Universidad de las Ciencias Informáticas

Facultad 6

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

Título: “Módulo de distribución automática de tareas a nodos de transcodificación en la Plataforma XILEMA AGORAV”

Autores:

Darian Pérez Fuentes

Amado Rafael Ramírez López

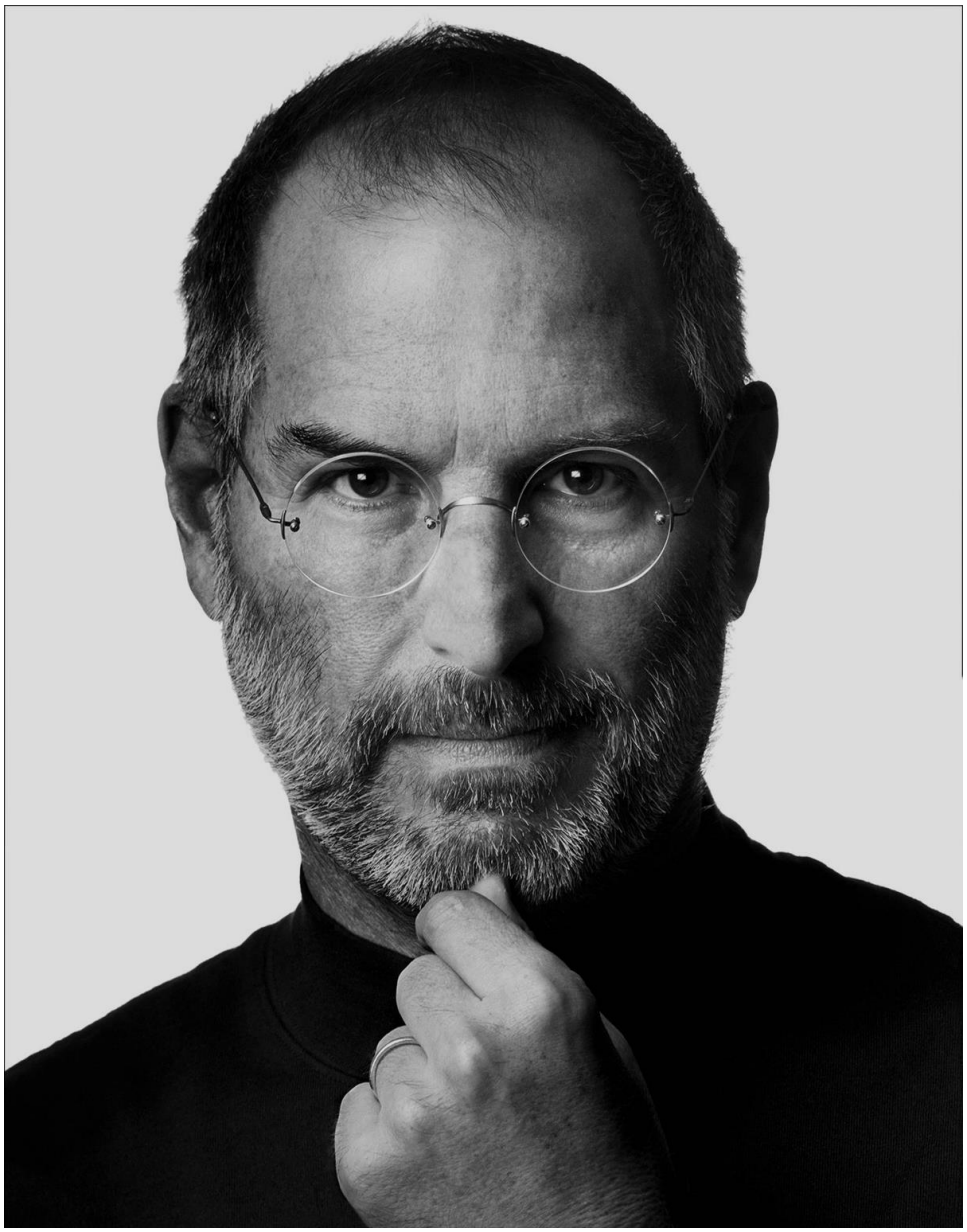
Tutores:

MSc. Rafael Leodán Caldero Álvarez

Ing. Vidal Orlando Acosta García

La Habana, julio del 2016

“Año 58 de la Revolución”



No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y, lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición.

Declaración de autoría

Declaramos ser los únicos autores del trabajo **Módulo de distribución automática de tareas a nodos de transcodificación** y se autoriza a la Universidad de las Ciencias Informáticas hacer el uso que estimen pertinente con el mismo.

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

_____	_____
Firma del autor	Firma del autor
Amado Rafael Ramírez López	Darian Pérez Fuentes
_____	_____
Firma del Tutor	Firma del Tutor
MSc. Rafael Leodán Caldero Álvarez	Ing. Vidal Orlando Acosta García

Dedicatoria

Amado:

A mis padres, por su confianza, por sus consejos, por ayudarme a ser cada día mejor y por estar junto a mí en todo momento...

A mi familia, por contar siempre con su apoyo...

Y muy especial a mi madre, que tanto deseó este momento.

Darian:

Dedico este trabajo de diploma, así como todos mis logros, a mi abuelita Idelina, a ella le debo todo lo que soy, y si hoy estoy aquí es por seguir todos sus consejos. A mi madre Isel, por todo su cariño y apoyo durante estos cinco años de lucha. A mi padre Daniel, por haber siempre confiado en mí. A mi tío Rey, por haberme acogido siempre como un hijo y responder por mí en todo momento. A mi primo Ernestico por estar siempre a mi lado en todos los momentos duros de la vida y darme tanto cariño; y a mi hermanito Bryan, por quererme tanto. A mi novia Jeniffer, por su amor y apoyo incondicional durante este largo camino. Mediante este trabajo exhorto a Bryan, Ernestico y Jeniffer, a que luchen por sus sueños y sigan adelante, aunque la vida le ponga pruebas y dificultades, todo con entrega y sacrificio se puede lograr.

Agradecimientos

Darian:

Primeramente quiero darle gracias a Dios por haber escuchado todas mis oraciones y haber caminado conmigo en esta larga travesía. Agradezco a mi abuelita Idelina por haberme dado todo el amor del mundo y más, por ser un ejemplo de entrega, sacrificio, entereza y lucha; por haber tenido en todo momento una sabia respuesta a cada una de mis preguntas, por hacerme sentir el nieto más feliz del mundo. Te amo mima. A mi mama Isel por enseñarme a luchar por lo que uno quiere realmente, por brindarme su apoyo y amor incondicional en todo tiempo, por llorar conmigo, reír conmigo y hacer las humbonancias juntos. A ti también te amo mami. A mi papá Daniel por haber estado siempre cuando lo necesité, porque nunca me faltó tu confianza y tu comprensión, por ser un ejemplo para mí. A mi tío Rey, por estar siempre pendiente de mí y quererme tanto, sin ti no hubiera sido posible haberme graduado, te quiero mucho. A mi primo Ernestico, que más que un primo es un hermano para mí, y a pesar de ser más pequeño que yo me ha enseñado muchas cosas de la vida. A ti primo te agradezco por quererme tanto y haber estado siempre conmigo en las buenas y malas. A mi hermanito Bryan, le agradezco todo su cariño y la felicidad que me transmite, gracias por tenerme en tu corazón. A mi novia Jeniffer, que aunque no pueda estar aquí con conmigo mi corazón está con ella porque me lo robó. Muchas gracias le doy por llenar mi vida de alegría y de amor, por ser promotora de momentos maravillosos, te amo mucho Jeni. Agradezco también a la familia de mi novia, en especial a mi suegra Consuelo por haberme acogido en su familia con tanto cariño. Sería un loco si no agradezco a mis amigos de toda la vida, Yordy, Noel, Gustavo, Carlos, Joel, Alejandro, Jessica juntos hemos pasado la mejor etapa de nuestras vidas, en las buenas y en las malas y si pudiera escoger de nuevo a mis amigos los elegiría a ustedes. Los quiero. Mis amigos de la UCI Jony, Manuel, Eddy, Araño, que juntos fundamos “La Wara” y se anexaros otros amigos como Joice, Orlando, Andy, Darian el blanco, Bermúdez, Marlon, entre otros para conformar así “La Wara ampliada”, a mi amiga Ivón, por haber sobrellevado todas mis pesadeces y siempre estar ahí cuando la necesité. Agradezco mis compañeros de apartamento, a mi gente de las fiestas. Todos han sido parte de mi formación profesional y como persona. A mi compañero de tesis el Rafa, que ha sido de gran ejemplo para mí, por la fuerza de voluntad que posee, por la entrega que lo caracteriza y siempre supo darme ánimo en los momentos más duros. Sin ti no lo hubiera logrado. A mis tutores que han sido incondicionales, y de lo mejor, este triunfo también es de ustedes. A los integrantes del proyecto AGORAV, Yeni, Pavel, Miguel, Ramón, que siempre estuvieron ahí para ayudarnos, cuando hizo falta y no les tembló la mano para señalar nuestras fallas.

Amado:

A mi mamá, por guiarme en el camino de la vida. Por todos los cuidados, por todo el amor y todos los sacrificios que ha hecho por mí. Gracias por hacer que mi vida sea feliz por tenerte a mi lado.

A mi hermano Yoel Alejandro por darme siempre alegría y por ser motivo de inspiración en mi vida.

A mi tía Mayelín, Esmerida y Maricel por apoyarme durante estos cinco años y brindarme su amor incondicional.

A mi novia Yoleisy por todo el amor que me da en cada momento y por siempre estar en los malos y buenos momentos de mi vida.

A mi padre por sus consejos, por educarme y por no perder la confianza en mí.

A mi primo Onayis por ser mi amigo, mi hermano y por darme los mejores consejos.

A mi abuelo, por aconsejarme y por enseñarme a ser mejor cada día.

A mis tíos, primos y primas por apoyarme durante estos cinco años.

A mis compañeros que a lo largo de la carrera he llegado a considerar hermanos, Edel, César, José Raúl, José Alejandro, Darian y Eddy, todos siempre presentes en las mejores y peores situaciones.

A nuestros tutores Rafael y Vidal, que con persistencia y rigor hicieron posible que este trabajo saliera adelante.

A las personas del proyecto en especial a los profesores Pavel y Yenisel.

Resumen

La distribución automática de tareas constituye en la actualidad un recurso indispensable en las empresas y negocios para lograr una mejora en sus procesos. En la presente investigación se identificó que el proceso de transcodificación en la Plataforma XILEMA AGORAV se ejecuta en un solo ordenador con recursos limitados, provocando retraso en el tiempo de publicación de los materiales audiovisuales. En busca de darle solución a la situación planteada anteriormente se decidió desarrollar un módulo para la distribución automática de tareas a nodos de transcodificación, que permita lograr en menor tiempo las publicaciones de los materiales audiovisuales en la Plataforma XILEMA AGORAV. Para ello se utilizaron lenguajes y tecnologías como HTML5, CSS, JavaScript, PHP5 y el CMS Drupal 7. Se obtuvo un módulo que distribuye automáticamente las tareas de transcodificación a los nodos, balanceando la carga de los mismos. Con su uso, se disminuye el tiempo de publicaciones de los materiales audiovisuales en la Plataforma XILEMA AGORAV. A dicho módulo se le realizaron pruebas de integración, de caja negra, de rendimiento y pruebas para la validación del mismo, constatándose que el mismo está listo para ser utilizado en la Plataforma XILEMA AGORAV.

Palabras clave: balanceo de carga, distribución automática, transcodificación.

.

ABSTRACT

Nowadays automatic distribution of tasks is an indispensable resource for companies and businesses to achieve an improvement in their processes. In this investigation it was identified that the transcoding process in the Platform XILEMA AGORAV is running on a single computer with limited resources, causing delay in the time of publication of audiovisual materials. In search to give solution to described situation above it was decided to develop a module for automatic distribution of tasks to transcoding nodes, which allows to achieve in less time publication of audiovisual materials in the Platform XILEMA AGORAV. For this purpose languages and technologies like HTML5, CSS, JavaScript, PHP 5 and CMS Drupal 7 were used. A module was obtained that automatically distributes transcoding tasks to nodes for balancing of load thereof. With its use, it is decreased the time of publication of audiovisual materials in the Platform XILEMA AGORAV. To this module were performed integration testing, black box testing, performance and testing for validation of the same, confirming that it is ready to be used in the Platform XILEMA AGORAV.

Keywords: *load balancing, automatic distribution, transcoding.*

Índice General

Índice de Figuras

Índice de Tablas

Introducción

En la actualidad, las Tecnologías de la Información y las Comunicaciones, conocidas como TICs, han sido incluidas en las esferas económica, política y social. Aparejado a la continua evolución de las TICs, en la década de 1990 fue posible incrustar videos en la web debido a un incremento del ancho de banda de la red de datos. Desde entonces, el video se ha convertido en un medio cotidiano de comunicación entre las personas, fomentando en ellas la necesidad de compartir contenidos y emociones a través del audiovisual. Dicha necesidad ha acelerado la evolución de la web hacia una nueva forma de buscar inmediatez audiovisual, mediante la cual se proporciona un amplio catálogo de opciones a los usuarios, así surgieron las plataformas audiovisuales (Bartolomé, y otros, 2007).

Las plataformas audiovisuales generalmente permiten la gestión, publicación y transmisión de contenido multimedia sobre la red de datos, a través de la web, mediante la tecnología *streaming*[1] (Creati, 2016). Las mismas cuentan generalmente con un flujo principal de trabajo, dicho flujo cuenta con los procesos de ingesta [2] del contenido procedente de diferentes fuentes, archivo multimedia, la catalogación, la transcodificación y la publicación del contenido audiovisual (Figura 1).

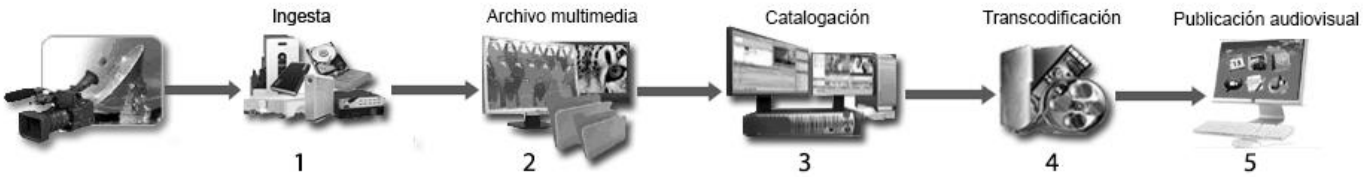


Figura 1. Flujo principal de trabajo de publicación de materiales audiovisuales

En las plataformas audiovisuales el proceso de transcodificación permite producir y servir videos en múltiples formatos o transformar varios formatos en uno solo, para utilidad de los consumidores. Además, dicho proceso requiere recursos de hardware más potentes que cualquier proceso básico ejecutado en un ordenador, como reproducir un audio, crear una carpeta o crear un documento. Lo dicho anteriormente se evidencia cuando se transcodifica de un formato de video a otro, lo cual implica comprimir y descomprimir el video, requiriéndose un alto tiempo y recursos de procesamiento. Además, cuanto más grande sea el video, más tardará en ser transcodificado.

[1] La tecnología *streaming* consiste en un servicio de transferencia de datos que se procesa como un flujo regular y continuo, y cuya principal ventaja es que el usuario no necesita descargar el contenido video o audio para poder visualizarlo o escucharlo. Además, permite la retransmisión en directo de un acontecimiento (Flumotion, 2015).

[2] Consiste en importar los materiales audiovisuales de diferentes fuentes, tanto analógicos como digitales (ISID, 2016).

Realizar la transcodificación de varios videos en un único ordenador con prestaciones convencionales, es muy lento. Como se evidencia en la Figura 1, lo dicho anteriormente provocaría demora en la publicación de los contenidos audiovisuales, ya que cuando se ingesta un video que no está en el formato requerido para ser publicado, el mismo debe ser transcodificado antes de la publicación. A diferencia de la situación descrita anteriormente, en la actualidad existen sistemas dedicados al trabajo con audiovisuales que realizan distribución de procesos. Plantea (Telestream, 2015) que dicha distribución generalmente permite un mejor aprovechamiento de los recursos de hardware donde se ejecuta cada proceso, además de lograrse una mayor velocidad de procesamiento del mismo, y por último se evita la sobrecarga de varios procesos en un único ordenador.

En Cuba también se trabaja con el proceso de transcodificación, un ejemplo es la Plataforma XILEMA AGORAV, la cual es desarrollada en el Centro de Desarrollo de Geoinformática y Señales Digitales (GEYSED) de la Universidad de las Ciencias Informáticas. La Plataforma XILEMA AGORAV tiene como objetivo la gestión, publicación y transmisión de contenido multimedia sobre la red de datos, a través de la web, mediante la tecnología *streaming*.

La Plataforma XILEMA AGORAV constituye una herramienta de gran utilidad para emisoras de radio y televisión que deseen publicar sus producciones audiovisuales, colocar su señal en vivo en una red interna o transmitirla vía Internet para hacerla accesible a un gran número de usuarios. Además, de ser utilizada en universidades, centros educativos con grandes redes de área local, ministerios y empresas de mediano y gran tamaño que deseen transmitir contenidos audiovisuales a sus trabajadores.

Actualmente en la Plataforma XILEMA AGORAV el proceso de transcodificación de los materiales audiovisuales se realiza en un solo ordenador con recursos limitados. Además téngase en cuenta que los videos poseen tamaños considerables y formatos avanzados en cuanto a la transcodificación, y debido a esto requieren una mayor disponibilidad de recursos de hardware, siendo aquí la Unidad Central de Procesos (CPU) el recurso más importante a tenerse en cuenta. Cuando varios usuarios ejercen el rol “Gestor de contenidos”, se incrementa el número de videos a transcodificar en dicho ordenador. Teniendo en cuenta la situación anteriormente planteada, se retrasa el tiempo de publicación de los materiales audiovisuales almacenados, afectando la actualidad de los mismos.

A partir de lo planteado anteriormente se define como el **problema de la investigación** “en la plataforma XILEMA AGORAV el proceso de transcodificación se ejecuta en un solo nodo, provocando retraso en el tiempo de publicación de los materiales audiovisuales”. El **objeto de estudio** de la investigación es “la distribución de procesos”, enmarcado en el **campo de acción** “la distribución de tareas en el proceso de transcodificación en la plataforma XILEMA AGORAV”. Para dar solución al problema planteado se define como **objetivo general** “desarrollar un Módulo para la distribución automática de tareas a nodos de

transcodificación, que permita lograr en menor tiempo la publicación de los materiales audiovisuales en XILEMA AGORAV".

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

- ¿Cuáles son los referentes teóricos para el desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación?
- ¿Qué elementos contribuyen a disminuir el tiempo de publicación de los materiales audiovisuales?
- ¿Cómo estructurar el proceso de desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación?

Para dar cumplimiento al objetivo general se definen las siguientes **tareas de la investigación**:

1. Definición de los principales conceptos, características y tendencias actuales relacionadas con la distribución de procesos.
2. Análisis del estado actual de la Plataforma XILEMA AGORAV, sus componentes y necesidades.
3. Caracterización de las soluciones similares existentes.
4. Selección de la metodología de desarrollo, las herramientas y tecnologías a utilizar durante el desarrollo del Módulo para la distribución automática de tareas a nodos de transcodificación.
5. Implementación del Módulo de distribución automática de tareas a nodos de transcodificación en la Plataforma XILEMA AGORAV.
6. Ejecución de pruebas al Módulo de distribución automática de tareas a nodos de transcodificación para validar el correcto funcionamiento del mismo.

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

Métodos Teóricos

- **Histórico-Lógico:** Los métodos históricos analizan la trayectoria completa del fenómeno, su condicionamiento a los diferentes períodos de la historia. Los métodos lógicos se basan en el estudio histórico del fenómeno y encuentran el conocimiento más profundo de su esencia. El método se utilizó para analizar la evolución de las investigaciones existentes referentes a la transcodificación y a la distribución de procesos, permitiendo la definición de términos propios.

- **Analítico-Sintético:** El análisis permite la división mental del fenómeno en sus múltiples relaciones y componentes para facilitar su estudio. La síntesis establece mentalmente la unión entre las partes previamente analizadas, posibilita descubrir sus características generales y las relaciones esenciales entre ellas. El método se utilizó para seleccionar las herramientas y tecnologías a utilizar durante el

desarrollo del módulo. Además, contribuyó al estudio de las soluciones existentes asociadas al dominio del problema, generándose nuevos conocimientos respecto a las cualidades y funcionalidades que el módulo debe cumplir.

Además, se empleó la siguiente técnica de recopilación de información:

Conflicto de diálogo: El conflicto de diálogo permite crear posiciones opuestas entre dos o más personas para estimular el debate y conocer los criterios de los interlocutores sobre el tema, abierta y espontáneamente. En el conflicto de diálogo participaron especialistas y directivos del proyecto AGORAV del Centro GEYSED. El debate generado permitió delimitar el dominio referente al problema de la investigación, obtener los requerimientos del módulo y analizar las consecuencias que genera el problema existente en la Plataforma XILEMA AGORAV.

Estructura del Documento

La presente investigación consta de tres capítulos. En el primer capítulo se fundamenta el marco teórico de la investigación, se caracterizan la metodología de desarrollo, las herramientas, tecnologías y lenguaje de programación utilizados en el proceso de desarrollo del Módulo para la distribución automática de tareas a nodos de transcodificación. En el segundo se exponen las principales características de la solución a implementar. Además, se identifican los requisitos, se escogen el patrón arquitectónico, los patrones de diseño y se diseñan los artefactos que componen la solución. En el tercero se muestra la organización del módulo mediante el diagrama de componentes y el diagrama de despliegue, se explica el estándar de codificación empleado y por último se evidencia con las pruebas necesarias el correcto funcionamiento de la solución.

Capítulo 1. Fundamentación teórica

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

1.1 Distribución de procesos

La distribución de procesos se enmarca en el campo de la computación distribuida. Es un modelo en el que los componentes situados en ordenadores en red se comunican y coordinan sus acciones mediante el paso de mensajes. Los componentes interactúan entre sí con el fin de lograr un objetivo común. A diferencia de un sistema de un solo ordenador, con la distribución de procesos al fallar alguno de los ordenadores otro puede hacerse cargo de la tarea del ordenador que ha fallado. Algunas de las ventajas y desventajas de la distribución de procesos son las siguientes (Wattenhofer, 2016):

Ventajas

- Permite el procesamiento distribuido.
- Usa distintos tipos de ordenadores.
- Puede proporcionar tolerancia a fallos.
- Permite escalabilidad, debido a que se puede incrementar el número de ordenadores según la demanda.

Desventajas

- Múltiples puntos de fallo: hay más puntos de fallo en la distribución de procesos, debido a que múltiples ordenadores están implicados en la misma, y todos son dependientes de la red para su comunicación, el fallo de uno o más ordenadores, o uno o más enlaces de red, puede suponer problemas para un sistema de distribución de procesos.
- Aspectos de seguridad: en la distribución de procesos hay más oportunidades de ataques no autorizados. Mientras que en un sistema centralizado todos los ordenadores y recursos están normalmente bajo el control de una administración única, en un esquema de distribución de procesos la seguridad es descentralizada. La descentralización hace difícil implementar y ejecutar políticas de seguridad; por tanto, la distribución de procesos es vulnerable a fallos de seguridad y accesos no autorizados.

En la distribución de procesos se analizan los modelos de paralelización con memoria compartida y distribuida, como se muestra en el siguiente epígrafe.

1.1.1 Análisis de paralelización con memoria compartida y memoria distribuida

Desde la perspectiva de procesamiento, una computadora con múltiples procesadores y/o con múltiples núcleos por procesador puede clasificarse en MIMD (*Multiple Instruction stream, Multiple Data stream*, Flujo de Instrucciones Múltiples y Flujo de Datos Múltiples) de memoria compartida o multiprocesador. Siguiendo la misma clasificación, un clúster (del inglés *cluster*, "grupo" o "raíz") es un MIMD de memoria distribuida o multicomputadora.

La diferencia a nivel de procesamiento y relación entre las múltiples unidades de procesamiento o CPUs de MIMD de memoria compartida y MIMD de memoria distribuida es la siguiente: en la primera clasificación la comunicación y la sincronización se realiza utilizando la memoria compartida y en la segunda clasificación se utiliza una comunicación mediante el envío de mensajes (Tinetti, y otros, 2008).

Procesamiento paralelo: multiprocesador y multicomputadora

El modelo de procesamiento en un **multiprocesador** puede considerarse como una extensión relativamente natural a partir del modelo de concurrencia que implementan los sistemas operativos. Coexisten varios procesos en el mismo sistema, que se coordinan y/o compiten para llevar a cabo procesamiento.

El modelo de procesamiento en una **multicomputadora** está esencialmente basado en memoria distribuida, donde los procesos corriendo en procesadores o, más aún, en las computadoras interconectadas deben enviar y recibir información mediante el envío de mensajes (Tinetti, y otros, 2008).

Teniendo en consideración las definiciones anteriores, en el Módulo de distribución automática de tareas a nodos de transcodificación se utiliza el modelo de procesamiento paralelo multicomputadora, debido a que se distribuyen las tareas de transcodificación de videos a varios nodos de transcodificación, donde la comunicación con los mismos se realiza mediante el envío de mensajes.

En la distribución de procesos, antes de realizar la distribución de tareas, se realiza un balanceo de carga. En el epígrafe 1.1.2 se describen los tipos de balanceo de carga que existen y en el epígrafe 1.1.3 se describe el método para el manejo de balanceo de carga realizado.

1.1.2 Tipos de balanceo de carga

Existen dos formas de balanceo de carga: estático y dinámico (William, y otros, 2009). El primero se caracteriza por un conocimiento previo de la aplicación a ejecutar, las características del sistema y la carga total de trabajo que se ha de repartir entre los procesadores (Aquiles, y otros, 2011).

El segundo es muy útil cuando el número de tareas es mayor que el número de procesadores disponibles o cuando el número de tareas es desconocido al comienzo de la aplicación. Una importante característica

del mismo es la capacidad que tiene la aplicación de adaptarse a los posibles cambios del sistema, no sólo en la carga de los procesadores sino también en posibles reconfiguraciones de los recursos del sistema (William, y otros, 2009).

El balanceo de carga estático tiene serios inconvenientes que lo sitúan en desventaja sobre el balanceo de carga dinámico. Entre ellos cabe destacar los siguientes: es muy difícil estimar de forma precisa el tiempo de ejecución de todas las partes en las que se divide un programa sin ejecutarlas y a veces los problemas necesitan un número indeterminado de pasos computacionales para alcanzar la solución.

Con el balanceo de carga dinámico todos los inconvenientes que presenta el balanceo de carga estático se tienen en cuenta. Lo anterior es posible porque la división de la carga computacional depende de las tareas que se están ejecutando y no de la estimación del tiempo que pueden tardar en ejecutarse. Aunque el balanceo de carga dinámico lleva consigo una cierta sobrecarga durante la ejecución del programa, resulta una alternativa mucho más eficiente que el balanceo de carga estática.

En el balanceo de carga dinámico, las tareas se reparten entre los procesadores durante la ejecución del programa. Dependiendo de dónde y cómo se almacenen y repartan las tareas el balanceo de carga dinámico se divide en: balanceo de carga dinámico centralizado (el nodo maestro o principal es el que tiene la colección completa de tareas a realizar y luego las tareas son enviadas a los nodos esclavos) y balanceo de carga dinámico distribuido o descentralizado (se utilizan varios nodos maestros y cada uno controla a un grupo de nodos esclavos) (William, y otros, 2009).

1.1.3 Método para el manejo del balanceo de carga y su descripción

El método propuesto para balancear la carga de trabajo a los nodos de transcodificación utiliza el balanceo de carga dinámico centralizado, debido a que existe una máquina servidora (máquina de cómputo donde se aloja la Plataforma XILEMA AGORAV) que es la encargada de realizar la distribución automática de tareas a nodos de transcodificación, haciendo uso de dicho método de balanceo de carga. La máquina servidora recopilará los siguientes parámetros de los nodos de transcodificación, con el propósito de obtener el mejor nodo para asignarle la tarea:

- Uso del CPU: es la cantidad de trabajo en por ciento que realiza la Unidad Central de Procesos (CPU) para procesar las instrucciones de un programa de computadora.
- Máxima capacidad de uso del CPU: atributo de la tabla de nodos que almacena la máxima capacidad de uso del CPU.
- Disponibilidad del CPU: Máxima capacidad de uso del CPU – Uso del CPU.
- Procesadores lógicos del CPU: consiste en que cada núcleo físico de una CPU se divida en dos

núcleos virtuales que actúan como procesadores independientes.

- Frecuencia del CPU: es el tiempo máximo en que las señales eléctricas pueden moverse en las varias bifurcaciones de los muchos circuitos de un CPU.
- Potencia del CPU: Procesadores lógicos del CPU * Frecuencia del CPU.
- Cantidad de tareas: es la cantidad de archivos multimedia que se están transcodificando en ese instante.
- Por ciento de conversión: es el por ciento de conversión general de los archivos multimedia que se están transcodificando.

Antes de asignarle la tarea a un nodo de transcodificación se evalúan los parámetros recopilados que fueron mencionados con anterioridad, verificando que:

- La Máxima capacidad de uso del CPU sea siempre mayor que el Uso del CPU recibido.
- Potencia del CPU sea la mayor.
    - En caso de tener valores iguales de Potencia del CPU, verificar que la Disponibilidad del CPU en ese instante sea la mayor.
    - En caso de tener valores iguales de Disponibilidad del CPU, verificar que la Cantidad de tareas en ese instante sea la menor.
    - En caso de tener valores iguales de Cantidad de tareas, verificar que el Porciento de conversión sea el mayor.

1.2 Definiciones asociadas al dominio del problema

1.2.1 Distribución de tareas

Para un mejor análisis del concepto de distribución de tareas es necesario el conocimiento del concepto de distribución. Plantea (Larousse, 2016) que la distribución es la división de una cosa en partes dando a cada una de ellas un destino o una posición.

En la presente investigación la distribución de tareas es: *"la asignación de la tarea de transcodificar un video a un nodo de transcodificación destinado"*.

1.2.2 Transcodificar

Se denomina transcodificar o transcodificación (del inglés *transcoding*) a la conversión directa (de digital a digital) de un formato de video a otro, para obtener la compatibilidad con otro programa o aplicación (Streaming Media, 2016). Dicha definición se acoge a la presente investigación.

1.2.3 Balanceo de carga

El balanceo de carga es un procedimiento que distribuye de una forma equitativa la carga computacional entre los recursos de cómputo disponibles, y con ellos conseguir una mejor velocidad de ejecución. Tomando como base, (William, y otros, 2009). Dicha definición se acoge a la presente investigación.

1.3 Estudio de las soluciones existentes

En la presente investigación se analizaron cuatro de las aplicaciones desarrolladas por empresas líderes en el trabajo con audiovisuales, dígase *Media Process Manager* (MPM), *Life Size Control, Media Application Server* (MAS) y Vantage.

1.3.1 Media Process Manager (MPM): Es un sistema de software que optimiza y automatiza cualquier transferencia de materiales audiovisuales. MPM posee herramientas de procesamiento de materiales audiovisuales que permiten las transferencias de archivos automáticas entre cualquier fuente y a cualquier destino. Los análisis de MPM permiten escanear todos los archivos multimedia de diferentes directorios para detectar cambios y poder inicializar un determinado proceso. Además, MPM prioriza automáticamente la carga de trabajo de procesamiento de medios de comunicación en tiempo real, de acuerdo con la urgencia y la importancia de cada flujo de trabajo (Tedial, 2015).

MPM posee características de interés para el desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación como son: la gestión de tareas en tiempo real, la distribución de tareas y automatización de tareas, aunque en la investigación realizada no se presenta un procedimiento de balanceo de carga. Dichas características aportaron una mejor comprensión en el desarrollo del módulo.

1.3.2 Life Size Control: Es una plataforma multifunción de software de gestión de video que ofrece una gestión centralizada de toda la red de videoconferencias. En la misma se procesan funciones claves de la gestión de videos de manera sencilla, con lo que se ofrece una supervisión permanente y un ahorro considerable de tiempo y recursos. Dentro de dichas funciones se encuentra *Smart Scheduler,* que permite la asignación automática de recursos. Además, dentro de sus funciones clave está gestión en tiempo real desde el panel de control a información e informes importantes (LifeSize, 2015).

Life Size Control posee características de interés para el desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación como son: la gestión de tareas en tiempo real, la distribución de tareas y automatización de tareas, aunque en la investigación realizada no se presenta un procedimiento de balanceo de carga. Dichas características aportaron una mejor comprensión en el desarrollo del módulo.

***1.3.3 Media Application Server* (MAS):** Es una plataforma que está basada en flujos de trabajo y es utilizada para la gestión de archivos de materiales audiovisuales. El MAS ofrece un centro de control común para todas sus tareas de procesamiento, como las transferencias, transcodificación, control de calidad y gestión de archivos.

La plataforma tiene integrada varias aplicaciones que funcionan como un todo. Entre las aplicaciones está *Workflow System* (WSF), que brinda la automatización mediante carpetas de inspección y una Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés) que incluye transferencias FTP (siglas en inglés de *File Transfer Protocol*, Protocolo de Transferencia de Archivos) automáticas. Además, brinda monitorizaciones remotas, alertas y notificaciones de tareas. WFS puede gestionar videos mediante la integración de nodos adicionales de terceros y procesamiento armónico. Otra de las aplicaciones del MAS es *Distribute Service*, que permite distribuir servicios de videos (Harmonic, 2015).

MAS Control posee características de interés para el desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación como son: la gestión de tareas en tiempo real, la distribución de tareas y automatización de tareas, aunque en la investigación realizada no se presenta un procedimiento de balanceo de carga. Dichas características permitieron una mejor comprensión en el desarrollo del módulo.

1.3.4 Vantage: Es una plataforma que permite el diseño y automatización de flujos de trabajo de video y puede integrarse con sistemas de terceros a nivel de API sin problemas y prácticamente con cualquier dispositivo de red. Permite crear flujos de trabajo adaptables que automáticamente toman decisiones y gestionan el contenido audiovisual. Además, mediante la integración de análisis, transcodificación, exploración de directorios, movimiento de archivos y el diseño de flujo de trabajo, puede crear inteligencia automatizada en los flujos de trabajo de video. *Vantage* realiza un balanceo de carga de las tareas y luego distribuye a cada una en un proceso independiente (Telestream, 2015).

Vantage posee como características de interés para el desarrollo del Módulo de distribución automática de tareas a nodos de transcodificación como son: la distribución de tareas y automatización de tareas. Dichas características aportaron una mejor comprensión en el desarrollo del módulo.

1.3.5 Resultados del análisis de las soluciones existentes

Luego del análisis realizado a las soluciones existentes vistas anteriormente se concluye que las mismas no constituyen una solución factible para el problema planteado, debido a que poseen un sistema de código privativo, por lo que presentan imposibilidad de copias, de modificación y redistribución. El costo de dichas aplicaciones es elevado, el soporte es exclusivo del propietario, y además el usuario que adquiere el software depende de la empresa propietaria. La integración con lodos nodos de transcodificación y la

plataforma XILEMA AGORAV se hace costosa en cuanto a recursos monetarios y tiempo, teniendo en cuenta que habría que modificar los mismos para adaptarlos al funcionamiento de dichos sistemas.

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

Tabla 1. Resumen del estudio de sistemas similares.

Características de interés / Sistemas analizados	Distribución de tareas	Gestión de tareas en tiempo real	Automatización de tareas
Life Size Control	✓	✓	✓
Media Application Server	✓	✓	✓
Media Process Manager	✓	✓	✓
Vantage	✓		✓

Tomando como base lo planteado con anterioridad, será necesario desarrollar un Módulo para la distribución automática de tareas a nodos de transcodificación, que permita lograr en menor tiempo la publicación de los materiales audiovisuales en XILEMA AGORAV, haciendo uso de tecnologías libres.

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

Teniendo en cuenta el ambiente de desarrollo del proyecto AGORAV, se utilizan la metodología, herramientas, tecnologías y lenguajes de programación definidas en el mismo. Su empleo permite desarrollar un módulo acorde con las políticas establecidas en el proyecto, lo que proporciona un mejor acoplamiento con la Plataforma XILEMA AGORAV.

1.4.1 Metodología de desarrollo de software AUP-UCI

Las metodologías para el desarrollo del software proponen un proceso disciplinado sobre el desarrollo de un sistema. El uso de las mismas proporciona un proceso más predecible en la confección de un producto y un aumento de la calidad del software en todas las fases del desarrollo del mismo (Pressman, 2002).

Para el desarrollo del módulo se utilizó como metodología AUP (Proceso Unificado Ágil, por sus siglas en inglés) en su variante UCI. La misma describe de manera simple la forma de desarrollar el módulo con el uso de técnicas ágiles, como el modelado ágil. El uso de dicha metodología permite una descripción concisa

utilizando poca documentación y brinda la posibilidad de utilizar cualquier conjunto de herramientas CASE (Ingeniería de Software Asistida por Computadora).

Debido a que en el proyecto AGORAV se utiliza como escenario “Proyectos que modelen el negocio con Modelo Conceptual solo pueden modelar el sistema con Casos de Uso del Sistema”, para el desarrollo del módulo se seleccionó el escenario antes mencionado. Con su uso se obtienen artefactos ingenieriles que el proyecto necesita, útiles para futuras labores de mantenimiento al módulo.

1.4.2 Unified Modeling Language (UML)

UML se define como un “Lenguaje cuyo vocabulario y reglas se centran en la representación conceptual y física de un sistema” (UML, 2016). El mismo permite visualizar, especificar, documentar y modelar los artefactos del módulo que el proyecto necesita.
También se opta por la utilización de UML porque la metodología utilizada lo exige.

1.4.3 Herramienta CASE para modelado

CASE comprende un amplio abanico de diferentes tipos de programas que se utilizan para ayudar a las actividades del proceso de software, como el análisis de requisitos, el modelado de sistemas, la depuración y las pruebas (Somerville, 2009).

Visual Paradigm es una herramienta CASE profesional, que utiliza el lenguaje de modelado UML (Visual Paradigm, 2015). Permite construir todos los diagramas del módulo brindando un claro entendimiento de los artefactos generados, además de obtener a partir de un diagrama otro que guarde relación con el mismo.

1.4.4 Lenguajes de programación

JavaScript: JavaScript es un lenguaje de programación interpretado. Su potencia está dada por el soporte que tiene en los distintos navegadores. Además, evita la sobrecarga del servidor debido a que es un lenguaje del lado del cliente (Sanchez, 2001). Dicho lenguaje aporta como ventajas significativas para el desarrollo del módulo que es liviano, utiliza poca memoria, es fácil de integrar y permite animar los elementos de la página web.

CSS: las Hojas de Estilo en Cascada (*Cascading Style Sheets*, por sus siglas en inglés CSS), se utilizan para dar estilo a documentos HTML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML (W3C, 2016). CSS permite controlar el estilo y el formato de las páginas web del módulo.

HTML5: HTML (*HyperText Markup Language,* por sus siglas en inglés) es el lenguaje para describir la estructura de las páginas web. HTML5 es la versión actual de HTML que incluye nuevos elementos como son (W3C, 2016):

- Los elementos semánticos como: <header>, <footer>, <article> y <section>.
- Los elementos multimedia como: <audio> y <video>.

El uso del lenguaje **HTML5** permite estructurar las páginas web del módulo que son interpretadas por el navegador y como resultado poder mostrar textos, tabla, botones, campos de textos y enlaces.

PHP5: PHP (*PHP Hypertext Preprocessor*) es un lenguaje interpretado y multiplataforma de alto nivel incrustado en páginas HTML y ejecutado en el servidor (Converse, y otros, 2004). PHP5 es una versión reciente la cual incluye el manejo de excepciones y posibilita el uso de técnicas de programación orientada a objetos. Existen librerías escritas en leguaje PHP que facilitan interfaces para una gran cantidad de sistemas de base de datos diferentes y dispone de conexión propia para estos sistemas. Entre sus ventajas se encuentra la facilidad de aprendizaje y portabilidad.

También se opta por PHP5 porque el módulo fue implementado con el Sistema Gestor de Contenidos Drupal, el cual utiliza el lenguaje de programación PHP.

1.4.5 Entorno de Desarrollo Integrado

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

PhpStorm es un IDE ligero, de fácil instalación y multiplataforma (JetBrains, 2014). Proporciona un editor para PHP, HTML y JavaScript, brindando análisis de código en la marcha y refactorizaciones automáticas. Además brinda un completamiento inteligente de código y detecta código duplicado.

1.4.6 Servidor web Apache2

Apache2 es el servidor web más utilizado en los sistemas Linux (Apache, 2016). El mismo se utiliza para servir páginas web solicitadas por los clientes web. Es multiplataforma, rápido, continuamente actualizado y de código abierto. Apache2 es usado en combinación con el motor de bases de datos PostgreSQL y el lenguaje PHP. Esta unión permite la interacción con la base de datos del módulo.

1.4.7 Sistema de Gestión de Contenidos Drupal

La Plataforma XILEMA AGORAV fue construida con el CMS (*Content Management System*) Drupal y la utilización de este facilitará la integración del módulo a desarrollar con dicha plataforma.

Un CMS es un software que permite crear una estructura base para la creación y administración de contenidos, principalmente de páginas web. Generalmente un CMS es una aplicación con una base de datos asociada en la que se almacenan los contenidos, separados de los estilos o diseño. El CMS controla también quién puede editar y visualizar los contenidos, convirtiéndose en una herramienta de gestión integral para la publicación de sitios web (Rodríguez, 2012).

Drupal en su versión 7.26 es un sistema de gestión de contenidos modular y muy configurable. Es un programa de código abierto con licencia GNU/GPL y escrito en PHP. Destaca por la calidad de su código, el respeto de los estándares web, la usabilidad y flexibilidad. Drupal puede ser instalado con diferentes gestores de bases de datos, como MySQL, SQLite, PostgreSQL y Oracle. La agregación de una gran cantidad de módulos existentes, hace que sea adecuado para la realización de muchos tipos de sitios web (Drupal Hispano, 2016).

En el desarrollo del módulo se hizo uso de las siguientes características del CMS Drupal:

- Fácilmente extensible a través de módulos que permiten adaptar rápidamente la aplicación a necesidades específicas o cambiantes.
- Administración vía web, registro e informes de errores.
- Sistema de Caché que elimina consultas a la base de datos incrementando el rendimiento y reduciendo la carga del servidor.

1.4.8 Sistema Gestor de Bases de Datos PostgreSQL

Se conoce como Sistema Gestor de Bases de Datos (SGBD) al conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos (Bertino, y otros, 1995).

PostgreSQL es un potente gestor de bases de datos de código abierto que soporta el esquema objeto-relacional. Cuenta con una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad e integridad de datos. Es multiplataforma e incluye almacenamiento de varios tipos de datos (PostgreSQL, 2016). El uso de PostgreSQL como SGBD en su versión 9.3 brinda ventajas como: fácil de administración, estable y de gran flexibilidad debido a que puede funcionar sobre la mayoría de los sistemas Unix.

1.4.9 PgAdmin 3

PgAdmin 3 es una herramienta de código abierto que permite administrar sistemas gestores de bases de datos PostgreSQL (ArPUG, 2016). El mismo posee una interfaz gráfica que soporta todas las características del SGBD PostgreSQL y hace simple la administración de dicho SGBD. Además, es multiplataforma, y su licencia es fácil de adquirir.

1.4.10 Protocolo WebSockets

Para la comunicación se hará uso del protocolo *WebSockets,* que es una tecnología que proporciona un canal de comunicación bidireccional y *full-duplex*[3] que opera a través de un *socket*[4] sobre el Protocolo de Control de Transmisión (TCP por sus siglas en inglés) (Websocket, 2016). La utilización de este protocolo brinda una manera sencilla de poder establecer la comunicación entre el cliente y servidor y reduce el tráfico innecesario sobre la red.

1.5 Conclusiones parciales

En este capítulo se concluye que:

- Mediante el estudio de sistemas que trabajan con audiovisuales se obtuvo como resultado que los mismos no constituyen una solución para el problema planteado, por lo que se requiere del desarrollo de una nueva solución. Sin embargo, se identificaron características comunes en dichos sistemas que aportaron una mejor comprensión en el desarrollo del módulo como son, la distribución de tareas, gestión de tareas en tiempo real y automatización de tareas
- Para guiar el proceso de desarrollo del módulo se seleccionó la metodología AUP variación UCI, haciendo uso del lenguaje de modelado UML, con el apoyo de la herramienta CASE *Visual Paradigm* para UML. Esto posibilitó la obtención de una solución bien documentada, ofreciendo facilidad para el desarrollo de futuras versiones del módulo.
- La elección de Drupal como Gestor de Contenidos, *WebSockets* como protocolo de comunicación, CSS, PHP5 y HTML5 ayudó a obtener un módulo desarrollado acorde a los sistemas web actuales.

---

[3] Una comunicación es *full-duplex* si puede enviar y recibir datos al mismo tiempo, lo que puede ocurrir entre dos nodos conectados directamente (Halsall, 2005).

[4] Es un "canal de comunicación" entre dos programas que se ejecutan sobre ordenadores distintos o incluso en el mismo ordenador. Está conformado por un número de identificación compuesto por la dirección IP (Protocolo de Internet) y el número de puerto TCP (Halsall, 2005).

Capítulo 2: Análisis y diseño

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

2.1 Modelo de dominio

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan los eventos que suceden en el entorno en el que trabaja el sistema (Sommerville, 2005). Para la presente investigación se plantea el uso de un modelo de dominio ya que no se tiene una estructura definida de los procesos del negocio y no se cuenta con un manual de procedimientos. A continuación, se describe el modelo de dominio del módulo.

2.1.1 Descripción general del modelo de dominio

La Plataforma XILEMA AGORAV contiene al módulo Gestor de contenidos, donde se adicionan archivos multimedia a dicha plataforma. Luego la plataforma verifica el formato de los archivos y si el formato es correcto (.webm) se almacenan directamente en el servidor de medias. En caso contrario se envían a un único nodo encargado de transcodificarlos y luego de concluida la operación anterior se almacenan en el servidor de medias. Para realizar la publicación de los archivos multimedia la plataforma los extrae del servidor de medias y los copia en el servidor streaming.

2.1.2 Descripción de las clases del modelo de dominio

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

- **Plataforma AGORAV:** plataforma que gestiona y publica contenidos audiovisuales.
- **Módulo Gestor de contenidos:** es el encargado de agregar a la plataforma todo el contenido que se gestiona.
- **Archivo multimedia:** fichero multimedia de audio y video que se almacena en el servidor de medias para ser publicado.
- **Servidor de medias:** servidor donde se guardan los archivos multimedia con el formato correcto.

- **Conversor o nodo de transcodificación:** subsistema que atiende peticiones a través del protocolo *websocket* y que realiza la función de transcodificación de los materiales audiovisuales.
- **Servidor *streaming*:** servidor donde se almacenan los archivos multimedia para ser publicados en la red.

2.2 Diagrama de clases del modelo de dominio

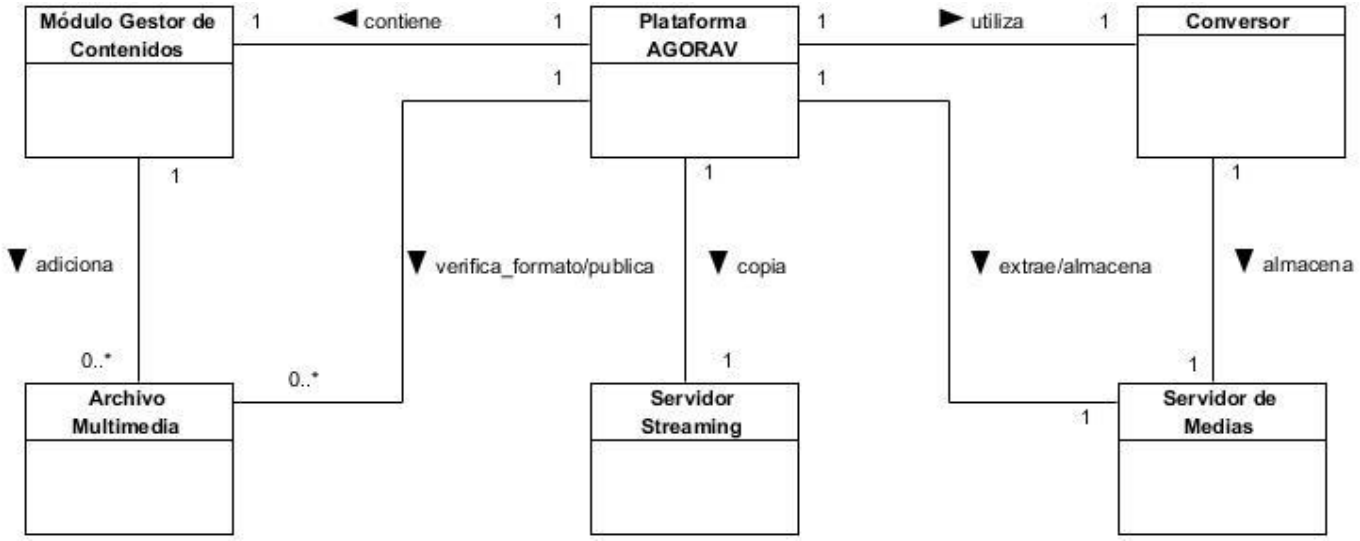


Figura 2. Representación del modelo de dominio.

2.3 Especificación de los requisitos

Un requisito es una condición o capacidad que debe cumplir o poseer un producto o componente de producto para satisfacer un contrato, un estándar, una especificación u otros documentos impuestos formalmente (Ramos, 2013).

2.3.1 Requisitos funcionales:

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir (Jacobson, y otros, 2000). Los requisitos funcionales identificados para el desarrollo del módulo son los siguientes:

RF1. Insertar nodo de transcodificación

Descripción: Permite adicionar un nodo de transcodificación en el módulo.

RF2. Listar nodos de transcodificación

Descripción: Permite mostrar una lista con los nodos de transcodificación existentes en el módulo.

RF3. Modificar nodo de transcodificación

Descripción: Permite editar los datos asociados a un nodo de transcodificación seleccionado.

RF4. Eliminar nodo de transcodificación

Descripción: Permite eliminar un nodo de transcodificación.

RF5. Buscar nodo de transcodificación

Descripción: Permite buscar un nodo de transcodificación.

RF6. Mostrar cantidad de tareas de cada nodo de transcodificación

Descripción: Permite mostrar la cantidad de tareas que está procesándose en cada nodo de transcodificación.

RF7. Mostrar por ciento del uso del CPU de cada nodo de transcodificación

Descripción: Permite mostrar el por ciento del uso del CPU de cada nodo de transcodificación.

RF8. Distribuir tarea

Descripción: Permite distribuir automáticamente la tarea de transcodificar un video a un nodo de transcodificación seleccionado para realizarla, teniendo en cuenta los siguientes parámetros principales:

- Uso del CPU.
- Disponibilidad del CPU.
- Potencia del CPU.
- Cantidad de tareas.
- Porciento de conversión.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Los requisitos no funcionales son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto (Jacobson, y otros, 2000).

Usabilidad

RNF1. El módulo podrá ser usado por personas con conocimientos básicos en el manejo de

computadoras.

RNF2. El módulo cuenta con una interfaz amigable e intuitiva que sirve de ayuda al usuario para una mejor interacción con las funcionalidades que brinda.

Requisitos del Software

RNF3. El módulo podrá ser utilizado en los navegadores Google Chrome versión 4, Safari versión 5, Mozilla Firefox versión 8 y en versiones superiores de ellos.

Requisitos del Hardware

RNF4. La máquina cliente debe poseer un Procesador Pentium 4 o superior y 512 MB de Acceso Aleatorio (RAM) como mínimo.

RNF5. La máquina servidora deben poseer un Procesador Quad Core o superior y una memoria RAM de 4GB o superior.

2.4 Descripción del módulo

Un Caso de Uso (CU) especifica una secuencia de acciones que el sistema puede llevar a cabo. El diagrama de CU se utiliza para ilustrar los requisitos del sistema. Además, es una representación gráfica de los procesos y su interacción con los actores (Jacobson, y otros, 2000).

2.4.1 Definición de los actores

Se le llama Actor a toda entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad. Esto incluye a los operadores humanos pero también incluye a todos los sistemas externos (Kruchten, 2003). A continuación, se describen los actores de la aplicación a desarrollar.

Tabla 2. Actores que intervienen en el problema.

Actor	Descripción
Gestor de contenidos	Usuario autenticado con permisos para Gestionar nodo de transcodificación y mostrar el estado de los nodos de transcodificación.
AGORAV	Inicia el proceso de distribución automática de tareas a nodos de transcodificación.

2.5 Diagrama de Casos de Uso del sistema

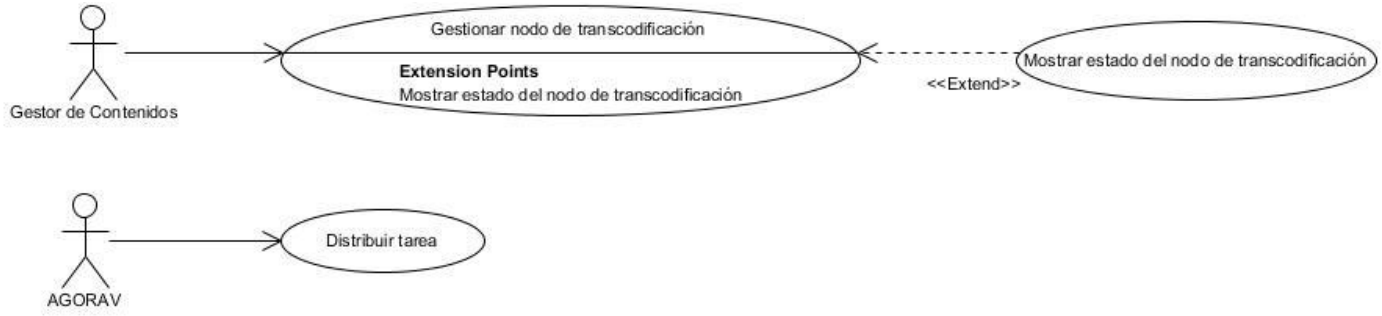


Figura 3. Diagrama de Casos de Uso del sistema.

2.6 Descripción de Casos de Uso

Tabla 3. Descripción del Caso de Uso: Gestionar nodo.

CU	Gestionar nodo.	
Objetivo	El CU se lleva a cabo para gestionar las funcionalidades referentes a los nodos de transcodificación.	
Actores	"Gestor de contenidos" (Inicia) Gestionar nodo.	
Resumen	El CU se inicia cuando el "Gestor de contenidos" desea insertar, modificar, eliminar, listar o buscar nodos de transcodificación. El CU termina cuando haya sido ejecutada alguna de las funcionalidades anteriormente especificadas.	
Complejidad	Alta.	
Prioridad	Media.	
Precondiciones	El usuario debe estar autenticado con el rol "Gestor de contenidos".	
Postcondiciones	Se inserta, lista, modifica, elimina o busca un nodo de transcodificación en el módulo.	
Referencias	RF1, RF2, RF3, RF4, RF5.	
Flujo de eventos		
Flujo básico: Gestionar nodo.		
	Actor	**Sistema**
1.	Selecciona la opción "Gestionar nodo".	
2.		Muestra una vista con un listado de los nodos registrados en el sistema.
3.	Puede realizar una de las opciones que se especifican a continuación: - Listar nodos. Ver Sección 1 "*Listar nodos*".	

	- Insertar nodo. Ver Sección 2: "*Insertar nodo*". - Modificar nodo. Ver Sección 3: "*Modificar nodo*". - Eliminar nodo. Ver Sección 4: "*Eliminar nodo*". - Buscar nodo. Ver Sección 5: "*Buscar nodo*".	
4.		El CU termina cuando haya sido ejecutada alguna de las funcionalidades anteriormente especificadas.
Sección 1: "Listar nodos".		
Flujo básico: Listar nodos.		
	Actor	**Sistema**
1.	Selecciona la opción "Listar nodos".	
2.		Verifica que exista algún nodo de transcodificación registrado.
3.		Muestra una vista con los nodos de transcodificación.
4.		Termina el CU.
Flujos alternos		
2.1: No hay ningún nodo de transcodificación registrado en el sistema.		
	Actor	**Sistema**
2.1.1		Muestra un mensaje "*La tabla de nodos de transcodificación se encuentra vacía".*
Sección 2: "Insertar nodo".		
Flujo básico: Insertar nodo.		
	Actor	**Sistema**
1.	Selecciona la opción "Insertar nodo"	
2.		Muestra un formulario con los datos del nodo de transcodificación: - Nombre (campo obligatorio) - IP (campo obligatorio) - Puerto (campo obligatorio) - Máxima capacidad de uso del CPU (campo obligatorio)

		- Procesadores lógicos del CPU (campo obligatorio) - Frecuencia del CPU (campo obligatorio)
3.	Llena los campos y presiona el botón "*Agregar*".	
4.		Verifica que los datos sean correctos.
5.		Verifica que no existan campos vacíos.
6.		Guarda los datos de la nueva acción.
7.		Termina el CU.
Flujos alternos		
5.1: El usuario deja campos en blanco.		
	Actor	**Sistema**
5.1.1		Muestra un mensaje "*El campo es obligatorio*."
5.1.2		Vuelve a ejecutar el paso 2.
4.1: El usuario ingresa un nombre o IP existente en la base de datos.		
	Actor	**Sistema**
4.1.1		Muestra un mensaje "*Existe otro nodo con ese valor*."
4.1.2		Vuelve a ejecutar el paso 2.
4.2 El usuario llena el campo "Dirección IP" con un formato incorrecto.		
	Actor	**Sistema**
4.2.1		Muestra un mensaje *"Entre un IP válido".*
4.2.2		Vuelve a ejecutar el paso 2.
4.3 El usuario llena el campo "Máxima capacidad de uso del CPU" con datos fuera del rango entre 0 y 100.		
	Actor	**Sistema**
4.3.1		Muestra un mensaje *"Entre una Máxima capacidad de uso del CPU válida".*
4.3.2		Vuelve a ejecutar el paso 2.
4.4 El usuario llena el campo "Puerto" con datos fuera del rango entre 1024 y 65535.		
	Actor	**Sistema**
4.4.1		Muestra un mensaje *"Entre un Puerto válido".*

4.4.2		Vuelve a ejecutar el paso 2.
4.5 El usuario llena el campo “Procesadores lógicos del CPU” con datos fuera del rango entre 0 y 100.		
	Actor	**Sistema**
4.5.1		Muestra un mensaje *“Entre una cantidad de Procesadores lógicos del CPU válido”.*
4.5.2		Vuelve a ejecutar el paso 2.
4.6 El usuario llena el campo “Frecuencia del CPU” con datos no válidos.		
	Actor	**Sistema**
4.6.1		Muestra un mensaje *“Entre una Frecuencia del CPU válida”.*
4.6.2		Vuelve a ejecutar el paso 2.
Sección 3: “Modificar nodo”.		
Flujo básico: Modificar nodo.		
	Actor	**Sistema**
1.	Selecciona la opción Listar nodos.	
2.		Muestra una vista con los nodos de transcodificación.
3.	Selecciona el nodo de transcodificación a modificar y se presiona el botón “Modificar”.	
4.		Muestra un formulario con los campos del nodo a modificar: - Nombre (campo obligatorio) - IP (campo obligatorio) - Puerto (campo obligatorio) - Máxima capacidad de uso del CPU (campo obligatorio) - Procesadores lógicos del CPU (campo obligatorio) - Frecuencia del CPU (campo obligatorio)
5.	Llena los campos y presiona el botón “Actualizar”.	
6.		Verifica que los datos sean correctos.
7.		Verifica que no existan campos vacíos.

8.		Actualiza los datos del nodo de transcodificación.
9.		Termina el CU.
Flujos alternos		
7.1: El usuario deja campos en blanco.		
	Actor	**Sistema**
7.1.1		Muestra un mensaje "*El campo es obligatorio*."
7.1.2		Vuelve a ejecutar el paso 2.
6.1: El usuario ingresa un nombre o IP existente en la base de datos.		
	Actor	**Sistema**
6.1.1		Muestra un mensaje "*Existe otro nodo con ese valor*."
6.1.2		Vuelve a ejecutar el paso 2.
6.2 El usuario llena el campo "Dirección IP" con un formato incorrecto.		
	Actor	**Sistema**
6.2.1		Muestra un mensaje *"Entre un IP válido".*
6.2.2		Vuelve a ejecutar el paso 2.
6.3 El usuario llena el campo "Máxima capacidad de uso del CPU" con datos fuera del rango entre 0 y 100.		
	Actor	**Sistema**
6.3.1		Muestra un mensaje *"Entre una Máxima capacidad de uso del CPU válida".*
6.3.2		Vuelve a ejecutar el paso 2.
6.4 El usuario llena el campo "Puerto" con datos fuera del rango entre 1024 y 65535.		
	Actor	**Sistema**
6.4.1		Muestra un mensaje *"Entre un Puerto válido".*
6.4.2		Vuelve a ejecutar el paso 2.
6.5 El usuario llena el campo "Procesadores lógicos del CPU" con datos fuera del rango entre 0 y 100.		
	Actor	**Sistema**
6.5.1		Muestra un mensaje *"Entre una cantidad de Procesadores lógicos del CPU válida".*
6.5.2		Vuelve a ejecutar el paso 2.
6.6 El usuario llena el campo "Frecuencia del CPU" con datos no válidos.		

	Actor	Sistema
6.6.1		Muestra un mensaje *"Entre una Frecuencia del CPU válida".*
6.6.2		Vuelve a ejecutar el paso 2.
Sección 4: "Eliminar nodo".		
Flujo básico: Eliminar nodo.		
	Actor	**Sistema**
1.	Selecciona la opción "Listar nodo".	
2.		Muestra una vista con los nodos de transcodificación.
3.	Selecciona el nodo de transcodificación a eliminar y presiona el botón "Eliminar".	
4.		Muestra un mensaje de confirmación "¿Está seguro que desea eliminar el nodo?" con las opciones "Aceptar" y "Cancelar".
5.	Presiona el botón "Aceptar"	
6.		Elimina el nodo de transcodificación. Muestra una vista de los nodos de transcodificación y un mensaje de información "El nodo ha sido eliminado".
7.		Termina el CU.
Flujos Alternos		
5.1 El usuario presiona el botón "Cancelar"		
	Actor	**Sistema**
5.1.1		Vuelve al paso 2 de la Sección 4.
Sección 5: "Buscar nodo".		
Flujo básico: Buscar nodo.		
	Actor	**Sistema**
1.		Muestra un formulario con el criterio de búsqueda.
2.	Se llena el criterio de búsqueda y se presiona el botón "Buscar".	
3.		Muestra una vista con los resultados.
4.		Termina el CU.
Flujos Alternos		

3.1: Se inserta el criterio de búsqueda en la opción buscar y no se encuentran resultados.		
	Actor	**Sistema**
3.1.1		Muestra el siguiente mensaje: "*No hay resultados que mostrar*".
3.1.2		Vuelve a ejecutar el paso 1 de la Sección 5.
Relaciones	CU extendido: Monitorizar Procesos	
Requisitos no funcionales	RNF 1, RNF 2, RNF 3, RNF 4.	
Asuntos pendientes	No	

2.7 Patrón arquitectónico

La arquitectura de software permite representar de forma concreta la estructura y funcionamiento interno de un sistema. La misma abarca los componentes de software, sus propiedades y relaciones (Clements, y otros, 2003). Al diseñar una arquitectura de software se debe crear y representar componentes que interactúen entre ellos, además de organizarlos de forma tal que se logren los requisitos establecidos. Se puede iniciar con patrones de soluciones ya probados, dichas soluciones se conocen como patrones arquitectónicos. Para el desarrollo del módulo se decide hacer uso del Patrón Arquitectónico, Modelo – Vista - Controlador (MVC) debido a que es la arquitectura empleada en la Plataforma XILEMA AGORAV.

En el patrón arquitectónico MVC se separa la lógica de negocio de la interfaz de usuario y se aumenta la posibilidad de evolución por separado de ambos aspectos. Mediante el uso de MVC es posible incrementar la reutilización de las aplicaciones debido a lo anteriormente mencionado.

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

2.8 Patrones de diseño de software utilizados

Los patrones de diseño son soluciones a problemas comunes en el desarrollo de software (Microsoft, 2015).

2.8.1 Patrones para la asignación de responsabilidades (GRASP)

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

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

El patrón **Experto (*Expert*)** indica que la clase que cuenta con la información necesaria para cumplir la responsabilidad es la responsable de manejar la información. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento) (Larman, 2003). La utilización de dicho patrón se ve aplicada en clases como: *distribucion_aut_tareas_controladora, distribucion_aut_tareas_controladora_insertar* y *distribucion_aut_tareas_controladora_modificar*, debido que cada una de ellas es responsable de manejar su información.

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

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

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

2.8.2 Patrones GoF

La línea base en el tema de patrones de diseño la impone el catálogo "*Design Patterns: Elements of Reusable Object-Oriented Software*". En el libro se presenta un conjunto de 23 patrones de diseño identificados a partir del estudio y la experiencia del grupo llamado Banda de los Cuatro o *The Gang of Four (GOF)* quienes se dedicaron a analizar los problemas recurrentes en el desarrollo de software y realizaron una clasificación y agrupación a partir de dos criterios, su propósito y alcance. Las categorías definidas son

(Guerrero, y otros, 2013):

- **Patrones creacionales:** inicialización y configuración de objetos.
- **Patrones estructurales:** separan la interfaz de la implementación. Se ocupan de cómo las clases y los objetos se agrupan para formar estructuras más grandes.
- **Patrones de comportamiento:** más que describir objetos o clases, describen la comunicación entre ellos.

Teniendo en cuenta que Drupal implementa al crear un módulo los patrones GOF, en el desarrollo del módulo se hace uso de las categorías siguientes:

2.8.2.1 Patrón estructural

Puente (Bridge): Su propósito es separar una abstracción de su implementación para permitir que ambos puedan variar independientemente. Dicho patrón es usado para evitar que posibles cambios en la implementación de una abstracción afecten a los clientes o cuando se quiere ocultar la implementación de la vista de los clientes. La conexión de Drupal con la base de datos es una abstracción que le permite al cliente ignorar el tipo de almacenamiento que utiliza, e incluso migrar de gestor de base de datos sin tener que modificar su código.

2.8.2.2 Patrón de comportamiento

Acción (Command): Patrón de comportamiento a nivel de objetos, su principio de funcionamiento es encapsular en un objeto la acción que satisface una petición (Larman, 2003). La utilización de este patrón se evidencia en la implementación de los ganchos (*hook*) necesarios para la comunicación del módulo con el núcleo de Drupal.

2.9 Diagramas de secuencia

Con el objetivo de describir cómo interactúan los objetos del diseño se realiza el diagrama de secuencia, el cual representa objetos a lo largo del eje X, y mensajes ordenados a lo largo del eje Y (Larman, 2003). A continuación, se presentan los diagramas de secuencia de cada uno de los flujos del CU Gestionar nodo.

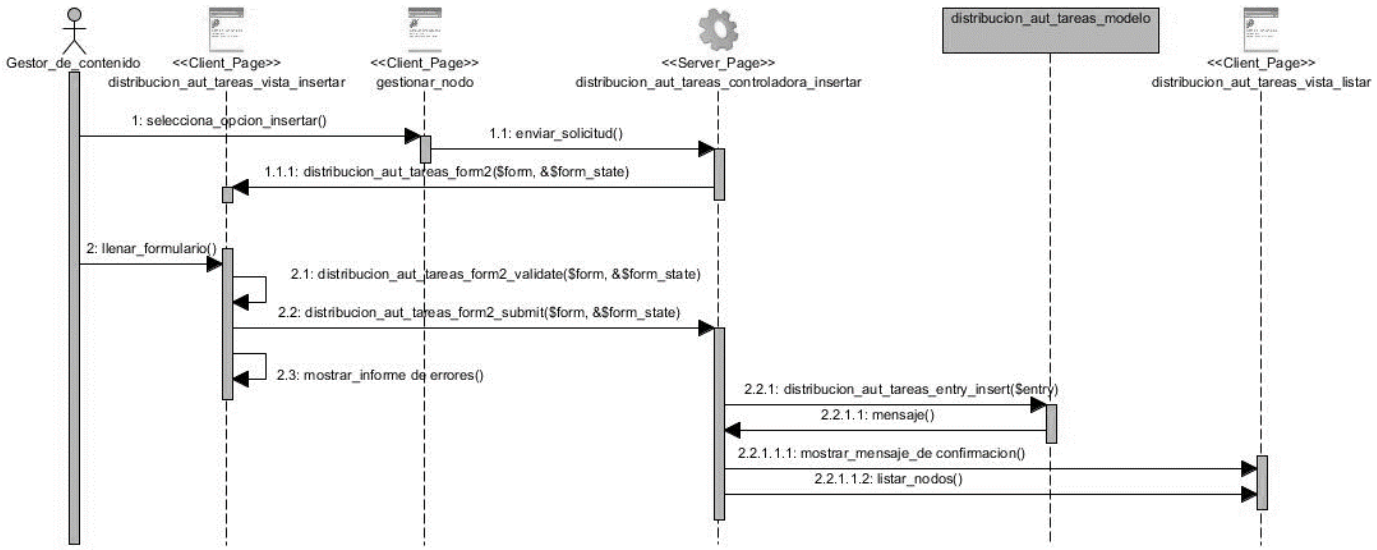


Figura 4. Diagrama de secuencia. Insertar nodo.

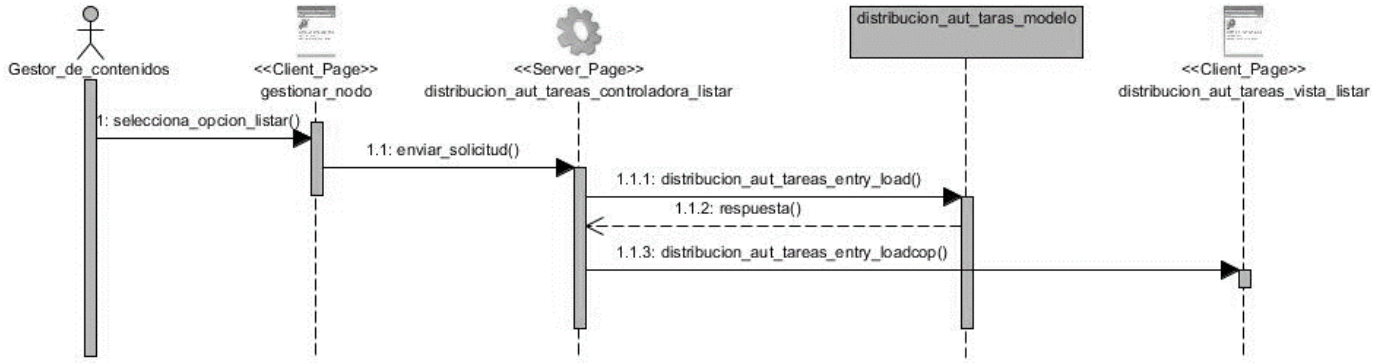


Figura 5. Diagrama de secuencia. Listar nodos.

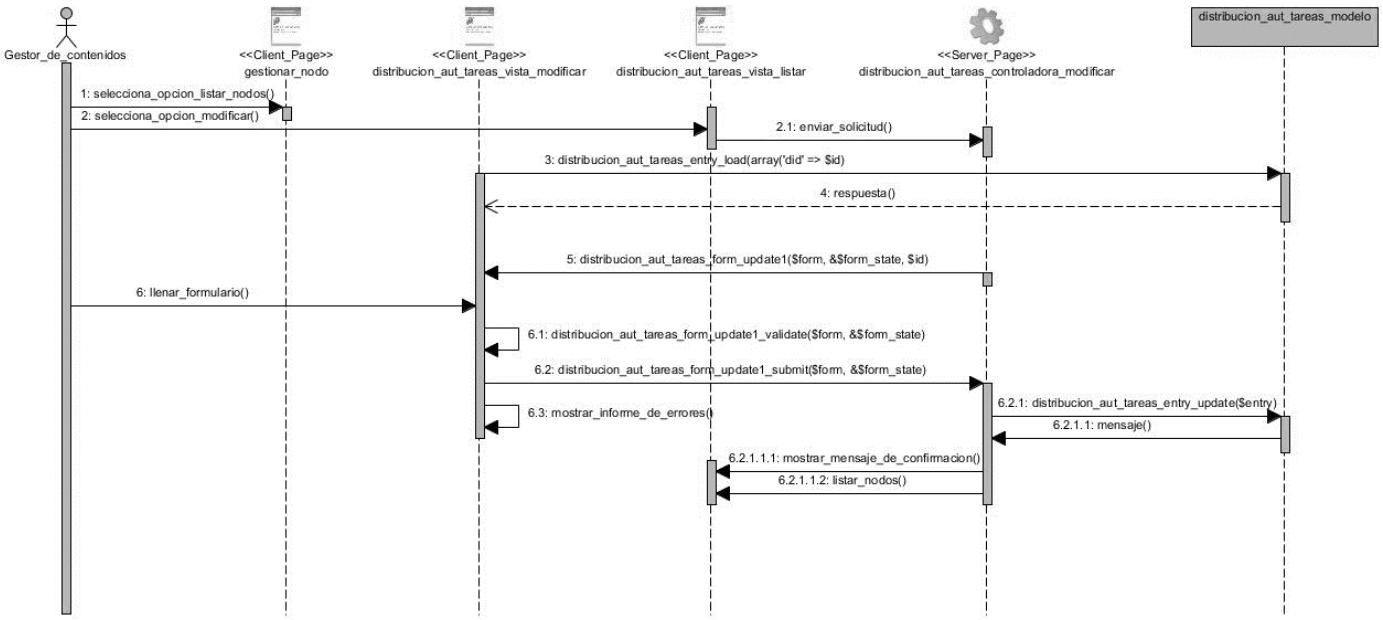


Figura 6. Diagrama de secuencia. Modificar nodo.

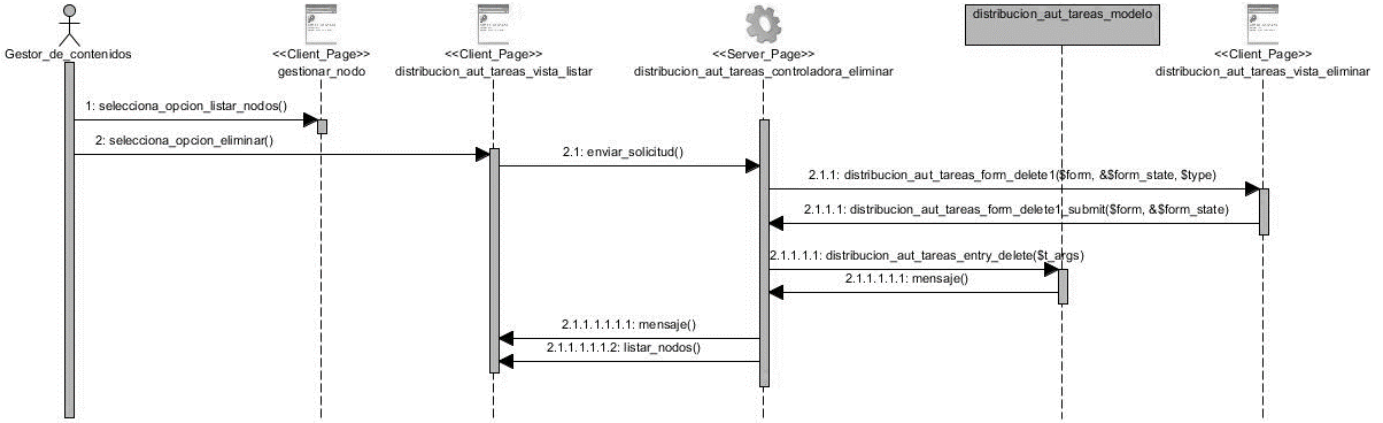


Figura 7. Diagrama de secuencia. Eliminar nodo.

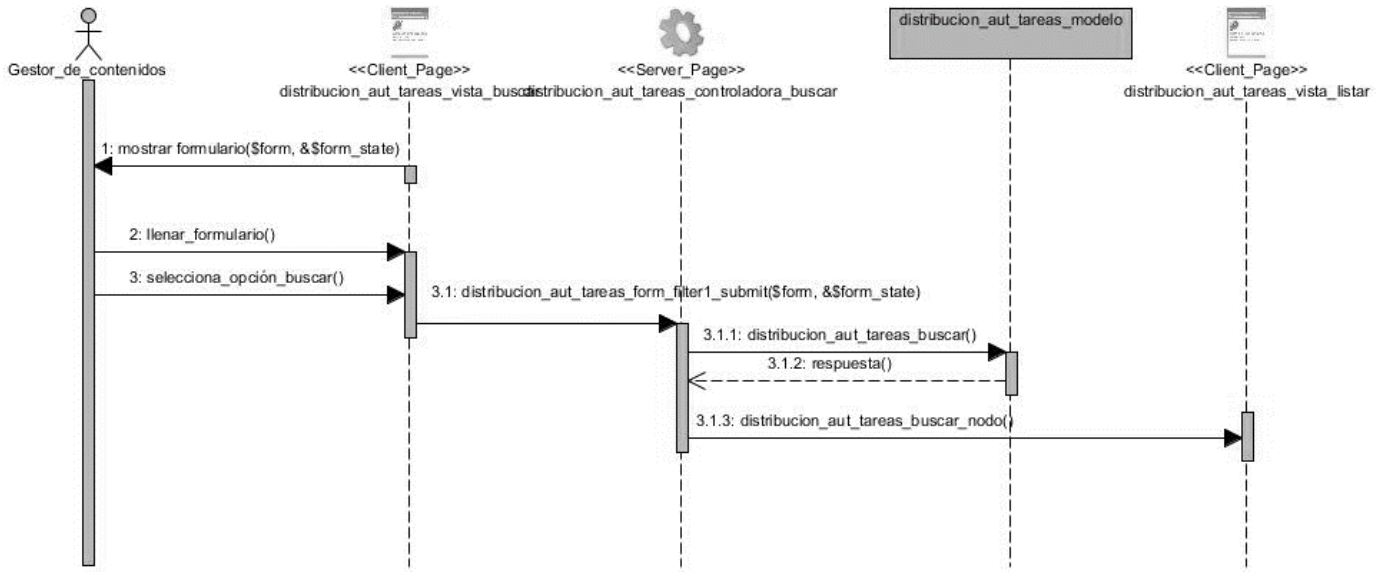


Figura 8. Diagrama de secuencia. Buscar nodo.

2.10 Diagrama de clases del diseño

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

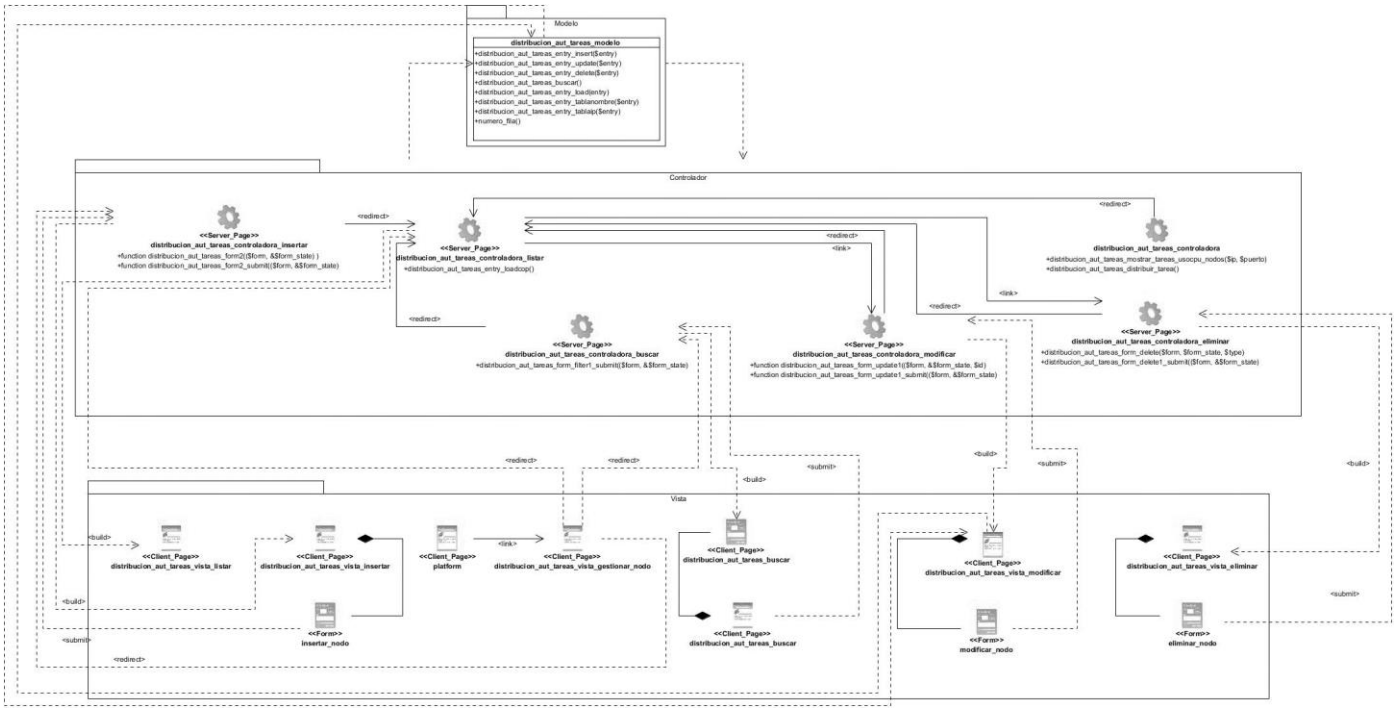


Figura 9. Diagrama de clases de diseño del Caso de Uso Gestionar nodo.

2.11 Conclusiones parciales

En este capítulo se concluye que:

- La descripción de CU del sistema, los diagramas de secuencia y los diagramas de clases de diseño sirvieron de guía para la implementación del módulo y para su documentación ingenieril, lo cual facilitará futuras labores de mantenimiento.
- La elección del patrón arquitectónico MVC y la aplicación de patrones de diseño, evidenciaron la utilización de buenas prácticas durante el desarrollo del módulo, lo cual repercute positivamente en la calidad final del mismo.

Capítulo 3. Implementación y Pruebas

En el presente capítulo se tienen en cuenta todos los aspectos del diseño del módulo con el fin de llevar a cabo el desarrollo de los flujos de trabajo de implementación y prueba. Se muestra la organización del módulo mediante el diagrama de componentes el cual representa la vista estática del módulo, y mediante el diagrama de despliegue se representa la situación física de los distintos componentes lógicos desarrollados. Se describe el estándar de codificación empleado. Por último se define el proceso de pruebas con el propósito de descubrir y corregir la mayor cantidad de errores en los requisitos implementados.

3.1 Estándares de codificación

Los estándares de codificación son muy importantes en cualquier aplicación, ya que permiten que el código sea uniforme y fácil de mantener. A continuación, se muestra de forma general el estándar de codificación definido para el módulo:

- Los nombres de cada uno de los elementos del programa deben ser significativos; su nombre debe explicar en lo posible el uso del elemento, como se muestra en la figura 10.
- No manejar en los programas más de una instrucción por línea, como se muestra en la figura 10.
- Declarar las variables en líneas separadas, como se muestra en la figura 10.
- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.
- La mayoría de los elementos se deben nombrar usando sustantivos.
- Siguiendo el estándar de codificación empleado en la Plataforma XILEMA AGORAV, sobre la cual se desarrolla el módulo, todas las declaraciones (atributos, clases y métodos) se escribirán completamente en minúscula, separando las palabras con guión bajo (_), como se muestra en la figura 11.

```
function distribucion_aut_tareas_distribuir_tarea(){

    $mensaje = "getUsageCPU";
    $json = json_encode($mensaje, JSON_FORCE_OBJECT);
    $contador = 0;
```

Figura 10. Estándares de codificación. Función.

```
class distribucion_aut_tareas_vista {

    function distribucion_aut_tareas_form_add1($form, &$form_state) {
        $form = array();
```

Figura 11. Estándares de codificación. Clase.

3.2 Diagrama de componentes

El diagrama de componentes permite describir los elementos físicos del sistema y sus relaciones que indican la utilización de servicios ofrecidos por otros componentes. Los componentes pueden ser simples archivos, paquetes, bibliotecas, entre otros (Larman, 2003). En la siguiente figura se muestra el diagrama de componentes del módulo en la Plataforma XILEMA AGORAV. Dicho diagrama contiene los ficheros .php que son clases, como la que hace referencias al modelo y a la vista implicadas en el funcionamiento del módulo. El fichero .module contiene funciones o ganchos *(hooks)* predefinidos por Drupal para ser utilizadas durante una petición. También se encuentra el fichero .install donde se definen las operaciones necesarias para el funcionamiento de la base de datos del módulo y el .info contiene información general del módulo como el nombre, las dependencias del mismo y las clases con que trabaja. A continuación, se muestra el diagrama de componentes del módulo.

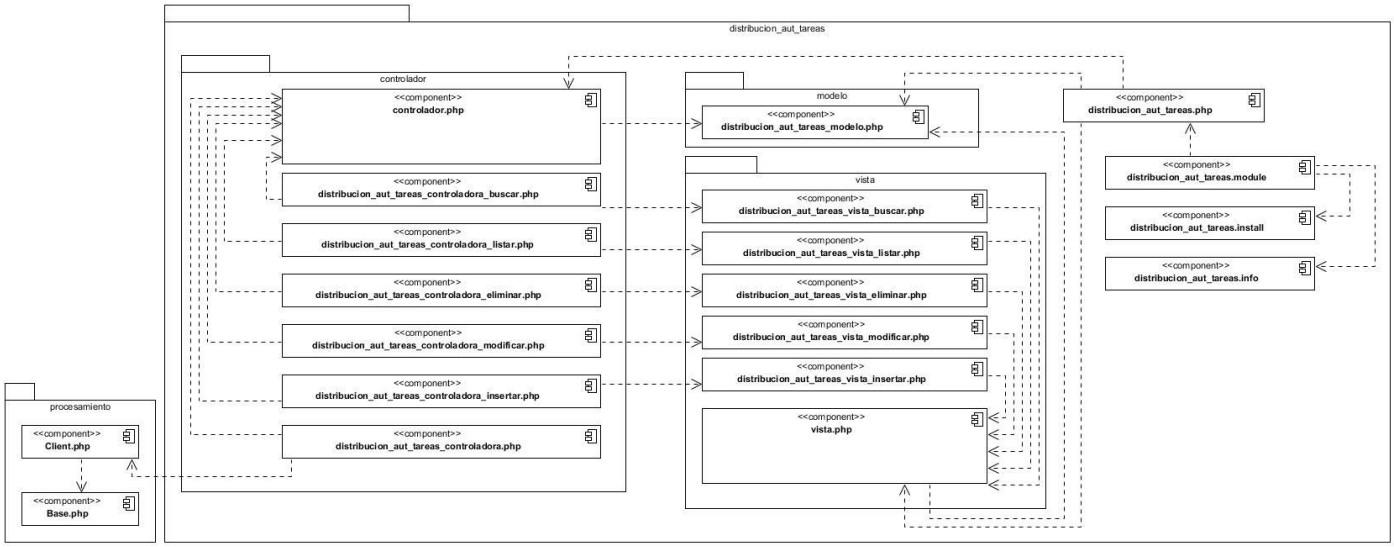


Figura 12. Diagrama de componentes

3.3 Diagrama de despliegue

Un diagrama de despliegue representa los recursos de cómputo que utiliza el sistema. Estos recursos son llamados nodos, que muestran su configuración y componentes. Las relaciones entre los nodos constituyen medios de comunicación (Larman, 2003). En la Figura 10 se muestra el diagrama de despliegue del módulo implementado.

Se representa un nodo PC Cliente el cual tiene como componente el navegador web Mozilla Firefox. Este nodo se comunica con la máquina servidora mediante el protocolo HTTPS (*Hypertext Transfer Protocol Secure*). A su vez la máquina servidora debe tener el servidor Apache versión 2 y la Plataforma XILEMA

AGORAV para su correcto funcionamiento, dicha plataforma incluye el Módulo de distribución automática de tareas a nodos de transcodificación. Además, se comunica con varios nodos como: el Servidor de Base de Datos que debe contener el gestor PostgreSQL versión 9.3, el Nodo de Transcodificación por el protocolo WSS (*Websocket Secure*) y con el Servidor de Medias mediante el protocolo NFS (*Network File System*). Por otro lado, el Nodo de Transcodificación incluye el Componente de codificación y decodificación y almacena el video convertido en el Servidor de Medias usando el protocolo NFS. El nodo Servidor *Streaming* cuenta con el componente *Icecast* 2.4.3 y recibe los archivos multimedia a publicar desde el Servidor de Medias empleando el protocolo NFS. Por último, la PC Cliente puede visualizar estas publicaciones mediante el protocolo HTTPS.

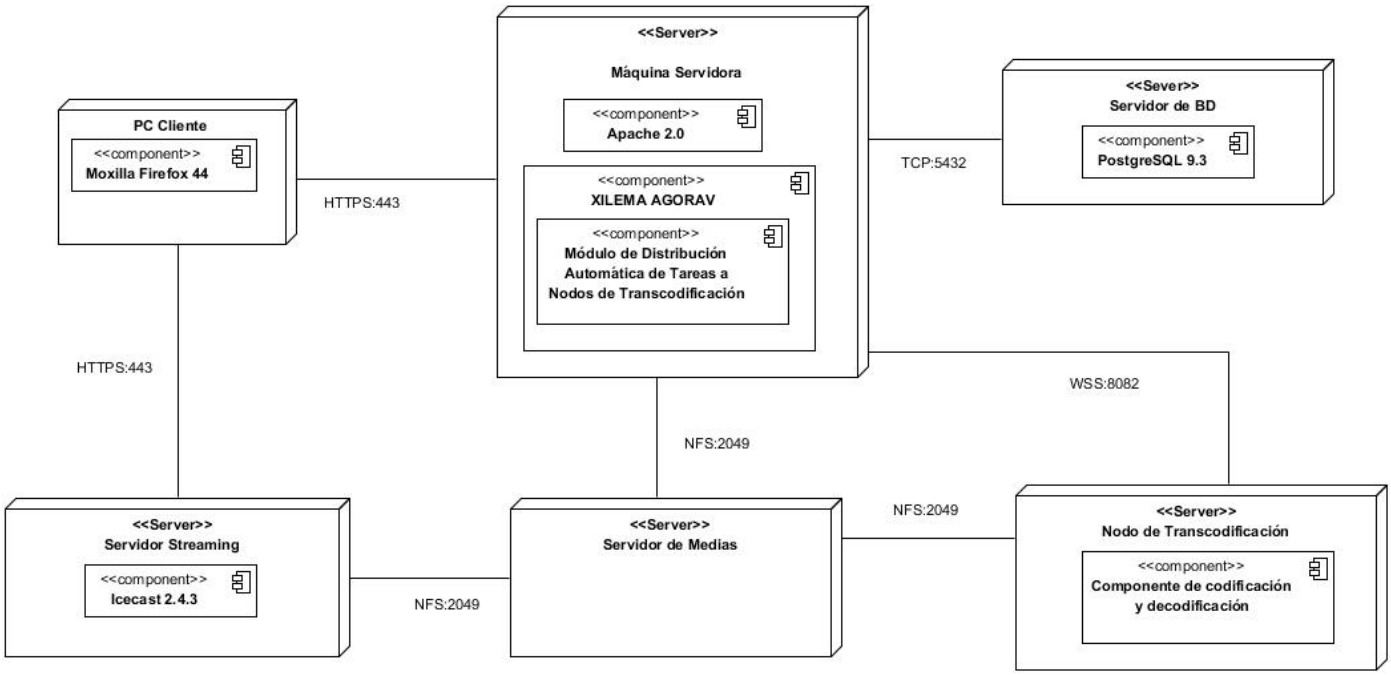


Figura 13. Diagrama de despliegue.

3.4 Modelo de pruebas

En la etapa de prueba se verifica el resultado de la implementación para la detección de errores y asegurar que la entrada definida produce resultados reales de acuerdo con los resultados requeridos (Pressman, 2005). Por lo que planificar estas pruebas y manejar los resultados de forma que los defectos puedan ser arreglados es de gran importancia para una correcta construcción del software. En esta etapa es generado el artefacto modelo de pruebas, (Kruchten, 2003) define que el mismo describe principalmente cómo se prueban los componentes ejecutables en el modelo de implementación con pruebas de integración y de sistema.

Una estrategia de prueba del software integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software (Pressman, 2010). Para el módulo se aplicará la estrategia de prueba de los niveles de integración y de sistema. Plantea (Pressman, 2010) que las pruebas de integración tienen como objetivo construir una estructura de programa que esté de acuerdo a lo que dicta el diseño. Las pruebas del sistema están constituidas por una serie de pruebas, y aunque cada una tiene un propósito diferente, todas trabajan para verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las funciones apropiadas.

3.4.1 Tipos de pruebas

En esta etapa se selecciona realizar al módulo las siguientes pruebas:

Prueba de integración ascendente: se integran primero los componentes de infraestructura, como el acceso a base de datos y a continuación se añaden los componentes funcionales (Sommerville, 2005). Para el módulo se utiliza el enfoque incremental de integración ascendente. Plantea (Pressman, 2010) que en el enfoque incremental el programa se construye y se prueba en pequeños incrementos, en los cuales resulta más fácil aislar y corregir los errores.

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

Pruebas de rendimiento: se centran en determinar la velocidad con la que un sistema bajo pruebas realiza una tarea en las condiciones particulares del escenario de pruebas (Globe, 2016). Las pruebas de rendimiento implican estresar el sistema (de ahí el nombre de prueba de estrés) realizando demandas que están fuera de los límites del diseño del software. Las pruebas de estrés van realizando pruebas acercándose a la máxima carga del diseño del sistema hasta que el sistema falla (Sommerville, 2005). Como estrategia para realizar las pruebas de rendimiento se hará uso de la prueba de estrés, para determinar cuánta capacidad puede manejar la funcionalidad distribuir tarea del módulo.

3.5 Aplicación y resultado de las pruebas

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

Tabla 4. Descripción de variables del caso de prueba del Caso de Uso Gestionar nodo.

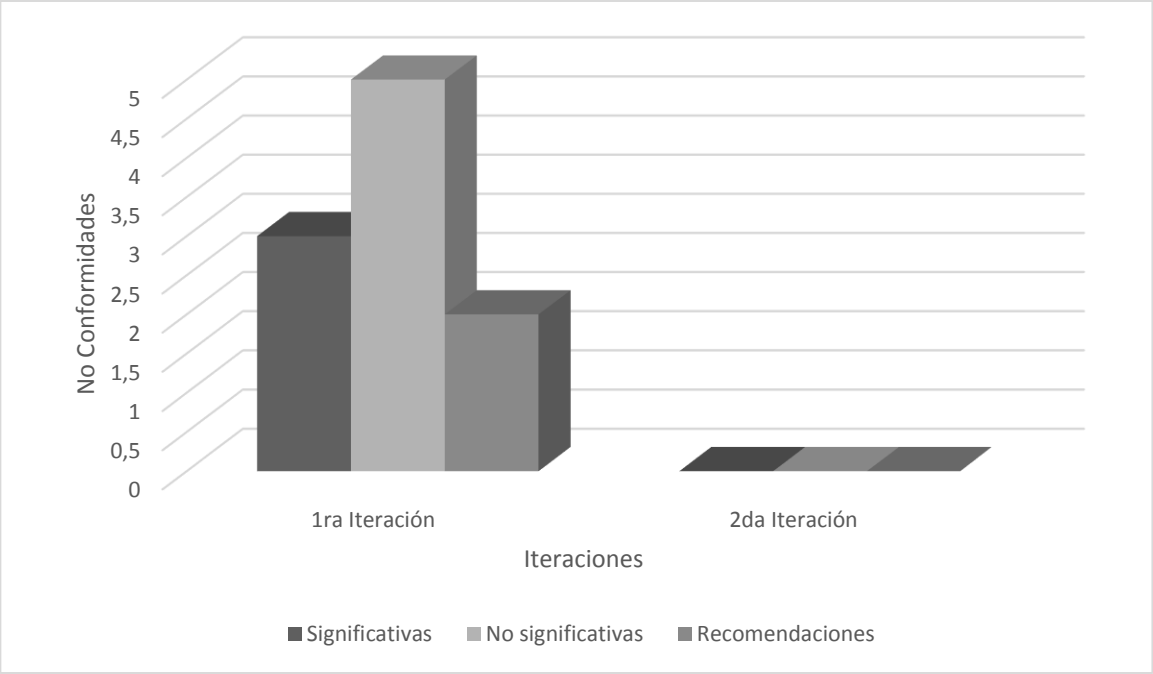
No.	Nombre del Campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Debe introducirse cualquier combinación del teclado
2	IP	Campo de texto	No	Debe introducirse el IP con el formato #. #. #. # (# es un número entero que está en el intervalo [0; 255]).
3	Puerto	Campo de texto	No	Debe introducirse un número entero que esté en el intervalo [1024; 65535].
4	Máxima capacidad de uso del CPU	Campo de texto	No	Debe introducirse un número entero que esté en el intervalo [0; 100].
5	Procesadores lógicos del CPU	Campo de texto	No	Debe introducirse un número entero que esté en el intervalo [0; 100].
6	Frecuencia del CPU	Campo de texto	No	Debe introducirse cualquier número mayor que 0.

Tabla 5. Descripción del Caso de Prueba para el Caso de Uso: Gestionar nodo. Sección: Insertar nodo.

Escenario	Descripción	Nombre	IP	Puerto	Máxima capacidad de uso del CPU	Procesadores lógicos del CPU	Frecuencia del CPU	Respuesta del sistema	Flujo central
EC 1.1 Insertar un nodo en el módulo correctamente.	El usuario introduce todos los parámetros correctamente y se adiciona un nodo al sistema.	V	V	V	V	V	V	Valida que los campos no se encuentren vacíos. Adiciona un nodo al sistema y muestra una vista con los nodos existentes.	• Se selecciona desde el menú principal la opción “Gestionar nodo”. • Se selecciona la opción “Insertar nodo”. • Se muestra un formulario con los campos a llenar de los nodos. • Se introducen los datos del nuevo nodo. • Se presiona el botón “Adicionar”. • Se verifican los datos.
		nodo uno	10.54.13.110	8802	100	2	2.2		
EC 1.2 Insertar un nodo en el módulo incorrectamente.	El usuario mantiene algún campo vacío o introduce datos inválidos.	I	V	V	V	V	V	Valida los datos introducidos. Mantiene la interfaz de insertar nodo y muestra un mensaje de error. En el primer caso "El campo nombre es obligatorio". En el segundo caso "El campo IP es obligatorio". En el tercer caso "Entre un puerto válido" y "Entre una Máxima capacidad de uso del CPU válida". En el último caso "Entre una cantidad de Procesadores lógicos del CPU válido" y "Entre una Frecuencia del CPU válida".	•Se selecciona desde el menú principal la opción “Gestionar nodo”. • Se selecciona la opción “Insertar nodo”. • Se muestra un formulario con los campos a llenar de los nodos. • Se introducen los datos del nuevo nodo. • Se presiona el botón “Adicionar”. • Se verifican los datos.
		vacío	10.54.13.10	3302	88	3	3.2		
		V	I	V	V	V	V		
		nodo dos	vacío	2406	32	2	4.1		
		V	V	I	I	V	V		
		3	10.8.107.106	dos	101	3	3.4		
		V	V	V	V	I	I		
		nodo 3	10.8.107.32	7834	32	2.2	cuatro		

Utilizando los casos de prueba diseñados se realizaron dos iteraciones de pruebas para encontrar la mayor cantidad de errores en el funcionamiento del módulo.

Tabla 6. Gráfico de resultado de las no conformidades.



En el gráfico anterior se evidencian 10 no conformidades detectadas al realizar la primera iteración, estas fueron clasificadas según su nivel de afectación en el funcionamiento del sistema, en significativas, no significativas y recomendaciones. Definidas como significativas aquellas no conformidades que afectaron el funcionamiento de la aplicación, las mismas fueron 3 errores de funcionalidad. Como no significativas se detectaron 2 errores ortográficos, 3 errores de mensajes y las restantes fueron recomendaciones o sugerencias de cambios a elementos del diseño. Al concluir la corrección de los errores detectados se dio fin a la primera iteración y se prosiguió a comenzar una nueva, para comprobar si existían errores. Una vez realizada la segunda iteración no se detectaron no conformidades.

Para el módulo se aplicó el enfoque incremental de **integración ascendente** ya que a medida que fueron desarrolladas las funcionalidades se agruparon y se coordinaron las entradas y salidas en cada iteración de la prueba realizada. Posteriormente se realizaron las pruebas al grupo de funcionalidades y al quedar probadas se continuó con el desarrollo de otras funcionalidades repitiendo el mismo enfoque. Una vez que se concluyó la implementación y se comprobó el correcto funcionamiento del módulo, se realizaron las pruebas para constatar su integración a la Plataforma XILEMA AGORAV y el Componente de codificación y decodificación para XILEMA AGORAV. A continuación se describen los resultados de las pruebas:

Se ejecutó la Plataforma XILEMA AGORAV y un usuario con rol “Gestor de contenidos” se autentificó en la misma. El usuario logró lo anterior con éxito e instantáneamente seleccionó entre los módulos el Módulo de distribución automática de tareas a nodos de transcodificación. El módulo debió mostrar correctamente un menú con todas sus funcionalidades, ocurrió que no se realizó pues existían clases que se repetían en el mismo, como *Client* y Base. Al concluir la corrección del error que dio fin a la primera iteración se prosiguió a comenzar una nueva iteración para comprobar si existían nuevos errores.

En esta nueva iteración se acciona el módulo, logrando así mostrarse correctamente el menú con todas sus funcionalidades. Seguidamente en el módulo se comprobaron todas sus funcionalidades, como por ciento actual del CPU de los nodos de transcodificación, cantidad de tareas de los nodos de transcodificación y distribuir tarea sin ocurrir ningún error, evidenciando en estas tres la integración con el Componente y con la plataforma de forma general. Así culmina exitosamente la prueba comprobándose la correcta integración del Módulo de distribución automática de tareas a nodos de transcodificación con la Plataforma XILEMA AGORAV y el Componente de codificación y decodificación para XILEMA AGORAV.

Las **pruebas de rendimiento** se realizaron haciendo uso de la estrategia de prueba de estrés. Las mismas fueron realizadas con la herramienta Apache Jmeter versión 2.8, que brinda la posibilidad de poder realizar pruebas de rendimiento con varios hilos de peticiones HTTP a la funcionalidad distribuir tarea del módulo. En el entorno donde se realizaron las pruebas se dispuso de una máquina servidora con sistema operativo Linux, un Procesador Core i3 a 3.3 GHz y 2 GB de memoria RAM.

Las pruebas fueron realizadas con 10, 50,100 y 101 muestras de datos, cuyas muestras son equivalentes a realizar 10, 50, 100 y 101 peticiones HTTP respectivamente a la funcionalidad distribuir tarea del módulo, con tres nodos de transcodificación insertados en el mismo. Dichas peticiones representan las ejecuciones de la funcionalidad distribuir tarea del módulo. Lo anterior se realizó cuando los usuarios con el rol “Gestor de contenidos” adicionaron concurrentemente archivos multimedia, en el formato no requerido, para ser publicado en la Plataforma XILEMA AGORAV.

Como configuración para la realización de las pruebas se tomó como IP (Protocolo de Internet) del servidor 127.0.0.1 o *localhost*, el puerto 8080 y la ruta para realizar la prueba fue la de la funcionalidad distribuir tarea del módulo como se muestra en la Figura 14 y Figura 15.

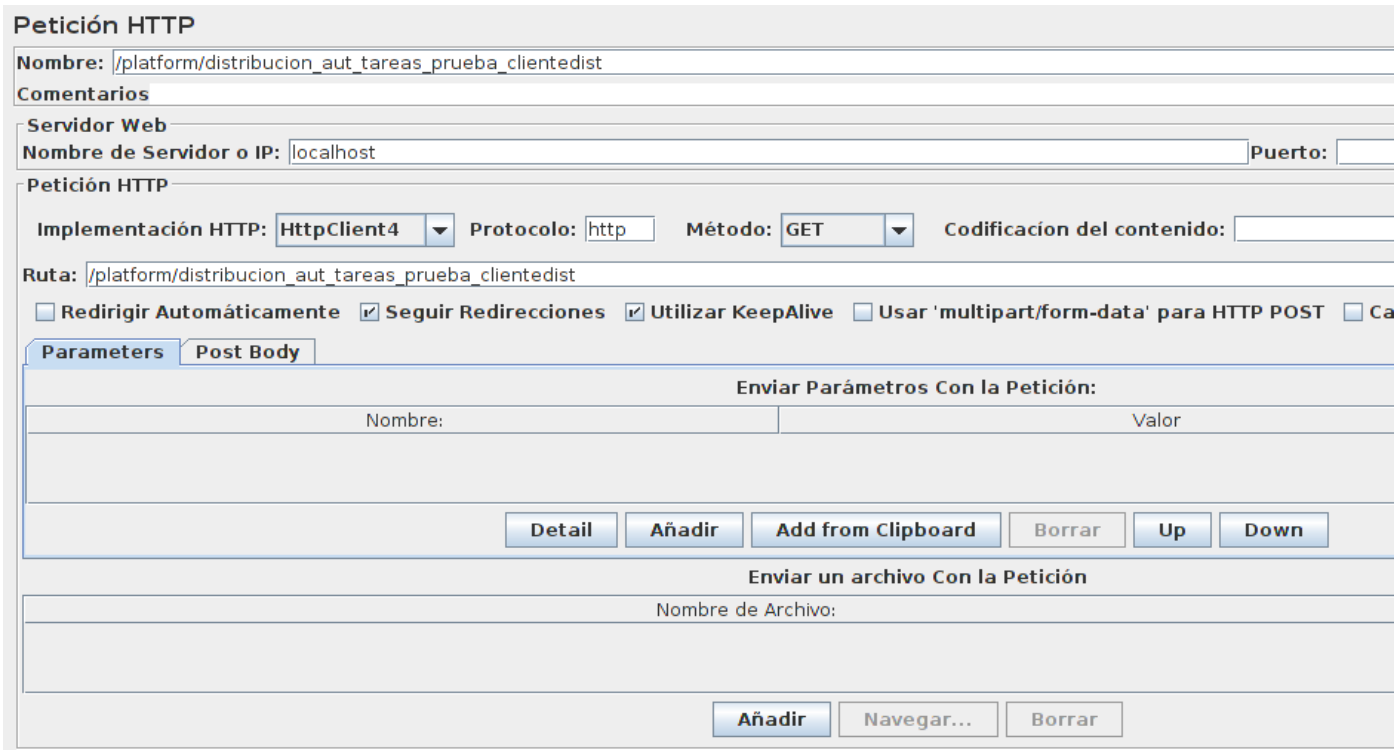


Figura 14. Representación de la configuración de las pruebas de rendimiento. Pate 1.

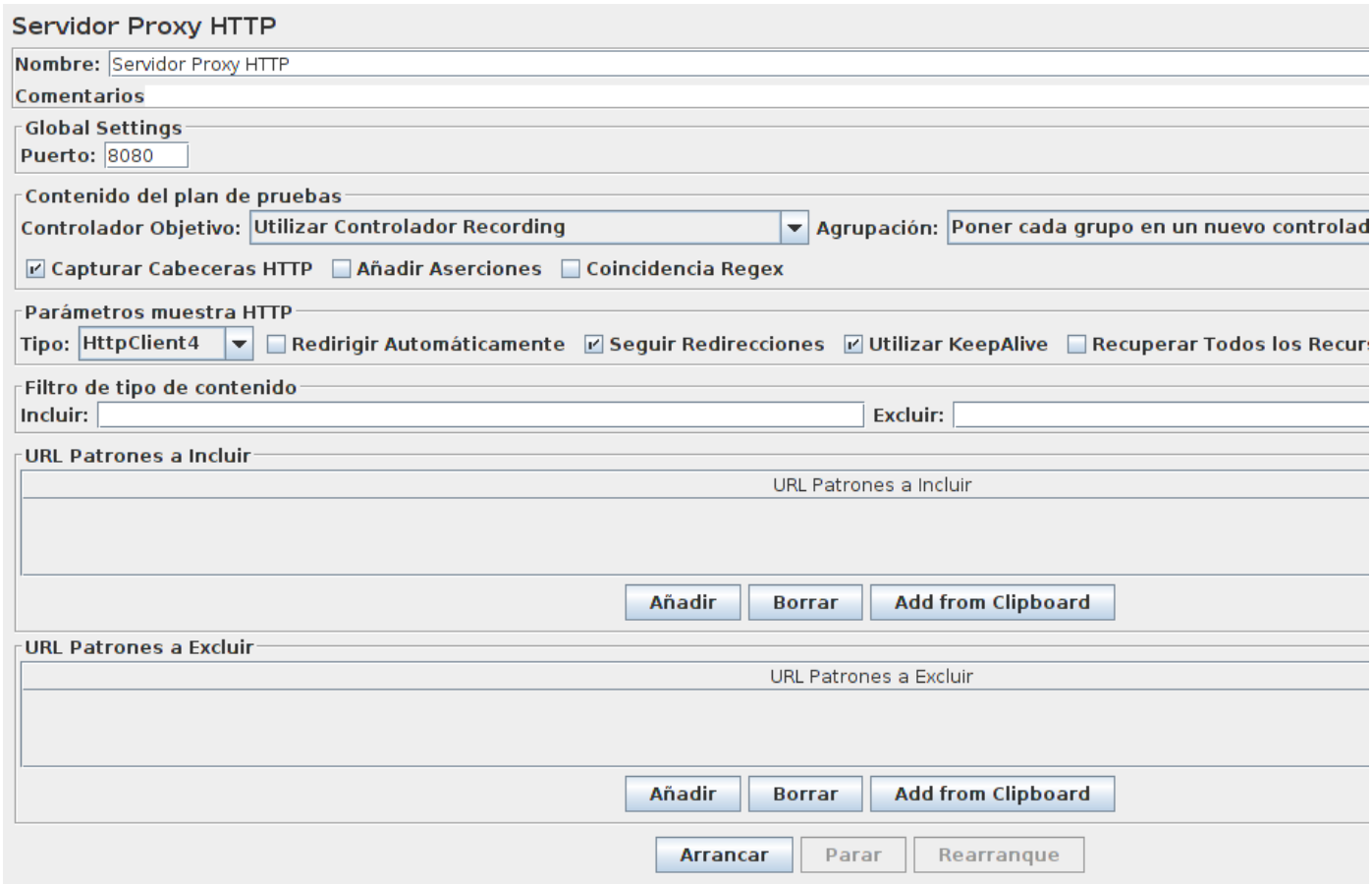
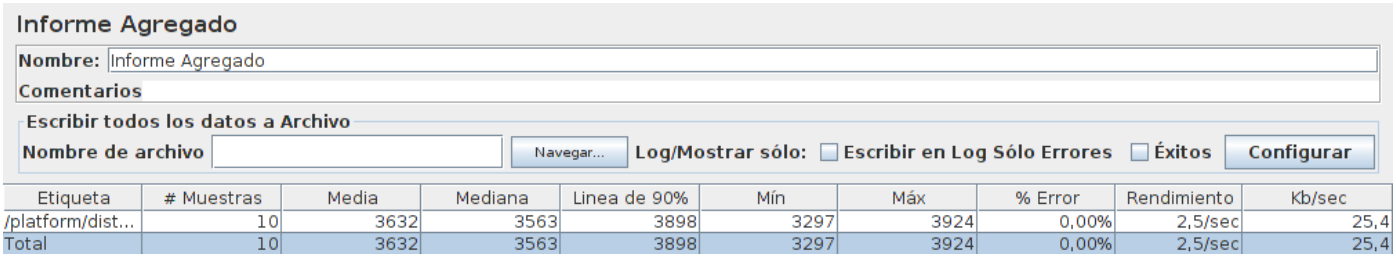


Figura 15. Representación de la configuración de las pruebas de rendimiento. Parte 2.

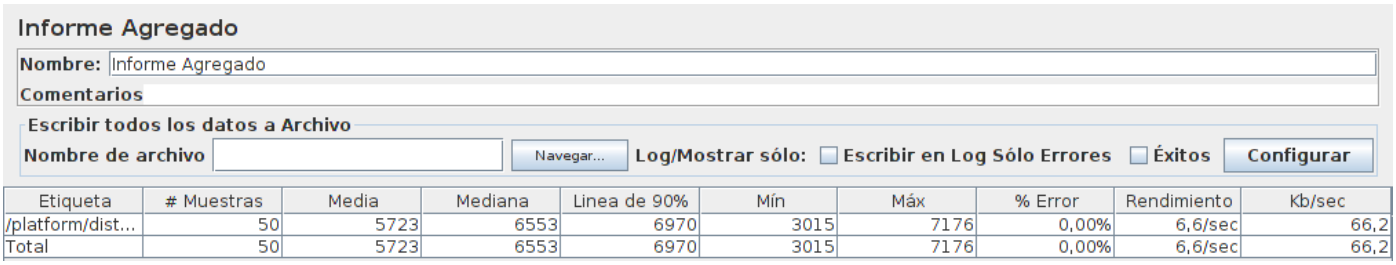
Al realizar la prueba con 10 muestras la funcionalidad distribuir tarea del módulo se ejecutó correctamente, como se muestra en la Figura 16, en un tiempo de 4 segundos como se evidencia en el siguiente cálculo: $\frac{\text{muestras}}{\text{rendimiento}} = \frac{10}{2.5} = 4.$



Etiqueta	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/platform/dist...	10	3632	3563	3898	3297	3924	0,00%	2,5/sec	25,4
Total	10	3632	3563	3898	3297	3924	0,00%	2,5/sec	25,4

Figura 16. Representación del resultado de la prueba de rendimiento con 10 muestras.

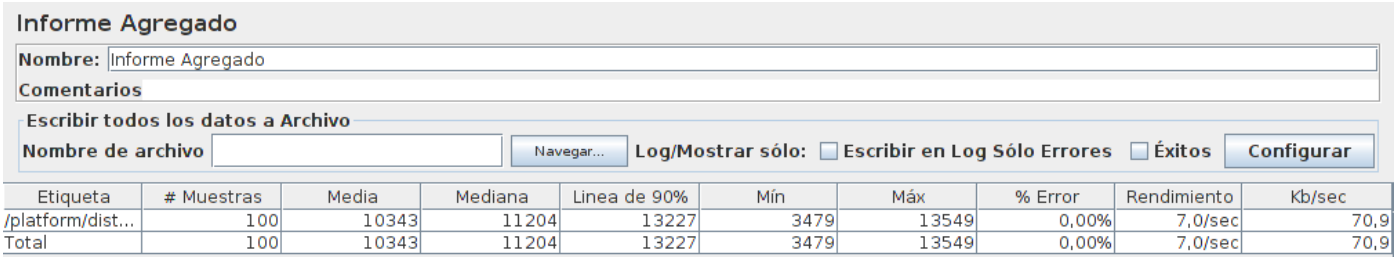
Al realizar la prueba con 50 muestras la funcionalidad distribuir tarea del módulo se ejecutó correctamente, como se muestra en la Figura 17, en un tiempo de 8 segundos aproximadamente como se evidencia en el siguiente cálculo: $\frac{\text{muestras}}{\text{rendimiento}} = \frac{50}{6.6} = 7.57 \approx 8.$



Etiqueta	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/platform/dist...	50	5723	6553	6970	3015	7176	0,00%	6,6/sec	66,2
Total	50	5723	6553	6970	3015	7176	0,00%	6,6/sec	66,2

Figura 17. Representación del resultado de la prueba de rendimiento con 50 muestras.

Al realizar la prueba con 100 muestras la funcionalidad distribuir tarea del módulo se ejecutó correctamente, como se muestra en la Figura 18, en un tiempo de 14 segundos aproximadamente como se evidencia en el siguiente cálculo: $\frac{\text{muestras}}{\text{rendimiento}} = \frac{100}{7.0} = 14.28 \approx 14.$



Etiqueta	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/platform/dist...	100	10343	11204	13227	3479	13549	0,00%	7,0/sec	70,9
Total	100	10343	11204	13227	3479	13549	0,00%	7,0/sec	70,9

Figura 18. Representación del resultado de la prueba de rendimiento con 100 muestras.

Al realizar la prueba con 101 muestras que son representadas en la Figura 19 como número de hilos, no se pudo observar el comportamiento de la funcionalidad distribuir tarea del módulo, debido a que la máquina servidora dejó de responder ante una carga de 101 usuarios con el rol “Gestor de contenidos”.

Figura 19. Representación de la prueba de rendimiento con 101 muestras.

Con los resultados obtenidos en cada prueba se demuestra que la funcionalidad distribuir tarea del módulo en un ámbito de trabajo normal bajo este entorno de pruebas, puede permitir una carga de 100 usuarios con el rol “Gestor de contenidos” concurrentemente.

3.5.1 Validación del módulo

Con el objetivo de verificar que el tiempo de publicación de los materiales audiovisuales disminuye, haciendo uso del módulo, se realizaron 4 pruebas, en la primera prueba se utilizó un único nodo de transcodificación y en las restantes se utilizaron varios nodos de transcodificación insertados en el módulo. En el entorno físico donde se realizaron las pruebas se dispuso de una máquina servidora con sistema operativo Linux, un Procesador Core i3 a 3.3 GHz y 2 GB de memoria RAM. También se utilizaron tres nodos de transcodificación con las siguientes propiedades:

- El primer nodo (N1): con sistema operativo Linux, un Procesador Core i3 a 3.3 GHz y 2 GB de memoria RAM.
- El segundo nodo (N2): con sistema operativo Linux, un Procesador Core i3 a 3.3 GHz y 4 GB de memoria RAM.
- El tercer nodo (N3): con sistema operativo Linux, un Procesador Core i7 a 2.10 GHz y 8 GB de memoria RAM.

En la primera prueba se contó con 5 archivos multimedia y un nodo de transcodificación (N1). Inició cuando un usuario con el rol de “Gestor de contenidos” accedió al módulo Gestor de contenidos en la Plataforma

XILEMA AGORAV a través de la máquina servidora. El usuario mencionado anteriormente adicionó los archivos multimedia, un archivo a cada instante. La plataforma AGORAV verificó en dicho instante el formato del mismo, y al no estar en el formato correcto (.webm) inició la funcionalidad distribuir tarea del módulo, seleccionando así el mejor nodo para transcodificar el archivo multimedia. Instantáneamente dicho nodo fue notificado para transcodificar el mismo en la dirección designada, logrando con esta iniciar su transcodificación.

Teniendo en cuenta el procedimiento descrito anteriormente, se tomó el tiempo en que inició y finalizó la transcodificación para la primera prueba, dichos tiempos fueron 10:03 am y 10:54 am respectivamente. La diferencia de los mismos permitió calcular el tiempo que demoró transcodificar los archivos multimedia, obteniéndose un resultado (T1) de 51 minutos.

La segunda prueba contó con 5 archivos multimedia y 2 nodos de transcodificación (N1 y N2). Se inició el procedimiento descrito en la primera prueba y los tiempos tomados fueron de 1:07 pm y 1:28 pm respectivamente. Obteniéndose un tiempo de demora de los archivos multimedia transcodificados (T2) de 22 minutos.

La tercera prueba contó con 10 archivos multimedia y 2 nodos de transcodificación (N1 y N2). Se inició el procedimiento descrito en la primera prueba y los tiempos tomados fueron de 1:40 pm y 2:29 pm respectivamente. Obteniéndose un tiempo de demora de los archivos multimedia transcodificados (T3) de 49 minutos.

La cuarta prueba contó con 10 archivos multimedia y 3 nodos de transcodificación (N1, N2 y N3). Se inició el procedimiento descrito en la primera prueba y los tiempos tomados fueron de 2:35 pm y 3:11 pm. Obteniéndose un tiempo de demora de los archivos multimedia transcodificados (T4) de 36 minutos.

Tabla 7. Resultado de las pruebas para la validación del módulo.

Pruebas	**Cantidad de archivos multimedia**	**Cantidad de nodos de transcodificación**	**Tiempo en que inició la transcodificación**	**Tiempo en que finalizó la transcodificación**	**Tiempo que demoró transcodificar los archivos multimedia**
1	5	1 (N1)	10:03 am	10:54 am	51 min
2	5	2 (N1, N2)	1:07 pm	1: 29 pm	22 min
3	10	2 (N1, N2)	1:40 pm	2:29 pm	49 min
4	10	3 (N1, N2, N3)	2:35 pm	3:11 pm	36 min

Una vez terminada la transcodificación de los archivos multimedia y tomados los tiempos de demora de dicha transcodificación, los mismos están listos para ser publicados. El tiempo de demora de la publicación de estos está dado por el tiempo que demore transcodificarlos, como se muestra en la Tabla 7.

Los tiempos de demora de los archivos multimedia transcodificados en las pruebas realizadas fueron comparados, como se evidencia a continuación: T1 = 51 min > T2 = 22 min y T3= 49 min > T4 = 36 min. Y se pudo concluir de dicha comparación, que cuando se adicionan varios archivos multimedia a transcodificar y se cuenta con varios nodos de transcodificación insertados en el módulo, el tiempo de transcodificación de los mismos disminuye con respecto a realizar dicho proceso en un único nodo como se evidencia en la Tabla 7.

La diferencia entre T1 y T2 fue calculada, como se evidencia a continuación: T1 = 51 min - T2 = 22 min = 29 min. Lo anterior demuestra que actualmente con el funcionamiento del Módulo de distribución automática de tareas a nodos de transcodificación, se logra realizar en menor tiempo la publicación de los materiales audiovisuales en XILEMA AGORAV.

3.6 Conclusiones parciales

En este capítulo se concluye que:

- El diagrama de componentes permitió comprender la vista estática del módulo.
- La realización del diagrama de despliegue permitió comprender la distribución física de los distintos componentes lógicos desarrollados.
- El uso del estándar de codificación permite el entendimiento del código por otros programadores que no sean del equipo de desarrollo en caso de cambios en el módulo en un futuro.
- Las pruebas aplicadas al módulo demostraron que el mismo se integra satisfactoriamente con Plataforma XILEMA AGORAV y el Componente de codificación y decodificación para XILEMA AGORAV y cumple con los requisitos trazados.

Conclusiones generales

En la investigación se concluye que:

- A partir del estudio del arte realizado se evidenció que no existen soluciones que permitan resolver el problema planteado, por lo cual se requirió desarrollar una nueva solución.
- Se desarrolló un Módulo de distribución automática de tareas a nodos de transcodificación que permite disminuir el tiempo de publicación de los materiales audiovisuales en la Plataforma XILEMA AGORAV, al cual se le realizaron pruebas de integración, rendimiento, caja negra y las pruebas realizadas para la validación del módulo, constatándose que el mismo está apto para ser utilizado en la Plataforma XILEMA AGORAV.
- El uso de la metodología AUP variante UCI posibilitó guiar el proceso de desarrollo del módulo acorde a las normas vigentes en la UCI y la generación de artefactos ingenieriles útiles para futuras labores de mantenimiento al módulo.
- El uso del patrón arquitectónico MVC, el protocolo seguro de comunicación *WebSockets* y el CMS Drupal posibilitó obtener un módulo flexible, escalable y acorde a las tendencias actuales en el desarrollo de este tipo de sistemas.

Recomendaciones

- Se recomienda que el algoritmo de distribución de tareas incluya como parámetros a tenerse en cuenta la duración de la media, el bitrate, el framerate y el formato de entrada y de salida. Dichos parámetros pueden aportar una mayor precisión al seleccionar el mejor nodo de transcodificación en el algoritmo.

Referencias Bibliográficas

Academic Developer Technology Group. 2015. *Academic Developer Technology Group.* [Online] 2015. [Cited: 12 12, 2015.] https://sites.google.com/site/gabineteutn/investigacion-y-desarrollo/html5/tutoriales/introduccion-a-websocket.

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

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

Apache. 2016. Apache. [Online] 2016. [Cited: 3 22, 2016.] http://httpd.apache.org/ABOUT_APACHE.html.

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

ArPUG. 2016. ArPUG. [Online] 2016. [Cited: 4 24, 2016.] http://www.arpug.com.ar/trac.

Bartolomé, Antonio, et al. 2007. *La web audiovisual.* 2007.

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

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

Citrix. 2016. Citrix. [Online] 2016. [Cited: 3 2, 2016.] https://www.citrix.com/glossary/load-balancing.html.

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

Computing Careers and Degrees. 2015. Computing Careers and Degrees. [Online] 2015. [Cited: 12 10, 2015.] http://computingcareers.acm.org.

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

Creati. 2016. Creati. [Online] 2016. [Cited: 2 18, 2016.] http://creati.es.

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

Datastax. 2016. Datastax. [Online] 2016. [Cited: 3 2, 2016.] http://www.datastax.com/dev/blog/what-persistence-and-why-does-it-matter.

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

Drupal. 2016. Drupal. [Online] 2016. [Cited: 1 25, 2016.] http://drupal.org.es/caracteristicas.

Drupal Hispano. 2016. Drupal Hispano. [Online] 2016. [Cited: 4 24, 2016.] http://drupal.org.es/caracteristicas.

Flumotion. 2015. Flumotion. [Online] 2015. [Cited: 10 28, 2015.] http://www.flumotion.com/plataforma-de-streaming/.

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

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

Globe. 2016. Globe. [Online] 2016. [Cited: 4 21, 2016.] http://www.globetesting.com/pruebas-de-rendimiento/.

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

Gustavo, Núñez Carvonel. 2010. 2010.

Harmonic. 2015. Harmonic. [Online] 2015. [Cited: 10 26, 2015.] http://www.harmonicinc.com/product/.

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

HTML5. 2015. CSS3, HTML5. [Online] 2015. [Cited: 24 10, 2015.] http://html5.dwebapps.com/que-es-css3/.

Hugo, Marca and Quisbert, Nancy. 2010. Diagrama de Despliegue. [Online] 2010. [Cited: 3 14, 2014.] virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc.

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

ifm. 2015. ifm. [Online] 2015. [Cited: 10 28, 2015.] http://www.ifm.com/ifmmx/web/pinfo015_010_040.htm.

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

ISID. 2016. ISID. [Online] 2016. [Cited: 4 25, 2016.] www.isid.com.

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

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

JetBrains. 2014. PHP Storm. [Online] 2014. [Cited: noviembre 20, 2015.] https://www.jetbrains.com/phpstorm/whatsnew/.

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

jWebSocktes. 2015. jWebSocktes. [Online] 2015. [Cited: noviembre 22, 2015.] https://jwebsocket.org/.

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

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

Kioskea. 2016. Kioskea. [Online] 2016. [Cited: febrero 3, 2016.] es.kioskea.net.

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

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

Larousse. 2016. Larousse. [Online] 2016. [Cited: 2 23, 2016.] http://www.larousse.mx.

Lenguajes de Programación. 2015. Lenguajes de Programación. [Online] 2015. [Cited: 12 3, 2015.] http://www.lenguajes-de-programacion.com.

LifeSize. 2015. LifeSize. [Online] 2015. [Cited: 10 20, 2015.] www.lifesize.com.

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

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

Microsoft. 2015. Microsoft. [Online] 2015. [Cited: febrero 2016, 23.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

Mosaic. 2004. Mosaic. [Online] 2004. [Cited: 12 15, 2015.] http://mosaic.uoc.edu/2004/11/29/introduccion-a-los-sistemas-de-gestion-de-contenidos-cms-de-codigo-abierto/.

Mozila. 2015. Mozilla. [Online] 2015. [Cited: 12 15, 2015.] https://developer.mozilla.org/es/docs/Web/API/WebRTC_API.

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

PHP. 2016. PHP. [Online] 2016. [Cited: 1 28, 2016.] php.net.

PostgreSQL. 2016. PostgreSQL. [Online] 2016. [Cited: 1 20, 2016.] http://www.postgresql.org/about/.

Powers, John, et al. 2006. *Distributed processing system.* 2006.

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

—. 2002. *Ingeniería del Software: Un enfoque Práctico.* 2002.

Pressman, Roger S and Maxim, Bruce R. 2014. *Software Engineering: A Practitioner's Approach.* 2014.

Pressman, Roger S. 2005. *Ingeniería del Software.* 2005.

Putti, Srinivasrao , et al. 2013. *Scalable Distributed Job Processing with Dynamic.* 2013.

Qin, Hang, ZHU, Li and WU, Zhongbo. 2009. *Dynamic Load Balancing with Overlay-Based Reconfiguration for Wireless Sensor Networks.* 2009.

RAE. 2016. RAE. [Online] 2016. [Cited: 3 30, 2016.] http://www.rae.es/.

Ramos, Kariné Blanco. 2013. *Proceso Base de Ingeniería de Requisitos para las pequeñas y medianas empresas de desarrollo de software.* Cuba : s.n., 2013.

Reyes, Yunia, Gonzáles, Lisbet and Ruiz, Yadira. 2008. *Estudio comparativo sobre las principales metodologías pesadas y orientadas a objetos en el desarrollo de software.* Habana : s.n., 2008.

Rodríguez, Fran Gil. 2012. *Experto en Drupal 7 Nivel Inicial.* 2012.

S. Dormido, R. Hernández, S. Ros y J. Sánchez. 2003. *Procesamiento Paralelo. Teoría y Programación .* Madrid : s.n., 2003.

Sanchez, Miguel Angel. 2001. *JavaScript.* s.l. : INNOVA 2001, 2001.

Sánchez, Tamara Rodríguez. 2015. *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana : s.n., 2015.

Savitz, Eric. 2015. *Forbes.* 2015.

Skvorc, Bruno. 2015. Best PHP IDE in 2015 - Survey Results. [Online] 2015. [Cited: noviembre 15, 2015.] http://www.sitepoint.com/best-php-ide-2014-survey-results/.

Somerville, Ian. 2009. *Ingeniería del Software Séptima Edición.* 2009.

Sommerville, Ian. 2005. *Ingeniería del Software Séptima Edición.* Madrid : Pearson Educación.SA, 2005.

SparxSystems. 2007. Diagrama de Componentes UML 2. [Online] Sparx Systems Pty Ltd., 2007. [Cited: marzo 11, 2016.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.html.

Stallings, William. 2003. *Sistemas Operativos.* Universidad Pontificia de Salamanca en Madrid : s.n., 2003.

Streaming Media. 2016. Streaming Media. [Online] 2016. [Cited: 4 13, 2016.] http://www.streamingmedia.com/Articles/Editorial/What-Is-./What-is-Encoding-and-Transcoding-75025.aspx.

Sullivan, Sean . 2010. *Programming WebSockets.* 2010.

Tanenbaum, Andrew S and Van Steen, Maarten. 2008. *Distributed Systems: Principles and Paradigms.* 2008.

Tecnología Streaming para radio digital Universitaria. **Acevedo, Edwin jeovanny Clavijo, Hernández Chacón, Sorey y Cardoza Vásquez, Edison. 2015.** Bucaramanga-Colombia : Revista ESCAICA, 2015, Vol. 1.

Tedial. 2015. MPM. [Online] 2015. [Cited: diciembre 1, 2015.] http://www.tedial.com/products/mpm/media-automation.

Tejedor, Jesús Ramón Milian. 2014. *WebRTC (Web Real-Time Communications).* 2014.

Telestream. 2015. Telestream. [Online] 2015. [Cited: 12 1, 2015.] http://www.telestream.net/vantage.

Tim Converse, Joyce Park, Clark Morgan. 2004. *PHP5 and MySQL Bible (Bible).* 2004.

Tinetti, Fernando and Wolfmann, Gustavo. 2008. *Análisis de Paralelización con Memoria Compartida y Memoria Distribuida.* La Plata : s.n., 2008.

Tomasi, Wayne. 2010. *Sistemas de Comunicaciones Electrónicas, 4ta Edición.* México : s.n., 2010.

Traitement des images par le contenu. 2016. Traitement des images par le contenu. [Online] enero 19, 2016. [Cited: enero 19, 2016.] http://www.irisa.fr/texmex/index.htm.

UCOL. 2015. Herramientas Case. [Online] 2015. [Cited: 12 15, 2015.] http://docente.ucol.mx/al961223/public_html/centro6.htm.

UML. 2016. UML. [Online] 2016. [Cited: 1 19, 2016.] http://www.uml.org/.

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

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

Versión 2.4 del Servidor HTTP Apache. 2016. Versión 2.4 del Servidor HTTP Apache. [Online] 2016. [Cited: 1 20, 2016.] http://httpd.apache.org/docs/2.4/.

Visual Paradigm. 2015. Visual Paradigm. What VP - UML Provides. [Online] 2015. [Cited: 10 20, 2015.]

http://www.visual-paradigm.com/product/vpuml/provides.

VSN. 2016. VSN. [Online] 2016. [Cited: 5 30, 2016.] https://www.vsn-tv.com/es.

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

W3C. 2016. W3C. [Online] 2016. [Cited: 5 18, 2016.] https://www.w3.org.

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

Websocket. 2016. *Websocket.* [Online] 2016. [Cited: 12 12, 2016.] http://www.websocket.org.

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

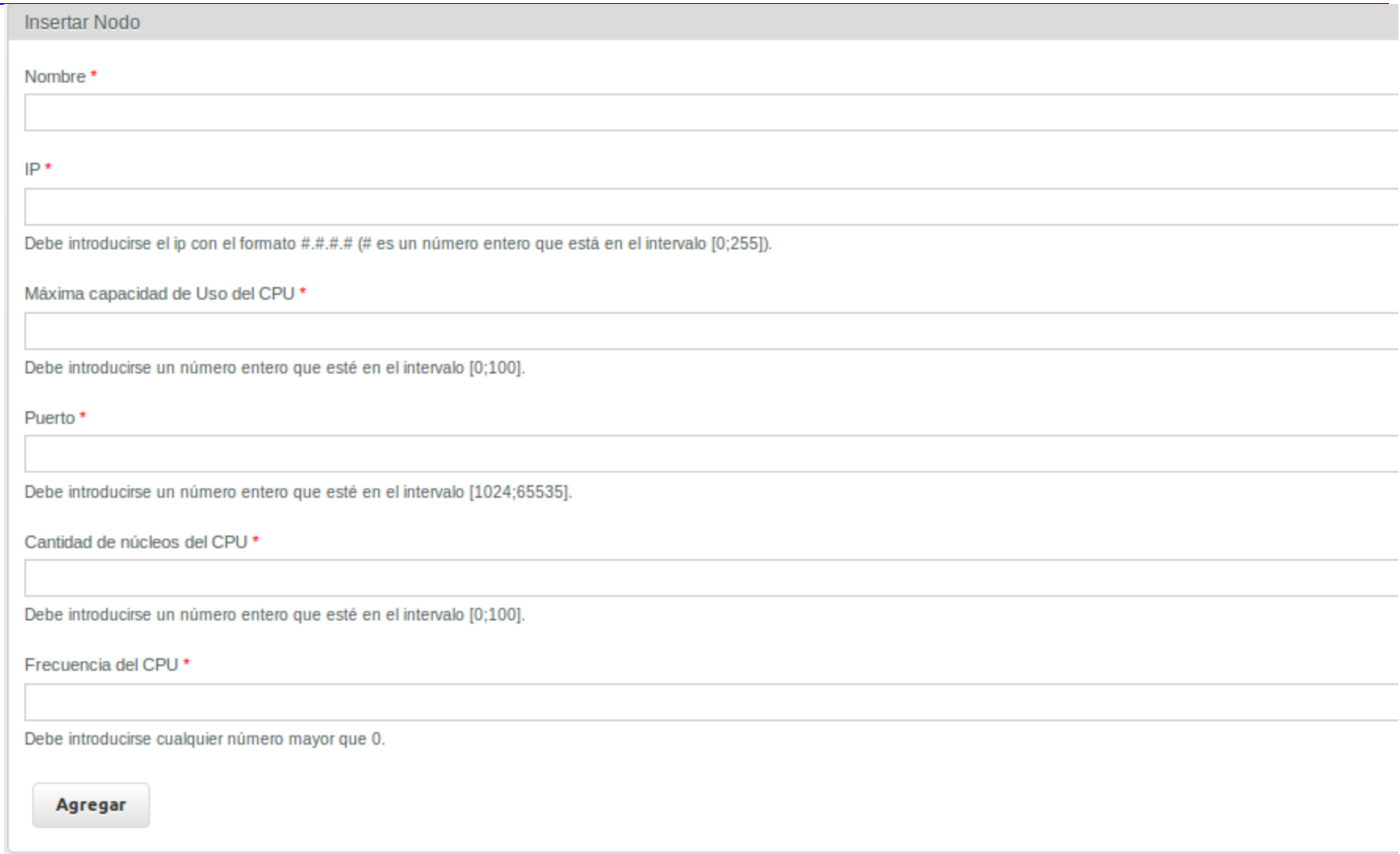
XTREAM. 2016. XTREAM. [Online] 2016. [Cited: 5 30, 2016.] http://www.xtreamsig.com.

Anexos

I. Descripción de requisitos funcionales

Tabla 8. Descripción de requisitos funcionales.

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
1.	Insertar nodo	El módulo debe permitir insertar un nodo al sistema	media	media
	Prototipo			



Campos	Tipos de Datos	Reglas o Restricciones
Entrada: Nombre del nodo	Cadena de caracteres.	Obligatorio

	Entrada: Dirección IP del nodo	Cadena de caracteres.	Obligatorio	
	Entrada: Puerto de comunicación	Número entero	Obligatorio	
	Entrada: Máxima capacidad de uso del CPU	Número entero	Obligatorio	
	Entrada: Procesadores lógicos del CPU	Número entero	Obligatorio	
	Entrada: Frecuencia del CPU del nodo	Número	Obligatorio	
	Salida: Lista nodos en el sistema	Lista	No procede	
	Observaciones			
2.	Modificar nodo	El módulo debe permitir modificar un nodo del sistema	media	media
	Prototipo			

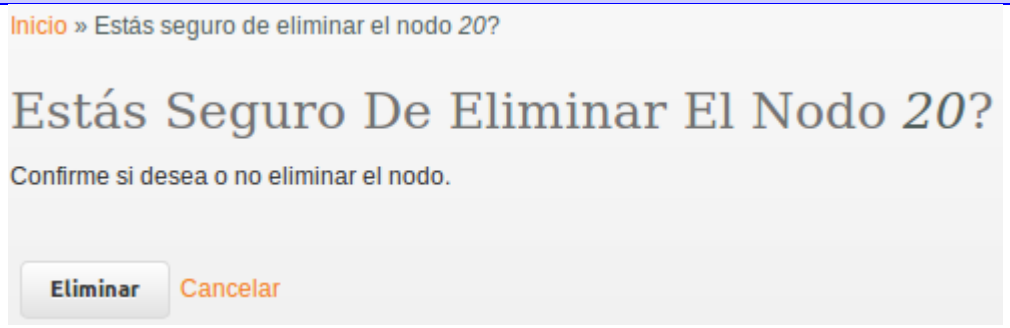
Campos	Campos	Campos
Entrada: Nombre del nodo	Cadena de caracteres	Obligatorio

	Entrada: Dirección IP del nodo	Cadena de caracteres	Obligatorio	
	Entrada: Puerto de comunicación	Número entero	Obligatorio	
	Entrada: Máxima capacidad de uso del CPU	Número entero	Obligatorio	
	Entrada: Procesadores lógicos del CPU	Número entero	Obligatorio	
	Entrada: Frecuencia del CPU del nodo	Número	Obligatorio	
	Salida: Lista nodos en el sistema	Lista	No procede	
	Observaciones			
3.	Listar nodos	El módulo debe permitir listar los nodos registrados en el sistema	media	media

Prototipo

Id	Nombre	IP	Puerto	Cantidad de núcleos del CPU	Frecuencia del CPU	Máxima capacidad de uso del CPU %	Actual CPU %	Actual cantidad de medias procesando
20	nodo uno	10.54.13.11	8802	2	3.3	100	sin conexión	sin conexión
21	nodo dos	10.54.13.110	8802	2	3.3	100	sin conexión	sin conexión
23	nodo tres	10.54.13.249	8802	2	2.6	98	sin conexión	sin conexión

Campos	**Tipos de Datos**	**Reglas o Restricciones**

	Salida: Lista de nodos	Lista	Tiene que haber al menos un nodo registrado en el sistema.	
	Observaciones			
4.	Eliminar nodo	El módulo debe permitir eliminar un nodo	media	media
	Prototipo			
				
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	Salida: Lista de nodos	Lista	No procede	
	Observaciones			
5.	Buscar nodo	El módulo debe permitir buscar un nodo	media	media
	Prototipo			
				

	Campos	**Campos**	**Campos**	
	Entrada: Nombre	Cadena de caracteres	No procede.	
	Entrada: Dirección IP	Cadena de caracteres.	No procede	
	Salida: nodo	Nodo de transcodificación	No procede	
	Observaciones			
6.	Mostrar Cantidad de Tareas de cada nodo	El módulo debe permitir mostrar la cantidad de tareas de cada nodo registrado en el sistema	baja	alta
	Prototipo			
				
	Campos	**Campos**	**Campos**	
	Salida: Cantidad de tareas de cada nodo	Número entero	Debe haber al menos un nodo registrado en el sistema	
	Observaciones			
7.	Mostrar porciento del uso del CPU de cada nodo	El módulo debe permitir mostrar el uso del CPU de cada nodo registrado en el sistema	baja	alta
	Prototipo			

		Actual cantidad de medias procesando sin conexión sin conexión sin conexión		
	Campos	**Campos**	**Campos**	
	Salida: uso del CPU del nodo en porciento	Número entero	Debe haber al menos un nodo registrado en el sistema	
	Observaciones			
8.	Distribuir tareas	El módulo debe permitir mostrar distribuir tareas al mejor nodo disponible en el sistema	alta	alta
	Prototipo			
	No procede.			
	Campos	**Campos**	**Campos**	
	Salida: Mejor nodo disponible en el sistema	Nodo de transcodificación	Debe haber al menos un nodo registrado en el sistema	
	Observaciones			

II. Especificación de Casos de Uso.

Tabla 9. Descripción del Caso de Uso. Mostrar estado del nodo.

CU	Mostrar estado del nodo.	
Objetivo	El CU de realiza para mostrar el estado de los procesos existentes en cada nodo.	
Actores	"Gestor de contenidos".	
Resumen	El CU se inicia cuando el "Gestor de contenidos" desea mostrar cantidad de tareas de cada nodo o mostrar por ciento del uso del CPU. El CU termina cuando se ejecuten las funcionalidades anteriormente especificadas.	
Complejidad	Baja.	
Prioridad	Crítica.	
Precondiciones	Que el "Gestor de contenidos" se autentique en la Plataforma AGORAV.	
Postcondiciones	Se muestra la cantidad de tareas de cada nodo y el por ciento del uso del CPU de cada nodo	
Referencias	RF6, RF7.	
Flujo de eventos		
Flujo básico: Mostrar estado del nodo		
	Actor	**Sistema**
1.	Selecciona la opción "Gestionar nodo".	
2.		Muestra una vista con un listado de los nodos registrados en el sistema.
3.	Selecciona la opción "*Listar nodos".*	
4.		El sistema verifica que haya nodos en el sistema.

5.		Muestra una vista con los nodos, incluyendo su por ciento de uso del CPU y la cantidad de tareas.
6.	Puede realizar una de las opciones que se especifican a continuación: - Mostrar por ciento de uso del cpu actual de cada nodo. - Mostrar cantidad de tareas actuales de cada nodo.	
7.		El CU termina cuando haya sido ejecutada alguna de las funcionalidades anteriormente especificadas.
Flujos alternos		
4.1: No hay ningún nodo registrado en el sistema.		
	Actor	**Sistema**
4.1.1		Muestra un mensaje "*La tabla de nodos de transcodificación se encuentra vacía*" y termina el CU.
Relaciones	Es extendido del CU Gestionar nodo.	
Requisitos no funcionales	RNF 1, RNF 2, RNF 3, RNF 4.	
Asuntos pendientes	No	

Tabla 10. Descripción del Caso de Uso. Distribuir tarea.

CU	Distribuir tarea.
Objetivo	Este CU se realiza para asignar las tareas a los nodos.
Actores	Sistema.

Resumen	El CU se inicia cuando el sistema desea asignar tareas a los nodos. El CU termina cuando haya sido ejecutada la funcionalidad anteriormente especificada.	
Complejidad	Alta.	
Prioridad	Crítica.	
Precondiciones	Que se adicione un archivo que no cumpla con el formato requerido.	
Postcondiciones	Se asigna tareas a los nodos.	
Referencias	RF8	
Flujo de eventos		
Flujo básico: Distribuir tarea.		
	Actor	**Sistema**
1.	Adiciona al sistema un archivo de multimedia que no tenga el formato (.webm).	
2.		Verifica que existan nodos en el sistema
3.		Distribuye la tarea al mejor nodo disponible.
4.		El CU termina cuando el sistema procesa la información.
Flujos alternos		
3.1: No hay ningún nodo registrado en el sistema.		
	Actor	**Sistema**
3.1.1		Muestra un mensaje "*La tabla de nodos de transcodificación se encuentra vacía*".
		Termina el CU.
Relaciones	No	
Requisitos no funcionales	RNF 1, RNF 2, RNF 3, RNF 4.	

Asuntos pendientes	No

III. Diagramas de secuencia.

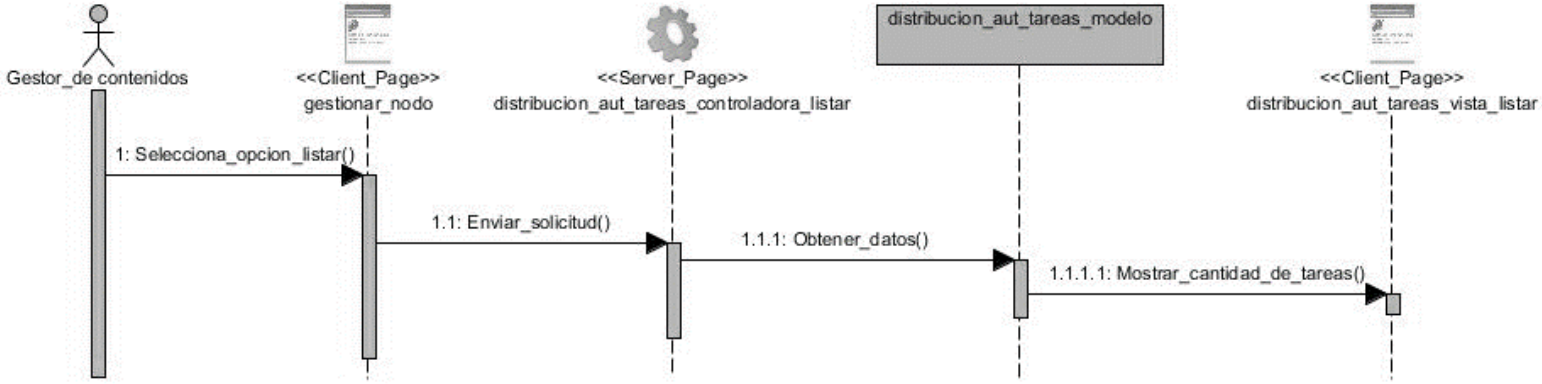


Figura 20. Diagrama de secuencia. Mostrar por ciento del uso del CPU de cada nodo.

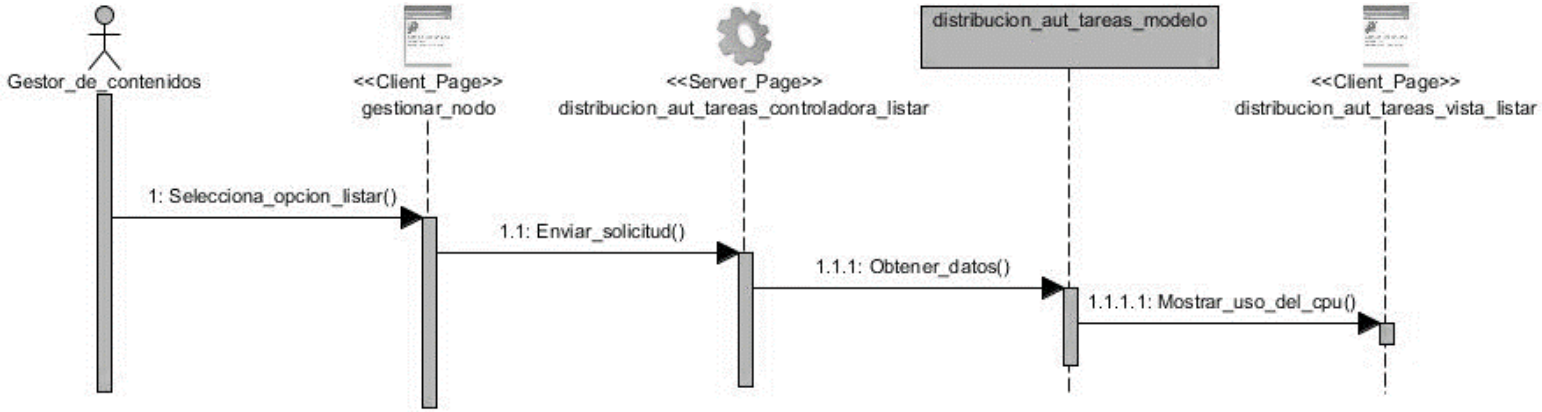


Figura 21. Diagrama de secuencia. Mostrar cantidad de tareas de cada nodo.

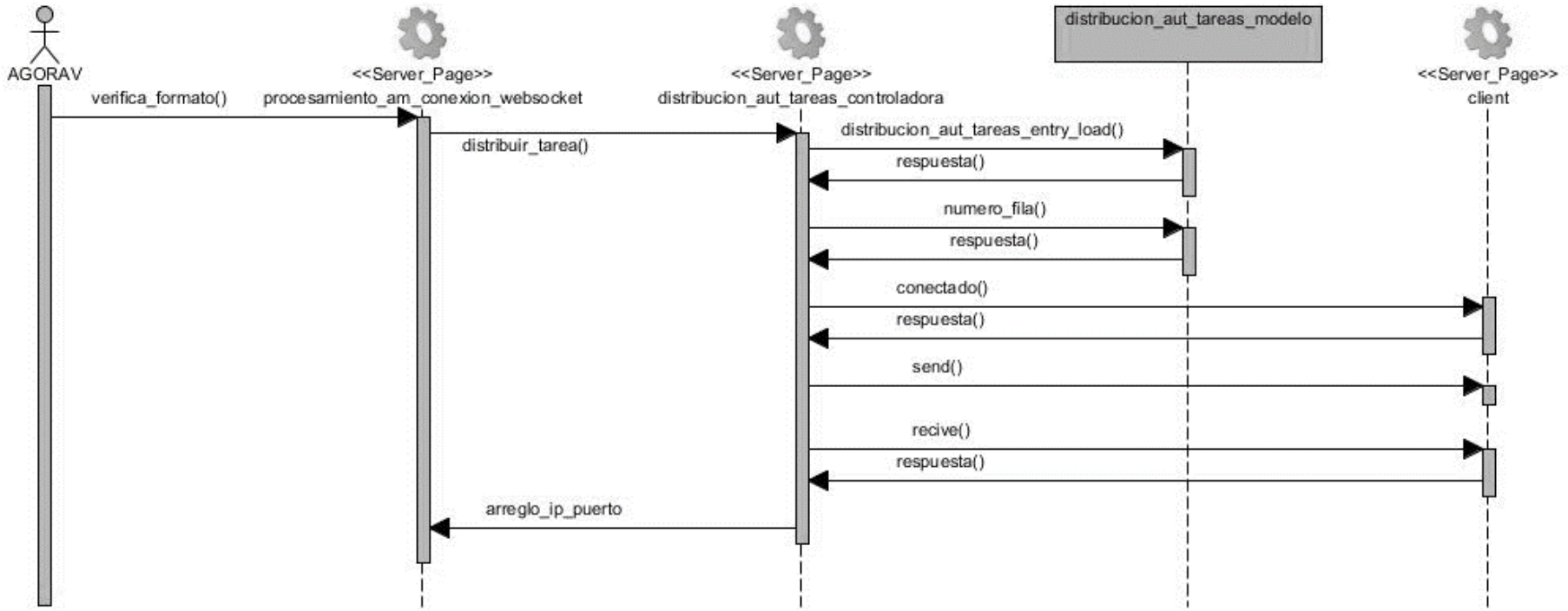


Figura 22. Diagrama de secuencia. Distribuir tarea.

IV. Diagramas de clases del diseño.

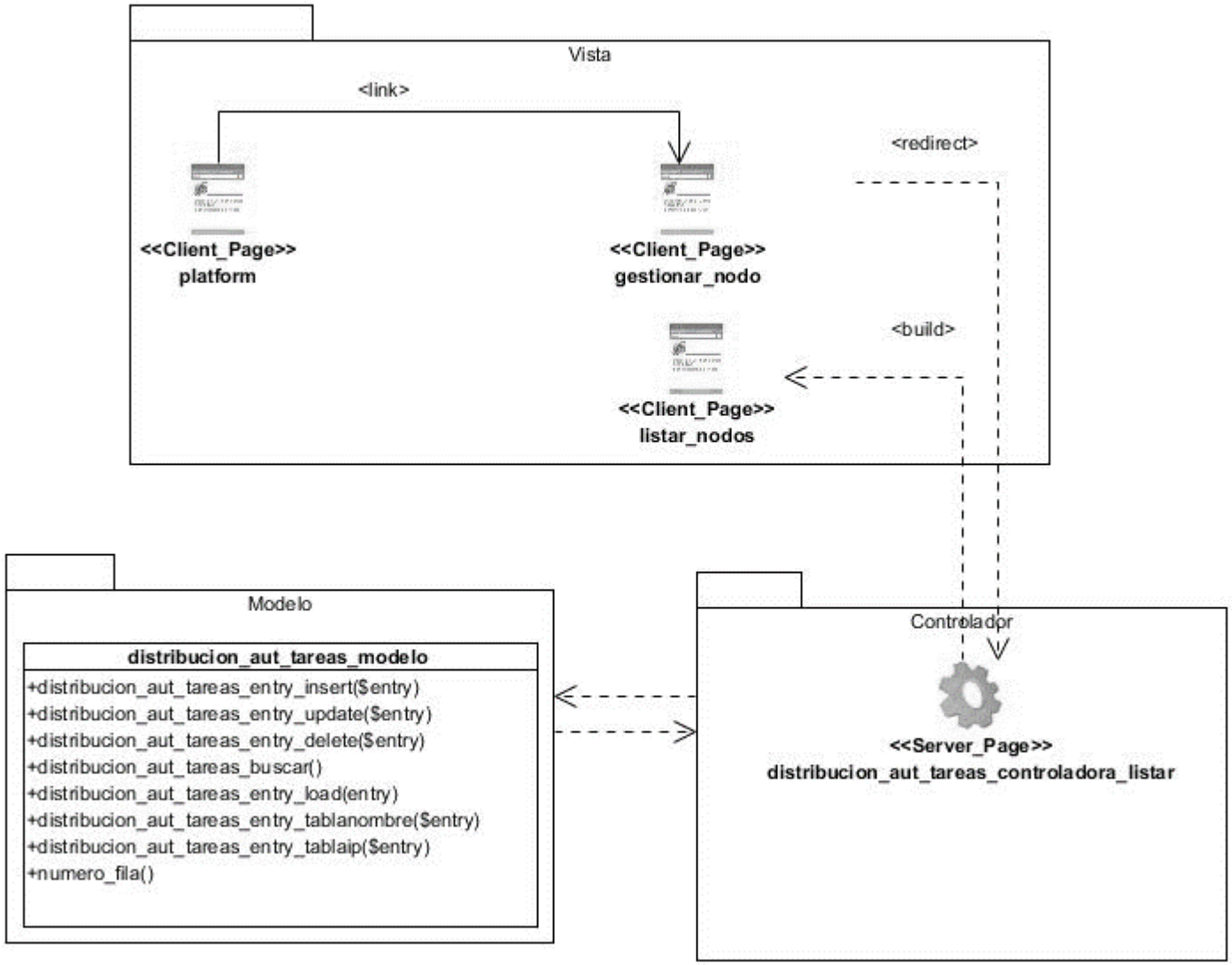


Figura 23. Diagrama de clases del diseño. Mostrar estado de cada nodo.

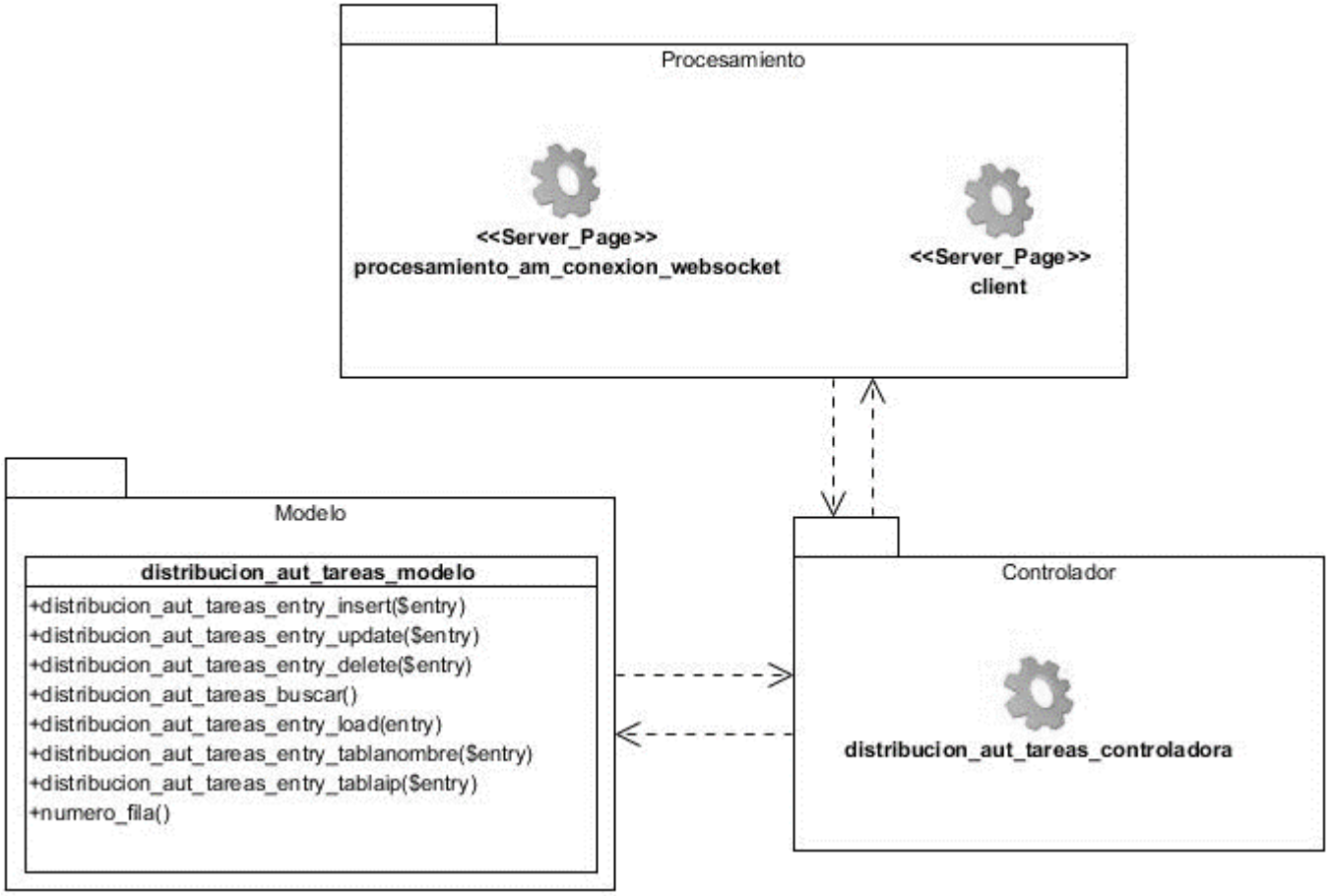


Figura 24. Diagrama de clases del diseño. Distribuir tarea.