ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ INTELIGENTE PARA LA GESTIÓN DE RECURSOS EN EL HOGAR MEDIANTE SMART TV, A PARTIR DE MODELOS Y SERVICIOS

AUTOR: SALAZAR ROBLES, ELIZABETH JACQUELINE

DIRECTOR: ING. ALULEMA FLORES, DARWIN OMAR, MSc.

SANGOLQUÍ

2017

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ INTELIGENTE PARA LA GESTIÓN DE RECURSOS EN EL HOGAR MEDIANTE SMART TV, A PARTIR DE MODELOS Y SERVICIOS"**, realizado por la señorita ***ELIZABETH JACQUELINE SALAZAR ROBLES,*** ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar a la señorita ***ELIZABETH JACQUELINE SALAZAR ROBLES*** para que lo sustente públicamente.

Sangolquí, 16 de Agosto del 2017

ING. DARWIN OMAR ALULEMA FLORES

DIRECTOR

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, ***ELIZABETH JACQUELINE SALAZAR ROBLES*****,** con cédula de identidad Nº 1718291386 declaro que este trabajo de titulación **"DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ INTELIGENTE PARA LA GESTIÓN DE RECURSOS EN EL HOGAR MEDIANTE SMART TV, A PARTIR DE MODELOS Y SERVICIOS"** ha sido desarrollado considerando los métodos de investigación existentes, así como también respetando los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ellos me declaro responsable del contenido, veracidad y alcance de la investigación mencionada.

Sangolquí, 16 de Agosto del 2017

ELIZABETH JACQUELINE SALAZAR ROBLES

CC. 1718291386

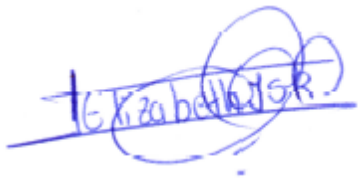
DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, ***ELIZABETH JACQUELINE SALAZAR ROBLES***, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de mi titulación **"DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ INTELIGENTE PARA LA GESTIÓN DE RECURSOS EN EL HOGAR MEDIANTE SMART TV, A PARTIR DE MODELOS Y SERVICIOS"** cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Sangolquí, 16 de Agosto del 2017



ELIZABETH JACQUELINE SALAZAR ROBLES

CC. 1718291386

DEDICATORIA

Deseo dedicar este trabajo de titulación especialmente a mis padres, quienes me han apoyado incondicionalmente, incluso cuando quise desistir con su sabiduría me hicieron entender que caerse no significaba rendirse, que tenía que levantarme y seguir adelante hasta alcanzar el objetivo tan anhelado, sin importar el tiempo pero siempre con empeño, esfuerzo y honestidad.

A mis hermanos, que han sido el impulso para hacer las cosas bien y convertirme en un modelo a seguir por ellos.

A mi tío José quien me brindó sus palabras de apoyo y estuvo siempre pendiente de mí, ofreciéndome su ayuda cuando la necesitara.

A Andrés Sosa que llegó a mi vida en un momento difícil de mi carrera, y quien con su paciencia y sabiduría supo apoyarme y guiarme basado en su experiencia.

A los docentes quienes impartieron sus valiosos conocimientos en mí.

A mi tutor, quien me brindó las bases necesarias y suficientes para que pueda desarrollar este trabajo de titulación, y además me guió durante el proceso.

A mis amigos y compañeros de estudio, que me supieron ayudar, aconsejar y escuchar en los momentos oportunos.

Elizabeth Jacqueline Salazar Robles

AGRADECIMIENTO

Quiero agradecer a mis padres que me acompañaron hasta altas horas de la noche para que no trabajara sola, y haciendo sacrificios de todo tipo me facilitaron las herramientas necesarias para poder culminar el trabajo de titulación.

A mi hermano Diego quien me ayudó y explicó más profundamente con temas de programación, que requerían más especialización.

A mi hermano Roger que me ayudó de la manera que más pudo sin importar la hora.

A Andrés Sosa quien estuvo durante todo el proceso de desarrollo de este trabajo de titulación apoyándome intelectual y financieramente cuando lo necesitaba, sin perder la confianza en mí.

A mí tutor por presionarme para que no deje pasar más tiempo y culmine lo antes posible con el trabajo de titulación.

A los Docentes que constantemente estuvieron pendientes y preocupados por el proceso de desarrollo de mi trabajo de titulación.

Elizabeth Jacqueline Salazar Robles

ÍNDICE

ÍNDICE DE TABLAS

INDICE DE FIGURAS

RESUMEN

El presente proyecto documenta el proceso de desarrollo de la aplicación SmartHomeTV. Esta aplicación gestiona los recursos del hogar a través de una interfaz inteligente basada en modelos y servicios, su composición es software desarrollado en Android, y hardware implementado en Arduino. La comunicación entre la interfaz y el servidor web se establece a través del uso de un servicio web que obtiene los valores de los registros almacenados en una base de datos, y los envía a la aplicación en Android, y esta a su vez envía los datos al hardware, el mismo que está constituido por el módulo shield de Ethernet montado sobre la tarjeta Arduino. Arduino se encarga de controlar las salidas hacía los actuadores conectados a las luces, persianas y ventilación, y además de monitorear las entradas de los sensores conectados a las puertas y ventanas. La transmisión y recepción de los datos se realiza utilizando el protocolo de transferencia de hipertextos (HTTP) tanto entre el servidor web y Android como entre Android y Arduino. Esta aplicación está diseñada para instalarse y ejecutarse en televisores inteligentes con sistema operativo Android, que integran características básicas como conexión a internet, para que puedan hacer uso del protocolo HTTP para su comunicación introduciendo en ellos el concepto de internet de las cosas, cambiando la percepción habitual de los usuarios. SmartHomeTV es una novedosa herramienta para que las personas gestionen los recursos de sus hogares, dejando atrás el concepto cotidiano y convirtiendo estas actividades en una nueva experiencia, sencilla, práctica y desde sus televisores.

Palabras claves:

- **SMART TV**
- **ARDUINO**
- **ANDROID**
- **INTERFACES INTELIGENTES**
- **DESARROLLO DIRIGIDO POR MODELOS**

ABSTRACT

This project documents the development process of the SmartHomeTV application. This application manages the resources of the home through an intelligent interface based on models and services, its composition is software developed in Android, and hardware implemented in Arduino. The communication between the interface and the web server is established through the use of a web service that obtains the values of the registers stored in a database, and sends them to the application in Android, and this in turn sends the data to the hardware, which is constituted by the shield module of Ethernet mounted on the Arduino card. Arduino controls the outputs to the actuators connected to the lights, blinds and ventilation, as well as monitoring the inputs of the sensors connected to the doors and windows.
The transmission and reception of the data is done using the hypertext transfer protocol (HTTP) between the web server and Android as well as between Android and Arduino. This application is designed to be installed and run on Smart televisions with Android operating system, which integrate basic features such as Internet connection, so that they can make use of HTTP protocol for their communication by introducing in them the concept of internet of the things, changing the perception of users. SmartHomeTV is a new tool for people to manage the resources of their homes, leaving behind the everyday concept and making these activities a new experience, simple, practical and from their TVs.

Keywords:

- **SMART TV**
- **ARDUINO**
- **ANDROID**
- **INTELLIGENT INTERFACES**
- **MODEL DRIVEN DEVELOPMENT**

CAPÍTULO I

INTRODUCCIÓN

1.1. Antecedentes

En los últimos años la tecnología ha ido avanzando en el desarrollo de nuevas formas para captar la atención del público, es el caso de la televisión que dejó atrás el concepto de sólo visualización y evolucionó a la interoperabilidad e interactividad. La televisión inteligente está generando grandes cambios en la perspectiva del usuario, la principal razón es que integra la televisión, las plataformas de entretenimiento, permite la socialización web y todo a través del Internet. (Significados, s/f)

Es así que los Smart TV han ido ganando popularidad, al punto de creer que serán el siguiente gran suceso. Entre las principales características y servicios están (Saraguro, 2014):

- Conectividad a internet cableada o por Wi-Fi.
- Vídeo por streaming.
- Navegación e Interfaz de usuario.
- Puertos de conexión HDMI o USB.
- Capacidades de procesamiento altas
- Navegación Web.
- Ejecución de aplicaciones.
- Control de ciertas aplicaciones a través del reconocimiento de gestos, voz y facial.

Por otro lado, existen numerosos dispositivos capaces de realizar streaming, aquellos que permiten proyectar contenido de aplicaciones de un celular, tablet o computador a un televisor, como Apple TV y Chromecast, utilizando funciones denominadas AirPlayFire y Cast respectivamente, o los

dispositivos más avanzados como Roku o Fire TV que ofrecen una variedad de canales libres o por suscripción, haciendo posible que los usuarios que posean televisores antiguos, con las características básicas requeridas para la adaptación de estos dispositivos de streaming no deban deshacerse de ellos, al contrario, al asemejarlo a un SMART TV, tengan la posibilidad de acceder a otro tipo de funcionalidades, por ejemplo, tener la capacidad de instalar y correr diferentes aplicaciones dedicadas. (Moskovclak, Katzmaler, 2014)

La televisión inteligente puede ser usada como un componente innovador, que permita al usuario satisfacer la necesidad de controlar y monitorear los recursos de su hogar, de forma que las tareas habituales se realicen a través de elementos tecnológicos, facilitando así su vida cotidiana y creando un ambiente mucho más confortable y seguro.

Es así como se pensó en el desarrollo de un sistema que se basa en una arquitectura que permita gestionar una interfaz que se ejecute en un Smart TV con sistema operativo Android, basándose en ingeniería de modelos para implementar la interfaz inteligente, y que además se adapte a las prácticas más comunes del usuario.

1.2. Justificación e Importancia

Los avances tecnológicos marcan cambios significativos, y acoplarse a ellos representa un precedente, es por eso que el Gobierno Ecuatoriano se está enfocando en crear una cultura digital y en el cambio de los procesos que a diario realiza, es así como generó el Plan Nacional de Gobierno Electrónico con el que busca modernizar el Estado, una de sus principales herramientas son las Tecnologías de la Información y las Comunicaciones (TICs), como menciona Proaño en el boletín Estadístico del Sector de las telecomunicaciones de la ARCOTEL (2015) las TICs representan una contribución para que la industria y la sociedad puedan observar el entorno tecnológico, las oportunidades vinculadas con el sector y la información para potenciales investigaciones en aspectos regulatorios, económicos y técnicos.

Además, el internet es el principal requerimiento del streaming, el mismo que tiene cifras de acceso a nivel nacional del 28,3% y varían de acuerdo a la población, Urbana 37% y Rural 9.1%, según las cifras del INEC en el año 2013. Es importante mencionar también que según el INEC, en el Ecuador el 94% de la población posee un televisor en su hogar, y el tiempo que dedican a este medio es de más de 12 horas a la semana.

Sumado el alto índice de penetración de la televisión con el acceso a internet, y la constante innovación orientada a las aplicaciones telemáticas, que pueden ser explotadas de manera que se generen soluciones, como por ejemplo, sistemas de protección para hogares, surge la necesidad de desarrollar aplicaciones que permitan esta integración y uno de los mecanismos para el efecto es emplear interfaces inteligentes, que a partir de un modelo abstracto y mediante aprendizaje de los requerimientos del usuario creen interfaces capaces de satisfacer las necesidades del usuario.

1.3. Alcance del proyecto

Se investigó sobre la tecnología Smart TV y las herramientas que permiten el desarrollo de aplicaciones, y se recopiló la información suficiente para el diseño y ejecución del proyecto.

Se estudió la ingeniería de modelos e interfaces inteligentes obteniendo así el conocimiento necesario a ser aplicado.

Se investigó el tipo de señales con las que trabajan los sistemas de domótica para acoplar adecuadamente las mismas dentro del sistema a desarrollar.

Se determinó el modelo para el desarrollo de una interfaz que puede ser adaptada a diferentes ambientes, cubriendo así las necesidades de los diferentes posibles usuarios.

Se diseñó una interfaz adaptativa, interactiva y que permite el acceso remoto, para que el usuario gestione los recursos de su hogar de una forma efectiva y sencilla.

La estación base está diseñada con los ambientes básicos que incluye un hogar habitable.

La implementación del sistema integra la aplicación con su interfaz y la estación base que contiene los actuadores y sensores que permiten visualizar de forma física y a través de un Smart TV los recursos gestionables del hogar.

El funcionamiento del sistema es determinado usando pruebas, y un análisis que incluye diferentes escenarios reales mediante los cuales se determina su eficiencia.

1.4. Objetivos

1.4.1. Objetivo general:

- Diseñar e implementar una interfaz inteligente para la gestión de recursos en el hogar, usando Smart TV a partir de modelos y servicios.

1.4.2. Objetivos específicos:

- Investigar la tecnología Smart TV y sus herramientas de desarrollo para la recopilación de información.
- Estudiar la ingeniería de modelos.
- Estudiar el desarrollo de interfaces inteligentes basado en componentes inteligentes.
- Determinar un modelo de interfaz inteligente.
- Diseñar el módulo de hardware.
- Implementar el sistema integrando la estación base con la interfaz adaptativa para su visualización.
- Documentar el proyecto.
- Ejecutar pruebas para validar el sistema.

1.5. Estado del arte

1.5.1. Smart city

En (Wang, De, Zhou, Huang y Moessner, 2017) indican que la tecnología IoT (Internet of Things) aumenta cada año con más rapidez, es decir, que el número de las “Cosas” altamente conectadas a Internet aumenta y aumenta, generando el requisito de tener interoperabilidad y escabilidad para el análisis, almacenamiento e integración de datos. Según los autores la forma más óptima de obtener los datos de los sensores es descubrir los servicios de sensores que proporcionen las funcionalidades necesarias y luego suscribirse a esos servicios, de esta manera, al tomar los datos se permite la creación de aplicaciones IoT, y se puede exponer a los sensores como servicios web o REST.

Desarrollan una plataforma de descubrimiento de servicios de sensores semánticos basada en la indexación espacial y la búsqueda semántica con la técnica aplicada para la inteligencia distribuida en el caso de estudio de la contaminación de aire urbano, centrado en redes de datos de sensores una red de sensores inalámbricos.

Este trabajo permite visualizar la importancia de tener todos los dispositivos conectados entre sí, utilizando la tecnología IoT, la misma a la que se hace referencia en el proyecto SmartHomeTV.

En (Pribyl, Lom y Pribyl, 2017) se hace énfasis en la importancia de utilizar herramientas de modelado, para lo que sus autores desarrollan SMACEF, una plataforma para planificar y evaluar estrategias de Smart cities, cuyas ventajas son la modularidad, la extensibilidad, la adaptación individual de los modelos y el enfoque en los resultados, que son la clave que influyen en el sistema. Este trabajo lo aplican en la Plaza Carlos de Praga.

Su objetivo es abordar los escenarios desde la perspectiva de los usuarios finales en lugar de proporcionar una visión general de la tecnología y sus capacidades.

Esto hace contraste al hecho de que realizar modelos, es una gran solución para mucho de los inconvenientes al momento de analizar, planificar y desarrollar software, una de las características que se abordan para el desarrollo de SmartHomeTV.

En (Bangui, Buhnova, Rakrak y Raghay, 2017) realizan un análisis de las nubes y las tecnologías, para diseñar sistemas de comunicación inalámbrica, que aporten al crecimiento de la ciudad inteligente, con el concepto de IOT. Indican que Li-Fi, al basarse en LEDs para la transferencia de datos permite resolver el problema de costos al momento de implementar una Smart city. Li- Fi es una tecnología inalámbrica eco-óptica que describe la comunicación entre la luz óptica y la inalámbrica de alta velocidad. Además mencionan que se debe utilizar una nube para almacenar los datos recogidos por Li-Fi para optimizar los servicios, sin embargo Li-Fi tiene sus limitaciones, por lo que mientras se logra quitar las restricciones de esta tecnología es mejor utilizar otras para implementar las ciudades inteligentes.

Al igual que en este trabajo, para SmartHomeTV se realiza el análisis de las tecnologías, optando por el uso de una red cableada para la interconexión de los recursos dentro del hogar, evitando así las restricciones, o interferencias.

Todos estos trabajos sobre Smart cities hablan de la automatización de los recursos de las ciudades para convertirlas en inteligentes, permitiendo que se sustenten por sí solas, por lo tanto, se debe partir desde la automatización de los recursos del hogar utilizando redes de comunicación seguras y confiables como las que se utiliza en el presente proyecto.

1.5.2. Smart home

En (Nisar, Ibrahim, Wu, Adamo y Deen, 2016) se documenta el desarrollo de un modelo para casa inteligente basado en las tecnologías de la información (TICs), utilizando la plataforma Android y las redes de sensores wireless para adquirir e integrar los datos en tiempo real y de forma segura. Los autores utilizan el algoritmo de observación OLA que combina los resultados de los sensores inalámbricos y los conceptos de inteligencia artificial, y dividen el proyecto en tres módulos principales, módulo de sensor, módulo de control y módulo de actuador. Especifican que esta aplicación es para ser utilizada en hogares de ancianos, permitiendo mejorar su estado de vida y colaborar con su salud. Al parecer este trabajo se asimila a SmartHomeTV, sin embargo, se utiliza dispositivos móviles para ejecutar la aplicación, es decir, aún no se habla de Smart TVs.

En (Krsriyanto y Dwi, 2016) se diseña un prototipo para hogar inteligente utilizando la tarjeta Arduino 2560 y la tecnología Ethernet gracias al módulo shield de Ethernet de Arduino. Los dispositivos electrónicos son controlados por Arduino, que tiene programado un servidor web con su respectiva dirección IP, esta dirección IP redirige al usuario al sitio web para que pueda monitorear desde ahí los elementos del hogar. Esta aplicación se ejecuta sobre dispositivos móviles, que son los encargados de enviar los datos a través de una red local al Arduino utilizando un enrutador conectado con el módulo shield de Ethernet.

Esta investigación no se centra en el diseño de una interfaz inteligente especial para el control de los recursos; SmartHomeTV se centra en el diseño de la interfaz inteligente para que se adapte al posible uso de diferentes usuarios.

En (Anuebunwa, Rajamani, Pillai y Okpako, 2017) se evalúa las capacidades de participación de los usuarios en las actividades que se pueda realizar a través de las aplicaciones para smart home. Utilizan lógica

difusa para entender los comportamientos de los consumidores y así mejorar el diseño y la planificación de una sólida y eficaz red de micro-red que interconecte los dispositivos en el hogar.

Según esta investigación, los usuarios o consumidores finales son el ente más importante en el proceso de desarrollo de software y hardware para hogares inteligentes, por lo tanto es de vital importancia entender en primer lugar sus necesidades, es así que, en SmartHomeTV se procura realizar un modelo que sea lo más real posible al entorno en el que habita el usuario.

El convertir un hogar normal en un Smart Home permite que los usuarios se sientan más cómodos, pero es necesario que se identifique claramente los subsistemas que intervienen para un eficiente desempeño en la gestión de los recursos dentro del hogar, tal como se lo realiza en el presente trabajo, identificando y detallando cual es la función a cumplir por cada subsistema.

1.5.3. Smart TV

En (Sun, Li, Wang, Li, Ma, Xu y Chen, 2015) exponen el desarrollo de la aplicación S3TV, para explotar las capacidades de los Smart TVs. El prototipo S3TV, utiliza comandos de gestos predefinidos y fáciles de aprender para reproducir contenido en la pantalla del televisor. El uso de un teléfono móvil con Android es indispensable, puesto que en este dispositivo se ejecuta la aplicación de control. Con este software logran reducir el desplazamiento visual de la pantalla del televisor, al no ser necesario ver la pantalla del móvil.

El desarrollo del software lo hacen en Android por ser uno de los sistemas operativos actuales que ha ido ganando gran popularidad. Este proyecto en sí, lo que hace es reemplazar el control remoto normal del Smart TV por uno más sofisticado, en cambio, lo que SmartHomeTV plantea es la implementación de un aplicación que pueda controlar los recursos del hogar

desde una aplicación propia para un Smart TV, desarrollada usando el mismo sistema operativo Android.

En (Kim, Jung, Lee, y Ryu, 2015) proponen un marco de control de electrodomésticos desde un decodificador de TV inteligente, utilizando una aplicación Web con protocolo WebSocket, que tenga registrada la ubicación de la red. Para esto, todos los electrodomésticos deben estar conectados a la red con el protocolo de comunicación serie RS-485.

A este trabajo le hace falta exponer el software de desarrollo y el controlador que se va a utilizar, o en su defecto especificar claramente el modelo independiente y adaptable que se podría utilizar para el desarrollo de la aplicación. En la documentación de SmartHomeTV se utiliza modelos que hagan que no sea una dependencia el entorno para desarrollar el software ni el medio para construir el hardware.

En (Perakakis y Ghinea, 2015) se expone que en la actualidad los Smart TV son dispositivos con un nivel alto de adquisición, sin embargo, aún no se ha logrado encontrar una forma óptima de navegación web para estos dispositivos, lo que genera una molestia en los usuarios, para esto, los autores proponen como solución a esta problemática el usar diseño web responsivo, que integre scripts de JavaScript para mejorar la interacción de los usuarios con la web, y además que permitan ser multiplataforma.

Como se puede notar el desarrollo de aplicaciones para Smart TV, es aún un campo extenso de exploración. La investigación de SmartHomeTV, incluye varios temas para la integración todos los componentes que sirven para mejorar las experiencias del usuario con el Smart TV.

La tecnología Smart TV tiene algunas desventajas, como el hecho de que los dispositivos trabajan con diferentes sistemas operativos, por lo que a diferencia de lo que realizan los trabajos de investigación consultados, en

este proyecto se procura incluir la mayoría de dispositivos posibles al elegir el sistema operativo Android.

1.5.4. Interfaces inteligentes

En (Jia y Zhao, 2015) se propone e implementa un tipo de interfaz de datos inteligente del internet de las cosas que soporta múltiples patrones, y puede ser usado para procesar automáticamente los datos de la capa de percepción y fusionarlos con los datos sin procesar de todo tipo de sensores, obteniendo un formato de datos unificados; esto se puede expresar como un diseño de arquitectura de software basada en red.

Como se observa, el internet de las cosas está en auge, la tendencia de tener interconectados los dispositivos entre sí, y poder obtener los datos de los sensores para manejarlos al gusto del usuario final, hace necesario el desarrollo de interfaces inteligentes, como es el caso de SmartHomeTV que genera la interfaz visual basada en los datos entregados por el usuario y almacenados en la base de datos.

En (Hedan, Xinyu y Chengen, 2010) se propone un sistema de interfaz inteligente implementada en Microsoft Visual C ++ 6.0, y para la gestión de los datos heterogéneos se utiliza una base de datos en Microsoft SQL Server 2000, con esta interfaz los datos de diferentes tipos y formatos pueden ser especificados, presentados y gestionados de manera unificada. La aplicación consta de dos módulos, el módulo de análisis de datos que convierte archivos de entrada en formato XML, y el módulo de visualización de datos que toma los archivos XML como su entrada para ser presentados y en la interfaz inteligente. De esta forma se tiene un enfoque que puede mejorar en gran medida la eficiencia de producción de los diseñadores de ingeniería.

Esta investigación enfatiza el desarrollo de software que brinde como resultado interfaces inteligentes, para mejorar la generación del software, tal

y como se lo hace en SmartHomeTV, a pesar de que la aplicación que se plantea en este trabajo es más completa, desde el proceso de modelado hasta su implementación.

Las interfaces inteligentes que se presentan en los trabajos investigados, no alcanzan las tecnologías actuales y tampoco muestran un desarrollo ordenado de software, a diferencia de este proyecto en el que se diseña el CIM y PIM con la utilización de la ingeniería de modelos para posteriormente realizar el código, de forma que la interfaz sea perdurable.

CAPITULO II

MARCO TEÓRICO

2.1. Conceptos básicos

2.1.1. Smart city

Una Smart city es una ciudad que emplea las Tecnologías de la Información y de la Comunicación (TICs) en conjunto con los recursos para emplearlos eficientemente, fomentando el crecimiento sostenible a fin de mejorar la calidad de vida de los ciudadanos que conviven en ella.

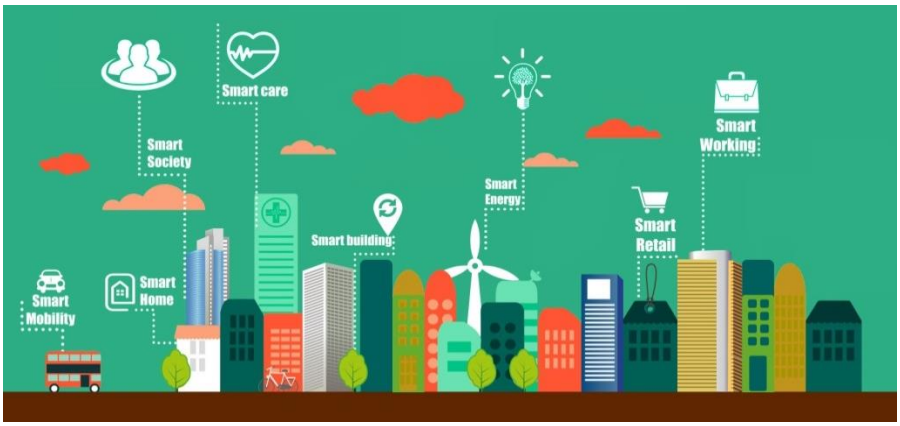


Figura 1. Smart city
Fuente. (Smart Lighting, 2017)

Se considera como un modelo ideal a una Smart city cuando se fundamenta primordialmente en los siguientes subsistemas (Endesa Educa, 2016):

- Generación distribuida: La ciudad inteligente debe poseer generación eléctrica distribuida por su territorio, en el que el abastecimiento sea individual y no centralizado.
- Smart grids: Hace referencia a las redes inteligentes interconectadas, que mantienen una comunicación de datos de ida y retorno entre el el usuario y el centro de control.
- Smart metering: El consumo energético de cada uno de los usuarios es medido de forma inteligente, por medio de telecontadores que realizan las lecturas en tiempo real y a distancia.
- Smart buildings: Los edificios deben ser inteligentes y poseer sistemas propios de producción de energía, respetando el medio ambiente.
- Smart sensors: Tiene la función de recolectar todos los datos suficientes para convertir a la ciudad en una Smart city. Su participación es primordial para conservar la conectividad en la ciudad y además permanecer informada y ser capaz de ejecutar las funciones de cada subsistema con éxito.
- eMobility: Se integra el vehículo eléctrico, y se coloca puestos de recarga públicos y privados.
- Smart citizen: La parte más importante de una Smart city son sus ciudadanos, ya que sin su participación activa, no tiene sentido llevar a cabo las iniciativas.

Las ventajas que generan las ciudades inteligentes son (Economipedia, s/f):

- Contribuir con el cuidado del medio ambiente.
- Economizar costos a los ciudadanos
- Mejorar los servicios públicos.
- Aportar en el proceso de trasparencia de la administración de los municipios.
- Conservar la inversión empresarial y llamar la atención de nuevos inversionistas.

- Aumentar la comunicación entre los ciudadanos.

En un futuro se espera que para el año 2050 al menos un 85% de la población mundial habite en ciudades, por lo que sus ciudadanos tendrían que afrontar la problemática de aumento de emisiones de Co2, falta de abastecimiento energético, tráfico automovilístico, escases de provisiones de materias primas, manejo adecuado de servicios sanitarios y control de la seguridad. Es por ello, que desde ahora se debe fomentar el desarrollo de tecnologías inteligentes que se puedan implementar poco a poco dentro de las ciudades. Algunos ejemplos de ciudades inteligentes son (Endesa Educa, 2016):

- Barcelona (España)
- Málaga (España)
- Búzios (Brasil)
- Santiago de Chile (Chile)
- Bogotá (Colombia)
- Montevideo (Uruguay)
- Tokio (Japón)
- Londres (Inglaterra)
- Nueva York (Estados Unidos)
- Zurich (Alemania)
- París (Francia)

Adicional, en otras ciudades alrededor del mundo existen proyectos que pretenden convertir sus entornos en ciudades inteligentes.

2.1.2. Smart home

Un Smart home, es un ambiente doméstico, que ha sido parcialmente automatizado, utilizando un control centralizado para las luces, ventilación, aire acondicionado, entre otros.

La historia de la domótica se origina en el año de 1975, la compañía Pico Electronics desarrolló y patentó la tecnología X10, cuyo fin era transmitir una señal de 120kHz a través de la línea eléctrica. Posteriormente en el año 1978 lanzaron sus primeros productos al mercado, y fueron implementados especialmente en edificios.

La década de los 80 fue el impulso para la automatización de los hogares, en el año 1983 Murata, Namekawa y Hamabe sugirieron un prototipo de estandarización para tener compatibilidad entre los sistemas de los desarrolladores domóticos en Japón. Un año después, siete fabricantes de estos sistemas lograron un acuerdo para la estandarización, ellos plantearon una estructura denominada HBS basada en tres bandas, banda base, para controlar las señales, subbandas, para las señales de datos de alta velocidad, y banda FM/TV, para la información visual. Como resultado de esta propuesta, en 1985, et.al, un producto basado en HBS fue desarrollado, este se constituía de cuatro subsistemas (Lips, 2012):

1. Subsistema de control del monitor de habitación: Controlaba las funcionalidades domésticas y de seguridad.
2. Subsistema telefónico: Para el control del teléfono y alarma de seguridad.
3. Subsistema de telecontrol: Controlaba los dispositivos y sensores de seguridad mediante los teléfonos que se ubicaban en la casa.
4. Subsistema de control de vídeo interno: Recibía señales de vídeo desde el receptor de imagen.

La Asociación de Industrias Electrónicas también trabajaba en los inicios de los años 80 en su propio proyecto, Consumer Electronic Bus (CEBus), diseñado para que los dispositivos electrónicos tengan la capacidad de interactuar con los productos de otros fabricantes a través de distintos canales de medios. Este sistema contaba con un modelo de red de cuatro capas (Lips, 2012):

1. Capa de aplicación: El dispositivo solicita la generación de un protocolo de datos de la capa de aplicación (APDU) y esto se transmite a la capa de red.
2. Capa de red: Recibe la información de la APDU y la convierte en un protocolo de datos de capa de unidad de red (NPDU). Luego la envía a la subcapa de control de enlace lógico (LLC) que es una parte de la capa de enlace de datos.
3. Capa de enlace de datos: La subcapa de control de enlace lógico agrega un encabezado con servicio adicional de información al NPDU para hacer una LPDU e invoca los servicios de control de acceso al medio (MAC).

   Control de acceso al medio: Aquí la LPDU se convierte en un MDPU y se envía a la capa física.
4. Capa física: Recepta las señales de video desde la pantalla de un televisor.

En el año 1988, un grupo denominado HES (Home Electronic System) se concentró en detallar los estándares para los hogares automáticos, y más tarde, en 1992 HES incluyó siete sistemas de control de hogares.

Para el año 2000 una de las propuestas de Sriskanthan, Tan y Karande fue incluir Bluetooth a un sistema de domótica, entre sus criterios de desarrollo, Bluetooth cubría las necesidades básicas, operaba en la banda no licenciada de 2,4GHz, y permitía interconectar dispositivos con un rango de cobertura de 10 metros, a este sistema lo denominaron HAP (Home Automation Protocol), consistía en un controlador de host implementado en un computador personal y en un microcontrolador que se comunicaba con el usuario a través de Bluetooth.

Todos estos inventos dieron pasos para que en posteriores años surgieran más opciones de sistemas domóticos, por ejemplo (Lips T, 2012):

- 2004 A. Alheraish propuso una solución M2M (máquina a máquina, hombre a máquina o móvil a máquina) basada en la red de comunicación celular GSM.

- 2005 A. R. Al-Ali y M. AL-Rousan presentaron un sistema domótico basado en Java.

Posterior al 2005, el diseño de los hogares inteligentes ha ido evolucionando significativamente de la mano de las nuevas tecnologías, como ZigBee o Wi-Fi.

En el 2006 se revolucionó el mercado de la automatización, gracias a un grupo de estudiantes italianos que introdujeron un microcontrolador de una sola placa y de código abierto, a la que denominaron Arduino, herramienta que tiene un sin número de fines, pero los desarrolladores de sistemas de hogares inteligentes, encontraron la forma de integrar esta placa a sus prototipos, haciéndolos más sencillos, pero sobretodo menos costosos. Es así como se creó en el 2009 DomotiHome.

Figura 2. Smart home

Fuente. (ChinaTechNews, 2016)

De igual forma, Google decidió introducirse en el mercado de la automatización con aplicaciones de Android que permiten la comunicación con dispositivos electrónicos y del hogar, su proyecto expuesto en la Conferencia de Desarrolladores de E/S de Google del año 2011, Android@Home, se basó en SNAP de Synapse Wireless (Lips, 2012).

Es así que la visión a futuro de los hogares inteligentes prevé que sea posible tener conectados todos los dispositivos electrónicos entre sí, manteniendo una comunicación entre ellos que permita brindar funcionalidades rápidas, confiables, seguras y en tiempo real.

2.1.3. Smart TV

La revolucionaria era tecnología permite visualizar los grandes cambios que se ha hecho con el paso de los años, a los conceptos de percepción que se tenía antes, un ejemplo claro, es la transformación de la televisión normal a una televisión que ofrece un variado número de prestaciones, entre ellas, capacidad de conexión a internet, comunicación con otros dispositivos electrónicos dentro de la misma red, ejecución de aplicaciones de ocio, entrenamiento y dedicadas que pueden ser descargadas desde la web.

Tabla 1.

Reseña histórica de la televisión

Año	Descripción
1884	Alemán Nipkow obtiene la patente del disco de exploración lumínica.
1906	Boris Rosig considera que con el tubo de rayos catódicos se puede proyectar imágenes de televisión.
1911	Boris Rosig logra producir únicamente imágenes deficientes.
1924	John Baird efectua las primeras pruebas de imágenes televisivas.
1925	Vladimir Zworykin obtiene la primera patente de televisión.
1926	John Logie Baird efectua la primera demostración pública de televisión.
1937	En Alemania los técnicos diseñan su sistema de televisión con 441 líneas de definición, Italia de igual forma crea un sistema de televisión de 441 líneas y la URSS opta por las 343 líneas.
1939	Estados Unidos fabrica dispositivos receptores televisivos para uso doméstico.
1947	La FCC establece, estandariza las emisiones televisivas con una normalización técnica
1953	Estados Unidos adopta el sistema de televisión en color denominado NTSC (National Television System Committee), que consistía en 496 líneas de definición.

Continua

1960	Sony fabrica la primera televisión portátil.
1998	La primera pantalla plana de plasma producida por Philips, se presenta en Estados Unidos.
2009	La empresa Samsung anuncia el primer televisor LED.
2011	Se introducen en el mercado las primeras Smart TV's.
2014	La compañía LG lanza el televisor OLED, de resolución 4K.
2016	La tecnología Quantumm dot, es presentada por Samsung.
2017	LG presenta un televisor cuya pantalla tiene una dimensión de 2,57mm, el LG SIGNTURE OLED TV W7.

La televisión posee una extensa trayectoria evolutiva, que se origina en los años 90 y se remonta hasta la actualidad, en la Tabla 1., se presenta una breve reseña (Juste, 2017).

La tecnología popular actual, Smart TV, asemeja su arquitectura a la de los dispositivos móviles, y consta de tres capas (Saraguro, 2014):

1. Capa Engine: Contiene los drivers y componentes de nivel bajo adheridos al sistema operativo, que permiten el control de bases de datos, gráficos, redes, broadcastig, entre otros.
2. Capa Intermedia: Permite la gestión de los elementos primordiales que suministra la plataforma para el crecimiento de aplicaciones tipo browser para la visualización de contenido web, multimedia y control DRM (Digital Right Managment).
3. Capa de Aplicación: Incluye a las aplicaciones finales que se visualiza en la pantalla del Smart TV.

En el mercado existe una gran variedad de marcas como Samsung, Sony, LG, Panasonic, Philips que ofertan Smart TV's, lo que ha generado una gran popularidad y adquisición de estos productos, llevando a los desarrolladores de software a interesarse en la creación de aplicaciones para televisión, convirtiendo a este proceso en una herramienta explotable para los negociantes en los años venideros.

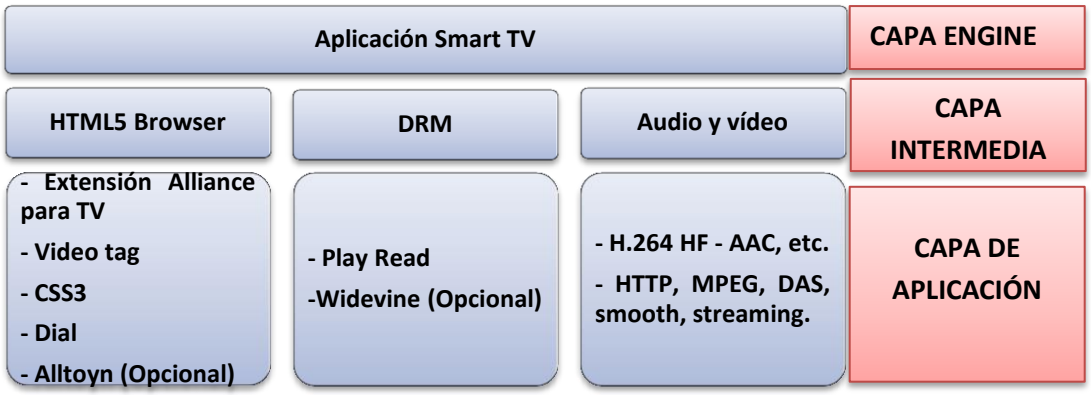


Figura 3. Arquitectura de Smart TV

Fuente. (Saraguro, 2014)

Se puede destacar a Samsung, pues ofrece a los desarrolladores un SDK con un IDE integrado basado en Eclipse y un emulador como máquina virtual, que desde su versión 5.1 hace posible crear aplicaciones para Smart home. Algunos de sus Smart Tv tienen integrado el reconocimiento por voz, que permite al usuario controlar el dispositivo a través de comandos predefinidos de voz. El control por gestos, es otra de las funcionalidades que incluyen, permitiendo que mediante comandos de movimientos básicos se puede manejar el televisor de acuerdo a las necesidades.

Si se parte de este punto Samsung se planteó la idea de proyecto Smart Home cuyo principal objetivo es que todos los dispositivos en el hogar estén conectados entre sí y puedan ser controlados desde un único punto. Los dispositivos electrónicos que planeó gestionar son electrodomésticos, aire acondicionado e iluminación, para que en conjunto estén conectados mediante una aplicación en el SMART TV, Smartphone o Tablet, poniendo así, énfasis en aumentar la comodidad de los usuarios.

2.1.3.1. TV BOX

Adicional, en el mercado se ofertan dispositivos denominados TV Box que al conectarlos a un televisor normal, lo convierten en un Smart TV, permitiendo que las tecnologías antiguas se adapten y conviertan en una

actual, un ejemplo de estos dispositivos es el MXQ PRO 4K, sus características principales se detallan en la Tabla 2.

Tabla 2.

Características MXQ Pro 4K

Nombre	Característica
Sistema operativo	Android 5.1
CPU	Amlogic S905 2.0ghz procesador A53 corteza
GPU	Brazo penta-core mali-450
RAM	DDR3 1GB
ROM	8GB Flash NAND
Conectividad WI-Fi	IEEE 802.11b / g / n
Ethernet	10/100m
Vídeo	hd MPEG1 / 2/4. h.265.4k-hd
Puerto HDMI	HDMI 2.0 estándar
Puertos	4 * USB, 1 * HDMI, 1 * RJ45. 1 ranura para tarjetas SD *, 1 *
Interfaces	av, 1 * pb en, 1 * SPDIF
3D	Hardware aceleración de gráficos 3D
Voltaje de alimentación	CC 5V / 2A

2.1.4. Desarrollo de aplicaciones para Smart TV

El desarrollo de aplicaciones para Smart TV, tomado desde un novedoso punto de vista comercial y experimental, se torna un tanto complejo al no existir una estandarización, puesto que cada televisor tiene su SDK. Estos SDK's funcionan como una extensión del IDE de Eclipse, actualmente las herramientas de desarrollo ofrecen generar dos tipos de aplicaciones para Smart TV (Saraguro, 2014):

1. Aplicaciones web que faciliten la portabilidad a diferentes plataformas.
2. Aplicaciones nativas, tal como, Android, Native Client, C, entre otros.

Se presenta una breve explicación de las tecnologías de Smart TV, con sus herramientas, lenguajes de desarrollo, y los tipos de proyectos que se puede crear. (Ver Tabla 3). (Saraguro, 2014).

Tabla 3.

Herramientas de desarrollo para Smart TV's

Smart TV	Herramienta de desarrollo	Tipo de proyecto	Lenguaje de programación
Google TV, Android TV	SDK Android	Android	Java
Samsung Smart TV	SDK Samsung Smart TV	Nativo, Web	HTML, CSS, Js, Flash, C
LG	SDK Smart TV Alliance, WebOs SDK.	Web	HTML, Js
Apple TV	Xcode	iOS	AppleScript, Objective C, Perl, Python, and Ruby
Toshiba	SDK Smart TV Alliance	Web	HTML
Sony	SDK Android Addon Sony	Android	Java
Panasonic	SDK Smart TV Alliance	Web	HTML

Fuente. (Saraguro, 2014)

Si se toma como referencia a la plataforma Android para el desarrollo de aplicaciones para Smart TV, es visible las ventajas que ofrece, al ser multiplataforma permite su reproducción en todos aquellos dispositivos que implementen como sistema operativo a Android. La arquitectura de Android está compuesta por capas que son: Kernel de Linux, entorno de ejecución Android, bibliotecas, marco de aplicación y capa de aplicación. Lo más importante de esta arquitectura es el Kernel de Linux, este es el que permite la gestión de la memoria y la programación de los procesos y servicios.

Bonifaz Kaufman interesado en la generación de aplicaciones que incluyan Android y Arduino creo Amarino, que es un conjunto de herramientas que permite la transmisión y recepción de datos a través de Bluetooth, gracias a su interfaz gráfica se puede conectar fácilmente la placa y el dispositivo celular.

El proceso de desarrollo de una aplicación para Smart TV debe seguir el siguiente lineamiento (Saraguro, 2014):

1. Planificar y diseñar la aplicación y la interfaz de usuario de la aplicación.
2. Implementar código en la herramienta IDE o SDK.
3. Depurar y probar la aplicación.
4. Empaquetar su aplicación para cargarlo al Smart TV.

Pero, esta tarea compleja por la variada existencia de sistemas operativos, puede ser solucionada si se utiliza herramientas como MDA (Model Driven Architecture) para el proceso de modelado, al aplicar este enfoque, la generación de la interfaz de usuario podría ser automatizada.

En el trabajo realizado por Sabraoui A., y Koutbi E. (2012), quienes se propone un enfoque para el diseño de la interfaz gráfica de aplicaciones móviles mediante la estructuración de un modelo independiente de la plataforma móvil, adoptando como mecanismo a MDA y utilizando plataformas Android OS, se llega a la conclusión de que, el modelado bajo conceptos y notación UML, además de generar ventajas, facilita la comunicación entre desarrolladores.

2.1.5. Android

Android es un sistema operativo diseñado por Google para dispositivos Smart basado en el S.O. Linux, por lo tanto, está basado en “Código Abierto”, es decir, el código del sistema y el del software en general están disponibles para realizar consultas y modificaciones.

En la actualidad existen varias plataformas de software para la programación de aplicaciones en Android, entre ellas tenemos: Android Studio, Eclipse, entre otros. En dichas aplicaciones se puede hacer uso total de los elementos del dispositivo, para que el programador desarrolle aplicaciones de acuerdo a sus necesidades.

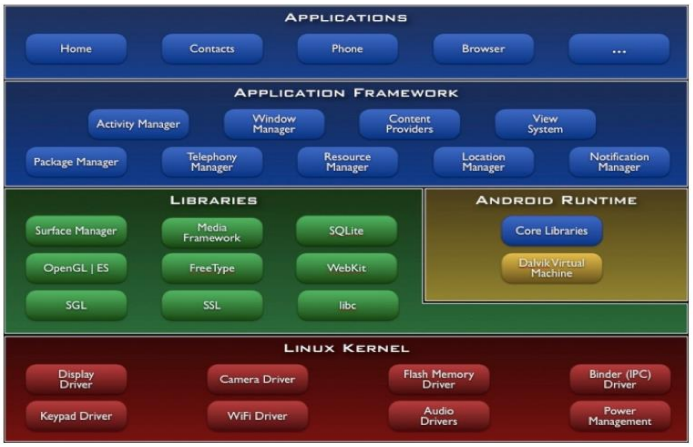


Figura 4. Estructura de Android

Fuente. (La Biblia del Programador, 2012)

En la Figura 4., se muestra la estructura manejada por Android, la que consta de cuatro niveles de jerarquización (Universidad Carlos III de Madrid, s/f):

1. Núcleo Linux: Usa el núcleo de Linux 2.6, tiene la capacidad de manejar servicios de : controlador de pantalla, controlador de cámara, controlador de teclado, controlador de red inalámbrica, controlador de audio, controlador de memoria, control de manejo de energía.
2. Tiempo de ejecución y librerías:

   Tiempo de ejecución: Se encuentra al mismo nivel que las librerías de Android y está conformado por librerías del núcleo y la máquina virtual “Dalvik”.

   Librerías: conformado por las librerías utilizadas por Android, las mismas que han sido escritas utilizando código fuente de C/C++ y proporcionan a Android todas sus características. A continuación se detallan las librerías utilizadas, por este nivel (Universidad Carlos III de Madrid, s/f):

   - Librería libc: Conformada por cabeceras y funciones según el estándar del lenguaje C.

- Librería surface Manager: Encargada de componer los elementos de navegación de pantalla y maneja las ventanas pertenecientes a las distintas aplicaciones activas.
- OpenGL/SL y SGL: Representan las librerías gráficas y por tanto sustentan la capacidad gráfica de Android. OpenGL/SL maneja gráficos en 3D y permite utilizar, en caso de que esté disponible en el propio dispositivo móvil. SGL proporciona gráficos en 2D utilizado por la mayor parte de aplicaciones.
- Librería media libraries: Proporciona los códecs suficientes para el contenido multimedia (vídeo, audio, imágenes estáticas y animadas, entre otros).
- FreeType: Permite realizar trabajos de forma sencilla y rápida con distintos tipos de fuentes.
- Librería SSL: Protocolo para establecer comunicaciones seguras.
- Librería SQLite: Crea y gestiona bases de datos relacionales.
- Librería WebKit: Ofrece un motor para las aplicaciones de tipo navegador y conforma el núcleo del actual navegador insertado por defecto en la plataforma Android.

3. Entorno de aplicaciones: Corresponde al conjunto de herramientas para el desarrollo de una aplicación. Toda aplicación desarrollada utiliza el mismo API y el mismo "framework" representado por este nivel sin importar el creador de la aplicación.

   Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes (Universidad Carlos III de Madrid, s/f):

   - Activity manager: API que dirige el ciclo de vida de las aplicaciones en Android.
   - Window manager: Realiza la gestión de las ventanas de las aplicaciones y usa la librería surface manager.

- Telephone manager: Incluye todas las API vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- Content provider: Permite a cualquier aplicación compartir sus datos con las demás aplicaciones de Android.
- View system: Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI). También incluye vistas estándar para las funcionalidades más frecuentes.
- Location manager: Permite a las aplicaciones el uso de la información de localización y posicionamiento.
- Notification manager: Permite a las aplicaciones comunicar al usuario eventos que ocurran durante su ejecución ejemplo una llamada entrante, un mensaje recibido. Si llevan asociada alguna acción se denominada intent.
- XMPP service: Protocolo API utilizada para el intercambio de mensajes basado en XML.

4. Aplicaciones: En esta se encuentran todas las aplicaciones incluidas por defecto de Android, como aquellas desarrolladas por terceros. Se caracterizan por que todas las aplicaciones utilizan los servicios de los niveles anteriores.

2.1.6. Arduino

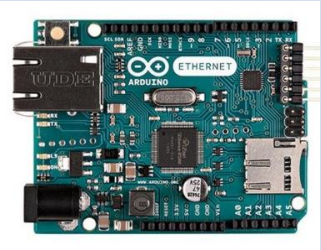
Se define como Arduino a una plataforma de prototipos electrónicos de “código abierto” constituida por hardware y software adaptativos, de uso fácil.

Tabla 4.

Características de placas Arduino, módulo Ethernet y Relays

Placa	Modelo	Características
	Arduino UNO	Microcontrolador: ATmega328P Pines digitales I/O: 14 PWM digitales I/O :6 Entradas analógicas: 6 Memoria Flash 32 KB (ATmega328P) SRAM: 2 KB (ATmega328P) EEPROM: 1 KB (ATmega328P) Clock speed: 16 MHz
	Arduino LEONARDO	Microcontrolador: ATmega32u4 Pines digitales I/O: 20 PWM digitales I/O :7 Entradas analógicas: 12 Memoria Flash 32 KB (ATmega32u4) SRAM: 2.5 KB (ATmega32u4) EEPROM: 1 KB (ATmega32u4) Clock speed: 16 MHz
	Arduino 101	Pines digitales I/O: 14 PWM digitales I/O :4 Entradas analógicas: 6 Memoria Flash 196 KB SRAM: 24 KB Clock speed: 32 MHz
	Arduino Mega 2560	Microcontrolador: ATmega2560 Pines digitales I/O: 54 PWM digitales I/O :15 Analógicas I/O: 16 Memoria Flash: 256 KB SRAM: 8 KB EEPROM: 4 KB Clock speed: 16 MHz
	Módulo de Relays de 4 canales	4 canales independientes protegidos con optoacopladores. 4 Relays de 1 polo 2 tiros. El voltaje de la bobina del relé es de 5 VDC. Voltaje entrada: 5 V. Voltaje de control: 3.3~9 V. Voltaje de salida: 250 VCA o 30 VDC.

Continua

		Corriente a la salida: 10 A. Activado por corriente: Corriente del circuito de control de 15 a 20 mA.
	Módulo de Relays de 2 canales	2 canales independientes protegidos con optoacopladores. 2 Relays de 1 polo 2 tiros. El voltaje de la bobina del relé es de 5 VDC. Voltaje entrada: 5 V. Voltaje de control: 3.3~9 V. Voltaje de salida: 250 VCA o 30 VDC. Corriente a la salida: 10 A. Activado por corriente: Corriente del circuito de control de 15 a 20 mA.
	Módulo Ethernet Arduino	Microcontrolador: ATmega328 Pines digitales I/O: 14 PWM digitales I/O :4 Analógicas I/O: 6 Memoria Flash: 32 KB SRAM: 2 KB EEPROM: 1 KB Clock speed: 16 MHz Controlador Ethernet embebido W5100 TCP/IP.

Arduino es capaz de captar el entorno a través de la recepción de entradas de variados sensores y puede monitorear a su alrededor actuadores de distintos tipos. Generar la programación para el microcontrolador de la placa es posible mediante el uso del “Lenguaje de Arduino” y del “Entorno de desarrollo”. Arduino en sus proyectos proporciona la capacidad de que sean autónomos o que posibiliten la comunicación con software en ejecución desde un ordenador.

En la Tabla 4., se presenta un detalle de las placas más utilizadas con sus respectivas características. (Arduino, s/f):

- Bajo costo: Las placas de Arduino son parcialmente económicas si se las compara con otras tarjetas microcontroladoras.

- Multiplataforma: Los sistemas operativos en los que se puede ejecutar el entorno de desarrollo de Arduino son Macintosh OSX, Windows y GNU/Linux.
- Entorno de programación simple y claro: El entorno de programación de Arduino es sencillo en su manejo para todo tipo de programadores.
- Código abierto y software extensible: Arduino se encuentra publicado como una herramienta de código abierto. El lenguaje permite ser complementado usando librerías C++ y se puede dar paso desde Arduino a la programación en lenguaje AVR C y viceversa.
- Código abierto y hardware extensible: Los microcontroladores ATMEGA8 y ATMEGA168 de Atmel fueron los que se utilizaron como base para Arduino. Y sus módulos están patentados bajo la licencia Creative Commons.

2.1.7. Base de datos

Se puede definir a una base de datos como una agrupación de información vinculada entre sí, que está estructurada o asociada.

En informática un sistema que se conforma por un conjunto de datos almacenados en discos que permitan el acceder directamente a ellos y un conjunto de programas que manipulan los datos, es denominado como base de datos.

La composición de cada base de datos es de una o más tablas que almacena un grupo de datos. Cada una de las tablas tiene una o más columnas y filas. En el que las columnas almacenan una porción de la información acerca de cada elemento que se desea guardar en la tabla, y cada una de las filas de la tabla constituye un registro. Se puede mencionar como características principales de las bases de datos a las siguientes (Pérez, 2007):

- Autonomía de los datos tanto lógica como físicamente.
- El porcentaje de redundancia que presentan es mínimo.
- Se permite acceder concurrentemente a varios usuarios.

- Los datos se integran.
- Las consultas complejas se optimizan.
- El acceso y la auditoría son seguros.
- Respaldo y recuperación.
- Se permite acceder a raíz de lenguajes estándar de programación.

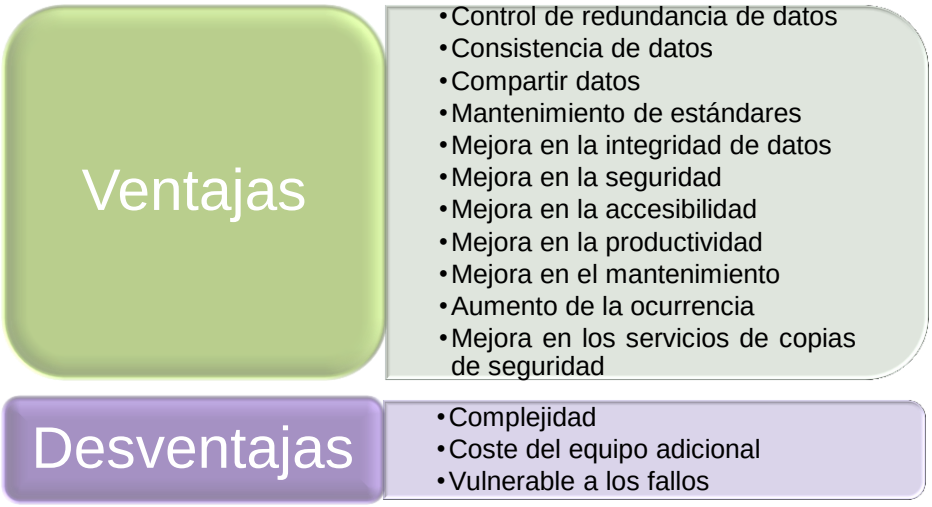


Figura 5. Ventajas y desventajas de las bases de datos

Fuente. (Pérez, 2007)

2.1.7.1. Tipos de bases de datos

Existen dos tipos de bases de datos, las bases de datos relacionales y las de datos no relacionales.

Las bases de datos relacionales son las que todos conocen, aquellas que su estructura se asemeja a la de un árbol; todo está interconectado entre sí y mantiene una relación. Las bases de datos SQL generalmente son utilizadas cuando no se necesita trabajar con volúmenes grandes de información.

Las ventajas de utilizar las bases de datos SQL son las siguientes (Pandorafm, 2015):

- Su uso es más adaptable, son más conocidas y económicas.

- El soporte que mantienen para gestionar bases de datos con mejores suites de productos y add-ons es mucho mayor, puesto que llevan mucho tiempo en el mercado.
- Se hace las operaciones en su totalidad o mejor no se las hace, utilizan la técnica del rollback.
- Se debe cumplir con la integridad en los datos y en la compatibilidad.

2.1.7.2. Tipos de motores de bases de datos

Los motores de bases de datos, son programas dedicados, cuyo principal objetivos es hacer de intermediarios entre las bases de datos y las aplicaciones que hacen uso de ellas. Las tareas que tienen son específicas, crear bases de datos, administrar la información de las bases de datos y funciones de acceso.

Tabla 5.

Motores de bases de datos Sql

Motores de bases de datos SQL
Oracle
MySQL
Microsoft SQL Server
PostgreSQL
Microsoft Access
SQLite

En la Tabla 5., se puede observar algunos de los principales motores de bases de datos.

El uso de cada uno de estos motores de bases de datos depende del tipo y volumen de información que se vaya a administrar.

2.1.7.3. Microsoft SQL Server

Es un lenguaje de programación utilizado para trabajar con conjuntos de hechos y las relaciones entre ellos dentro de una base de datos. Dentro de este lenguaje se realiza consultas que se transforman en lenguaje de comandos y que facilitan los métodos de selección, inserción, actualización, averiguación de la ubicación de los datos, y más. (Rouse, 2015)

SQL es considerado como un lenguaje declarativo de alto nivel, porque utiliza conjuntos de registros y no registros individuales, ofreciendo una productividad superior en la codificación y orientación a objetos.

Dentro de una base de datos podemos encontrar el DDL (Lenguaje Definición de Datos) que permite modificar objetos contemplados en la base de datos. SQL es un lenguaje DML que permite manipular datos por medio de consultas.

2.2. Arquitectura Orientada a Servicios (SOA)

SOA surge por los problemas comunes a la hora de desarrollar software, pues en mucho de los casos no se tiene compatibilidad entre las aplicaciones, modelos de datos, lenguajes de programación o sistemas de comunicación obligando así a volver a diseñar y reescribir las aplicaciones para que puedan operar entre sí. Por lo tanto, una aplicación en la actualidad debe procurar en lo posible tener características de extensibilidad y comunicarse con un mayor nivel de abstracción, pero sobre todo no debe ser restringida. SOA está conformado por elementos básicos (García, Gonzales, Castillo, Arenas, s/f):

- Proveedores de servicios: La aplicación presenta operaciones que pueden ser usadas por cualquier otra aplicación.
- Consumidores de servicios: Hacen uso de las operaciones del proveedor para adquirir información.
- Bus de servicios empresariales: Integran a los servicios de manera lógica y extensible.

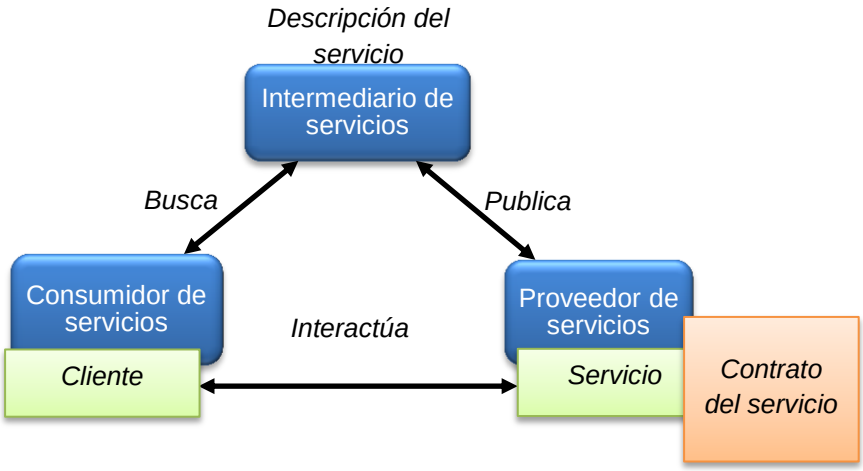


Figura 6. Esquema de interacción de servicios

Se hace referencia a un servicio como un componente de software relacionado a una interfaz claramente definida, y que su ejecución no depende del lenguaje en el que ha sido creado, ni de la plataforma que lo provee o consume.

Tabla 6.

Beneficios de SOA

Beneficio	Descripción
Aplicaciones más productivas y flexibles	Hace posible la obtención de mejor productividad de los recursos existentes
Desarrollo de aplicaciones más rápidas y económicas	Aceleración en el proceso de desarrollo de proyectos y rebaja de los costos de producción de soluciones al eliminar redundancias.
Aplicaciones más seguras y manejables	Facilita la agregación de nuevas funciones y servicios que permitan administrar los procesos de negocio con riesgo, al ingresar a los servicios y no directamente a las aplicaciones.
Minimación de riesgo de tiempo de inactividad o pérdida de datos	Gracias a sus capacidades de rendimiento, escabilidad, seguridad y disponibilidad alta y sin precedentes.
Mejora de la capacidad para innovar y diferenciarse	Desarrollar las funcionalidades nuevas de forma rápida con los datos integrales, precisos y oportunos.

Fuente. (Vera, 2014)

Un servicio puede estar compuesto por servicios de mayor granularidad, de un nivel de abstracción superior, permitiendo aplicar seguridades de acceso y de confiabilidad de los datos, y puede ser ejecutado local o remotamente. A este tipo de servicios se los denomina, servicios web, y son básicamente software que ha sido desarrollado para la interacción máquina a máquina en una red. La Tabla 6, presenta los beneficios de utilizar SOA.

2.2.1. Servicios Web

La necesidad de estandarizar la comunicación entre varias plataformas, y lenguajes de programación, hacen que surjan los servicios web.

Entre los intentos fallidos por crear estándares están DCOM y CORBA – ORB, estos dependían de la implementación del propietario. Además de que se requería del uso de RCP (Remote Procedure Call) para establecer la comunicación entre los distintos nodos, y su seguridad no era tan confiable, por lo que implementarlos en internet, era casi imposible.

Para el control de RCP, se creó RMI, mecanismo que ofrecía invocar métodos de forma remota y comunicar fácilmente los servidores de aplicaciones distribuidas que se basaban únicamente en java.

La solución para estos problemas de confiabilidad, escabilidad, conectividad y comunicación fue crear REST y SOAP.

2.2.1.1. REST (Representational State Transfer)

Hace referencia a una forma de arquitectura de software orientada a sistemas hipermedia distribuida, como la Web, es básicamente un conjunto de principios para diseñar arquitecturas en la red.

REST permite el uso de links entre diferentes recursos, se basa en operaciones, no permite representaciones XML y no cuenta con una gama amplia de estándares.

Concretamente sirven para aquellos sistemas en los que prima la manera en la se realizan las peticiones al servidor desde el cliente basados

únicamente en el recurso de interés. Las ventajas que ofrece REST son (Mayta, 2012):

- Realizar pocas operaciones con muchos recursos.
- Centrarse en el rendimiento y escabilidad a nivel superior para sistemas distribuidos de hipermedia.
- Permiten HTTP GET, HTTP POST, HTTP PUT y HTTP DEL.
- Autodescritivo XML
- Síncrono
- Comunicación segura de punto a punto.

2.2.1.2. SOAP (Simple Object Access Protocol)

SOAP es un protocolo utilizado para intercambiar mensajes en las redes de computadoras, utilizando generalmente HTTP, se basa en XML, por lo que se facilita la lectura de los datos. Esto ayuda a que las empresas integren sus servicios web de manera sencilla y puedan construir procesos complejos de negocio. Las ventajas y desventajas de usar SOAP en los servicios web se muestran en la Tabla 7.

Es posible crear un servicio web que ofrezca métodos que se puedan llamar con RCP desde cualquier lugar de Internet usando el protocolo SOAP, es decir, el servicio debe ser capaz de interpretar los mensajes SOAP de petición para invocar luego el método solicitado y con el resultado obtenido se debe construir el mensaje de respuesta SOAP y devolvérsela al cliente.

La generación de los servicios web, se facilita con la utilización de herramientas como visual Studio, que implementa plantillas con SOAP en lenguajes como C, Java, C++ o C# para su desarrollo de forma efectiva, sencilla y en menos tiempo, solamente se debe colocar el código correspondiente a los WebMethod que para obtener la información que el usuario necesite.

Tabla 7.

Ventajas y desventajas de SOAP.

Ventajas	Desventajas
Contribuyen con la interoperabilidad entre las aplicaciones de software, sin depender de las propiedades o plataformas en las que se las instalen.	Se apoyan en HTTP, por lo que se pueden obviar medidas de seguridad basadas en firewall, cuyas normas tratan de bloquear.
Es más fácil acceder a su contenido y entender su funcionamiento, puesto que fomentan el uso de protocolos basados en texto.	La comunicación entre los programas se representa como una dificultad.
Permite la provisión de servicios integrados, al hacer posible la combinación de servicios y software de diferentes compañías ubicadas en diferentes posiciones geográficas.	No existe información para la creación de servicios web en algunos lenguajes de programación, y no tiene definidos algunos estándares
Se basa en estándares.	No se representa como una solución a todos los problemas.
	Depende de la disponibilidad de servidores y comunicaciones.

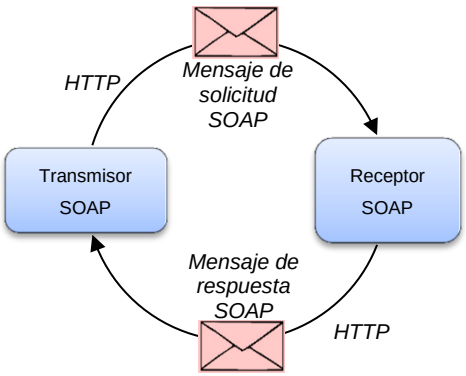


Figura 7. Transmisión y recepción de mensajes SOAP

Fuente. (Propia)

2.2.1.3. Internet Information Service (IIS)

Es un conjunto de prestaciones utilizado para servidores de Microsoft Windows usualmente empleados en servidores web.

El IIS ofrece una plataforma segura, de administración sencilla, modular y extensible en la que se puede alojar aplicaciones, servicios y sitios web confiablemente. Adicional la compartición de información con los usuarios en Internet es aceptada, en una intranet o en una extranet IIS, además, la plataforma web que utiliza, contiene IIS, ASP.NET, servicios de FTP, PHP y WCF “Windows Communication Foundation”.

Entre las distintas ventajas que presenta IIS tenemos (Microsoft Developer Network, s/f):

- Seguridad web que es reforzada con el apoyo de una superficie reducida de servidor y la separación automática de aplicaciones.
- Se permite implementar y ejecutar aplicaciones web de ASP.NET, ASP clásico y PHP dentro del mismo servidor sin mayor dificultad.
- Se le asigna una identificación única a los procedimientos tanto de trabajo como de configuración en un ambiente aislado reduciendo así los riesgos de seguridad.
- Los componentes del IIS se pueden agregar o eliminar sin presentar riesgos o dificultades.
- Alta velocidad en el sitio web, gracias al alojamiento en el caché.

Aplicaciones: Dentro de las distintas aplicaciones que se pueden realizar se tiene (Microsoft Developer Network, s/f):

- Se puede configurar características de IIS y administrar sitios web.
- Se puede cargar y descargar archivos por medio del uso del protocolo FTP.
- La configuración de las aplicaciones web con distintas tecnologías como ASP clásico, ASP.NET y PHP es soportada.
- Permite automatizar la administración de tareas del servidor web.
- Configuración de varios servidores web y administración con IIS.

2.2.1.4. ASP.NET

Es un sitio web que permite establecer las páginas predeterminadas para la aplicación, facilitando la exploración del sitio a los usuarios, al poder escribir en el código la dirección de la página a la que se desea referir como página principal. Además proporciona los mecanismos para hacer referencia a la lectura o escritura de archivos ubicados en el servidor, y establecer las rutas de accesos a estos archivos.

Para realizar la administración del sitio web, se debe configurar previamente el archivo Web.config.

Tabla 8.

Ciclo de Vida de ASP.NET

Fase	Descripción
Solicitud de página	Se produce antes de que inicie el ciclo de vida de la página. Al ser solicitada la página por el usuario, ASP.NET determina si debe ser analizada y compilada o si es factible enviar una versión en caché de la página como respuesta sin ejecutar la página.
Inicio	En el paso de inicio, se establecen las propiedades de la página, como Request y Response. En esta fase, la página también determina si la solicitud es una devolución de datos o una nueva solicitud, y establece la propiedad IsPostBack. Además, durante esta fase se establece la propiedad UICulture de la página.
Inicialización de página	Durante el inicio de la página, están disponibles los controles insertados en ella y adicional se determina la propiedad UniqueID de cada control y se aplican los temas respectivos de la página.
Carga	Si el usuario solicita una devolución de datos, el control carga sus propiedades con la información retomada del estado de vista y control.
Validación	Se llama al método Validate disponible en todos los controles de validación, que determina la característica IsValid de cada uno de los controles de validación y de la página.
Control de devolución de datos	Si la solicitud es una devolución de datos, se llama a los controladores de eventos.
Representación	Se guarda en la página el estado de vista y, posteriormente, se llama a cada uno de los controles para que aporte su salida representada al valor OutputStream de la propiedad Response de la página.

Continua

Descarga	Si la página se ha representado completamente, si se ha enviado al cliente y si está lista para ser descartada se llama a la descarga. E inicia, la descarga de las propiedades de la página, como Response y Request, y las actividades de limpieza se ejecutan respectivamente.

Fuente: (Microsoft Developer Network, 2007)

Es necesario conocer el ciclo de vida de una aplicación de ASP.NET para que inicie y pueda procesar oportunamente las solicitudes, estas fases se detallan brevemente en la Tabla 8. (Microsoft Developer Network, 2007).

Cuando se crea un sitio web de ASP.NET su puede incluir diferentes tipos de archivos soportados por ASP.NET, y, se puede generar carpetas cuyo propósito específico sea guardar código fuente. Entre los tipos de archivo que administra ASP.NET, se tiene, .asmx, su ubicación es en la raíz de la aplicación o un subdirectorio, y es un archivo de un controlador genérico que consta de código que genera la interfaz. A su vez, los archivos de ASP.NET (.asp) pueden ser gobernados desde el Administrador IIS.

2.3. Ingeniería de modelos

La problemática de desarrollar software, debe ser enfrentada de la misma manera en la que los ingenieros generan otros tipos de sistemas complejos, basándose en metodologías precisas apoyadas en la aplicación de herramientas.

En los años 70 Tom Demaco, utilizó el terminó MBD (Model Based Development), en su libro, “Structured Analysis and System Specification”, en el que determinaba que la creación de un sistema de software debe tener como precedente la creación de un modelo. Este modelo representa el dominio del problema y de la solución de forma abstracta, tomando los elementos principales para estructurarlos explícitamente.

El modelado de un sistema debe brindar un mecanismo de comunicación y negociación entre los desarrolladores de software, minimizando las características vinculadas a la tecnología de desarrollo. En general, los

métodos actuales para desarrollar software, en sí, ya implementan modelos, con tendencia a los métodos matemáticos o métodos diagramáticos.

El ciclo de vida basado en modelo debe permitirnos generar eficiencia y corrección del producto final, hoy en día el proceso de desarrollo es conducido por el diseño de bajo nivel y por el código. (Ver Figura 8.)

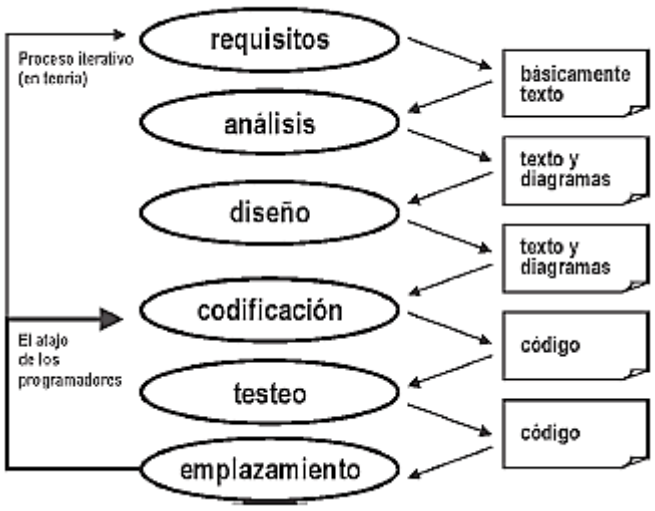


Figura 8. Ciclo de vida del desarrollo de software basado en modelos
Fuente. (Pons, Giandini y Pérez, 2010)

El problema de los programadores al momento de desarrollar software es que no realizan cambios en la etapa de diseño, sino, solamente lo hacen en el código, al no tener tiempo para actualizar los diagramas y otros documentos de alto nivel. Esto representa que la tarea de mantenimiento de los sistemas se haga compleja, luego de que ha transcurrido tiempo de la elaboración del software, o que ya no se cuente con la persona que realizó el código.

La industria del software permanece en constante evolución, lanzando al mercado nuevas tecnologías que de forma rápida llegan a ser populares, y que las compañías necesitan aplicar estas nuevas tecnologías en sus procesos de actualización para ofrecer mejores beneficios y no quedarse atrás, no se puede permanecer con sistemas de los que sólo se tenga la programación, al contrario, es indispensable tener el modelo, estos

inconvenientes dieron paso a la creación del concepto de Desarrollo de Software Dirigido por Modelos (MDD).

MDD promete mejorar el proceso de construcción de software basándose en un proceso guiado por modelos soportado por potentes herramientas. Los modelos que se genera van desde los más abstractos hasta los más concretos siguiendo pasos de transformación y/o refinamientos, hasta llegar al código aplicando una última transformación. La transformación entre modelos constituye el motor de MDD (Pons, Giandini y Perez, 2010).

La Figura 9, muestra el ciclo de vida del software dirigido por modelos, este no luce diferente al del desarrollo normal, sin embargo, en este no se salta ninguna etapa ni se toma atajos.

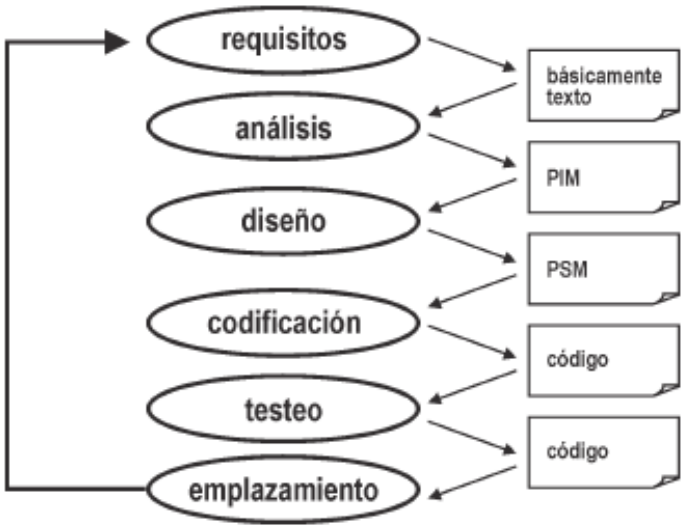


Figura 9. Ciclo de vida del desarrollo de software dirigido por modelos
Fuente. (Pons., Giandini y Pérez, 2010)

MDD identifica diferentes modelos (Pons, Giandini y Pérez, 2010):

- Modelos de alto nivel de abstracción independientes de las variadas metodologías computacionales. (CIMs, Computational Independent Model).
- Modelos independientes de la tecnología en la que se implementen. (PIMs, Platform Independent Model).

- Modelos que detallan el sistema en terminología de construcción de implementación disponible en alguna tecnología especifica. (PSMs, Platform Specific Model).
- Modelos que representan el código fuente en sí mismo. (IMs, Implementatio Model).

Las transformaciones MDD son ejecutadas comúnmente por herramientas como se indica en la Figura 10.

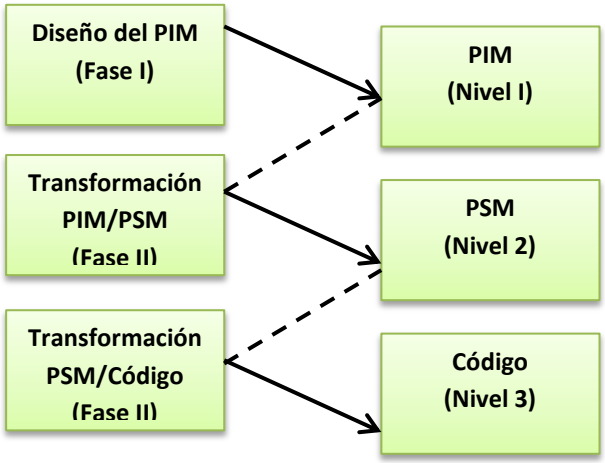


Figura 10. Pasos en el proceso de desarrollo MDD

Fuente. (Pons, Giandini y Pérez, 2010)

El desarrollo de software dirigido por modelos hace posible la mejora de las prácticas comunes, además presenta las siguientes ventajas (Pons, Giandini y Pérez, 2010).

- Incremento en la productividad.
- Adaptación a los cambios tecnológicos.
- Adaptación a los cambios en los requisitos.
- Consistencia:
- Mejoras en la comunicación con los usuarios
- Mejoras en la comunicación entre los desarrolladores:

- Captura de la experiencia:
- Los modelos son productos de larga duración:
- Posibilidad de demorar las decisiones tecnológicas:

De las propuestas más conocida en MDD, se tiene a MDA (Model Driven Arquitecture) y MDS (Domain Specific Modeling), estas dos técnicas mantienen una fuerte conexión con los conceptos básicos de MDD. Especiamente MDA tiene a enfocarse en el lenguaje de modelado que se base en los entándares de OMG.

2.3.1. MDA (Model Driven Arquitecture)

Es un concepto promovido por OMG (Object Managment Group), desde el año 2000 cuyo objetivo es afrontar los retos de integración de las aplicaciones y la continua actualización tecnológica. MDA no está limitado al desarrollo de sistemas de software, al contrario, también es posible adaptarlo para el desarrollo de otros tipos de sistemas.

MDA se relaciona con algunos estándares, como Unified Modeling Language (UML), Meta-Object Facility (MOF), XML Metadata Interchange (XMI), Enterprise Distributed Object Computing (EDOC), Software Process Engineering Metamodel (SPEM) y Common Warehouse Metamodel (CWM). (Pons, Giandini y Pérez, 2010).

Las dos motivaciones principales de los sistemas de software de MDA para cumplir con las directrices de OMG, son la interoperabilidad (independencia de los fabricantes a través de estandarizaciones) y la portabilidad (independencia de la plataforma). La postulación que hace OMG como objetivo fundamental de MDA es, separar el diseño del sistema de la arquitectura y de las tecnologías de construcción, así se logra facilitar la alteración independiente del diseño y de la arquitectura.

2.3.2. UML en MDD

UML es el lenguaje preferido para definir los modelos de MDD al ser un estándar abierto y de facto para el modelado de software. Al ser un lenguaje de propósito general, es posible aplicarlo de distintas maneras y en varios casos.

Los beneficios de usar UML en MDD son los siguientes (Pons, Giandini y Pérez, 2010):

- Es posible generar un entorno de desarrollo personalizado, evitando el complejo proceso de implementarlo desde cero.
- La industria en general utiliza este estándar abierto y de facto para el modelado de software.
- Desde 1995 se ha mantenido en el mercado.
- Existe numerosos cursos de excelente calidad y libros que proporcionan información y aprendizaje de UML.
- Las herramientas que soportan UML son numerosas.
- La ortogonalidad de los perfiles UML permite que se pueda aplicar varios perfiles simultáneamente.

2.4. Interfaces inteligentes

En el año de 1956, en la Conferencia de Dartmouth, se presentó como reto, el generar una máquina que fuera capaz de simular las facetas de la inteligencia humana, pensada específicamente en la faceta de la comunicación tanto con otras máquinas como con los humanos, esto dio paso a que en los últimos 60 años la comunidad investigadora ponga toda su atención en este tema, para mejorar las técnicas de comunicación, puesto que, la comunicación es la base fundamental para desarrollar escenarios que simulen las capacidades humanas.

Una interfaz de usuario (IU) es la interacción entre el usuario y la computadora, que presenta un alto grado de usabilidad, en sí, es la imagen que el usuario percibe de la aplicación, sin embargo, una interfaz inteligente

es una herramienta que colabora en el perfeccionamiento de comunicación entre el hombre y la máquina, generando mejores sensaciones en los usuarios, introduciendo naturalidad de la interacción, esto es posible al mejorar la calidad de comunicación entre el usuario y la máquina, aplicando técnicas de inteligencia artificial en el proceso de desarrollo de las interfaces.

Uno de los estilos de interacción más exitosos, y usados para el desarrollo de IUs es la manipulación directa, esta técnica permite que el usuario maneje directamente la representación gráfica de la aplicación en la pantalla, ejemplo de esto son los sistemas de administración de datos, los videojuegos y las hojas de cálculo electrónico, su popularidad se basa en la retroalimentación continua que proporciona a la tarea que el usuario realiza, es decir, permite realizar cambios o correcciones rápidas en las operaciones. Pero, este concepto se limita a que el usuario sea quien realice todas las actividades.

La interacción asistida por otro lado utiliza la metáfora del agente que contribuye con el usuario en el mismo entorno de trabajo, es decir, el usuario trabaja cooperativamente en comunicación con el agente, quien controla los eventos y las tareas realizadas (López, Jaquero, Montero, Molina y González, 2006).

Es posible que un agente de interfaz haga uso de las IU, sin necesariamente seguir las directrices del usuario, el agente de interfaz interpreta la entrada que presenta el usuario en la pantalla y puede realizar cambios a los objetos que el usuario visualiza en la pantalla, estos trabajan en segundo plano y su accionar es por iniciativa propia, esta acción da paso a las Interfaces Inteligentes de Usuario (IUI). (López, Jaquero, Montero, Molina y González, 2006).

Uno de los problemas que presenta el desarrollo de interfaces inteligentes es la relación y adaptación de las interfaces a las aplicaciones, pues su objetivo es que la eficiencia, efectividad y naturalidad de la interacción hombre – máquina sea eficiente y valorada, siguiendo el comportamiento de una serie de modelos. Las IUIs, deben ser capaces de

modelar al usuario, sus tareas, el entorno en el que labora, y analizando las entradas generar salidas adecuadas.

Figura 11. Objetivos habituales de una interfaz de usuario inteligente
Fuente. (López, Jaquero, Montero, Molina y González, 2006)

Como se visualiza en la Figura 11, los retos que enfrentan las IUIs son muchos, pues se incluye características humanas como, aprendizaje, razonamiento, autocorrección, entre otras, por lo tanto, los métodos y técnicas para el desarrollo de estas interfaces son numerosos, sin embargo, en la última década, la tendencia predominante es la de los entornos de desarrollo que se basan en modelos.

Lo que se pretende especificar con esta tendencia, debe ser claro, y poseer una serie de modelos que describan todas las características del usuario y se almacenen en formatos que permitan el desarrollo posterior de la aplicación, sin necesidad de especificar una plataforma específica.

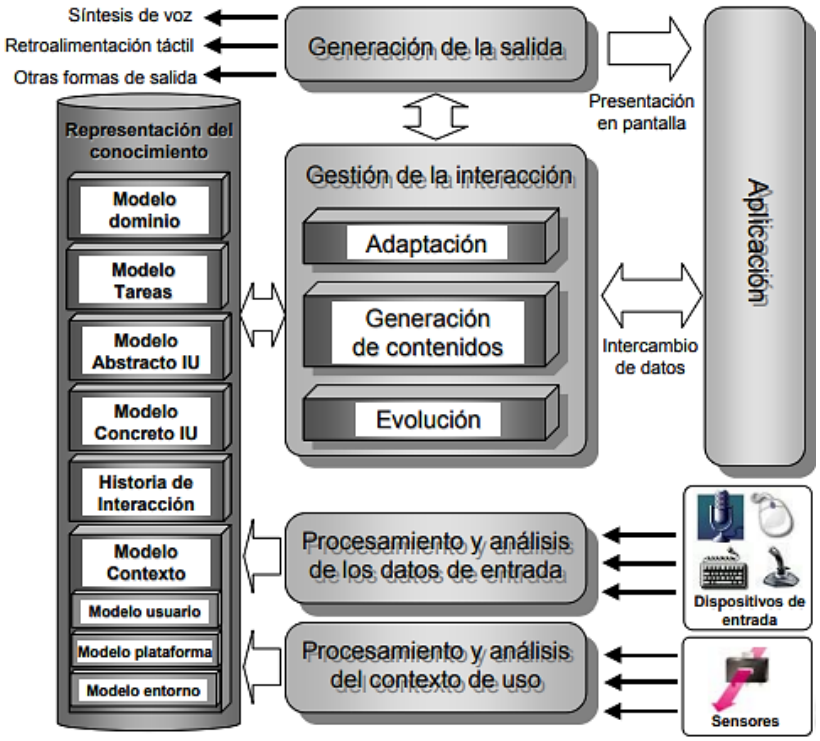


Figura 12. Arquitectura general de una interfaz inteligente de usuario
Fuente. (López, Jaquero, Montero, Molina y González, 2006)

En la actualidad se presentan desafíos en torno a la Inteligencia Artificial, que posibilita decir que la interfaz ideal, para la interacción entre humano – máquina, es aquella que no existe, en un futuro, el ordenador ideal, será un agente que conozca el entorno, los gustos, y la manera de ser de una persona, y que de manera discreta se adelante a las necesidades del usuario sin necesitar una orden, únicamente hablándole o realizándole gestos faciales, de igual forma como si se comunicara con otro ser humano. (López, Jaquero, Montero, Molina y González, 2006).

2.4.1. Interfaces inteligentes adaptativas

Una interfaz de usuario adaptativa consiste en cada uno de los elementos que la conforman, es decir, está compuesta por cada elemento de

los cuales se la construye, su objetivo es la comunicación entre el software y los usuarios.

Las interfaces de usuario inteligentes deben ser elementos capaces de adaptarse y modificarse a lo largo del ciclo de vida de una aplicación, por lo que para el desarrollo de las interfaces de usuario es importante la utilización de modelos, así se facilita su proceso de desarrollo, para este efecto, es importante conocer los niveles de representación del modelo que se va a utilizar. Los niveles de representación son (Criado J., 2015):

a) Tareas y conceptos: En el sistema se deben llevar a cabo diferentes tareas, las mismas que son descritas en este nivel. Además contiene las definiciones de aquellos conceptos que intervienen en el dominio, los conceptos fundamentales como resultado de su aplicación y dependiendo de sus capacidades, dan paso a la aparición de las diferentes tareas que se puede realizar. Este nivel hace referencia al nivel independiente de la computación (CIM, Computation Independent Model) de MDA, debido a que se describen los requisitos de la interfaz de usuario sin incluir detalles de la estructura ni del funcionamiento.

b) Interfaz de usuario abstracta (AUI, Abstract User Interface): representa el concepto de las interfaces de usuario a nivel estructural con un punto de vista abstracto, en otras palabras, los elementos que intervienen en las interfaces de usuario, no equivalen directamente a lo que se tiene en el mundo real, sin embargo, su representación es similar a un conjunto de elementos con características en común (tipos de elementos).

   Las relaciones que se establecen entre los elementos, son abstractas, y se utilizan como referencia para conocer qué elementos pueden ser enlazados entre sí. Este nivel hace equivalencia al nivel PIM (Platform Independent Model) de la aproximación MDA. Es decir, una AUI detalla la estructura y la funcionalidad de una interfaz de usuario de forma independiente a la plataforma en la cual va a ser desplegada.

c) Interfaz de usuario concreta (CUI, Concrete User Interface): concretamente representa interfaces de usuario a nivel estructural. Los elementos que representa tienen correspondencia en el mundo real, y las relaciones establecidas entre los elementos representados detallan enlaces específicos que pueden hacerse entre dichos elementos. Las interfaces de usuario concretas hacen referencia al nivel PSM (Platform Specific Model) de MDA. Las CUIs adicionan información sobre la implementación de sus elementos y su descripción está vinculada a una plataforma especifica.

d) Interfaz de usuario final (FUI, Final User Interface): este nivel de interfaz de usuario hace referencia a las interfaces de usuario reales que se ejecutan en una plataforma, incluye las interfaces de usuario que son compiladas y luego ejecutadas. En este caso, las interfaces de usuario finales hacen referencia al nivel de código de MDA.

La adaptación de las interfaces es posible a través de la manipulación de las arquitecturas que las detallan, tanto a nivel abstracto como a nivel concreto. Por lo tanto los trabajos de investigación no deben centrar su atención en la obtención de las arquitecturas abstractas a partir de la información del PIM, ni en el despliegue de las interfaces de usuario final a partir de las definiciones de arquitecturas abstractas. El punto de partida debe ser la existencia de arquitecturas abstractas y concretas, y que se puedan aplicar en diferentes operaciones permitiendo la adaptación de las interfaces de usuario en tiempo de ejecución.

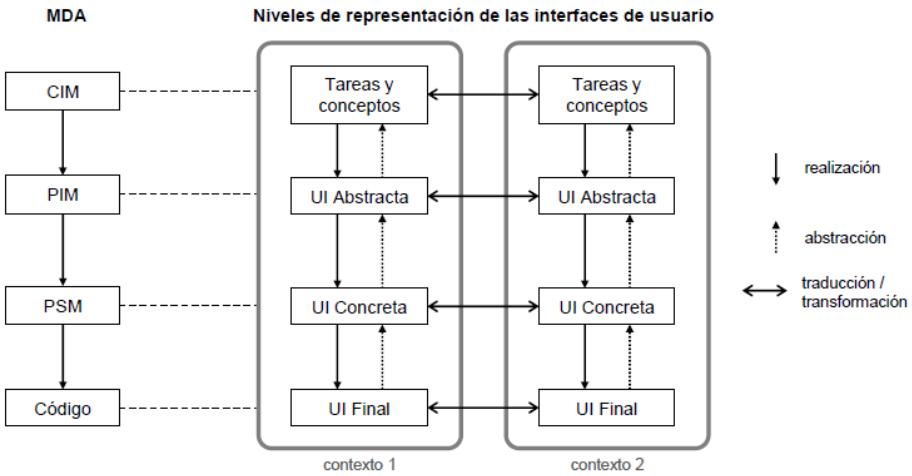


Figura 13. Niveles de Representación de usuario

Fuente. (Criado, 2015).

2.4.1.1. Operaciones en el nivel abstracto

Las arquitecturas abstractas están orientadas a la representación de los tipos de componentes y de las relaciones que existen en una arquitectura, previamente a la obtención de las arquitecturas concretas y de, posteriormente, la generación de la arquitectura software final. Sin embargo, existe la posibilidad de obtener una arquitectura abstracta a partir de otra, al aplicar operaciones de traducción o transformación. El objetivo en sí de las operaciones de transformación es obtener una arquitectura abstracta adaptada a partir de una arquitectura abstracta inicial. (Criado, 2015)

2.4.1.2. Operaciones en el nivel concreto

Las arquitecturas concretas se alcanzan luego de la operación de realizar las arquitecturas abstractas. Este es un proceso inverso a un mecanismo de abstracción, debido a que está orientado al aumento del nivel de detalle de la representación de la arquitectura.

Para el efecto, se define un proceso de regeneración que escoge los componentes concretos que cumplen eficientemente con la definición abstracta y que pueden construir la arquitectura concreta correspondiente.

El objetivo de las operaciones de regeneración es obtener una arquitectura concreta a partir de una arquitectura abstracta que ha sido previamente adaptada por una operación de transformación, también se da lugar a las operaciones de transformación encargadas de la adaptación de los modelos de arquitecturas concretas. Las operaciones de transformación a nivel concreto se dan en caso de que no sea necesario realizar una transformación a nivel abstracto para ejecutar el proceso de adaptación (Criado, 2015).

2.4.1.3. Esquema de la Metodología

Es importante tener clara la descripción de los niveles de representación que se utilizan para la descripción de las interfaces de usuario, así como las operaciones que se llevan a cabo en la metodología, además es necesario establecer el modo de funcionamiento que se debe seguir para la adaptación de las interfaces de usuario al momento que se está ejecutando (Criado, 2015).

CAPITULO III

DISEÑO E IMPLEMENTACIÓN

En este capítulo se presenta los procesos llevados a cabo para generar el diseño completo de SmartHomeTV, aplicando técnicas de ingeniería de desarrollo de software dirigido por modelos, además se detalla el diseño del Software con su respectivo código de programación y Hardware con su respectivo diseño.

3.1. Determinación del modelo del sistema

La ingeniería de modelos como se explicó en el capítulo anterior, es una técnica que permite que el software no tenga un ciclo de vida corto, por lo tanto, de la manera en que se presenta este modelo, se brinda la oportunidad de que cualquier desarrollador pueda utilizar este modelo para generarlo con las plataformas que más desee.

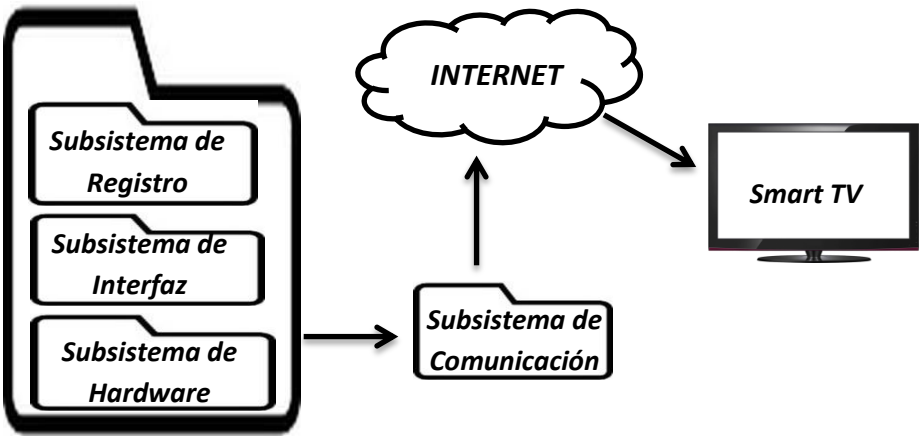


Figura 14. Modelo independiente de la computación (CIM) del sistema

El primer paso es analizar el propósito del desarrollo. La Figura 14, muestra un esquema grafico general de los subsistemas que intervienen en el Sistema, los mismos que se detallan a continuación:

- Subsistema de registro: Es donde se almacena la información de los datos personales del usuario y de los recursos de su hogar.
- Subsistema de interfaz: Se encarga de obtener la información del subsistema de registro, tanto de los datos de acceso del usuario como los de los recursos para generar automáticamente la pantalla de visualización.
- Subsistema hardware: Su función es conectar físicamente los recursos del hogar y controlarlos.
- Subsistema de comunicación: Mediante el subsistema de comunicación se estable la metodología lógica a través de la que se transmitirá y receptará la información entre los subsistemas de registro e interfaz.

El diseño del CIM no es complejo, pues al ser un modelo independiente de la computación sólo debe mostrar las acciones y roles de cada uno de los componentes que conforman el Sistema.

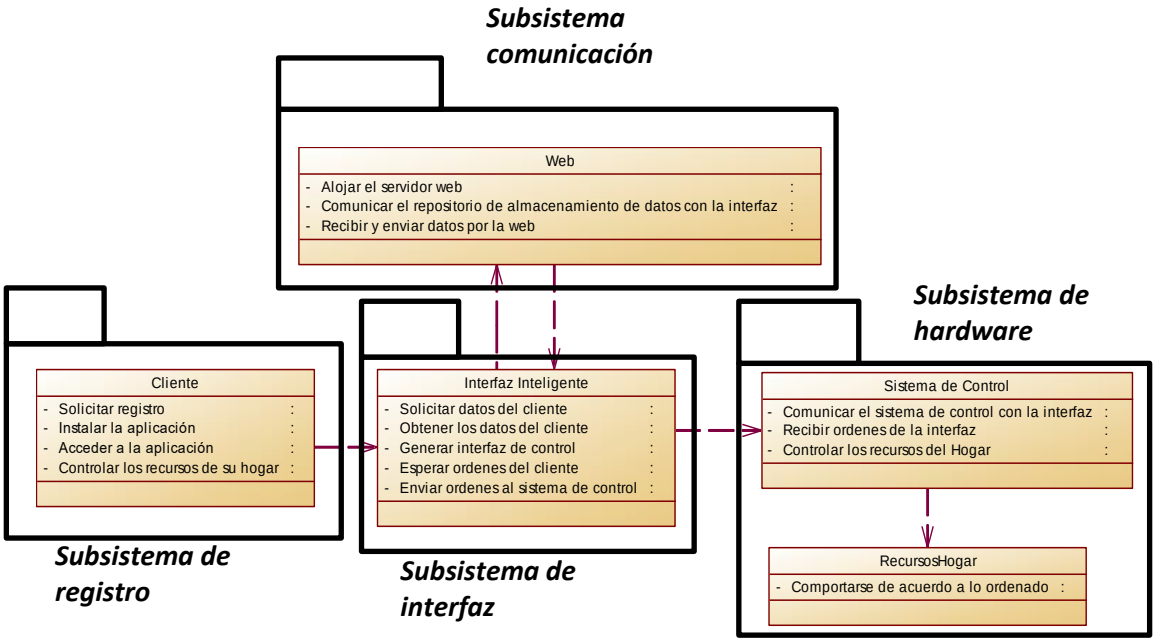


Figura 15. Modelo independiente de la plataforma (PIM)

Una vez que se ha determinado el CIM, el siguiente paso es transformarlo en un PIM (Platform Independent Model), tal como su nombre lo indica, no debe especificar la plataforma en el que se lo va a desarrollar, es decir, debe ser detallado, pero general para que pueda se implementar usando cualquier herramienta, plataforma o lenguaje de programación. Para realizar el diagrama del diseño se utiliza el lenguaje casi universal, UML, para mejorar el entendimiento del mismo. (Ver Figura 15.)

El PIM también está dividido en cuatro subsistemas, algunos de ellos se especifican en dos niveles. La Tabla 9 presenta un detalle del diagrama.

Tabla 9.

Detalle de modelo independiente de la plataforma (PIM)

Color	Nombre	Descripción
	Tablas	Etapa de registro y almacenamiento de datos. Subsistema de registro
	Clases	Corresponde a las clases para la programación del servicio web y de la interfaz. Subsistema de comunicación e interfaz
	Código	Define a la programación del software para el sistema de control. Subsistema de Hardware
	Variables	Detalla las variables de cada uno de los métodos de cada una de las clases. Nivel 2 de subsistemas de comunicación e interfaz.

En los apartados posteriores se explica detalladamente cada uno de los subsistemas que se definen en el PIM, con sus respectivos niveles y código.

SUBSISTEMA DE REGISTRO **SIBSISTEMA DE COMUNICACIÓN** **SUBSISTEMA DE INTERFAZ** **SUBSISTEMA DE HARDWARE**

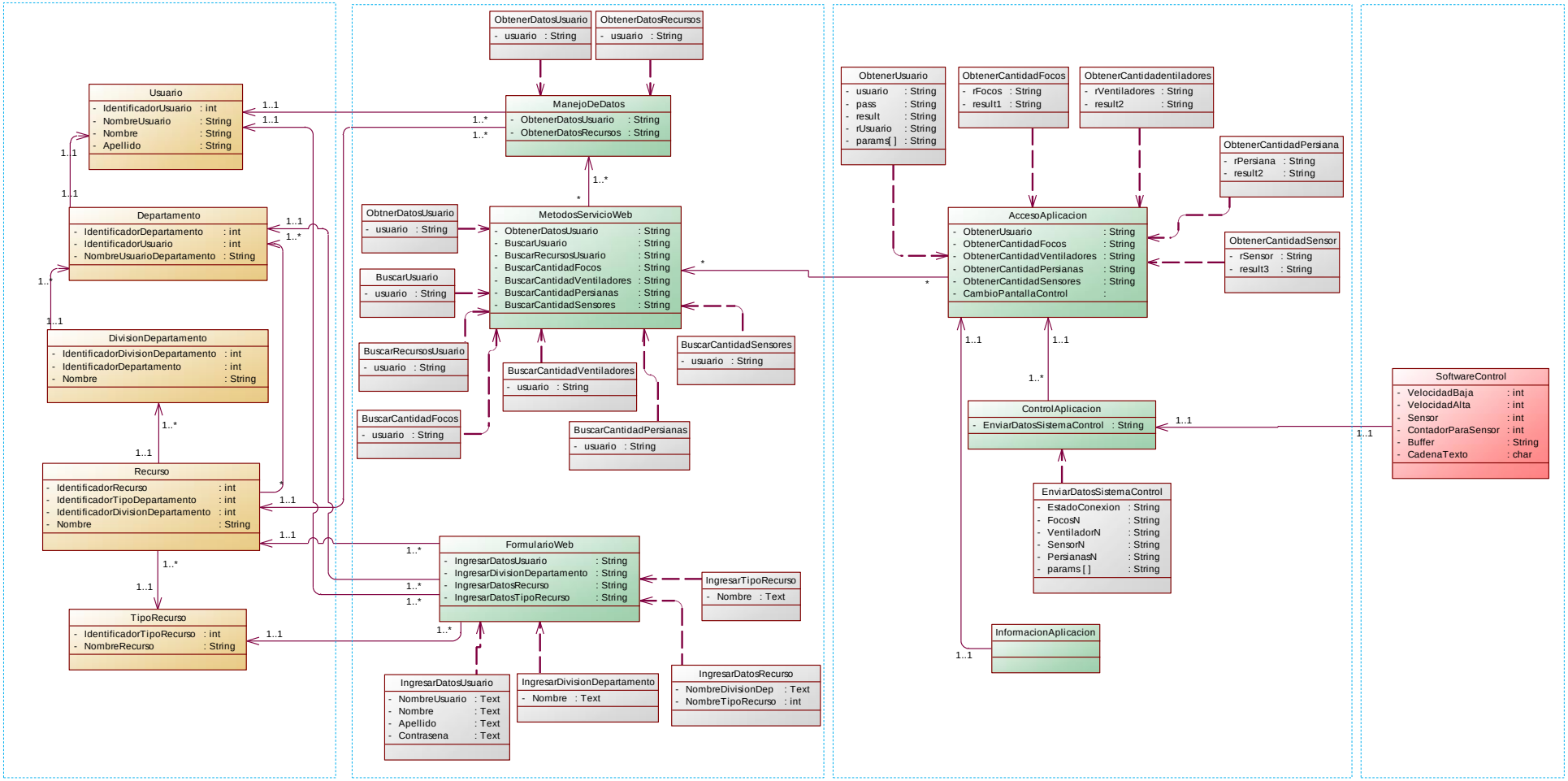


Figura 16. Modelo independiente de la plataforma de la interfaz

3.2. Arquitectura del sistema

Con el modelo del PIM de la Figura 16, es posible definir una arquitectura de red que permita la comunicación tanto física como lógica del sistema. (Ver Figura 17.)

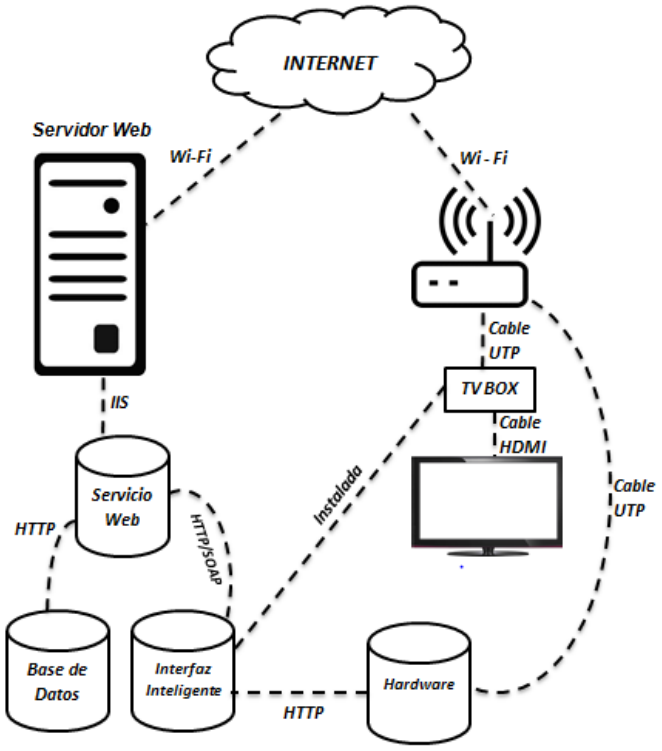


Figura 17. Arquitectura del sistema

3.3. Diseño del software

El desarrollo en sí del software depende del diseño de los subsistemas de registro y comunicación. El subsistema de registro corresponde a la generación de una base de datos y el subsistema de comunicación representa la creación de un servicio web. (Ver Figura 18.)

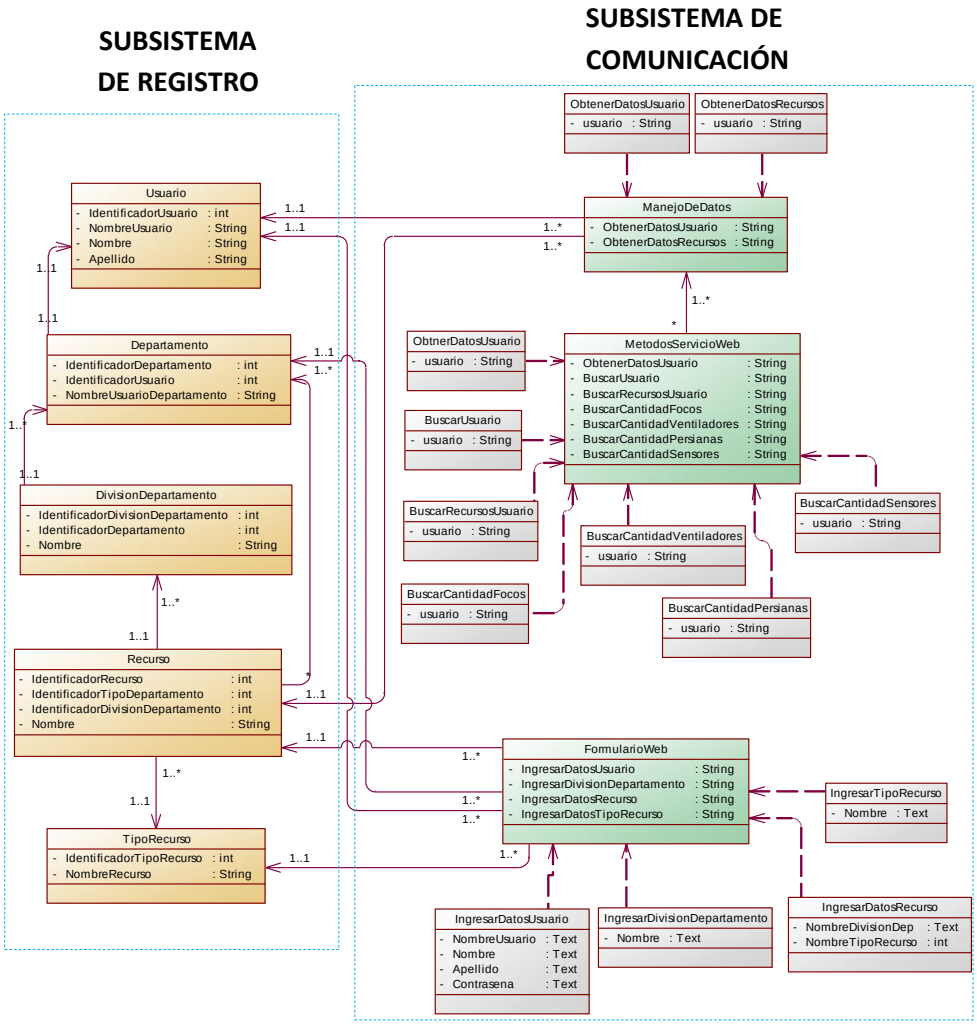


Figura 18. Subsistema de registro y comunicación del software

Las plataformas y lenguajes de programación elegidos en base a los requerimientos que se indican en el PIM son:

- Microsoft SQL Server para diseñar e implementar la base de datos.
- Visual Studio 2012 para implementar el servicio web

3.3.1. Diseño de la base de datos.

El diseño de la base de datos se realiza basado en el subsistema de registro de la Figura 18, en la que se especifica la información que contendrá cada tabla, por lo tanto, utiliza cinco tablas, algunas de ellas obtienen

atributos de otras tablas para que sea más sencillo realizar las consultas de los datos que se hace sobre la base. La Figura 19, muestra el diseño del diagrama de clases con las tablas, los atributos y especifica las claves primarias de cada una.

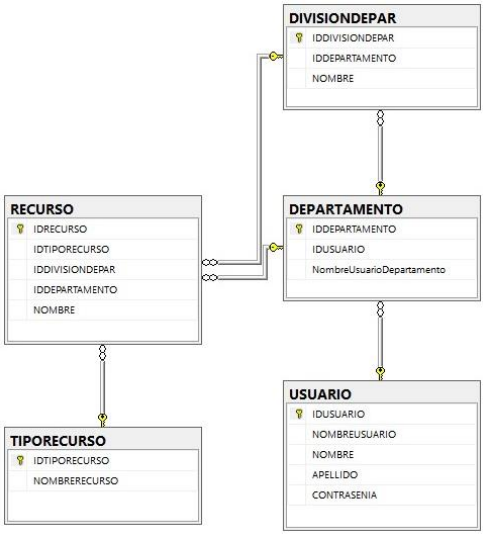


Figura 19. Diagrama de clases de la base de datos

La base de datos se genera por medio de consultas, especificando cada tabla, con sus atributos, claves primarias y foráneas, las líneas de código que se muestran en la Figura 20, son un ejemplo de cómo crear la tabla usuario:

```
create table USUARIO (
   IDUSUARIO            integer              not null,
   NOMBREUSUARIO        varchar(20)          null,
   NOMBRE               varchar(50)          null,
   APELLIDO             varchar(50)          null,
   CONTRASENIA          varchar(10)          null,
   constraint PK_USUARIO primary key (IDUSUARIO)
)
go
```

Figura 20. Código para crear la tabla usuario de la base de datos

Si se desea agregar una clave foranea a una tabla se debe utilizar las líneas de código de la Figura 21:

```
alter table DEPARTAMENTO
   add constraint FK_DEPARTAM_REFERENCE_USUARIO foreign key
(IDUSUARIO)
      references USUARIO (IDUSUARIO)
go
```

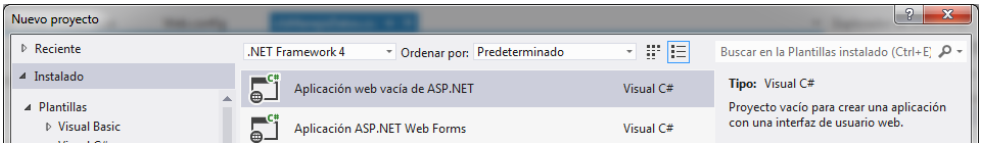
Figura 21. Código para agregar al campo IdUsuario como clave foránea de la tabla Departamento

El código para generar las tablas faltantes es similar al presentado, por lo tanto, se escribió el código completo en un script para luego generar la base de datos ejecutando el script en una query de Microsoft SQL Server Managment Studio, obteniendo como resultado la base de datos dbSmartHome.

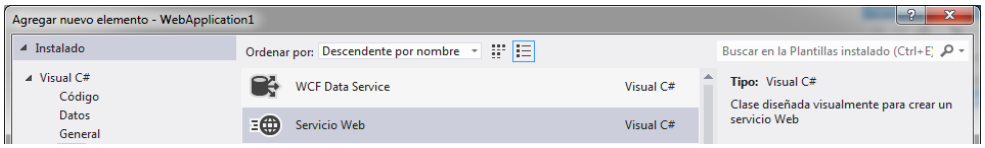
3.3.2. Diseño del servicio web.

La función del servicio web, es consultar los datos almacenados en dbSmartHome utilizando el protocolo de comunicación SOAP, este proceso corresponde al subsistema de comunicación de la Figura 18 y claramente se puede visualizar que tiene dos niveles.

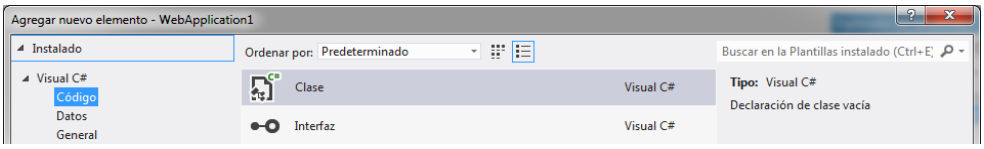
La ventaja de utilizar Visual Studio, es que entre las plantillas se puede encontrar una para Web con ASP.NET, esta fue la que se utilizó; una vez que se tiene cargada la plantilla, se le agrega una clase diseñada visualmente para crear un servicio web (*.asmx) y una clase para poder obtener los datos de la base de datos.



a) **Plantilla para crear el servicio web**



b) **Clase para visualizar el servicio web**



c) **Clase para obtención de datos de base para servicio web**

Figura 22. Plantilla y elementos para creación de servicio web

Las dos primeras clases utilizan lenguaje de programación C# y la tercera HTML. (Ver Figura 22)

A la plantilla de Servicio Web se la denominó: AAWebProject.sln, se le añadió una clase AAWebService.asmx.cs y una clase clsManejoDatos.cs.

Es primordial declarar primero los métodos de la clase clsManejoDatos, que de acuerdo al diagrama de la Figura 18 tiene dos métodos, se utiliza la librería “System.Data.SqlClient” y las líneas de código de la Figura 23, para implementar como ejemplo el método ObtenerDatosUsuario.

```
public DataSet getDatosUsuario(string username)
{
        DataSet dataTable = new DataSet();
        using (SqlConnection con = new SqlConnection(
        ConfigurationManager.ConnectionStrings["SHDataConnectionString"]
        .ConnectionString))
        {
        con.Open();
        using (SqlDataAdapter sqlAdapter = new SqlDataAdapter("SELECT
        Contrasenia FROM USUARIO WHERE NombreUsuario ='" +
        username.ToString() + "'", con))
                {
                sqlAdapter.Fill(dataTable);}
         } return dataTable;
}
```

Figura 23. Código para obtener los datos del usuario de la base de datos

Se puede observar claramente, que se define el método getDatosUsuario con el nombre de usuario como parámetro, luego se crea un nuevo objeto de la clase DataSet, para posteriormente usando un objeto de la clase SqlConnection se establezca en su constructor la conexión con la base de datos. Se abre la comunicación y con ayuda de un objeto de la clase SqlDataAdapter se realiza la consulta y se devuelve como respuesta la contraseña asociada al usuario.

La implementación del método ObtenerDatosRecursos se hace de la misma forma.

Una vez que se tiene la clase clsManejoDatos lista, se procede a la programación de la clase AAWebService.asmx.cs. Esta clase tiene 7 métodos implementados, cada uno devuelve diferente información correspondiente a cada uno de los usuarios.

Primero se debe iniciar la programación del Servicio Web indicando el nombre del Namespace, luego se crea un objeto de la clase clsManejoDatos, en este caso dataEmployess, con este objeto se invoca el método getDatosUsuarios y se envía como parámetro el usuario. (Ver Figura 24)

```
[WebService(Namespace = "http://ely.org/")]

    clsManejoDatos dataEmployess = new clsManejoDatos();

    [WebMethod]
    public DataSet getDatosUsuario(string usuario)
    {
            return dataEmployess.getDatosUsuario(usuario);
    }
```

Figura 24. Asignación del nombre Namespace del servicio web y declaración del método web

Se debe realizar el mismo procedimiento con cada uno de los métodos restantes. Para crear el formulario web cuya utilidad es permitir de forma más ágil y sencilla el ingreso de los datos de cada uno de los usuarios, a ser

almacenados en la base de datos, se selecciona la plantilla de la Figura 25, y se procede con la redacción del código.

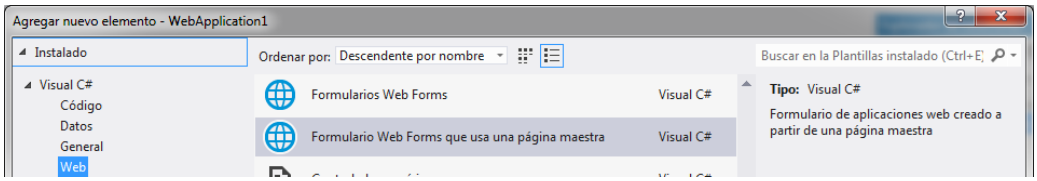


Figura 25. Plantilla para crear el formulario web de registro de datos

Se utilizó una plantilla prediseñada para mejorar el estilo de la presentación, las líneas principales de programación y que se utilizan para el registro de datos en todas las tablas se muestran en la Figura 26:

```
<li class="active">

 <%-- Usuario --%>
   <div class="form" runat="server">
    <input      type="text"      id="txtUsuario"      class="active      textbox"
    value="NombreUsuario"  runat="server"  onfocus="this.value  =  '';"  onblur="if
    (this.value == '') {this.value = 'Nombre de Usuario';}">
    <input  type="text"  id="txtNombre"  class="active  textbox"  value="Nombre"
    runat="server"  onfocus="this.value  =  '';"  onblur="if  (this.value  ==  '')
    {this.value = 'Nombre';}">
    <input type="text" id="txtApellido" class="active textbox" value="Apellido"
    runat="server"  onfocus="this.value  =  '';"  onblur="if  (this.value  ==  '')
    {this.value = 'Apellido';}">
    <input type="text" id="txtPassword" class="active textbox" value="Contraseña"
    runat="server"  onfocus="this.value  =  '';"  onblur="if  (this.value  ==  '')
    {this.value = 'Contraseña';}">
     <input         type="submit"         runat="server"         value="Registrar"
onserverclick="clkRegistrarUsuario">
    </div>
</li>
```

Figura 26. Código para crear la pestaña de ingreso de datos del usuario en el formulario web

Este formulario, contiene cuatro pestañas, en la primera se ingresa la información del usuario, en la segunda la información de las divisiones de la casa, en la tercera la información de ubicación de los recursos en cada división de la casa y la cuarta el catálogo de recurso. El diseño se presenta en la Figura 27.

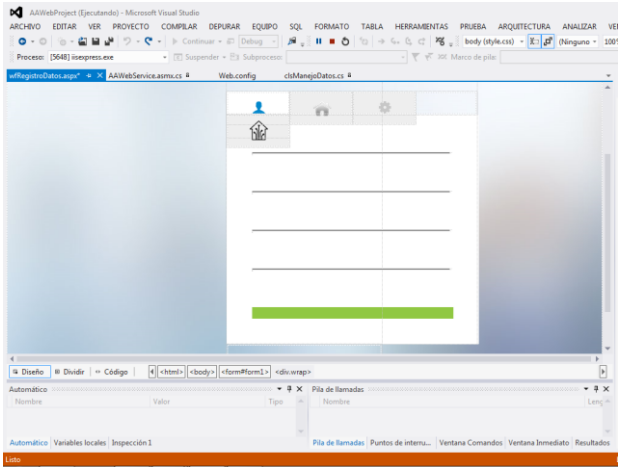


Figura 27. Diseño del formulario web para el ingreso de datos

Con esto prácticamente se tiene listo el servicio web utilizando SOAP, para ser consumido por la interfaz inteligente que se diseña más adelante. Como se ha indicado, solamente se ha tomado el ejemplo de creación de las tablas y métodos para la información de usuario, puesto que para lo demás se sigue el mismo lineamiento.

3.4. Diseño de la interfaz inteligente

Con la base de datos y el servicio web listos, se procede con la parte esencial de este proyecto, el diseño de la interfaz inteligente, para este efecto se utilizará la diagramación del subsistema de interfaz del PIM. (Ver Figura 28).

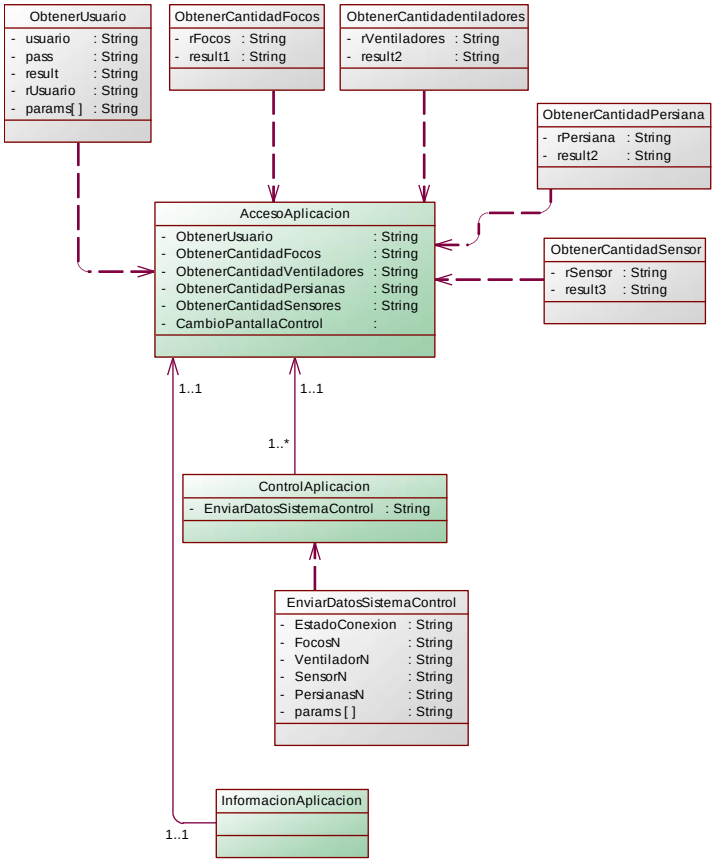


Figura 28. Diagramación UML para el desarrollo de la interfaz inteligente

Se ha optado por utilizar Android, debido a que últimamente la mayoría de los Smart TV han adoptado a este como su sistema operativo, y la demanda de las aplicaciones desarrolladas bajo este lenguaje van en aumento, además Android cuenta con su propio entorno de desarrollo, el mismo que se ha utilizado para crear SmartHomeTv.

Para comenzar con el desarrollo de la aplicación se debe crear un nuevo proyecto de Android Studio, colocar el nombre de la aplicación, en este caso SmartHomeTV, seleccionar el SDK (Software Development Kit) mínimo, Api 16, se debe tomar en cuenta que mientras más alto se elija el nivel de API, en menos dispositivos funcionará.

Se selecciona una Actividad en blanco, se le da un nombre a la actividad y se finaliza, en este caso la primera clase se denomina LoginActivity.java y el layout relacionado a ella login.xml.

Es de suma importancia que se de permisos de internet a la aplicación, esto se configura insertando la línea de código de la Figura 29, en el archivo AndroidManifest.xml:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Figura 29. Código para darle permisos de Internet a la aplicación

Posteriormente se debe agregar al proyecto la librería que permitirá el consumo de recursos a través del servicio web con SOAP. Para este efecto se debe copiar el .jar de ksoap2, que se ha descargado de internet, dentro de la carpeta libs, dar clic derecho sobre el archivo y seleccionar la opción “Add As Library”.

Figura 30. Diseño del layout para el Loggin

Lo siguiente es generar la pantalla de layout, esta es la que el usuario podrá ver cuando inicie la aplicación, este diseño es realizado en base a los gustos y necesidades del autor. (Ver Figura 30).

La clase LogginActivity es la que se contiene el código que permite que los objetos insertados se muestren en la pantalla, tomen datos y los envíen. Cabe recalcar que la clase debe ser pública y extendida a una actividad. Luego, se debe declarar las variables y objetos globales necesarios, en este caso se muestra algunas líneas de código como un ejemplo (Ver Figura 31):

```
public class LoginActivity extends Activity {

// Botón que permite verificar la contraseña devuelta por el método
y la ingresada en el cuadro de texto
private Button btnVerificar;

//Variable que guarda la contraseña del Usuario y la envía a la
siguiente clase
public String rUsuario;

// Objeto que permite obtener los datos devueltos por la tarea
asíncrona en el post execute.
private Handler mHandler = new Handler();
```

Figura 31. Código para declarar variables y objetos globales

El método onCreate viene declarado por default, aquí es necesario declarar los elementos con los que se desea que el usuario interactúe, y las imágenes a ser mostradas, como se indica en la Figura 32, se debe considerar que las imágenes deben estar guardadas en la carpeta "drawable" del proyecto para poder ser utilizadas.

```
@Override
protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);
    setContentView(R.layout.login);

    ImageView imgLogosh = (ImageView)findViewById(R.id.imgLogosh);
```

Figura 32. Declaración del método onCreate

Cada uno de los objetos tiene propiedades, en este caso, se utiliza la propiedad OnClick, para que cuando se haga clic sobre el botón verificar, se ejecuten las líneas de código que se especifican dentro de este método y se muestran en la Figura 33.

```
@Override
public void onClick(View view)
{
        //Se declara el campo de texto en el que se ingresa el
        usuario, se obtiene el texto y se lo almacena en una variable.
        String usuario
        =((EditText)findViewById(R.id.txtUsuario)).getText().toString(
        );


        //Se declara el campo de texto en el que se ingresa la
        contraseña, se obtiene el texto y se lo almacena en una
        variable.
        String pass =
        ((EditText)findViewById(R.id.txtContraseña)).getText().toStrin
        g();

        //Se crea un vector que almacena los parámetros a ser enviados
        a la tarea asíncrona, es sumamente importante que la dirección
        ip que se indica, sea la de la máquina que contiene el
        servidor web.
       String[] params = new String[]{"192.168.0.101", usuario};

        //Se ejecuta la tarea asíncrona.
       new MyAsyncTask().execute(params);
```

Figura 33. Código para que cuando se de el evento clic se ejecute la tarea asíncrona en base al parámetro usuario enviado

Se comprueba que la contraseña que se ha ingresado en el campo de texto sea igual a la que se obtuvo como resultado de invocar la tarea asíncrona, en el caso de ser verdadero, crea un objeto Intent para hacer el cambio de la pantalla y se envía el nombre de usuario (Ver Figura 34).

```
if (pass.equals(rUsuario))
{
Intent nuevapant = new
Intent(LoginActivity.this,ControlActivity.class);
nuevapant.putExtra("NUsuario",usuario);
startActivity(nuevapant);
}
```

Figura 34. Código para comprobar que la contraseña del usuario que se ha ingresado sea la correcta

Para tener información del uso de la aplicación se creó un botón que aparece en la parte superior derecha, mediante las líneas de código de la Figura 35.

```
public void InfoPant(View trans){
    if (trans.getId() == findViewById(R.id.btnInfo).getId()) {
        Intent i = new Intent(this,InformacionActivity.class);
        startActivity(i);
    }
}
```

Figura 35. Código para mostrar pantalla de información de SmartHomeTv

La parte esencial del código, con el que se puede obtener los datos, corresponde a la tarea asíncrona que se ejecuta en segundo plano para evitar que posibles errores al momento de la ejecución puedan provocar que la aplicación se detenga drásticamente, además al hacerlo en segundo plano no se consume tantos recursos como si se ejecutara en el método principal, el fragmento de código se puede observar en la Figura 36.

```
class MyAsyncTask extends AsyncTask<String, Void, String> {
    public String SOAP_ACTION = "http://ely.org/buscarUsuario";
    public String OPERATION_NAME="buscarUsuario";
    public String WSDL_TARGET_NAMESPACE="http://ely.org/";
    public String SOAP_ADDRESS;
    private SoapObject request;
    private HttpTransportSE httpTransport;
    private SoapSerializationEnvelope envelope;
    Object response = null;
```

Figura 36. Asignación de los encabezados respectivos para la solicitud que realiza la tarea asíncrona

En esta porción de código se puede observar, que se debe indicar los nombres requeridos para que la operación SOAP se ejecute, es decir, se debe indicar SOAP_ACTION, OPERATION_NAME, y WSDL_TARGET_NAMESPACE, luego se crea un objeto SOAP para la solicitud, un objeto para transportar por http y un objeto contenedor tipo SOAPSerializationEnvelope.

Con estos datos especificados se procede a ejecutar en segundo plano todo el proceso, en la implementación del método doInBackground se debe especificar el nombre del Servicio Web, que sirve como puente para la comunicación y con la implementación del código de onPostExecute se

obtiene la respuesta a la solicitud SOAP que se realiza, el valor se almacena en una variable global y se lo utiliza para poder consultar el resto de información que se necesita. (Ver Figura 37)

```
    @Override
    protected String doInBackground(String... params){
        SOAP_ADDRESS= "http://"+params[0]+"/AAWebService.asmx";
        request = new SoapObject(WSDL_TARGET_NAMESPACE,
OPERATION_NAME);
        PropertyInfo pi = new PropertyInfo();
        pi.setName("usuario");
        pi.setValue(String.valueOf(params[1]));
        pi.setType(String.class);
        request.addProperty(pi);
        pi = new PropertyInfo();
        envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
        envelope.dotNet= true;
        envelope.setOutputSoapObject(request);
        httpTransport = new HttpTransportSE(SOAP_ADDRESS);
        try{
            httpTransport.call(SOAP_ACTION, envelope);
            response = envelope.getResponse()}
        catch(Exception exp){
            response= exp.getMessage()}
        return response.toString()}
    @Override
    protected void onPostExecute(final String result){
        super.onPostExecute(result);
        mHandler.post(new Runnable(){
            @Override
            public void run(){
                rUsuario = result;} } ); } }
```

Figura 37. Código para implementar el método doInBackground y obtener la contraseña del usuario ingresado

Con el nombre de usuario es posible obtener la cantidad de recursos de cada tipo, información que el usuario hizo registrar en la base de datos, solamente se debe indicar el nombre de cada método y enviar como parámetro el nombre de usuario.

Pasando a la clase de control, se tiene que, esta clase obtiene mediante las variables tipo String FocosN, VentiladorN, SensorN, String PersianaN, UsuarioN, todos los datos que se enviaron de la clase LoginActivity consultando con el nombre de usuario como parámetro, estos resultados permiten la generación automática de la interfaz.

Figura 38. Ubicación inicial de los objetos antes de generar la interfaz inteligente

La interfaz utilizando dos variables y con la ayuda de la importación de las librerías android.view.Display, android.view.WindowManager y android.content.Context; obtiene el largo y ancho de la pantalla, luego lo divide en porciones tanto en el eje X como en el eje Y, y va moviendo los objetos hasta que queden centrados.

Como ejemplo, si el usuario uno, registra que tiene cuatro luces, un ventilador y una persiana, en la pantalla de control se muestra el mismo número de elementos, como indica la Figura 39, pero si el usuario dos tiene una sola luz, entonces la interfaz será como la que se indica en la Figura 40.

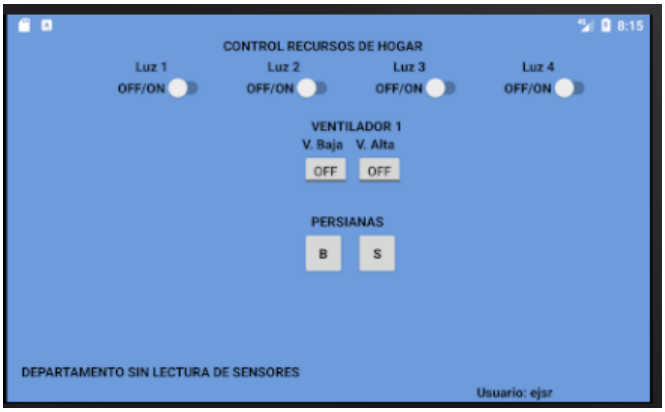


Figura 39. Interfaz generada para usuario uno

Adicional, en la interfaz se indica si el usuario no tiene alguno de los 4 tipos de recursos que se permite, y se muestra el nombre del usuario.

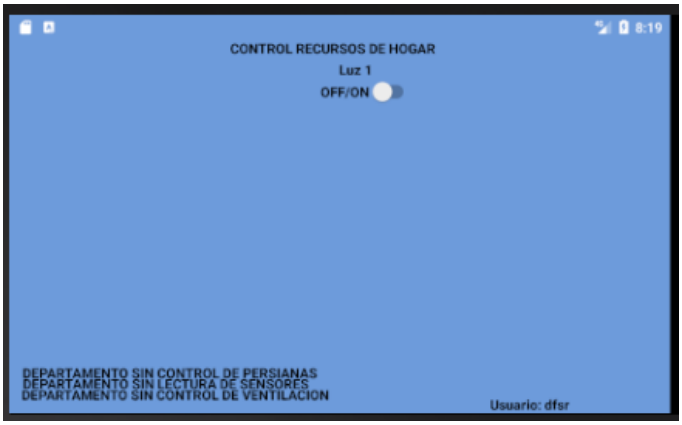


Figura 40. Interfaz generada para usuario dos

Esto permite que el espacio físico y real en el que se puede implementar el sistema SmartHomeTV no se limite, por el momento, está diseñado para controlar cinco recursos de cada tipo, número que se podría ampliar en el caso de que el desarrollador lo necesite.

```
private class Background_get extends AsyncTask<String, Void, String>
{
    @Override
    protected String doInBackground(String... params) {
        try {
            URL url = new URL("http://192.168.0.103/?" + params[0]);
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            estado = url.toString();

            BufferedReader in = new BufferedReader(new
InputStreamReader(connection.getInputStream()));

            StringBuilder result = new StringBuilder();
            String inputLine;
            while ((inputLine = in.readLine()) != null)
                result.append(inputLine).append("\n");
            in.close();
            connection.disconnect();
            return result.toString();
        } catch (IOException e) {
            e.printStackTrace();}
        return null;    }
}
```

Figura 41. Método de tarea asíncrona para comunicar la interfaz con el sistema de control

La Figura 41, en una de las líneas de programación muestra una dirección IP que se coloca en el URL, esta debe ser la misma que se le asigna a la placa Arduino.

```
<string name="app_name">SmartHomeTV</string>
<string name="control">SmartHomeTV</string>
<string name="informacion">SmartHomeTV</string>
```

Figura 42. Configuración del archivo string.xml

3.5. Diseño del hardware

El diseño del Hardware inicia con la selección de los elementos a utilizar, que están relacionados directamente con las actividades de control. Por lo que se determina:

- Selección de controlador
- Cantidad de actuadores a gobernar y sensores a leer.
- Modo de comunicación entre el controlador y la aplicación SmartHomeTV.

3.6. Selección de controlador

Se ha optado por utilizar una placa Arduino ya que presenta facilidad de integración con módulos de control y comunicación, adicional el mismo puede comunicarse con aplicaciones creadas con el entorno de desarrollo de software Android.

La placa utilizada es Arduino MEGA 2560, cuyas características se detallan en la Tabla 4. Otro factor de selección fue su bajo costo. (Ver Figura 43)

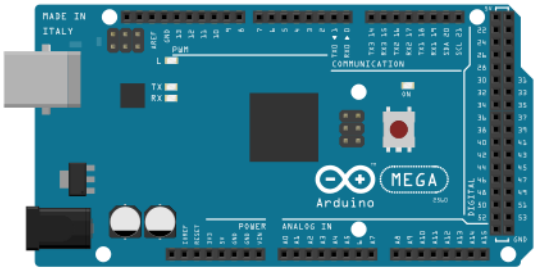


Figura 43. Placa Arduino Mega 2560

3.6.1. Cantidad de actuadores a controlar y sensores a leer.

La cantidad de dispositivos a controlar se encuentran relacionados con la cantidad de salidas digitales o analógicas, por lo que su número no podrá exceder la capacidad de pines del Arduino Mega 2560 detallados en la Tabla 4.

Para el desarrollo del proyecto se ha determinado que los elementos a controlar son:

- Iluminación: Con un máximo de cinco dispositivos de iluminación, por lo tanto, se utiliza un máximo de cinco salidas digitales. El tipo de control que se realiza es ON/OFF.
- Ventilación: Con un máximo de cinco dispositivos de ventilación, es decir, se utiliza un máximo de cinco salidas PWM digitales. Se controla dos velocidades con ON/OFF.
- Control de persianas: Con un máximo de cinco persianas, por lo tanto, se utilizará un máximo de 10 salidas digitales. El motor gira mientras se mantiene presionado el botón de la interfaz. Se ubica un final de carrera para controlar que la persiana ha subido en su totalidad y el motor no se mantenga girando.
- Alarma: Un pin digital, se activa o desactiva cuando se abre o cierra una puerta o una ventana respectivamente.

Es necesario mencionar que los elementos a controlar no operan con el mismo voltaje, se trabaja con fuentes de alimentación de 24 VDC y 9VDC, por lo tanto, se ha utilizado dos módulos de relés uno de cuatro canales y

uno de dos canales para la conexión, la función que cumplen los módulos de relés es separar la etapa de control de la etapa de potencia. Estos módulos de relés tienen la capacidad de soportar altas cargas ya que su corriente de trabajo es de 10 A a 250 VAC, por lo que se puede conectar cualquier tipo de dispositivo ya sea con alimentación AC o DC. Adicional dichos módulos se encuentran constituidos por opto acopladores, transistores y diodos conectados con polaridad invertida a las bobinas para la protección de los circuitos.

Para la lectura de los sensores:

- Control de presencia: Se ha determinado un máximo de cinco sensores, se utiliza un máximo de 5 entradas digitales.

3.6.2. Modo de comunicación entre el controlador y la aplicación SmartHomeTV.

Para la comunicación entre el controlador y la aplicación SmartHomeTV se ha considerado que el dispositivo de mando en este caso una Smart TV deberá estar conectada a la red, para dicha acción se ha usado el módulo Ethernet Arduino cuyas características se detallan en la Tabla 4. (Ver Figura 44)

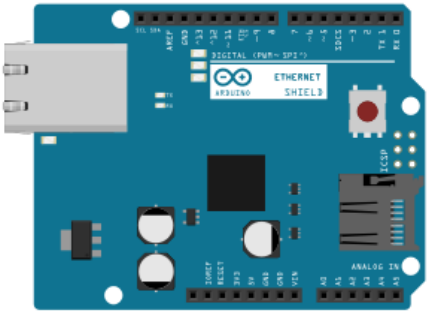


Figura 44. Módulo shield de Ethernet para Arduino

La conexión del mismo es sencilla ya que el módulo de Ethernet se monta en placa Arduino MEGA, como lo indica la Figura 44.

El montaje del Arduino con el módulo shield de Ethernet es relativamente fácil, solamente se debe colocar el módulo sobre la placa puesto que los pines están listos para su conexión. (Ver Figura 45)

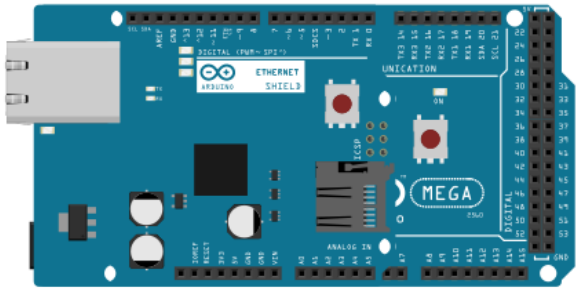


Figura 45. Montaje del módulo de Ethernet sobre Arduino Mega

El circuito con la placa y el módulo respectivamente queda como se indica en la Figura 46.

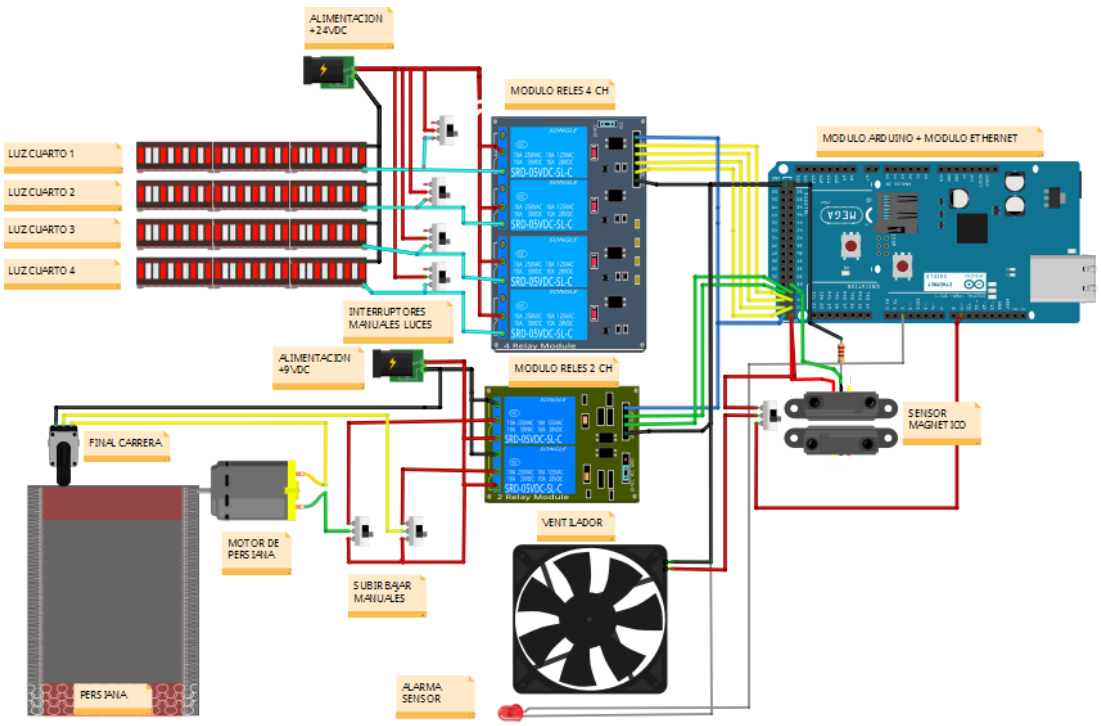


Figura 46. Circuito del Sistema de Control

3.7. Diseño del software del controlador

El esquema de diagramas UML que responde al diseño del software del controlador responde al subsistema de hardware. (Ver Figura 47).

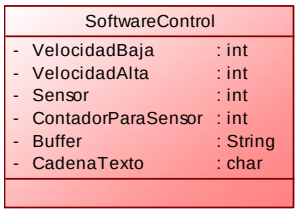


Figura 47. Diagramación UML para el diseño del software del controlador

Además para este diseño se ha utilizado el entorno de desarrollo Arduino Genuino V1.8.2, el mismo que dispone de varias librerías que permiten la integración con el módulo Ethernet.

Las librerías utilizadas para el uso del módulo Ethernet son: #include <SPI.h>, la cual es un protocolo serial asíncrono que permite la comunicación con varios dispositivos, y la librería #include <Ethernet.h> que permite la configuración del módulo Ethernet y la conexión a la red.

A continuación la Figura 48 indica el diagrama de flujo que caracteriza la programación:

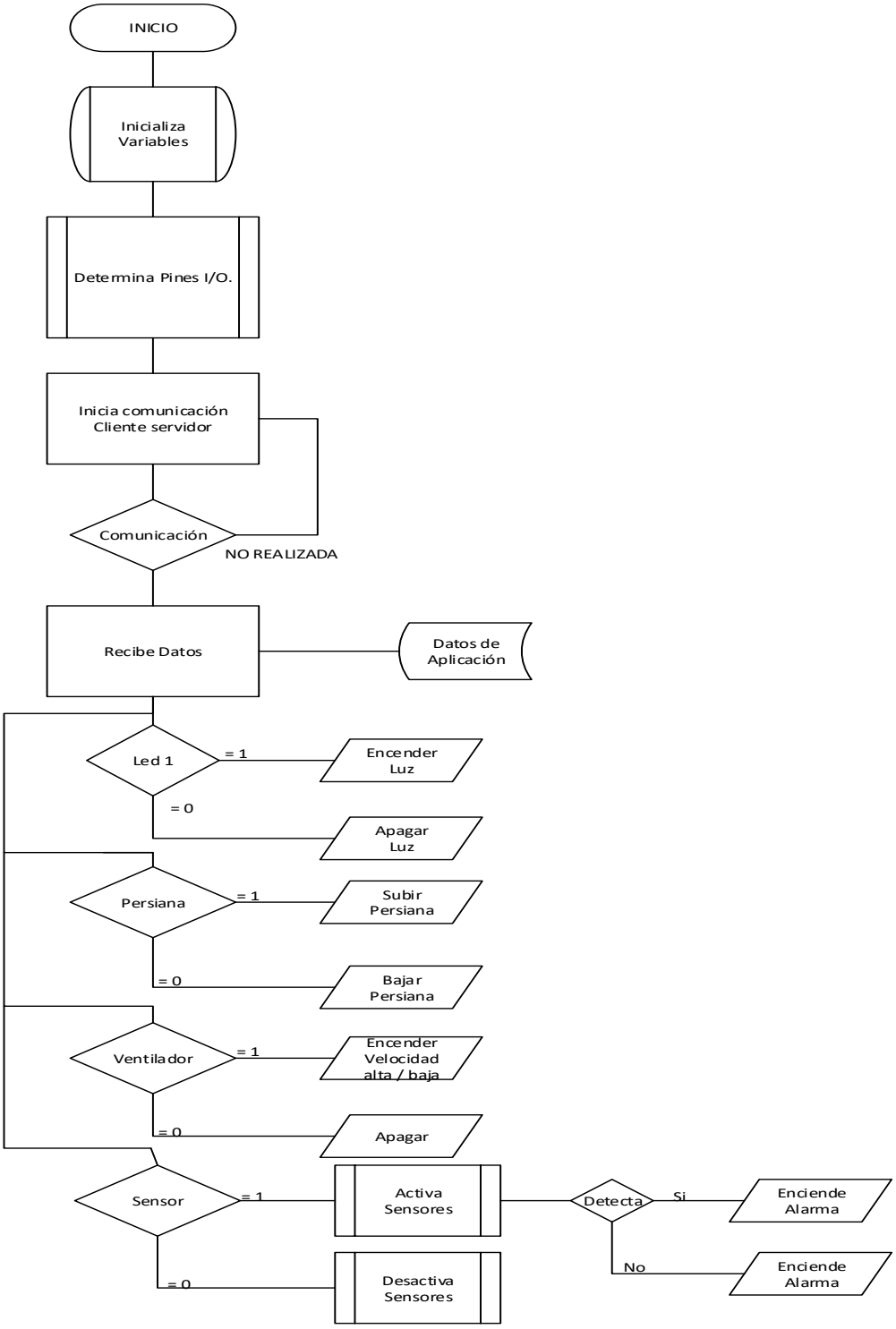


Figura 48. Diagrama de flujo del software del controlador

La programación para el controlador inicia con la declaración de las variables, y se determina los pines de entrada y salida a utilizar. En la Figura 49 se indica el modo de asignación de un pin para salida o entrada.

Es indispensable asignarle una dirección IP al módulo Ethernet de Arduino, para que esté en la capacidad de conectarse a la red.

```
IPAddress ip(192, 168, 0, 103);


pinMode(2, OUTPUT); // Pin salida de alarma
pinMode(28, INPUT);  // SENSOR
```

Figura 49. Asignación de dirección IP y pines de la tarjeta Arduino

Una vez realizada la configuración de los pines se inicia la comunicación serial y el servidor, utilizando las líneas de código mostradas en la Figura 50.

```
Serial.begin(9600);
Ethernet.begin(mac, ip);
server.begin();
```

Figura 50. Código para iniciar la comunicación del puerto serial, la del servidor y la comunicación Ethernet a partir de la MAC e IP del módulo

La programación principal del software se hace en el método void loop(), para el control de la velocidad del ventilador se utiliza la función map. (Ver Figura 51).

```
vbaja= map(111,0,255,0,255); //VELOCIDAD BAJA
valta= map(191,0,255,0,255); //VELOCIDAD ALTA
```

Figura 51. Función para mapear la velocidad del ventilador en alta y baja

Se crea un objeto de la clase EthernetClient, para saber si el servidor está disponible y si lo está, se envía una línea en blanco; se declara una variable de tipo string para enviar datos por el buffer y mientras el cliente esté conectado se envía la cadena de conexión que define el protocolo de comunicación de HTTP. (Ver Figura 52)

```
EthernetClient client = server.available();

if (client) {
  boolean currentLineIsBlank = true;
  String buffer = "";

  while (client.connected()) {

   if (client.available()) {
    char c = client.read();
    buffer += c;

    if (c == '\n' && currentLineIsBlank) {
     client.println("HTTP/1.1 200 OK");
     client.println("Content-Type: text/html");
     client.println("Connection: close");
     client.println();
     break;
    }
   if (c == '\n') {
     currentLineIsBlank = true;
     buffer = "";
```

Figura 52. Código para comprobar la comunicación con el protocolo HTTP

Si existe una línea definida como ‘\n’ se limpia la línea de envío, y si es un retorno de carro ‘\r’ se envía el índice del buffer que va a ejecutar. Las líneas de código de la Figura 53 se utilizan para encender las luces, para los actuadores restantes se utiliza la misma metodología.

```
if(buffer.indexOf("GET /?led1=1")>=0) { // If led1 = 1
        digitalWrite(22, LOW); // led > on
      }
      if(buffer.indexOf("GET /?led1=0")>=0) { // If led1 = 0
       digitalWrite(22, HIGH); // led > off
      }
```

Figura 53. Lectura del buffer para seleccionar valores a escribir en los pines digitales de salida

Para la lectura del sensor, se utiliza la función digitalRead, y en el caso de que detecte un cambio en el estado envía una señal de activación a una alarma. (Ver Figura 54)

```
sensor=digitalRead(28);
  if(sensor ==1){
    digitalWrite(2,LOW);
  }else{
    digitalWrite(2,HIGH);
  }
```

Figura 54. Lectura del sensor de estado de puertas y ventanas

Por último, cuando se completa la programación del software, se verifica que no contenga errores y se sube a la placa Arduino, con esto, el hardware y software del hardware están listos para ser usado.

3.8. Especificaciones técnicas

Las especificaciones técnicas mínimas del equipo que se necesita para montar el servidor son:

- Procesador: Intel Core i3 2620M de 2.1 GHz
- Memoria Ram: 4 GB
- HDD: 750GB
- Sistema Operativo: Windows 7

El router que se necesita para implementar el sistema debe tener como especificaciones técnicas mínimas las siguientes:

- Estándar IEEE802.11n
- 1 puerto WAN 10/100Mbps y 2 puertos LAN 10/100Mbps
- Métodos de configuración criptográfica WPS, WPA2, WPA, WEP
- Admitir el control de ancho de banda (QoS)

Las especificaciones técnicas del Arduino luego de implementar el sistema son:

- 5 pines digitales para 5 luces, con un manejo de voltaje de 24VDC y 500mA.

- 10 pines digitales para 5 persianas con un manejo de voltaje de 9VDC y 500mA.
- 5 pines digitales para 5 sensores, con un manejo de voltaje de 5VDC y 20mA.
- 5 pines digitales pwm para 5 ventiladores, con un manejo de voltaje de 5VDC y 20mA.
- 1 pin digital para 1 alarma, con un manejo de voltaje de 5VDC y 20mA.

Con el sistema ya implementado se tiene aún 28 pines digitales disponibles en los que se podría conectar 20 recursos más, tomando las consideraciones de que para el ventilador se utiliza pines digitales pwm.

CAPITULO IV

PRUEBAS

Este capítulo presenta las especificaciones técnicas y las configuraciones que se llevó a cabo para que sea posible tener disponible el servidor Web, que permite comunicar los componentes de software de la aplicación.

La metodología que se utiliza es a través de la aplicación de la técnica observacional, para comprobar que la interfaz obtenga la información de los usuarios desde la base de datos y adapte los controles dependiendo de esa información.

En la Figura 55 se muestra la arquitectura de la red ya constituida

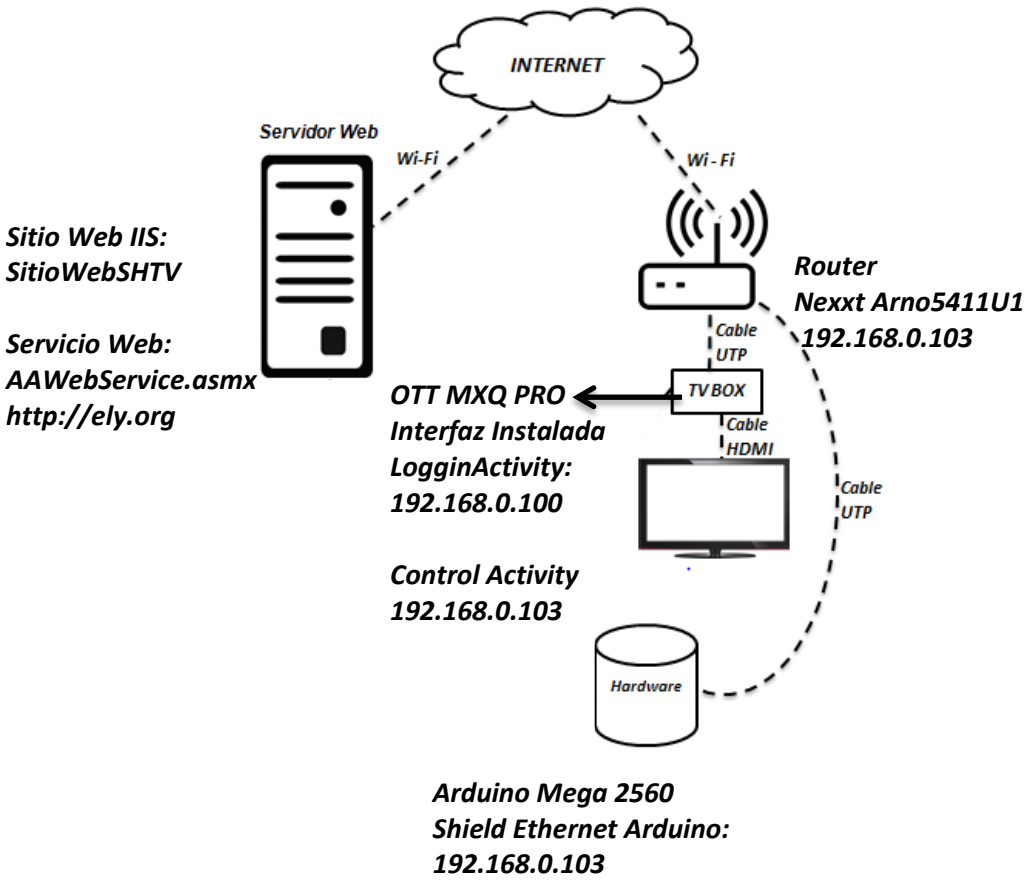


Figura 55. Arquitectura de red del sistema

En la Figura 56, se puede observar un plano de la maqueta con los ambientes definidos.

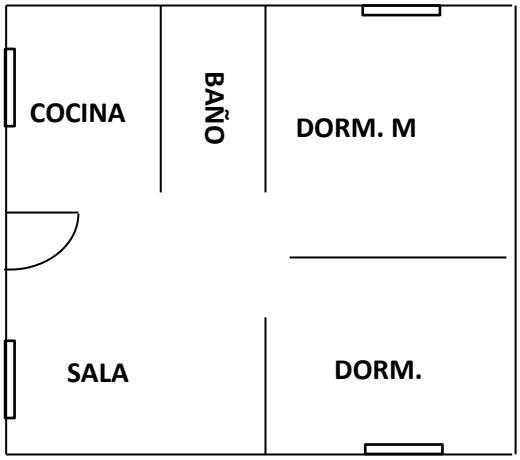


Figura 56. Estructura física y detalle de ambientes

También se documenta el resultado de los procesos que se puso en marcha con la aplicación de pruebas de concepto, se define el ambiente de pruebas que se utiliza y finalmente se realiza el montaje del sistema completo.

4.1. Configuración del servidor

Windows entre las distintas utilidades que presenta, permite que en el panel de control sea posible activar o desactivar sus características, el primer paso es seleccionar la opción de Internet Information Service, esta permite instalar servicios que hagan posible tener en línea un servidor web, adicional se deben marcar todas las casillas que se pueden visualizar en la Figura 57, en el caso de que estén marcadas ya, se las deja así.

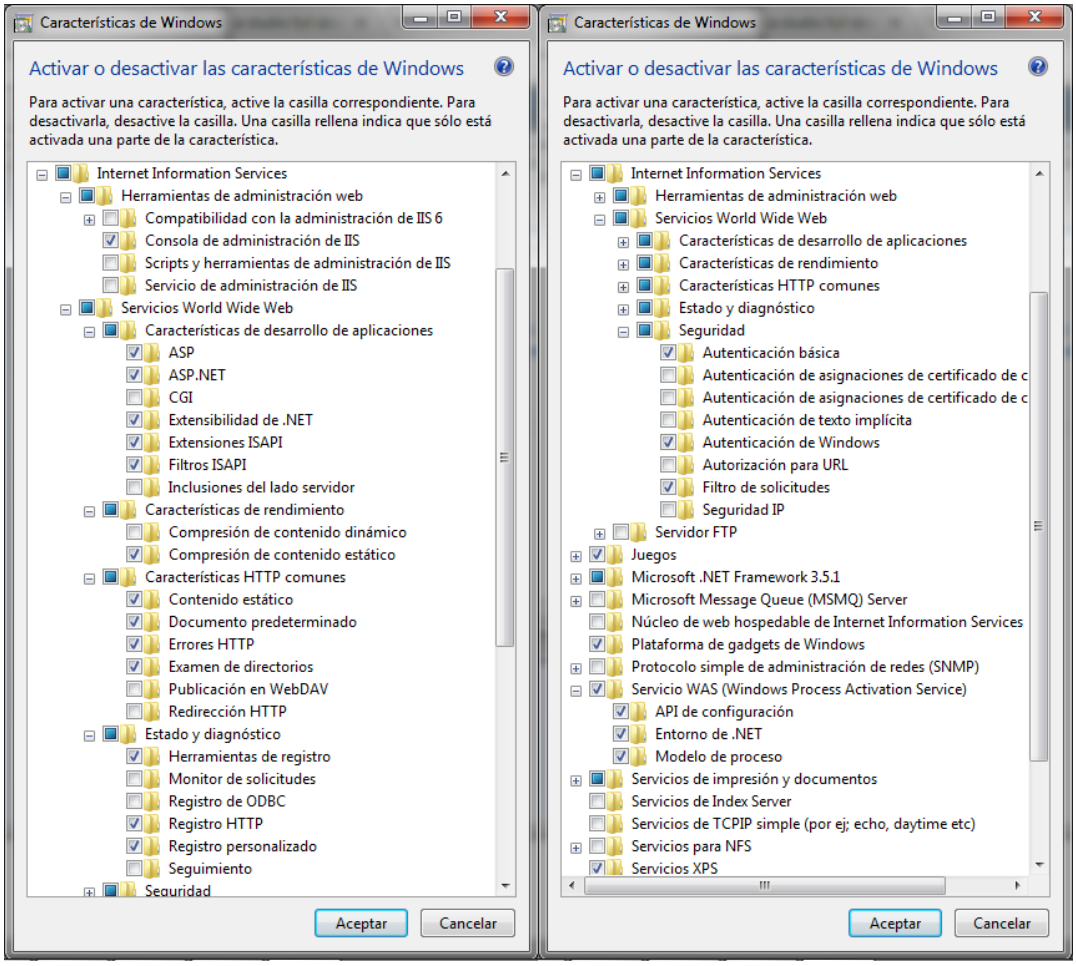


Figura 57. Activación de las características de Windows

Al activar estas características se instala la consola del IIS, la misma que posibilita el manejo de las configuraciones de los servicios web desarrollados con ASP.NET, permite también controlar las características de http comunes, evaluar el estado y diagnóstico de los Servicios World Wide Web, Autentificar mediante Windows y usar Microsoft .NET.

Posterior a la activación es necesario reiniciar el ordenador para poder acceder al Administrador de Internet Information Service (IIS).

Una vez que se está ya en el IIS, se debe cargar el formulario web, para lo que se sigue los pasos indicados a continuación:

1. Se agrega un nuevo sitio web.

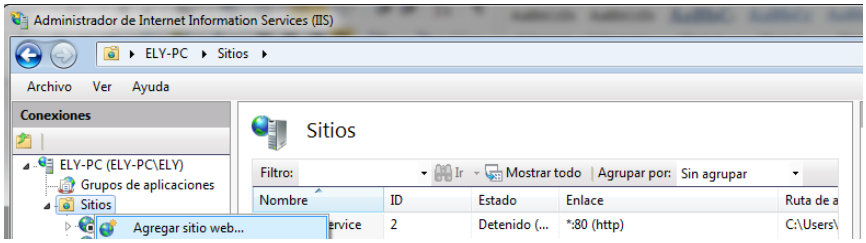


Figura 58. Agregación de un nuevo sitio web

2. Se le asigna un nombre al servidor web, se ubica la dirección física en la que se almacenará el servidor y se autentifica.

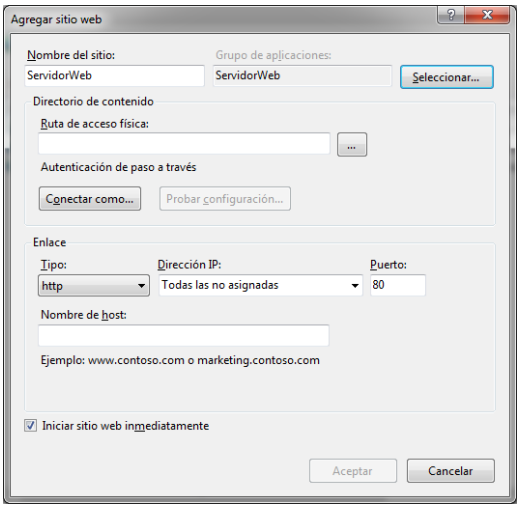


Figura 59. Configuración de las características básicas del servidor web

3. Para autentificar se da clic en “Conectar como…” y se establece un usuario específico:

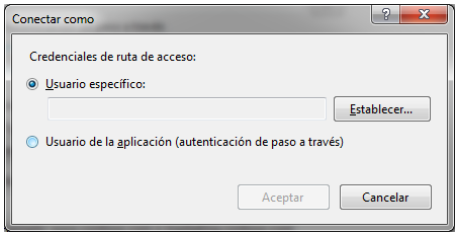


Figura 60. Asignación de usuario para establecer la conexión

4. Se coloca el nombre de usuario, en este caso el nombre de la máquina y la contraseña con la que se inicia la sesión de Windows.

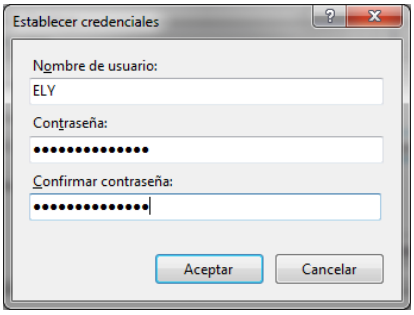


Figura 61. Introducción de los valores con los que se autentifica el inicio de sesión en Windows

5. Se comprueba la conexión y si es correcta se da clic en Aceptar.

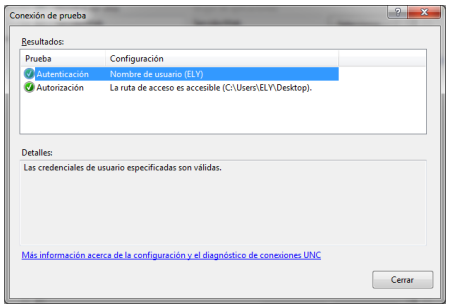


Figura 62. Comprobación de que se estableció la conexión

6. Se observa que el sitio web ya aparece en la consola, es decir, ya se tiene creado el servidor web, en este proyecto el nombre del servidor web a utilizar es SitioWebSHTV.

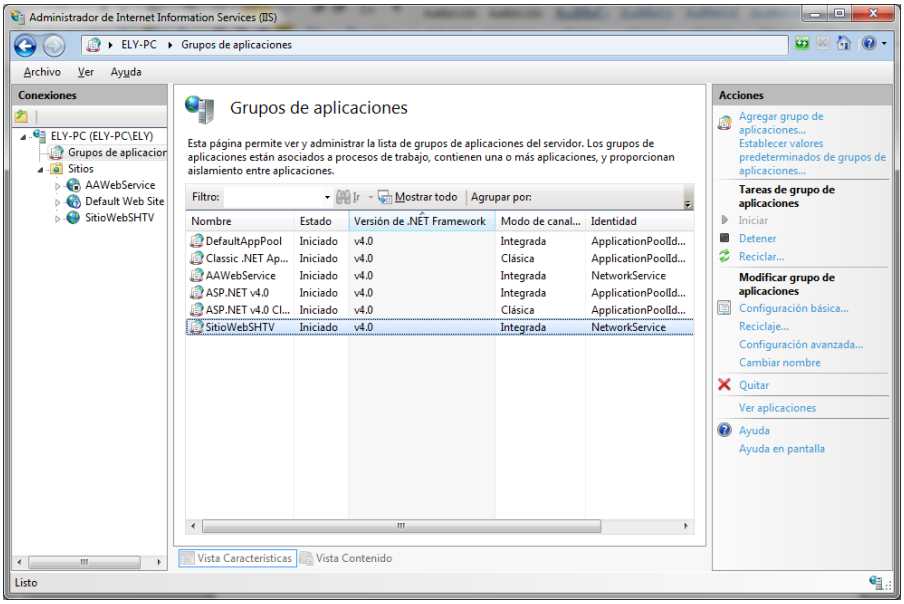


Figura 63. Sitio web creado en el Administrador de IIS

7. Se debe configurar el sitio web con la versión 4.0 de .NET Framework y la Identidad debe ser tipo NetworkService, esto se cambia en las configuraciones avanzadas, que aparecen al dar clic derecho sobre el sitio.

8. Para poder verificar su correcto funcionamiento desde la consola, es necesario instalar la versión 4.0 para el ASP.NET ingresando en el cmd con permisos de administrador y ejecutar la línea de código que se muestra en la Figura 61.

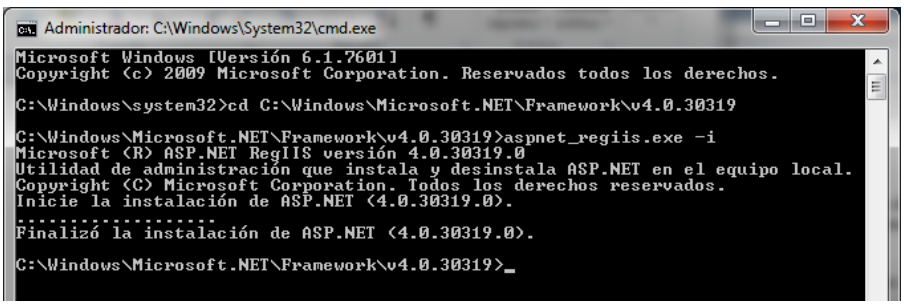


Figura 64. Instalación de la v4.0 de ASP.NET para el Framework

9. Dando clic derecho sobre el servidor web, se accede a las propiedades, y en la pestaña seguridad se edita los permisos, para agregar el de “Servicio de red”.

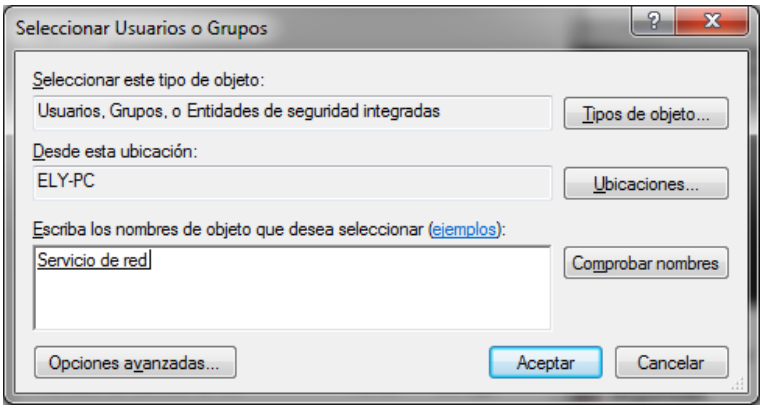


Figura 65. Adición de los permisos para los servicios de red

10. Se acepta y se da marca las casillas para proporcionar al permiso de servicio de red control total.

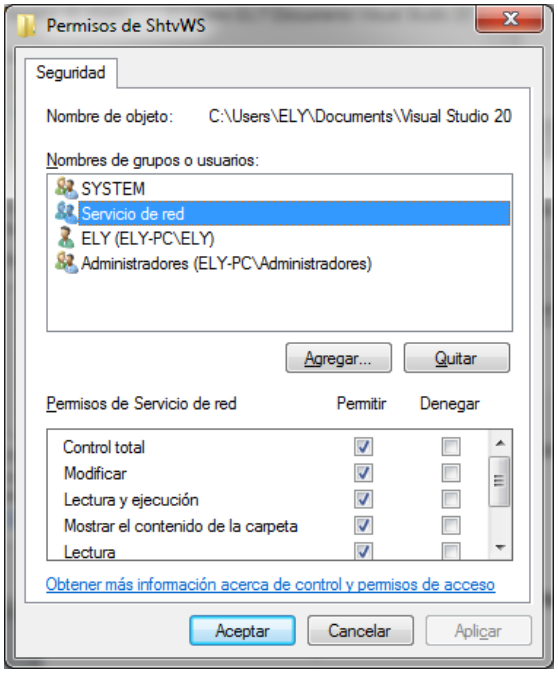


Figura 66. Permisos de servicio de red con control total

11. Se reemplaza los permisos anteriores por los nuevos y se acepta.

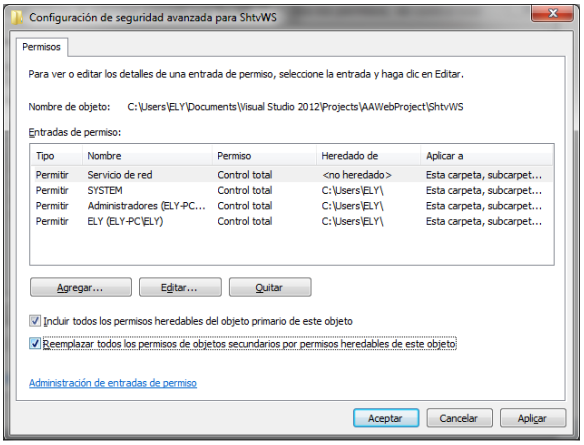


Figura 67. Reemplazo de los permisos anteriores por los nuevos para servicio de red

12. Se acepta y con esto, el servidor está listo para ser probado.

4.2. Pruebas de concepto

Al tener implementado cada uno de los componentes que conforman la aplicación es posible ya probarlos de forma individual, para posteriormente realizar las pruebas en conjunto y comprobar el correcto funcionamiento de todo el sistema.

4.2.1. Base de datos

Las pruebas para la base de datos, se realizan primero ingresando a la base de datos desde Microsoft SQL Server Managment, y se registra valores manualmente con la información referente a un usuario en cada una de las tablas, para que esto sea posible se utiliza la opción editar las primeras 200 filas. (Ver Figura 68)

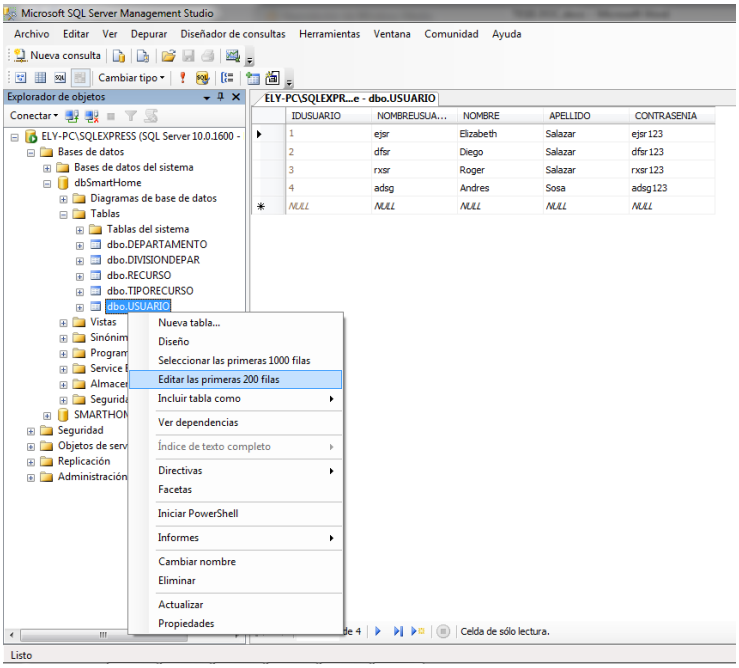


Figura 68. Registro de información en la base de datos dbSmartHome

Se confirma el ingreso de datos, a través de la opción seleccionar las primeras 1000 filas. Observando que están los registros ingresados, se considera como un resultado satisfactorio utilizar esta opción. Luego se prueba la segunda opción, esta es ingresar datos con el formulario web diseñado en visual studio, para lo que usando un navegador web, cargamos la dirección: http://localhost/wfRegistroDatos.aspx, y procedemos con el ingreso de los datos d un usuario al igual que en el primer caso se logra el objetivo.

4.2.2. Servidor Web

Una vez que se tiene todo el código, se ejecuta la plantilla del proyecto desde Visual Studio, esta se carga utilizando un navegador web a través de uno de los puertos como se muestra en la Figura 69, en la que se puede visualizar los métodos implementados.

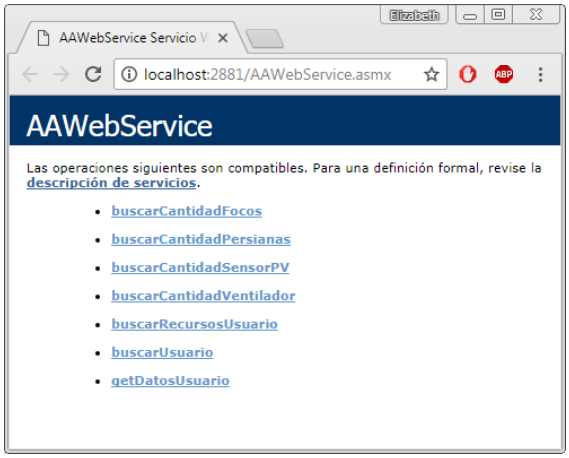


Figura 69. Operaciones del servicio web AAWebService.asmx

Si seleccionamos el método buscarUsuario como ejemplo, se carga la pantalla de la Figura 70, cabe recalcar que en la parte inferior del campo en el que se solicita que se ingrese el nombre de usuario, se detalla en texto como es el formato de una solicitud y respuesta SOAP, luego al colocar el nombre de usuario y dar clic en invocar, se desplegará la información que se solicita, en este caso la contraseña (Ver Figura 71)

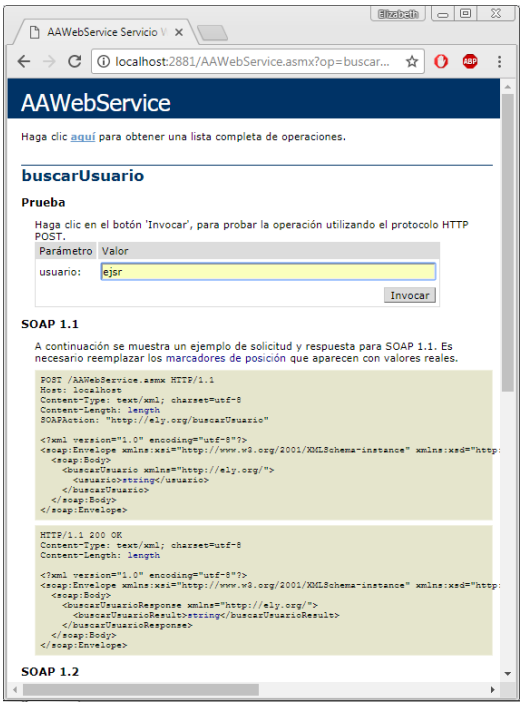


Figura 70. Pantalla para invocar método buscarUsuario con SOAP, con ejemplo de solicitud y respuesta SOAP

Figura 71. Respuesta a la solicitud SOAP

Como se observa en la Figura 72, los métodos del servicio web están operando funcionalmente, ejecutándolas desde Visual Studio. El siguiente paso, es publicar este servicio web para realizar las pruebas respectivas, el proceso se puede observar en las Figuras 73 y 74.

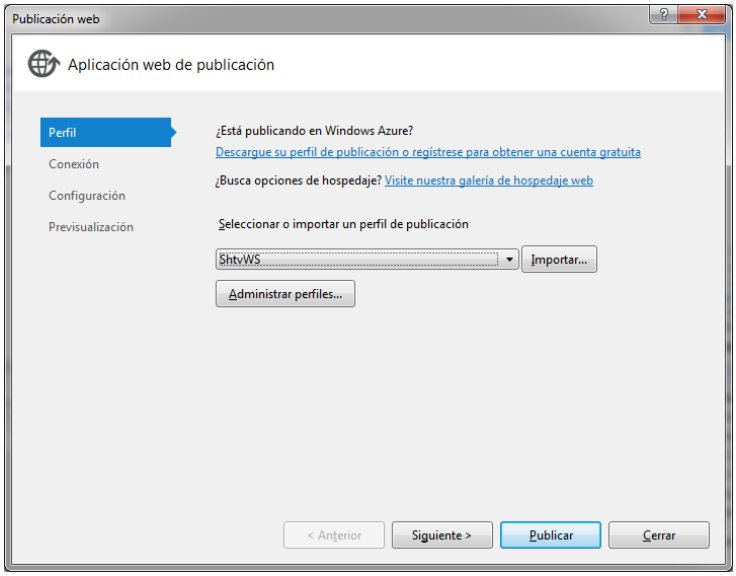


Figura 72. Perfil para la publicación del servicio web

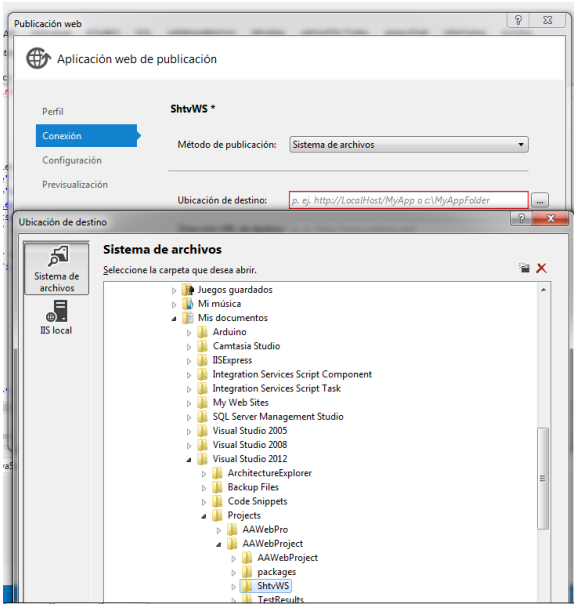


Figura 73. Selección de la ubicación física para publicar el servicio web

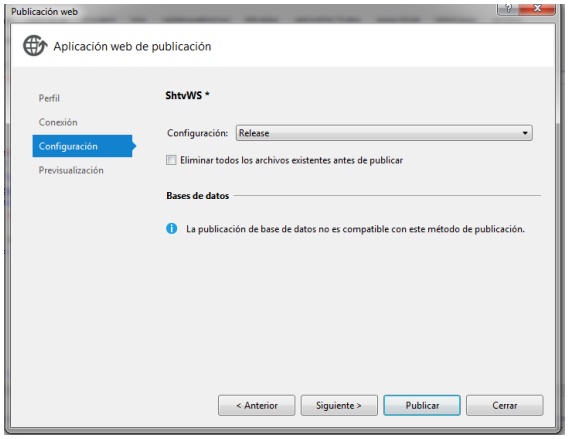


Figura 74. Pantalla para aceptar la publicación del servicio web

Una vez que se tiene publicado el servicio web, se lo configura y añade al IIS como se indica en la sección 4.1 de este capítulo.

Las pruebas para corroborar el correcto funcionamiento, se realizan ingresando a un navegador web y escribiendo en la URL, http://localhost/AAWebService.asmx, para verificar que se cargue el servicio web y http://localhost/wfRegistroDatos.aspx para verificar que se cargue el formulario web de registro. El resultado de estas pruebas se muestra en la Figura 75.

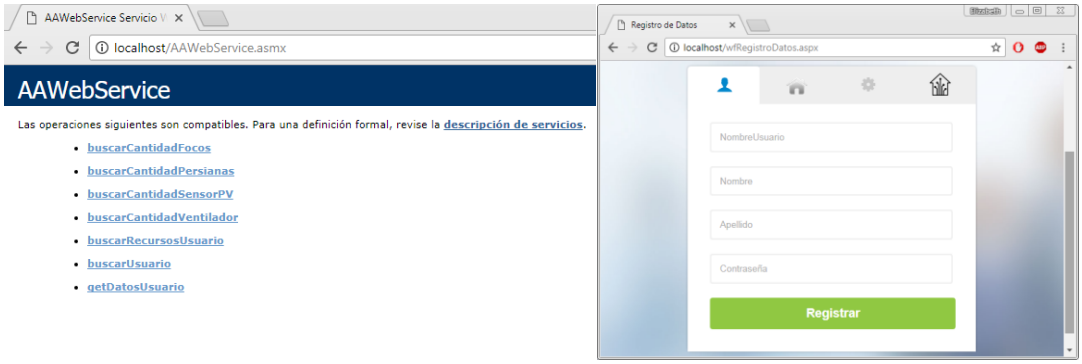


Figura 75. Pruebas del servicio web y formulario web en el localhost

4.2.3. Interfaz inteligente

La interfaz se prueba usando un emulador de celular que proporciona el entorno de desarrollo Android Studio; al usar el mismo sistema operativo que un televisor smart no genera problemas.

Figura 76. Pruebas de la interfaz en el emulador de Android Studio

Como se observa en la Figura 76, las actividades se ejecutan con normalidad y realizan correctamente las funciones, por lo que se considera que las pruebas son validas.

Por último se prueba todo el sistema conectado entre sí usando el ordenador para enviar los datos desde el emulador al sistema de control, y los resultados de estas pruebas son correctas, por lo que se puede notar que el concepto de lo que se deseó realizar es fácil de entender y además cumple su cometido.

4.3. Definición del Ambiente de Pruebas

El ambiente de pruebas se ha definido tomando en cuenta los requisitos básicos para que una persona pueda habitar en un sitio, es decir, se ha tomado en cuenta un departamento que incluya los recursos necesarios que permitan un estilo de vida normal, para tal efecto se realizó una maqueta con una disposición de cinco ambientes, sala, baño, dormitorio master, dormitorio y cocina con comedor, en la que se incluyó ventanas, una cortina y luces.

Esta estructura es realizada con materiales accesibles que permiten una producción fácil y agradable visualmente.

Figura 77. Estructura básica de un departamento hecho maqueta con madera

Se utiliza un Router Nexxt ARN01154U1, al que se le asigna la dirección IP 192.168.0.1. Se ingresa a las configuraciones y se realiza una lista de acceso por MAC a los siguientes dispositivos:

- Portátil con Servidor Web: 192.168.0.100
- Módulo Ethernet Shiel: 192.168.0.103.

El resto de dispositivos no necesitan ser configurados con una dirección IP estática.

4.4. Montaje del Sistema

El proceso de integración del sistema se logra con la ayuda de los siguientes materiales:

- TV BOX OTT
- Router Nexxt
- Arduino Mega 2560
- Shield de Ethernet
- Maqueta de departamento.
- Cables de red
- Elementos electrónicos: 1 led, 4 tiras de leds, 1 motor DC, 1 Ventilador, 5 sensores magnéticos.
- Software de la aplicación: Base de datos, servidor web, interfaz inteligente, controlador.

El proceso de construcción de la maqueta inicia con la estructura de madera que se observa en la Figura 77, a esta estructura se le añade un sistema de cableado para conectar tiras de leds, una en cada cuarto, se coloca el motor DC integrado con un cortina que simula la acción de subir y bajar una persiana, y se coloca los cinco sensores magnéticos en una puerta y cuatro ventanas, todos conectados a una alarma representada por un led que se enciende al cambiar el estado de cerrado por abierto, por último se

coloca un ventilador que puede funcionar en dos velocidades. (Ver Figura 78)

Figura 78. Luces, persiana, ventilador y sensores en maqueta

Continuando con el proceso de montura, se conecta los pines de los sensores y actuadores al sistema de control conformado por el Arduino y el shield de Ethernet y la placa queda como muestra la Figura 79.

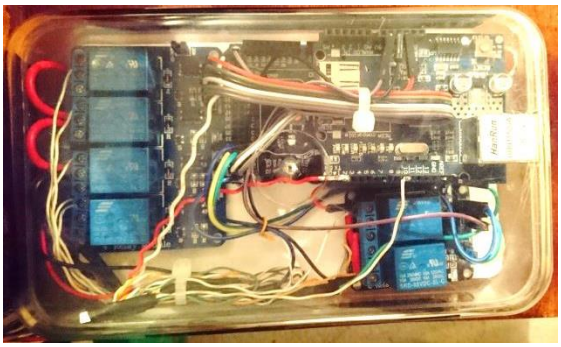


Figura 79. Circuito de control del sistema

Una vez que se tiene el sistema de control, se procede a conectarlo a la red, a través del Router. (Ver Figura 80)

Figura 80. Controlador y router conectados

El siguiente paso es integrar la conexión del TV Box al router. Se instala la Interfaz en el TV Box, y se lo prueba. (Ver Figuras 81 y 82).

Figura 81. Conexión del TV Box al router

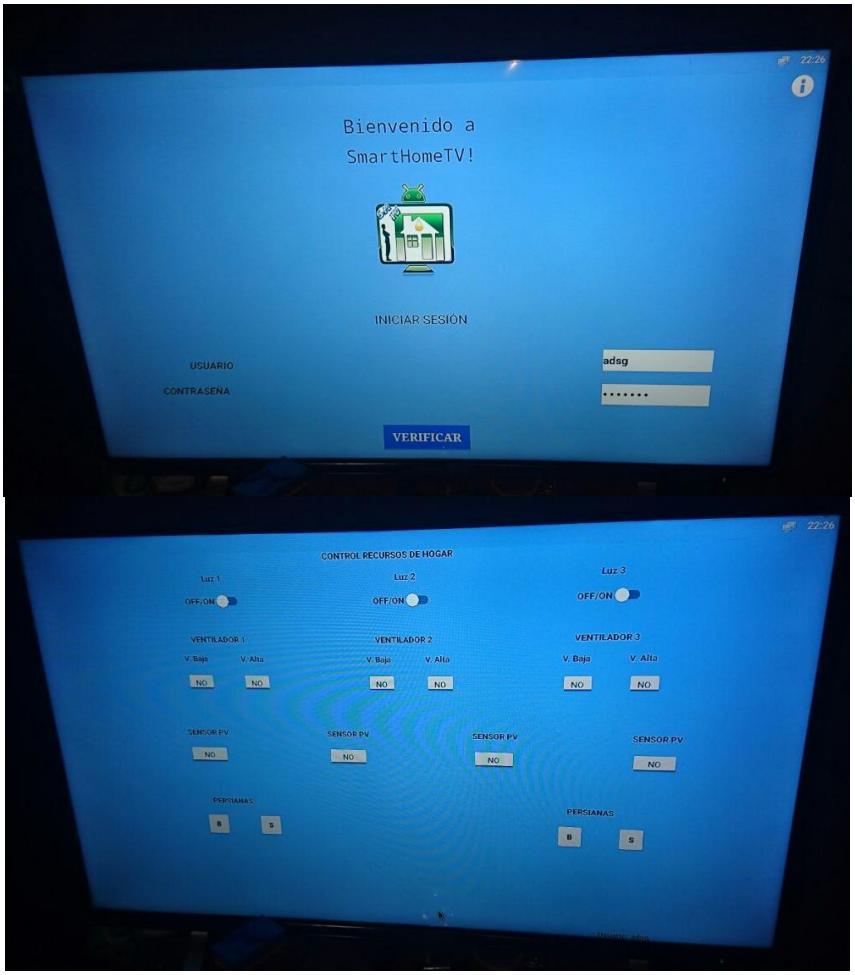


Figura 82. Interfaz inteligente en el Tv Box y mostrada en televisor

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- Se diseñó e implementó una interfaz inteligente que gestiona los recursos del hogar a través de una aplicación desarrollada en Android que se instala sobre un Smart TV, la misma que se diseña utilizando los conceptos de software desarrollado por modelos en sus etapas de establecimiento del CIM y del PIM, y además implementa el protocolo SOAP para la transmisión de datos entre el servidor web y la interfaz adaptativa.

- La ingeniería de modelos permite que se logre la abstracción de las actividades que rigen el dominio de la aplicación SmartHomeTV, para que así se facilite el determinar las etapas del modelo CIM cuyas características se diseñan en el PIM, el mismo que es el más cercano a implementar y que sirve de guía para la generación de las tablas en la base de datos SQL, y la generación del código tanto como para el servicio web de ASP.NET con C#, las clases java en Android y el método de Arduino en C.

- La interfaz de usuario Inteligente funciona a partir de la adquisición de datos de un repositorio, en la que se encuentra almacenada la información personal y de los recursos del hogar del usuario, estos valores no son comunes entre usuarios, por lo que al generarse la interfaz, los objetos toman las dimensiones del ancho y largo de la pantalla y se van ubicando de forma automática para que se visualicen las luces, ventiladores, persianas, y sensores, en el caso de que un usuario no cuente con algún recurso mencionado, la

interfaz muestra un mensaje indicando la ausencia del mismo en la parte inferior de la pantalla.

- El subsistema de hardware del proyecto está compuesto por una tarjeta Arduino que permite realizar la comunicación con la interfaz utilizando el módulo shield Ethernet, recibiendo los datos con el protocolo HTTP y permite el control de los actuadores y la habilitación de sensores, manipulados desde la interfaz.

- El servicio web diseñado es uno de los componentes principales que permite que Android y la base de datos se comuniquen con el protocolo interoperable SOAP, con la inclusión de la librería Ksoap2 que permite el consumo de los servicios almacenados en el repositorio al estar incluida dentro del proyecto de Android y al configurar adecuadamente las características del Administrador de Internet Information Service, permitiendo que se ejecuten las plantillas de los métodos web desde el localhost.

- El sistema de comunicación implementado permite que los protocolos HTPP y SOAP interactúen con el servidor de forma que los subsistemas de registro, interfaz, control y hardware permitan la transmisión y recepción de datos tanto del usuario como de las acciones que el realice permitiendo su visualización en el Smart TV.

5.2. Recomendaciones

- Es preferible que se coloque las direcciones IP de la tarjeta Arduino y del ordenador donde se encuentra el servidor, como estáticas en el Router, así se les asignará la misma dirección IP cada vez que se conecten a la red.

- Se recomienda que el dispositivo de almacenamiento en el que se trasladara la aplicación debe tener un formato FAT 32, para evitar inconvenientes con el TV BOX.

- Se recomienda que para la etapa de potencia se utilice elementos discretos, como SCR, TRIAC o transistores de potencia, para evitar el ruido que generan los relés, adicional estos dispositivos deben ser configurados con opto acopladores para separar etapa de control con etapa de potencia.

- Es recomendable que se analice el uso de una tarjeta Raspberry Pi para realizar el control, ya que la misma posee el módulo Ethernet integrado evitando así el comprar este módulo por separado como es en el caso de Arduino.

REFERENCIAS BIBLIOGRÁFICAS

1. Significado de Smart Tv, (s/f.). Significados. Recuperado de: https://www.significados.com/smart-tv/
2. Moskovclak M. y Katzmaler D, (2014), Transmisores Digitales, Recuperado de: https://www.cnet.com/es/noticias/chromecast-vs-apple-tv-vs-roku-3-cual-elegir/
3. Saraguro A., (2014). Desarrollo de Aplicaciones en TV inteligentes (Smart TV) basado en Tecnologías web (Tesis de maestría). Universidad de Granada. Granada. Recuperado de: http://repositorio.educacionsuperior.gob.ec/bitstream/28000/1728/1/T-SENESCYT-00924.pdf
4. Agencia de Regulación y Control. (2015). Telefonía fija, Audio – Video por suscripción y Radio-TV. Recuperado de: http://www.arcotel.gob.ec/wp-content/uploads/downloads/2015/04/BOLETIN-No-4-AVS_RTV_TF.pdf
5. Instituto Nacional de Estadística y Censos. (2013). Tecnologías de la Información (TIC'S) 2013. Recuperado de: http://www.ecuadorencifras.gob.ec/documentos/web-inec/Estadisticas_Socialess/TIC/Resultados_principales_140515.Tic.pdf
6. Secretaría Nacional de la Administración Pública, Gobierno Nacional de la República del Ecuador. (s/f). Plan Nacional de Gobierno Electrónico 2014-2017. Recuperado de: https://www.gobiernoelectronico.gob.ec/wp-content/uploads/downloads/2016/03/PlanGobiernoElectronicoV1.pdf
7. López V., Jaquero F., Montero J., Molina P., González P., (2006). Interfaces de Usuario Inteligentes: Pasado, Presente y Futuro. Recuperado de: http://www.dsi.uclm.es/personal/VictorManuelLopez/mipagina/archivos/Interaccion2006.pdf
8. Maybury, M. Intelligent user interfaces: an introduction. In Proceedings of the 4th international Conference on intelligent User interfaces (Los

Angeles, California, United States, January 05 - 08, 1999). IUI '99. ACM Press, New York, NY, 3-4, 1999.

9. Alheraish A., Design and implementation of home automation system, Consumer Electronics, IEEE Transactions on 50 (2004).
10. Al-Ali A., Al-Rousan M., Java-based home automation system, Consumer Electronics, IEEE Transactions on50 (2004).
11. DomoticHome, Domotichome (2009). Recuperado de: http://www.domotichome.net
12. Juste M., La historia del televisor: 90 años de continua transformación. Recuperado de: http://www.expansion.com/economia-digital/innovacion/2017/02/18/58a59f91e2704e9a308b4588.html
13. Sabraoui A., M. El Koutbi, I. Khriss, GUI Code Generation for Android Applications using a MDA Approach. 1th Edition of the International Conference on Complex Systems (ICCS'12) (Agadir, Moroco, November 05 – 06, 2012).
14. Pons C., Giandini R., Perez G., (1º Edición). (2010). Desarrollo de Software Dirigido por Modelos. Conceptos teóricos y su aplicación práctica. Buenos Aires, Argentina: Editorial de la Universidad Nacional de la Plata.
15. Estructura de Android. (2012). La Biblia del Programador. Recuperado de: http://labibliadelprogramador.blogspot.com/2012/09/estructura-de-android.html
16. Software de Comunicaciones, Arquitectura Android. (s/f). Universidad Carlos III de Madrid. Recuperado de: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android
17. SQL o lenguaje de consultas estructuradas. (2015). SearchDataCenter en Español. Recuperado de: http://searchdatacenter.techtarget.com/es/definicion/SQL-o-lenguaje-de-consultas-estructuradas
18. ¿Qué son las bases de datos?. (2007). Maestros del Web. Recuperado de: http://www.maestrosdelweb.com/que-son-las-bases-de-datos/

19. Introducción al Servidor Web IIS. (s/f). Microsoft Developer Network. Recuperado de: https://msdn.microsoft.com/es-es/library/hh831725(v=ws.11).aspx
20. Información general sobre el ciclo de vida de una página ASP.NET. (2007). Recuperado de: https://msdn.microsoft.com/es-es/library/ms178472.aspx
21. García P., Gonzales J., Castillo P., Arenas M. (s/f). Docencia de Arquitectura Orientada a Servicios. UPCommons. Recuperado de: http://upcommons.upc.edu/bitstream/handle/2099/12004/a43.pdf?sequence=1
22. Wang W., De S., Zhou Y., Huang X., Moessner K., (2017). Distributed Sensor Data Computing in Smart City Applications. IEEE. (s/v). (1-5). DOI: 10.1109/WoWMoM.2017.7974338
23. Pribyl O., Lom M., y Pribyl P., (2017). Smart Charles Square: Modeling interconnections of basic building blocks in Smart Cities. IEEE. (s/v). (1-6). DOI: 10.1109/SCSP.2017.7973351
24. Bangui H., Buhnova B., Rakrak S., y Raghay S. (2017). Smart Mobile Technologies for the City of the Future. IEEE. (s/v). (1-6). DOI: 10.1109/SCSP.2017.7973851
25. Nisar K., Ibrahim A., Wu L., Adamov A. y Deen M. (2016). Smart Home for Elderly Living Using Wireless Sensor Networks and an Android Application. IEEE. (s/v). (1-8). DOI: 10.1109/ICAICT.2016.7991655
26. Krsriyanto M., Dwi B., (2016). SMART HOME USING LOCAL AREA NETWORK (LAN) BASED ARDUINO MEGA 2560. IEEE. (s/v). (1-5). DOI: 10.1109/ICWT.2016.7870866
27. Anuebunwa U., Rajamani H., Pillai P., Okpako O. (2017). Evaluation of User Participation Capabilities in Demand Response Programs for Smart Home Applications. IEEE. (s/v). (1-6). DOI: 10.1109/PowerAfrica.2017.7991273
28. Sun J., Li Y., Wang L., Li X, Ma X., Xu J, Chen G., (2015). Controlling Smart TVs using Touch Gestures on Mobile Devices. IEEE. (s/v). (1-8). DOI: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.222

29. Kim J., Jung E, Lee Y., y Ryu W. (2015). Home Appliance Control Framework Based on Smart TV Set-top Box. IEEE. (v.61). (1-7). DOI: 10.1109/TCE.2015.7298086
30. Perakakis E.,Ghinea G., (2015). Responsive web design for the internet connected TV: The answer to more smart TV content?. IEEE. (s/v). (1-5). DOI: 10.1109/ICCE-Berlin.2015.7391286
31. Jia X., Zhao S., (2015). Intelligent Interface of Multiple Mode Internet of Things Aware Data. IEEE. (01). (1-4). DOI: 10.1109/ICCSNT.2015.7490846
32. Hedan L., Xinyu W., Chengen W. (2010). Intelligent interface for engineering design. IEEE. (01). (1-5). DOI: 10.1109/ICETC.2010.5529206
33. Smart Cities. (2016). Endesa Educa. Recuperado de : http://www.endesaeduca.com/Endesa_educa/recursos-interactivos/smart-city/
34. Criado J. (2015). Un Metodología Basada en Modelos y Mediación para la Adaptación de Interfaces de Usuario Dinámicas (Tesis Doctoral). Universidad de Almería, Almería.
35. Smart Lighting. Smart Cities. (2016). Recuperado de: http://smart-lighting.es/smart-city-ecuador-2017/
36. Economipedia. (s/f). ¿Qué es una ciudad inteligente o smart city?. Recuperado de: http://economipedia.com/definiciones/que-es-una-ciudad-inteligente-o-smart-city.html
37. Gadgets & Electronics, Security. (2016). IoT Smart Home Development Boost As Midea Joins With Huawei. Recuparado de: https://www.chinatechnews.com/2016/07/21/23790-iot-smart-home-development-boost-as-midea-joins-with-huawei
38. Arduino. (s/f). ¿Qué es arduino?. Recuperado de: http://arduino.cl/que-es-arduino/
39. Pandorafm. (2015). NoSQL vs SQL: Principales diferencias y cuándo elegir cada una de ellas. Recuperado de: https://blog.pandorafms.org/es/nosql-vs-sql-diferencias-y-cuando-elegir-cada-una/

40. Vera M. (2014). ¿Qué se entiende por SOA, y cuáles son sus beneficios?. Tech Deployment. Recuperado de: http://www.i2btech.com/blog-i2b/tech-deployment/que-se-entiende-por-soa-y-cuales-son-sus-beneficios/
41. Mayta C. (2012), Debate entre Servicios Web Basados en Rest Y Soap. Recuperado de: http://carlosmayta.blogspot.com/
42. Microsoft Developer Network, (s/f). Introducción al servidor web (IIS). Recuperado de: https://msdn.microsoft.com/es-es/library/hh831725(v=ws.11).aspx