UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Centro de Ideoinformática.

FACULTAD 1

TÍTULO: Sistema basado en servicios web para el análisis de redes sociales.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTORES: Richard Luciano García

Carlos Rafael Isequilla Lima

TUTORES: Lic. Karel Antonio Verdecia Ortiz

Ing. Alfonso Claro Arceo

La Habana, Junio 2014

"Año 56 de la Revolución"

"Siempre que te pregunten si puedes hacer un trabajo, contesta que sí y ponte enseguida a aprender cómo se hace."

Franklin Delano Roosevelt (1882-1945).

DECLARACIÓN DE AUTORÍA

Se declara que somos los únicos autores de este trabajo y autorizamos a la Facultad 1 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2014.

_____	_____
Carlos Rafael Isequilla Lima	Richard Luciano García
_____	_____
Karel Antonio Verdecia Ortiz	Alfonso Claro Arceo

DATOS DE CONTACTO

Tutor: Lic. Karel Antonio Verdecia Ortiz. Lic. en Ciencias de la Computación. Graduado en la Universidad de Oriente en el 2004. Durante su trabajo en la Universidad de las Ciencias Informáticas ha participado en proyectos productivos de programación web, minería de texto y análisis de redes sociales.
Correo electrónico: kverdecia@uci.cu . Teléfono: 835-8823

Tutor: Ing. Alfonso Claro Arceo. Ingeniero en Ciencias Informáticas. Graduado en la Universidad de las Ciencias Informáticas en el 2007. Durante su trabajo en la Universidad de las Ciencias Informáticas ha participado en proyectos productivos de genética médica.
Correo electrónico: aclaro@uci.cu

Dedicatoria

A mi mamá y a mi papá por su amor, dedicación y confianza, por su paciencia y consejos para ustedes es este regalo, los admiro con todo mi corazón

A mi hermano para que le sirva de ejemplo y siga mis pasos y se convierta en un hombre de bien. Y a mi familia en general por confiar en mí, les dedico este momento.

Carlos Rafael Isequilla Lima

A mi abuelo aunque hoy no esté presente entre nosotros, por haber sido la fuerza que me llevó a terminar la carrera.

A toda mi familia por su amor, cariño, confianza, preocupación y aliento en todos los momentos de mi vida.

Richard Luciano García.

Agradecimientos

Primeramente quiero agradecerle a mi madre por estar incondicionalmente en los buenos y malos momentos de mi vida.

A mi padre por ser un ejemplo de fuerza y consagración en la vida.

A mi hermano por ser tan competitivo y espero ser un ejemplo para él y ojalá cumpla el sueño de su vida como yo estoy cumpliendo el mío ahora.

A mi abuela Mercedes por darme todo su amor sin dudar ni un minuto.

A mi familia en general por su apoyo incondicional.

A mi Lau por dejarme entrar en su corazón y espero nunca salir, porque en estos pocos meses se ha convertido en la luz de mis ojos.

A mi compañero de tesis Richard que sin él no estuviera parado aquí en este momento.

A mis tutores por sus consejos precisos y consideraciones.

A mis amigos de toda la vida la gente de mi barrio, Harold, Javier etc.

A mis amigos de acá de la UCI por estar presente en buenas batallas en todos los años de mi carrera.

A mis suegros que ya son mi familia y que los quiero.

Carlos Rafael Isequilla Lima

A mi mamá por siempre estar para mí, dándome amor y confianza.

A mi papá por siempre preocuparse por mí y estar al tanto de las cosas de la escuela.

A mis abuelas por su comprensión, cariño y por darme ánimos para seguir adelante.

Al resto de mi familia por estar siempre ahí para mí.

A mi tía Margarita por siempre estar atenta y preocupada por mí.

A Carlos Augusto, Yohanna, Esperanza y Carlitos por hacerme parte de su familia.

A mi familia habanera Noa, Ada, Laura y Sofía por su cariño.

A Ale y su familia por quererme como a uno más de la suya

A mis tutores por su esfuerzo, dedicación y colaboración.

A mi tutora Aylín por su ayuda incondicional y su preocupación.

A todos los que de una manera u otra han contribuido con mi formación.

Richard Luciano García.

Resumen.

Las redes sociales constituyen una plataforma virtual para la interacción humana, permitiendo la rápida difusión de información de cualquier índole. El análisis de las redes sociales está teniendo un gran protagonismo a nivel mundial. En la Universidad de las Ciencias Informáticas (UCI) se aplica análisis a las redes sociales en internet para determinar el comportamiento de los usuarios, pero en esta institución no se realiza otro tipo de análisis, por lo que surgió la necesidad de desarrollar un sistema que mediante una serie de métricas de Centralidad, Cercanía, Intermediación, brinde otra información relacionada al análisis de redes sociales.

Desde la perspectiva anterior, este trabajo de diploma tiene como objetivo desarrollar un sistema basado en servicios web para favorecer la aplicación del análisis de redes en la Universidad de Ciencias Informáticas. El mismo permite gestionar nodos, aristas, comunidades y redes, calcular métricas, detectar comunidades y mostrar una visualización de la red que se analiza, con las comunidades que la componen diferenciadas por diferentes colores. Además otras aplicaciones podrán interactuar con este sistema ya que ofrece una capa de servicios web para gestionar redes, nodos, aristas y comunidades, calcular métricas y detectar comunidades. La significación práctica de este trabajo consiste en un sistema basado en servicios web que permita realizar análisis de redes sociales.

El objetivo de este documento es recoger todo el análisis teórico así como los artefactos generados por la metodología de desarrollo *OpenUp* y modelado a través del Lenguaje Unificado de Modelado (UML). Para realizar la implementación del sistema, se escogió como sistema gestor de base de datos *PostgreSql*, lenguaje de programación *Python*, utilizando el *framework Django*, y como entorno de desarrollo integrado Eclipse.

Palabras clave: análisis de redes sociales, redes sociales, métricas de centralidad

Abstract.

Social networks are a virtual platform for human interaction, which allow a fast diffusion of information of any kind. The social networks analysis in this moment is a very important issue for everyone worldwide. To determine the behaviour of its members in social networks many organizations performs social networks' analytics. The University of Informatics Sciences is one of these organizations however the social network analysis performed is not enough. Therefore there is a need to develop a web system that used Centrality, Closeness and Intermediation metrics to analyse what happens in social networks. To avoid this situation a new web system was elaborated. This research is mainly aimed at developing a system based on web services so as to define graphs and make analysis upon them. The social network analysis system provides a set of web services to allow other applications to interact with the information it handles. The system displays the analysed network and also assigns colours to the communities that are discovered on it. Communities are determined by metrics included in the systems. The purpose of this document is to collect information about the theoretical framework and artefacts generated by OpenUp methodology and modeled throughout the Unified Modelling Language (UML) making use of the case tool Visual Paradigm. PostgreSql was chosen for its implementation as a database managing system. Python was used as a programming language. Django web framework was also applied and Eclipse was used as integrated development environment. The practical significance of this work consists of the implementation of a system based on web services so as to define graphs and make analysis upon them.

Keywords: *social network analysis, social networks, centrality metrics*

ÍNDICE

Introducción.

"El desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) ha sobrepasado los estrechos marcos sectoriales de ingenieros informáticos, cibernéticos, programadores e investigadores para invadir la vida cotidiana de millones de personas en todos los ámbitos del saber humano, revelando la ineludible interrelación entre ciencia-tecnología-sociedad, convirtiéndose a su vez en un tema de gran impacto en la cultura global" [1].

Los avances científico-técnicos y el desarrollo de las telecomunicaciones en el mundo contemporáneo conducen al incremento de la comunicación internacional y al procesamiento de grandes volúmenes de información en diferentes modalidades lo que ha provocado un aumento de las relaciones personales a través de diferentes aplicaciones informáticas. Con herramientas tales como: el *e-mail*, *blogs*, *microblogs*, *wikis*, *Facebook*, *Twitter*, *Google+,* los usuarios crean un espacio virtual para la interacción mediante el cual envían opiniones, mensajes personales o públicos, comentan post con sus experiencias o contribuyen al conocimiento de la comunidad para desarrollar asociaciones, facilitar ayuda y promover la cultura general.

La creación de las redes sociales constituye una plataforma para la interacción humana, permitiendo la rápida difusión de información, como respuesta a prioridades nacionales entre las que se pueden mencionar: la ocurrencia de eventos climatológicos o desastres naturales, denuncias de fraudes electorales, abusos, convocatoria a protestas sociales convirtiéndose en una vía efectiva para la movilización social. Esto ha propiciado que se desarrolle una disciplina de la ciencia llamada análisis de redes sociales.

"El análisis de redes sociales ha pasado de ser una metáfora sugerente para fundamentar un enfoque analítico y un paradigma, con sus principios teóricos, métodos de software y líneas de investigación propios. Los analistas estudian la influencia del todo en las partes y viceversa, el efecto producido por la acción selectiva de los individuos en la red; desde la estructura hasta la relación y el individuo, desde el comportamiento hasta la actitud" [2].

En Cuba desde hace varios años se han realizado una serie de estudios aplicando el análisis de redes sociales. Los mismos han consistido en el análisis de redes sociales para la determinación de relaciones de colaboración/conocimiento en algunas de las principales revistas científico-técnicas del país cubiertas por *SciELO*, *Scopus* y *Web of Sciences*. En las mismas se han aplicado algunas métricas para el análisis de redes sociales.

En el Centro de Ideoinformática (CIDI), de la facultad 1 de la Universidad de las Ciencias Informáticas (UCI) específicamente en el Departamento de Operaciones *Web* y Análisis de Información se aplica análisis a las redes sociales en internet para determinar el comportamiento de los usuarios, así como la actividad de los mismos en cuanto a la publicación de contenido, la frecuencia y la intensidad con que utilizan determinada red social de internet (por ejemplo el número de mensajes publicados), así como su tipo de uso (publicación de contenido propio, interacción, en el sentido de reenviar/compartir el de otros usuarios, diálogo o comentario sobre él de éstos). Lo que le permite no sólo analizar la identidad de un usuario/entidad sino su impacto, influencia y reputación *online*. Este análisis se hace mediante herramientas que permiten su uso gratis donde se necesita de internet y la automatización del proceso. Pero no se constató que en esta institución se realizara otro tipo de análisis a las redes sociales por lo que se desaprovecha el conocimiento de información que pudiera ser valiosa a la hora de tomar decisiones importantes en esta casa de altos estudios. Esto dificulta obtener resultados como:

- Seguir y comparar la dinámica de las comunidades y la influencia de su contribución individual.
- Encontrar una guía en el análisis de las redes sociales para la promoción de los productos que se desarrollan y servicios que se prestan en la universidad.
- Ofrecer información social abundante que apoye la toma de decisiones importantes en dicha institución empleando el análisis de la red social creada a partir del correo electrónico de la universidad.
- Determinar las relaciones de colaboración que se establecen dentro y fuera de su estructura organizativa.
- Identificar los autores más prominentes en materia de cooperación, intercambio de información/conocimiento y las relaciones de colaboración más frecuentes.
- Obtener las relaciones de colaboración entre sus miembros para generar y transferir conocimientos en la organización, como vía para contribuir al desarrollo de nuevos productos o servicios, y perfeccionar su competitividad a nivel institucional, social, nacional e internacional.

La situación antes expuesta permite plantear como **problema de investigación:** ¿Cómo contribuir a la aplicación del análisis de redes sociales en la Universidad de las Ciencias Informáticas?

Según el problema identificado el **objeto de estudio** se enmarca en**:** El análisis de redes sociales y se define como **campo de acción:** el análisis de redes sociales en la Universidad de las Ciencias Informáticas.

Se plantea como **objetivo general**: desarrollar un sistema basado en servicios web para contribuir a la aplicación del análisis de redes sociales en la Universidad de Ciencias Informáticas.

Se proponen como **objetivos específicos:**

1. Realizar la fundamentación teórica del análisis de redes sociales.
2. Realizar el diseño del sistema basado en servicios web para el análisis de redes sociales.
3. Implementar el sistema basado en servicios web para el análisis de redes sociales.
4. Realizar pruebas al sistema basado en servicios web para el análisis de redes sociales.

Definiendo las siguientes **tareas de la investigación:**

1. Fundamentación teórica del análisis de redes sociales.
2. Selección de algoritmos y bibliotecas que se utilizarán en el sistema basado en servicios web para el análisis de redes sociales.
3. Diseño del sistema basado en servicios web para el análisis de redes sociales.
4. Implementación del sistema basado en servicios web para el análisis de redes sociales.
5. Realización de pruebas al sistema basado en servicios web para el análisis de redes sociales.

Como **posible resultado** se espera un sistema basado en servicios web que permita definir grafos y una interfaz web de administración para el análisis de redes sociales.

Se asume la dialéctica-materialista como método general de la ciencia, que permite abordar el objeto en su desarrollo histórico-lógico, para lo cual se emplearon métodos del nivel teórico.

Métodos del nivel teórico

Histórico-lógico: se utilizó para el estudio de la trayectoria histórica y evolución del análisis de las redes sociales. Lo que permitió elaborar una sucesión cronológica y argumentada que facilitó el estudio del objeto en el presente trabajo de diploma. Apoyándose en la lógica de todo el proceso que permitió definir la línea de investigación según los antecedentes, características, necesidades, carencias y dificultades de los

productos analizados los cuales motivaron la realización de esta aplicación y tener un correcto orden en los epígrafes que componen esta investigación.

Analítico-sintético: la utilización de este método posibilitó extraer e identificar conceptos, características, expectativas, necesidades y otros elementos mediante la revisión bibliográfica que posteriormente ayudaron a establecer una propuesta adecuada a las necesidades de implementación del sistema. Después del análisis de la información fue necesario su organización y síntesis para la elaboración del presente trabajo de diploma.

Modelación: se utilizó para modelar el sistema basado en servicios web lo cual permitió definir grafos y una interfaz web de administración para el análisis de redes sociales a través de la representación gráfica de los contenidos tales como: modelo de dominio, modelo de casos de uso, modelo de datos, diagramas de clases y diagramas de secuencia, utilizando las herramientas necesarias que permitieran el establecimiento de las relaciones entre los elementos del problema y el objeto de estudio.

Estructura del trabajo de diploma.

El trabajo de diploma está estructurado de la siguiente forma: introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, glosario de términos y anexos.

Capítulo 1: Fundamentación teórica del sistema basado en servicios web para el análisis de redes sociales.

En este capítulo, se aborda el marco teórico conceptual relacionado con la investigación. Se realiza un análisis de los antecedentes que propician el surgimiento del análisis de redes sociales. Se hace un estudio de las soluciones homólogas existentes. Se definen tanto las herramientas, como las tecnologías y la metodología a utilizar para la construcción del sistema basado en servicios web para el análisis de redes sociales.

Capítulo 2: Diseño del sistema basado en servicios web para el análisis de redes sociales.

En este capítulo se aborda la propuesta de solución a la problemática planteada. Se describen los procesos asociados al negocio a partir de un modelo de dominio y los principales conceptos que estos engloban, para lograr una mejor comprensión tanto del sistema como del contexto donde se desarrolla el mismo. Se describen los requisitos funcionales y no funcionales, para dar cumplimiento a los objetivos planteados. A partir de la derivación de los requisitos funcionales se representa un diagrama de casos de uso, su descripción y la interacción con los actores. Se define el estilo y los patrones

arquitectónicos a utilizar. Se precisan los patrones de diseño. Se realizan los diagramas de clases y de secuencia.

Capítulo 3: Implementación y prueba del sistema basado en servicios web para el análisis de redes sociales.

Se realiza el modelo de componentes. Se describe el estilo de código utilizado. Además se realiza el modelo de despliegue. Se definen los niveles de pruebas. Se diseñan y aplican las pruebas inherentes al sistema para comprobar el correcto funcionamiento del sistema basado en servicios web para el análisis de redes sociales y se ofrecen los resultados de las mismas.

Capítulo 1: Fundamentación teórica del sistema basado en servicios web para el análisis de redes sociales.

En este capítulo, se aborda el marco teórico conceptual relacionado con la investigación. Se realiza un análisis de los antecedentes que propician el surgimiento del análisis de redes sociales. Se hace un estudio de las soluciones homólogas existentes. Se definen tanto las herramientas, como las tecnologías y la metodología a utilizar para la construcción del sistema basado en servicios web para el análisis de redes sociales.

1.1 Antecedentes del análisis de redes sociales.

Los predecesores de las redes sociales en el siglo XVIII incluyen a Émile Durkheim y a Ferdinand Tönnies. Este último expresó que los grupos sociales pueden existir como lazos sociales personales y directos que relacionan a las personas con aquellas con quienes comparten creencias, valores; o bien como relaciones sociales formales e instrumentales. Émile Durkheim dio una explicación no individualista al hecho social expresando que los fenómenos sociales nacen cuando las personas que interactúan forman parte de una realidad que ya no puede explicarse en base a las características de los actores individuales. A principios del siglo XX Georg Simmel pensó concisamente en términos de red social. Sus escritos versaban sobre la naturaleza del tamaño de la red sobre la interacción y la probabilidad de interacción en redes ramificadas, de punto flojo, en lugar de grupos.

En 1929 el escritor húngaro Frigyes Karinthy en una corta historia llamada Chains propuso la Teoría de los Seis grados de separación. La misma plantea que todas las personas del planeta están conectadas a través de no más de seis personas. Las redes sociales parten de esta teoría que se fundamenta en el planteamiento de que el número de conocidos crece exponencialmente con el número de enlaces de la cadena, y solo un pequeño número de enlaces son necesarios para el conjunto de conocidos se convierta en la población humana entera.

Luego de pasadas las primeras décadas del siglo XX, nacieron tres corrientes principales en las redes sociales. En los años 1930 Jacob L. Moreno fue pionero en el registro sistemático y en el análisis de la interacción social de pequeños grupos, en especial dentro de la rama de la sociometría, y otro grupo de autores pertenecientes a la universidad de Harvard liderado por W. Lloyd Warner y Elton Mayo examinaron las relaciones interpersonales en el trabajo. Sobre la década de los 40 del mismo siglo los antropólogos británicos A.R. Radcliffe-Brown instaron al estudio sistemático de las redes, pero tomó unos 15 años antes que sus estudios se siguieran de forma sistemática.

En los años 1950 en Inglaterra se llevaron a cabo estudios de parentesco por Elizabeth Bott y estudios de urbanización por parte de un grupo de antropólogos de la Universidad de Manchester con lo que se desarrolló el análisis de redes sociales. Durante ese tiempo el antropólogo británico SF Nadel codificó una teoría de la estructura social que más tarde se aplicó en el análisis de redes.

Varios académicos investigaron entre los años 1960 y 1970 la combinación de diferentes temas y tradiciones. Entre ellos se puede citar a Harrison White y estudiantes del Departamento de Relaciones Sociales de la Universidad de Harvard. Además otros investigadores fueron Charles Tilly quien se centró en la sociología política y movimientos sociales y Stanley Milgram quien desarrolló la tesis de los seis grados de separación.

1.2 Conceptos fundamentales.

1.2.1 Análisis de Redes Sociales.

"El Análisis de Redes Sociales (*Social Network Analysis,* SNA por sus siglas en inglés) es un análisis metódico de las redes sociales. El SNA observa las relaciones sociales en términos de la teoría de redes, consistiendo en nodos (los cuales representan a cada actor de la red) y lazos (que representan las relaciones entre los individuos, tales como la amistad, el parentesco, la posición de la organización, las relaciones sexuales, etc.). Estas redes son a menudo representadas en un diagrama de red social, donde los nodos se representan como puntos y los lazos se representan como líneas" [3].

"El análisis de redes sociales es la representación visual de las relaciones sociales en términos de la teoría de redes consistente en nodos y aristas" [4]. Los nodos representan a los actores individuales en las redes, y las aristas las relaciones entre ellos. A menudo el grafo resultante es una estructura bastante compleja. Investigaciones en distintos campos académicos han mostrado que las redes sociales operan en muchos niveles, desde los familiares hasta niveles nacionales, y juegan un papel crítico en la determinación del camino a seguir en la solución a problemas, en la forma de organizar empresas, y el grado en que un individuo tiene éxito en lograr sus objetivos.

A partir del análisis de las definiciones anteriores se establece como hilo común que el análisis de redes sociales es el análisis de las redes sociales en términos de la teoría de redes consistiendo en nodos que representan los actores y aristas las que representan las relaciones o vínculos entre individuos.

1.2.2 Grafo.

"En matemáticas, un grafo es una representación abstracta de un conjunto de objetos donde los pares de objetos son conectados por líneas. Los objetos interconectados son

representados por abstracciones matemáticas llamados nodos y las líneas que conectan los pares de nodos son llamados aristas" [4]. Los grafos son representados de manera esquemática como un conjunto de puntos para los nodos unidos por líneas o curvas para las aristas. Los grafos son parte del objeto de estudio de las matemáticas discretas.

En su libro Estructuras de Datos en C los autores Aaron M. Tanenbaum, Yedidyah Langsam, M. J. A plantean que "un grafo consta de un conjunto de nodos (o vértices) y un conjunto de arcos (o aristas). Cada arco de un grafo se especifica mediante un par de nodos. Si el par de nodos que constituyen el arco son pares ordenados, se dice que el grafo es n grafo dirigido (o dígrafo). Un aspecto a tener en cuenta es que un nodo no necesita tener ningún arco asociado al él" [5].

Los autores de esta investigación consideran que el concepto de grafo dado por los autores Aaron M. Tanenbaum, Yedidyah Langsam, M. J. A se ajusta más al tema de investigación y es la representación que se utiliza en el análisis de redes sociales.

1.2.3 Red Social.

"Una red social es una estructura social hecha por individuos u organizaciones llamados nodos" [4]. Los nodos (que a menudo representan a los individuos) están conectados por uno o más tipos específicos de relaciones.

"Una red social es un conjunto de actores vinculados entre sí" [6]. Los actores pueden ser personas o grupos de estas: empresas, comunidades, organizaciones, países, ciudades, etc. Los vínculos son cualquier cosa que relacione a los actores, por ejemplo: amor, poder, alianzas, amistad, parentesco familiar, contacto por correo electrónico, creencias religiosas comunes, rivalidad, conocimiento o prestigio, etc.

El concepto red social en el ámbito de internet se define como: "páginas que permiten a las personas conectar con sus conocidos, familiares y amigos, incluso realizar nuevas amistades, a fin de compartir contenidos, interactuar, crear comunidades sobre intereses similares: trabajo, lecturas, juegos, amistad, relaciones interpersonales" [7].

Un conjunto de personas que se relacionen entre sí de manera física y sin la intervención de tecnologías es también una red social. Las redes sociales se pueden representar como una red o grafo. Naturalmente, los vínculos pueden ser aristas o arcos (con dirección), y pueden tener uno o más pesos.

1.2.4 Comunidades.

"Las grandes redes son una compleja combinación de pequeñas comunidades (también llamados clústeres, grupos o subgrafos). Las comunidades son focos de vértices densamente conectados que están escasamente conectados a otros focos" [8]. Identificar

comunidades dentro de una red y acotar sus relaciones con otras puede ser esencial para la toma de decisiones estratégicas. El análisis de redes puede ayudar a identificar comunidades competentes o complementarias, alianzas potenciales para formar una comunidad, y los individuos que se pueden conectar a ella.

1.2.5 Métricas.

Métrica: "Es una medida efectuada sobre algún aspecto del sistema en desarrollo o del proceso empleado que permite, previa comparación con unos valores (medidas) de referencia, obtener conclusiones sobre el aspecto medido con el fin de adoptar las decisiones necesarias" [9].

Métricas de Centralidad: "se refiere a un grupo de métricas que cuantifican la importancia o influencia de un nodo en particular o grupo dentro de una red" [8]. Las métricas de centralidad incluyen: cercanía, centralidad de vector propio, grado de centralidad.

Cercanía: "El grado en que un persona está cerca de todas las demás en una red (directa o indirectamente). Refleja la capacidad de acceder a la información a través de los miembros de la red. Así, la cercanía es la inversa de la suma de las distancias más cortas entre cada individuo y cada una de las otras personas en la red" [4]. En la teoría de grafos la cercanía es una métrica de centralidad de un vértice dentro de un grafo. Los vértices que tienden a tener "distancias geodésicas[1]" más cortas con respecto a otros vértices en el grafo tienen mayor cercanía. La cercanía preferentemente en el análisis de redes significa la longitud del camino más corto, el cual proporciona valores más altos al vértice central y por tanto es positivamente asociado con otras medidas de ese mismo grado.

Centralidad de Vector Propio: "Una medida de la importancia de un nodo en una red. Asigna puntuaciones relativas a todos los nodos de la red basadas en el principio de que las conexiones a los nodos que tienen una puntuación más alta, contribuyen más a la puntuación del nodo en cuestión" [4]. Es una vista más sofisticada de la centralidad: una persona con pocas conexiones pudiera tener un alto vector de centralidad si esas pocas conexiones estuvieran muy bien conectadas a sí mismas. El vector propio de centralidad permite tener para las conexiones un valor variable de manera tal que conectando algunos vértices se obtengan mayores beneficios que conectando otros. El algoritmo *PageRank* utilizado por Google en su motor de búsqueda es una variante del Vector Propio de Centralidad.

[1] El camino más corto entre dos puntos es llamado distancia geodésica.

Grado de Centralidad: "es la primera y más simple de las medidas de centralidad y es definida como el número de enlaces que tiene un nodo con los demás" [4]. El grado es con frecuencia interpretado en términos del riesgo inmediato de un nodo para adoptar cualquier corriente a través de la red (ejemplo un virus, o cualquier información). Si el grafo es dirigido (significa que las conexiones tienen una dirección), entonces se definen dos métricas diferentes de grado de centralidad llamadas grado de entrada y grado de salida.

Grado de entrada: Está definido por el número de aristas incidentes hacia él. En las relaciones como amistad o asesoramiento demuestra el grado de popularidad.

Grado de salida: Está definido por el número de aristas que son incidentes desde él. En las relaciones como amistad o asesoramiento demuestra el grado de sociabilidad.

Intermediación: "El grado en que un nodo contribuye a la suma del flujo máximo entre todos los pares de nodos (excluyendo ese nodo). La intermediación es una medida de centralidad de un vértice dentro del grafo" [4]. Los vértices que ocurren en muchos de los pasos más cortos entre otros vértices tienen mayor intermediación que otros vértices que no lo hacen. La intermediación es la extensión que media entre otros nodos en la red. Esta medida toma en consideración la conectividad de los nodos vecinos proporcionándole un mayor valor a los nodos que se conectan con un grupo de clústeres. Esta medida refleja el número de personas las cuales están conectadas indirectamente a través de vínculos directos.

Rango de Páginas (PageRank): asigna valores comprendidos en el rango de 1 a 10 a la relevancia de los documentos o páginas web que se indexan por un motor de búsqueda. Este algoritmo se basa en la vasta estructura de enlaces de la web como un indicador del valor de una página en concreto. Google interpreta un enlace de una página **A** a una página **B** como un voto de **A** hacia **B**. también se tiene en cuenta el volumen de votos y los enlaces que una página recibe, así como la página que emite el voto. Las páginas consideradas más importantes, es decir con un *PageRank* elevado, emiten un voto mayor y que ayuda a hacer otras páginas más importantes. Por tanto el *PageRank* de una página refleja la importancia que esta tiene en Internet.

1.2.6 Servicios web

Durante el desarrollo de aplicaciones informáticas se han utilizado varias alternativas para comunicar sistemas logrando cierta interoperabilidad. Por lo que se define como:

Servicio web (*Web Services* en inglés): "colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de *software*

desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet o Intranet. La interoperabilidad se consigue mediante la adopción de estándares abiertos" [10].

Los servicios web son independientes de la plataforma en la que corran dos aplicaciones que se comunican entre sí, tributan independencia entre la aplicación que brinda el servicio y la que lo consume, lo que permite establecer enlaces entre nuevas implementaciones con otras anteriores, sin que la tecnología de los más antiguos afecte a los más modernos, así los cambios en una no afecta a la otra, lo que aporta una flexibilidad importante.

1.3 Estudio de herramientas y bibliotecas utilizadas en el análisis de redes sociales.

EgoNet Active Development or Explanation.

EgoNet es un software para la recolección y análisis de datos de la red egocéntrica. Contiene asistentes para crear cuestionarios, recoger datos y proporciona medidas generales de la red y matrices de datos que se pueden utilizar en el análisis por otros programas de *software*.

Gephi

Gephi es un software que permite la manipulación y exploración de grafos. *Gephi* es una plataforma interactiva de visualización y exploración para todos tipos de redes y sistemas complejos, grafos dinámicos y jerárquicos. Es una herramienta para explorar y comprender los grafos donde el usuario interactúa con la representación, manipulación de las estructuras, formas y colores para revelar las propiedades ocultas. Utiliza un motor de renderizado 3D para visualizar grandes redes en tiempo real y acelerar el proceso de exploración. Tiene una arquitectura flexible y multitarea que brinda nuevas posibilidades para el trabajo con colecciones de datos complejos y producir resultados visuales valiosos. *Gephi Toolkit* es una librería estándar de Java que compone el núcleo de *Gephi* y se puede importar desde cualquier proyecto en Java. *Gephi Toolkit* recibe un archivo gexf, le aplica un tipo de *layout* y retorna el resultado en un archivo gexf, exporta un archivo de grafo a (*portable document format*, pdf), tiene módulos para almacenamiento, filtrado, búsqueda, análisis de redes sociales y visualización de grafos.

GraphStream

GraphStream es una biblioteca de grafos estática y dinámica. Esta biblioteca permite crear, visualizar y renderizar grafos comenzando desde cero, cargarlos de un archivo o cualquier otra fuente.

Graph-tool

Graph-tool es una biblioteca de *Python* para el análisis eficiente y la visualización de grafos. El núcleo de las estructuras de datos y los algoritmos son implementados en C++, con un amplio uso de plantillas de metaprogramación, basado en la librería *Boost Graph*. Contiene una lista de algoritmos comprensibles.

Graphviz

Graphviz constituye un *framework* de visualización de grafos de código abierto. Contiene algunos programas con capas para grafos, adecuados para la visualización de redes sociales.

Java Universal Network/Graph (*JUNG*) Framework

*Java Universal Network/Graph (*JUNG*)* permite el análisis, manipulación y visualización de grafos. JUNG es una API en Java que provee en lenguaje común y extensible para la modelación, análisis y visualización de datos relacionales. Admite variedad de grafos (incluyendo los hipergrafos), elementos de grafos de cualquier tipo y propiedades, permitiendo visualizaciones personalizadas e incluye algoritmos de la teoría de grafos, minería de datos, y análisis de redes sociales (ejemplo *clustering*, descomposición, optimización, generación aleatoria de grafo, análisis estadístico, distancia, flujo y centralidad. Ha sido utilizado para analizar redes que superan el millón de nodos y es limitada solo por la cantidad de memoria asignada a Java.

***Network Overview Discovery Exploration for Excel* (NodeXL)**

NodeXL es un *plugin* libre y de código abierto para Excel 2007 implementado en C#.Net para el análisis y visualización de redes. Se integra a Excel 2007 y 2010 y añade grafos dirigidos como tabla en la hoja de cálculo y calcula el conjunto de métricas y mantiene el resultado en memoria. Admite extraer *e-mail*, *Twitter*, *YouTube*, *WWW* y proyectar redes sociales. Acepta la representación de grafos por listas y matrices de adyacencia. Permite la manipulación y filtrado de los datos subyacentes en formato de hoja de cálculo, múltiples diseños de visualización de redes, la lectura y escritura de archivos en formato *UCINet* y *GraphML*.

Networkx

Networkx es una biblioteca desarrollada en *Python* que cuenta con un conjunto de herramientas integradas para la creación, manipulación, análisis, visualización, estudios de estructuras, la dinámica y funciones complejas de redes. La interfaz de usuario es a través de secuencias de comandos en la consola proporcionada por *Python*. Incluye grandes conjuntos de algoritmos claves, métricas y generadores de grafos. La visualización se realiza a través de pylab y *Graphviz*. Es un proyecto de código abierto en desarrollo activo desde el 2004, por *Los Alamos National Lab*. Por las características que presentadas por esta biblioteca y las facilidades que ofrece se decide su uso como base para el desarrollo de la aplicación.

R

R es un entorno versátil y popular que contiene varios paquetes relevantes para el análisis de redes sociales entre ellos: *igraph* que es un paquete de análisis de redes genéricas; *sna* realiza análisis sociométrico de redes, manipula y visualiza redes; *tnet* realiza análisis de redes pesadas, redes de dos modos y longitudinales; *ergm* implementa modelos de grafos aleatorios y exponenciales para redes; *latentnet* funciona para redes de posicionamiento latente y modelos de conglomerados; *degreenet* provee herramientas para el modelado estadístico de las redes distribución de niveles; y *networksis* provee herramientas para la simulación bipartita de redes con los marginales fijos.

Sigmajs

Sigmajs es una biblioteca de JavaScript dedicada a la visualización de grafos. La misma permite publicar grafos en las páginas web de manera muy sencilla y proporciona a los desarrolladores integrar exploración de grafos en aplicaciones web. Provee muchas características integradas como renderizado *Canvas* y *WebGL*, soporte para *mouse* y *touch* lo que proporciona la manipulación de las páginas *web* de forma más rápida para el usuario. *Sigmajs* permite diferentes opciones para personalizar, visualizar e interactuar con los grafos así como implementar funciones o *scripts* para renderizar los nodos y las aristas de la forma exacta que el desarrollador desee. *Sigmajs* posee además un motor de renderizado y una API que hace posible modificar los datos, mover la cámara y actualizar el renderizado. Permite cargar grafos en archivos JSON o GEXF y un *plugin* que se encarga de leer y parsear el archivo.

Conclusiones del estudio de herramientas y bibliotecas más utilizadas para el análisis de redes sociales.

En el estudio realizado anteriormente se identificaron y analizaron una serie de herramientas y bibliotecas que son utilizadas para el análisis de redes sociales. De las herramientas analizadas:

- No son multiplataforma, *Graph-tool* que es una biblioteca que está disponible para sistemas *Linux* y *Mac*, *NodeXL* es una biblioteca para *Microsoft Office* solo está disponible para *Windows*.
- Solo permiten visualización web *Sigmajs*.
- Solo permiten la generación de archivos con el formato .gexf la biblioteca *Networkx*, R y *Gephi*.

Por lo que no se puede seleccionar solo una de ellas para la solución de este trabajo. Por lo que se decide la utilización de la biblioteca *Networkx* para realizar el análisis de redes sociales y exportar un fichero .gexf y la biblioteca *JavaScript Sigmajs* que se encargará de la visualización *web.*

1.4 Análisis de redes sociales en Cuba.

En los últimos años en Cuba se han realizado una serie de estudios en diferentes entidades aplicando el análisis de redes sociales. Entre estos estudios podemos encontrar el realizado por Lic. Rosa Lidia Vega Almeida, Lic. Ricardo Arencibia Jorge, Ing. Juan Antonio Araújo Ruiz para determinar la producción científica de los institutos de salud de Cuba en el *Web of Science* en el período 2000-2004 donde la colaboración y las temáticas de investigación se presentaron mediante grafos basados en el análisis de redes sociales, utilizando el empleo de indicadores de análisis fundamentalmente, el grado de centralidad y la intermediación entre los nodos de cada red.

Otro ejemplo es el estudio realizado Lic. Ricardo Arencibia Jorge enfocado en determinar la visibilidad internacional alcanzada por la revista Acimed ya que la misma se encuentra al alcance internacional por medio de Scopus y cubierta por *SciELO*. Por medio de técnicas de análisis de redes sociales, se representaron todas las redes de coautoría presentes en la producción científica de esta revista, cuyo análisis permite definir los principales grupos de investigación conformados en torno a las temáticas de investigación tratadas en la revista.

En el año 2009 se publicó el estudio El análisis de las redes sociales en la identificación de las relaciones de colaboración: estudio de la *Revista Cubana de Ciencia Agrícola* por las autoras Yeter Caraballo Pérez, Anays Más Basnuevo para determinar las relaciones

de colaboración en dicha revista. Con el objetivo de identificar los autores más prominentes en materia de cooperación e intercambio de información/conocimiento y las relaciones de colaboración más frecuentes. Se desarrolló un análisis de la centralidad, cercanía e intermediación con vista a determinar la posición de cada uno de los actores en el conjunto de la red y se determinaron los egos de esta, es decir, de los individuos clave a la luz de la cooperación y el intercambio en materia de información/conocimiento en la organización objeto de estudio.

Durante la realización de esta investigación no se encontraron resultados diversos del desarrollo de aplicaciones enfocadas a realizar análisis de redes sociales a excepción del sistema Motor de clasificación inteligente de contenidos para correo electrónico (MOCICE) desarrollada en la Universidad de las Ciencias Informáticas (UCI). MOCICE es una herramienta que permite analizar de forma automática flujos de correo electrónico para:

1. Detectar grupos de usuarios con intereses comunes o Comunidades de Interés (CI).
2. Detectar los roles que juegan los usuarios dentro de las CI.
3. Detectar tópicos y los usuarios vinculados a estos tópicos.

1.5 Tecnologías a utilizar.

Para la realización de una aplicación se deben tener en cuenta las tendencias actuales en torno a las herramientas a utilizar para el desarrollo de la misma, de forma que el trabajo se haga más sencillo y a la vez con mayor calidad. Se deben utilizar tecnologías avanzadas y en continuo progreso, para que de esta forma la aplicación cumpla con las expectativas de los usuarios finales y además se pueda actualizar la misma con gran facilidad. Las tecnologías que se describen en este capítulo, son las que serán utilizadas para el desarrollo de esta herramienta.

1.5.1 Metodologías ágiles.

Hoy día las metodologías de desarrollo se clasifican en ágiles y tradicionales o pesadas. Las metodologías tradicionales o pesadas son muy efectivas a la hora de planificar el proceso de desarrollo de software cuando es para un proyecto muy grande y donde se requiere una gran cantidad de tiempo. Mientras que las metodologías ágiles son muy empleadas en proyectos pequeños, con poco personal, de corta duración y de menor magnitud, lo que no quiere decir que se separen de las prácticas esenciales para asegurar la calidad del producto.

***Xtreme Programming* (XP).**

Entre las metodologías clasificadas como ágiles podemos encontrar XP. Esta metodología se enfoca en fomentar las relaciones interpersonales como forma para llegar al éxito en el desarrollo de *software*, propiciando el trabajo en equipo, la superación de los desarrolladores, y un clima de trabajo. XP se fundamenta en una constante retroalimentación y comunicación fluida entre el cliente y el equipo de desarrollo, sencillez en las soluciones y valentía a la hora de enfrentar cambios. Es una metodología muy utilizada cuando el proyecto a desarrollar tiene características como las siguientes:

- Proyectos donde los requisitos son imprecisos y muy cambiantes.
- Proyectos en los que existe un alto riesgo técnico.

XP exige un alto grado de disciplina de los involucrados en el proyecto. Además de que integra como práctica de programación usualmente la orientada a objetos y una gran utilización de patrones de diseño.

***OpenUp*.**

Dentro de las metodologías existentes entre las metodologías ágiles está *OpenUp* que permite mantener la filosofía de RUP. Es una metodología que ofrece un proceso mínimo que no proporciona lineamientos para todos los componentes que se empleen en un proyecto de software, ya que solo tiene los componentes esenciales que puede servir de base a procesos específicos. Los componentes están enfocados a fomentar el intercambio de información entre los involucrados en el proyecto (equipos de desarrollo) para mantener el entendimiento, objetivos y alcance del proyecto, sincronizar intereses, desarrollo evolutivo, retroalimentación y mejoramiento continuo.

OpenUp está centrada en articular la arquitectura para posibilitar la colaboración técnica, minimizar el riesgo y el sobreesfuerzo de desarrollo. Propone un equilibrio entre las necesidades de los implicados con los resultados del proyecto y los costos, con el propósito de maximizar el valor de los involucrados con los resultados y las guías del proceso de desarrollo, por lo que se realiza un ciclo de vida iterativo que reduce el riesgo a tiempo y proporciona resultados en curso al cliente del proyecto.

Fundamentación de la metodología de desarrollo.

Se selecciona como metodología de desarrollo *OpenUp* ya que es apropiado para proyectos pequeños y de bajos recursos. Permite disminuir las probabilidades de fracaso e incrementar las probabilidades de éxito. Detecta errores tempranos a través de un ciclo iterativo. Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.

1.5.2 Sistema Gestor de Base de Datos.

***PostgreSql*.**

"*PostgreSql* es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es un sistema de gestión de bases de datos de código abierto potente que en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. *PostgreSql* utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando" [11].

Su desarrollo comenzó hace más de 16 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. *PostgreSql* funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

A continuación se mencionan algunas de las características más importantes soportadas por *PostgreSql* [11]:

- Es una base de datos 100% ACID
- Integridad referencial
- Replicación asincrónica/sincrónica / *Streaming replication - Hot Standby*
- Copias de seguridad en caliente (*Online/hot backups*)
- Múltiples métodos de autentificación
- Acceso encriptado vía SSL
- Funciones/procedimientos almacenados (*stored procedures*) en numerosos lenguajes de programación, entre otros *PL/pgSQL* (similar al *PL/SQL* de *Oracle*), *PL/Perl*, *PL/Python* y *PL/Tcl.*
- Numerosos tipos de datos y posibilidad de definir nuevos tipos.
- Soporta el almacenamiento de objetos binarios grandes (gráficos, videos, sonido).

***MySql*.**

MySql es un sistema de gestión de bases de datos relacional. Su diseño multihilos le permite soportar una gran carga de información de forma eficiente. Está diseñado para entornos de producción críticos, con alta carga de trabajo, así como para integrarse en aplicaciones para ser distribuido. *MySql* es una marca registrada de *MySql AB*, aunque tiene una doble licencia. Los usuarios pueden elegir entre usar el producto *MySql* como

un sistema *Open Source* bajo los términos de la licencia *GNU* (del inglés *General Public License*) o pueden adquirir una licencia comercial estándar de *MySql AB*.

"Se encuentra programado en *C* y en *C++,* probado con un amplio rango de compiladores diferentes, multiplataforma, proporciona sistemas de almacenamiento transaccionales y no transaccionales. El servidor está disponible como un programa separado para usar en un entorno de red *cliente/servidor*. También está disponible como biblioteca y puede ser incrustado en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible. Los clientes pueden conectar con el servidor *MySql* usando *sockets TCP/IP* en cualquier plataforma" [12].

Fundamentación del sistema gestor de base de datos.

El sistema gestor de base de datos seleccionado es *PostgreSql,* principalmente por ser multiplataforma lo cual permitirá ser utilizado tanto en *Windows* como en *Linux*. Otro aspecto fundamental a tener en cuenta es que es libre, parámetro en el cual lleva cierta ventaja sobre *MySql*, el cual conlleva la necesidad de una licencia para su uso.

1.5.3 Lenguaje de programación *Python*.

La biblioteca seleccionada para realizar análisis de redes sociales es *Networkx*, que es implementada en *Python*, por tanto se selecciona *Python* como lenguaje de programación. Además es un lenguaje de programación de propósito general, dinámico y orientado a objetos cuya expansión y popularidad es relativamente reciente y puede ser usado de muchas maneras en el desarrollo de *software*. Ofrece gran soporte e integración con otros lenguajes y herramientas, contiene una extensa cantidad de bibliotecas y puede ser aprendido en pocos días.

Python es interpretado, lo que quiere decir que no hace falta compilar el código, sólo hay que escribir el programa y ejecutarlo. Es distribuido bajo la licencia *open source OSI* que lo hace libre para ser usado incluso en el desarrollo de productos comerciales. Entre las principales características que posee *Python* se encuentran:

- Propósito general: Se pueden desarrollar cualquier tipo de programas.
- Multiplataforma: Se puede correr en cualquier sistema operativo siempre y cuando esté instalado el intérprete de *Python* programado para él, lo que ofrece una gran ventaja cuando portamos código a diferentes sistemas operativos.
- Interactivo: *Python* tiene un intérprete por línea de comandos en el que se pueden introducir sentencias que se ejecutan y se muestra el resultado. Esto ofrece que se pueda comprender mejor el lenguaje y comprobar los resultados de la ejecución de código rápidamente.

- Funciones y bibliotecas: en el propio lenguaje de programación están incorporadas una serie de funciones como por ejemplo: tratamiento de cadenas, números, archivos, etc., se pueden incorporar bibliotecas que se importan en los programas para temas específicos.
- Sintaxis clara: Tiene una sintaxis muy particular ya que para separar segmentos de código en *Python* se debe tabular hacia adentro, colocando un margen al código que estaría dentro de una función o un bucle[2], lo que permite que todos los desarrolladores sigan las mismas notaciones y que los programas desarrollados en este lenguaje luzcan de manera semejante.

1.5.4 Framework de desarrollo web.

Web2py

Web2py es un *framework* para el desarrollo de aplicaciones web funcionales escrito en *Python.* Es *open source* y permite al desarrollador aplicar buenas prácticas de desarrollo de software. En las fases de diseño, implementación y pruebas el desarrollador puede utilizar una serie de bibliotecas que le facilitan el trabajo en cada una de ellas por separado, y las hace trabajar juntas. En la implementación del *framework* se contemplan muchos problemas que pueden convertirse en vulnerabilidades de seguridad por lo que valida las entradas para evitar inyecciones de código, cambia el nombre de los archivos leídos para evitar ataque de directorio transversal, tiene una capa de abstracción que escribe SQL de forma dinámica, todo esto enfocado en la seguridad. La principal desventaja que presenta es la falta de flexibilidad ya que a diferencia de *Django* las aplicaciones de *web2py* no son independientes del *framework* ya que radican dentro del mismo y podría originar problemas a la hora de la integración con otros sistemas.

***Django*.**

Django es un *framework* que permite el desarrollo de aplicaciones web escrito en *Python* que permite ahorrar tiempo, crear y mantener aplicaciones de alta calidad con el menor esfuerzo. Es *open source* y se enfoca en la reutilización, conectividad y extensibilidad de componentes para el desarrollo rápido basados en el principio DRY (del inglés *Don´t Repeat Yourself*) haciendo las cosas una sola vez y rehusándolas en la medida que sea posible. *Python* es utilizado completamente en el *framework* desde los archivos, los modelos de datos hasta los archivos de configuración. Algunas de las características con que cuenta este *framework* son:

[2]Es una sentencia que se realiza repetidas veces a un trozo aislado de código, hasta que la condición asignada a dicho bucle deje de cumplirse.

- Sistema de mapeo de objetos relacionales (ORM): Los modelos se implementan completamente en *Python* proporcionando una API rica y dinámica para el acceso a la base de datos, aunque también permite construir su propio código SQL.
- Interfaz automática de administración: Es un mecanismo simple que permite la creación de interfaces para los administradores donde estos puedan insertar o modificar contenido de su sitio.
- Sistema de plantillas: Implementa un mecanismo de plantillas que permite la separación del diseño, el contenido y el código de *Python.*
- Sistema de cacheo: Para mejorar el rendimiento de la aplicación proporciona un conjunto de APIs para el cacheo de objetos en memoria usando Memcached[3] u otras aplicaciones de cacheo.
- Gran cantidad de aplicaciones reusables y *plugins*: Cuenta con una gran cantidad de aplicaciones y plugins que pueden ser usados en las tareas más comunes.

Selección del *framework* de desarrollo.

Se selecciona el *framework* de desarrollo *Django* ya que mantiene de forma rigurosa un diseño limpio en su propio código y facilita que el programador siga las mejores prácticas de desarrollo *web* en las aplicaciones que crea. Fomenta el bajo acoplamiento, la filosofía de programación que dice que las distintas partes de la aplicación deben ser intercambiables y deben comunicarse unas con otras mediante APIs[4] claras y concisas. *Django* es utilizado por muchas compañías para el desarrollo de sus aplicaciones empresariales por la rica cantidad de características con que cuenta.

1.5.5 Entorno de Desarrollo Integrado (IDE).

NetBeans

NetBeans es un entorno de desarrollo integrado que permite escribir, compilar, ejecutar y corregir errores a programas. Desarrollado en *Java* también tiene soporte para otros lenguajes de programación mediante el soporte nativo (C/C++, Ruby y PHP) y el uso de bibliotecas. Es un producto libre y gratuito que no tiene restricciones de utilización. Permite el desarrollo de aplicaciones web, de escritorio, móviles empleando las plataformas de *Java.* El proyecto *NetBeans* es apoyado por una variada comunidad de desarrolladores y ofrece una amplia documentación y una variada selección de componentes.

[3]Es un sistema distribuido de cacheo de objetos en memoria en la forma clave-valor.

[4]API es una interfaz de programación de aplicaciones.

Eclipse

Eclipse es un entorno de desarrollo integrado que es multiplataforma y *open source.* A diferencia de otros entornos monolíticos que proporcionan todas las funcionalidades las requiera el usuario o no, *Eclipse* emplea plugins lo que permite una plataforma ligera para componentes de software. Puede extenderse a otros lenguajes de programación como son C/C++ y *Python.* Entre las características que presenta ***Eclipse*** se encuentran:

- Dispone de un Editor de texto con resaltado de sintaxis.
- Compilación en tiempo real.
- Tiene pruebas unitarias con *JUnit*, control de versiones con CVS, integración con *Ant,* asistentes (*wizards*) para creación de proyectos, clases*, tests*, etc., y refactorización.
- Asimismo, a través de "*plugins"* libremente disponibles es posible añadir control de versiones con *Subversion* e integración con *Hibernate*.

Integrar Eclipse con Python.

PyDev es una biblioteca para Eclipse que agrega a esta herramienta la potencialidad de un editor *Python, Jython* y *Iron Python* de gran nivel. Se distribuye bajo la licencia *Eclipse Public License*. Es un IDE completo que integra editor de código, depurador e intérprete. Algunas características que presenta son:

- Incorpora un depurador gráfico de *Python*, resaltado de sintaxis, auto completado.
- Permite desarrollar módulos en *Jython*, que es la implementación de *Java* en *Python* y cuyo principal fuerte es el uso de la biblioteca estándar de *Java* en módulos *Python*.
- Estructura los proyectos en directorios, recursos, etc. lo que ayuda a organizarlos y establecer una jerarquía.

Fundamentación del IDE seleccionado.

Se selecciona como IDE desarrollo *Eclipse* con el *plugin PyDev* ya que es multiplataforma, permite la integración con el lenguaje de desarrollo Python, dispone de un editor de código con resaltado de sintaxis, compilación en tiempo real, depurador e intérprete, autocompletado entre otras características.

1.5.6 JavaScript.

JavaScript es lenguaje de programación que se utiliza esencialmente en el desarrollo de páginas *web*. Es interpretado y que no requiere compilación. Está basado en prototipos, es imperativo, débilmente tipado y dinámico. Los navegadores web en sus versiones más

recientes interpretan código *JavaScript*. Permite añadir propiedades interactivas a las páginas web de manera de forma sencilla. Este es un lenguaje de programación que se basa en la utilización de guiones que son integrados directamente al código *HTML* por lo que el código es transferido al cliente para que este lo interprete al cargar la página por lo que si no llena correctamente un formulario no tiene que esperar a que el servidor muestre de nuevo el formulario indicando los errores existentes. Las principales características de este lenguaje son:

- Es un lenguaje interpretado.
- No necesita compilación.
- Multiplataforma.
- Lenguaje de alto nivel.
- Admite programación estructurada.
- Basado en objetos.
- Maneja la mayoría de los eventos que se pueden producir sobre la página *web*.
- No se necesita ningún *kit* o entorno de desarrollo.

1.5.7 Celery

Celery es una aplicación que permite crear tareas de trabajo asíncronas gestionadas por un gestor de colas que está basada en el envío de mensajes de manera distribuida. Se focaliza en operaciones en tiempo real, pero también soporta la calendarización de tareas, es decir, puede lanzar tareas que se tengan que ejecutar en un momento determinado o de manera periódica. Las unidades de ejecución, llamadas tareas, se ejecutan de manera concurrente en uno o más nodos de trabajo. Estas tareas pueden ejecutarse de manera asíncrona bien de manera síncrona (esperando hasta que la tarea está lista). Se crean dos proyectos a partir del original, *Celery* mantiene la estructura básica de la biblioteca y se puede utilizar desde cualquier aplicación *Python* y *django-celery* que se encarga de la integración con *Django*.

1.5.8 Sentry

Sentry es un registro de eventos en tiempo real y la plataforma de agregación. Su núcleo se especializa en el control de errores y extraer toda la información necesaria para hacer una autopsia adecuada y sin ninguno de los inconvenientes del *bucle* estándar de retroalimentación de los usuarios. Es importante señalar que *Sentry* no debe ser considerado como un secuenciador de registro, sino como un adicionador de evento. Es

algo intermedio entre una solución de métricas simples (como *Graphite*) y un adicionador de secuencia de registro completo (como *Logstash*).

1.5.9 Lenguaje Unificado de Modelado (UML).

UML es un lenguaje de modelado para visualizar, construir y documentar los artefactos de un sistema, incluyendo modelado del negocio y sistemas de *software*. Proporciona un estándar para describir los modelos, incluyendo aspectos conceptuales como procesos de negocio, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. UML presenta un conjunto de diagramas y notaciones para el modelado de sistemas de software orientado a objetos que describen la semántica de los que estos diagramas y símbolos significan.

1.5.10 Herramienta *Case Visual Paradigm*.

Visual Paradigm es una herramienta *CASE* que utiliza el lenguaje modelado unificado (UML), multiplataforma y que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, implementación y pruebas. Permite construir diagramas de diversos tipos, generar código inverso y desde diagramas. Las principales características con las que cuenta son:

- Soporta un conjunto de estándares entre los que se encuentran UML, SysML, BPMN, XML y XMI.
- Ofrece herramientas para la generación de reportes en formatos *HTML*, pdf y doc.
- Generación de documentación: brinda la posibilidad de documentar todo el trabajo sin necesidad de utilizar herramientas externas.
- Interoperabilidad e integración. Permite la integración con un conjunto de herramientas (*Visio drawing, Rational Rose, ERwin Data Modeler Project, Microsoft Excel y Microsoft Word Document*) e intercambiar diagramas UML y modelos usando representaciones industriales comunes.
- Modelado de base de datos. Proporciona una mayor documentación de la base de datos y diagramas de mapeo de relación de objetos.
- Integración con herramientas para el control de versiones.
- Diseño de prototipo de Interfaz de Usuario. Permite insertar información adicional a los diagramas mediante notas y comentarios para describir sus elementos lo que facilita la revisión de los prototipos así como el trabajo en equipo.

Teniendo las características y los beneficios que presenta para el desarrollo de *software*, referentes al modelado, se decidió utilizar *Visual Paradigm* para el modelado del sistema.

Además se tuvo en cuenta que esta constituye la herramienta que utiliza la universidad para el desarrollo de *software*.

1.5.11 Bootstrap.

Bootstrap es un *framework* de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en *HTML* y *CSS*, así como, extensiones de JavaScript adicionales. Desde la versión 2.0 soporta diseños sensibles. *Bootstrap* es modular y consiste esencialmente en una serie de hojas de estilo *LESS* que implementan la variedad de componentes de la herramienta. El uso del lenguaje de hojas de estilo *LESS* permite el uso de variables, funciones y operadores, selectores anidados, así como clases *mixin*. Desde la versión 2.0, la configuración de *Bootstrap* también tiene una opción especial de "Personalizar" en la documentación. Los componentes de *JavaScript* para *Bootstrap* están basados en la librería *jQuery* de *JavaScript*. Los plugins se encuentran en la herramienta de plugin de *jQuery*. Proveen elementos adicionales de interfaz de usuario como diálogos, *tooltips* y carruseles.

1.5.12 Representational State Transfer (REST).

"REST es un estilo arquitectónico de software orientado a sistemas de hipermedia distribuido, como la *World Wide Web* (www) y a su vez es una abstracción de los elementos arquitectónicos dentro de tales sistemas. *REST* ignora los detalles de implementación de los componentes y la sintaxis del protocolo centrándose en las funciones de los mismos, las limitaciones de la interacción de estos con otros y su interpretación de elementos de datos significativos. Abarca las limitaciones sobre los componentes, los conectores y los datos que definen la base de la arquitectura web y por consiguiente la esencia de su comportamiento como una aplicación basada en la red" [13].

Los servicios web *REST* utilizando HTTPS, permite que a hacer una solicitud al servicio este no retorne la base de datos completa, sino un tipo de datos interpretable por el cliente (por ejemplo JSON, XML, HTML). Estos sistemas se conocen como *RESTful* y a los servicios que se implementan bajo *REST* como *RESTful Web Service.*

1.6 Conclusiones.

En este capítulo se concluye que la sistematización de los elementos teóricos aportó una guía para el comienzo de la investigación. Además el estudio del estado del arte permitió obtener conocimiento sobre las métricas de centralidad definiendo la de Cercanía,

Centralidad del Vector propio y Grado de Centralidad como las más importantes a la hora aportar información valiosa para la toma de decisiones. El estudio de las herramientas y bibliotecas en el estado del arte permitió seleccionar la biblioteca *Networkx* para el trabajo con grafos y *Sigmajs* para la visualización. La Fundamentación de la metodología de desarrollo permitió definir *OpenUp* como la más indicada para el desarrollo de todos los procesos y análisis del producto. El estudio de los lenguajes y las herramientas propuestas permitió seleccionar *Django* como *framework* y *Python* como lenguaje de programación para una mejor implementación.

Capítulo 2: Diseño del sistema basado en servicios web para el análisis de redes sociales.

En este capítulo se aborda la propuesta de solución a la problemática planteada. Se describen los procesos asociados al negocio a partir de un modelo de dominio y los principales conceptos que estos engloban, logrando una mejor comprensión tanto del sistema como del contexto donde se desarrolla el mismo. Se describen los requisitos funcionales y no funcionales, para dar cumplimiento a los objetivos planteados. A partir de la derivación de los requisitos funcionales se representa un diagrama de casos de uso, su descripción y la interacción con los actores. Se define el estilo y los patrones arquitectónicos a utilizar. Se precisan los patrones de diseño. Se realizan los diagramas de clases y de secuencia.

2.1 Propuesta de solución.

Se implementará un sistema capaz de gestionar la información asociada a una red social, así como de los nodos, aristas y comunidades asociadas a ella. Los usuarios tendrán la posibilidad de calcular tanto las métricas asociadas a una red, como las asociadas a una comunidad en específico. Además de detectar las comunidades de una red. También se podrán analizar estos resultados mediante la visualización de los resultados de las métricas, lo que constituirá un apoyo a la toma de decisiones en la Universidad de las Ciencias Informáticas. Además el sistema contará con una capa de servicios *web* para que otras aplicaciones puedan interactuar con el sistema y utilizar la información brindada por este. Los cuales se muestran en la siguiente tabla:

Servicios	**Descripción**
Autenticar Usuario	Permite que un usuario entre al sistema utilizando dos parámetros fundamentales el usuario y la contraseña, existen dos tipos de usuario: el usuario común y el usuario administrador.
Crear Nodo	Permite crear un nuevo nodo relacionado con usuario en la red que se está trabajando.
Modificar Nodo	Permite modificar un nodo seleccionado, cambiando su nombre o descripción.
Eliminar Nodo	Permite eliminar un nodo seleccionado.
Listar Nodo	Permite ver en una lista los nodos de la red.
Mostrar Detalles del Nodo	Permite ver los detalles de un nodo de la red. También permite añadir un nodo desde esta vista.

Crear Arista	Permite crear una arista asociada a un par de nodos la cual representa una relación entre ese par de nodos.
Modificar Arista	Permite que una arista seleccionada sea modificada cambiando las características que se desea.
Eliminar Arista	Permite que una arista seleccionada sea eliminada si se desea.
Listar Arista	Permite ver la lista de las aristas pertenecientes a una red.
Mostrar Detalles de una Arista	Permite ver todas las características que definen una arista. También permite agregar una arista desde esta vista.
Crear Red	Permite crear una nueva red, la cual va a tener una serie de usuarios asociados a ella.
Modificar Red	Permite modificar los datos de una red seleccionada.
Eliminar Red	Permite que una red seleccionada sea eliminada.
Listar Red	Permite ver la lista de redes que se encuentran registradas en el sistema.
Mostrar Detalles de Red	Permite ver los datos que identifica a la red, además permite al usuario añadir un nodo y una arista y tambien una comunidad desde esta vista en el sistema.
Visualizar Red	Permite hacer una visualización de la red que se esta analizando enseñando varias comunidades diferenciadas por varios colores.
Detectar Comunidad	Permite dentro de la red que se analiza detectar los grupos de usuarios relacionados por intereses comunes, lo cual se le llama comunidad.
Modificar Comunidad	Permite modificar los datos que identifican una comunidad.
Listar Comunidad	Permite ver todas las comunidades definidas en la red que se está analizando.
Mostrar Detalles de la Comunidad	Permite ver los datos que identifica una comunidad además permite añadir un nodo y una arista desde esta vista también.
Calcular Métricas	Permite hacer el cálculo de las métricas asociadas a una comunidad y a una red, se puede hacer el cálculo de la métrica sin necesidad de ver el resultado al momento que se manda a calcular.
Métricas	Dentro de la vista de los detalles de la red y comunidad hay una opción vinculada con el resultado de las métricas

	que previamente se habían mandado a calcular.

Tabla 1 Descripción de los Servicios

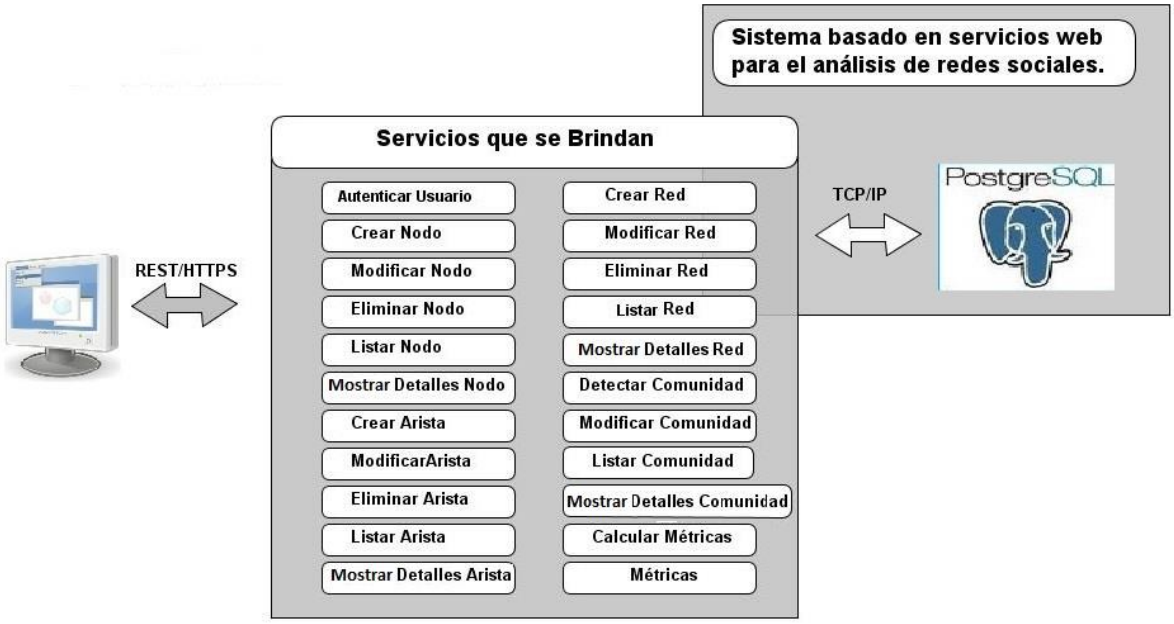


Figura 1 Servicios que se brindan.

2.2 Modelo de Dominio.

"Un Modelo de Dominio es una representación visual estática del entorno real del proyecto. Puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis, como paso previo al diseño de un sistema, ya sea de software o de otro tipo. El objetivo del modelado del dominio es comprender y describir las clases más importantes dentro del contexto del sistema y ayudar a los usuarios, clientes, desarrolladores y otros interesados a utilizar un vocabulario común" [14].

Durante el desarrollo de la herramienta no se pudo constatar procesos bien definidos en el entorno del negocio. Se hizo difícil determinar los elementos más importantes del sistema y sus interconexiones, así como el establecimiento de las reglas de funcionamiento. Sin embargo, se pueden identificar eventos, transacciones y objetos involucrados en ese entorno por lo que se hizo necesario un modelado del dominio de la solución.

Esto ayuda a los usuarios a utilizar un vocabulario común para poder entender el contexto en el cual se ubica el sistema, para capturar correctamente los requisitos y poder construir un sistema correcto se necesita tener un firme conocimiento del funcionamiento

del objeto de estudio. El objetivo de este modelo es capturar los términos necesario para comprender cómo funciona el análisis de redes sociales.

En el modelo de dominio que se representa a continuación se tiene que el usuario puede analizar una o varias redes, las cuales pueden tener uno o varios nodos que representan a los usuarios de la red, una o varias aristas que representan las relaciones que existen entre cada uno de los nodos de la red y una o varias comunidades las cuales de definen por las los intereses comunes que tengan los usuarios de la red. Una arista está compuesta por dos nodos y no necesariamente un nodo puede estar asociado a una arista. Las métricas se aplican sobre los nodos, las comunidades y la red.

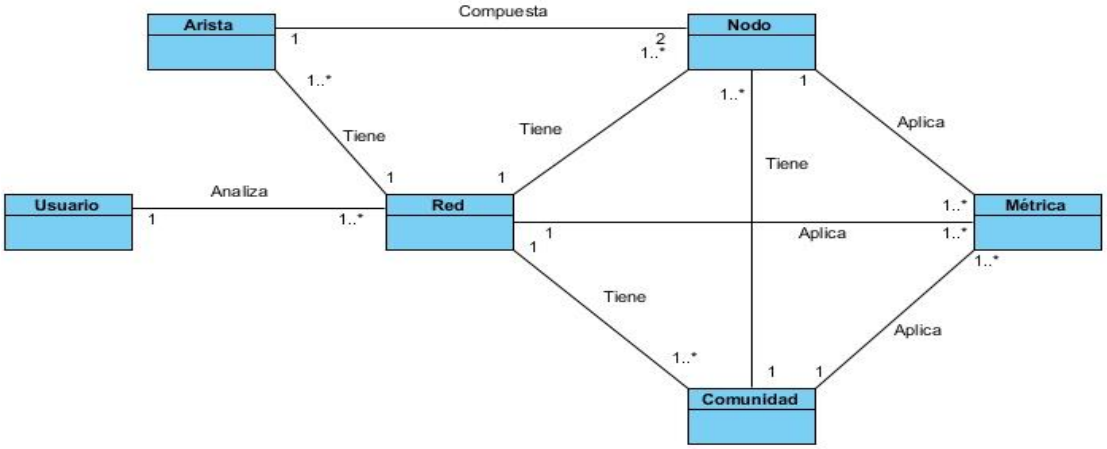


Figura 2 Modelo de dominio.

A continuación se definen los principales conceptos que se utilizan en el diagrama Modelo de Dominio:

Usuario: Persona que se encarga del análisis de las redes, así como de toda la información asociada a las mismas.

Red: Es un conjunto de actores relacionados entre sí. Donde los nodos representan a los actores y las aristas a las relaciones que se establecen entre ellos.

Comunidad: Conjunto de nodos que están relacionados entre sí por intereses comunes.

Nodo: Representa a los actores. No necesariamente tiene que estar asociado a una arista.

Arista: Está formada por dos nodos, uno donde inicia y otro que significa el fin, puede tener además un peso asociado.

Métrica: las métricas se pueden calcular de manera general a los nodos y serían entonces las métricas de la red o a una comunidad.

2.3 Especificación de requisitos.

Según el estándar 1233 de la IEEE: Guía para el desarrollo de Especificaciones de Requerimientos de Sistemas, un requisito se define como [15]:

a) "Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo".
b) "Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento".

A partir de lo planteado se puede definir que los requisitos de software son características y funcionalidades que debe cumplir un sistema. Los requisitos están enfocados hacia todo lo que debe hacer el sistema, el usuario, los miembros del equipo de proyecto y se clasifican en requisitos funcionales y requisitos no funcionales.

2.3.1 Requisitos funcionales (RF).

"Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, no alteran la funcionalidad del producto, por lo que se mantienen invariables sin importarle con que propiedades o cualidades se relacionen" [16]. A continuación se muestran los requisitos funcionales:

RF1_ Autenticar Usuario
RF2_ Adicionar Perfil
RF3_ Modificar Perfil
RF4_ Eliminar Perfil
RF5_ Listar Perfil
RF6_ Mostrar Detalles de Perfil
RF7_ Adicionar Nodo
RF8_ Modificar Nodo
RF9_ Eliminar Nodo
RF10_ Listar Nodo
FR11_ Mostrar Detalles de Nodo
RF12_ Adicionar Arista
RF13_ Modificar Arista
RF14_ Eliminar Arista
RF15_ Listar Arista
RF16_ Mostrar Detalles de Arista
RF17_ Modificar Comunidad

RF18_ Listar Comunidad

RF19_ Mostrar Detalles de Comunidad

RF20_ Adicionar Red

RF21_ Modificar Red

RF22_ Eliminar Red

RF23_ Listar Red

RF24_ Mostrar Detalles de Red

RF25_ Visualizar Red

RF26_ Calcular Métrica de centralidad

RF27_ Calcular Métrica de Cercanía

RF28_ Calcular Métrica de Centralidad de Vector Propio

RF29_ Calcular Métrica de intermediación

RF30_ Calcular Métrica de PageRank

RF31_ Calcular Métrica de Grado de Entrada

RF32_ Calcular Métrica de Grado de Salida

RF33_ Cambiar contraseña.

RF34_ Detectar comunidades

RF35_ Brindar servicios web

2.3.2 Requisitos no funcionales (RNF).

"Los requisitos no funcionales son los requerimientos que no se refieren a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste, como la fiabilidad y el tiempo de respuesta" [16].

RNF1_Usabilidad:

- El sistema deberá ser una aplicación web que pueda ser usado por cualquier persona que posea un nivel básico de conocimientos de computación.
- El sistema deberá utilizar nombres sugerentes para lograr que el usuario encuentre lo que busca en el menor tiempo posible, las acciones a realizar serán fáciles de acceder.

RNF2_Confiabilidad:

- El sistema debe validar la captación de datos para evitar entradas inadecuadas.

RNF3_Seguridad:

- El sistema debe usar roles para especificar los privilegios de cada usuario.
- El sistema debe garantizar que el acceso a la información se realice de acuerdo al rol que desempeñan los usuarios.

RNF4_Interfaz:

- El sistema debe contar con un diseño de interfaz sencillo, amigable, interactivo y poseer un uso de colores adecuados.
- El sistema debe garantizar una correcta organización de la información para permitir una adecuada interpretación.

RNF5_Idioma:

- El sistema debe utilizar el idioma español para los mensajes y textos de la interfaz.

RNF6_Hardware:

- El sistema se desplegará en un servidor que cuente como mínimo con microprocesador Core i3 a 2,4 GHz o superior, que disponga de mínimo 2 Gb de memoria RAM y que tenga al menos 5 Gb de espacio libre en el disco duro.

RNF7_Software:

- El sistema se desplegará en un servidor que tenga instalado: sistema gestor de bases de datos PostgreSql 9.1 o superior, servidor de aplicaciones web Nginx, Python 2.7.4 o superior.

2.4 Descripción de los actores

En el sistema intervienen dos usuarios, el administrador del sistema y el usuario común que son los que interactúan con el *software*.

Actor	Descripción
Usuario Administrador	Es el usuario encargado de gestionar los perfiles, así como los roles y permisos que tendrán los usuarios.
Usuario Común	Es el usuario que se encarga de la gestión de las redes, la visualización de las mismas, la gestión de los nodos y aristas.

Tabla 2 Descripción de los actores

2.5 Patrones de casos de usos.

"Los patrones de casos de uso son comportamientos que deben existir en el sistema, ayudan a describir qué es lo que el sistema debe hacer, es decir, describen el uso del sistema y cómo este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen cómo deberán ser estructurados y organizados los casos de uso. Son patrones que capturan mejores prácticas para modelar casos de uso" [17].

Estos patrones fueron utilizados durante la especificación de casos de uso, para la confección del diagrama de casos de uso del sistema, a continuación se explicarán en qué consiste cada patrón aplicado.

- **Extensión:** "Este patrón consiste en dos casos de uso y una relación de extensión entre ellos. El caso de uso extendido es concreto, es decir, este puede ser instanciado por sí solo, así como, ser una extensión del caso de uso base. El caso de uso base puede ser concreto o abstracto. Este patrón es aplicable cuando un flujo de datos puede ser extendido del flujo de datos de otro caso de uso, así como ser ejecutado por sí solo" [18].
- **CRUD (*Creating*, *Reading*, *Updating*, *Deleting*):** este patrón se basa en la fusión de Casos de Uso simples para formar una unidad conceptual. Tiene dos tipos de patrones el Parcial y el Completo, ambos son patrones de estructura.

2.6 Diagrama de caso de usos del sistema.

"Un diagrama de casos de uso del sistema representa de manera gráfica a los procesos y su interacción con los actores" [19].

Casos de uso: Es una descripción de secuencia de interacciones que se realizan entre un actor y la aplicación, cuando el primero usa el sistema para la realización de una tarea específica. Un caso de uso es una unidad coherente de funcionalidad y su nombre debe reflejar la tarea específica que el actor quiere llevar a cabo usando el sistema.

En el diagrama de casos de uso se encuentran dos actores principales que pueden interactuar con el sistema los cuales son el usuario administrador y el usuario común, que pueden hacer varias actividades como son autenticarse en el sistema, cambiar su contraseña, gestionar nodos, aristas, redes y comunidades además pueden o no hacer el cálculo de las métricas sobre las redes y las comunidades en las que se encuentran trabajando. Los usuarios comunes pueden modificar su perfil y los usuarios administradores pueden gestionar el perfil.

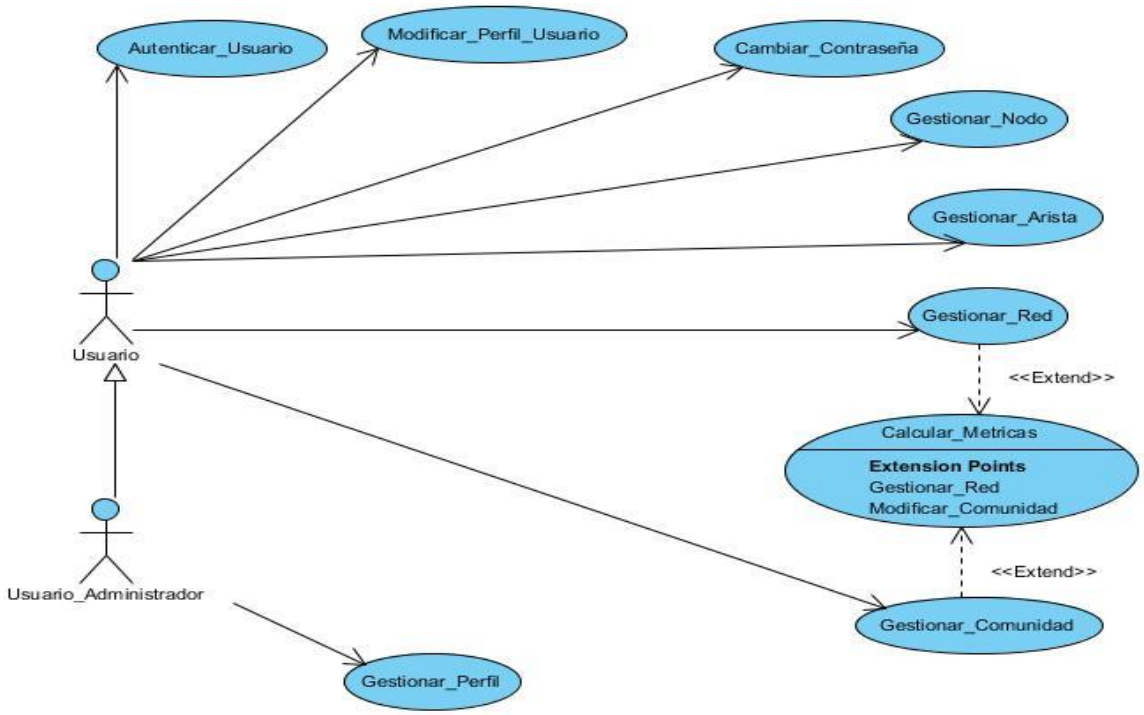


Figura 3 Diagrama de casos de uso del sistema

2.7 Definición de los casos de uso del sistema.

Después de investigar cuales son los patrones que se ajustan a los casos de uso, a continuación se refleja en una tabla como se le aplican los patrones a los requerimientos funcionales para agruparlos por cada casos de uso al cual corresponden.

Casos de Uso	**Requerimientos Funcionales**
Autenticar Usuario	Autenticar Usuario
Cambiar contraseña	Cambiar contraseña
Modificar Perfil Usuario	Modificar Perfil Usuario
Gestionar Perfil	Adicionar Perfil
	Modificar Perfil
	Listar Perfil
	Eliminar Perfil
	Mostrar Detalles de Perfil
	Adicionar Nodo
	Modificar Nodo
	Listar Nodo

Gestionar Nodo	Eliminar Nodo
	Mostrar Detalles de Nodo
Gestionar Arista	Adicionar Arista
	Modificar Arista
	Listar Arista
	Eliminar Arista
	Mostrar Detalles de Arista
Gestionar Red	Adicionar Red
	Modificar Red
	Listar Red
	Visualizar Red
	Eliminar Red
	Mostrar Detalles de Red
	Listar Métricas de Red.
Gestionar Comunidad	Modificar Comunidad
	Listar Comunidad
	Mostrar Detalles de Comunidad
	Detectar comunidades
Métrica	Calcular Métrica de centralidad
	Calcular Métrica de Cercanía
	Calcular Métrica de Centralidad de Vector
	Calcular Métrica de intermediación
	Calcular Métrica de PageRank
	Calcular Métrica de Grado de Entrada
	Calcular Métrica de Grado de Salida

Tabla 3 Definición de los Casos de Uso del sistema.

2.8 Descripción de los casos de uso del sistema.

CU # 6. Gestionar Red	
Objetivo	Gestionar las redes que se analizan por el sistema basado en servicios *web* para el análisis de redes sociales.

Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario consulta la lista de redes que se analizan en el sistema basado en servicios web para el análisis de redes sociales. A partir de aquí el actor según su nivel de acceso, puede realizar diferentes operaciones relacionadas con las redes.
Complejidad	Media.
Prioridad	Alta crítico.
Precondiciones	El usuario esté autenticado en el sistema.
Pues condiciones	La Red es gestionada por el usuario con éxito
Flujo de eventos	
Flujo básico Gestionar Red	

	Actor	**Sistema**
1	Selecciona el ítem listar red en el menú red para mostrar las redes que se encuentran en el sistema.	
2		Muestra la lista de las redes y permite realizar varias acciones con una red: ✓ Detalle de Red. Ver Sección 1:Detalle de la red (Prototipo) ✓ Adicionar red. Ver Sección 2: Adicionar red (Prototipo 2). ✓ Modificar red. Ver Sección 3: Modificar red (Prototipo 3). ✓ Eliminar red. Ver Sección 4: Eliminar red (Prototipo 4). ✓ Visualizar red. Ver Sección 5: Visualizar red (Prototipo 5). ✓ Calcula Métricas: Ver Caso uso: Calcular Métricas
3	Pulsa el enlace o el botón asociado a alguna acción: ✓ Mostrar detalles de una red. ✓ Adicionar red. ✓ Modificar red. ✓ Eliminar red. ✓ Visualizar red.	Verifica si el usuario tiene acceso a la acción seleccionada.
4		Muestra la vista asociada a la acción (Ver las secciones asociadas a las acciones).

5		Terminar el caso de uso.

Prototipo

Sección 1: "Mostrar detalles de red "

Flujo básico Mostrar detalles de red

	Actor	Sistema
1		Se muestran los detalles de la red seleccionada. Además muestra las opciones de: ✓ Modificar red. ✓ Eliminar red.
2		Termina Sección 1.

Sección 2: "Adicionar red"

Flujo básico Adicionar red

	Actor	Sistema
1	Introduce los campos correspondientes a la red: ✓ Nombre de la red. ✓ Descripción de la red ✓ Red. ✓ Ruta del archivo. Pulsa el botón Crear.	
2		Valida los datos entrados por el usuario.
3		Registra los datos de la red. Muestra el mensaje "La red se ha registrado correctamente".
4		Redirecciona a la vista "Listar redes".
5		Termina Sección 2.

Flujos alternos

Nº 1 ***El usuario pulsa el botón cancelar.***

	Actor	Sistema
1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar redes".
2		Vuelve al paso 2 del flujo básico del caso de uso.

Flujos alternos

Nº 1 ***Datos incorrectos***

	Actor	Sistema
1		Muestra un mensaje "Error: Algunos datos son incorrectos."
2	Corrige los datos incorrectos y pulsa el botón "Adicionar red".	Vuelve al paso 2 del flujo básico de la sección 2.
3		Termina Sección 2.

Prototipo

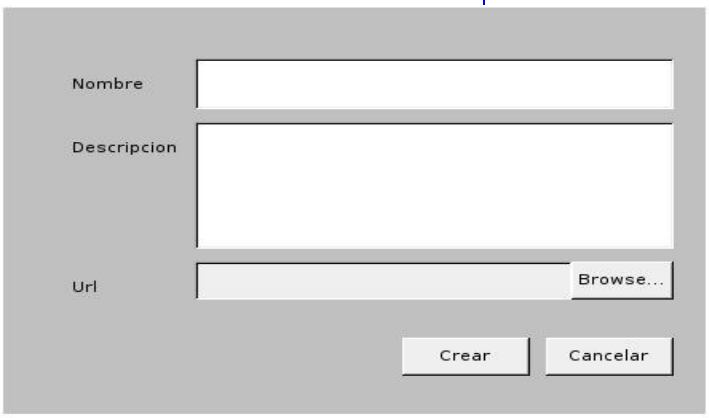


Tabla 4 Descripción del caso de uso Gestionar Red.

Para consultar las demás descripciones de casos de usos consultar Anexo #1.

2.9 Descripción de la arquitectura.

"La arquitectura de software es una vista del sistema que incluye los componentes principales del mismo, consiste en la vista conceptual de toda su estructura. Un estilo arquitectónico es un conjunto coordinado de restricciones arquitectónicas que restringe los roles o rangos de los elementos arquitectónicos y las relaciones permitidas entre esos elementos dentro de la arquitectura que se conforma a ese estilo" [13].

2.9.1 Arquitectura Modelo-Vista-Controlador (MVC).

"La arquitectura Modelo Vista Controlador surgió como patrón arquitectónico para el desarrollo de interfaces gráficas de usuario en entornos *Smalltalk*. Su concepto se basa en la necesidad de reducir el esfuerzo de programación necesario en la implementación

de sistemas múltiples y sincronizados de los mismos datos, mediante la división de la aplicación en tres partes fundamentales" (20):

- El modelo, que contiene la lógica de negocio de la aplicación.
- La vista, que muestra al usuario la información que este necesita.
- El controlador, que recibe e interpreta la interacción del usuario, actuando sobre modelo y vista de manera adecuada para provocar cambios de estado en la representación interna de los datos, así como en su visualización.

"Estos tres elementos juntos, la lógica de acceso a la base de datos, la lógica de negocios, y la lógica de presentación comprenden un concepto, que a veces es llamado, el patrón de arquitectura de software Modelo-Vista-Controlador (MVC). En este patrón, el modelo hace referencia al acceso a la capa de datos, la vista se refiere a la parte del sistema que selecciona qué mostrar y cómo mostrarlo, y el controlador implica la parte del sistema que decide qué vista usar, dependiendo de la entrada del usuario, accediendo al modelo si es necesario" [20]. El Modelo, la Vista y el Controlador se separan en Django de la siguiente manera:

- Modelo, la porción de acceso a la base de datos, es manejada por la capa de la base de datos de Django.
- Vista, la porción que selecciona qué datos mostrar y cómo mostrarlos, es manejada por la vista y las plantillas.
- Controlador, la porción que delega a la vista dependiendo de la entrada del usuario, es manejada por el framework.

Debido a que el Controlador es manejado por el mismo framework Django es conocido como un Framework MTV. En el patrón de diseño MTV:

- M significa (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.
- T significa (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.
- V significa (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: puedes pensar en esto como un puente entre los modelos y las plantillas.

2.10 Patrones de diseño.

"Un patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos; en teoría, indican la manera de utilizarlo en circunstancias diversas" [17].

Los patrones de diseño consisten en la descripción de un problema y la solución del mismo, de forma tal que se puedan utilizar en diferentes contextos para resolver interrogantes comunes. Constituyen una solución efectiva que se le dio a un problema en determinado momento y que puede reutilizarse aplicándose en diferentes problemas de diseño en distintas circunstancias. Para el desarrollo de la aplicación se utilizan los patrones GRASP (*General Responsibility Assignment Software Patterns*) y GOF (Gang of Four) su utilización corresponde a las buenas prácticas para el desarrollo del software.

Creador: Ayuda a identificar quien debe ser el responsable de la creación o instanciación de nuevos objetos o clases. Este patrón se evidencia en la función *createGraph*.

Controlador: Sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, por lo que recibe las peticiones de los usuarios y se encarga de enviarlos a las distintas clases según el método llamado. Este patrón se puede evidenciar en las clases controladoras como *NetworkDeleteView*, *NetworkNodesView*.

Alta cohesión: La información que sea almacenada en la clase debe ser coherente y debe poseer alta relación con la misma. En *Django* este patrón se evidencia manteniendo cada capa en archivos diferentes logrando una alta cohesión y flexibilidad en el *software*.

Bajo acoplamiento: La idea fundamental es tener las clases lo menos relacionadas posibles, lo que permite que la modificación de una de las clases repercuta lo menos posible en cada una de las restantes. En una aplicación *web* hecha en *Django* las definiciones *URL* y la implementación de las funciones de vistas a las que estas URL llaman están poco acopladas ya que están en lugares separados. Esto nos permite modificar las *URL* o la función de vista asociada sin tener que hacer grandes cambios en el resto de la aplicación.

Decorador: este patrón se pone en evidencia cuando se separan las vistas y se forma una plantilla base y varias plantillas que heredan de ella. Esta plantilla bases es la que tiene el código global y común de todas las páginas del sistema, para que no se tenga que repetir una y otra vez en cada página. Esto sería una implementación del patrón Decorador.

2.11 Modelo de diseño.

"El modelo de diseño es utilizado para modelar los aspectos dinámicos del sistema. Consta de un conjunto de objetos que describen las realizaciones de los casos de uso y sus relaciones. Se centra en los impactos que producen en el sistema los requisitos funcionales y no funcionales. De manera general el modelo de diseño constituye una abstracción al modelo de implementación y del código fuente, o sea, es la entrada o el punto de partida para las posteriores actividades de implementación del sistema que se desea desarrollar" [14].

2.11.1 Diagrama de clases del diseño.

"Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases en sí, muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los mismos se utilizan para reflejar la vista de diseño estática de un sistema. Son la base para los posteriores diagramas de componentes y los de despliegue. Su importancia radica en que permitirá construir sistemas ejecutables aplicando ingeniería directa e inversa" [14].

En el diseño se modela el sistema y se encuentra su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales y otras restricciones. Una entrada esencial en el diseño es el resultado del análisis. El propósito principal es adquirir una comprensión con profundidad de los aspectos relacionados con los requisitos funcionales, no funcionales y restricciones asociadas a la programación, sistemas operativos y tecnologías de interfaz de usuario. UML posee una extensión para el modelado de aplicaciones *web*, esa extensión es usada para el diseño de las clases. Los estereotipos que usa esta extensión son:

<<*Client Page*>> Una instancia de Página Cliente es una página *web*, con formato HTML. Cada página cliente solo puede ser construida por una página servidor.

<<*Form*>> Grupo de elementos de entrada que son parte de una página cliente. Sus atributos son los elementos de entrada del formulario.

<<*Server page*>> Representa la página web que contiene código que se ejecuta en el servidor.

Link: Representa un apuntador desde una "client page" hacia una "*client page*" o "*server page*". Corresponde directamente con una etiqueta <a> (ancla) de HTML.

Submit: Esta relación siempre se da entre una "form" y una *"server page",* por supuesto, la "*server page*" procesa los datos que la "*form*" le envía (*submits*).

Build: Sirve para identificar cuales "*server page*" son responsables de la creación de una "*client page*".

Redirect: Esta es también una relación unidireccional que indica que una página *Web* redirige hacia otra.

Las *client page* las cuales son las páginas que se muestran en la interfaz del sistema y son las que interactúan con el usuario además construyen un formulario en el cual el usuario llena los datos necesarios para adicionar una nueva red, modificar una red seleccionada o eliminar la red que se seleccione. Todas las *client page* que se encuentran representadas en el diagrama son construidas por una *server page* la cual se encarga de manejar la comunicación con la clase controladora, la clase controladora se encarga de pedir la información a la clase entidad que es la que contiene toda la información relacionada con la red.

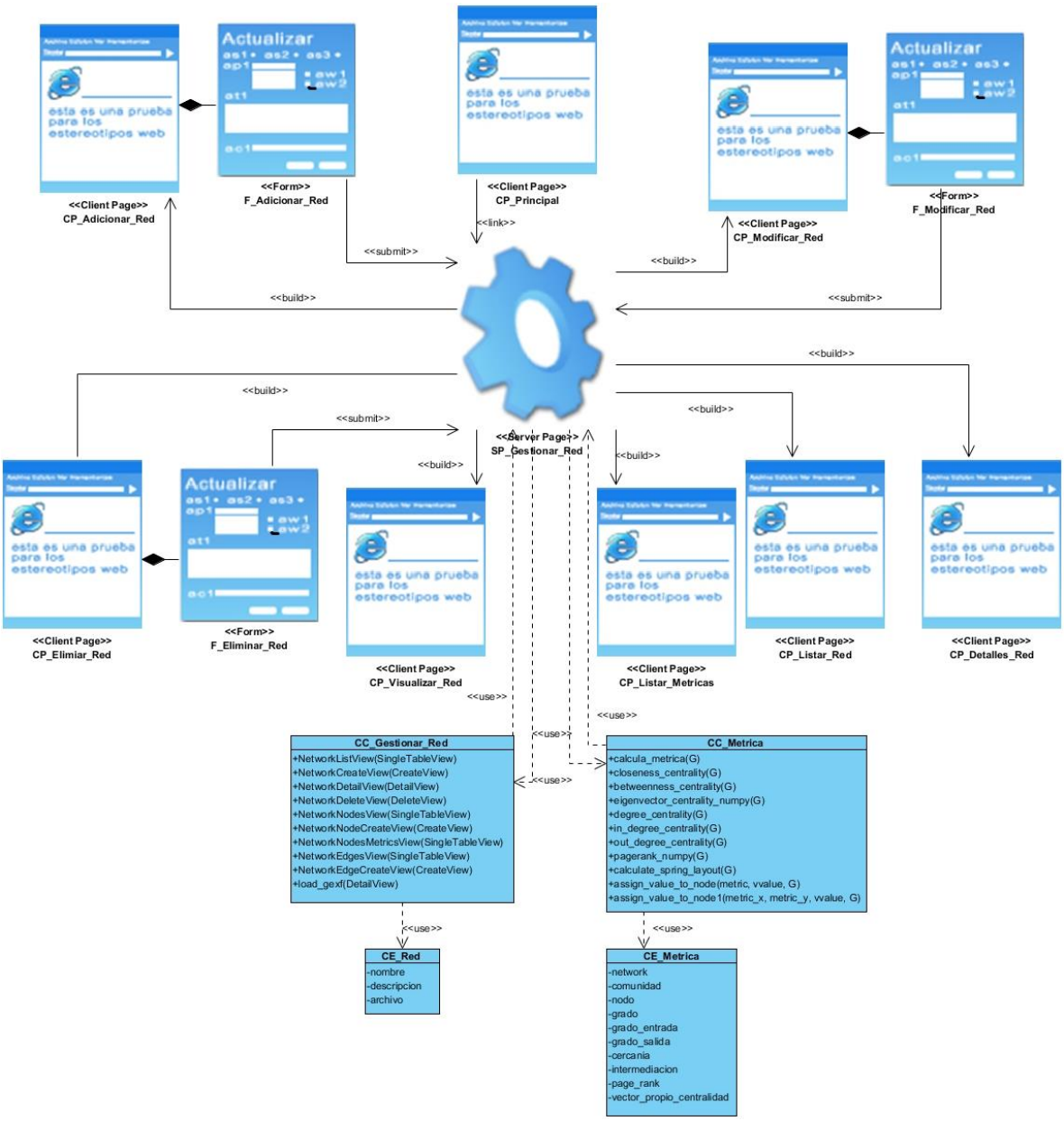


Figura 4 Diagrama de clases del diseño Gestionar Red.

Para consultar los demás diagramas de clases consultar Anexo #3.

2.11.2 Diagramas de secuencia.

"Es el diagrama que muestra las interacciones entre los objetos organizados en una secuencia temporal. En particular muestra los objetos participantes en la interacción y la secuencia de mensajes intercambiados. Dentro del conjunto de mensajes representados dispuestos en una secuencia temporal, cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical que representa el rol durante cierto plazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre líneas

de vida. Un diagrama de secuencia puede mostrar un escenario, es decir, una historia individual de transacción. Un uso de un diagrama de secuencia es mostrar la secuencia del comportamiento de un caso de uso" [21].

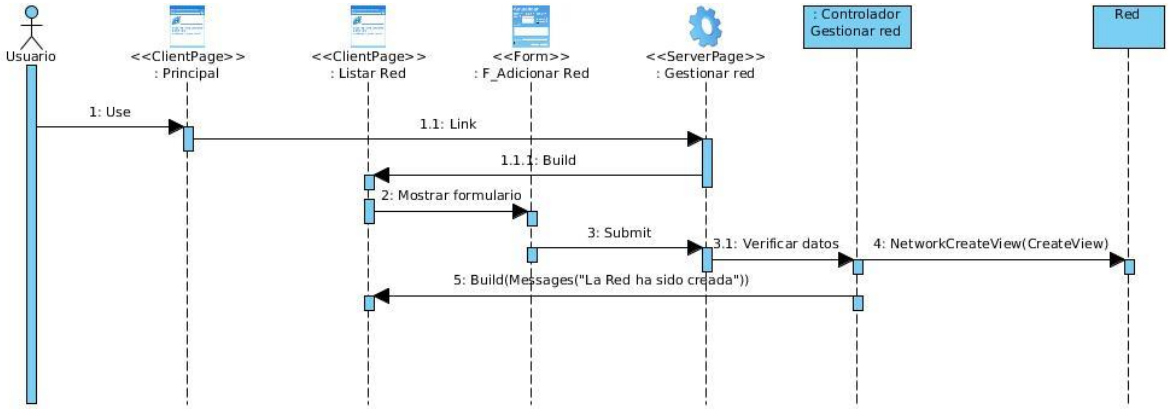


Figura 5 Diagrama de secuencia Adicionar Red.

En diagrama de secuencia adicionar red se tiene el actor (usuario en lo adelante) el cual se encuentra utilizando la interfaz principal y desea adicionar una red mediante un vínculo accede a la página servidora Gestionar Red que a su vez esta construye interfaz de Adicionar Red que muestra el formulario para llenar los datos relacionados con la nueva red. El usuario mediante un botón en el formulario ordena a la página servidora Gestionar Red que comience el proceso de adicionar los nuevos datos de la red en la base de datos mediante el controlador Gestionar Red, el sistema construye los datos en la base datos y le envía al usuario la notificación de que se adicionó la nueva red.

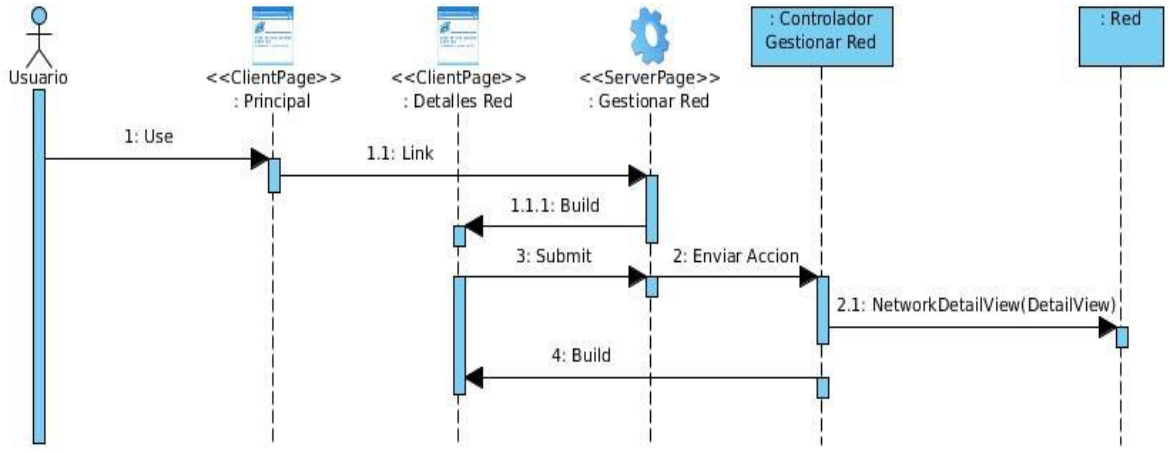


Figura 6 Diagrama de secuencia Detalles de Red.

En el diagrama de secuencia Mostrar Detalles de Red el usuario se dispone a ver los detalles de una red y accede a la página servidora mediante un vínculo, la cual construye la interfaz Mostrar Detalles de Red y mediante un botón envía la acción a la página

servidora y esta a su vez verifica la acción con la controladora y el usuario recibe como respuesta del sistema los detalles de la red seleccionada (**Para consultar los demás diagramas de secuencia consultar Anexo #3**).

2.12 Modelo de datos.

Un modelo de datos: "Es un conjunto de conceptos que permite describir los datos, las relaciones que existe entre ellos, la semántica y las restricciones de consistencia" [22].

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información. El modelo de datos físico plantea cómo se maneja la información en la base de datos en el cual existen varias tablas relacionadas con los nodos, las redes, las comunidades, las aristas, las métricas y los usuarios que guardan toda la información necesaria para una posterior gestión de ella.

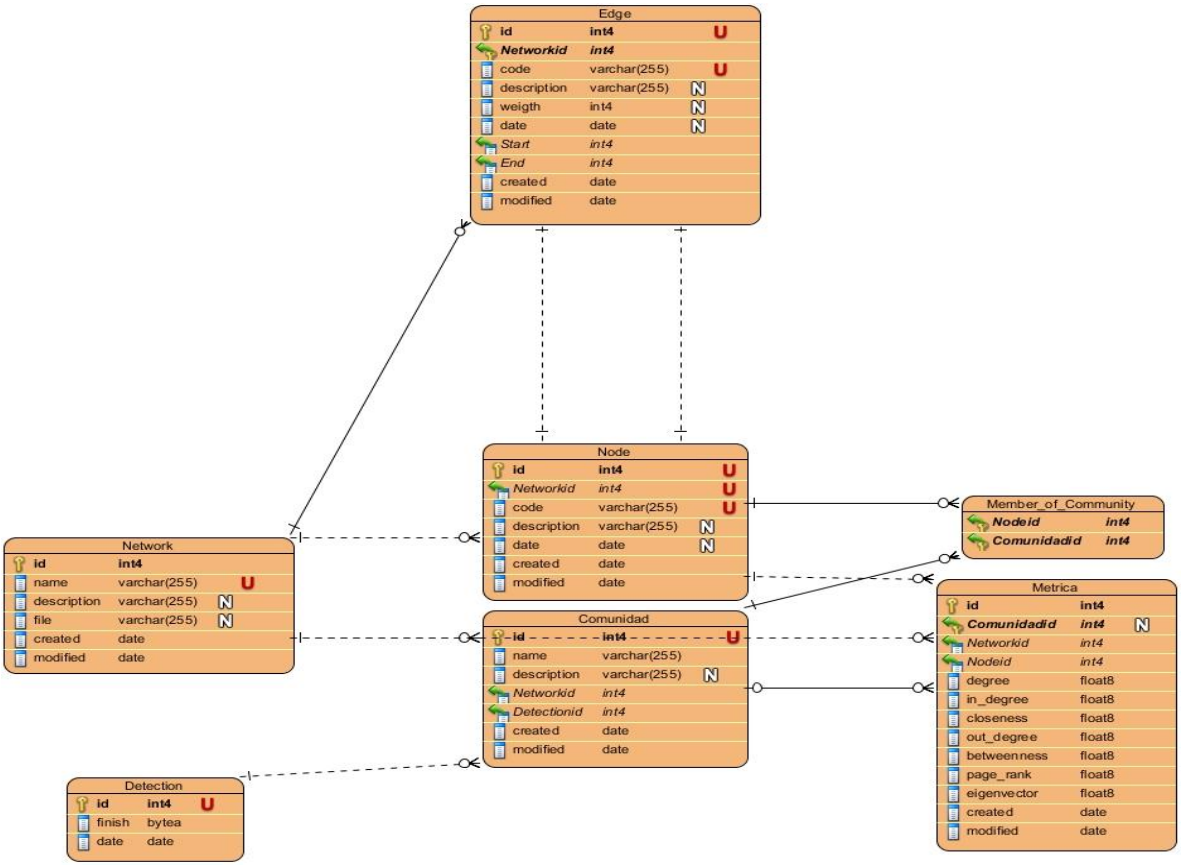


Figura 7 Modelo de Datos físico del sistema.

2.13 Conclusiones.

La realización de la propuesta de solución permitió definir los servicios web a brindar por la aplicación. La identificación de los requerimientos funcionales, así como la realización del diagrama de casos de uso proporcionó un mejor entendimiento de las principales funcionalidades del sistema a la hora de su implementación. Además la modelación de los diagramas de clases del diseño y diagramas de secuencia permitió tener una mejor visión de la interacción entre el usuario y el sistema. La realización del modelo físico de datos proporcionó un buen entendimiento de como almacenar y gestionar la información proporcionada para el análisis de las redes sociales. La definición de la arquitectura Modelo-Vista-Controlador y dentro de esta la definición Modelo-Vista-Plantilla utilizada por *Django* dejó sentadas las bases para la confección e implementación del sistema.

Capítulo 3: Implementación y prueba del sistema basado en servicios web para el análisis de redes sociales.

Se realiza el modelo de componentes. Se describe el estilo de código utilizado. Además se realiza el modelo de despliegue. Se definen los niveles de pruebas. Se diseñan y aplican las pruebas inherentes al sistema para comprobar el correcto funcionamiento del sistema basado en servicios web para el análisis de redes sociales y se ofrecen los resultados de las mismas.

3.1 Diagrama de componentes.

"Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño. El diagrama de componentes muestra la vista física del software en términos de componentes ejecutables y librerías de clases con sus relaciones y sus dependencias" [14]. Estos representan los elementos físicos del sistema y las relaciones entre ellos. Los componentes representan todos los elementos de *software* que intervienen en la fabricación de aplicaciones informáticas. Un diagrama de componentes representa las dependencias entre componentes de *software*, incluyendo componentes de código fuente, códigos binarios y ejecutables, hace parte de la vista física de un sistema, la cual modela la estructura de implementación de la aplicación por sí misma. Esta vista proporciona la oportunidad de establecer correspondencias entre las clases y los componentes de implementación. **Para consultar los demás diagramas de componentes consultar Anexo #6.**

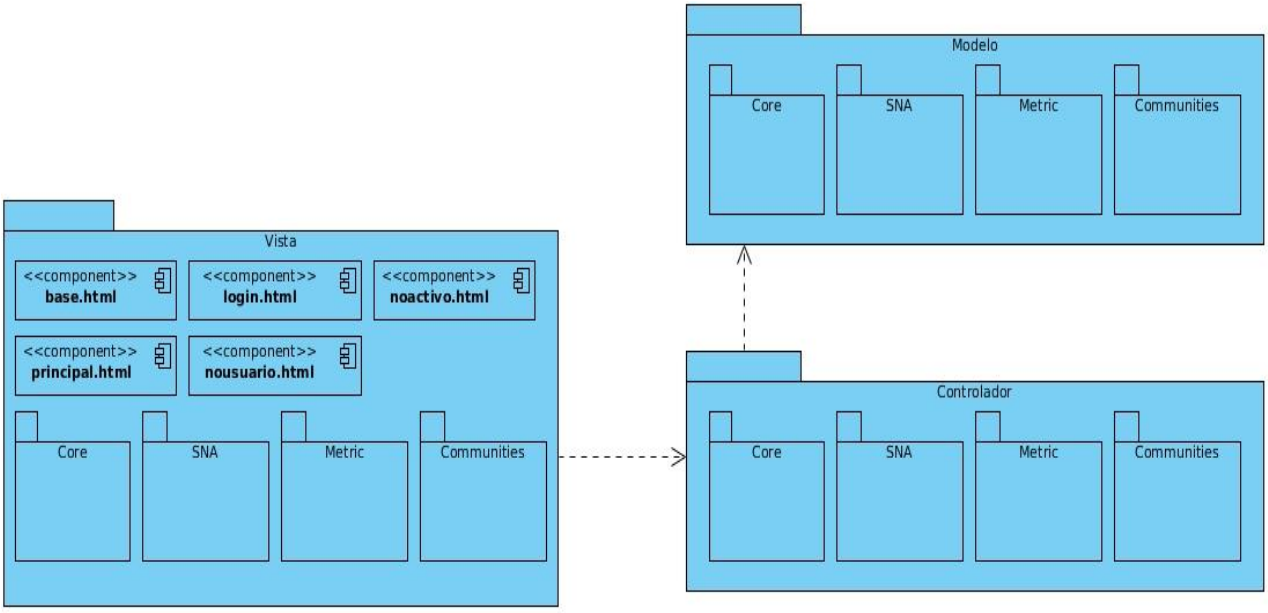


Figura 8 Diagrama de Componentes.

3.2 Estilo de código.

El código es importante que sea legible, ya que escribir un segmento de código individual toma momentos hacerlo, pero al depurarlo puede llevar desde minutos hasta horas. Además el desarrollador u otras personas revisan el código tiempo después, es de ayuda que este escrito con un estilo de código limpio y consistente, lo que facilita darle mantenimiento y enriquecer sistemas de cualquier tamaño. Para hacer el código lo más legible posible:

- Evitar usar abreviaturas en los nombres de las variables.
- Escribir fuera de las funciones los nombre de los argumentos.
- Documentar las clases y métodos.
- Utilizar las líneas repetidas de código en funciones reusables o métodos.

PEP8 es el estilo de código definido en *Python*. El mismo describe convenciones de código tales como:

- Usar 4 espacios por nivel de indentación.

```
class NetworkListView(SingleTableView):
    model = Network
    table_class = NetworkTable
    table_pagination = {'per_page':5}
    template_name = 'networks/network_list.html'
```

Figura 9 Estilo de código: Niveles de indentación.

- Separar funciones de alto nivel y definiciones de clase con dos líneas en blanco.

```
class NetworkListView(SingleTableView):
    model = Network
    table_class = NetworkTable
    table_pagination = {'per_page':5}
    template_name = 'networks/network_list.html'

    def get_table_pagination(self):
        return self.table_pagination


class NetworkCreateView(CreateView):
    model = Network
    template_name = 'networks/network_create.html'
```

Figura 10 Estilo de código: Separación de funciones de alto nivel y clases.

- Las definiciones de métodos dentro de una clase son separadas por una línea en blanco.

```
class NetworkListView(SingleTableView):
    model = Network
    table_class = NetworkTable
    table_pagination = {'per_page':5}
    template_name = 'networks/network_list.html'

    def get_table_pagination(self):
        return self.table_pagination
```

Figura 11 Estilo de código: Separación de las definiciones de métodos.

- Las importaciones deben estar en líneas separadas.

```
import networkx as nx

import uuid
```

Figura 12 Estilo de código: Importaciones

- Las importaciones deben estar agrupadas en el siguiente orden:
  1. importaciones de la librería estándar
  2. importaciones del núcleo de *Django*.
  3. importaciones terceras relacionadas
  4. importaciones locales de la aplicación / librería.

  (Debería haber una línea en blanco entre cada grupo).

```
from django_tables2 import SingleTableView
from django.contrib.auth.models import User
from django.core.urlresolvers import reverse
from django.views.generic import (CreateView, UpdateView, DeleteView, DetailView,
    ListView)
from django.views.generic.detail import BaseDetailView
from django.shortcuts import get_object_or_404
from django.contrib import messages
from django.http import request

from rest_framework import generics
from rest_framework.views import APIView

from .forms import EdgeForm, NodeForm
from .models import Network, Node, Edge
from .tables import NetworkTable, NodeTable, EdgeTable

from metrics.models import Metric
from metrics.tables import MetricsTable
from metrics.tasks import metrics_db, metrics_communities
from communities.tasks import detect_communities1
```

Figura 13 Estilo de código: Orden de las importaciones

3.3 Diagrama de despliegue.

"El diagrama de despliegue se utiliza para modelar los aspectos físicos sobre los que se ejecutarán los componentes del sistema de *software* en términos de nodos, donde los nodos sirven para modelar la topología del *hardware* sobre el que se ejecuta un sistema y representan dispositivos sobre los cuales se pueden desplegar los componentes" [14].

En el diagrama de despliegue que se muestra a continuación N pc clientes se conectarán al servidor web mediante una conexión HTTPS. El servidor web se conectará al servidor de base de datos mediante una conexión TCP/IP.

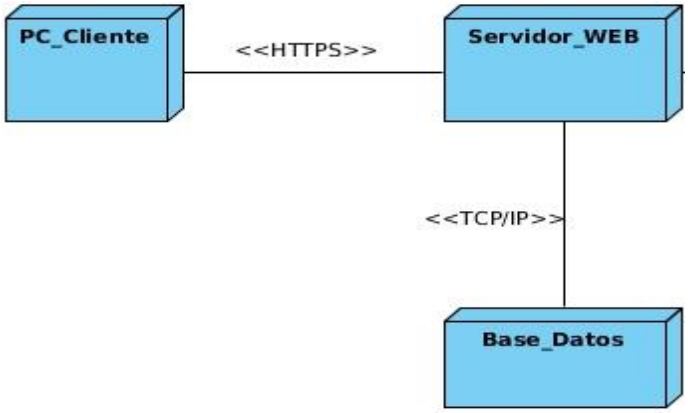


Figura 14 Diagrama de Despliegue.

3.4 Pruebas de *software*.

"Las pruebas del *software* son un elemento fundamental para la garantía de calidad del sistema. Estas pruebas representan una revisión final de las especificaciones del diseño y de la codificación, es decir, las pruebas verifican que el *software* funcione como se diseñó y que los requerimientos son satisfechos, además de brindar soporte para encontrar y documentar defectos del sistema" [23]. El objetivo de la prueba de *software* es descubrir errores.

Para la realización de las pruebas al sistema los autores decidieron utilizar las Pruebas de Rendimiento por la necesidad de probar el rendimiento que tendrá el sistema ante un número de conexiones concurrentes de usuarios o peticiones de los mismos. Además se utilizarán las pruebas de seguridad para comprobar que se cumplan los niveles de acceso según los roles establecidos y restringir el acceso a la información. También se utilizarán las Pruebas de Caja Negra o Funcional para verificar que se cumplan los requisitos del cliente.

3.4.1 Pruebas de Seguridad

"Permite asegurar que el sistema modificado o nuevo, incluya controles de acceso apropiados y no contenga ningún agujero de seguridad que pudiera comprometer otros sistemas" [26].

Las pruebas de seguridad están enfocadas a garantizar que el acceso a la información por parte de los usuarios sea según el rol que desempeñan y a las funcionalidades específicas para ese rol. Dependiendo del rol que esté asignado al usuario que se autentique será capaz de ejecutar determinadas funciones según su nivel de acceso. Este tipo de pruebas permite comprobar los niveles de seguridad lógica del sistema.

Las pruebas de seguridad aseguran que los usuarios tengan acceso solo a la información que están autorizados a acceder, garantizan que no existan brechas de seguridad en el sistema pues solo el personal autorizado puede acceder a la aplicación.

En el grupo de Seguridad del Departamento de Software (DEPSW) de la UCI se han establecido niveles para este tipo de pruebas. La evaluación de la seguridad de las aplicaciones en un primer nivel (nivel 1) se definió 15 indicadores agrupados en 4 tipos de pruebas.

Resultados Pruebas de Seguridad nivel 1.

1. Pruebas de Autorización: Ningún usuario estándar pudo modificar sus privilegios ni los de otro usuario en la aplicación.

2. Pruebas de Gestión de Sesiones: No se pudo acceder a la aplicación copiando la URL después de estar autenticado, cerrar el navegador y volver a abrirlo. Al cerrar la sesión de un usuario y dar clic en el botón del navegador "Atrás" la aplicación no entró a la sesión autenticada.

3. Validación de Datos: Se enmascararon los datos confiables cuando se visualizan en la aplicación. Solamente se permiten contraseñas alfanuméricas, que incluyan caracteres especiales y que tengan seis caracteres, como mínimo, de longitud. El sistema no mostró mensajes indebidos al colocar en la barra de dirección o en campos de entrada los caracteres: comillas simples (´´), signos de *ampersand* (&), o signos*: + - /.*

4. Comprobación del Sistema de Autenticación: Los mensajes de error para distintas combinaciones de autenticación muestraron la misma información. Los tiempos de respuestas usuario correcto - contraseña incorrecta y usuario incorrecto - contraseña incorrecta fueron los mismos. El sistema protegió el envío de los datos mediante protocolo

seguro (*HTTPS*). Se bloqueó la sesión del usuario después de 1 hora de inactividad, pero no se bloqueó la cuenta del usuario después de un número de intentos de autenticación fallidos. El campo usuario de la autenticación al sistema tiene el autocompletamiento desactivado (no guarda los usuarios que se autentican). El sistema usa un certificado.

Resultados de las pruebas de seguridad nivel 2.

Para valorar la seguridad en el nivel número dos se utilizó la herramienta *Acunetix Web Vulnerability Scanner 8*, que se emplea para detectar vulnerabilidades de seguridad en el sistema. *Acunetix Web Vulnerability Scanner* es una herramienta capaz de escanear sitios *web* en busca de posibles fallos de seguridad que puedan poner en peligro la integridad de la página una vez publicada en Internet. Esta aplicación ejecuta una serie de pruebas, configurables por el usuario, para identificar las vulnerabilidades tanto en la programación de la página como en la configuración del servidor, detecta técnicas de *hacking* como pueden ser *SQL injection*, *Cross Site Scripting*, *Passwords* débiles, etc. Una vez concluido este proceso, la versión genera una serie de informes, especificando los fallos detectados en el análisis de la página *web*. Estos informes se mostrarán en la pantalla en forma de gráfica para una mejor perfección, o bien podrán visualizarse los resultados más exhaustivamente con la descripción de la vulnerabilidad.

Durante la ejecución del programa se detectaron cuatro no conformidades de las cuales dos son de prioridad media y dos de prioridad informativa. Las alertas de prioridad media señalan el certificado ssl, que el sistema trae un certificado de nivel bajo por defecto y cuando se haga el despliegue del sistema es el cliente el que aplica su certificado. Las alertas de prioridad informativa destacan que se muestra la dirección IP en la URL del sistema y se rectificará cuando el sistema se monte en un servidor DNS que es el que hace la traducción de la dirección física por un nombre.

3.4.2 Pruebas funcionales.

Este tipo de pruebas tienen como objetivo principal verificar que se cumplan los requisitos funcionales del *software*. Consiste en estudiar las entradas que recibe y las respuestas que produce determinado sistema, sin tener en cuenta la lógica del programa, solo la funcionalidad que debe realizar. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores en estructuras de datos, de rendimiento y de inicialización y terminación.

"Para definir bien las entradas y salidas del *software* hay que encontrar una serie de datos de entrada que causen un comportamiento erróneo en el sistema y como

consecuencia producen una serie de salidas que revelan la presencia de defectos" [14]. Esto se denomina como Casos de prueba.

Casos de prueba de caja negra.

"Para comprobar la calidad del producto desarrollado, se realizarán los casos de prueba de caja negra, con el objetivo de demostrar que el mismo cumple con los requisitos funcionales definidos. En la realización de estos casos, la técnica a emplear será de partición de equivalencia, pues la misma permite examinar los valores válidos y no válidos para las condiciones de entrada existentes en el *software*" [23].

SC Adicionar Red

Escenario	Descripción	Nombre	Descripción	Red	Respuesta del sistema	Flujo central
EC 1.1 El usuario solicita crear una red	*El usuario solicita crear una red con un nombre válido*	*V*	*NA*	*NA*	*El sistema crea la red y muestra un mensaje "La red se ha creado satisfactoriamente"*	*1 El usuario autenticado solicita gestionar red.* *2 Solicita crear una red haciendo clic en el botón Adicionar red.* *3 Introduce los datos pertenecientes a una red y da clic en el botón Crear.*
		Nombre de red válido	*NA*	*NA*		
EC 1.2 El usuario solicita crear una red	*El usuario solicita crear una red con un nombre existente*	*I*	*NA*	*NA*	*Muestra un mensaje de error "La red no ha sido creado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor ya se ha utilizado."*	
		Nombre de la red existente	*NA*	*NA*		
EC 1.3 El usuario solicita crear una red	*El usuario solicita crear una red con un nombre no válido*	*I*	*NA*	*NA*	*Muestra un mensaje de error "La red no ha sido creada" e informa del campo*	

					incorrecto: en caso de introducir una cadena no válida; "Solo letras, números y espacios" en caso de ser la cadena menor de 2 caracteres; "Este valor es demasiado corto. Debería tener 2 caracteres o más." y en caso de tener más de 30 caracteres; "Este valor es demasiado largo. Debería tener 30 caracteres o menos."	
		Nombre de la red no válido	*NA*	*NA*		
EC 1.4 El usuario solicita crear una red	*El usuario solicita crear una red con el campo nombre vacío*	*I*	*NA*	*NA*	*Muestra un mensaje de error "Campo no válido" o "Invalid field"*	
		Campo vacío	*NA*	*NA*		
EC 1.5 El usuario solicita crear una red	*El usuario solicita crear una red con el campo red incorrecto*	*NA*	*NA*	*I*	*Muestra un mensaje de error "La red no ha sido creada" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*Campo Red no válido*		

SC Modificar Red

Escenario	Descripción	Nombre	Descri	Red	Respuesta	Flujo

			pción		del sistema	central
EC 1.7 El usuario solicita modificar una red	*El usuario solicita modificar la red seleccionado con nombre no válido*	*I*	*NA*	*NA*	*Muestra un mensaje de error "La red no ha sido modificada" e informa del campo incorrecto: en caso de introducir una cadena no válida; "Solo letras, números y espacios" en caso de ser la cadena menor de 2 caracteres; "Este valor es demasiado corto. Debería tener 2 caracteres o más." y en caso de tener más de 30 caracteres; "Este valor es demasiado largo. Debería tener 30 caracteres o menos."*	*1 El usuario autenticado solicita gestionar red. 2 Solicita modificar una red haciendo clic en la red que desee modificar. 3 Modifica los datos y da clic en el botón Guardar.*
		Nombre de red no válido	*NA*	*NA*		
EC 1.8 El usuario solicita modificar una red	*El usuario solicita guardar la red seleccionada con un nombre válido*	*V*	*NA*	*NA*	*Muestra un mensaje "La red seleccionado ha sido modificado satisfactoriamente"*	
		Nombre de la red válido	*NA*	*NA*		
EC 1.9 El usuario solicita modificar una red	*El usuario solicita modificar la red seleccionada con un nombre existente diferente del propio*	*I*	*NA*	*NA*	*Muestra un mensaje de error "La red seleccionada no ha sido modificada" e informa del campo incorrecto con un mensaje sobre el mismo "Este*	
		Nombre de la red existente	*NA*	*NA*		

					valor ya se ha utilizado."	
EC 1.10 El usuario solicita modificar una red	*El usuario solicita modificar la red seleccionada con el campo nombre vacío*	*I*	*NA*	*NA*	*Muestra un mensaje de error "Campo no válido" o "Invalid field"*	
		Campo vacío	*NA*	*NA*		
EC 1.11 El usuario solicita modificar una red	*El usuario solicita modificar una red con el campo red incorrecto*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "La red no ha sido modificada" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*Campo Red no válido*		

(Para consultar los demás casos de pruebas consultar Anexo #5).

Resultados de las pruebas de caja negra.

Se realizaron 3 iteraciones de estas pruebas para la validación de los requisitos funcionales donde se ejecutaron 35 diseños de casos de pruebas. A continuación se muestran los resultados obtenidos por cada una de las iteraciones:

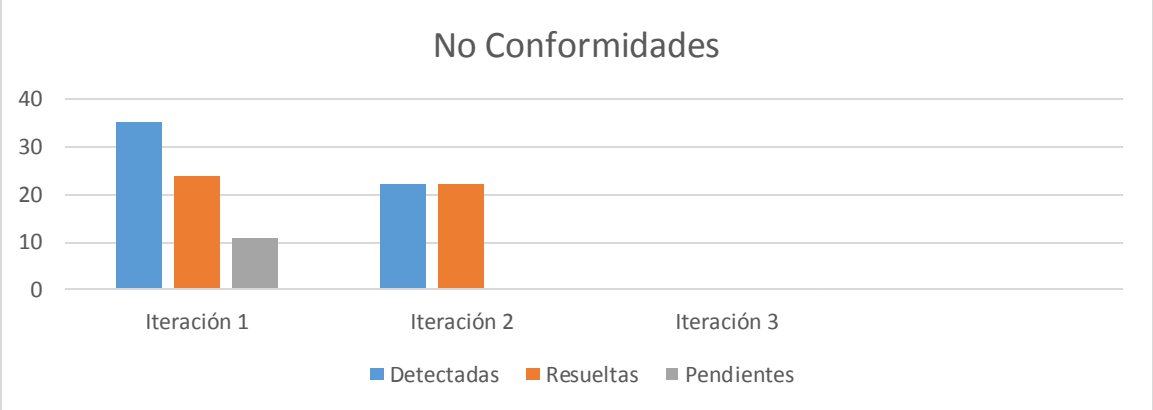


Figura 15 Comportamiento de las no Conformidades

Algunas de las no conformidades encontradas durante el proceso de pruebas fueron:

- No se muestran correctamente los mensajes de error o de que una acción finalizó correctamente.
- Errores en las validaciones de datos.
- Acciones que conllevaban a datos duplicados.

3.4.3 Pruebas de rendimiento.

Las pruebas de rendimiento están enfocadas en evaluar el desempeño que tiene determinado sistema o componente. Estas pruebas generalmente se realizan mediante herramientas automatizadas que generan una cantidad de usuarios, carga y volumen de información, monitorizando también el rendimiento del *hardware*.

- **Pruebas de carga:** Dentro de las pruebas de rendimiento, es la más sencilla. Consiste en observar el comportamiento del sistema bajo una cantidad de peticiones. La carga puede ser el número de usuarios de manera concurrente usando la aplicación y que durante el tiempo que dura la carga realizan un número de peticiones. Si se monitorizan durante esta prueba el servidor de aplicaciones y la base de datos entonces puede mostrar el cuello de botella de la aplicación.
- **Pruebas de estrés:** Estas pruebas se utilizan generalmente para colapsar la aplicación. Se va duplicando el número de usuarios que se agregan a la aplicación y se ejecutan pruebas de carga hasta que el sistema falla. Estas pruebas se realizan para determinar la solidez en los momentos de carga extrema del sistema y constituye una ayuda para los administradores en función de conocer si la aplicación rendirá en el caso de que la carga real supere a la carga esperada.

Herramienta utilizada para la realización de las pruebas de carga y estrés.

La realización de estas pruebas se hizo de manera automatizada mediante el *software* JMeter en su versión 2.8.4 la cual es desarrollada en *Java* y permite la realización de pruebas de rendimiento en aplicaciones web. JMeter es un *software* que se encarga de realizar pruebas de carga y estrés de código abierto. En sus inicios fue desarrollado para probar aplicaciones *web*, y se ha expandido a otras funciones de prueba. Se utiliza para probar tanto recursos estáticos como dinámicos y se puede utilizar para simular una carga pesada en un servidor de red, probar su resistencia o analizar el rendimiento ante diferentes tipos de cargas.

Entorno de las pruebas de carga y estrés.

***Hardware* de prueba (PC cliente)**:

- Tipo de procesador: *Intel (R)* Core(TM) i3-2100 CPU @ 3.10Hz
- Memoria RAM: 2,00 GB
- Tipo de Red: Ethernet 10/100Mbps

Hardware de prueba (PC servidor):

- Tipo de procesador: *Intel (R)* Core(TM) i3-2100 CPU @ 3.10Hz
- Memoria RAM: 2,00 GB
- Tipo de Red: Ethernet 10/100Mbps

Software instalado en ambas PC:

- Tipo de servidor web: *Nginx*
- Memoria máxima: 2048 MB
- Máximo de hilos concurrentes: 150
- Plataforma: SO Ubuntu 13.10 *Saucy Salamander*
- Servidor de BD: PostgreSQL v9.1

Resultados de las pruebas de carga y estrés.

Las pruebas de rendimiento fueron ejecutadas con las mínimas prestaciones que se recomiendan, arrojaron resultados que ofrecen un contacto directo con el comportamiento del sistema en situaciones complicadas de funcionamiento. Para llevar a cabo esta ejecución se simuló un total de 150 usuarios conectados concurrentemente, con un período entre una petición y otra de 1 segundo. Bajo estas condiciones el sistema dio los siguientes resultados:

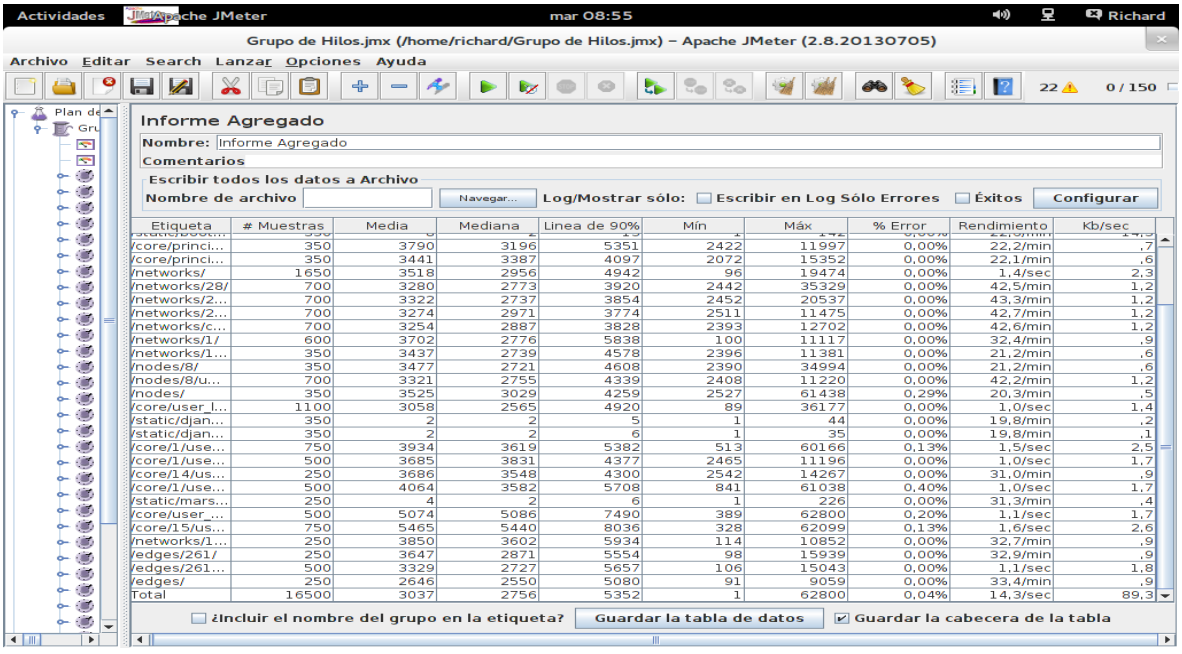


Figura 16 Resultado de la prueba de rendimiento.

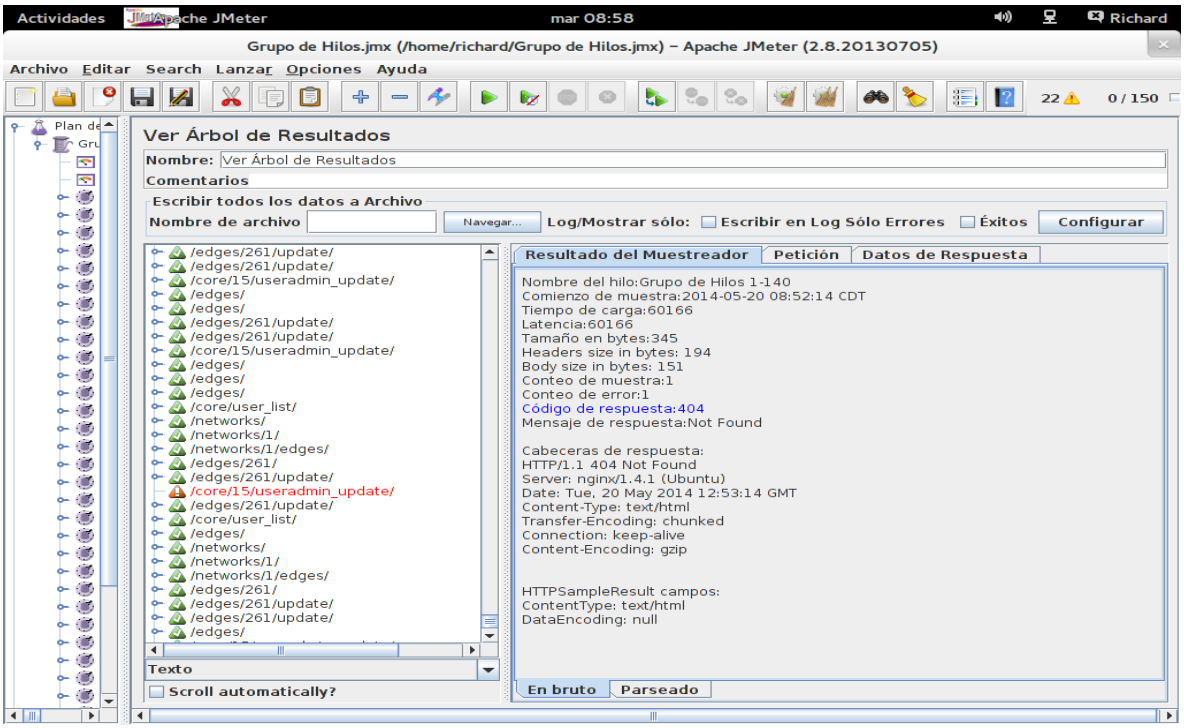


Figura 17 Resultados de las peticiones realizadas a la aplicación y sus respuestas.

Interpretación de resultados de las pruebas de rendimiento

- **Etiqueta:** El nombre de la muestra (conjunto de muestras).
- # **Muestras:** El número de muestras para cada *URL*.
- **Media:** El tiempo medio transcurrido para un conjunto de resultados.
- **Mín:** El mínimo tiempo transcurrido para las muestras de la *URL* dada.
- **Máx:** El máximo tiempo transcurrido para las muestras de la *URL* dada.
- **% Error:** Porcentaje de las peticiones con errores.
- **Rendimiento:** Rendimiento medido en base a peticiones por segundo/minuto/hora.
- **Kb/sec:** Rendimiento medido en Kilobytes por segundo.
- **Media de Bytes:** Tamaño medio de la respuesta de la muestra medido en bytes.

Teniendo en cuenta que se realizaron las pruebas con un total de 150 usuarios concurrentes, con un período de peticiones enviadas al servidor, por los usuarios, de 1 segundo, equivalente a 1 petición por cada 0.0699 segundos aproximadamente. Se puede observar que las pruebas se han realizado con un margen de error igual a 0.04% sobre un total de 16500 peticiones realizadas para un rendimiento aproximado de 14.3 peticiones por segundo. Atendiendo a la cantidad de peticiones por segundo que se enviaron, la

velocidad de respuesta de las peticiones enviadas y las prestaciones del *hardware* donde se realizaron las pruebas, los resultados alcanzados son considerablemente buenos y si se mejoran las características del entorno de prueba se puede pronosticar que se obtendrán resultados de muy alta calidad.

3.5 Conclusiones.

La representación y modelado de cada uno de los elementos del diagrama de componentes permitió una mejor compresión y gestión del mismo. Además la utilización de un estilo de código permitió trabajar con una mayor limpieza y claridad en la implementación, también ayudará a entender con facilidad el código a otros desarrolladores si en un futuro se desea realizar otras versiones del sistema. La realización de las pruebas garantizó que el sistema cumpla con todos los requerimientos funcionales previamente acordados, además garantizan un completo funcionamiento del mismo. Todas las no conformidades detectadas en la fase de pruebas quedaron resueltas.

Conclusiones.

La investigación realizada permitió arribar a las siguientes conclusiones:

- Se realizó la fundamentación teórica del análisis de redes sociales lo que permitió dejar sentadas las bases para el comienzo de la investigación, ya que se seleccionaron las métricas a utilizar, se definieron las bibliotecas y las tecnologías a utilizar.
- También se realizó el diseño del sistema basado en servicios web para el análisis de redes sociales, lo que permitió un mejor entendimiento de cómo funcionaría el sistema, la identificación de los requerimientos funcionales y la arquitectura, así como el modelado de los diagramas de caso de uso, de clases de diseño y de secuencia, y el modelo de datos que proporcionaron un mejor entendimiento del sistema.
- Se implementó el sistema basado en servicios web para el análisis de redes sociales utilizando el estilo de código PEP8 definido para el lenguaje de programación *Python* que aportó mayor claridad y limpieza a la hora de desarrollar el sistema.
- Además se realizaron las pruebas al sistema basado en servicios web para el análisis de redes sociales solucionándose todas las no conformidades detectadas, obteniéndose un producto calificado para su total explotación.

Recomendaciones.

Se recomienda integrar el sistema con otras aplicaciones desplegadas por la Dirección de Redes y Seguridad Informática y otras áreas de la universidad donde se aproveche el análisis de redes sociales. Así también en versiones posteriores evaluar la inclusión de funcionalidades que permitan gestionar el contenido que se publica por parte de los usuarios en la red.

Bibliografía

1. ***Las Tecnologías de la Información y las Comunicaciones (TIC) en la enseñanza de idiomas.*** **Rojas, Orlando Luciano. 29, Bayamo : s.n., Olimpia.**

2. Pérez, Jorge Arturo Pérez. ***La investigación REDES SOCIALES VIRTUALES Y LA BIOÉTICA.*** **Medellín : s.n., 2011. 978-958-8474-23-6.**

3. Roldán, Carlos Santana. ¿Qué es el Análisis de Redes Sociales (Social Network Analysis)? ***Code Jobs Aprende a programar.*** **[En línea] [Citado el: 15 de enero de 2013.] http://www.codejobs.biz/es/blog/2013/04/11/que-es-el-analisis-de-redes-sociales-social-network-analysis#sthash.pOmio76V.dpbs.**

4. ***Social Network Analysis Theory and Applications.*** **2011.**

5. Aaron M. Tenenbaum, Yedidyah Langsam, M. J. A. ***Estructuras de Datos en C.*** **1ra edition. s.l. : Prentice Hall Hispanoamerica, S.A., 1993.**

6. ***Analisis de redes sociales: un tutorial.*** **Moreno, Mauricio Monsalve. 2008.**

7. Marquina-Arenas, Julián. ***Plan Social Media y Community Manager.*** **s.l. : Editorial UOC, 2013. 978-84-9029-855-8.**

8. Derek Hansen, Ben Shneiderman, Marc A. Smith. ***Analyzing Social Media Networks with NodeXL.*** **2011. 978-0-12-382229-1.**

9. Lider de Proyecto. [En línea] [Citado el: 1 de mayo de 2014.] http://www.liderdeproyecto.com/glosario/#m.

10. Ferris, Chris. W3C Working Group Note. [En línea] [Citado el: 1 de mayo de 2014.] http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/..

11. Martínez, Rafael. PostgreSQL-es. [En línea] 02 de 10 de 2010. [Citado el: 17 de diciembre de 2013.] http://www.postgresql.org.es/sobre_postgresql.

12. José Antonio Castillo González, Luis Felipe Matos Marinas. ***SISTEMA INFORMÁTICO PARA LA GESTIÓN DE LOS TESTS FÍSICOS EN LOS ENTRENAMIENTOS DEPORTIVOS.*** **UCI. Ciudad de La Habana : s.n., 2010. pág. 73.**

13. Fielding, Dr. Roy Thomas. ***Architectural Styles and the Design of Network-based.*** **s.l. : Addison Wesley, 2000. ISBN 0-599-87118-0.**

14. Jacobson, Ivar. ***El Proceso Unificado de Desarrollode Software.*** **La Habana : Félix Varela, 2004. 84-7829-036-2.**

15. Delgado, José Miguel Martínez Guerrero / Camilo Andrés Silva. ***Anexo 2 Ingeniería de Requerimientos Guía Metodológica para el levantamiento y análisis de Requerimientos de Software en base a Procesos de Negocio.*** **PONTIFICIA UNIVERSIDAD JAVERIANA : s.n., 2010.**

16. Sommerville, Ian. ***Ingeniería de Software.*** **La Habana : Félix Varela, 2005. 84-7829-074-5.**

17. Larman, Craig. ***UML y Patrones. Una introducción al análisis y diseño orientado a.*** **La Habana : Félix Varela, 2004. 842-053-438-2.**

18. Overgaard, Gunnar y Palmvist, Karin. ***Use Cases- Patterns and Blueprints.*** **2005.**

19. ***Ayuda Extendida de Rational Rose Enterprise Edition. .*** **2003.**

Bibliografía.

20. Legg, Jesse. ***Django 1.2 e-commerce.*** **s.l. : Packt Publishing Ltd, 2010. pág. 244. ISBN 978-1-847197-00-9.**

21. Thelma Jefferson Betancourt, Dayami Pompa Hechavarría. ***Sistema para la Planificación de Pruebas dentro del Grupo de Calidad.*** **La Habana : s.n., 2010.**

22. González, Fuertes G. ***Psicología Comunitaria.*** **España : s.n., 1988.**

23. Pressman, Roger. ***Ingeniería del Software. Un enfoque práctico.*** **La Habana : Félix Varela, 2005.**

24. Garcerant, Iván. Tecnología y Synergix. [En línea] 2008. [Citado el: 15 de mayo de 2014.] http://synergix.wordpress.com/2008/03/15/definimos-pruebas-de-unidad-como./.

25. Mansilla, Ricardo. Slideshare. [En línea] [Citado el: 15 de mayo de 2014.] http://www.slideshare.net/cliceduca/pruebas-de-software-2420588.

26. Díaz, Ivis Pages. ***Diseño y aplicación de pruebas a la Plataforma Soberana GeneSIG.*** **La Habana : s.n., 2012.**

27. Sueiras, Edita. Scribd. ***Scribd.*** **[En línea] 2013. http://es.scribd.com/doc/24658747/Redes-sociales-definicion.**

28. The Intelligent Development Environment for Python Programmers - Wingware Python IDE. [En línea] WINGWARE, 2013. [Citado el: 15 de diciembre de 2013.] http://wingware.com/.

29. Guzman, Ricardo. Metodologías de desarrollo ágil. [En línea] Academia.edu. [Citado el: 15 de diciembre de 2013.] http://www.academia.edu/1740166/Metodologias_de_desarrollo_agil.

30. Open UP: Metodología Open UP. ***Open UP: Metodología Open UP.*** **[En línea] septiembre de 2013. [Citado el: 15 de diciembre de 2013.] http://openupeaojmp.blogspot.com/2013/09/metodologia-open-up.html.**

31. Python Software Foundation. Python Programming Language- Official WebSite. ***Python Software Foundation. Python Programming Language- Official WebSite.*** **[En línea] 2013. [Citado el: 17 de diciembre de 2013.] http://www.python.org/about/.**

32. Django web framework | Django en Español, django-es. ***Django web framework | Django en Español, django-es.*** **[En línea] [Citado el: 2013 de diciembre de 2013.] http://django.es/.**

33. django CMS – The easy-to-use and developer-friendly CMS. ***django CMS – The easy-to-use and developer-friendly CMS.*** **[En línea] 2013. [Citado el: 18 de diciembre de 2013.] https://www.django-cms.org/en/.**

34. Hourieh, Ayman. ***Django 1.0 Web Site Development.*** **s.l. : Pack Publishing, 2009. pág. 272. ISBN 978-1-847196-78-1.**

35. Heilman, Gregory L. ***Estructuras de datos, algoritmos, y programación orientada a objetos.*** **La Habana : Félix Varela, 2003. pág. 305. 84-481-1173-7.**

36. Radcliffe-Brown, A.R. ***On Social Structure.*** **s.l. : Journal of the Royal Anthropological Institute: 70, 1940.**

37. Nadel, SF. ***The Theory of Social Structure. .*** **London : Cohen and West., 1957.**

38. Granovetter, Mark. ***Introduction for the French Reader.*** **2007.**

39. ***Medios de comunicación y solidaridad: reflexiones entorno a la (des)articulicación social.*** **Ed. Universitat Jaume I, España : s.n., 2006.**

Bibliografía.

40. Celery: Distributed Task Queue. [En línea] Celery, 2014. [Citado el: 23 de enero de 2014.] http://www.celeryproject.org/.

41. Cramer, David. *Sentry Documentation Release 5.4.1.* 2013. pág. 67.

42. L. CUADERNO, O. *Herramientas de soporte al proceso de desarrollo dirigido por modelos y su implementación con DSL Tools.* Argentina. : s.n.

43. *Modelado de Sistemas con UML.*

44. UML, BPMN and Enterprise Architecture Tool for Software Development. [En línea] [Citado el: 23 de enero de 2014.] http://www.visual-paradigm.com/.

45. Franz Inc. [En línea] Franz Inc., 2014. [Citado el: 2014 de enero de 15.] http://franz.com/agraph/allegrograph/.

46. Gephi make graph handy. *Gephi.org.* [En línea] 2012. [Citado el: 15 de enero de 2014.] https://gephi.org/.

47. GraphStream A Dynamic Graph Library. *GraphStream.* [En línea] 2013. [Citado el: 15 de enero de 2014.] http://graphstream-project.org/.

48. graph-tool Efficient network analysis. [En línea] http://graph-tool.skewed.de/.

49. Graphviz - Graph Visualization Software. [En línea] [Citado el: 15 de enero de 2014.] http://www.graphviz.org/.

50. JUNG JAVA UNIVERSAL NETWORK/GRAPH FRAMEWORK. [En línea] [Citado el: 15 de enero de 2014.] http://jung.sourceforge.net/.

51. NodeXL Network Overview, Discovery and Exploration. [En línea] CodePlex, 2014. http://nodexl.codeplex.com/.

52. Netminer4. [En línea] 2012. [Citado el: 15 de enero de 2014.] http://www.netminer.com/index.php.

53. Networkx. [En línea] 2013. [Citado el: 15 de enero de 2014.] http://networkx.github.io/.

54. sigmajs. *sigmajs.* [En línea] [Citado el: 15 de enero de 2014.] http://sigmajs.org/.

55. Data Visualization Software Tulip. *Data Visualization Software Tulip.* [En línea] [Citado el: 15 de enero de 2014.] http://tulip.labri.fr/TulipDrupal/.

56. Henry Fretzen, Jeff y Sobotka. *Superutilidades para JavaScript.* Madrid : MacGRAW-HILLINTERAMERICAN, 2004. 0-07-882364-1.

57. Ivar Jacobson, Grady Booch, James Rumbaugh. *El Proceso Unificado de Desarrollo de Softaware. .* 2005.

58. IBM. IBM. [En línea] 2010. [Citado el: 10 de 02 de 2014.] . http://www.ibm.com/ .

59. GSINNOVA. Grupo de Soluciones. . [En línea] 2011. [Citado el: 10 de 02 de 2014.] http://www.rational.com.ar/index.html .

60. visual-paradigm. . [En línea] http://www.visual-paradigm.com/ visual paradigm..

61. José H. Canós, Patricio Letelier yMª Carmen Penadés. *Métodologías Ágiles en el Desarrollo de Software.* s.l. : Universidad Politécnica de Valencia.

62. Fernández., Mayté Sánchez. *Análisis y diseño del componente alertas y avisos del Marco de Trabajo Sauxe.* Ciudad de la Habana : UCI, 2011. pág. 84.

Bibliografía.

63. NetBeans IDE. [En línea] 2013. https://netbeans.org/features/index.html.

64. [En línea] https://sites.google.com/site/redessociales93/historia-de-analisis-de-redes-sociales.

65. Noel Rodríguez Arias, Lissete González Gallo, Joselín Miló Pérez, Ruber González Pedraza, Alexander Rodríguez Rabelo. INTEGRACIÓN DE LAS APLICACIONES alasBQO Y GALEN A TRAVÉS DE LOS SERVICIOS . [En línea] [Citado el: 1 de mayo de 2014.] http://repositorio_institucional.uci.cu/jspui/handle/ident/4004.

66. Otto, Mark. Developer Blog. [En línea] Twitter, 19 de agosto de 2011. [Citado el: 1 de mayo de 2014.] https://blog.twitter.com/2011/bootstrap-twitter.

67. FIELDING, Roy Thomas. *Architectural Styles and the Design of Network-based Software Architectures.* Irvine : University of California, 2000.

68. RESTful Web services: The basics. [En línea] 2008. [Citado el: 10 de mayo de 2014.] http://www.ibm.com/developerworks/webservices/library/ws-restful/.

69. Larman, Craig. *UML y Patrones Introducción al análisis y diseño orientado a objetos.* s.l. : Prentice Hall, 1999. 970-17-0261-1.

Glosario de Términos.

API: interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

Biblioteca: En ciencias de la computación, una biblioteca (o librería) es un conjunto de subprogramas utilizados para desarrollar *software*. Las bibliotecas contienen código y datos, que proporcionan servicios a programas independientes, es decir, pasan a formar parte de éstos.

Blogs: Es un sitio *web* que recopila cronológicamente textos o artículos de uno o varios autores, apareciendo primero el más reciente, donde el autor conserva siempre la libertad de dejar publicado lo que crea pertinente. En cada artículo, los lectores pueden escribir sus comentarios y el autor darles respuesta

Framework: Un *framework* se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta.

IDE: Entorno de Desarrollo Integrado, es un programa compuesto por un conjunto de herramientas para un programador. Puede ser exclusivo de un lenguaje o puede ser utilizado en más de uno.

JSON: (*JavaScript Object Notation*) Es un formato alternativo de envío y recepción de datos, es decir remplaza a XML o el envío de texto plano. Este formato de datos es más liviano que XML.

Lenguaje de programación: Conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Metodología de desarrollo de *software*: Es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo de *software*.

Motor de búsqueda: Programa que selecciona y jerarquiza resultados entre las entradas de una base de datos en función de unos términos y criterios de búsqueda.

Open source: es una marca de certificación propiedad de la *Open Source Initiative*. El *software Open Source* se define por la licencia que lo acompaña, que garantiza a cualquier persona el derecho de usar, modificar y redistribuir el código libremente.

Paquete: Un paquete de *software* es una serie de programas que se distribuyen conjuntamente. Algunas de las razones suelen ser que el funcionamiento de cada uno complementa a o requiere de otros.

Plugin: Un *plugin* o componente enchufable es una aplicación informática que interactúa

Renderizado: Proceso de generar una imagen a partir de un modelo, usando una aplicación de computadora.

Sistema: Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, *hardware*, *software* y de Recurso Humano. Un sistema informático típico emplea una computadora que usa dispositivos programables para capturar, almacenar y procesar datos.

Sistema gestor de bases de datos: Es el *software* que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

Teoría de redes: Teoría que constituye una técnica matemática que ha aportado una ayuda eficaz en el tratamiento de los problemas de transportación de la producción.

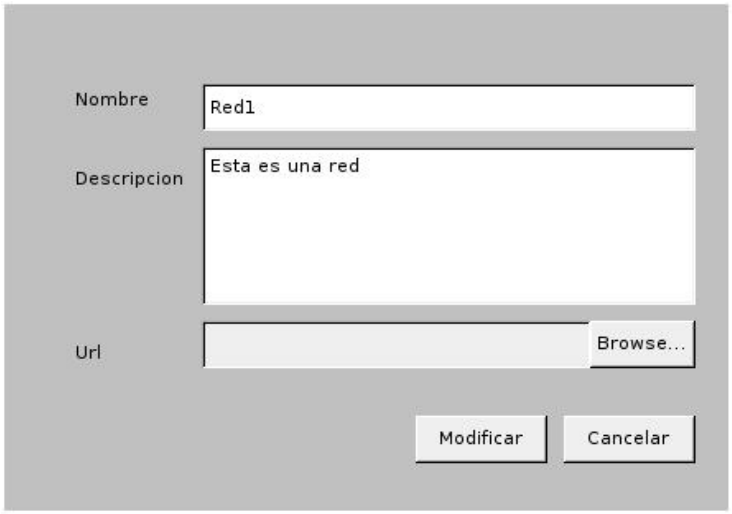
Tecnologías de la Información y las Comunicaciones: conjunto de servicios, redes, *software* y dispositivos que tienen como fin la mejora de la calidad de vida de las personas dentro de un entorno, y que se integran a un sistema de información interconectado y complementario.

Wiki: Es una herramienta que permite hacer páginas *web* de una forma fácil y rápida de manera colaborativa, puede ser editado por varios usuarios.

Anexo # 1. Descripciones de casos de uso.

Sección 3: "Modificar Red"		
Flujo básico Modificar Red		
	Actor	**Sistema**
1	Modifica los campos correspondientes de la red: 1. Nombre de la red. 2. Descripción de la red 3. Red. 4. Ruta del archivo. Pulsa el botón modificar.	Valida los datos entrados por el usuario.
2		Registra los datos modificados de la red. Muestra el mensaje "La red se ha modificado correctamente".
3		Redirecciona a la vista "Listar redes". Ver caso de uso Listar redes.
4		Termina sección 3.
Flujos alternos		
Nº 1 ***El usuario cancela la edición de la red***		
	Actor	**Sistema**
1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar redes".
2		Vuelve al paso 2 del flujo básico del caso de uso.
Nº 2 ***Los datos modificados son incorrectos***		
	Actor	**Sistema**
1		Muestra un mensaje "Error: Algunos datos son incorrectos."
2	Corrige los datos incorrectos y pulsa el botón "Modificar red".	Vuelve al paso 2 del flujo básico de la sección Modificar red.
3		Termina sección 3.

Prototipo



Sección 4: "Eliminar Red"

Flujo básico eliminar red

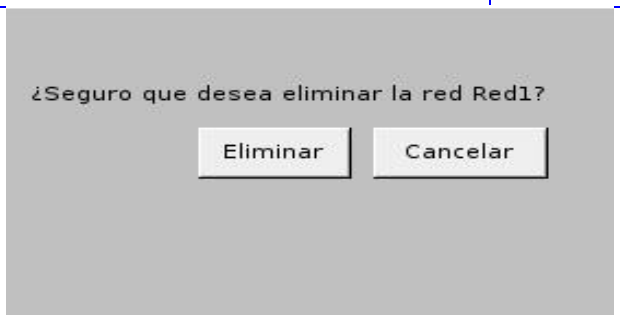
	Actor	Sistema
1	Pulsa el botón eliminar.	Muestra la vista eliminar red.
2	Selecciona la red que desea eliminar	Muestra el mensaje "Esta seguro que desea eliminar la red seleccionada"
3	Confirma que desea eliminar la red seleccionada	Elimina los datos de la red. Muestra el mensaje "La red se ha eliminado correctamente".
4		Redirecciona a la vista "Listar redes"-
5		Termina Sección 4.

Flujos alternos

Nº 1 *El usuario cancela la edición de la red*

	Actor	Sistema
2.1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar redes".
2.2		Termina Sección 4.

Prototipo



Sección 5: "Visualizar Red"		
Flujo básico Visualizar Red		
	Actor	**Sistema**
1	Pulsa el enlace Visualizar red.	Muestra la vista gráfica Visualizar red.
2		Termina Sección 5.
Relaciones	**CU extendidos**	Calcular Métricas, Paso 5 del flujo básico

Tabla 5 Descripción de caso de usos Gestionar Red (Continuación).

CU # 1. Autenticar Usuario	
Objetivo	Entrar en el sistema con usuario y una contraseña para poder acceder a sus principales funcionalidades.
Actores	Usuario
Resumen	El caso de uso inicia cuando el actor (en lo adelante usuario) va a acceder a la vista principal del sistema el cual debe introducir el usuario que lo identifica y su contraseña.
Complejidad	Baja
Prioridad	Opcional
Precondiciones	RF-1
Postcondiciones	Una vez que el usuario introduzca con éxito su usuario y su contraseña el sistema debe garantizar todas las funcionalidades disponibles para el rol que representa el usuario.
Flujo de eventos	
Flujo básico <Autenticar Usuario>	

	Actor	**Sistema**
1	El usuario introduce su usuario y contraseña al sistema.	Verifica si los datos introducidos por el usuario están correctos.
2		Se termina el Caso de Uso.
3		

Flujos alternos	
Nº 1 Datos incorrectos	

	Actor	Sistema
2.1		Si los datos del usuario no están correctos muestra un mensaje “El nombre de usuario o la contraseña son incorrectos.”
2.2		Vuelve al paso 1 del flujo básico del caso de uso.
Prototipo		

Tabla 6 Descripción de caso de usos Autenticar Usuario.

CU # 2. Cambiar Contraseña	
Objetivo	Cambiar la contraseña del usuario.
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario decide cambiar la contraseña con que el accede al sistema
Complejidad	Baja
Prioridad	Opcional
Precondiciones	RF-2
Postcondiciones	La contraseña es cambiada satisfactoriamente
Flujo de eventos	
Flujo básico Cambiar Contraseña	

	Actor	Sistema
1	El usuario selecciona la opción Cambiar contraseña.	Muestra dos campos de texto donde se ponen la contraseña nueva y se confirma esa misma contraseña.
2	El usuario escribe la nueva contraseña y la repite de nuevo en el otro campo de texto.	Verifica que la contraseña escrita cumpla con todos los parámetros.
3		Cambia la contraseña.

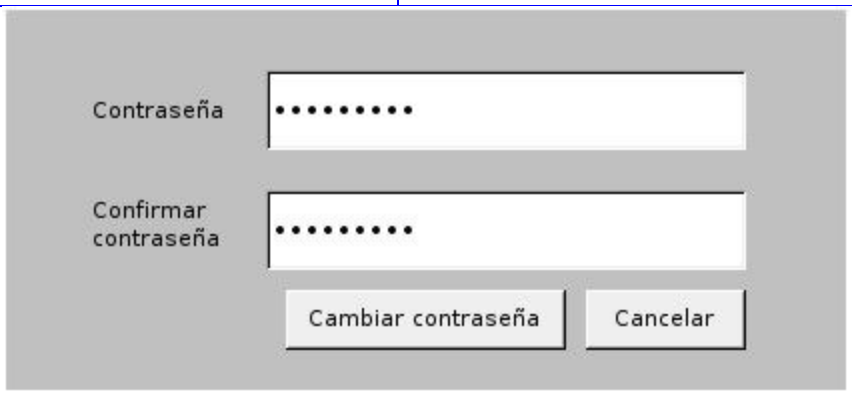
4		Muestra una notificación del cambio de la contraseña.
5		Se termina el caso de uso.
Flujos alternos		
Nº 1 Datos incorrectos		
	Actor	**Sistema**
2.1		Muestra un mensaje "La contraseña no cumple con los parámetros"
2.2		Vuelve al paso 2 del flujo básico del caso de uso.
Prototipo		

Tabla 7 Descripción de caso de usos Cambiar Contraseña.

CU # 3. Modificar Perfil de Usuario	
Objetivo	Modificar el perfil del usuario común
Actores	Usuario Común
Resumen	El caso de uso inicia cuando el usuario decide modificar sus datos personales.
Complejidad	Baja
Prioridad	Opcional
Precondiciones	RF-27
Postcondiciones	Los datos del usuario son cambiados satisfactoriamente

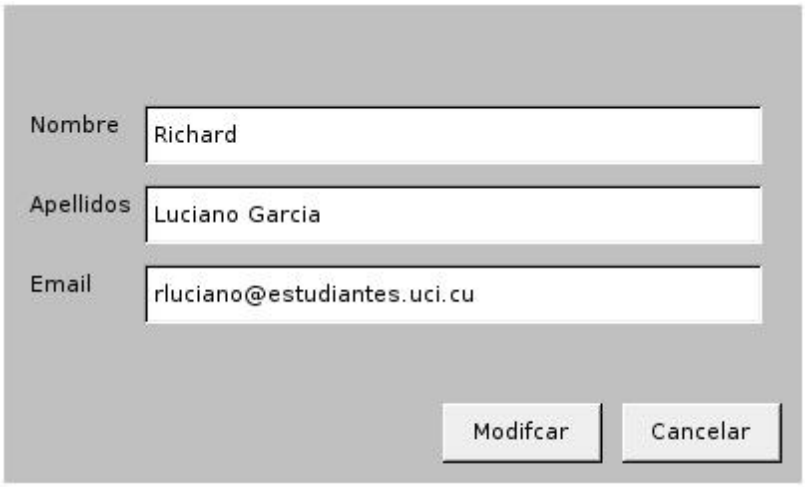
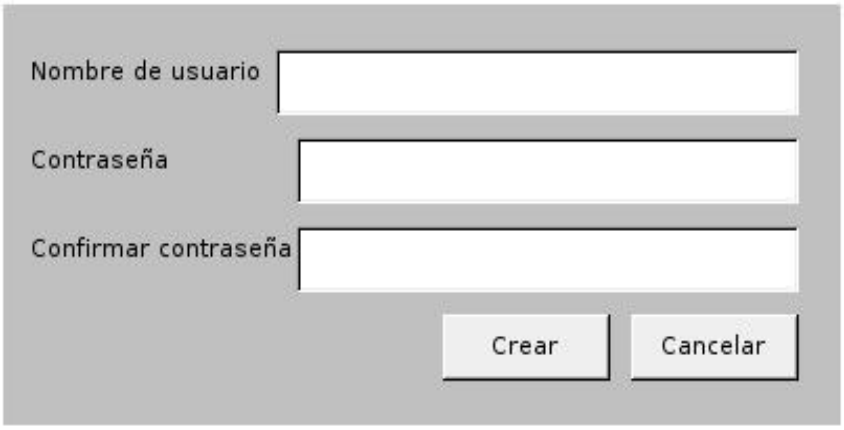
Flujo de eventos		
Flujo básico Modificar Perfil de Usuario		
	Actor	**Sistema**
1	El usuario selecciona la opción de modificar perfil de usuario.	Muestra los campos correspondientes con la opción seleccionada
2	Escribe sus nuevos datos personales.	Verifica que los datos no estén en la base datos.
3		Modifica los datos.
4		Muestra una notificación del cambio de los datos del perfil.
		Se termina el caso de uso.
Flujos alternos		
Nº 1 Datos incorrectos		
	Actor	**Sistema**
2.1		Muestra un mensaje “Los datos ya existen”
2.2		Vuelve al paso 2 del flujo básico del caso de uso.
Prototipo		

Tabla 8 Descripción de caso de usos Modificar Perfil de Usuario.

CU # 4. Gestionar Perfil		
Objetivo	Gestionar los perfiles de los usuarios	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario según su nivel de acceso tenga que hacer operaciones sobre los perfiles, se muestra un panel con las opciones Adicionar Perfil, Modificar Perfil y Eliminar Perfil. El usuario escoge la tarea que desea realizar.	
Complejidad	Baja	
Prioridad	Opcional	
Precondiciones	RF-3 RF-4, RF-5	
Postcondiciones	El perfil es gestionado por el usuario con éxito	
Flujo de eventos		
Flujo básico Gestionar Perfil		
	Actor	**Sistema**
1	Pulsa el enlace correspondiente a la opción perfil.	Permite realizar varias acciones relacionada con la gestión del perfil. ✓ Adicionar perfil. ✓ Modificar perfil ✓ Eliminar perfil.
2	El usuario selecciona la opción en la que desea trabajar. Trabaja con los datos correspondientes con el perfil.	Verifica los datos.
3		Actualiza los cambios.
4		Se termina el caso de uso.
Sección 1: "Adicionar Perfil"		
Flujo básico Adicionar perfil		
	Actor	**Sistema**
1	Accede a la opción Adicionar perfil.	Muestra la vista correspondiente a la opción Adicionar perfil.
2	Introduce los datos.	Verifica si los datos están correctos
3		Introduce el nuevo perfil en el sistema.
4		Termina sección 1
Flujos alternos		

Nº 1 Datos Incorrectos		
	Actor	**Sistema**
2.1		Muestra un mensaje “Los datos ya existen en el sistema.”
2.2		Vuelve al paso 2 del flujo básico de la sección 1 del caso de uso.



Sección 2: “Modificar Perfil”

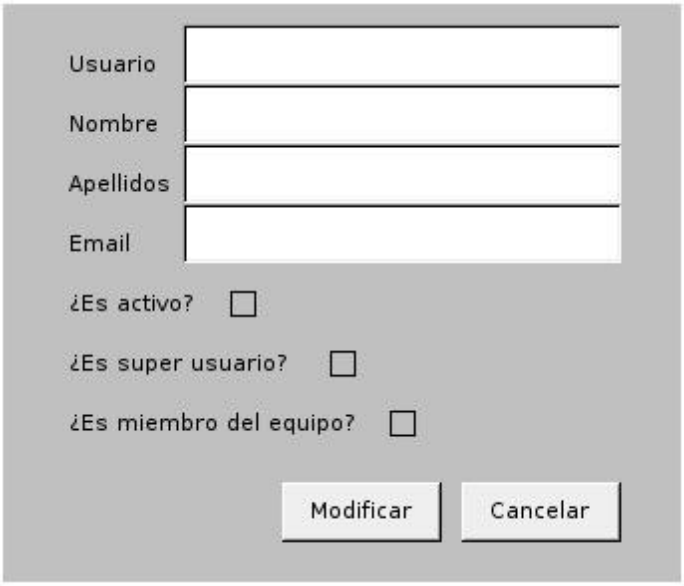
Flujo básico Modificar perfil

	Actor	**Sistema**
1	Accede a la opción modificar perfil.	Muestra la vista correspondiente a la opción modificar perfil.
2	Introduce los datos.	Verifica si los datos están correctos
3		Introduce el perfil modificado en el sistema.
4		Termina sección 2

Flujos alternos

Nº 1 Datos Incorrectos

	Actor	**Sistema**
2.1		Muestra un mensaje “Los datos ya existen en el sistema.”
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.

Prototipo	

Sección 3: "Eliminar Perfil"

Flujo básico Eliminar Perfil

	Actor	Sistema
1	Accede a la opción eliminar perfil.	Muestra la vista correspondiente a la opción eliminar perfil.
2	Selecciona el perfil que quiere eliminar.	Muestra un mensaje "Está seguro que sesea eliminar el usuario."
3	Confirma que desea eliminar el usuario seleccionado	Elimina el perfil.
4		Termina sección 3

Flujos alternos

Nº 1 El usuario pulsa el botón cancelar

	Actor	Sistema
1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar nodos".
2		Vuelve al paso 2 del flujo básico del

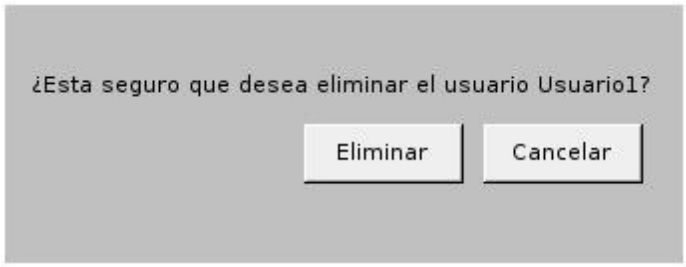
		caso de uso.
Prototipo		

Tabla 9 Descripción de caso de usos Gestionar Perfil de Usuario.

CU # 4. Gestionar Nodo		
Objetivo	Gestionar los nodos de las redes	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario decida Adicionar, modificar, listar y eliminar un nodo, el cual representa un usuario de una red.	
Complejidad	Baja	
Prioridad	Opcional	
Precondiciones	RF-6, RF-7, RF-8, RF-9	
Postcondiciones	El nodo es gestionado por el usuario con éxito	
Flujo de eventos		
Flujo básico Gestionar Nodo		
	Actor	**Sistema**
1	Pulsa el enlace correspondiente a la opción Nodo.	Permite realizar varias acciones relacionada con la gestión del Nodo. ✓ Adicionar Nodo. ✓ Modificar Nodo. ✓ Listar Nodo ✓ Eliminar Nodo.
2	El usuario selecciona la opción en la que desea trabajar. Trabaja con los datos correspondientes con el perfil.	Verifica los datos.
3		Actualiza los cambios.
4		Se termina el caso de uso.
Sección 1: "Adicionar Nodo"		

Flujo básico Adicionar nodo

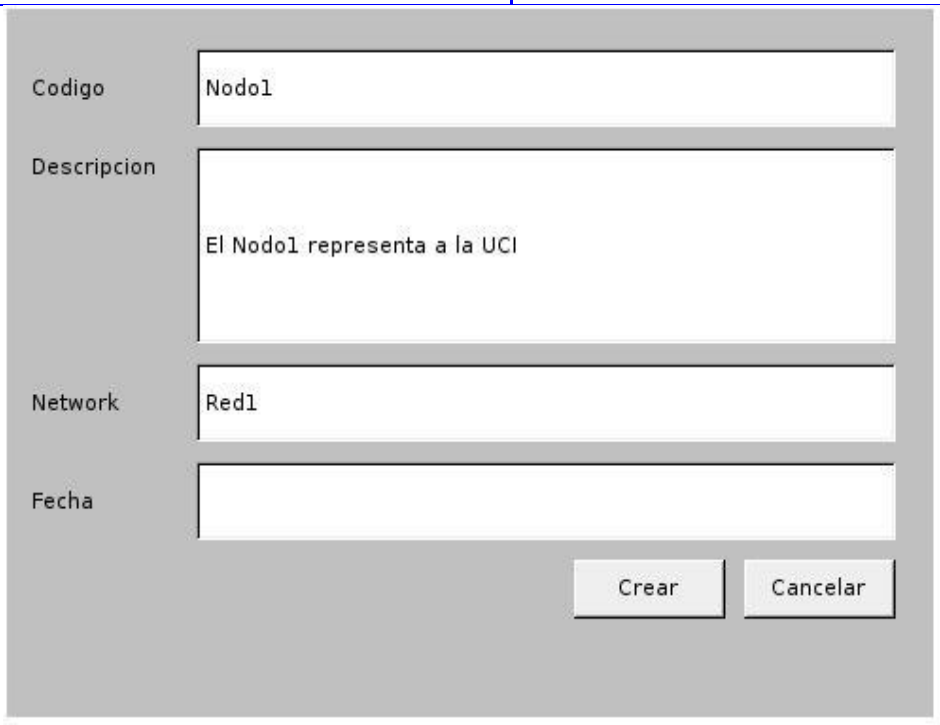
	Actor	Sistema
1	Accede a la opción Adicionar Nodo.	Muestra la vista correspondiente a la opción Adicionar nodo.
2	Introduce los datos. ✓ Código ✓ Descripción ✓ Red ✓ Fecha	Verifica si los datos están correctos
3		Introduce el nuevo nodo.
4		Termina sección 1

Flujos alternos

Nº 1 Datos incorrectos

	Actor	Sistema
2.1		Muestra un mensaje "El Nodo que quiere Adicionar existe."
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.

Prototipo



Sección 2: "Modificar Nodo"

Flujo básico Modificar nodo

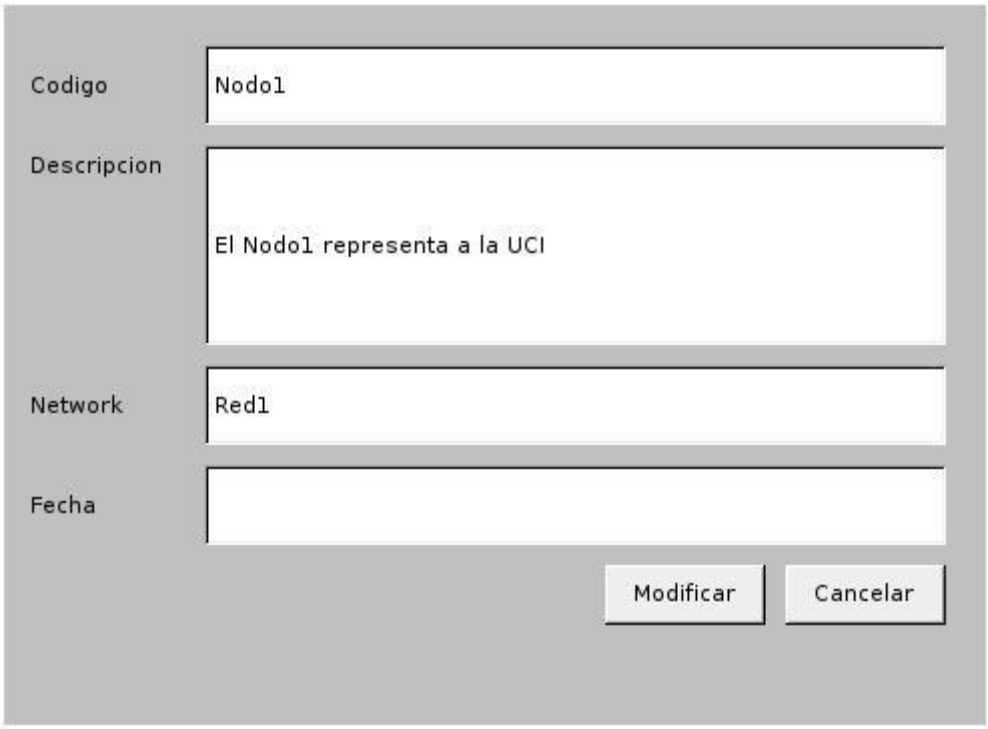
	Actor	Sistema
1	Accede a la opción modificar nodo.	Muestra la vista correspondiente a la opción modificar nodo.
2	Introduce los datos. 1. Código 2. Descripción 3. Red 4. Fecha	Verifica si los datos están correctos
3		Actualiza los datos del nodo modificado en el sistema.
4		Termina sección 2

Flujos alternos

Nº 1 Datos incorrectos

	Actor	Sistema
2.1		Muestra un mensaje "Error al modificar el nodo seleccionado."
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.

Prototipo

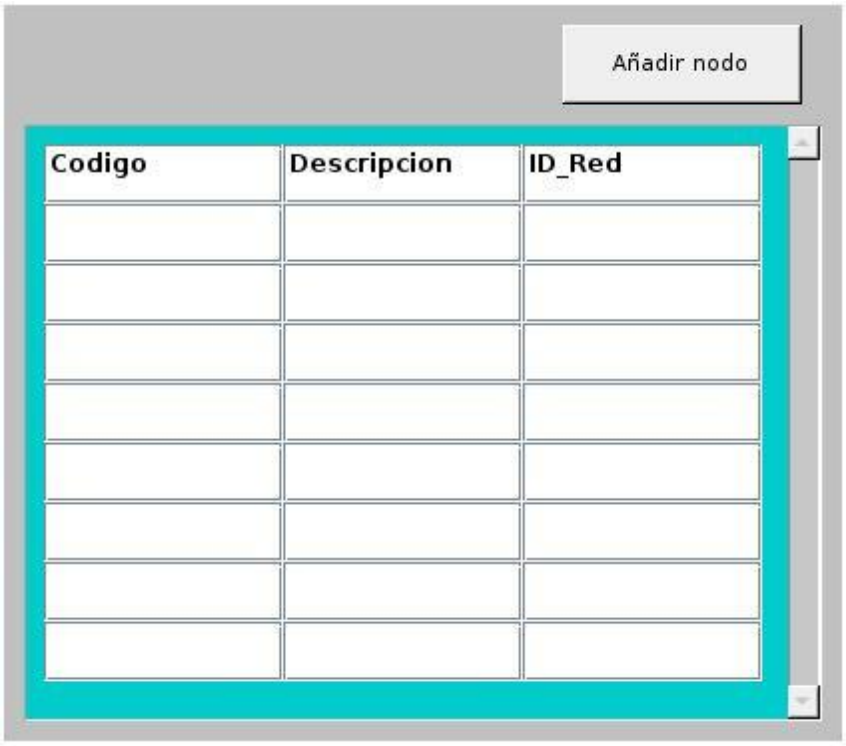


Sección 3: "Listar Nodo"

Flujo básico Listar nodo

	Actor	Sistema
1	Accede a la opción listar nodo.	Muestra la vista correspondiente a la opción listar nodo.
2		Termina sección 3

Prototipo



Sección 4: "Eliminar Nodo"

Flujo básico Eliminar nodo

	Actor	Sistema
1	Accede a la opción eliminar nodo.	Muestra la vista correspondiente a la opción eliminar nodo.
2	Selecciona el nodo que quiere eliminar.	Muestra un mensaje "Está seguro que sesea eliminar el nodo."
	Confirma que desea eliminar el nodo seleccionado	Elimina el nodo.
		Termina sección 4

Flujos alternos

Nº 1 El usuario pulsa el botón cancelar

	Actor	Sistema
1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar nodos".

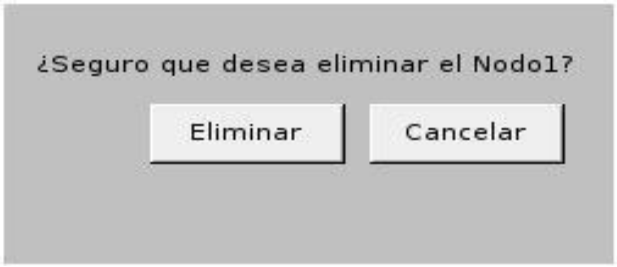
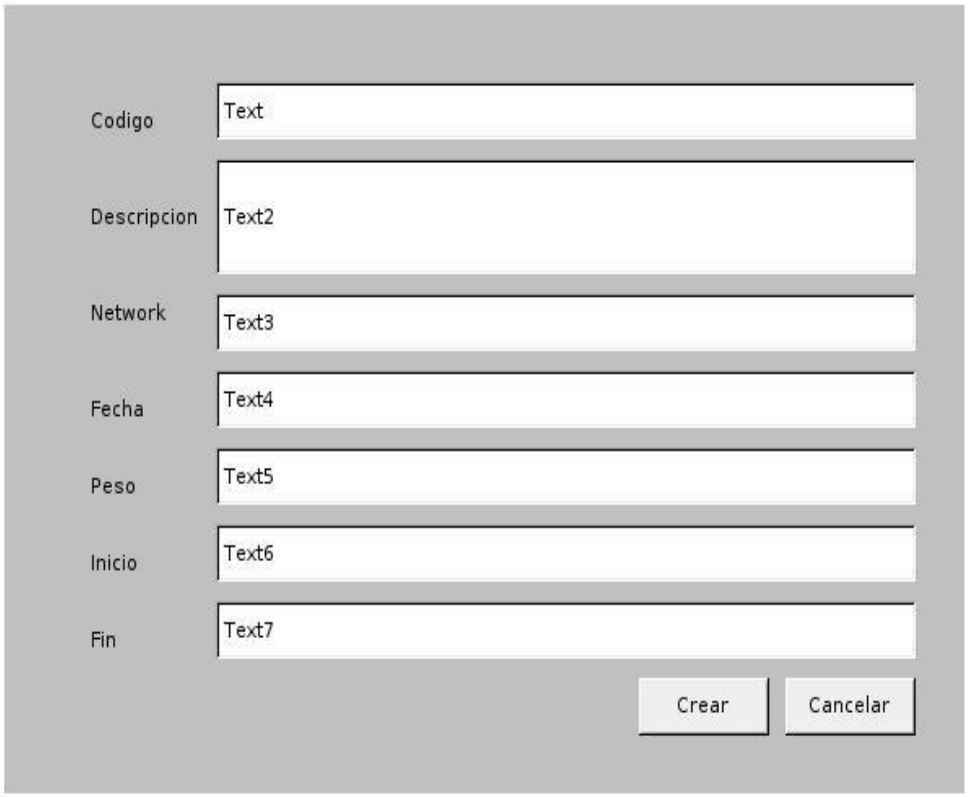
2		Vuelve al paso 2 del flujo básico del caso de uso.
Prototipo		

Tabla 10 Descripción de caso de usos Gestionar Nodo.

CU # 5. Gestionar Arista		
Objetivo	Gestionar las aristas que relacionan a los nodos	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario decida Adicionar, modificar, listar y eliminar una arista, el cual representa un usuario de una red.	
Complejidad	Baja	
Prioridad	Secundario	
Precondiciones	RF-10, RF-11, RF-12, RF-13	
Postcondiciones	La arista es gestionada por el usuario con éxito	
Flujo de eventos		
Flujo básico Gestionar Arista		
	Actor	**Sistema**
1	Pulsa el enlace correspondiente a la opción Arista.	Permite realizar varias acciones relacionada con la gestión de una Arista. ✓ Adicionar Arista. ✓ Modificar Arista. ✓ Listar Arista ✓ Eliminar Arista.
2	El usuario selecciona la opción en la que desea trabajar.	Verifica los datos.
3		Actualiza los cambios.
4		Se termina el caso de uso.

Sección 1: “Adicionar Arista”		
Flujo básico Adicionar arista		
	Actor	**Sistema**
1	Accede a la opción Adicionar Arista.	Muestra la vista correspondiente a la opción Adicionar Arista.
2	Introduce los datos correspondientes a la nueva arista.	Verifica si los datos están correctos
3		Introduce la nueva Arista.
4		Termina sección 1
Flujos alternos		
Nº 1 Datos incorrectos		
	Actor	**Sistema**
2.1.		Muestra un mensaje “Los datos son incorrectos”
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.
Flujos alternos		
Nº 2 Datos existentes		
2.1.		Muestra un mensaje “La arista ya existe”
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.
Prototipo		



Sección 2: "Modificar Arista"

Flujo básico Modificar arista

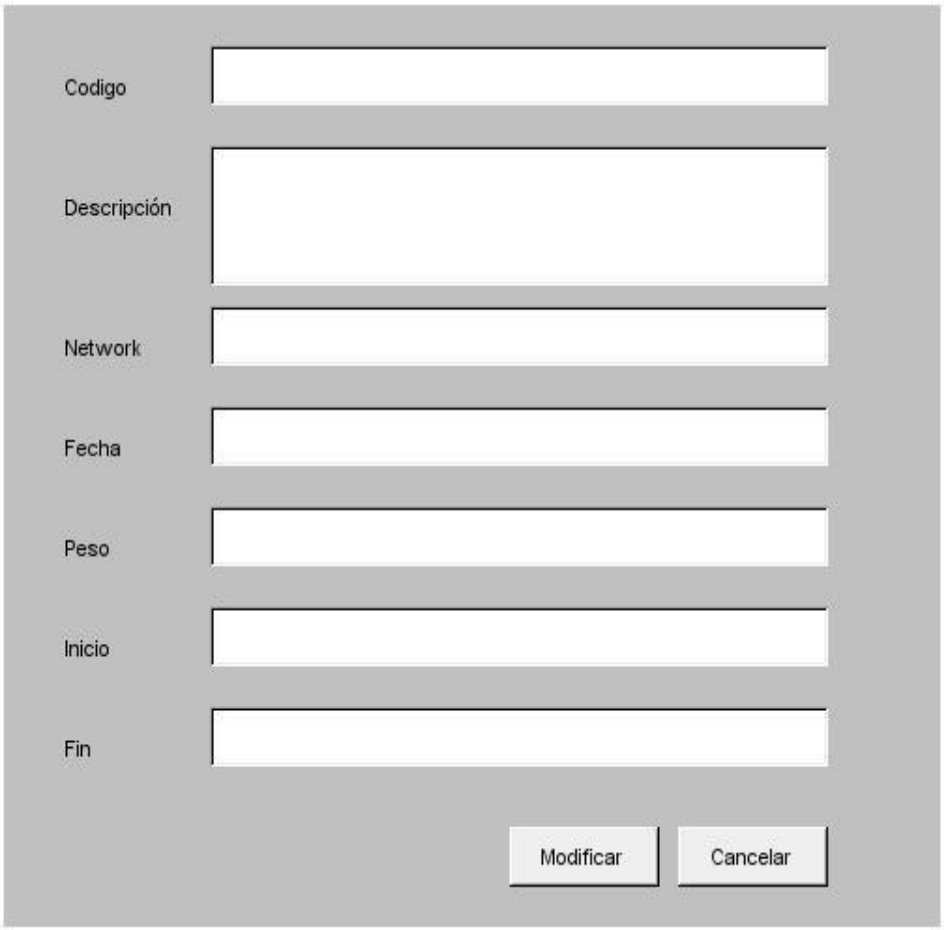
	Actor	Sistema
1	Accede a la opción modificar Arista.	Muestra la vista correspondiente a la opción modificar arista.
2	Introduce los datos correspondientes con la arista que se quiere modificar.	Verifica si los datos están correctos
3		Actualiza los datos de la arista modificada en el sistema.
4		Termina sección 2

Flujos alternos

Nº 1 Datos incorrectos

	Actor	Sistema
2.1		Muestra un mensaje "Error al modificar la arista seleccionada."
2.2		Vuelve al paso 2 del flujo básico de la sección 2 del caso de uso.

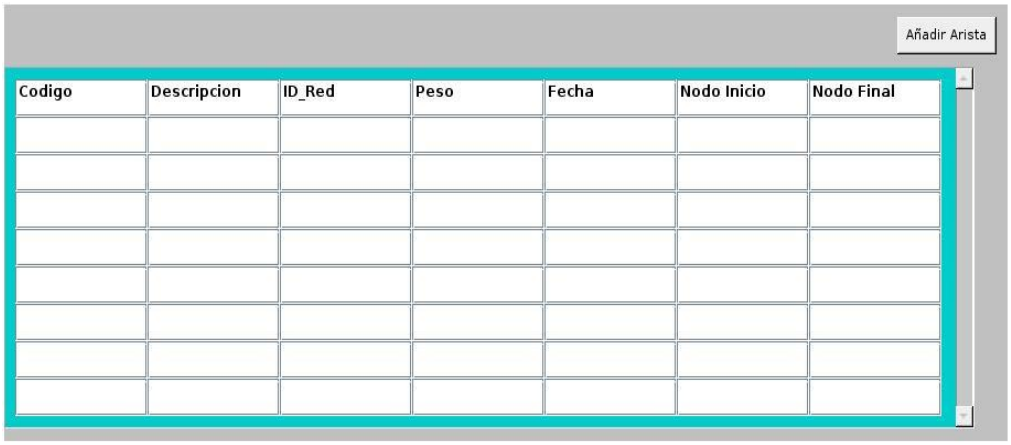
Prototipo	



Sección 3: "Listar Arista"

Flujo básico Listar Arista

	Actor	Sistema
1	Accede a la opción listar arista.	Muestra la vista correspondiente a la opción listar arista.
2		Termina sección 3

Prototipo	

Sección 4: "Eliminar Arista"

Flujo básico Eliminar Arista

	Actor	Sistema
1	Accede a la opción eliminar arista	Muestra la vista correspondiente a la opción eliminar arista.
2	Selecciona la arista que quiere eliminar.	Muestra un mensaje "Está seguro que desea eliminar la arista."
3	Confirma que desea eliminar la arista seleccionada	Elimina la arista.
4		Termina sección 4

Flujos alternos

Nº 1 El usuario pulsa el botón cancelar

	Actor	Sistema
1	Pulsa el botón cancelar.	Redirecciona a la vista "Listar aristas".
2		Vuelve al paso 2 del flujo básico del caso de uso.

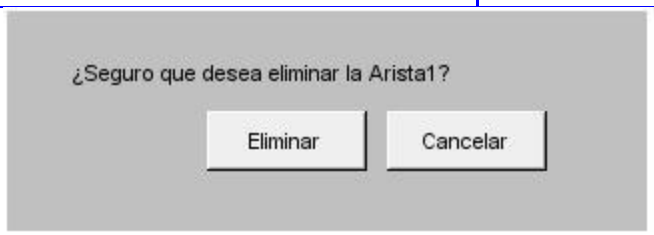
Prototipo	

Tabla 11 Descripción de caso de usos Gestionar Arista.

CU # 7. Gestionar Comunidad	
Objetivo	Gestionar las comunidades que se generen
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario decida modificar o listar una comunidad en específico, finalizando cuando se ejecuta la acción.
Complejidad	Baja
Prioridad	Opcional
Precondiciones	RF-14
Postcondiciones	La comunidad es modificada por el usuario con éxito

Flujo de eventos

Flujo básico Gestionar Comunidad

	Actor	Sistema
1	El usuario selecciona la opción Gestionar Comunidad.	Carga la página con los diferentes campos de textos que pertenecen a la Gestión de una Comunidad. Como son: ✓ Listar Comunidad. ✓ Modificar Comunidad Además permite aplicar el cálculo de las métricas
2	El usuario llena todos los campos de textos con la información necesaria para modificar los datos.	Valida que los datos estén correctamente.
3		Actualiza los cambios
4		Se termina el caso de uso.

Flujos alternos

Nº Evento Datos incorrectos

	Actor	Sistema
3.1		Muestra un mensaje de error “Los datos están incorrectos”.
3.2		Vuelve al paso 2 del flujo básico del caso de uso.

Sección 1: “Listar Comunidad”

Flujo básico listar comunidad

	Actor	Sistema
1	Decide obtener una lista de las comunidades que existen en una red.	Muestra la lista de las comunidades obteniendo una descripción y un identificador.
2		Termina Sección 1

Sección 2: "Modificar Comunidad"

Flujo básico Modificar comunidad

	Actor	Sistema
1.	Decide modificar los campos correspondientes a una comunidad ✓ Nombre de la comunidad ✓ Descripción de la comunidad	Valida los datos entrados por el usuario.
2		Registra los datos modificados de la comunidad. Muestra el mensaje "La comunidad se ha modificado correctamente".
3		Redirecciona a la vista "Listar comunidades".
4		Termina sección 2.

Flujos alternos

Nº 1 *Los datos modificados son incorrectos*

	Actor	Sistema
1		Muestra un mensaje "Error: Algunos datos son incorrectos."
2		Vuelve al paso 1 del flujo básico de la sección modificar comunidad.
3		Termina sección 3.

Flujos alternos

Nº 2 El usuario pulsa el botón cancelar

	Actor	Sistema
1	Pulsa el botón cancelar	Redirecciona a la vista "Listar comunidad"
2		Termina sección 2.

Relaciones	CU extendidos	Calcular Métricas, Paso 2 del flujo básico

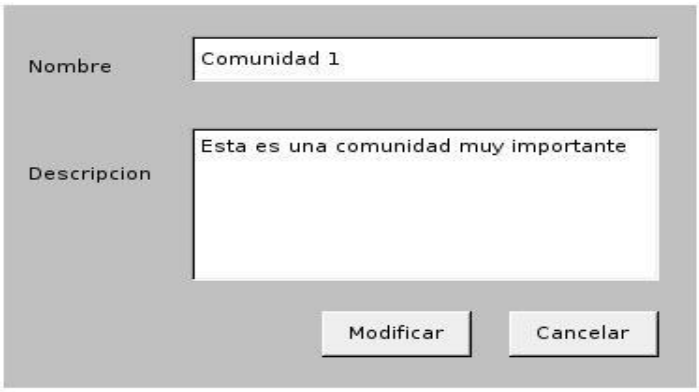
Prototipo	

Tabla 12 Descripción de caso de usos Gestionar Comunidad.

CU # 8. Calcular Métricas		
Objetivo	Hacer el cálculo de las métricas para gestionar la información.	
Actores	Usuario	
Resumen	El caso de uso inicia cuando el usuario dentro del flujo de trabajo de los casos de uso, gestionar red y modificar comunidad decide hacer el cálculo de las métricas.	
Complejidad	Media.	
Prioridad	Alta crítico.	
Precondiciones	RF-20, RF-21, RF-22, RF-23, RF-24, RF-25, RF-26	
Postcondiciones	El cálculo de las métricas son realizadas satisfactoriamente.	
Flujo de eventos		
Flujo básico <Calcular métricas>		
	Actor	**Sistema**
1	Dentro de los casos de uso Gestionar Red y modificar red, accede a la opción métricas.	Muestra la vista relacionada con el cálculo de las métricas.
2	Obtiene la información.	Se termina el caso de uso.

Tabla 13 Descripción de caso de usos Calcular Métricas.

Anexo # 2 Arquitectura Modelo-Vista-Controlador de Django.

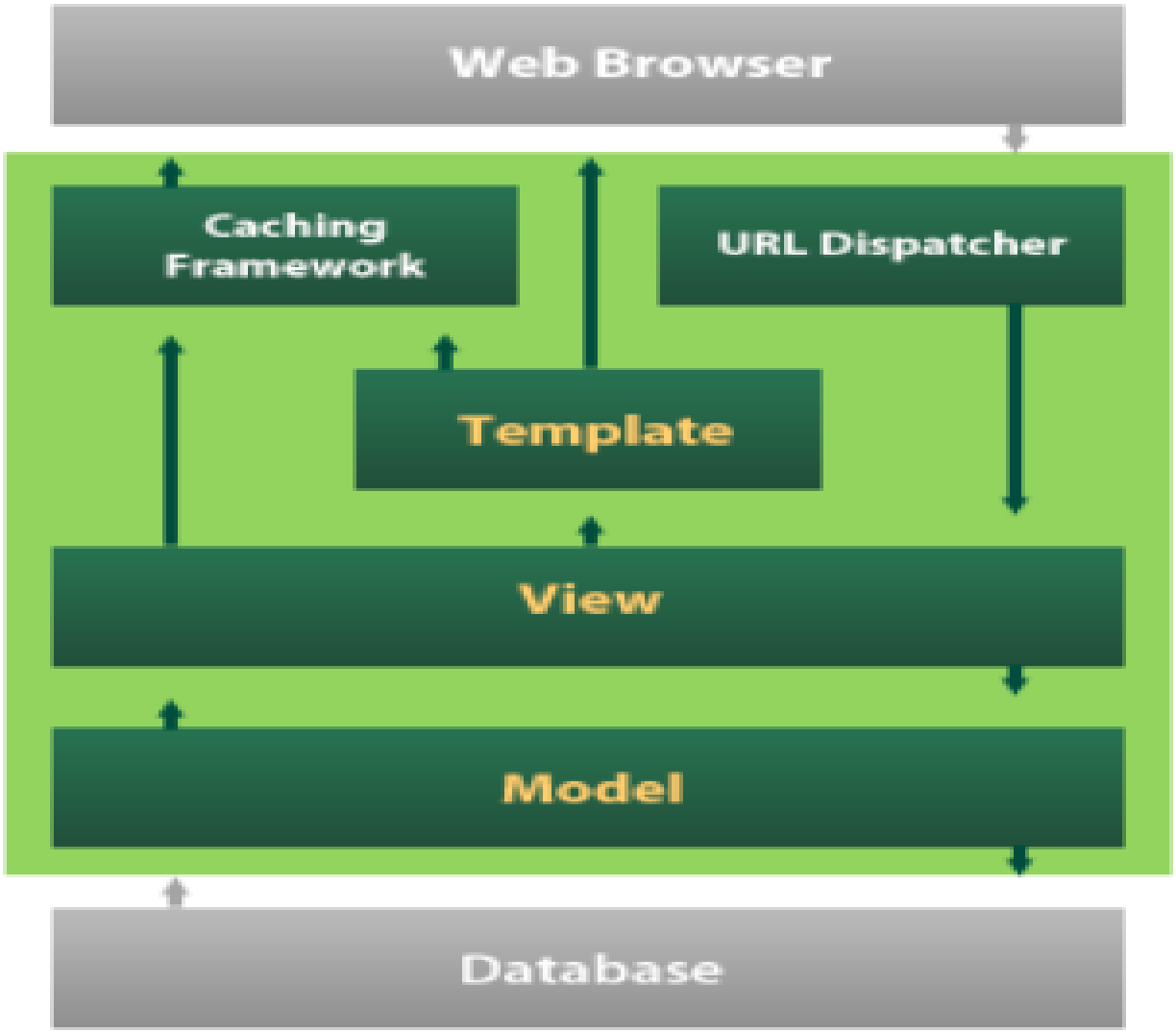


Figura 18 Arquitectura de Django.

Anexo # 3. Diagramas de Clases del diseño.

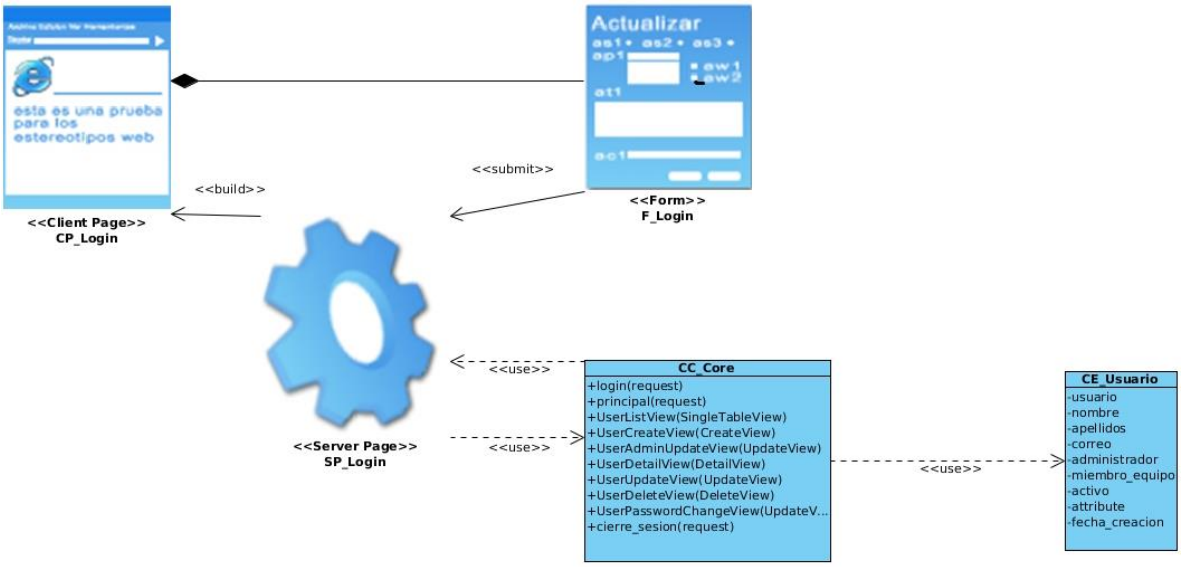


Figura 19 Diagrama de clases del diseño Caso de uso Autenticar Usuario.

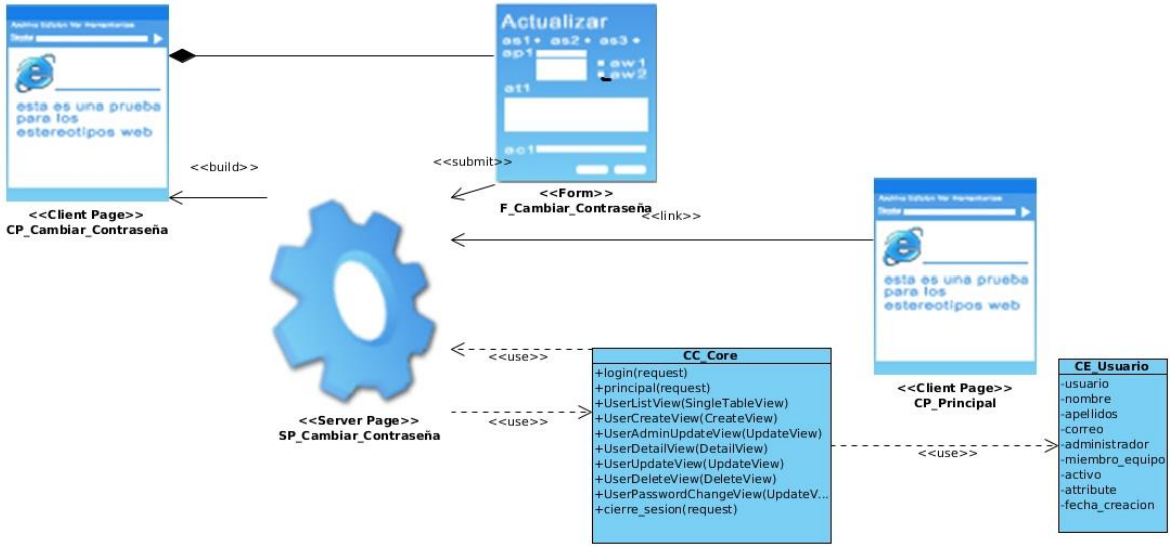


Figura 20 Diagrama de clases del diseño Caso de uso Cambiar Contraseña.

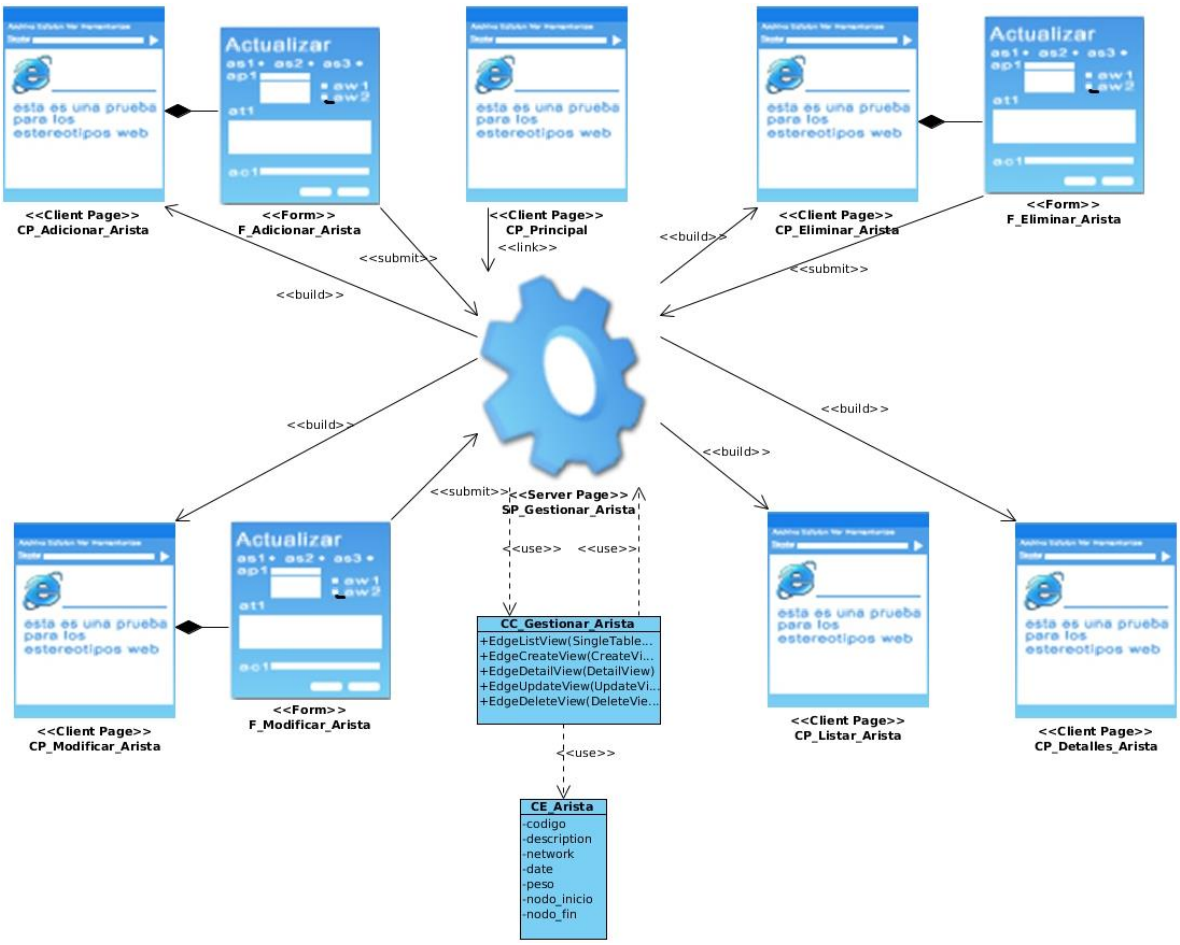


Figura 21 Diagrama de clases del diseño Caso de uso Gestionar Arista.

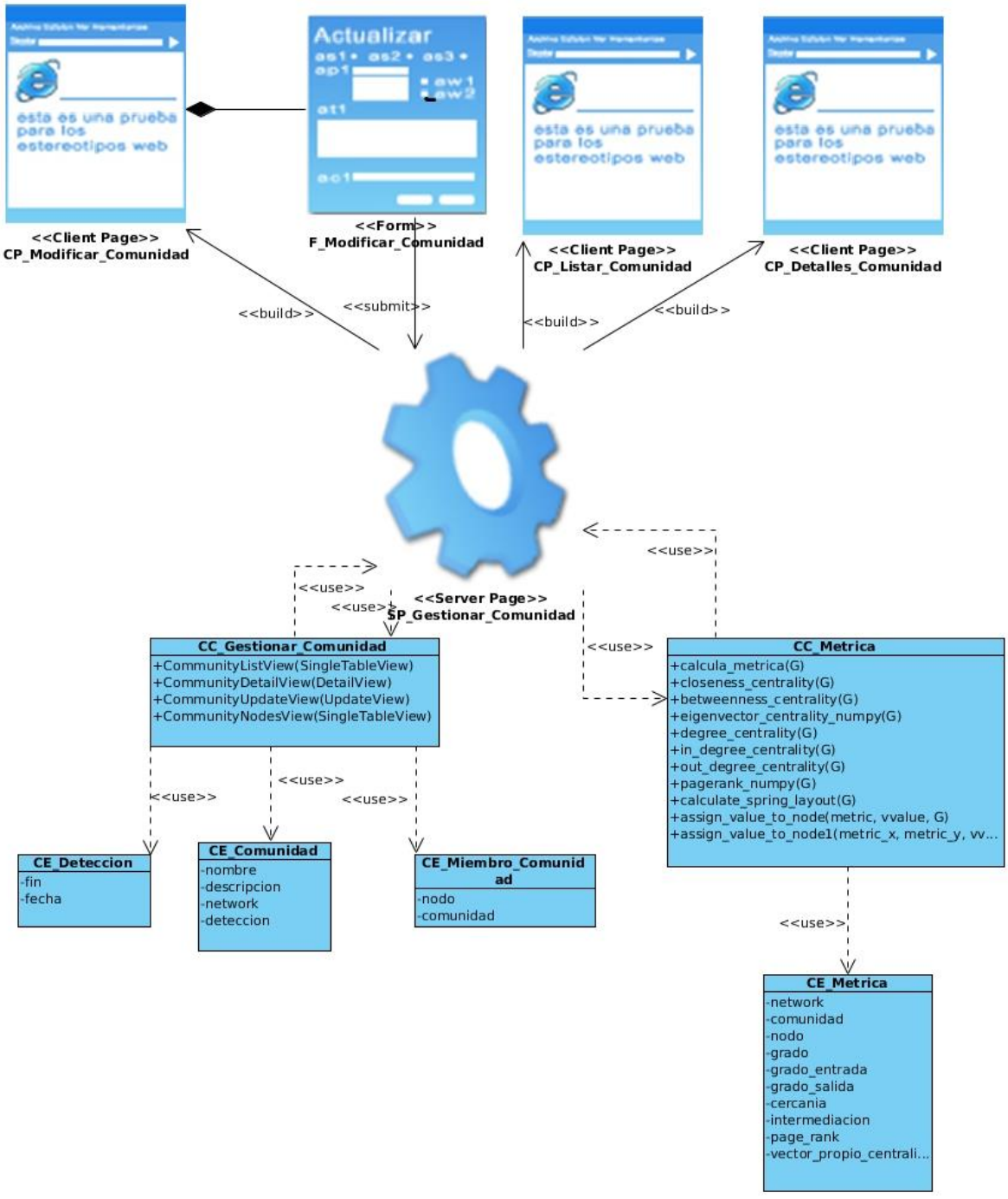


Figura 22 Diagrama de clases del diseño Caso de uso Gestionar Comunidad.

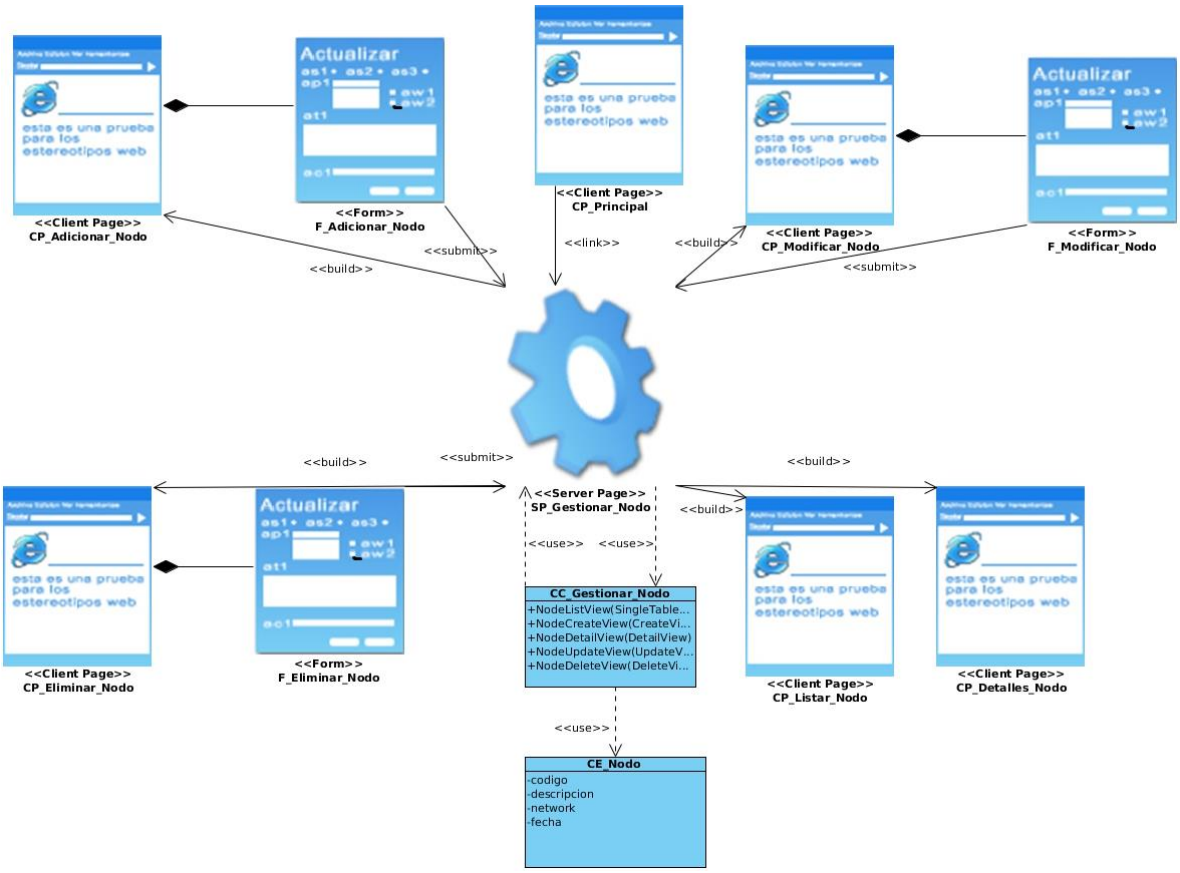


Figura 23 Diagrama de clases del diseño Caso de uso Gestionar Nodo.

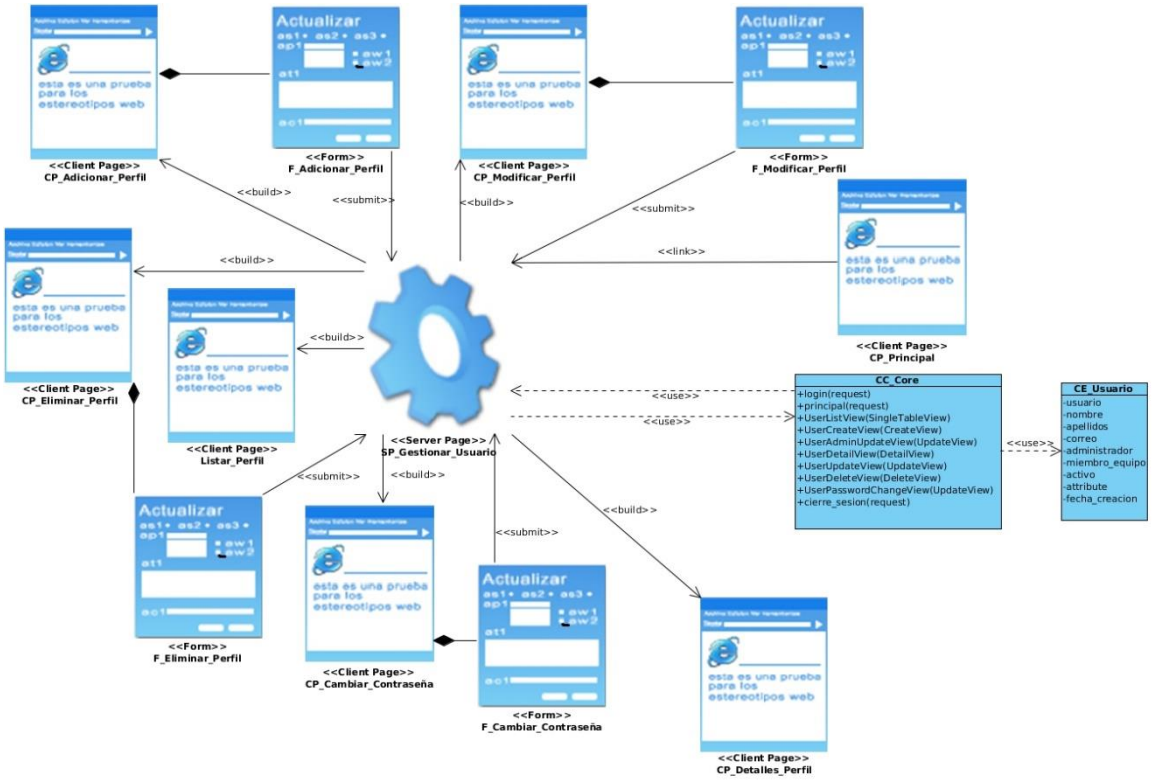


Figura 24 Diagrama de clases del diseño Caso de uso Gestionar Perfil de Usuario.

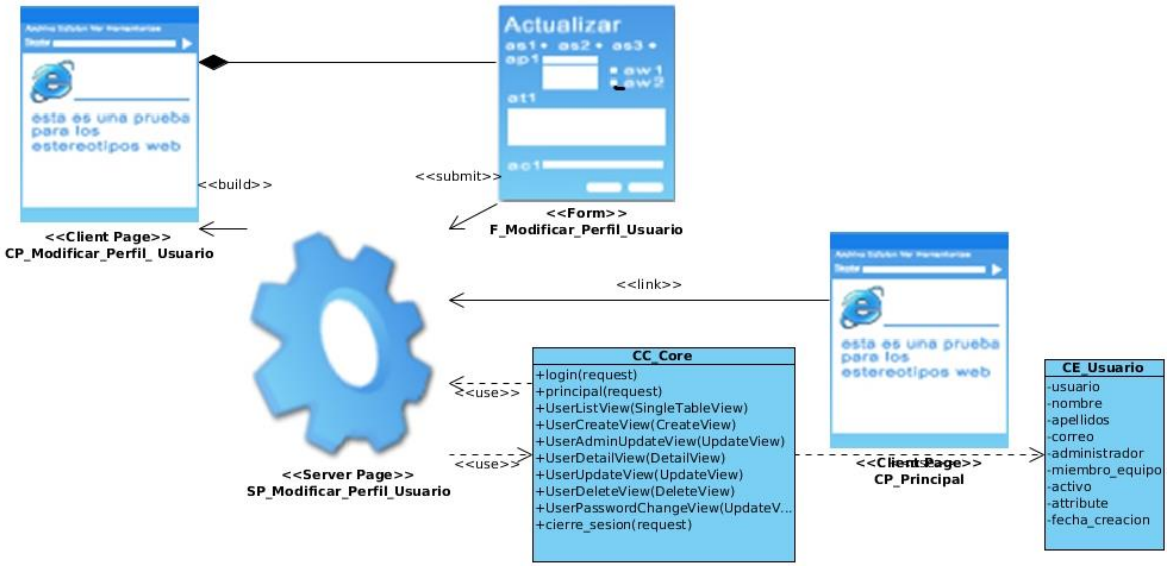


Figura 25 Diagrama de clases del diseño Caso de uso Modificar Perfil de Usuario.

Anexo # 4. Diagramas de secuencia.

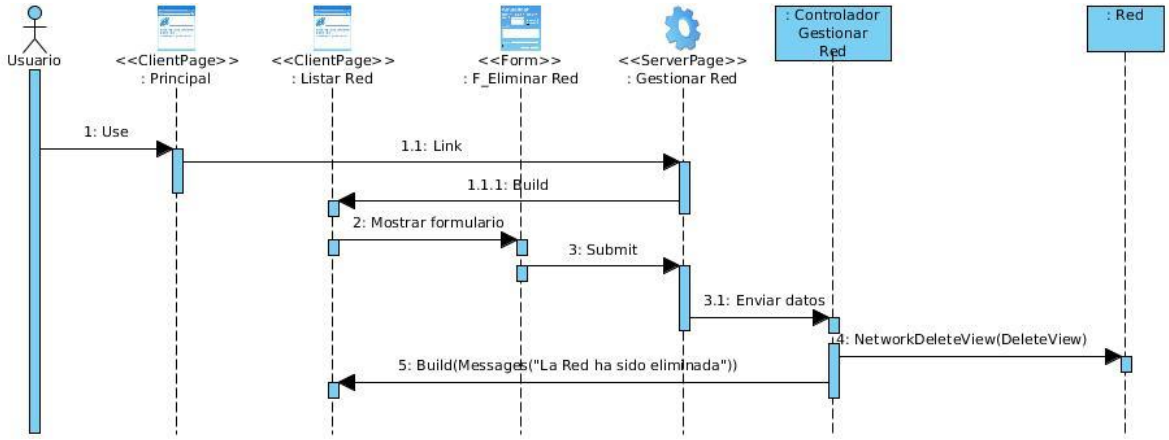


Figura 26 Diagrama de secuencia Eliminar Red.

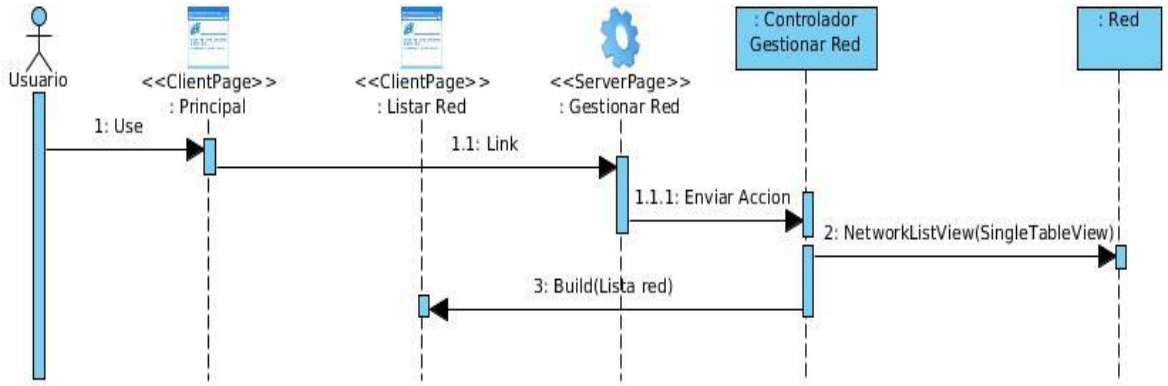


Figura 27 Diagrama de secuencia Listar Red.

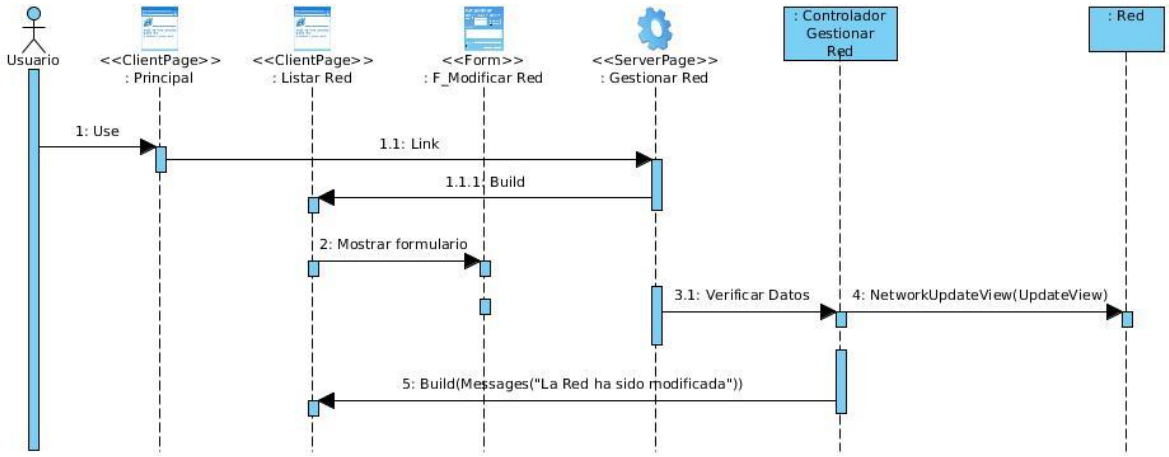


Figura 28 Diagrama de secuencia Modificar Red.

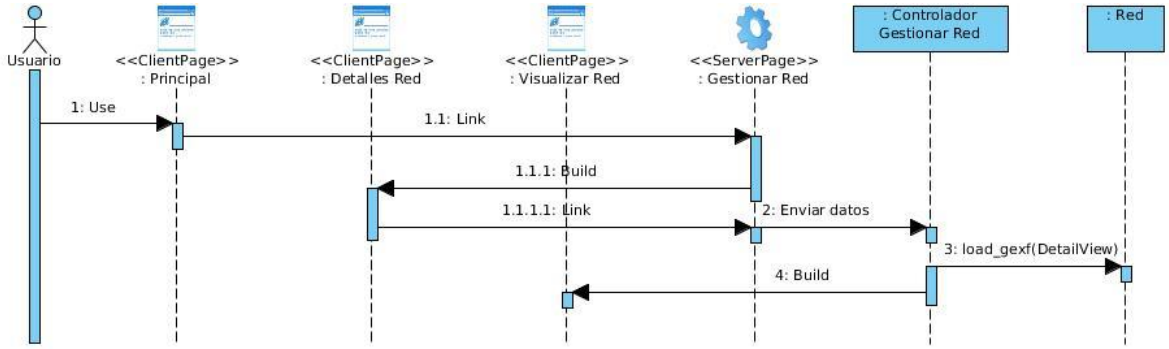


Figura 29 Diagrama de secuencia Visualizar Red.

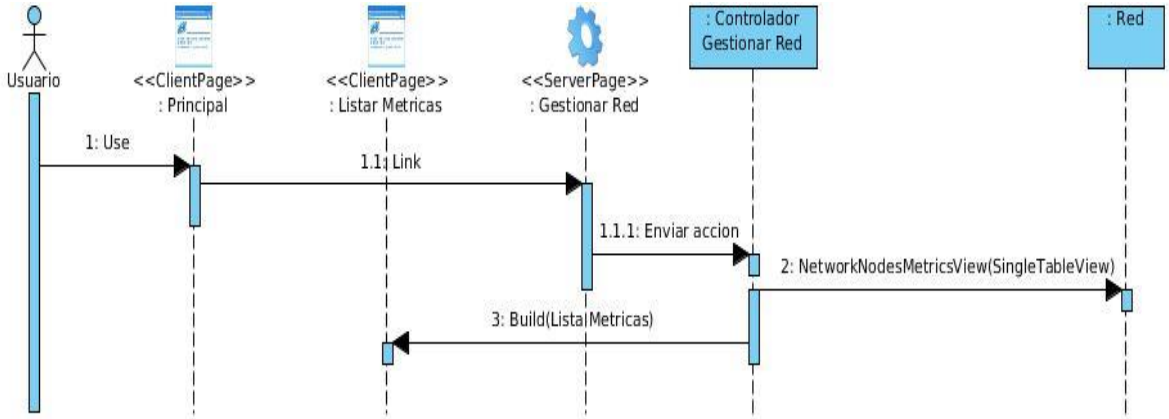


Figura 30 Diagrama de secuencia Listar Métricas de Red.

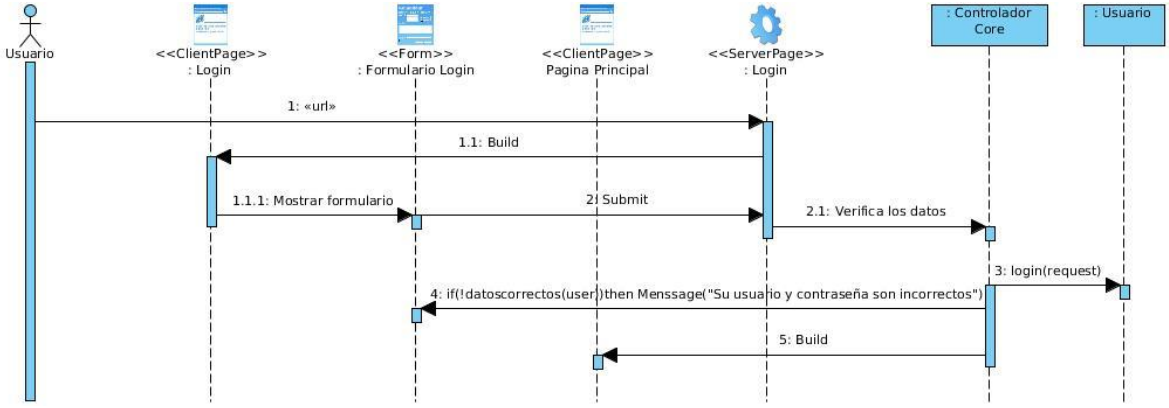


Figura 31 Diagrama de secuencia Autenticar Usuario

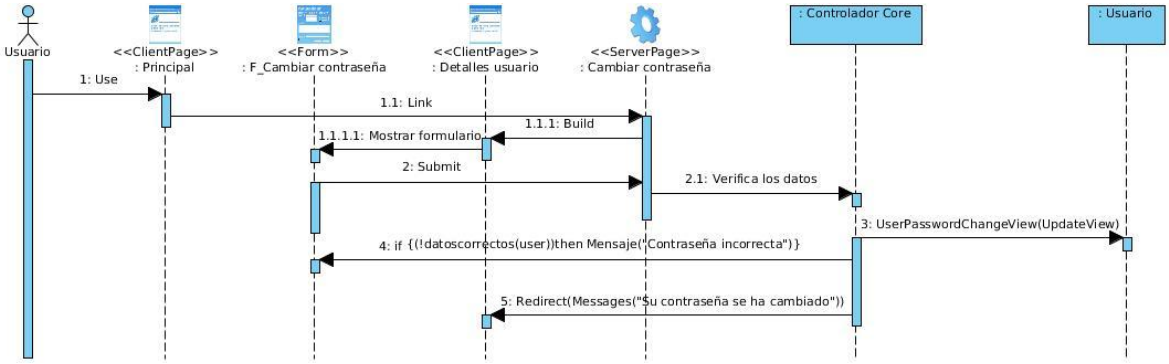


Figura 32 Diagrama de secuencia Cambiar Contraseña.

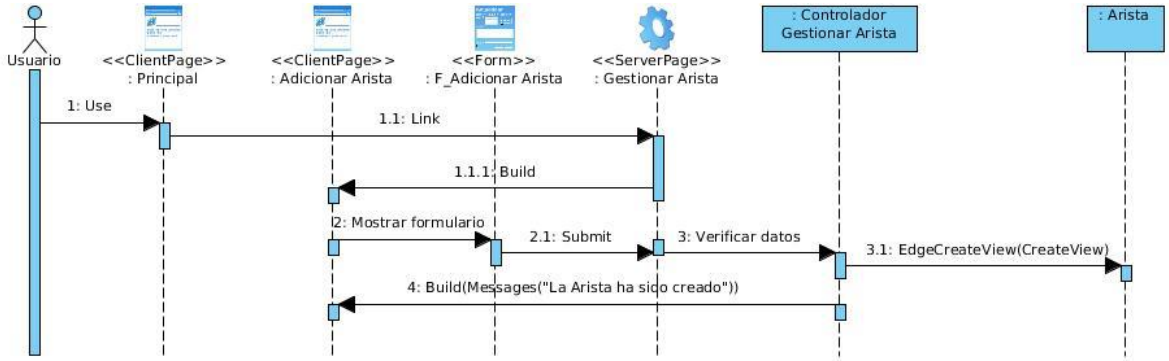


Figura 33 Diagrama de secuencia Adicionar Arista

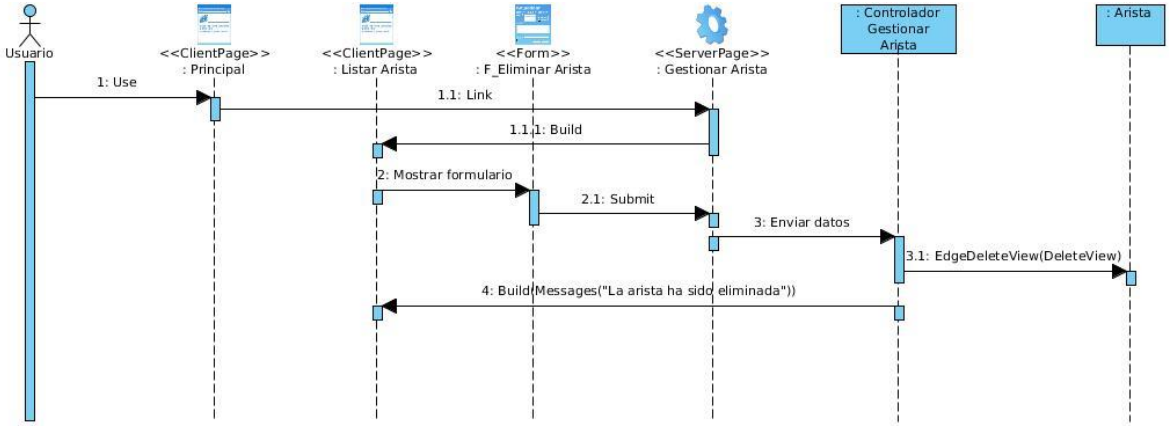


Figura 34 Diagrama de secuencia Eliminar Arista

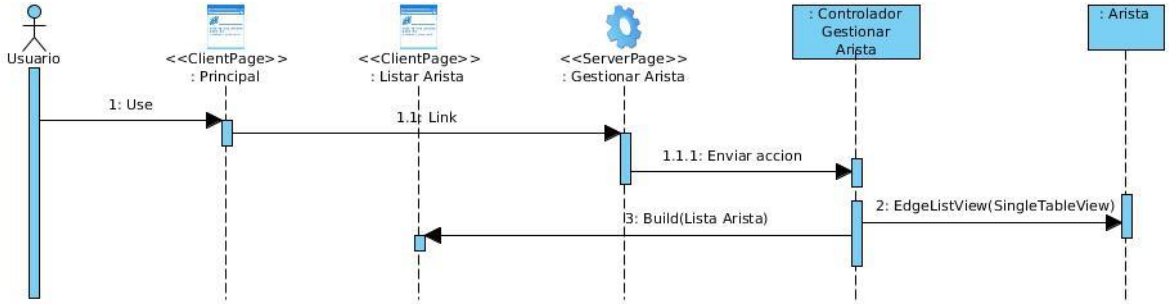


Figura 35 Diagrama de secuencia Listar Arista.

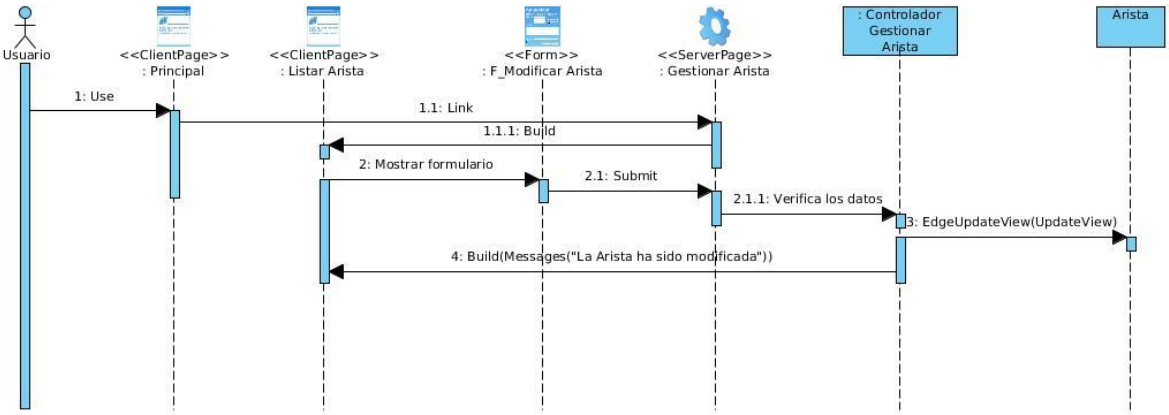


Figura 36 Diagrama de secuencia Modificar Arista.

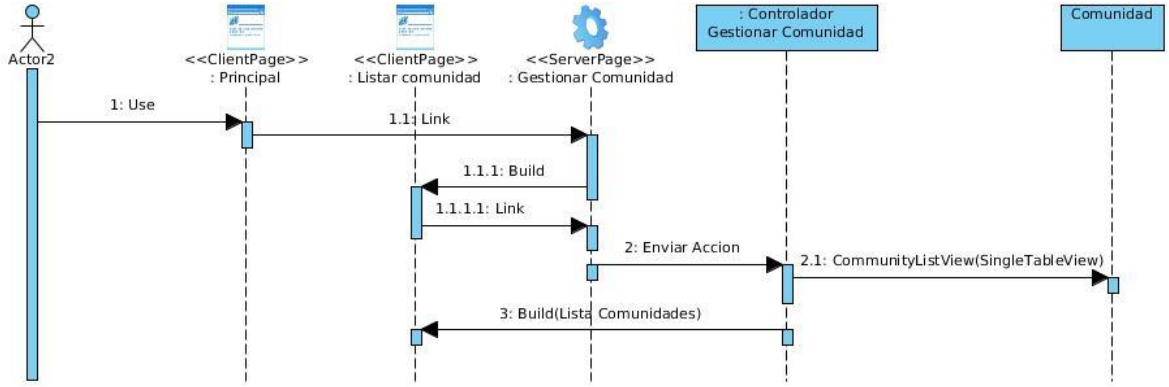


Figura 37 Diagrama de secuencia Listar Comunidad

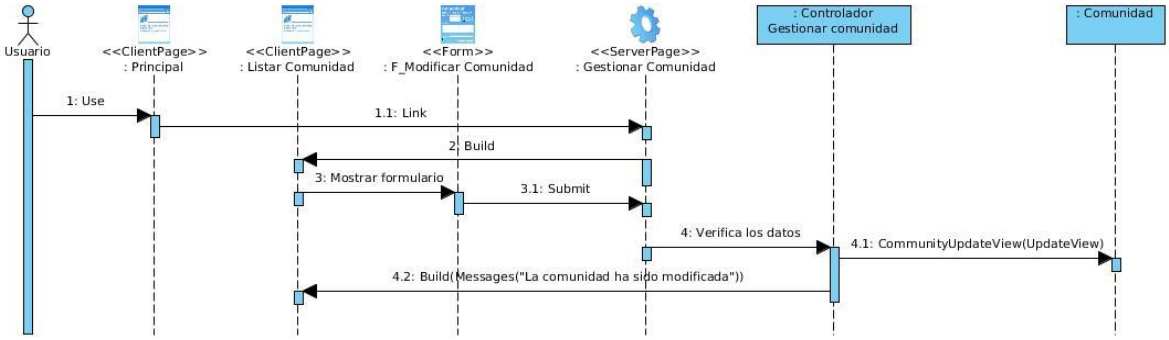


Figura 38 Diagrama de secuencia Modificar Comunidad.

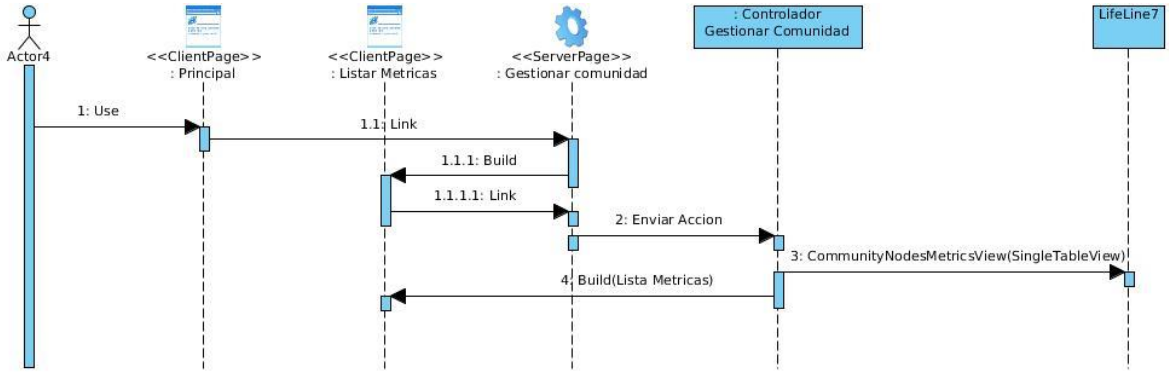


Figura 39 Diagrama de secuencia Listar Métricas de la Comunidad

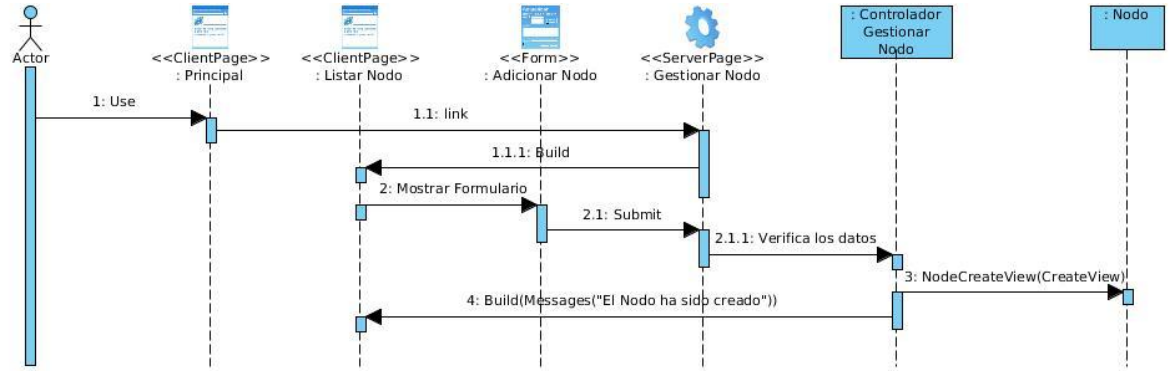


Figura 40 Diagrama de secuencia Adicionar Nodo.

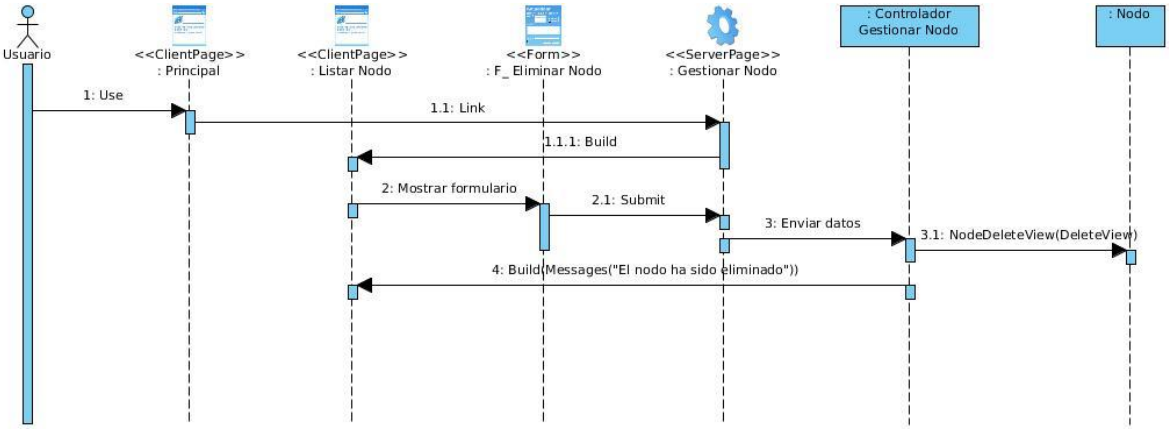


Figura 41 Diagrama de secuencia Eliminar Nodo.

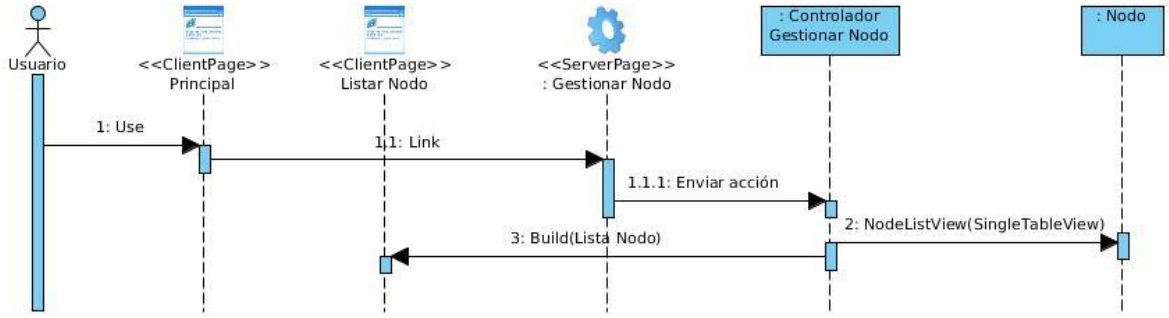


Figura 42 Diagrama de secuencia Listar Nodo.

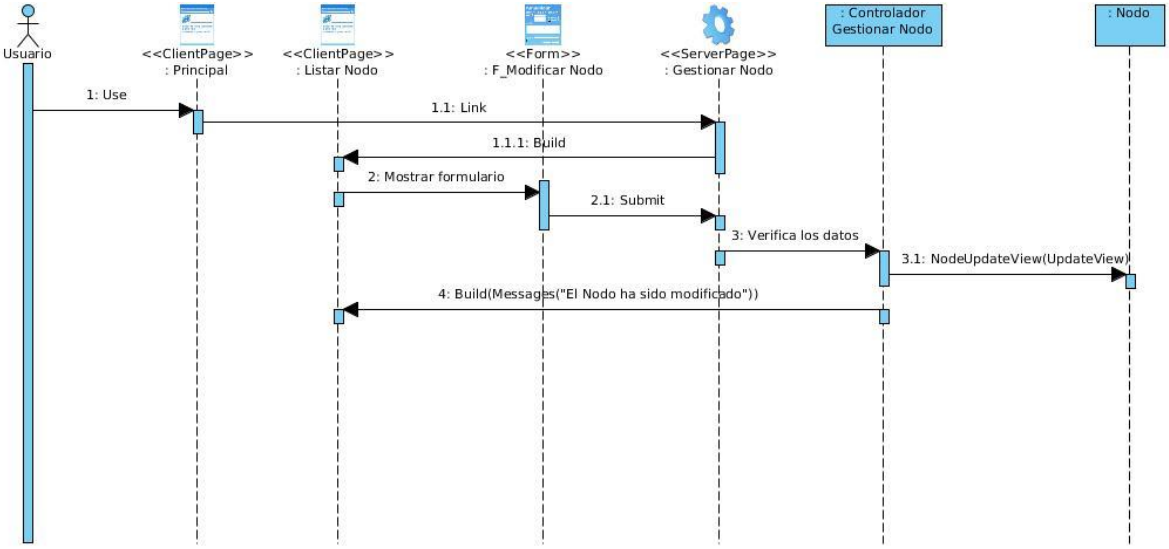


Figura 43 Diagrama de secuencia Modificar Nodo.

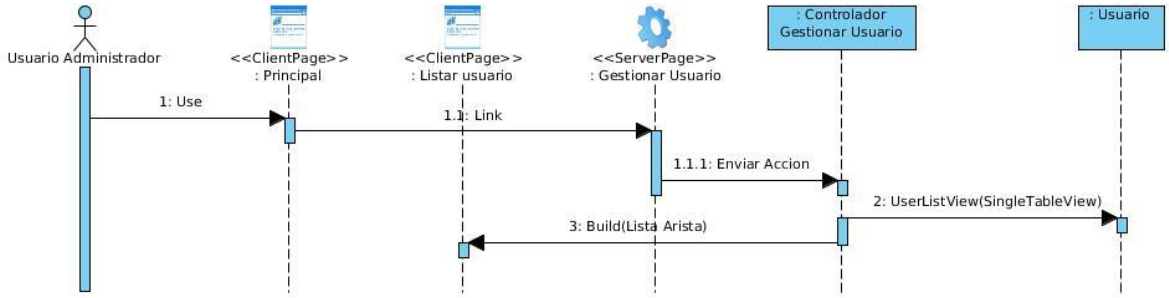


Figura 44 Diagrama de secuencia Listar Usuarios.

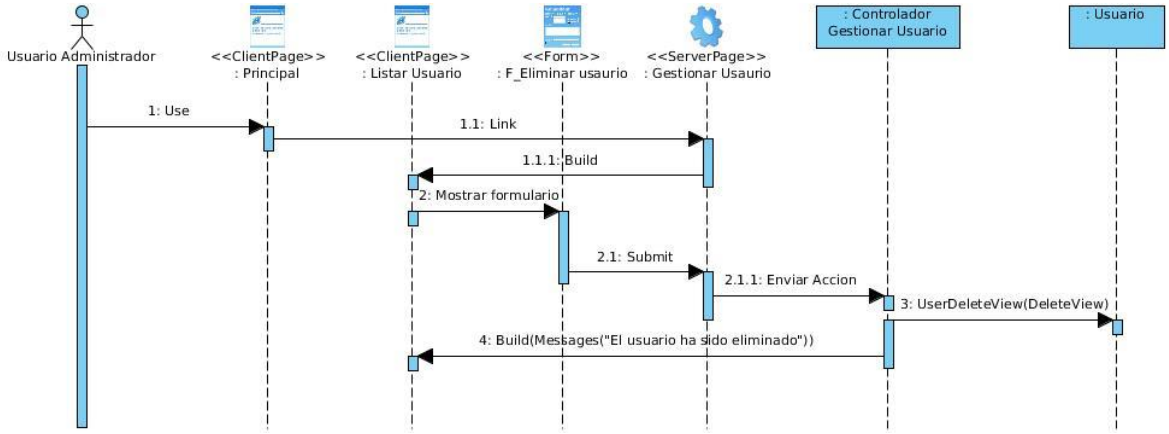


Figura 45 Diagrama de secuencia Eliminar Usuario.

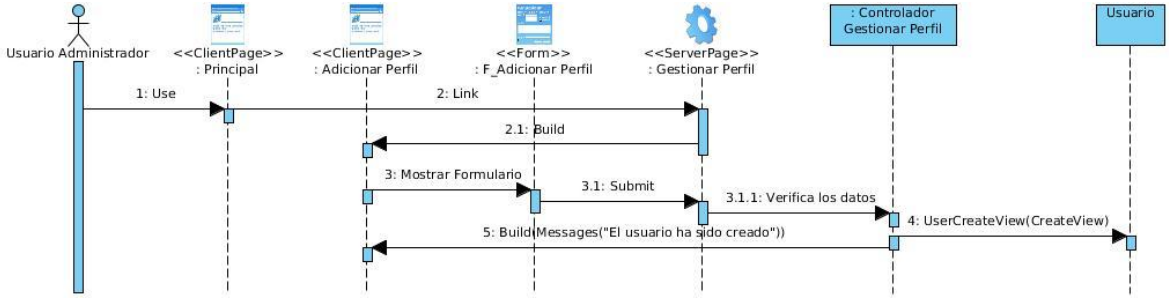


Figura 46 Diagrama de secuencia Adicionar Usuario.

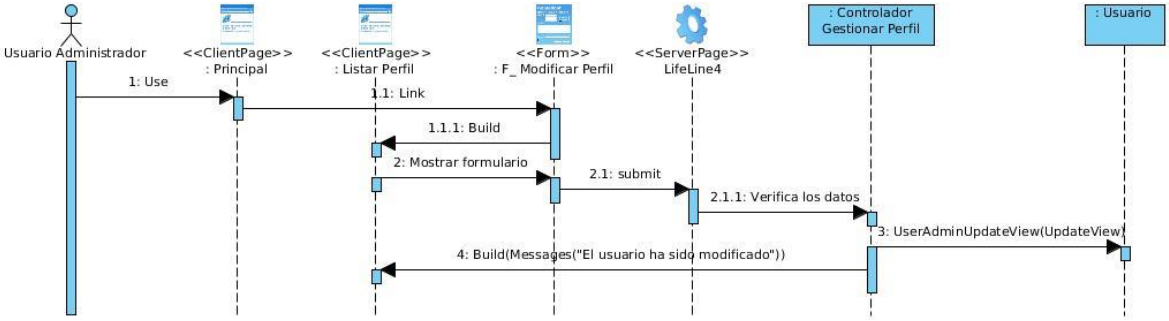


Figura 47 Diagrama de secuencia Modificar Usuario.

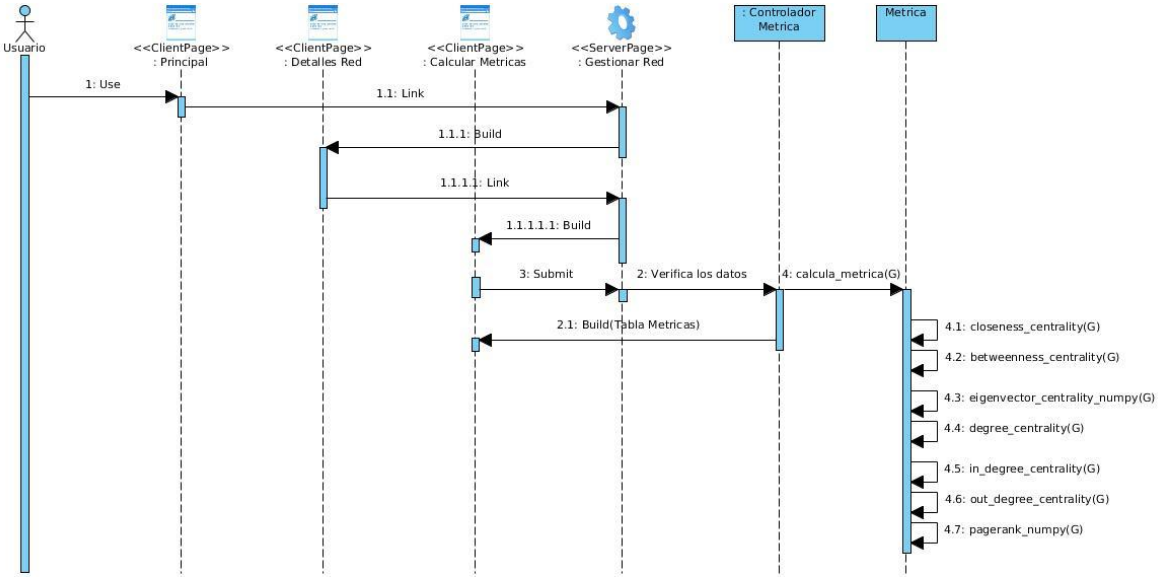


Figura 48 Diagrama de secuencia Calcular Métricas.

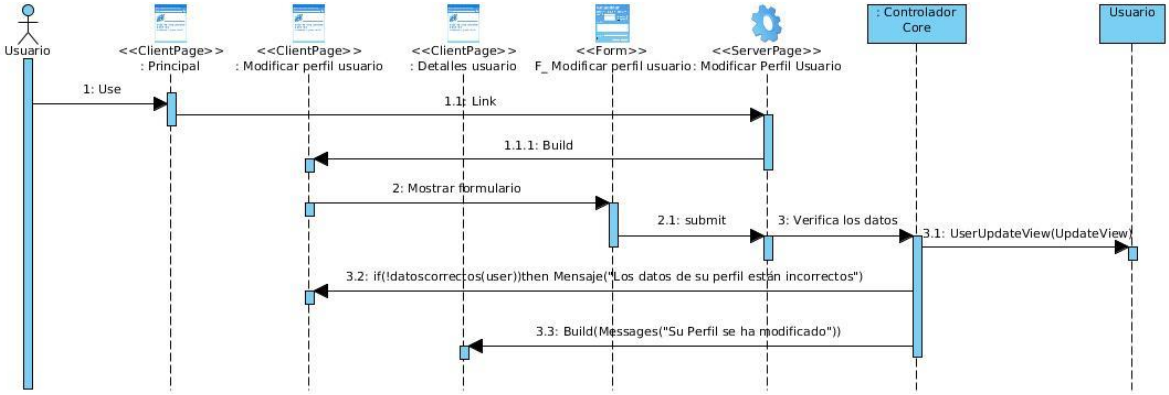


Figura 49 Diagrama de secuencia Modificar Perfil de Usuario.

SC Listar Red

Escenario	Descripción	Nombre	Descripción	Red	Respuesta del sistema	Flujo central
EC 1.13 El usuario solicita listar las redes	*El usuario solicita listar las redes*	*NA*	*NA*	*NA*	*Se muestra el listado de las redes*	*1 El usuario autenticado solicita gestionar redes.*
		NA	*NA*	*NA*		

SC Eliminar Red

Escenario	Descripción	Nombre	Descripción	Red	Respuesta del sistema	Flujo central
EC 1.14 El usuario solicita eliminar una red	*El usuario solicita eliminar una red*	*NA*	*NA*	*NA*	*Se elimina la red se muestra el listado de las redes y un mensaje "La red ha sido eliminada satisfactoriamente"*	*1 El usuario autenticado solicita gestionar red.* *2 Luego hace clic en la red que desee eliminar.* *3 Por último da clic en el botón Eliminar.*
		NA	*NA*	*NA*		

Tabla 14 Caso de Prueba Gestionar Red (Continuación).

Anexo # 5. Casos de prueba.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	*Usuario*	*Entrada de texto*	*No*	*Corresponde al nombre de usuario. Puede ser cualquier cadena de caracteres alfanuméricos, que no comiencen con un número, y un límite de 20 caracteres.*
2	*Contraseña*	*Entrada de contraseña*	*No*	*Corresponde a la contraseña del usuario. Puede ser cualquier cadena de caracteres*

Tabla 15 Caso de Prueba Autenticar Usuario.

Escenario	Descripción	Usuario	Contraseña	Respuesta del sistema	Flujo central
EC 1.1 El usuario cambia su contraseña	*En el formulario de cambio de contraseña el usuario introduce un par su nueva contraseña y repetir la entrada de la nueva contraseña, luego hace clic en el botón "Entrar".*	*V* *Campo de contraseña válida*	*V* *Campo de repetir contraseña válida*	*La contraseña del usuario es cambiada satisfactoriamente.*	*1 El usuario accede al formulario de cambiar contraseña. 2. El sistema muestra al usuario la vista del formulario del cambio de contraseña. 3 El usuario se dispone a llenar los campos contraseña y repetir contraseña.*
EC 1.2 El usuario falla al cambiar la contraseña	*En el formulario de cambio de contraseña, el usuario introduce un par su nueva contraseña y repetir la contraseña incorrecta, luego hace clic en el botón "Entrar".*	*V* *Campo de contraseña no válida*	*I* *Campo de repetir contraseña no válida*	*Muestra un mensaje de error "Error al cambiar la contraseña" o "La contraseña debe tener más de 6 caracteres".*	
EC 1.3 El usuario falla al cambiar la contraseña	*En el formulario de cambio de contraseña, el usuario introduce un par su nueva contraseña correcta y repetir*	*I* *Campo de contraseña válida*	*V* *Campo de repetir contraseña no válida*	*Muestra un mensaje de error "Error al cambiar la contraseña" o "La contraseña no coincide".*	

	la contraseña incorrecta, luego hace clic en el botón "Entrar".				

Tabla 16 Caso de Prueba Cambiar Contraseña.

SC Crear Nodo

Escenario	Descripción	Nombre	Descripción	Red	Fecha	Respuesta del sistema	Flujo central
EC 1.1 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con un nombre válido*	*V*	*NA*	*NA*	*NA*	*El sistema crea el nodo y muestra un mensaje "El nodo se ha creado satisfactoriamente"*	*1 El usuario autenticado solicita gestionar nodo.* *2 Solicita crear un nodo haciendo clic en el botón Nuevo.* *3 Introduce los datos y da clic en el botón Guardar.*
		Nombre de nodo válido	*NA*	*NA*	*NA*		
EC 1.2 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con un nombre existente*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido creado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor ya se ha utilizado."*	
		Nombre de nodo existente	*NA*	*NA*	*NA*		
EC 1.3 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con un nombre no válido*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido creado" e informa del campo incorrecto: en caso de introducir una cadena no válida; "Solo letras, números y*	

		Nombre de nodo no válido	*NA*	*NA*	*NA*	*espacios" en caso de ser la cadena menor de 2 caracteres; "Este valor es demasiado corto. Debería tener 2 caracteres o más." y en caso de tener más de 30 caracteres; "Este valor es demasiado largo. Debería tener 30 caracteres o menos."*	
EC 1.4 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con el campo nombre vacío*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "Campo no válido" o "Invalid field"*	
		Campo vacío	*NA*	*NA*	*NA*		
EC 1.5 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con el campo red incorrecto*	*NA*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido creado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*Campo Red no válido*	*NA*		
EC 1.6 El usuario solicita crear un nodo	*El usuario solicita crear un nodo con el campo fecha incorrecto*	*NA*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido creado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*NA*	*Campo fecha no válido*		

SC Modificar Nodo

Escenario	Descripción	Nombre	Descripción	Red	Fecha	Respuesta del sistema	Flujo central
EC 1.7 El usuario solicita modificar un nodo	*El usuario solicita modificar el nodo seleccionado con nombre no válido*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El elemento no ha sido modificado" e informa del campo incorrecto: en caso de introducir una cadena no válida; "Solo letras, números y espacios" en caso de ser la cadena menor de 2 caracteres; "Este valor es demasiado corto. Debería tener 2 caracteres o más." y en caso de tener más de 30 caracteres; "Este valor es demasiado largo. Debería tener 30 caracteres o menos."*	*1 El usuario autenticado solicita gestionar nodo. 2 Solicita modificar un nodo haciendo clic en el nodo que desee modificar. 3 Modifica los datos y da clic en el botón Guardar.*
		Nombre de nodo no válido	*NA*	*NA*	*NA*		
EC 1.8 El usuario solicita modificar un nodo	*El usuario solicita guardar el nodo seleccionado con un nombre válido*	*V*	*NA*	*NA*	*NA*	*Muestra un mensaje "El nodo seleccionado ha sido modificado satisfactoriamente"*	
		Nombre de nodo válido	*NA*	*NA*	*NA*		
EC 1.9 El usuario solicita modificar un nodo	*El usuario solicita guardar el nodo seleccionado con un nombre existente diferente del propio*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo seleccionado no ha sido modificado" e informa del campo*	
		Nombre del nodo existente	*NA*	*NA*	*NA*		

						incorrecto con un mensaje sobre el mismo "Este valor ya se ha utilizado."	
EC 1.10 El usuario solicita modificar un nodo	*El usuario solicita modificar el nodo seleccionado con el campo nombre vacío*	*I*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "Campo no válido" o "Invalid field"*	
		Campo vacío	*NA*	*NA*	*NA*		
EC 1.11 El usuario solicita modificar un nodo	*El usuario solicita modificar un nodo con el campo red incorrecto*	*NA*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido modificado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*Campo Red no válido*	*NA*		
EC 1.12 El usuario solicita modificar un nodo	*El usuario solicita modificar un nodo con el campo fecha incorrecto*	*NA*	*NA*	*NA*	*NA*	*Muestra un mensaje de error "El nodo no ha sido modificado" e informa del campo incorrecto con un mensaje sobre el mismo "Este valor es incorrecto."*	
		NA	*NA*	*NA*	*Campo fecha no válido*		

SC Listar Nodo

Escenario	**Descripción**	**Nombre**	**Descripción**	**Red**	**Fecha**	**Respuesta del sistema**	**Flujo central**
EC 1.13 El usuario solicita listar	*El usuario solicita listar los nodos*	*NA*	*NA*	*NA*	*NA*	*Se muestra el listado de los nodos*	*1 El usuario autenticado solicita*
		NA	*NA*	*NA*	*NA*		

los nodos							*gestionar grupos.*

SC Eliminar Nodo

Escenario	Descripción	Nombre	Descripción	Red	Fecha	Respuesta del sistema	Flujo central
EC 1.14 El usuario solicita eliminar un nodo	*El usuario solicita eliminar un nodo.*	*NA*	*NA*	*NA*	*NA*	*Se elimina el nodo se muestra el listado de los nodos y un mensaje "El nodo ha sido eliminado satisfactoriam ente"*	*1 El usuario autenticado solicita gestionar nodo.* *2 Luego hace clic en el nodo que desee eliminar.* *3 Por último da clic en el botón Eliminar.*
		NA	*NA*	*NA*	*NA*		

Anexo # 6 Diagramas de Componentes.

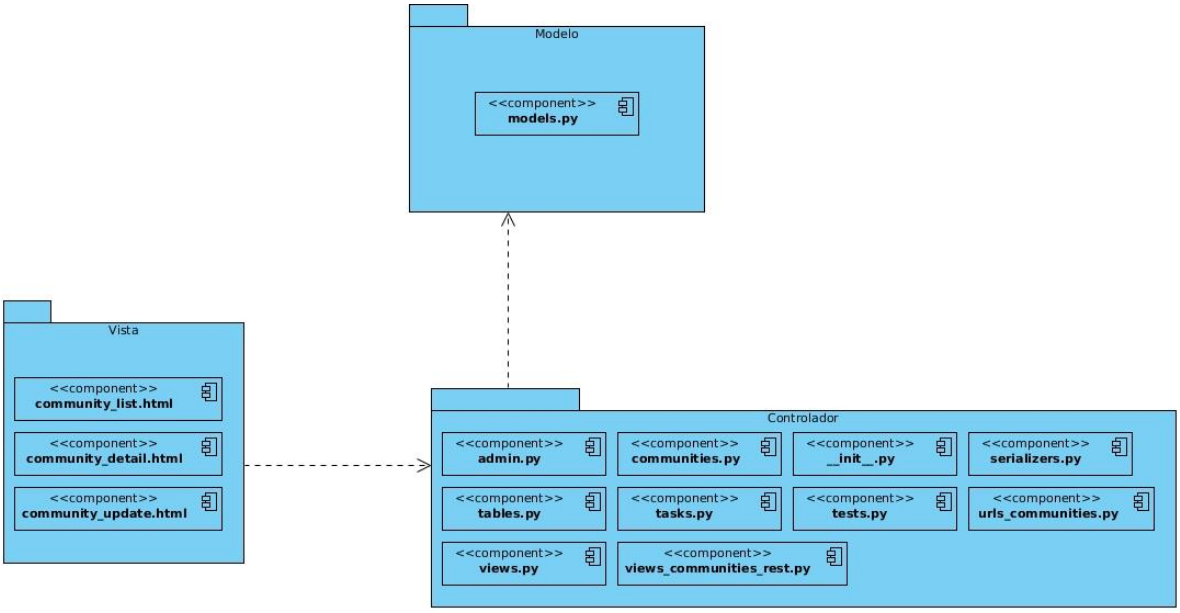


Figura 50 Diagrama de componentes: Aplicación Comunidad.

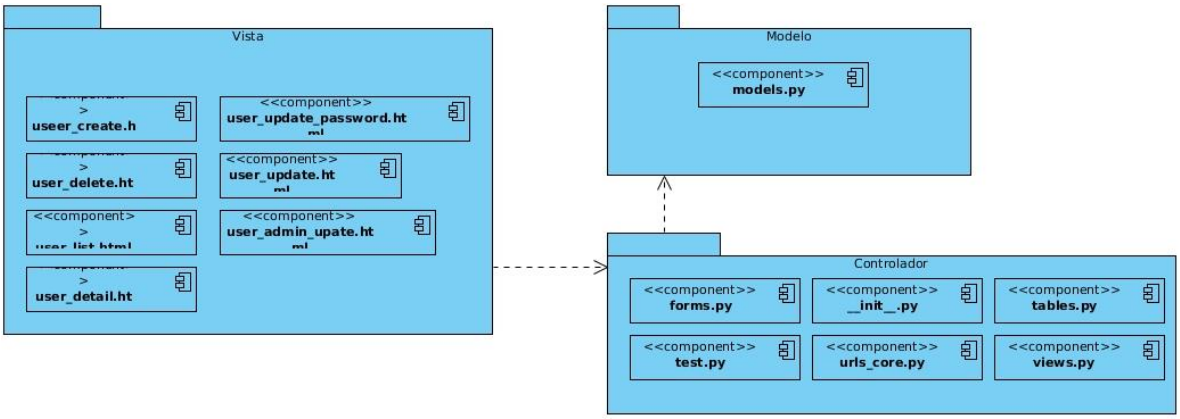


Figura 51 Diagrama de componentes: Aplicación Núcleo.

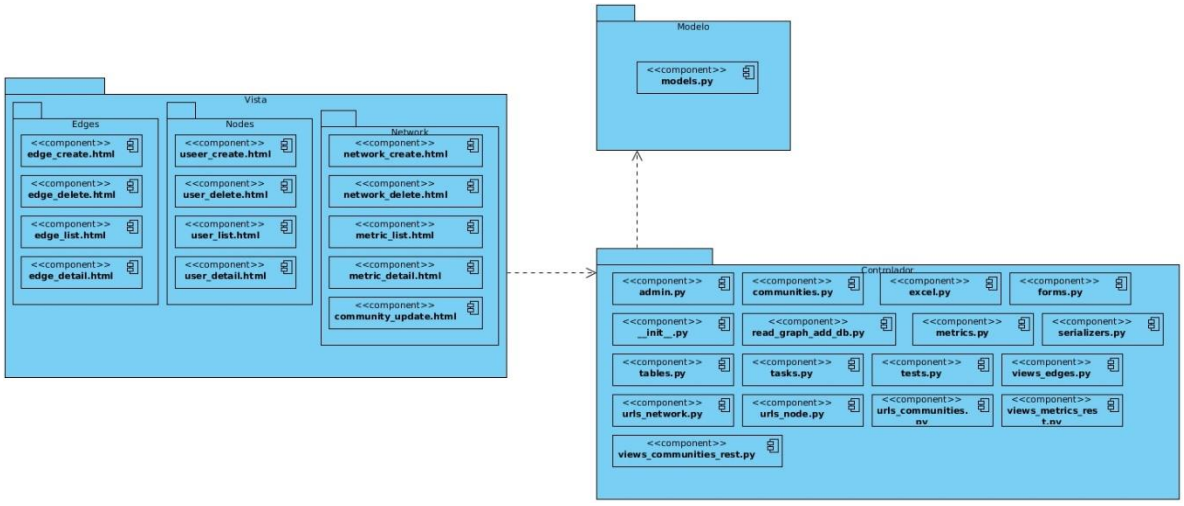


Figura 52 Diagrama de componentes: Aplicación SNA.

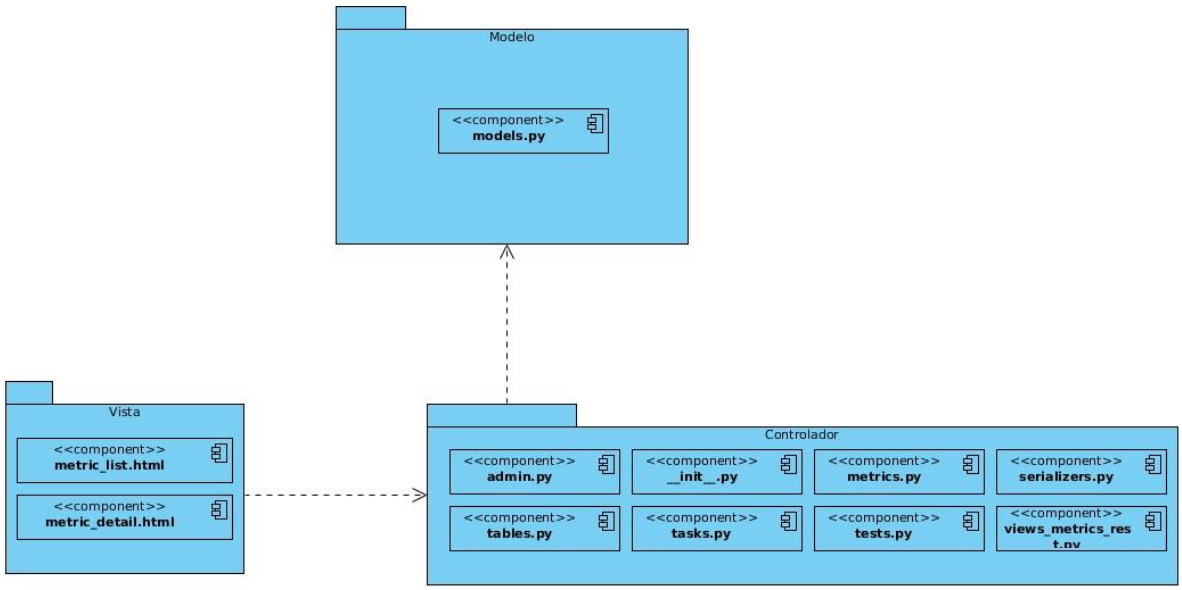


Figura 53 Diagrama de componentes: Aplicación Métrica.