Universidad de las Ciencias Informáticas

FACULTAD 6

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

Título: Estación integrada a un sistema de laboratorios virtuales y a distancia para la práctica de microbiología en la carrera de enfermería.

Autores: Daldis Gómez García

Gustavo Moré Fernández Guzmán

Tutores: MSc. Omar Mar Cornelio

Ing. Bárbara Bron Fonseca

La Habana, junio 2015

"Año 57 de la Revolución"

“Prefiero morir de pie, a vivir arrodillado.”

Che.

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autores:

_____	_____
Firma de autor	Firma de autor
Daldis Gómez García	Gustavo Moré Fernández Guzmán

Tutores:

_____	_____
Firma de tutor	Firma de tutor
MSc. Omar Mar Cornelio	Ing. Bárbara Bron Fonseca

Tutores:

MSc. Omar Mar Cornelio

- Universidad de las Ciencias Informáticas
- La Habana, Cuba
- Correo: omarmar@uci.cu

Ing. Bárbara Bron Fonseca

- Universidad de las Ciencias Informáticas
- La Habana, Cuba
- Correo: bbron@uci.cu

Autores:

Daldis Gómez García.

- Universidad de las Ciencias Informáticas
- La Habana, Cuba
- Correo: dgomezg@estudiantes.uci.cu

Gustavo Moré Fernández Guzmán.

- Universidad de las Ciencias Informáticas
- La Habana, Cuba
- Correo: gmfernandez@estudiantes.uci.cu

Agradecimientos

Primeramente quiero agradecerles a las personas más importantes en mi vida, empezar por mis viejos, mis abuelos "Papa Dago" y "Mamá Mayi", más que abuelos fueron y son padres para mí, ellos que siempre estuvieron ahí enseñándome, forjándome tal cual soy hoy. Gracias, no existe forma ni en un millón de años de pagarle todo lo que hicieron por mí. Continuo con mi madre, esa persona tan especial, "*La mejor mamá el mundo*", ella que siempre me apoyó en cada decisión que tome, siempre estuvo para decirme que continuara, que no importaba, para darme aliento después de cada caída, para ser mi consejera y confidente. Gracias, si hoy he llegado a esto es gracias a ustedes.

A una personita muy especial, a mi novia querida, ella que ha estado este último año para apoyarme y darme aliento para superar cada montaña. A mí papa, a mi tía Dailen, mi tía Ana, a mi hermana Darlen, gracias por darme esa gordita tan linda, mi sobrinita, al resto de mis tías y tíos que han estado siempre ahí. A mis hermanitos, Daniela y "Pombo" y a mi hermano mayor Ariel.

A mis amigos de tantos años a Tito, Renides, Rolando, Eduardo, Trujillo, Eliecer, Eiler, Fernando y muchos otros que me han acompañado en cada travesía. A la gente del aula que tanto moleste durante cada prueba o problema.

A mi tribunal, por ayudarme y soportarme día tras día. A los tutores y al resto de los que me ayudaron de una forma u otra.

Daldis.

Primero que todo agradecerles a mis padres, ya que sin ello no hubiese terminado nunca esta carrera, por todo el apoyo que me han dado. A mi familia por siempre haber confiado en mí, en especial a mis tías y mi hermano. A mi novia por su dedicación y comprensión. A mi dúo de tesis por ser tan paciente. A mis amigos, los viejos y los que he hecho en esta escuela. A todos los que de alguna forma me han ayudado a ser lo que soy hoy, gracias, esto es de ustedes.

Gustavo.

Dedicatoria

“Le dedico esta tesis a mi familia, en especial a mis abuelos y a mi mamá.”

Daldis.

“A mis padres que todo me lo han dado.”

Gustavo.

Resumen

Las prácticas de laboratorios para el estudio de la microbiología juegan un papel fundamental en la carrera de enfermería. Sin embargo no se cuenta en el país con los recursos necesarios para establecer un laboratorio en cada centro educativo donde se imparta esta materia. Teniendo en cuenta el problema anterior, la presente investigación, describe una solución a la problemática planteada a partir de la implementación de una aplicación que, desplegada en una estación de trabajo, permitirá la visualización de las muestras colocadas en el microscopio de forma local o de forma remota a través del Sistema de Laboratorios Virtuales y a Distancia. Para el proceso de implementación de la solución planteada fue utilizada la metodología OpenUp, generando los artefactos pertinentes en cada fase y culminando con las pruebas para comprobar su correcto funcionamiento. Gracias al desarrollo de la presente solución los estudiantes de la carrera pueden observar la evolución de las muestras de microorganismos en tiempo real a través del Sistema de Laboratorios Virtuales y a Distancia, eliminando la necesidad de habilitar un laboratorio en cada centro educativo. Esto trae asociado un ahorro considerable al país, además de erradicar completamente el riesgo de accidentes al estar en contacto con microorganismos peligrosos. Con esta solución se crearon las bases para el desarrollo de nuevas funcionalidades a incluir en el módulo de microbiología.

Palabras clave: estación de trabajo, microbiología, muestras, Sistema de Laboratorios Virtuales y a Distancia.

Abstract

The laboratory practice for the microbiology study play a key role in nursing career. However in the country are not available the necessary resources to establish a laboratory in each school where teach this subject matter. Because of this, the present research describes a solution to the issues raised from the implementation of a workstation for displaying samples of microorganism and its integration into the system of virtual laboratories and distance. For the development of the proposed solution it was used the methodology OpenUP generating relevant artifacts at every stage and ending with tests to verify proper operation. Because of this race students can observe the evolution of samples of microorganisms in real time through Virtual Labs System and Distance, eliminating the need to have a lab in each school. This brings considerable savings associated with the country, and fully eliminate the risk of accidents when in contact with dangerous microorganisms. With this solution the foundation for the development of new features included in the microbiology module is created.

Keywords: microbiology, workstation, samples, Virtual and Distance Labs System.

Índice

Índice de figuras

Índice de tablas

INTRODUCCIÓN

El desarrollo de las Tecnologías de la Información y las Comunicaciones (TICs) ha potenciado un cambio dramático en las formas de enseñanza, debido a la posibilidad de eliminar las limitaciones que suponía la distancia física entre el estudiante y el profesor. Además de posibilitar el acceso a grandes volúmenes de información de forma rápida y eficiente. Siendo posible en gran medida, al progreso de las tecnologías Web y al auge que con ello ha alcanzado Internet, encabezando la “red de redes” el ranking de medios para la obtención de información.

Las posibilidades que brinda Internet son muy variadas, lo que ha repercutido en la creación e implantación de nuevas y variadas formas de educación, entre las que se encuentra la “Educación a distancia”, constituyendo una nueva vía para realizar el proceso educativo. Por otro lado permite al estudiante la obtención de conocimientos a través de materiales de estudio que le son enviados por correo postal o electrónico o son colocados en sitios Web especializados en el asunto.

Entre las ventajas fundamentales de la educación a distancia se encuentran el poder incorporar herramientas como las plataformas o sistemas virtuales, tales como los Sistemas de Laboratorios Virtuales y a Distancia (SLVD). Los SLVD permiten controlar de forma remota estaciones de trabajo y los distintos dispositivos que la componen, además de poder recrear el comportamiento de diversos fenómenos a través de modelos matemáticos (López et al. 2009). Además permiten la visualización de los resultados obtenidos, tanto de la interacción con los dispositivos físicos como con los simulados.

Los experimentos en los SLVD son más baratos en términos de tiempo, dinero y energía que los modelos tradicionales, ya que no precisan de equipamiento técnico en cada una de las infraestructuras docentes. Utilizando una estación centralizada y debidamente equipada se realizan todas las conexiones vía internet o intranet. Esta característica exige que estos sistemas se encuentren accesibles desde cualquier localización del planeta (Ferre et al. 2010). Incluyendo además entre sus características fundamentales que los estudiantes no interactúan directamente con el dispositivo, lo cual elimina el riesgo de accidentes.

Debido a las bondades que brinda este tipo de tecnología, en el país se desea extender su uso a las ciencias médicas, siendo de gran importancia en la carrera de enfermería, enfocándose fundamentalmente en el estudio de la microbiología y las prácticas de laboratorio asociadas a esta. Esto posibilitaría la visualización de la evolución de las muestras en tiempo real, lo que conlleva a un mejor

entendimiento de sus procesos naturales que tienen lugar en microorganismos como los virus, bacterias, parásitos y hongos, así como la observación de tejidos y células.

Sin embargo luego de realizar un estudio de la situación actual de los centros educativos del país fueron identificadas las siguientes insatisfacciones:

- No existe una estación de trabajo que permita visualizar las muestras colocadas en microscopios remotos desde un SLVD.
- La imposibilidad de integrar armónicamente la estación de trabajo para microbiología a un SLVD.
- No existen en el mercado herramientas que permitan extender y reutilizar funcionalidades para elaborar las estaciones de trabajo transmisoras.

Luego de realizar el estudio se define como **problema de la investigación:** ¿Cómo contribuir a la mejora de la realización de las prácticas de la microbiología en la carrera de enfermería?

Teniendo como **objeto de estudio** de la investigación: Sistema de Laboratorios Virtuales y a Distancia, enmarcado en el **campo de acción**: Estaciones de trabajo que permiten las prácticas de la microbiología en un Sistema de Laboratorios Virtuales y a Distancia.

Para dar respuesta al problema antes planteado, se tiene como **objetivo general:** Desarrollar un sistema que integre una estación de trabajo al Sistema de Laboratorios Virtuales y a Distancia que permita las prácticas de microbiología en la carrera de enfermería.

Preguntas científicas:

1. ¿Cuáles son los fundamentos teóricos que rigen el desarrollo de los Sistemas de Laboratorios a Distancia y sus estaciones de trabajo?
2. ¿Cuáles son las tecnologías y mecanismos necesarios para el desarrollo de una solución informática que permita integrar una estación de trabajo al Sistema de Laboratorio Virtuales y a Distancia?
3. ¿Cómo estructurar el proceso de desarrollo de una solución informática que permita integrar una estación de trabajo de microbiología al Sistema de Laboratorio Virtuales y a Distancia?
4. ¿Cómo validar que la estación de trabajo de microbiología se integra al Sistema de Laboratorios Virtuales y a Distancia?

Para dar cumplimiento al objetivo general y las preguntas científicas ya planteados se trazaron las siguientes **tareas de la investigación:**

1. Realizar el diseño teórico para planificar, organizar y ejecutar la investigación.
2. Describir el estado del arte de los Sistemas de Laboratorios Virtuales y a Distancia para conocer las funcionalidades con que cuentan estos en el mundo.
3. Seleccionar las tecnologías y herramientas a utilizar en el proceso de desarrollo del software.
4. Realizar el análisis y diseño de la solución según defina la metodología seleccionada.
5. Realizar el modelo de implementación para definir los componentes que lo integran.
6. Implementar los componentes del modelo para obtener las funcionalidades definidas.
7. Diseñar los casos de prueba y ejecutar las pruebas de funcionalidad para validar la calidad del producto.

Para dar cumplimiento a las tareas de la investigación fueron utilizados los siguientes **métodos científicos**:

Métodos teóricos

Los métodos teóricos facilitan el estudio de las características propias de la investigación que no pueden ser identificadas a través de otros métodos científicos como la observación en el método empírico (Hernández and González 2011). Permiten además definir modelos e hipótesis de la investigación, lo cual incrementa el nivel de información que es obtenida, al apoyarse principalmente en los procedimientos teóricos Análisis y Síntesis, aportando al desarrollo de teorías científicas. A continuación se describen los métodos teóricos utilizados.

Analítico y Sintético: Permite la división mental del fenómeno en sus componentes y establecer la unión entre las partes analizadas (Hernández and González 2011). Se empleó para estudiar y llegar a una conclusión o resumen de la bibliografía consultada.

Inductivo y Deductivo: Comprende los procedimientos que permiten llegar a proposiciones generales a partir del estudio realizado e inferir casos particulares por un razonamiento lógico (Hernández and González 2011). Durante la investigación se utilizó este método para arribar a conclusiones luego de realizar los estudios relacionados con la problemática existente.

Métodos empíricos

Los métodos empíricos permiten describir las características del objeto en cuestión a través de la experiencia obtenida y llevando a cabo un análisis racional (Hernández and González 2011). A continuación se describe el método empírico utilizado en la investigación.

Observación: Método que se enfoca en la percepción planificada y objetiva del fenómeno en cuestión, siendo realizada de forma consciente (Zayas 1995). Fue utilizado, en la etapa inicial, para determinar el problema de la investigación y ayudar en el diseño. Mientras que en el transcurso de la misma fue utilizada para ir comprobando las ideas planteadas.

La investigación está estructurada en tres capítulos:

Capítulo 1: Fundamento teórico. En este capítulo se fundamentan términos técnicos de importancia para la investigación. Se realiza un estudio del arte sobre las aplicaciones existentes con características similares para ver si pueden solucionar el problema planteado. Además se analiza el objeto de estudio y se seleccionan tecnologías, lenguajes de programación, herramientas y la metodología para el desarrollo de la solución.

Capítulo 2: Características y diseño del sistema. En este capítulo se describen los procesos actuales mediante el modelado del dominio, se identifican los requisitos funcionales y no funcionales, y se agrupan en casos de uso del sistema, además se generan los diagramas correspondientes que los modelan, según la metodología escogida.

Capítulo 3: Implementación y pruebas. En este capítulo se plantea la construcción de la solución propuesta en el capítulo anterior, mostrándose los artefactos generados por la metodología utilizada, se genera el diagrama de despliegue, se realizan las descripciones de las pruebas y se exponen los resultados de las mismas.

Capítulo 1: Fundamentación Teórica

Introducción

En el presente capítulo se exponen conceptos y términos relevantes para el desarrollo de la investigación. Se efectúa un estudio del estado del arte relacionado con soluciones informáticas existentes para los SLVD. Se desarrolla un análisis de la problemática presentada anteriormente, para poder comprender y argumentar la necesidad actual, además de una valoración de las tecnologías actuales utilizadas en la propuesta de solución. Se aborda distintos aspectos teóricos para la investigación, donde se encuentra el análisis de los estándares, herramientas y metodología de desarrollo para la construcción del sistema, así como sus características fundamentales.

1.1. Educación a Distancia

A continuación se muestra la definición de educación a distancia:

- La educación a distancia es la gestión o proceso de educar o ser educado cuando este proceso autodirigido se realiza a distancia. Tal acto educativo es apoyado por el material didáctico elaborado por un equipo experimentado y multidisciplinario (Gómez 2012). Este tipo de educación se centra en hacerle llegar la información a los estudiantes, eliminando la necesidad de que se encuentren presentes en el momento de impartir la clase.
- Es el proceso de enseñanza aprendizaje que se da cuando el profesor y participante (discente/alumno) no se encuentran frente a frente como en la educación presencial, sino que emplea otros medios para la interactividad síncrona o asíncrona; entre ellos, Internet, CD, videos, video conferencia, sesiones de chat y otros (Martínez 2008).

Se denomina educación a distancia al proceso de enseñanza y aprendizaje en el que el educador no imparte de forma directa y presencial los conocimientos al estudiante, apoyándose para ello en otros medios como videos conferencias, sesiones de chat, documentos, multimedia, videos, presentaciones u otra de las posibilidades que brinda Internet.

1.2. Conceptos asociados a la investigación

1.2.1. Sistema de Laboratorios Virtuales

A continuación se muestra la definición de Sistema de Laboratorios Virtuales:

- Alternativa informática a los laboratorios presenciales y a distancia en que se simula en un ordenador el comportamiento de los sistemas a estudiar haciendo uso de modelos matemáticos, permitiendo la realización de experimentos y obteniendo a través de gráficas la evolución temporal de los resultados (López, Gangoiti, Zulueta and Calvo 2009).
- Los Sistemas de Laboratorios Virtuales (SLV) son una representación informática de los laboratorios tradicionales y permiten realizar experimentos, investigaciones, prácticas académicas y científicas, dando la sensación de su existencia real; apoyan y promueven el aprendizaje de los estudiantes y aumentan las opciones de experimentos disponibles en las instituciones educativas (Pascuas et al. 2012).
- Sistema computacional accesible vía Web, mediante el que se simula un laboratorio en donde los experimentos se llevan a cabo siguiendo un procedimiento similar al que se sigue en un laboratorio convencional, pudiendo inclusive ofrecer la visualización de fenómenos mediante objetos dinámicos, programados mediante applets de Java, Flash, CGIS, JavaScript, PHP, etc., incluyendo imágenes y animaciones (Medina et al. 2011).

Un SLV enfocado al trabajo de la microbiología es una simulación de los instrumentos de un laboratorio de microbiología a través de modelos matemáticos. Mediante los SLV se pueden realizar experimentos, investigaciones y prácticas académicas o científicas, siendo posible acceder vía Web a sus funcionalidades y visualizar los resultados, observando mediante gráficas su evolución temporal.

1.2.2. Sistema de Laboratorios a Distancia

A continuación se muestra la definición de Sistema de Laboratorios a Distancia:

- Los Sistemas de Laboratorios a Distancia (SLD) se definen también como un sistema computacional que permite el control y acceso a instrumentación y equipos de un laboratorio real, en el que se realizan prácticas de forma local o remota a través de Internet o Intranet, permitiendo la transferencia de información de forma bidireccional (Medina, Hermida, Hernández and Ladrón 2011). Estos son enfocados a uno o varios temas en específico, lo que permite su uso principalmente en la pedagogía, permitiendo a los educandos realizar prácticas con instrumentación real de forma remota.
- Debido a sus características los laboratorios a distancia generan un apoyo en los procesos de formación en los distintos niveles educativos, al permitir la visualización a través de la Web de las

prácticas que se encuentra realizando el estudiante (Ariza and Amaya 2011). El educando al poder realizar las prácticas aumenta sus conocimientos sobre el tema en cuestión. Los laboratorios a distancia están compuestos por una o varias estaciones de trabajo especializadas cada una, en un tema específico.

Un Sistema de Laboratorios a Distancia enfocado en la realización de prácticas de microbiología permite acceder y visualizar remotamente una muestra colocada en un microscopio colocado en una estación de trabajo. Esto permite fomentar el proceso educativo a través de clase prácticas e investigaciones, que pueden realizar los estudiantes de forma didáctica, sin necesidad de acceder directamente al equipamiento.

1.2.3. Sistemas de Laboratorios Virtuales y a Distancia

Los Sistemas de Laboratorios Virtuales y a Distancia (SLVD) permiten integrar en un todo las características de los SLV y los SLD. Al incluir las características de ambos permiten la simulación a través de modelos matemáticos del comportamiento de diversos sistemas y el control, visualización y acceso a instrumentos o equipos de un laboratorio real. Permiten además potenciar la realización de experimentos y prácticas de forma local o remota, obteniendo transferencias de información de forma unidireccional y bidireccional.

1.2.4. Otros términos asociados a la investigación

Estaciones de trabajo: Una estación de trabajo consiste en una computadora con acceso a la red o a un ordenador central, que ofrece mayor rendimiento para el trabajo con respecto al rendimiento del procesador o la conexión de red (Dictionary.com 2015). Cada estación de trabajo puede contar con uno o varios periféricos conectados, los que se gestionan a través de algún software especializado, permitiendo además el control y visualización de dichos dispositivos de forma remota.

Por otro lado en el caso específico de las estaciones de trabajo de los SLVD son computadoras que permiten el control y visualización de un determinado dispositivo a través de la plataforma. Para ello es necesario contar con un software que, luego de ser desplegado en esta computadora, se encargue de habilitar las funcionalidades que se desean del dispositivo para el acceso remoto. En el caso de las estaciones de trabajo para las prácticas de microbiología permiten el acceso a microscopios, permitiendo la visualización de forma remota de la muestra de microorganismos colocada en estos.

Microscopio: Un microscopio es un instrumento óptico destinado a observar objetos extremadamente

diminutos, haciendo perceptible lo que no lo es a simple vista (RAE 2014). Entre las principales aplicaciones de estos se encuentra su uso en la microbiología, para la observación de microorganismos, células y tejidos.

Cámara digital: Dispositivo electrónico que permite capturar de forma digital una imagen o una sucesión de estas (Alegsa 2015), creando así un flujo de video que puede ser almacenado o mostrado haciendo uso de diversos software.

Códec: Un códec comprime o descomprime archivos multimedia como canciones o vídeos. Un códec puede constar de dos partes: un codificador que comprime el archivo de medios (codificación), y un decodificador que descomprime el archivo (decodificación). Algunos códec incluyen ambas partes, y otros códec incluyen solamente uno de ellos (Enterprise 2015).

Video: Según (Digitales 2008-2009) *"...suele llamarse video a la captura, grabación, almacenamiento, y reconstrucción de una serie de imágenes y sonidos, las cuales representan escenas en movimiento."*

Transmisión de video: Flujo de video que es enviado de forma continua por la red a través de un protocolo de transmisión de datos.

RTSP: RTSP es un protocolo de nivel de aplicación para el control de la entrega de datos con propiedades en tiempo real. RTSP proporciona un marco extensible para permitir a controlarse, entrega bajo demanda de datos en tiempo real, como audio y video. Establece y controla stream de audio o video, aunque no entrega el flujo continuo el mismo, lo que trata es de intercalar el flujo de media con el control del stream si es posible. Básicamente RTSP actúa como un "control remoto de la red" para servidores multimedia (Schulzrinne 1998).

Streaming o difusión del flujo: Técnica para la transferencia de datos de manera que se puede procesar como un flujo constante y continuo. Gracias a la transmisión, el navegador del cliente o plugin puede empezar a mostrar los datos antes de que se haya transmitido la totalidad del expediente.

Portlet: Un portlet es un componente de un portal Web que ofrece acceso a alguna fuente de información específica o una aplicación, tales como actualizaciones de noticias, gestión de multimedia, soporte técnico, o un programa de correo electrónico, entre otras muchas posibilidades (TechTarget 2015a). Estos permiten crear portales de contenido agregado diferentes en una sola interfaz; los portlets conectan al usuario a un contenido específico dentro de esa interfaz.

1.3. Sistemas de Laboratorios Virtuales y a Distancia existentes en la actualidad

1.3.1. Soluciones Internacionales

Laboratorio Remoto de Automática (LRA): El Grupo de Investigación de Automática SUPPRESS de la Universidad de León en España permite la evaluación o la auto-evaluación de los estudiantes. Se pueden clasificar los sistemas accesibles en cuatro grupos: Plantas piloto y maquetas en las que se implementan procesos industriales reales, Tecnologías para la Automatización, Control y Supervisión, Equipos didácticos e Instalaciones industriales reales (SUPPRESS 2014).

Desde el punto de vista de la investigación, la plataforma tecnológica permite trabajar en las siguientes líneas de investigación:

- Supervisión remota de procesos industriales complejos.
- Desarrollo de herramientas avanzadas de supervisión basadas en técnicas de reducción de la dimensionalidad.
- Supervisión energética de edificios e instalaciones industriales.
- Análisis inteligente de datos.
- Supervisión y seguridad en infraestructuras críticas.
- Control avanzado.

UNILabs: La Red Universitaria de Laboratorios Interactivos (UNILabs, por sus siglas en inglés) posibilita la realización de prácticas de Automática, Matemáticas, Control de Procesos Avanzados, Técnicas Experimentales de Física. Para poder acceder a cada uno de sus módulos primeramente se tiene que haber realizados una reserva para el mismo. Las simulaciones de los experimentos están desarrolladas en Easy Java Simulation (EJS), una herramienta de código abierto desarrollada en Java, diseñada para la generación de simulaciones interactivas dinámicas (UNILabs 2014).

Javaoptics (JOptics): El proyecto Javaoptics (JOptics) es una iniciativa del Grupo de Innovación Docente en Óptica Física y Fotónica de la Universidad de Barcelona, cuyo objetivo es fomentar el aprendizaje de la Óptica a través de Internet. JOptics consiste en una serie de recursos docentes, dirigidos principalmente al estudiante universitario, que permiten afianzar los conocimientos de Óptica Física. Además, una parte de

los materiales desarrollados pueden ser utilizados por estudiantes y profesores de ESO[1] y bachillerato, para ilustrar y ampliar conceptos de Física, a ese nivel educativo.

JOptics permite realizar experimentación virtual sobre 12 fenómenos ópticos, incluyendo fibras ópticas, modelo del ojo, difracción, colorimetría, etc. Las simulaciones pueden ejecutarse directamente como *applets* en el navegador, o bien descargarse en el ordenador para usarse en cualquier momento, sin necesidad de tener que disponer de conexión a internet (Universidad 2014).

En la actualidad existen SLVD cuyas funcionalidades fomentan el desarrollo de la educación en la rama de la microbiología entre los se encuentran:

Virtual Labs: Portal de laboratorios virtuales en 97 campos de 9 disciplinas distintas de ciencia e ingeniería, incluyendo electrónica, comunicaciones, informática y biotecnología, entre otras. Se encuentra orientado a estudiantes universitarios y cualquier persona interesada por la investigación científica, permitirá llevar a cabo experimentos en línea, para lo que sólo es necesario un ordenador con conexión a internet (Lab 2014).

Labster: Laboratorio a distancia que permite realizar experimentos a través de la Web en equipos de un laboratorio real, entre los que se encuentran la fermentación, genética médica, clonación molecular, cromatografía y microbiología (Labster 2014).

1.3.2. Soluciones Nacionales

En la actualidad en el país se encuentra desplegado un sistema de laboratorios a distancia en la Universidad Central “Marta Abreu” de las Villas (UCLV), enfocado en el estudio de la automática. El principal objetivo de este laboratorio es permitir a los usuarios aprender a ajustar controladores predefinidos y a diseñar sus propios controladores para posteriormente probarlos sobre un conjunto de dispositivos físicos a través de Internet.

El SLD utiliza herramientas de diseño de sistemas de control asistido por computadora como Matlab-Simulink para la creación de nuevos controladores, lo que facilita a los usuarios el uso del sistema. El SLD permite una rápida y fácil integración de nuevos procesos para experimentos de control y en la actualidad

[1] ESO: La **Educación Secundaria Obligatoria** (ESO) es el sistema educativo español de enseñanza secundaria que tiene como objetivo preparar al alumnado de entre 12 y 16 años para sus próximos estudios y/o el mundo laboral.

están disponibles: un motor de corriente directa y un robot manipulador (Castellanos et al. 2004).

1.3.3. Comparación entre las soluciones

A continuación se muestra una comparación entre las distintas soluciones, tanto nacionales como internacionales, teniendo en cuenta los siguientes aspectos: disponibilidad, facilidad de uso, accesibilidad y si presenta un módulo para el estudio de la microbiología.

Tabla 1: Comparación entre los SLVD existentes en la actualidad

Soluciones	Privadas	Objetos de estudio	Accesibilidad	Módulo de microbiología
LRA	No	La Automática.	A través de internet, haciendo uso de un navegador Web.	No
UNILabs	No	La Automática, Matemáticas, Control de Procesos Avanzados y Técnicas Experimentales de Física.	A través de internet, haciendo uso de un navegador Web.	No
JOptics	No	La Física Óptica.	A través de internet, haciendo uso de un navegador Web.	No
Labster	Si	La Biología y las biotecnologías.	A través de internet, haciendo uso de un navegador Web.	Si
Virtual Labs	No	Disciplinas tales como: electrónica, comunicación, informática, biotecnología, mecánica, química, psicología, ingeniería civil.	A través de internet, haciendo uso de un navegador Web.	Si
SLD UCLV	No	Los sistemas de control, tales como el modelado de dispositivos físicos y el diseño de controladores.	A través de internet, haciendo uso de un navegador Web.	No

Luego de ser realizada las comparaciones entre las soluciones existentes en la actualidad, tanto internacionales como nacionales, se arribó a las siguientes conclusiones:

- Los SLVD en la actualidad se encuentran enfocados principalmente al estudio de las ingenierías (matemáticas, automática, física, etc.), siendo pocos los que incorporan módulos para el estudio de las ciencias médicas.
- Se encuentran accesibles a través de internet, permitiendo el acceso a gran variedad de estaciones de trabajo, y en su mayoría de forma gratuita, salvo algunas excepciones.
- Solo permiten el acceso a sus estaciones de trabajo a través de la plataforma que se encuentre desplegada, siendo imposible la reutilización de estas para otras soluciones informáticas.

1.4. Herramientas, tecnologías y metodología para el desarrollo de la solución

Enfocado en el desarrollo de la solución se representan las principales características de varias herramientas y tecnologías que posibilitan su creación. Fueron escogidas teniendo en cuenta el aporte que brindan a la implementación de la solución, además de ser definidas en conjunto con los demás equipos de desarrollo de los distintos módulos del SLVD, lo que potencia el mejor entendimiento de la documentación generada.

1.4.1. Metodologías de Desarrollo de Software

OpenUp

Proceso unificado que aplica enfoques iterativos e incrementales dentro de un ciclo de vida estructurado del desarrollo de un software. OpenUp divide el ciclo de vida de un software en iteraciones o intervalos planificados, lo que define un plan que debe ser entregado dentro de cada iteración, lo cual debe dar como resultado un producto estable y coherente, con los que se construye el sistema de forma gradual (Yang 2014). A continuación se mencionan los principios fundamentales de OpenUp.

Principios de OpenUp:

1. Colaborar para sincronizar intereses y compartir conocimiento. Este principio promueve prácticas que impulsan un ambiente de equipo saludable, facilitan la colaboración y desarrollan un conocimiento compartido del proyecto.

2. Equilibrar las prioridades para maximizar el beneficio obtenido por los interesados en el proyecto. Este principio promueve prácticas que permiten a los participantes de los proyectos desarrollar una solución que maximice los beneficios obtenidos por los participantes y que cumple con los requisitos y restricciones del proyecto.
3. Centrarse en la arquitectura de forma temprana para minimizar el riesgo y organizar el desarrollo.
4. Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo. Este principio promueve prácticas que permiten a los equipos de desarrollo obtener retroalimentación temprana y continua de los participantes del proyecto, permitiendo demostrarles incrementos progresivos en la funcionalidad.

Esta metodología está concebida para proyectos con poco tiempo de desarrollo, equipos pequeños y con pocos roles. Propone que el diseño debe ser sencillo, que funcione con todas las pruebas y con el menor número de clases y métodos posible. Todas estas características contribuyeron a que la solución desarrollada mantuviera una programación organizada y con una taza pequeña de errores.

Para el desarrollo de la aplicación fue escogida la metodología OpenUp por basarse en la creación de la documentación del proyecto en las metodologías tradicionales e incorporar características de las ágiles. Posibilita generar solo aquellos artefactos que son necesarios para el desarrollo, adecuándose a equipos de trabajo pequeños.

1.4.2. Lenguaje de Modelado de Sistemas

UML

El Lenguaje Unificado de Modelado, en sus siglas en inglés (UML), es un lenguaje de propósito general para el modelado orientado a objetos y visual, que permite una abstracción del sistema y sus componentes (OMG 2015). En la presente investigación es utilizado en su versión 2.1 para visualizar, especificar y documentar los artefactos que se generan a lo largo del ciclo de desarrollo de la solución, fundamentalmente en las fases de análisis, diseño e implementación. Esto permite entender, diseñar, configurar, mantener y controlar la información sobre el sistema a construir.

1.4.3. Herramienta CASE

Visual Paradigm

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar

todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar, actualizar y compatible entre ediciones (Visual 2015).

En la solución desarrollada fue utilizada el Visual Paradigm en su versión 8.0, facilitando el modelado del diagrama conceptual y diagrama de despliegue, así como la representación de patrones de diseño, mediante el soporte de UML que proporciona esta herramienta, lo que aportó un entendimiento de conceptos claves para el negocio en cuestión.

1.4.4. Lenguaje de Programación y Bibliotecas

Un lenguaje de programación está diseñado para escribir un conjunto de acciones consecutivas que un equipo debe ejecutar, lo que lo convierte en un modo práctico para dar instrucciones a un equipo. Estos facilitan las tareas de programación ya que poseen formas adecuadas para su entendimiento y resultan independientes de la computadora a utilizar.

C++

C++ es un lenguaje de programación de propósito general que se inclina por el desarrollo de sistemas. Este soporta la abstracción de datos, la programación orientada a objetos y genérica (Stroustrup 1985). Es una versión ampliada del lenguaje C, por demás incluye una gran variedad de librerías. Se utilizará este lenguaje por las siguientes características:

- Se puede realizar una aplicación que se puede comercializar sin la necesidad de pagar una licencia, lo cual posibilita la realización de la aplicación en su totalidad sin tener que pagar por la utilización de este lenguaje.
- Presenta la facilidad de que es portable, propiciando que el código del simulador creado en este lenguaje se puede compilar en cualquier sistema operativo.
- Proporciona facilidades para utilizar código o bibliotecas existentes además de ser uno de los lenguajes más rápidos en cuanto a ejecución, ventaja que puede ser aprovechada para el procesamiento de los flujos de video.

Java

Java es un lenguaje orientado a objetos basado cuya principal característica es ser un lenguaje compilado e interpretado. Todo programa se compila y el codigo generado es interpretado por la máquina virtual. Al ser multiplataforma permite trasladar la aplicación de una computadora a otra sin importar el sistema operativo, ademas de ser un lenguaje de código abierto (Downey 2012).

La tecnología Java cuenta con la plataforma Java 2 Enterprise Edition (J2EE),"por sus siglas en inglés". J2EE es la extensión de Java para el desarrollo Web, dispone de varias herramientas de código abierto como servidores, marcos de trabajo y API[2]. Esta engloba dentro de sí un conjunto de especificaciones y APIs, tales como: JDBC[3], RMI[4], JMS[5], XML[6], Servicios Web, etc., además configura algunas especificaciones únicas como Enterprise JavaBeans, *servlets*, Java Server Pages (JSP) y portlets (Aumaille 2002).

1.4.5. Entorno de Desarrollo Integrado

Los entornos de desarrollo integrado (IDEs), "por sus siglas en inglés", proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación ya que son un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existente.

QT Creator

En esta investigación se utiliza el IDE Qt Creator 3.0 como plataforma de desarrollo de la aplicación de escritorio pues proporciona amplios beneficios para los desarrolladores debido a que existe abundante documentación sobre cómo trabajar en su entorno que ayuda a los nuevos usuarios de Qt a aprender y comenzara desarrollar rápidamente. Ademas QT Creator ofrece las siguientes características:

- Las bibliotecas Qt se encuentran escritas en C++ lo que facilita el desarrollo. Estas cuentan con QTDesigner para diseñar formularios visualmente, QtAssistant que permite el acceso rápido a la documentación, QtLinguist la traducción rápida de programas y Qmake simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

[2] API: *Applications Programming Interface, "Interfaces de programación de aplicaciones" traducido al español.*
[3] JDBC: *Java Database Connectivity*
[4] RMI: *Remote Method Invocation*
[5] JMS: *Java Message Service*
[6] XML: *eXtensible Markup Language*

- Posee un avanzado editor de código C++, lo cual facilita la programación de la aplicación ofreciendo completado de código.
- Posee una GUI integrada y diseñador de formularios.

Eclipse

Eclipse es un IDE de código abierto y multiplataforma. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI del inglés Graphical User Interface). Este IDE provee las herramientas y funciones necesarias para el desarrollo de software, recogidas además en una interfaz que lo hace fácil y agradable de usar.

Eclipse emplea complementos (en inglés plugins) para extender sus funcionalidades y brindar soporte a otros lenguajes de programación como C, C++, PHP y Python. El IDE Eclipse Luna en su version 4.4.2 será utilizado para el trabajo con el lenguaje de programación Java, siendo fundamental este en la creación del portlet que será integrado al SLVD[7].

1.4.6. Marcos de trabajo

LibVLC

El VLC es un reproductor y también un marco de trabajo (*framework, por sus siglas en inglés*) multimedia del proyecto VideoLAN. Es además un software libre distribuido bajo la licencia GPL que soporta gran variedad de formatos de audio y video, además de varios protocolos de streaming[8]. VLC provee del API LibVLC, que permite el acceso a una interfaz externa para la programación, siendo posible embeber todas las funciones del reproductor VLC Media Player en una aplicación (Organization 2015).

La librería LibVLC en su versión 2.2.0 es utilizada debido a que posee la capacidad de potenciar la transmisión de video a través de redes y convertir archivos multimedia en formatos distintos al original.

[7] SLVD: Sistema de Laboratorios Virtuales y a Distancia.

[8] Streaming: El streaming (también denominado transmisión, difusión en continuo o descarga continua) es la distribución digital de multimedia a través de una red de computadoras, de manera que el usuario consume el producto (generalmente archivo de video o audio) en paralelo mientras se descarga. La palabra streaming se refiere a una corriente continuada, que fluye sin interrupción.

Incluye además la posibilidad de recibir un flujo de video en un formato dado, realizar la transcodificación[9] y emitirlo en el nuevo formato.

Spring

Spring proporciona un modelo de programación y configuración completa para las aplicaciones empresariales modernas basadas en Java, en cualquier tipo de plataforma de despliegue. Un elemento clave de Spring es el apoyo infraestructural a nivel de aplicación: Spring se centra en la "fontanería" de las aplicaciones empresariales para que los equipos pueden centrarse en la lógica de negocios a nivel de aplicación, sin ataduras innecesarias a los entornos de implementación específicos (Software 2015).

Por otro lado se utiliza Spring en su versión 3.0, debido a que elimina las responsabilidades de generar los diversos objetos necesarios para la aplicación de las manos del desarrollador, al encargarse de la creacion de los mismo, basándose en ficheros xml y anotaciones. Spring se basa en el patrón Modelo Vista Controlador (MVC), estructurado en tres capas, el modelo de datos, las vistas de la interfaz de usuario y los controladores, esto permite dividir en tres capas las clases de Spring Portlet MVC Framework.

1.4.7. Liferay Portal

Sistema de gestión de contenidos Web que permite la fácil creación de sitios Web, proporcionando mayor responsabilidad y flexibilidad para el desarrollo y mantenimiento de los contenidos. Liferay permite diseñar flujos de trabajo asociados a contenidos y documentos para dar agilidad a sus procesos de negocio. Al ser basado en Java se ejecuta sobre multiples plataformas y utiliza Hibernate como herramienta ORM[10] para la capa de persistencia, lo que facilita el soporte de gran variedad de bases de datos tales como MySQL, Oracle o PostgreSQL.

Liferay es un contenedor de portlets, los cuales visualiza y gestiona, siendo posible implementar portlets que brinden un determinado servicio. Puede ser desplegado sobre multiples servidores como: JBoss, GlassFish o Apache Tomcat. Será utilizado en su versión 6.2.0 como contenedor del portlet y brindará las funcionalidades necesarias para dar respuesta a las necesidades de la investigación.

[9] Transcodificación: Se denomina transcodificar a la conversión directa de un códec a otro.

[10] ORM: El mapeo objeto-relacional (más conocido por su nombre en inglés, Object-Relational mapping, o sus siglas ORM) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y la utilización de una base de datos relacional como motor de persistencia.

1.4.8. Hibernate

Es una herramienta de software libre de mapeo objeto-relacional para ambientes de desarrollo en Java. Hibernate facilita el mapeo de atributos entre una Base de Datos (BD) relacional tradicional y el modelo orientado a objetos de una aplicación. Posee un lenguaje de consultas propio llamado Hibernate Query Language (HQL) que puede ser usado por los programadores sin tener conocimientos de SQL, además de ser uno de los módulos que compone la arquitectura de Spring (Peak and Heudecker 2005). Brinda diferentes beneficios como son la generación del código Java a partir del modelo de base datos y viceversa, así como el mapeo de clases.

Al ser una herramienta ORM permite solucionar el problema que se crea entre los modelos de datos coexistentes en una aplicación, los modelos orientados a objetos y los modelos relacionales. Hibernate en su versión 4.1.3 es utilizado en la aplicación Web para fomentar la persistencia de datos, al permitir gestionar los objetos de las bases de datos, facilitando la inserción, consulta y eliminación de los datos referentes a las estaciones.

1.4.9. Sistemas Gestores de Bases de Datos (SGBD)

PostgreSQL

Como Sistema Gestor de Base de Datos (SGBD) se utilizó la versión 9.4 de PostgreSQL en la solución desarrollada. Por otro lado permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Cuenta con una arquitectura que ha ganado mucha reputación dada su confidencialidad e integridad en los datos (Momjian 2000). Es una herramienta orientada a objetos, de bajo coste y multiplataforma, siendo un potente gestor de base de datos relacional libre (liberado bajo licencia BSD).

PostgreSQL potencia el manejo y el aprendizaje relacionado con la utilización de las bases de datos, al permitir almacenar gran variedad de información y contar con una extensa documentación. En la presente investigación fue utilizado para la persistencia de los datos relacionados con las direcciones URLs, y su posterior listado.

Conclusiones Parciales

Una vez realizado el estudio de la literatura cientifica referente a los Sistemas de Laboratorios Virtuales y a Distancia se pudo identificar las posibilidades que brindan este tipo de herramientas, a su vez se evidencío la imposibilidad de reutilizar las estaciones de trabajo con las que cuentan. Sobre el estudio de

las tendencias de herramientas y tecnologías actuales para el desarrollo de software se concluyó utilizar aquellas que mejor se adaptan para dar cumplimiento a las necesidades planteadas.

Capítulo 2: Características y Diseño de la Solución

Introducción

En el presente capítulo se realiza un análisis de la solución propuesta a través del modelo de dominio, donde se representan los principales componentes del problema, con el fin de comprender las características fundamentales del mismo, para luego ser implementadas en la solución. También se representan los diagramas de clases del diseño, además de otros artefactos resultantes del flujo de trabajo de la metodología OpenUp.

2.1. Descripción de la solución propuesta

La solución que se propone desarrollar se encuentra dividida en dos aplicaciones, una aplicación de escritorio que se desplegará en la estación de trabajo y una aplicación Web en forma de portlet[1] que se integrará al SLVD. La aplicación de escritorio se encontrará en la estación de trabajo y contará entre sus funcionalidades con la transmisión de un flujo de video, mientras que la aplicación Web se encargará de visualizar el flujo de video transmitido. A continuación se describe detalladamente cada una de ellas:

1. Aplicación de escritorio (estación de trabajo): La aplicación de escritorio se encuentra desplegada en la estación de trabajo. Es la encargada de listar los dispositivos de tipo cámara que se encuentren conectados y permitir la visualización de uno de ellos. El dispositivo seleccionado debe encontrarse acoplado a un microscopio, lo que permite la observación de la muestra que se encuentre colocada en este. Otra de sus funcionalidades es la transmisión por la red del flujo de video que se encuentra emitiendo el dispositivo seleccionado. Además permite la grabación y captura de fotografías de las muestras colocadas en el microscopio seleccionado.
2. Aplicación Web (SLVD): La aplicación Web consiste en un portlet que permite integrar una estación de trabajo al SLVD. Se encarga de visualizar los flujos de video que se encuentran siendo transmitidos por la aplicación de escritorio desde las estaciones de trabajo. Además de permitir capturar y guardar fotografías del flujo de video que se encuentra visualizando.

En su conjunto la solución propuesta en su totalidad permitirá a los educandos de universidades y otros centros educativos del país potenciar la adquisición de conocimientos en la rama de la microbiología, al poder observar el progreso de muestras reales en tiempo real, además de realizar fotos de las mismas,

[1] Portlet: Componentes modulares de las interfaces de usuario gestionadas y visualizadas en un portal Web.

todo esto haciendo único uso de un navegador Web.

2.2. Modelo de dominio

El modelo de dominio engloba los principales conceptos que conforman el negocio, mas no incluye las responsabilidades de los actores. El modelo de dominio recoge los tipos o clases más importantes, por lo que hay que tener presente que se realiza un modelo del entorno del software, no del mismo como tal (Falgueras and Catalunya 2002).

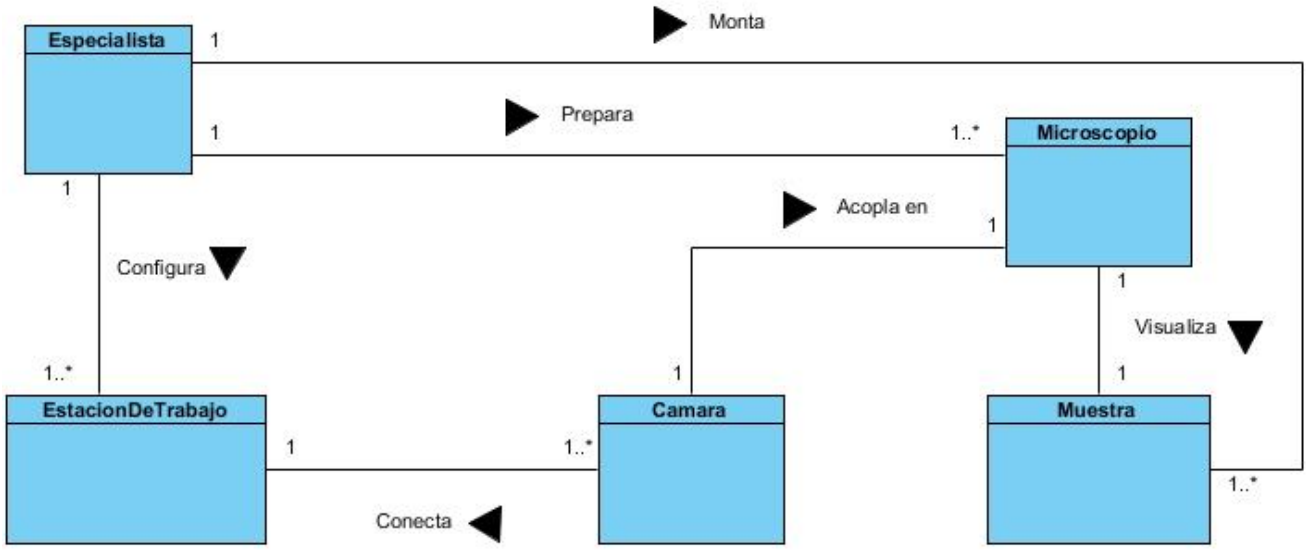


Figura 1: Modelo de Dominio

2.2.1. Definición de clases del modelo del dominio

Para un mejor entendimiento del modelo de dominio se realiza una descripción de las clases existentes:

- **Especialista**: Persona encargada de interactuar con las estaciones de trabajo, los microscopios y las muestras.
- **Estación de trabajo**: Computadora con acceso a la red a la que se encuentra conectada la cámara que esta acoplada al microscopio.
- **Cámara**: Dispositivo destinado a registrar imágenes en movimiento.
- **Microscopio**: Dispositivo que permite observar objetos u organismos imperceptibles a la vista.
- **Muestra**: Pequeña porción de microorganismos o de tejido celular que se visualizará con el

microscopio.

El proceso de observación de la muestra se inicia cuando el especialista coloca la misma en el microscopio. La cámara se encuentra acoplada al microscopio y se encarga de visualizar la muestra colocada en este. El especialista a su vez gestiona la cámara a través de la estación de trabajo, permitiéndole visualizar la muestra que anteriormente haya colocado en el microscopio.

2.3. Especificación de los Requisitos del Sistema

2.3.1. Requisitos Funcionales

Los requisitos funcionales son declaraciones de servicios que el sistema debe proporcionar, define la manera en que éste debe reaccionar a determinadas entradas y cómo se debe comportar en situaciones particulares (Pressman 2010). Estos requisitos son las características fundamentales que debe cumplir el sistema y expresan la capacidad de acción del mismo.

Requisitos Funcionales de la aplicación de escritorio (estación de trabajo)

RF1: Listar cámaras disponibles para su uso.

RF2: Seleccionar una cámara de la lista de disponibles.

RF3: Visualizar la muestra con la cámara seleccionada.

RF4: Capturar fotografía de la muestra con la cámara seleccionada.

RF5: Guardar la fotografía capturada.

RF6: Iniciar grabación de la evolución de la muestra con la cámara seleccionada.

RF7: Detener grabación de la cámara.

RF8: Guardar video grabado de la evolución de la muestra.

RF9: Configurar los parámetros del video para la transmisión.

Descripción: La aplicación de escritorio debe permitir escoger los parámetros con los que se transmitirá el flujo de video (ajustes de resolución y ajustes de codificación).

Entradas:

- Ajuste de protocolo (Formato alfanumérico).

- Ajuste de códec (Formato alfanumérico).
- Ajuste de resolución (Formato alfanumérico).

Salidas: La transmisión del video se realiza de acuerdo a los parámetros escogidos.

RF10: Iniciar la transmisión del flujo de video emitido por la cámara seleccionada.

RF11: Detener la transmisión del flujo de video.

RF12: Mostrar los parámetros con los que se realiza la transmisión.

RF13: Configurar las opciones de guardado de las capturas.

Descripción: La aplicación de escritorio permite seleccionar donde se desea almacenar las fotos y grabaciones realizadas.

Requisitos Funcionales de la aplicación Web (SLVD)

RF14: Listar las estaciones transmisoras disponibles.

Descripción: La aplicación Web muestra las estaciones transmisoras que hayan sido previamente insertadas en la base de datos.

RF15: Seleccionar una estación transmisora de la lista de disponibles.

RF16: Visualizar flujo de video de la estación transmisora seleccionada.

RF17: Capturar fotograma del flujo de video de la estación seleccionada.

Descripción: La aplicación Web permite capturar un fotograma del flujo de video transmitido por la estación de trabajo que haya sido seleccionada.

RF18: Guardar fotograma tomado del flujo de video de la estación seleccionada.

Descripción: La aplicación Web permite guardar el fotograma tomado del flujo de video, dándole al usuario la posibilidad de almacenarlo en el directorio que desee.

RF19: Insertar una estación transmisora de un flujo de video.

Descripción: La aplicación Web permite insertar los parámetros (Nombre de la estación y URL[2] mediante la que se puede acceder al flujo de video que se encuentre siendo transmitido).

[2] URL: *Uniform Resource Locator, en español localizador uniforme de recurso.*

Entrada:

- Nombre de la estación (Formato alfanumérico)
- URL (Formato alfanumérico)

Salida: Si todos los valores son correctos se inserta una nueva estación transmisora, sino se muestra un mensaje de error.

RF20: Actualizar los datos de una estación transmisora.

Descripción: La aplicación Web permite actualizar los parámetros “Nombre de la estación” y “URL” correspondientes a una estación transmisora.

RF21: Eliminar una estación transmisora.

2.3.2. Requisitos no Funcionales

Los Requisitos no Funcionales son las restricciones de servicios o funciones ofrecidas por el sistema, restringiendo el espacio de sus posibles soluciones (Pressman 2010). Se enfocan fundamentalmente en las características del sistema y las condiciones que se deben cumplir para su correcto funcionamiento.

Requisitos de usabilidad:

RNF1: La solución en general poseerá un diseño que permita potenciar la facilidad de aprendizaje, tanto para aprender sus funcionalidades básicas como para realizar correctamente la tarea que desea ejecutar el usuario, y poder volver a utilizarlas en el futuro. Para ello indicará a través de mensajes emergentes la acción que realiza cada botón, además de contar estos con iconos o nombres sugerentes a la acción que realizan.

Requisitos de software:

RNF2: Para el funcionamiento del sistema se debe requerir de los siguientes recursos de software.

- **PC Cliente de la aplicación de escritorio**: Sistema Operativo Linux. Biblioteca LibVLC en su versión 2.2.0.
- **Servidor de aplicaciones:** Sistema Operativo Linux, debe contar con el servidor Web Apache Tomcat 7.0.1 con la aplicación Liferay Portal 6.2.0, JDK6, PostgreSQL 9.4.
- **PC Cliente Web:** Las estaciones clientes para acceder al sistema deben utilizar como navegador

Mozilla Firefox versión 36.0, Google Chrome 40.0 o cualquiera de la versiones superiores.

Requisitos de hardware

RNF3: Para el correcto funcionamiento del sistema se debe requerir con los siguientes recursos de hardware.

- **PC Cliente de la aplicación de escritorio:** 2 Gb de RAM o superior, procesador Dual Core a 2.20 GHz o superior. Tarjeta de red con una velocidad de 100 Mbps o superior.
- **Servidor de aplicaciones:** 4Gb de memoria RAM, procesador Dual Core a 2.20 GHz o superior. Tarjeta de red con una velocidad de 100 Mbps o superior.
- **PC Cliente Web:** 512 Mb de RAM o superior, procesador Pentium IV o superior. Tarjeta de red con una velocidad de 100 Mbps o superior.

Requisitos de interfaz

RNF4: El sistema está diseñado para que se visualicen los mensajes de error en ventanas emergentes con bordes de color rojo.

RNF5: Cuando se ordene una acción en cualquiera de los componentes del sistema debe mostrarse un mensaje que indique la acción que se encuentra ejecutándose: “Grabando”, “Transmitiendo” o ambas. En el caso de encontrarse transmitiendo se muestran otros detalles significativos, tales como el puerto, códec, resolución y la URL mediante la que se puede acceder al flujo de video.

Requisitos de diseño

RNF6: Lenguaje de programación. Se emplea en el desarrollo de la aplicación de escritorio el lenguaje C++, haciendo uso del framework QT Creator en su versión 3.0, mientras que para el lado del cliente Web se utilizará el lenguaje Java Server Page (JSP), haciendo uso del framework Spring.

RNF7: Bibliotecas. Para el correcto funcionamiento de la aplicación de escritorio se utiliza la biblioteca escrita en lenguaje C, LibVLC en su versión 2.2.0.

2.4. Modelo de Casos de Uso del Sistema

2.4.1. Definición de actores del sistema

Un actor del sistema es un rol determinado que cumple un usuario, el cual determina su nivel de interacción con determinado caso de uso.

Tabla 2: Descripción de actores del sistema

Actor del Sistema	Descripción
Especialista	El especialista interactúa con la aplicación de escritorio, permitiéndole seleccionar de la lista de cámaras disponibles la que desea visualizar y transmitir, además de permitirle capturar fotos y grabaciones de video, y configurar las propiedades de dichas capturas.
Usuario	El usuario tiene permitido seleccionar de la lista de flujos de video, el que desea visualizar, además de capturar fotogramas y almacenar el mismo en un directorio de su preferencia, todo a través de la aplicación Web.
Administrador	El administrador tiene permitido seleccionar de la lista de flujos de video, el que desea visualizar, además de capturar fotogramas y almacenar el mismo en un directorio de su preferencia. Además puede insertar, modificar o eliminar el origen del flujo de video, todo a través de la aplicación Web.

2.4.2. Diagrama de Casos de Uso del Sistema

Un Diagrama de Casos de Uso muestra la relación entre los actores y los casos de uso del sistema, de esta forma se representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa (Jacobson et al. 2004). A continuación se mencionan los requisitos funcionales agrupados en Casos de Uso del Sistema, encontrándose separados teniendo en cuenta la aplicación en la que se encuentran implementados.

Definición de Casos de Uso del Sistema de la aplicación de escritorio (estación de trabajo).

CU1. Visualizar la muestra.

- **RF1:** Listar cámaras disponibles para su uso.

- **RF2:** Seleccionar una cámara de la lista de disponibles.
- **RF3:** Visualizar la muestra con la cámara seleccionada.

CU2. Realizar capturas de la evolución de la muestra.

- **RF4:** Capturar fotografía de la muestra con la cámara seleccionada.
- **RF5:** Guardar la fotografía capturada.
- **RF6:** Iniciar grabación de la evolución de la muestra con la cámara seleccionada.
- **RF7:** Detener grabación de la cámara.
- **RF8:** Guardar video grabado de la evolución de la muestra.

CU3. Administrar la transmisión del flujo de video.

- **RF9:** Configurar los parámetros del video para la transmisión.
- **RF10:** Iniciar la transmisión del flujo de video emitido por la cámara seleccionada.
- **RF11:** Detener la transmisión del flujo de video.
- **RF12:** Mostrar los parámetros con los que se realiza la transmisión.

CU4. Configurar opciones de guardado de las capturas.

- **RF13:** Configurar las opciones de guardado de las capturas.

Definición de Casos de Uso del Sistema de la aplicación Web (SLVD).

CU5. Mostrar flujo de video transmitido.

- **RF14:** Listar las estaciones transmisoras disponibles.
- **RF15:** Seleccionar una estación transmisora de la lista de disponibles.
- **RF16:** Visualizar flujo de video de la estación transmisora seleccionada.

CU6. Capturar fotograma del flujo de video.

- **RF17:** Capturar fotograma del flujo de video de la estación seleccionada.
- **RF18:** Guardar fotograma tomado del flujo de video de la estación seleccionada.

CU7. Gestionar estaciones transmisoras del flujo de video.

- **RF19:** Insertar una estación transmisora de un flujo de video.
- **RF20:** Actualizar los datos de una estación transmisora de un flujo de video.
- **RF21:** Eliminar una estación transmisora de un flujo de video.

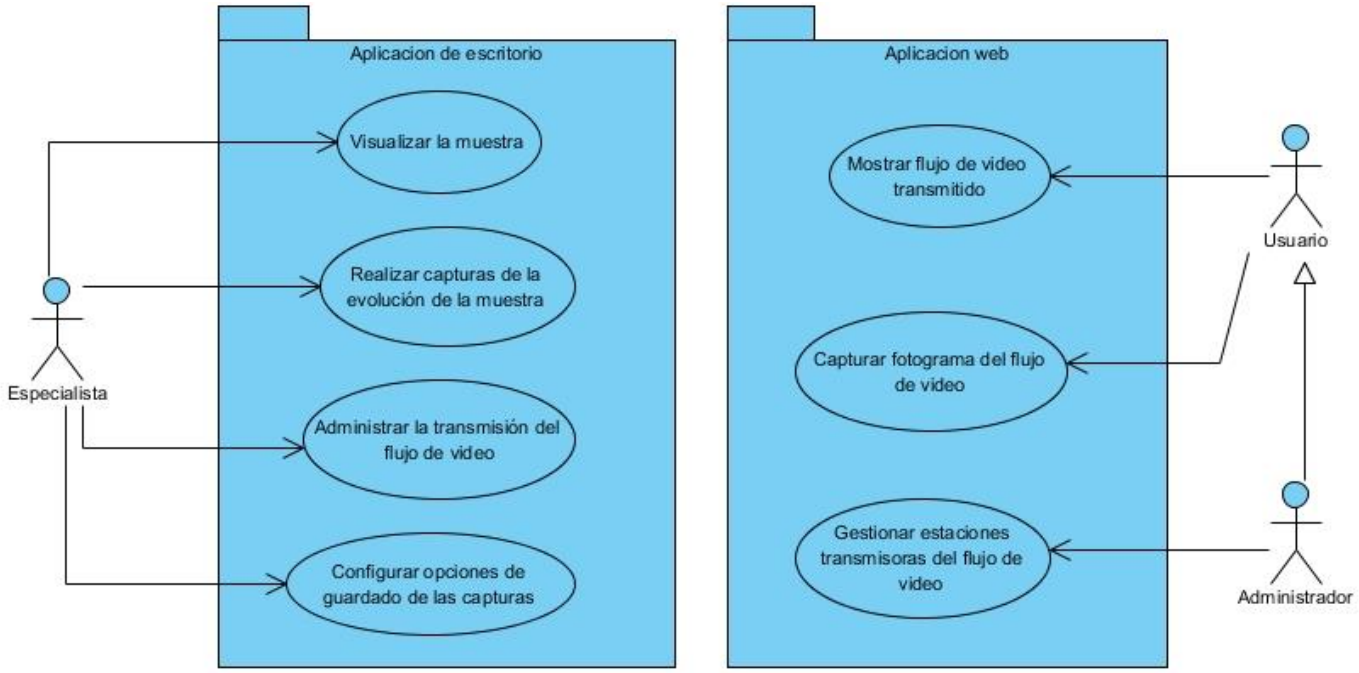


Figura 2. Diagrama de Casos de Uso del Sistema

2.4.3. Descripción de los Casos de Uso críticos del Sistema

La descripción de los Casos de Uso describe paso a paso la interacción entre el actor y el sistema, siendo una relación de acción del actor y respuesta del sistema. A continuación se describen los CU: "Administrar la transmisión del flujo de video" y "Mostrar flujo de video transmitido".

CU “Administrar la transmisión del flujo de video”

Tabla 3: Especificación CU "Administrar la transmisión del flujo de video”

Objetivo	Iniciar la transmisión de un flujo de video por la red.
Actores	Especialista (inicia).
Resumen	El caso de uso inicia cuando el Especialista realiza algunas de las siguientes operaciones:

	Iniciar la transmisión, Detener la transmisión o Configura los parámetros con los que se transmitirá el flujo de video.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	Ha sido seleccionada una cámara y mostrada la muestra.
Postcondiciones	En dependencia a la acción realizada por el Especialista: se transmite un flujo de video por la red, se detiene el flujo de video transmitido, se modifican los parámetros con los que se transmitirá el flujo de video.

Flujo de eventos		
Flujo básico "Transmisión del flujo de video"		
	Actor	**Sistema**
1	Selecciona la opción que desea realizar: a) Iniciar transmisión b) Detener transmisión c) Configurar transmisión	
2		El sistema, en dependencia a la acción solicitada realiza la operación correspondiente: a) Iniciar transmisión, ver Sección 1: "Iniciar transmisión" b) Detener transmisión, ver Sección 2: "Detener transmisión" c) Configurar parámetros, ver Sección 3: "Configurar parámetros"
Sección 1: "Iniciar transmisión"		

Flujo básico "Se inicia la transmisión del flujo de video con los parámetros por omisión"		
1		Configura la transmisión con los parámetros protocolo, códec y resolución por omisión.
2		Comienza a transmitir el flujo de video. Muestra en el área de notificaciones el mensaje "Transmitiendo". Termina el CU.
Flujo alterno		
1a. "El Especialista ha seleccionado los parámetros de la transmisión"		
	Actor	**Sistema**
1a		Configura la transmisión con los parámetros protocolo, códec y resolución seleccionados por el Especialista.
Flujo alterno		
2a. "Cámara no soporta los parámetros establecidos para la transmisión"		
	Actor	**Sistema**
2a		Muestra en pantalla un mensaje indicando que no se ha podido iniciar la transmisión con los parámetros establecidos. Termina el CU.
Sección 2: "Detener transmisión"		
Flujo básico "Se detiene la transmisión del flujo de video"		
	Actor	**Sistema**
1		Detiene la transmisión del flujo de video.

		Termina el CU.
Flujo alterno		
1a. "No existe ninguna transmisión iniciada"		
	Actor	**Sistema**
1a		Muestra el mensaje: "No existe ninguna transmisión iniciada"
Sección 3: "Configurar transmisión"		
Flujo básico "Se seleccionan los parámetros con los que se realizará la transmisión"		
	Actor	**Sistema**
1		Muestra la interfaz para modificar los parámetros con los que se realizará la transmisión del flujo de video.
2	Modifica los parámetros protocolo, códec o resolución que desea.	
3	Acepta las modificaciones realizadas.	
4		Actualiza los parámetros para la transmisión. Termina el CU.
Flujo alterno		
3a. "Cancela las modificaciones realizadas"		
	Actor	**Sistema**
3a		Se cancela la operación y se mantienen los parámetros por omisión para realizar la transmisión.

Relaciones	**CU Incluidos**	Ninguno.
	CU Extendidos	Ninguno.
Requisitos no funcionales	RNF5,RNF7	
Asuntos pendientes	Ninguno.	

CU "Mostrar flujo de video transmitido"

Tabla 4: Especificación del CU " Mostrar flujo de video transmitido"

Objetivo	Visualizar a través de la plataforma Web el flujo de video que se encuentre siendo transmitido.
Actores	Usuario (inicia).
Resumen	El caso de uso inicia cuando el Usuario selecciona un flujo de video de la lista de disponibles y se visualiza este en la aplicación Web.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	Debe existir al menos una estación de trabajo transmitiendo un flujo de video. Lista de flujos de video obtenida correctamente del servidor.
Postcondiciones	Se visualiza el flujo de video seleccionado.
Flujo de eventos	
Flujo básico "Mostrar flujo de video transmitido"	

	Actor	Sistema
1	Selecciona la estación que desea visualizar.	
2		Establece la conexión con la estación transmisora mostrando el flujo de video. Termina el CU.
Flujo alterno		
2a. Evento: “No se establece la conexión con el servidor”		
	Actor	**Sistema**
2a.		Muestra en el visualizador un mensaje indicando que no se ha podido establecer conexión con la estación seleccionada. Termina el CU.

Relaciones	**CU Incluidos**	Ninguno.
	CU Extendidos	Ninguno.
Requisitos no funcionales	RNF1,RNF4	
Asuntos pendientes	Ninguno.	

2.5. Diseño del Sistema

2.5.1. Arquitectura del sistema

La arquitectura del software alude a “la estructura general del software y las formas en que la estructura proporciona una integridad conceptual para un sistema”. En su forma más simple, la arquitectura es la estructura u organización de los componentes del programa, la manera en que estos componentes interactúan, y la estructura de datos que utilizan los componentes. En un sentido más amplio, sin embargo, los componentes pueden generalizarse para representar elementos

importantes del sistema y sus interacciones (Pressman 2005).

El sistema al encontrarse dividido en dos aplicaciones distintas, una de escritorio y una Web se decidió la utilización de una arquitectura basada en componentes, donde cada componente se desarrolla con un patrón arquitectónico diferente. Esto simplifica las pruebas y el mantenimiento, al enfocarse en cada componente por separado, además de permitir su reutilización en otros sistemas.

2.5.2. Patrones arquitectónicos.

Un patrón arquitectónico expresa un paradigma fundamental para estructurar u organizar un sistema. Es la expresión de un esquema estructural de organización para sistemas de software, que proporciona un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye normas y directrices para las relaciones entre ellos (Buschman et al. 1996).

Para el desarrollo de la aplicación de escritorio fue utilizada el patrón arquitectónico N Capas en su versión 2 Capas, lo que facilita el diseño modular de la aplicación y minimiza la dependencia entre capas. Al no contar con la necesidad de acceder a datos, fueron definidas únicamente las siguientes:

- Capa de Presentación: Reúne todos los aspectos relacionados con las interfaces y la interacción con los diferentes tipos de usuarios. Estos aspectos típicamente incluyen el manejo y aspecto de las ventanas, el formato de los reportes, menú y gráficos en general. Representa la capa que interactúa con el usuario, también se le puede llamar interfaz de usuario.
- Capa de Negocio: Esta capa incluye todo lo relacionado con la lógica de negocio por lo que incluye las tareas que forman parte de los procesos, reglas y restricciones que se aplican. Es la capa donde se almacena el código implementado.

En el desarrollo del cliente Web se utilizará el patrón arquitectónico Modelo Vista Controlador (MVC) separa la vista o interfaz que se le mostrará al usuario del modelo de datos y de la clase controladora, facilitando el desarrollo de estas por separado e incrementando la flexibilidad y reutilización. La controladora procesa todos los datos que se generan en las interfaces y los envía al modelo, regresando estos a las interfaces y así sucesivamente en un ciclo que solo termina al cerrar la aplicación.

- Modelo: Representa la información con la que trabaja la aplicación, la lógica del negocio. Maneja los datos y controla todas sus transformaciones.
- Vista: Representa la interfaz de usuario, los datos que a este se le mostrarán de una manera más

amigable.

- Controlador: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y en las vistas.

2.5.3. Diagramas de clases del diseño

El diagrama de clases del diseño es utilizado para modelar la vista de diseño estática de un sistema, estos describen gráficamente las especificaciones de las clases además de visualizar las relaciones entre estas. A continuación se muestran los diagramas de clases del diseño para los CU: “Administrar la transmisión del flujo de video” y “Mostrar flujo de video transmitido”.

CU “Administrar la transmisión del flujo de video”

En el diagrama de clases del diseño referente al CU “Administrar la transmisión del flujo de video” se identifican dos paquetes: Presentación y Negocio, los que corresponden al patrón de diseño N-Capas, en su versión dos capas.

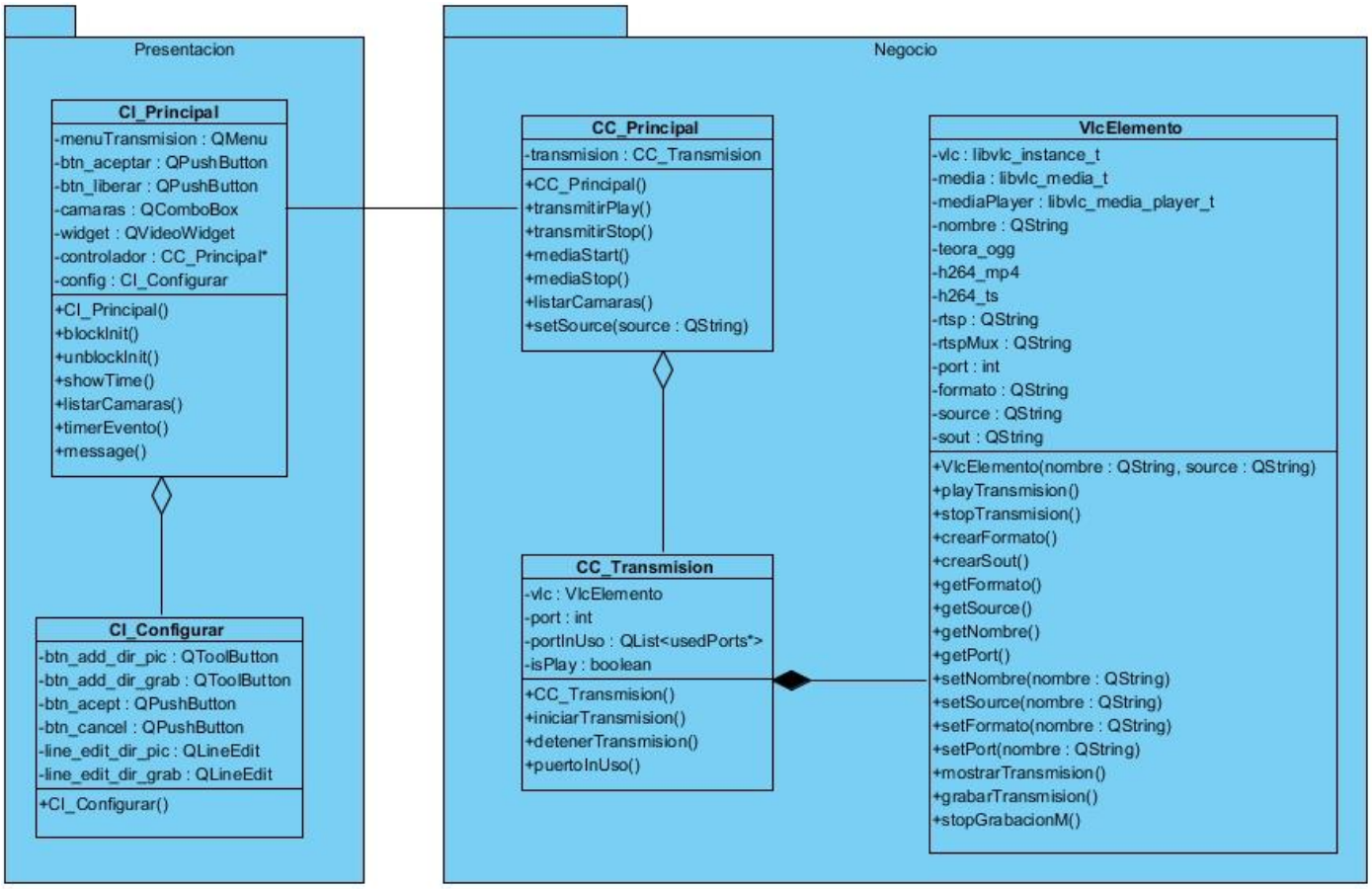


Figura 3: Diagrama de Clases del Diseño CU “Administrar la transmisión del flujo de video”

Mediante la clase CI_Principal el especialista puede acceder a todas las funcionalidades que posee la aplicación de escritorio. Posee un objeto de tipo CC_Principal que es la clase controladora. La clase CC_Principal es la encargada de recibir las peticiones que le son enviados desde la clase interfaz, esta redirecciona a la clase controladora relacionada con la petición realizada, en este caso la clase CC_Transmisión, encargada de realizar la transmisión del video a través del protocolo RTSP o HTTP, haciendo uso de un objeto de tipo VlcElemento.

CU “Mostrar flujo de video transmitido”

En el diagrama de clases del diseño referente al CU “Mostrar flujo de video transmitido” se identifican tres paquetes: Vista, Controlador y Modelo, los que corresponden a las capas del patrón de diseño MVC.

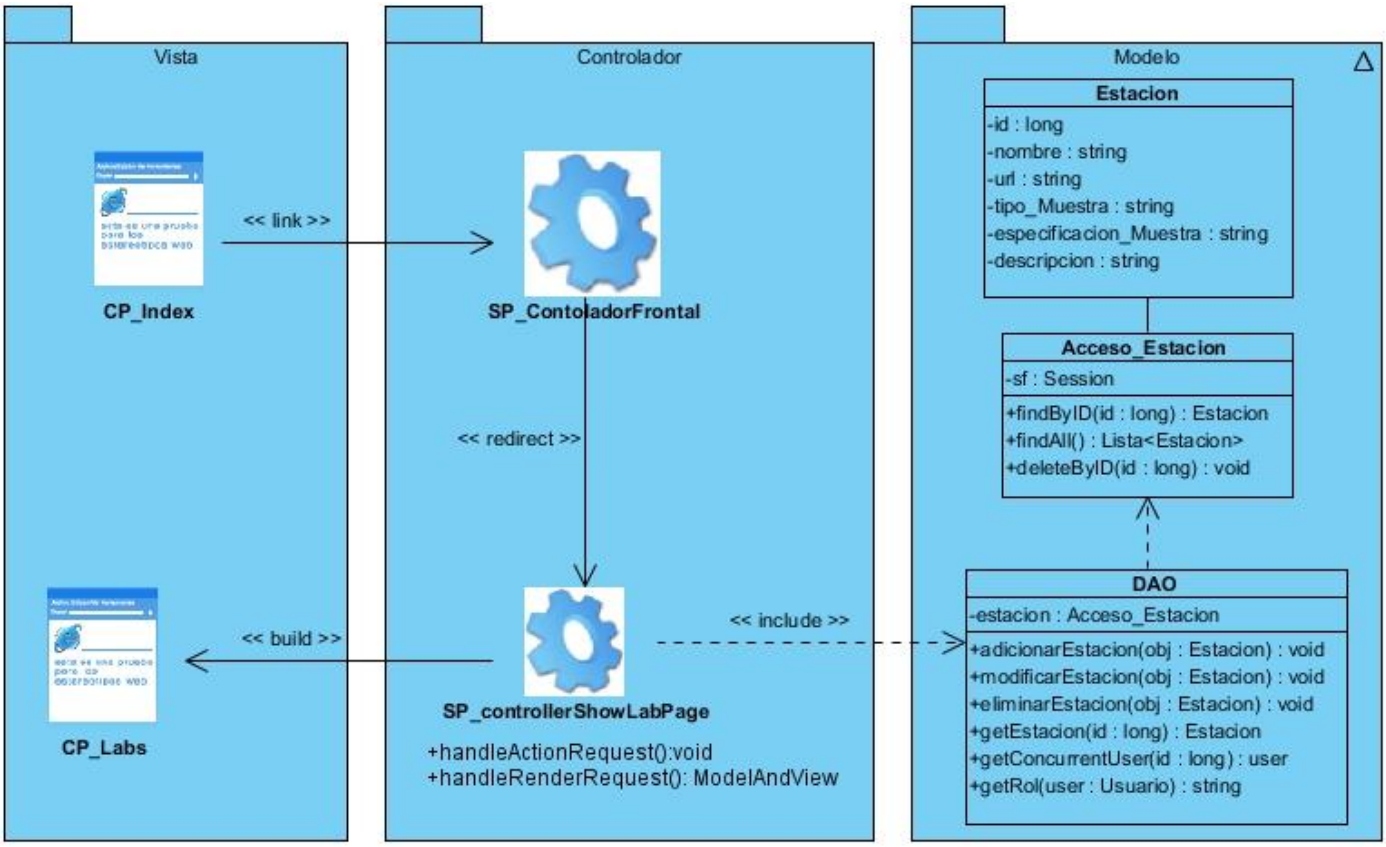


Figura 4: Diagrama de Clases del Diseño CU “Mostrar flujo de video transmitido”

El usuario accede a la interfaz principal, “CP_Labs”, a través de la clase “CP_Index”, esta genera una solicitud que es enviada a la clase “SP_ControladoFrontal”, que a su vez redirecciona a la clase “SP_controllerShowLabPage”. La clase “CP_Labs” se construye con una lista de las estaciones que se encuentren disponibles, además contiene todas las funcionalidades necesarias para la visualización de los flujos de video.

En el diagrama de clases del diseño del CU “*Mostrar flujo de video transmitido*” se define la clase “DAO”. Esta contiene todos los métodos necesarios para el acceso y gestión de las tablas de la base datos, utilizando para ello las clases “Acceso_Estacion” y “Estacion”. Esta clase implementa el patrón de diseño DAO, ya que se encarga de la gestión de los datos entre las clases que contienen la lógica del negocio y las entidades.

2.5.4. Patrones de diseño

Los patrones de diseño son soluciones a problemas específicos y comunes que se presentan en el diseño de un software, permiten volver a utilizar la experiencia obtenida por otros desarrolladores, al clasificar y describir formas de solucionar los problemas más frecuentes (Tedeschi 2013).

Patrones GRASP

Los patrones GRASP (acrónimo de General Responsibility Assignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos, son una serie de buenas prácticas de aplicación recomendable en el diseño de software (Larman 1999).

- **Experto**

La clase que contiene toda la información que es necesaria para que realice las responsabilidades asignadas. Esto posibilita que los sistemas sean más fáciles de entender, mantener y ampliar, y existen más oportunidades para reutilizar componentes en futuras aplicaciones.

Este patrón se pone de evidencia a través del Controlador Frontal o Distpacher, ya que se encapsulan todas las responsabilidades e información en el mismo, o en el caso de la aplicación de escritorio a través de la clase VlcElemento, ya que esta es la experta en conocer todo lo relacionado con el video.

- **Creador**

Se le asigna a la clase B la responsabilidad de crear instancias de la clase A al agregar, contener, registrar instancias, utilizar o inicializar los datos de uno o varios objetos de tipo A (Larman 1999). Este patrón se evidencia en el Controlador Frontal, pues este contiene toda la información para crear instancias de los objetos mediante el Bean Factory y pasar las dependencias mediante la inversión de control, manifestándose así el patrón inyección de dependencia.

- **Controlador**

El patrón controlador asigna una responsabilidad de recibir o manejar un mensaje de evento del sistema (Larman 1999). Este se evidencia en cada uno de los controladores que se encargan de procesar una petición en particular. Esta tarea le es asignada a dichos controladores por el Controlador Frontal, representando un sistema global. En la aplicación de escritorio se utilizó para decidir qué clase es la encargada de manejar los eventos, estando presente en la clase CC_Principal pues es la encargada de recibir los datos que son enviados desde la interfaz y enviarlo a las distintas clases según el método

llamado.

- **Alta cohesión**

Las clases con alta cohesión tienen una serie moderada de responsabilidades en un área determinada, pero siempre colaborando con otras para realizar las tareas (Larman 1999).

En la aplicación de escritorio fue utilizado para asignar responsabilidades a clases que estén altamente relacionadas con la trasmisión de video garantizando la contribución entre estas para realizar tareas de elevada complejidad en vez de utilizar una sola clase para ello.

En la aplicación Web el Controlador Frontal o Distpacher se encarga de mantener una alta cohesión entre vistas y controladores delegando funcionalidades en dependencia de la petición generada por el usuario.

- **Bajo acoplamiento**

Las clases con bajo acoplamiento no dependen de muchas otras. Este es una medida la fuerza con la que las clases están conectadas con otras (Larman 1999). Permite la reutilización aumentando la productividad, el diseño de las clases más independientes y reduce el impacto de los cambios.

En la aplicación de escritorio se utilizó para asignar responsabilidades a otras clases reduciendo las dependencias al mínimo, lo más que se pueda facilitando la reutilización de código. Por otro lado en la aplicación Web el sistema posee un bajo acoplamiento entre las vistas y los controladores, ya que estos no conocen el controlador encargado de procesar su información.

Patrones GoF

Los patrones "*grupo de los cuatro*", acrónimo de "*Gang of Four*" (GoF), también forman parte del diseño y proponen soluciones para diferentes clases de problemas que pueden presentarse durante el desarrollo del software. A continuación se muestran los más utilizados en el diseño de la solución.

- **Fachada**

El patrón Fachada proporciona una única interfaz a un conjunto de clases de un subsistema. se utilizó para simplificar el acceso al conjunto de clases proporcionando una única clase a utilizar para comunicarse con las demás, en la aplicación Web lo hace a través de la clase DAO del paquete de Acceso a datos.

- **Comando**

Este patrón soluciona el problema de cuando un objeto o sistema recibe varias peticiones o comandos, para ello cada comando define una clase que lo represente y le asigna la responsabilidad de ejecutarse el mismo, de esta forma reduce la responsabilidad del receptor en el manejo de los comandos, aumenta la facilidad con que pueden agregarse otros comandos y ofrece las bases para registrar los comandos, formar colas de espera con ellos y cancelarlos.

Otros patrones utilizados en la solución:

- **DAO**

El patrón *DAO* (por sus siglas en inglés *Data Access Object*) es un patrón de diseño que centraliza todo el acceso a datos en una capa independiente, aislándolo del resto de la aplicación. Su principal beneficio es que reduce la complejidad de los objetos de negocio al abstraerlos de la implementación real de la comunicación con la fuente de datos y de esta forma permitir una migración más fácil de fuente de datos.

Conclusiones Parciales

En este capítulo fueron definidos los conceptos asociados al modelo de dominio logrando una mejor visión del sistema y de sus componentes, permitiendo el análisis de la solución a implementar. Se analizaron todos los requisitos de software tanto los funcionales como no funcionales. Fueron descritos los casos de uso del sistema, diagrama de casos de uso del sistema y actores, además de realizarse los diagramas de análisis y secuencia de los casos de uso más significativos, creando la base para sustentar las siguientes fases de desarrollo.

Capítulo 3: Implementación y Pruebas

Introducción

En el transcurso del siguiente capítulo se realiza el flujo de trabajo de implementación, describiéndose detalladamente las técnicas utilizadas en el desarrollo de la aplicación, se analizan los artefactos principales que incluyen componentes, subsistemas de implementación y diagramas de componentes. Además de determinar los tipos de pruebas a realizar y los casos de pruebas que serán aplicados al sistema.

3.1. Modelo de implementación

El modelo de implementación describe como se implementan los elementos del diseño en términos de componentes tales como ficheros de código fuente, ejecutables, librerías y documentos.

3.1.1. Diagrama de componentes

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces (Microsoft 2015b). Representa como cada subsistema es dividido en componentes y muestra las dependencias entre ellos. A continuación se muestra el diagrama de componentes para los CU: “Administrar la transmisión del flujo de video” y “Mostrar flujo de video transmitido”.

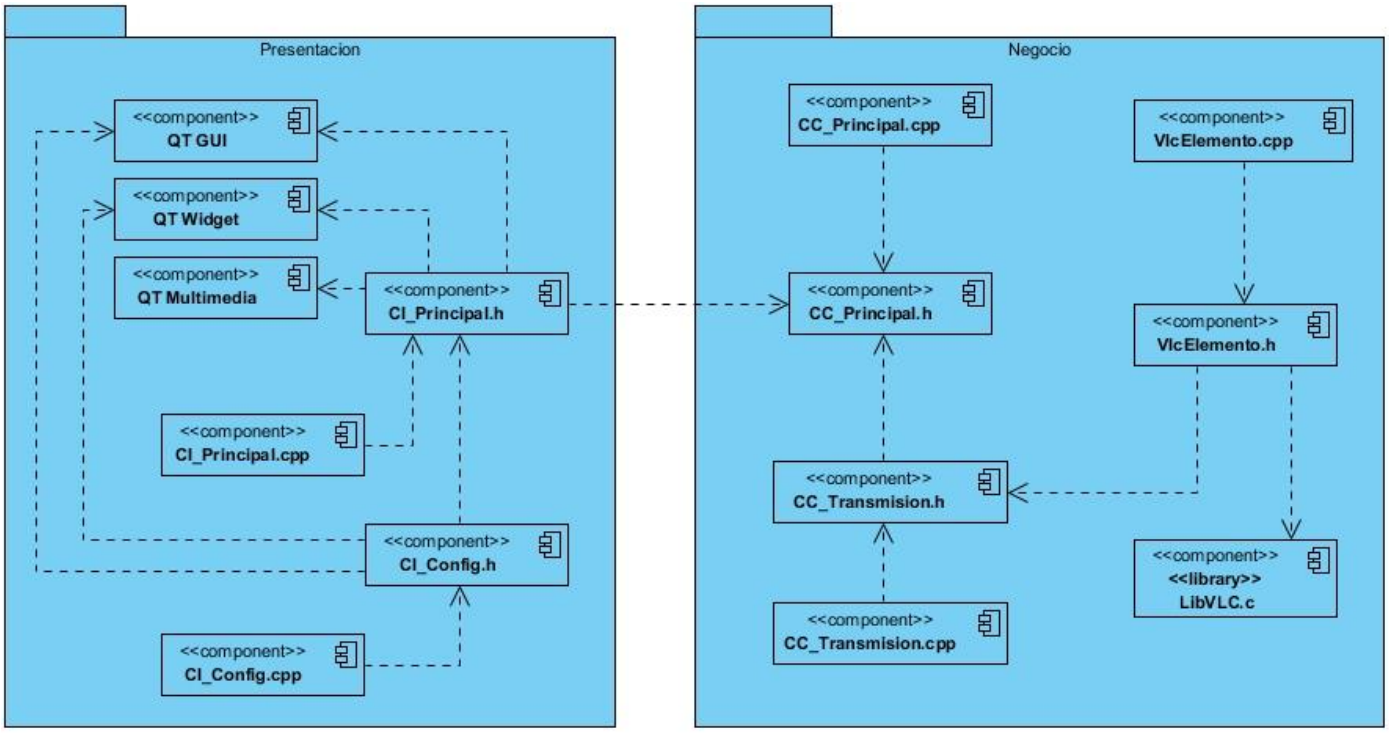


Figura 5: Diagrama de componentes CU "Administrar la transmisión del flujo de video"

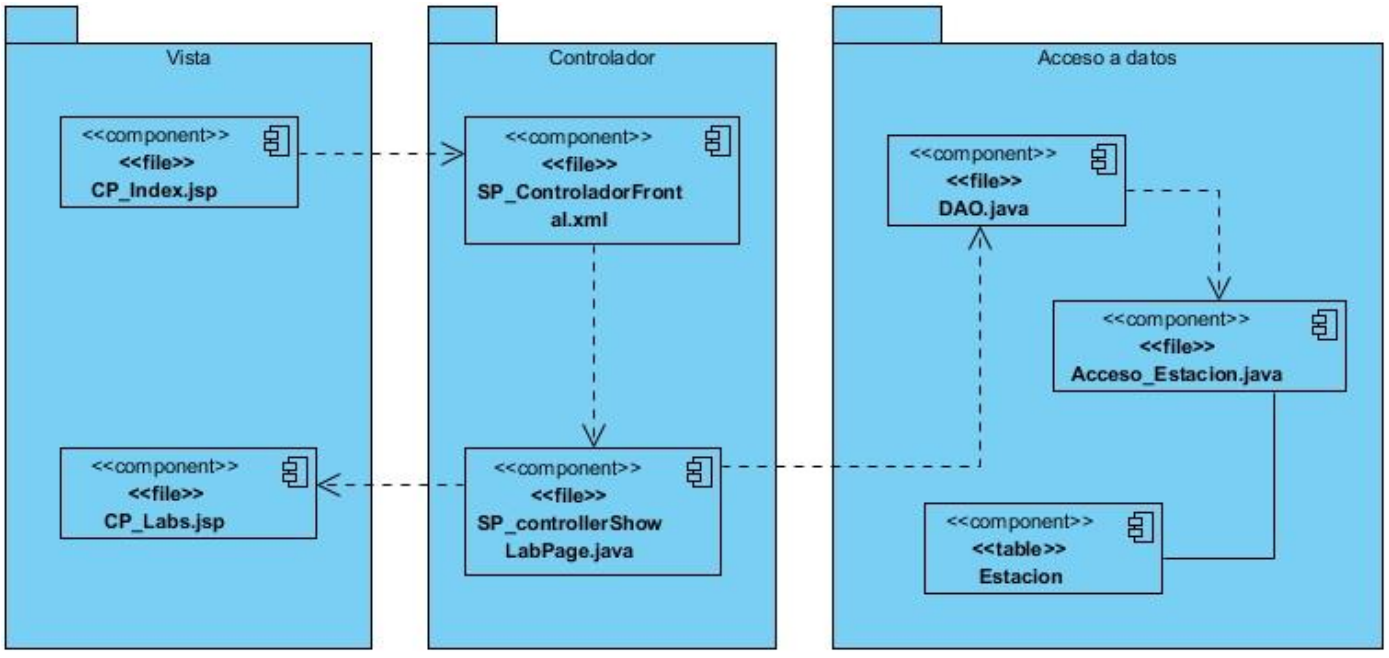


Figura 6: Diagrama de componentes CU "Mostrar flujo de video transmitido"

3.1.2. Modelo de Despliegue

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación que muestra las relaciones físicas entre los componentes hardware y software en el sistema propuesto. A continuación se muestra el diagrama de despliegue con la descripción de los elementos que lo conforman.

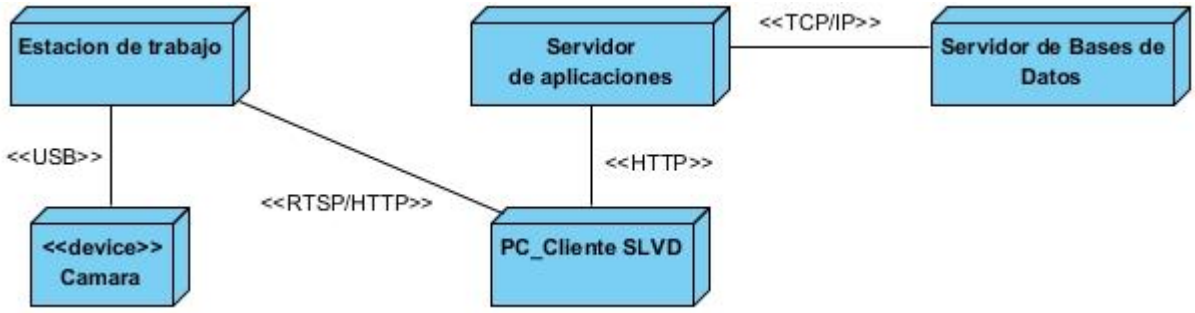


Figura 7: Diagrama de Despliegue

A continuación se describe cada uno de los nodos que conforman el diagrama de despliegue:

- Cámara: Dispositivo físico encargado de emitir flujos de video.
- Estación de trabajo: Se encuentra desplegada la aplicación de escritorio desarrollada. Por otro lado la estación de trabajo debe contar con la librería LibVLC en su versión 2.2.0 instalada, es utilizada por la aplicación de escritorio para la transmisión del flujo de video de la cámara haciendo uso del protocolo RTSP[1] o HTTP.
- Servidor de aplicaciones: Se encuentra alojado el servidor Web Apache Tomcat en su versión 7.0.1, y cuenta con el contenedor de portlet[2] Liferay Portal, versión 6.2.0, en este se encuentra desplegado el portlet para la microbiología.
- Servidor de bases de datos: Contiene el servidor de bases de datos PostgreSQL en su versión 9.4, al cual se accede a través del puerto 5432, haciendo uso del protocolo TCP/IP.
- PC cliente del SLVD: Terminal conectada a la red utilizada por el cliente para acceder al portlet de microbiología en el SLVD a través de un navegador Web.

El dispositivo de tipo cámara que se encuentra conectado a la Estación de trabajo emite un flujo de video, es decodificado y transmitido por la aplicación de escritorio desarrollada haciendo uso del protocolo RTSP

[1] RTSP: *Real Time Streaming Protocol, en español "Protocolo de Flujo en Tiempo Real".*
[2] Portlet*: Componentes modulares de las interfaces de usuario gestionadas y visualizadas en un portal Web.*

o el HTTP. El cliente accede a través de una computadora al portlet alojado en el SLVD, a través del protocolo HTTP y permite visualizar los flujos de video que se encuentran siendo transmitidos desde las estaciones de trabajo. Previamente se deben haber colocados en la bases de datos, accedida a través del protocolo TCP/IP, las URLs de acceso al flujo de video.

3.2. Aspectos fundamentales de la implementación

Una vez concluida la fase de análisis y diseño se procedió a la implementación de los diferentes componentes de la solución planteada. El proceso de desarrollo de los mismos fue guiado por los diagramas de clases y se respetaron las características de cada uno de los patrones arquitectónicos escogidos según el componente. A continuación se describe cada uno de estos componentes.

3.2.1. Descripción por áreas de las interfaces de usuario

Aplicación de Escritorio (Estación de trabajo)

a) Panel de visualización: Área donde se muestra lo que se encuentre visualizando la cámara.

b) Opciones de configuración: Área que permite entrar a la interfaz de configuración.

c) Opciones de transmisión: Permite al usuario acceder a las opciones correspondientes con la transmisión.

d) Opciones de realizar capturas: Permite a los usuarios realizar fotografía o grabaciones con la cámara escogida.

e) Lista de cámaras: Muestra la lista de cámaras disponibles para su utilización. El usuario puede escoger una de estas y visualizarla.

f) Detalles y notificaciones: En esta área se muestran al usuario las notificaciones del sistema. Se muestra además información adicional sobre la transmisión (puerto, códec, resolución y URL[3]), de encontrarse realizando una.

[3] URL: *Uniform Resource Locator (en español, " Localizador de Recursos Uniforme")*

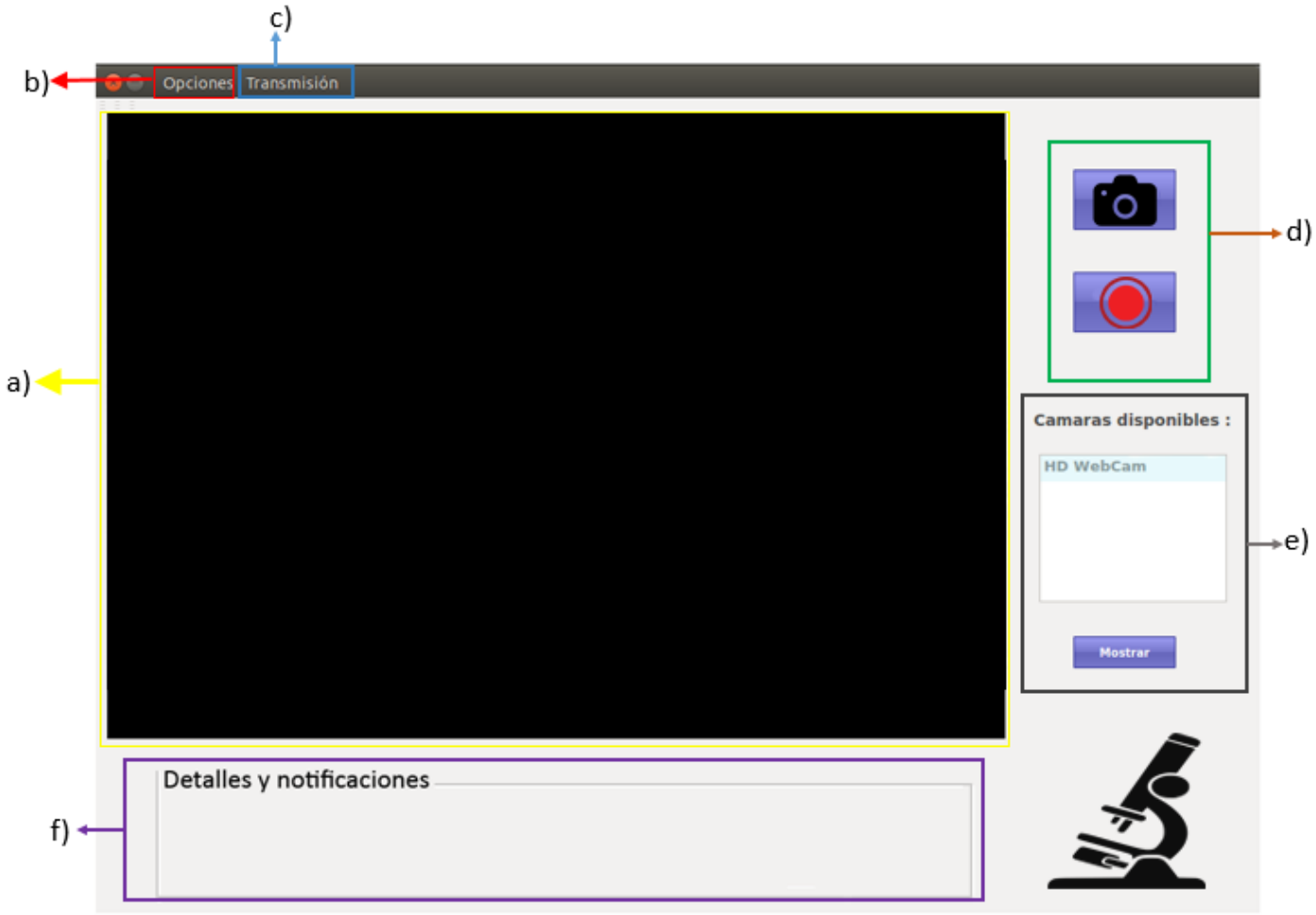


Figura 8: Interfaz gráfica de la Aplicación de Escritorio

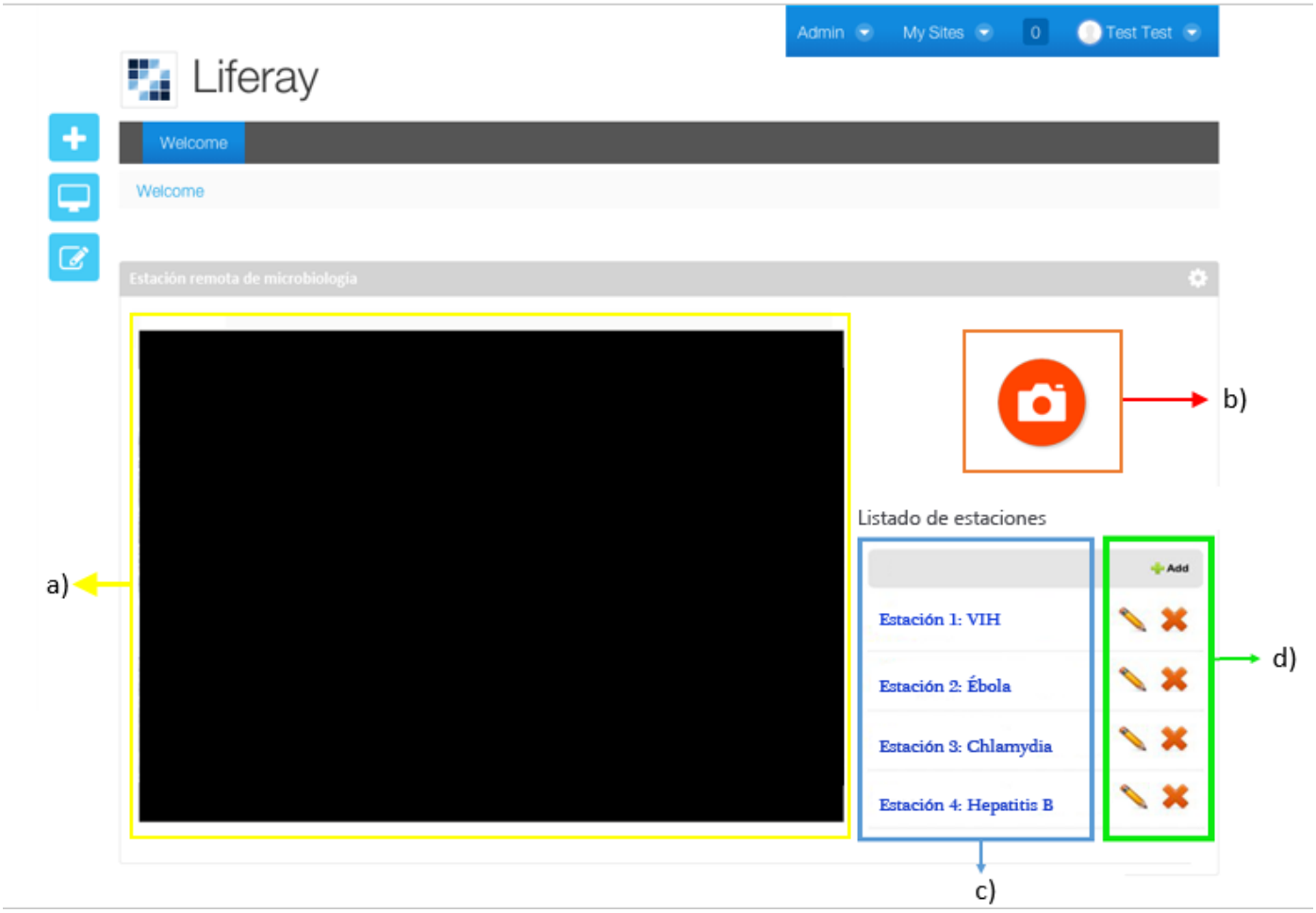
Aplicación Web (Portlet en el SLVD)

a) Panel de visualización: Área donde se muestra el flujo de video que se encuentre transmitiendo la estación escogida.

b) Opciones para realizar captura: Permite al usuario realizar capturas el flujo de video que se encuentre visualizando.

c) Lista de estaciones: Muestra la lista de estaciones disponibles para su visualización.

d) Opciones de origen: Esta área solo podrá visualizarla un usuario con permisos de administración pues permite la gestión de la lista de estaciones accesibles.



3.2.2. Estándares de codificación

Los estándares de codificación comprenden todos los aspectos de la generación de código, este refleja un estilo armonioso estableciendo como operar con la base de código existente (Microsoft 2015a). Estos facilitan que el código generado en la fase de implementación se encuentre bien organizado, sea comprensible, sencillo y rápido de detectar errores. Se logra así una aplicación eficiente en términos de mantenimiento y extensión de acuerdo a las nuevas necesidades que surjan en el negocio.

En el desarrollo de la solución planteada se utilizan los siguientes estándares de codificación:

- lowerCamelCase: Es una convención de nombres que forma parte de CamelCase en el que el nombre se forman de múltiples palabras que se unen como una sola donde la primera letra del nombre comienza con mayúscula (TechTarget 2015b). Específicamente en el caso de

lowerCamelCase la primera letra del nombre comienza con minúscula.

```
void VlcElemento::crearSout()
{
    mySout.clear();
    QString auxSout="";
    auxSout.append(soutFormat);
    auxSout.append(soutProtocol);
    mySout=auxSout.arg(w,h);

}
```

Figura 9: Fragmento de código de la clase: "VlcElemento"

- UpperCamelCase: Esta convención de nombres también forma parte de CamelCase, pero esta específica que la primera letra es con mayúscula, siendo fácil distinguirla de los nombres lowerCamelCase. Se utiliza mayormente para definir los nombres de las clases, a continuación se muestra un ejemplo.

```
private:
    CC_Transmision *transmision;
    CC_Media *media;
    CC_Media *getMedia();
    CC_Transmision *getTransmision();
    QString name;
    QString source;
    QString format;
    int port;
```

Figura 10: Fragmento de código de la clase: "CC_Principal"

- Sentencias de inclusión o importación: No existen líneas en blanco entre estas. A continuación se muestra un ejemplo en la clase “VlcElemento”

```
#include "vlc/vlc.h"
#include "vlc/libvlc.h"
#include "vlc/libvlc_events.h"
#include "vlc/libvlc_media.h"
#include "vlc/libvlc_vlm.h"
#include <QObject>
#include "QDebug"
#include <QString>
```

Figura 11: Fragmento de código de la clase: "VlcElemento"

- Las secciones de código se encuentran alineadas.
- Número de declaraciones por líneas: Se declara cada variable en una línea distinta.
- Se utilizan espacios antes y después de los operadores.
- Se emplean las letras i, j, k como contadores.

3.3. Pruebas de software

Las pruebas de software son un elemento esencial para la garantía de la calidad del software y representan una revisión final de las especificaciones del diseño y codificación, Estas consisten en la verificación del comportamiento del sistema ante un conjunto de casos de pruebas. Su principal objetivo es evaluar y valorar la calidad del sistema a través de:

- Encontrar y documentar los defectos del software.
- Validar el correcto funcionamiento del software de acuerdo a su diseño.
- Probar los requisitos que debe cumplir el software.
- Validar la correcta implementación de los requisitos.

Para el cumplimiento de estos objetivos, debido a su carácter crítico para la garantía de haberse alcanzado una buena calidad en el desarrollo del sistema se escogió realizar pruebas al sistema, al centrarse, este nivel, en detectar fallas en el cubrimiento de los requerimientos.

3.3.1. Niveles de Pruebas

Los niveles de pruebas son diferentes ángulos de verificar y validar en determinados momentos el ciclo de vida del software. En el desarrollo de las fases de pruebas se aplicarán las pruebas que abarcan los

siguientes niveles:

- **Unidad:** Las pruebas unitarias tienen el objetivo de detectar errores en los datos, la lógica y los algoritmos y comprobar el flujo de datos entre los componentes, comprobando el correcto funcionamiento de cada método o función por separado. Las pruebas unitarias tienen que poder repetirse tantas veces como uno quiera, por este motivo, la rapidez de las pruebas tiene un factor clave. Dentro del nivel unidad fueron utilizadas pruebas funcionales a través del método de cajas blancas haciendo uso de la técnica de camino básico.
- **Integración:** Las pruebas de integración tienen como objetivo detectar errores de interfaces y de las relaciones entre los componentes. Por otra parte permiten coger los módulos probados mediante la prueba de unidad y crear una estructura de programa que esté de acuerdo con lo que dicta el diseño. Fueron desarrolladas pruebas de tipo integración a través de los casos de pruebas del método de cajas negras y la técnica de partición y equivalencia.
- **Sistema:** Las pruebas al sistema tienen como propósito ejercitar profundamente el sistema, para verificar que se han integrado adecuadamente todos los elementos de este (hardware y software). Estas tratan de detectar fallas en el cubrimiento de los requerimientos al comprobar los requisitos del sistema. Se ejecutaron pruebas relacionadas con el rendimiento del sistema, siendo utilizadas las pruebas de carga para comprobar la cantidad de usuarios que pueden conectarse al flujo de video.

3.3.2. Tipos de pruebas

A partir de los niveles de pruebas anteriormente explicados fueron seleccionados los siguientes tipos de pruebas:

- **Pruebas Funcionales:** Las pruebas funcionales se centran en validar si el comportamiento del software cumple o no con los requerimientos propuestos. Estas se centran desde el punto de vista del usuario, para así validar su conformidad con esta. Raramente es considerada la estructura interna, enfocándose fundamentalmente en realizar una serie de entradas al programa y comprobar que las salidas de este son las esperadas.
- **Pruebas de integración:** Las pruebas de integración contribuyen a determinar la correcta interacción entre los distintos componentes del sistema.

- **Pruebas de rendimiento del sistema:** Las pruebas relacionadas con el rendimiento del sistema están diseñadas para probar el software en tiempo de ejecución dentro del contexto de un sistema integrado. Por otro lado se puede comprobar la cantidad de usuarios que pueden ejecutar una determinada acción de forma concurrente.

3.3.3. Métodos y técnicas de pruebas

Por otro lado, para la realización de los distintos tipos de pruebas seleccionados se definieron los siguientes métodos y técnicas de pruebas.

Método de Cajas Blancas

El método de Cajas Blancas se realiza sobre las funciones internas de un módulo, por otro lado utiliza la estructura de control del diseño procedimental para obtener los casos de pruebas (Pressman 2002). Mediante la aplicación de estas pruebas se busca:

- Garantizar que se ejercita por lo menos una vez todos los caminos independientes de cada módulo.
- Ejercitar todas las decisiones lógicas en sus vertientes verdadera y falsa.
- Ejecutar todos los bucles en sus límites y con sus límites operacionales.
- Ejercitar las estructuras internas de datos para asegurar su validez.

Técnica de Cajas Blancas "Camino básico"

Esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico (Pressman 2002). La idea es derivar casos de prueba a partir de un conjunto de caminos independientes por los cuales puede circular el flujo de control. Este camino se representa a través de un grafo de flujos, lo que permite determinar la complejidad ciclomática y la cantidad de caminos independientes, dando el número de pruebas a realizar.

Método de Cajas Negras

El método de pruebas Cajas Negras se centra en las interfaces del sistema, ya que se conocen sus funciones específicas y se pretende demostrar que cada una es operativa, no sin dejar de buscar posibles errores (Pressman 2002). Se demuestra así la operatividad del sistema, ya que ante determinada entrada del usuario se produce la respuesta indicada.

Las pruebas de Cajas Negras buscan encontrar errores que se encuentren dentro de las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en las estructuras de datos o en el acceso a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Técnica de Cajas Negras "Partición y equivalencia"

Para el diseño de los casos de prueba de Cajas Negras se hará uso de la técnica de partición de equivalencia, que divide los campos de entrada de un programa en variables de equivalencia con juegos de datos de entrada y salida. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa.

3.4. Aplicación del método de Cajas Blancas

Para la realización de las pruebas de Cajas Blancas haciendo uso de la técnica de camino simple fue utilizada la función *"handlerRenderRequest"* del controlador "controllershowLabPage". En esta se identificaron y enumeraron los bloques de ejecución, alcanzando un total de 8, identificando el camino básico entre ellos. Para cada sentencia condicional fue identificado un nodo predicado del cual se derivan varios caminos a seguir.

Por otro lado con el objetivo de obtener una medición cuantitativa de la complejidad lógica del camino determinado fue obtenida la complejidad ciclomática ("V (G)") restando a la cantidad de aristas (A) la cantidad de nodos (N) y sumándole dos.

$$V(G) = A - N + 2$$

Además se comprobó el resultado obtenido realizando el cálculo a través de otra fórmula, donde se utiliza la cantidad de nodos predicados (P):

$$V(G) = P + 1$$

A continuación se muestran los pasos seguidos para la realización de las pruebas:

Figura 12: Fragmento de código utilizado en las pruebas unitarias.

La figura 12 hace alusión al método "handlerRenderRequest" el cual se encarga de manejar los permisos de los usuarios a las distintas interfaces y funcionalidades implementadas. El código se encuentra dividido en fragmentos y enumerados para así construir el camino básico. A continuación se muestra el grafo de flujos asociado a este método.

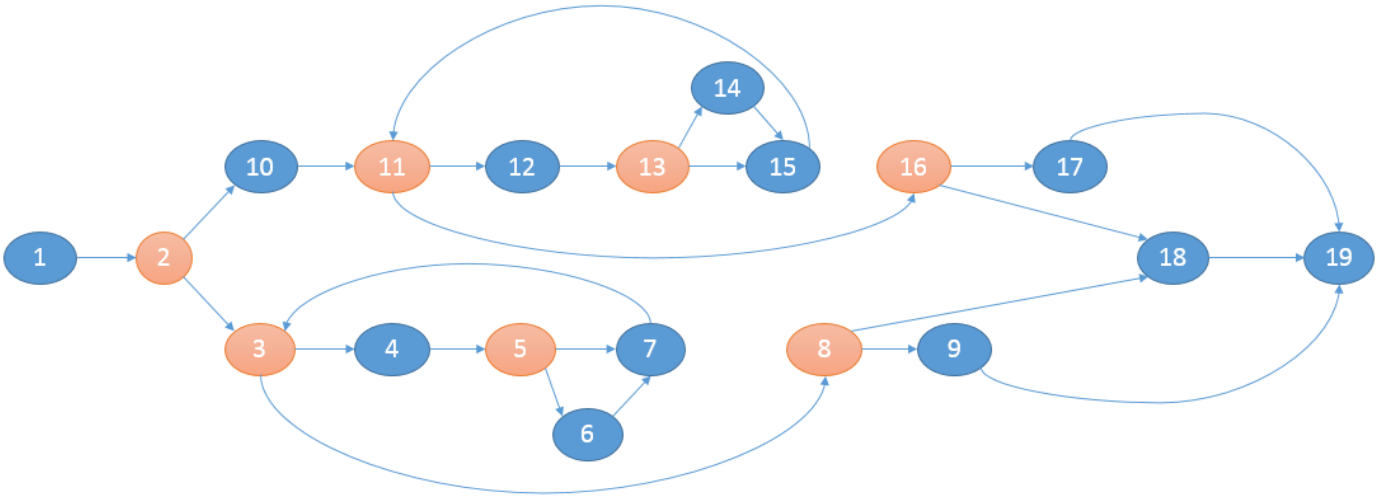


Figura 13: Grafo de flujo asociado a las pruebas unitarias.

Luego de realizado el grafo y calculada la complejidad ciclomática (“V(G)= 8”) se procede a determinar los caminos linealmente independientes, tomando como nodos predicados a el nodo 2, 3, 5, 8, 11, 13 y 16:

- Camino 1: 1-2-3-8-9-19
- Camino 2: 1-2-3-8-18-19
- Camino 3: 1-2-10-11-16-17-19
- Camino 4: 1-2-10-11-16-18-19
- Camino 5: 1-2-3-4-5-7-3- …
- Camino 6: 1-2-3-4-5-6-7-3-…
- Camino 7: 1-2-10-11-12-13-15-11-…
- Camino 8: 1-2-10-11-12-13-14-15-11-…

Una vez determinados los caminos independientes se conforman los casos de pruebas, a continuación se muestra el caso de pruebas para el camino 6.

Caso de prueba “Camino 6”

El método “handlerRenderRequest” se encarga de validar el rol que desempeña el usuario que se encuentra autenticado en el sistema y darle acceso a las funcionalidades asociadas a su rol.

Primeramente se instancias una serie de variables entre las que se encuentra la lista de roles, una variable boolean, que permite el tratamiento de errores y una variable de tipo String que contendrá la información referente al “request”. Luego se comprueba que el valor que contiene la variable de tipo String es igual 1, de ser así se realiza un ciclo hasta que se termine la lista de roles y comparando si posee el rol de “Administrador”. De no poseer este rol retorna un mensaje de error.

3.5. Diseño de los Casos de Pruebas

Los casos de pruebas se realizan con el objetivo de determinar que una funcionalidad ha sido implementada satisfaciendo las necesidades del cliente. Para cada caso de uso debe estar asociado un caso de prueba que recoja la especificación de ese caso de uso, dividido en secciones y escenarios, detallando las funcionalidades descritas y describiendo cada variable que recoge el caso de uso en cuestión.

A continuación se muestran una serie de casos de pruebas, que servirán como muestra visual del proceso de pruebas realizadas a la solución desarrollada. Dichos casos de pruebas se describirán en tablas que contienen los siguientes campos:

- **Escenario:** Diferentes condiciones sobre las cuales se realizan las pruebas.
- **Descripción:** Breve descripción del escenario.
- **Variable de tipo orden:** Describe la orden o acción que se le manda a ejecutar al sistema a través de un botón.
- **Respuesta:** Descripción del resultado que debe dar el sistema al realizar la prueba.
- **Flujo central:** Describe los pasos que se deben seguir para realizar la prueba.

Caso de Prueba #1

Tabla 5: Caso de Prueba #1 CU “Administrar la transmisión del flujo de video”

Escenario	Descripción	Variable de tipo orden	Respuesta	Flujo central

EC 1.1 Se transmite un flujo de video correctamente.	En este escenario se obtiene como salida la transmisión del flujo de video que emite la cámara seleccionada.	Clic en el botón “Iniciar” del menú “Transmitir”.	El sistema comienza la transmisión del flujo de video haciendo uso del protocolo RTSP. Además muestra en el área de notificaciones el icono de “Transmitiendo” y los detalles referentes a la transmisión (puerto, URL, etc.).	Interfaz principal de la aplicación de escritorio/ seleccionar cámara/ clic en “Mostrar”/ Menú “Transmisión”/ clic en “Iniciar”
EC 1.2 No se transmite el flujo de video	En este escenario no se obtiene como salida la transmisión del flujo de video.	Clic en el botón “Iniciar” del menú “Transmitir”.	El sistema muestra un mensaje de error: “La transmisión no ha podido iniciarse, verifique su conexión a la red o los parámetros de video escogidos.”	Interfaz principal de la aplicación de escritorio/ seleccionar cámara/ clic en “Mostrar”/ Menú “Transmisión”/ clic en “Iniciar”

Caso de Prueba #2

Tabla 6: Caso de Prueba #2 CU “Mostrar flujo de video transmitido”

Escenario	Descripción	Variable de tipo orden	Respuesta	Flujo Central
EC 2.1 Visualización del flujo de video.	En este escenario se visualiza correctamente un flujo de video.	Clic en un elemento de la lista mostrada.	El sistema comienza a mostrar el flujo de video.	Interfaz principal del componente Web/ clic para seleccionar la estación transmisora.
EC 2.2 No se establece conexión con	En este escenario no se visualiza el flujo de video.	Clic en un elemento de la lista mostrada.	El sistema muestra un mensaje de error: “No se ha podido establecer	Interfaz principal del componente Web/ clic para seleccionar

el servidor			conexión con el servidor”	la estación transmisora.

3.5.1. Resultados de las pruebas de cajas negras

Para que un proceso de pruebas tenga éxito se requiere al final de un análisis de los resultados obtenidos, es decir, la evaluación del producto que se encuentra probando de acuerdo a todos los defectos y fallos del sistema encontrados a lo largo del proceso. A continuación se muestran los resultados que arrojó el proceso de pruebas.

Cajas Negras

Luego de realizarse las pruebas de cajas negras a través de los casos de pruebas asociados a cada caso de uso se comprobó el correcto funcionamiento de la solución desarrollada. Las respuestas fueron verificadas en cada caso, independientemente de que fuesen correctas o incorrectas, arrojando un total de 22 no conformidades en la primera iteración. Estas fueron siendo eliminadas en el transcurso de las siguientes iteraciones, a continuación se muestran las no conformidades por cada una de las iteraciones realizadas.

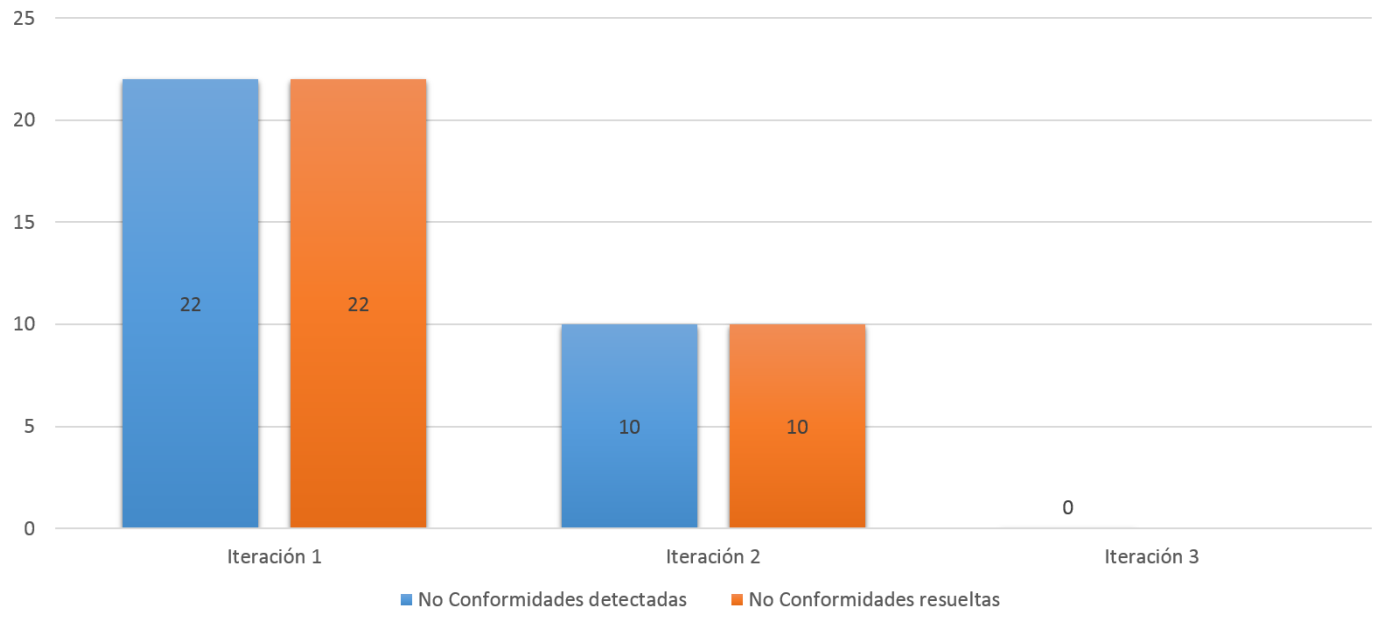


Figura 14: Resultados de los casos de pruebas

Integración

Luego de ser eliminadas todas las no conformidades resultantes del desarrollo de las pruebas de Cajas Negras se pudo llegar a la conclusión que la solución desarrollada se integra al Sistema de Laboratorios Virtuales y a Distancia, evidenciando su correcto funcionamiento. Al aplicar la estrategia de integración se pudo ir escalando en la integración de los distintos componentes del software, lo cual culminó con la puesta en marcha del módulo a través de la plataforma.

Pruebas de Carga

Las pruebas de carga permiten determinar y validar la respuesta de la aplicación cuando es sometida a una carga de usuarios y/o transacciones que se espera en el ambiente de producción. En esta prueba se comprobó el desempeño de la aplicación de escritorio al responder ante múltiples peticiones de video de forma simultánea. Para el análisis de las respuestas se debe tener en cuenta que fueron realizadas sobre una computadora con las siguientes características:

- Procesador: AMD A8-4555M a 1.6 GHz
- Memoria RAM: 6 GB
- Sistema Operativo: Ubuntu 15.04

A continuación se presentan los resultados de la prueba de carga realizada.

Tabla 7: Resultados de las Pruebas de Carga

Cantidad de usuarios	CPU (%)	RAM (%)	Respuesta del sistema
10	15	4.4	Transmite correctamente el flujo de video.
20	21	8.0	
40	28	12.9	
60	33	20.3	

Conclusiones Parciales

En este capítulo se construyó el modelo de implementación, para ello se realizaron los diagramas de componentes por cada caso de uso. Se definió el modelo de despliegue que muestra los elementos

necesarios para la solución: un ordenador donde se desplegará la aplicación de escritorio, un servidor de aplicaciones, un servidor de bases de datos, un servidor de medias y un ordenador cliente para acceder a través de la red a la aplicación Web. Además se definieron los casos de pruebas con el objetivo de comprobar la solución desarrolla, utilizando el método de cajas negras a través de la técnica de partición y equivalencia lo que arrojó una serie de no conformidades, las cuales fueron eliminadas en las siguientes iteraciones, obteniéndose la aceptación del cliente.

Conclusiones Generales

Tras haber culminado los tres capítulos con los que cuenta el presente trabajo de diploma, se logró dar cumplimiento a las tareas de la investigación, permitiendo arribar a las siguientes conclusiones:

1. La caracterización y revisión del estado del arte actual de los Sistemas de Laboratorios Virtuales y a Distancia y sus estaciones de trabajo permitió afirmar que las soluciones existentes, no resuelven la problemática planteada, expresándose la necesidad del desarrollo de la propuesta.
2. Las tecnologías y herramientas propuestas para el desarrollo de la solución se corresponden con las políticas de soberanía tecnológica que impulsa la universidad y el país.
3. Los artefactos generados durante el proceso de desarrollo permitirán continuar su implementación en el futuro y adecuarla a diferentes contextos.
4. Los casos de pruebas diseñados permitieron validar que lo solución desarrollada cumple con los requerimientos exigidos.

Recomendaciones

Al concluir esta investigación se recomienda para el desarrollo de estudios futuros:

1. Automatizar el funcionamiento del zoom del microscopio y agregarle esta funcionalidad, tanto a la estación de trabajo como al módulo de microbiología en el Sistemas de Laboratorios Virtuales y a Distancia.
2. Incorporar a la solución un servidor que se encargue de retransmitir el flujo de video que se encuentra siendo transmitido desde las estaciones de trabajo a los clientes del Sistema de Laboratorios Virtuales y a Distancia.
3. Automatizar el proceso de inserción de las estaciones que se encuentren transmitiendo en la lista que se le muestra a los usuarios en el Sistema de Laboratorios Virtuales y a Distancia.
4. Socializar la solución propuesta con el objetivo de potenciar la educación en las carreras que precisen realizar observaciones de la evolución de organismos microscópicos.

Referencias Bibliográficas

ALEGSA. ¿Cuál es la definicion de cámara digital? In., 2015.

ARIZA, C. AND D. AMAYA. Laboratorio remoto para la enseñanza de la programación de un robot industria. 2011, vol. 2, pp. 33-39. Available from Internet:<http://web.usbmed.edu.co/usbmed/fing/v2n1/v2n1a7.pdf>. ISSN 2027-5846.

AUMAILLE, B. *"J2EE Desarrollo de aplicaciones web"*. Edtion ed., 2002. ISBN 9782746019126.

BUSCHMAN, F., R. MEUNIER AND H. ROHNERT. Pattern-Oriented Software Architecture. A Sustem of Patterns. 1996, pp. 512.

CASTELLANOS, A., L. HERNÁNDEZ, R. ARACIL, E. RUBIO, et al. Sistema de Laboratorios a Distancia(SLD): laboratorio para la enseñanza del control automático a distancia. 2004, vol. 7, pp. 11-24. ISSN 1516-084X.

DICTIONARY.COM. Workstation. In. http://dictionary.reference.com/browse/workstation, 2015.

DIGITALES, S. "Caracteristicas del video. Concepto". 2008-2009.

DOWNEY, A. B. Pensando la computación como un científico (con Java) [online]. UNGS. Universidad Nacional de General Sarmiento, 2012. Available from World Wide Web:<http://www.ungs.edu.ar/areas/publicaciones/476/>.

ENTERPRISE, Q. Streaming. 2015.

FALGUERAS, B. C. AND U. O. D. CATALUNYA *Ingeniería del software*. Edtion ed.: Universitat Oberta de Catalunya, 2002. ISBN 9788484297932.

FERRE, M., I. SANTANA, R. ARACIL, L. HERNÁNDEZ, et al. Aplicación del Sistema de Laboratorios a Distancia en Asignaturas de Regulación Automática. *Revista Iberoamericana de Automática e Informática Industrial* [Type of Work]. 2010, vol. 7, pp. 46-53. ISSN 1627-7912.

GÓMEZ, P. El papel de la autoría en la mediación de los procesos de enseñanza aprendizaje en las

unidades didácticas de la educación a distancia. 2012, vol. 12, pp. 1-16. Available from Internet:<http://revista.inie.ucr.ac.cr/uploads/tx_magazine/papel-autoria-mediacion-procesos-ensenanza-aprendizaje-unidades-didacticas-educacion-distancia-gomez.pdf>. ISSN 1409 - 4703.

HERNÁNDEZ, R. A. AND S. C. GONZÁLEZ. El proceso de investigación científica [online]. [Ciudad de La Habana]: Editorial Universitaria 2011. Available from World Wide Web:<**¡Error! Referencia de hipervínculo no válida.**.

JACOBSON, I., G. BOOCH AND J. RUMBAUGH El proceso unificado de desarrollo de software 2004.

LAB, V. Virtual Lab. [Fecha de consulta 15 noviembre 2014]. In. Disponible en: http://www.vlab.co.in: Ministry of Human Resource Development (MHRD), 2014, p. National Mission on Education through ICT.

LABSTER. Virtual laboratories for High School and College. [Fecha de consulta 21 noviembre 2014]. In. Disponible en: http://www.labster.com/ 2014.

LARMAN, C. UML y Patrones. In. Prentice Hall, 1999.

LÓPEZ, J. M., U. GANGOITI, E. ZULUETA AND I. CALVO. Laboratorios remotos y virtuales en enseñanzas técnicas y científicas. 2009, vol. 3. Available from Internet:<http://www.ehu.es/ikastorratza/3_alea/laboratorios.pdf>. ISSN 1988-5911.

MARTÍNEZ, C. H. La educación a distancia: sus características y necesidad en la educación actual. 2008, vol. 17, pp. 7-27. ISSN 1019-9403.

MEDINA, A., G. HERMIDA, J. HERNÁNDEZ AND E. LADRÓN. Los Laboratorios Virtuales y Laboratorios Remotos en la Enseñanza de la Ingeniería. 2011, vol. 4. Available from Internet:<http://academiajournals.com/downloads/LorandiLabsEd11.pdf>. ISSN 1940-1116.

MICROSOFT. Revisiones de código y estándares de codificación. In. https://msdn.microsoft.com/es-es/library/aa291591(v=vs.71).aspx, 2015a.

MICROSOFT, D. N. Diagramas de componentes de UML: Referencia [online]. [https://msdn.microsoft.com/es-es/library/dd409390.aspx]: 2015b.

MOMJIAN, B. PostgreSQL Introduction and Concepts [online]. [http://www.cs.earlham.edu/~millebe/CS65/aw_pgsql_book.pdf]: Addison Wesley, 2000.

OMG. UML. In. http://www.uml.org/, 2015.

ORGANIZATION, V. LibVLC. In. http://www.videolan.org/vlc/libvlc.html, 2015.

PASCUAS, S., J. J. BOCANEGRA, E. J. ORTIZ AND J. N. PÉREZ. Desarrollo dirigido por modelos para la creación de laboratorios virtuales 2012, vol. 17, no. 51, pp. 119-125. Available from Internet:<http://www.redalyc.org/articulo.oa?id=84923910018>.

PEAK, P. AND N. HEUDECKER. Hibernate quickly, Dreamtech Press [online]. 2005.

PRESSMAN, R. S. *Ingeniería de Software, Un enfoque práctico*. Edtion ed.: Editorial McGraw-Hill, 2002. 640 p. ISBN 8448132149.

PRESSMAN, R. S. *"Ingeniería del Software. Un enfoque práctico"*. Edtion ed., 2005. ISBN 9701054733.

PRESSMAN, R. S. *"Software Engineering"*. Edtion ed.: New York: Higher Education, 2010.

RAE, R. A. E. Microscopio. In., 2014.

SCHULZRINNE, H. Real Time Streaming Protocol (RTSP). 1998.

SOFTWARE, P. Spring Framework. In. http://projects.spring.io/spring-framework/, 2015.

STROUSTRUP, B. Stroustrup: C++. In. http://www.stroustrup.com/C++.html, 1985.

SUPPRESS, G. D. I. E. A. Laboratorio Remoto de Automática. [Fecha de consulta 20 noviembre 2014]. In. Disponible en: http://lra.unileon.es: Universidad de Leon, España, 2014.

TECHTARGET. Portlets. In. http://searchsoa.techtarget.com/definition/portlet, 2015a.

TECHTARGET. What is lowerCamelCase? In. http://searchsoa.techtarget.com/definition/lowerCamelCase, 2015b.

TEDESCHI, N. Microsoft developer network. In. http://msdn.microsoft.com/es-es/library/bb972240.aspx, 2013.

UNILABS. UNILabs. [Fecha de consulta 21 noviembre 2014]. In. Diponible en: http://unilabs.dia.uned.es: Universidad Nacional de Educación a Distancia (UNED), 2014.

UNIVERSIDAD, B. Javaoptics (JOptics). [Fecha de consulta 20 noviembre 2014]. In. Disponible en: http://www.ub.edu/javaoptics/index-es.html: Grupo de Innovación Docente en Óptica Física y Fotónica, 2014.

VISUAL, P. Visual Paradigm. In. http://www.visual-paradigm.com, 2015.

YANG, M. Introduction to OpenUp. In. http://epf.eclipse.org/wikis/openup/index.htm, 2014.

ZAYAS, D. C. C. A. D. *METODOLOGIA DE LA INVESTIGACION CIENTIFICA* Edtion ed., 1995. ISBN 123254