UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 2

Módulo para medir la calidad de las publicaciones científicas en el Sistema para repositorios digitales REPXOS 3.0

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

Autor: Daniel Cruz González

Tutor: Gleidis Yuriannis Rosabal Espinosa

Co-Tutor: Yorjander Tan Guevara

La Habana, junio del 2015

DECLARACIÓN DE AUTORÍA

Declaración de Autoría

Declaro ser el legítimo autor del trabajo titulado: *Módulo para medir la calidad de las publicaciones científicas en el Sistema para repositorios digitales REPXOS 3.0* y reconozco a la Universidad de las Ciencias Informáticas (UCI) los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____

Firma del Autor
Daniel Cruz González

_____

Firma del Tutor
Gleidis Yuriannis Rosabal Espinosa

_____

Firma del Co-Tutor
Yorjander Tan Guevara

DEDICATORIA

Le dedico esta tesis a mis padres por no haberme matado en estos años de la universidad, por apoyarme cada vez que tenía mundiales y confiar en mi cuando les decía ***aun queda tiempo*** *o* ***eso lo apruebo yo con dos días de estudio.*** *Por ser unos padres ejemplares y enseñarme muchoooo.*

A Dayana, mi gordi del alma, por ser mi mamá aquí en la escuela estos 5 años, por ser mi amiga en todo momento sin importarle las condiciones en que se encuentre, por cargar mis culpas y caprichos, por hacer suyos mis problemas. Mimi gracias por todo, te llevo en mi corazón para Pinar del Río, recuerda que eres la mejor hermanita del mundo, te quiero mucho y nunca te olvidaré.

AGRADECIMIENTO

A las personas que más quiero en el mundo por apoyarme siempre, por brindarme su amor y confianza y creer en mí a mi papá Daniel, mi mamita Xiomara mi abuela Rita, son lo más grande que tengo en la vida.

A mi familia completa a todos mis tíos de la habana, Dulce, Sonia Beni, Rafael. A los de Pinar Enrique, el pity y el negro.

A mis primos Rafelito y Dianita por quererme como hermano de ellos y `por compartir todas sus cosas conmigo, los quiero.

Anita, que agradecerte tengo un mundo, aprendí mucho a tu lado y gracias a ti fui muy feliz y encontré un motivo para seguir el sueño de graduarme. Siempre tendrás un lugar en mi corazón. TKM

Ernesto, socio gracias por enseñarme a hacer diagramas por compartir buenos ratos y sobre todo por cuidar mucho a Dayana

A Wendy y Sarahi por toda su ayuda desde el politécnico hasta hoy en día.

A mis tutores que siempre me apoyaron y me dieron aliento para cumplir este
Objetivo y en especial a gleidis por ser mi compañera de tesis. A Luis Carlos por enseñarme a trabajar con DSpace y a mi profesora de PP Daira por ayudarme mucho revisándome el documento.

Al piquete de Pinar, Chino, Migue, Pedro, Alain, Jochy, Leo, Roilo y Leonardo el habanero que era como de pinar por estar siempre jodiendo en las buenas y las malas, piquete aquí va mi tesis como se los prometí, más vale tarde que nunca.

A Zuniet y Lianne por ser una de las mejores amigas que he tenido en el corto tiempo que estuvimos juntos.

Al piquete de Matanzas Handy, Nino, Viper, Alejandro, Eyidey por compartir el ***lotar*** *para que no me vieran en las travesuras.*

A Malvín, Osvel y Ledesma por ser mis guardaespaldas gracias a ustedes aún estoy vivo, miren que me salve a cuentas de ustedes.

RESUMEN

Resumen

Las publicaciones científicas son utilizadas para almacenar los conocimientos adquiridos por la humanidad en cualquier campo del saber. En la Universidad de las Ciencias Informáticas (UCI) existe un repositorio institucional el cual es el encargado de conservar y difundir las publicaciones científicas de los investigadores mediante el sistema REPXOS 3.0 que utiliza la herramienta DSpace para la creación de repositorios digitales. En el repositorio institucional no existe una forma de medir la calidad de las publicaciones científicas que se almacenan en el mismo. Para solucionar este problema, se desarrolló un módulo que permite medir la producción científica de un investigador. Para ello se realizó un estudio y análisis de los sistemas nacionales e internacionales utilizados para medir la calidad de las publicaciones científicas. Se estudiaron y analizaron los indicadores bibliométricos propuestos por estos sistemas, seleccionando los indicadores SCImago Journal Ranking (SJR) y el impacto normalizado de fuente por publicación (SNIP). Se diseñó e implementó un conjunto de clases que dan cumplimiento a los requisitos funcionales del módulo a desarrollar. Se utilizaron las herramientas, tecnologías y lenguajes especificados por el proyecto. Con la investigación realizada se logró medir la calidad de las producciones científicas de un investigador permitiendo que se puedan hacer comparaciones entre los mismos, además de incentivar la publicación de artículos científicos en revistas referenciadas internacionalmente.

Palabras claves:

Bibliometría, Cienciometría, Indicadores bibliométricos, Ranking, Repositorio Institucional.

Índice

ÍNDICE

Índice de Figuras

Introducción

Las publicaciones científicas son depositarias de los conocimientos documentales que la humanidad acumula en cualquier campo del saber y constituyen la vía fundamental para transmitir dichos conocimientos. La medición de la calidad de las producciones científicas no es posible conocerlas de forma absoluta, pero existen indicadores cuantitativos que permiten valorar de una forma relativa su impacto en la comunidad científica y son de utilidad para el personal docente e investigador. Un alto grado en la calidad de las producciones científicas contribuye a una mejora de la visibilidad e impacto científico de las instituciones. La publicación de *rankings* científicos proporciona una exitosa vía para iniciar y consolidar los procesos de cambio en el modo de publicación del mundo académico, aumentando el compromiso de los profesores e investigadores.

En esta dura batalla los países más pobres del mundo y en especial los latinoamericanos, dentro de ellos por supuesto Cuba, llevan una gran desventaja. Tratar de escalar posiciones en los sistemas de rankings internacionales se ha hecho una prioridad, producto a que la ciencia que se hace en esta parte del mundo es muchas veces ciencia perdida. Las leyes regulatorias internacionales hacen que sea más difícil lograr una ubicación de privilegio ante la gran avalancha de información que llega del llamado primer mundo.

Hay que tener presente que la visibilidad mundial de revistas científicas es determinada cualitativa y cuantitativamente por factores como: (1)

- La presencia y capacidad de consulta a través de librerías digitales o bases globales de indexación.
- El nivel de reconocimiento por la comunidad científica mundial.
- Su patrón de citación que determina posicionamiento en los buscadores, sus ventajas acumulativas.
- La pertinencia de autores y consejo editorial a sociedades científicas.

La cantidad de documentos científicos, citación e influencia intelectual es dominada por Estados Unidos, China, Inglaterra y otros países desarrollados no solo porque dominan los factores anteriormente mencionados, sino porque en esos países las publicaciones académicas son en sí mismas una industria. (2)

Hoy Latinoamérica hace grandes esfuerzos por tratar de eliminar la brecha existente. Latinoamérica tiene que enfocarse a lograr un alto reconocimiento internacional sobre la base

de la calidad o lo que es lo mismo, creatividad, originalidad y contribución en un área, eliminar en lo posible las barreras lingüísticas de sus autores y buscar disponibilidad de acceso. Cuba al ser un país subdesarrollado y bloqueado se ve inmersa en el movimiento *Open Access* (OA), de esta forma se eliminan las barreras físicas y económicas que impiden el acceso a la información de los usuarios. Según la declaración de Budapest *(Budapest Open Access Initiative)* se establecen dos rutas para alcanzar el OA. La **ruta dorada** o la de publicación en revistas científicas con revisión por pares (*peer-review*) cuyos contenidos están accesibles sin necesidad de compra o suscripción y la **ruta verde** que alude al archivo o depósito de recursos digitales en repositorios institucionales o temáticos. En los últimos años los repositorios (RIs) han cobrado importancia en la sociedad académica y científica, porque representan una fuente de información digital especializada, organizada y accesible para los lectores de diversas áreas. Los RIs son sistemas informáticos dedicados a gestionar los trabajos científicos y académicos de diversas instituciones de forma libre y gratuita.

Las universidades juegan un papel importante como generadoras y consumidoras de información. Su función principal, como institución de estudios superiores, es impulsar la creación intelectual al servicio de la enseñanza y la investigación. La Universidad de las Ciencias Informáticas (UCI) surge como un centro docente-productor que desarrolla aplicaciones y servicios informáticos orientados a diversos sectores de la economía y los servicios, dentro y fuera de Cuba. La informatización de la sociedad cubana pasa necesariamente por este centro de altos estudios que hace suya la máxima de estar conectados al futuro de Cuba y su Revolución.

La UCI cuenta con un repositorio institucional que tiene como objetivo atesorar, divulgar y conservar la memoria científica. Permite incrementar la visibilidad, y el prestigio de la institución, así como el impacto y el reconocimiento de la producción científica de sus investigadores. Es el encargado de centralizar y mantener disponible las producciones científicas que se generan en la institución.

En el repositorio institucional no existe una forma de medir la calidad de la producción científica de los investigadores de la UCI. En la UCI existe un sistema (SindiCIT) que mide los indicadores de ciencia, tecnología e innovación; pero la evaluación es a nivel de área y no de investigador. Actualmente en el sistema la información es enviada por los jefes de las áreas trayendo consigo errores en la manipulación de los datos. Teniendo en cuenta la poca madurez del personal científico con que cuenta la universidad, se puede decir que los indicadores

relacionados con la visibilidad y la relevancia sean aún débiles con relación a los obtenidos por otros centros del país con muchos años de experiencia y un claustro sólido. Actualmente no se comparan los resultados entre los investigadores de la institución, lo cual dificulta que se inicie y consolide el aumento de publicación en revistas referenciadas internacionalmente que tengan alto índice de impacto, y que a su vez se logre un mayor compromiso de los investigadores con la institución.

Teniendo en cuenta la problemática anterior surge el siguiente **problema a resolver**. ¿Cómo medir la producción científica de un investigador en la UCI?

Se define como **objeto de estudio** la medición de la calidad de las producciones científicas, siendo el **campo de acción** Sistemas para la medición de la calidad de las producciones científicas en repositorios digitales basados en DSpace.

La investigación tiene como **objetivo general** desarrollar un Módulo para el Sistema de repositorios digitales REPXOS 3.0 que permita medir la producción científica de un investigador en la UCI.

Aporte social.
El módulo permitirá medir la calidad de las publicaciones científicas a nivel de investigador que se encuentren almacenadas en el repositorio. Además servirá para incentivar a los investigadores a publicar artículos en revistas referenciadas. Permitirá hacer comparaciones a nivel de investigador para la toma oportuna de decisiones. También facilitará la búsqueda de artículos de un investigador específico y podrá ser accedido por usuarios de la comunidad científica y por los propios administradores del sistema.

Para la presente investigación se trazaron los siguientes **objetivos específicos**:

- Confeccionar el marco teórico de la investigación, relacionado con los principales fundamentos teóricos de la medición de la producción científica.
- Desarrollar los artefactos correspondientes al análisis y diseño de acuerdo a la metodología de desarrollo que se utiliza.
- Implementar el módulo de medición de la calidad de las publicaciones científicas para el sistema REPXOS 3.0 mediante las funcionalidades definidas.
- Realizar una estrategia de pruebas para verificar el funcionamiento del módulo.

A continuación se muestran los métodos científicos que se usan durante la investigación:

Métodos empíricos

Observación: Como método científico permite obtener conocimiento acerca del comportamiento del objeto de estudio. Es utilizado en una primera etapa para formular el problema y en el diseño de la investigación.

Entrevista: Se realiza una entrevista estructurada con el arquitecto del proyecto Repositorio Institucional para determinar las principales funcionalidades con las que debe cumplir el módulo.

Métodos teóricos

Analítico-Sintético: Este método se ha usado durante la confección del diseño teórico de la investigación. Se realizó un análisis sobre los principales conceptos relacionados con la medición de la calidad de las publicaciones científicas, dividiendo su estudio en partes para un mejor entendimiento y luego realizar una síntesis específica y concreta de estos elementos. Además, permitió analizar e identificar las principales características del entorno de desarrollo (metodología, herramientas, tecnologías y lenguaje de desarrollo).

Modelación: Se utilizó para modelar las diferentes etapas del proceso de ingeniería de software por las que transitó el desarrollo del módulo. Mediante el lenguaje unificado de modelado (UML), se refleja la estructura, relaciones internas y características de la solución a través de la modelación de diagramas.

Análisis Histórico-Lógico: Este método ha sido usado en el estudio de la trayectoria y evolución de los indicadores bibliométricos encargados de medir la calidad de las producciones científicas. De igual forma fue usado para investigar las aplicaciones informáticas de este tipo implementadas en Cuba y el resto del mundo.

Para dar cumplimiento a los objetivos trazados, el presente documento está estructurado en tres capítulos. A continuación se expone una breve descripción de cada uno de ellos.

Capítulo 1: Fundamentación teórica. En este capítulo se documentan los conceptos fundamentales que serán tratados a lo largo de la investigación. Se estudian y analizan algunos sistemas encargados de medir la calidad de las publicaciones científicas. Se realiza un estudio de la metodología, las tecnologías, herramientas, y lenguaje que será utilizado para darle solución al problema propuesto.

Capítulo 2: Análisis y diseño. En este capítulo se describe la solución propuesta para la situación problemática. Se presentan las características y funcionalidades del módulo a partir de los requisitos funcionales y no funcionales capturados. Además se abordan las actividades definidas para el diseño del módulo siguiendo la metodología RUP y se especifica la arquitectura del módulo.

Capítulo 3: Implementación y pruebas. En este capítulo se definen los diagramas de componentes y despliegue, obteniendo una visión del módulo. Se justifican los estándares de codificación utilizados y se valida la solución propuesta a través de una estrategia de pruebas.

Capítulo.1 Fundamentación Teórica

Introducción

En el presente capítulo se lleva a cabo la fundamentación teórica del trabajo, se tratan las principales definiciones relacionados con la medición de las publicaciones científicas. Se estudian y analizan elementos relacionados con las herramientas, metodología y tecnologías a utilizar para la creación de una posible solución al problema. Además se realiza un estudio de sistemas encargados de medir la calidad en las producciones científicas.

1.1. Conceptos relacionados con el objeto de estudio

La generación de nuevos productos es el resultado de la investigación científica, esto se traduce en el uso de la innovación tecnológica como factor clave para el crecimiento y avance de la ciencia. Como consecuencia el progreso de la ciencia depende, en gran medida, de la difusión y divulgación del conocimiento. Las publicaciones científicas permiten hacer cambios, transformaciones, avanzar hacia el futuro haciendo un uso racional, eficiente y eficaz de los recursos de que dispone y crea el hombre en su labor científica.

¿Qué es la bibliometría?

A continuación se define el concepto de bibliometría según diferentes autores.

La bibliometría refiere a la ciencia que permite el análisis cuantitativo de la producción científica a través de la literatura, estudiando la naturaleza y el curso de una disciplina científica. El uso de indicadores bibliométricos para analizar la actividad investigadora de un equipo científico, un área o un país, se basa en que las publicaciones científicas son un resultado esencial de dicha actividad. Un nuevo conocimiento adquiere valor cuando se da a conocer y se difunde, ya que es así como progresa la ciencia. La bibliometría permite valorar la calidad científica e influencia tanto del trabajo como de las fuentes, se define como la aplicación de métodos matemáticos y estadísticos a libros y otros medios de comunicación escrita. (3)

La bibliometría es definida como la rama que estudia los aspectos cuantitativos de la producción y uso de la información registrada, a cuyo efecto desarrolla modelos y medidas matemáticas. Los indicadores que se edifican a partir de prácticas bibliométricas cuantifican el número de documentos publicados por un país, institución, grupo de investigación o individuo. (4)

El autor del presente trabajo define como la bibliometría los instrumentos utilizados para medir la calidad de las publicaciones científicas. Además permiten manejar y clasificar grandes volúmenes de publicaciones científicas.

La *Propuesta de Bases para una Política sobre Publicaciones Científicas en los centros adscritos al Ministerio de Educación Superior, en la versión 1 Junio/09* plantea que a nivel del investigador, su producción científica se mide en buena medida por la cantidad de artículos publicados en revistas referenciadas, y el impacto de sus resultados por la cantidad de citas que recibe. (5)

1.1.1. Indicadores bibliométricos

Factor de impacto

El factor o índice de impacto mide la repercusión que ha tenido una revista en la literatura científica a partir del análisis de las citaciones que han recibido los artículos que se han publicado en ella. Permite comparar revistas, establecer *rankings* en función de este factor y reflejar la relevancia relativa de cada título. (6)

¿Cómo se calcula?

El factor de impacto corresponde a la división del número de citaciones que han recibido los artículos de una revista entre el número de artículos que se publicaron en ella. A pesar de que pueden variar ciertos elementos en el cálculo del factor (como el período de años o el tipo de artículos a contar) y que se han de tener en cuenta ciertas variables (artículos breves con gran número de referencias, procedencia y ámbito temático de las publicaciones incluidas), la fórmula más extendida es esta:

Número de citaciones recibidas en un año concreto por los artículos de los dos años anteriores de una revista dividido por el número de artículos publicados en la revista estos dos últimos años. (6)

Es decir:

$$\textbf{Factor de Impacto 2015} = \frac{\text{Citaciones que han recibido en 2015 los artículos publicados en 2013 y 2014}}{\text{Número de artículos publicados en el período 2013 y 2014}}$$

Índice de Inmediatez

El índice de inmediatez de una revista permite conocer la rapidez con la que los artículos publicados son citados en el mismo año en que se publica. Se calcula dividiendo el número de citas a artículos publicados en un año dado, entre el número de artículos publicados en ese año.

$$\text{Índice de Inmediatez} = \frac{\text{Número de citas(Año de estudio)}}{\text{Número de artículos(Año de estudio)}}$$

Índice H

Fue propuesto por Jorge Hirsch, investigador de Física de la Universidad de California, en 2005 con el objetivo de medir la distribución de las citas que han recibido los trabajos científicos de un investigador. Se trata de un balance entre el número de citas que recibe un investigador y el número de publicaciones que ha realizado a lo largo de su carrera, es decir, una media entre cantidad y calidad. (7)

Número de publicaciones	1	2	3	…	28	**29**	…	184	185
Recuento de citas	192	141	103	…	29	**29**	…	0	0

Tabla 1 Índice h de un autor

La tabla muestra el índice h de un autor. Este autor tiene índice h 29, porque 29 de sus 185 publicaciones tienen al menos 29 citas cada una y las otras 154 publicaciones (185-29) tienen menos de 29 citas cada una.

Índice G

El índice G es un indicador que, al igual que el índice H, cuantifica la productividad bibliométrica basada en el historial de publicaciones de los autores. El índice G es un indicador propuesto por Leo Egghe en 2006. También se calcula a partir de la distribución de citas recibidas por las publicaciones de un investigador determinado. Es similar al índice H, más complejo en su cálculo, pero al ser mayor y más variable, permite distinguir entre autores con índice H similar. Se calcula ordenando las publicaciones de un investigador por el número de citas recibidas en orden descendente, numerando la posición, y generando dos nuevas columnas: número de citas recibidas acumulado y número de posición al cuadrado. A continuación se identifica el

número de orden de la posición en la que el número de citas acumuladas es igual o mayor que el número de posición al cuadrado. (8)

Un autor tiene un índice de “G” cuando, considerando los “G” artículos más citados de dicho autor, la cantidad de citas acumuladas por estos "G" artículos es superior a "G" al cuadrado.

Índice G = 15 (la cantidad de citas acumuladas por estos 15 artículos más citados es superior a 255.

Cuartil

Es la unidad utilizada para la medida de posición de una revista. Se utiliza separando los grupos de revistas de una determinada especialidad, ordenados de mayor a menor visibilidad (factor de impacto). Si un listado de revistas ordenadas de mayor a menor factor de impacto se dividiera en 4 grupos iguales, cada una de las partes constituiría un cuartil. Aquellas que contengan mayor factor de impacto estarían en el primer cuartil. (9)

SNIP

Fuente Normalizada de Impacto por Papel: Creado por el profesor Henk Moed. Mide el impacto de la citación contextual mediante la ponderación de las citas en función del número total de citas en un campo. El impacto de una única citación se le da mayor valor en las áreas temáticas en las que tienen menos probabilidades de citas, y viceversa. Su objetivo es permitir la comparación directa de fuentes en diferentes campos temáticos. (10)

Acerca de SNIP:

- Mide el impacto de citas contextuales ‘normalizando’ los valores de las citas.
- Toma en cuenta la frecuencia de la cita en un campo de investigación.
- Considera la inmediatez – la rapidez con que un artículo probablemente tendrá impacto en un área determinada.
- Calcula sin hacer uso de una clasificación temática de la revista científica, para evitar delimitaciones.

Eigen factor

Creado en el año 2007 por C.T. Bergstrom. El Eigen factor es un indicador del prestigio[1] de las revistas, que se basa, al estilo *PageRank*[2], en el cálculo iterativo del prestigio de las citas recibidas por una revista en un período establecido. Eigen factor se calcula en forma conjunta para las revistas en los campos de la ciencia y las ciencias sociales para una ventana temporal de cinco años; ajusta las diferencias entre los patrones de citación de los diferentes campos del conocimiento y elimina las autocitas. La normalización de las conexiones entre las revistas se realiza por medio del número total de referencias emitidas y no en función del número de trabajos publicados. (11)

SJR

Desarrollado en 2007 por Félix Moya de Anegón, del Grupo SCImago, es una medida del prestigio científico de una revista, que se basa en esquemas para la ponderación de las citas recibidas. Su ventana temporal es de tres años. La clasificación de las publicaciones en áreas y categorías temáticas se ejecuta según el esquema de clasificación de Scopus. El SJR se calcula en dos fases: una primera, en la que se calcula el prestigio de la revista a partir del cómputo de las citas ponderadas y recibidas por cada revista, dividido por el total de referencias emitidas en el año de estudio.

El prestigio de cada revista depende de:

- Un valor mínimo de prestigio que se obtiene al ser seleccionada para su procesamiento por la base de datos.
- El prestigio de la publicación, determinado por el número de artículos de la revista incluido en la base de datos.

[1] Prestigio/influencia. Cada vez que ocurre una cita, se produce un intercambio de influencia y de reconocimiento/prestigio entre la obra citante (la que cita) y la obra citada, que va en uno y otro sentido. Si la obra A cita a la obra B, por ejemplo, puede decirse que B influyó sobre A y que A recibió el reconocimiento de B. Ese reconocimiento en el contexto de la teoría del poder en las redes, se denomina prestigio. En la medida en que aumenta el número de citas que recibe una revista aumenta su influencia pero, a la vez también su reconocimiento/prestigio.

[2] Un algoritmo empleado por Google para el análisis de los enlaces, que asigna pesos numéricos a cada documento, página o recurso de un conjunto de ellos (siempre que al menos presenten un vínculo o enlace) en el *World Wide Web* (*WWW*) con el propósito de medir su importancia relativa dentro del conjunto.

- El prestigio de citación, calculado a partir del número y la importancia de las citas recibidas de otras revistas.

La segunda fase convierte esta medida, dependiente del tamaño de la revista (número de trabajos publicados), a otra independiente del tamaño con vista a hacerlas comparables entre sí: el número de artículos, revisiones y materiales de conferencias publicados por la revista. El valor resultante se incrementa proporcionalmente al multiplicarlo por una constante con vistas a facilitar su empleo en las comparaciones. (11)

Puede consultarse en SCImago Journal & Country Rank
Acerca de SJR:

- Es ponderado por el prestigio de la revista científica, igualando así las condiciones entre todas las revistas.
- Elimina la manipulación: la única forma de elevar la clasificación del SJR es ser publicado en revistas científicas de mayor reputación.
- Comparte igualmente el prestigio de una revista científica entre el número total de citas en dicha revista.
- Estandariza las diferencias en el patrón de citación entre las áreas de investigación.

1.2. Análisis de sistemas informáticos

1.2.1. Internacionales

EigenFactor.org

Es un proyecto de investigación académica co-fundada por Jevin West y Carl Bergstrom y patrocinado por el Laboratorio Bergstrom en el Departamento de Biología de la Universidad de Washington. Es creado con el objetivo de desarrollar métodos para evaluar la influencia de las publicaciones científicas y para el mapeo de la estructura de la investigación académica. Ofrece los indicadores bibliométricos *Eigen factor y Article influence* para evaluar la importancia de las revistas científicas. Eigenfactor.org no sólo clasifica las revistas especializadas en las ciencias naturales y sociales, sino también periódicos, tesis doctorales, revistas populares. Las revistas tienen diferentes estándares de citación y diferentes escalas de tiempo en las que se producen las citas; por ejemplo el promedio de un artículo en una revista líder de biología celular podría recibir de 10 a 30 citas en dos años y el promedio de un artículo en revista de matemática serían 2 citas. El algoritmo empleado por este sistema corrige estas diferencias y permite hacer una mejor comparación entre diferentes áreas. (12)

La puntuación *Eigen factor* valora la importancia relativa de una publicación para la comunidad científica, de forma que la suma de las puntuaciones de todas las revistas suman 100; sin embargo esta puntuación está influenciada por el tamaño de una publicación medida por el número de artículos anuales publicados (una revista que duplica el número de artículos que publica, duplica igualmente su puntuación *Eigen factor*). *Article influence* mide la media de la influencia de los artículos de una publicación, por lo que es comparable al factor de impacto del Instituto para la Información Científica (ISI)[3]. (13)

SCImago Journal Rank & Country Rank

El SCImago Journal Rank & Country Rank es un portal que incluye las revistas y los indicadores bibliométricos, contenida en la base de datos Scopus. Estos indicadores pueden ser utilizados para evaluar y analizar los campos científicos. Provee acceso libre a estos indicadores. Esta plataforma debe su nombre al indicador *SCImago Journal Ranking* (SJR). SCImago es un grupo de investigación del Consejo Superior de Investigaciones Científicas (CSIC), la Universidad de Granada, Extremadura, Carlos III (Madrid) y Alcalá de Henares; dedicada al análisis de la información, la representación y la recuperación por medio de técnicas de visualización. Además SCImago ha desarrollado el proyecto Atlas de la Ciencia cuyo objetivo principal es lograr una representación gráfica de la Ciencia Iberoamericana de Investigación. Una representación se concibe como una colección de mapas interactivos, permitiendo funciones de navegación a través de los espacios semánticos formados por los mapas. (14)

En el portal de SCImago se pueden realizar búsquedas por áreas y categorías temáticas, así como por países y ordenar sus resultados por diferentes indicadores (opción Journal rankings). Permite buscar una revista por el título, ISSN o por el nombre del autor (opción Journal Search). Además se pueden comparar países y revistas (opción Compare). El conjunto de revistas puede ser analizada por separadas simplemente haciendo clic en el título de la revista para obtener su perfil individual que incluye tablas y gráficos para comparar las métricas del desempeño de la revista. Entre los indicadores que brinda SCImago se encuentran el SJR y el índice h.

[3] Por sus siglas en inglés Institute for Scientific Information

CWTS Journal Indicators

Provee acceso libre a los indicadores bibliométricos de las revistas científicas. Los indicadores se han calculado por el Centro de la Universidad de Leiden para Estudios Científicos y Tecnológicos (CWTS), basado en la base de datos bibliográfica Scopus. Los indicadores están disponibles para más de 20.000 revistas indexadas en la base de datos Scopus. Puede consultarse en CWTS Journal Indicators Website. (15). En este sitio es posible hallar la información por título de revista y áreas temáticas y además permite organizar la búsqueda según los indicadores bibliométricos.

El CWTS Journal Indicators actualmente ofrece cuatro indicadores:

- **P**: El número de publicaciones de una fuente en los últimos tres años.
- **RIP**: Término que se refiere al impacto bruto por publicación, calculado como el número de citas brindadas en el presente año , dividido por el número total de publicaciones de los últimos tres años. RIP es muy similar al reconocido factor de impacto de las revistas. Al igual que dicho factor, RIP no hace correcciones a las diferencias entre las prácticas de citas entre los campos científicos.
- **SNIP**: Término que se refiere al impacto normalizado de fuente por publicación. calculado como el número de citas brindadas en el presente año dividido por el número total de publicaciones de los últimos tres años. La diferencia con RIP es que en el caso de las citas SNIP, estas son normalizadas con el objetivo de corregir las diferencias en las prácticas de citas entre los campos científicos.
- **% self cit**. Porcentaje de citas autónomas de una fuente, que es calculado como el porcentaje de todas las citas que se originan de la misma fuente, las cuales son dadas en el presente año a las publicaciones de los últimos tres años.

1.2.2. Nacionales

Sistema de Indicadores para medir la Ciencia en la Universidad de las Ciencias Informáticas (SindiCIT)

El actual sistema brinda la posibilidad de llevar un control de los indicadores relacionados con la ciencia, tecnología e innovación que se realiza en la Universidad de las Ciencias Informáticas. El mismo se encuentra en desarrollo y brinda la posibilidad de calcular una serie de parámetros preestablecidos, los cuales pueden dar una idea de cómo se encuentra la institución. Es importante tener en cuenta que el mismo ha sido diseñado para que lo usen personas con determinados conocimientos y no por todos. Para poder ingresar datos en el mismo debe estar

registrado en la aplicación y tener los permisos necesarios sobre la facultad que se quiere trabajar. Estos permisos los concede el administrador del sistema. (16)

Este sistema depende de una serie de entradas que consiste en información cuantitativa de los diferentes indicadores cienciométricos que son introducidos por los asesores de investigación en las diferentes áreas. A partir de estas entradas y a través de un mecanismo de ponderación, el sistema permite un conjunto de salidas que posibilitan la evaluación de la producción científica de la universidad. El sistema posibilita comparar la universidad con otras instituciones en el país, así como establecer un *ranking* entre las diferentes áreas del centro, respectivamente. (17)

1.2.3. Valoración sobre los sistemas estudiados

El sistema **SindiCIT** presenta como dificultad que está diseñado para que solo tengan acceso al mismo personal con determinados conocimientos. SindiCIT es capaz de ponderar de manera diferenciada aquellas investigaciones científicas de ciclo completo (Investigación + Desarrollo + Producción + Comercialización) con relación a aquellas investigaciones puramente académicas y medir, en sentido general, la actividad de la Ciencia, Tecnología e Innovación en la universidad. Además la medición de la actividad científica no se realiza a nivel de investigador sino en un área determinada. Una de las dificultades existentes es que pese a que la información asociada a estos procesos se recopila digitalmente, los métodos para el procesamiento y la gestión de la misma son los tradicionales y no los más efectivos debido a que la información es enviada por los diferentes jefes de las áreas.

Los sistemas **SCImago Journal Rank & Country Rank** y **Eigen Factor.org** brindan, entre otros factores, el SJR y *Eigen factor*. Estos presentan grandes similitudes al intentar imitar ambos el funcionamiento de la fórmula de *PageRank* de Google; es decir, tratan en esencia de asignar un peso a las fuentes que emiten los enlaces. *SCImago Journal Rank & Country Rank* brinda anualmente un archivo .xls con SJR de cada revista. Para el desarrollo del sistema de medición de la calidad de las publicaciones científicas en el repositorio institucional el autor del trabajo selecciona el SJR. Su selección se debe a que con este indicador se puede obtener el prestigio de cada revista que contiene las publicaciones científicas. Además, la calidad de las producciones científicas está dada en gran medida por la cantidad de artículos referenciados en revistas científicas.

El **Eigen factor.org** permite buscar el indicador *Eigen factor* de cada revista por el nombre, categoría a la que pertenece o por un año determinado. No permite obtener de forma directa el *Eigen factor* de todas las revistas en un archivo que brinde la posibilidad de importarlo a la base de datos del repositorio.

En el sistema **CWTS Journal Indicators** el indicador clave ofrecido es el SNIP. Este indicador permite hacer comparaciones directas de fuentes en diferentes campos temáticos. Este portal brinda la posibilidad de exportar un archivo .xls con el SNIP de las revistas indexadas en la base de datos de Scopus para insertarlo en la base de datos del repositorio.

En el repositorio institucional no existe una forma de contar las citas que reciben las publicaciones científicas. Para lograr contar las citas de las publicaciones es necesario emplear un algoritmo matemático que realice búsquedas en base de datos internacionales, lo que trae consigo que se necesite una cuenta de navegación con libre acceso. Los sistemas internacionales estudiados son capaces de contar las citas de las publicaciones científicas y con ello logran crear indicadores que miden el prestigio de las revistas. Además proveen acceso libre a los indicadores.

Se puede afirmar que no existe un indicador perfecto para evaluar la calidad de las publicaciones en revistas científicas. Cada uno tiene su función, ofrecen una información diferente y tienen su contexto de aplicación. Debido a la dificultad que presenta el repositorio institucional se decide utilizar el indicador SJR de *SCImago* y el SNIP de *CWTS Journal Indicators* respectivamente. Luego de obtener estos dos indicadores se decide utilizar las siguientes fórmulas para elaborar un *ranking*, ya que como plantea la resolución del Ministerio de Educación Superior (MES) “Propuesta de Bases para una Política sobre Publicaciones Científicas en los centros adscritos al Ministerio de Educación Superior, en la versión 1 Junio/09” que la producción científica de un investigador se mide en buena medida por la cantidad de artículos publicados en revistas referenciadas y cada revista tiene un valor de prestigio obtenido anteriormente.

$$RankigSJR = \sum_{i=1}^{N} CP_i * SJR_i$$

$$RankigSNIP = \sum_{i=1}^{N} CP_i * SNIP_i$$

RankingSJR: Ranking calculado según el indicador brindado por SCImago.

RankingSNIP: Ranking calculado según el indicador brindado por *CWTS Journal Indicators*.

SJR: Indicador de SCImago.

SNIP: Indicador de *CWTS Journal Indicators*.

CP: Cantidad de publicaciones en una revista.

1.3. Ambiente de desarrollo

En el proceso de desarrollo del software es importante definir la metodología, herramientas tecnologías y lenguaje de desarrollo, para obtener el producto final con la calidad requerida. La metodología, herramientas, tecnologías y lenguajes que se utilizan para la construcción del sistema, fueron establecidas como políticas del proyecto Repositorio Institucional, mediante un estudio realizado por el equipo de arquitectura, por lo que la selección realizada queda fuera del alcance del trabajo. En el presente trabajo solo se realiza una breve descripción de las mismas.

1.3.1. Metodología

La metodología RUP[4], divide en 4 fases el desarrollo del software:

Inicio: El objetivo en esta etapa es determinar la visión del proyecto.

Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.

Construcción: En esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.

Transmisión: El objetivo es obtener el producto final.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. La metodología RUP, es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Este se caracteriza por ser iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura. (18)

[4] Por sus siglas en inglés Rational Unified Process

Dirigido por los casos de uso: Los casos de uso se utilizan para describir los requisitos funcionales del sistema desde la perspectiva del usuario. Establecen el comportamiento que se desee del sistema, para validar y verificar la arquitectura definida en el sistema así como para determinar el alcance de cada iteración y el contenido de trabajo de los integrantes del equipo de desarrollo.

Centrado en la arquitectura: La arquitectura del sistema permite ganar control sobre el proyecto para manejar su complejidad y controlar su integridad, mostrando una visión común del sistema para los clientes y el equipo de proyecto. Se utiliza para construir, conceptualizar y evolucionar el sistema. Hace posible la reutilización a gran escala y provee una base para la gestión del proyecto.

Iterativo e incremental: El proceso iterativo e incremental consta de una secuencia de iteraciones a partes pequeñas o mini-proyectos siendo más práctico. Cada iteración aborda una parte de la funcionalidad total, pasando por todos los flujos de trabajo relevantes y refinando la arquitectura. Toda la retroalimentación de la iteración pasada permite reajustar los objetivos para las siguientes iteraciones. Se continúa con esta dinámica hasta que se haya finalizado por completo con la versión actual del producto.

Herramientas, tecnologías y lenguajes

El uso de herramientas, tecnologías y lenguajes se hace imprescindible para el desarrollo de un sistema informático. A continuación se especifican y se describen las herramientas, las tecnologías, el lenguaje de modelado, la herramienta CASE y los lenguajes de programación utilizados en la construcción del sistema.

1.3.2. Servidor de aplicaciones

Usualmente se trata de un dispositivo de software que proporciona servicios de aplicación a las computadoras cliente. Un servidor de aplicaciones generalmente gestiona la mayor parte (o la totalidad) de las funciones de lógica de negocio y de acceso a los datos de la aplicación. Los principales beneficios de los servidores de aplicación son la centralización y la disminución de la complejidad en el desarrollo de aplicaciones.

Apache Tomcat v7.0.39

Es multiplataforma, lo que lo convierte en un servidor prácticamente universal; es un contenedor de servlets/JSP, utiliza la tecnología Apache que es gratuita y de código abierto lo que le da

transparencia al software. Permite escribir y desplegar aplicaciones web complejas de forma sencilla, proporcionando el soporte para características de desarrollo que de otra forma tendrían que ser desplegadas manualmente. Resuelve las opciones de autenticación de usuarios, los registros de log, así como la infraestructura y los filtrados a nivel de aplicación y es altamente configurable. (19) Esta herramienta posee las siguientes características:

- Es de código abierto.
- Permite funcionar en cualquier sistema operativo que se encuentre la máquina virtual de Java.
- Fácil de instalar.
- Ocupa muy poco espacio.

Estas características hacen que Apache Tomcat se integre al conjunto de tecnologías utilizadas para la creación del sistema. Es de gran ayuda porque al ser un contenedor web escrito en Java funciona en cualquier sistema operativo que disponga de una máquina virtual de java previendo posible migración de sistema operativo.

1.3.3. Sistema Base

DSpace v4.2

DSpace es un software de código abierto diseñado por el Instituto Tecnológico de Massachusetts (MIT) y los laboratorios de *Hewlett Packard* (HP), sin fines de lucro y organizaciones comerciales que construyen depósitos digitales abiertos. Es gratuito y fácil de instalar y completamente personalizable para adaptarse a las necesidades de cualquier organización. DSpace preserva y permite el acceso fácil y abierto a todo tipo de contenido digital, incluyendo texto, imágenes, fotografías, videos, datos de investigación. Los datos son organizados como ítems que pertenecen a una colección; cada colección pertenece a una comunidad. (20)

1.3.4. Herramienta de Instalación

Apache ANT v1.8

Es una herramienta de automatización de tareas que ayuda al desarrollador en la integración del trabajo involucrado en el proceso de desarrollo de software. Presta un conjunto de tareas desarrolladas en Java, útiles para la integración de procesos multiplataforma y de diversas fuentes. Algunas ventajas de ANT radican en su facilidad de uso dado que sin este tipo de herramientas debían realizarse scripts con comandos del sistema. Mediante la utilización de

ANT es posible reutilizar el proceso descrito en el XML de configuración en distintas plataformas. ANT es flexible y no impone convenciones de codificación o diseños de directorio para los proyectos de Java que lo adoptan como una herramienta de construcción. El principal uso conocido de ANT es la acumulación de las aplicaciones Java. Suministra una serie de tareas integradas que permiten compilar, ensamblar, probar y ejecutar aplicaciones Java. (21)

1.3.5. Herramientas de modelado

Una herramienta CASE [5](Ingeniería de Software Asistida por Ordenador) es una aplicación informática que permite la realización de un buen diseño para el proyecto, implementando a partir de él, parte del código automáticamente. Las herramientas CASE nacen para auxiliar a los desarrolladores de software, lo que permite el apoyo computarizado en todo o en parte del ciclo de vida del desarrollo de un sistema de software. Son un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida de los sistemas de información, completamente en alguna de sus fases. El empleo de herramientas CASE permiten integrar los procesos del ciclo de vida. (22)

Visual Paradigm for UML 8.0 Enterprise Edition

Es una herramienta UML[6] profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una construcción más rápida de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (23)

Los beneficios que se consiguen al utilizar Visual Paradigm son varios, por un lado el uso de lenguajes visuales facilita su asimilación y entendimiento por parte del desarrollador; el tiempo invertido en el desarrollo de la arquitectura se minimiza y la trazabilidad y documentación del proyecto se realiza de una forma ordenada.

[5] Por sus siglas en inglés Computer Aided Software Engineering

[6] Por sus siglas en inglés Unified Modeling Language

1.3.6. Sistema gestor de base de datos

Un sistema de bases de datos es un sistema computarizado para llevar registros. Es posible considerar a la propia base de datos como una especie de armario electrónico para archivar; es decir, es un depósito o contenedor de una colección de archivos de datos computados. (24)

PostgreSQL v9.1

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD[7] y con su código fuente disponible libremente. Es un sistema de gestión de bases de datos de código abierto potente. 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. (25)

PostgreSQL brinda muchas ventajas a la hora de desarrollar el módulo debido a que es un gestor de base de datos multiplataforma, presentando una licencia que puede distribuirse gratis para cualquier propósito. Existe una abundante documentación sobre la misma. Además favorece en gran medida proceso de transición y migración hacia software libre en el que se encuentra el país.

1.3.7. Entorno de desarrollo integrado

Netbeans v7.4.1

Es un entorno de desarrollo integrado (IDE[8]) una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java que permite crear aplicaciones web utilizando los lenguajes Java, Java Script y, HTML. Netbeans IDE es un producto libre y gratuito sin restricciones de uso. (26)

1.3.8. Herramienta de compilación

Maven v3.0

Significa acumulador de conocimientos. Se inició originalmente como un intento de simplificar los procesos de construcción del proyecto de turbina de Yakarta, una forma fácil de publicar la información del proyecto y una forma de compartir archivos JAR a través de varios proyectos. El

[7] La licencia BSD es la licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution). Es una licencia de software libre. La licencia BSD permite el uso del código fuente en software no libre.

[8] Por sus siglas en Inglés *integrated development environment*

resultado es una herramienta que se puede utilizar para la construcción y la gestión de cualquier proyecto basado en Java. Asume la estructura de los ficheros del proyecto, con lo cual a la hora de compilar no se pierde tiempo indicando donde se encuentra el código fuente. El objetivo principal de Maven es permitir al desarrollador comprender el estado completo de un esfuerzo de desarrollo en el menor período de tiempo. (27)

Las características de la herramienta permiten la compilación del módulo como un subsistema dentro del sistema REPXOS. Por tanto se hace imprescindible el uso de esta herramienta para el desarrollo del Módulo para medir la calidad de las publicaciones científicas en el Sistema para repositorios digitales REPXOS 3.0.

1.3.9. Generador de reportes

IText: Es una biblioteca de código abierto para crear y manipular archivos PDF, RTF, y HTML en Java. El mismo documento puede ser exportado en múltiples formatos, o múltiples instancias del mismo formato. Los datos pueden ser escritos a un fichero o desde un Servlet [9] a un navegador web. Más recientemente, ha sido extendida a una biblioteca PDF de propósito general, capaz de rellenar formularios, mover páginas de un PDF a otro. Estas extensiones son a menudo mutuamente excluyentes. Una clase te permite rellenar en formularios, mientras una clase diferente e incompatible hace posible copiar páginas de un PDF a otro. (28)

1.3.10. Lenguaje de Modelado

El Lenguaje de Modelado Unificado:

El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño. Una de las metas principales de UML es avanzar en el estado de la integración institucional proporcionando herramientas de interoperabilidad para el modelado visual de objetos. Sin embargo para lograr un intercambio exitoso de modelos de información entre herramientas, se requirió definir a UML una semántica y una notación. (29)

Se puede decir que es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad y se centra en la representación gráfica de un sistema. Este lenguaje permite:

[9] Un servlet es una clase que se ejecuta en el contexto de un servidor web

- **Visualizar**: Permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- **Especificar**: Permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir**: A partir de los modelos especificados se pueden construir los sistemas diseñados.
- **Documentar**: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

1.3.11. Lenguaje de desarrollo

Java

Fue diseñado como un lenguaje orientado a objetos. Proporciona una colección de clases para su uso en aplicaciones, además de numerosas comprobaciones en compilación y en tiempo de ejecución. Está diseñado para soportar variados sistemas operativos. Java se utiliza para crear aplicaciones y procesos que funcionen en multitud de dispositivos. Las aplicaciones Java se comunican con la máquina virtual Java, y no con el sistema operativo, lo cual permite a los programadores desentenderse de la compatibilidad con el hardware. (30)

JavaScript

JavaScript es un lenguaje de programación que ha permitido el gran desarrollo de la Web. Se utiliza en la creación de páginas Web dinámicas y exactas en cuanto a posición y presentación de su contenido. Es un lenguaje robusto y a la vez ligero, el cual a pesar de ser considerado por muchos como un lenguaje no orientado a objetos permite implementar varias de las características de este paradigma de programación. Está basado en prototipos, las nuevas clases se generan clonando las clases bases y extendiendo su funcionalidad. Cualquier navegador puede interpretar el código JavaScript dentro de las páginas Web. Permite la máxima interactividad entre el usuario y la página, además la verificación de los datos introducidos por el usuario antes de enviar el formulario al servidor. (31)

Este lenguaje, además de las características planteadas se ajusta al desarrollo del módulo por su compatibilidad con la mayoría de los navegadores modernos. Permite crear efectos especiales en las páginas y definir interactividades con el usuario.

Conclusiones

Durante la realización de este capítulo se obtuvo el siguiente resultado. Se logró recopilar información importante sobre los indicadores bibliométricos y determinar que no existe un indicador general para medir la calidad. Se obtuvo que las soluciones informáticas estudiadas brindan indicadores encargados de medir la calidad que estos ya están calculados y pueden ser utilizados en la solución del problema tal es el caso del SJR proporcionado por *SCImago* y el SNIP proporcionado por *CWTS Journal Indicators*. Se determinó que las herramientas propuestas por el proyecto Repositorio Institucional se ajustan para el desarrollo del sistema.

Capítulo.2 Análisis y Diseño

Introducción

En el presente capítulo se describe la propuesta de solución del Módulo para medir la calidad de las publicaciones científicas en el Sistema para repositorios digitales REPXOS 3.0. Se desarrollan los primeros flujos de trabajo de la metodología antes propuesta (RUP). Se realiza la confección del modelo de dominio, también se definen los requerimientos funcionales y no funcionales del sistema. Se diseñan los casos de uso del con sus descripciones textuales, también se muestran los diagramas y descripciones correspondientes.

2.1. Propuesta de solución

Después de realizar un estudio de la situación problemática, se propone como solución, la implementación de un módulo que sea capaz de medir la calidad de las producciones científicas en el repositorio institucional. El módulo será el encargado de mostrar un ranking de los autores, utilizando para ello los indicadores SJR de *SCImago* y SNIP de *CWTS Journal Indicators*. Estos indicadores ya están calculados en dichos sistemas, por lo que no es necesario emplear fórmulas para calcularlos; además que en el repositorio no existe una forma de contar las citas que obtiene un artículo almacenado en el mismo. Para calcular el *ranking* se utilizan dos fórmulas creadas por el autor llamadas **RankingSJR** y **RankingSNIP** basadas en la definición del MES. Cuando el usuario acceda a la opción que le permita buscar el *ranking* se muestra una lista con los autores según el indicador seleccionado.

El módulo además servirá para obtener un listado de las publicaciones de un autor. De esta forma permitirá que cada autor tenga asociado a él una página con sus publicaciones. Estas últimas van a estar ordenadas cronológicamente, además se pueden seleccionar por tema. El sistema facilitará la búsqueda de artículos asociados a un autor específico. El módulo incentivará a los investigadores a publicar en revistas científicas con un alto valor de prestigio. Permite depositar las publicaciones científicas en el repositorio institucional para que sean consultadas por la comunidad universitaria y poder ocupar lugares privilegiados en el ranking de las producciones científicas.

2.2. Modelo de dominio

El modelo de dominio o conceptual se utiliza para capturar y expresar el entendimiento en un área bajo análisis como paso previo al diseño de un sistema, además de ser utilizado como un

medio para comprender el sector de negocios al cual el sistema va a servir, tomándose como el punto de partida para el diseño del sistema. (32)

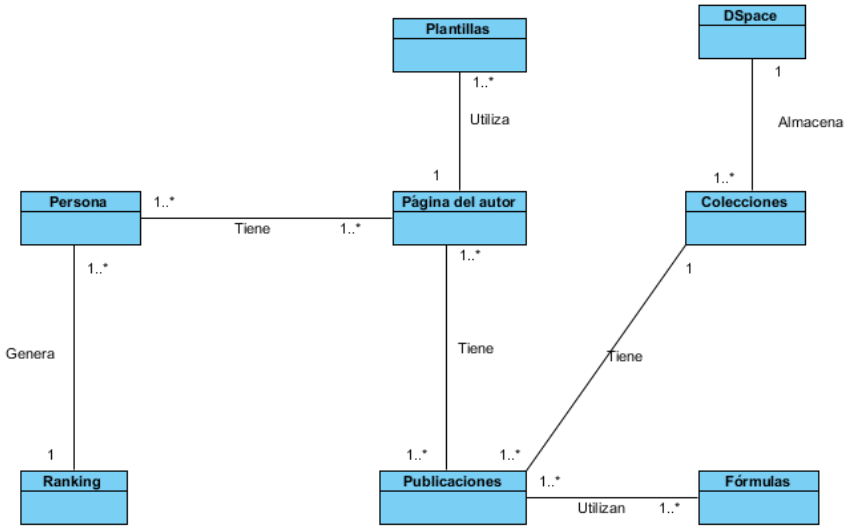

Figura 1: Diagrama de dominio

Definición de los conceptos del modelo del dominio:

- **Colecciones**: Conjunto de publicaciones ordenadas por un tema en común.
- **DSpace**: Herramienta base para la creación de repositorios institucionales.
- **Fórmulas**: Son utilizadas para calcular el ranking de un autor.
- **Página del autor**: Contiene las publicaciones de un autor.
- **Persona**: Es la persona que interactúa con el sistema RepXOS 3.0.
- **Plantillas**: Son utilizadas para crear la página de un autor.
- **Publicaciones**: Conjunto de investigaciones de un autor.
- **Ranking**: Muestra una lista de autores ordenados según la calidad de sus publicaciones.

Una vez comprendido los principales aspectos a manejar en el módulo, se puede pasar a la definición de los requisitos funcionales y no funcionales.

2.3. Requisitos funcionales

Un requisito funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requisitos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir. Los requerimientos de comportamiento para cada requisito funcional se muestran en los casos de uso. Los requisitos

funcionales establecen los comportamientos del sistema. (33) Los requisitos funcionales del módulo se muestran a continuación:

RF1: Gestionar plantilla.
RF1.1: Crear plantilla.
RF1.2: Modificar plantilla.
RF1.3: Eliminar plantilla.
RF2: Ver listado de publicaciones por autor.
RF3: Exportar página del autor a pdf.
RF4: Mostrar ranking de autores.
RF5: Generar ranking.

2.4. Requisitos no funcionales

Son requisitos que imponen restricciones en el diseño o la implementación. Son propiedades o cualidades que el producto debe tener. Existen múltiples categorías para clasificar a los requisitos no funcionales. (18) A continuación se muestran los requisitos no funcionales con las que debe cumplir el módulo:

1. **RNF1: Confiabilidad**

Disponibilidad. El sistema contribuye a medir la calidad de las publicaciones científicas en el repositorio institucional, por lo cual debe estar disponible las 24 horas todos los días.

2. **RNF2: Seguridad**

El sistema debe garantizar la protección de los datos almacenados. Para ello se establecerán diferentes niveles de acceso, garantizando que cada usuario tenga acceso solamente a las funcionalidades permitidas. Además de que la arquitectura del sistema REPXOS garantiza de forma general la seguridad del sistema esto se logra mediante la correcta configuración de web.xml, encargado de controlar el acceso.

3. **RNF3: Requerimientos de Software.**

- La PC cliente debe de contar con un navegador web: Navegador Web, lector de PDF.
- La PC servidor tiene que contar con el servidor de aplicaciones web Apache Tomcat v7.0.39. Servidor de base de datos PostgreSQL 9.1.Máquina Virtual de Java 7. Sistema operativo Windows o Linux.

4. **RNF3: Requerimientos de Hardware.**

- La PC cliente debe de contar con una tarjeta de red.
- La PC servidor microprocesador Intel Core i3-2120 a 3.30 Ghz. 2GB de memoria RAM y un espacio libre en disco duro de 4 GB para la instalación del sistema.

Por lo tanto los recursos mínimos para que el módulo funcione de forma adecuada deben estar en correspondencia con los descritos anteriormente, de lo contrario no se garantiza el correcto funcionamiento del mismo.

5. **RNF4: Restricciones en el diseño y la implementación**

Se usará lenguaje de programación Java. Las herramientas que se utilizan son de distribución bajo licencias libres.

Luego de haber definido los requisitos funcionales, se procede a la agrupación de los mismos en Casos de Uso, con la representación del diagrama obtenido y las especificaciones de estos.

2.5. Diagrama de casos de uso

Un diagrama que muestra un conjunto de casos de uso y de actores, los diagramas de caso de uso muestran los casos de uso de un sistema desde el punto de vista estático. Le permite al programador comprender el contexto del negocio y la agrupación de los requisitos. (18) A continuación se observa el diagrama de casos de uso del Módulo para medir la calidad de las publicaciones científicas en el repositorio institucional.

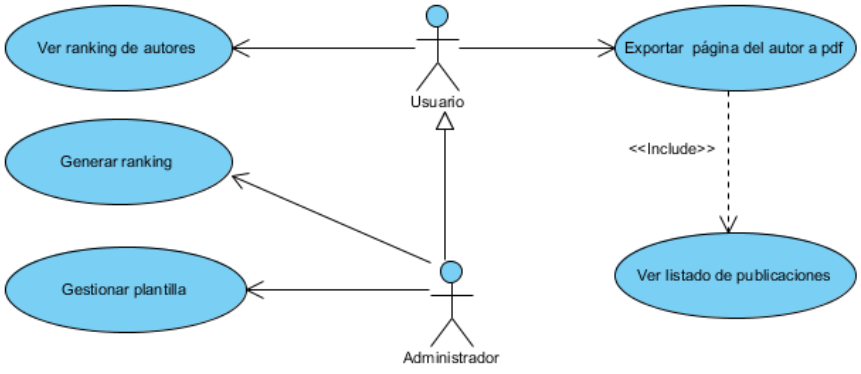

Figura 2: Diagrama de casos de uso

2.6. Definiciones de actores

Un actor es una idealización de una persona externa, de un proceso, o de una cosa que interactúa con un sistema, un subsistema, o una clase. Un actor caracteriza las interacciones

que los usuarios exteriores pueden tener con el sistema. En tiempo de ejecución, un usuario físico puede estar limitado a los actores múltiples dentro del sistema. Diferentes usuarios pueden estar ligados al mismo actor y por lo tanto pueden representar casos múltiples de la misma definición de actor. (18)

Actor	Descripción
Usuario	Es la persona que posee permisos limitados, en este caso es la encargada de ver las publicaciones de un autor y ver el lugar que ocupa un autor en el ranking de publicaciones.
Administrador	Es la persona que posee permisos especiales, en este caso es la encargada de gestionar la plantilla y actualizar el ranking, además de las funciones que realiza un usuario.

2.7. Especificación y resumen de los casos de uso

La descripción de casos de uso permite describir paso a paso el orden de los eventos que los actores siguen para completar un proceso mediante el sistema. A continuación se muestra un resumen de los casos de uso del sistema. La especificación de los casos de uso se encuentran en el anexo1 **(Ver Anexo 1).**

CU 1. Gestionar plantilla

CU_1	Gestionar plantilla
Actor	Administrador
Descripción	El caso de uso se inicia cuando el actor desea crear, modificar, o eliminar la plantilla. Si el actor selecciona la opción de crear, el sistema le permite especificar los datos y registrarlos. Si desea modificar, el sistema muestra los datos y permite modificar los datos requeridos. Si el actor selecciona la opción eliminar, el sistema elimina la plantilla. Termina así el caso de uso.
Referencia	RF1

CU2.Ver listado de publicaciones de un autor

CU_2	Ver listado de publicaciones de un autor
Actor	Usuario
Descripción	Permite buscar las publicaciones de un autor específico, dando la posibilidad de seleccionar un rango de fecha determinando, ordenarlas cronológicamente, seleccionar las colecciones, para después poder ver un listado de las publicaciones de un autor.
Referencia	RF2

CU3. Exportar página del autor a pdf

CU_3	Exportar página del autor a pdf
Actor	Usuario.
Descripción	El caso de uso se inicia cuando el actor desea exportar a formato pdf las publicaciones de un autor. El actor puede seleccionar el tipo de plantilla donde desea exportar las publicaciones.
Referencia	RF3

CU4. Mostrar ranking de autores

CU_4	Mostrar ranking de autores
Actor	Usuario
Descripción	El caso de uso se inicia cuando el actor desea buscar el lugar que ocupan en el ranking uno o varios autores. Además permite mostrar un ranking según indicadores predefinidos.
Referencia	RF4

CU5. Generar ranking

CU_5	Generar ranking
Actor	Administrador
Descripción	El caso de uso se inicia cuando el actor desea actualizar el ranking porque se hayan subido nuevas publicaciones al repositorio o actualizados los indicadores bibliométricos. Si el actor selecciona la opción que le permite generar el ranking el sistema actualiza el ranking existente.
Referencia	RF5

Luego de haber definidos los CU del módulo, se comienza el flujo de trabajo análisis y diseño.

2.8. Modelo de análisis

El modelo de análisis es una primera aproximación al diseño que ofrece una especificación más precisa de los requerimientos utilizando el lenguaje de los desarrolladores. El modelo de análisis ofrece una visión conceptual precisa y unificada de alternativas para la implementación de los sistemas. La estructura impuesta por el modelo de análisis se define mediante una jerarquía y su estructura se representa mediante un sistema de análisis que denota el paquete de más alto nivel del modelo. Este modelo crece incrementalmente conforme se analizan más casos de uso, de manera que el sistema se construye como una estructura de clases de análisis y relaciones entre dichas clases. (18) A continuación se presentan los estereotipos utilizados en los diagramas de clases del análisis, que se muestran más adelante.

Imagen	Nombre	Descripción
	Interfaz	Modelan la interacción entre el sistema y sus actores.
	Entidad	Modelan información que posee larga vida y que es a menudo persistente.
	Control	Implementan la funcionalidad del caso de uso.

Tabla 2 Estereotipos del diagrama de clases del análisis

2.8.1. Diagrama de clases del análisis

El Diagrama de Clases del Análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Constituye una vista estática de las clases que conforman el modelo del análisis y las asociaciones entre las mismas. Este va a representar el funcionamiento del mundo real, no de la implementación automatizada del mismo. Estos diagramas representan las definiciones y relaciones entre las clases.

A continuación se presenta el diagrama de clases de análisis correspondiente al caso de uso Ver ranking de autores, para consultar el resto de los diagramas de clases de análisis **(Ver Anexo 2)**.

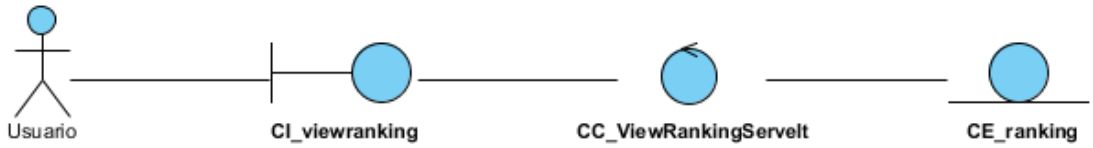

Figura 3 Diagrama de clases del análisis Ver ranking de autores

- **CC_ViewRankingServlet**: Muestra la clase controladora encargada de implementar las funcionalidades utilizadas para mostrar el ranking.
- **CE_ranking**: Representa la clase entidad encargada de almacenar los datos relacionados con el ranking.
- **CI_viewranking**: Es la clase interfaz que se encarga de mostrar el ranking.

2.8.2. Diagrama de colaboración

Un diagrama de colaboración recuerda un diagrama de clase pero contiene instancias y enlaces en lugar de clases y asociaciones. Muestra cómo interactúan los objetos secuencialmente o en paralelo numerando los mensajes que se envían unos a otros. Como complemento a un diagrama de colaboración, los desarrolladores pueden utilizar textos para explicar cómo interactúan los objetos para llevar a cabo el flujo de eventos del caso de uso.

A continuación se presenta el diagrama de colaboración correspondiente al caso de uso Ver ranking de autores para consultar el resto de los diagramas de colaboración **(Ver Anexo 3)**.

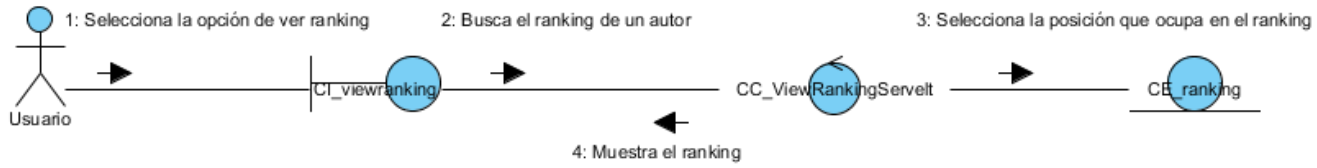

Figura 4: Diagrama de colaboración Ver ranking de autores

La imagen muestra el diagrama de colaboración correspondiendo al caso de uso Ver ranking de autores. El usuario selecciona la opción que le permite ver el ranking y se muestra en la clase interfaz **CI_viewranking**. La clase **CC_ViewRanking** es la clase controladora encargada de implementar la funcionalidad utilizada para ver el ranking y esta selecciona la posición que ocupa en el ranking.

2.9. Modelo de diseño

El modelo de diseño constituye el conjunto de diagramas que describen el diseño lógico de un sistema. Comprende los diagramas de clases software, diagramas de interacción, diagramas de paquetes, ofreciendo una perspectiva de especificación o implementación, como quiere el modelador. (34)

2.9.1. Diagrama de clases del diseño

Un diagrama de clases de diseño representa las especificaciones de las clases e interfaces software en una aplicación. Entre la información general se encuentra:

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Información acerca del tipo de los atributos.
- Navegabilidad.
- Dependencias.

A diferencia de las clases conceptuales del Modelo del Dominio, las clases de diseño de los diagrama de clases de diseño muestran las definiciones de las clases software en lugar de los conceptos del mundo real. (34)

A continuación se presenta el diagrama de clases del diseño correspondiente al caso de uso Ver ranking de autores, para consultar el resto de los diagramas de clases del diseño **(Ver Anexo 4)**.

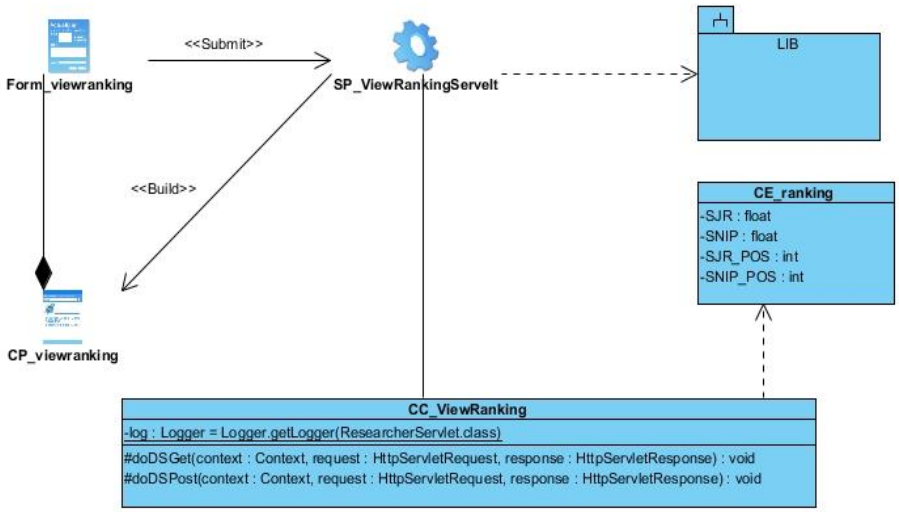

Figura 5: Diagrama de clases del diseño Ver ranking de autores

La imagen muestra el diagrama de clases del diseño correspondiente al caso de uso Ver ranking de autores. La clase **SP_ViewRankingServlet** representa la clase servidora que crea la página cliente que representa la clase interfaz que muestra el ranking de los autores y tiene asociado a ella un formulario que envía datos a la clase servidora. El subsistema **Lib** representa las librerías utilizadas por la clase servidora para su correcto funcionamiento. La clase **CE_ranking** muestra la clase entidad.

Imagen	Nombre	Descripción
	ClientPage	Representa una clase interfaz del sistema.
	ServerPage	Es una clase servidora, que se encarga de manejar los datos.
	Formulario	Permite recopilar información.

Tabla 3: Estereotipos del Modelo de Diseño

2.9.2. Diagrama de paquete

El diagrama de paquete es un mecanismo de organización de los paquetes y sus elementos que subdividen el modelo en otros más pequeños que colaboran entre sí. Están constituidos por dos tipos de elementos: (18)

- **Paquetes**: Permiten dividir un modelo en partes manejables mediante la agrupación de elementos que pueden ser casos de uso, clases o componentes. Pueden anidar otros paquetes dentro de sí.
- **Dependencias**: Indican que un elemento de un paquete requiere a otro de otro paquete distinto.

A continuación se presenta el diagrama de paquete correspondiente al Módulo.

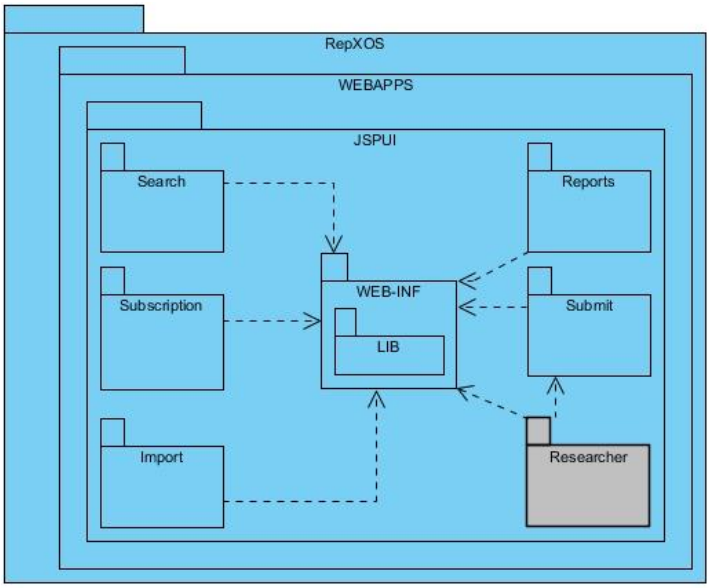

Figura 6: Diagrama de Paquete

En este diagrama se puede apreciar un paquete que contiene todos los demás paquetes el cual es el sistema REPXOS. Dentro de este paquete se encuentran varios paquetes pero el principal para la solución del problema es el paquete webapps. Este paquete contiene el paquete JSPUI, aquí se encuentran todas las clases JSP y clases java Servlets que componen el sistema. En el paquete WEB-INF se encuentra las librerías necesarias para el correcto funcionamiento del sistema REPXOS.

A continuación se muestra el propósito de cada paquete para su mejor entendimiento.

- **WEB-INF**: Es el paquete principal que contiene los elementos de configuración que permiten ejecutar las operaciones en sistema, garantiza la seguridad y los roles de usuarios.
- **Submit**: Es el paquete que contiene todas las clases relacionadas con el almacenamiento de las publicaciones en el sistema.
- **Import**: Contiene todos los elementos relacionados con la importación de archivos en el sistema.
- **Search**: Es el paquete que tiene todas las clases relacionadas con el proceso de búsqueda.
- **Subscription**: Es el paquete que tiene todas las clases para realizar las suscripciones de cada usuario.
- **Recommendation**: Es el paquete que contiene las clases para realizar las recomendaciones de materiales a un usuario.
- **Researcher***:* Es el paquete que contiene todas las funcionalidades encargadas de medir la calidad de las publicaciones científicas, se destaca en color diferente, pues este contiene todas las clases que se implementan para el desarrollo del módulo propuesto.

2.10. Modelo de datos

El modelo de datos es el conjunto de herramientas conceptuales para describir la representación de la información en términos de datos. Los modelos de datos comprenden aspectos relacionados con: estructuras y tipos de datos, operaciones y restricciones. (35)

2.10.1. Diagrama entidad relación

Un Diagrama o Modelo Entidad Relación (E-R)[10], es una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información así como sus interrelaciones y propiedades. En el mismo se pretenden visualizar los elementos que pertenecen a una base de datos, que reciben el nombre de entidades, las cuales se corresponden con el concepto de clase de la programación orientada a objeto y donde cada tupla de una futura relación representaría un objeto de la programación orientada a objetos. (36)

[10] Por sus siglas en inglés Entity Relationship

La imagen muestra el diagrama entidad relación

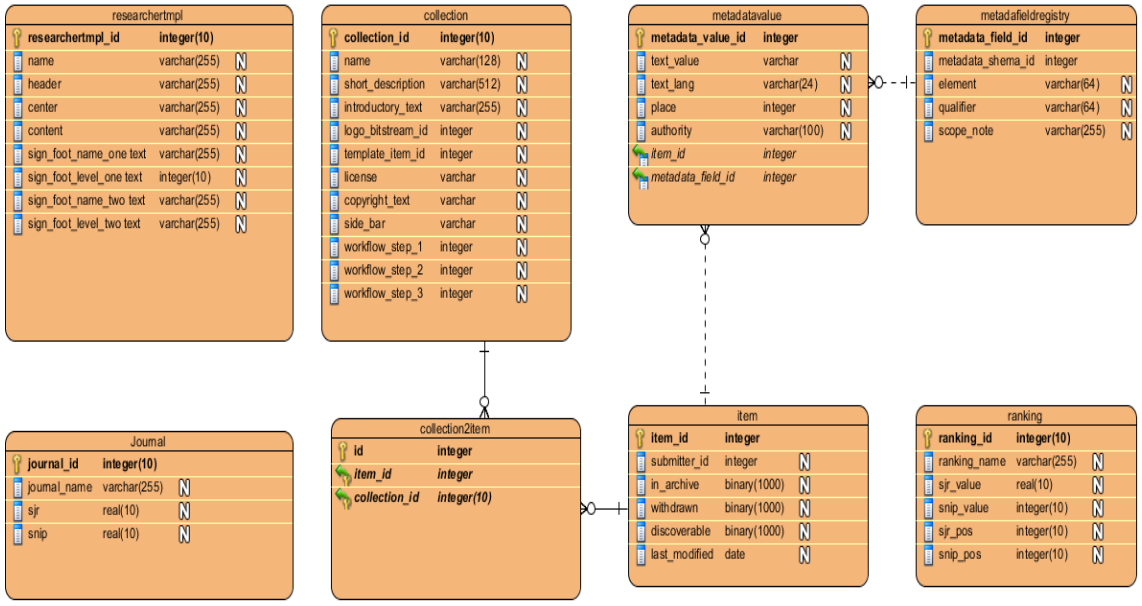

Figura 7: Diagrama entidad relación

A continuación se muestra el propósito de cada entidad para su mejor entendimiento.

- **Collection**: Entidad que contiene un conjunto de Ítem.
- **Collection2item**: Contiene la relación de las colecciones con un ítem.
- **Ítem**: Entidad donde se almacenan los datos de los documentos subidos al repositorio así como últimas modificaciones.
- **Journal**: Entidad que contiene los indicadores bibliométricos SNIP y SJR de las revistas científicas.
- **Metadatavalue**: Entidad que contiene los valores de los metadatos.
- **Metadatafieldregistry**: Contiene los tipos de metadatos del sistema.
- **Ranking**: Entidad que contiene el lugar que ocupa en el ranking un autor.
- **Researchertmpl**: Esta entidad es la encargada de gestionar los datos de la plantilla del autor.

2.11. Patrones utilizados

Un patrón es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos; en teoría indica la manera de utilizarlo en circunstancias diversas. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas. Los patrones de diseño son la base para la

búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. (37)

2.11.1. Patrón de arquitectura

Los patrones de arquitectura definen la estructura general de un software, imponen una transformación en el diseño de una arquitectura, indican la relación entre los subsistemas y los componentes del software y definen las reglas para especificar las relaciones entre los elementos (clases, paquetes, subsistemas) de la arquitectura. El sistema informático REPXOS 3.0 está basado en la herramienta para la creación de repositorios DSpace, la cual presenta el patrón arquitectónico de tres capas. Las capas en que está dividido DSpace son las siguientes: capa de aplicación, capa de lógica de negocios y capa de almacenamiento. Estas ofrecen servicios a la capa superior por medio de APIs[11], y utiliza los servicios de la capa inferior. El modelo de tres capas es una forma lógica de agrupar los componentes que se crean. Está basado en el concepto de que todos los niveles de la aplicación son una colección de componentes que se proporcionan servicios entre sí o a otros niveles adyacentes. (38)

Capa de aplicación: Se encarga de que el sistema interactúe con el usuario y viceversa, muestra el sistema al usuario, le presenta la información y obtiene la información del usuario en un mínimo de proceso. Se comunica únicamente con la capa intermedia o de negocio. También cuenta con un conjunto de módulos que permiten la interacción con el mundo exterior. (38)

En esta capa se encuentran almacenadas las interfaces contenidas en los ficheros jsp. Estas clases son las que hacen posible que el usuario pueda intercambiar con el sistema ya que se comunican con las páginas servidoras que contienen las funcionalidades implementadas. En el caso del módulo un ejemplo concreto es el uso del fichero *viewranking.jsp*.

Capa de lógica de negocios: Esta capa contiene las funciones que se ejecutan, se reciben las peticiones del usuario, se procesa la información y se envían las respuestas tras el proceso. Se denomina capa de negocio o capa de lógica del negocio, porque es aquí donde se establecen

[11] Por sus siglas en inglés Application Programming Interface

todas las reglas que deben cumplirse. Esta capa se comunica con la de aplicación, para recibir las solicitudes y presentar los resultados, y con la capa de almacenamiento, para solicitar al gestor de base de datos almacenar o recuperar datos de él. (38)

Dentro de esta capa en el módulo se implementan los Servlets, que son clases que contienen la lógica de negocio de la aplicación. En estos se implementan las funcionalidades del módulo y son los encargados de obtener información de la base datos ya que esta capa se conecta con la capa de almacenamiento. Entre los Servlets implementados en el módulo se encuentra el *CreateTemplateServlet.java.*

Capa de almacenamiento: La capa de almacenamiento es el puente entre la capa de Lógica de Negocio y el Sistema de Base de Datos. Encapsula la lógica de acceso a datos. Aquí se encuentran componentes que hacen transparente el acceso a la base de datos. Este es el lugar idóneo para implementar los objetos de acceso a datos, permitiendo ingresar, obtener, actualizar y eliminar información del Sistema de Bases de Datos. Es responsable del almacenamiento físico de los metadatos y el contenido. (38)

A continuación se muestra una figura donde se representan las capas que conforman la arquitectura del sistema.

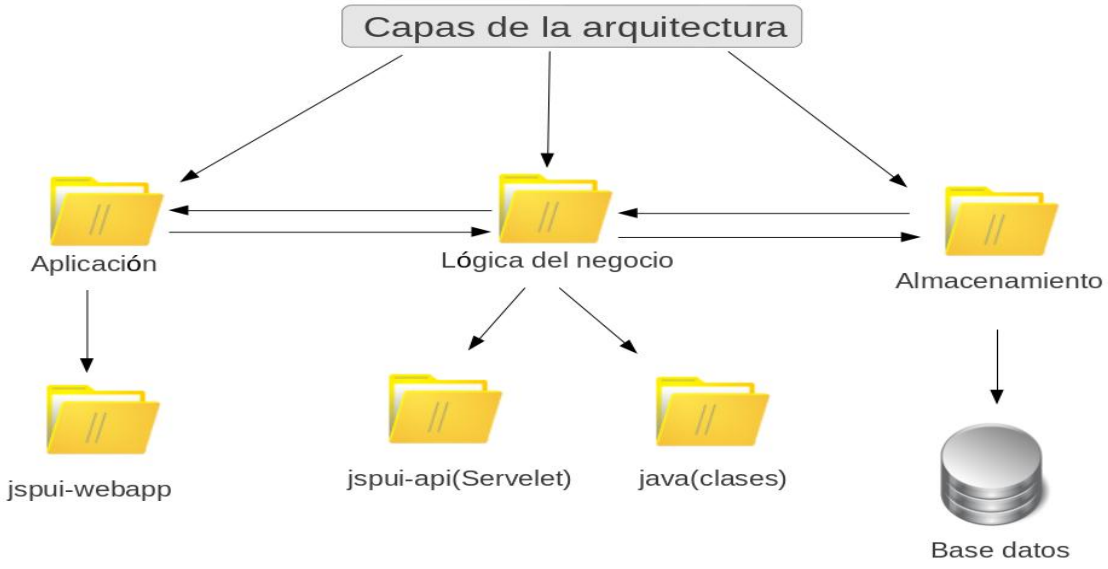

Figura 8: Capas que conforman la arquitectura del sistema

2.11.2. Patrones de diseño GRASP

Los patrones GRASP[12], en español Patrones Generales de Software para la Asignación de Responsabilidades) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. (34)

Patrón experto: Es el encargado de asignar una responsabilidad al experto en información, es decir la clase que cuenta con la información necesaria para cumplir con la misma. Esto permite que los sistemas tiendan a ser más fáciles de entender, mantener y ampliar, además presenta la oportunidad de reutilizar los componentes en futuras aplicaciones. (39) Este patrón está presente en la clase ManageTemplateServlet.java que es la encargada de gestionar las plantillas.

Patrón creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debe conectar con el objeto producido en cualquier evento. (39) Este patrón está presente en la clase ResearcherPrintServlet.java que es la encargada de exportar a formato pdf la página del autor.

Patrón controlador: La mayor parte de los sistemas reciben eventos de entrada externa, los cuales generalmente incluyen una interfaz gráfica para el usuario operado por una persona. La misma clase controlador debería utilizarse con todos los eventos sistémicos de un caso de uso, de modo que se puede conservar la información referente al estado del caso. Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. (39) Este patrón está presente en la clase *CreateTemplateServlet.java* que es la encargada de crear la plantilla.

Conclusión

Se definieron un total de siete requisitos funcionales, los cuales fueron agrupados en cinco Casos de Uso (CU). A partir de la obtención de los CU se elaboró el Diagrama de CU y se describieron textualmente para una mejor comprensión. También se definieron los requisitos no funcionales del módulo. Estos fueron elaborados teniendo en cuanta el software, el hardware, las restricciones en el diseño y la implementación, la confiabilidad y la seguridad del sistema.

[12] Por sus siglas en inglés General Responsibility Assignment Software Patterns

Se construyeron los artefactos fundamentales del flujo de trabajo análisis y diseño. Entre estos se encuentran los diagramas de clases del diseño y el diagrama de paquetes, los cuales mostraron una propuesta para el desarrollo del módulo. Además estos artefactos sirvieron de base para la implementación del módulo propuesto.

La utilización de patrones de diseño reconocidos, y el apoyo en el patrón arquitectónico tres capas, evidenció la construcción de un módulo legible y de fácil mantenimiento, elementos que favorecen el aporte tecnológico que representa este módulo para UCI.

Capítulo.3 Implementación y Prueba

Introducción

En este capítulo se abordará sobre los flujos de trabajo de Implementación y Prueba los cuales son determinantes en el proceso de desarrollo de software. Por parte de la implementación se elaboran los diagramas de componentes, despliegue así como la justificación de los estándares de codificación y los principales métodos. Mientras que por parte de las pruebas se define la estrategia de pruebas y se realizan las pruebas al módulo para obtener los resultados correspondientes.

3.1. Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, como ficheros de código fuente, ejecutables, entre otros. El modelo de implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en los lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. (18)

3.1.1. Diagrama de Componente

Los diagramas de componentes describen los elementos físicos del sistema y su relación, también muestran las dependencias lógicas entre componentes de software. El diagrama de componente forma parte de la vista física de un sistema, donde se modela la estructura de implementación de la aplicación, su organización en componentes y su despliegue en nodos de ejecución. La vista de implementación se representa con los diagramas de componentes. (40)

A continuación se presenta el diagrama de componente correspondiente al caso de uso Ver ranking de autores. Para consultar el resto de los diagramas de componentes **(Ver Anexo 5)**.

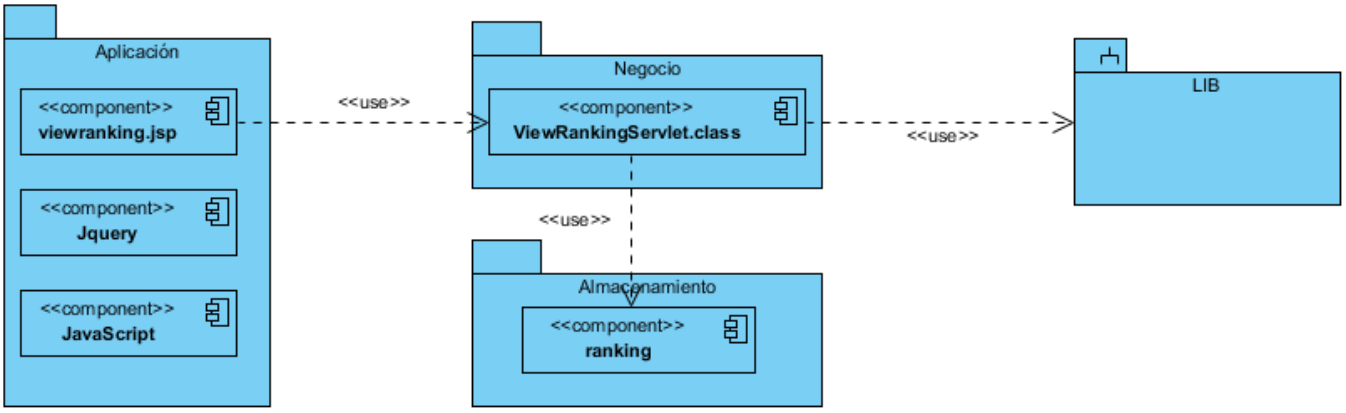

Figura 9: Diagrama de componentes Ver ranking de autores

La imagen muestra el diagrama de componentes correspondiente al caso de uso Ver ranking de autores. En la capa de aplicación contiene el componente **viewranking.jsp** que se conecta con el componente de la capa de negocio **ViewRanking.class**; además utiliza los componentes **Jquery** y **JavaScript** para validar los campos. El componente de la capa de negocio se relaciona con el componente **ranking** ubicado en la capa de almacenamiento además utiliza librerías para su correcto funcionamiento.

3.1.2. Diagrama de Despliegue

Un diagrama de despliegue muestra la configuración de los nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos. Es el encargado de la distribución, entrega, instalación de las partes que constituyen el sistema físico. (18)

A continuación se muestra el diagrama de despliegue del módulo:

.

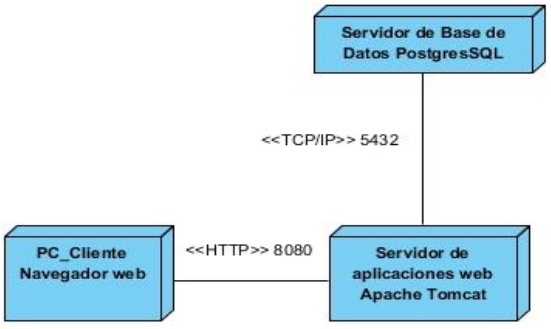

Figura 10: Diagrama de Despliegue

Componentes o nodos que interactúan en el despliegue del módulo:

- **PC_Cliente**: Representa las computadoras clientes que se conectan al servidor de aplicaciones mediante el protocolo HTTP por el puerto 8080.
- **Servidor de Aplicaciones Web**: Se encarga de atender las peticiones del usuario y proporcionar respuestas a este. Es el servidor donde se encuentra desarrollada la aplicación web. Este accede al servidor de Base Datos para el manejo de la información mediante el protocolo TCP/IP por el puerto 5432.
- **Servidor de Bases de Datos:** Se utiliza para gestionar datos del sistema REPXOS.

3.1.3. Justificación del estándar de codificación

Los estándares de codificación son términos que describen convenciones para escribir código fuente en ciertos lenguajes de programación, este es dependiente de la selección del lenguaje, de manera tal que facilita leer, escribir y mantener el código. Aplicar una estandarización al código, reduce el tiempo y el esfuerzo del entrenamiento de otros desarrolladores que trabajen con el módulo. Al usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. Además, si se aplica de forma continua un estándar de codificación bien definido, caben muchas posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener.

Clases Java

Las clases Java ya existentes en el sistema REPXOS contienen las siguientes secciones en este orden: comentarios de inicio, sentencia de paquete, sentencias de importación, declaraciones de clases e interfaces. El módulo debe adaptarse al mismo y aplicar esta estructura como estándar.

Organización de los Ficheros

Cada fichero fuente Java contiene una única clase o interface pública. Cuando algunas clases o interfaces privadas están asociadas a una clase pública, pueden ponerse en el mismo fichero que la clase pública. La clase o interfaz pública debe ser la primera clase o interface del fichero.

Los ficheros fuentes Java tienen la siguiente ordenación:

- Comentarios de comienzo.
- Sentencias package e import.

- Declaraciones de clases e interfaces.

Comentarios de comienzo

A continuación se muestra un comentario de inicio de la clase *CreateTemplateServlet.java.*

```
/**
 * The contents of this file are subject to the license and copyright detailed
 * in the LICENSE and NOTICE files at the root of the source tree and available
 * online at
 *
 * http://www.dspace.org/license/
 */
```

Figura 11: Comentarios iniciales de la clase CreateTemplateServlet

Sentencias package e import

La primera línea no comentada de un fichero fuente debe ser la sentencia de paquete, que indica el paquete al que pertenecen las clases incluidas en el fichero fuente. Tras la declaración del paquete se incluirán las sentencias de importación de los paquetes necesarios. A continuación se muestra un ejemplo de la clase *CreateTemplateServlet.java*.

```
package org.dspace.app.webui.servlet.admin;
import java.io.IOException;
import java.sql.SQLException;
```

Figura 12: Sentencias package e import de la clase CreateTemplateServlet

Declaraciones de clases e interfaces

Al codificar clases e interfaces de Java, se siguen las siguientes reglas de formato:

- Ningún espacio en blanco entre el nombre de un método y el paréntesis ***"("*** que abre su lista de parámetros.
- La llave de apertura **"{"** aparece al final de la misma línea de la sentencia declaración.
- La llave de cierre **"}"** empieza una nueva línea indentada para ajustarse a su sentencia de apertura correspondiente.

A continuación se muestra un ejemplo en la clase *CreateTemplateServlet.java.*

```
protected void doDSGet(Context context, HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException,
        SQLException, AuthorizeException {
    // Simply forward to the plain form
    JSPManager.showJSP(request, response, "/researcher/createtemplate.jsp");
}
```

Figura 13: Declaración de un método de la clase CreateTemplateServlet

Declaración de variables

Los nombres de las variables deben ser cortos pero con significado. Los nombres de variables de un solo caracter se deben evitar, excepto para variables índices temporales. Nombres comunes para variables temporales son i, j, k, m, y n para enteros; c, d, y e para caracteres.

```
String author_name = request.getParameter("author_name");
String fecha_inicio = request.getParameter("fecha_inicio");
String fecha_fin = request.getParameter("fecha_fin");
String[] valsearch = request.getParameterValues("checkbox_search");
```

Figura 14: Declaración de variables de la clase ResearcherServlet

Sentencias for

Las sentencias **for** tienen la siguiente forma.

```
for (int i = 0; i < val.length; i++) {
    String paragraph = final_list.get(Integer.parseInt(val[i]));
    document.add(new Paragraph(paragraph));
}
```

Figura 15: Sentencia for de la clase ResearcherServlet

Las sentencias **while** tienen la siguiente forma, comienzan con la palabra reservada **while** seguido por una condición entre paréntesis y posteriormente las sentencias. A continuación se muestra un ejemplo.

```
while (row2 != null) {
    metadata_field_id.add(row2.getIntColumn("metadata_field_id"));
    text_value.add(row2.getStringColumn("text_value"));
    row2 = tri2.next();
}
```

Figura 16 Sentencia while de la clase ResearcherServlet

Las sentencias **if-else** y las sentencias **try-catch** tienen las siguiente forma.

```
if (!fecha.isEmpty()) {
    try {
        date = new java.sql.Date(Fdate.parse(fecha).getTime());
    } catch (ParseException ex) {
        java.util.logging.Logger.getLogger(ResearcherServlet.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Figura 17 Sentencias if-else y try de la clase ResearcherServlet

Clases JSP

Directiva(s) JSP page

Una directiva Page define atributos asociados con la página JSP en tiempo de traducción. La especificación JSP no impone ninguna obligación sobre cuántas directivas Page se pueden definir en la misma página. Ejemplos de estas son:

```
<%@ page import="org.dspace.core.I18nUtil" %>
<%@ page import="org.dspace.core.Utils" %>
```

Figura 18 Directivas JSP page

Directiva(s) tag library opcionales

Una directiva *taglib* declara las librerías de etiquetas usadas por el JSP, esta puede ser declarada corta en una sola línea. Si se tienen varias directivas *taglib* se deben almacenar juntas en la misma localización dentro del cuerpo JSP. Al igual que la directiva Page, si la longitud de una directiva *taglib* excede la anchura de 80 caracteres, debemos dividirla en varias líneas.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<%@ taglib uri="http://www.dspace.org/dspace-tags.tld" prefix="dspace" %>
```

Figura 19: Directivas tag library opcionales

3.2. Pruebas

Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto de software. Son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un sistema informático. Básicamente es una fase en el desarrollo de software, consistente en probar las aplicaciones construidas. Para determinar el nivel de calidad se deben

efectuar unas medidas o pruebas que permitan comprobar el grado de cumplimiento respecto a las especificaciones iniciales del sistema. (41) Las pruebas son aplicadas para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo, agrupadas por niveles de prueba aplicada por etapas. (18)

Se distinguen los siguientes niveles de pruebas:

- **Prueba de desarrollador**: Es la prueba diseñada e implementada por el equipo de desarrollo.
- **Prueba independiente**: Es la prueba que es diseñada e implementada por alguien independiente del grupo de desarrolladores.
- **Prueba de Unidad**: Se centra en el esfuerzo de verificación de la unidad más pequeña del diseño del software, el componente o sistema de software. El objetivo es comprobar que el sistema, entendido como una unidad funcional, está correctamente codificado.
- **Prueba de Integración**: Es una técnica sistemática para construir la arquitectura del software, mientras, al mismo tiempo, se aplican las pruebas para descubrir errores asociados con la interfaz. Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso.
- **Prueba de sistema:** Son las pruebas que se hacen cuando el software está funcionando como un todo. Es la actividad de prueba dirigida a verificar el programa final, después que todos los componentes de software y hardware han sido integrados.
- **Prueba de aceptación:** Es la prueba final antes del despliegue del sistema. El propósito es confirmar que el sistema está terminado, que desarrolla puntualmente las necesidades de la organización y que es aceptado por los usuarios finales.

Dentro de los niveles de pruebas mencionados anteriormente se decide utilizar el nivel de prueba de aceptación. En este nivel de prueba se pudo verificar que el software está listo. Además se utiliza el nivel pruebas del sistema.

Tipos de pruebas

Cada nivel de prueba engloba una técnica de prueba específica según los atributos de calidad que se deseen verificar con las pruebas al software. Entre las técnicas de pruebas que se realizan en el sistema está la que evalúa la funcionalidad de éste. En esta investigación se realiza el tipo de pruebas funcionales y de regresión.

- **Función**: Las pruebas de función fijan su atención en la validación de las funciones, métodos, servicios y casos de usos. Permite comprobar el correcto funcionamiento de los requisitos funcionales del módulo.

- **Regresión**: Las pruebas de regresión son utilizadas para disminuir los efectos colaterales. Se aplican cada vez que el software se le hace algún cambio. Cada vez que el software se corrige cambia. La prueba de regresión se aplica manualmente al ejecutar nuevamente todos los casos de prueba. Fueron utilizadas en una segunda iteración realizada a los casos de prueba diseñados.

Método de prueba

Para la realización de las pruebas a la aplicación se utiliza el método de caja negra. El método de prueba de caja negra se aplica a la interfaz de la aplicación. Pretende demostrar que las funcionalidades del sistema son operativas, las entradas se aceptan correctamente y que se producen los resultados esperados.

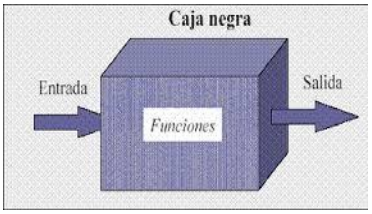

Figura 20: Pruebas de caja negra

Dentro del método de caja negra se utiliza la técnica “Partición Equivalente” siendo considerada como una de las más efectivas en la evaluación de los valores válidos, inválidos y los que no es necesario proporcionar un valor del dato en las entradas existentes en la aplicación. La Partición Equivalente divide el dominio de entrada de un programa en clases de datos a partir de las cuales se derivan casos de prueba. (32)

3.2.1. Resultados de las pruebas

Luego de haber realizado las pruebas mencionadas anteriormente se obtuvieron los siguientes resultados:

No.Iteración	Aplicación			Casos de Prueba	Resueltas	No procede	Total NC
	Alta	Media	Baja				
1ra Iteración	1	1	1	4	7	0	7
2da Iteración	0	0	0	0	0	0	0

Tabla 4 Resultados de las pruebas

Como se muestra en la tabla las no conformidades (NC) son aplicadas tanto a los Casos de Prueba (CP) como a la aplicación. En la parte de la aplicación fueron detectadas tres NC estas están clasificadas en una alta, una media y una baja. Por parte de los CP solo cuatro y todas fueron resueltas. Posteriormente se procede a realizar las pruebas de Regresión en la segunda iteración. En la segunda iteración no fueron detectadas NC. Es válido destacar que las NC detectadas fueron resueltas.

Las no conformidades detectadas fueron:

Aplicación

- El caso de prueba describía un mensaje que la aplicación no mostraba en la interfaz de modificar plantilla.
- El mensaje *Plantilla actualizada* presentaba falta de ortografía.
- Al seleccionar una fecha de inicio mayor que la fecha de fin no está validado el campo en la interfaz Ver listado de publicaciones.

Casos de Prueba

- Los números que identifican a los escenarios no seguían un orden ascendente en CP Gestionar plantilla.
- Fueron detectados algunos errores ortográficos en la descripción de los casos de prueba.
- En la descripción de variable *nombre de la plantilla* se especifica que admite solo letras, pero además admite números, por lo que debería especificarse que admite caracteres alfanuméricos.

- La descripción de un escenario no está en correspondencia con su nombre. El escenario 2.4 del CP Gestionar plantilla debería llamarse *Nombre de la plantilla ya existe*.

Conclusiones

Luego de desarrollado el capítulo se representaron los elementos necesarios para la comprensión de como el módulo fue implementado, elaborándose los artefactos más importantes correspondientes al flujo de trabajo de implementación. Se realizaron los tipos de pruebas de regresión y funcionalidad mediante el método de caja negra, en ambas pruebas se realizaron dos iteraciones. En la primera iteración se detectaron varias no conformidades las cuales fueron resueltas, demostrando la calidad de solución que se propone y el cumplimiento con los requisitos funcionales definidos.

CONCLUSIONES

Conclusiones

La presente investigación se centró en el objetivo general, el cual se basó en desarrollar un Módulo para el sistema de repositorios digitales REPXOS 3.0 que permita medir la producción científica de un investigador en la UCI. Durante el avance del mismo se cumplieron los objetivos propuestos y se arribó a los siguientes resultados:

- El estudio de los diferentes sistemas homólogos permitió obtener los conocimientos necesarios para una mayor comprensión del objeto de estudio y para sentar las bases de la investigación.
- Se diseñó un módulo para medir la calidad de las producciones científicas en el repositorio institucional, mediante el cual se generaron los artefactos más significativos propuestos por la metodología de desarrollo RUP.
- Se implementaron todas las funcionalidades definidas para el desarrollo del módulo, utilizando para ello las herramientas y lenguajes definidos.

.

.

Recomendaciones

- Se recomienda para versiones posteriores que se le agregue un parámetro nuevo al ranking utilizando para ello las visitas que recibe una publicación almacenada en el repositorio.

- Incorporar el proceso de generar el ranking a los scripts del sistema.

Referencias

1. Revista española de Documentación Científica. [En línea] 2012. [Citado el: 15 de 03 de 2015.] http://redc.revistas.csic.es/index.php/redc/article/view/773/862.

2. **Bergstrom, Carl T y Bergstrom, Theodore C.** *The costs and benefits of library site licenses to academic journals.* 2004.

3. *Estudio bibliométrico de los modelos de medición del concepto de calidad percibida del servicio en Internet.* **Duque Oliva, Edison Jair, Rodríguez Romero, Carlos y Cervera, Amparo.** Colombia : s.n.

4. Universidad Tecnológica de Pereira. [En línea] [Citado el: 1 de 06 de 2015.] http://repositorio.utp.edu.co/dspace/bitstream/11059/1008/1/025072C157.pdf.

5. **Pedagógicas, Universidad de las Ciencias.** Pedagogia y Sociedad. [En línea] [Citado el: 15 de 03 de 2015.] http://www.pedsoc.rimed.cu/FTP/articulos%20pdf/A%C3%B1o%2015/no_34_julio_2012/informate/Bases_Pol%C3%ADtica_de_Publicaciones_del_MES._niveles.pdf.

6. Universidad de León. [En línea] 01 de 2009. [Citado el: 25 de 03 de 2015.] http://biblioteca.unileon.es/documentos/guia_factorimpacto.pdf.

7. Biblioteca de la Universidad de Extremadura. [En línea] [Citado el: 6 de 06 de 2015.] http://biblioguias.unex.es/content.php?pid=408804&sid=3344542.

8. *Theory and practise of the g-index.* **Egghe, Leo.** 1, Budapest : Scientometrics, 2006, Vol. 69. 131-152.

9. **BiblioSaúde.** BiblioSaúde. [Online] [Cited: 03 15, 2015.] http://bibliosaude.sergas.es/DXerais/481/COMO%20CALCULAR%20EL%20CUARTIL%20DE%20UNA%20REVISTA%20CIENT%C3%8DFICA.%20isi%20wEB%20OF%20kNOWLEDGE.pdf.

10. Journal Metrics. [Online] [Cited: 03 27, 2015.] http://www.journalmetrics.com/snip.php.

11. *Nuevos indicadores métricos para la evaluación de las publicaciones seriadas científicas y académicas.* **Cañedo Andalia, Rubén and Cruz Font, Jaime.** 1, Ciudad de la Habana : s.n., 2012, Vol. 23. 1024-9435.

12. eigenfactor. [En línea] 2012. [Citado el: 30 de 03 de 2015.] http://www.eigenfactor.org/.

13. Universidad de las Palmas de Gran Canaria. [En línea] 19 de 03 de 2015. [Citado el: 30 de 03 de 2015.] https://biblioteca.ulpgc.es/valoracion_revistas.

14. **Scimago Lab.** SCImago Journal & Country Rank. [En línea] [Citado el: 25 de 03 de 2015.] http://www.scimagojr.com/.

15. **Leiden University.** CWTS Journal Indicators. [En línea] 2014. [Citado el: 28 de 03 de 2015.] www.journalindicators.com.

16. Sistemas de Indicadores de Ciencia,Tecnología e Innovación. [En línea] Universidad de las Ciencias Informaticas. [Citado el: 25 de 03 de 2015.] https://investigaciones.uci.cu/SIndiCIT/index.php#.

17. **Ortiz Batista, Yunaysy and Gulín González, Jorge .** *Sistema integrado de gestión de Ciencia, Tecnología e Innovación en la Universidad de las Ciencias Informáticas.* Habana : s.n., 2011.

18. **Ivar, Jacobson, Grady Booch and Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* 2000. ISBN 84-7829-036-2.

19. **tomcat.apache.** Apache Tomcat. [En línea] 2014. [Citado el: 10 de 04 de 2014.] http://tomcat.apache.org/.

20. **dspace.org.** Dspace. [En línea] 2014. [Citado el: 10 de 03 de 2014.] http://www.dspace.org/introducing.

21. **Ant.Apache.** Apache ant site. [En línea] 2014. [Citado el: 12 de 03 de 2014.] http://ant.apache.org/.

22. **Sosa Vázquez, Yailis y Ruiz Cardoso, Alberto.** *Análisis y diseño de los procesos de Agregación, Notificaciones y Ajustes al Plan del subsistema de Planificación perteneciente al Sistema Integral de Gestión de Entidades Cedrux.* La habana : s.n., 2010.

23. **Visual-paradigm.** Visual-paradigm. [En línea] 2014. [Citado el: 16 de 03 de 2014.] http://www.visual-paradigm.com.

24. **Date, C J.** *Introducción a los sistemas de base de datos.* Mexico : s.n., 2001. ISBN/968-444-419-2.

25. **Postgresql.** postgresql site. [En línea] 2014. [Citado el: 13 de 03 de 2014.] http://www.postgresql.org.es/sobre_postgresql.

26. **Netbeans.** netbeans site. [En línea] 2014. [Citado el: 15 de 01 de 2014.] https://netbeans.org/index_es.html.

27. **Maven.** Maven site. [En línea] 2014. [Citado el: 15 de 1 de 2014.] http://maven.apache.org/what-is-maven.html.

28. **GROUP, ITEXT.** ITEXT. [Online] 2014. [Cited: 02 9, 2014.] http://itextpdf.com/.

29. **Cornejo, J EGonzález.** Document Information Retrieval. [Online] 2 17, 2012. [Cited: 1 20, 2014.] http://www.docirs.cl/uml.htm.

30. **Java.** Java. [En línea] 2013. [Citado el: 10 de 01 de 2014.] http://www.java.com/es/about/.

31. **Castro, Lilian Teresa Mecías y Bermúdez, Rolando Peña.** *Desarrollo de una Librería de Componentes JavaScript basado en Dojo Toolkit.* La Habana : s.n., 2008.

32. **Pressman, Roger S.** *Ingeniería de Software, Un enforque práctico.* Séptima. New York : McGraw-Hill, 2010. ISBN 978-0-07-337 597 -7.

33. **Sommerville, Ian.** *Software Engineering.* 2006. ISBN 0-321-31379-8.

34. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objeto.* Mexico : Prentice Hall,, 1999. ISBN/970-17-0261-1..

35. Universidad de Cantabria. [En línea] 2011. [Citado el: 22 de 04 de 2015.] http://personales.unican.es/zorrillm/BasesDatos/02%20-%20Modelos%20de%20datos%20ER-UML-relacional.pdf.

36. Entorno Virtual de Aprendizaje de la Universidad de las Ciencias. [Online] [Cited: 04 22, 2015.] http://eva.uci.cu/file.php/180/2._Clases/Tema_1/Materiales_basicos/2.Modelo_Entidad_Relacion_y_extensiones.pdf.

37. **Wesle, Addison, Gamma, Erich y Helm, Richard.** *Design Patterns. Elements of Reusable Object-Oriented Software.* 1995. ISBN 0-201-63361-2.

38. **Developer, Team The DSpace.** [Online] 02 24, 2012. [Cited: 02 8, 2014.] https://wiki.duraspace.org/display/DSDOC18/Architecture.

39. **Hernán, Astudillo Marcello Visconti y.** *Fundamentos de Ingeniería de Software.* s.l. : Departamento de Informática de la Universidad Técnica Federico Santa Maria, 2009.

40. **Garcia, Y y PÉREZ, D.** *Sistema de Gestión de la Trayectoria Productiva de cada estudiante en la facultad 1.* 2009.

41. **Bruceta , Yaniel y Perez, Sandy.** *Despliegue Jerárquico de Servidores de XILEMA GRHS.* 2014.

42. **Biblioteca de la Universidad de Extremadura.** Biblioteca de la Universidad de Extremadura. [Online] [Cited: 03 10, 2015.] http://biblioteca.unex.es/aprendizaje-e-investigacion/investigacion/evaluacion-impacto/434.html#descripcion.