DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

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

TEMA: DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA PARA LA INTEROPERABILIDAD DE APLICACIONES, BASADA EN SERVICIOS PARA EL ESTÁNDAR ISDBT-TB

AUTOR: TAPIA CABEZAS CRISTIAN PABLO

DIRECTOR: ING. ALULEMA FLORES DARWIN OMAR MSc.

SANGOLQUÍ

2017

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

Ing. Darwin Alulema, MSc.

CERTIFICA

Que el trabajo titulado "DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA PARA LA INTEROPERABILIDAD DE APLICACIONES, BASADA EN SERVICIOS PARA EL ESTÁNDAR ISDBT-TB", realizado por Cristian Pablo Tapia Cabezas, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la Universidad de las Fuerzas Armadas - ESPE en su reglamento.

Sangolquí, 14 de agosto de 2017

Ing. Darwin Alulema, Msc.

DIRECTOR

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

CRISTIAN PABLO TAPIA CABEZAS

DECLARO QUE:

El proyecto de grado denominado "DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA PARA LA INTEROPERABILIDAD DE APLICACIONES, BASADA EN SERVICIOS PARA EL ESTÁNDAR ISDBT-TB", ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas se presentan en las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 14 de Agosto de 2017

Cristian Pablo Tapia Cabezas

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

CRISTIAN PABLO TAPIA CABEZAS

Autorizo a la Universidad de las Fuerzas Armadas - ESPE la publicación, en la biblioteca virtual de la Institución del trabajo "DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA PARA LA INTEROPERABILIDAD DE APLICACIONES, BASADA EN SERVICIOS PARA EL ESTÁNDAR ISDBT-TB", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, 14 de Agosto de 2017

Cristian Pablo Tapia Cabezas

Dedicatoria

A Dios por la vida y las abundantes bendiciones recibidas.

A mi padre, mi madre y mi hermana por su apoyo incondicional a lo largo de toda mi carrera.

A mi abuelita que desde el cielo ha estado cuidándome e intercediendo por mi éxito.

Cristian

Agradecimientos

Gracias a Dios por la inteligencia, la oportunidad de estudiar y los regalos que día a día nos brinda.

Gracias a mi madre que desde mis primeros días ha estado presente con su inmenso cariño, siempre apoyándome en todo cuanto necesitaba y brindándome la fuerza necesaria para vencer las adversidades.

Gracias a mi padre que con su consejo ha sabido encaminar mis decisiones sabiamente, gracias por su ejemplo que ha sido en todo momento un referente y una guía para alcanzar mis metas.

Gracias a mi hermana que siempre me ha brindado su apoyo y confianza lo cual fue motor principal para continuar adelante sin vacilar.

A mis profesores que supieron impartir en mi la semilla del conocimiento, de manera especial al ingeniero Darwin Alulema quien con su sincera amistad ha estado siempre pendiente de la culminación exitosa de mi carrera, agradezco al doctor Gonzalo Olmedo por su guía técnica en la realización práctica de la tesis, también al ingeniero Derlin Morocho por su participación en la labor de lector.

A todos quienes hicieron parte de este camino, a mis compañeros que compartieron tanto tiempo entre conferencias, proyectos y desveladas. A cada uno mi más sincero agradecimiento.

Cristian

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo describe el diseño e implementación de una arquitectura para el desarrollo de aplicaciones interoperables basadas en el estándar de televisión digital vigente en Ecuador IDFBT-Tb. En la implementación se desarrolló una aplicación T-health para televisión digital interoperable con otra desarrollada para la plataforma móvil Android. Se realizaron pruebas en laboratorio utilizando un servidor de Playout y el software Village Flow. La arquitectura detalla el proceso de creación de los modelos CIM, PIM, PDM Y PSM y puede ser utilizada para el posterior desarrollo de aplicaciones similares en estructura pero diversas en los campos de aplicación. La comunicación entre las plataformas se realiza mediante la utilización de la arquitectura REST para servicios WEB. La aplicación desarrollada para la prueba de la arquitectura está orientada a la educación en salud para el público televidente, de esta forma el usuario en casa puede interactuar con el programa y conocer consejos de médicos que aporten a mejorar la calidad de vida. La aplicación también podría ser usada como material didáctico que permita a los galenos en formación aumentar su experiencia en el diagnóstico de enfermedades y el tratamiento con los pacientes.

PALABRAS CLAVE:

- **MDA**
- **REST**
- **ARQUITECTURA DE SOFTWARE**
- **ANDROID**
- **TELEVISIÓN DIGITIAL**

ABSTRACT

The present work describes the design and implementation of an architecture for the development of interoperable applications based on the IDFBT-Tb standard of digital television in Ecuador. In the implementation was developed a T-health application for digital television interoperable with another developed for the Android mobile platform. Laboratory tests were performed using a Playout server and Village Flow software. The architecture details the process of creating the CIM, PIM, PDM and PSM models and can be used for the subsequent development of similar applications in structure but diverse in the Fields of application. Communication between platforms is done by using the REST architecture for WEB services. The application developed for the architecture test is oriented to health education for the viewer public, so the home user can interact with the program and get advice from physicians who contribute to improve the quality of life. The application could also be used as training material to enable trained physicians to increase their expertise in disease diagnosis and treatment with patients.

KEYWORDS:

- **MDA**
- **REST**
- **SOFTWARE ARCHITECTURE**
- **ANDROID**
- **DIGITIAL TELEVISION**

CAPÍTULO 1

INTRODUCCIÓN

1.1 Antecedentes

El 26 de marzo de 2010, Ecuador adoptó oficialmente el estándar japonés-brasileño (ISDB-Tb), abriendo las posibilidades para la creación de aplicaciones que permitan explotar este recurso de diversas maneras. Uno de los puntos presentes en el Plan Nacional del Buen Vivir propuesto por el gobierno es democratizar la prestación de servicios públicos de telecomunicaciones y de tecnologías de información y comunicación (TIC), incluyendo radiodifusión, televisión y espectro radioeléctrico, y profundizar su uso y acceso universal.

La Televisión digital Terrestre "TDT", es el resultado de la aplicación de la tecnología digital a la señal de televisión analógica, proceso que permitirá optimizar el espectro radioeléctrico e implementar nuevos servicios audiovisuales e interactivos con una programación diversa a través de este medio de comunicación. Es una oportunidad que posibilitará el desarrollo de múltiples programas y aplicaciones como Telegobierno,

Telesalud y Teleducación, para el buen vivir de la población. (Bustamante Ramírez & Alvarez Mozoncillo, 2002)

"La gran innovación de la televisión digital consiste en introducir el concepto de interactividad en un medio pasivo. Esta novedad tendrá un efecto directo en el aspecto económico y en el de la producción, que por la interactividad llevará a un aumento de la demanda en el contexto de la convergencia entre los tres grandes sectores: televisión, telecomunicaciones e informática.

La llegada de la digitalización, al igual que todas las innovaciones tecnológicas, no es un fenómeno repentino, sino un fenómeno en continuidad con las grandes tendencias que recorren el sector desde hace años de forma paralela y solapándose: multiplicación y especialización creciente de la oferta; segmentación paralela de los consumidores; avance de la lógica de pago por el consumidor; concentración creciente y globalización de los productos, las programaciones y los capitales" (Bustamante Ramírez & Alvarez Mozoncillo, 2002). Así mismo el desarrollo de aplicaciones interactivas para Televisión Digital aún no ha sido explotada debido a que el apagón analógico esta por darse**.**

1.2 Justificación

La integración de la televisión digital con tecnologías desplegadas como las móviles expande los medios para la utilización de nuevos programa o aplicaciones con diversos contenidos, las cuales deben estar de acuerdo a los conceptos de convergencia tecnológica en el cual las nociones de ingeniería de modelos permite desarrollar sistemas que se puedan integrar con mayor facilidad a otras plataformas ya que se maneja estándares como es el caso de SOA (Arquitectura orientada a servicios).

De ahí que establecer un mecanismo que permita el desarrollo de aplicaciones de distinta naturaleza permitirá que el mercado para TV digital crezca consiguiendo así la masificación del servicio, la mejora de las aplicaciones y el nacimiento de nuevos modelos de negocio.

Una alternativa para alcanzar la interoperabilidad de aplicaciones de TV digital con plataformas como la móvil, es el empleo de una arquitectura orientada a modelos, encauzada hacia la representación de sistemas en componentes funcionales o lógicos, definiendo también sus interfaces de comunicación, cuya implementación emplee servicios. Por tanto, surge la necesidad de trabajar con arquitecturas que permitan definir de forma abstracta la comunicación de distintos tipos de tecnología, como por ejemplo la móvil.

"La Arquitectura basada por modelos o MDA (Model-Driven Architecture) lleva a resolver problemas de tiempo, costos y calidad asociados al desarrollo del software" (Loor, 2014). La ventaja de utilizar esta metodología es que una vez creado el modelo independiente o PIM (Platform-Independent Model), este se lo puede traducir a través de distintos compiladores a los diversos lenguajes de programación existentes. Las características de MDA son portabilidad, la interoperabilidad y la reutilización, estas prestaciones permiten reducir el costo, mitigar riesgos e incrementar la agilidad en la ejecución de un proyecto debido a su facilidad de implementación.

1.3 Alcance

Se plantea una arquitectura que se base en modelos para desarrollar aplicaciones de TV digital con el fin de generar un modelo abstracto que permita la evolución del mismo, además de admitir la convergencia con diversas tecnologías, para validar la arquitectura propuesta se desarrolla una aplicación de TV digital interoperable con la plataforma móvil Android.

En la interactividad de la televisión digital cabe recalcar dos modos de realizarla: la interactividad local que permite funcionar sin un canal de retorno mediante contenido almacenado en el receptor que puede ser renovada a medida que el usuario solicite información. La evidente limitación de este tipo de interactividad es que el usuario es

incapaz de enviar datos lo que restringe el tipo de aplicaciones que puedan ser desarrolladas mediante este concepto.

La otra opción para el desarrollo de aplicaciones interactivas en TV digital es utilizar un canal de retorno mediante el cual el usuario puede enviar información y por tanto este puede participar de concursos votando o enviar solicitudes y mensajes a través de la aplicación interactiva. Para el desarrollo del proyecto se utiliza una metodología mixta que incluye tanto aspectos cualitativos como cuantitativos.

Dentro de la metodología cualitativa se lleva a cabo una investigación bibliográfica del estado del arte sobre el desarrollo de proyectos similares en el Ecuador y la región, además se investiga sobre la implementación de la televisión digital en Ecuador y las políticas del gobierno en la implementación de la misma. Y finalmente se realiza también el estado del arte de las Arquitecturas de Software basadas en modelos y basadas en componentes y servicios.

En la metodología cuantitativa se desarrolla el diseño de la arquitectura basada en modelos con sus distintos niveles de abstracción, posteriormente se establece escenarios de prueba con una clasificación a alto nivel de los requerimientos a probar, además se desarrolla la prueba de concepto que comprende el desarrollo de la aplicación para su ulterior implementación. Posteriormente se realiza el análisis del escenario de prueba mediante un estudio de los resultados obtenidos, todo el procedimiento queda registrado en una memoria técnica la cual evidenciará los resultados de la investigación.

Los recursos de hardware a utilizar son un ordenador core i5 con 4Gb de memoria RAM para el desarrollo de las aplicaciones y las pruebas de escenario posteriores, un Smartphone con sistema operativo 4.0 o superior y un televisión con capacidad de recepción digital. En cuanto al software a utilizar, se necesita el programa GINGA NCL (Middleware Abierto del Sistema Nipo-Brasileño de TV Digital ISDB-TB) para desarrollar la aplicación además de su ambiente de ejecución GINGA-J que permite su

vinculación con Java, se utilizará el programa appinventor para el desarrollo de la aplicación móvil.

1.4 Estado del arte

En (Canchi, González, & Mendel, 2013), se presenta una arquitectura para el estándar SBTVD-T, encaminada a explotar las características interactivas de la Televisión Digital dentro del ámbito de la publicidad y el comercio electrónico haciendo uso del canal de retorno. "Se considera los objetivos de negocio de cada actor interesado (publicistas, radiodifusores e ingenieros de infraestructura tecnológica) y se formula metas tecnológicas alineadas a los mismos, los objetivos de negocio en los cuales se basa la propuesta son: 1) maximizar la audiencia, (2) proveer maneras novedosas de difundir el mensaje publicitario y (3) ampliar la variedad de servicios publicitarios a ofrecer." (Canchi, González, & Mendel, 2013).

Se hace uso de un Servidor de aplicaciones, el cual mediante un repositorio de datos que almacena toda la información concerniente a la publicidad, genera las aplicaciones pertinentes de acuerdo a los contenidos solicitados. También existe un servicio de publicidad encargado de gestionar y responde las solicitudes de los usuarios las cuales son enviadas desde su televisor haciendo uso del canal de retorno. La arquitectura propuesta está compuesta básicamente por tres subsistemas:

- "El Subsistema de Transmisión de Aplicaciones de Publicidad administra la transmisión de aplicaciones de publicidad desde el Transmisor al Receptor a través del Canal de transmisión.
- En el Receptor se ejecuta la Aplicación de Publicidad, cuya función principal es mostrar la información de publicidad y atender la interacción con el televidente.
- El Subsistema de Servicios Publicitarios Adicionales se encarga de atender las peticiones que lleguen desde la Aplicación de Publicidad Interactiva a través del Canal de retorno o de interactividad y enviar las respuestas por la misma vía. A su vez, la Aplicación

de Publicidad en el Receptor es la responsable de mostrar dichas respuestas al televidente.” (Canchi, González, & Mendel, 2013).

Toda la arquitectura es evaluada de acuerdo a los requerimientos de los interesados (publicistas, radiodifusores e ingenieros de infraestructura tecnológica) y se comprueba el cumplimiento de los mismos. Por último se presenta un análisis de escenarios basada en “la técnica clásica desarrollada en el Instituto de Ingeniería de Software de la Universidad Carnegie Mellon” que propone evaluar las respuestas de la arquitectura frente a diversos estímulos externos que puedan impactar en ella.

El trabajo de (Canchi, González, & Mendel, 2013) sirve como una guía para la construcción de la arquitectura de la propuesta a desarrollarse pues tiene algunos puntos en común como es el uso del canal de retorno y el uso de subsistemas que a su vez albergan servicios que son consumidos por las distintas entidades de la arquitectura.

Una de las principales diferenciaciones con el trabajo expuesto en esta memoria técnica es la incorporación de un usuario multiplataforma que obedece a la tendencia tecnológica de convergencia e internet de las cosas. Para la implementación del mismo se hace uso del Representational State Transfe o REST como estilo de arquitectura el cual será abordado más adelante.

En (Paucar Curasma, Ugarte Vera, & Paucar Curasma, 2014), se desarrolla una aplicación interactiva en el campo de T-Health que permite al usuario agendar una cita médica por medio de su televisor. El aporte de social de la aplicación radica en que “el registro de cita médica por televisión permite la inclusión social de las personas de menor ingreso económico y con domicilio alejado de los hospitales. Los pobladores, al solicitar citas por televisión, no generan gastos de transporte al desplazarse hasta los hospitales, que normalmente están alejados de sus domicilios.”

También se presenta un modelamiento en lenguaje UML (Unified Modeling Language, Lenguaje Unificado de Modelado) que detalla diagramas de caso de uso, diagramas de secuencia y modelado de datos de la aplicación.

"Para el desarrollo de este aplicativo interactivo, se utilizó el lenguaje NCL. NCL es una extensión de XML (eXtensible Markup Language, Lenguaje de Etiquetado Extensible)" (Paucar Curasma, Ugarte Vera, & Paucar Curasma, 2014) y su complemento Lua que permite la gestión de los datos enviados y el control del canal de retorno. Las pruebas para el análisis de resultados se las realizaron en laboratorio donde se utilizó el Virutal Set Top Box Ginga-Ncl instalado en una laptop y un servidor en el cual se alojaba la base de datos en SQL. No se especifica la arquitectura de servicios web utilizada.

El trabajo presente difiere de la propuesta de (Paucar Curasma, Ugarte Vera, & Paucar Curasma, 2014) en la arquitectura utilizada para brindar los servicios web al usuario, se trabaja como se mencionó con REST lo que permite que los servicios no sean únicamente consumidos por el usuario en Tv digital sino también por usuarios multiplataforma, sin embargo la aplicación desarrollada por (Paucar Curasma, Ugarte Vera, & Paucar Curasma, 2014) sirve como referente para el desarrollo de la arquitectura pues coincide en muchos puntos en la labor a realizar. Otro importante aporte a la arquitectura es que se implementará un subsistema de seguridad que permite un campo más amplio de aplicación en instancias donde se requiera seguridad en los datos a transmitir.

En (Martínez Martínez & Narváez Morales, 2013) despliegan una implementación de un prototipo de plataforma de software para TDT con Ginga-Ncl-Lua para desarrollar un mecanismo de Teleformación sobre dispositivos de servicio móvil avanzado, este trabajo es una primera aproximación de la interoperabilidad de la Televisión Digital con plataformas móviles, sin embargo se trata únicamente de una aplicación de interactividad local que no hace uso del canal de retorno, esta puede visualizarse en la plataforma móvil al encontrarse en formato "one-seg", un servicio del estándar IDB-Tb que transmite el

contenido haciendo uso de un ancho de banda estrecho lo cual permite la visualización en dispositivos portátiles.

"Para el diseño del software que ayudó a estructurar la plataforma de teleformación se usó Ginga-NCL-LUA que provee una infraestructura de presentación para aplicaciones interactivas de tipo declarativas escritas en el lenguaje NCL y LUA. NCL es una aplicación de XML con facilidades para los aspectos de interactividad, sincronismo, espacio - temporal entre objetos de adaptabilidad, soporte a múltiples dispositivos y soporte a la producción de programas interactivos en vivo no-lineares." (Martínez Martínez & Narváez Morales, 2013)

El diseño de la aplicación presenta 5 niveles a través de los cuales el usuario puede acceder al aula virtual para visualizar contenido relacionado con las materias académicas que está cursando. Las pruebas se realizaron en las instalaciones de la Universidad de las Fuerzas Armadas Espe en el laboratorio de Tv digital donde se usó el equipo necesario para la transmisión de la aplicación.

"Se usó un servidor que tiene instalado el software VILLAGEFLOW y que también tiene integrado la tarjeta moduladora DTA-115. Este servidor funciona como un transmisor de televisión digital ISDB-Tb. Del lado del receptor se utilizó Set Top Box como decodificador de las señales enviadas y un televisor para visualizar el contenido de audio, video y datos." (Martínez Martínez & Narváez Morales, 2013)

La interacción lograda por la aplicación es de tipo local pues no se utiliza el canal de retorno para el envío de datos del usuario, lo que quiere decir que toda la información es descargada en el receptor del usuario lo cual expone una limitación, al contrario lo que se busca con la arquitectura propuesta en este trabajo, es optimizar recursos y utilizar todo el potencial de las aplicaciones interactivas al hacer uso del canal de retorno.

Un punto en el que convergen ambos trabajos es que la aplicación puede ser utilizada no únicamente en el televisor sino también en dispositivos móviles como es el

caso de smartphones , sin embargo en la arquitectura propuesta se sortea la limitante de que el teléfono deba tener un receptor de televisión digital, al requerir únicamente de acceso a internet.

En (Peñerory Beltran, 2015) se muestra un modelo para el desarrollo de aplicaciones de televisión Digital Interactiva basado en el diseño centrado en el Usuario, **"s**e presenta un modelo que facilita el desarrollo de aplicaciones para Televisión Digital Interactiva y que permite el uso adecuado de técnicas de evaluación, de usabilidad y accesibilidad, desde la perspectiva del diseño centrado en el usuario." (Peñerory Beltran, 2015). La propuesta pretende solventar las falencias que presentan las aplicaciones de Tv digital que no involucran en el proceso de diseño al usuario, sus requerimientos y necesidades. La construcción del prototipo presenta 5 fases resumidas a continuación:

Primera fase: identificación de componentes y definición de la experiencia de usuario: Como acercamiento al público objetivo se elaboró una encuesta para conocer a fondo sus intereses televisivos y sus experiencias con diversas plataformas. Con el análisis de los datos, se creó una ficha de caracterización del usuario que presenta los distintivos de cada tipo de usuario que se detalló en la encuesta.

Segunda fase: diseño y arquitectura de la plataforma, el contenido multimedia y la interacción: Se define la estructura de la interfaz del usuario tomando en cuenta aspectos de usabilidad y accesibilidad establecidos en la fase anterior, así mismo se especifica la organización del contenido a desplegar.

Tercera fase: implementación de la aplicación de televisión digital interactiva e integración de componentes: "Luego de haber diseñado los diferentes elementos multimedia como iconos, botones, fondos y textos, se procedió a la implementación de la interfaz de usuario. A través del uso del lenguaje de programación Livecode." (Peñerory Beltran, 2015)

Cuarta fase: evaluación funcional, de usabilidad y accesibilidad: La estimación de la aplicación se dividió en tres actividades: Evaluación de la experiencia de usuario, evaluación de la interfaz gráfica, evaluación de la funcionalidad de la aplicación.

Quinta fase: entrega y feedback de la aplicación: Se creó los manuales de usuario y se receptó las experiencias de los usuarios con la aplicación.

Para el desarrollo de la aplicación se presenta un modelado bastante detallado de la aplicación lo cual constituyó una guía para el presente trabajo, pues ambos poseen los mismos elementos y la forma de interactuar entre estos es también similar, cabe recalcar que a diferencia de la arquitectura a desarrollarse que es para el estándar ISDBT-Tb, el modelo de (Peñerory Beltran, 2015), se realizó para el estándar Europeo DVB vigente en Colombia.

Como resultado de la investigación previa a la elaboración de este trabajo de titulación no se ha encontrado artículos que ahonden en el tema de interoperabilidad de la televisión digital con plataformas móviles, por tanto es un tema innovador que plantea varios trabajos futuros que profundicen en esta línea de investigación.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1 MBD Desarrollo de software basado en modelos

El concepto principal que propone la ingeniería de software es que la creación del mismo debe realizarse como toda obra de ingeniería compleja que corresponde a otras disciplinas, bajo esta premisa es necesario un proceso bien estructurado que permita la utilización de herramientas como el caso de modelos.

"Hacia finales de los años 70, Tom Demarco en su libro "Structured Analysis and System Specification" introdujo el concepto de desarrollo de software basado en modelos o MBD (por sus siglas en inglés Model Based Development). DeMarco destacó que la construcción de un sistema de software debe ser precedida por la construcción de un modelo." (Pons, Giandini, & Pérez, 2010). Un modelo es una representación de un

sistema donde se puede apreciar sus características y funcionalidad, así mismo sirve para desarrollar o comprobar hipótesis con el fin de refinar el sistema y depurarlo de errores.

Utilizar arquitecturas basadas por modelos, requiere también de una documentación clara de todas las etapas que se van desarrollando, de esta forma estas quedan listas para una posterior modificación que permita al software adaptarse a nuevas plataformas. La descripción de los modelos a utilizarse puede basarse en 2 métodos:

Métodos Matemáticos.- "Existen técnicas y herramientas formales que ayudan a modelar el problema y razonar sobre la solución de una manera precisa y rigurosa. Estos métodos utilizan lenguajes de especificación de naturaleza matemática, tales como: Z [DB 01], VDM [EF 94], B [BM 00] y OCL [OCL], los cuales permiten demostrar si la especificación cumple ciertas propiedades." (Pons, Giandini, & Pérez, 2010)

Métodos diagramáticos.- Los métodos diagramáticos son más utilizados debido a su interpretación intuitiva, a pesar de mantener un lenguaje formal al igual que los métodos matemáticos, los diagramáticos son de fácil comprensión, uno de los más utilizados es el UML (Lenguaje Unificado de Modelo).

La abstracción es uno de los principales elementos que ayuda en la creación de un modelo, requiere una idealización del problema, aislando los elementos menos importantes y centrándose en la esencia del mismo lo cual permite la comprensión y manejabilidad de un sistema complejo.

Cuando se empieza con las etapas iniciales de la arquitectura, es necesario crear los siguientes modelos que ayudarán al desarrollador a definir los alcances y limitaciones del proyecto:

- Requisitos (escritos generalmente en lenguaje natural).
- Modelos de Análisis
- Modelos de Diseño

A continuación se muestra el proceso de desarrollo de software basado en el paradigma basado en modelo:

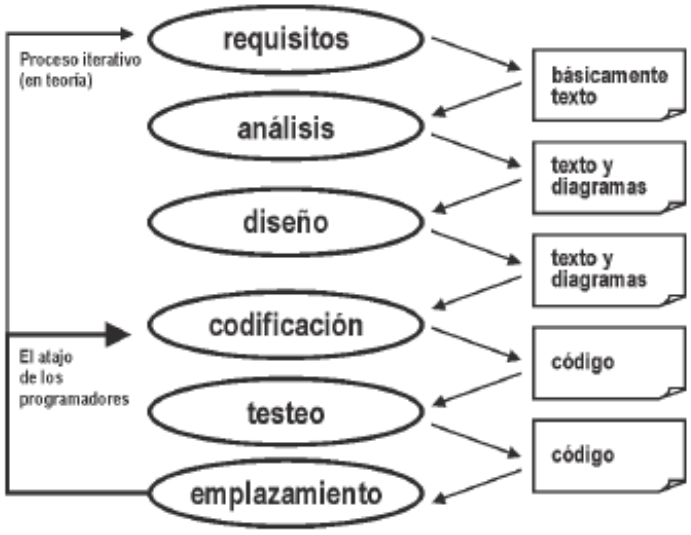

Figura 1. Ciclo de vida del desarrollo de software basada en modelos
Fuente: (Pons, Giandini, & Pérez, 2010)

2.2 MDD desarrollo de software dirigido por modelos

Es necesario diferenciar el desarrollo de software "basado" en modelos al "dirigido" por modelos, este adjetivo empleado en esta herramienta, denota una mayor importancia de los modelos en el desarrollo llegando a ser al menos tan importante como el mismo código fuente. Se trabaja creando modelos abstractos hasta llegar a modelos concretos y finalmente al código fuente empleando una transformación del cual se hace cargo el motor de MDD.

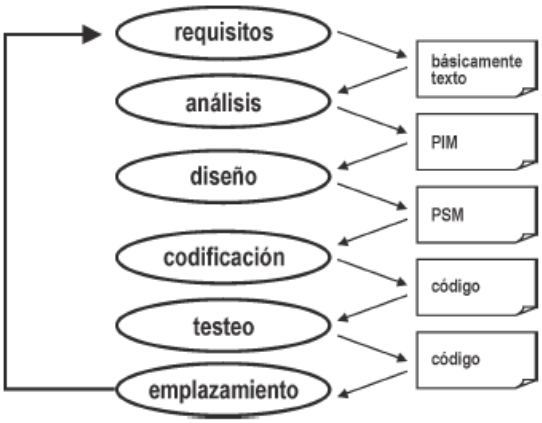

Figura 2. Ciclo de vida del desarrollo de software dirigido por modelos
Fuente: (Pons, Giandini, & Pérez, 2010)

"En el proceso de desarrollo de software tradicional, las transformaciones de modelo a modelo, o de modelo a código son hechas mayormente con intervención humana. Muchas herramientas pueden generar código a partir de modelos, pero generalmente no van más allá de la generación de algún esqueleto de código que luego se debe completar manualmente." (Pons, Giandini, & Pérez, 2010). Sin embargo la innovación que propone MDD es la automatización de todos los modelos, es decir que existe una herramienta capaz de transformar todos los modelos a partir del principal.

No obstante el desarrollo de software tradicional implica siempre intervención humana en la etapa de la transformación de modelos, pues si bien es cierto existen programas que generan el código a partir del modelado, este no es más que una base que debe ser pulida y detallada para satisfacer las necesidades finales de los clientes.

El origen de MDD no es reciente pues se basa en ideas antes propuestas para el desarrollo de software, se puede citar el ejemplo del libro escrito por W. Dijkstra "A Discipline of Programming" donde ya se menciona la idea de la abstracción para crear el modelo de especificaciones que luego pueda ser transformado en el código fuente.

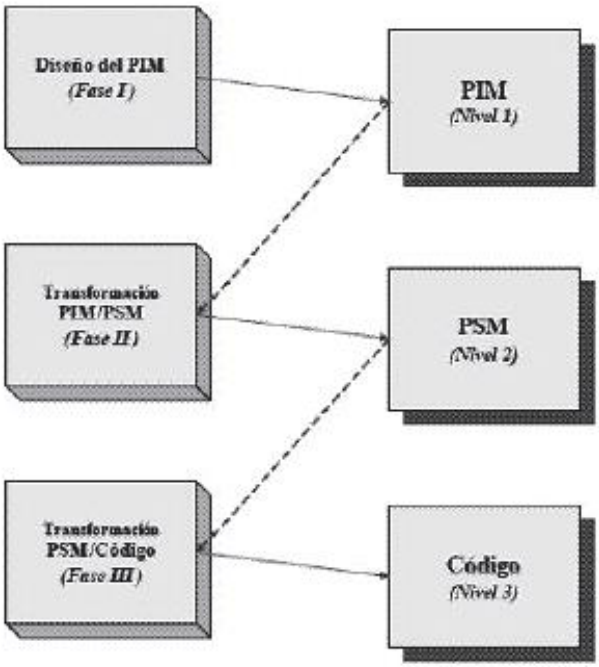

Figura 3. Los tres pasos principales en el proceso del desarrollo MDD

Fuente: (Pons, Giandini, & Pérez, 2010)

Dentro de los beneficios de utilizar MDD se puede citar (Pons, Giandini, & Pérez, 2010):

- Incremento de la productividad
- Adaptación a los cambios tecnológicos
- Adaptación a los cambios en los requisitos
- Consistencia
- Re-uso
- Mejoras en la comunicación con el usuario
- Mejoras en la comunicación entre desarrolladores
- Captura de experiencia
- Los modelos son productos de larga duración

- Probabilidad de demorar las decisiones tecnológicas

Para el desarrollo de software dirigido por modelos existen dos propuestas que son las más conocidas y utilizadas:

- MDA.- Arquitectura Dirigida por Modelos que utiliza un lenguaje basados en estándares OMG (Object Managment Group)
- DSM.- Domain Specific Modeling, el cual utiliza un lenguaje específico de dominio DSLs (Domain Specific Language)

A continuación se detallará la propuesta MDA que es la más utilizada para la arquitectura desarrollada en los capítulos posteriores.

2.3 MDA Model Driven Architecture

2.3.1. Introducción

Dentro de MDA se pueden distinguir 3 etapas que comprenden del proceso de desarrollo de software:

CIM (Computation Independent Model).- es el modelo independiente de la computación y especifica lo que se espera del sistema de manera práctica comprende la parte de estudios básicos.

PIM (Platform-Independent Model).- "es una descripción de la funcionalidad del sistema en forma independiente de las características de plataformas de implantación específicas, pues crea los procesos y las estructuras del sistema, sin pensar en el sistema operativo, lenguajes, el hardware o las redes a utilizar" (wordpress, 2014).

PSM (Platform-Specific Model).- en este modelo se crea la descripción específica de sistema tomando en cuenta la plataforma en la cual se va a implementar y los lenguajes de programación a utilizar. Para la construcción de PSMs se usan los perfiles UML que

son extensiones de UML que permiten añadir información semántica a los modelos para expresar detalles específicos de la plataforma.

Cumpliendo con las directivas del OMG, las dos principales motivaciones de MDA son la interoperabilidad (independencia de los fabricantes a través de estandarizaciones) y la portabilidad (independencia de la plataforma) de los sistemas de software.

Evidentemente la ventaja más significativa del uso de MDA es que su modelo PIM debe ser creado de forma tal que garantice los cambios tecnológicos y permita su posterior evolución según se requiera. Una vez creado el modelo PIM, este puede ser reutilizado en diversas aplicaciones con la posibilidad de ser migrado a diversas plataformas que utilicen distintos lenguajes de programación.

Cuando se habla de MDA es preciso definir un dominio específico de lenguaje o DSL (Domain Specific Languaje). Se define a un dominio como un área de interés para un esfuerzo de desarrollo en particular (Pons, Giandini, & Pérez, 2010). Generalmente se trabaja con dominios pequeños pues son más fáciles de definir.

Los DSL son lenguajes que sirven para crear modelos, sin embargo no están estandarizados, estos pueden ser gráficos o no y depende mucho del dominio de aplicación (Pons, Giandini, & Pérez, 2010).

2.3.2. Clasificación de los modelos

Varios modelados pueden describir un mismo modelo pero de manera más o menos detallada, dependiendo del nivel de abstracción y la etapa de desarrollo en que se encuentra. Así las primeras etapas podrían únicamente mostrar generalidades en los elementos de los diagramas por ejemplo solo nombres, y posteriormente en un modelado más avanzado se puede especificar los atributos de cada objeto.

Para el modelo CIM o independiente de la plataforma y la computación se puede describir incluso el modelo de negocio al cual se quiere aplicar el proyecto, si bien son dos conceptos distintos, el software sirve para asistir el modelo de negocio y tiene que tener en cuenta los requerimientos específicos que este necesita.

Los tipos de modelos más comúnmente utilizados son:

- Modelos estáticos (o estructurales): Estos hacen referencia a los elementos que constituyen un sistema, sus propiedades y sus conexiones entre sí.
- Modelos dinámicos (o de comportamiento): Especifican el comportamiento de los elementos y cómo interactúan con el sistema.
- Modelos independientes de la plataforma: Excluyen los detalles técnicos que supone la implementación de un sistema en un lenguaje de plataforma específica.
- Modelos específicos de la plataforma: Detallan la arquitectura incluyendo los conceptos técnicos de lenguaje específico de una plataforma en particular.

2.3.3. Metamodelado

"Hace algunos años, la sintaxis de los lenguajes se definía casi exclusivamente usando Backus Naur Form (BNF). Este formalismo es una meta sintaxis usada para expresar gramáticas libres de contexto, es decir, una manera formal de describir cuáles son las palabras básicas del lenguaje y cuáles secuencias de palabras forman una expresión correcta dentro del lenguaje. Una especificación en BNF es un sistema de reglas de la derivación, escrito como:

<símbolo> ::= <expresión con símbolos> "

Para estandarizar el uso de lenguajes gráficos se desarrolló una técnica denominada "metamodelo" el cual define qué elementos pueden existir en un modelo y como pueden conectarse entre sí.

Es posible usar también lenguajes gráficos los cuales poseen sus propias características que los diferencian de los lenguajes textuales, por ejemplo, la lectura de un lenguaje textual se la realiza desde arriba hacia abajo y de izquierda a derecha, sin embargo en un lenguaje gráfico esta idea no es siempre aplicable y su lectura depende de su sintaxis concreta.

2.3.4. Módulos o Subsistemas

Agrupan los distintos componentes que contribuyen a realizar una tarea dentro de la arquitectura. Pueden existir componentes de primer orden o segundo nivel de acuerdo a la granularidad y tamaño de la aplicación (M01). Sus características son:

- Deben tener un conjunto definido y claro de responsabilidades
- Un límite claramente definido
- Una o más interfaces definidas
- Protocolo de comunicación claro
- Reutilizable (dentro y fuera del sistema)
- Testeable o comprobable.

2.3.5. La arquitectura de 4 capas de modelado del OMG (Object Management Group)

"El metamodelado como se mencionó anteriormente, es entonces un mecanismo que permite definir formalmente lenguajes de modelado, como por ejemplo UML. La Arquitectura de cuatro capas de modelado es la propuesta del OMG orientada a estandarizar conceptos relacionados al modelado, desde los más abstractos a los más concretos." (Pons, Giandini, & Pérez, 2010)

Dentro de las especificaciones del OMG, se definen los modelos M3, M2, M1 y M2 de la siguiente manera:

Nivel M0: Instancias

En este nivel se definen todos los objetos de aplicación del sistema, es decir las entidades físicas que lo conforman y dentro de estas los elementos que pueden contener, en esta etapa no se mencionan las clases ni atributos.

Nivel M1: Modelo del sistema

Los elementos de las entidades definidos en la capa M0 se convierten en instancias de M1 y se le asignan ciertas condiciones, como por ejemplo la clase de variable con la que pueden ser representadas.

Nivel M2: Metamodelo

De la misma forma los elementos del nivel M1 son las instancias del nivel M2. En esta capa se introducen los conceptos de Clase, Atributo y Operación.

Nivel M3: Meta-metamodelo

Se continúa con el mismo procedimiento, los elementos del nivel M2 son las instancias del nivel M3, este es el nivel más abstracto, el lenguaje que generalmente se utiliza en esta capa es MOF.

El rol del metamodelado en MDD es sumamente importante pues a partir de este se definen los lenguajes del modelado para que programas de transformación puedan entenderlos y trabajar correctamente, de esta forma se pueden traducir a diferentes lenguajes de programación mediante reglas preestablecidas.

Dentro de OMG se detalla el uso de MOF (Meta-Object Facility) el cual es un lenguaje de modelado definido por OMG más abstracto, sus características son:

- A cada elemento de una capa anterior le corresponde un elemento de la capa superior.
- Se basa en el paradigma orientado a objetos.
- Usa la misma sintaxis de los diagramas UML
- Está separada por dos partes: EMOF (Essential Mof) y CMOF.

2.3.6. Software de Desarrollo

ECILPSE EMF

Eclipse es una herramienta de programación basada en código abierto, es multiplataforma y se puede extender su funcionalidad mediante la creación de plugins. Fue desarrollado en un inicio por IBM para el desarrollo en lenguaje Java, pero más luego fue adaptado para operar con distintos lenguajes de programación. El programa Eclipse tiene un plugin llamado Ecore (EMF) para la implementación de MOF.

"El proyecto EMF es un framework para modelado, que permite la generación automática de código para construir herramientas y otras aplicaciones a partir de modelos de datos estructurados" (Pons, Giandini, & Pérez, 2010).

XOMEGA

Si se quiere utilizar herramientas de Microsoft Office como Visual Studio, existe un plugin denominado Xomega que permite trabajar con ingeniería dirigida por modelos, para instalarlo hay que ir a la opción Tools y luego dar clic sobre Extensions and Updates, ahora en este menú se selecciona en la parte izquierda la opción online y en la barra superior derecha se digita el plugin que se requiere instalar. Una vez visualizado el resultado de la búsqueda, se procede a dar clic en descargar y posteriormente en instalar, por último se debe reiniciar Visual Studio para que se apliquen los cambios.

2.4 Arquitectura basada en componentes

"Una arquitectura basada en componentes describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas." (Pelaez, 2009).

La descomposición del sistema implica el detalle de los componentes individuales y como estos interaccionan entre sí. Algunos beneficios de utilizar una arquitectura basada en componentes son:

- Arquitectura reutilizable.
- Los componentes del diseño pueden ser extraídos para formar aplicaciones en distintos campos independientemente de la plataforma.
- La encapsulación de los componentes permite su utilización sin necesidad de conocer detalles técnicos internos de estos.
- Se reduce los costos puesto que se permite la distribución del costo de desarrollo y mantenimiento.

2.5 SOA Arquitectura orientada a servicios

"SOA (Arquitectura orientada a servicios) es un marco de trabajo conceptual que establece una estructura de diseño para la integración de aplicaciones" (Bussines, 2016). La implementación de SOA permite la reducción de costos, flexibilidad, adaptación y aumento en la competitividad.

La arquitectura basada en servicios permite la interoperabilidad entre las distintas plataformas independientemente de las características técnicas de estas. El intercambio de información se da cuando un cliente realiza una petición, el recurso al cual quiere acceder tiene un identificador denominado URL y la comunicación se ejecuta bajo el estándar SOAP.

En la arquitectura que se propone, se utilizará el diseño basado en componentes y servicios con el fin de brindar interoperabilidad de la TV digital con diversas plataformas a través del uso de webservice, que al basarse en HTTP puede ser reconocido por otras plataformas como la móvil.

2.4.1. SOAP Simple Object Access Protocol

"Un servicio web SOAP es una interfaz que describe un conjunto de operaciones a las cuales se tiene acceso mediante un mensaje XML" (Vallecillos, 2016). Una vez implementado el servicio, el desarrollador presenta una descripción sobre que protocolos de comunicación y transporte deben usarse para acceder al servicio. Por otra parte, la interfaz de usuario debe ofrecer una comunicación lo más transparente posible hacia el usuario ocultando los detalles técnicos de la implementación.

"La capa más baja de la arquitectura es la capa de red. Los servicios web deben estar en localizaciones que sean accesibles para poder ser invocados por los consumidores de servicios. Los servicios web públicos usan protocolos de red que son estándares, como por ejemplo, el protocolo HTTP, o los protocolos de Internet, como SMTP o FTP, entre otros. La siguiente capa es la capa de mensajería, basada en XML.

La capa de mensajería utilizada en esta pila conceptual es SOAP debido a que es un protocolo bastante simple basado básicamente en mensajes POST HTML que contiene XML. (Vallecillos, 2016).

La capa WSDL (Web Services Description Language) describe como debe ser consumido el servicio por el usuario, se define la interfaz y los mecanismos de interacción cliente-servidor. Dentro de estas descripciones está también la publicación y la localización de servicios en la red los cuales pueden ser definidos a través de la siguiente capa denominada UDDI (Universal Description, Discovery and Integration). "Por último, la composición de servicios y el flujo de información se describen a través de un documento WSFL (Web Services Flow Language)" (Vallecillos, 2016).

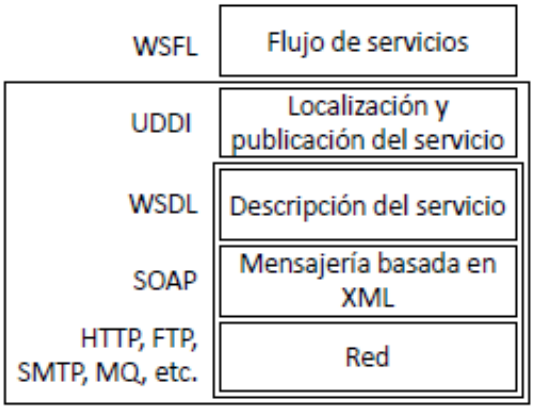

Figura 4. Pila conceptual de servicios de red

Fuente: (Vallecillos, 2016)

Hoy en día es comúnmente usada la tecnología de Servicios Web la cual se define como un objeto remoto con el cual puede interactuar un cliente para solicitar datos mediante métodos de entrada haciendo uso de una URL

Para la interacción cliente-servidor se maneja la invocación de métodos para lo cual se utiliza el estándar "SOAP que es un protocolo de la capa de aplicación basado en XML para el intercambio de mensajes en servicios web." (Vallecillos, 2016)

2.4.2. Representational State Transfer (REST)

Los servicios web construidos bajo la arquitectura REST no se basan en objetos SOA, su funcionamiento implica no guardar el estado del servicio tras la solicitud. "Los servicios web REST solo contienen información de estado en la invocación de sus mensajes y en sus respuestas. Los mensajes de entrada y salida ya contienen la información necesaria para gestionar los recursos." (Vallecillos, 2016).

"Los servicios RESTful se basan en tres características principales: direccionamiento, interfaces uniformes y la carencia de estados." (Richardson & Ruby, 2008). En el direccionamiento se utilizan un identificador de recurso denominado URI,

un recurso es todo aquello a lo que el servidor puede tener acceso como por ejemplo, una base de datos, un documento, un correo, etc.

Otra de las características de RESTful es la utilización de una interfaz uniforme y estándar lo cual permite la interoperabilidad de diversas plataformas sobre un mismo servicio web. Los formatos de comunicación soportados por esta arquitectura son varios incluyendo texto plano, HTML, XML, y JSON (JavaScript Object Notation). Para el uso de una interfaz uniforme en HTML se utilizan un conjunto de operaciones identificadas con los verbos POST, PUT, DELETE y GET.

Para el diseño de un servicio web basado en la arquitectura REST se siguen los siguientes pasos: "(a) primero, se nombra el recurso utilizando una URI; (b) a continuación, se identifica el tipo de interfaz específica que se ofrece a través del servicio; (c) seguidamente, se diseña cómo se representar el recurso que será solicitado luego por los clientes, y cómo se enviará dicho recurso al cliente; (d) y finalmente, se define cómo será el comportamiento del servicio y qué ocurre durante una ejecución satisfactoria del mismo." (Vallecillos, 2016)

REST es una arquitectura utilizada para brindar servicios Web. Una de las principales ventajas de su uso es que no está condicionado a retornar datos en formato XML únicamente como es el caso de SOAP, sino que puede hacer uso de otros lenguajes como JSON (JavaScript Object Notation), HTML (Hypertext Markup Languaje) o texto plano.

Para la comunicación entre distintas tecnologías en este trabajo se emplea Web Service en la arquitectura REST (Transferencia de Estado Representacional) que "es un método de comunicación entre dos aparatos electrónicos en una red que utiliza una colección de protocolos abiertos y estándares usados para intercambiar datos entre aplicaciones o sistemas. Las aplicaciones escritas en varios lenguajes de programación

que funcionan en plataformas diferentes pueden utilizar web services para intercambiar información a través de una red". (Pressman & Ojeda, 1998)

2.6 Ginga NCL

Ginga es el Middleware (lógica de intercambio de información entre aplicaciones) que permite la creación de aplicaciones interactivas que potencializan el uso de la televisión digital. Funciona para el estándar Japonés-Brasileño o ISDB-Tb. Fue desarrollado en la Pontificia Universidad Católica de Río de Janeiro PUC-Rio.

"GINGA presenta una infraestructura para desarrollar aplicaciones interactivas bajo el lenguaje declarativo NCL (Nested Context Languaje). Ginga-NCL separa el contenido de la estructura de presentación, en definitiva los lenguajes declarativos permiten al desarrollador proporcionar en conjunto las tareas a realizar sin definir una gran cantidad de código para hacer las mismas tareas, en comparación con las implementaciones algorítmicas, a fin de cuentas, los lenguajes declarativos centran su uso en la adaptabilidad, edición y producción de contenido y el soporte a múltiples dispositivos." (Alejandro, 2011). Dentro de GINGA están presentes tres módulos complementarios entre sí: Ginga-NCL, Ginga-J y Ginga-CC.

Ginga NCL procesa los programas desarrollados en el lenguaje declarativo NCL, una opción para ampliar las propiedades del paradigma imperativo es la utilización de scripts en lenguaje LUA que permiten incorporar funcionalidades como animaciones, utilización del canal de retorno y transición de contenido dinámico.

NCL facilita la organización de los objetos media y especifica su posición en el tiempo y espacio además de concatenarlos mediante eventos lo cual es de mucha utilidad para el desarrollador al momento de crear aplicaciones no lineales que responden de acuerdo a las necesidades del usuario.

Una de las ventajas que presenta la arquitectura de GINGA es su adaptabilidad, se puede añadir extensiones según necesidad del desarrollador para permitir por ejemplo la transmisión de video bajo demanda, otra de las extensiones más usadas es GINGA-J que añade la posibilidad de ejecutar aplicaciones implementadas en JAVA.

2.5.1. Tipos de interactividad en Ginga

Interactividad Local

Al desarrollar una aplicación para TV digital es posible contar con una interacción de usuario que no requiere del uso del canal de retorno, el televidente visualiza contenido descargado previamente en su receptor, la limitación de este tipo de interactividad es evidente, el usuario no pude enviar datos y la información que obtienen no pueden ser actualizada en tiempo real.

Interactividad con canal de retorno

La otra opción de interactividad requiere del uso del canal de retorno proporcionado por el estándar ISDBT-Tb el cual requiere que el receptor posea una conexión a internet. Las opciones para el desarrollador se amplían puesto que ahora el usuario puede enviar datos hacia un servidor externo el cual podría procesarlos para enviarlos de regreso. Otra de las ventajas de utilizar este tipo de interactividad es que la información que el usuario visualiza puede estar actualizada en tiempo real.

2.5.2. Estructura de un programa NCL

Un programa NCL tiene una estructura similar a un programa HTML y se pueden distinguir básicamente 2 partes:

Head: La cabecera está delimitada por las etiquetas < HEAD > y </HEAD >. Dentro de este campo se declaran las siguientes instancias:

<regionBase>: define las posiciones en la pantalla de los distintos elementos que conforman la aplicación, así como su tamaño.

<descriptorBase>: “Es la base que agrupa a los elementos <descriptor>, los cuales definen los valores iniciales para las propiedades de los elementos <media>. Estos valores iniciales son definidos como atributos de los elementos los cuales definen si el medio se ve con transparencia, por cierto tiempo, etc”. (Universidad Técnica Federico Santa María, 2014)

<connectorBase>: Es la declaración de la biblioteca que contiene “condiciones simples o compuestas así como también acciones simples o compuestas con el fin de asignarle roles a los elementos de media que se desee estén envueltos en relaciones causa-efecto. Presionar un botón para desplegar una imagen, o esperar 10 segundos para que se detenga un video, son ejemplos de relaciones causa-efecto para elementos de media.” (Universidad Técnica Federico Santa María, 2014).

Body: Está delimitado por las etiquetas <BODY> que indican el inicio del campo y < /BODY > que indican su finalización. Dentro de Body se declara las siguientes instancias:

<port>: Cuando algún elemento dentro de la aplicación no necesita de algún evento para iniciar, se debe asociarlo a una puerta en la cual se especifica también su región y su correspondiente descriptor.

<media>: Con esta etiqueta se declaran todos los objetos de texto, imagen u objetos lua que se utilicen en la aplicación, es necesario asignarles nombre e indicar su ubicación, es posible también agregarles propiedades las cuales son de utilidad para los objetos lua.

<link>: “Los elementos <link> son aquellos que permiten la interactividad y el que suceda, en general, el inicio y término de los componentes de media. Aquí se le asigna un rol a un <media> en la relación causa-efecto definida en los conectores en <head>.”

(Universidad Técnica Federico Santa María, 2014). Un ejemplo de uso de un conector o link es el siguiente código:

```
<region id= "region1" height= "37%" width= "44%" right= "45%" top="33%" zIndex= "1"/>
<descriptor id= "descriptor1" region= "region1" />
<media id= "imagen" src="imagen.jpg" descriptor= "descriptor1"/>
<link xconnector= "onKeySelectionStartNStopN">
<bind role= "onSelection" component= "imagen" >
<bindParam name= "keyCode" value="VK-ENTER"/>
</bind>
<bind role= "start" component= "imagen1" />
<bind role= "stop" component= "imagen2" />
</link>
```

La primera línea hace referencia a la región en la cual se desplegará “imagen” la cual será utilizada como botón posteriormente, luego se hace la declaración del descriptor que hará referencia la región antes creada.

Con el comando media id, se da nombre al objeto media imagen y se especifica su ubicación. Link xconeector = "onKeySelectionStartNStopN" define un evento de tecla el cual controlará el inicio y fin de objetos multimedia.

Al especificar “bindParam name= "keyCode" value="VK-ENTER” se está indicando que el evento será iniciado al pulsar la tecla “ENTER” del control remoto, el parámetro “keyCode” puede tener también los siguientes valores: "0" al "9", "A" hasta la "Z", "*", "#", "MENU", "INFO_, "GUIDE", "CURSOR_DOWN", "CURSOR_LEFT", "CURSOR_RIGHT", "CURSOR_ UP", "CHANNEL_DOWN", "CHANNEL_UP", "VOLUME_DOWN", "VOLUME_UP", "ENTER", "RED", "GREEN", "YELLOW",

"BLUE", "BACK", "EXIT", "POWER", "REWIND", "STOP", "EJECT", "PLAY", "RECORD" y "PAUSE". (Martínez Martínez & Narváez Morales, 2013). Por último se define que componente iniciará y cual se detendrá al ejecutarse el evento.

2.5.3. NCLUA

Dentro de un programa NCL es posible añadir objetos media del tipo .lua que pasan a llamarse objetos NCLua, lo cual amplia la opciones para realizar una interactividad con el televidente más específica. De esta forma es posible añadir por ejemplo campos de texto para la autentificación de usuarios o envío de información textual por medio del canal de retorno.

El control del objeto NCLUA es al igual que cualquier otro objeto añadido al programa y se ejecutará en la secuencia en la que el programador haya especificado en sus propiedades.

Una de las características más resaltables de utilizar LUA en el desarrollo de aplicaciones para TV digital es la opción de implementar un cliente Web que permita al televidente realizar peticiones hacia un servidor el cual responderá con el contenido solicitado. La notificación de los eventos externos recibidos por el programa .lua requiere una función que se estructura de la siguiente manera:

```
function handler(evt)
end
event.post(handler)
```

Existen 3 tipos de eventos que LUA puede gestionar:

- NCL { class = ‘ncl’, type = ‘presentation’, action = ‘start’, }
- Control remote { class = ‘key’, type = ‘press’, key = ‘RED’ }
- Eventos de uso del canal de retorno { class = ‘tcp’, type = ‘connect’, }

Donde:

- Class: Indica la clase de evento.
- Type: India el tipo de evento.
- Action: Define la acción a ser tomada
- Property: Propiedad afectada
- Key: Tecla presionana
- Label: Nombre del ancla asociada al evento

2.5.3.1. Clases en NCLUA

Sirve para la comunicación de LUA con NCL cuando se generan eventos del tipo:

<link xconnector=”OnBeginStart”>

<bind role=”OnBegin” component=”video”/>

<bind role=”start” component=”lua”/>

</link>

El evento lua que se genera es:

event.post{ class='ncl', type='presentation', action='start' }

De la misma forma se pueden crear eventos similares como por ejemplo uno que tome acción de detener un objeto media en lugar de inicializarlo como el ejemplo anterior.

event.post { class='ncl', type='presentation', action='stop' }

Existen 2 tipos de evento que se pueden generar con la clase NCL:

Tipo presentación:

- Class: ‘ncl’
- Type: ‘presentation’
- Area: [string] nombre del label asociado al evento.
- Action: [string] puede tomar los siguientes valores: 'start', 'stop', 'abort', 'pause' o 'resume'.

Tipo atributo:

- Class: ‘ncl’
- Type: ‘atribution’
- Property: [string] nombre de la propiedad asociada al evento.
- Action: [string] puede tomar los siguientes valores: 'start', 'stop', 'abort', 'pause' o 'resume'.
- Value: [string] nuevo valor a ser asignado a la propiedad.

Class KEY

Se utiliza para gestionar los eventos generados al pulsar alguna tecla del control remoto. Tiene los siguientes parámetros:

- Class: ‘key’
- Type: [string] puede asumir los valores de ‘press’ o ‘relase’.
- Key: [string] valor de la tecla presionada.

Class TCP

Con el uso de esta clase es posible conectarse a cualquier servidor a través del canal de retorno disponible en televisión digital.

Para enviar se utiliza la siguiente estructura:

event.post { class = 'tcp', type = 'connect', host = , port = , [timeout = ,] }

Para recibir se utiliza la siguiente estructura:

evt = { class = 'tcp', type = 'data', value = , connection = , error = , }

(Universidad Técnica Federico Santa María, 2014)

2.5.3.2. Módulos de NCLUA

Los 2 módulos básicos que maneja NCLua son el módulo Event y el módulo Canvas.

Módulo Event:

El módulo event sirve para gestionar la conexión entre el programa NCL y el programa LUA así como los datos que o parámetros seteados en la ejecución. La siguiente figura muestra la relación entre un evento de tipo ncl con el archivo NCL y con el script LUA, generando un puente entre ambos para el traspaso de información de ejecución y de parámetros:

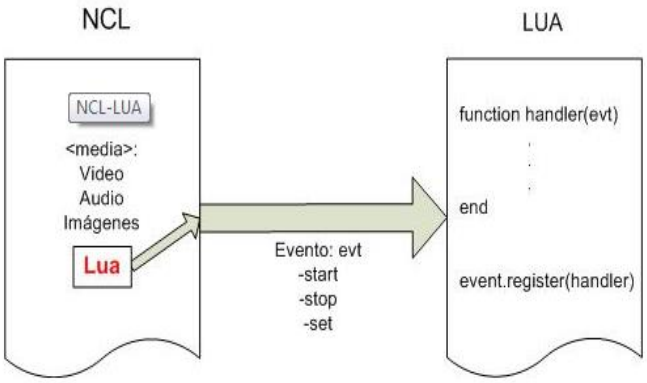

Figura 5. Relación NCL-LUA

Fuente: (Paucar Curasma, Ugarte Vera, & Paucar Curasma, 2014)

Módulo Canvas:

Con el uso de este módulo es posible dibujar elementos en la pantalla y es posible también mediante la programación en Lua especificar en qué momentos estos deben aparecer y con qué atributos específicos:

Para definir en qué posición de la pantalla se dibujará el elemento, es necesario primeramente definir la región que ocupará el objeto Lua dentro del programa NCL, luego haciendo uso del módulo Canvas se puede definir la posición del objeto mediante coordenadas de eje X e Y.

Estos son algunos de los métodos que permite el módulo:

- canvas:clear(): Borra todos los elementos dejando vacía la región
- canvas:attrColor(r,g,b,a): Define el color de un elemento en formato rgba
- canvas:attrFont(font,size): Cuando dibujamos texto haciendo uso del módulo, con el uso de este método podemos setear el tamaño y la fuente de la letra.
- canvas:drawText(x,y,text): Dibuja un texto definido en el parámetro 'text' en la posición 'x' e 'y'
- canvas:flush(): Actualiza o refresca la pantalla si ha existido cambios, siempre que se utiliza un método del tipo 'draw', es necesario utilizar flush.
- canvas:drawRect('fill', 0,0, dx,dy): Dibuja un rectángulo en la posición especificada luego del parámetro 'fill' con ancho igual a dx y altura igual a dy.

2.5.3.3. Uso del canal de retorno de Tv digital

Para hacer uso del canal de retorno es necesario el archivo tcp.lua el cual contiene el programa necesario para establecer la conexión con un servidor Web y gestionar peticiones y respuestas generadas por los mismos.

Los métodos más importantes de tcp.lua son:

- socket.tcp()
- master:close()
- client:close()
- server:close()
- master:connect(address, port)
- client:receive([pattern [, prefix]])
- client:send(data [, i [, j]])

2.5.4. Village Flow

El software Village Flow, integra las opciones de codificación y multiplexación de audio, video y datos. El programa fue desarrollado por una empresa japonesa e integra un sistema muy versátil que permite la integración de varios módulos de modulación RF y adaptadores de entrada y de salida. Dentro de las funcionalidades que permite Village Flow están: Transmisión de datos (GINGA), monitoreo de las señales, generación de guía de programación, demodulación RF, subtítulos, además es posible generar el video en todos los formatos soportados por el estándar ISDB-Tb, estos son: definición estándar, alta definición, y one-seg para plataformas móviles.

2.5.5. Decodificador EITV

El decodificador EITV es un Set Top Box para desarrolladores de aplicaciones interactivas completamente compatible con el estándar ISDB-Tb, tiene las siguientes principales características:

Tabla 1:

Características Decodificador EITV

Características
Completa interactividad (DTVi - Ginga)
Cargar las aplicaciones por vía aérea a través de Internet o red local
Soporte simultáneo de canales ISDB-T y IPTV (a través de UDP y RTP)
Aplicación gráfica (GUI) a través de un servidor de web para instalación y configuración de aplicaciones Ginga (DTVi) y canales de IPTV
Ajuste de la imagem en la pantalla de televisión
Control de bloqueo de los canales por grupo de edad con contraseña
Compatible con los estándares de color PAL-M y NTSC
Búsqueda automática de canales
Guía Electrónica de Programación (EPG)
Menú en inglés, portugués o español
Selección de audio

Fuente: (EITV, 2011)

2.5.6. Televisión digital en Ecuador

El 26 de marzo de 2010, luego de un análisis de los distintos estándares de Tv digital propuestos por los diversos países, el Consejo Nacional de Telecomunicaciones CONATEL, resolvió adoptar el estándar japonés-Brasileño ISDBT-tb siguiendo la tendencia de la región Latinoamericana cuyos únicos países que no han adoptado este estándar son Colombia, Guyana, Surinam y Guyana Francesa.

Algunas de las ventajas del estándar adoptado por Ecuador, además de ser el más reciente, son:

- La posibilidad de poder transmitir hasta tres canales en definición estándar en vez de uno solo en alta definición haciendo uso de un solo canal.
- Posee la opción de transmitir una guía de contenido
- Sus aplicaciones interactivas soportan el uso de canal de retorno para enviar datos de regreso hacia servidores externos o estaciones de transmisión.

- Es posible la recepción de la señal incluso con movilidad a velocidades de hasta 400 Km/h en el caso de alta definición y 100 Km/h en el caso de definición estándar. (Moreno Quinche & Salazar Baculimba, 2011)
- Proporciona el midlware Ginga con el que es posible la transmisión de aplicaciones interactivas a la par del contenido visual.
- Incorpora la tecnología "one-seg" que permite la transmisión de contenido dirigido a dispositivos móviles.

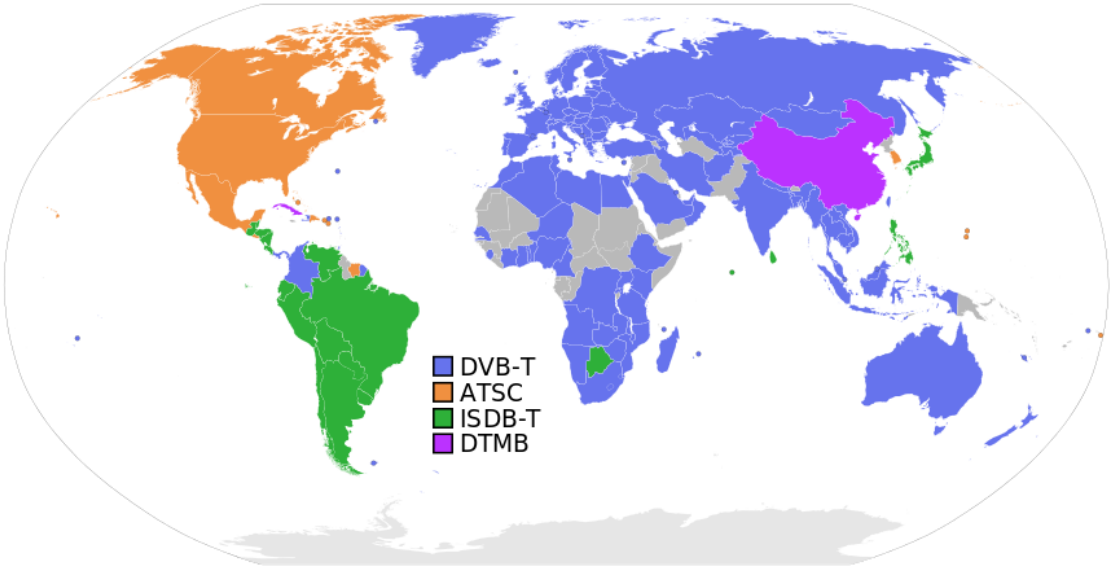

Figura 6. Estándares de Televisión Digital en el mundo

Fuente: (Dominio Público)

Uno de los datos importantes a analizar referente a la aceptación de contenido por Televisión en Ecuador, es la posesión de televisiones en los hogares, de acuerdo a los datos recolectados por el INEC (Instituto de Estadísticas y Censos del Ecuador) en una encuesta realizada en el año 2010, se obtuvo los siguientes datos:

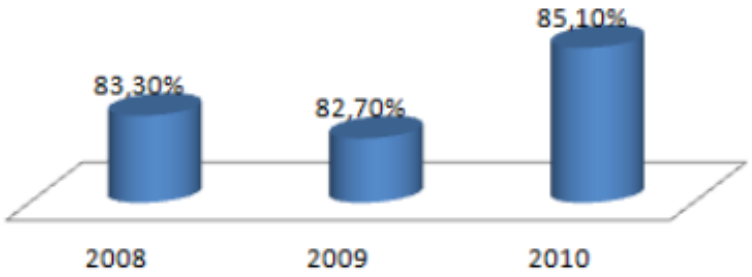

Figura 7. Posesión de televisores en hogares Ecuatorianos
Fuente: (Moreno Quinche & Salazar Baculimba, 2011)

El denominado "apagón analógico" que comprende la supresión de contenido para la televisión analógica, estaba previsto para finales del año 2016, sin embargo debido a una falta de difusión del tema entre los ecuatorianos, esta fecha fue pospuesta. Existen ya en el mercado diversos modelos de televisores compatibles con el estándar y muchos de ellos también ofrecen la característica que permite al usuario el uso del canal de retorno por medio de internet. Si el usuario no posee una televisión nueva que permita la recepción de la señal digital, se puede optar también por la adquisición de un decodificador, en muchos casos estos también poseen conexión a internet. A la fecha Ecuador cuenta ya con 23 estaciones de transmisión de televisión digital.

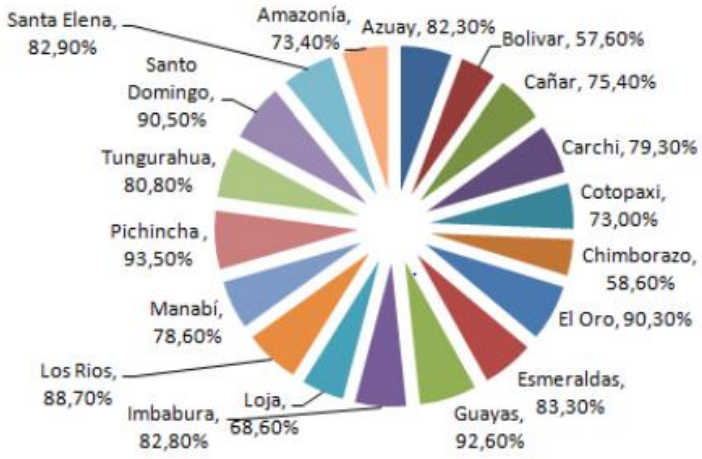

Figura 8. Posesión de televisores por provincia
Fuente: (Moreno Quinche & Salazar Baculimba, 2011)

2.5.7. Penetración del internet en Ecuador

En una encuesta realizada en diciembre de 2016 por el INEC en el cual se recolectó datos a nivel nacional, regional, urbana y rural en 31092 viviendas se establecieron los siguientes resultados.

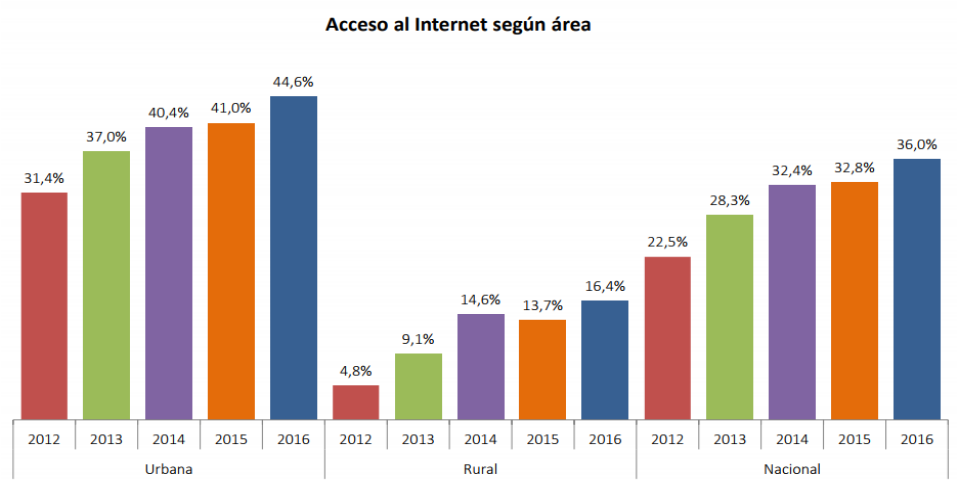

Figura 9. Acceso al Internet según área
(INEC, 2016)

"El 36,0% de los hogares a nivel nacional tienen acceso a internet, 13,5 puntos más que hace cinco años. En el área urbana el crecimiento es de 13,2 puntos, mientras que en la rural de 11,6 puntos." (INEC, Acceso al Internet, 2016)

"El 36,0% de los hogares tiene acceso a Internet, de ellos el 24,5% accede a través de algún medio inalámbrico, 4,0 puntos más que en 2012." (INEC, 2016)

"Según un estudio del Instituto de Estadísticas y Censos (INEC), en el 2006 solo el 2.5% de los ecuatorianos tenía acceso a internet. En la actualidad la cifra sobrepasa el 1000 %, es decir que el 26.1% usa internet." Según las proyecciones se estima que a este ritmo, dentro de ocho años, el 50% de la población contara con internet. (Suarez, 2015)

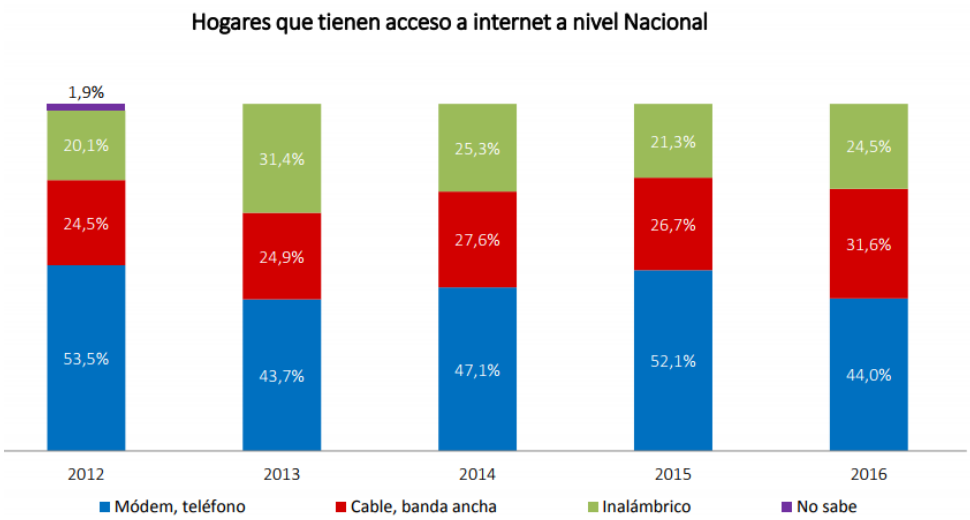

Figura 10. Hogares que tienen acceso a internet

Fuente: (INEC, 2016)

"En cuanto al uso de dispositivos móviles como Smartphone o teléfono inteligente, en los últimos años ha sido exponencial. En el 2015, 3'084.886 ecuatorianos declararon tener un teléfono inteligente (Smartphone). Esta cifra es casi cinco veces más que la reportada en el 2011 cuando 522.640 ecuatorianos tenían celulares." (Metro, 2016)

2.5.8. Aplicaciones interactivas para Tv Digital

Aplicaciones T-Commerce o aplicaciones de comercio electrónico:

Consiste en aplicaciones que aprovechan la publicidad en la televisión, permitiendo que el usuario final realice compras, o interactúe con propagandas de un producto específico. Esta actividad tiene fines de lucro, abriendo un nuevo concepto en el uso del televisor. Dentro de esta aplicación, se incorpora además los servicios bancarios, permitiendo al usuario realizar pagos, o transferencias con su código de cuenta secreto. (Martínez Martínez & Narváez Morales, 2013)

Aplicaciones de T-Health o de salud:

Las aplicaciones T-Health tienen por propósito fomentar estilos de vida saludable además de colaborar en tareas del campo médico, mediante una aplicación T-health un usuario puede acceder a los servicios de salud tales como unidades terapéuticas o de diagnóstico. Es posible por ejemplo que el usuario envíe sus signos vitales por medio del uso de sensor en su teléfono o consultar el resultado de exámenes médicos de importancia.

El uso de aplicaciones T-Health supone el uso de una base de datos en donde el usuario guarda su historial médico en el cual se encuentran todos sus datos, diagnósticos y exámenes realizados, lo cual es una ayuda para los profesionales de la salud pues se cuenta con un formato estandarizado y de fácil acceso a los informes médicos de cada paciente.

Aplicaciones T-Goverment o de administración gubernamental

Las aplicaciones de T-goverment son todas aquellas que facilitan a la población a acceder a los servicios públicos como por ejemplo información, solicitudes, certificados de registro, pago de servicios, etc. Con el uso de este tipo de aplicaciones, se simplifica la administración de servicios y se propaga más eficientemente información de interés ciudadano.

Aplicaciones T-Learning o de aprendizaje

Las aplicaciones de T-Learnig son aquellas que apoyan al aprendizaje del alumno mediante el uso de las tecnologías de la información, un ejemplo típico de una aplicación de este tipo es un aula virtual en donde el usuario puede acceder a contenido multimedia que facilita la comprensión de las materias que está cursando.

CAPÍTULO 3

DISEÑO DE LA ARQUITECTURA E IMPLEMENTACIÓN

3.1 Desarrollo de los modelos

La siguiente es una vista general donde se visualizan los principales elementos que conforman la arquitectura.

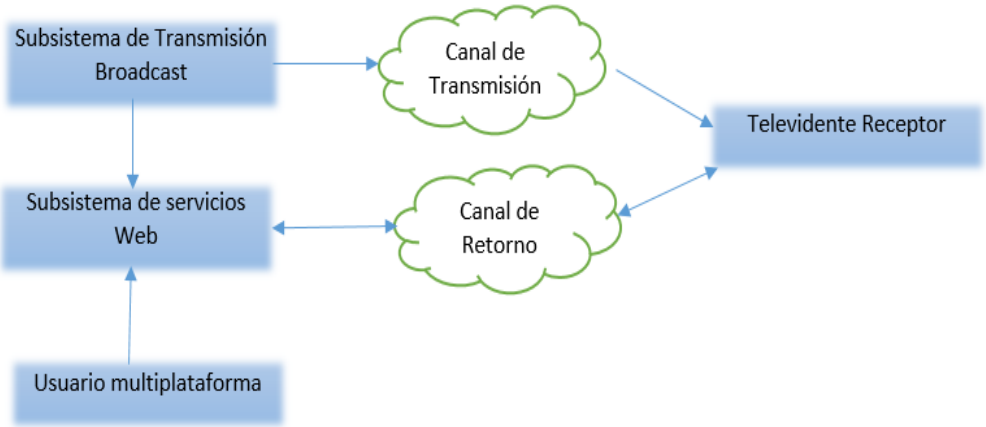

Figura 11. Vista general de la arquitectura

El subsistema de transmisión Broadcast lo compone la estación de televisión que es la encargada de generar el contenido visual y la aplicación interactiva, el canal de transmisión es el aire por el cual viajan las ondas electromagnéticas en el rango UHF, una vez recibido el contenido, haciendo uso de su receptor el televidente pude interactuar con la aplicación, en caso de que desee ingresar a la plataforma, el usuario pude enviar sus datos a través del canal de retorno (internet). Todas las solicitudes son atendidas por el subsistema de servicios web al cual el consumidor también puede acceder desde otra plataforma distinta a la televisión digital.

3.1.1. Modelo CIM

El modelo denominado CIM (Modelo independiente de la computación) es la representación más general que contiene todos los subsistemas que integran la arquitectura e indica cómo se relacionan de manera concisa.

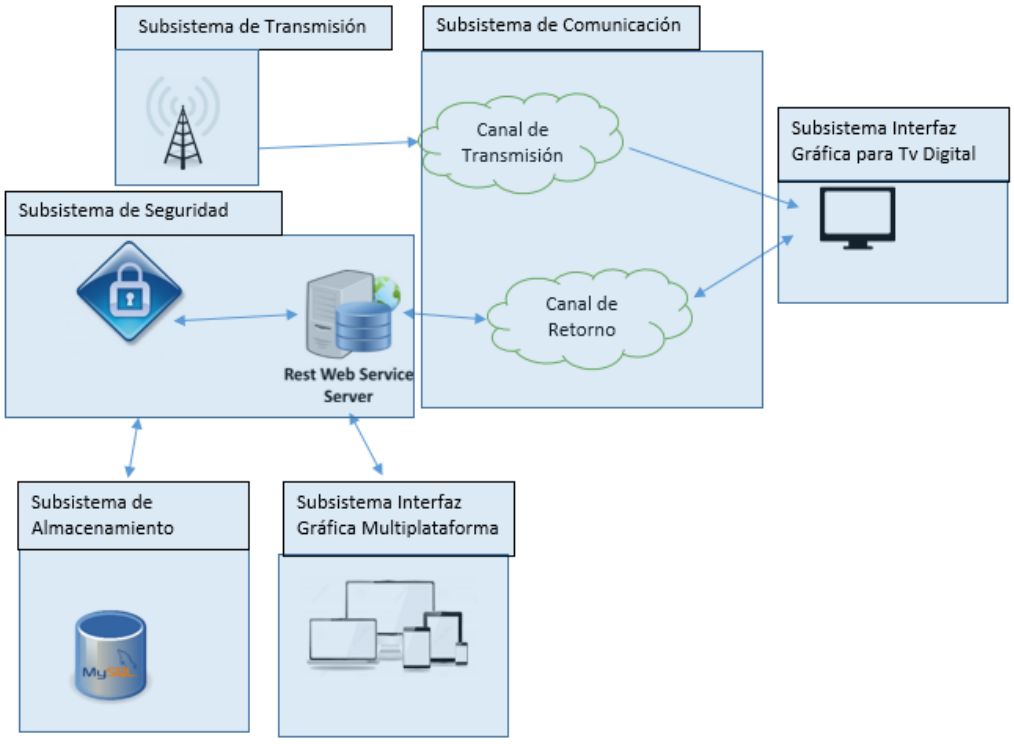

Figura 12. Modelo CIM

En contraste a la vista general de la arquitectura, aquí se define explícitamente cuales son los subestimas y cómo interactúan unos con otros. Tanto el usuario multiplataforma con el usuario de televisión digital interactúan con la plataforma haciendo uso de Rest Web Service el cual a su vez interacciona con un subsistema de seguridad que garantiza la confidencialidad de los datos de usuario guardados en el subsistema de almacenamiento

3.1.2. Modelo PIM

En el modelado PIM o modelo independiente de la plataforma es común utilizar el lenguaje UML. En esta etapa del diseño, se establece la interacción entre los distintos subsistemas que conforman la arquitectura.

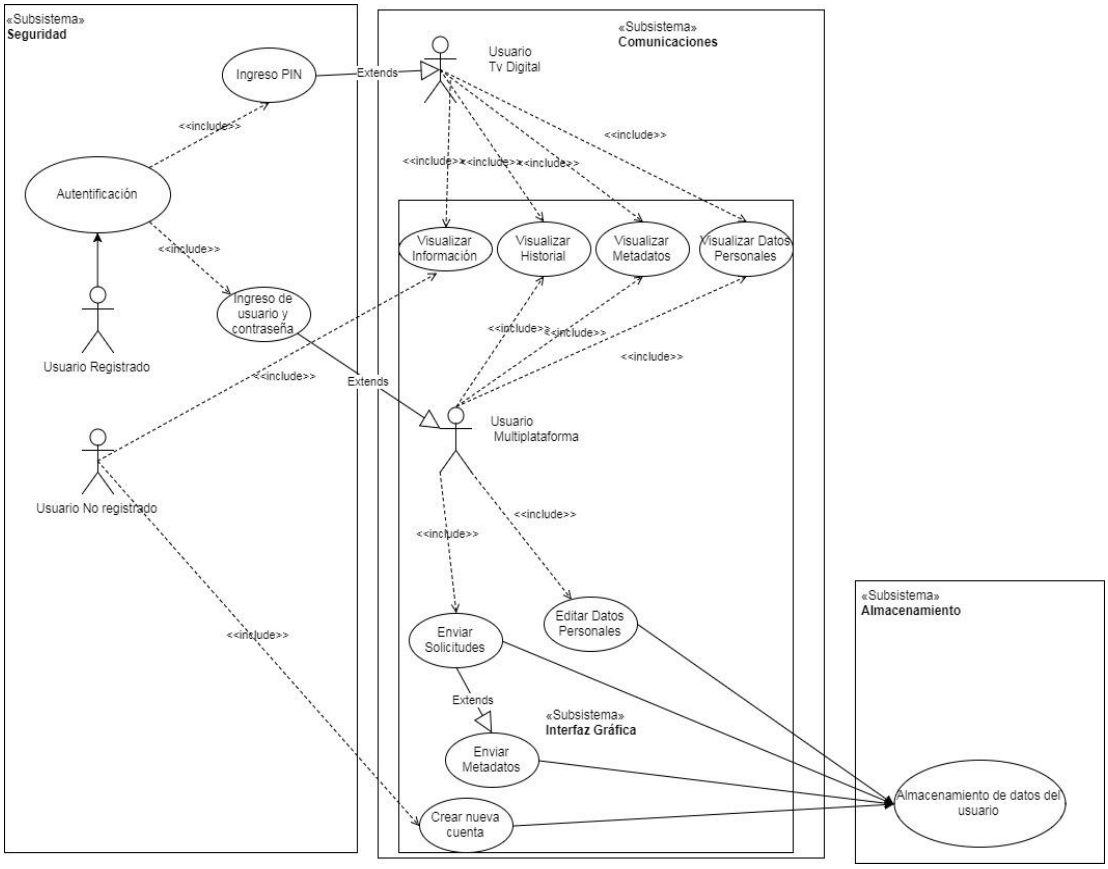

Figura 13. Modelo PIM

En la primera etapa de la arquitectura el usuario interactúa con la aplicación haciendo uso del subsistema de interfaz gráfica para autentificarse con sus datos mediante dos métodos dependiendo de qué plataforma está utilizando, en caso de ingresar desde la televisión digital, el usuario deberá ingresar su PIN, en caso de ingresar desde la plataforma móvil, deberá ingresar su usuario y contraseña, en este punto está en funcionamiento el subsistema de seguridad que valida los datos enviados para comprobar si el usuario es correcto.

Una vez validado el usuario, entra en marcha el subsistema de comunicaciones que contempla los dos posibles escenarios globales, en caso de que el usuario use la televisión digital para interactuar con la aplicación, este tiene la posibilidad de visualizar su historial, sus datos de sensores y sus datos personales. En caso de que el usuario esté utilizando la plataforma móvil, este tendrá las opciones de editar datos personales, enviar solicitudes, enviar metadatos (datos de sensor), adicionalmente también podrá acceder a las opciones de visualización que permite la televisión digital, toda esta interacción del usuario con la plataforma es posible a través del al subsistema de interfaz gráfica.

Todos los datos enviados al crear un usuario, enviar una solicitud o metadatos, son gestionados y guardados en el subsistema de almacenamiento.

3.1.3. Modelo PDM

En esta etapa de modelo definición de la plataforma, se especifican los atributos que tendrán cada uno de las entidades que conforman la arquitectura. También se detallan cada uno de los subsistemas y las variables que utiliza cada uno, mediante el uso de diagramas UML se puede detallar también los alcances y limitaciones de la arquitectura defiendo la cardinalidad de cada uno de sus elementos. A continuación se exponen todos los subsistemas que conforman la arquitectura propuesta:

Subsistema de Almacenamiento

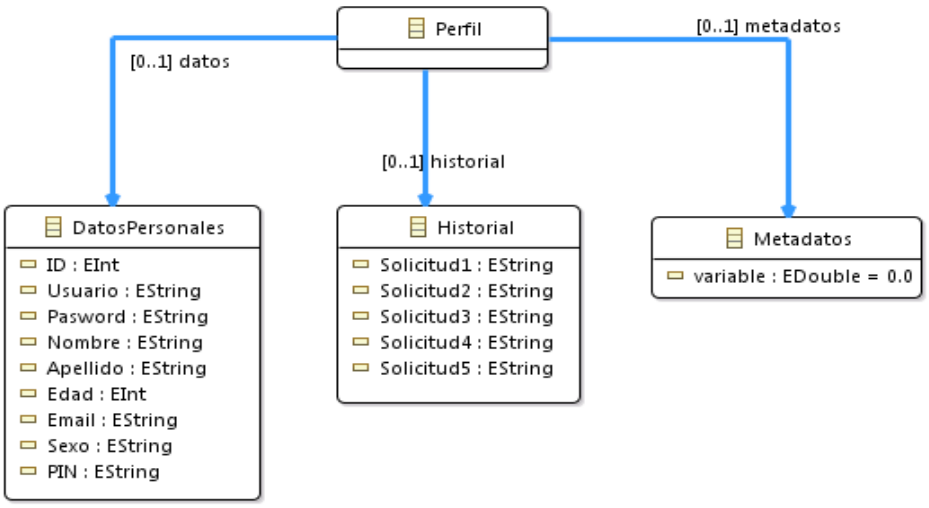

Figura 14. Subsistema de Almacenamiento

Cada usuario posee un perfil que almacena sus datos personales, su historial de solicitudes y sus metadatos, es posible almacenar hasta 5 solicitudes. El subsistema de persistencia o almacenamiento comprende la base de datos Sql y los métodos necesarios para que el Web Service pueda extraer datos ordenados de esta, más adelante en el modelado PSM se detalla el código utilizado para la creación del subsistema.

Subsistema de Seguridad

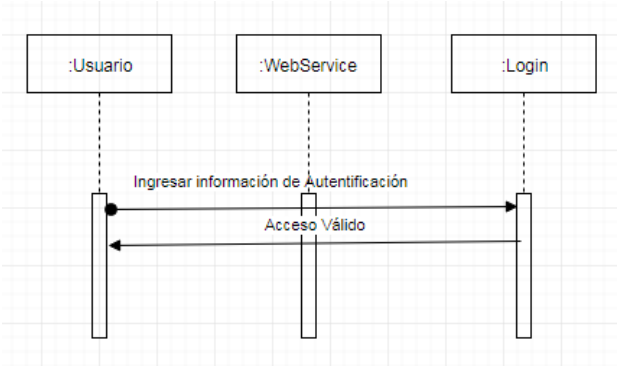

Figura 15. Subsistema de Seguridad

En el subsistema de seguridad intervienen los siguientes actores: el usuario que envía la información de autentificación a través del Web Service y el método Login alojado en el servidor el cual es el encargado de gestionar las peticiones y responder a estas indicando si el usuario es válido o no.

Subsistema de Comunicaciones

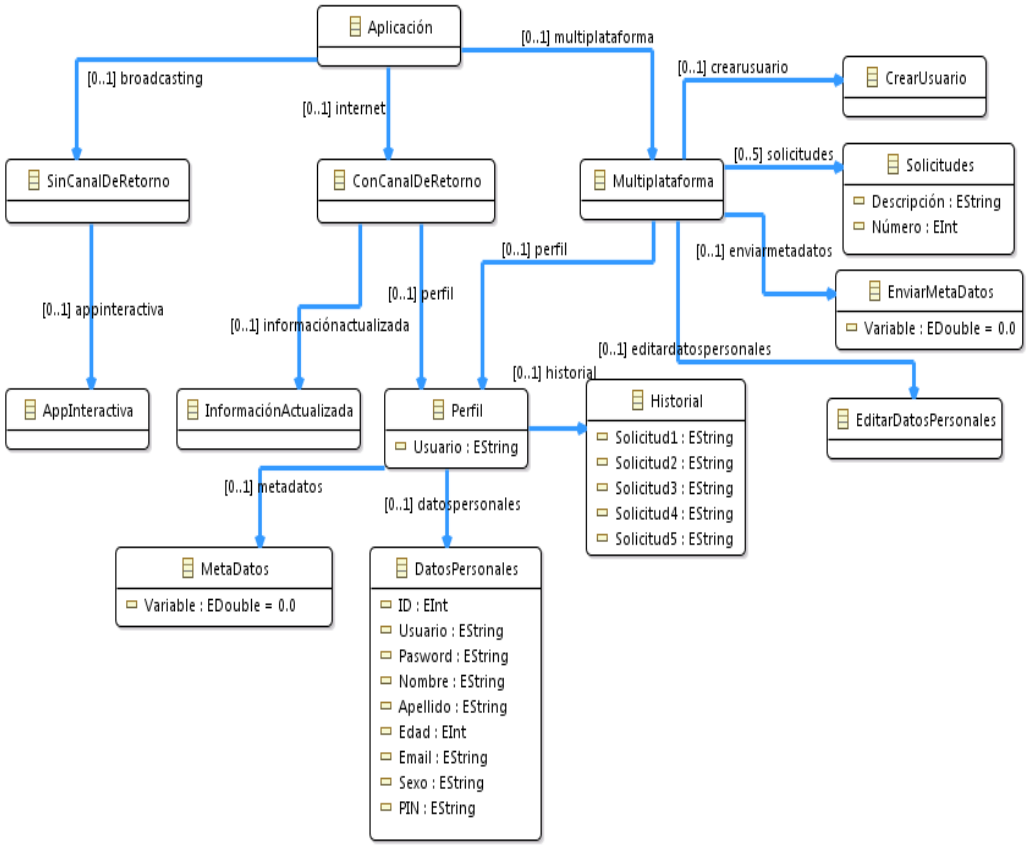

Figura 16. Subsistema de Comunicaciones

En el subsistema de comunicaciones se consideran los 3 posibles escenarios:

Usuario de Tv Digital sin canal de retorno: Puede acceder a la aplicación transmitida vía Broadcast e interactuar de manera mínima visualizando información acerca del programa.

Usuario de Tv Digital con canal de retorno: Tiene la posibilidad de ingresar a la plataforma enviando su PIN, una vez validado podrá acceder a la visualización de sus datos personales, su historial de solicitudes y sus metadatos. Si el televidente no desea ingresar, puede hacer uso del canal de retorno al solicitar información dinámica de la aplicación la cual es actualizada frecuentemente dependiendo del tema que se esté tratando en el programa.

Usuario Multiplataforma: Ingresando desde su móvil, el usuario puede editar sus datos personales, enviar solicitudes o metadatos y visualizar su historial.

Modelo de Negocio

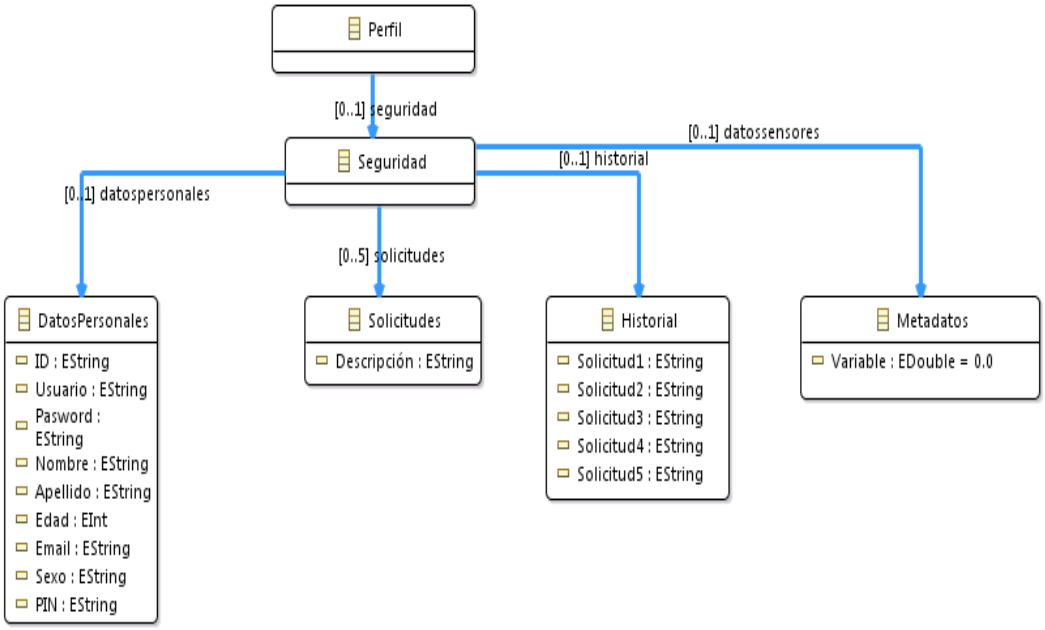

Figura 17. Modelo de Negocio

El modelo de negocio presenta las posibilidades que tiene el usuario al acceder a la plataforma, cada perfil puede almacenar datos personales que comprende el id o identificador, el usuario, nombres, apellidos, edad, email y género, dentro de datos personales también se encuentra en PIN que es una cadena de texto y números que permite al usuario ingresar a la plataforma por medio de su televisión digital, adicionalmente existe

la opción de generar hasta 5 solicitudes, las cuales se guardarán en el historial por último el usuario puede enviar datos recolectados por el sensor de su teléfono inteligente.

Subsistema Interfaz Gráfica Multiplataforma

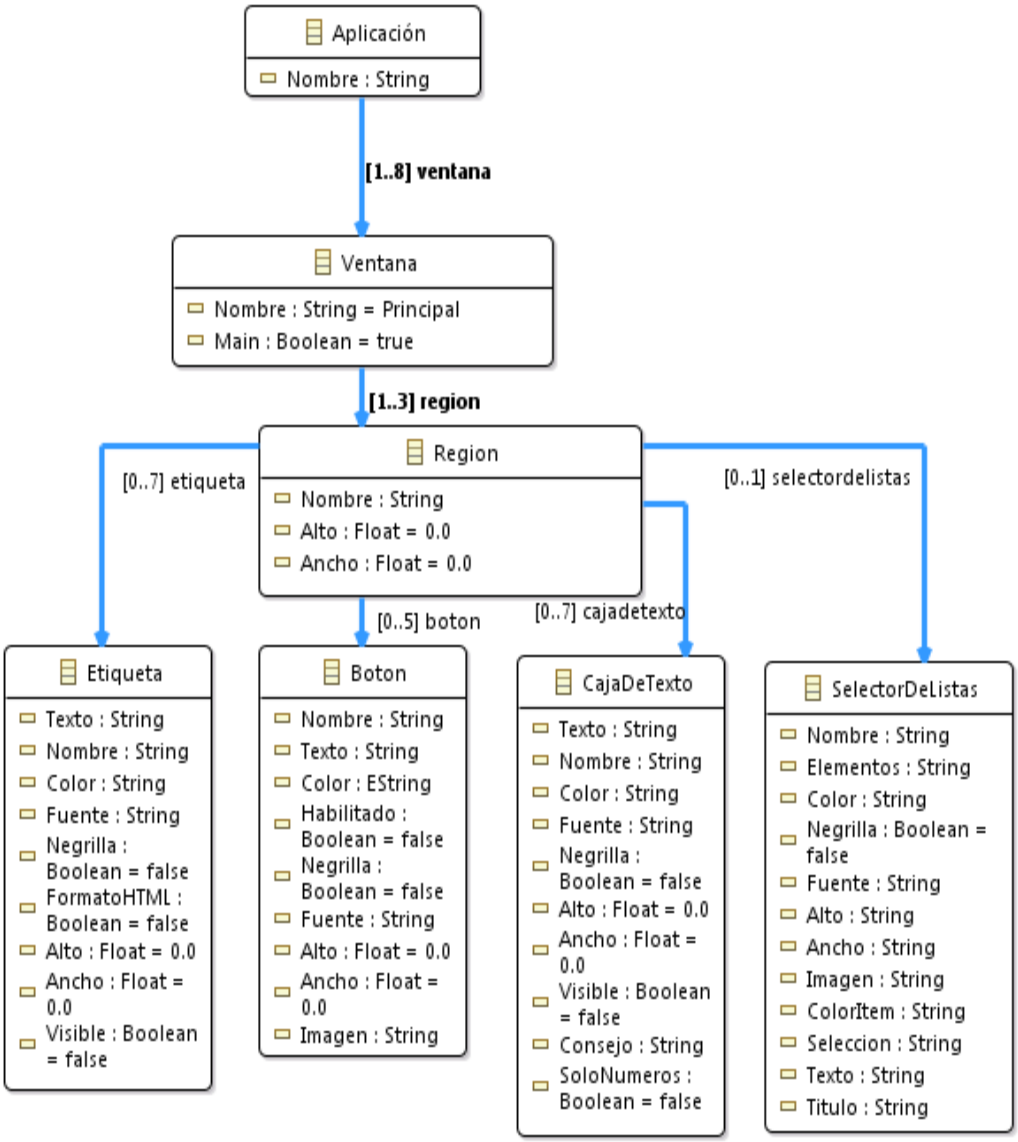

Figura 18. Subsistema Interfaz Gráfica Multiplataforma

En la interfaz gráfica multiplataforma se admite hasta 8 ventanas distintas las cuales a su vez pueden contener hasta 10 cajas de texto donde el usuario puede ingresar

información que posteriormente puede ser enviada al servidor, la ventada puede contener también hasta 10 etiquetas con información de guía y 5 contenedores que pueden desplegar hasta 5 botones distintos.

Subsistema Interfaz Gráfica para TV Digital

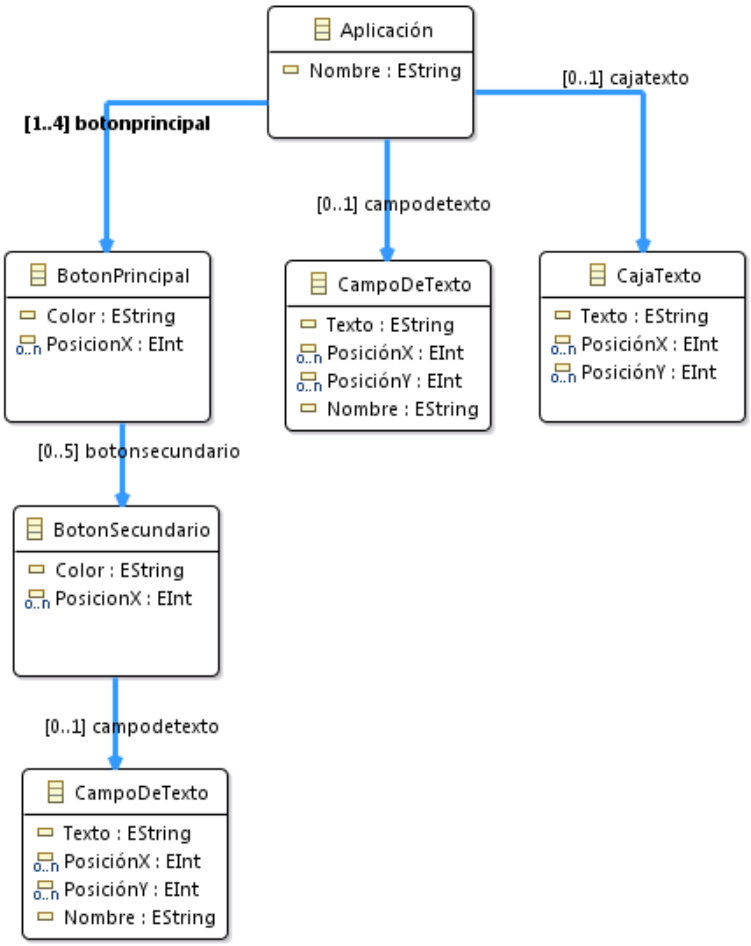

Figura 19. Subsistema Interfaz Gráfica para TV Digital

En la interfaz gráfica para televisión digital, puede desplegarse hasta 4 botones principales controlados por los botones rojo, azul, amarillo y verde del control remoto, estos a su vez al ser presionados pueden presentar 5 botones secundarios lo cuales pueden ser seleccionados con el cursor del control remoto, al presionar estos se puede desplegar un campo de texto que contenga la información solicitada por el televidente.

3.1.4. Modelo PSM

Para el modelado PSM o modelo de plataforma específica, se define el lenguaje de programación a utilizar, se trata del código fuente. A continuación se indica cómo se generaron los modelos para las plataformas específicas de Televisión digital y Android.

Subsistema de Almacenamiento

Para el subsistema de persistencia o almacenamiento se utilizan los siguientes comandos que generan la base de datos y crean los índices necesarios para facilitar las búsquedas:

```
CREATE SCHEMA `saluddatos` DEFAULT CHARACTER SET latin1 ;
CREATE TABLE `saluddatos`.`usuario` (
`id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
`usuario` VARCHAR(255) NOT NULL,
`password` VARCHAR(255) NOT NULL,
`nombre` VARCHAR(255) NOT NULL,
`apellido` VARCHAR(255) NULL,
`edad` INT(2) UNSIGNED NULL,
`email` VARCHAR(255) NOT NULL,
`sexo` VARCHAR(255) NULL,
PRIMARY KEY (`id`),
UNIQUE INDEX `id_UNIQUE` (`id` ASC),
UNIQUE INDEX `usuario_UNIQUE` (`usuario` ASC))
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;
ALTER TABLE `saluddatos`.`usuario`
DROP INDEX `id_UNIQUE` ,
ADD INDEX `idx_id` (`id` ASC),
DROP INDEX `usuario_UNIQUE` ,
```

ADD INDEX `idx_usuario` (`usuario` ASC);

En el anexo 2 se detalla la creación de la base de datos usando el software MySql Workbench.

Subsistema de Seguridad

Para el subsistema de seguridad se utilizó el siguiente método para la plataforma Android: cuando el usuario hace una petición hacia el servidor este le responde con la cadena de texto en formato REST que indica si el usuario es válido o no. Si es válido el usuario accede al menú principal.

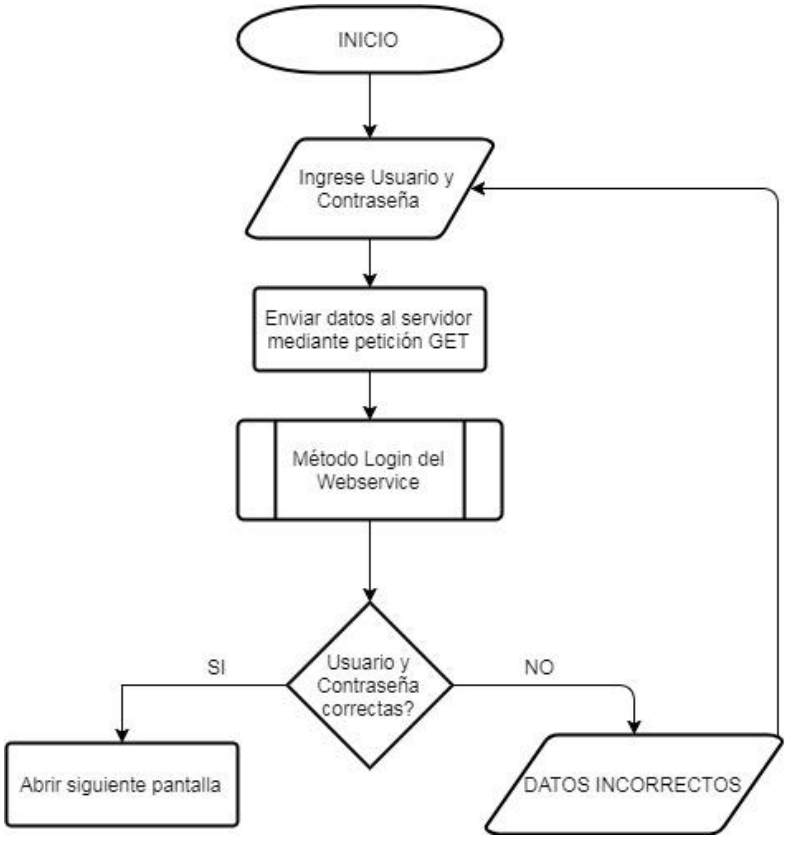

Figura 20. Diagrama de flujo de Autentificación en Android

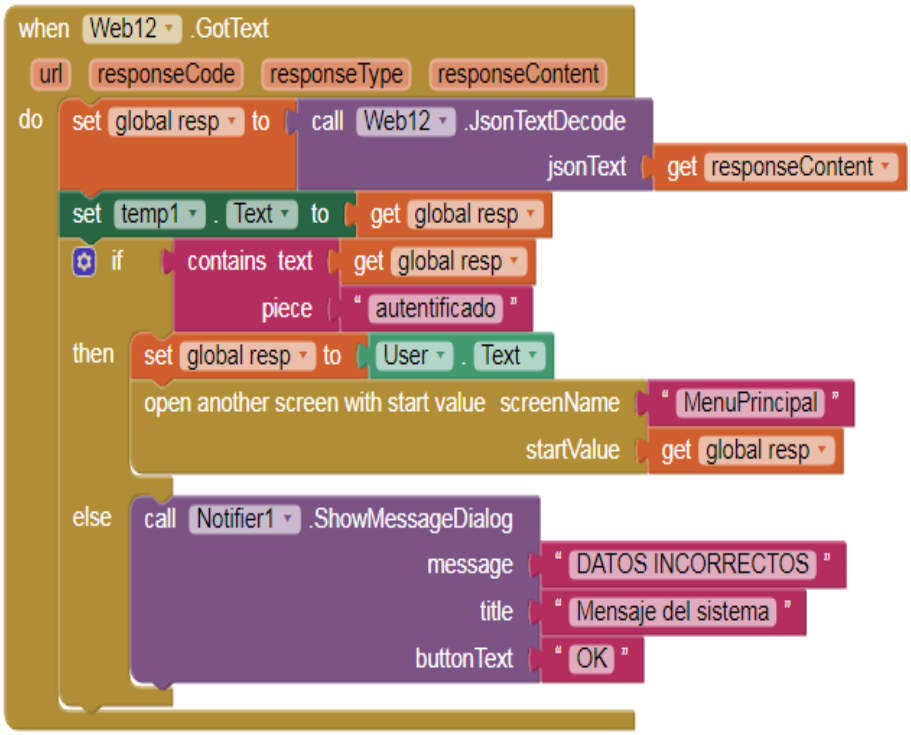

Figura 21. Código para autentificación en Smartphone

Tabla 2:

Descripción de variables para autentificación

Nombre	**Tipo**	**Descripción**
Resp	String	Esta variable almacena la respuesta del servidor para ser procesada posteriormente
Web12	Object	Este objeto permite gestionar la comunicación cliente – servidor mediante los métodos de envío y recepción de datos.
Temp1	String	Funciona como control para el desarrollador, permite procesar las respuestas enviadas por el servidor.
Notifier	Object	El objeto funciona para configurar los mensajes mostrados al usuario de la aplicación.

Para la plataforma de Tv digital se utilizó la siguiente metodología: cuando el usuario hace la petición de ingresar al sistema por medio de su televisor, se envía su PIN

hacia el servidor en donde se ha especificado un método denominado “logintv” el cual devuelve los datos del usuario en caso de existir o validar su PIN:

```
tcp.send('GET /jsonweb/rest/usuario/logintv/'..evt.value..' HTTP/1.1\r\n')
```

Caso contrario se envía una palabra bandera que indica que la identificación es incorrecta, de esta forma los datos de usuario son descargados en el equipo únicamente cuando la autentificación es verificada.

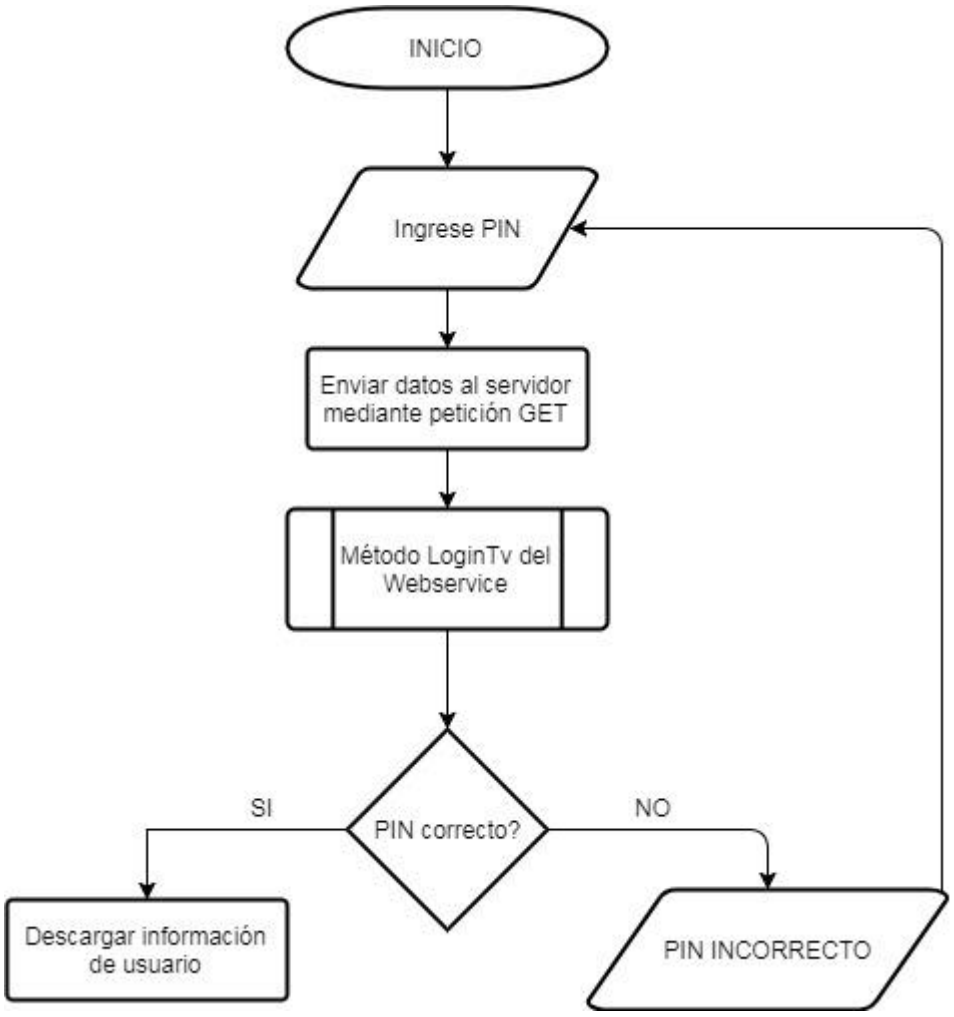

Figura 22. Diagrama de flujo de Autentificación en Tv digital

```
if a ~= 'no existen datos' then
                    writeText("Auntentificado")
                     else
                     writeText("PIN INCORRECTO")
 end
```

Subsistema de Comunicaciones

Para el montaje del servidor se utilizó Apache Tomcat que es una implementación de Java que permite desplegar aplicaciones web de diversos fines. Una de las mayores ventajas de la utilización de Apache es que está desarrollado en código abierto por lo que su utilización es gratuita.

La comunicación multiplataforma se lleva a cabo por medio del servicio Web REST el cual está programado en JAVA y contiene las siguientes clases:

Abstract Facade: Dentro de este programa se encuentran el código que permite la interacción de Java con SQL, creando los métodos de creación, edición y eliminación de registros (create, edit, remove, find, etc.) haciendo uso del método "EntityManager" propio de la librería "javax.persistence". Como se observa en la figura…. el método principal que permite la comunicación entre las diversas plataformas, es aquel que permite obtener los datos de usuario según lo requiera el cliente, para establecer esta comunicación se utiliza el siguiente código, el cual permite localizar un usuario por su nombre y obtener de esta forma todos sus datos.

```
public List<T> findAllByOwner(String owner) {
        EntityManager em = getEntityManager();
        Query query = em.createNamedQuery("Usuario.findByNombre",
Usuario.class);
        query.setParameter("nombre", owner);
        return query.getResultList();     }
```

Dentro de “AbstractFacade” se incluye también el manejo de transacciones que permite la comunicación del programa JAVA con MySql Server, a continuación se muestra el código utilizado para la inserción de un registro dentro de la base de datos:

```
public String create(T entity) {
        try {
            EntityManager em = getEntityManager();
            em.getTransaction().begin();
            em.persist(entity);
            em.getTransaction().commit();
            em.refresh(entity);
            return "ok";
        } catch (Exception e) {
            return e.getLocalizedMessage();
        }
    }
```

UsuarioFacadeREST: se encuentra el llamado a los métodos necesarios que le permitirán al usuario crear, editar y eliminar los registros de la base de datos, aquí se especifica la arquitectura propia de REST, asignando los nombres de PUT, POST y DELETE a los métodos edit, create y remove creados en el archivo “AbstractFacade”. Para la creación de los métodos se utiliza el siguiente esquema de código que cambia de acuerdo al método requerido:

```
    @POST
@Override
@Consumes({MediaType.APPLICATION_JSON})
@Produces({MediaType.APPLICATION_JSON})
public String create(Usuario entity) {
    String resultado = super.create(entity);
```

```
    return String.format(JSON_RESPONSE, entity.getId(), entity.getAndroidid(), "insert", resultado);
  }
```

La utilización del protocolo REST es el que permite que todos los elementos de la arquitectura se comuniquen utilizando un mismo lenguaje, para que esto suceda es necesario especificar el formato de respuesta que debe generar el servidor en contestación de la petición de un cliente, por lo tanto se crea un "formato de cadena" que posibilita el intercambio de información de la manera más transparente posible hacia el usuario:

```
private final String JSON_RESPONSE = "{\"serverId\":%d, \"androidId\":d%, \"operacion\":\"%s\", \"resultado\":\"%s\"}";
```

Con este formato se notifica al cliente e si la operación solicitada fue admitida o rechazada por el servidor. Por ejemplo para la notificación que se genera en caso de que un nuevo perfil haya sido creado exitosamente se envía la siguiente cadena de texto:

```
return String.format(JSON_RESPONSE, entity.getId(), entity.getUsuario(), "creado", resultado);
```

El cliente por su parte debe tener configurado la recepción de contenido en formato JSON así como la correcta decodificación que permita la interpretación de la cadena.

ApplicationConfig: En este archivo se encuentra el "path" o camino por el cual el usuario accederá al servicio Web una vez establecida la comunicación con el servidor, se lo define de la siguiente manera:

```
@javax.ws.rs.ApplicationPath("rest")
```

En el anexo 1 se encuentra detallada la creación del servicio web.

Subsistema Interfaz Gráfica Multiplataforma

El subsistema de interfaz gráfica multiplataforma está pensado para las posibilidades y limitaciones de las plataformas móviles, por tanto su diseño se rige a la distribución general de los teléfonos inteligentes o dispositivos de interfaces similares, se permite la creación de 8 ventanas distintas bajo el siguiente esquema:

Figura 23. Esquema Interfaz Gráfica Multiplataforma

Cada región puede contener hasta 3 regiones, que a su vez pueden contener hasta 7 etiquetas, 7 cajas de texto, 5 botones y un selector de lista, estos elementos pueden estar dispuestos de acuerdo a conveniencia del programador, a continuación se muestra un ejemplo:

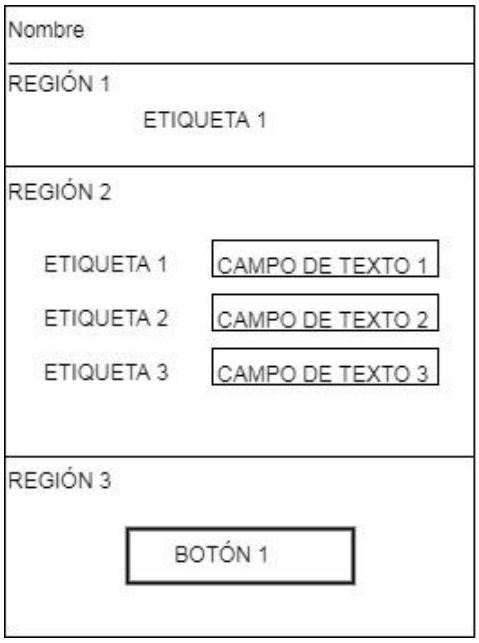

Figura 24. Ejemplo Interfaz Gráfica Android

Subsistema Interfaz Gráfica para TV Digital

Para la creación del subsistema Interfaz gráfica para Tv digital se utiliza el lenguaje Ginga NCL, como se observó en el diagrama en el modelado PDM, la aplicación puede contener hasta 4 botones principales que a su vez pueden desplegar 5 botones más, estos al ser seleccionados pueden presentar una etiqueta.

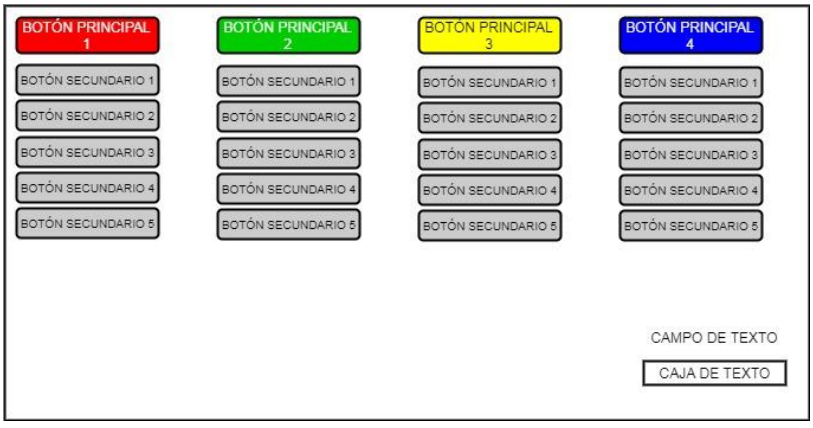

Figura 25. Interfaz Gráfica Televisión Digital

Los 4 botones principales están controlados por los botones rojo, verde, amarillo y azul del control remoto y están programados de la siguiente forma:

```
<link xconnector="conector#onKeySelectionNStartNStopN" id="info">
        <bind role="onSelection" component="binfo" />
        <linkParam name="keyCode" value="RED" />
        <bind role="stop" component="consulta1"/>
        <bind role="start" component="conex1"  />
</link>
```

Los botones secundarios que se presentan al presionar un botón principal, están programados bajo el siguiente código:

```
<link xconnector="conector#onSelectionStartN">
        <bind role="onSelection" component="binfo1"/>
        <bind role="start" component="infotxt"  />
        <bind role="start" component="fondo2"  />
        </link>
```

Al presionar uno de estos botones se despliega la información que ha solicitado el usuario en una etiqueta cuyo código es:

```
 <region id="r11" left="3%" bottom="2%" height="15%" width="75%" zIndex="3"/>
        <descriptor id="d11" region="r11" />
<media id="infotxt" src="info.txt" descriptor="d11">
                <property name="fontSize" value="16"/>
        </media>
```

3.2 Implementación de la arquitectura

A continuación se muestra una representación del funcionamiento de la arquitectura y su respuesta frente a las diversas acciones que tome el usuario.

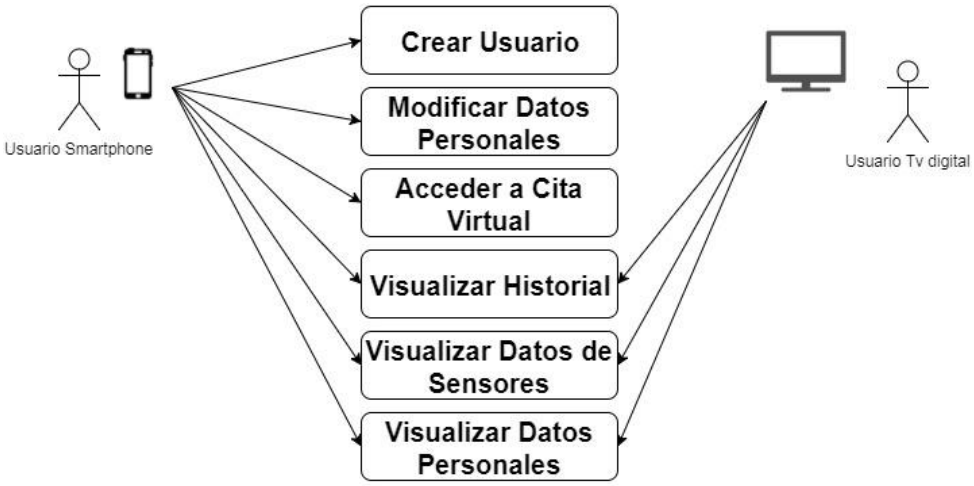

Figura 26. Casos de uso

3.2.1. Desarrollo de la aplicación para TV Digital

Con el fin de poner a prueba la arquitectura propuesta se ha desarrollado una aplicación para TV digital interoperable con la plataforma móvil Android, se trata de una consulta médica virtual que se opera de la siguiente forma:

El programa dirigido por un médico se transmite en un canal predeterminado, a la par se transmite la aplicación interactiva, en este punto existe 3 posibles escenarios:

* Usuario sin acceso a Internet: En este caso no se cuenta con canal de retorno y el televidente únicamente podrá observar el contenido de la aplicación enviada vía broadcast la cual contiene un menú de información acerca del tema que se está tratando.

* Usuario con acceso a Internet: En este escenario el usuario podrá acceder a su perfil mediante un usuario y contraseña, una vez autentificado se podrá observar su historial médico, sus datos personales y los datos de sensores.

* Usuario con acceso a Internet y Smartphone: El televidente podrá acceder tanto desde su Smartphone como desde su televisor a su perfil, en el Smartphone, el usuario podrá modificar sus datos personales y subir datos recolectados por los sensores del teléfono como por ejemplo el ritmo cardiaco, en la prueba de la arquitectura desarrollada se utilizó el sensor de podómetro. También tendrá la opción de "pedir una cita" en este caso deberá enviar la descripción de su problema desde la aplicación Android, la solicitud se enviará hacia el servidor en donde se gestionará para permitir al usuario participar del programa en el momento en que llegue su turno. Cuando un usuario crea una cuenta, se le asigna aparte de la contraseña, un pin que le servirá para ingresar a los servicios de la plataforma desde su Televisor Digital.

La aplicación está orientada a la educación en salud para el público televidente, de esta forma el usuario en casa puede interactuar con el programa y conocer consejos de médicos que aporten a mejorar la calidad de vida. La aplicación también podría ser usada como material didáctico que permita a los galenos en formación aumentar su experiencia en el diagnóstico de enfermedades y el tratamiento con los pacientes.

La interfaz de la aplicación para Tv Digital consta de 4 opciones manejados por los botones Rojo, Verde, Amarillo y Azul del control remoto. En la parte inferior derecha se encuentra el campo que el usuario debe utilizar para ingresar su pin y de esta forma autentificarse como usuario válido de la plataforma. En esta parte de la pantalla está también el botón "Cerrar Sesión" que borra todos los datos del usuario descargados en el equipo, la descripción del funcionamiento del resto de botones se especifica a continuación:

Información: Controlado por el botón rojo

Al presionar este botón se despliega 2 botones más, al apretar la opción “Internet” el usuario podrá acceder a información actualizada acerca del tema que se está tratando en ese momento, la información obtenida mediante este botón se la puede actualizar en cualquier instante.

Figura 27. Botón información

Fuente Video: (Echeverría, 2013)

Historial: Controlado por el botón verde

El usuario visualizará las opciones para consultar sus consultas realizadas previamente desde su Smartphone. Mediante la utilización de las flechas del control remoto, podrá desplazarse desde la opción de consulta número 1 hasta la número 5, al presionar el botón “ok” de despliega la información en la parte inferior izquierda de la pantalla.

Figura 28. Botón historial

Fuente Video: (Echeverría, 2013)

Perfil: Controlado por el botón amarillo

Dentro de este menú el usuario puede acceder a visualizar sus datos personales o los datos de sensor enviados desde su Smartphone.

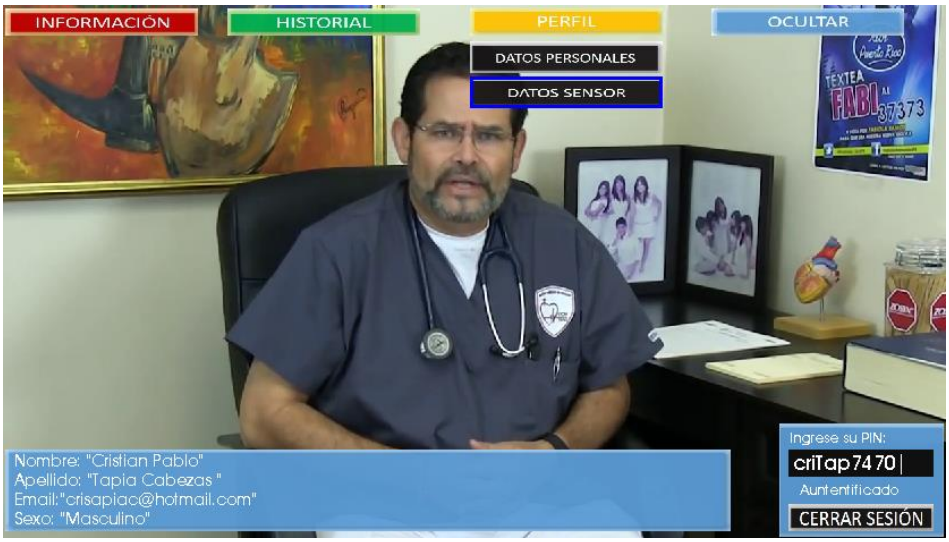

Figura 29. Botón perfil

Fuente Video: (Echeverría, 2013)

Ocultar: Controlado por el botón azul

Si el usuario no desea interactuar con la aplicación, el uso de este botón le permite ocultar todo el contenido.

Figura 30. Interfaz de la aplicación para Tv Digital
Fuente Video: (Echeverría, 2013)

3.2.1.1. Estructura del programa NCL para la prueba de la arquitectura

La estructura del programa en ncl realizado para esta aplicación es el siguiente:

En la declaración de "regionbase".- en esta parte se especifica la posición de cada uno de los elementos de la aplicación así como su tamaño.

```
<region id="r0"  height="100%" width="100%"  zIndex="1"/>
  <region id="r1" left="2%" top="2%" height="5%"    width="20%" zIndex="3"/>
  <region id="r2" left="2%" top="8%"  height="4%"    width="20%"
zIndex="3"/>
```

La región 0 por ejemplo, que corresponde a la región utilizada por el video establece que el elemento multimedia ligado a esta región, utilizara el cien por ciento de la pantalla

y tiene zIndex de 1 lo cual indica que su transparencia o su ubicación respecto a otros objetos es la más posterior.

Dentro de “descriptorbase” se asigna el “focusIndex” que indica en qué orden se seleccionan los elementos cuando el usuario está navegando por la aplicación con las flechas de su control remoto.

```
<descriptor id="d13" region="r13" focusIndex="1" moveDown="2" moveUp="7" />
        <descriptor id="d13a" region="r13a" focusIndex="2" moveUp="1" moveDown="3" />
```

En el código anterior se estable el índice el objeto multimedia en 1 lo que quiere decir que cuando este sea inicializado, el foco de navegación se moverá hasta este objeto si no existiese uno de menor índice.

Dentro de “body” se hace la declaración de los objetos multimedia especificando su ubicación dentro de la carpeta que contiene la aplicación y sus propiedades, por ejemplo para el objeto multimedia “infotxt” que corresponde al texto que se desplegará cuando el usuario apreté el botón de información sin internet se tiene el siguiente código:

```
<media id="infotxt" src="info.txt" descriptor="d11">
        <property name="fontSize" value="16"/>
</media>
```

Se puede observar que se establece la propiedad “fontSize” en 16 lo cual significa que el texto tendrá un tamaño de 16.

Para que el usuario puede autentificarse ingresando se agrega los objetos media LUA que permiten al usuario ingresar su PIN haciendo uso del teclado numérico del control remoto:

```
<media id="input" src="input.lua"  descriptor="d13">
        <property name="text"/>
        <area id="select"/>
</media>
<media id="conex" src="conex.lua" descriptor="d14">
        <property name="search"/>
        <property name="result"/>
</media>
```

El objeto input.lua es una caja de texto creado mediante el módulo Canvas de lua que permite el ingreso del pin de usuario haciendo uso de las teclas numéricas del control remoto. El objeto conex.lua es el que permite la conexión con el servicio REST, será explicado más adelante.

Para que el programa ejecute las acciones necesarias cuando el usuario apreté un botón se utilizó conectores de la siguiente forma:

```
<link xconnector="conector#onBeginStartN" id="cbinfo">
        <bind role="onBegin" component="video"/>
        <bind role="start" component="binfo" />
                        …
</link>
```

Este conector es utilizado para iniciar objetos multimedia cuando otro objeto ya fue iniciado, de esta forma el momento en el que inicie el video, los botones rojo, verde, amarillo y azul serán también iniciados o presentados en la interfaz.

Cuando el usuario termina de ingresar su pin y aplasta “enter”, se dispara el siguiente conector que hace un pase de parámetros hacia el servidor para la autentificación de usuario:

```
<link xconnector="conector#onBeginSet">
        <bind role="onBegin" component="input" interface="select"/>
        <bind role="get" component="input" interface="text"/>
        <bind role="set" component="conex" interface="search">
            <bindParam name="var" value="$get"/>
        </bind>
</link>
```

Lo que indica este código es que cuan inicie la interface “select” del objeto input, es decir cuando se presione “enter” estando señalado el objeto “input” por el cursor, se estableza la variable “search” del objeto conex en el valor ingresado por el usuario en el objeto “input”, de esta forma se hace el envío del pin hacia el servidor el cual validará o no al usuario.

El resto de botones utilizar el conector “OnSelectionStartStop” que permite desplegar el contenido que requiera el usuario y ocultar el que ya no es de su interés, por ejemplo cuando el usuario aplasta el botón de datos de sensor, se ejecuta el siguiente código:

```
<link xconnector="conector#onSelectionStartNStopN">
        <bind role="onSelection" component="bsensor"/>
        <bind role="start" component="sensor"  />
        <bind role="start" component="fondo2"  />
        <bind role="stop" component="nombre"  />
</link>
```

El código permite iniciar el texto que contiene los datos de sensor enviados por el usuario y detener el contenido que el usuario estaba visualizando antes de hacer clic en el botón.

3.2.2. Creación de un Cliente Web por medio de LUA

Para la utilización del canal de retorno en televisión digital, se requiere del lenguaje de programación Lua. Los scripts Lua pueden ser embebidos dentro del programa NCL el cual los ejecutará conforme el programa requiera de estos. El ejemplo de arquitectura propuesto utiliza 2 archivos de LUA que permiten la conexión al servidor REST, estos son tcp.lua y conexion.lua. Dentro de tcp.lua se encuentran todas las directivas necesarias para realizar las llamadas asincrónicas y ejecutar los procedimientos de conexión, envío, recepción y desconexión propios de la comunicación TCP.

"La función principal, `tcp.execute`, puede ejecutar los siguientes comandos:

* `Tcp.connect`: recibe la dirección y el puerto de destino

* `Tcp.send`: recibe una cadena con el valor que se va a transmitir

* `Tcp.receive`: devuelve la cadena con el valor que se va a recibir

* `Tcp.disconnect`: cierra la conexión"

Dentro del archivo conexion.lua se hace el llamado a la función tcp.execute con los parámetros de configuración para establecer la conexión con el servidor REST, así mismo se envía los parámetros de requerimiento al servidor mediante el comando tcp.send con el fin de transaccionar la autentificación del usuario, de la misma manera enviando una nueva solicitud al servidor, la aplicación interactiva puede desplegar información respecto al historial del usuario autentificado. Finalmente se ejecuta el comando tcp.disconnect() para cerrar la conexión con el servidor. El código utilizado se detalla en el anexo 3.

3.2.3. Desarrollo de la aplicación Android

Para la interfaz gráfica multiplataforma se utilizó el programa appinventor. Dentro de la aplicación desarrollada, el usuario puede navegar por 7 ventanas, la primera es la correspondiente al subsistema de seguridad en donde el usuario debe ingresar su usuario y contraseña para poder acceder:

Figura 31. Acceso plataforma Android

Si el usuario no tiene una cuenta deberá crear una pulsando el botón "Crear nueva cuenta", a continuación se muestra la pantalla que permite ingresar datos de: usuario (debe ser único en la base de datos, si no lo es se emite una notificación indicando que se elija otro usuario), contraseña, nombre, apellido, edad y email. Una vez creado el usuario el sistema genera automáticamente el código pin que será mostrado en la pantalla de datos de usuario como se muestra más adelante.

Para la creación de un nuevo usuario siguiendo las directrices de la arquitectura REST, es necesario utilizar el método POST adjuntado los datos del usuario para

agregarlos al subsistema de almacenamiento. Para tal propósito se sigue la siguiente lógica de programación mostrada en el siguiente diagrama de flujo

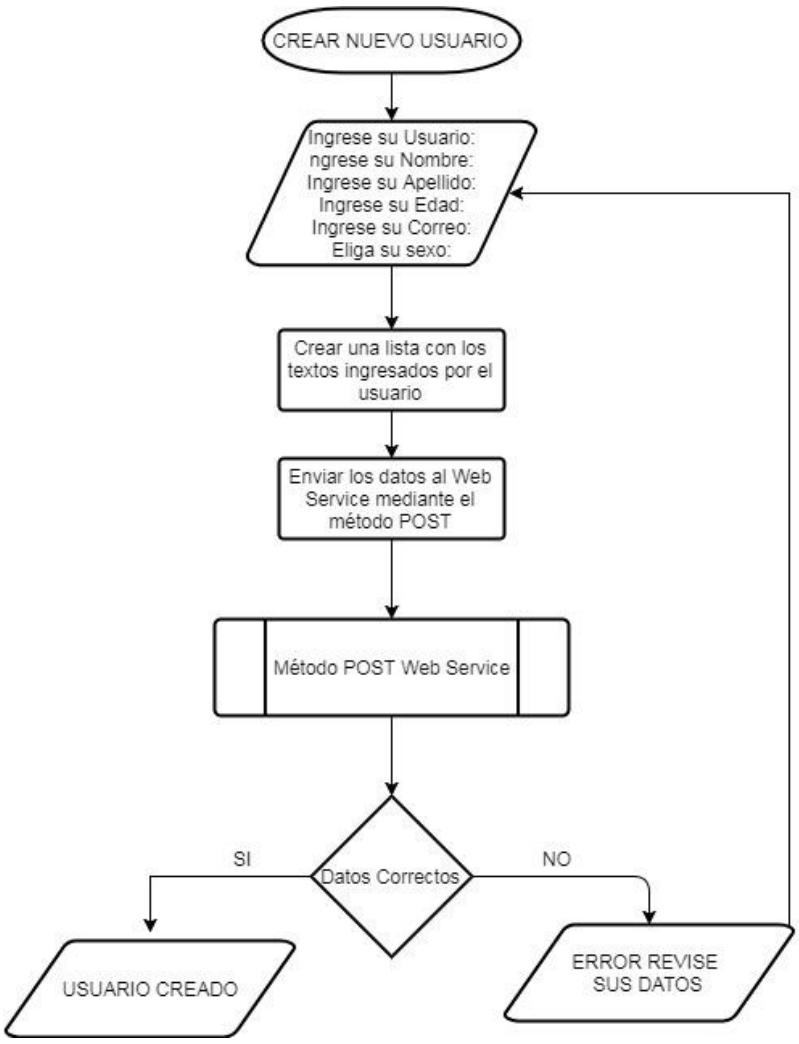

Figura 32. Diagrama de Flujo Creación de Usuario

En el siguiente segmento de código se muestra la rutina que se realiza para la creación de un nuevo usuario, se crea una lista que contiene tanto el identificador del campo que se va a enviar como el texto ingresado por el usuario:

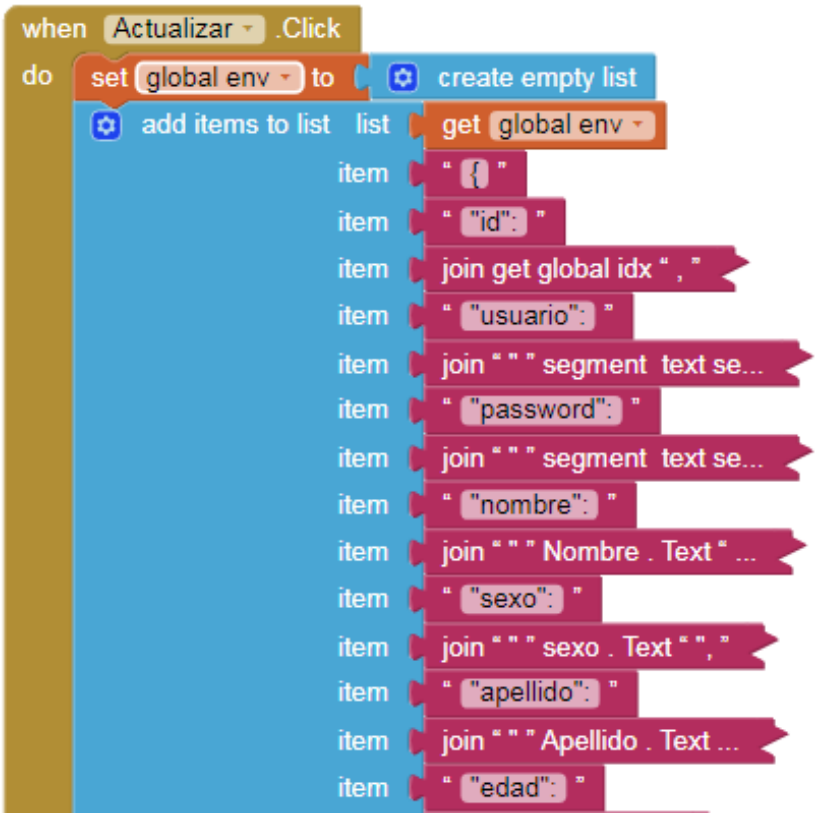

Figura 33. Código creación de usuario

Luego se envía la información con el método POST que como se mencionó anteriormente, sirve para la creación de un registro en la base de datos:

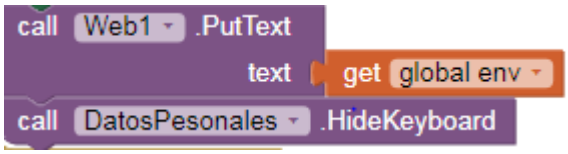

Figura 34. Código envío de información

Si el usuario ya está registrado en el sistema, una vez autentificado, puede navegar por la aplicación a través del menú principal de la misma:

Figura 35. Menú Principal Aplicación Android

Al seleccionar el botón de datos personales, el usuario pude modificar su información generada cuando creo su cuenta por primera vez, una vez que ha efectuado los cambios el botón actualizar permite enviar la información hacia la base de datos.

Figura 36. Actualizar datos personales

Al ingresar en la opción “enviar datos de sensor” el usuario tiene la posibilidad de guardar datos generados por el sensor de podómetro de su celular:

Figura 37. Enviar datos de Sensor

Dentro de la interfaz “Pedir Cita”, el usuario puede enviar la descripción de su problema hacia los servidores y guardar un registro de hasta 5 citas médicas virtuales. Primero deberá elegir el número de consulta que desea enviar y luego deberá ingresar su inquietud en el campo de texto.

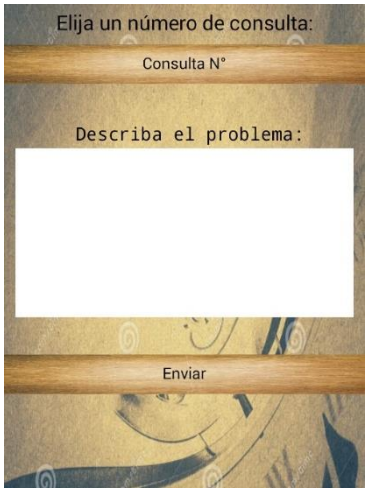

Figura 38. Enviar datos de consulta médica

Para enviar la información de la consulta, se utiliza el método PUT pues se trata de una modificación del registro de usuario:

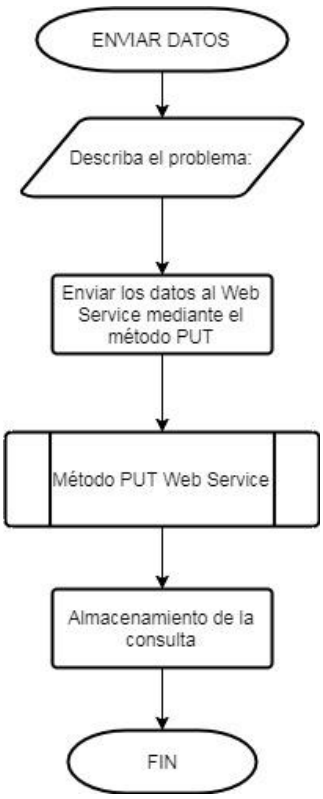

Figura 39. Diagrama de flujo de Enviar consulta

Para el envío de la consulta al servidor se utiliza el mismo código exhibido en la figura número 35.

Al hacer clic en el botón de historial, el usuario pude acceder a revisar 5 consultas virtuales almacenadas en su perfil:

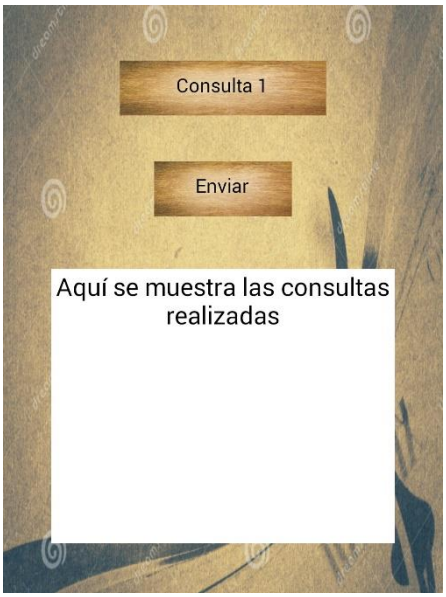

Figura 40. Historial Aplicación Android

Si el usuario hace clic en la opción "cerrar sesión", regresará a la interfaz de autentificación de usuario donde deberá ingresar sus datos para acceder a la plataforma.

3.2.4. Web Service REST en Android

Para consumir un servicio REST mediante una aplicación Android se utiliza el componente no visible "web", este posee los comandos necesarios para establecer la comunicación con el servidor y enviar solicitudes del tipo GET, POST, PUT o DELETE.

El primer paso es conectarse al servidor mediante la instrucción "set Web.url to "especificando la dirección http del servidor.

A continuación se ejecuta la instrucción "call Web.Get" para realizar la petición que se requiera al servidor:

GET: en caso de requerir información del usuario

POST: en caso de creación de un usuario

PUT: en caso de edición de la información de un usuario

DELETE: en caso de requerir borrar el registro de un usuario.

Una vez respondida la solicitud se puede gestionar su contenido mediante la instrucción “Web.GotText”, ahora bien es necesario decodificar esta información con el uso de “Web.JsonDecode” de esta forma estamos transformando la respuesta del servidor a texto simple el cual será más fácil gestionarlo.

CAPÍTULO 4

PRUEBAS Y ANÁLISIS DEL ESCENARIO

4.1 Pruebas en laboratorio

4.1.1. Descripción del escenario

Las pruebas se realizaron en el laboratorio de Televisión Digital de la Universidad de las Fuerzas Armadas ESPE donde se utilizó el software Village Flow versión 2 instalado en un servidor de Playout con las siguientes características:

- ROM 680 Gb
- Sistema Operativo Windows 7
- 16 Gb RAM
- Procesador Intel(R) Core(TM) i7-2600 CPU @ 3.400Ghz 3.40 Ghz
- Tarjetas de video Nvidia GeForce 9500 GT
- Adaptador de red 802.11bgn 1T1R Wireless Adapter
- DTA-2111 – Multi – standard VHF/UHF modulator for PCI Express
- DTA – 2144 – Quad ASI/SDI in+out for PCI Express

Para la utilización del canal de retorno que permite la comunicación de la aplicación visualizada en el televisor con el servidor REST se utilizó un router marca TP-link modelo TL-WR841N, el decodificador posee en su panel posterior una entrada para un cable RJ-45 que fue utilizada para acceder a la red configurada para las pruebas de la aplicación.

Para la integración de la plataforma móvil con Tv digital se instaló la aplicación desarrollada para Android en un Smartphone marca Lenovo Vibe X, este igualmente se conectó vía Wi-fi a la red configurada en el router Tp-link.

El servidor REST se desplegó en un equipo portátil de las siguientes características:

- ROM 600 Gb
- 8 Gb RAM
- Procesador Intel(R) Core(TM) i5-2450 CPU @ 2.50Ghz
- Sistema Operativo Windows 7
- Adaptador de red 802.11bgn 1T1R Wireless Adapter

Para la etapa de recepción se utilizó un decodificador marca EITV el cual es un Set-top Box orientado al desarrollo de aplicaciones para Tv Digital que posee características de compatibilidad completa con los estándares "declarativos (Ginga-NCL / Luna) y el imperativo (Ginga-J / JavaDTV) que se han especificado para el Sistema Brasileño de Televisión Digital (SBTVD)." (EITV, 2011)

Figura 41. Decodificador EITV
Fuente: (EITV, 2011)

Las principales características técnicas del decodificador se resumen en el siguiente cuadro:

Tabla 3:

Características Técnicas Decodificador

Característica	Especificación
Front-end (Sintonizador + Demodulador)	
Frecuencia de entrada	UHF: 470 MHz (CH14) a 806 MHz (CH69) VHF: 174 MHz (CH07) a 216 MHz (CH13)
Ancho de banda	5,6 MHz
Nivel de siñal	-85 dBm a -20 dBm
Siñal	Compatible con el estándar ISDB-T
Impedancia de entrada	75 Ohms (nominal)
Conexión de E/S	Conector F
Unidad de procesamiento	
Procesador	STi 7105 (CPU 450 MHz)
Memoria RAM	256 MBytes
Memoria Flash	128 MBytes
Descodificación de video	
Padrão	Rec. ITU-T H.264 (MPEG-4 AVC)
Profile	HP@L4.0
Formatos	480i y 1080i
Frame rate	25, 30, 50 y 60 MHz
Frecuencia de video	50 y 60 Hz
Descodificación de audio	
Patrón	ISO/IEC 14496-3 (MPEG-4 AAC) (SBR)
Profile	AAC@L4 y HE-AAC@L4
Panel Frontal	
Botones	Cambio de canales y on/off (Standby)
	Sensor de infrarrojos para mando a distancia

Fuente: (EITV, 2011)

La estación de televisión simulada en el servidor de Playout mediante el software Village Flow es la encargada de la generación del Transport Stream en el cual se encuentra embebida la aplicación GINGA, luego crea el carrusel de información que viajará a través del cable coaxial utilizado como medio de transmisión, el decodificador re ensambla el contenido y lo decodifica para enviarlo hacia la televisión a través del cable HDMI , el decodificador a su vez está conectado mediante cable UTP categoría 5 a la red configurada con las siguientes direcciones ip:

Tabla 4:

Distribución de direcciones IP para escenario de laboratorio

Dirección Ip	Dispositivo
192.168.1.1	Router Tp-link
192.168.1.2	Servidor REST
192.168.1.3	Smarphone
192.168.1.4	Decodificador EITV

Para las pruebas realizadas en el laboratorio se utilizó el siguiente diagrama de conexión:

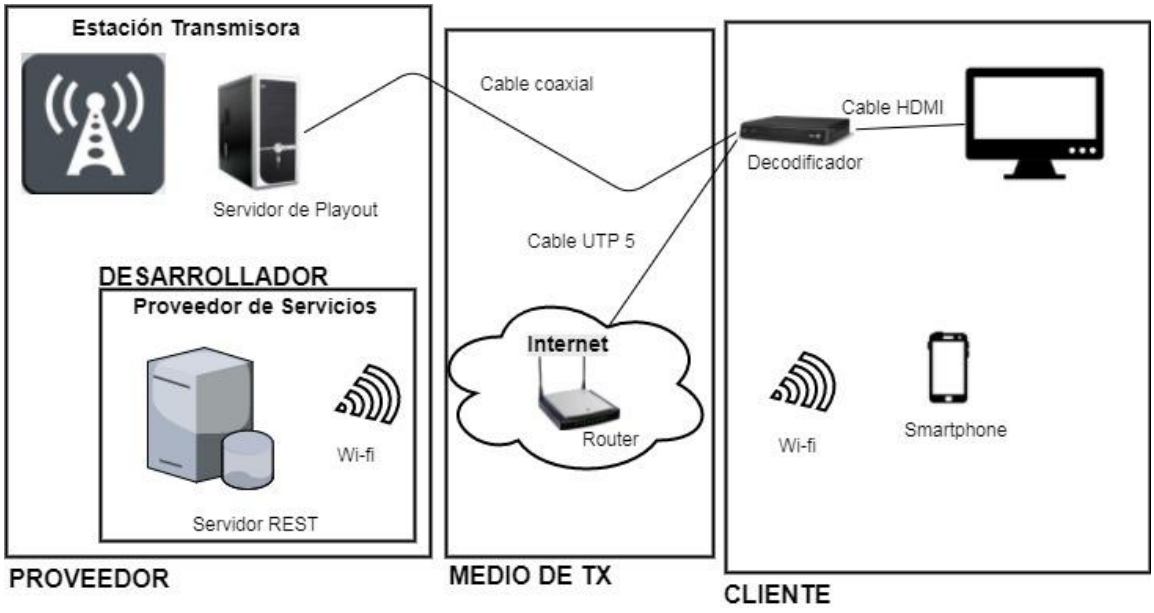

Figura 42. Escenario de prueba en laboratorio

El smartphone en el cual se encuentra instalada la aplicación desarrollada se conecta vía wi-fi a la red configurada en el router Tp-link mediante la utilización del protocolo DHCP, el proveedor del servicio q a su vez es el desarrollador es el encargado de actualizar la información y atender a las peticiones que los clientes realizan por medio del televisor o el Smartphone.

4.1.2. Configuración de los equipos

La interfaz de usuario de Village Flow presenta un diagrama de bloques a través del cual se configura los parámetros necesarios para la transmisión, para las pruebas realizadas se obvió la transmisión de video y audio y se envió únicamente la interactividad.

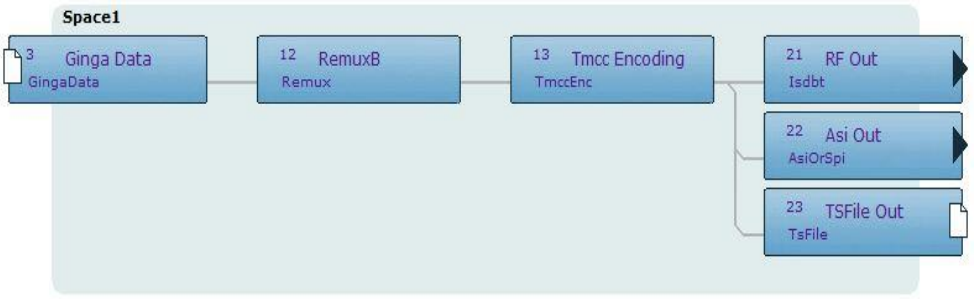

Figura 43. Diagrama de bloques Village Flow

El primer bloque es el correspondiente a los datos de la aplicación Ginga, aquí se especifica la ruta del archivo XML que se encarga de re direccionar hacia el archivo ejecutable principal de la aplicación NCL, además se configuran los siguientes parámetros:

Figura 44. Configuración del bloque Ginga Data

En el directorio especificado en el parámetro path se encuentra la carpeta app13 la cual contiene todos los archivos necesarios para la ejecución de la aplicación interactiva, es decir todos los objetos media y archivos .lua que permiten la interacción con canal de retorno. Se especifica también el número de identificador de paquete que tendrá el TS correspondiente a los datos de la aplicación GINGA.

El bloque Remux o remultiplexor es el encargado de "adaptar el flujo TS-MPEG a las necesidades de funcionamiento establecidas en el sistema ISDB-T", (Pisciotta, 2014). Como se observa en el diagrama de bloques, a la entrada se colocan todos los TS correspondientes a los programas que se quieren transmitir y también para el caso de prueba, el archivo de la aplicación de interactividad.

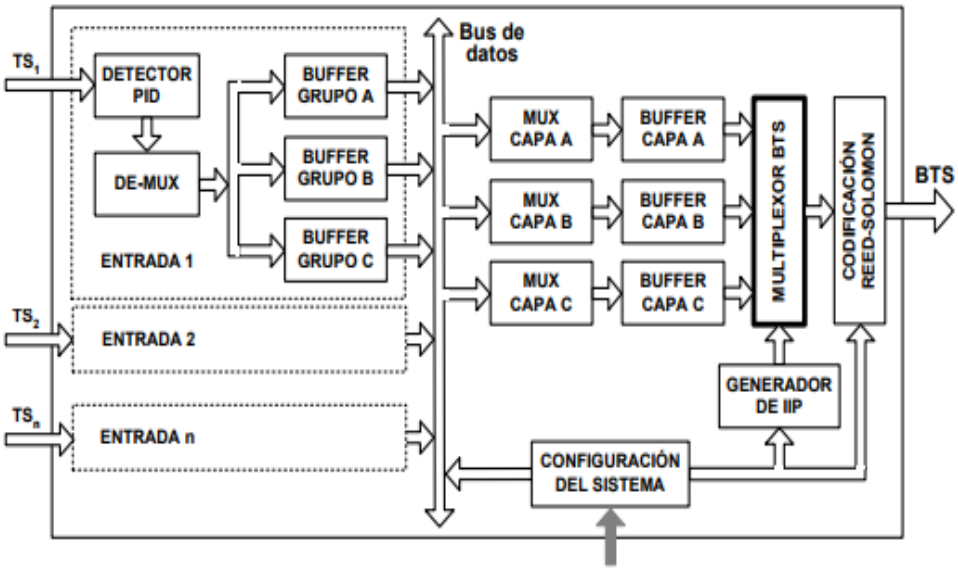

Figura 45. Diagrama de Bloques del Remultiplexor (Pisciotta, 2014)

En el bloque REMUX de la interfaz de usuario de Village Flow, es necesario configurar los PID o identificadores de paquetes para identificar cada uno de los programas transportados en los flujos TS entrantes:

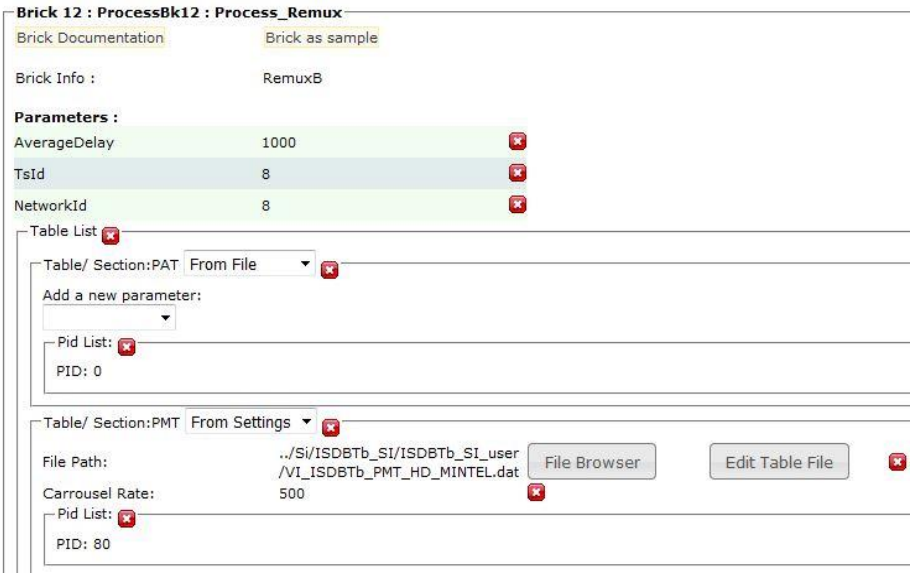

Figura 46. Configuración bloque Remux

En la etapa de codificación que utiliza los parámetros propios de ISDBT-tb MPEG-2 para audio y MPEG-2 y MPEG-4 AVC /H.264 para video se configuran los siguientes parámetros:

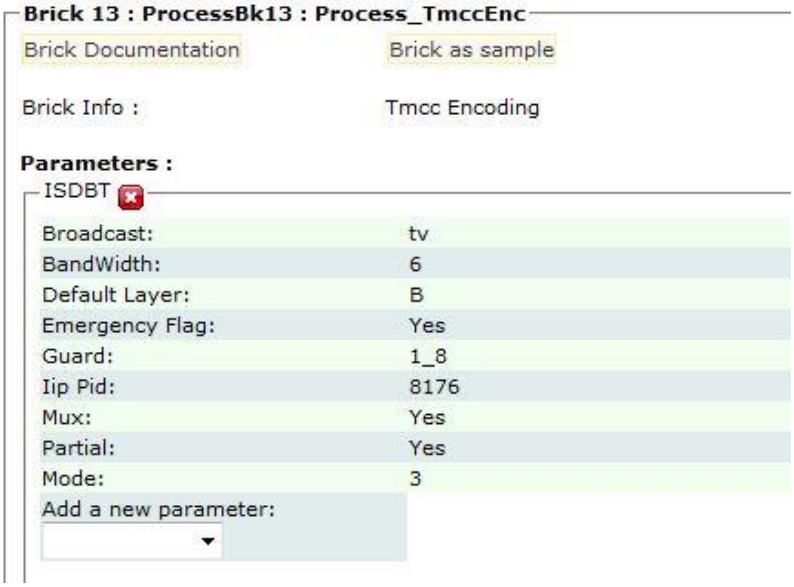

Figura 47. Configuración bloque de Codificación

Por último de los tres bloques de salida, se utilizó el de RF que controla la tarjeta de video del servidor el cual posee un conector de salida RF para cable coaxial que va conectado directamente al decodificador de la televisión. Es necesario precisar parámetros como el canal de RF y la potencia de transmisión, así como la modulación que en este caso se utilizó QAM-64:

Figura 48. Configuración bloque Salida RF

4.1.3. Pruebas realizadas

Para el botón de información se presentan dos escenarios:

Cuando el usuario utiliza el botón de información sin conexión a internet, se comprobó que únicamente podrá visualizar contenido que ha sido transmitido por la aplicación, si se requiere un cambio de este, es necesario realizar nuevamente la transmisión de la aplicación.

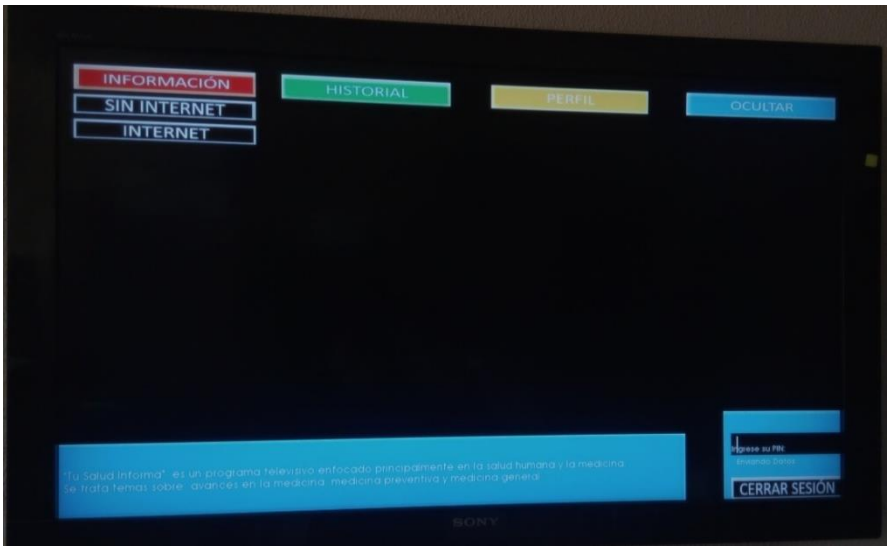

Figura 49. Botón de información sin canal de retorno

Cuando el usuario utiliza el botón de información con internet, el administrador únicamente tiene que editar la página web informacion.html con el contenido que desee que el usuario visualice en su televisor.

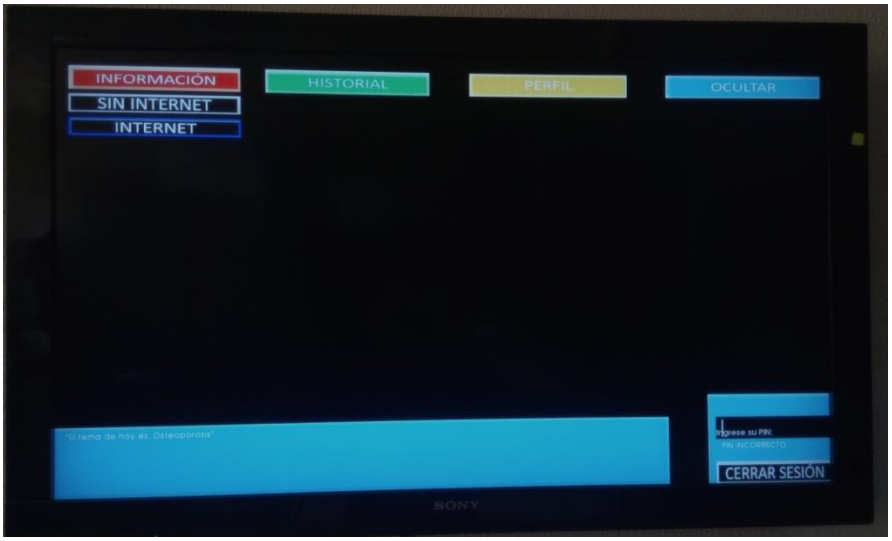

Figura 50. Botón de información con canal de retorno

En el Smartphone haciendo uso de la aplicación desarrollada se procedió a crear un usuario y enviar datos de consulta y de sensor:

Figura 51. Creación de usuario de prueba

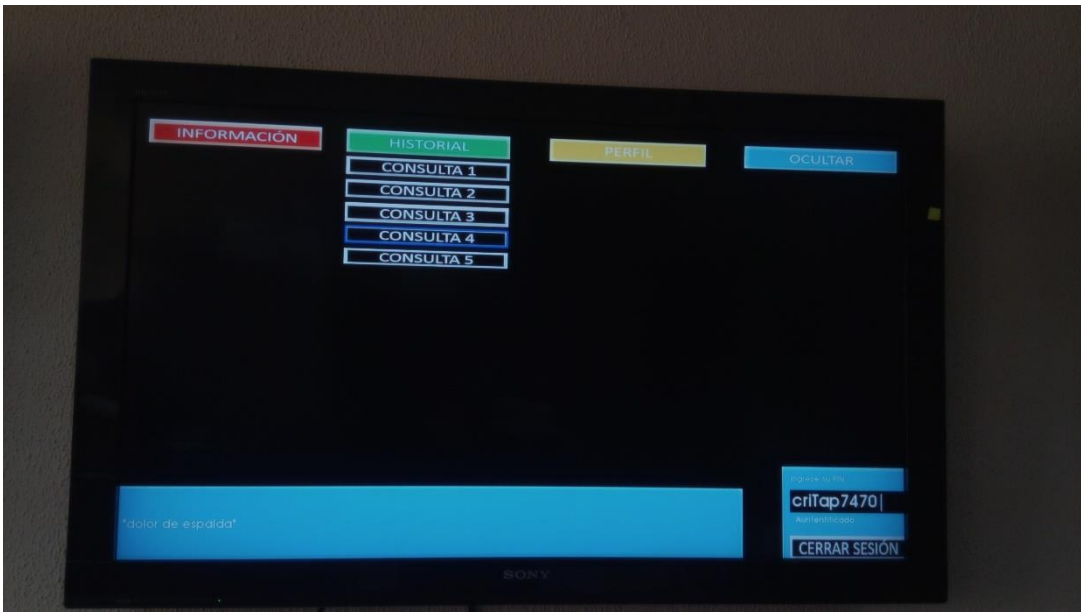

Figura 52. Visualización de historial en GINGA

Como se observa en la Figura 53, se procedió a transmitir la aplicación y posteriormente a autentificar el usuario en la Televisión Digital haciendo uso de los números del control remoto, accediendo al historial de usuario se visualizó la consulta generada desde la aplicación Android.

Luego en el botón perfil se visualizó los datos personales y los datos de sensor enviados desde el Smartphone:

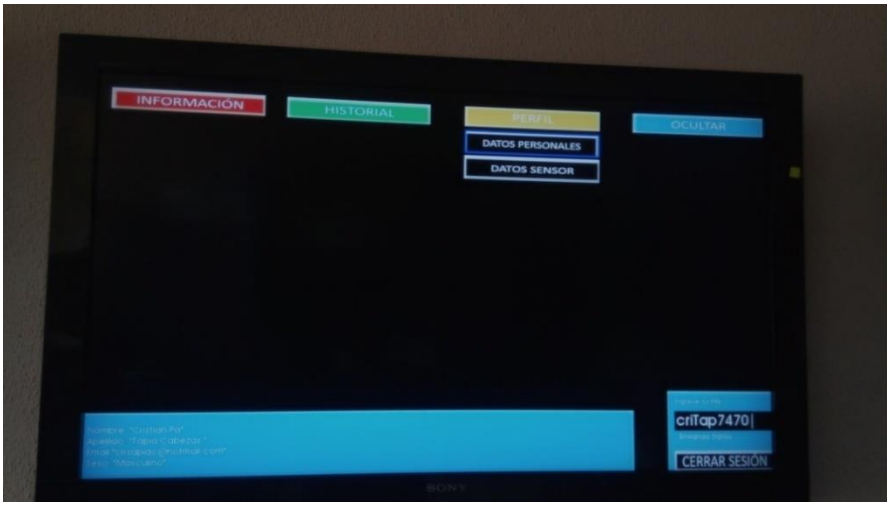

Figura 53. Visualización de datos personales en GINGA

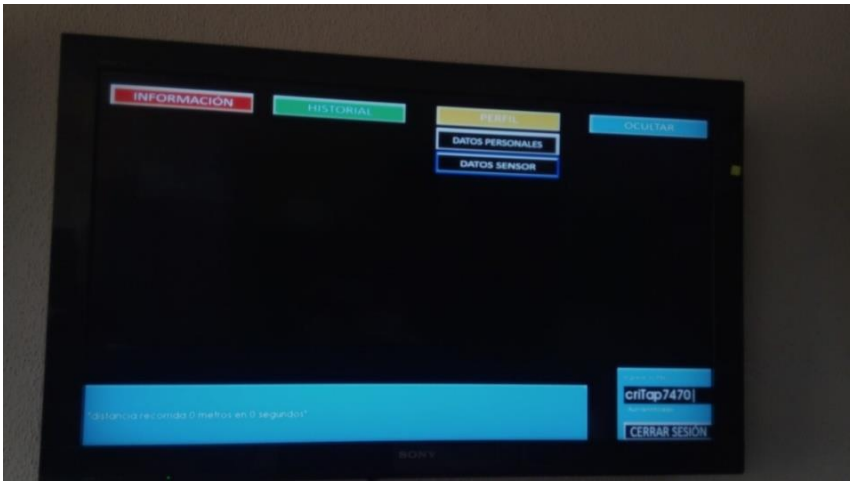

Figura 54. Visualización de datos de sensor en GINGA

4.2 Pruebas en simulador

Las pruebas en simulador se realizaron en una laptop con las siguientes características:

- ROM 600 Gb
- 8 Gb RAM
- Procesador Intel(R) Core(TM) i5-2450 CPU @ 2.50Ghz
- Sistema Operativo Windows 7
- Adaptador de red 802.11bgn 1T1R Wireless Adapter

A diferencia del escenario planteado en la sección anterior, en este caso tenemos embebido en el simulador los sistemas de transmisión Broadcast, proveedor de servicios y cliente de Televisión Digital.

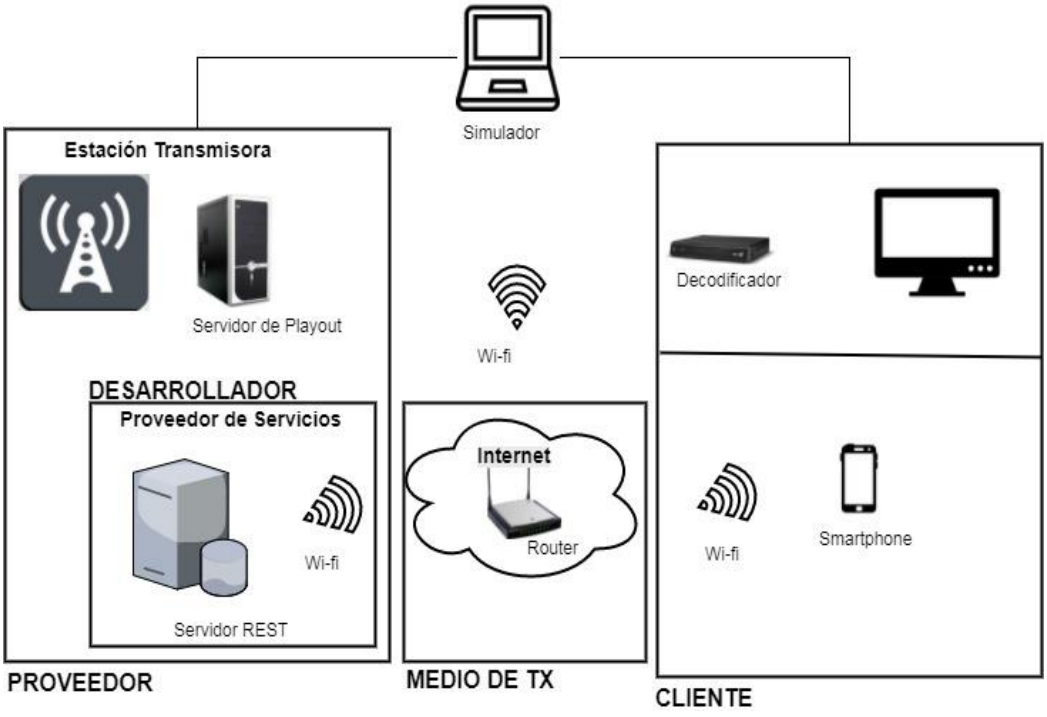

Figura 55. Escenario para pruebas en simulador

El simulador fue montado utilizando una máquina virtual en la cual se aloja un sistema operativo Linux Ubuntu, se configura de la siguiente manera:

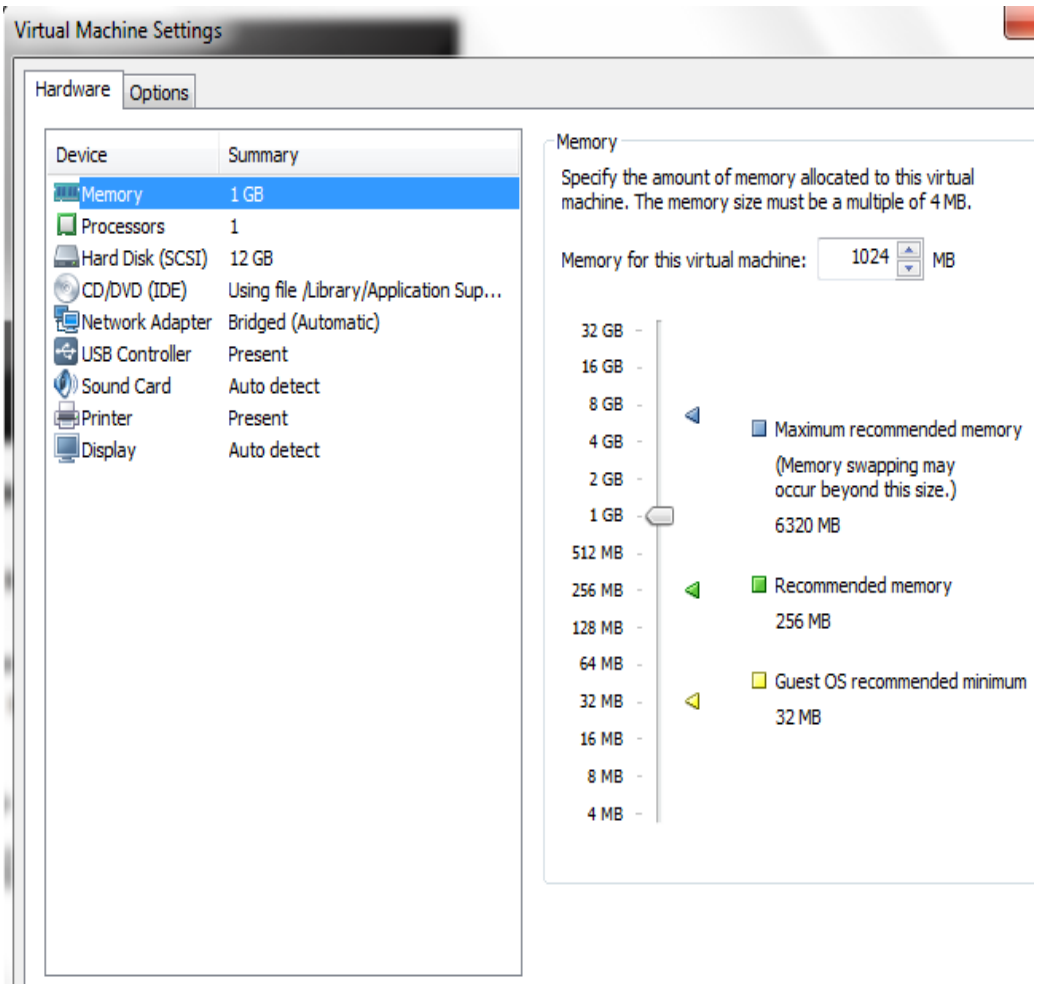

Figura 56. Configuración de la máquina virtual

La aplicación que se va a ejecutar requiere de la utilización del canal de retorno por tanto es necesario configurar los parámetros de red de la siguiente manera:

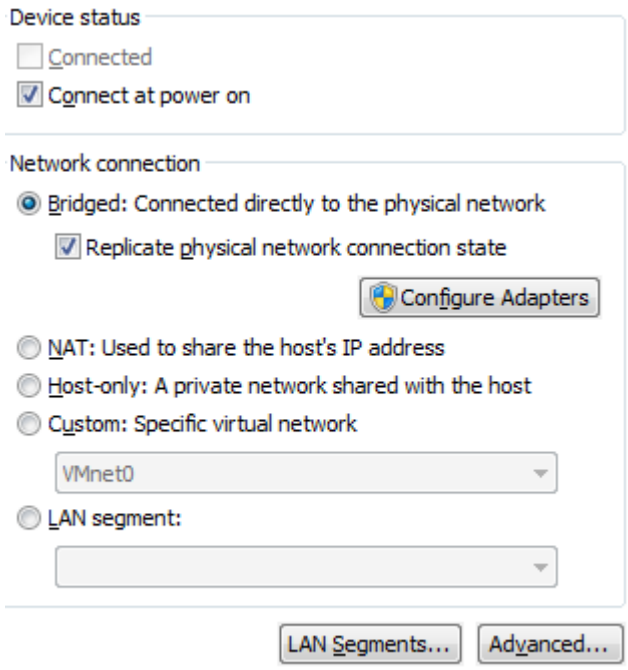

Figura 57. Configuración de Red en la máquina virtual

Con los parámetros establecidos la máquina virtual se conectará a la red configurada para la realización de las pruebas, el router le asignará una dirección ip en el segmento de red correspondiente mediante el protocolo DHCP. De la misma manera, el cliente multiplataforma, haciendo uso de su Smartphone se conecta vía wifi a la red configurada con la siguiente asignación de direcciones ip:

Tabla 5:

Tabla de Direcciones Ip para escenario simulado

Dirección Ip	Dispositivo
192.168.1.1	Router Tp-link
192.168.1.2	Servidor REST
192.168.1.3	Smarphone
192.168.1.4	Simulador Ginga

Al igual que en el escenario anterior el cliente multiplataforma ingresa a su perfil desde su smartphone y crea su cuenta para posteriormente enviar datos de consulta y de

sensor los cuales son visualizados desde el simulador accediendo a la cuenta con el pin proporcionado por el sistema:

Figura 58. Prueba de Simulación

Fuente Video: (Echeverría, 2013)

CAPÍTULO 5

ANÁLISIS DE RESULTADOS

5.1 Conclusiones

1. En el presente trabajo de investigación se presentó una arquitectura que permite incorporar dos plataformas distintas como es la móvil y la televisión digital bajo el estándar ISDBT-Tb, la integración de la arquitectura REST para servicios web permite la comunicación entre ambas tecnologías y presenta la posibilidad de una integración a mayor escala con otras plataformas existentes.
2. La principal innovación de la televisión digital es la incorporación de contenido interactivo lo cual permite al televidente un nuevo nivel en cuanto a experiencia de usuario, por tanto la plataforma ya no se limita únicamente a la presentación de contenido gráfico fijo, sino que el usuario puede acceder a mayor información de un tema de su interés convirtiendo en una experiencia personalizada.
3. La arquitectura propuesta considera los dos tipos de usuario de televisión digital, aquellos que poseen acceso a internet y aquellos que no, en consecuencia se presenta opciones de interactividad para ambos.

4. Una vez ejecutado el apagón analógico en Ecuador, será posible explotar todo el potencial que brinda el estándar ISDBT-Tb en cuando a aplicaciones interactivas en diversos campos, sin embargo uno de los riesgos que se pueden presentar es el estancamiento en el desarrollo de la televisión digital como ha ocurrido en otros países donde debido a la falta de integración con otras plataformas se ha relegado a un segundo plano, una de las soluciones se presenta en este trabajo de investigación que permite adherir a la televisión digital al internet de las cosas.
5. Para la prueba de la arquitectura se ha desarrollado una aplicación para teléfonos inteligentes, sin embargo como trabajo futuro se puede utilizar la misma arquitectura en distintas multiplataformas.
6. En la realización de las pruebas fue necesario la utilización de un cable HDMI para la conexión del codificador con la televisión puesto que con un cable RCA la visualización de la aplicación era sumamente borrosa y a veces incluso ininteligible para el usuario.
7. El desarrollo de la aplicación para Tv digital tiene que tomar en cuenta las restricciones de la plataforma tales como limitaciones en el hardware de procesamiento del televisor y la interfaz de usuario no debe ser invasiva permitiendo al usuario visualizar tanto la aplicación como el programa que se transmite.

5.2 Recomendaciones

1. Conocer la sintaxis de los lenguajes de programación necesarios para el desarrollo de aplicaciones interactivas de televisión digital.
2. Utilizar servidores con hardware sofisticado que permita abastecer las necesidades de las aplicaciones.
3. Vaciar el buffer de salida del servidor de playout en caso que la transmisión presente algún inconveniente

4. No utilizar demasiado contenido gráfico o de animaciones en las aplicaciones de GINGA puesto que existe la limitante de procesamiento en los televisores digitales.
5. La utilización de conexión de buena calidad certifican una baja pérdida y por tanto una satisfactoria recepción.

REFERENCIAS

Alejandro, A. A. (Julio de 2011). Diseño de aplicaciones interactivas t-government, t-health y t- learning para su aplicación en el sistema de televisión digital terrestre del ecuador (sbtvd) para la empresa televisión del pacífico s.a. Gamatv. Quito, Pichincha, Ecuador: Universidad Politécnica Salesiana.

Bussines, I. t. (2016). I2B. Recuperado el 2017, de I2B: http://www.i2btech.com/

Bustamante Ramírez, E., & Alvarez Mozoncillo, J. (2002). Comunicación y cultura en la era digital (1st ed.). Barcelona: Gedisa.

Canchi, S. D., González, M. H., & Mendel, R. H. (2013). Propuesta de una Arquitectura de Software para. Argentine Symposium on Software Engineering(14).

Echeverría, D. I. (Dirección). (2013). Programa tu salud Informa [Película].

EITV. (02 de Septiembre de 2011). Manual de usuario EITV DeveloperBox. Brasil.

INEC. (Diciembre de 2016). Acceso al Internet. Ecuador: Gobierno Nacional.

INEC. (Diciembre de 2016). Acceso al Internet. Acceso al Internet. Ecuador: Gobierno Nacional.

Loor, L. V. (2014). Arquitectura Manejada por modelos. San Gregorio(8). Recuperado el 2017

Martínez Martínez, R. F., & Narváez Morales, F. J. (2013). Implementación de un prototipo de plataforma de software para tdt con ginga-ncl-lua para desarrollar un mecanismo de teleformación sobre dispositivos de servicio móvil avanzado. 45. Riobamba, Ecuador.

Metro. (16 de Agosto de 2016). 2,8 millones de ecuatorianos usa las redes sociales. Metro.

Moreno Quinche, B. G., & Salazar Baculimba, J. C. (Noviembre de 2011). Estudio y anális de factibilidad para la implementación de la televisión digital terreste en Ecuador. Estudio y anális de factibilidad para la implementación de la televisión digital terreste en Ecuador. Cuenca, Ecuador: Universidad Politécnica Salesiana .

Paucar Curasma, H., Ugarte Vera, N., & Paucar Curasma, R. (2014). Aplicación Interactiva para registro de citas médicas por Televisión. III Congreso Internacional de Computación y Telecomunicaciones. Recuperado el 2017

Pelaez, J. C. (18 de abril de 2009). geeks. Recuperado el 2017, de geeks: www.geeks.ms

Peñerory Beltran, V. M. (2015). Modelo para el desarrollo de aplicaciones de televisión digital interactiva basado en el diseño centrado en el usuario. Santiago de Cali, Colombia: Universidad Autónoma de Occidente.

Pisciotta, O. N. (2014). Remultiplexor ISDB-Tb. Memoria Investigaciones en Ingeniería.

Pons, C., Giandini, R., & Pérez, G. (2010). Desarrollo de software dirigido por modelos. Buenos Aires: Editorial de la Universidad Nacional de la Plata.

Pressman, R., & Ojeda, M. R. (1998). Ingeniería del software (1st ed.). Aravaca, Madrid, España: McGraw-Hill.

Richardson, L., & Ruby, S. (2008). RESTful web services. O'Reilly Media, Inc.

Suarez, J. (17 de Abril de 2015). Usabilidad de internet se incrementa en el Ecuador, según el INEC. Ecuavisa.

Universidad Técnica Federico Santa María, U. U. (2014). Ginga y TVD. Recuperado el 2017, de http://www2.elo.utfsm.cl/~elo323/

Vallecillos, R. J. (16 de Marzo de 2016). COScore: una infraestructura de servicios para el despliegue de aplicaciones Mashup. COScore: una infraestructura de servicios para el despliegue de aplicaciones Mashup. Almería, España: Universidad de Almería.

wordpress. (2014). Ingsoftwarei. Recuperado el 2017, de https://ingsoftwarei2014.wordpress.com/tag/arquitectura-dirigida-por-modelos-mda/