



Universidad de las Ciencias Informáticas

Facultad 6

**DESARROLLO DE LA APLICACIÓN XILEMA AGORAV PARA
DISPOSITIVOS MÓVILES CON SISTEMA OPERATIVO *ANDROID*.**

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores:

Orlando Javier Vilaú Jiménez
Joice Osvaldo Fernández Rodríguez

Tutor:

Ing. Miguel Morciego Verona

La Habana, julio de 2016
“Año 58 de la Revolución”

Declaración de autoría

Declaramos ser autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a que haga uso del mismo en su beneficio, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Orlando Javier Vilaú Jiménez

Firma del Autor

Joice Osvaldo Fernández Rodríguez

Firma del Autor

Ing. Miguel Morciego Varona

Firma del Tutor

Datos de contacto

Tutor:

Ing. Miguel Morciego Varona

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: mmorciego@uci.cu

Agradecimientos

Orlando Javier Vílariú Jiménez

Agradecer a todas aquellas personas, familiares y amigos, que de una forma u otra me han ayudado y apoyado en mi paso por esta universidad.

Joíce Osvaldo Fernández Rodríguez

A mi mamá por estar siempre presente, por enseñarme a valorar a las personas, por corregirme cuando me equivoco y siempre alentarme a seguir adelante.

A mi papá por ser un ejemplo para mí, por guiarme solo como él sabe, por sacrificarse todos estos años para tratar de darme siempre lo mejor.

A mi tía por estar siempre pendiente de mí, por darme los mejores consejos, por su compañía.

A mi familia en general por experimentar junto a mí mis logros y derrotas.

Dedicatoria

Orlando Javier Vilcaú Jiménez

Por y para mis padres, gracias por todo el apoyo, por guiarme en la vida y querer para mí todo lo que ustedes no pudieron tener.

Joíce Osvaldo Fernández Rodríguez

*A mis padres por el sacrificio de tantos años, por poner mi bienestar ante todas las cosas y por esa educación que solo ustedes han sabido darme.
A mi tía Dunia por estar siempre a mi lado y quererme como su hijo, por apoyarme siempre y darme los mejores consejos.*

Resumen

Con el surgimiento de Internet y el acelerado avance de las tecnologías, la comunicación ha evolucionado de manera radical, siendo actualmente la red de redes uno de los medios de comunicación principal. Una de las formas de transmisión de información más usuales en la Internet son los contenidos audiovisuales, por tanto, los sitios o plataformas dedicados a la gestión de este tipo de contenido son ampliamente visitados por los usuarios. En el centro de Geoinformática y Señales Digitales se desarrolla la plataforma XILEMA AGORAV, encargada de la gestión y administración de materiales audiovisuales a través de la web. Esta plataforma, aunque posee un diseño adaptable para dispositivos móviles, puede proporcionar una experiencia de usuario poco agradable debido a la cantidad de contenidos mostrados al usuario. Como consecuencia, esta deficiencia puede devenir en un bajo rendimiento del dispositivo móvil al ser preciso un gran consumo de recursos de *hardware*. El presente trabajo tiene como objetivo desarrollar una aplicación para dispositivos con Sistema Operativo *Android* para mejorar la usabilidad y facilitar el consumo de los contenidos audiovisuales de la plataforma XILEMA AGORAV.

Palabras claves: *Android*, contenidos audiovisuales, dispositivos móviles, rendimiento, usabilidad, XILEMA AGORAV.

Abstract

With the emergence of Internet and the accelerated advance of technologies, the communication has evolved in a radical way, being nowadays one of the main means of communications. Audiovisual is one of the most usual ways of information transmission, therefore, sites and platforms dedicated to the management of this type of content are widely visited by users. In the Geoinformatics and Digital Signals center, the platform XILEMA AGORAV was developed with the purpose of managing audiovisual materials throughout the web. Accessing this platform through a mobile browser, even though it has an adaptable design for mobile devices, may leave an unpleasant user experience due to the density of contents to show. As a consequence, this deficiency may result in a low performance of the mobile device because of the consumption of *hardware* resources that is required. The present paper has as objective to develop an application for mobile devices with Android technology that provides better user experience and facilitate consumption of the audiovisuals contents from the platform XILEMA AGORAV.

Keywords: *Android*, audiovisual content, efficiency, mobile devices, usability, XILEMA AGORA.

Introducción	1
CAPÍTULO 1: Fundamentación teórica.	5
1.1. Elementos asociados al dominio del problema	5
1.1.1. Dispositivos móviles	5
1.1.2. Usabilidad	6
1.1.3. Experiencia de usuario	7
1.1.4. Material Design	7
1.2. Caracterización de la situación problemática	9
1.3. Soluciones existentes	11
1.4. Caracterización de las tecnologías, herramientas, y metodología	15
1.4.1. Tecnologías	16
1.4.2. Metodología de desarrollo de software	20
1.4.3. Lenguaje de modelado	21
1.4.4. Lenguaje de programación	21
1.4.5. Herramientas de desarrollo	22
1.5. Conclusiones parciales	23
CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución	25
2.1. Modelo de dominio	25
2.1.1. Descripción de los conceptos del modelo de dominio	25
2.2. Requisitos de <i>software</i>	26
2.2.1. Requisitos funcionales	26
2.2.2. Requisitos no funcionales	28
2.3. Casos de uso	29
2.3.1. Descripción de los actores del sistema	30

2.3.2.	Descripción de los casos de uso del sistema:	31
2.4.	Arquitectura de <i>software</i>	34
2.4.1.	Estilo arquitectónico	34
2.4.2.	Patrón arquitectónico	34
2.4.3.	Patrones de diseño	36
2.5.	Modelo de diseño	38
2.5.1.	Diagrama de clases del diseño	38
2.6.	Estándar de codificación	39
2.7.	Conclusiones parciales	41
CAPÍTULO 3: Implementación y validación de la solución		42
3.1.	Modelo de Implementación	42
3.1.1.	Diagrama de Componentes	42
3.2.	Modelo de despliegue	43
3.2.1.	Diagrama de despliegue	43
3.3.	Validación de la solución	44
3.3.1.	Pruebas funcionales	44
3.3.2.	Pruebas de rendimiento	52
3.3.3.	Pruebas de usabilidad	53
3.3.4.	Pruebas de integración	56
3.4.	Conclusiones parciales	57
Conclusiones generales		58
Recomendaciones		59
Referencias bibliográficas		60
Bibliografía		64
Anexos		68

Índice de figuras

Figura 1: Arquitectura de <i>Android OS</i>	17
Figura 2: Modelo de Dominio.	25
Figura 3: Diagrama de casos de uso del sistema.....	30
Figura 4: Diagrama orientado a dominio de la propuesta arquitectónica.	36
Figura 5: Diagrama de clases del diseño Reproducir Audiovisual.	39
Figura 6: Diagrama de componentes.	43
Figura 7: Diagrama de despliegue.	44
Figura 8: Gráfica de no conformidades de las pruebas de caja negra.....	50

Índice de tablas

Tabla 1: Características principales de las plataformas analizadas.....	14
Tabla 2: Descripción de los actores del sistema.	30
Tabla 3: Descripción del caso de uso Reproducir Audiovisual.	31
Tabla 4: Nomenclatura de los elementos para el estilo de codificación.....	40
Tabla 5: Caso de prueba del caso de uso Reproducir Audiovisual.....	46
Tabla 6: Promedio de los resultados de las pruebas de rendimiento.....	53

Introducción

Los contenidos audiovisuales actualmente constituyen una de las principales formas de transmisión de información en la sociedad, donde cada individuo puede identificar sus preferencias y disfrutar de ellas gracias a la variedad de contenidos disponibles. Estos poseen dos componentes: la información que contiene y el medio a través del cual se transmite (Ruano, 2010). A raíz del desarrollo tecnológico los medios de transmisión se han expandido, donde Internet ha llegado a jugar un papel fundamental para la distribución de los contenidos audiovisuales.

“Internet es un medio de comunicación que compite con los demás en cuanto a sus posibilidades de llegar a muchos lugares y a muchas personas en tiempo real” (Aguaded Gómez, 2010). Lo planteado anteriormente, unido a la distribución de audiovisuales ha traído consigo la proliferación de plataformas web que permiten la publicación y consumo (o visualización) de este tipo de contenido. Asimismo, con los nuevos avances en las tecnologías móviles y las redes inalámbricas se ha hecho posible ampliar el acceso a los materiales brindados en estas plataformas. Características como la portabilidad y el tamaño reducido de los dispositivos móviles, además de su conectividad, han hecho posible prescindir de un ordenador y permitir el consumo de contenidos publicados en la web desde casi cualquier lugar.

En la Universidad de las Ciencias Informáticas (UCI), específicamente en el centro de desarrollo Geoinformática y Señales Digitales (GEYSED), se desarrolla una solución para la administración y gestión de contenidos audiovisuales basada en tecnología web: XILEMA AGORAV. La misma tiene como objetivo general proveer a cualquier entidad con una red de datos una solución integral para la transmisión y publicación de contenido audiovisuales. Esta plataforma permite la gestión, publicación y transmisión de contenido audiovisual sobre la red de datos, a través de la web mediante la tecnología *streaming*¹. La plataforma puede ser adaptable a las necesidades de cualquier cliente, ya sea para transmitir en una red interna o vía Internet. (Plataforma VideoWeb, televisión en la web, 2013).

XILEMA AGORAV cuenta con diferentes funcionalidades agrupadas en seis módulos según sus características. Uno de estos es: Presentación de Contenidos, que permite desde una interfaz de usuario visualizar los contenidos audiovisuales, conformar listas de reproducción a partir de las publicaciones, compartir las listas con otros usuarios del sistema, comentar las publicaciones y efectuar un voto referente a la misma.

¹ Tecnología que se utiliza para mejorar la descarga y reproducción de archivos de audio y video.

Introducción

Con el actual auge de las tecnologías móviles, cada día es más frecuente el consumo de contenidos audiovisuales desde terminales móviles a través de un navegador web. No obstante, acceder desde este tipo de dispositivos, aunque el sistema cuente con un diseño adaptable, puede dificultar el acceso y la usabilidad en dicho sistema debido a la cantidad de contenidos de valor a mostrar. Puede resultar complejo para el usuario la navegación por la plataforma, pues en ocasiones es necesario acercarse a la interfaz del sitio web para leer o acceder a un elemento, o utilizar una barra de desplazamiento debido a la densidad de contenido. Como consecuencia de esto, la usabilidad queda reducida ya que el usuario en cuestión puede demorarse en la búsqueda de un elemento o resultar complejo el acceso a este. Otro inconveniente es el consumo excesivo de recursos del dispositivo al acceder a la plataforma XILEMA AGORAV desde un navegador web. Debido a ello el usuario puede experimentar calentamiento en el dispositivo, retardo en el tiempo de respuesta o demora al cambiar la aplicación del navegador web a ejecución en segundo plano y viceversa.

Partiendo de la situación planteada se deriva el siguiente **problema de investigación**: ¿Cómo facilitar el consumo de los materiales audiovisuales de la plataforma XILEMA AGORAV desde un dispositivo móvil con sistema operativo *Android*?

Se define como **objetivo general**, desarrollar una aplicación que facilite el consumo de contenidos audiovisuales de la plataforma XILEMA AGORAV desde un dispositivo móvil con sistema operativo *Android*.

La investigación se centra en el proceso de consumo de contenidos audiovisuales sobre dispositivos móviles, lo cual constituye el **objeto de estudio**. Acotando **el campo de acción** al proceso de consumo de los contenidos audiovisuales de la plataforma XILEMA AGORAV desde dispositivos móviles con sistema operativo *Android*.

Para cumplir el objetivo general de la investigación se trazaron las siguientes **preguntas de investigación**:

- ¿Cuáles son los fundamentos teóricos que rigen el desarrollo de plataformas compatibles con dispositivos móviles que permitan el consumo de contenidos audiovisuales?
- ¿Cómo ha evolucionado el desarrollo de plataformas compatibles con dispositivos móviles que permitan el consumo de contenidos audiovisuales?
- ¿Qué métodos y herramientas hay que tener en cuenta para el desarrollo de plataformas compatibles con dispositivos móviles que permitan el consumo de contenidos audiovisuales?

Introducción

Para dar cumplimiento a las preguntas formuladas anteriormente se hace necesario desarrollar las siguientes **tareas de investigación**:

1. Fundamentación de los métodos factibles para el consumo de los contenidos audiovisuales de la plataforma XILEMA AGORAV desde dispositivos móviles con sistema operativo *Android*.
2. Determinación de las herramientas adecuadas para el desarrollo la aplicación AGORAV para dispositivos móviles con sistema operativo *Android*.
3. Elaboración de la documentación necesaria según la metodología de desarrollo seleccionada.
4. Implementación de las funcionalidades que garanticen el correcto consumo de los contenidos audiovisuales publicados en la plataforma XILEMA AGORAV.
5. Validación de la solución mediante la aplicación de pruebas funcionales, de rendimiento, de usabilidad y de integración.

Una vez definidas las tareas de investigación se determinó como **idea a defender** el desarrollo de una aplicación nativa que use servicios web para el consumo de contenidos audiovisuales publicados en la plataforma XILEMA AGORAV desde un dispositivo móvil con sistema operativo *Android*, mejora la usabilidad y rendimiento en comparación con su versión web móvil.

Se plantea como **resultado esperado** que la aplicación nativa *Android* para la plataforma XILEMA AGORAV mejora la usabilidad y el rendimiento de los dispositivos móviles en comparación con su equivalente para la web.

Métodos Teóricos:

- **Método histórico – lógico:** Este método se utilizó para analizar la trayectoria y evolución histórica de plataformas compatibles con dispositivos móviles que permitan el consumo de contenidos audiovisuales, así como establecer las tendencias generales que rigen el funcionamiento y desarrollo de las mismas.
- **Método análisis – sintético:** Este método se utilizó con el objetivo de descomponer el proceso de consumo de contenidos audiovisuales desde dispositivos móviles, procesar los fundamentos y las

Introducción

teorías relacionadas a este proceso, lo que permite extraer los aspectos significativos que sustentan la propuesta de solución.

- **Observación científica:** Este método se utilizó con el objetivo de comprender el comportamiento de los usuarios al interactuar con la solución, lo que permite mejorar la solución en función a dicho comportamiento.

Técnicas de investigación científica:

- **Cuestionario:** Esta técnica se utilizó con el objetivo de documentar las opiniones de los usuarios al interactuar con la solución, lo que permitió realizar un análisis del nivel de aceptación de la misma.

El presente trabajo de diploma está estructurado en 3 capítulos:

Capítulo 1: “Fundamentación teórica”: En este capítulo se brinda una visión general de los aspectos teóricos relacionados con la gestión de contenidos audiovisuales. Se definen las herramientas, lenguaje de programación, así como la metodología de desarrollo a utilizar.

Capítulo 2: “Levantamiento de requisitos y diseño de la solución”: En este capítulo se expone el modelo de dominio para una mejor comprensión de los conceptos del problema. Se definen los requisitos funcionales y no funcionales, se generan artefactos como el diagrama de caso de uso y la descripción de los casos de uso. Además, se define la arquitectura del sistema y presentan los estándares de codificación.

Capítulo 3: “Implementación y validación de la solución”: En este capítulo se detallan los elementos relacionados con la implementación de la solución, para dar cumplimiento a las funcionalidades propuestas en el capítulo anterior. Se detalla la estrategia de prueba a seguir para validar la calidad de la aplicación AGORAV.

CAPÍTULO 1: *Fundamentación teórica*

CAPÍTULO 1: Fundamentación teórica.

En este capítulo se brinda una visión general de los aspectos teóricos relacionados con la gestión de publicación de contenidos audiovisuales y los conceptos básicos asociados al dominio del problema de investigación. Además se realiza un estudio crítico de las principales tendencias, técnicas y tecnologías usadas en la actualidad para la gestión de publicación de contenidos audiovisuales, así como una valoración de las más factibles para dar solución al problema en cuestión.

1.1. Elementos asociados al dominio del problema

Al desarrollar la solución es necesario tener presente conceptos y definiciones asociados al proceso de consumo de contenidos audiovisuales desde dispositivos móviles, para lograr una mayor comprensión de los problemas que afectan a la plataforma XILEMA AGORAV. A continuación, se abordan los conceptos asociados a la investigación.

1.1.1. *Dispositivos móviles*

Los dispositivos móviles se definen como un grupo de dispositivos electrónicos que poseen las siguientes características que los diferencian de otros dispositivos: movilidad, tamaño reducido, comunicación inalámbrica e interacción con las personas. Algunos tipos de dispositivos móviles son los *smartphones* (en español teléfonos inteligentes), *tablets* (en español tabletas), *e-reader* (en español lector de libros electrónicos) y *PDA*s (en español asistentes personales digitales). A continuación, se describen brevemente las características mencionadas anteriormente (Morillo Pozo, 2011):

- La movilidad es la cualidad de un dispositivo para ser transportado o movido con frecuencia y facilidad.
- El tamaño reducido es la cualidad de un dispositivo móvil de ser fácilmente usado con una o dos manos sin necesidad de ayuda o soporte externo.
- La comunicación inalámbrica es la capacidad que tiene un dispositivo de enviar o recibir datos sin la necesidad de un enlace cableado.
- La interacción es el proceso de uso que establece un usuario con un dispositivo. Un factor que incide en esta interacción es la usabilidad.

CAPÍTULO 1: Fundamentación teórica

En virtud de su creciente capacidad de conectividad, procesamiento y transmisión de datos, el dispositivo móvil se ha convertido en un medio primordial de consumo debido a su condición de meta-dispositivo tecnológico (algunos incluyen componentes como cámara, agenda, TV, videoconsola, reproductor de audio/video, GPS...), además de sus rasgos de objeto cultural fuertemente ligado a la identidad del usuario y como un medio para la producción, distribución y consumo de datos y contenidos (Aguado Terrón, y otros, 2009).

Es necesario comprender que la posibilidad misma del acceso a Internet desde un dispositivo móvil hizo evidente la necesidad de adaptar los estándares de la WWW² a las características del dispositivo de bolsillo: menor tamaño y resolución de pantalla, menor capacidad de procesamiento de datos, configuración vertical, problemas específicos de usabilidad en cuanto a entrada de datos y presentación de contenidos. Como consecuencia de estas necesidades surge un nuevo modelo de desarrollo centrado en el usuario, enfocado principalmente a la experiencia del mismo, donde el contenido no es el objetivo esencial (Aguado Terrón, y otros, 2009).

Mediante este nuevo paradigma de desarrollo se comenzaron a tener en cuenta principios de usabilidad que antes no eran tan importantes. Otro concepto fuertemente ligado a este paradigma es la accesibilidad, donde se valoran factores como que la información y los servicios deben ser accesibles para todos, ya sean débiles visuales, personas de la tercera edad o con deficiencias motrices; además, que la presentación contenidos sea clara y simple, al igual que los mecanismos de navegación.

1.1.2. Usabilidad

Según la norma ISO 9241-11 la usabilidad se define como: *“el grado en que un producto puede ser utilizado por usuarios específicos para lograr los objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso”*.

La usabilidad es parte de la Interacción Humano-Computadora (del inglés HCI / *Human-Computer Interaction*), rama dentro de las ciencias de la computación que estudia todos los aspectos relacionados con la interacción entre el ser humano y las máquinas. Incluye el análisis de los aspectos sociales, físicos, emocionales, ergonómicos, entre otros. Además se basa en aspectos evaluativos relacionados con las interfaces desde donde el usuario pueda desenvolverse fácilmente, le sea sencillo encontrar lo que busca

² *World Wide Web*.

CAPÍTULO 1: *Fundamentación teórica*

dentro del sitio, cometa la menor cantidad de errores y sea capaz de realizar un manejo eficiente de los contenidos. Todos estos aspectos se deben cumplir para lograr la satisfacción por parte del usuario al utilizar un sistema (Evaluación de "Web Site" utilizando principios de Ingeniería de Usabilidad, 2015).

Se define también por usabilidad la capacidad de respuesta y consumo de recursos de una aplicación o programa. Observándose en el sobrecalentamiento del dispositivo, retardo en las respuestas o colapso del sistema; traducidos como impaciencia, enojo o insatisfacción en el usuario.

1.1.3. Experiencia de usuario

El diseño de la experiencia del usuario como disciplina se ocupa de todos los elementos que componen la interfaz, incluyendo el diseño visual, el texto, el sonido y la interacción. También abarca cada aspecto de la interacción del usuario con el producto, servicio o empresa. La experiencia de usuario trabaja para coordinar estos elementos y permitir la mejor interacción posible por los usuarios (User Experience Professionals Association, 2014).

Son muchos los autores que tratan este tema y definen un concepto de experiencia de usuario, cada uno con sus variantes y puntos de vistas diferentes. Sin embargo la esencia sigue siendo la misma descrita como la sensación, sentimiento, respuesta emocional, valoración y satisfacción del usuario respecto a un producto, resultado del fenómeno de interacción con el producto y la interacción con su proveedor (Hassan Montero, y otros, 2005).

La experiencia de usuario es un concepto que se debe tener en cuenta al diseñar cualquier plataforma o aplicación, pues constituye un punto decisivo para determinar si el proyecto tendrá éxito o no. Es decir, si un usuario no se siente a gusto cuando interactúa con un producto, es muy probable que no lo vuelva a usar o lo use menos. Actualmente con la competencia existente en el mercado mundial es necesario hacer sentir a gusto al usuario, para mantener la fidelidad de los mismos.

1.1.4. Material Design

Según Google en la conferencia Google I/O del 2014 donde fue presentado el concepto de *Material Design* lo define como: “un concepto, una filosofía, unas pautas enfocadas al diseño utilizado en Android, pero también en la web y en cualquier plataforma”. Se trata de un diseño limpio, en el que predominan animaciones y transiciones de respuesta; el relleno y los efectos de profundidad tales como la iluminación

CAPÍTULO 1: *Fundamentación teórica*

y las sombras. *Material Design* posee superficies físicas y bordes donde las escenas y sombras proporcionan significado sobre lo que se puede tocar y cómo se va a mover (Google, 2016).

Material Design es una guía completa para el diseño, animación e interacción de las interfaces visuales para todas las plataformas y dispositivos. Está basado en objetos materiales, piezas colocadas en un lugar y con un movimiento determinado, donde la profundidad, las superficies, los bordes, las sombras y los colores juegan un papel principal. Esta guiado por las leyes de la física, donde las movimientos son lógicos, los objetos se superponen pero no pueden atravesarse el uno al otro, los menús poseen una tipografía clara, casillas bien ordenadas, colores e imágenes llamativos para no perder el foco y dar un sentido de orden y jerarquía (Google, 2016).

Material Design posee tres principios básicos explicados a continuación (Google, 2016):

- Enfocado a un espacio racionalizado y a un sistema de movimiento, simulando una realidad táctil. Donde las superficies y los bordes proporcionan indicadores visuales que se basan en la realidad; apoyado con el uso de atributos táctiles familiares que ayuda a los usuarios a entender rápidamente el comportamiento del sistema. El uso de efectos de luz, de superficie, y movimiento son la clave para transmitir el sentido de los objetos
- Los elementos de diseño como la tipografía, el espacio, la escala, el color y las imágenes, son la guía para el tratamiento visual. El correcto uso de estos elementos hacen más agradable la vista, pues definen la jerarquía y el significado de los mismos.
- Todas las acciones se realizan en un único entorno mediante el movimiento de los elementos, donde los objetos se presentan al usuario sin romper la continuidad, a medida que se transforman y se reorganizan. Las transiciones deben ser correctas y coherentes.

El concepto de *Material Design* es una estrategia de Google para ganar en uniformidad en las aplicaciones, donde el usuario se sienta cómodo, sea posible una navegación sencilla e intuitiva, además de respetar las reglas de usabilidad. Este estándar está enfocado a la experiencia de usuario, por lo que muchas aplicaciones y plataformas han modificado sus interfaces teniendo en cuenta estos principios definidos.

CAPÍTULO 1: *Fundamentación teórica*

1.2. Caracterización de la situación problemática

Actualmente el centro de desarrollo GEYSED cuenta con varios proyectos productivos entre los que se encuentra AGORAV. Este trabaja en una plataforma con el mismo nombre para la gestión de contenidos audiovisuales en la Web. Entre las funcionalidades que brinda la plataforma al usuario se encuentra la reproducción de videos, búsqueda por nombre, descarga de audiovisuales y una interacción social, ya sea comentar o realizar una votación a un contenido publicado.

Se puede acceder a AGORAV desde cualquier equipo que contenga un navegador web y que posea una conexión a la red donde se encuentra desplegada la plataforma, ya sea una PC (escritorio o laptop) o a través de un dispositivo móvil. Mediante una prueba basada en el consumo de las funcionalidades de la plataforma entre los diferentes tipos de equipos se evidenció una diferencia con respecto a la navegabilidad. Mientras que en las PC es sencillo navegar e interactuar con las distintas secciones o botones presentes en la plataforma, en los dispositivos móviles no lo es tanto; pues aunque la plataforma cuenta con un diseño adaptable, se complejiza la navegación, teniendo que acercarse para poder leer las secciones o acertar al seleccionar un botón.

La navegabilidad es la diferencia más evidente al comparar entre los dos equipos, pero una vez adentrados en el problema de los dispositivos móviles se pueden apreciar otras características en estos, donde los de menores prestaciones son los más afectados. Al acceder a la plataforma desde un navegador web móvil, dichos dispositivos pueden responder de manera inesperada debido al gran consumo de recursos, lo que ocasiona:

- Retardo en las respuestas.
- Inestabilidad en el sistema, evidente al cambiar entre una aplicación y otra.
- Calentamiento en el dispositivo.

Existen diferentes soluciones para optimizar las capacidades de los diferentes dispositivos. La forma de acceder a las plataformas web puede variar en dos formas: *native apps* (en español aplicaciones nativas) o *web apps* (en español aplicaciones web), donde cada una de estas tiene sus ventajas y sus desventajas.

Aplicación Web Móvil

CAPÍTULO 1: *Fundamentación teórica*

Una aplicación web móvil es una aplicación web con formato para dispositivos móviles y se accede a esta a través de un navegador web móvil. Como una aplicación web tradicional, la aplicación web móvil se construye con tres tecnologías: HTML, CSS y JavaScript (Lionbridge, 2012).

Ventajas:

- La principal ventaja de las aplicaciones web móviles con respecto a las aplicaciones móviles nativas es la compatibilidad entre plataformas, lo que les permite llegar a un público más amplio por el menor esfuerzo. Son relativamente baratas, fáciles y rápidas de construir, aunque en algunos dispositivos específicos por lo general requieren personalización. Los navegadores web para móviles son bastantes estándares, por lo que es mucho más fácil crear una aplicación web móvil universal que una aplicación nativa.
- Las aplicaciones web son más fácil de mantener que las aplicaciones nativas.
- Acceso simple: los usuarios no tienen que descargar una aplicación, sino simplemente acceder a una URL a través de su navegador móvil.

Desventajas:

- Los navegadores móviles tienen capacidades limitadas en comparación con los navegadores de escritorio tradicionales. Muchas veces la aplicación puede parecer torpe en comparación con la de un navegador web de escritorio.
- Las aplicaciones web no pueden tener acceso al *hardware* y el *software* del dispositivo móvil como son la cámara, el GPS o uso de gráficos personalizados pesados.
- Las aplicaciones web requieren de una conexión lo que puede ocasionar problemas de rendimiento si el sitio es lento o no está disponible.

Aplicación Nativa

Una aplicación móvil nativa es una aplicación construida para los dispositivos móviles con un sistema operativo específico. Las aplicaciones nativas tienen una gran ventaja sobre las aplicaciones web: la capacidad de aprovechar el *hardware* y el *software* específico del dispositivo (Lionbridge, 2012).

CAPÍTULO 1: *Fundamentación teórica*

Ventajas:

- Una experiencia de usuario más convincente: Las aplicaciones nativas pueden aprovechar las capacidades del dispositivo móvil, incluyendo el *hardware* como el GPS, la cámara y gráficos, y el *software* como el correo electrónico, calendario, contactos, galería de fotos / video, administradores de archivos.
- Capacidad de ejecutarse *off-line*: Dado que la aplicación está instalada en el dispositivo, no se requiere conexión a Internet. Los usuarios pueden obtener el máximo rendimiento en todo momento, con todos los gráficos, imágenes, secuencias de comandos y datos. En caso de pérdida de conexión, la transferencia de datos se puede reanudar cuando se restablece la misma.
- En caso de querer conseguir ganancias económicas con las aplicaciones nativas es mucho más fácil, pues se obtienen mediante la compra de la misma en una tienda de aplicaciones, mientras que las aplicaciones web garantizan esto mediante una suscripción.

Desventajas:

- Para las aplicaciones nativas puede resultar difícil el acceso desde distintas plataformas, ya que presentan requisitos predefinidos para su uso.
- Si el sistema sufre modificaciones, es más costoso realizar cambios a la aplicación nativa para adaptarla a las nuevas funcionalidades.

A raíz del análisis de las diferentes formas de dar solución al problema de investigación, se llegó a la conclusión que sería más factible para ello la implementación de una aplicación nativa. Pues esta aprovecha mejor las capacidades presentes en los dispositivos móviles. Asimismo, en comparación con las aplicaciones web presentan un menor consumo de recursos, además de poder realizar un diseño flexible a las interfaces de usuario.

1.3. Soluciones existentes

En la actualidad existen disímiles sitios que brindan servicios de hospedaje de contenidos audiovisuales, en los cuales los usuarios pueden interactuar con los contenidos expuestos. Entre ellos podemos encontrar: www.youtube.com, www.dailymotion.com, www.vimeo.com y www.hulu.com como principales. El acceso

CAPÍTULO 1: *Fundamentación teórica*

por parte de los usuarios a estos sistemas desde dispositivos móviles ha ido en aumento a partir de los avances tecnológicos en este campo. Por tanto, estas plataformas han optado por el desarrollo de aplicaciones móviles nativas que brinden una mejor usabilidad en cuanto a la navegación y rendimiento en cuanto al consumo de recursos del dispositivo.

Youtube:

Youtube es un sitio web en el cual los usuarios pueden subir y compartir videos. Aloja una variedad de clips de películas, programas de televisión, videos musicales, también se pueden encontrar filmaciones caseras como *video blogs*. Fue creado por tres antiguos empleados de PayPal en febrero de 2005. En octubre de 2006, fue adquirido por Google Inc. a cambio de 1650 millones de dólares y ahora opera como una de sus filiales. Existe una ampliación de la plataforma para dispositivos móviles con sistema operativo *Android* disponible a partir de la versión 2.2 del mismo. Esta presenta una interfaz amigable y cómoda para sus usuarios (Google Play, 2016). Entre las características que presenta esta aplicación se destacan algunas como:

- Explorar recomendaciones personales en una sección llamada Página Principal
- Ver lo más reciente de los canales favoritos de los usuarios en la sección Suscripciones.
- Buscar los videos reproducidos recientemente y que se le asignaron el estado “Me gusta”
- Permite una interacción y retroalimentación con otros usuarios mediante los comentarios y los elementos compartidos.

Dailymotion:

Dailymotion es una plataforma propiedad del operador Orange, posee una aplicación para los dispositivos que usan sistema operativo *Android*. Compatible con todos los dispositivos que dispongan de la versión del sistema operativo 2.1 o superior. Como característica principal permite organizar los videos por categorías o tipologías para cuando se necesite buscar contenidos por determinados criterios (Google Play, 2016). Entre sus funcionalidades se pueden encontrar:

- Sugerencias personalizadas para descubrir contenidos.
- Compartir los videos favoritos y seguir los canales preferidos.

CAPÍTULO 1: *Fundamentación teórica*

- Reproducir videos en alta calidad.
- Buscar por palabras claves o explora por categorías
- Emitir eventos en directo o sincronizar videos para verlos más tarde.

Vimeo:

Vimeo es una red social de Internet basada en videos, lanzada en noviembre de 2004 por la compañía InterActiveCorp (IAC). Este sitio también ha implementado su propia aplicación para la plataforma *Android*, compatible con versiones 2.3 o superior. Como sistema presenta ventajas como la limpieza y claridad de su interfaz principal que hace posible desplazarse muy bien por toda la aplicación, funcionalidades muy claras, la aplicación es muy intuitiva (Google Play, 2016). Entre sus funcionalidades se pueden encontrar:

- Ver y administrar todos los videos existentes en Vimeo y descargarlos en la galería.
- Búsqueda rápida de los videos en vimeo.com.
- Cargar fácilmente el material sin editar en HD.
- Pausar y reanudar las subidas de videos.
- Reemplazar los videos existentes.
- Editar título, descripción, etiquetas, privacidad y créditos de los videos.
- Compartir con Facebook, Twitter, correo electrónico o SMS.
- Añadir clips a grupos, canales y álbumes.
- Estadísticas en gustos y comentarios.

Hulu:

Sitio web que ofrece gratis, programas de televisión y películas de NBC, FOX, y otras cadenas televisivas mediante la tecnología *streaming*. Actualmente sólo se ofrece a los usuarios de los Estados Unidos. Hulu ofrece videos en formato Flash Video en una resolución más alta que YouTube, pero inferior a la definición estándar de televisión. Hulu también ofrece servicios web de otros sitios web, incluido AOL, MSN, MySpace

CAPÍTULO 1: *Fundamentación teórica*

o Yahoo!. Cuenta con una aplicación para dispositivos con sistema operativo *Android* compatible con las versiones 4.0.3 o superiores. Una peculiaridad de esta aplicación es ofrecer publicidad adaptada a los intereses del usuario (Google Play, 2016).

Internos:

Plataforma web para la publicación de contenidos audiovisuales, actualmente desplegada en la Universidad de las Ciencias Informáticas (UCI). Esta plataforma cuenta con una aplicación nativa compatible con las versiones 4.0 o superior del sistema operativo *Android*. En comparación con el acceso por el navegador web móvil a la plataforma, la aplicación posee una mejor presentación de los contenidos y una mejor usabilidad. Entre las características que presenta esta aplicación se destacan algunas como:

- Reproducir audiovisuales mediante un reproductor propio de la aplicación, sin necesidad de códec³ de video externos.
- Descargar contenidos audiovisuales al dispositivo.
- Correcta presentación de los contenidos.
- Inexistencia de interacción con el usuario como: votar o comentar.
- Opciones de personalización, como: color de la aplicación, especificación del directorio de descarga o la sección de inicio por defecto.

A continuación, se muestra en la Tabla 1 un resumen comparativo entre las principales características de las plataformas analizadas:

Tabla 1: Características principales de las plataformas analizadas.

Plataforma	Usabilidad	Rendimiento dispositivo	Experiencia de usuario	Integración con XILEMA AGORAV
Youtube	X	X	X	
Dailymotion	X	X	X	
Vimeo	X	X	X	

³ Programa o dispositivo *hardware* capaz de codificar o decodificar una señal o flujo de datos digitales.

CAPÍTULO 1: *Fundamentación teórica*

Hulu	X	X	X	
Internos	X		X	

Las aplicaciones nativas *Android* de las plataformas propuestas como posibles soluciones, se han desechado como soluciones, debido a que están diseñadas específicamente para el modelo de negocio de sus plataformas, por lo que ninguna de estas es posible integrarlas a la plataforma XILEMA AGORAV.

A la aplicación Internos desarrollada para la plataforma del mismo nombre publicada en la Intranet de la UCI, se le realizó un análisis más detallado que a las demás soluciones existentes. Fue descartada como posible solución, pues está basada en la técnica *web scraping* para extraer la información del sitio web de la plataforma. Esta línea de trabajo mejora la usabilidad, pues permite un mejor manejo de la presentación de contenidos, pero mantiene el rendimiento del dispositivo igual que al acceder por el navegador web, ya que es necesario realizar una lectura y carga completa del sitio web. Además, el uso de la técnica *web scraping* presenta otros inconvenientes: si es modificado en la fuente de origen el nombre de cualquier etiqueta, o de cualquier sección en la estructura HTML del sitio web, la aplicación dejaría de funcionar, ya que no recibiría el contenido esperado. No obstante esta solución aportó ideas tales como el reproductor interno sin necesidad de códec externos y las opciones de personalización.

Las demás posibles soluciones aportaron ideas que sirvieron de modelos para la creación de la nueva solución. De la aplicación Youtube se analizó el diseño de las interfaces de usuario, tomándose la idea de mostrar las publicaciones en forma de listas, agregando la opción de reproducir y descargar sin necesidad de abrir la especificación de la publicación. El uso de *tabstrips*⁴ para navegar entre los diferentes contenidos fue una de las características que se tomó de la aplicación Dailymotion. También se analizaron las características presentes en el diseño de las interfaces de Vimeo y Hulu, tomándose algunas como referencias. Destacar que en las aplicaciones nativas de estas plataformas se evidencian mejoras en la usabilidad y en el rendimiento del dispositivo, además de una mayor aceptación por parte de los usuarios.

1.4. Caracterización de las tecnologías, herramientas, y metodología

Desarrollar un *software* implica que sea necesario definir las tecnologías, herramientas y metodología de desarrollo que se van a emplear para lograr el objetivo planteado. A continuación, se exponen las

⁴ Organización de los contenidos en pestañas, las cuales agrupan un conjunto de componentes en su interior.

CAPÍTULO 1: *Fundamentación teórica*

herramientas a utilizar, la metodología de desarrollo seleccionada y un análisis de las tecnologías actuales para el consumo de contenidos desde dispositivos móviles con sistema operativo *Android*.

1.4.1. *Tecnologías*

A continuación, se describen las tecnologías seleccionadas que inciden en el consumo de contenidos audiovisuales, mostrando una caracterización y análisis de las mismas.

Sistema operativo *Android*

Google adquirió la compañía *Android Inc.* y desarrolló la plataforma *Android*. Posteriormente en acuerdo con el consorcio *Open Handset Alliance* decide promocionar los estándares de código abierto para dispositivos móviles. Como característica distintiva del SO *Android* se tiene que su núcleo está basado en el Kernel⁵ de Linux (*Android OS*, 2012).

Entre las características notables se tiene que la plataforma es adaptable a pantallas más grandes, VGA, biblioteca de gráficos 2D y 3D basada en las especificaciones de OpenGL; utiliza SQLite para el almacenamiento de datos; emplea SMS y MMS como vías de mensajería; soporta múltiples tecnologías de conectividad: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE y WiMAX; maneja varios formatos multimedia: AMR, MP3, MIDI, WAV, JPEG, PNG, GIF, BMP, entre otros; incluye además soporte para *hardware* adicional: cámaras de fotos, de video, pantallas táctiles, GPS (del inglés *Global Positioning System*), acelerómetros, giroscopios, magnetómetros, termómetros, sensores de proximidad y de presión (Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con *Android*, 2012). La Figura 1 muestra los componentes principales de la arquitectura *Android*:

⁵ *Software* que constituye una parte fundamental del sistema operativo, encargado de gestionar recursos, a través de servicios de llamada al Sistema.

CAPÍTULO 1: Fundamentación teórica

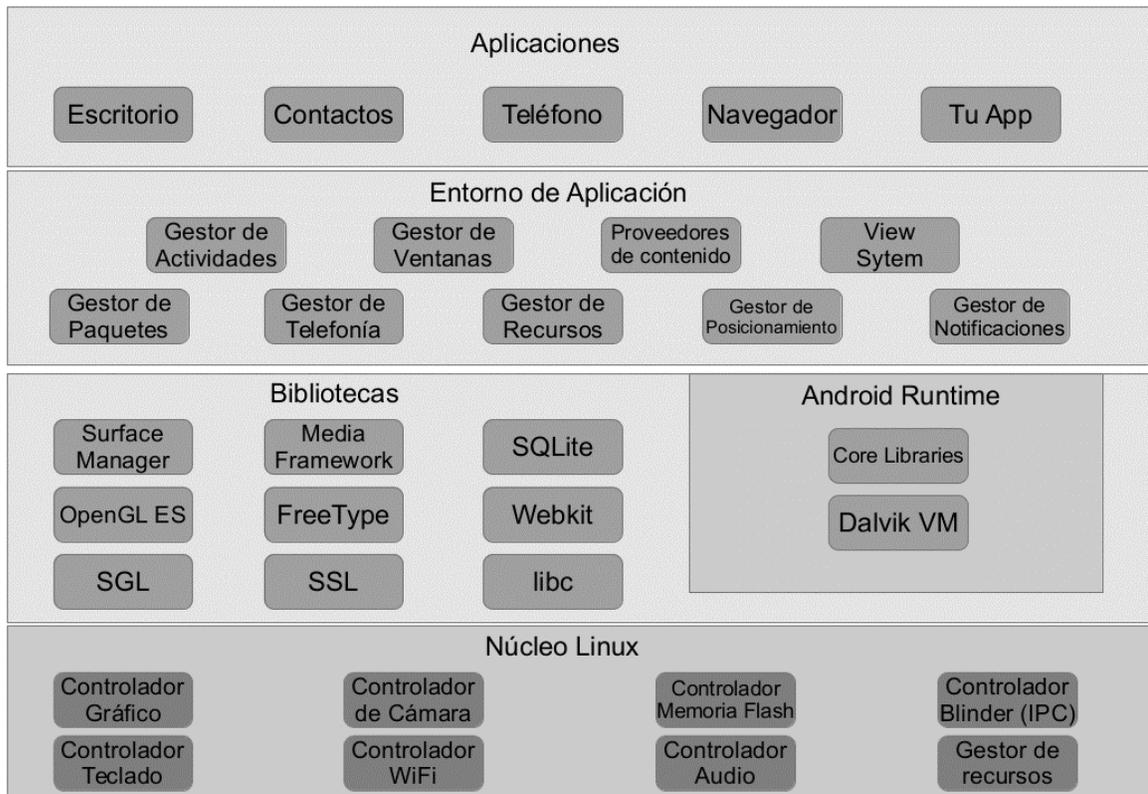


Figura 1: Arquitectura de *Android* OS.

Entre los principales fabricantes de dispositivos móviles con SO *Android* se encuentran: Dell, HTC, Kyocera, LG, Motorola, y Samsung.

Componentes básicos de una aplicación *Android*

Hay cinco tipos de componentes en una aplicación *Android*. Cada uno de ellos tiene un propósito y un ciclo de vida distinto que define cómo se crea y se destruye el componente (Pavón Pulido, 2013).

- **Activity:** Una actividad o *activity* permite crear componentes que interactúan con el usuario. Cada actividad tiene asociada una vista o *view*. La vista constituye el conjunto de elementos gráficos que se presentan al usuario para que interactúe con el sistema. Por ejemplo, cajas de edición, etiquetas o botones, entre otros. Para crear actividades personalizadas es necesario definir una clase que herede de la clase *Activity*.

CAPÍTULO 1: *Fundamentación teórica*

- **Intent:** Una intención o *intent* permite el envío de mensajes entre componentes. El paso de mensajes a través de *Intents* puede considerarse una facilidad para asociar (*binding*) componentes de forma tardía y en tiempo de ejecución en la misma aplicación o entre diferentes aplicaciones.
- **Service:** Un servicio o *service* permite llevar a cabo operaciones en segundo plano o *background* y, por tanto, no proporcionan al usuario una interfaz de usuario.
- **Broadcast Receiver:** Los receptores de notificaciones o *broadcast receivers* permiten que se reciban notificaciones, desde ciertos componentes. El componente que envía la notificación lo hace a través de un *sendbroadcast*. Los componentes que la reciben se asocian o suscriben a un *Receiver*.
- **Content Provider:** Los proveedores de contenido o *content providers* proporcionan facilidades de acceso a datos estructurados. Además de encapsular los datos, proporcionan mecanismos para definir la seguridad de los mismos.

Servicios web

Un servicio web se define como un sistema de *software* diseñado para permitir interoperabilidad máquina a máquina en una red. En general, los servicios web son sólo APIs⁶ (del inglés *Application Programming Interface*) Web que pueden ser accedidas en una red, como Internet, y ejecutadas en un sistema de alojamiento remoto (Rao, 2004).

Ventajas de los servicios web

- Aumentan la interoperabilidad entre programas independientemente de la plataforma en donde están instalados.
- Aumentan la interoperabilidad entre servicios y programas de diferentes compañías y ubicados en diferentes lugares geográficos.
- Fomentan los estándares y protocolos basados en texto, haciendo más fácil acceder y entender su contenido y funcionamiento, pero, en general, produciendo una baja en su rendimiento.

⁶ Conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

CAPÍTULO 1: *Fundamentación teórica*

- Al emplear HTTP, pueden utilizar un sistema firewall sin cambiar las reglas de filtrado.

Simple Object Access Protocol (SOAP):

El término "servicios web SOAP" designa una tecnología que permite que las aplicaciones se comuniquen en una forma que no depende de la plataforma ni del lenguaje de programación. Un servicio web SOAP es una interfaz de *software* que describe un conjunto de operaciones a las cuales se puede acceder por la red a través de mensajería XML estandarizada. Usa protocolos basados en el lenguaje XML con el objetivo de describir una operación para ejecutar sobre los datos o para intercambiar con otro servicio web. Los servicios web SOAP, pueden describir cualquier tipo de datos de manera independiente de la plataforma para el intercambio entre sistemas, lo que permite la creación de aplicaciones de bajo acoplamiento. Además, pueden funcionar a un nivel más abstracto que permita reevaluar, modificar o manejar tipos de datos dinámicamente *on demand* (mediante solicitud). Por tanto, en términos técnicos, este tipo de servicios puede manejar datos con mucha más facilidad y permite una comunicación más libre entre los *software* (IBM developerWorks).

Los *frameworks*⁷ para servicios web SOAP se dividen en tres áreas: protocolos de comunicación, descripción de servicios y descubrimiento de servicios (Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI, 2002).

REpresentational State Transfer (REST)

REST (en español transferencia de representación de estado) es una arquitectura de servicio que no posee estados, por lo que no se puede realizar una llamada a un servicio e insertarle datos. Por tanto, el estado lo mantiene el cliente y debe ser él quien lo envíe en la llamada al servicio. REST se refiere a un conjunto de principios para el diseño de arquitecturas en red que se ejecutan sobre HTTP, estos principios abarcan cómo son definidos los recursos y cómo son diseccionados (Modelado, Diseño e Implementación de servicios web, 2007).

REST es un conjunto de técnicas orientadas a crear servicios web en las que se renuncia a la posibilidad de especificar la interfaz de los servicios de forma abstracta, a cambio de contar con una convención que

⁷ Estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de *software*, pueden incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

CAPÍTULO 1: *Fundamentación teórica*

permite manejar la información mediante una serie de operaciones estándar. La convención utilizada no es otra que el protocolo HTTP. La idea detrás de REST es el desarrollo de servicios orientados a la manipulación de recursos. En un servicio de este tipo, se tiene una URL por cada recurso (documento, entidad, etc.) que se gestiona, y que realiza una tarea diferente sobre dicho recurso en función del método HTTP que se utilice (Brito Calahorro, 2009).

1.4.2. *Metodología de desarrollo de software*

Teniendo en cuenta las necesidades para la solución del problema investigativo, la línea de desarrollo del proyecto AGORAV y la sugerencia por parte del cliente, se utiliza la metodología de desarrollo *Agile Unified Process*⁸ (AUP) en su variante UCI. Esta describe de una manera fácil de entender la forma de desarrollar aplicaciones de *software* de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP.

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de **Inicio**, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, llamada **Ejecución** y se agrega la fase de **Cierre**. (Rodríguez Sanchez, 2014)

- **Inicio:** En esta fase se lleva a cabo un estudio inicial de lo que desea el cliente para obtener información acerca del alcance del proyecto. Se decide si se realiza o no el proyecto.
- **Ejecución:** En esta fase se realiza la modelación del negocio, el levantamiento de requisitos, se define la arquitectura a utilizar, se implementa el *software* y por último se libera el producto.
- **Cierre:** En esta fase se realizan las actividades formales de cierre de proyecto.

AUP-UCI define 4 escenarios para modelar el sistema, de los cuales se escogió el escenario No. 2, el cual se basa en un modelo conceptual y define modelar el negocio mediante casos de uso. Se seleccionó este escenario porque no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, además es el recomendado para los proyectos donde el objetivo primario es la gestión y presentación de la información.

⁸ Proceso Unificado Ágil.

CAPÍTULO 1: *Fundamentación teórica*

AUP-UCI propone 7 disciplinas: **Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas liberación, Pruebas de aceptación.** Dependiendo del escenario escogido para el desarrollo del proyecto, varían los artefactos generados.

Entre las flexibilidades que posee AUP-UCI, permite la omisión del flujo de trabajo análisis, permitiendo pasar de la disciplina de requisitos directamente al diseño. En la solución se decidió explotar esta posibilidad, debido a que se cuenta con una descripción detallada de las funcionalidades del sistema y los requisitos se encuentran bien definidos.

1.4.3. Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML), permite visualizar, especificar, construir y documentar los artefactos de un sistema que involucra el desarrollo de un *software*. Un modelo UML describe lo que supuestamente hará un sistema, no cómo implementarlo. Este lenguaje dispone de reglas para combinar tales elementos y permite la modelación de sistemas con tecnología orientada a objetos. Los diagramas son entes importantes de UML, cuya finalidad es representar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. El modelo gráfico de UML tiene un vocabulario en el que se identifican: elementos, relaciones y diagramas (Jacobson, y otros, 2000).

Herramienta CASE Visual Paradigm 8.0

Visual Paradigm es una herramienta CASE que brinda un entorno para la realización de diagramas con UML y BPMN. Soporta todo el ciclo de desarrollo: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Posibilita una mejor comunicación entre los miembros del equipo ya que establece un lenguaje estándar, común para todos (Visual Paradigm, 2016).

Se decide utilizar la herramienta por la facilidad que brinda para el trabajo colaborativo en equipos de desarrollo, también por su robustez y el uso del lenguaje UML.

1.4.4. Lenguaje de programación

Lenguaje artificial que puede ser usado para controlar el comportamiento de una máquina, especialmente una computadora. Estos se componen de un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas (Joyanes Aguilar, 1996).

Java

CAPÍTULO 1: *Fundamentación teórica*

Java es un lenguaje de programación orientado a objetos ideado por la Sun Microsystems en 1995, supone un significativo avance en el desarrollo de *software*, es un lenguaje simple, distribuido, interpretado, robusto, de arquitectura neutral, portable y dinámico (Schildt, 2003).

A pesar de que existen otros lenguajes de programación para desarrollar aplicaciones *Android*, se seleccionó Java porque Google lo propone como lenguaje estándar para el desarrollo de sus aplicaciones nativas. Además existe una gran cantidad de bibliografía para apoyarse durante la fase de implementación.

1.4.5. Herramientas de desarrollo

IDE Android Studio 2.0

Android Studio es un entorno de desarrollo integrado (sus siglas en inglés IDE) para la plataforma *Android*, basado en el *software* IntelliJ IDEA de JetBrains. Este IDE posee características que facilitan el desarrollo de aplicaciones, algunas de ellas son: al escribir código, permite ver los cambios que realizamos en nuestra aplicación en tiempo real; es capaz de realizar completamiento de código avanzado, refactorizar y analizar código; es posible crear aplicaciones para cualquier dispositivo con SO *Android* ya sean *smartphones*, *tablets*, *Android Wear* o *Android TV*, también permite probarlas en emuladores virtuales con cualquier configuración de los tipos de dispositivos antes mencionados (*Android Studio*, 2016). A continuación, se muestran algunas ventajas de este IDE:

- Indexa todo el proyecto y hace predicciones sobre él.
- Posee renderizado en tiempo real de *layouts*⁹.
- Es capaz de asociar automáticamente carpetas y archivos con su papel en la aplicación.

JDK 8.66

Java Development Kit, es un *software* que provee herramientas de desarrollo para la creación de programas en Java. JDK consta de una serie de aplicaciones y componentes, para realizar cada una de las tareas de las que es capaz de encargarse (Oracle Corporation, 2016).

⁹ Contenedor de una o más vistas que controla su comportamiento y posición.

CAPÍTULO 1: *Fundamentación teórica*

Gradle 2.9

Gradle es un sistema de compilación basado en JVM (del inglés *Java Virtual Machine*) a su vez es un *plugin*, lo que facilita su actualización y su exportación de un proyecto a otro. Gradle permite reutilizar código fácilmente, hace sencilla la tarea de configurar y personalizar la compilación, permite la distribución sencilla de código al resto del mundo y gestiona las dependencias de forma potente y cómoda ya que está basado en Maven¹⁰ (Android Studio faq, 2014).

Características

- Gradle se basa en el lenguaje Groovy¹¹ para definir dependencias, tareas de ejecución de *scripts*¹² y variables del proyecto.
- Está presente a nivel de proyecto y de módulo.
- Es sencillo definir dependencias de módulos y librerías.

Software Development Kit (SDK) 24.4.1

El SDK (en español kit de desarrollo de *software*) es necesario para desarrollar aplicaciones y ejecutar un emulador del sistema *Android* de la versión que sea. Todas las aplicaciones *Android* se desarrollan en lenguaje Java con este kit. Cada vez que aparece una nueva versión de *Android*, Google libera el código fuente y publica el SDK con la nueva versión de *Android* para que los desarrolladores puedan adaptarse a los cambios y funcionalidades del nuevo sistema (Subirats, 2014).

1.5 Conclusiones parciales

- Mediante el análisis de las diferentes formas de dar solución al problema, se llegó a la conclusión que una aplicación nativa resuelve los problemas de rendimiento y usabilidad presentes hoy en el

¹⁰ Herramienta de *software* para la gestión y construcción de proyectos Java, que posee la capacidad de realizar tareas definidas, como la compilación del código y su empaquetado, siendo capaz de utilizar dependencias externas.

¹¹ Lenguaje de programación orientado a objetos implementado sobre la plataforma Java.

¹² Archivo de órdenes, archivo de procesamiento por lotes.

CAPÍTULO 1: Fundamentación teórica

consumo de contenidos audiovisuales desde dispositivos móviles en la plataforma XILEMA AGORAV, además de brindar una mejor experiencia de usuario.

- El estudio de las soluciones existentes demuestra que no pueden ser utilizadas para dar solución al problema planteado, ya que sus funcionalidades responden al negocio para el que están diseñadas, además no son compatibles con los servicios que brinda la plataforma XILEMA AGORAV. A pesar de esto se tomaron como referencia para el diseño y la presentación de la información.
- Las tecnologías escogidas para el desarrollo de la aplicación, fueron determinadas por las características de éstas. Fueron escogidos como IDE *Android Studio* 2.0 con el SDK 24.4.1 y Gradle 2.9. Visual Paradigm como herramienta para la generación de artefactos y lenguaje de programación Java, lo cual posibilitará la obtención de una solución bien documentada, ofreciendo facilidad para el desarrollo de futuras versiones de la aplicación, además de seguir los estándares utilizados internacionalmente.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

En el presente capítulo se hace una descripción del modelo de dominio y se muestran los conceptos principales y su relación. Se enuncian los requisitos para determinar el dominio de la aplicación, el desempeño, así como las restricciones que el mismo debe poseer, la especificación de los requisitos para determinar lo que el sistema debe realizar, además de las descripciones de los casos de uso para mayor comprensión de su funcionamiento. Se describe la arquitectura del sistema y se argumentan los patrones de diseño a utilizar. Además se establecen los estándares de codificación que se utilizan en la fase de implementación.

2.1. Modelo de dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés, su relación entre cada uno de los objetos y su forma de interacción (Larman, 2005). A partir del modelo de dominio se puede realizar un mejor diseño del sistema, facilitar el levantamiento de requisitos y comprender el flujo de acciones en el sistema. A continuación, en la Figura 2 se muestra el modelo de dominio:

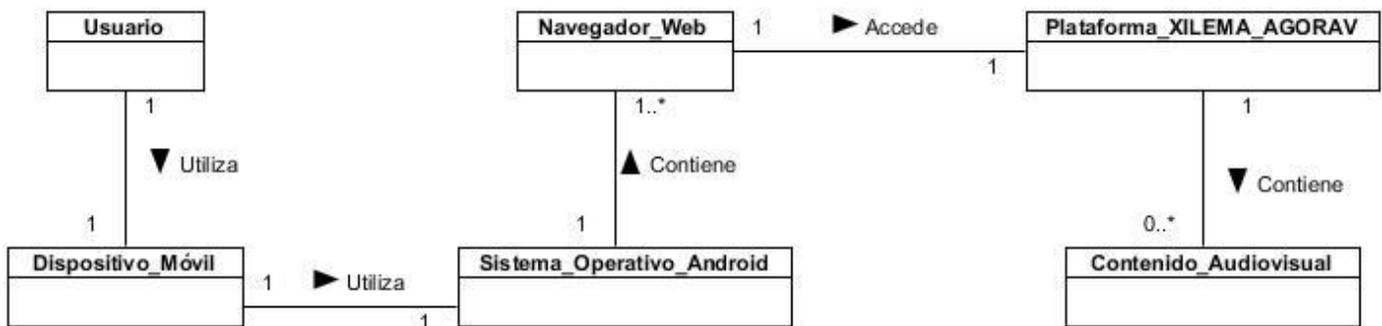


Figura 2: Modelo de Dominio.

2.1.1. Descripción de los conceptos del modelo de dominio

- **Usuario:** Es el actor del sistema que usa un dispositivo móvil para posteriormente acceder a la plataforma XILEMA AGORAV y consumir los contenidos audiovisuales publicados.
- **Dispositivo Móvil:** Dispositivo electrónico que se utiliza como terminal para acceder a la plataforma XILEMA AGORAV.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

- **Sistema_Operativo_Android:** Sistema operativo que usan algunos tipos de dispositivos móviles.
- **Navegador_Web:** Aplicación nativa *Android* para acceder a la Web, diseñado para el uso en dispositivos móviles.
- **Plataforma_XILEMA_AGORAV:** Plataforma de gestión de contenidos audiovisuales.
- **Contenido_Audiovisual:** Archivo de video publicado en la plataforma XILEMA AGORAV.

2.2. Requisitos de software

La especificación de requisitos es el proceso de desarrollo de *software* donde se analizan las necesidades exactas de los usuarios de un sistema y traducir éstas a precisas funciones y acciones que subsecuentemente serán usadas en el desarrollo del sistema. Un correcto levantamiento de requisitos permite posteriormente la construcción de un *software* de calidad (Pressman, 2010).

2.2.1. Requisitos funcionales

Para saber qué debe hacer el sistema y su funcionamiento es necesario conocer los requisitos funcionales (RF), de la manera en que este debe reaccionar a entradas particulares y como debe comportarse en situaciones particulares. Se utilizó la técnica tormenta de ideas para la recopilación de los requisitos funcionales. A continuación, se muestran los requisitos funcionales y una breve descripción de los mismos.

RF 1: Autenticar Usuario

Descripción: La aplicación debe permitir que los usuarios autenticarse en la plataforma XILEMA AGORAV mediante el uso de servicios web.

RF 2: Cerrar sesión.

Descripción: La aplicación debe permitir a los usuarios cerrar su sesión en la plataforma XILEMA AGORAV mediante el uso de servicios web.

RF 3: Configurar aplicación.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Descripción: La aplicación debe permitir al usuario cambiar los parámetros de configuración.

RF 4: Reproducir material audiovisual.

Descripción: La aplicación debe permitir al usuario reproducir un contenido audiovisual mediante el consumo de servicios web desde la plataforma XILEMA AGORAV.

RF 5: Pausar reproducción de material audiovisual.

Descripción: La aplicación debe permitir al usuario pausar la reproducción de un material audiovisual.

RF 6: Detener reproducción de un material audiovisual.

Descripción: La aplicación debe permitir al usuario detener la reproducción de un material audiovisual.

RF 7: Subir volumen.

Descripción: La aplicación debe permitir al usuario subir el volumen mientras se reproduce un material audiovisual.

RF 8: Bajar el volumen

Descripción: La aplicación debe permitir al usuario bajar el volumen mientras se reproduce un material audiovisual.

RF 9: Ir a un instante de tiempo en la reproducción.

Descripción: La aplicación debe permitir al usuario ir a un instante de tiempo específico en la reproducción de un material audiovisual.

RF 10: Iniciar descarga de un material audiovisual.

Descripción: La aplicación debe permitir al usuario descargar un material audiovisual desde la plataforma XILEMA AGORAV.

RF 11: Cancelar descarga de un material audiovisual.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Descripción: La aplicación debe permitir al usuario cancelar la descarga de un material audiovisual.

RF 12: Pausar descarga de un material audiovisual.

Descripción: La aplicación debe permitir al usuario pausar la descarga de un material audiovisual.

RF 13: Listar descargas de materiales audiovisuales.

Descripción: La aplicación debe permitir al usuario visualizar la lista de descargas de materiales audiovisuales.

RF 14: Efectuar comentario referente a un material audiovisual.

Descripción: La aplicación debe permitir al usuario realizar un comentario referente a un material audiovisual mediante el uso de servicios web.

RF 15: Efectuar voto referente a un material audiovisual.

Descripción: La aplicación debe permitir al usuario realizar un voto referente a un material audiovisual mediante el uso de servicios web.

RF 16: Buscar material audiovisual.

Descripción: La aplicación debe permitir al usuario realizar la búsqueda de materiales audiovisuales por el nombre mediante el uso de servicios web.

2.2.2. Requisitos no funcionales

Los requisitos no funcionales se refieren a funciones específicas que proporciona el sistema. Son restricciones de los servicios o funciones ofrecidas por el sistema (fiabilidad, tiempo de respuestas, capacidad de almacenamiento). Generalmente se aplican al sistema en su totalidad. Surgen de las necesidades del usuario (restricciones de presupuesto, políticas de la organización, necesidad de interoperabilidad). En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto. Están vinculados a requisitos funcionales, es decir, una vez que se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

grande debe ser (Pressman, 2010).A continuación, se muestran los requisitos no funcionales y la descripción de los mismos:

Requisitos de usabilidad: Las interfaces gráficas implementadas por el sistema deben concebirse con un ambiente sencillo y de navegación fácil para el usuario, debido a que la aplicación podrá ser usada por usuarios con conocimientos básicos de dispositivos móviles.

Requisitos de *software*: Se debe disponer en el dispositivo móvil una versión 2.3.3 o superior del sistema operativo *Android*, ya que el 99% de los dispositivos móviles con sistema operativo *Android* actualmente utilizan esta versión o una superior.

Requisitos de *hardware*:

- El dispositivo debe soportar tecnología Wi-Fi 802.11 b/g/n.
- Se requiere un microprocesador con una velocidad de 650 MHz o superior.
- Se requiere un mínimo de 256 Mb de memoria RAM.
- Dispositivo con una resolución igual o superior a 320 x 480 píxeles, para una correcta visualización de la aplicación.

2.3. Casos de uso

Todo *software* ofrece a su entorno un conjunto de servicios, un caso de uso es la utilización de estos servicios por alguien o algo fuera del sistema. Los casos de uso definen un escenario que identifica una línea de utilización para el sistema, facilitando una descripción de cómo se utilizará el mismo (Pressman, 2010). A continuación, se listan los casos de uso del sistema:

CU 1: Autenticar Usuario.

CU 2: Reproducir Audiovisual.

CU 3: Configurar Aplicación.

CU 4: Cerrar Sesión.

CU 5: Gestionar Descarga.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

CU 6: Efectuar Comentario.

CU 7: Efectuar Voto.

CU 8: Buscar Contenido Audiovisual.

A continuación, en la Figura 3 se muestra el diagrama de casos de uso del sistema para una mejor comprensión de las funcionalidades, evidenciándose la pertenencia de cada caso de uso con el actor que lo ejecuta, donde el actor Usuario_Autenticado es una especialización del actor Usuario.

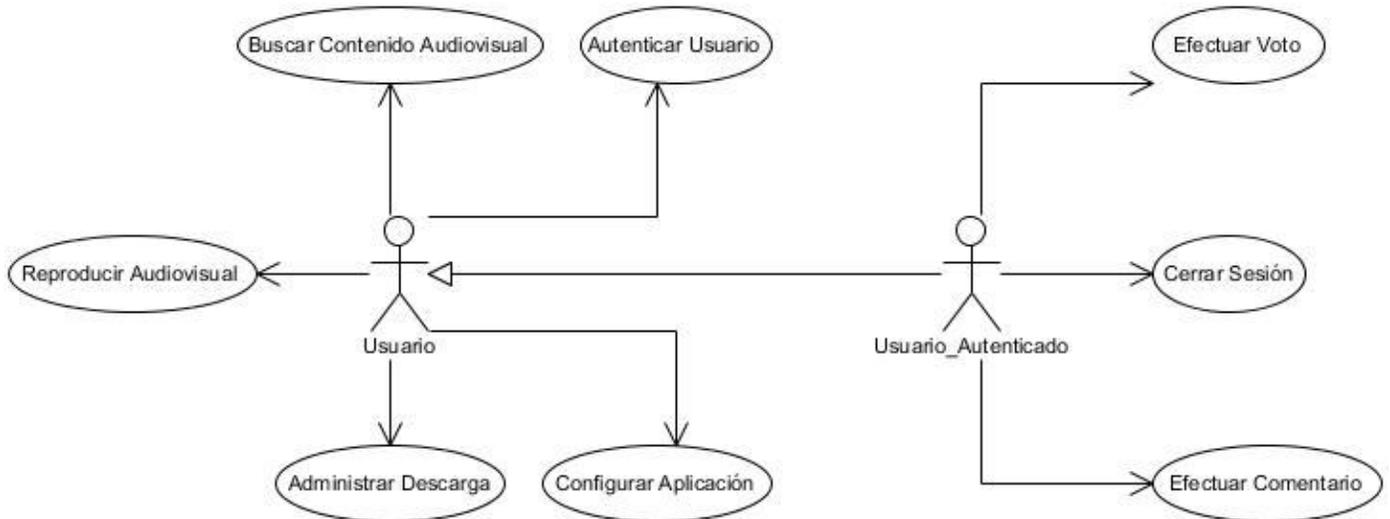


Figura 3: Diagrama de casos de uso del sistema.

2.3.1. Descripción de los actores del sistema

Los actores del sistema se definen como los agentes externos al sistema que interactúan con este. En el diagrama de caso de uso del sistema se especifica con que servicios específicos interactúa cada actor.

Tabla 2: Descripción de los actores del sistema.

Actor	Descripción
Usuario	Es la persona que interactúa con la aplicación y consume algunos servicios que brinda la plataforma XILEMA AGORAV.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Usuario_Autenticado	Es la persona que interactúa con la aplicación y tiene total acceso a los servicios que brinda la plataforma XILEMA AGORAV.
---------------------	---

2.3.2. Descripción de los casos de uso del sistema:

La descripción detallada de los casos de uso posibilita el posterior diseño, implementación y diseño de las pruebas de interfaz. A continuación, en la Tabla 3 se muestra la descripción del caso de uso Reproducir Contenido Audiovisual.

Tabla 3: Descripción del caso de uso Reproducir Audiovisual.

Objetivo	Realizar acciones de reproducción seleccionadas por el actor.	
Actores	Usuario, Usuario_Autenticado	
Resumen	El caso de uso se inicia cuando se selecciona el material que se desea reproducir. Las opciones de reproducción estarán disponibles mientras se esté reproduciendo un material audiovisual. El caso de uso termina cuando el actor realiza alguna opción de reproducción sobre el material o la reproducción del mismo llega a su fin.	
Complejidad	Alta	
Prioridad	Alta	
Precondiciones	Se seleccionó un material para su reproducción.	
Poscondiciones	Acciones de reproducción seleccionadas por el usuario sobre el material seleccionado.	
Flujo de eventos		
Flujo básico:	Reproducir material.	
	Actor	Sistema
1.	Selecciona el material que desea reproducir.	
2.		Carga el material seleccionado en el reproductor. El material comienza a reproducirse por defecto.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

3.	<p>Puede realizar las siguientes acciones de reproducción:</p> <ul style="list-style-type: none"> - Pausar/Continuar reproducción. - Parar reproducción. Ver sección 1: “Parar reproducción”. - Ir a un instante de tiempo. Ver sección 2: “Ir a un instante de tiempo”. - Controlar volumen. Ver sección 3: “Controlar volumen”. 	
4.	<p>Presiona el botón “Pausar o Continuar”.</p> <p>Observación: Si el material está reproduciéndose el botón estará activo con la función de pausa, si el material está pausado el botón estará activo con la función de continuar la reproducción.</p>	
5.		Se pausa o continúa la reproducción del material. Termina el CU.
Sección 1: “Parar reproducción”.		
Flujo básico: Parar reproducción.		
	Actor	Sistema
1.	Presiona el botón “Parar”.	
2.		El reproductor detiene la reproducción. Termina el CU.
Sección 2: “Ir a un instante de tiempo”.		
Flujo básico: Ir a un instante de tiempo.		
	Actor	Sistema
1.	El usuario hace clic sobre una posición en la línea de tiempo de reproducción.	
2.		El reproductor sitúa la reproducción del material en el tiempo indicado por el usuario. Termina el CU.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Flujos alternos		
1.1 Evento: botón "Avanzar"		
1.1	El usuario selecciona el botón "Avanzar".	
2.1		El sistema avanza 10 segundos en la reproducción del material. Termina el CU.
1.2 Evento: botón "Atrasar"		
1.2	El usuario selecciona el botón "Atrasar".	
2.2		El sistema atrasa 10 segundos en la reproducción del material. Termina el CU.
Sección 3: "Controlar volumen".		
Flujo básico: Controlar volumen.		
	Actor	Sistema
1.	Presiona el botón "Aumentar volumen" del dispositivo.	
2.		Se percibe el aumento en el volumen del sonido del material que se reproduce. Termina el CU.
Flujos alternos		
1.1 Evento: Disminuir el volumen.		
	Actor	Sistema
1.1	Presiona el botón "Disminuir volumen" del dispositivo.	
2.1		Se percibe disminución en el volumen del sonido del material que se reproduce. Termina el CU.
Relaciones		CU Incluidos
		No existe.
		CU Extendidos
		No existe.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución



2.4. Arquitectura de software

La arquitectura de un sistema consiste en la descripción de los subsistemas y componentes de un sistema informático y las relaciones que se establecen entre ellos (Pressman, 2010). La correcta selección y aplicación de una arquitectura en el diseño ingenieril de un proyecto es fundamental, pues permite organizar el desarrollo, promover la reutilización de código, facilitar el mantenimiento y modificaciones posteriores.

2.4.1. Estilo arquitectónico

Carlos Billy Reynoso en su libro *Introducción a la Arquitectura de Software* plantea: “Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica.” Los estilos arquitectónicos indican los tipos de componentes y conectores utilizados, además provee una serie de patrones óptimos para el desarrollo.

Para el desarrollo de la solución propuesta se escoge el estilo arquitectónico **Llamada y Retorno**, pues permite la realización de estructuras de programación escalables y fáciles de modificar.

2.4.2. Patrón arquitectónico

Los patrones arquitectónicos proveen una descripción de los componentes, elementos y conectores; así como la interacción entre ellos. Además, exponen un conjunto de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico organiza estructuralmente un sistema de *software*, definiendo subsistemas a partir de sus responsabilidades e interrelaciones (Reynoso, 2004).

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Para el desarrollo de la solución propuesta se escoge el patrón arquitectónico **N-Capas**, en específico 3-Capas. Este patrón organiza jerárquicamente en capas, donde cada capa provee de información a la capa superior y se sirve de la inferior, permitiendo agrupar en tres capas los componentes de la aplicación (Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con Android, 2012). Las cuales se detallan a continuación:

- **Capa de presentación:** es la única que interactúa directamente con el usuario, permitiendo el intercambio de información con ellos. En esta capa se agrupan todas las vistas.
- **Capa de lógica:** recibe las peticiones de la capa de presentación y le envía la respuesta tras el procesamiento de las mismas. Agrupa los componentes lógicos del flujo de trabajo, los de acceso a interfaces de servicios y el consumo de los mismos.
- **Capa de acceso a datos:** interactúa con la capa de lógica, recibe solicitudes de almacenamientos o recuperación de información. Agrupa clases de Java para el acceso a ficheros físicos y consultas a bases de datos.

La principal ventaja de esta arquitectura, es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Además a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables. A continuación, en la Figura 4 se presentan la pertenencia de las diferentes clases a cada una de las capas propuesta para la implementación de la solución:

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

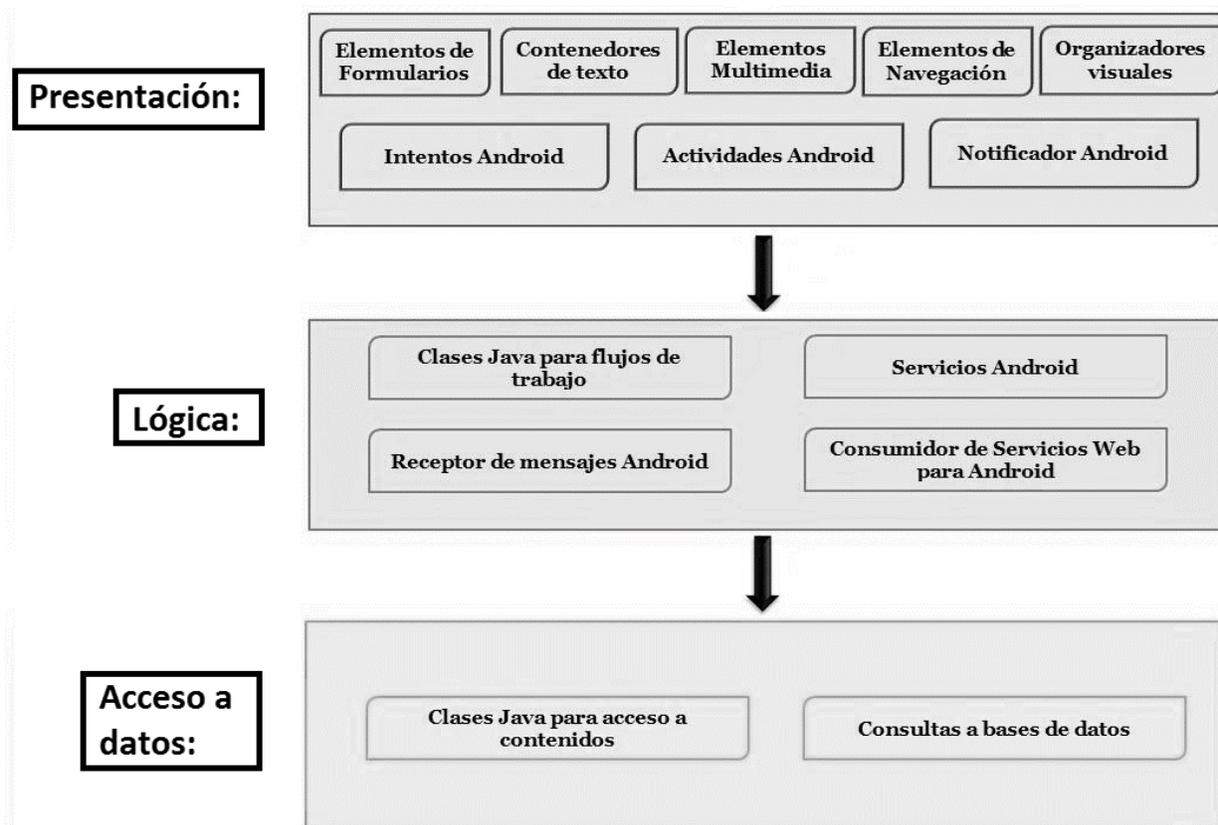


Figura 4: Diagrama orientado a dominio de la propuesta arquitectónica.

2.4.3. Patrones de diseño

Los patrones de diseño elevan la calidad de los sistemas, además de brindar una solución probada y documentada a problemas de desarrollo de *software* que están sujetos a contextos similares, también de representar una excelente manera de trabajar en base a la calidad (Fowler, 2007). Existen clasificaciones para estos patrones entre los que se encuentran los Patrones Generales de *Software* para Asignar Responsabilidades y los Patrones del Grupo de los Cuatro. A continuación, se presentan los patrones de diseño que son utilizados para el desarrollo de la solución.

Patrones Generales de *Software* para Asignar Responsabilidades

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Los patrones generales de *software* para asignar responsabilidades (por sus siglas en inglés GRASP¹³) permiten asignar responsabilidades a objetos de forma tal que estos conozcan el rol que les corresponde. (Larman, 2005). Los patrones que se utilizan son:

Experto: Asigna una responsabilidad a la clase que cuenta con la información necesaria para cumplirla. Constituye un principio básico que frecuentemente se utiliza en el diseño orientado a objetos. Las ventajas que presenta este patrón son: la conservación del encapsulamiento, debido a que los objetos para hacer lo que se les pide se valen de su propia información, además de posibilitar que el sistema sea robusto y de fácil mantenimiento.

Creador: Asigna la responsabilidad a una clase de crear objetos, práctica muy frecuente en los sistemas orientados a objetos. Entre sus beneficios se encuentra el brindar soporte a un bajo acoplamiento lo que trae como consecuencia una menor dependencia respecto al mantenimiento y la reutilización en mayor medida.

Controlador: Asigna la responsabilidad a una clase de administrar un mensaje de los eventos del sistema. *“Un evento del sistema es un evento de alto nivel generado por un actor externo; es un evento de entrada externa. Se asocia a operaciones del sistema: las que emite en respuesta a los eventos del sistema”* (Larman, 2005).

Alta cohesión: Asigna las responsabilidades en función de mantener la complejidad dentro de los límites posibles. La cohesión es una medida que refleja cuán enfocadas y relacionadas se encuentran las responsabilidades en una clase. Las clases que presentan una alta cohesión se caracterizan por poseer responsabilidades estrechamente relacionadas que no hagan un enorme trabajo; sin embargo, una clase con baja cohesión realiza muchas cosas no afines a un trabajo excesivo. Entre los beneficios que presenta el patrón está la mejora de la facilidad y claridad para comprender el diseño.

Bajo acoplamiento: Asigna la responsabilidad a las clases garantizando que las mismas se encuentren lo más independiente posibles, lo que trae consigo una dependencia escasa y un aumento en la reutilización.

¹³ GRASP: *General Responsibility Assignment Software Patterns*.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

Patrones Grupo de los Cuatro (GOF)

Los patrones del Grupo de los Cuatro (por sus siglas en inglés GOF¹⁴) se dividen en tres categorías: patrones de creación, patrones estructurales y patrones de comportamiento (Larman, 2005). A continuación, se describen los patrones que se evidencian en la solución:

Instancia única (del inglés *Singleton*): es un patrón de creación y garantiza que una clase posea una única instancia, a la vez que provee un punto de acceso global a ella. Permite por su diseño restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. En la solución se empleó para la construcción de la clase *Controller*.

Adaptador (del inglés *Adapter*): convierte la interfaz de una clase en otra distinta que es la que esperan los clientes. Permite que cooperen clases que de otra manera no podrían por tener interfaces incompatibles. Este patrón permitió el diseño e implementación de varias clases de la capa Presentación, específicamente las clases *Adapter*.

2.5. Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema (Larman, 2005).

2.5.1. Diagrama de clases del diseño

Un diagrama de clases sirve para visualizar las relaciones entre las clases que involucran el sistema, los cuales responden al cumplimiento de los requisitos funcionales. Los diagramas de clases del diseño son subproductos del modelo de diseño proporcionando una perspectiva estática que representa el diseño estructural del sistema, mostrando un conjunto de clases y sus atributos (Larman, 2005). A continuación, en la Figura 5 se muestra el diagrama de clases del diseño del caso de uso Reproducir Audiovisual:

¹⁴ GOF: *Gang Of Four*.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

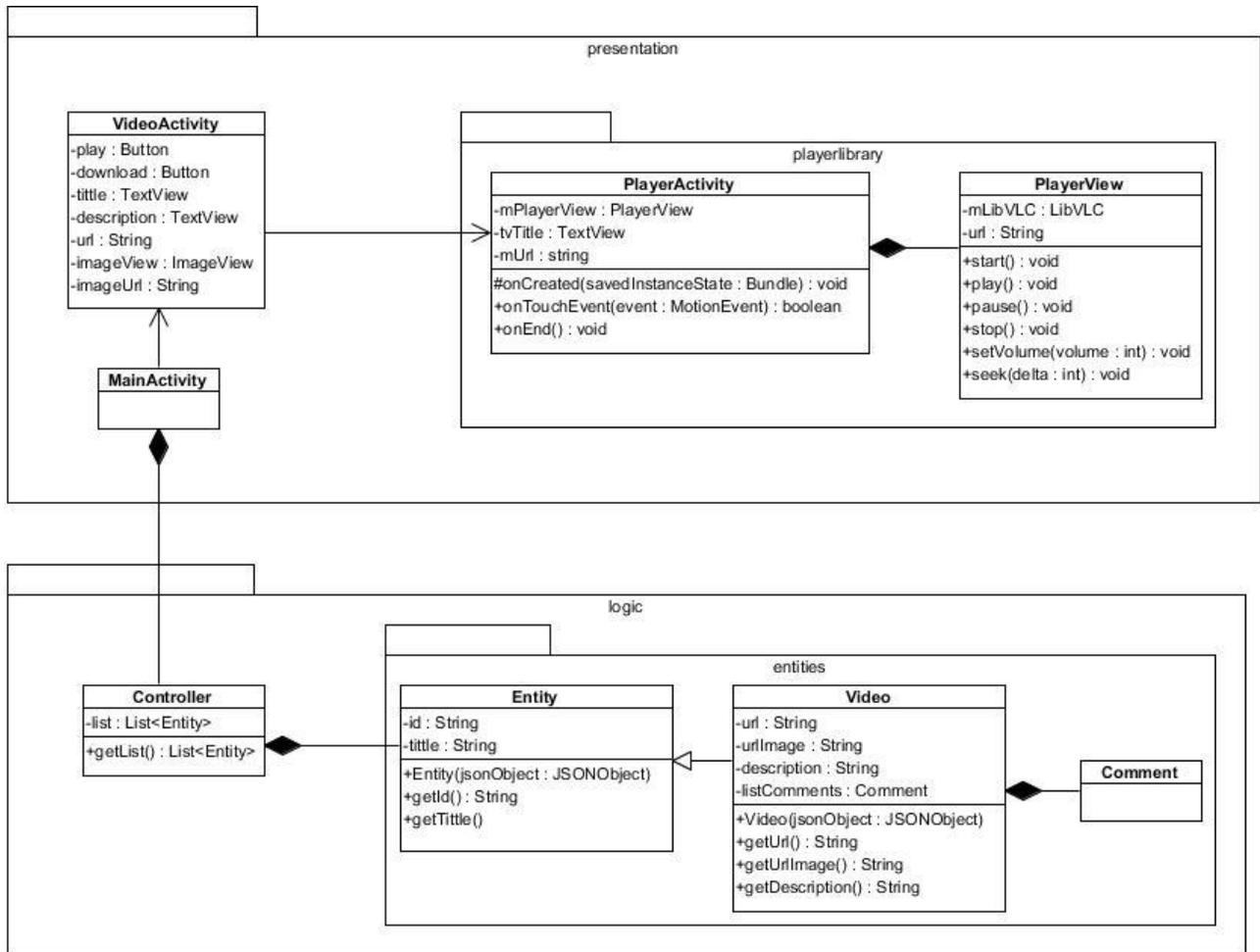


Figura 5: Diagrama de clases del diseño Reproducir Audiovisual.

2.6. Estándar de codificación

El estándar en el código de programación es muy importante en cualquier producto, principalmente si este involucra muchos desarrolladores. Esto asegura que el código sea de alta calidad, que contenga una cantidad baja de errores y sea fácil de mantener (Becerra, 2011).

Principios generales

- Los nombres de cada uno de los elementos del programa deben ser significativos; su nombre debe explicar en lo posible el uso del elemento.
- No manejar en los programas más de una instrucción por línea.

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

- Declarar las variables en líneas separadas.
- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.
- La mayoría de los elementos se deben nombrar usando sustantivos.
- Los atributos deben comenzar con letra minúscula y los métodos deben comenzar con letra mayúscula.
- Especificar importaciones: especificar la ruta completa de la clase a utilizar. Esto posibilita conocer cuál es la clase utilizada en cada paquete importado, el código es más legible y entendible.
- Escribir métodos cortos: siempre que sea posible, escribir métodos pequeños y concretos. Sin embargo, no hay límites estrictos para la cantidad de líneas de código.

Nomenclatura de los elementos

Para una mejor organización y comprensión del código es recomendable definir reglas de nombramiento para cada tipo de elementos. Cumplir con esta nomenclatura permite crear *software* más escalables y fáciles de mantener. A continuación, se define la nomenclatura de los elementos a seguir en la solución:

Tabla 4: Nomenclatura de los elementos para el estilo de codificación.

Elemento	Reglas de nombramiento
Clases, interfaces	Nombre sustantivo singular, con la primera letra en mayúscula y las demás en minúsculas. Si el nombre de la clase está comprimido en más de una palabra la primera letra debe de ser mayúscula. Letras mayúsculas seguidas no son permitidas. No hay separación entre palabras. Ejemplo: "ZendPDF" no es permitido mientras que "ZendPdf" sí lo es.
Archivos fuente	Nombre sustantivo singular completamente en minúscula y sin

CAPÍTULO 2: Levantamiento de requisitos y diseño de la solución

	separación entre las palabras.
Constantes	Nombre sustantivo en mayúsculas. Para separar palabras se usará el guion bajo: _
Paquetes y directorios	Nombre sustantivo singular en minúsculas

2.7. Conclusiones parciales

- La elaboración del modelo de dominio permitió un mejor entendimiento de los conceptos principales en el entorno del consumo de contenidos audiovisuales desde dispositivos móviles con sistema operativo *Android*.
- Mediante la tormenta de ideas se logró definir los requisitos funcionales y no funcionales de la aplicación AGORAV, donde el diagrama de casos de uso del sistema junto con la descripción de los casos de uso permitieron una detallada descripción del flujo y comportamiento de cada requisito.
- El uso del estilo arquitectónico Llamada y retorno y el patrón arquitectónico N-Capas, en conjunto con el uso de patrones de diseño GRASP Y GOF contribuyeron a una adecuada organización de la aplicación en 3 componentes o capas (Presentación, Lógica y Acceso a datos), lo que permitió obtener una solución escalable.
- El uso de los estándares de codificación definidos por el proyecto AGORAV junto con los definidos por Google, permitieron la obtención de un código legible.

CAPÍTULO 3: Implementación y validación de la solución

CAPÍTULO 3: Implementación y validación de la solución

En el presente capítulo se hace un análisis de los resultados obtenidos durante el desarrollo de la solución propuesta. Se realiza el diagrama de componentes para describir la organización del sistema y sus relaciones, así como el modelo de despliegue. Además, se especifican las diferentes métricas para la validación de los requisitos y el diseño.

3.1. Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes, como ficheros de código fuente ejecutables. Este modelo refiere también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración (Larman, 2005).

3.1.1. Diagrama de Componentes

Según Craig Larman *“Un diagrama de componentes muestra las dependencias lógicas entre componentes de software, sean éstos componentes: fuentes, binarios o ejecutables, ilustran las piezas del software, controladores embebidos, etc. Prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema, es decir para describir la vista de implementación estática de un sistema”*. Llegado el momento de la implementación de la solución, todos los diagramas generados en fases anteriores se materializan en un sistema integrando las partes necesarias para la obtención de un producto final. Estas relaciones de dependencia se modelan a través del diagrama de componentes.

A continuación, se muestra el diagrama de componentes donde se aprecia la división del sistema en componentes y las dependencias que se generan entre los mismos. Primeramente, se presenta la *Aplicación_AGORAV_Android* que consiste en un ejecutable que depende de las bibliotecas *materialsbstrip:library:1.1.0*, *FileDownloader-0.3.2*, *ksoap2-Android-assembly-3.0.0* y del paquete *bibliotecas del reproductor*. La biblioteca *materialsbstrip:library:1.1.0* permite crear las vistas de la aplicación, que a su vez depende de la biblioteca *nineoldAndroids-2.4.0* usada para dar soporte a las versiones superiores a la 9 del SO *Android*. La biblioteca *ksoap2-Android-assembly-3.0.0* permite consumir los servicios SOAP brindados por la plataforma XILEMA AGORAV. El paquete *bibliotecas del reproductor* contiene un conjunto de bibliotecas como: *gdb.setup*, *gdbserver*, *libiomx-ics.so*, entre otras, las cuales son utilizadas por el reproductor de la aplicación. La biblioteca *FileDownloader- 0.3.2* se utiliza para la gestión de descargas de los contenidos audiovisuales.

CAPÍTULO 3: Implementación y validación de la solución

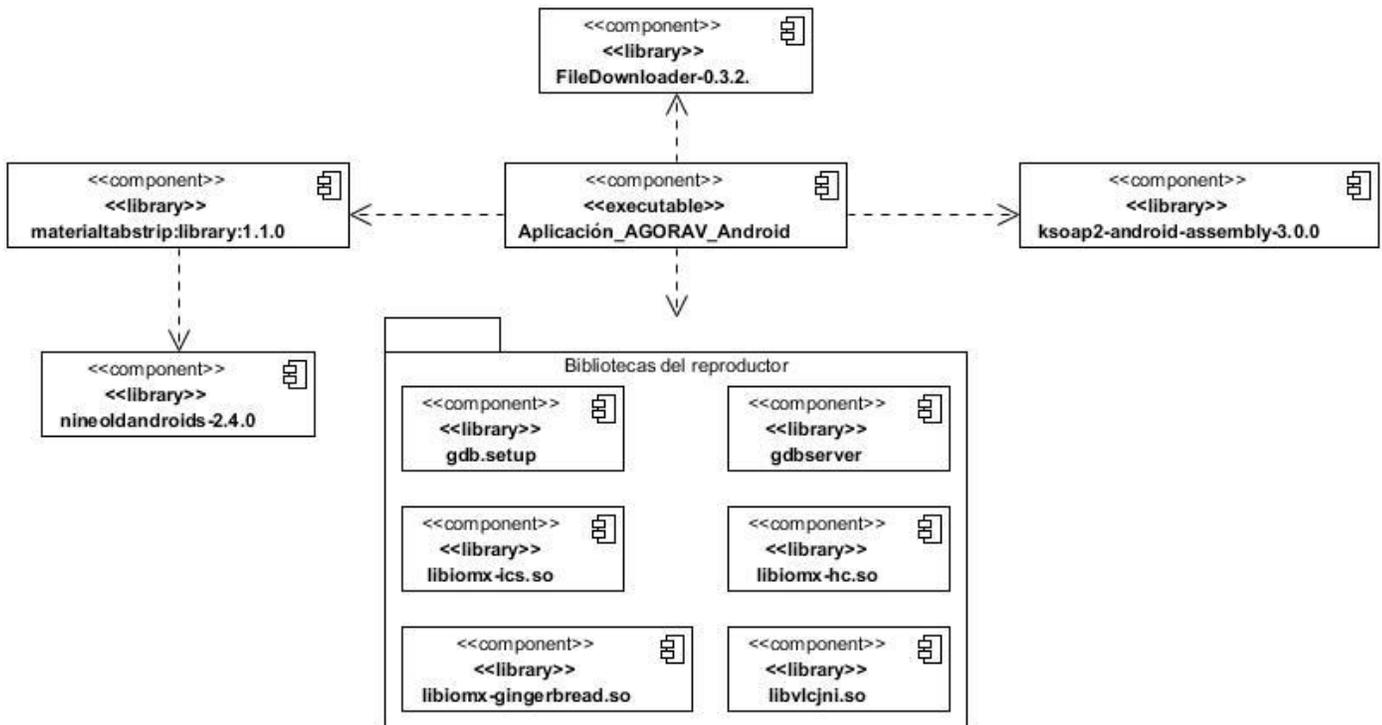


Figura 6: Diagrama de componentes.

3.2. Modelo de despliegue

Ivar Jacobson y otros en el libro *El proceso unificado de desarrollo de software* definen el modelo de despliegue como: “*un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos del cómputo*”. Este modelo es primordial en las actividades de diseño e implementación, pues la distribución que posee el sistema tiene mucha influencia en su diseño. El modelo de despliegue representa en sí mismo una correspondencia entre el *software* y el *hardware*. Un nodo es un objeto físico que existe en tiempo de ejecución y que representa algún tipo de recurso computacional, en él se ejecutan los componentes y además posee relaciones que representan medios de comunicación entre ellos. Los componentes son los elementos que intervienen en la ejecución del sistema.

3.2.1. Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas entre los componentes *hardware* y *software* en el sistema final, es decir la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes de *software* (Martínez Zurita, 2013). En la Figura 7 se muestra el diagrama de despliegue perteneciente al sistema propuesto.

CAPÍTULO 3: Implementación y validación de la solución

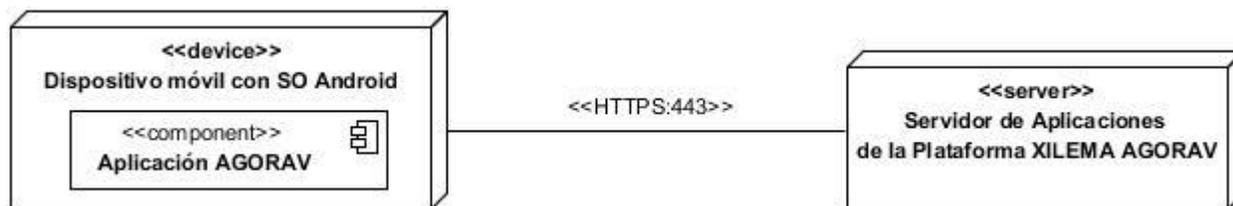


Figura 7: Diagrama de despliegue.

Descripción de los nodos físicos del sistema:

<<device>> Dispositivo móvil con SO Android: Dispositivo móvil donde se encuentra instalada la aplicación AGORAV.

<<component>> Aplicación AGORAV: Aplicación que se comunica con la plataforma XILEMA AGORAV y consume los servicios que brinda esta.

<<server>> Servidor de Aplicaciones de la Plataforma XILEMA AGORAV: Plataforma que brinda servicios web a la aplicación AGORAV.

3.3. Validación de la solución

Ian Sommerville define que la validación tiene como objetivo: “*asegurar que el sistema de software satisface las expectativas del cliente*”. Así como también establecer la seguridad de que el *software* está hecho para un propósito, o sea, debe ser lo suficientemente bueno para su uso pretendido. A continuación, se especifican los tipos de pruebas a realizar para garantizar la calidad de la solución.

3.3.1. Pruebas funcionales

Las pruebas funcionales tienen como objetivo asegurar que el *software* cumpla con las especificaciones requeridas y eliminar los posibles errores que este pueda tener. Para la realización de estas pruebas previamente se deben crear diseños de modelos de pruebas, donde se especifican las entradas y salidas del sistema, los flujos de eventos y descripción detallada de las respuestas del sistema (López Ceballos, 2012). Las pruebas funcionales se aplican de forma manual y desde el punto de vista del usuario.

Pruebas de caja negra

El método de prueba caja negra o prueba de comportamiento se centra en los requisitos funcionales del sistema. La correcta aplicación de este tipo de prueba puede arrojar no conformidades como: funciones incorrectas o faltantes, errores de interfaz, errores en estructura de datos o en el acceso a bases de datos

CAPÍTULO 3: Implementación y validación de la solución

externas, errores de comportamiento o desempeño y errores de inicialización y término (Pressman, 2010). Con el fin de garantizar el correcto funcionamiento de la aplicación AGORAV se ejecutaron los casos de prueba diseñados para cada caso de uso del sistema, utilizando la técnica partición de equivalencia. En la Tabla 5 se muestra el diseño de caso de prueba del caso de uso Reproducir Audiovisual. Para los demás casos de prueba dirigirse al artefacto Diseño de casos de prueba.

CAPÍTULO 3: Implementación y validación de la solución

Tabla 5: Caso de prueba del caso de uso Reproducir Audiovisual

SC 1: Reproducir material.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Reproducir material correctamente.	El usuario selecciona el material que desea reproducir de los publicados en la plataforma XILEMA AGORAV.	Reproduce el archivo seleccionado a pantalla completa Observación: - En caso de no existir conexión con la plataforma, muestra el mensaje de error "Inténtelo más tarde".	Aplicación. - Seleccionar el botón reproducir asociado a una publicación.

SC 2: Pausar reproducción de material

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Pausar material correctamente.	El usuario pulsa el botón pausar y el material deja de reproducirse.	Pausa el material.	Aplicación. - Seleccionar el botón reproducir asociado a una publicación. - Seleccionar el botón pausar del reproductor.
EC 2.2 Reanudar reproducción del material correctamente.	El usuario pulsa el botón reproducir y el material comienza a reproducirse a partir de la pausa.	Se reproduce el material pausado.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Seleccionar el botón pausar del reproductor.

CAPÍTULO 3: Implementación y validación de la solución

			- Seleccionar el botón de reproducir del reproductor.
--	--	--	---

SC 3: Parar reproducción de material

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 3.1 Parar material correctamente.	El usuario pulsa el botón parar y el material audiovisual deja de reproducirse.	Deja de reproducirse el material.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Seleccionar el botón parar del reproductor.

SC 4: Ir al tiempo de reproducción especificado

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 4.1 Ir a tiempo de reproducción especificado correctamente.	El usuario pulsar sobre una posición en la línea de tiempo del reproductor.	Reproduce el material a partir de la posición seleccionada.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Seleccionar una posición en la línea de tiempo del reproductor.

CAPÍTULO 3: Implementación y validación de la solución

EC 4.2 Adelantar reproducción del material correctamente	EL usuario pulsa el botón adelantar reproducción.	Adelanta la reproducción 10 segundos en la línea de tiempo del reproductor.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Seleccionar el botón adelantar reproducción.
EC 4.3 Atrasar reproducción del material correctamente	EL usuario pulsa el botón atrasar reproducción.	Atrasa la reproducción 10 segundos en la línea de tiempo del reproductor.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Seleccionar el botón atrasar reproducción.

SC 5: Controlar volumen

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 5.1 Aumentar volumen correctamente	El usuario aumenta el volumen mediante el botón "Aumentar volumen" del dispositivo.	Aumenta el volumen del material en reproducción.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Aumentar el volumen mediante el botón "Aumentar volumen" del dispositivo.

CAPÍTULO 3: Implementación y validación de la solución

EC 5.2 Disminuir volumen correctamente	El usuario disminuye el volumen mediante el botón "Disminuir volumen" del dispositivo.	Disminuye el volumen del material en reproducción.	Aplicación. - Seleccionar el botón de reproducir asociado a una publicación. - Disminuir el volumen mediante el botón "Disminuir volumen" del dispositivo.
--	--	--	--

CAPÍTULO 3: Implementación y validación de la solución

Resultados de las pruebas de caja negra:

Durante el transcurso de la ejecución de las pruebas de caja negra se identificaron 10 no conformidades (NC) en 3 iteraciones como se muestra en la Figura 8:

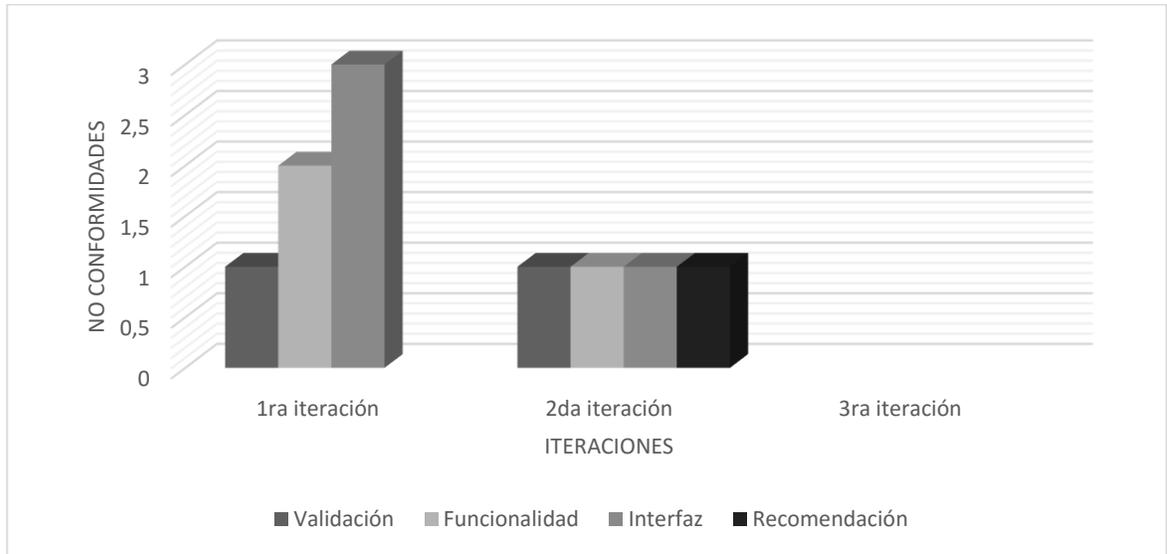


Figura 8: Gráfica de no conformidades de las pruebas de caja negra

Entiéndase por NC de validación a los errores relativos a todo tipo de validación, NC de funcionalidad a cuando el resultado mostrado al realizar una acción no es el esperado, NC de interfaz a los errores visuales relacionados con el diseño de las interfaces y las NC de recomendaciones son sugerencias hechas por los probadores.

No conformidades detectadas en la 1ra iteración

NC de validación:

- En la funcionalidad Autenticar Usuario era posible enviar campos vacíos a la plataforma, generando que la aplicación AGORAV colapsara.

NC de funcionalidad:

- Al presionar el botón Reproducir Audiovisual asociado a una publicación, no se ejecutaba correctamente el caso de uso, la aplicación no ejecutaba el reproductor.

CAPÍTULO 3: Implementación y validación de la solución

- Al presionar el botón Comentar asociado a una publicación, la aplicación mostraba el mensaje de error “Compruebe su conexión”, esto ocurría existiendo una conexión con la plataforma.

NC de interfaz:

- El tamaño de letra en el *drawer*¹⁵ era muy pequeño por lo que era difícil leer el contenido que mostraba.
- En las publicaciones el título del contenido audiovisual no se mostraba.
- El botón Reanudar Descargas no aparecía al pasar al modo pausa de descarga, pero sí se ejecutaba la funcionalidad.

No conformidades detectadas en la 2da iteración

NC de validación:

- Al reproducir un audiovisual sin existir conexión la aplicación AGORAV colapsaba, pues no estaba funcionando la validación para cuando no existiese conexión.

NC de funcionalidad:

- Al presionar el botón Detener Descarga asociado a una descarga no se ejecutaba el caso de uso.

NC de interfaz:

- El tamaño de letra en la descripción asociada a un contenido presente en la interfaz Publicaciones era muy pequeña causando dificultad al leer.

NC de recomendación:

- Mostrar un mensaje al realizar un comentario, pues el usuario no sabía si su comentario había sido enviado o no.

¹⁵ Menú oculto deslizable, ubicado en la mayoría de los casos en la parte lateral izquierda de la interfaz de usuario.

CAPÍTULO 3: Implementación y validación de la solución

Al culminar cada iteración de pruebas se procedió a resolver las no conformidades detectadas, dando paso a una nueva iteración. Al concluir la 3ra iteración no se detectaron no conformidades, garantizando así el correcto funcionamiento y comportamiento de todos los casos de uso.

3.3.2. Pruebas de rendimiento

Las pruebas de rendimiento se realizan para determinar lo rápido y capaz que realiza una tarea un *software* en condiciones particulares de trabajo. En este tipo de pruebas se valora el tiempo de respuesta, los retardos en la entrega de mensajes, si la aplicación se bloquea, el consumo de memoria y del CPU¹⁶ o la carga de los problemas de velocidad (Mejores prácticas en las pruebas de aplicaciones móviles, 2013).

Al realizar la evaluación del rendimiento se seleccionaron los parámetros: tiempo de respuesta, uso del CPU y consumo de memoria RAM¹⁷. Estos fueron evaluados en varios dispositivos móviles al acceder a la plataforma XILEMA AGORAV y consumir sus servicios, tanto desde la aplicación AGORAV como desde el navegador web. Con estos valores se realizó una comparación entre las dos formas de acceso. A continuación, se muestran específicamente los parámetros de evaluación:

- Tiempo de respuesta, uso del CUP y consumo de memoria RAM al entrar a la plataforma.
- Tiempo de respuesta, uso del CUP y consumo de memoria RAM al acceder a una publicación.
- Tiempo de respuesta, uso del CUP y consumo de memoria RAM al iniciar reproducción de un contenido audiovisual.
- Tiempo de respuesta, uso del CUP y consumo de memoria RAM durante la reproducción de un audiovisual.

Resultados de las pruebas de rendimiento

Las pruebas fueron ejecutadas en 5 dispositivos móviles con distintas propiedades, los tiempos de respuesta fueron tomados con un cronómetro y registrados en una tabla comparativa para su posterior análisis. Para comparar el uso del CPU y el consumo de memoria RAM se utilizó la herramienta *Android Monitor* del IDE

¹⁶ *Central Process Unit* (en español Unidad Central de Procesamiento).

¹⁷ *Random Access Memory* (en español Memoria de Acceso Aleatorio).

CAPÍTULO 3: Implementación y validación de la solución

Android Studio, que muestra en tiempo real el comportamiento de estos parámetros durante la ejecución de una aplicación específica.

En el parámetro Reproducir audiovisual, al aplicar la prueba a la aplicación nativa se observó un comportamiento similar en todos los dispositivos utilizados, el uso del CPU osciló entre el 13.3% y el 15%, el consumo de memoria RAM entre los 37 MB y 41 MB, y el tiempo de espera no varió observándose siempre 0.2 segundos. Al aplicar la prueba al navegador web se observó un aumento en los resultados de los parámetros, siendo aún más evidentes en los dispositivos de bajas propiedades. Al finalizar la prueba de rendimiento se observaron mejores tiempos de respuestas, además de un menor uso del CPU y consumo de memoria RAM al utilizar la aplicación nativa AGORAV con respecto al navegador web móvil.

A continuación, se muestra una tabla comparativa donde se evidencia el promedio de los resultados de las pruebas:

Tabla 6: Promedio de los resultados de las pruebas de rendimiento.

Parámetros	Tiempo de respuesta (segundos)		Uso del CPU (%)		Consumo de RAM (MB)	
	App Nativa	Nav. Web	App Nativa	Nav. Web	App Nativa	Nav. Web
Acceder a la plataforma	2.86	10.73	16.82	41.92	36.8	129.6
Acceder a una publicación	0.62	6.63	14.93	39.24	36.6	129.6
Iniciar reproducción de un contenido audiovisual	0.2	3.98	16.96	47.23	38.8	145.4
Reproducción de un contenido audiovisual	0.84	6.45	14.27	48.94	39.4	147.2

3.3.3. Pruebas de usabilidad

Las pruebas de usabilidad se refieren a la evaluación de un sistema realizada por usuarios. Durante una prueba, los participantes tratarán de ejecutar las funcionalidades del *software* mientras que los observadores ven, escuchan y toman notas (Usability.gov). El objetivo es identificar cualquier problema de usabilidad, recopilar datos cualitativos y cuantitativos, además de determinar la satisfacción del participante (Universidad de Colima, 2010).

Entrevista contextual

CAPÍTULO 3: Implementación y validación de la solución

Se utilizó la técnica Entrevista contextual, la cual consiste en observar y escuchar a un usuario mientras utiliza la aplicación, sin especificar el funcionamiento de la misma (Usability.gov). Se realizaron preguntas y se documentaron las opiniones de los usuarios, para luego identificar y rectificar las deficiencias de usabilidad y errores existentes. Al final se aplicó un cuestionario para evaluar la satisfacción del usuario.

Para ejecutar este tipo de pruebas se seleccionó una muestra de personas y se les pidió utilizar la aplicación, mientras se tomaban notas sobre las reacciones de los usuarios. Al terminar de usar la aplicación se les pidió a los participantes responder un cuestionario conformado por 6 preguntas cerradas con dos tipos de respuestas: dicotómicas¹⁸ y tricotómicas¹⁹. Todas las preguntas fueron elaboradas para determinar lo que los probadores opinan sobre la solución. En el Anexo no. 1, se muestra el cuestionario aplicado a los probadores.

Resultados de las pruebas de usabilidad

La prueba se efectuó en el laboratorio de producción, en un ambiente de desarrollo donde el probador utilizó la aplicación AGORAV sin ninguna explicación de su funcionamiento o guía de prueba. La prueba fue aplicada a 22 usuarios, esta muestra abarcó personas con diferente nivel de conocimiento del trabajo con dispositivos móviles con SO *Android*: trabajadores del proyecto AGORAV, estudiantes de diversos años de la facultad 6, profesores de la facultad 6 y técnicos de laboratorios. Durante la ejecución de las pruebas se documentaron algunas observaciones sobre el desempeño de los mismos al utilizar la aplicación. Al concluir, se aplicó un cuestionario donde los probadores respondieron en dependencia de sus experiencias al utilizar la aplicación.

Entre las observaciones más notables se encontraron:

- Dificultad al encontrar las opciones: Ajustes y Salir, que se encontraban al final del *drawer*. Esta deficiencia se solucionó agregando un botón nuevo a la derecha en la *ToolBar*²⁰ con estas dos opciones.

¹⁸ Respuestas de dos opciones. Por lo general sí o no.

¹⁹ Respuestas de tres opciones.

²⁰ Barra superior de la interfaz en la que se encuentran algunas opciones de la aplicación, como ajuste, buscar e inicio.

CAPÍTULO 3: Implementación y validación de la solución

- Al realizar una votación sobre una publicación, el usuario no podía ver su voto particular, solamente observaba el *rating*²¹ de la publicación y el número de votos. Esta deficiencia se solucionó modificando el *Ratingbar*²² para que se mantuviera su voto particular en esta y agregando dos atributos al componente de la vista que indicaran el *rating* de la publicación y la cantidad de votos.
- La única manera de cerrar la aplicación era a través del botón Salir. Esta deficiencia se solucionó agregando una opción que garantizara que al presionar el botón Atrás del dispositivo y no existiera una sección a la que retroceder, se mostrara el siguiente mensaje: “Presione nuevamente atrás para salir”, y al presionarlo por segunda vez la aplicación se cierre.

Los resultados arrojados por el cuestionario fueron positivos. A continuación, se muestra un análisis de los mismos por cada una de los elementos evaluados en el cuestionario:

- En cuanto al nivel de dificultad a la hora de navegar por la aplicación AGORAV el 86% de los probadores seleccionaron baja, el 14% seleccionaron media, mientras que ninguno seleccionó alta.
- En cuanto al diseño de la interfaz de usuario de la aplicación el 95% de los probadores seleccionaron que es sencillo, mientras que el 5% opinan que es complejo.
- En cuanto a la velocidad de navegación el 86% de los probadores opinan que la navegación es rápida, el 14% que es media, mientras que ninguno seleccionó que es lenta.
- En cuanto a la comparación de consumir los servicios brindados por la plataforma XILEMA AGORAV desde la aplicación y el navegador web móvil, el 100% opinó que acceder desde la aplicación AGORAV es mejor.
- De los probadores un 9% evidenció un sobrecalentamiento en el dispositivo al utilizar la aplicación AGORAV.
- El 100% de los probadores encuentra satisfactoria la experiencia de usuario al utilizar la aplicación AGORAV.

²¹ Evaluación promedio de todas las votaciones realizadas referentes a un contenido audiovisual específico.

²² Barra deslizante en forma de estrellas, en la que el usuario puede deslizar y definir una evaluación.

CAPÍTULO 3: Implementación y validación de la solución

3.3.4. Pruebas de integración

Las pruebas de integración son una técnica sistemática para construir la arquitectura del *software*. Son utilizadas para comprobar que los módulos o partes de un *software* funcionan correctamente como un todo. Existen dos formas de realizar la integración del *software*: de forma no incremental y de forma incremental. La forma no incremental no es recomendable debido al gran cúmulo de errores que puede generar (Pressman, 2010). Para la realización de este tipo de prueba se aplicó un enfoque incremental de integración ascendente.

Incremental ascendente

La prueba de integración ascendente se aplica con la construcción y prueba de módulos atómicos. Debido a que los componentes se integran de abajo hacia arriba, siempre está disponible el procesamiento requerido para los componentes subordinados a un determinado nivel (Pressman, 2010).

Esta prueba se aplicó en la medida que se avanzó en la implementación de la solución, pues una vez terminada la implementación de una funcionalidad del sistema, se procedió a probar su correcta ejecución en conjunto con la plataforma, comprobando así la integración de la funcionalidad con la misma. Al terminar la implementación de otra funcionalidad se repitió el procedimiento anterior, sumado con la comprobación de las funcionalidades ya implementadas. Este procedimiento se repitió a lo largo de todo el desarrollo de la solución hasta quedar implementadas todas las funcionalidades. Una vez conformada la solución se procedió a probar la aplicación de forma general para constatar su integración a la plataforma XILEMA AGORAV.

Resultados de las pruebas de integración

A medida que se fueron implementando las funcionalidades del sistema se fue probando su correcta ejecución con la plataforma XILEMA AGORAV. Durante estas pruebas iterativas se detectaron no conformidades tales como: al realizar un comentario en una publicación no fue posible ejecutar la funcionalidad, debido a que la aplicación no enviaba correctamente los datos a la plataforma mediante el servicio web; al iniciar sesión un usuario, el servicio web no retornaba los datos asociados al usuario autenticado.

Aplicar este tipo de prueba durante todo el proceso de implementación permitió corregir el mal comportamiento de las funcionalidades con respecto a su interacción con la plataforma. De esta manera al

CAPÍTULO 3: Implementación y validación de la solución

culminar la aplicación y ejecutar la última iteración de la prueba de integración se aseguró la correcta integración de la aplicación con la plataforma XILEMA AGORAV.

3.4. Conclusiones parciales

- En el modelo de implementación, la realización del diagrama de componentes permitió comprender mejor las dependencias en cuanto a librerías, apreciándose la interacción entre los componentes. El diagrama de despliegue ayudó a entender mejor la relación entre los componentes físicos que intervienen en el sistema final.
- Con la realización de las pruebas de caja negra e integración se encontraron no conformidades referentes a las funcionalidades de la aplicación, las cuales fueron erradicadas posteriormente. La realización de estas pruebas aportó positivamente a la calidad del *software*.
- La realización de las pruebas de rendimiento y usabilidad demostró que la aplicación AGORAV mejora el consumo de contenidos publicados en la plataforma XILEMA AGORAV desde dispositivos móviles con SO *Android*.

Conclusiones generales

Al concluir la presente investigación se obtienen resultados que dan cumplimiento satisfactorio al objetivo general planteado inicialmente. A partir de estos resultados se arriba a las siguientes conclusiones:

- Las herramientas y tecnologías seleccionadas permitieron la obtención de una amplia documentación, el desarrollo de una solución siguiendo estándares utilizados internacionalmente, además de sentar las bases para futuras versiones.
- El desarrollo de una aplicación nativa aumentó el rendimiento en los dispositivos móviles con sistema operativo *Android* al consumir los servicios brindados por la plataforma XILEMA AGORAV en comparación con un navegador web móvil.
- El desarrollo de la aplicación nativa AGORAV utilizando principios de usabilidad y enfocada a la experiencia de usuario mejoró el consumo de contenidos audiovisuales publicados en la plataforma XILEMA AGORAV.
- La realización de pruebas de Integración, Caja Negra, Usabilidad y Rendimiento demostró el correcto funcionamiento de la aplicación nativa AGORAV para dispositivos móviles con sistema operativo *Android*.

Recomendaciones

- Incrementar el número de funcionalidades a la aplicación AGORAV, ya que la plataforma brinda más funcionalidades que las implementadas en la solución.
- Agregar funcionalidades a la solución enfocadas a la interacción con las redes sociales: Facebook, Twitter e Instagram.

Referencias bibliográficas

Referencias bibliográficas

Aguaded Gómez, José Ignacio Aguaded. 2010. *Internet, una red para la información, la comunicación y la educación.* 2010.

Aguado Terrón, Juan Miguel y Martínez Martínez, Inmaculada J. 2009. *De la Web social al Móvil 2.0: el paradigma 2.0 en el proceso de convergencia mediática de la comunicación móvil.* s.l. : El profesional de la información, 2009.

Android OS. 2012. Historia Android OS. *Android OS.* [En línea] 2012. <http://androidos.readthedocs.org/en/latest/data/historia/>.

2015. Android Studio 2.0 Preview: el nuevo IDE construye las aplicaciones mucho más rápido. *XatakaAndroid.* [En línea] 23 de Noviembre de 2015. <http://www.xatakandroid.com/programacion-android/android-studio-2-0-preview-el-nuevo-ide-construye-las-aplicaciones-mucho-mas-rapido>.

Android Studio. 2016. Android Studio The Official IDE for Android. *Android Studio.* [En línea] 11 de mayo de 2016. <https://developer.android.com/studio/index.html?hl=es#features>.

Android Studio faq. 2014. *Android Studio faq.* [En línea] 2014. <http://androidstudiofaqs.com/conceptos/que-es-gradle-en-android-studio>.

Becerra, Tristán. 2011. Revisiones de código y estándares de codificación. *Revisiones de código y estándares de codificación.* [En línea] 2011. [Citado el: 24 de marzo de 2016.] <https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx>.

Brito Calahorro, Nacho. 2009. *Manual de desarrollo web con Grails.* España : s.n., 2009. 978-84-613-2651.

CCM. 2014. Lenguajes de programación. *High-Tech CCM.* [En línea] junio de 2014. <http://es.ccm.net/contents/304-lenguajes-de-programacion.pdf>.

Desarrollo de Aplicaciones para dispositivos Móviles sobre la plataforma Android de Google. **Tudela, Jaime Aranz. 2009.** Madrid : Universidad Carlos III, 2009.

Developers Android. 2015. Android Studio Overview. *Developers Android.* [En línea] 13 de diciembre de 2015. <http://android-studio.uptodown.com/>.

Evaluación de "Web Site" utilizando principios de Ingeniería de Usabilidad. **Valez Rubio, Miguel. 2015.** 2015.

Fowler. 2007. *UML distilled, A brief guide to standard object modeling lenguaje.* Tercera. s.l. : Pearson Education, 2007. ISBN: 978-8131-715-65-9.

Google. 2016. Google. *Material design.* [En línea] Google, 2016. <https://www.google.com/design/spec/material-design/introduction.html>.

Referencias bibliográficas

- Google Play. 2016.** Dailymotion. *Google Play*. [En línea] Google, 28 de enero de 2016. https://play.google.com/store/apps/details?id=com.dailymotion.dailymotion&hl=es_419.
- . **2016.** Hulu. *Google Play*. [En línea] Google, 21 de diciembre de 2016. https://play.google.com/store/apps/details?id=com.hulu.plus&hl=es_419.
- . **2016.** Vimeo. *Google Play*. [En línea] Google, 17 de enero de 2016. https://play.google.com/store/apps/details?id=com.vimeo.android.videoapp&hl=es_419.
- . **2016.** YouTube. *Google Play*. [En línea] Google, 12 de enero de 2016. https://play.google.com/store/apps/details?id=com.google.android.youtube&hl=es_419.
- Hassan Montero, Yusef y Martín Fernández, Francisco. 2005.** *La experiencia del usuario*. 2005.
- IBM developerWorks.** Introducción a SOA y servicios web. *IBM developerWorks*. [En línea] <http://www.ibm.com/developerworks/ssa/webservices/newto/service.html>.
- IBM. 2006.** IBM Corp Ayuda del Rational (Español). *Rational Unified Process Versión 7.0.1*. [En línea] 2006. [Citado el: 21 de Abril de 2014.] <http://www.ibm.com/developerworks/ssa/rational/newto/>.
- Jacobson, Ivar, Booch, G. y Rumbaugh, J. 2000.** *El proceso unificado de desarrollo de software*. Madrid : Pearson Edicación, 2000.
- Joyanes Aguilar, Luis. 1996.** *Programación Orientada a Objetos*. Madrid : s.n., 1996.
- Larman, Craig. 2005.** *Applying UML and patterns: an introduction to object-oriented analysis and design and iterative development*. s.l. : Pearson Education India, 2005.
- Lionbridge. 2012.** Mobile Web Apps vs. Mobile Native Apps: How to Make the Right Choice. *Lionbridge*. [En línea] 2012. http://www.lionbridge.com/files/2012/11/Lionbridge-WP_MobileApps2.pdf.
- López Ceballos, Alfonso Grabiél. 2012.** Pruebas Funcionales. *SlideShare*. [En línea] 26 de octubre de 2012. <http://es.slideshare.net/AlfonsoLopezCeballos/pruebas-funcionales>.
- Martínez Zurita, Alberto. 2013.** Diagrama de despliegue. *SlideShare*. [En línea] 9 de marzo de 2013. <http://es.slideshare.net/albertozurita96/diagrama-de-despliegue-17071673>.
- Mejores prácticas en las pruebas de aplicaciones móviles.* **Díaz, Santiago. 2013.** Madrid : atSistemas, 2013.
- Modelado, Diseño e Implementación de servicios web.* **Navarro Maset, Rafael. 2007.** 2007.
- Morillo Pozo, Julián David. 2011.** *Introducción a los dispositivos móviles*. s.l. : Universitat Oberta de Catalunya, 2011.
- Nielsen, Jacob. 1994.** *Usability engineering*. s.l. : Elsevier, 1994.

Referencias bibliográficas

- Oracle Corporation. 2016.** ¿Cómo puedo empezar a desarrollar programas Java con Java Development Kit (JDK)? *Java*. [En línea] 2016. <https://www.java.com/es/download/faq/develop.xml>.
- Pavón Pulido, Nieves. 2013.** *Componentes básicos de Android*. 2013.
- Plataforma VideoWeb, televisión en la web.* **Torres Rodríguez, Frank, y otros. 2013.** La Habana : XVII Fórum de Ciencia y Técnica, 2013.
- Pressman, Roger S. 2010.** *Software engineering: a practitioner's approach - 7th ed.* s.l. : Palgrave Macmillan, 2010.
- Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con Android.*
Larramendi Ferras, Claudia Beatriz. 2012. La Habana : s.n., 2012.
- Rámirez García, Higinio. 2014.** Análisis de sistemas operativos smartphones. 2014.
- Rao, Jinghai, Su, Xiaomeng. 2004.** *A survey of automated web service composition methods.* s.l. : Springer, 2004.
- Reynoso, Carlos Billy. 2004.** *Introducción a la Arquitectura de Software.* s.l. : Universidad de Buenos Aires, 2004.
- Rodríguez Sanchez, Tamara. 2014.** *Metodología de desarrollo para la actividad productiva de la UCI.* La Habana : Universidad de las Ciencias Informáticas, 2014.
- Ruano, Soledad. 2010.** *Internet y la telefonía móvil, nuevos soportes para distribuir contenidos audiovisuales.* 2010.
- Schildt, Herbert. 2003.** *Manual de referencia Java 2 4ta Edición.* 2003.
- Sommerville, Ian. 2005.** *Ingeniería de Software 7ma. Ed.* Madrid : Pearson Education, 2005. 84-7829-074-5.
- Subirats, Joan. 2014.** Qué es y para qué sirve el SDK. *fandroides*. [En línea] 3 de abril de 2014.
<http://www.fandroides.com/que-es-y-para-que-sirve-el-sdk/>.
- Universidad de Colima. 2010.** Pruebas de Usabilidad. *SlideShare*. [En línea] 16 de febrero de 2010.
<http://es.slideshare.net/guest041f5db/pruebas-de-usabilidad-3199194>.
- Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI.* **Curbera, Francisco and Duftler, Matthew and Khalaf, Rania and Nagy, William and Mukhi, Nirmal and Weerawarana, Sanjiva. 2002.** s.l. : IEEE Computer Society, 2002, Vol. 6.
- Usability.gov. 2015.** Contextual Interview. *Usability.gov Improving the User Experience*. [En línea] Usability.gov, 2015. <http://www.usability.gov/how-to-and-tools/methods/usability-testing.html>.
- . Usability Testing. *Usability.gov Improving the User Experience*. [En línea] Usability.gov.
<http://www.usability.gov/how-to-and-tools/methods/usability-testing.html>.
- User Experience Professionals Association. 2014.** Definitions of User Experience and Usability. *User Experience Professionals Association*. [En línea] 3 de marzo de 2014. <https://uxpa.org/resources/definitions-user-experience-and-usability>.

Referencias bibliográficas

Visual Paradigm. 2016. Visual Paradigm. *Visual Paradigm*. [En línea] 2016. <http://www.visual-paradigm.com>.

Bibliografía

Aguaded Gómez, José Ignacio Aguaded. 2010. *Internet, una red para la información, la comunicación y la educación.* 2010.

Aguado Terrón, Juan Miguel y Martínez Martínez, Inmaculada J. 2009. *De la Web social al Móvil 2.0: el paradigma 2.0 en el proceso de convergencia mediática de la comunicación móvil.* s.l. : El profesional de la información, 2009.

Android OS. 2012. Historia Android OS. *Android OS.* [En línea] 2012. <http://androidos.readthedocs.org/en/latest/data/historia/>.

2015. Android Studio 2.0 Preview: el nuevo IDE construye las aplicaciones mucho más rápido. *XatakaAndroid.* [En línea] 23 de Noviembre de 2015. <http://www.xatakandroid.com/programacion-android/android-studio-2-0-preview-el-nuevo-ide-construye-las-aplicaciones-mucho-mas-rapido>.

Android Studio. 2016. Android Studio The Official IDE for Android. *Android Studio.* [En línea] 11 de mayo de 2016. <https://developer.android.com/studio/index.html?hl=es#features>.

Android Studio faq. 2014. *Android Studio faq.* [En línea] 2014. <http://androidstudiofaqs.com/conceptos/que-es-gradle-en-android-studio>.

Becerra, Tristán. 2011. Revisiones de código y estándares de codificación. *Revisiones de código y estándares de codificación.* [En línea] 2011. [Citado el: 24 de marzo de 2016.] <https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx>.

Brito Calahorro, Nacho. 2009. *Manual de desarrollo web con Grails.* España : s.n., 2009. 978-84-613-2651.

CCM. 2014. Lenguajes de programación. *High-Tech CCM.* [En línea] junio de 2014. <http://es.ccm.net/contents/304-lenguajes-de-programacion.pdf>.

Desarrollo de Aplicaciones para dispositivos Móviles sobre la plataforma Android de Google. **Tudela, Jaime Aranaz. 2009.** Madrid : Universidad Carlos III, 2009.

Developers Android. 2015. Android Studio Overview. *Developers Android.* [En línea] 13 de diciembre de 2015. <http://android-studio.uptodown.com/>.

Evaluación de "Web Site" utilizando principios de Ingeniería de Usabilidad. **Valez Rubio, Miguel. 2015.** 2015.

Fowler. 2007. *UML distilled, A brief guide to standard object modeling lenguaje.* Tercera. s.l. : Pearson Education, 2007. ISBN: 978-8131-715-65-9.

Google. 2016. Google. *Material design.* [En línea] Google, 2016. <https://www.google.com/design/spec/material-design/introduction.html>.

Bibliografía

- Google Play. 2016.** Dailymotion. *Google Play*. [En línea] Google, 28 de enero de 2016. https://play.google.com/store/apps/details?id=com.dailymotion.dailymotion&hl=es_419.
- . **2016.** Hulu. *Google Play*. [En línea] Google, 21 de diciembre de 2016. https://play.google.com/store/apps/details?id=com.hulu.plus&hl=es_419.
- . **2016.** Vimeo. *Google Play*. [En línea] Google, 17 de enero de 2016. https://play.google.com/store/apps/details?id=com.vimeo.android.videoapp&hl=es_419.
- . **2016.** YouTube. *Google Play*. [En línea] Google, 12 de enero de 2016. https://play.google.com/store/apps/details?id=com.google.android.youtube&hl=es_419.
- Hassan Montero, Yusef y Martín Fernández, Francisco. 2005.** *La experiencia del usuario*. 2005.
- IBM developerWorks.** Introducción a SOA y servicios web. *IBM developerWorks*. [En línea] <http://www.ibm.com/developerworks/ssa/webservices/newto/service.html>.
- IBM. 2006.** IBM Corp Ayuda del Rational (Español). *Rational Unified Process Versión 7.0.1*. [En línea] 2006. [Citado el: 21 de Abril de 2014.] <http://www.ibm.com/developerworks/ssa/rational/newto/>.
- Jacobson, Ivar, Booch, G. y Rumbaugh, J. 2000.** *El proceso unificado de desarrollo de software*. Madrid : Pearson Edicación, 2000.
- Joyanes Aguilar, Luis. 1996.** *Programación Orientada a Objetos*. Madrid : s.n., 1996.
- Larman, Craig. 2005.** *Applying UML and patterns: an introduction to object-oriented analysis and design and iterative development*. s.l. : Pearson Education India, 2005.
- Lionbridge. 2012.** Mobile Web Apps vs. Mobile Native Apps: How to Make the Right Choice. *Lionbridge*. [En línea] 2012. http://www.lionbridge.com/files/2012/11/Lionbridge-WP_MobileApps2.pdf.
- López Ceballos, Alfonso Grabiél. 2012.** Pruebas Funcionales. *SlideShare*. [En línea] 26 de octubre de 2012. <http://es.slideshare.net/AlfonsoLopezCeballos/pruebas-funcionales>.
- Martínez Zurita, Alberto. 2013.** Diagrama de despliegue. *SlideShare*. [En línea] 9 de marzo de 2013. <http://es.slideshare.net/albertozurita96/diagrama-de-despliegue-17071673>.
- Mejores prácticas en las pruebas de aplicaciones móviles.* **Díaz, Santiago. 2013.** Madrid : atSistemas, 2013.
- Modelado, Diseño e Implementación de servicios web.* **Navarro Maset, Rafael. 2007.** 2007.
- Morillo Pozo, Julián David. 2011.** *Introducción a los dispositivos móviles*. s.l. : Universitat Oberta de Catalunya, 2011.
- Nielsen, Jacob. 1994.** *Usability engineering*. s.l. : Elsevier, 1994.

Bibliografía

- Oracle Corporation. 2016.** ¿Cómo puedo empezar a desarrollar programas Java con Java Development Kit (JDK)? *Java*. [En línea] 2016. <https://www.java.com/es/download/faq/develop.xml>.
- Pavón Pulido, Nieves. 2013.** *Componentes básicos de Android*. 2013.
- Plataforma VideoWeb, televisión en la web. Torres Rodríguez, Frank, y otros. 2013.* La Habana : XVII Fórum de Ciencia y Técnica, 2013.
- Pressman, Roger S. 2010.** *Software engineering: a practitioner's approach - 7th ed.* s.l. : Palgrave Macmillan, 2010.
- Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con Android.*
- Larramendi Ferras, Claudia Beatriz. 2012.** La Habana : s.n., 2012.
- Rámirez García, Higinio. 2014.** Análisis de sistemas operativos smartphones. 2014.
- Rao, Jinghai, Su, Xiaomeng. 2004.** *A survey of automated web service composition methods.* s.l. : Springer, 2004.
- Reynoso, Carlos Billy. 2004.** *Introducción a la Arquitectura de Software.* s.l. : Universidad de Buenos Aires, 2004.
- Rodríguez Sanchez, Tamara. 2014.** *Metodología de desarrollo para la actividad productiva de la UCI.* La Habana : Universidad de las Ciencias Informáticas, 2014.
- Ruano, Soledad. 2010.** *Internet y la telefonía móvil, nuevos soportes para distribuir contenidos audiovisuales.* 2010.
- Schildt, Herbert. 2003.** *Manual de referencia Java 2 4ta Edición.* 2003.
- Sommerville, Ian. 2005.** *Ingeniería de Software 7ma. Ed.* Madrid : Pearson Education, 2005. 84-7829-074-5.
- Subirats, Joan. 2014.** Qué es y para qué sirve el SDK. *fandroides*. [En línea] 3 de abril de 2014. <http://www.fandroides.com/que-es-y-para-que-sirve-el-sdk/>.
- Universidad de Colima. 2010.** Pruebas de Usabilidad. *SlideShare*. [En línea] 16 de febrero de 2010. <http://es.slideshare.net/guest041f5db/pruebas-de-usabilidad-3199194>.
- Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. Curbera, Francisco and Duftler, Matthew and Khalaf, Rania and Nagy, William and Mukhi, Nirmal and Weerawarana, Sanjiva. 2002.* s.l. : IEEE Computer Society, 2002, Vol. 6.
- Usability.gov. 2015.** Contextual Interview. *Usability.gov Improving the User Experience*. [En línea] Usability.gov, 2015. <http://www.usability.gov/how-to-and-tools/methods/usability-testing.html>.
- . Usability Testing. *Usability.gov Improving the User Experience* . [En línea] Usability.gov. <http://www.usability.gov/how-to-and-tools/methods/usability-testing.html>.
- User Experience Professionals Association. 2014.** Definitions of User Experience and Usability. *User Experience Professionals Association*. [En línea] 3 de marzo de 2014. <https://uxpa.org/resources/definitions-user-experience-and-usability>.

Bibliografía

Visual Paradigm. 2016. Visual Paradigm. *Visual Paradigm*. [En línea] 2016. <http://www.visual-paradigm.com>.

Anexos

Anexo no.1 Prueba de usabilidad

Entrevista Contextual

El presente cuestionario tiene como objetivo recoger las impresiones de los usuarios al utilizar e interactuar con la aplicación AGORAV para dispositivos móviles con sistema operativo Android.

1. ¿Qué nivel de dificultad encuentra a la hora de navegar por la aplicación AGORAV?
 Alta Media Baja

2. ¿Cómo encuentra el diseño de la interfaz de usuario de la aplicación AGORAV?
 Complejo Sencillo

3. ¿Cómo encuentra la velocidad de navegación?
 Rápida Media Lenta

4. ¿En comparación con el acceso por navegador web, cómo encuentra el consumo de contenidos audiovisuales desde la aplicación AGORAV?
 Mejor Igual Peor

5. ¿Evidencia sobrecalentamiento del dispositivo al utilizar la aplicación AGORAV?
 Si No

6. Teniendo en cuenta que la experiencia de usuario es la sensación, sentimiento, respuesta emocional, valoración y satisfacción del usuario respecto a un producto, resultado del fenómeno de interacción con el producto. ¿Encuentra satisfactoria la experiencia de usuario al interactuar con la aplicación AGORAV?
 Si No