

Universidad de las Ciencias Informáticas
Facultad de Ciencias y Tecnologías Computacionales



Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

**“Subsistema de Gestión de Canales Virtuales para el
Sistema de Transmisión de Canales Virtuales v2.0”**

Autores:

Yudeisy Valdés Fernández
Osmín García Prieto

Tutores:

MSc. Rafael Leodán Cardero Álvarez
Ing. Vidal Orlando Acosta García

Cotutor:

Ing. Vladimir Milán Núñez

La Habana, 21 de junio del 2017

“Año 59 de la Revolución”



“Aquellos que están locos como para pensar que pueden cambiar el mundo, son aquellos que lo hacen.”

Steven Paul Jobs

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores del trabajo **Subsistema de Gestión de Canales Virtuales para el Sistema de Transmisión de Canales Virtuales v2.0** y se autoriza a la Universidad de las Ciencias Informáticas a hacer el uso que estime pertinente con el mismo.

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

Firma del autor

Yudeisy Valdés Fernández

Firma del autor

Osmín García Prieto

Firma del Tutor

MSc. Rafael Leodán Cardero Álvarez

Firma del Tutor

Ing. Vidal Orlando Acosta García

Firma del Cotutor

Ing. Vladimir Milán Núñez

DEDICATORIA

“A mis padres, mi hermana, mi pequeño ángel Sofía y mis abuelos”.

Yudeisy Valdés Fernández.

A mis abuelos, por ser la luz que guía mi vida y ser mi mayor motivo de inspiración.

A mis padres, por su apoyo y amor incondicional.

A toda mi familia y amigos por ser parte importante de mi vida.

Osmín García Prieto.

AGRADECIMIENTOS

“A mis familiares, en especial a mis padres que les doy gracias por haberme dado la vida que sin ellos no estuviese hoy aquí. Mami yo sé que te he dado mucho que hacer en estos 25 años gracias por aguantarme y malcriarme tanto y también por hacerme tan feliz todo lo que he vivido hasta ahora. Mi papi lindo tengo mucho que decirte sin que mi mami se ponga celosa yo te quiero cantidad, siempre me diste tu hombro para llorar en los momentos difíciles y también me sacaste sonrisas en los momentos alegres, aunque sé que en el mundo todos tienen un papá bueno el mío es el mejor”.

“A mi hermana que ha sido más que una hermana, una amiga, una madre, mi pilar, mi soporte, y que cuando necesite levantarme siempre estuvo allí dándome la mano y sacudiéndome el polvo”.

“A mi sobrina linda Sofía que es el ángel de mi vida y lo que más yo quiero en este mundo porque siendo alguien tan pequeña me pudo robar mi corazón”.

“A mis abuelos que fueron mis ángeles protectores y mis guías en encaminarme en la vida, a mi novio que me ha aguantado tanto mal estrés y me enseñó a sonreír en los momentos más difíciles secándome las lágrimas”.

“A mis tíos, a mis suegros que siempre me trataron como una hija más, a mis cuñados que son lo mejor, y en especial al Ber que me ha tratado siempre como se hermana pequeña”.

“A Vladi y Yanet que fueron dos personas excepcionales que aunque el tiempo no les alcanzaba y se les tornaba difícil las circunstancias siempre estuvieron allí para mí, gracias por ser mis amigos”.

“A mis tutores, que siempre estuvieron ahí cuando tenía dudas, por ser perfeccionistas con cada detalle y hacerme dar lo mejor de mí en cada momento”.

“A mi compañero de tesis, que me aguantó un montón de peleas, estrés, lágrimas, sonrisas y me trató como su hermana, y así marchando unidos pudimos llegar a este resultado”.

“A mis amigos que son muchos y la verdad no me gustaría dejar a nadie fuera, pero en especial a mi piquete José Joel, Odiel, Dieter, Arletis, Diannet, Suse, Pabel, Mary, Lázaro, Yanet, Rolando, Yadira, Arian, William, Cristina quiero decirle a todos que demostraron ser buenos amigos”.

“A mis compañeros y amigos de la facultad 2 en especial a Bienvenido, Joangel, Manchón, Lourdes, Glenda, Marian, Yeny, Yunio”.

“A Yeny y Sol por estar ahí a regañadientes siempre con nosotros, pero demostrándonos su apoyo incondicional y mostrándonos que siempre podemos ser mejores”.

“A mis amigos y compañeros de trabajo de la FEU, en especial a Yoelvis”.

“A la profe Ailec, y todos los profes que ofrecieron su apoyo incondicional”.

“A todo aquel que contribuyó con mi formación, en especial a los profesores de esta casa de altos estudios”.

“A la FEU y la UJC, que me quitó tiempo pero me enseñó a aprender y abrirme camino en la vida, por lo que ahora ganaré mucho más”.

“A Fidel Castro Ruz y a la Revolución, que me brindó la posibilidad de estudiar en esta casa de altos estudios y ser hoy Ingeniera en Ciencias Informáticas”.

Yudeisy Valdés Fernández

Durante estos cinco años en la universidad he vivido experiencias únicas, unas muy buenas y unas pocas no tanto. Hoy quiero agradecer aquellas personas que me han dado su apoyo en el transcurso de mi formación.

A mi mamá, por ser mi novia en todo momento y saber aliviar mis más amargos días con su amor y sus mimos.

A mi papá, por la confianza que siempre deposita en mí, por sus regaños muy comunes, que me hacen aprender de mis errores, por su amor incondicional, por sus cálidos abrazos que me hacen sentirme seguro de mí mismo y por su apoyo en todos los momentos de mi vida.

A mi tío Manolito, por ser mi otro padre, mi hermano y mi amigo, por compartir momentos importantes de mi juventud y sobre todo por su sentido del humor.

A mis abuelos María, José Manuel, Justa y Juan por brindarme todas sus experiencias, por su amor en todo momento y sobre todo por sus mimos.

A mi hermano Idiel, que me ha enseñado que lo que hace a un hombre grande no son sus errores, sino lo que hacemos para sobrepasarlos y aprender de ellos.

A mi tía Amarilis, por ser esa persona que en los días más amargos te sacan una sorprendente sonrisa.

A mi prima Maridalis, por ser mi amiga y sobre todo mi hermana, por sus consejos, que, aunque fuertes, ciertos.

A la nueva integrante de la familia Idielis, mi sobrina del alma que tanto quiero.

A Rigoberto Pita Medina, por ser ese profesor y amigo que tantos valores aportó a mi formación personal y profesional.

A mis mejores amigos Roly y Alexander, esos que están ahí en los momentos difíciles y que te brindan su ayuda sin pedir nada a cambio.

A mi cotutor Vladimir, por demostrarme que existen personas que brindan su ayuda, incluso sin conocerte y sobre todo por sus grandes consejos, gracias Vlado.

A Yenisel y a Solange por su apoyo y ayuda incondicional.

A Yanet Parra, por demostrarme que no existen obstáculos, cuando se quiere ayudar a otra persona.

A mis demás tutores Vidal y Rafael, por apoyarnos y ayudarnos en el transcurso de la investigación.

A todos mis amigos, esos locos que en los días más amargos saca lo mejor de ti con sus locuras y jodederas.

A todos aquellos que brindaron su granito de arena en la presente investigación.

A mi compañera de tesis, por los momentos alegres, por esas comidas para ahogar el estrés, por los dolores de cabeza que en ocasiones me daba y sobre todo por compartir todos estos momentos de gran importancia en mi carrera.

Osmín García Prieto.

RESUMEN

La gestión de canales virtuales es un proceso necesario para el procesamiento de señales digitales debido al alto consumo de materiales audiovisuales que existe en la actualidad. Este proceso tiene irregularidades, debido a que no existe una descripción detallada de los espacios televisivos en la planificación de canales. Además, no se pueden realizar cambios en tiempo real en la parrilla de programación y la periodicidad de la planificación es limitada. La investigación realizada permitió desarrollar un Subsistema de Gestión de Canales Virtuales para el Sistema de Transmisión de Canales Virtuales. Dicho subsistema permite que la periodicidad de la planificación de las parrillas de programación sea mayor a una semana. Además, facilita a los usuarios poder mantenerse actualizados de los últimos cambios que se realicen en la programación de los canales a transmitir. La solución desarrollada es capaz de gestionar la programación audiovisual la cual brinda una importante fuente de abastecimiento de información para los usuarios. Para lograr el resultado esperado se utilizó las tecnologías y lenguajes: HTML5, CSS3, JavaScript (del lado del cliente), PHP5 (PHP del lado del servidor) y como framework de desarrollo Symfony. También se utilizó como metodología de software AUP-UCI en el escenario 2; y se realizaron pruebas de caja negra y caja blanca, demostrando que el subsistema está apto para ser utilizado.

Palabras Claves: audiovisuales, canales virtuales, parrillas de programación, planificación.

ABSTRACT

The management of virtual channels is a necessary process for the processing of digital signals due to the high consumption of audiovisual materials that exists today. This process has irregularities, because there is no detailed description of the television spaces in channel planning. In addition, real-time changes cannot be made to the scheduling grid and the scheduling periodicity is limited. The research carried out allowed the development of a Subsystem of Management of Virtual Channels for the System of Transmission of Virtual Channels. Said subsystem allows the scheduling schedule of the programming grids to be greater than one week. In addition, it facilitates the users to be able to keep up to date of the last changes that are made in the programming of the channels to be transmitted. The solution developed is capable of managing audiovisual programming, which provides an important source of information for users. To achieve the expected result we used the technologies and languages: HTML5, CSS3, JavaScript (client side), PHP5 (PHP server side) and as Symphony development framework. AUP-UCI software methodology was also used in scenario 2; and tests of black box and white box were applied, demonstrating that the subsystem is suitable to be used.

Keywords: *audiovisuals, planning, programming grids, virtual channels.*

ÍNDICE GENERAL

| | |
|---|----|
| Introducción | 1 |
| Capítulo 1. Fundamentación teórica | 5 |
| 1.1 Elementos asociados al dominio del problema. | 5 |
| 1.1.1 Canal virtual | 5 |
| 1.1.2 Parrilla de programación de televisión..... | 6 |
| 1.1.3 Espacio Televisivo | 6 |
| 1.2 Estudio de las soluciones existentes..... | 7 |
| 1.2.1 Resultados del análisis de las soluciones existentes..... | 8 |
| 1.3 Metodología, herramientas y tecnologías a utilizar para el desarrollo del módulo..... | 9 |
| 1.3.1 Metodología de desarrollo de software AUP-UCI. | 9 |
| 1.3.2 Lenguaje de Modelado Unificado. | 10 |
| 1.3.3 Herramienta CASE..... | 10 |
| 1.3.4 Framework Symfony | 11 |
| 1.3.5 Entorno de Desarrollo Integrado..... | 12 |
| 1.3.6 Lenguajes de programación | 13 |
| Conclusiones parciales | 15 |
| Capítulo 2. Análisis y diseño del Subsistema de Gestión de Canales Virtuales. | 16 |
| 2.1 Modelo de dominio. | 16 |
| 2.1.1 <i>Modelo de Dominio</i> | 16 |
| 2.1.2 <i>Descripción de las clases del Modelo de Dominio</i> | 17 |
| 2.2 Especificación de los requisitos. | 17 |
| 2.2.1 <i>Requisitos funcionales</i> | 17 |
| 2.2.2 <i>Requisitos no funcionales</i> | 19 |
| 2.3 Descripción del Subsistema de Gestión de Canales Virtuales. | 20 |
| 2.3.1 Definición de los actores. | 20 |

| | | |
|---------|--|-----------|
| 2.3.2 | Diagrama de casos de uso del sistema. | 20 |
| 2.3.3 | Descripción de casos de uso del subsistema. | 21 |
| 2.4 | Patrón arquitectónico. | 27 |
| 2.5 | Patrones de diseño de software utilizados para el Subsistema de Gestión de Canales Virtuales. | 28 |
| 2.5.1 | Patrones para la asignación de responsabilidades (GRASP). | 28 |
| 2.5.2 | Patrones GOF. | 29 |
| 2.5.2.1 | Patrón creacional. | 30 |
| 2.5.2.2 | Patrón estructural. | 30 |
| 2.5.2.3 | Patrón de comportamiento. | 30 |
| 2.6 | Diagrama de clases del diseño. | 30 |
| 2.7 | Modelo físico de la Base de Datos. | 31 |
| | Conclusiones parciales. | 32 |
| | Capítulo 3. Implementación y Pruebas. | 33 |
| 3.1 | Diagrama de componentes. | 33 |
| 3.2 | Estándares de codificación. | 34 |
| 3.3 | Modelo de despliegue. | 35 |
| 3.4.1 | <i>Descripción de los componentes físicos del Modelo de Despliegue.</i> | 35 |
| 3.4 | Modelo de Pruebas. | 36 |
| 3.5 | Aplicación y resultado de las pruebas. | 37 |
| | Conclusiones Parciales. | 44 |
| | Conclusiones generales. | 45 |
| | Referencias Bibliográficas. | 46 |
| | Anexos. | 54 |
| | Anexos 1: Notificación de aceptación en evento. | 54 |
| | Anexos 2: Avaluos otorgados en eventos científicos. | 56 |
| | Anexos 3: Diagramas de clases del diseño. | 57 |
| | Anexos 4: Diagramas de secuencia. | 58 |

ÍNDICE DE FIGURAS

| | |
|---|----|
| Figura 1 Modelo de dominio..... | 16 |
| Figura 2 Diagrama de Casos de uso del sistema..... | 21 |
| Figura 3 Patrón arquitectónico..... | 28 |
| Figura 4 Diagrama de clases del diseño CU Gestionar canal..... | 31 |
| Figura 5 Modelo físico de la Base de Datos CU Gestionar Canal..... | 31 |
| Figura 6 Diagrama de componentes del Subsistema de Gestión Canales Virtuales..... | 33 |
| Figura 8 Modelo de despliegue..... | 35 |
| Figura 9 Gráfico de resultado de las no conformidades..... | 40 |
| Figura 10 Grafo de flujo asociado al código fuente..... | 42 |
| Figura 11 Diagrama de clases del diseño del Caso de Uso Gestionar Usuario..... | 57 |
| Figura 12 Diagrama de clases del diseño del Caso de Uso Asignar Roles..... | 57 |
| Figura 13 Diagrama de Secuencia CU Gestionar Canal escenario Listar Canal..... | 58 |
| Figura 14 Diagrama de Secuencia CU Gestionar Canal escenario Adicionar Canal..... | 58 |
| Figura 15 Diagrama de Secuencia CU Gestionar Canal escenario Modificar Canal..... | 59 |
| Figura 16 Diagrama de Secuencia CU Gestionar Canal escenario Eliminar Canal..... | 59 |
| Figura 17 Diagrama de Secuencia CU Gestionar Canal escenario Buscar Canal..... | 60 |

ÍNDICE DE TABLAS

| | |
|---|--------------------------------------|
| Tabla 1 Resumen del estudio de sistemas similares..... | 8 |
| Tabla 2 Descripción de los Actores del Sistema. | 20 |
| Tabla 3 Agrupación de requisitos por caso de uso..... | 21 |
| Tabla 4 Descripción del caso de uso: Gestionar canal televisivo en el Subsistema de Gestión de Canales Virtuales..... | 22 |
| Tabla 5 Descripción de las variables del caso de prueba del caso de uso Gestionar canal. | 37 |
| Tabla 6 . Descripción del Caso de Prueba para el Caso de Uso: Gestionar canal. Sección: Adicionar canal. | 38 |
| Tabla 7 Trayectoria básica obtenida en el grafo..... | 42 |
| Tabla 8 Caso de Prueba para la Trayectoria Básica 1. | 42 |
| Tabla 9 Caso de Prueba para la Trayectoria Básica 2 | 42 |
| Tabla 10 Descripción del Caso de Uso Gestionar usuario en el Subsistema de Gestión de Canales Virtuales. | ¡Error! Marcador no definido. |
| Tabla 11 Descripción del Caso de Uso Gestionar rol en el Subsistema de Gestión de Canales Virtuales. | ¡Error! Marcador no definido. |

Introducción

Con el transcurso de los años el hombre ha desarrollado diversas formas de comunicarse, apegado a la evolución de las tecnologías y la gran necesidad de incrementar la difusión de la información de una forma rápida y segura. Hoy en día la televisión es uno de los medios indispensables para la sociedad, ya que constituye uno de los avances de la revolución científico-tecnológica. Los sectores empresariales, de turismo, la salud y educación han utilizado la televisión como medio primordial para difundir la información en la sociedad actual, por lo que se ha vuelto necesario en el mundo la gestión de canales virtuales que se usan para transmitir la información. La gestión de canales virtuales ha ido evolucionando en conjunto con los cambios en la televisión por lo que se han desarrollado sistemas informáticos para facilitar la gestión de canales virtuales.

En Cuba los sectores empresariales hacen uso de la televisión para difundir información al cliente con el fin de mantenerlo informado; en el caso del turismo se usa la televisión por cable para la transmisión de la cultura y recreación de sus huéspedes. En el sector de la salud y educación es usada para la formación de valores y el desarrollo de conocimientos; y en el sector del transporte para mantener informados a sus pasajeros de los horarios de rutas de vuelos, barcos u ómnibus. También es usada para hacer llegar una sana recreación a los hogares y mantener al pueblo informado en cuanto acontece en el mundo y en nuestro país. (Rascón, 2010)

La Universidad de las Ciencias Informáticas (UCI) es una de las universidades del país que cuenta con un área encargada para la televisión que pertenece a la Dirección General de Comunicación Institucional. La UCI utiliza la televisión por cable para hacer llegar a la comunidad universitaria la información; ya sea tanto para la formación docente como para el entretenimiento y recreación. Con el objetivo de automatizar los procesos de planificación, transmisión y monitorización de canales virtuales que se ejecutan en una entidad dedicada a la gestión y transmisión de contenidos audiovisuales, la UCI cuenta con un Sistema de Transmisión de Canales Virtuales (STCV). Este sistema se desarrolló en el Centro de Desarrollo de Geoinformática y Señales Digitales (GEYSED), perteneciente a la Facultad de Ciencias y Tecnologías Computacionales.

El STCV está compuesto por un subsistema que utiliza tecnología streaming ¹para la transmisión de

¹ Streaming: es la distribución digital de contenido multimedia a través de una red de computadoras, de manera que el usuario utiliza el producto a la vez que se descarga. La palabra *retransmisión* se refiere a una corriente continua que fluye sin interrupción, y habitualmente a la difusión de audio o vídeo. (Internet, Televisión y Convergencia: nuevas pantallas y plataformas , 2012)

audiovisuales, y un subsistema para la monitorización de las transmisiones. Además, de un subsistema interactivo el cual permite al usuario consultar la cartelera de la programación, visualizar videos en vivo o bajo demanda, y tiene la opción de pausar y reiniciar una transmisión en vivo. También contiene un subsistema de programación encargado de realizar las planificaciones y gestionar los canales virtuales. (Farías, y otros, 2013)

El Subsistema de Programación permite la autenticación de cualquiera de sus usuarios del sistema mostrándole la información de los canales; lo que afecta la confidencialidad de la información de los canales y su programación, ya que cualquier usuario del sistema tiene acceso al mismo. También afecta la integridad de la información, ya que pueden realizar cambios no deseados en la planificación, lo que trae como consecuencia que el televidente que recibe el servicio le llegue la información que no es la esperada porque no se realiza una correcta gestión de roles.

La elaboración de la parrilla de la programación de los canales está restringida a siete días, por lo que se ve limitada a realizarse semanal, e impide la periodicidad de las planificaciones que realiza el planificador no pudiendo hacerlas cuando estime conveniente. En caso de que pueda ocurrir alguna interrupción en la transmisión, el subsistema no permite realizar cambios en la planificación de la programación actual en tiempo real por lo que el usuario no puede visualizar los cambios realizados en último instante durante la transmisión del canal. Además, el usuario no puede recibir la información de cómo quedó la cartelera ni tampoco tiene conocimiento de los materiales audiovisuales que serán transmitidos trayendo consigo que irrumpa con su planificación personal.

Otra dificultad que tiene el subsistema es la distorsión de las planificaciones existentes al no contar con un control sobre los tiempos disponibles que poseen los canales para estar en transmisión. Esto permite a los especialistas planificar más archivos multimedia de los que pueden ser transmitidos según el tiempo del canal. Además, provoca que se corran los horarios de la programación y los usuarios no reciban los materiales audiovisuales en los horarios fijados por cada espacio televisivo, ya que si estaba planificado el horario de la película para las 2:00 pm lo reciben a las 3:00 pm.

A partir de lo planteado se define como el **problema de la investigación**: ¿Cómo contribuir a la mejora de los procesos de gestión de canales virtuales en el STCV?

El **objeto de estudio** de la investigación es “los procesos de gestión de canales virtuales”, enmarcado en el **campo de acción**, “los procesos de gestión de canales virtuales en el Sistema de Transmisión de Canales Virtuales”.

Para dar solución al problema planteado se define como **objetivo general** “Desarrollar un Subsistema de

Gestión de Canales Virtuales para el Sistema de Transmisión de Canales Virtuales v2.0”.

El objetivo propuesto indujo a formular, como guía para el desarrollo de la investigación, las siguientes **preguntas científicas**:

- ¿Cuáles son los elementos fundamentales asociados al proceso de gestión de canales virtuales?
- ¿Cuáles son los diferentes sistemas que planifican y gestionan canales virtuales?
- ¿Cómo implementar un Subsistema de Gestión de Canales Virtuales que permita que la periodicidad de la planificación sea mayor a una semana, se puedan realizar cambios en tiempo real en la parrilla de programación y exista una descripción detallada de los espacios televisivos en la planificación de los canales?
- ¿Qué resultados se obtienen al aplicar técnicas y/o métodos ingenieriles al Subsistema de Gestión de Canales Virtuales?

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

1. Definición de los principales conceptos, características y tendencias actuales relacionadas con la gestión de canales virtuales.
2. Caracterización de las soluciones similares existentes.
3. Selección de las herramientas y tecnologías a utilizar durante el desarrollo del subsistema.
4. Conceptualización, elaboración, y construcción del diseño de software del Subsistema de Gestión de Canales Virtuales.
5. Implementación del Subsistema de Gestión de Canales Virtuales para el Sistema de Transmisión de Canales Virtuales.
6. Aplicación de las pruebas de software para validar el correcto funcionamiento del Subsistema de Gestión de Canales Virtuales.

Para la investigación se utilizaron los siguientes **métodos de investigación** (Hernández, y otros, 2011):

Métodos Teóricos

Permiten estudiar las características del objeto de investigación que no son observables directamente, facilitan la construcción de modelos e hipótesis de investigación y crean las condiciones para ir más allá de

las características fenomenológicas y superficiales de la realidad, contribuyendo al desarrollo de las teorías científicas y para su ejecución se apoyan en el proceso de análisis y síntesis.

- **Analítico-Sintético:** El análisis permite la división mental del fenómeno en sus múltiples relaciones y componentes para facilitar su estudio. La síntesis establece mentalmente la unión entre las partes previamente analizadas, posibilita descubrir sus características generales y las relaciones esenciales entre ellas. Este método contribuye a analizar y comprender los referentes teóricos, así como toda la documentación relacionada con el objeto de estudio, facilitando la obtención de los elementos más importantes relacionados con el mismo. Será determinante para definir el camino a seguir en cada paso de la investigación a partir de la exploración de la realidad.
- **Modelación:** se evidencia con la creación de reproducciones simplificadas del entorno, como son los modelos y diagramas representados, permite la representación de la realidad y facilita estudiar nuevas relaciones y cualidades del objeto de estudio.

Estructura del Documento

El trabajo de diploma se divide en 3 capítulos, los cuales están estructurados de la siguiente forma:

Capítulo 1. “Fundamentación Teórica”: en este capítulo se describen los principales conceptos a tratar, tales como canal virtual, parrilla de programación y espacio televisivo. Además se definen un grupo de herramientas y tecnologías a utilizar para el desarrollo del subsistema. También se realiza un estudio de las posibles soluciones existentes.

Capítulo 2. “Análisis y diseño del Subsistema de Gestión de Canales Virtuales”: en este capítulo se identifican los requerimientos, se escogen la arquitectura, los patrones de diseño, y se diseñan las clases de la solución a implementar. Además, se utiliza la metodología de software AUP-UCI para el desarrollo del subsistema, y se detallan los pasos de la misma, incluyendo varios de sus diagramas.

Capítulo 3. “Implementación y Pruebas”: en este capítulo se muestra los distintos componentes lógicos desarrollados a través del modelo de despliegue y la organización del subsistema mediante el modelo de componentes. Se explican los estándares de codificación empleados. Además, se implementan cada una de las funcionalidades hasta lograr el producto completo y por último se realizan las pruebas respectivas para validar que la solución elaborada está funcionando de manera correcta.

Capítulo 1. Fundamentación teórica

En el presente capítulo se fundamenta el marco teórico de la investigación, se caracterizan la metodología de desarrollo, las herramientas, tecnologías y lenguaje de programación utilizados en el proceso de desarrollo del subsistema.

1.1 Conceptos asociados al dominio del problema.

Para entender el problema a resolver es necesario explicar algunos términos relacionados con el objeto y campo de acción de la investigación. Estos términos se muestran a continuación:

1.1.1 Gestión de canales virtuales

Gestión

El término gestión es utilizado para referirse al conjunto de acciones, o diligencias que permiten la realización de cualquier actividad o deseo. Dicho de otra manera, una gestión se refiere a todos aquellos trámites que se realizan con la finalidad de resolver una situación o materializar un proyecto. En el entorno empresarial o comercial, la gestión es asociada con la administración de un negocio. (Real Academia Española, 2016)

Del latín *gestio*, el concepto de gestión hace referencia a la acción y a la consecuencia de administrar o gestionar algo. Al respecto, hay que decir que gestionar es llevar a cabo diligencias que hacen posible la realización de una operación comercial o de un anhelo cualquiera. (Porto, y otros, 2012)

Estos conceptos se consideran fundamentales para la presente investigación, por lo que se toman como base de conocimientos en el desarrollo investigativo. Además, se adicionan a los mismos que, la gestión es la asunción y ejercicio de responsabilidades sobre un proceso (es decir, sobre un conjunto de actividades) lo que incluye: la preocupación por la disposición de los recursos y estructuras necesarias para que tenga lugar; y la coordinación de sus actividades (y correspondientes interacciones).

Canal virtual

Un canal de programación para programas televisivos que mediante la utilización de parrillas de programación va a permitir que los espacios que se brinden en el canal tengan una secuencia ordenada; y donde los usuarios o televidentes podrán identificar y acceder con oportunidad a la programación de su elección y el acceso a nuevos canales (Castillo, 2016).

Canal de transporte unidireccional de células del modo de transferencia asíncrono asociadas mediante un valor de identificador común único (RAI, 2016).

Estos conceptos se consideran fundamentales para la presente investigación, por lo que se toman como base de conocimientos en el desarrollo investigativo. Además, se adicionan a los mismos que, un canal virtual es un canal de programación para programas televisivos donde los usuarios o televidentes pueden visualizar y acceder con oportunidad a la programación de su elección. También pueden acceder a nuevos canales y la cartelera correspondiente a los mismos.

Los conceptos vistos no contemplan algunos aspectos con relación al tema tratado en la investigación, por lo que a partir de ellos los autores definen un nuevo concepto enriquecido por los estudiados y que se adapta al marco del problema. Se define como Gestión de Canales Virtuales los procesos que se desarrollan en torno a la gestión de los canales virtuales, la creación de los mismos y su planificación. Además, la generación de las parrillas de programación y la creación de los espacios televisivos.

1.1.2 Parrilla de programación de televisión

La parrilla de programación de televisión (o cartelera) es un espacio en el cual se planifican las emisiones de audio y audiovisuales² dentro de un rango de tiempo determinado o un espacio televisivo, que puede ser consultado bajo demanda (Castillo, 2016).

Este concepto se considera fundamental para la presente investigación, por lo que se toman como base de conocimiento en el desarrollo investigativo. Además, se adicionan al mismo que, la parrilla de programación televisiva es la disposición en el tiempo de los programas de un canal de televisión o una emisora de radio. Indica la hora de comienzo y fin de cada programa a lo largo del día.

1.1.3 Espacio Televisivo

Es un programa hecho para la televisión y para diferenciarlo de otros medios, bien sea prensa, radio o cine es denominado espacio televisivo. Además, el espacio será creado para determinados horarios, se transmite en la fecha y hora planificada; y también contiene audiovisuales asociados a ese espacio (Castillo, 2016).

Este concepto se considera fundamental para la presente investigación, por lo que se toman como base de conocimiento en el desarrollo investigativo. Además, se adicionan al mismo que, el espacio televisivo puede estar repetido en días y canales diferentes; y con materiales audiovisuales diferentes.

² Audiovisuales se refiere conjuntamente al oído y a la vista, o los emplea a la vez. Se dice especialmente de métodos didácticos que se valen de grabaciones acústicas acompañadas de imágenes ópticas. (RAE, 2016)

1.2 Estudio de las soluciones existentes.

La generación, publicación y actualización automática de la parrilla de programación (o cartelera) de cada canal es de gran importancia en la actualidad para agilizar la gestión de canales virtuales. La mayoría de los sistemas relacionados con la temática de la investigación usan la parrilla de programación para realizar la planificación; pero no pueden realizar cambios en tiempo real y algunas ocasiones los espacios televisivos se sobrescriben unos con otros. También se refleja la gestión de más de un canal a la vez, y la planificación de materiales audiovisuales y espacios televisivos en uno o varios canales por medio de una parrilla de programación. A continuación se exponen un conjunto de sistemas relacionados con la temática de la investigación con el fin de comprender la necesidad de realizar el Subsistema de Gestión de Canales Virtuales.

Digiton: Es un conjunto integral de herramientas de software e integra todos los componentes periféricos del sistema: estaciones de captura, editoras de vídeo, sistemas de almacenamiento y sistemas de emisión. La suite de Digiton permite la captura y gestión de contenido, la edición de vídeo y audio, la emisión por el play-out, archivo de contenidos: selección y documentación y búsqueda y recuperación de contenidos (Pedregosa, 2014).

Digiton posee características de interés para el desarrollo del Subsistema de Gestión de Canales Virtuales como son: gestión de canales en tiempo real, almacenamiento de las medias en un servidor local, aunque en la investigación realizada no se evidencia el control del tiempo en un canal determinado, la gestión de más de un canal a la vez.

VSNCREATV: es un software de planificación de la programación y gestión de TV. Permite gestionar los flujos de trabajo de los departamentos de publicidad, producción y emisión para planificar la emisión de TV (desde simulaciones en general para toda la temporada, a la programación y los horarios para una semana en particular). Además, VSNCREATV está diseñado sobre arquitectura Web, lo que favorece una implementación veloz y ligera en un canal o sistema multicanal. Además, es multiplataforma, lo que permite a los clientes trabajar desde cualquier ordenador y sistema operativo (Windows, OSx) desde un navegador (VSN, 2015).

VSNCREATV posee características de interés para el desarrollo del Subsistema de Gestión de Canales Virtuales como son: está diseñado sobre una arquitectura web, es multiplataforma, lo que permite a los clientes trabajar desde cualquier ordenador y sistema operativo. Es un sistema multicanal aunque en la investigación realizada no se evidencia el control del tiempo en los canales, así como la seguridad en el

acceso a la plataforma.

XILEMA STCV (Sistema de Transmisión de Canales Virtuales) es un sistema encargado principalmente de la gestión de todas las planificaciones de transmisión televisiva. También se encarga de la gestión de los usuarios que acceden al sistema de transmisión canales virtuales, para ello el usuario que acceda debe tener el rol de *superadmin* o *admin*.

STCV posee características de interés para el desarrollo del Subsistema de Gestión de Canales Virtuales como son: es un sistema multicanal, es multiplataforma, se encarga de la gestión de los usuarios que acceden al sistema, lo que permite a los clientes trabajar desde cualquier ordenador y sistema operativo. Aunque en la investigación realizada no se evidencia el control del tiempo en los canales, la existencia de parrillas de programación que faciliten la planificación en los canales y no permite generación, publicación y actualización automática de la parrilla de programación (o cartelera) de cada canal.

Ustream es una plataforma online que dispone de diversos canales que permiten la transmisión de eventos en vivo. Esta transmisión es realizada por los usuarios si se encuentran registrados, y es posible hacerlo desde un dispositivo móvil. Los usuarios tienen la opción de registrarse gratis o en forma de páginas, teniendo en este último caso mayores ventajas como la opción de restringir la incrustación de reproductores de video o introducir anuncios sobre su canal sin la necesidad de tener amplios conocimientos de HTML5 (Ustream, 2016).

Ustream posee características de interés para el desarrollo del Subsistema de Gestión de Canales Virtuales como son: es un sistema multiplataforma, se encarga de la gestión de los usuarios que acceden al sistema, lo que permite a los clientes trabajar desde cualquier ordenador y sistema operativo. La investigación realizada no se evidencia el control del tiempo en los canales, la existencia de parrillas de programación que faciliten la planificación en los canales y no es un sistema multicanal.

1.2.1 Resultados del análisis de las soluciones existentes

A continuación, se muestra una tabla que permite una mayor comprensión de las principales características identificadas por cada sistema analizado.

Tabla 1 Resumen del estudio de sistemas similares.

| Sistemas analizados | VSNCREATV | Digiton | XILEMA STVC | Ustream |
|---------------------|-----------|---------|-------------|---------|
| | | | | |

| | | | | |
|--|---|---|---|---|
| Características de interés | | | | |
| Generación, publicación y actualización automática de la parrilla de programación (o cartelera) de cada canal | - | - | - | - |
| Sistema multicanal | X | - | X | X |
| Planificación de materiales audiovisuales y espacios televisivos en uno o varios canales por medio de una parrilla de programación | X | X | X | - |

Luego del análisis realizado a las soluciones existentes vistas se concluye que las mismas no constituyen una solución para la problemática planteada en la presente investigación, debido a que poseen un sistema privativo, por lo que presentan imposibilidad de copias, de modificación y redistribución. El soporte es exclusivo del propietario, y además el usuario que adquiere el software depende de la empresa propietaria, por lo que es necesario desarrollar un Subsistema de Gestión de Canales Virtuales para el Sistema de Transmisión de Canales Virtuales. Además, se analizaron un conjunto de características de interés las cuales evidencian que los sistemas estudiados no cumplen con su totalidad y no resuelven la problemática planteada. Aunque se debe destacar que las soluciones estudiadas aportaron características a tener en cuenta para el desarrollo de la propuesta de solución.

1.3 Metodología, herramientas y tecnologías a utilizar para el desarrollo del módulo.

Para desarrollar la plataforma propuesta en la presente investigación, se hace necesario indagar sobre los lenguajes, herramientas y tecnologías a ser utilizados en el desarrollo de la propuesta de solución. A continuación se procede con el estudio y selección de las mismas.

1.3.1 Metodología de desarrollo de software AUP-UCI.

La metodología para el desarrollo del software es un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Proceso disciplinado sobre el desarrollo de un sistema en el que se van indicando detalladamente todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. El uso de la misma proporciona un proceso más predecible en la construcción

de un producto y un aumento de la calidad del software en todas las fases del desarrollo del mismo (Pressman, 2002).

Para el desarrollo del subsistema se utilizó como metodología AUP (Proceso Unificado Ágil, por sus siglas en inglés) en su variante UCI, empleando el escenario 2. Teniendo en cuenta las necesidades para la solución del problema investigativo, la línea de desarrollo del proyecto STCV, se utiliza la metodología de desarrollo AUP-UCI. La misma define fases, principios, etapas o flujos de trabajo, disciplinas de soporte, entre otros elementos, para mejorar y organizar el trabajo desde el comienzo. Además, desde un inicio están bien definidos los requisitos de software, sobre los que no ocurrirán cambios constantes.

1.3.2 Lenguaje de Modelado Unificado.

Lenguaje de Modelado Unificado (UML por sus siglas en inglés) se define como un “lenguaje cuyo vocabulario y reglas se centran en la representación conceptual y física de un sistema” (UML, 2016). Se utiliza con el objetivo de visualizar, especificar, documentar y modelar los artefactos del subsistema.

1.3.3 Herramienta CASE.

Una herramienta CASE supone la automatización del desarrollo del software, contribuyendo a mejorar la calidad y la productividad en varios aspectos del ciclo de vida de desarrollo del software en tareas como el diseño de proyectos, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores, entre otras. Permite la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta conseguimos agilizar el trabajo. Además, facilita la realización de prototipos y el desarrollo de conjunto de aplicaciones, simplifica el mantenimiento de los programas, mejora y estandariza la documentación, aumenta la portabilidad de las aplicaciones y facilita la reutilización de componentes software. También permite un desarrollo y un refinamiento visual de las aplicaciones mediante la utilización de gráficos (Somerville, 2009).

Visual Paradigm es una herramienta CASE profesional, que utiliza el lenguaje de modelado UML; la misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software y fue diseñada para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos (Visual Paradigm, 2015).

Se realizó la selección de Visual Paradigm como herramienta CASE, ya que es la que se usa en el centro para el modelado. Además, se caracteriza por su disponibilidad en múltiples plataformas como Windows y

Linux. También se caracteriza por el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación; capacidades de ingeniería directa e inversa; modelo y código que permanece sincronizado en todo el ciclo de desarrollo; y disponibilidad de múltiples versiones, con diferentes especificaciones. Posee una licencia: gratuita y comercial, soportar aplicaciones Web, es fácil de instalar y actualizar, y permite construir todos los diagramas del subsistema brindando un claro entendimiento de los artefactos generados.

1.3.4 Framework Symfony

Symfony v2.8.4 es un *framework* de tipo *full-stack*³ construido con varios componentes independientes y librerías que se utilizan para crear aplicaciones y sitios web en PHP basado en el patrón Modelo Vista Controlador. Su código, todos los componentes y librerías que incluye, se publican bajo la licencia MIT⁴ de software libre. También la documentación del proyecto es libre e incluye varios libros y decenas de tutoriales específicos.

Symfony permite establecer los parámetros de configuración de las aplicaciones a través de variables de entorno del propio servidor para evitar el uso de contraseñas en archivos de configuración. Dispone de un plan de lanzamientos predecible, con versiones estables mantenidas durante tres años. Symfony incluye varias herramientas gráficas y de consola para depurar fácilmente los errores que se produzcan en las aplicaciones (symfony, 2016).

Symfony es utilizado en la presente investigación para el desarrollo del subsistema, ya que este *framework* de desarrollo es el que utiliza el proyecto y actualmente es estable. Además, con la utilización del patrón Modelo Vista Controlador permite a los desarrolladores y los diseñadores trabajar simultáneamente, incluyendo la capacidad de hacer prototipos rápidos, hacer cambios en una parte del subsistema sin afectar a los demás. Permite que el subsistema sea compatible con bibliotecas e infraestructuras que existen y que se pueda adaptar a entornos de negocio en cambio permanente. También es compatible con la mayoría de los sistemas gestores de base de datos en el caso de la presente investigación PostgreSQL y permite la utilización de lenguajes como HTML5, PHP7 y CSS3 (LibrosWeb, 2016).

³ **Full-stack**: no es más que un conjunto completo (JAMSTACK, 2017).

⁴ **Licencia MIT**: es una licencia de software libre permisiva; compatible con muchas licencias Copyleft, como la GNU General Public License; y se origina en el Instituto Tecnológico de *Massachusetts* (MIT, *Massachusetts Institute of Technology*) (Symfony, 2016).

1.3.5 Entorno de Desarrollo Integrado

Un Entorno de Desarrollo Integrado (IDE por sus siglas en inglés) es un programa informático compuesto por un conjunto de herramientas de programación, dígame un editor de código, un compilador, un depurador y un constructor de interfaces gráficas de usuario (Skvorc, 2015).

PhpStorm es un IDE ligero, de fácil instalación y multiplataforma. Soporta los lenguajes PHP, HTML y JavaScript los cuales son utilizados en el desarrollo del subsistema brindando análisis de código en la marcha y autocompletado automático (JetBrains, 2014). Es utilizado para la implementación del subsistema ya que brinda un completamiento inteligente de código, detecta código duplicado, y sintaxis abreviada. Además de las características que brinda, este IDE es usado por el proyecto.

Servidor web Apache v2

El servidor web Apache v2 es multiplataforma, rápido, brinda soporte a largo plazo y de código abierto. Apache v2 es usado en combinación con el motor de bases de datos PostgreSQL y el lenguaje PHP (Apache, 2016). Esta unión permite la interacción con la base de datos del Subsistema de Gestión de Canales Virtuales. Además de las características que brinda, es el servidor web que utiliza el proyecto.

Sistema Gestor de Bases de Datos (SGBD) PostgreSQL

El gestor de base de datos PostgreSQL es un potente gestor de bases de datos de código abierto que soporta el esquema objeto-relacional. Cuenta con una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad e integridad de datos. Es multiplataforma e incluye almacenamiento de varios tipos de datos (PostgreSQL, 2016).

El uso de PostgreSQL como SGBD en su versión 9.3 brinda ventajas como: fácil administración, estabilidad y flexibilidad. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Por lo que se utilizó en la presente investigación como el gestor de base de datos para almacenar la información relacionada con el Subsistema de Gestión de Canales Virtuales; además, es utilizado por el proyecto.

PgAdmin 3

PgAdmin 3 es una herramienta de código abierto que permite administrar sistemas gestores de bases de datos PostgreSQL (ArPUG, 2016). Posee una interfaz gráfica que soporta todas las características del SGBD PostgreSQL y hace simple la administración de dicho SGBD. Además, es multiplataforma, y su licencia de PostgreSQL es fácil de adquirir. También es esta herramienta usada por el proyecto para el trabajo con el sistema gestor de base de datos.

Protocolo HTTPS

Para la comunicación se hace uso del Protocolo Seguro de Transferencia de Hipertexto (más conocido por sus siglas en inglés HTTPS), el cual es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto. HTTPS utiliza un cifrado basado en SSL⁵/TLS⁶ para crear un canal seguro más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará difícil de descifrar (Lawrence, 2006).

1.3.6 Lenguajes de programación

JavaScript, aka Mocha, aka LiveScript, también conocido como JScript, también conocido como ECMAScript, es uno de los lenguajes de programación más populares del mundo. Es un lenguaje de programación interpretado, su potencia está dada por el soporte que tiene en los distintos navegadores y es un muy agradable lenguaje dinámico orientado a objetos de programación de propósito general. Se caracteriza por ser imperativo y estructurado, ya que es compatible con gran parte de la estructura de programación de C. Además, es dinámico, ya que es compatible con varias formas de comprobar el tipo de un objeto y permite evaluar expresiones expresadas como cadenas en tiempo de ejecución; es funcional y prototípico (Crockford, 2016). Para el desarrollo del subsistema de la presente investigación ofrece una lista de características que permiten agilizar el proceso de desarrollo por lo que se tomó este lenguaje; además de que el proyecto trabaja con el mismo.

CSS3 u hojas de estilo en cascada (en inglés Cascading Style Sheets) es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML. Permite hacer aplicaciones web con mayor separación entre estilos y contenidos. Brinda soporte a muchas necesidades de los sitios web actuales, sin tener que recurrir a trucos de diseñadores o lenguajes de programación. Trae consigo muchas novedades altamente esperadas, como las esquinas redondeadas, sombras, gradientes, transiciones o animaciones, y

⁵ SSL (Secure Socket Layer), es un protocolo de seguridad que hace que sus datos viajen de manera íntegra y segura, es decir, la transmisión de los datos entre un servidor y usuario web, y en retroalimentación, es totalmente cifrada o encriptada (Moller, y otros, 2014).

⁶ Protocolo TLS (Transport Layer Security): Es un protocolo mediante el cual se establece una conexión segura por medio de un canal cifrado entre el cliente y servidor. Así el intercambio de información se realiza en un entorno seguro y libre de ataques (Gutmann, 2014).

nuevos layouts como multi-columnas, cajas flexibles o maquetas de diseño en cuadrícula (grid layouts) (Celik, 2013). CSS3 permite controlar el estilo y el formato de las páginas Web a utilizar en el subsistema.

HTML5 (HyperText Markup Language, versión 5) esta nueva versión (aún en desarrollo), y en conjunto con CSS3, define los nuevos estándares de desarrollo web, rediseñando el código para resolver problemas y actualizándolo así a nuevas necesidades. No se limita solo a crear nuevas etiquetas por lo que incorpora las etiquetas (canvas 2D y 3D, audio, vídeo) con codecs para mostrar los contenidos multimedia. También incorpora etiquetas para manejar grandes conjuntos de datos: Datagrid, Details, Menu y Command, las que permiten generar tablas dinámicas que pueden filtrar, ordenar y ocultar contenido en cliente.

Además, realiza mejoras en los formularios, con nuevos tipos de datos (eMail, number, url, datetime...) y facilidades para validar el contenido sin Javascript. También posee Drag & Drop, que no es más que una nueva funcionalidad para arrastrar objetos como imágenes; y los visores: MathML (fórmulas matemáticas) y SVG (gráficos vectoriales), que de forma general se deja abierto a poder interpretar otros lenguajes XML. Esta versión establece una serie de nuevos elementos y atributos que reflejan el uso típico de los sitios Web modernos (Garro, 2015). El uso del lenguaje HTML5 permite estructurar las páginas web del subsistema que son interpretadas por el navegador y como resultado poder mostrar textos, tabla, botones, campos de textos y enlaces; por lo que en la presente investigación se hace uso del mismo.

PHP7 (PHP Hypertext Preprocessor, versión 7) es un lenguaje interpretado y multiplataforma de alto nivel incrustado en páginas HTML y ejecutado en el servidor. PHP7 es una versión la cual incluye el manejo de excepciones y posibilita el uso de técnicas de programación orientada a objetos. Existen librerías escritas en el lenguaje PHP que facilitan interfaces para una gran cantidad de sistemas de base de datos diferentes y dispone de conexión propia para estos sistemas. Entre sus ventajas se encuentra la facilidad de aprendizaje y portabilidad (Luis Miguel Cabezas Granado, 2004). Por sus características es usado en la investigación, además de que es el lenguaje que usa el proyecto para el desarrollo.

jQuery es una biblioteca multiplataforma escrita en JavaScript. Permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Además, es libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privados. También se caracteriza por su selección de elementos DOM, interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath. Manipulación de la hoja de estilos CSS; efectos y animaciones, que pueden ser animaciones personalizadas; y soporta extensiones (The jQuery Foundation, 2017). JQuery es utilizado en el desarrollo de Subsistema de Gestión

de Canales Virtuales ya que permite el manejo de eventos, la animación, y las interacciones Ajax para un desarrollo web rápido; además, el proyecto trabaja con esta biblioteca.

Conclusiones parciales

En este capítulo se han caracterizado los principales conceptos relacionados a la gestión de canales virtuales, lo cual permitió una mayor comprensión del objeto de estudio, realizando un análisis de los factores que influyen en la situación problemática. A partir del análisis de los diferentes sistemas que se utilizan para la gestión y planificación de canales virtuales en el mundo, se definió que la vía más factible para dar solución al problema es desarrollar un Subsistema de Gestión de Canales Virtuales. También, la caracterización de las diferentes herramientas y tecnologías a utilizar, posibilita mayor seguridad para hacer uso de las mismas en aras de cumplir el objetivo general.

Capítulo 2. Análisis y diseño del Subsistema de Gestión de Canales Virtuales.

En el presente capítulo se exponen las principales características del subsistema a implementar, mediante la identificación de los requisitos funcionales y no funcionales con los que debe de cumplir la solución. También se describen el patrón arquitectónico y los patrones de diseño de software empleados. Se realiza el modelo de dominio, la definición de actores y relaciones entre ellos, además el diagrama de Caso de Uso del sistema y las descripciones textuales. Posteriormente, se muestra el diagrama de clases del diseño propuesto.

2.1 Modelo de dominio.

El Modelo de Dominio o Modelo Conceptual es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Para la realización del mismo se utiliza un lenguaje común, lo que facilita la comunicación entre los desarrolladores y usuarios de la aplicación; además, favorece un mayor entendimiento del contexto en que se desarrolla el sistema. El modelo de dominio se describe mediante diagramas UML (diagrama de clases) que muestra las clases del dominio y cómo se relacionan unas con otras mediante asociaciones (Larman, 2003). Para la presente investigación se plantea el uso de un modelo de dominio ya que no se tiene una estructura definida de los procesos del negocio, no se cuenta con un manual de procedimientos y los flujos de información son difusos.

2.1.1 Modelo de Dominio del Subsistema de Gestión de Canales Virtuales

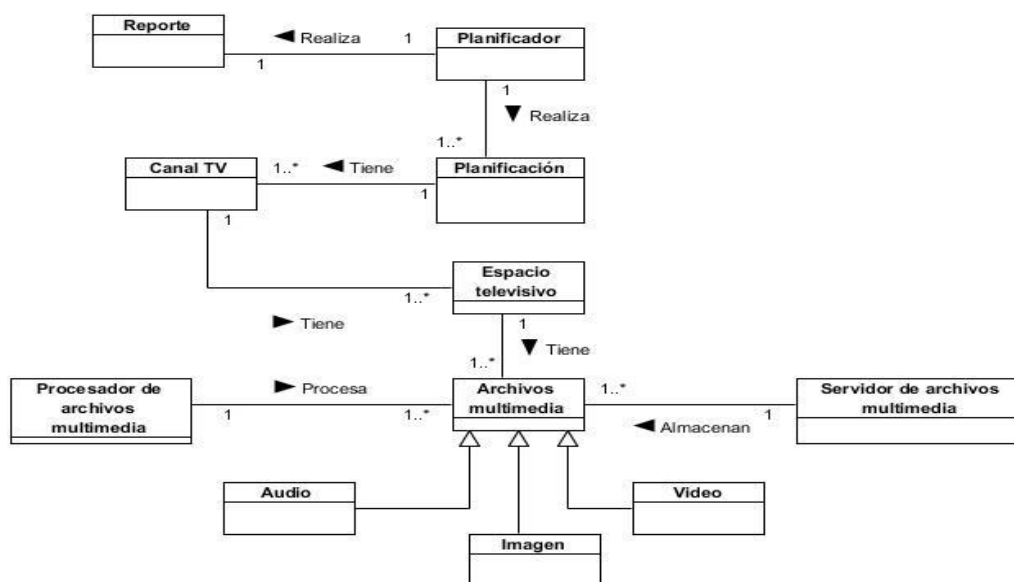


Figura 1: Modelo de dominio

2.1.2 Descripción de las clases del Modelo de Dominio del Subsistema de Gestión de Canales Virtuales

El modelo de dominio contribuye en el proceso de desarrollo del software a identificar las clases necesarias para modelar el sistema. A continuación, se identifican los conceptos fundamentales que se emplean en el modelo.

- **Sistema de Transmisión de Canales Virtuales:** sistema capaz de automatizar los procesos de planificación, transmisión y monitorización de canales virtuales que se ejecutan en una entidad dedicada a la gestión y transmisión de contenidos audiovisuales.
- **Archivos multimedia:** representa los materiales audiovisuales (audio y/o video) digitales que serán transmitidos por los canales de televisión.
- **Servidor de archivos multimedia:** servidor donde se almacenan los archivos multimedia.
- **Procesador de archivos multimedia:** representa al sistema encargado de procesar los archivos multimedia.
- **Canal TV:** constituye un tipo de estación emisora que transmite audio y video a receptores de televisión en un área concreta. Las transmisiones se realizan a partir de la consulta de la parrilla de programación.
- **Planificación:** constituye la secuencia de transmisión de las programaciones televisivas que serán transmitidas. Contiene la hora de inicio, la hora de fin, la fecha en la que se va a transmitir y el canal al cual pertenece dicha planificación.
- **Reporte:** constituye el reporte generado por el usuario, contiene los parámetros del cual dicho usuario desea conocer el estado.

2.2 Especificación de los requisitos.

El Instituto de Ingeniería de Software (SEI, por sus siglas en inglés) plantea como parte del glosario del modelo CMMI que un requisito es: (1) Una condición o capacidad necesitada por un usuario para solucionar un problema o lograr un objetivo. (2) Una condición o capacidad que debe cumplir o poseer un producto o componente de producto para satisfacer un contrato, un estándar, una especificación u otros documentos impuestos formalmente. (3) Una representación documentada de una condición o capacidad como en (1) o en (2).

2.2.1 Requisitos funcionales

Los requisitos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir. Son establecidos entre el cliente y el equipo de proyecto sobre lo que debe o no hacer el sistema (Sommerville,

2002). Para poder identificar qué debe hacer el subsistema y entender su funcionamiento, es fundamental conocer los requisitos funcionales que el subsistema debe cumplir. La técnica utilizada para la obtención de requisitos fue la entrevista a los especialistas del STCV.

A continuación se muestran los requisitos funcionales (RF) identificados con una breve descripción:

RF1. Adicionar usuario

El subsistema debe permitir adicionar un nuevo usuario.

RF2. Modificar usuario

El subsistema debe permitir modificar los datos de un usuario seleccionado.

RF3. Eliminar usuario

El subsistema debe permitir eliminar uno o varios usuarios seleccionados.

RF4. Buscar usuario

El subsistema debe permitir buscar un usuario por nombre de usuario.

RF5. Listar usuarios

El subsistema debe permitir listar todos los usuarios existentes.

RF6. Autenticar usuario

El subsistema debe permitir autenticar a los usuarios.

RF7. Cerrar sesión

El subsistema debe permitir a los usuarios autenticados cerrar la sesión.

RF8. Adicionar rol

El subsistema debe permitir adicionar un nuevo rol.

RF9. Modificar rol

El subsistema debe permitir modificar un rol.

RF10. Eliminar rol

El subsistema debe permitir eliminar uno o varios roles.

RF11. Asignar rol

El subsistema debe permitir asignar uno o varios roles a un usuario seleccionado.

RF12. Asignar permisos

El subsistema debe permitir asignar uno o varios permisos a un rol.

RF13. Adicionar canal

El subsistema debe permitir adicionar un nuevo canal.

RF14. Modificar canal

El subsistema debe permitir modificar los datos de un canal creado.

RF15. Eliminar canal

El subsistema debe permitir eliminar uno o varios canales.

RF16. Listar canales

El subsistema debe permitir listar los canales existentes.

RF17. Buscar canal

El subsistema debe permitir buscar un canal a partir del nombre, dirección IP, URL, puerto.

RF18. Adicionar espacio televisivo

El subsistema debe permitir adicionar un espacio televisivo a un canal.

RF19. Modificar espacio televisivo

El subsistema debe permitir modificar los datos de un espacio televisivo de un canal.

RF20. Eliminar espacio televisivo

El subsistema debe permitir eliminar un espacio televisivo de un canal.

RF21. Listar espacios televisivos de un canal

El subsistema debe permitir listar los espacios televisivos de un canal.

RF22. Adicionar archivo multimedia

El subsistema debe permitir adicionar un archivo multimedia.

RF23. Modificar archivo multimedia

El subsistema debe permitir modificar los datos de un archivo multimedia.

RF24. Eliminar archivo multimedia

El subsistema debe permitir eliminar un archivo multimedia.

RF25. Asignar archivo multimedia a un espacio televisivo

El subsistema debe permitir asignar uno o varios archivos multimedia a un espacio televisivo.

2.2.2 Requisitos no funcionales

Los requisitos no funcionales (RNF) son propiedades o cualidades que el producto debe tener. Especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, seguridad y fiabilidad. Para determinar los requisitos no funcionales se tuvo en cuenta las clasificaciones dadas por Sommerville en el libro Ingeniería de Software. 6ta. Edición (Sommerville, 2002).

Requisito de Software:

RNF 1. El Subsistema debe ser utilizado sobre el navegador Mozilla Firefox v5.0 o versiones superiores.

Requisito Hardware:

RNF 2. La PC servidor de aplicaciones debe cumplir con los siguientes requisitos mínimos que se especifican a continuación: una RAM de 4 GB o superior y procesador Intel Core i3 a 3.0 GHz o superior.

RNF 3. La PC cliente sería recomendable que cumpla con los siguientes requisitos mínimos que se especifican a continuación: una RAM de 2 GB o superior, un procesador Intel Pentium Dual-Core a 1.5 GHz o superior

2.3 Descripción del Subsistema de Gestión de Canales Virtuales.

2.3.1 Definición de los actores

Se le llama actor a toda entidad externa al sistema que guarda una relación con este y que le demanda una funcionalidad. Los actores pueden representar roles jugados por usuarios humanos, hardware externo, u otros sujetos. Un actor no necesariamente representa una entidad física específica, sino simplemente una faceta particular (es decir, un "rol") de alguna actividad que es relevante a la especificación de sus casos de uso asociados. Así, una única instancia física puede jugar el rol de muchos actores diferentes y, asimismo, un actor dado puede ser interpretado por múltiples instancias diferentes. (Kruchten, 2003)

Tabla 2 Descripción de los Actores del Sistema.

| Actor | Descripción |
|---------------|--|
| Usuario | Se encarga de interactuar con la interfaz principal del subsistema, permitiéndole la autenticación al mismo. |
| Planificador | Se encarga de realizar la gestión de los canales virtuales en el subsistema. |
| Administrador | Se encarga de administrar los usuarios, roles y permisos del subsistema. |

2.3.2 Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema explica gráficamente un conjunto de casos de uso (CU) del sistema, los actores y cómo se relacionan entre ellos. El diagrama tiene por objetivo ofrecer una clase de diagrama contextual que permite conocer rápidamente los actores externos de un sistema y las formas básicas en que lo utilizan (Larman, 2003).

Tabla 3 Agrupación de requisitos por caso de uso.

| Nombre del caso de uso | Referencia a requisitos |
|---------------------------------------|-------------------------|
| Gestionar usuario | RF: 1,2,3,4,5 |
| Autenticar usuario | RF: 6 |
| Cerrar sesión | RF:7 |
| Gestionar rol | RF: 8,9,10 |
| Asignar rol | RF: 11 |
| Asignar permiso | RF:12 |
| Gestionar canal | RF:13,14,15, 16, 17 |
| Gestionar espacio televisivo | RF:18,19, 20 |
| Gestionar media | RF:22,23,24 |
| Asignar media a un espacio televisivo | RF:25 |

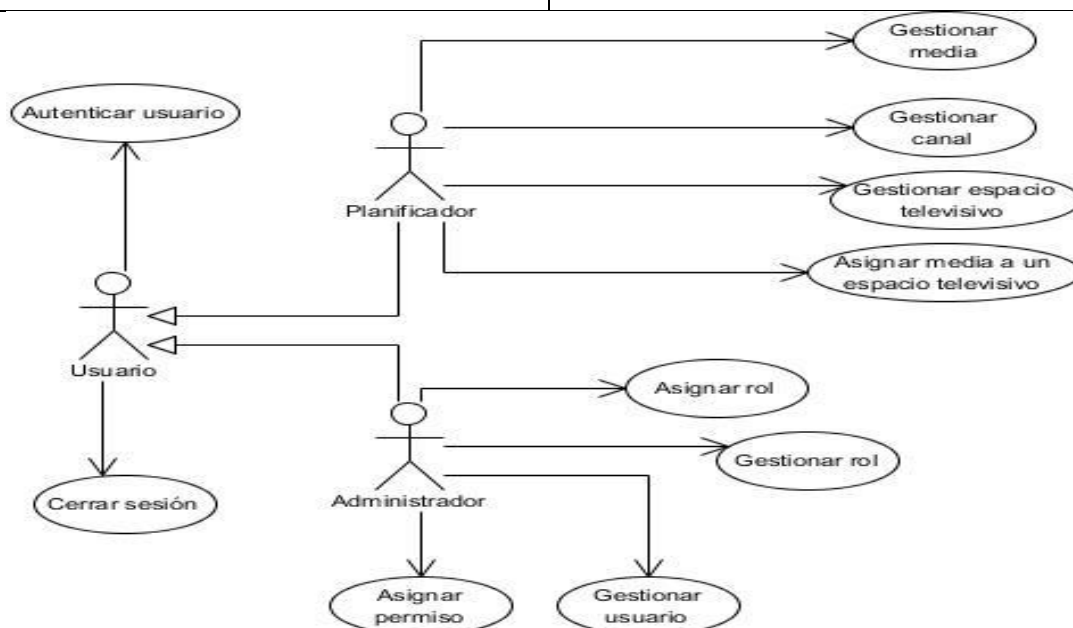


Figura 2 Diagrama de Casos de uso del sistema

2.3.3 Descripción de casos de uso del subsistema

Los casos de uso (CU) dirigen el proceso de desarrollo en su totalidad. Los casos de uso son una descripción de un conjunto de secuencias de acciones, incluyendo variaciones, que un sistema realiza para lograr un

resultado observable de valor para un actor (Pressman, 2002). A continuación, se describen los casos de uso críticos en el subsistema.

Tabla 4 Descripción del caso de uso: Gestionar canal televisivo en el Subsistema de Gestión de Canales Virtuales.

| | | |
|-------------------------------------|--|---|
| Objetivo | Adicionar, eliminar, modificar, buscar, y listar canales | |
| Actores | Planificador (Inicia): Gestionar canal televisivo | |
| Resumen | El caso de uso se inicia cuando el planificador selecciona la opción "Gestionar Canales" y procede a adicionar, modificar, buscar, listar o eliminar un canal. Termina cuando se haya realizado alguna de estas operaciones. | |
| Complejidad | Alta. | |
| Prioridad | Media. | |
| Precondiciones | El usuario con rol planificador deberá estar autenticado y debe existir conexión la Base de Datos. | |
| Postcondiciones | Se gestiona un canal. | |
| Flujo de eventos | | |
| Flujo básico Gestionar canal | | |
| | Actor | Sistema |
| 1 | Selecciona de la interfaz principal la opción Gestionar Canales | |
| 2 | | <p>Muestra una interfaz con todos los canales existentes, permitiendo seleccionar alguna de las siguientes acciones:</p> <ul style="list-style-type: none"> • Adicionar canal (Ver Flujo básico de eventos). • Buscar canal (Ir a la Sección 1 Buscar canal). • Eliminar canal (Ir a la Sección 2 Eliminar canal). • Modificar canal (Ir a la Sección 3 Modificar canal). |
| 3 | Selecciona la opción "Nuevo" | |
| 4 | | <p>Muestra un formulario con los datos del canal</p> <ul style="list-style-type: none"> • nombre (campo de texto) (campo obligatorio) • puerto (campo de texto) (campo obligatorio) • dirección IP (campo de texto) (campo |

| | | |
|---|---|--|
| | | obligatorio) • descripción canal (campo de texto) |
| 5 | Llena los campos y presiona el botón "Aceptar". | |
| 6 | | Valida los datos introducidos. |
| 7 | | Muestra el canal creado en la lista. Termina el CU. |
| Flujos alternos | | |
| 6a. Existen campos vacíos. | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "No deben existir campos vacíos". |
| 2 | | Vuelve a ejecutar el paso 2 |
| 6b. El nombre o el puerto ya existen en la base de datos | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "Datos ya existentes." |
| 2 | | Vuelve a ejecutar el paso 2 |
| 6c. El usuario llena el campo "Dirección IP" con un formato incorrecto | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "Entre un IP válido". |
| 2 | | Vuelve a ejecutar el paso 2. |
| Sección 1: "Buscar canal" | | |
| Flujo básico | | |
| | Actor | Sistema |
| 1 | | Muestra en la interfaz una pestaña, con un campo vacío a llenar donde se insertan los datos. |

| | | |
|---|--|---|
| 2 | Inserta el nombre del canal que desea buscar | |
| 3 | | Consulta la base de datos para ver si existe el canal. |
| 4 | | Muestra el canal en una lista (Ver figura A). Termina el CU. |
| Flujos alternos | | |
| 3a. Existen datos incorrectos. | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "No existe canales con estas características". |
| Sección 2: "Eliminar canal" | | |
| Flujo básico | | |
| | Actor | Sistema |
| 1 | Selecciona la opción Eliminar del canal al que se desea efectuar la operación. | |
| 2 | | Muestra un mensaje ¿Está seguro que desea eliminar el canal? |
| 3 | Selecciona la opción Aceptar. | |
| 4 | | Elimina el canal y actualiza la lista de canales. Termina el CU. |
| Flujos alternos | | |
| 3a. Selecciona la opción Cancelar. | | |
| | Actor | Sistema |
| 1 | Pulsa en la opción Cancelar. | |
| 2 | | Cancela la operación. Termina el CU. |
| Sección 3: "Modificar canal" | | |
| Flujo básico | | |

| | | |
|---|--|---|
| 1 | Selecciona la opción Editar del canal al que se desea efectuar la operación. | |
| 2 | | Muestra un formulario con los datos del canal <ul style="list-style-type: none"> • nombre (campo de texto) (campo obligatorio) • puerto (campo de texto) (campo obligatorio) • dirección IP (campo de texto) (campo obligatorio) • descripción canal (campo de texto) |
| 3 | Llena los campos y presiona el botón "Actualizar". | |
| 4 | | Valida que no existan campos vacíos. |
| 5 | | Valida que los campos estén correctamente escritos. |
| 6 | | Muestra en el canal modificado en la lista. Termina el CU. |
| Flujos alternos | | |
| 4a. Existen campos vacíos | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "No deben existir campos vacíos". |
| 5a. El nombre o el puerto ya existen en la base de datos | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "Datos ya existentes." |
| 2 | | Vuelve a ejecutar el paso 2 |
| 5b. El usuario llena el campo "Dirección IP" con un formato incorrecto | | |
| | Actor | Sistema |
| 1 | | Muestra un mensaje "Entre un IP válido". |
| 2 | | Vuelve a ejecutar el paso 2. |
| Relaciones | CU Incluidos | No existen. |
| | CU Extendidos | No existen. |
| Requisitos no funcionales | RNF 1, RNF 2, RNF 3, RNF 4. | |
| Asuntos pendientes | No existen. | |

Prototipo de interfaz de usuario Adicionar Canal

Adicionar Canal

Nombre del canal:

Puerto:

IP:

Descripción:

Estado: Activo No Activo

Logo: 

Prototipo de interfaz de usuario Buscar Canal

Canales

Imprimir CSV PDF Nuevo

| Nombre | Puerto | IP | Descripción | Estado | Logo | Acción |
|-------------------|--------|---------------|-------------|--------|--|---|
| Tele Rebelde | 12 | 253.121.111.2 | | Activo |  | <input type="button" value="Editar"/> <input type="button" value="Borrar"/> <input type="button" value="Plan"/> |
| Canal Educativo 2 | 1000 | 100.10.114.25 | | Activo |  | <input type="button" value="Editar"/> <input type="button" value="Borrar"/> <input type="button" value="Plan"/> |

Prototipo de interfaz de usuario Eliminar Canal

¿Está seguro que desea eliminar el canal?

Aceptar

Cancelar

Prototipo de interfaz de usuario Modificar Canal

Modificar Canal

Nombre del canal:

Puerto:

IP:

Descripción:

Estado: Activo No Activo

Logo: 

Guardar

Cancelar

2.4 Patrón arquitectónico.

La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el equipo de proyecto compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establece la estructura, funcionamiento e interacción entre las partes del software (Larman, 2003). Al diseñar una arquitectura de software se debe crear y representar componentes que interactúen entre ellos y tengan asignadas tareas específicas, además de organizarlos de forma tal que se logren los requisitos establecidos. Se puede iniciar con patrones de soluciones ya probados, y utilizar modelos que han funcionado. Estas soluciones probadas se conocen como patrones arquitectónicos, que van de lo general a lo particular.

Para el desarrollo del subsistema se decide hacer uso del Patrón Arquitectónico, Modelo – Vista - Controlador (MVC) debido a que es la arquitectura empleada en el sistema. El patrón arquitectónico MVC separa la lógica de negocio de la interfaz de usuario y se aumenta la posibilidad de evolución por separado de ambos aspectos. Mediante el uso de MVC es posible incrementar la reutilización de las aplicaciones debido a lo anteriormente mencionado.

El patrón consiste en separar el código del subsistema en tres partes: el modelo, la vista y el controlador. El modelo es la representación de la información con la cual el sistema opera, por lo tanto, gestiona todos los accesos a dicha información. La vista maneja la visualización de la información. El controlador responde a eventos (usualmente acciones del usuario) e invoca peticiones al modelo o las vistas (Venete, 2011).

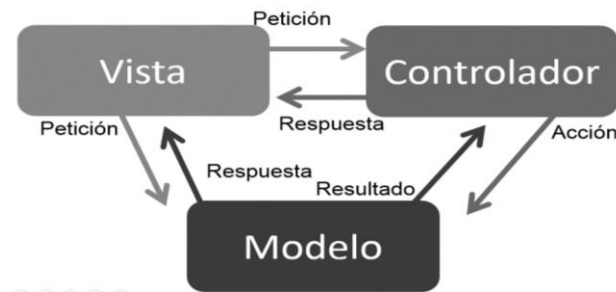


Figura 3 Patrón arquitectónico.

2.5 Patrones de diseño de software utilizados para el Subsistema de Gestión de Canales Virtuales.

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces (Microsoft, 2015).

2.5.1 Patrones para la asignación de responsabilidades (GRASP).

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos (Larman, 2003). Para la implementación del subsistema se hizo uso de los patrones siguientes:

El patrón **Creador (Creator)** tiene la información necesaria para realizar la creación del objeto. Es decir, que el patrón ayuda a identificar quién debe ser el responsable de la creación de nuevos objetos (Larman, 2003). El mismo se ve aplicado en la clase *CanalController*, que tiene varios métodos donde se crean objetos de la clase *Canal*.

El patrón **Experto (Expert)** indica que la clase que cuenta con la información necesaria para cumplir la responsabilidad es la responsable de manejar la información. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantendrá encapsulada (Larman, 2003). La utilización de dicho

patrón se ve aplicada en clases como: *CanalController* y *CanalRespository*, debido que cada una de ellas es responsable de manejar su información.

El patrón **Controlador**, asigna a clases específicas la responsabilidad de controlar el flujo de eventos de la aplicación. Lo anterior facilita la centralización de actividades (validaciones, seguridad, entre otros) (Larman, 2003). Un ejemplo de la utilización del patrón se encuentra en la clase *app*, la cual controla flujo de eventos del subsistema.

El patrón **Bajo Acoplamiento (Low Coupling)** indica una menor dependencia entre clases. De tal forma que, en caso de producirse una modificación en alguna de ellas, se tenga la menor repercusión posible en el resto de las clases (Larman, 2003). En el subsistema se evidencia el uso de este patrón ya que cada clase solo crea instancias de otras que necesite. De esta forma si se produce algún cambio en algunas de ellas se afecte lo menos posible las otras clases. La aplicación de dicho patrón se ve reflejado en todo el diseño del subsistema.

El patrón **Alta Cohesión (High Cohesion)** plantea que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase. El mismo se utiliza debido a que se recomienda tener un mayor grado de cohesión con un menor grado de acoplamiento. De esta forma se tiene menor dependencia y se especifican los propósitos de cada objeto en el sistema (Larman, 2003). Un ejemplo de utilización de este patrón es cuando se definen las responsabilidades y tareas de cada clase en el subsistema. La aplicación de dicho patrón se ve reflejado en todo el diseño del subsistema.

2.5.2 Patrones GOF

Los patrones GoF se descubren como una forma indispensable de enfrentarse a la programación a raíz del libro “Design Patterns—Elements of Reusable Software” de Erich Gamma, Richard Helm, Ralph Jonson y John Vlissides, a partir de entonces estos patrones son conocidos como los patrones de la pandilla de los cuatro (GoF, gang of four). Los mismos proponen soluciones a problemas concretos, no son teorías genéricas e indican resoluciones técnicas basadas en Programación Orientada a Objetos (POO). En ocasiones tienen más utilidad con algunos lenguajes de programación y en otras son aplicables a cualquier lenguaje. Se utilizan en situaciones frecuentes y estos es debido a que se basan en la experiencia acumulada al resolver problemas reiterativos. Se clasifican según su propósito en:

- **Patrones creacionales:** se encargan de la creación de instancias de los objetos. Abstraen la forma en que se crean los objetos, permitiendo tratar las clases a crear de forma genérica, dejando para después la decisión de qué clase crear o cómo crearla. Según donde se tome dicha decisión se pueden clasificar los patrones de creación en: patrones de creación de clases; la decisión se toma en los constructores de las clases y usan la herencia para determinar la creación de las instancias.

- **Patrones estructurales:** son los que plantean las relaciones entre clases, las combinan y forman estructuras mayores. Tratan de conseguir que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos. Lo fundamental son las relaciones de uso entre los objetos, y estas están determinadas por las interfaces que soportan los objetos. Estudian cómo se relacionan los objetos en tiempo de ejecución. Sirven para diseñar las interconexiones entre los objetos.
- **Patrones de comportamiento:** plantea la interacción y cooperación entre las clases. Los patrones de comportamiento estudian las relaciones entre llamadas de los diferentes objetos, normalmente ligados con la dimensión temporal.

2.5.2.1 Patrón creacional

Instancia única (Singleton): es un patrón de diseño diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Este patrón garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

2.5.2.2 Patrón estructural

Decorador (Decorator): añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad (Larman, 2003). El patrón Decorador se ve implementado en el comportamiento de Symfony que dispone de un archivo llamado layout.php que contiene las etiquetas <html> y <body>. Este archivo, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en el layout, o si se mira desde el otro punto de vista decora la plantilla.

2.5.2.3 Patrón de comportamiento

Acción (Command): encapsula una petición en un objeto, permitiendo así parametrizar a los clientes con distintas peticiones, encolar o llevar un registro de las peticiones y poder deshacer la operaciones (Larman, 2003). Este patrón se ve ejemplificado en el caso de uso gestionar canal a la hora de modificar los canales, ya que se puede cancelar la modificación si se requiere, y los cambios que se produzcan o no, serán vistos directamente por el usuario solo en el listado cuando actualice la base de datos.

2.6 Diagrama de clases del diseño.

Los diagramas de clases del diseño son un tipo de diagrama estático que describen gráficamente la estructura de una aplicación (Pressman, 2010). Para el desarrollo del subsistema se hace uso de diagramas de clases del diseño, ya que sirven de guía a los desarrolladores al constituir una aproximación del sistema

que se desea implementar. A continuación, se representa el diagrama de clases del diseño correspondiente al caso de uso del sistema Gestionar Canal del subsistema.

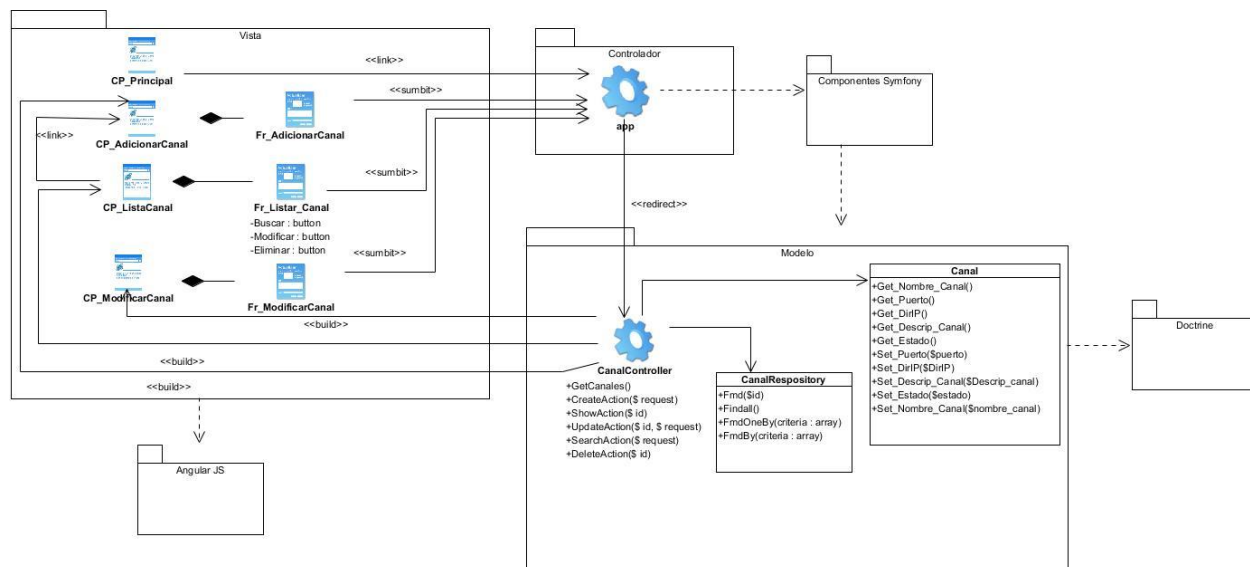


Figura 4 Diagrama de clases del diseño CU Gestionar canal.

2.7 Modelo físico de la Base de Datos.

Para la implementación del subsistema se definió el siguiente modelo físico de la base de datos, el cual esta compuesto por 10 tablas. Estas tablas almacenan toda la información referente al Subsistema de Gestion de Canales Virtuales, y procesan todos los cambios que se ejecuntan en el mismo. Las tablas que lo componen son: Canal, Planificación, Planificación_Espacio_televisivo, Espacio_televisivo, Espacio_televisivo_Activo_Multimedia, Activo_Multimedia, Seg_Usuario, Seg_rol, Seg_Usuario_Seg_rol, y Seg_Permisio.

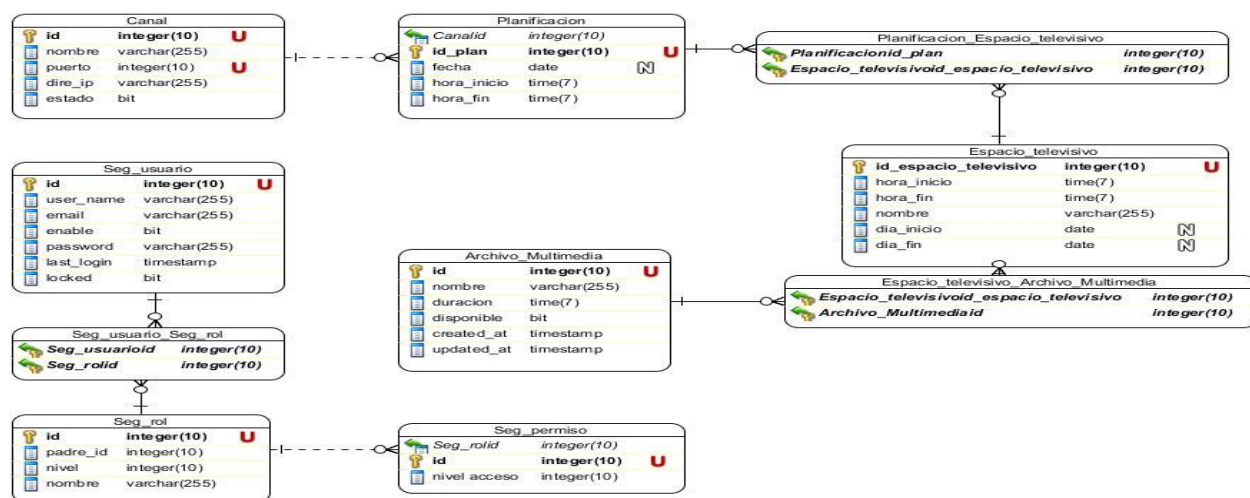


Figura 5 Modelo físico de la Base de Datos CU Gestionar Canal.

Conclusiones parciales.

La confección del modelo de dominio y su descripción permitió visualizar la relación existente entre los conceptos u objetos significativos en el dominio del problema y facilitó la obtención de los requisitos del subsistema. Se realizó el modelado del sistema obteniendo el diagrama de casos de uso del sistema y su especificación, propiciando un mejor entendimiento al equipo de desarrollo del subsistema en cuanto a funcionalidades y características que debe poseer. Los diagramas de clases de diseño y la modelación física de la base de datos del subsistema sirvieron de guía para la implementación del mismo y para su documentación ingenieril. La elección del patrón arquitectónico MVC y la aplicación de patrones de diseño, evidenció la utilización de buenas prácticas durante el desarrollo del subsistema, lo cual incide positivamente en la calidad final del mismo.

Capítulo 3. Implementación y Pruebas

En el presente capítulo se tienen en cuenta todos los aspectos relacionados con la implementación y validación del subsistema. Se muestra la organización del mismo mediante los diagramas de componentes, así como su distribución física mediante el diagrama de despliegue. Se describe el estándar de codificación a tener en cuenta para su implementación; además, se define el proceso de pruebas con el propósito de descubrir y corregir la mayor cantidad de errores en los requisitos implementados.

3.1 Diagrama de componentes.

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Uno de los usos principales es que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema (Larman, 2003). En la siguiente figura se muestra el diagrama de componentes del CU Gestionar Canal.

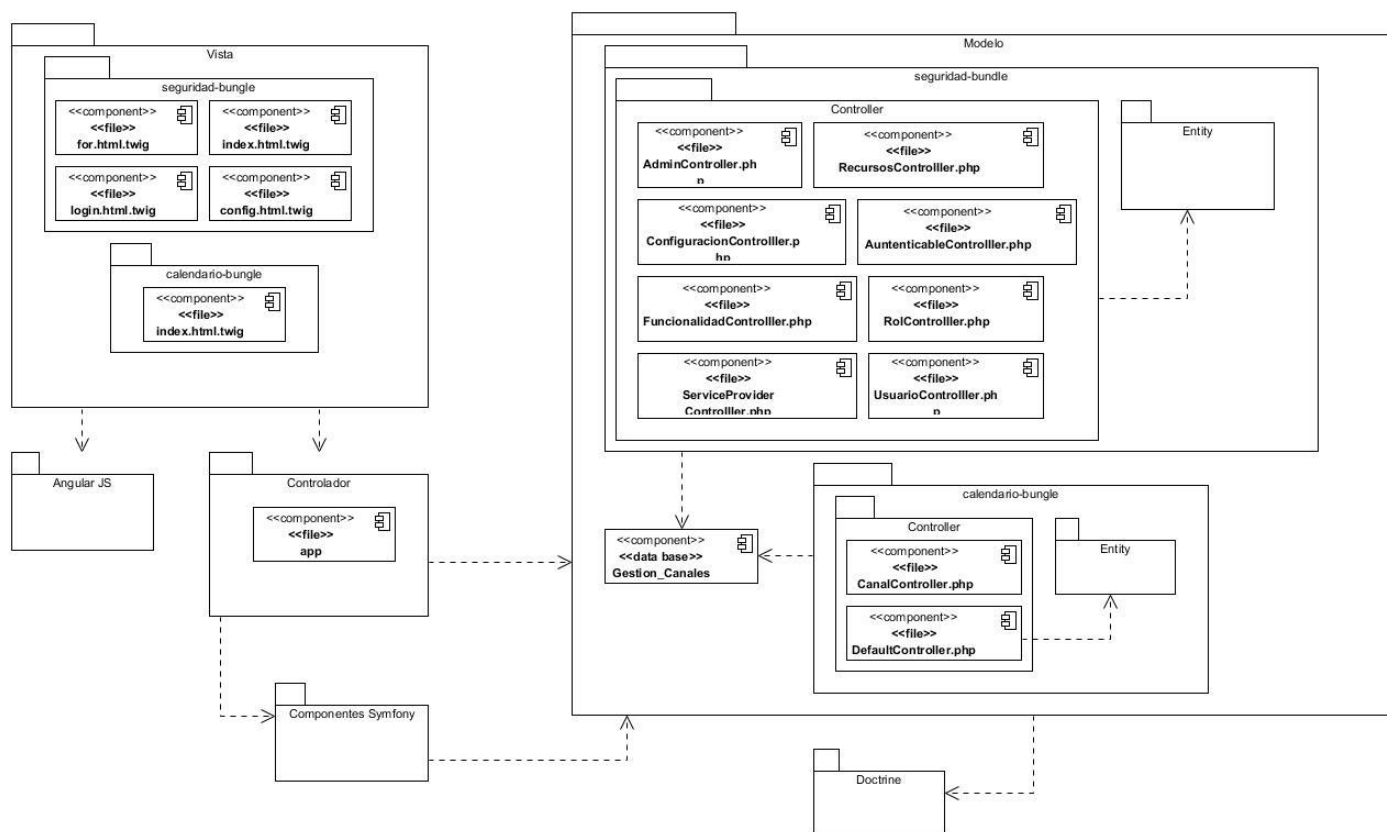


Figura 6 Diagrama de componentes del Subsistema de Gestión Canales Virtuales.

3.2 Estándares de codificación.

Un estándar de codificación comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. En la actualidad existen diferentes estándares de codificación cuya utilización favorece la comunicación fluida entre los desarrolladores, permite la reutilización y el mantenimiento de los sistemas. Para implementar la solución propuesta se hacen uso de los siguientes estándares establecidos por los autores de esta investigación. Dichos estándares se describen a continuación:

- Los nombres de cada uno de los elementos del programa deben ser significativos; su nombre debe explicar en lo posible el uso del elemento, como se muestra:

```
/**
 * Deletes a Canal entity.
 *
 * @Route("/{id}", name="canal_delete", options={"expose"=true})
 * @Method("DELETE")
 */
public function deleteAction($id)
{
    //$em = $this->getDoctrine()->getManager();
    $em = $this->get('doctrine.orm.entity_manager');
    $entity = $em->getRepository('InicioBundle:Canal')->find($id);

    if (!$entity) {
        return new Response('Unable to find Canal entity.', Response::HTTP_NOT_FOUND);
    }
    $em->remove($entity);
    $em->flush();
    return new Response("The Canal with id '$id' was deleted successfully.");
}
```

- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.
- La mayoría de los elementos se deben nombrar usando sustantivos.
- Siguiendo el estándar de codificación empleado en el Sistema STVC, sobre la cual se desarrolla el

subsistema, las declaraciones (atributos, clases y métodos) se escribirán en minúscula, separadas las palabras con guion bajo (_), como se muestra:

```
private function createCreateForm(Canal $entity)
{
    $form = $this->createForm(new CanalType(), $entity, array(
        'action' => $this->generateUrl('canal_create'),
        'method' => 'POST',
    ));
    return $form;
}
```

3.3 Modelo de despliegue.

Un diagrama de despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. Los diagramas de despliegue son los complementos de los diagramas de componentes que, unidos, proveen la vista de implementación del sistema. Describen la topología del sistema la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP. Se utilizan para describir la vista de despliegue estática de un sistema (Larman, 2003). En la figura 12 se muestra el diagrama de despliegue del subsistema implementado.

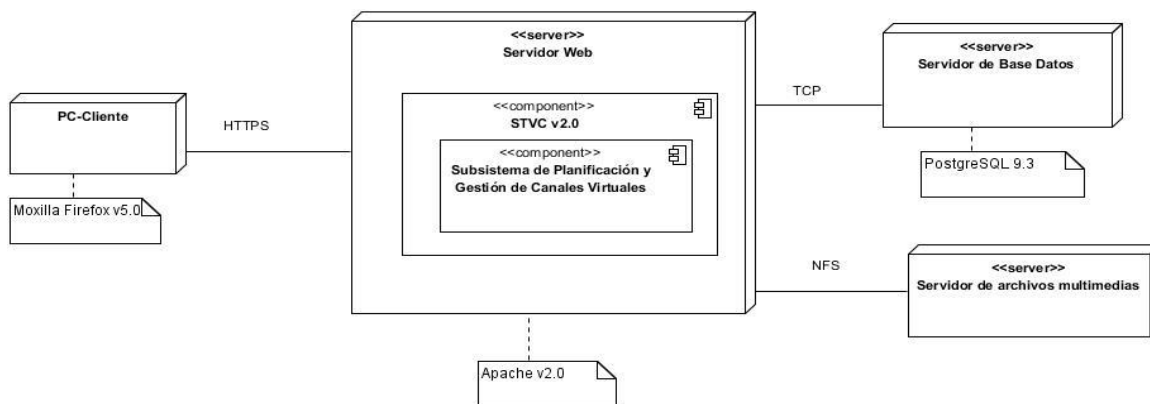


Figura 7 Modelo de despliegue.

3.4.1 Descripción de los componentes físicos del Modelo de Despliegue.

Se representa un nodo PC Cliente el cual tiene como componente el navegador web Mozilla Firefox v5.0. Este nodo se comunica con el servidor web mediante el protocolo HTTPS (Hypertext Transfer Protocol

Secure). A su vez el servidor web debe tener el servidor Apache v2.0 y el Sistema de Transmisión de Canales Virtuales v2.0 para su correcto funcionamiento, dicho sistema incluye Subsistema de Gestión de Canales Virtuales. Además, se comunica con varios nodos como: el Servidor de Base de Datos que debe contener el gestor PostgreSQL v9.3 y con el Servidor de archivos multimedia mediante el protocolo NFS (Network File System). Por último, la PC Cliente puede visualizar la planificación y la gestión de los canales mediante el protocolo HTTPS.

3.4 Modelo de Pruebas.

Durante esta etapa se prueban todas las partes que componen el producto por el equipo de desarrollo, con el objetivo de medir la calidad del subsistema. El proceso de pruebas está encaminado a medir el cumplimiento de las funcionalidades, reduciendo de esta manera el número de errores no detectados; por lo que se desarrollan las pruebas de software.

Las pruebas de software (en inglés software *testing*) son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada. Es una actividad más en el proceso de control de calidad donde se detecta los defectos que pueda tener el software; también se verifica que todos los requisitos se hayan implementado correctamente y verifica que haya una integración adecuada de los componentes. Además identifica y asegura que los defectos encontrados se han corregido antes dar por finalizado el software (Pressman, 2010).

3.5.1 Tipos de pruebas

Para comprobar el correcto funcionamiento del subsistema se aplicaron varios tipos de pruebas planteadas por (Pressman, 2010), las cuales se explican a continuación:

Pruebas de función: ejercitan el contenido y las unidades funcionales dentro de la aplicación y se enfocan sobre un conjunto de pruebas que intentan descubrir errores en la misma. Para realizar las pruebas de función al Subsistema de Gestión de Canales Virtuales serán harán uso de los métodos o pruebas de caja negra y caja blanca.

El **método o prueba de caja negra** se concentra en los requisitos funcionales del software. Permite derivar un conjunto de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa. La partición equivalente es una variante para la prueba de caja negra, la cual divide el dominio de entrada en clases de datos a partir de las cuales pueden derivarse casos de prueba. La partición equivalente se dirige a la definición de casos de prueba que descubran la mayor cantidad de errores, reduciendo así el número total de casos de prueba que hay que desarrollar (Pressman, 2010).

El **método o pruebas de caja blanca** también conocidas como pruebas de caja de cristal o pruebas estructurales se centran en los detalles procedimentales del software, por lo que su diseño está fuertemente

ligado al código fuente. Comprueba los caminos lógicos del software proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el estado deseado o esperado. La prueba de caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba (Pressman, 2010).

3.5 Aplicación y resultado de las pruebas.

A continuación se expondrá todo los resultados de las pruebas realizadas al subsistema desarrollado, dentro de las que están las pruebas de caja negra y caja blanca.

- **Prueba de caja negra**

Aplicando el método de participación de equivalencia se realizaron las pruebas de caja negra, arrojando los resultados siguientes:

La realización de las **pruebas de caja negra** al subsistema requirió del diseño de los casos de prueba en correspondencia con las descripciones de CU. A continuación, se presenta la Descripción del Caso de Prueba (DCP) correspondiente al CU Gestionar Canal del subsistema.

Tabla 5 Descripción de las variables del caso de prueba del caso de uso Gestionar canal.

| No. | Nombre del Campo | Clasificación | Valor Nulo | Descripción |
|-----|-------------------|----------------|------------|--|
| 1 | Nombre canal | Campo de texto | No | Admite cualquier combinación de caracteres alfabética y numérica. |
| 2 | Puerto | Campo de texto | No | Debe introducirse un número entero que esté en el intervalo [1; 65535]. |
| 3 | Dirección IP | Campo de texto | No | Debe introducirse el IP con el formato #. #. #. # (# es un número entero que está en el intervalo [0; 255]). |
| 4 | Descripción canal | Campo de texto | No | Admite cualquier combinación de caracteres alfabética y numérica. |
| 5 | URL logo | Campo de texto | Si | Debe introducirse la dirección donde se encuentran los logos de los canales. |
| 6 | Estado | Checkbox | No | Debe seleccionar un campo. |

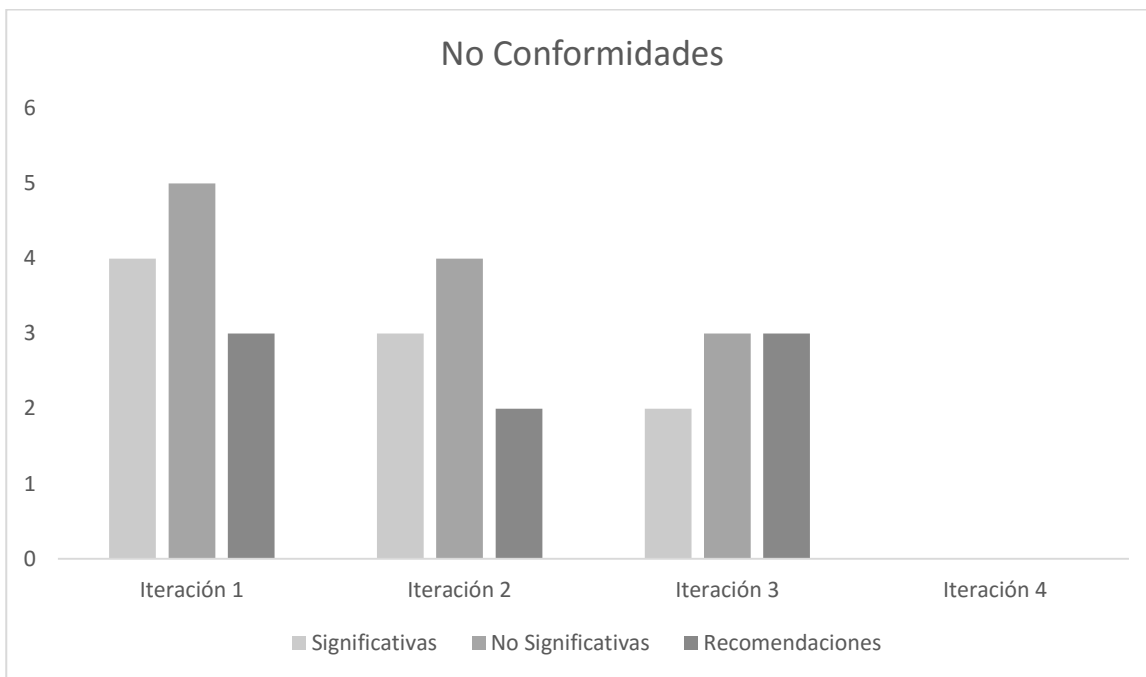
Tabla 6 . Descripción del Caso de Prueba para el Caso de Uso: Gestionar canal. Sección: Adicionar canal.

| Escenario | Descripción | Nombre del canal | Puerto | Dirección IP | Descripción canal | URL logo | Estado | Respuesta del sistema | Flujo central |
|--|---|------------------|--------|--------------|-----------------------------|-----------------------|--------|---|---|
| EC 1.1 Adicionar un canal en el subsistema correctamente. | El usuario introduce todos los parámetros correctamente y se adiciona un canal al subsistema. | V | V | V | V | V | V | Adiciona un canal al subsistema y muestra la lista de los canales existentes actualizada. | Clic en el menú principal en el botón "Gestionar canal"/ clic en el botón "Adicionar"/ introducir los datos del nuevo canal / clic en el botón "Adicionar". |
| | | Cubavisión | 8080 | 10.8.101.103 | Canal de la Familia Cubana. | \\:updatelimagen.jpg | Activo | | |
| EC 1.2 Adicionar un canal en el subsistema con campos vacíos incorrectamente. | El usuario intenta adicionar un canal con campos vacíos. | F | V | V | V | V | V | Mantiene la interfaz de "Nuevo "y muestra un mensaje de error: "No deben existir campos vacíos". En el primer caso "El campo nombre del canal es obligatorio". En el segundo caso "el campo puerto es obligatorio". | Clic en el menú principal en el botón "Gestionar canal"/ clic en el botón "Adicionar"/ clic en el botón "Adicionar". |
| | | vacío | 2320 | 10.8.101.104 | Canal del deporte en Cuba. | \\:updatelimagen1.jpg | Activo | | |
| | | V | F | V | V | V | V | | |
| | | Tele Rebelde | vacío | 10.8.101.105 | Canal del deporte en Cuba. | \\:updatelimagen1.jpg | Activo | | |
| | | V | V | F | F | V | V | | |
| EC 1.3 Adicionar un canal en el subsistema | El usuario intenta adicionar un canal con | Tele Rebelde | 2527 | 101 | ajkfasdjaksdj | \\:updatelimagen1.jpg | Activo | En el primer caso "El IP no está escrito de forma correcta" y "El campo descripción del | Clic en el menú principal en el botón "Gestionar canal"/ clic en el botón "Adicionar"/ introducir los datos del |

| | | | | | | | | | |
|---|---------------------|--------------|-------|--------------|----------------------------|--------------------------|----------|--|---|
| introduciendo datos incorrectos. | campos incorrectos. | V | V | V | V | F | F | canal no debe está escrito de forma incorrecta". ". En el cuarto caso "El campo URL Logo es no está correctamente escrito" y "En el campo estado debe elegir una opción ". El quinto caso "El campo puerto no está correctamente escrito" y " El campo dirección IP esta incorrectamente escrito" | nuevo canal / clic en el botón "Adicionar". |
| | | Tele Rebelde | 2829 | 10.8.101.106 | Canal del deporte en Cuba. | Imágen es, carpeta | Ninguno | | |
| | | V | F | F | V | V | V | | |
| | | Tele Rebelde | Cinco | 255@123 | Canal del deporte en Cuba. | \\:updat e\ imagen 1.jpg | Inactivo | | |

A partir del diseño y ejecución de los casos de prueba, introduciendo juegos de datos, tanto correctos como incorrectos, se detectaron errores tales como: opciones que no funcionan, errores de idioma, validación de los datos de entradas, errores de redacción y funcionalidades incorrectas, para un total de 12 no conformidades.

Figura 8 Gráfico de resultado de las no conformidades.



En el gráfico anterior se evidencian 12 no conformidades detectadas al realizar la primera iteración, estas fueron clasificadas según su nivel de afectación en el funcionamiento del sistema como: significativas, no significativas y recomendaciones. Definidas como significativas aquellas no conformidades que afectaron el funcionamiento del subsistema, las mismas fueron 4 errores de funcionalidad. Como no significativas se detectaron 2 errores ortográficos, 3 errores de mensajes y las restantes fueron recomendaciones o sugerencias de cambios a elementos del diseño. Al concluir la corrección de los errores detectados se dio fin a la primera iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores. Una vez realizada la segunda iteración se detectaron errores significativos 3 de ellos, 4 errores de mensajes en el caso de las no significativas, y el resto fueron sugerencias de cambios para corregir los errores. Al concluir la corrección de los errores detectados se dio fin a la segunda iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores. Por lo que en la tercera iteración se encontraron 8 no conformidades. Por último en la cuarta iteración no se detectaron no conformidades.

- **Prueba de caja blanca**

Aplicando el método del camino básico se realizaron las pruebas de caja blanca, arrojando los resultados siguientes:

La realización de las **pruebas de caja blanca** al subsistema requirió de la prueba del **camino básico**, la cual arrojó los siguientes resultados:

Esta prueba permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

Los pasos que se siguen para aplicar esta técnica son:

1) A partir del diseño o del código fuente, se dibuja el grafo de flujo asociado.

Código fuente:

```

1 { public function createAction(Request $request)
    {
2     [ $entity = new Canal();
        $form = $this->createCreateForm($entity);
        $form->handleRequest($request);
3     ] if ($form->isValid())
        {
4         [ $em = $this->get('doctrine.orm.entity_manager');
            $em->persist($entity);
            $em->flush();
            return new Response('The Canal was created successfully.');
```

```

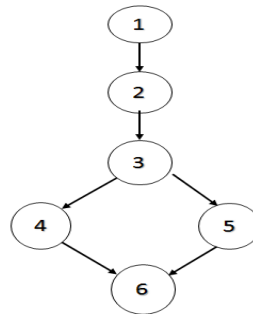
        ]
5     [ $errors = $this->getAllErrorsMessages($form);
        return new Response($this->serialize($errors),Response::HTTP_INTERNAL_SERVER_ERROR);
```

```

6 { }

```


Figura 9 Grafo de flujo asociado al código fuente.



2) Se calcula la complejidad ciclomática del grafo.

$$V(G) = NA \text{ (Número de Aristas)} - NN \text{ (Número de Nodos)} + 2$$

$$V(G) = 6 - 6 + 2$$

$$V(G) = 2$$

3) Se determina un conjunto básico de caminos independientes.

Tabla 7 Trayectoria básica obtenida en el grafo.

| No. de trayectorias | Trayectorias |
|---------------------|---------------|
| 1 | 1, 2, 3, 4, 6 |
| 2 | 1, 2, 5, 6 |

4) Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Tabla 8 Caso de Prueba para la Trayectoria Básica 1.

| Caso de prueba para la Trayectoria Básica 1 | |
|---|--|
| Trayectoria | 1, 2, 3, 4, 6 |
| Descripción de los parámetros de entrada | Se introducen por parámetro las variables para crear el nuevo canal, las cuales contienen los datos del nuevo canal que se quiere obtener. |
| Resultado esperado | Se obtiene un nuevo canal. |

Tabla 9 Caso de Prueba para la Trayectoria Básica 2

| Caso de prueba para la Trayectoria Básica 2 | |
|--|--|
| Trayectoria | 1, 2, 5, 6 |
| Descripción de los parámetros de entrada | Se introducen por parámetro las variables para crear el nuevo canal, las cuales contienen los datos del nuevo canal que se quiere obtener con los datos incorrectos o campos vacíos. |
| Resultado esperado | Muestra un mensaje de error: "No deben existir campos vacíos". |

Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

Conclusiones Parciales.

Con la confección del diagrama de componentes se permitió comprender la vista estática del subsistema, mostrando la organización y dependencia que existe entre los componentes que se necesitan para ejecutarlo. Además, la realización del diagrama de despliegue permitió comprender la distribución física de los distintos componentes lógicos desarrollados. El uso del estándar de codificación permite el entendimiento del código por otros programadores que no sean del equipo de desarrollo en caso de mejoras en el subsistema en un futuro. Con el desarrollo del proceso de pruebas se pudo identificar y resolver los errores en la implementación aumentando la calidad del producto final obtenido. También las pruebas aplicadas demostraron que el mismo cumple con los requisitos trazados.

Conclusiones generales

Con la realización del presente trabajo de diploma se caracterizaron los principales conceptos relacionados a los sistemas de planificación de canales virtuales, permitiendo sentar las bases para el desarrollo del Subsistema de Gestión de Canales Virtuales en su versión 2.0. Además, facilitó una mayor comprensión del objeto para el cumplimiento del objetivo general. También fueron seleccionadas las herramientas, metodología de desarrollo de software y tecnologías a emplear. Se hizo una descripción detallada del subsistema desarrollado, haciendo énfasis en la metodología empleada. Se determinó la arquitectura y patrones arquitectónicos para el subsistema, así como los patrones de diseño utilizados en el subsistema.

Con el desarrollo del subsistema se cumple lo siguiente:

- Permite que la periodicidad de la planificación sea mayor a una semana; por lo que los usuarios del sistema no se ven limitados a la hora de realizar la planificación.
- Realiza cambios en tiempo real en la parrilla de programación; lo cual facilita que si existe algún cambio de último momento pueda realizarse sin inconvenientes.
- Se puede realizar una descripción detallada de los espacios televisivos en la planificación de canales, permitiendo facilidades a la hora de realizar la parrilla de programación.

Por lo anteriormente expuesto, se concluye que el objetivo general de la presente investigación se ha cumplido satisfactoriamente ya que el subsistema da solución a la situación problemática que le dio origen.

Referencias Bibliográficas

Abramson, Albert. 2003. *The History of Television*. North Carolina : s.n., 2003.

Academic Developer Technology Group. 2015. *Academic Developer Technology Group*. [En línea] 2015. [Citado el: 12 de 12 de 2015.] <https://sites.google.com/site/gabineteutn/investigacion-y-desarrollo/html5/tutoriales/introduccion-a-websocket>.

Aguilera López, Purificación y Morente Fernández, María. 2012. *Ofimática y proceso de la información*. 2012.

Alanyali, Murat y Hajek, Bruce. 1997. *Analysis of Simple Algorithms for Dinamic Load Balancing*. 1997.

Alvarez, Carlos de Zayas. 1995. *Metodología de la Investigación Científica*. Santiago de Cuba : s.n., 1995.

Apache. 2016. Apache. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.] http://httpd.apache.org/ABOUT_APACHE.html.

—. 2016. Apache. [En línea] 2016. [Citado el: 22 de 3 de 2016.] http://httpd.apache.org/ABOUT_APACHE.html.

Aquiles, Barreto y Cardinale, Yudith. 2011. *Modelo de balance de carga para un clúster computacional*. 2011.

ArPUG. 2016. ArPUG. [En línea] 2016. [Citado el: 24 de 4 de 2016.] www.arpug.com.ar/trac.

Bartolomé, Antonio, y otros. 2007. *La web audiovisual*. 2007.

Bertino, E A y Martino, L A. 1995. *Sistemas de bases de datos orientadas a objetos*. 1995.

Castillo, Jéssica Izquierdo. 2016. *Teoría de programación de radio y televisión*. Castellón, España : Universitat Jaume, 2016.

Celik, Tantek. 2013. W3C. W3C. [En línea] 2013. [Citado el: 31 de 3 de 2017.] <https://www.w3.org/Style/CSS/>.

Centeno, Vicente Luque. 2003. *Automatización de tareas en la web*. 2003.

Citrix. 2016. Citrix. [En línea] 2016. [Citado el: 2 de 3 de 2016.] www.citrix.com/glossary/load-balancing.html.

Clements, P, Bass, L y Kazaman, R. 2003. *Software Architecture in Practice*. s.l. : SEI Series in Software

Engineering: Addison Wesley, 2003.

Clements, P, Bass, L y Kazman, R. 2003. *Software Architecture in Practice*. . s.l. : SEI Series in Software Engineering: Addison Wesley, 2003.

Computing Careers and Degrees. 2015. Computing Careers and Degrees. [En línea] 2015. [Citado el: 10 de 12 de 2015.] <http://computingcareers.acm.org>.

Converse, Tim, Park, Joyce y Morgan, Clark. 2004. *PHP5 and MySQL Bible*. Indiana : Wiley Publishing, 2004.

Creati. 2016. Creati. [En línea] 2016. [Citado el: 18 de 2 de 2016.] <http://creati.es>.

Cristiá, Maximiliano . 2011. *Introducción a la Ingeniería de Requerimientos*. 2011.

Crockford, Douglas. 2016. JavaScript.com. *JavaScript.com*. [En línea] 2016. [Citado el: 31 de marzo de 2017.] <https://www.javascript.com/>.

Datastax. 2016. Datastax. [En línea] 2016. [Citado el: 2 de 3 de 2016.] www.datastax.com/dev/blog/what-persistence-and-why-does-it-matter.

Date, C J. 2009. *Introducción a los sistemas de Bases de Datos*. 2009.

Drupal. 2016. Drupal. [En línea] 2016. [Citado el: 25 de 1 de 2016.] <http://drupal.org/es/caracteristicas>.

Drupal Hispano. 2016. Drupal Hispano. [En línea] 2016. [Citado el: 24 de 4 de 2016.] <http://drupal.org/es/caracteristicas>.

Flumotion. 2015. Flumotion. [En línea] 2015. [Citado el: 28 de 10 de 2015.] <http://www.flumotion.com/plataforma-de-streaming/>.

García, Alex Ribelles. 2013. *Plataformas de publicación y distribución de audio y video*. Catalunya : s.n., 2013.

Garro, Arkaitz. 2015. *HTML5*. 2015.

Gilster, Ron. 2004. *Guía Completa para PC*. 2004.

Globe. 2016. Globe. [En línea] 2016. [Citado el: 21 de 4 de 2016.] <http://www.globetesting.com/pruebas-de-rendimiento/>.

Granado, Luis Miguel Cabezas. 2004. *Manual Imprescindible de PHP5*. Madrid : Anaya Multimedia,

2004. 84-415-1785-1.

Guerrero, Carlos A, Suárez, Johana M y Gutiérrez, Luz E. 2013. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. [En línea] 2013. [Citado el: 28 de febrero de 2016.] http://www.scielo.cl/scielo.php?pid=S0718-07642013000300012&script=sci_arttext.0718-0764.

Gustavo, Núñez Carvonel. 2010. 2010.

Harmonic. 2015. Harmonic. [En línea] 2015. [Citado el: 26 de 10 de 2015.] [http://www.harmonicinc.com/product/..](http://www.harmonicinc.com/product/)

Hernández, Rolando Alfredo y Coello, Sayda. 2011. *El proceso de investigación científica*. La Habana : Universitaria, 2011.

HTML5. 2015. CSS3, HTML5. [En línea] 2015. [Citado el: 10 de 24 de 2015.] <http://html5.dwebapps.com/que-es-css3/>.

Hugo, Marca y Quisbert, Nancy. 2010. Diagrama de Despliegue. [En línea] 2010. [Citado el: 14 de 3 de 2014.] virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc.

IBM. 2017. IBM Knowledge Center. *IBM Knowledge Center*. [En línea] 2017. [Citado el: 31 de 3 de 2017.] https://www.ibm.com/support/knowledgecenter/es/ssw_aix_72/com.ibm.aix.networkcomm/tcpip_protocols.htm.

IEEE. 2016. *A Model for Types and Levels of Human Interaction*. s.l. : Pueblo y Educación., 2016.

ifm. 2015. ifm. [En línea] 2015. [Citado el: 28 de 10 de 2015.] http://www.ifm.com/ifmmx/web/pinfo015_010_040.htm.

INEI. 1999. *Herramienta Case: Colección Cultura Informática*. s.l. : Talleres de la Oficina de Impresiones de la Oficina Técnica de Difusión del Instituto Nacional de Estadística e Informática (INEI), 1999.

ISID. 2016. ISID. [En línea] 2016. [Citado el: 25 de 4 de 2016.] www.isid.com.

Jacobson, Ivar, Booch, Grady y Ramburg, James. 2000. *El Proceso Unificado del Desarrollo de Software*. Madrid : Addyson Wesley, 2000.

Javier Isidro, Becerril, y otros. 2012. *La web 2.0: un análisis de su impacto en lo social, político.*, 2012.

JetBrains. 2014. PHP Storm. [En línea] 2014. [Citado el: 20 de noviembre de 2015.]

<https://www.jetbrains.com/phpstorm/whatsnew/>.

Jiménez, Javier Bustos. 2010. *Balance de Carga Dinámico*. Chile : Departamento de Ciencias de la Computación, 2010.

jWebSockets. 2015. jWebSockets. [En línea] 2015. [Citado el: 22 de noviembre de 2015.]
<https://jwebsocket.org/>.

Kalle Lyytinen, V.-P. Tahvanainen. 1992. *Next Generation CASE Tools*. Netherlands : s.n., 1992.

Keidar, Idit y Rajsbaum, Sergio . 2013. *ACM SIGACT News Distributed*. 2013.

Kioskea. 2016. Kioskea. [En línea] 2016. [Citado el: 3 de febrero de 2016.] es.kioskea.net.

Kruchten, Philippe. 2003. *Rational Unified Process, The: An Introduction, Third Edition*. s.l. : Addison Wesley, 2003. ISBN: 0-321-19770-4.

—. **2003.** *Rational Unified Process, The: An Introduction, Third Edition*. s.l. : Addison Wesley, 2003. ISBN: 0-321-19770-4.

Larman, Craig. 2003. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*. 2003.

—. **2003.** *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*. 2da Edición. 2003.

Larousse. 2016. Larousse. [En línea] 2016. [Citado el: 23 de 2 de 2016.] <http://www.larousse.mx>.

Lenguajes de Programación. 2015. Lenguajes de Programación. [En línea] 2015. [Citado el: 3 de 12 de 2015.] <http://www.lenguajes-de-programacion.com>.

Libros Web. 2016. Libros Web. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.]
http://librosweb.es/libro/symfony_2_x/.

LifeSize. 2015. LifeSize. [En línea] 2015. [Citado el: 20 de 10 de 2015.] www.lifesize.com.

López, Alberto Tallón. 2011. *Microlopez*. 2011.

Luis Miguel Cabezas Granado. 2004. *Manual Imprescindible de PHP5*. Madrid : Anaya Multimedia, 2004. 84-415-1785-1.

Luzua, Guillermo Farias y Díaz, Abel Berengue. 2013. *PLATFORM MANAGEMENT AND*

TRANSMISSION OF DIGITAL TV CHANNELS. La Habana : s.n., 2013.

María, Luigi Santa. 2016. staffdigital. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.]
<https://www.staffdigital.pe>.

Mejuto, Jesús y Verdera, Marc . 2013. *Estudio de Caso*. 2013.

Microsoft. 2015. Microsoft. [En línea] 2015. [Citado el: 6 de marzo de 2017.]
<http://msdn.microsoft.com/es-es/library/bb972240.aspx..>

—. **2015.** Microsoft. [En línea] 2015. [Citado el: 2016 de febrero de 23.] <http://msdn.microsoft.com/es-es/library/bb972240.aspx..>

Millo Sánchez, Reinier, y otros. 2015. *Propuesta de SET-TOP Box cubano empleando alternativas de software libre*. La Habana : Revista Digital de las Tecnologías de la Información y las Telecomunicaciones, 2015. Vol. 15.

Mosaic. 2004. Mosaic. [En línea] 2004. [Citado el: 15 de 12 de 2015.]
<http://mosaic.uoc.edu/2004/11/29/introduccion-a-los-sistemas-de-gestion-de-contenidos-cms-de-codigo-abierto/>.

Mozilla. 2015. Mozilla. [En línea] 2015. [Citado el: 15 de 12 de 2015.]
https://developer.mozilla.org/es/docs/Web/API/WebRTC_API.

Müller, Luis. 2011. *Programación Concurrente*. 2011.

Pedregosa, Jaume. 2014. mosaic. *mosaic*. [En línea] 2 de Diciembre de 2014. [Citado el: 21 de Noviembre de 2016.] <http://mosaic.uoc.edu>.

PHP. 2016. PHP. [En línea] 2016. [Citado el: 28 de 1 de 2016.] php.net.

postgreSQL. 2016. pgAdmin postgresQL tools. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.]
<https://www.pgadmin.org>.

PostgreSQL. 2016. PostgreSQL. [En línea] 2016. [Citado el: 20 de 1 de 2016.]
<http://www.postgresql.org/about/>.

Poulin, Michel y Robin, Michael. 2013. *Digital Television Fundamentals*. New York : s.n., 2013.

Powers, John, y otros. 2006. *Distributed processing system*. 2006.

Pressman, Roger. 2010. *Ingeniería de Software. Un enfoque Práctico*. 2010.

- . 2002. *Ingeniería del Software: Un enfoque Práctico*. 2002.
- Pressman, Roger S. 2005.** *Ingeniería del Software*. 2005.
- Pressman, Roger S y Maxim, Bruce R. 2014.** *Software Engineering: A Practitioner's Approach*. 2014.
- Putti, Srinivasrao , y otros. 2013.** *Scalable Distributed Job Processing with Dynamic*. 2013.
- Qin, Hang, ZHU, Li y WU, Zhongbo. 2009.** *Dynamic Load Balancing with Overlay-Based Reconfiguration for Wireless Sensor Networks*. 2009.
- RAE. 2016.** RAE. [En línea] 2016. [Citado el: 30 de 3 de 2016.] <http://www.rae.es/>.
- Ramos, Kariné Blanco. 2013.** *Proceso Base de Ingeniería de Requisitos para las pequeñas y medianas empresas de desarrollo de software*. Cuba : s.n., 2013.
- . 2013. *Proceso Base de Ingeniería de Requisitos para las pequeñas y medianas empresas de desarrollo de software*. Cuba : s.n., 2013.
- Rascón, Marco. 2010.** La televisión cubana. *Instituto Cubano de Radio y Televisión*. 2010.
- Real Academia Española. 2016.** Real Academia Española. [En línea] 2016. [Citado el: 30 de 3 de 2016.] <http://www.rae.es/>.
- Reyes, Yunia, Gonzáles, Lisbet y Ruiz, Yadira. 2008.** *Estudio comparativo sobre las principales metodologías pesadas y orientadas a objetos en el desarrollo de software*. Habana : s.n., 2008.
- Rodríguez, Fran Gil. 2012.** *Experto en Drupal 7 Nivel Inicial*. 2012.
- S. Dormido, R. Hernández, S. Ros y J. Sánchez. 2003.** *Procesamiento Paralelo. Teoría y Programación* . Madrid : s.n., 2003.
- Sanchez, Miguel Angel. 2001.** *JavaScript*. s.l. : INNOVA 2001, 2001.
- Sánchez, Tamara Rodríguez. 2015.** *Metodología de desarrollo para la Actividad productiva de la UCI*. Habana : s.n., 2015.
- Savitz, Eric. 2015.** *Forbes*. 2015.
- Skvorc, Bruno. 2015.** Best PHP IDE in 2015 - Survey Results. [En línea] 2015. [Citado el: 15 de noviembre de 2015.] <http://www.sitepoint.com/best-php-ide-2014-survey-results/>.
- Somerville, Ian. 2009.** *Ingeniería del Software Séptima Edición*. 2009.

- Sommerville, Ian. 2005.** *Ingeniería del Software Séptima Edición*. Madrid : Pearson Educación.SA, 2005.
- SparxSystems. 2007.** Diagrama de Componentes UML 2. [En línea] Sparx Systems Pty Ltd., 2007. [Citado el: 11 de marzo de 2016.]
http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.
- Stallings, William. 2003.** *Sistemas Operativos*. Universidad Pontificia de Salamanca en Madrid : s.n., 2003.
- Streaming Media. 2016.** Streaming Media. [En línea] 2016. [Citado el: 13 de 4 de 2016.]
<http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-Encoding-and-Transcoding-75025.aspx>.
- Sullivan, Sean . 2010.** *Programming WebSockets*. 2010.
- symfony. 2016.** symfony. [En línea] 2016. [Citado el: 23 de 03 de 2017.] <http://symfony.es/pagina/que-es-symfony/>.
- Tanenbaum, Andrew S y Van Steen, Maarten. 2008.** *Distributed Systems: Principles and Paradigms*. 2008.
- Tecnología Streaming para radio digital Universitaria. Acevedo, Edwin jeovanny Clavijo, Hernández Chacón, Sorey y Cardoza Vásquez, Edison. 2015.* Bucaramanga-Colombia : Revista ESCAICA, 2015, Vol. 1.
- Tedial. 2015.** MPM. [En línea] 2015. [Citado el: 1 de diciembre de 2015.]
<http://www.tedial.com/products/mpm/media-automation>.
- Tejedor, Jesús Ramón Milian. 2014.** *WebRTC (Web Real-Time Communications)*. 2014.
- Telestream. 2015.** Telestream. [En línea] 2015. [Citado el: 1 de 12 de 2015.]
<http://www.telestream.net/vantage>.
- Tim Converse, Joyce Park, Clark Morgan. 2004.** *PHP5 and MySQL Bible (Bible)*. 2004.
- Tinetti, Fernando y Wolfmann, Gustavo. 2008.** *Análisis de Paralelización con Memoria Compartida y Memoria Distribuida*. La Plata : s.n., 2008.
- Tomasi, Wayne. 2010.** *Sistemas de Comunicaciones Electrónicas, 4ta Edición*. México : s.n., 2010.
- tools, pgAdmin postgresQL. 2016.** pgAdmin postgresQL tools. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.] <https://www.pgadmin.org>.

Traitement des images par le contenu. 2016. Traitement des images par le contenu. [En línea] 19 de enero de 2016. [Citado el: 19 de enero de 2016.] <http://www.irisa.fr/texmex/index.htm>.

UCOL. 2015. Herramientas Case. [En línea] 2015. [Citado el: 15 de 12 de 2015.] http://docente.ucol.mx/al961223/public_html/centro6.htm.

UML. 2016. UML. [En línea] 2016. [Citado el: 19 de 1 de 2016.] <http://www.uml.org/>.

Ustream. 2016. Ustream. [En línea] 2016. www.ustream.tv.

Vega, John Freddy y Van Der Henst, Christian. 2011. *Guía HTML5, el presente de la web*. 2011.

Venete, Adriana. 2011. *Introducción a los Patrones de Arquitectura*. 2011.

Versión 2.4 del Servidor HTTP Apache. 2016. Versión 2.4 del Servidor HTTP Apache. [En línea] 2016. [Citado el: 20 de 1 de 2016.] <http://httpd.apache.org/docs/2.4/>.

Visual Paradigm. 2015. Visual Paradigm. What VP - UML Provides. [En línea] 2015. [Citado el: 20 de 10 de 2015.] <http://www.visual-paradigm.com/product/vpuml/provides>.

VSN. 2016. VSN. [En línea] 2016. [Citado el: 30 de 5 de 2016.] <https://www.vsn-tv.com/es>.

—. 2015. VSN innovation y media solutions. [En línea] 2015. <https://www.vsn-tv.com>.

Vyemura, John P. 2007. *Diseño de Sistemas Digitales. Un enfoque integrado*. 2007.

W3C. 2016. W3C. [En línea] 2016. [Citado el: 18 de 5 de 2016.] <https://www.w3.org>.

Wattenhofer, Roger. 2016. *Principles of Distributed Computing*. 2016.

Web, Libros. 2016. Libros Web. [En línea] 2016. [Citado el: 23 de Noviembre de 2016.] <http://librosweb.es>.

Websocket. 2016. *Websocket*. [En línea] 2016. [Citado el: 12 de 12 de 2016.] <http://www.websocket.org>.


William, Bedoya John Branch, y otros. 2009. *Proposición de un método para balanceo de carga en un clúster*. Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV)– Unidad : s.n., 2009.

XTREAM. 2016. XTREAM. [En línea] 2016. [Citado el: 30 de 5 de 2016.] <http://www.xtreamsig.com>.

Anexos


Anexos 1: Notificación de aceptación en evento.

LATIN AMERICAN AND CARIBBEAN CONSORTIUM OF ENGINEERING INSTITUTIONS
CONSORCIO DE ESCUELAS DE INGENIERÍA DE LATINOAMÉRICA Y DEL CARIBE




Boca Ratón, FL, USA, May 15, 2017


LACCEI partner Organizations




IFEES
International Federation of Engineering Education Societies




OAS
Organization of American States




EftA
Engineering for the Americas




Asibeit
Asociación Iberoamericana de Instituciones de Enseñanza de la Ingeniería




American Association for Engineering Education




Ister
Iberoamerican Science, Technology and Education Consortium




Réseau Caribbéen de Ingénierie




IGIP
Internationale Gesellschaft für Ingenieurpädagogik



SEFI
Société Européenne pour la Formation des Ingénieurs



JEE
Journal for Engineering Education



SPEED
Student Partners for Engineering Education Development

Yudeisy Valdés Fernández
Universidad de las Ciencias Informáticas
La Habana, Cuba.

Re: Letter of Invitation for Yudeisy Valdes Fernandez (CUBA), to participate in the LACCEI 2017 International Multi-Conference (July 19-21, 2017), Boca Raton, FL - USA, its pre-conference sessions (July 17-18, 2017) and in the 2nd OAS Summit of Engineering for the Americas. **Source:** <http://wp.eng.fau.edu/laccei/>

Dear Prof. Yudeisy Valdes,

I have the pleasure to invite you to participate in the XV International Multi-Conference of LACCEI 2017 and to also celebrate with us the XV Anniversary of the founding of LACCEI here at the Florida Atlantic University Campus and to attend our annual conference (July 19-21, 2017) and pre-conference sessions (in July 17-21, 2017).

Moreover, the 2nd OAS Summit of Engineering for the Americas will be a fantastic opportunity for you to exchange ideas and engage in networking sessions with many members of Latin American governments that are involved in engineering education and university research projects. Engineering for the Americas is an initiative adopted by the Ministers of Science and Technology of the 34 member countries of the Organization of American States (OAS). It is headquartered at the OAS in Washington, D.C., USA and reports annually to the Ministers.

The OAS Engineering for the Americas (EftA) Summit will be held on Wednesday July 19th. We invite you to be part of this event and attend the rest of the activities of LACCEI and the 2nd OAS Summit of Engineering for the Americas.

To attend the LACCEI 2017 International Conference, as it is customary, you will have to cover your own travel and lodging expenses and you are expected to give a presentation. A second email will be sent to you with further details on the time of your presentation. We look forward to further coordinate your speech as one of this year's parallel session presenters.

The Summit of Engineering for the Americas is included as part of the registration of the LACCEI 2016 International Multi-Conference and will have a simultaneous translation into English or Spanish. All events are held at Florida Atlantic University.

We recommend that you make your hotel reservations before May 30th, at the Wyndham Hotel Boca Raton, where there is an available and affordable fine dining restaurant and it is the official hotel sponsor for LACCEI 2017 Multi-Conference and the OAS Summit. We hope to welcome you at the opening Session of LACCEI 2017 and the OAS Conferences.

Important links:

- **LACCEI 2017 International Multi-conference Website:**
<http://wp.eng.fau.edu/laccei/>
- **Wyndham Boca Raton Hotel:**
<http://wp.eng.fau.edu/laccei/wyndham/>
<http://www.wyndhamboca.com/>
Email: aberrio@wyndhamboca.com


Best Regards,



Maria Larrondo Petrie, PhD
Directora Ejecutiva de LACCEI



Anexos 2: Avaes otorgados en eventos científicos.


Universidad de las Ciencias
Informáticas

**Universidad de las Ciencias Informáticas
Federación Estudiantil Universitaria**

Otorga el presente:
Reconocimiento

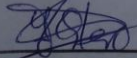
A: Yudisy Valdés Fernández
Osmin García Pardo

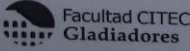
Por haber obtenido la categoría de: Doutorado en la
Jornada Científica Estudiantil


Curso 2016-2017

"El futuro de nuestra patria tiene que ser necesariamente un futuro de hombres de ciencia, tiene que ser un futuro de hombres de pensamiento, porque precisamente es lo que más estamos sembrando; o que más estamos sembrando son oportunidades a la inteligencia (...)"

Fidel Castro Ruz


Yoelvis Quintero Díaz
Presidente de la FEU-CITEC


Facultad CITEC
Gladiadores


Yadira Barroso Rodríguez
Vice-Decana de Formación

Anexos 3: Diagramas de clases del diseño.

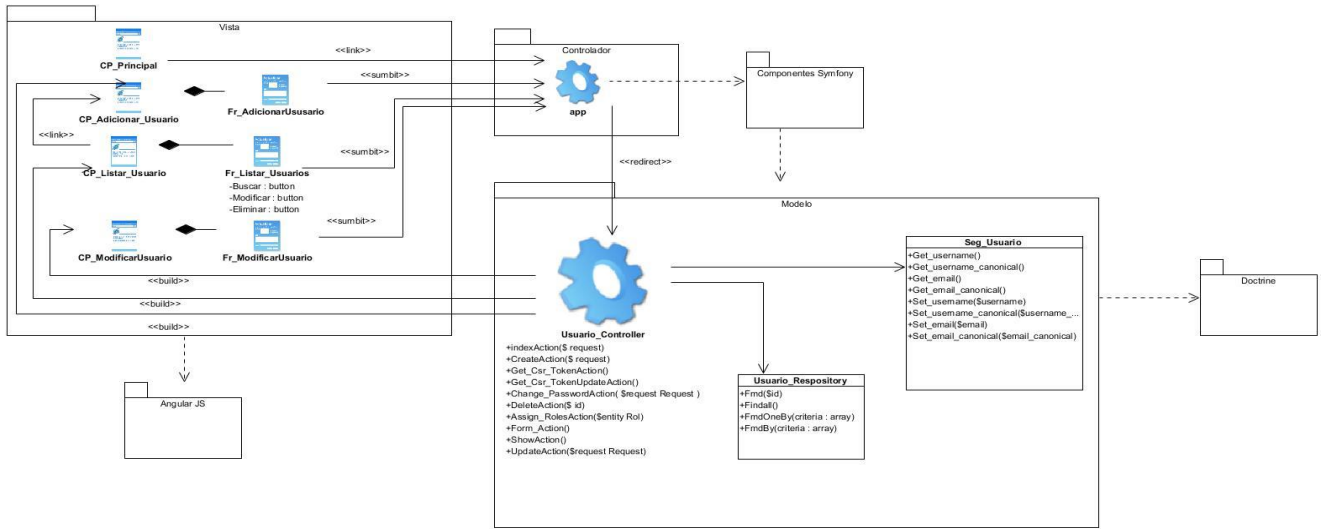


Figura 10 Diagrama de clases del diseño del Caso de Uso Gestionar Usuario.

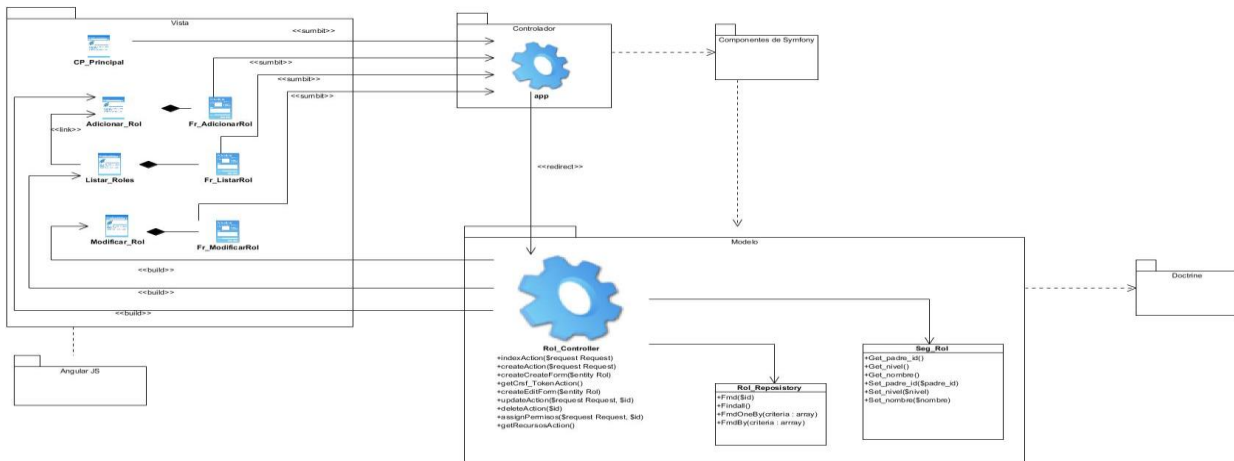


Figura 11 Diagrama de clases del diseño del Caso de Uso Asignar Roles.

Anexos 4: Diagramas de secuencia.

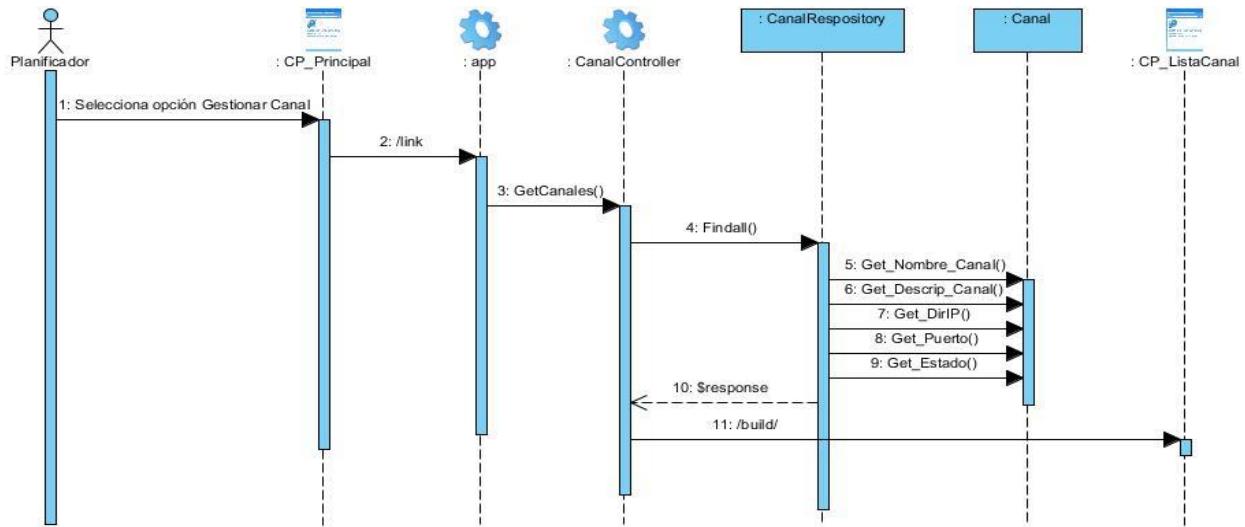


Figura 12 Diagrama de Secuencia CU Gestionar Canal escenario Listar Canal.

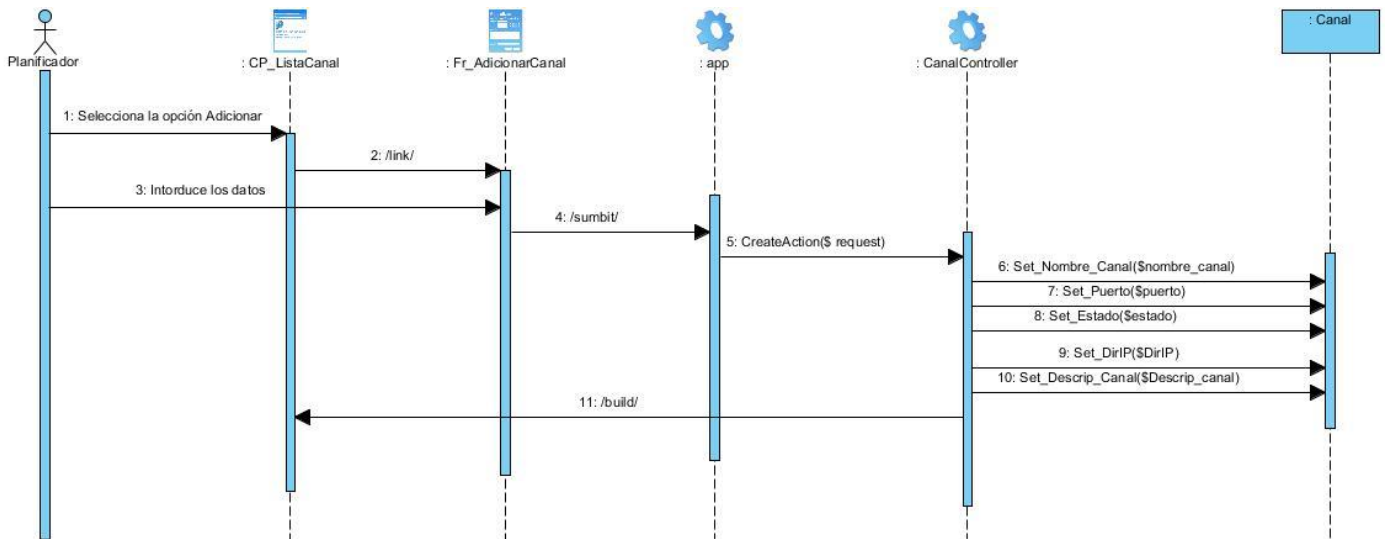


Figura 13 Diagrama de Secuencia CU Gestionar Canal escenario Adicionar Canal.

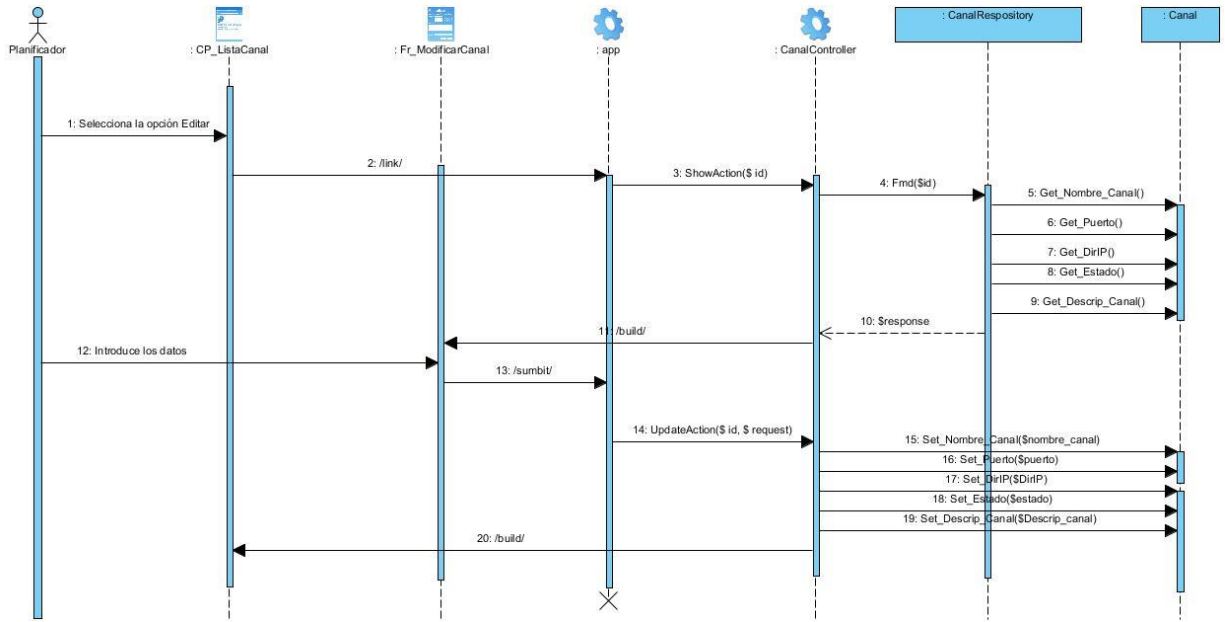


Figura 14 Diagrama de Secuencia CU Gestionar Canal escenario Modificar Canal.

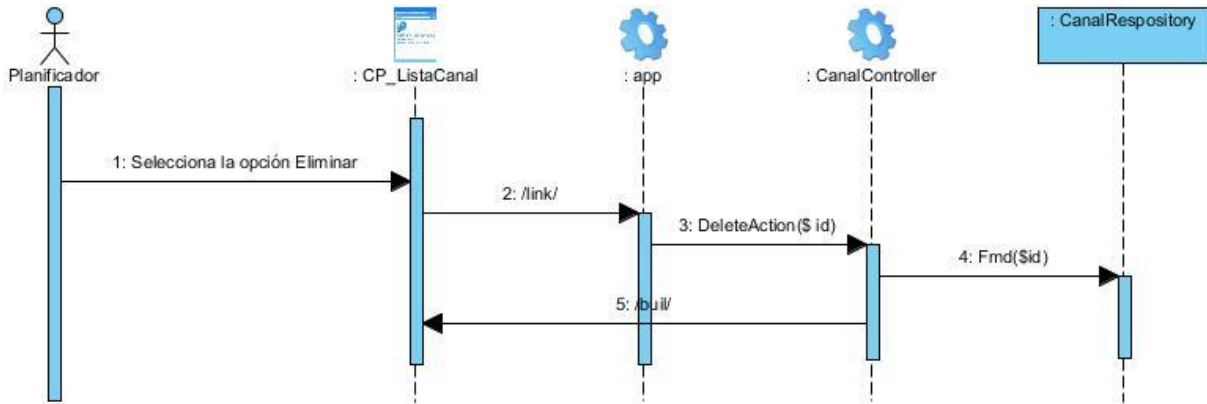


Figura 15 Diagrama de Secuencia CU Gestionar Canal escenario Eliminar Canal.

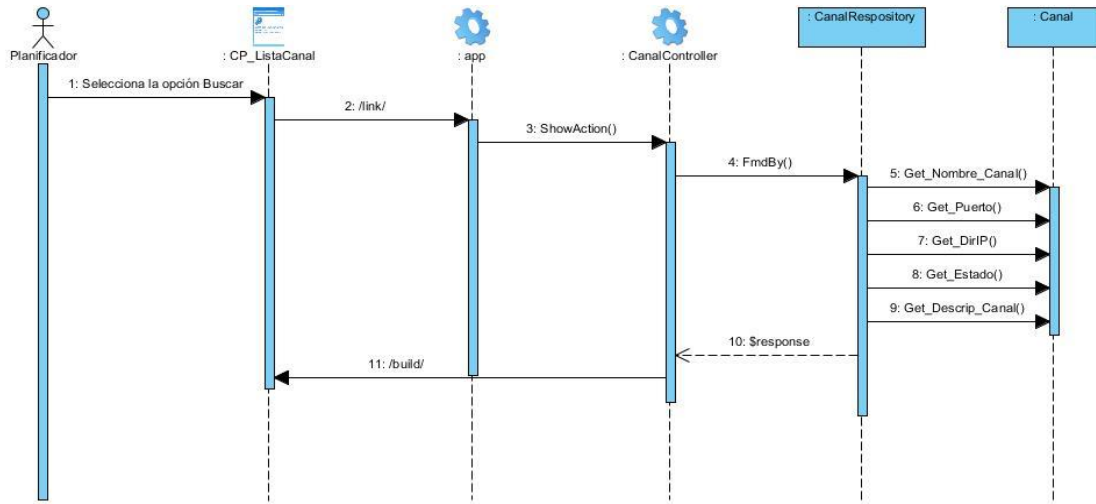


Figura 16 Diagrama de Secuencia CU Gestionar Canal escenario Buscar Canal.