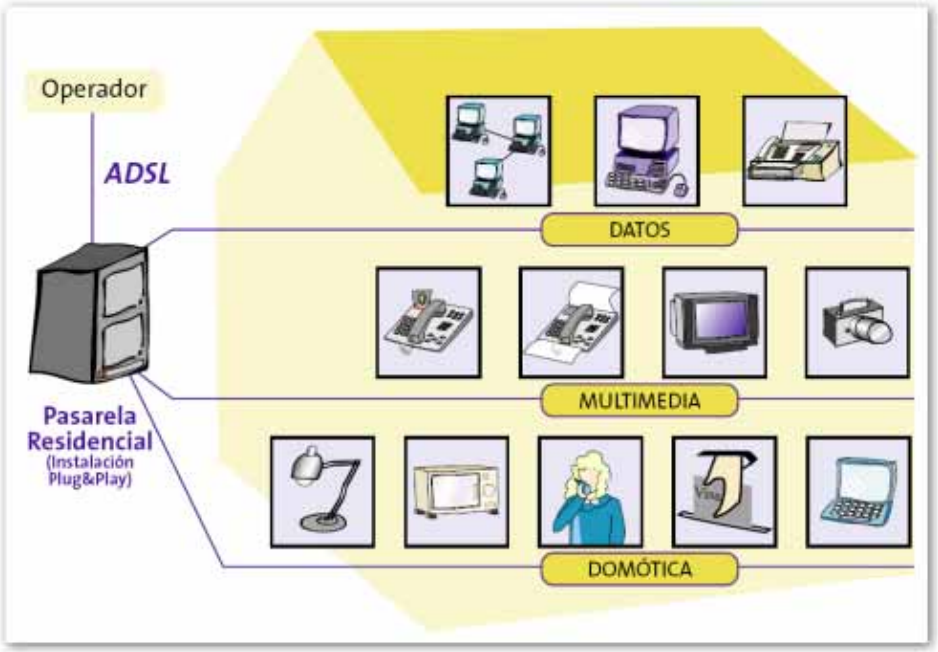
Domótica y Pasarela Residencial

La vigésima edición del diccionario de la Real Academia Española define "domótica" como el "conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda".

La domótica es la automatización y control de aparatos y sistemas eléctricos y electrotécnicos en la vivienda. Su principal objetivo es aumentar el confort, ahorrar energía y mejorar la seguridad.

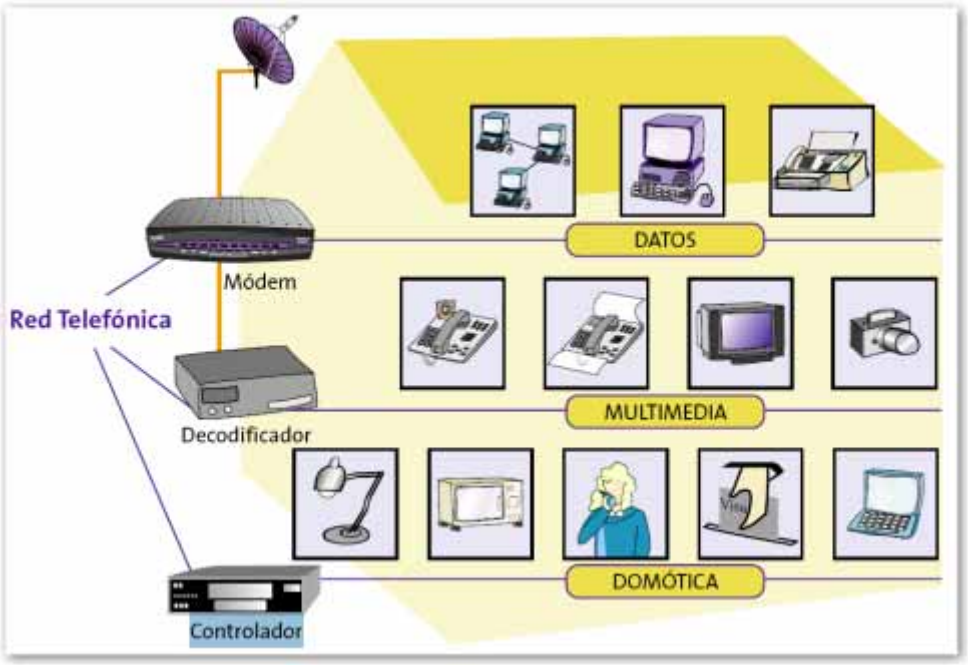
Dentro del contexto de la domótica y la digitalización del hogar se encuentran las pasarelas residenciales. Una Pasarela Residencial[2] es un dispositivo que conecta las infraestructuras de telecomunicaciones (datos, control, automatización, etc.) del hogar digital a una red pública de datos, como por ejemplo Internet.

Una pasarela residencial es el producto que permitirá la conectividad total de los hogares con el mundo exterior. Y el software desarrollado por este proyecto consiste en una aproximación de una pasarela residencial.



Siendo estrictos, el sistema desarrollado no es exactamente una pasarela residencial, sino un controlador domótico, ya que no integra las redes de datos multimedia y domótica. Esto lo podemos observar en la figura siguiente.

[2] Ver Anexo I.



LonWorks

LonWorks[3] es una plataforma especialmente creada para satisfacer los requerimientos de performance, confiabilidad, flexibilidad, costo, instalación y mantenimiento necesarios para redes dedicadas a aplicaciones de control.

LonWorks es un protocolo domótico que permite cubrir desde el nivel físico al nivel de aplicación en casi la totalidad de los proyectos de redes de control. Está diseñada para cubrir los requisitos de la mayoría de las aplicaciones de control y ha tenido gran éxito en instalaciones de control y monitoreo para edificios comerciales, de viviendas e industrias.

Cualquier dispositivo LonWorks, también llamado nodo, está basado en un microcontrolador particular llamado Neuron Chip. Cada Neuron Chip tiene un identificador único (Neuron ID), lo que permite distinguir cada uno de los dispositivos. A su vez posee un modelo de comunicaciones independiente del medio físico, y el firmware que implementa el protocolo LonTalk proporciona servicios de transporte y ruteo extremo a extremo.

Los dispositivos se comunican entre sí por medio de datagramas que contienen la dirección de destino, información para el ruteo, datos de control, incluyen datos de la aplicación de usuario, y CRC como código detector de errores.

[3] Ver Anexo I.

Aplicaciones Multicapas.

La arquitectura multicapa se utiliza por los beneficios de separar los componentes, formando su suma el todo. Y permitiendo que la integridad de cada una quede independiente de las otras.

Una aplicación típica como la nuestra está compuesta de tres componentes.

La capa de presentación maneja la vista de interfaces con el sistema operativo, o como en nuestro caso la presentación de la aplicación Web.

La capa de negocios es quien determina como responde la lógica de la aplicación. Aquí se encuentra la inteligencia del sistema.

En la capa persistencia es el almacén de la información, esta interactúa y maneja la base de datos del sistema y vuelve estos accesibles a la lógica de negocios y a la vista.

La decisión de utilizar una arquitectura de capas permite que no todos los integrantes del equipo de desarrollo deban dominar todas las tecnologías, esto permite especializarse y obtener un producto de mayor calidad.

Model-View-Controler (MVC)

El Model-View-Controler (MVC) es un patrón de arquitectura muy difundido en el uso de aplicaciones web. Fue aportado por SmallTalk y tiene tres piezas claves que se reparten la responsabilidad de la aplicación, el modelo, la vista y el controlador.

El modelo (model), es el responsable de la lógica de negocio del sistema. Este puede estar dado por una única capa interactuando directamente con la base de datos o por más de una como es en nuestro caso.

La vista (view), es la presentación del negocio en la interfaz grafica. En nuestro caso está compuesta por las paginas jsp (Java Server Pages) y jspf (Java Server Pages Fragments). La mayoría de estos contenidos son generados dinámicamente en el servidor de aplicaciones.

El controlador (controler), es quien maneja el flujo de control entre la lógica y la vista, la navegabilidad y el estado de los usuarios. Al utilizar Java Server Faces (JSF) como Frameworks para implementar el patrón este se ve transparente al desarrollo y todo es configurable mediante archivos XML. Este componente es central en el patrón y responsable de interceptar las peticiones http del cliente, traducir la petición en la operación de negocio especifica, determinar la siguiente vista para el cliente y retornar el control al cliente.

El patrón MVC es el seguido por la aplicación desarrollada y fue utilizado JSF como framework para implementarlo.

Mapeo objeto – relacional (ORM)

El mapeo objeto-relacional (más conocido por su nombre en inglés, Object-Relational Mapping, o sus siglas O/RM, ORM, y O/R Mapping) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional. En la práctica esto crea una base de datos orientada a objetos virtual, por sobre la base de datos relacional. Esto posibilita el uso de las características propias de la orientación a objetos (básicamente herencia y polimorfismo). Hay paquetes comerciales y de uso libre disponibles que desarrollan el mapeo relacional de objetos.

En la programación orientada a objetos, las tareas de manejo de datos son implementadas generalmente por la manipulación de objetos, los cuales son casi siempre valores no escalares. Sin embargo, la gran mayoría de los motores de base de datos, como los productos basados en el lenguaje SQL, solamente puede almacenar y manipular valores escalares como enteros y cadenas, organizados en tablas. Por lo que se deben convertir los valores de los objetos en grupos de valores simples para almacenarlos en la base de datos (y volverlos a convertir luego de recuperarlos de la base de datos), o solo usar valores escalares simples en el programa. El mapeo relacional de objetos es utilizado para implementar esta primera aproximación.

El punto de inflexión del problema reside en traducir estos objetos a formas en las cuales puedan ser almacenadas en la base de datos, y los cuáles pueden ser recuperados más tarde fácilmente, mientras se preserven las propiedades de los objetos y sus relaciones; estos objetos se dice entonces que son persistentes. Para solucionar estos problemas es que existen frameworks como Hibernate, TopLink, etc.

Especificación de Políticas

Por política nos referimos a la serie de eventos que implican la realización de una serie de acciones por el sistema. En todas ellas puede estar en pelígro la integridad física de las personas o la violación de la propiedad privada.

A continuación explicaremos todas las políticas que son utilizadas, indicando cuales son las condiciones por las que se realiza el disparo de las mismas así como las acciones a tomar en cada una de ellas.

Política de intrusión

Esta política pretende detectar cualquier tipo de intrusiones que ocurra en la vivienda o el edificio, incluso también en habitaciones interiores, de personas no habilitadas para ingresar a dicho lugar. Por no habilitación nos referimos a que el usuario no existe en la base de datos del sistema, a un usuario externo que intentó ingresar a la propiedad. El fin de esta política es aumentar el nivel de seguridad de los usuarios, ya que pueden estar más tranquilos al ir al trabajo o incluso al salir de vacaciones, ya que frente a cualquier tipo de intrusión serán avisados.

El disparo de esta política se podrá detectar a través de la activación de una alarma o por medio de los sensores de presencia o sensores de apertura, que puedan estar en puertas o ventanas. Otra forma de activación es a través del botón de pánico, este puede ser activado por un usuario frente a la presencia de personas ajenas, o frente a ataques personales a usuarios, ya sea dentro o en las inmediaciones cercanas a la vivienda.

Las acciones a tomar serán primeramente será el aviso a las autoridades correspondientes, luego también el envió de SMS y e-mail a una serie de usuarios previamente cargados, la activación de un juego de luces que desoriente al intruso y la activación de cámaras de la zona afectada (si se contara con ellas) para poder detectar la mayor información posible. Todo esto se registrará en el log de eventos del sistema, para la futura consulta.

Nombre	Política de intrusión
Qué lo dispara	Alarma
	Sensores de apertura
	Sensores de presencia
	Botón de pánico (intrusión)
Acciones a tomar	Aviso a autoridades
	Juego de luces
	Envío SMS
	Envío mails
	Log

Política de incendio

Este tipo de política es otro ejemplo de acciones a tomar que permiten aumentar la seguridad de los hogares o edificios con estos sistemas domóticos, ya que el sistema alertará a los usuarios incluso antes de que estos se den por aludidos del incidente y evitando que estos avancen a una gravedad importante. La forma de detectar un incendio o posible incendio será a través de la activación de una serie de sensores, como ser de humo o temperatura. Las primeras acciones a tomar, como medida de prevención, serán cortar el suministro eléctrico de la zona afectada y el cierre de llaves de gas de la casa, para evitar propagaciones indeseables. También se avisaran a los usuarios a través de una alarma o aviso hablado del incidente y se habilitaran los ingresos a dicho lugar para permitir la libre circulación de personas, ya sea para la salida de los moradores como para el ingreso de bomberos o personal especializado. Se procederá también a dar avisos a los usuarios por sms o mail y se registrará el incidente en el log de eventos.

Nombre	Política incendio
Qué lo dispara	Sensor de temperatura
	Sensor de humo
Acciones a tomar	Aviso autoridades
	Corte de suministro eléctrico
	Cierre de llaves de gas
	Activar alarma incendios
	Habilitar ingreso a lugar del incidente
	Envío SMS
	Envío mails
	Log

Política de inundación

La política de inundación permite detectar tanto roturas en caños de la casa como posibles olvidos de canillas abiertas y etc. La forma de detección es a través de sensores especialmente diseñados para este fin los que se ubican a una cierta altura del suelo. Cuando dicho sensor se activa se pasa a cortar el suministro principal, para evitar que dicha inundación se siga extendiendo, y también el corte del suministro eléctrico en la zona donde se detectó dicha falla. Se procederá a avisar al grupo de usuarios seleccionados del incidente por sms o mail y el registro de dicho evento en el log del sistema.

Nombre	Política inundación
Qué lo dispara	Sensor de inundación
Acciones a tomar	Corte de suministro principal de agua
	Corte de suministro eléctrico en zona afectada
	Envío SMS
	Envío mails
	Log

Política de fuga de gas

Frente a una fuga de gas, los sensores especialmente diseñados y colocados en el sistema, detectaran esta falla y procederán a realizar las acciones programadas para evitar accidentes mayores. Como primer paso se procederá a cortar todo el suministro eléctrico, manteniendo si fuera posible solamente el sistema de ventilación. Se habilitará un sistema de ventilación hasta que los sensores de gas tengan medidas normales. Como viene ocurriendo en todas las políticas ya descriptas, se procederá a avisar a la serie de usuarios programados por los medios acordados del incidente, así como la escritura del evento en el log.

Nombre	Política fuga de gas
Qué lo dispara	Sensor de gas
Acciones a tomar	Corte de suministro eléctrico en zona afectada
	Ventilación
	Envío SMS
	Envío mails
	Log

Política falla eléctrica

Este tipo de política pretende realizar las acciones pre programadas frente al salto en llaves termo magnéticas. Las acciones a tomar son principalmente el aviso a los usuarios configurados del incidente por los medios configurados y el registro del incidente en el log de eventos.

Nombre	Política falla eléctrica
Qué lo dispara	Salto de llave termo magnética
Acciones a tomar	Envío SMS
	Envío mails
	Log

Política seguridad personas

Esta política se activa principalmente frente a casos de necesidad de asistencia personal, sobretodo en caso de existir personas de tercera edad o con problemas médicos. El sistema se encargará de dar el aviso al sistema de emergencia configurado así como a los usuarios, generalmente los familiares, del incidente ocurrido. También se procurará habilitar el recorrido de ingreso a la habitación donde se detectó la alarma, facilitando el acceso hacia el usuario en problemas. Como todas las acciones del sistema, se adjuntará en el log de eventos.

Nombre	Política seguridad personas
Qué lo dispara	Botón de pánico (seguridad personas)
Acciones a tomar	Aviso servicio de emergencia
	Habilitar ingreso a zona del incidente
	Envío SMS
	Envío mails
	Log

Política alerta meteorológica

Esta política pretende alertar frente a posibles alertas meteorológicas detectadas a través del sistema de monitoreo de AccuWeather. Se realizará un aviso en pantalla, así como los avisos usuales a usuarios a través de mail y SMS. También se actuará sobre ventanas o toldos que se encuentren abiertos, en caso que se requiera. Como todas las acciones, se escribirá en el log de eventos del sistema.

Nombre	Política alerta metereológica
Qué lo dispara	Aviso por medio del servicio web de AccuWeather
Acciones a tomar	Aviso en pantalla
	Cierre ventanas, etc
	Envío SMS
	Envío mails
	Log

Dispositivos de Campo

En el presente capítulo se busca presentar el hardware disponible para la implementación de la red de control de dispositivos. Esta red integra el hardware de prueba utilizado, este podrá ser controlado y monitoreado desde la aplicación SAICOH. Se comentarán también las cuestiones referentes a las principales decisiones de diseño adoptadas.

Hardware de Prueba

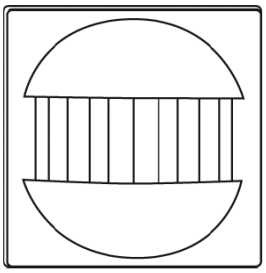
Sensor de temperatura para interior - AP RTS -10 (63325-246)

Este sensor cumple la función de relevar temperatura ambiente y poner a disposición de la red la variable. Puede participar activamente en el proceso de control generando señales de activación para otros dispositivos. A modo de ejemplo, pueden configurársele parámetros como para ejecutar un control on/off. En este caso el algoritmo de control se realiza en el propio sensor obteniendo como resultado la señal de activación del dispositivo controlado.

Descripción técnica:

- Rango de medida: -20..+50ºC
- Grado de protección: IP20
- Dimensiones: 73 x 73 x 24 mm (H x W x D)
- Reporta el valor medido a la red LON según el perfil LonMark “Temperature Sensor (1040)”
- Capaz de realizar análisis de medidas en base a parámetros configurables según perfil LonMark “Temperature Switch Level (3200)”

Sensor de movimiento para interior - LON 42015-538



Este sensor cumple principalmente dos funciones, detectar movimiento y niveles de iluminación. La detección de movimiento posibilita la generación de acciones según una persona entre o salga del recinto. La medición de la cantidad de luz en un ambiente puede utilizarse en conjunto con otros módulos para regular iluminación, etc.

Descripción técnica:

- Uso interior
- Detección de movimiento dentro de ángulo horizontal de 180 grados
- Detección de movimiento en un perímetro de hasta 8 metros
- Umbrales ajustables para control de iluminación por nivel de luminosidad
- Traduce los movimientos detectados según los perfiles LonMark "Occupancy Sensor (1060)" y "Occupancy Controller (3071)" en mensajes LON para ejecución de control de iluminación dependiente del desplazamiento.

Dimmer - AP DIM 500- AB (37222-095)

Este módulo cumple la función de variar la forma de onda de alimentación de las luminarias conectadas haciendo posible la regulación de la cantidad de luz que estas generan. Puede utilizarse también como control on/off de luminarias. En caso de poseer sensores de luminosidad (luxómetros) conectados a la red puede utilizar esta información para ejecutar algoritmos de control capaces de mantener niveles determinados de luz ambiente.

Descripción técnica:

- Dimerización por control de fase (atrazo de fase) para lámparas incandescentes, halógenas HV y transformadores eléctricos
- Carga manejable: 20..500VA

- Dos entradas (230V) para operación mediante botones
- Protección contra cortocircuitos y sobrecarga
- Grado de protección: IP20
- Dimensiones: 40 x 280 x 28 mm (H x W x D)
- Incluye timers, control de prioridades y comportamiento configurable al reinicio.
- Provee control de iluminación y escenas de según el perfil LonMark "Lamp Actuator (3040)", "Constant Ligh Controller (3250)", "Scene Controller (3251)" y "Switch (3200)".

LON I/Os - REG-S 4W 4DI 24 V (35236-150)

Este módulo con entradas y salidas digitales posee cuatro entradas digitales de 24VAC las cuales pueden ser utilizafas para censar señales y cuatro salidas digitales por relé capaces de manipular cargas. Se alimenta directamente de la red de suministro local. De agregarse un sensor de proximidad al sistema el controlador del módulo puede implementar funciones avanzadas de iluminación con manejo de cargas por prioridades incorporando pulsadores, sensores, timers, etc.

<u>Descripción técnica:</u>

- 4 entradas (24 VAC) y 4 salidas (10 A Max)
- Conmutación independiente para 4 grupos de carga
- Voltaje de entradas: aprox. 24 VAC
- Corriente máxima de entrada: aprox. 10mA
- Posibilidad de operación manual por botones
- Status de entradas y salidas por LEDs
- Detección de caída de alimentación
- Voltaje de alimentación: AC 230V
- Ancho: aprox. 105mm

- Funciones de operación según perfiles LonMark “Lamp actuator (3040)” que incluye timers, operaciones lógicas, control de prioridades y estado configurable de las salidas al reinicio de la alimentación.
- Controladores de escena (“Scene Controllers (3251)”) con manejo simultaneo de grupos de entradas y salidas.
- Flancos detectables según perfiles Lonmark “Switch (3200)”, “Scene panel (3250)” o “Occupancy Sensor (1060)”

LON Power Supply - LPS-W (11031-004)

Esta Módulo es una fuente de alimentación que inyecta en la red LonWorks un voltaje de continua de 42.8V. Varios componentes de la red se alimentan de este voltaje para su operación por ejemplo, sensores. Es posible utilizar el mismo par tranzado para alimentación y comunicaciones ya que estas se realizan por codificación diferencial.

Descripción técnica:

- Fuente de alimentación de los dispositivos que contienen LPTs
- Corriente de salida máx.: 1.5 A
- Terminación de bus ajustable para topologías libres o en línea
- Alimentación: AC 230 V
- Ancho del dispositivo: 215mm aprox.

Interfase LonWorks – USB

Permite la utilización de un PC como estación de monitoreo y control de los elementos de la red. Si se usa sin la PC el sistema continua operando según los últimos parámetros cargados.

Red

Para realizar el diseño de la red[4] se hizo necesario tomar decisiones al respecto de la definición de la topología a utilizar, variables de red en juego, lazos de control a implementar, etc.

A continuación se presentarán las principales elecciones realizadas junto con sus justificaciones.

LonWorks soporta múltiples canales de comunicación. Un canal es el medio físico por el cual se intercambian datos pudiendo contener hasta 32.285 nodos.

Habiendo realizado un estudio previo de las diferentes topologías soportadas por cada conjunto de transductores, y considerando las restricciones impuestas por los equipos de prueba ya presentados definimos la topología que utilizaremos en la red Lonworks a implementar.

La tabla siguiente presenta los módulos Lonworks junto con las principales características en cuanto a la conformación de la red física.

Módulo	Función	Xver	Alimentación
37222-095	Dimmer	FTT	230VAC
35236-150	IOs Digitales	FTT	230VAC
63325-246	Sensor Temperatura	LPT	42.8VDC
42315-383	Sensor Movimiento	LPT	42.8VDC
11031-004	Fuente DC	-	230VDC
-	Interfaz LON-USB	FFT	5VDC

Como se observa en la tabla, módulos son alimentados externamente (230VAC) mientras otros se alimentan de la propia red Lonworks (42.8VDC).

[4] Ver Anexo II.

Esto es posible ya que la modulación de los datos sobre el bus de campo es compatible con la existencia de voltajes en modo común altos. La mayoría de las veces este es el sistema preferido al momento de alimentar sensores minimizando el cableado necesario y evitando la utilización de baterías (con la consecuente disminución del mantenimiento). Finalmente, la interfaz que permite integrar un PC a la red Lonworks se alimenta a través del propio bus USB. De la tabla se concluye también, que todos los tranceptores son compatibles entre sí lo que hace innecesaria la utilización de routers.

La red que implementaremos será de topología libre cableada mediante par trenzado de cobre acorde con la categoría 5e.

La gran flexibilidad que posee esta topología es un factor determinante en su elección ya que es usual en este tipo de instalaciones que se sucedan cambios a posteriori de la puesta en marcha. Con una topología de sencilla modificación se minimizan y abaratan los problemas a futuro.

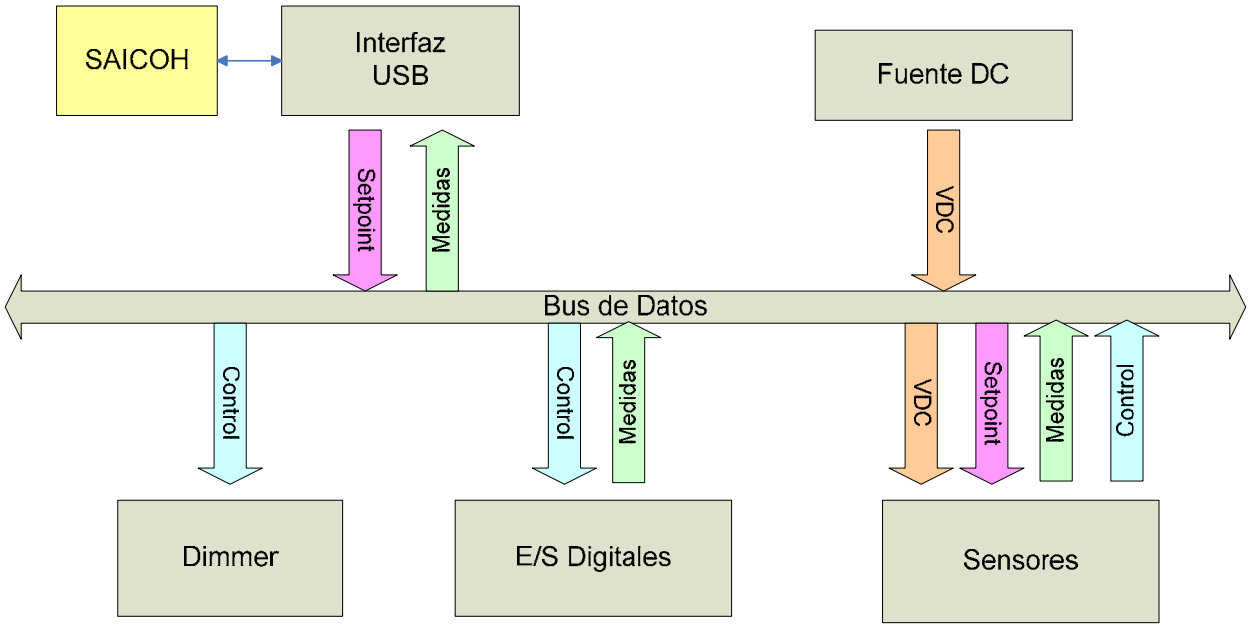
La utilización de Link Power Transceivers (LPTs) nos obliga a revisar el consumo de los sensores con el fin de comprobar que la fuente se encuentre dimensionada acorde a las necesidades. De las hojas técnicas de los equipos obtenemos los consumos de potencia máximos de cada uno así como la potencia de la fuente.

Equipo	Consumo Aprox.
Sensor Movimiento	285mW
Sensor Temperatura	50mW
Fuente	60W

Como era de suponer, la potencia de la fuente es más que suficiente para alimentar ambos sensores.

En conclusión, decidida la topología de red a utilizar como libre resta ubicar los equipos en sus lugares respectivos y trazar el tendido que optimice las distancias de cableado.

Conceptualmente, el sistema global que obtendremos es el que se presenta en el diagrama a continuación donde se identifican claramente los módulos con sus interacciones.

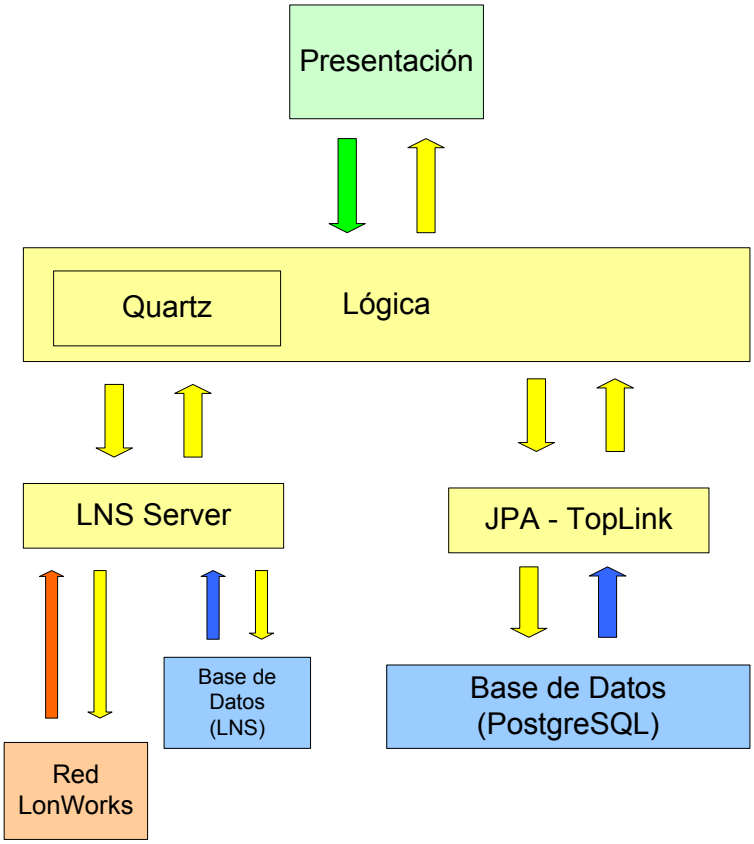


Tecnologías de Software

Previo a abocarnos al análisis del software desarrollado, plantearemos las consideraciones pertinentes a la elección del lenguaje.

En este capítulo hablaremos de las decisiones que tomamos a la hora de elegir un software para desarrollar, que fuera un sistema multicapa, cual fue la elección del motor de base de datos y la forma de comunicación entre el sistema y la base, y al final se comenta el API que utilizamos para realizar la comunicación del sistema con el hardware (sensores, actuadores, etc.).

Cabe destacar que en el momento de realizar todas estas decisiones, nos planteamos asumir el desafío de utilizar tecnologías de última generación.



Elección del Lenguaje y frameworks.

Por tratarse de un proyecto académico y plantearnos como uno de los objetivos obtener un sistema robusto escalable y estable se decidió trabajar con tecnologías de punta en el desarrollo de software en todas las áreas críticas de la aplicación, persistencia de datos, separación de la capa grafica, manejo de eventos temporales y logueos del sistema.

La primera decisión tomada fue la plataforma de desarrollo, se opto por la utilización de Java en la plataforma Java Enterprise Edition (JEE[5]). La idea de desarrollar en Java surge principalmente por razones prácticas. Al ser parte formal de la formación académica, todos los integrantes del grupo estábamos familiarizados con el lenguaje. JEE permite desarrollar aplicaciones empresariales distribuidas, basadas en componentes mediante el uso de un modelo de aplicación multicapa. En general se suele aplicar en sistemas de tres capas o más.

Este fue elegido por ser un lenguaje que permite desarrollar aplicaciones portables, robustas, escalables y seguras. Por lo comentado anteriormente, como nuestro equipo poseía conocimientos básicos de Java, existían varias posibilidades para realizar en forma óptima la separación en capas.

Para trabajar con JEE es necesario un Servidor de Aplicación, un dispositivo de software que proporciona servicios de aplicación, gestionando la mayor parte de las funciones de lógica de negocios y de accesos a los datos de la aplicación. Incluyendo este los servicios de Servidor Web y de Mapeo Objeto Relacional.

Entre las opciones analizadas optamos por la utilización de Glassfish. Esta elección se apoyo en que es la implementación de referencia para la generación del estándar JEE, tiene licencias OpenSource (CDDL y GPL) y GlassFish cuenta con plugins para Eclipse y con excelente soporte en NetBeans.

Seleccionada la plataforma de programación, definido el servidor de aplicaciones y tomada la decisión de implementar el desarrollo en capas se

[5] Ver Anexo III.

definieron los frameworks sobre los que se sostendría la arquitectura. Se establece como primer paso que se implementara el patrón de diseño MVC para separar la interfaz grafica[6] de la lógica de control y se utilizara un mapeador Objeto – Relacional para persistir los objetos en la base de datos relacional.

Para implementar el patrón MVC optamos por Java Server Faces (JSF) debido a que al no ser necesario conocerlo en detalle para utilizarlo resulto simple la realización de pruebas con el mismo previo a la decisión. La otra opción analizada STRUTS posee más documentación pero es necesario avanzar más en su curva de aprendizaje antes de comenzar a utilizarlo en producción.

Base de datos

Para trabajar con la base de datos[7] se utilizo un mapeador Objeto – Relacional. La elegida es el API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar de Enterprise Java Beans (EJB3), Java Persistance Api (JPA). El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos y permitir usar objetos regulares.

JPA se ocupa de la forma en que los datos de la base de datos se representan en objetos de java (entidades de persistencia), o sea, estandariza el mapeo objeto relacional. Cuando hablamos de entidades de persistencia nos referimos a clases de java que típicamente representan tablas de la base de datos relacional. Las instancias de las entidades corresponden a las filas de las tablas. Las entidades tienen típicamente relaciones con otras entidades, esto también se representa en esta estructura de clases.

Para realizar este mapeo de clases de java en tablas de la base de datos se utilizan frameworks ORM (Object-Relational Mapping). Existen muchas opciones, como ser Hibernate, TopLink, etc. En nuestro caso se eligió TopLink.

---

[6] Ver Anexo IV.

[7] Ver Anexo V.

TopLink es un package ORM open source de java, que provee un framework flexible y poderoso para realizar el mapeo de objetos java a una base de datos relacional o para convertir objetos java en documentos XML.

Fue desarrollado por Oracle, y en el 2006 el código fuente del producto TopLink y recursos de desarrollo al proyecto open source de Sun Microsystems java.net Glassfish. Este proyecto fue llamado TopLink Essentials y es la referencia de implementación de Java EE EJB 3.0 JPA. Actualmente TopLink es la opción por defecto de Glassfish al integrar JPA a las aplicaciones.

Como última decisión a tomar para la base de datos, aunque cronológicamente fue la primera, hubo que decidir que motor de base de datos se iba a utilizar. Para este proyecto se eligió PostgreSQL, que es un motor de base de datos relacional open source. Ya tiene más de 15 años de desarrollo activo y una arquitectura que se ha ganado la reputación de ser confiable, mantener la integridad y exactitud de los datos. Corre tanto en windows como linux, tiene todas las prestaciones necesarias de una base de datos y a su vez contiene una interfaz nativa de programación para C/C++, Java, .Net, Perl, Phyto, Ruby, y otros. También tiene un lenguaje propio de programación llamado PL/PgSQL, el cual es muy parecido al lenguaje PL/SQL que utiliza Oracle.

Como muchos otros proyectos open source, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

LNSHMI

Hay una serie de funcionalidades criticas del sistema que son la comunicación del Software con el Hardware de monitoreo y control, la realización de logueos del sistema y el manejo de eventos temporales que son realizados con APIs o frameworks específicos para dichas tareas. De esta forma se consigue un sistema mucho más robusto que implementando de cero estas funciones.

LNSHMI[8] (LonWork Network Server Human Machine Interface) fue utilizado para conectar la Red LonWorks con el software desarrollado en Java. El funcionamiento se basa en proveer una interfaz para la conexión de la aplicación Java con el LNS Server. Éste es quién se encarga del manejo de las comunicaciones con la red LonWorks gerenciando los dispositivos conectados y manteniendo actualizado los valores de las variables en uso. La realización de los logueos del sistema son mediante el uso del framework Apache Log4j, desarrollado con el fin de facilitar la inclusión controlada de logueos en aplicaciones. Su velocidad y flexibilidad se basan en mantener las sentencias de logueo empaquetadas de manera de que se pueda habilitar el logueo en tiempo de ejecución sin necesidad de modificar el archivo compilado. Todo esto puede realizarse a un costo de procesamiento razonable

Para el manejo de los eventos temporales Quartz, un framework open source pensado e implementado con el fin de simplificar y potenciar el uso de tareas planificadas en cualquier aplicación JEE (Java Enterprise Edition) o JSE (Java Standard Edition).

La base tecnológica del sistema es la expuesta anteriormente. Durante la implementación del mismo se utilizaron también API específicas, como por ejemplo JavaMail y SAX entre otras. Durante la descripción de los paquetes será presentada la utilización de todos estos componentes.

[8] Ver Anexo I.

Comunicaciones

Para analizar las necesidades y tipo de comunicación del sistema desarrollado, es necesario tener una clara definición de que se entiende por comunicaciones.

En este contexto definimos comunicaciones como la interacción del sistema con los dispositivos que no manejen el protocolo LonTalk y las necesidades de comunicación con el exterior por medios que no sean la interfaz gráfica.

A continuación se presentara una breve introducción a las necesidades y tipos de comunicación con referencia a las tecnologías que se utilizan en el proyecto.

Interacción con huellas digitales

Se contó con un dispositivo detector de huellas digitales que no utiliza el protocolo de comunicaciones LonTalk. El dispositivo elegido para el proyecto es de la marca Granding, modelo BioSH-F7.

El detector se conecta al PC a través de Ethernet, y éste a su vez escribe la información en una base de datos. Originalmente se esperaba acceder a dicha información a través del software desarrollado, y que luego la lógica de control del sistema se ocupara de habilitar o no la apertura del cerrojo, según los permisos que contenga el usuario.

Durante el trabajo con el detector nos encontramos con una serie de inconvenientes. El software de gestión del detector dice que soporta cualquier motor de base de datos, pero en la realidad es diferente, ya que espera una base Access que trae por defecto, por lo que no pudimos comunicarnos con PostgreSQL, que es el motor que elegimos utilizar para nuestro proyecto. Luego decidimos comunicarnos directamente Access, en este punto nos encontramos con que JDBC (el api de java para manejo de base de datos) no es compatible con ODBC (el conector de Microsoft, que es el que utiliza Access) y la manera de comunicarlos es por medio de puentes (bridges). Buscando en Internet los que encontramos disponibles eran pagos.

Luego de discutirlo con el tutor del proyecto decidimos que, debido que el detector no es un componente central, decidimos no utilizarlo. Este fue incluido originalmente con el objetivo de conectar a nuestro sistema un dispositivo no LonWorks.

Respecto a la inclusión de un detector de huellas en el hogar, recientes encuestas en España muestran que esta funcionalidad no es de mayor interés para los usuarios finales.

Lectura de temperatura XML (Extensible Markup Language)

Una de las funcionalidades del sistema desarrollado es la posibilidad de que el usuario acceda a datos del estado del tiempo, por ejemplo: temperatura, viento, humedad relativa, etc.

Los datos presentados al usuario son obtenidos mediante información suministrada por AccuWeather.com. Esta se nos brinda en forma gratuita durante un plazo de dos años por tratarse de un proyecto académico.

La información que nos interesa es actualizada en los servidores cada 4 horas y accedemos a ella a través de un archivo XML.

Una vez obtenido el XML, los datos de este se extraen con la utilización del API de java SAX (Simple API for XML) y se devuelven a la interfaz gráfica en el formato del dato correspondiente.

Se realizo la elección de SAX frente a las demás herramientas ya que esta se distingue por ser una herramienta rápida y eficiente. SAX requiere menor memoria de procesamiento que DOM (Document Object Model), dado que a diferencia de DOM, esta no construye una representación interna del archivo XML, SAX simplemente envía la información a la aplicación a medida que la lee, por esta razón no es posible retornar a una posición previa o saltar en la lectura a otra etiqueta del archivo. Este tipo de parseadores funciona bien cuando simplemente se necesita realizar la lectura sobre un archivo XML. Además SAX se utiliza para realizar el procesamiento de archivos donde no existe dependencia entre las etiquetas, el cual es nuestro caso. En el caso de existir una dependencia del estado DOM hubiese sido la herramienta elegida.

Envío de mails y sms

Mediante JavaMail se realizaron una serie de clases encargadas de enviar un e-mail o sms, cuando las políticas de acción automática del sistema lo requieren.

Para realizar los envíos, el asunto y el contenido son definidos por la política que genera el evento. El usuario, contraseña, e-mail y puerto con el que el sistema envía el mensaje son obtenidos de la base de datos del sistema en las tablas de configuración del software. Para saber a quienes se debe enviar el aviso, en las tablas de usuarios, entre los permisos de estos, figura una propiedad que determina si hay que avisarles en caso de dispararse una política.

Arquitectura de Software

En la búsqueda de un sistema flexible y con posibilidades de incorporar nuevas funcionalidades se separa la lógica, el mapeo de la base de datos y la vinculación con la interfaz gráfica en proyectos individuales. Estos proyectos son divididos en paquetes que reúnen los conceptos básicos del software.

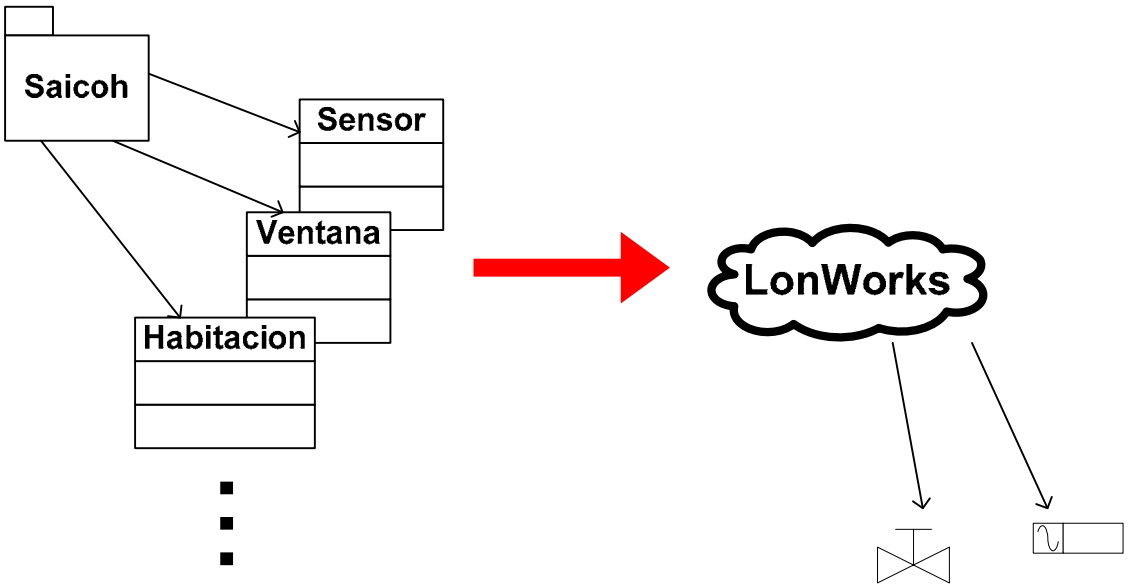
A continuación se presentará como está estructurado el sistema, a nivel de paquetes.

Arquitectura de capas y paquetes

Cada uno de los proyectos realizados contempla un paquete principal. Estos son *saicohGUI*, *saicoh* y *saicohDB.*

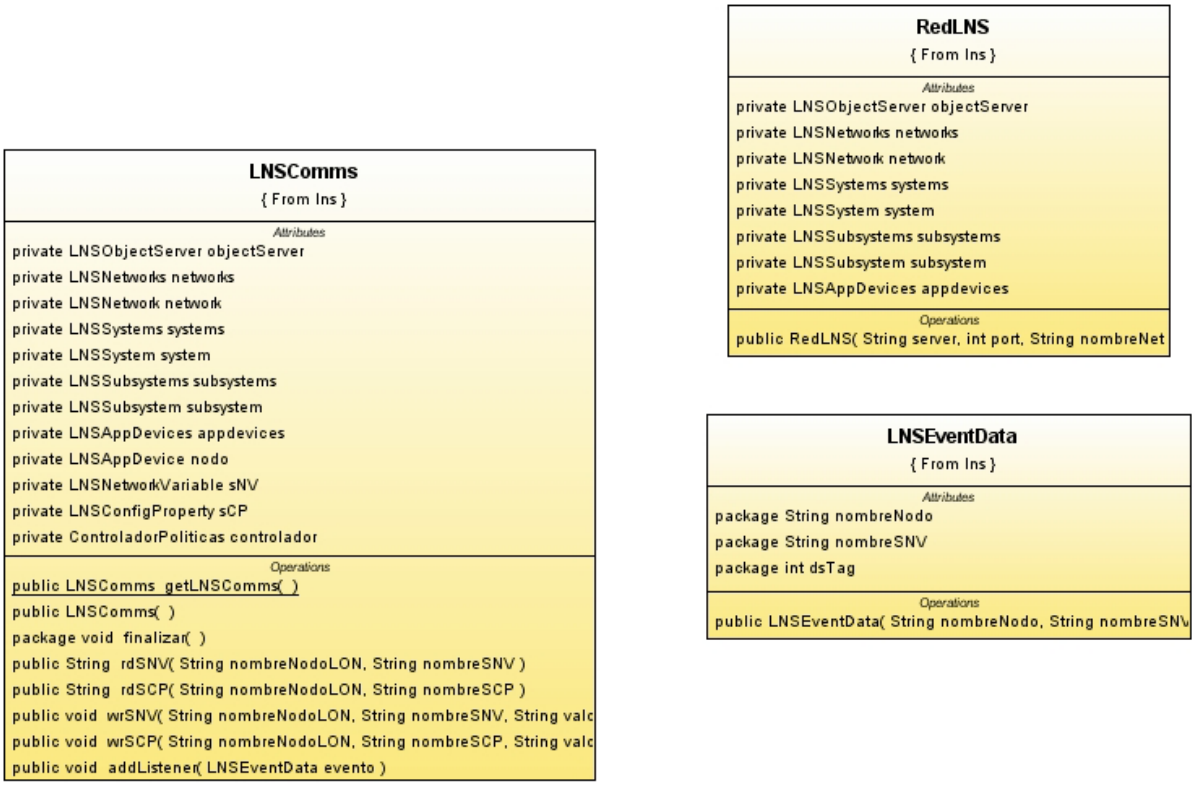
Paquete saicoh

El paquete *saicoh* agrupa la capa de negocio, contiene toda la lógica del sistema.



Este paquete está formado por un grupo de paquetes que explicaremos a continuación.

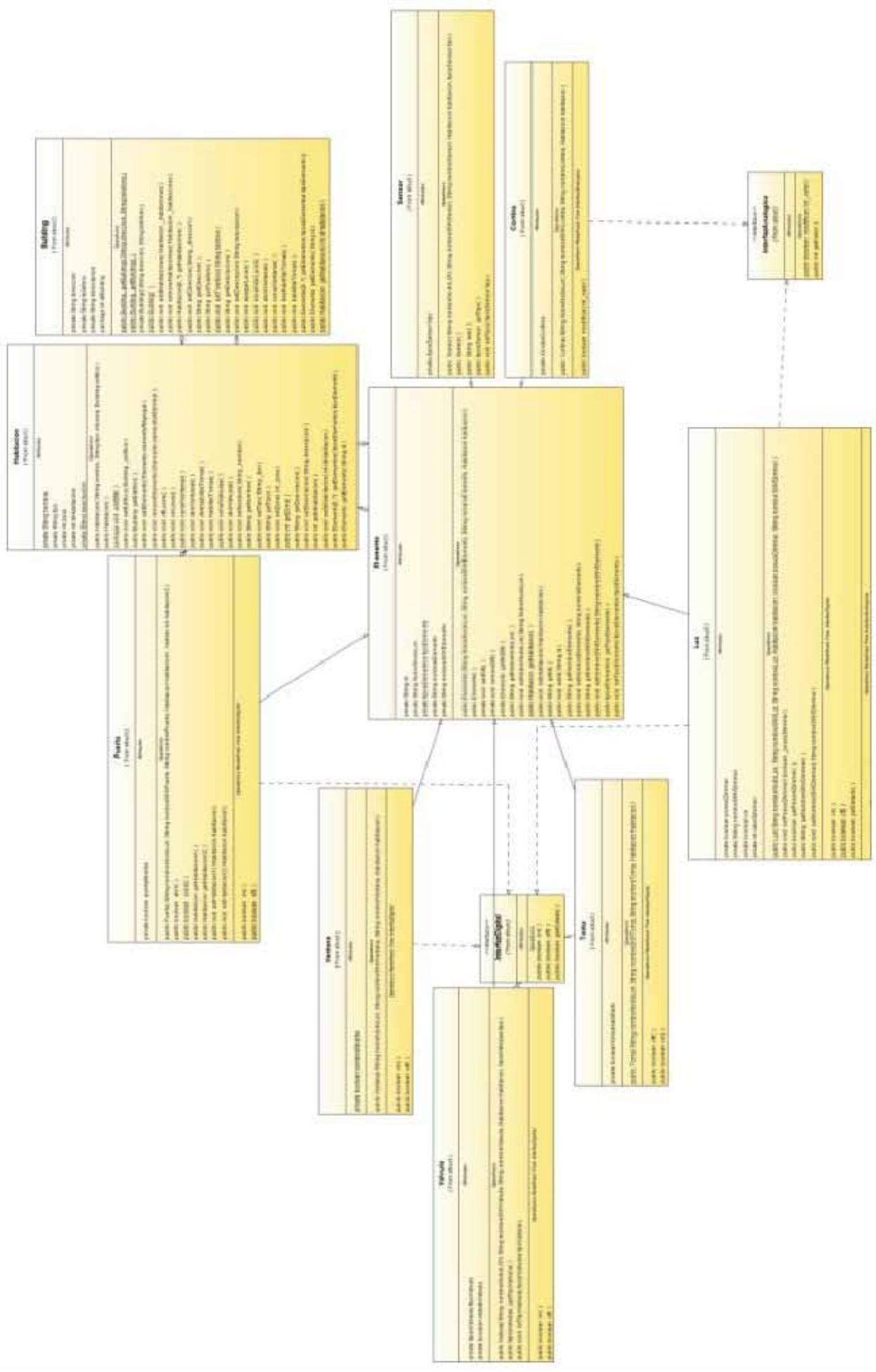
- El paquete *net* es quien interactúa directamente con la red LonWorks y el protocolo Lontalk manejando la totalidad de la adquisición y escritura de variables en la red de control. Ver diagrama de clases a continuación.



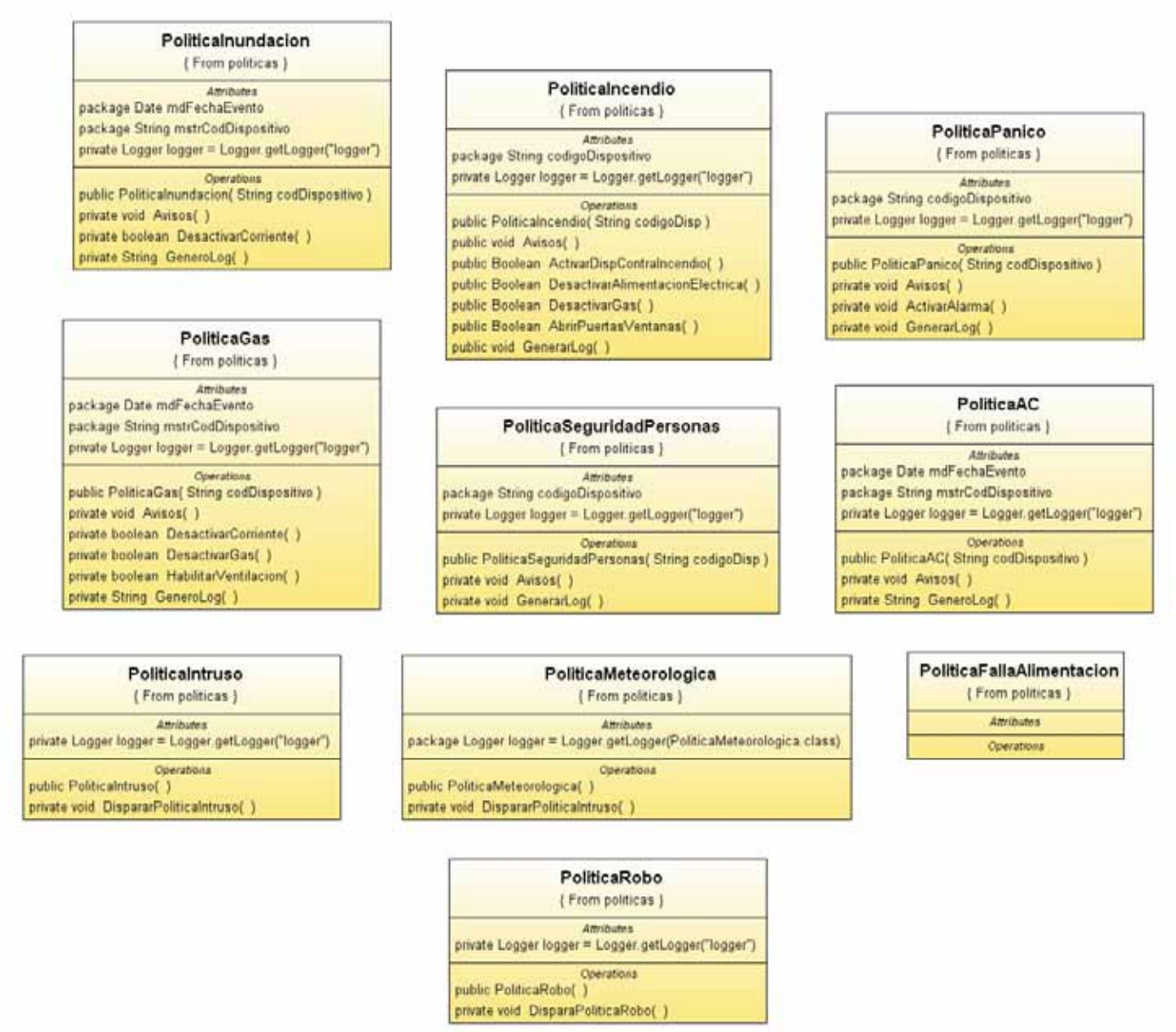
- En *comms.out* encontramos las comunicaciones salientes del sistema.

- Dentro del paquete soporte se incluyen subpaquetes que brindan herramientas accesorias al núcleo del sistema. Estos son: soporte.levantaEstructuraXML (para levantar el sistema en el inicio), soporte.meteorologia (maneja la lectura metereológica) y soporte.bateria (gestiona la batería del notebook).

- El paquete *struct* es el que representa en la lógica toda la estructura física del hogar digital y los dispositivos (sensores, actuadores, etc) que se encuentren en él. Ver diagrama de clases a continuación.



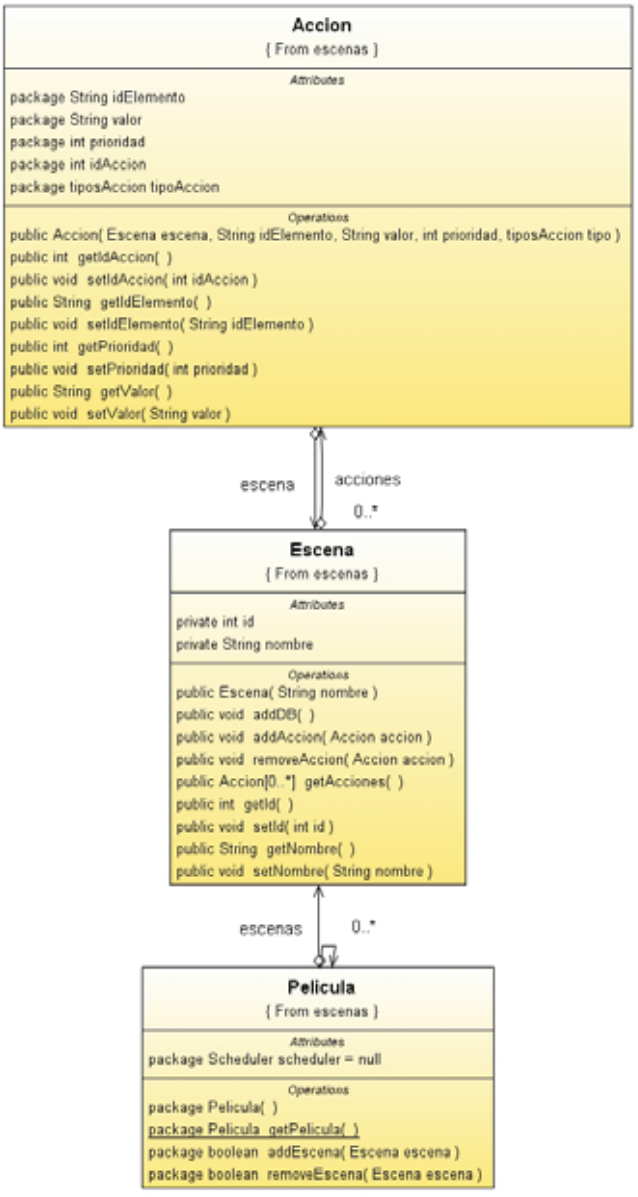
- El paquete *control* está compuesto por una clase para realizar el control por histéresis y un subpaquete control.políticas donde son manejadas e implementadas todas las políticas. Ver diagrama de clases a continuación.



- Para levantar la estructura del building en el arranque inicial del sistema es que fué concevido y es utilizado soporte.levantaEstructuraXML.

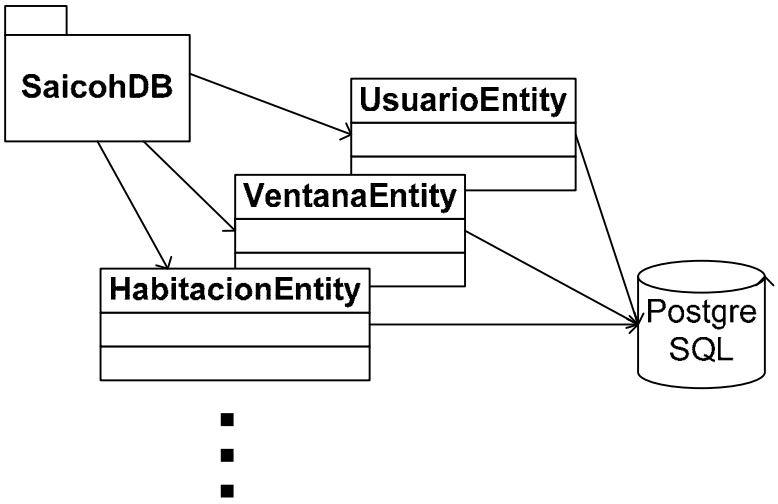
- La adquisición de los datos meterológicos se realiza mediante soporte.meteorologia. Estos son presentados en la interfaz gráfica y utilizados por la política alerta meteorológica.
- La utilización de soporte.bateria se da en la PolíticaAC, este provee actualmente datos de la carga de la batería de un notebook con sistema operativo windows XP. En caso de que el software fuera llevado a producción es necesario que brinde servicios compatibles con otros sistemas operativos.

- El paquete escenas y el paquete escenas.jobs se utilizan para gestionar las escenas del sistema. Ver diagrama a continuación.



Paquete saicohDB

El paquete s*aicohDB* maneja la capa de persistencia del sistema, lo que realiza toda la comunicación con la base de datos.

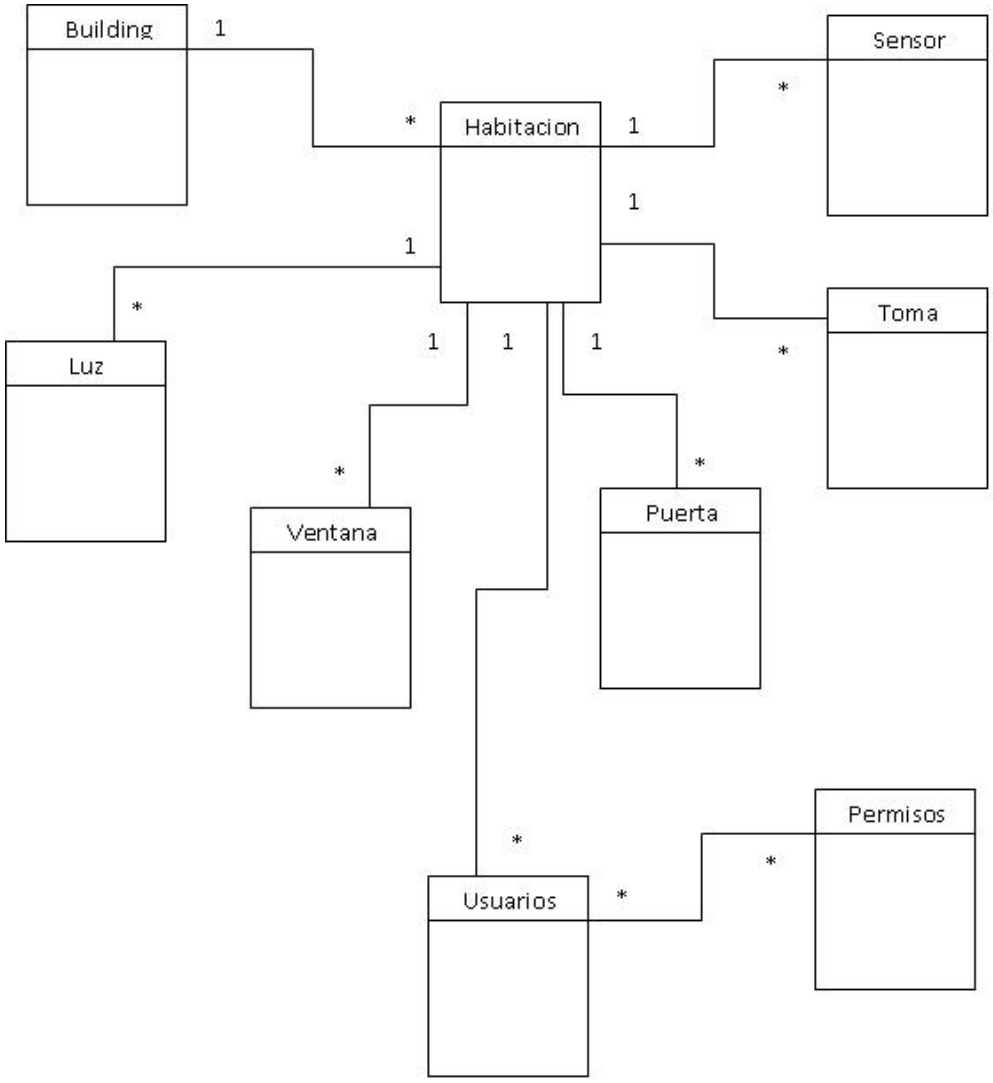


En el diagrama se puede ver un ejemplo de algunas de las clases incluidas en el paquete, las cuales son las que se comunican con la base de datos en PostgreSQL.

Las clases que se comunican directamente con la base de datos son las llamadas entidades, la cuales utilizan, como se explicó anteriormente, persistencia para realizar el mapeo. Existe también un archivo XML de configuración que es el encargado de definir el mapeo de las tablas de la base de datos con las entidades.

Por último hay una gama de clases que contienen todos los métodos que precisan la lógica de negocio y la lógica de vista.

La estructura de base de datos se puede observar en el siguiente diagrama MER, Modelo Entidad Relacional, diagrama que representa las relaciones entre las tablas de la base de datos.



Paquete saicohGUI

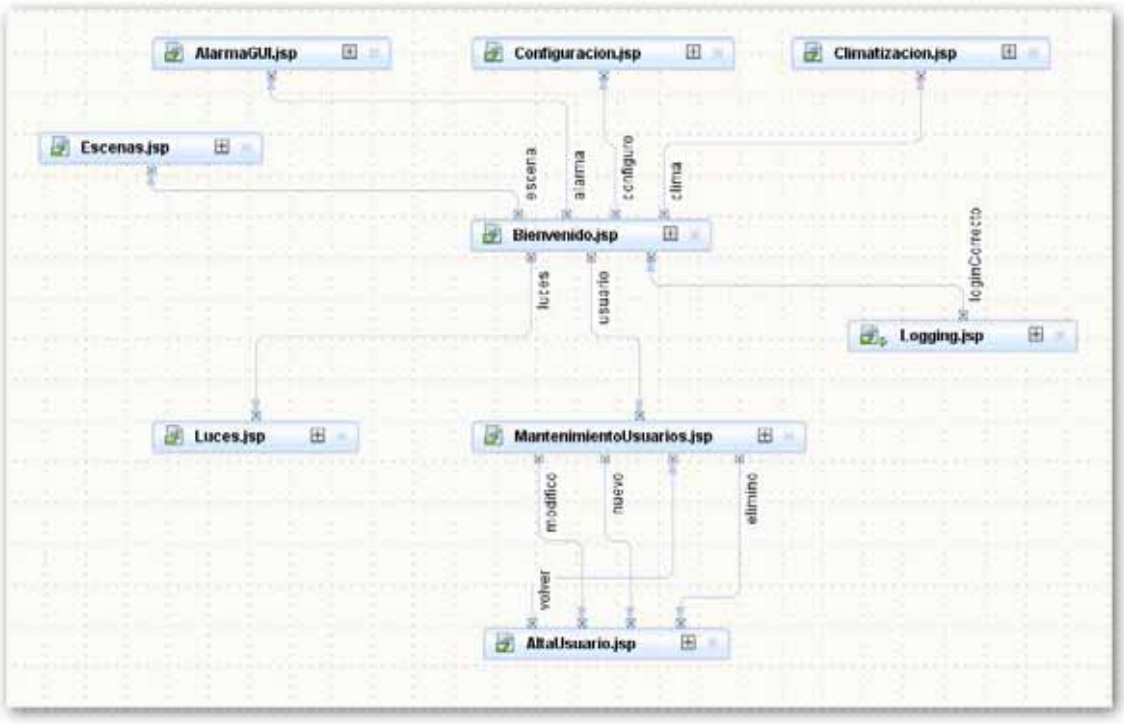
El paquete s*aicohGUI* maneja la capa de vista del sistema, la interfaz con el usuario. El mismo se basa en JavaServerFaces, para implementar el patrón de diseño MVC que ya se comentó anteriormente.

El SessionBean1 es el objeto encargado de persistir datos durante toda la sesión web. Maneja el ingreso de usuarios al sistema, a través de la conexión a la base de datos para verificar los datos, y obtiene de la lógica de control todos los datos necesarios para ser mostrados en pantalla, y también aquellos que se modifican a través de la interfaz.

El RequestBean1 es un bean de request, una consulta web, o sea, un objeto que vive durante la consulta web. Este objeto es el encargado de manejar los datos para aquellas páginas que pueden tener varios estados, por ejemplo, para el alta y la modificación de usuarios se accede a la misma página pero según el modo con el que se ingresa, son los controles que están habilitados o los datos que están visibles.

Cada página tiene asociado un bean de request que es el encargado de interactuar con el SessionBean1 y el RequestBean1, para escribir la página.

Un archivo XML es el encargado de definir las interacciones entre las páginas. A continuación presentamos un diagrama que indica la navegación entre páginas.



Funcionamiento Dinámico

A continuación será presentado el análisis del comportamiento del sistema frente a circunstancias típicas, que se pueden encontrar a lo largo del funcionamiento del sistema.

Creación de un Building

Para poner el software en funcionamiento es necesario disponer de la topología del edificio. Estos datos son necesarios para levantar la estructura del mismo al realizar el arranque inicial del sistema.

¿Qué es un Building?

En la lógica el Building mapea la estructura del edificio a ser automatizado y los estados de los elementos que lo componen.

Los edificios poseen similares características, todos contienen habitaciones y las mismas pueden contener otros elementos en diferentes cantidades, como pueden ser puertas, luces, sensores, ventanas, tomas, cortinas, y válvulas entre otros. No sería viable tener que ingresar esta información al sistema reiteradas veces ya que es una información fija y necesariamente conocida previo a la ejecución inicial del sistema. Además, en tiempo real se almacenará para cada elemento su estado, y por esto debe existir el elemento sin volver a ser creado. Por esta razón la creación del Building implementa el patrón singleton, para que nuestra instancia de building sea única, evitando que se hagan más inicializaciones y permitiendo que sea accesible a todo el sistema.

El software está diseñado para que el administrador del sistema ingrese el esqueleto del building mediante el uso de un archivo XML. En este archivo se especificará la cantidad de habitaciones y elementos que posee cada una. Pasando como atributos de la etiqueta los parámetros necesarios para crear el objeto y ser introducidos en el Building.

Es necesario respetar ciertas condiciones para la correcta creación del Building. La inclusión de los elementos debe darse dentro de las etiquetas correspondientes a la habitación que pertenezcan, aunque el orden de precedencia en el cual se incluyan es irrelevante ya que van a pasar a almacenarse en una lista de elementos de la misma especie dentro de esa habitación y no existe orden lógico que respetar.

Decidimos utilizar XML porque aporta muchas ventajas y principalmente una de las ventajas prácticas en el campo práctico que nos interesa es el uso de XML en la comunicación de datos. Si la información se transfiere en XML, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esa información y trabajar con ella.

Las etiquetas que utilizamos para nuestro sistema son:

< Building > : referencia al building , con los siguientes atributos:

- Nombre: nombre con el cual se identifica el building.
- Dirección: dirección física del building.
- Teléfono: teléfono del building.

<Habitacion>: referencia a la habitación, debe estar adentro del Building. Con los siguientes atributos:

- Nombre: nombre con el cual se identifica la habitación.
- Tipo: tipo de habitación.
- Zona: zona a la que pertenece la habitación.
- Edificio: edificio en el cual se encuentra la habitación.

<Luz>: referencia a la luz, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLon: nombre del nodo que controla la luz.
- NombreSNVLuz: SVN que maneja la luz.
- NombreLuz: nombre con el cual se identifica la luz.
- Habitacion: habitación a la cual pertenece la puerta.
- PoseeDimmer: especifica si la luz está conectada a un dimmer.

- NombreSNVDimmer: nombre de la variable LON que maneja el dimmer.

<Puerta>: referencia a la puerta, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLon: nombre del nodo que controla la puerta.
- NombreSNVPuerta: SVN que maneja la puerta.
- NombrePuerta: nombre con el cual se identifica la puerta.
- Habitacion1: habitación 1 a la cual pertenece la puerta.
- Habitacion2: habitación 2 a la cual pertenece la puerta.

<Sensor>: referencia al sensor, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLON: nombre del nodo que controla el sensor.
- NombreSNVSensor: SVN que maneja el sensor.
- NombreSensor: nombre con el cual se identifica el sensor y se despliega en la GUI.
- Habitacion: habitación a la cual pertenece el sensor.
- Tipo: tipo de sensor.

<Toma>: referencia a la toma, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLon: nombre del nodo que controla la toma.
- NombreSNVToma: SVN que maneja la toma.
- NombreToma: nombre con el cual se identifica la toma.
- Habitacion: habitación a la cual pertenece la toma.

<Valvula>: referencia a la válvula, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLon: nombre del nodo que controla la válvula.
- NombreSNVValvula: SVN que maneja la válvula.

- NombreLlave: nombre con el cual se identifica la llave de la válvula.
- Habitacion: habitación a la cual pertenece la válvula.
- Tipo: tipo de válvula.

<Ventana>: referencia a la ventana, debe de estar dentro del Building. Con los siguientes atributos:

- NombreNodoLon: nombre del nodo que controla la ventana.
- NombreSNVVentana: SVN que maneja la ventana.
- NombreVentana: nombre con el cual se identifica la llave de la ventana y se despliega en la GUI.
- Habitacion: habitación a la cual pertenece la ventana.

Usuarios

Para realizar el ingreso al sistema se requiere la autenticación de los usuarios. Siendo estos previamente ingresados al sistema, por lo que se contempló todo el mantenimiento de usuarios.

Por esta razón es necesario agregar al software los usuarios que interactúan con el sistema, así como los datos y permisos de los mismos. Tanto el alta, modificación y baja de usuarios se realiza mediante la interfaz gráfica, donde allí se pueden ingresar y/o modificar los datos obligatorios que deben poseer los usuarios, o darse de baja los mismos.

En la interfaz gráfica se parte del listado de los usuarios que existen actualmente en la base de datos del sistema, así como alguno de los datos más relevantes de estos.

El sistema permite realizar el ingreso de nuevos usuarios, la modificación de usuarios ya existentes o la baja de aquellos que ya no utilizarán más el sistema. Para hacer estas acciones se contemplaron botones en la interfaz que realizan cada una de estas funcionalidades.

La pantalla para realizar todo esto, es la misma en todos los casos, simplemente cambian que en el alta se muestran todos los campos vacíos, en

la modificación se muestran todos los datos previamente cargados, y en la baja se muestra también todos los datos pero no se permiten modificar.

Los datos que se tienen en cuenta para los usuarios son:

- Nombre completo
- Usuario
- Contraseña
- Permisos
- Email
- Celular
- Envía Correo

Estos son los datos a especificar de cada uno de los usuarios. Cabe aclarar que por usuario nos referimos al nombre con el que se va a ingresar al sistema, mientras que el nombre completo sirve para identificar cada una de las personas.

Tanto el email como el número del celular ingresados son los que luego serán utilizados por el controlador de políticas para realizar los avisos pertinentes a aquellos usuarios configurados (los que tienen seleccionada la bandera Envía Correo).

Toda la información es guardada en la base de datos del sistema, en donde cada usuario tiene asignada una clave única que lo identifica.

Los permisos asociados a los usuarios son los siguientes:

- **Administrador:** es el permiso más alto que puede tener un usuario, puede realizar modificaciones en todo el sistema y a su puede visualizar toda la información que se muestra. Corresponde al usuario que realiza los mantenimientos al sistema, el cual cuando se requiera, el dueño de casa ingresará con dicho usuario.

- **Normal**: solamente puede disparar escenas y eventos puntuales. Corresponde al usuario que utiliza normalmente el sistema.

- **Restringido:** este usuario solamente puede visualizar la información en pantalla pero no puede modificar ni disparar ningún evento. Tiene asignado un rango horario y días de la semana, esto pensando en que el sistema a futuro pueda realizar el control de ingreso a la casa a través de algún dispositivo, por ejemplo un detector de huella, utilizando el sistema para ello. Este tipo de permiso está pensado para empleados de la casa o niños, los cuales se está interesado que no puedan acceder a ciertas secciones de la casa o el uso de ciertas instalaciones.

Manejo de Políticas de Control

Para manejar las políticas hay que tener en cuenta las etapas involucradas en el disparo de las mismas.

Cada política debe ejecutar una serie de acciones en forma automática dada una serie de condiciones de disparo. Las acciones deben de ejecutarse para todas las políticas que sean disparadas en forma simultánea y sin impedir que el sistema continúe con el funcionamiento normal.

Para permitir que el sistema continúe en funcionamiento los objetos que implementan las políticas son instanciados en hilos de ejecución independientes dentro de la maquina virtual.

El manejo del disparo de las políticas es a través del objeto ContoladorPoliticas. Este objeto se encuentra suscripto a listeners de todas las variables relevantes al disparo de políticas. Dadas las condiciones necesarias abre un hilo de ejecución independiente donde ejecuta la política, cuando esta termina su ejecución se cierra el hilo.

Las políticas durante su ejecución hacen tres tipos de acciones. La principal acción es correctiva y actúa directamente sobre el hardware del sistema. Un ejemplo que esclarece este tipo de acciones es enviar a la red LonWorks las variables de red necesarias para que cerrar la llave de gas en caso de activarse las políticas de incendio o de pérdida de gas.

Otro tipo de acciones consiste en el envió de e-mail y/o sms a usuarios registrados para recibir los mismos. La política correspondiente instancia la clase Mailer con el texto a enviar y esta se encarga de pedir consultas hacia la base de datos para recuperar los e-mail a los cuales enviar el mensaje.

El último tipo de acción es el de loguear el disparo de la política, este se implementa por medio del objeto LoggerInfo. Este objeto despliega en consola la información, la introduce en un archivo te texto y la almacena en un buffer que es utilizado por la interfaz gráfica para su despliegue en pantalla.

Manejo de eventos temporales (Películas)

Los eventos temporales son manejados a través de la creación de escenas y la incorporación de las mismas en a una película. Para el manejo de eventos temporales se utiliza el framework Quartz.

Una escena es un conjunto de acciones a ser ejecutadas en un momento predeterminado de tiempo definido en la Película. Las escenas son instancias de la clase Escena las cuales una vez definidas en la interfaz gráfica son guardadas para su posterior incorporación en la película.

La clase Pelicula implementa el patrón singleton y es donde se instancia el Scheduler de Quartz (la agenda). En la interfaz gráfica de gestión de la película se selecciona entre las escenas guardadas la que se desea agregar. Se le asigna fecha de inicio/fin, hora de inicio/fin y periodicidad, con la posibilidad de ejecutar la misma escena repetitivamente varios días de la semana. Por ejemplo, ejecutar una determinada escena los días lunes, miércoles y viernes durante un mes.

Una vez aceptada la inclusión de la escena en la película, a nivel de programa, se crea un trigger para cada acción de la misma, se asignan las acciones a los triggers y se registran los triggers en el Scheduler de la Pelicula.

Los triggers son utilizados para disparar tres posibles tareas (jobs), encender, apagar o realizar la asignación de un valor a una variable de red. OnJob, OffJob y SetJob son las tres clases que implementan la interfaz Jobs.

Ejemplos de aplicación

Un hogar digital posibilita la convergencia de servicios y, por lo tanto, la posibilidad de integrar funciones pertenecientes a diferentes ámbitos.

Entre los servicios que pueden brindarse se detallarán a continuación algunos ejemplos prácticos de utilización de SAICOH mediante el uso de supuestos escenarios. Estos escenarios son una adaptación libre del Capítulo 1.4 del ¨Libro Blanco del Hogar Digital y las Infraestructuras Comunes de Telecomunicaciones¨.

Los escenarios descriptos a continuación pueden ser ya una realidad con la utilización de nuestro sistema. El común denominador de estos escenarios es que los usuarios se benefician mediante el uso de la tecnología.

Los conceptos manejados son de común aplicación tanto en hogares como oficinas por lo que nos restringiremos a abordar el tema únicamente en el ámbito del hogar.

Escenarios

Juan y María son un joven matrimonio con 3 hijos que vive en Montevideo. Además, viven con la abuela, que lamentablemente fue operada de la cadera hace unas semanas.

Escenario 1: En cualquier lugar

Juan y María están en el trabajo y no vuelven en varias horas. Recordaron que debían dejar descongelando la heladera para vaciarla antes de irse de viaje por el fin de semana. Juan abre el navegador y se conecta a la aplicación de su casa, identificándose con nombre de usuario y contraseña, en ese momento nota la baja temperatura exterior y además de apagar la heladera, configura el encendido del aire acondicionado una hora antes de que regresen del trabajo.

Escenario 2: Seguridad personal

María y Juan contrataron la empresa de asistencia médica de emergencia que consideraron más adecuada para la abuela y colocaron botones de emergencia en toda la casa. Ellos quieren lograr de ese modo que la abuela este fuera de peligro en todo momento. Por esto si se siente mal o tiene cualquier problema, bastará con que pulse alguno de los botones para que la empresa reciba la alarma y se realicen avisos vía SMS y correo electrónico a Juan y María.

Además, tanto Juan como María pueden desde cualquier PC con acceso a internet acceder a la aplicación, ingresando al sistema con usuario y contraseña, y realizar la incorporación al sistema de la enfermera que fue contratada para el cuidado de la abuela.

Escenario 3: En casa

María está en casa, en una semana realizará una cena de trabajo allí y noto que no hay una escena adecuada para el evento. Entra a la aplicación y realiza la creación de una escena a su gusto, configurando las luces que considera más convenientes y seleccionando la temperatura de las habitaciones que se van a utilizar, así como el encendido del audio.

Además, María va a llevar a los niños al zoológico, por lo cual controla la temperatura exterior y el pronóstico para la tarde mientras realiza la configuración de escenas.

Escenario 4: De vacaciones

La familia se va de vacaciones durante un par de semanas.

Juan decide chequear que todo esté bien en el hogar, ingresando al sistema con su usuario y contraseña a la aplicación y comprueba que olvidaron activar el ¨simulador de presencia¨, lo activa y aprovecha para ver el estado de los electrodomésticos y controlar que las tomas estén cerradas.

Validación y Conclusiones

Validación

Analizaremos a continuación si los criterios de éxito y aceptación planteados lograron ser cumplidos independientemente de lo observable en la aplicación concreta del sistema en el modelo.

Se proponía el acceso al sistema mediante un cliente local y remotamente vía Internet. Basados en la decisión de diseño de que la interfaz grafica de la aplicación sea web, es necesario incorporar un servidor de aplicaciones donde se ejecute la misma. Si el cliente se encuentra en la misma máquina o ésta posee una IP estática asociada o un DNS público, la aplicación puede accederse desde cualquier punto local como remoto. Se probó esto en una LAN donde se fijo una IP a la maquina servidor y una maquina cliente accedió correctamente a la aplicación.

El requisito de obtener información de Internet, ingresarla y procesarla en el sistema fue llevado a cabo satisfactoriamente pudiendo apreciarse en la interfaz grafica condiciones meteorológicas actualizadas. Se concluye que esta es una manera simple y válida para presentar otros datos de interés como pueden ser titulares de diarios, etc.

Otro objetivo consistió en el envío de e-mails y/o SMS por parte del sistema. Este requisito es parte fundamental ya que se ve asociado directamente a los reportes de alarmas técnicas las cuales según estudios recientes realizados en la Unión Europea son muy apreciadas por los usuarios finales en el Hogar Digital. Se cumplió la pauta exitosamente generando avisos por e-mails y SMS a todos los usuarios del sistema (registrados a tales efectos) según se disparan las alarmas.

La robustez fue planteada como parte fundamental. A nivel de software, se implementó un diseño en capas cuidando las interfaces con el objetivo de minimizar los accesos a recursos no permitidos. Se idearon políticas de aviso frente a fallas de alimentación y caídas en las comunicaciones dejando la puerta abierta a continuar el desarrollo para la incorporación de dispositivos

redundantes como UPSs, módems, etc. Para controlar la caída de un nodo en la red LonWorks, se realiza periodicamente una consulta (ping) del estado de cada uno de los dispositivos.

En cuanto a la exigencia planteada de realizar pruebas prolongadas en el tiempo con el fin de testear estabilidad y performance, se ha mantenido el sistema en funcionamiento durante períodos prolongados de tiempo, sin encontrar inconvenientes.

La incorporación al sistema de un dispositivo “no LonWorks” fue dejado de lado como ya se explicara al referirnos a los inconvenientes plateados con el detector de huellas dactilares.

En resumen, el proyecto llegó a su fin satisfactoriamente, en funcionamiento y con las exigencias superadas. El producto se podrá considerar como para salir al mercado si se mejora el diseño gráfico y se le agregan facilidades en la etapa de provisión (primera configuración general luego de la instalación).

Conclusiones

Se diseño y desarrollo un sistema que permite el monitoreo y el control remoto de una muestra descriptiva de dispositivos que implementan el protocolo LonWorks.

Se utilizaron tecnologías Java Enterprise Edition y frameworks seleccionados para realizar funciones específicas de manera óptima. La arquitectura del software fue implementada en capas, pensado de forma que el sistema sea escalable y simple al realizar mejoras y ampliaciones en las funcionalidades actuales.

El sistema implementado es capaz de obtener y enviar datos a través de internet y de realizar acciones automáticas sobre el hogar ante situaciones críticas (Políticas).

A diferencia de otras ofertas domóticas en el Uruguay este desarrollo se encuentra basado en un estándar internacional para redes domóticas de tercera generación, LonWorks.

Ofrece a su vez servicios de valor agregado sobre las funciones básicas como ser la presentación del estado del tiempo actualizado vía internet. Además, resulta importante destacar que el sistema brinda a través de las políticas el manejo de alarmas técnicas, funcionalidad más valorada por usuarios finales de tecnologías domóticas según la encuesta “Estudio Mint-Casadomo 2008” realizada en España y publicada el 17 de Julio del 2008.

En resumen, se obtuvo una versión de pasarela residencial implementada con tecnología de última generación la cual trabaja con dispositivos que implementan el protocolo LonWorks. Ésta ofrece funcionalidades básicas y avanzadas de control domótico.

Posibles Líneas de Trabajo a Futuro

Para proyectar el futuro de la aplicación debemos considerar dos caminos, uno es la salida en producción con las funcionalidades actuales y el otro, la incorporación de más funcionalidades de interés para posibles usuarios finales.

En cuanto a llevar la aplicación a manos de usuarios finales debe trabajarse principalmente en el testing, un diseño grafico de mayor calidad y optimizar la usabilidad apuntando a lograr una interfaz simple y amigable. Las bases para el correcto manejo de la etapa de provisión y la recuperación del sistema frente a caídas se encuentran establecidas más no aún implementados en su totalidad.

Como nuevas funcionalidades a incorporar figuran posibilidades limitadas únicamente por la imaginación. Al integrar el monitoreo y control del hogar con una aplicación web desarrollada con la última tecnología disponible es posible agregar valor con servicios vinculados al manejo de la información. A modo de ejemplo la obtención de datos vía Internet como titulares de noticias obtenidos por medio de la lectura de RSS o información financiera de bolsas de valores.

Incorporar a la interpretación de XML disponibles en la Web la utilización de Web Services es una alternativa prometedora. Integrar el sistema con una PBX basada en Asterisk abre una nueva gama de posibilidades entre las que tenemos el reporte de alarmas técnicas telefónicamente con mensajes predeterminados, la utilización de la aplicación vía DMTF, etc.

Implementar aplicaciones basadas en reconocimiento de voz en el cual el usuario pueda ejecutar acciones sobre el hogar desde el teléfono o a través de micrófonos ambiente, es otra de las posibilidades.

Otra etapa a desarrollar sería la incorporación de dispositivos de audio y video.

Queda más que demostrada la potencialidad de la aplicación restando focalizar los futuros desarrollos en las áreas de interés deseadas.

Referencias

- http://www.echelon.com, Agosto 2007.
- LonWorks Installation Handbook, second edition, Julio 2007. ISBN 978-3-8007-2906-7.
- Open Control Networks, Dietmar Dietrich, Julio 2007.ISBN 0-7923-7406-1.
- Análisis del estado del arte de los buses de campo aplicados al control de procesos industriales - Dr. Ing. Héctor Kaschel C. y el Ing. Ernesto Pinto L. Fac. de Ingeniería, Depto. de Ingeniería Eléctrica - Universidad de Santiago de Chile.
- Thinking in Java.. Eckel, Bruce., third edition, Julio 2007.
- Simple API for XML. http://java.sun.com/j2ee/1.4/docs/tutorial/doc/JAXPSAX.html, Enero 2008.
- APIs and Documentation on Sun Developer Network. http://java.sun.com/j2se/1.5.0/docs/api/, Julio 2007.
- UML y patrones. Una introducción al análisis y diseño orientado a objetos y al Proceso unificado, Larman, Craig., 2ª edición. Madrid, 2003. ISBN 84-205-3438-2.
- Guías de clase 2006 de la materia Desarrollo de Software para Ingeniería Eléctrica, IIE, UDELAR.
- http://es.wikipedia.org/wiki
- http://es.wikipedia.org/wiki/Dom%C3%B3tica, Agosto 2007.
- http://www.domotica.net/504.html , Agosto 2007.
- http://www.domodesk.com/content.aspx?co=51&t=21&c=43, Agosto 2007.
- http://www.casadomo.com, Agosto 2007.
- http://www.casadomo.com/noticiasDetalle.aspx?c=49&idm=60&m=15&n2=14&pat=14, **Setiembre 2008.**
- http://www.domoticaviva.com, Agosto 2007.
- http://www.domoticaviva.com/noticias/013-200702/pasarela1.htm **setiembre 2008.**
- http://www.prosegur.com.uy/home.prosegur?seccion=alarmas, Agosto 2007.

- http://articulo.mercadolibre.com.uy/MLU-4689404-sistemas-de-domotica-e-instalaciones-de-audio-_JM, Agosto 2007.
- http://www.todo.com.uy/guia2.php3?nidrubro=139045&nidpadre=11600 , Agosto 2007.
- http://www.archisound.com.uy/, Agosto 2007.
- http://www.domotica.com.uy , Agosto 2007.
- http://www.aldeadomotica.com , Agosto 2007.
- http://www.domotec.com.uy/, Agosto 2007.
- http://www.buyusa.gov/uruguay/es/49.html, Agosto 2007.
- http://www.indomotika.com, Agosto 2007.
- http://java.sun.com
- http://java.sun.com/j2ee/1.4/docs/tutorial/doc/JAXPSAX2.html
- http://www.oracle.com
- http://www.postgresql.org/
- http://www.service-architecture.com/
- http://www.rae.es/rae.html
- Software de Análisis de Propagación Outdoor, Documentación de Proyecto de fin de carrera, Katz-Larroca-Martino, Marzo 2006.
- Libro Blanco del Hogar Digital y las Infraestructuras Comunes de Telecomunicaciones (ICT), Telefónica España, 2007.
- La casa digital, Manuales Plan Avanza, 2007.
- ISBN: 84-611-4741-3.
- www.starlims.com, Setiembre 2008.
- , Java a Tope: JavaMail en ejemplos, Gálvez-García. Edición Electrónica, 2006. ISBN: 84-690-0697-5.
- Administración SGBD PostgreSQL, J.M. Alarcón, Noviembre 2006.
- Definición de una arquitectura software para el diseño de aplicaciones web basadas en tecnología Java-J2EE, D. Fernández
- Estudio Mint-CasaDomo 2008: Sistemas de domótica y seguridad en viviendas de nueva promoción, Madrid, España, Julio 2008

Glosario

AccuWeather – compañía estadounidense dedicada a la publicación de pronósticos meteorológicos

Actuador – es el nodo en el que se realizan acciones. Nuestra manera de "intervenir" en el mundo real.

Building – estructura jerárquica que representa el edificio, todas las habitaciones y elementos que se encuentran

Boletín – información sobre los sucesos que el sistema considere de importancia para el usuario y por lo tanto despliega en la GUI

Domótica – Conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda.

Escena – cualquier configuración del estado de la casa (iluminación, cargas, temperatura, etc.) que es almacenada para su exacta reproducción posteriormente.

Framework –estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado

Glassfish – servidor de aplicaciones

Java - lenguaje de programación orientado a objetos

JEE – Java Enterprise Edition, plataforma de desarrollo de aplicaciones empresariales de java

Job – Tarea que se asocia a un trigger en Quartz. Al darse la condición temporal de disparo Quartz ejecuta los jobs (o tareas) asociadas.

JPA – Java Persistance Api, api de java para manejo de la persistencia con bases de datos

LNSHMI – Lonwork Network Server Human Machine Interface, api de java que conecta la red lonworks con el desarrollo en java

LonWorks – Protocolo domótico

MVC – Model View Controller

NetBeans – entorno de desarrollo de java

Nodo – cada dispositivo con un Neuron id diferente que puede estar formado por sensores y actuadores.

Open Source – tipo de licenciamiento de software

ORM – Object Relation Mapping

Paquete – en software, conjunto de clases que tienen realizan acciones similares por lo que pueden ser agrupadas

Pasarela Residencial – dispositivo que conecta las infraestructuras de telecomunicaciones del hogar digital a una red pública de datos

Película – ejecución en una secuencia predeterminada de escenas

Persistencia – acción de preservar la información de un objeto de forma permanente (guardar), pero a su vez también se refiere a poder recuperar la información del mismo (leer) para que pueda ser utilizada nuevamente

Política – procedimiento que se ejecutará automáticamente frente a la ocurrencia de determinados incidentes

PostgreSQL – motor de base de datos open source

Quartz – motor de ejecución de eventos temporales basado en java.

Recomendaciones – avisos no urgentes que el sistema dará al usuario sobre la práctica a seguir frente a determinado suceso. El sistema no actuará por sí solo en estos casos.

Scheduler – Entidad del entorno Quartz que se encarga de gestionar y monitorear triggers y jobs registrados.

Sensor – es el nodo en el que se realiza adquisición de datos. Nuestra manera de “ver” el mundo real.

Sistema – es el software que controla la red y por esta la realidad

Suceso – cualquier evento que el TTR necesite saber para controlar la casa

TopLink – framework para manejar la persistencia en aplicaciones java

Trigger – entidad de Quartz a la que se le asocia una tarea a ejecutar y una o varias condiciones temporales de disparo. Quartz monitorea todos los triggers registrados y al darse la condición de disparo ejecuta la tarea asociada.

TTR (Tarea en Tiempo Real) – es el motor encargado del control de la casa a lo largo del tiempo teniendo capacidad de manejar todos los sucesos que en ella se den (cambio de escenas, imprevistos, alarmas, avisos, etc.).

Usuario – persona que controlará su casa mediante la cooperación del sistema

XML – Extensible Markup Language, un estándar para el intercambio de información estructurada entre diferentes plataformas

Anexos

Buses de Campo

Un bus de campo es un sistema de transmisión de información que simplifica enormemente la instalación y operación de máquinas y equipamientos, tanto industriales como domésticos. Entre los objetivos de un bus de campo se encuentra el de sustituir las antiguas conexiones punto a punto entre los elementos de campo y el equipo de control.

Típicamente, son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie, que conectan dispositivos de campo como PLCs, transductores, actuadores y sensores. Cada dispositivo de campo incorpora cierta capacidad de proceso, que lo convierte en un dispositivo inteligente, intentando mantener siempre un bajo costo.

El objetivo principal es reemplazar los sistemas de control centralizados por redes de control distribuido mediante la cual se mejora la calidad del producto, se reducen los costos y se mejora la eficiencia. La idea principal tras esto es que gracias a compartir un mismo medio físico de transmisión, los datos adquiridos por un equipo pueden ser distribuidos al resto de los que se encuentran en el bus, adoptándose precauciones para no saturar el medio físico innecesariamente. Si a esto sumamos que cada dispositivo de campo es un dispositivo inteligente que puede llevar a cabo funciones propias de control, m antenimiento y diagnóstico obtenemos una plataforma distribuida de control.

Ventajas de los Buses de Campo

La principal ventaja que ofrecen los buses de campo, y la que los hace más atractivos a los usuarios finales, es la reducción de costos. El ahorro proviene fundamentalmente de tres fuentes: ahorro en el costo de instalación, ahorro en el costo de mantenimiento y ahorros derivados de la mejora del funcionamiento del sistema. Una de las principales características de los buses de campo es su significativa reducción en el cableado necesario para el control de una instalación. Cada componente sólo requiere un cable para la conexión de los diversos nodos. Se estima que utilizando un bus integrado de control puede ofrecerse una reducción de 5 a 1 en los costos de cableado. Ciertos buses de

campo (como en el caso particular de LonWorks) soportan la utilización de topologías “libres” lo que ofrece la posibilidad de extender el bus en cualquier dirección a bajo costo.

Los buses de campo permiten a los operadores monitorear todos los dispositivos que integran el sistema e interpretar fácilmente las interacciones entre ellos. Esto es útil al momento de optimizar el sistema y simplifica la detección de las posibles fuentes de fallas, reduciendo costos de mantenimiento y tiempos de parada. El usuario obtiene mayor flexibilidad en el diseño del sistema.

Con la incorporación del procesamiento distribuido, algoritmos y procedimientos de control que en sistemas tradicionales debían incluirse en las centrales, pueden realizarse ahora en los propios dispositivos de campo, simplificando los sistemas de control, disminuyendo el tráfico en el bus, optimizando los tiempos de respuesta y parada.

Las prestaciones del sistema mejoran con el uso de la tecnología de los buses de campo debido a la simplificación en la forma de obtener información de la planta desde los distintos elementos. Las mediciones de estos dispositivos estarán disponibles, de ser necesario, para todos los nodos. Esto permitirá el diseño de sistemas de control más eficientes y robustos.

La Guerra de los Buses

Ante la variedad de opciones existente, parece razonable pensar que fabricantes y usuarios hicieran un esfuerzo en la búsqueda de normativas comunes para la interconexión de sistemas industriales y domésticos.

Lo que ha venido llamándose "la guerra de los buses" tiene que ver con la permanente confusión reinante en los entornos normalizadores en los que se debate la especificación del supuesto "bus de campo universal". Desde mediados de los años ‘80 la Comisión Electrotécnica Internacional (IEC-CEI) y la Sociedad de Instrumentación Americana (ISA) ha sido escenario del supuesto esfuerzo de los fabricantes para lograr el establecimiento de una norma única de bus de campo de uso general. En 1992 surgieron dos grupos, el ISP (Interoperable Systems Project) y WorldFIP cada uno promoviendo su propia versión del bus de campo. En el primer grupo estaban fabricantes como

Siemens, Fisher-Rosemount, Foxboro y Yokogawa. En el segundo Allen-Bradley, HoneyWell, Square D y diversas empresas francesa. En 1994 ambos grupos se unieron en la Fieldbus Foundation. El debate se trasladó luego, y continúa en la actualidad, a la conjunción de Fieldbus y el mundo Profibus. Los años pasan, la norma del supuesto bus universal nunca se acaba de generar y en el camino aparecen nuevas opciones como CAN, LonWorks, Ethernet, etc. Incluso el debate es confuso y totalmente incomprensible, otras empresas participantes en el debate generaban en paralelo soluciones propias, es el caso de Allen-Bradley con DeviceNet y HoneyWell con SDS. La realidad es que sólo los usuarios están realmente interesados en la obtención de normas de uso general. Los fabricantes luchan por su cuota de mercado y, en general, sólo están a favor de una norma cuando ésta recoge las características de su propia opción, lo cual es comprensible dadas las fuertes inversiones necesarias para el desarrollo de un bus industrial normalizado. El debate sigue abierto…

Comparación de Tecnologías Domóticas.

En los entornos de control y automatización existen protocolos y estándares que permiten el intercambio de paquetes de datos con bajas latencias (tiempo de respuesta limitados). Entre estas tecnologías destacamos LonWorks, EIB y X-10 por ser las más usadas al momento de implementar soluciones para espacios domésticos, comerciales o de oficinas.

Como se mencionó anteriormente, uno de los aspectos que caracteriza los inicios de la domótica es la diversidad de entidades que intentaron forzar el mercado con la adopción de sus propias iniciativas.

A continuación se presenta procedencia, origen y ámbito de aplicación de algunos productos así como diferentes características relevantes a tener en cuenta.

Iniciativa	Procedencia		Ámbito de aplicación
	Promotor	País	
Batibus	Merlin Gerin (Schneider Electric)	Francia	Europa
EIB	Siemens	Alemania	Europa
EHS	Comisión Europea	Unión Europa	Europa
X-10	Pico Electronics Ltd	UK	Mundial
LonWorks	Echelon	EE.UU.	Mundial
CEBus	Asociación de Industrias Electrónicas de EE.UU.	EE.UU.	EE.UU.
HBS		Japón	Japón

REDES DE CONTROL Y AUTOMATIZACIÓN			
Tecnología	**Medio de Transmisión**	**Velocidad de Transmisión**	**Distancia máxima al dispositivo**
Konnex	1. TP0 2. TP1 3. PL100 4. PL132 5. Ethernet 6. Radio	2. 9600 bps 3. 1200/2400 bps 4. 2.4 Kbps	2. 1000 m 3. 600 m
Lonworks	1. TP 2. Cable eléctrico 3. Radio 4. Coaxial 5. FO	1. 78 Kbps – 1.28 Mbps 2. 5.4 Kbps	1. 500 – 2700 m
X10	Cable eléctrico	60 bps en EEUU 50 bps en Europa	185 m^2
BacNet	• Cable Coaxial • TP • FO	1 Mbps – 100 Mbps	Con Ethernet sobre TP: 100 m
EIB	1. TP 2. Cable eléctrico 3. RF 4. Infrarrojos	1. 9600 bps 2. 1200/2400 bps	1. 1000 m 2. 600 m 3. 300 m
EHS	1. Cable eléctrico 2. TP	1. 2.4 Kbps 2. 48 Kbps	
Batibus	TP	4800 bps	200 m a 1.500 m en función de la sección de cable
Cebus	• TP • Cable eléctrico • Radio • Coaxial • Infrarrojos	10.000 bit/s	En función de las características del medio
DALI	Par de cable	----------	200 m
Metasys	N2 Bus	9600 bps	1219 m
SCP	Cable eléctrico	<10 Kbps	----------
ZigBee	Inalámbrico	20 Kbps-250Kbps	10 m – 75 m

Principales sistemas

X-10

X-10 es uno de los protocolos más antiguos que se utiliza en aplicaciones domóticas. Su origen se remonta a los años comprendidos entre 1976 y 1978 en Escocia. Fue diseñado inicialmente para transmitir datos por líneas de baja tensión a muy baja velocidad y bajo costo. Es un protocolo no propietario, es decir, cualquiera puede producir dispositivos X-10 y ofrecerlos al mercado. La única restricción existente es la necesidad de utilizar la implementación del circuito de comunicaciones concebido por el fabricante escocés dueño de la tecnología, disponible a un bajo costo.

A nivel físico el protocolo usa una modulación muy sencilla. El transceptor X-10 esta pendiente de los pases por cero de la onda sinusoidal existente sobre la línea de alimentación, para insertar un instante después una ráfaga muy corta de señal en una frecuencia fija, pudiéndose insertar esta señal en el semiciclo positivo o negativo de la onda sinusoidal.

La transmisión completa once ciclos divididos en tres campos de información. Dos ciclos para el “Código de Inicio”, cuatro para el “Código de Casa” y cinco para un “Código Numérico o de Función”. Además para aumentar la fiabilidad del sistema la trama se transmite dos veces separadas por dos ciclos.

Existen tres tipos de dispositivos X-10; los que solo transmiten ordenes, los que solo reciben y los que hacen ambas cosas..

European Installation Bus o EIB

EIB es un sistema desarrollado bajo los auspicios de la Unión Europea. El objetivo era crear un estándar europeo que permita comunicarse todos los dispositivos de una instalación eléctrica (climatización, seguridad, electrodomésticos, etc.).

Esta basado en una estructura de niveles OSI y tiene una arquitectura descentralizada definiendo una relación extremo a extremo entre dispositivos que permite distribuir la inteligencia entre sensores y actuadores instalados en una vivienda.

A nivel físico permite utilizar par trenzado, corrientes portadoras, Ethernet, radiofrecuencia o infrarrojo. En la práctica el único medio físico de utilización masiva es el par trenzado.

La EIBA es una asociación de 113 empresas europeas con sede en Bruselas que se unieron en 1990 para impulsar el uso e implantación del sistema EIB. Sus tareas son entre otras dar directrices técnicas, distribuir el conocimiento y experiencia, conceder a los productos y fabricantes una licencia EIB. Colabora además con organismos europeos e internacionales en todas las fases de normalización y es líder en el proceso de convergencia de los tres buses europeos (EIB, Batibus y EHS) en Konnex.

LonWorks (EIA 709.1)

Echelon presentó la tecnología LonWorks en el año 1992. Está diseñada para cubrir los requisitos de la mayoría de las aplicaciones de control y ha tenido gran éxito en instalaciones de control y monitoreo para edificios comerciales, de viviendas e industrias.

Desde su origen ofrece una solución con arquitectura descentralizada, extremo a extremo, que permite distribuir la inteligencia entre los sensores y los actuadores instalados en la red. El estándar LonWorks tiene la particularidad de cubrir desde el nivel físico al de aplicación de la mayoría de los proyectos de redes de control.

Echelon define su arquitectura como un sistema abierto a cualquier fabricante que desee utilizar esta tecnología, sin depender de sistemas propietarios, lo que permite reducir costos y aumentar la flexibilidad de la aplicación de control distribuida.

Cualquier dispositivo LonWorks, también llamado nodo, esta basado en un microcontrolador particular, el Neuron Chip. El circuito integrado y el firmware que implementa el protocolo de comunicación entre los dispositivos (LonTalk) fueron desarrollados completamente por Echelon en 1990.

Se puede destacar del Neuron Chip que tiene un identificador único (Neuron ID), un modelo de comunicaciones independiente del medio físico y el firmware que implementa el protocolo LonTalk proporciona servicios de transporte y ruteo extremo a extremo.

Los dispositivos se comunican entre si por medio de datagramas que contienen la dirección de destino, información para el ruteo, datos de control, que incluyen datos de la aplicación de usuario, y CRC como código detector de errores.

Los datos pueden tener dos formatos, mensajes explícitos o variables de red. Los mensajes explícitos son la forma más sencilla de intercambiar datos, las variables de red proporcionan un modelo estructurado para el intercambio estandarizado de datos distribuidos, permitiendo al programador de la aplicación estar menos pendiente de los detalles de comunicación.

Respecto a la fabricación del Neuron Chip, Echelon solo ha concedido licencia a tres fabricantes de semiconductores (Cypress Semiconductor, Toshiba y Motorola), que además pagan un royalty por cada circuito fabricado.

A nivel físico, el Neuron Chip proporciona un puerto específico de cinco pines que puede ser configurado como interfaz para diferentes tranceptores de línea a diferentes velocidades. El tranceptor es el encargado de adaptar las señales del Neuron Chip a los niveles que necesita cada medio físico.

A continuación se presentan los tranceptores más utilizados con algunas de sus principales características.

Transceptor	Medio Físico	Velocidad binaria	Topología de red	Distancia máxima	Nodos máximos	Otros
PLT-22	Ondas Portadoras	5-4 Kbps	Cualquiera en redes de baja tensión o par trenzado sin alimentación	Depende de la atenuación entre emisor y receptor y del ruido en la línea	Depende de la atenuación entre emisor y receptor y del ruido en la línea	Compatible con PLT-20 y PLT-21
FTT-10A	Par Trenzado	78 Kbps	Bus, estrella o lazo. Cualquier combinación	500 metros, hasta 2700 metros con doble bus e impedancias de carga en	64	Compatible con FTT-10 y LPT10

				los extremos		
LPT-10	Par Trenzado	78 Kbps	Bus, estrella o lazo. Cualquier combinación	500 metros, hasta 2700 metros con doble bus e impedancias de carga en los extremos	32, 64, 128 en función del consumo	Capaz de tele-alimentar nodos por el mismo par trenzado
TPT/XF-78	Par Trenzado	78 Kbps	Bus	1400 metros	64	Aislado con transformador
TPT/XF-1250	Par Trenzado	1,25 Mbps	Bus	130 metros	64	Aislado con transformador

Existe una asociación de fabricantes que desarrollan productos y servicios basados en LonWorks llamada LonMark. Los objetivos de esta son especificar las variables que se intercambian en la red de control a nivel de aplicación, definir perfiles funcionales que estandarizan las funciones de forma que diversos fabricantes puedan ofrecer el mismo producto funcional pero con un hardware distinto.

Luego de analizar las diferentes opciones que existen en el mercado, nos volcamos por el sistema LonWorks para realizar nuestro proyecto, ya que este cumplía de forma más acertada con nuestros objetivos a llegar.

Anexo II – Transceptores y medios de transmisión

Transceptores

Cada dispositivo LonWorks (nodo) contiene un transceptor. Estos periféricos proveen de la interfaz física entre un equipo LonWorks y su red y se hallan para una amplia variedad de medios de comunicación y topologías. Su estandarización simplifica el desarrollo de dispositivos LonWorks y asegura la interoperabilidad. De esto se desprende lo importante que es conocer de qué transceptor dispone cada producto ya que este es un factor determinante en el tipo y forma de la red a utilizar. Dispositivos con diferentes tipos de transceptores pueden interconectarse requiriéndose la utilización de enrutadores.

Transceptores más utilizados:

Transceiver Type	Bit Rate	Network topology	Nodes per channel	Maximum Length	Type of isolation of Neuron
TP/XF1250	1.25 Mbps	Bus	64	130m	Transformer
FTT10A	78 kbps	Bus	64	2700m	Transformer
FTT10A	78 kbps	Free	64	500m	Transformer
LPT10	78 kbps	Bus	128	2200m	Transformer
PowerLine	4.8 kbps	Free	64	Up to 5km	Custom

Características de los medios de transmisión

Par Trenzado

Estabilidad, bajo costo y manejabilidad

Velocidad de transmisión de hasta 1.25Mbps

Alta confiabilidad (Menos propenso a interrupciones)

Par Trenzado con LinkPower

Similar al par trenzado

Suministro de comunicaciones y alimentación a través del mismo par

Terminación de red en la propia fuente de alimentación

Corrientes Portadoras (Power Line)

Medio de transmisión red eléctrica (220V AC)

Varios rangos de frecuencia

Sin cableado adicional, ni preinstalación

Flexibilidad

Resistente a interrupciones

Velocidad Tx. ~10 Kbps

Grandes distancias

Radiofrecuencia

Transmisión Mediante ondas electromagnéticas

Flexibilidad total de posicionamiento de los dispositivos

No requiere cableado

Costo elevado

Alta sensibilidad a las interferencias

Velocidad de transmisión ~5 Kbps

Dificultad en la estructura de red

Rango de alcance limitado

Bandas 400-470 Mhz y 900Mhz

Fibra Óptica

Transmisión basada en el principio de reflexión de las ondas de luz

Elevadas velocidades

Robustez frente a interferencias electromagnéticas

Alcanzan largas distancias.

Fundamentalmente empleado en conexiones punto a punto con largas distancias (como backbone)

Cable Coaxial

Velocidad de Tx. Media

Propenso a interrupciones en las comunicaciones

Especial atención a la instalación del cable

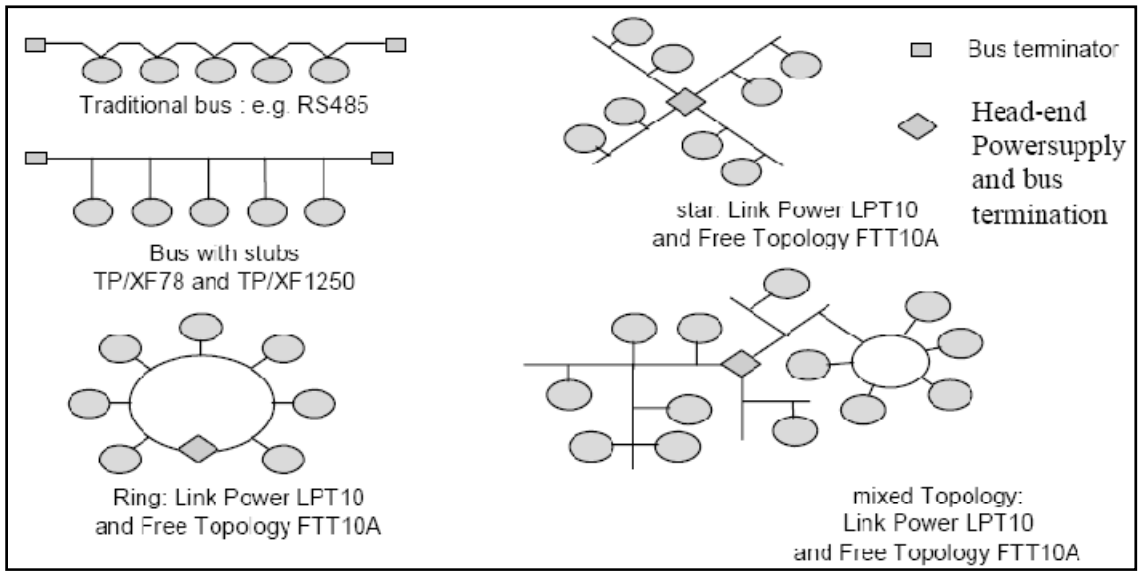
En desuso

Dentro de cada uno de los medios comentados, existen diferentes tipos de cable a emplear, los cuales proporcionarán mayores o menores velocidades de acuerdo a las características de los mismos. Deben considerarse las topologías permitidas en cada uno de los medios ya que en general no resultan ser variables totalmente independientes.

Topologías de redes cableadas

LonWorks soporta múltiples canales de comunicación. Un canal es el medio físico por el cual se intercambian datos pudiendo contener hasta 32.285 nodos.

Las topologías soportadas son bus, anillo, estrella y cualquier combinación de éstas. El diseño del transceptor es lo que determina la cantidad máxima de nodos por canal, la distancia máxima de comunicación y las topologías soportadas por cada red de dispositivos.



RS485 (Bus Tradicional)

El cable empleado puede ser de dos o cuatro hilos y mallado o no (según implementación y blindaje). Si el tendido es largo se requiere de terminaciones de línea (resistencias) para evitar la posible reflexión de la señal en los extremos del cable. Esta topología era utilizada principalmente para formar el backbone de la red ya que presenta una alta tasa de transferencia de datos (hasta 1.25Mbps). Actualmente se encuentra cada vez más en desuso.

FTT/LPT (Estrella – Bus – Anillo – Libre)

La compatibilidad de los FTT (Free Topology Transceiver) y los LPT (Link Power Transceiver) brindan una gran flexibilidad a la hora de proyectar una red. Estos transceptores permiten utilizar topologías en estrella, bus o anillo. Pero la gran ventaja de estos dispositivos es que posibilitan que las topologías puedan mezclarse eliminando la rigidez, abaratando costos de instalación y facilitando la posterior expansión. FFT y LPT se emplean en general en las zonas más cercanas a los puntos de control ya que no poseen altas tasas de transferencia de datos (78Kbps.) pero sí gran flexibilidad en cuanto a la conformación física de la red.

Debe considerarse que empleando topologías en estrella, anillo o mixta es necesaria la utilización de al menos una terminación de bus (en la mayoría de los casos basta con una resistencia de 52.3Ω). En caso de utilizar una topología en anillo deberá de tenerse en cuenta la polaridad de la red por lo que es altamente recomendado mantener un orden claro en las conexiones.

En el caso de estar empleando una fuente de alimentación para los LPT, no será necesario el empleo de terminaciones de bus ya que esta se encuentra incorporado en la propia fuente.

En Topología Libre, la distancia máxima entre nodos permitida, varía en relación al tipo de cable empleado, encontrándose entre 500m y 250m. En cuanto a la distancia máxima del tendido (distancia a la terminación de bus), está se encuentra comprendida entre 450m y 500m.

FFT- 10A / LPT 10A en Topología Libre, tendidos máximos por tipo de cable

Tipo de Cable	Distancia máxima entre nodos	Tendido máximo
Cat 5	250m	450m
JY(St)Y 2x2x0.8	320m	500m
UL Level IV, 22 AWG	400m	500m
Belden 8471	400m	500m

En Topología de Bus, se dispone de una estructurada ideada para extenderse grandes distancias. En estos casos, los máximos oscilan entre los 500m y 2700m para FFT y de 750m a 2200m en el caso de utilizar FFT/LPT. En todos los casos, es recomendable que la distancia entre el troncal principal y los nodos no supere los 3m.

Independientemente de la topología y tipo de cable empleado las limitaciones de cantidad de nodos por segmento son de 64 para FFTs y 128 para LPTs. En caso de topología mixta (FFTs y LPTs), podrán ubicarse hasta 128 nodos contando cada FFT como dos nodos y cada LPT como uno. En el caso de utilizar LPTs debe considerarse el consumo total de los nodos y la potencia de la fuente, valores que pueden limitar la máxima cantidad de dispositivos por segmento.

PLT (Libre)

La utilización de PLTs (Power Line Transceivers) en redes mediante línea portadora permite topología libre. Dado que las comunicaciones se realizan a través del tendido eléctrico existente del edificio llegando a todos los puntos en donde halla acceso al mismo. Este sistema resulta muchas veces en la alternativa más viable (tanto técnica como económicamente) al momento de hacer instalaciones en edificaciones existentes. La velocidad de transmisión es menor a 10Kbps por lo que es empleada en redes cercanas a los nodos de control. Debido a su baja tasa de transferencia no requiere de terminaciones de red.

Lon IP (Libre)

Esta topología de red emplea el medio físico de una red Ethernet razón por la cual se manejan tasas de trasferencia de datos mucho mayores respecto al resto de las opciones. Es empleada normalmente como backbone de la red.

Fibra Óptica (Bus - Anillo)

Este medio no empleado habitualmente dados sus altos costos presenta altas tasas de transferencia de datos. Su uso se reduce prácticamente a la interconexión entre edificios separados considerablemente cuando no es viable usar un enlace “público” como Internet. La topología preferida es la de bus ya que la conexión entre distintos segmentos de red se hace complicada debido a la naturaleza de la fibra óptica. Topologías de anillo pueden ser utilizadas para fortalecer la redundancia del sistema.

Dadas las posibilidades analizadas, los beneficios en flexibilidad al momento del diseño de una red Lon son notables. Este punto entre otros es lo que motiva a instaladores e integradores de sistemas a considerar seriamente entre sus opciones la plataforma Lonworks para sus proyectos.

La plataforma Java Enterprise Edition

Entre las primeras decisiones tomadas al momento de la elección de las diversas tecnologías a utilizar se encontró la de optar por la plataforma de desarrollo. Nos inclinamos hacia la plataforma Java Enterprise Edition más conocida como JEE.

Esta plataforma cuenta con un poco menos de diez años de desarrollo, en este momento se encuentra en la versión 5, la cual será utilizada en nuestro proyecto. Cabe destacar que hasta la versión anterior a esta (versión 1.4) la plataforma se denominaba J2EE. Al definir el salto a la versión actual se opto por renombrar la plataforma.

JEE permite desarrollar aplicaciones empresariales distribuidas, basadas en componentes mediante el uso de un modelo de aplicación multicapa. En general se suele aplicar en sistemas de tres capas (Fig. 1) o más.

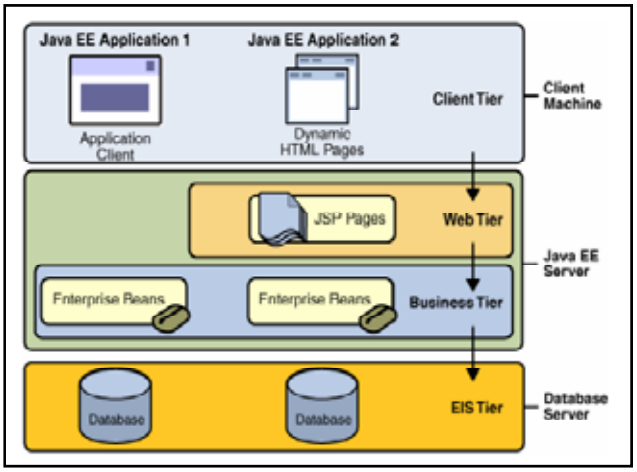


Fig. 1 - Modelo de Aplicación en tres capas

La evolución de la plataforma Java Enterprise Edition ha sido notable desde su origen como el proyecto JPE a mediados de 1998. El siguiente bosquejo (Fig. 2) ilustra los principales agregados conforme las siguientes versiones fueron liberadas al uso público.

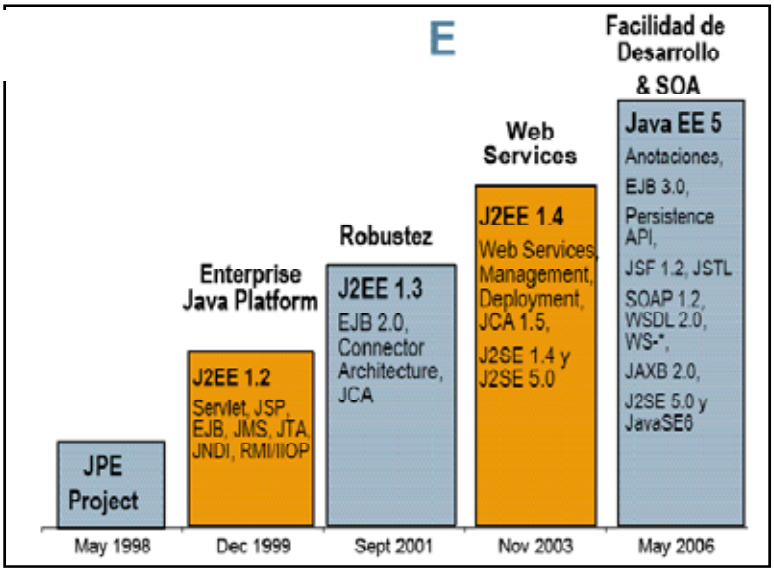


Fig. 2 - Evolución de la Plataforma Java Enterprise Edition

La versión 5 de Java Enterprise Edition permite desarrollar aplicaciones portables, robustas, escalables y seguras. Generada a partir de otra plataforma Java, la Standard Edition, JEE contiene todo aquello que podemos encontrar en J2SE y le agrega la posibilidad de desarrollar aplicaciones con arquitecturas orientadas a servicios (SOA) así como aplicaciones web de última generación.

En grandes rasgos, JEE se basa en componentes, un componente JEE es una unidad de software autocontenida que es integrada a una aplicación y se comunica con otros componentes.

La especificación define diferentes componentes para cada nivel entre los que tenemos:

- Cliente
- Aplicaciones
- Applets
- Componentes web
- Java Servlets
- Java Server Faces (JSF)
- Java Server Pages (JSP)
- Componentes de negocio
- Enterprise Java Beans (EJB)

Servidor de Aplicaciones

Un servidor de aplicaciones es un dispositivo de software que proporciona servicios de aplicación, gestionando la mayor parte de las funciones de lógica de negocios y de accesos a los datos de la aplicación. Aunque los servidores de aplicaciones no existen únicamente para Java en general se asocian a la tecnología Java Enterprise Edition (JEE).

Existe una amplia gama de éstos tanto en código abierto como cerrado. Como ejemplos tenemos:

Propietarios

- WebSphere (IBM)
- Oracle Application Server (Oracle Corporation.)
- WebLogic (BEA)
- Sun Java Application Server (SUN Microsistems)

Open Source

- JOnAS (ObjectWeb, el primero libre)
- Jeronimo (Apache Fundation)
- JBoss (de los mas populares)
- GlassFish (con base en el servidor Sun Java Applicaton Server y con el modulo de persistencia TopLink provisto por Oracle).

Los servidores de aplicación para JEE sirven como contenedor de los componentes que conforman la aplicación. Es común confundir Tomcat con un servidor de aplicaciones, siendo este en realidad, un contenedor de servlets y servidor web.

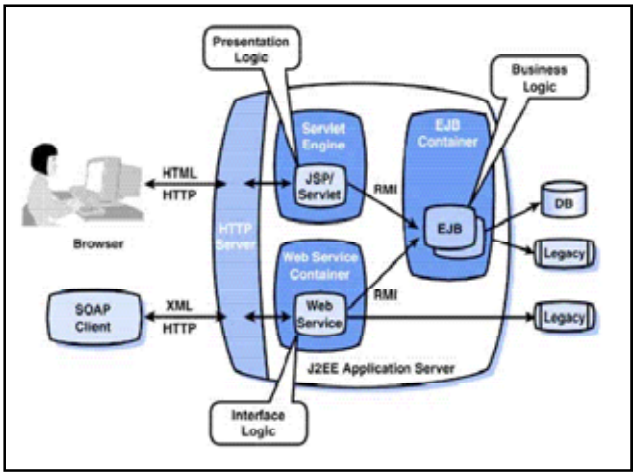


Fig. 3 - Servidor de Aplicación

Hasta hace un tiempo gracias a un modelo de negocio basado en código abierto, costos de licencias nulos y con servicios de soporte y mantenimiento, cursos y consultaría, JBoss se hizo de una cuota importante en el mercado de los servidores de aplicaciones.

En mayo del 2005 se inicia el proyecto GlassFish que genera, un año después, la primera versión estable. Es en este momento que GlassFish v1 pasa a ser una seria alternativa a JBoss. Desde setiembre del 2007 se cuenta con GlassFish v2 mientras la comunidad trabaja ya en la tercera versión prometiéndose notables mejoras (a modo de ejemplo tiempo de startup menores al segundo).

El código de Sun Java System Application Server fue la base inicial de GlassFish y es la versión certificada y con soporte ofrecida por Sun. Las diferencias relevantes con el GlassFish open source son su instalador grafico específico por plataforma que lo integra con el sistema operativo y la inclusión de distintos conectores a bases de datos.

GlassFish y JBoss son las dos opciones consideradas al momento de la elección del servidor de aplicaciones JEE, luego de investigadas ambas opciones nos inclinamos por la utilización de GlassFish.

A continuación se presentan algunos de los fundamentos en los que basamos nuestra decisión.

- GlassFish cumple con el estándar JEE siendo la implementación de referencia para la generación del estándar.
- Incluye Tomcat como contenedor web
- GlassFish cuenta con plugins para Eclipse y con excelente soporte en NetBeans 5.5
- GlassFish incluye las nuevas librerías de Web Services (JAX-WS 2.0) y es la base de las nuevas plataformas SOA (Service Oriented Architecture) en JAVA
- GlassFish tiene licencia CDDL (tipo Mozilla) y GPL que permite su mezcla con otros proyectos Open Source y otros tipos de licencias como la de Apache, Mozilla, GNU LGPL, GNU GPL, etc.

Enterprise JavaBeans

La tecnología Enterprise JavaBeans (EJB) es la arquitectura de componentes del lado del servidor para la plataforma Java Enterprise Edition (JEE).

Esta tecnología permite desarrollar de forma rápida y simplificada aplicaciones distribuidas, transaccionales, seguras y portables basadas en la tecnología Java.

El objetivo de los EJBs es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables.

Frameworks

El término framework refiriere a una estructura de software compuesta de componentes personalizables e intercambiables utilizables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las piezas faltantes para construir una aplicación concreta. Los objetivos principales que persigue la creación y la utilización de un framework son:

acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

Generalmente un framework incluye programas de soporte, librerías de código y alguna forma sencilla de utilizarlo (en general un lenguaje de scripting o una aplicación). Es usual que al menos una de las partes de un framework sea accesible mediante un API (Applicarion Programming Interface).

A continuación se mencionarán los frameworks más relevantes utilizados en el desarrollo de la aplicación junto con una breve descripción de los mismos.

Quartz

Quartz es un framework open source pensado e implementado con el fin de simplificar y potenciar el uso de tareas planificadas en cualquier aplicación JEE (Java Enterprise Edition) o JSE (Java Standard Edition). Quartz es capaz de manejar miles de tareas independientes con gran eficiencia minimizando el trabajo para los desarrolladores. Estas tareas pueden ser componentes Java estándar o EJBs (Enterprise Java Beans).

Su utilización es sencilla siendo simplemente necesario crear una condición de disparo (trigger) y una tarea a disparar (job); luego basta con asociarlas e ingresarlas en el planificador.

JFreeChart

JFreeChart es un framework Java para el despliegue de información mediante gráficos de calidad. Fue ideado y diseñado para ser utilizado en aplicaciones, applets, servlets y JSPs (Java Server Pages). Se distribuye gratuitamente junto con su código fuente bajo los términos GNU-LGPL (Lesser General Public Licence).

Entre sus características destacamos:

- Un muy bien documentado API que facilita su uso
- Una amplia gama de formatos de gráficas
- Formatos de salida tanto en imágenes (PNG, JPG, etc) como en gráficos vectoriales (PDF, SVG, EPS, etc.)

- Distribución totalmente gratuita (inclusive su código) siendo pago únicamente el manual de uso (no siendo este necesario para su utilización)

Apache Log4J

Log4j es un framework open source cuyo desarrollo fue promovido por el proyecto “Apache’s Yakarta”. Fue desarrollado con el fin de facilitar la inclusión controlada de logueos en aplicaciones. Su velocidad y flexibilidad se basan en mantener las sentencias de logueo empaquetadas de manera de que se pueda habilitar el logueo en tiempo de ejecución sin necesidad de modificar el archivo compilado. Todo esto puede realizarse a un costo de procesamiento razonable.

Comparando con otras opciones para la generación de logs tenemos:

	Log4J	Java Logging API	Jakarta Commons Logging
Niveles de logueo	FATAL, ERROR, WARN, INFO, DEBUG, TRACE	SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST	FATAL, ERROR, WARN, INFO, DEBUG, TRACE
Salidas	Asyncrona, JDBC, JMS, LF5, NTEvent, Null, SMTP, Socket, SocketHub, Syslog, Telnet, Writer	Console, File, Socket, Memory	Depende del framework con el que se utilice
Popularidad	Ampliamente utilizado	Poca	Ampliamente utilizado
Costo / Licencia	Apache License	Incluido en JRE	Apache License

LNSHMI

EL LNSHMI (LonWork Network Server Human Machine Interface) es una librería que posibilita la conexión entre redes LonWorks y aplicaciones desarrolladas en Java. Esto permite implementar aplicaciones, applets, servlets, etc. que puedan interactuar con equipos que implementen el estándar.

El funcionamiento se basa en proveer una interfaz para la conexión de la aplicación Java con el LNS Server. Éste es quién se encarga del manejo de las

comunicaciones con la red LonWorks gerenciando los dispositivos conectados y manteniendo actualizado los valores de las variables en uso.

Integrar plataformas Java a las redes Lonworks posibilita a nivel de usuario, el uso de interfaces basadas en clientes o servicios web. A nivel comercial se puede explotar la posibilidad de integrar en aplicaciones de porte empresarial las funciones de los dispositivos Lonworks.

Anexo IV - Interfaz grafica

La interfaz grafica del sistema es una parte importante del proyecto ya que como lo dice su nombre será la interfaz mediante la que los usuarios accederán a las diferentes posibilidades que brinda la aplicación informática. Nuestra aplicación, en particular, es una aplicación de tipo web, por lo que para desarrollar dicha interfaz se utilizarán herramientas existentes para el desarrollo web.

Los framework web, proveen de estructuras definidas y reutilizables que facilitan la creación de aplicaciones web. En cierto modo, los framework proveen una capa de abstracción sobre la arquitectura original ocultándola o adaptándola para no tener que utilizar el protocolo HTTP (HyperText Transfer Protocol) original de manera nativa y así poder acelerar los tiempos de desarrollo y mantenimiento del código.

Veamos algo de la evolución de los framework web y como, con ello, han avanzado las aplicaciones orientadas a Internet. Al inicio de los noventa nos encontramos con la web estática, el protocolo HTTP solamente se utilizaba para transferir texto almacenado en documentos.

Luego aparece la interfaz CGI (Common Gateway Interface) la cual define una interfaz mediante la cual las aplicaciones y el servidor web pueden comunicarse entre sí lo que permite que cualquier usuario con un cliente web pueda ejecutar programas en el servidor. Esto impulsó la generación de contenidos dinámicos con la aparición de problemas como su difícil escalabilidad, la necesidad de chequear recurrentemente el correcto formato de los parámetros de entrada, entre otros. Otra gran desventaja es lo difícil que resulta separar el contenido estático del dinámico generado por el programa.

Con la aparición de Java y sus servlets se solucionaron varios de los problemas de CGI, como la escalabilidad y el chequeo inicial de los datos, persistiendo el problema de la mezcla entre contenidos estático y el código programado. El desarrollo de JSP (Java Server Pages) introdujo la posibilidad de generar una página HTML permitiendo integrar, mediante etiquetas especiales, código Java en una página web. Con el uso de JSPs es más sencillo y rápido desarrollar y mantener páginas web.

A partir de las primeras especificaciones de JSP es que se describen patrones básicos para la construcción de aplicaciones basadas en esta tecnología. Estos son los llamados Modelo 1 (Fig. 4) y Modelo 2 (Fig. 5) que se presentan a continuación.

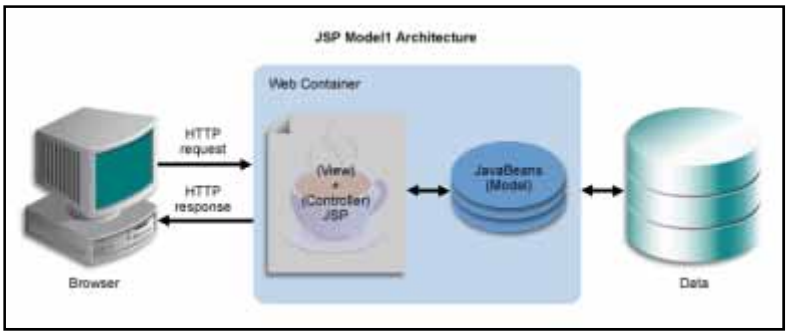


Fig. 4 - JSP Arquitectura Modelo 1

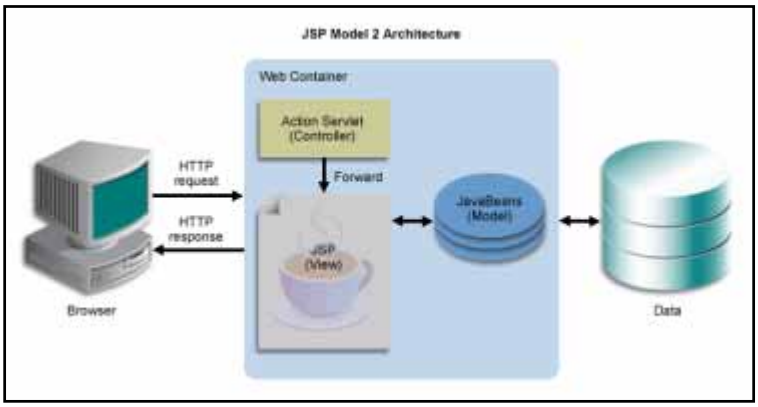


Fig. 5 - JSP Arquitectura Modelo 2

Para el desarrollo en Java de nuestra interfaz disponemos de una amplia gama de frameworks de posible utilización entre los más conocidos tenemos: Struts, JSF, Barracuda, Cocoon, Expresso, Freemarker, Velocity, Webmacro, Maverick, Sitemesh, Yakarta Turbina, Spring y muchos otros. A efectos comparativos nos enfocaremos en solo dos de estos (JSF y Struts) por ser los de más amplia utilización.

Struts es un framework basado en el modelo 2 y consta de un servlet como controlador central que recibe las peticiones de los clientes. A través de este es que se logra la separación entre la vista y la lógica de negocio.

La configuración de la lógica de navegación entre páginas se realiza mediante archivos XML facilitando el manejo. Struts provee una base de tags que permiten embeber fácilmente en código HTML el acceso a beans y la

generación de vistas dinámicas. Soporta internacionalización, extendiendo la funcionalidad que provee java, y brinda mecanismos de validación de entradas.

Por otro lado, Java Server Faces es una especificación (JSR127) aprobada por el Java Community Process (JCP) para construir interfaces de usuario orientadas a aplicaciones que corren en un servidor. Siendo un estándar, existen varias implementaciones del mismo.

El modelo de JSF posee un controlador central que maneja todas las peticiones y gestiona el ciclo de vida basado en un modelo de componentes para la interfaz de usuario.

El ciclo de vida de una petición web en JSF se observa a continuación (Fig. 6).

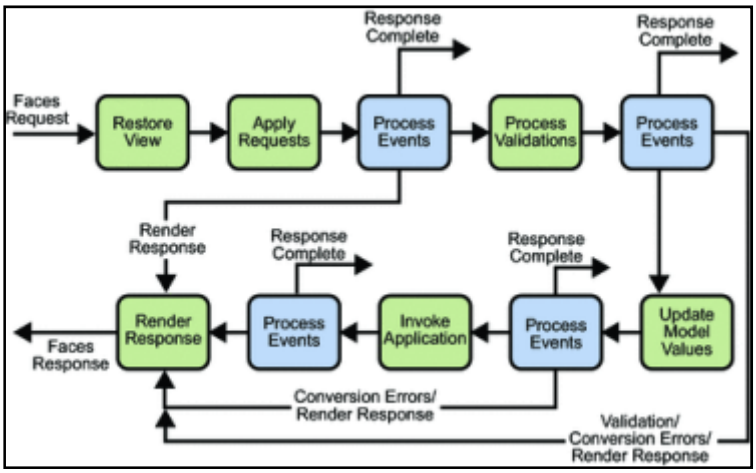


Fig. 6 - Ciclo de Vida de una petición web estándar

JSF posee lógica de navegación entre páginas, soporte de internacionalización y validación de entradas, características también encontradas en Struts. También permite binding de valores, métodos y componentes. Soporta el manejo de eventos mediante listeners y es totalmente independiente del dispositivo de presentación.

Para finalizar resulta ilustrativo realizar una comparación de pros y contras entre ambos frameworks.

	STRUTS	JSF
Pros	- Años demostrando que funciona - Muchos desarrollos de gran envergadura - Buenas prácticas conocidas - Mucho material en la Web - Mucha documentación - Open Source (Licencia Apache)	- Clara separación entre presentación y lógica - Es una especificación (existen varias implementaciones) - No es necesario conocerlo en detalle para utilizarlo - Comunidad y herramientas en aumento
Contras	- No abstrae completamente al desarrollador del protocolo http - No diseñado para la creación de componentes propios - No es natural el mapeo de datos - No es fácil armar vistas independientes del dispositivo	- La creación de componentes propias es compleja - Requiere javascript - Faltan proyectos de gran envergadura

De la tabla anterior se puede observar que ambos tienen varios puntos a favor y en contra no resultando fácil una elección. Finalmente optamos por JSF debido a que al no ser necesario conocerlo en detalle para utilizarlo resulto simple la realización de pruebas con el mismo previo a la decisión.

Anexo V – Base de Datos

Las bases de datos son un capítulo importante de los sistemas informáticos. En esta se almacenan datos que serán utilizables posteriormente de forma sistemática por el software desarrollado. Al enfrentarse con la necesidad del uso de una base de datos y la utilización de la misma por parte de una aplicación desarrollada en Java debemos tomar decisiones referentes a distintos tópicos. Por un lado decidir que modelo de base de datos se utilizara para luego optar por un sistema de gestión de base de datos (SGBD) que satisfaga nuestros requerimientos y por otro lado cual de las alternativas del lenguaje de programación se utilizara para comunicarse con este.

Al optar por el modelo de base de datos a elegir, aparecen dos opciones. Por un lado las base de datos relacionales y por otro las orientadas a objetos.

El modelo relacional es el más utilizado en la actualidad, sus postulados datan de 1970 y su idea principal es el uso de “relaciones”. Este es fácil de entender y de utilizar, en el la información puede ser recuperada o almacenada mediante consultas, con mucha flexibilidad y poder para administrar la información.

Las base de datos orientadas a objetos son bastante recientes y tratan de almacenar en la base objetos completos. Estas incorporan todos los conceptos importantes del paradigma de objetos (encapsulamiento, herencia, polimorfismo, etc.), los usuarios pueden definir operaciones sobre los datos y los programas de aplicación para operar sobre los datos invocando a dichas operaciones.

En la aplicación a desarrollar se opto por la utilización de base de datos relacional. El fundamento de la elección es la experiencia previa del equipo de trabajo con este tipo de bases de datos.

En el momento de optar por SGBD las alternativas MySQL y PostgreSQL fueron las manejadas. Para ambos se encuentra mucha información y se puede obtener distinto tipos de ventajas de la utilización de una u otra. Respecto a las licencias de estos SGBD, nos encontramos con que MySQL tiene doble licencia, GPL y propietario; mientras PosgreSQL tiene licencia BSD.

	MySQL	PosgreSQL
Pros	- Lo mejor de MySQL es su velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento. - Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema. - Utilidades de administración con gran facilidad de configuración e instalación. - Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está. - El conjunto de aplicaciones Apache-PHP-MySQL es de los más utilizados en Internet	- Posee una gran escalabilidad. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta. - Implementa el uso de rollback's, subconsultas y transacciones. - Desde la versión 7.0, claves ajenas. - Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle. - Tiene mejor soporte para triggers y procedimientos en el servidor.
Contras	- Carece de soporte para transacciones, rollback's, subconsultas. - No maneje la integridad referencial. - No considera las claves ajenas. - No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.	- Consume gran cantidad de recursos. - Tiene un límite de 8K por fila, aunque se puede aumentar a 32K, con una disminución considerable del rendimiento. - Es de 2 a 3 veces más lento que MySQL.

En general se puede concluir que, sistemas en los que la velocidad y el numero de accesos concurrentes sea algo primordial, y la seguridad no sea muy importante MySQL es la mejor. En cambio, para sistemas mas serios en las que la consistencia de la BD sea fundamental (BD con información realmente importante, bancos, etc.) PostgreSQL es una mejor opción pese a su mayor lentitud.

Para el sistema que nos encontramos desarrollando optamos por PostgreSQL debido a los requerimientos de seguridad de la aplicación en desarrollo, ítem fundamental para la seguridad del hogar en que se aplique el sistema desarrollado.

Ya optado por el modelo de base de datos y el SGBD, es necesario decidir con que tecnología se desarrollada la interacción entre el software y el SGBD. Dentro de la familia JEE encontramos varias opciones posibles, nosotros optamos por la utilización de Java Persistence Api (JPA), para fundamentar

esta elección se puede hacer un poco de historia, nombrando algunas opciones y sus puntos a favor y en contra según surgieron las mismas.

La primera opción que surge naturalmente es JDBC, un estándar, independiente del SGBD relacional y fácil de usar. Pero de muy bajo nivel, propenso a errores e incomodo de realizar cambios.

Con J2EE Entity beans tenemos un contenedor que maneja persistencia, seguridad y transaccionalidad. Se configura desde fuera del código con XML, pero es complejo, lento (RMI/Corba) y difícil de configurar.

Java data objects (JDO) es un lenguaje de consultas orientado a objetos, aparece con las mismas ventajas que JDBC y suma ser de alto nivel y fácil de aprender. Pero es muy poco aceptado comercialmente.

Una forma de seguir utilizando las bases de datos relacionales desde un sistema orientado a objeto, es con frameworks de persistencia, estos se encargan en hacer un mapeo objeto relacional (ORM). Algunos de los más populares son Hibernate y Toplink (comercial). El problema que resuelven es porque los modelos son similares pero con comportamientos diferentes; y el mapeo no es único, ni es trivial. El mapeo puede automatizarse con información adicional y en estos son hechos en archivos de configuración y la persistencia no es intrusita en los objetos del dominio. Son los antecesores directos de JPA.

JPA aparece en uno de los tres documentos que definen EJB 3.0, nos abstrae de la base de datos y nos permite codificar en términos de objetos aunque provee un lenguaje de consulta muy similar a SQL.

En JPA la configuración está basada en anotaciones (incorporadas en Java 5.0) o en archivos de configuración. Posee mecanismos avanzados para el manejo de ciclos de vida de entidades persistentes.

En conclusión, JPA reúne lo mejor de sus antecesores y estandariza las funciones de ORM de otros frameworks, siendo a su vez de simple utilización y popularidad en aumento.