Vol. 19, n.° 2, pp. 11-20, 2020
Revista UIS Ingenierías
Página de la revista: revistas.uis.edu.co/index.php/revistauisingenierias

Implementación de un repositorio para el catálogo, búsqueda y uso de componentes software reutilizables en el desarrollo de aplicaciones web

Implementation of a repository for cataloguing, searching and using reusable software components in web application development

Jhon Carlos Vargas-Fandiño[1a], Jhon Jairo Sandoval-Ramirez [1b], Fredy Humberto Vera-Rivera [1c, 2, 3]

[1]Grupo de Investigación en Inteligencia Artificial (GIA), Programa de Ingeniería de Sistemas, Universidad Francisco de Paula Santander, Colombia. Orcid: [c] 0000-0003-4003-497X. Correo electrónico: [a] jhonlavigne07@gmail.com, [b] jhon.jairo.sandoval.ramirez@gmail.com, [c] fredyhumbertovera@ufps.edu.co
[2] Grupo de Estudios Doctorales en Informática (GEDI), Escuela de Ingeniería de Sistemas y Computación, Universidad del Valle, Colombia. Correo electrónico: fredy.vera@correounivalle.edu.co
[3] Foundation of Researchers in Technology of Materials (Foristom), Colombia.
Correo electrónico: freve9@gmail.com

Recibido: 13 junio, 2019. Aceptado: 20 enero, 2020. Versión final: 5 marzo, 2020.

Resumen

El desarrollo de software se caracteriza por ser un proceso complejo, requiere de inversión de tiempo, conocimiento de herramientas tecnológicas para su elaboración, su depuración y su despliegue. Sin embargo, con el paso de los años, se han desarrollado avances tecnológicos y metodológicos, que reducen considerablemente esta complejidad, convirtiendo dicha actividad en un proceso más intuitivo, controlable y rápido. La reutilización aparece como una alternativa para desarrollar aplicaciones y sistemas de una manera más eficiente y rápida. La idea es reutilizar elementos y componentes en lugar de tener que desarrollarlos desde un principio. A estas unidades se les conoce como componentes software reutilizables. La biblioteca o repositorio de componentes reutilizables es el corazón del desarrollo de software basado en componentes, permite catalogar, organizar, descubrir y reutilizar estos componentes. En este trabajo se desarrolló e implementó un repositorio de componentes de software reutilizables que apoye los procesos de desarrollo de aplicaciones web y la formación de estudiantes en el programa de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander (Cúcuta, Colombia) llamada Colossal, permitiendo a los estudiantes y profesores buscar, compartir y reutilizar los componentes disponibles en el repositorio y usarlos en sus proyectos. Para evaluar el repositorio de componentes y verificar que cumple con su propósito se realizó un estudio cuantitativo descriptivo donde se evalúa el nivel de aceptación de la herramienta tecnológica, como resultado se obtuvo que la mayoría (94%) de los estudiantes y profesores respondió de manera positiva y se evidenció una buena aceptación del funcionamiento y uso del mismo.

Palabras clave: reutilización de software; repositorio de componentes software; Colossal; componentes software, ingeniería del software basada en componentes.

Abstract

Software development is a complex process that requires time investment and knowledge of technological tools for its elaboration, debugging and deployment. However, over the years, technological and methodological advances have

ISSN impreso: 1657 - 4583. ISSN en línea: 2145 - 8456, **CC BY-ND 4.0**
J. C. Vargas-Fandiño, J. J. Sandoval-Ramírez, F. H. Vera-Rivera, "Implementación de un repositorio para el catálogo, búsqueda y uso de componentes software reutilizables en el desarrollo de aplicaciones web," *Rev. UIS Ing.*, vol. 19, no. 2, pp. 11-20, 2020. doi: 10.18273/revuin.v19n2-2020002

been developed, which considerably reduce this complexity, making this activity a more intuitive, controllable and fast process. Reuse appears as an alternative to develop applications and systems more efficiently and quickly. The idea is to reuse elements and components instead of having to develop them from scratch. These units are known as reusable software components. The library or repository of reusable components is the heart of component-based software development, allowing these components to be catalogued, organised, discovered and reused. In this work, a reusable software component repository was developed and implemented to support the web application development processes and the training of students in the Systems Engineering program of the Universidad Francisco de Paula Santander (Cúcuta, Colombia) called Colossal, allowing students and professors to search, share and reuse the components available in the library and use them in their projects. In order to evaluate the component repository and verify that it fulfills its purpose, a quantitative descriptive study was carried out to evaluate the level of acceptance of the technological tool. As a result, it was obtained that the majority (94%) of students and professors responded positively and a high acceptance of the operation and use of the tool was evidenced.

Keywords: software reuse; software components repository; Colossal; software components, component-based software engineering.

1. Introducción

La reutilización de componentes software permite agilizar los procesos de producción y despliegue de las aplicaciones web. Dichos componentes han sido debidamente probados y verificados con anterioridad, lo que reduce los tiempos de codificación ayudando a crear sistemas informáticos más confiables y seguros [1], permitiendo a los desarrolladores construir nuevos sistemas realizando la búsqueda de componentes en un repositorio, combinándolos y adaptándolos en vez de codificar y construir las mismas aplicaciones una y otra vez. Un componente de software es una unidad que cuenta con una interfaz debidamente especificada que cumple con una función y puede ser implementado por terceros para dar lugar a nuevos sistemas o componentes [2].

La Ingeniería de Software Basada en Componentes (ISBC o CBSE), es una rama de la ingeniera del software que se encarga de la construcción de un sistema a partir de componentes reutilizables. Cuenta con dos áreas fundamentales, la primera, la ingeniería de dominio, que consiste en la creación de cada componente y su catalogación en el repositorio de componentes; cada uno de los componentes deben ser probados y validados para su publicación y el desarrollador los puede reutilizar en sus aplicaciones; y la segunda, el desarrollo basado en componentes, el cual enmarca las actividades que se deben realizar para construir un sistema a partir de esos componentes ya creados, entre estas actividades se encuentran la identificación, la búsqueda, la composición, la adaptación y las pruebas de los componentes seleccionados y de la aplicación que se esté desarrollando [3], [4].

Este trabajo abarca la primera de estas dos actividades; el desarrollo e implementación de "Colossal" un repositorio de componentes software reutilizables que apoye los procesos de construcción de aplicaciones web, permitiendo a los desarrolladores, estudiantes y profesores de la universidad buscar, compartir y reutilizar los componentes que se encuentren ahí disponibles y usarlos en sus propios desarrollos y proyectos.

La evolución de los componentes software se ha dado a largo de los años a través de la creación de unidades de software que son reutilizadas en la creación de diferentes aplicaciones, desde clases, paquetes, librerías, Apis, módulos, servicios, microservicios y funciones como servicio. La arquitectura de aplicaciones basadas en microservicios actualmente es un área de mucho interés, es un tema actual de investigación y se viene utilizando en muchas áreas tecnológicas, por ejemplo, el internet de las cosas (IoT), la computación en la nube, las telecomunicaciones y los vehículos autónomos; además están siendo utilizados por las empresas tecnológicas más importantes como son Google, Amazon, Airbnb, Netflix, Spotify y otras. Los microservicios son unidades de software independientes, desplegadas en su propio proceso, que se unen a través de una interfaz bien definida [5], por lo tanto, pueden ser considerados como componentes software reutilizables y ser catalogados por Colossal para su posterior reutilización [6].

Por otro lado, en un equipo de desarrollo es importante conocer y poder usar el código y funcionalidades que ya han sido desarrolladas con el fin de ahorrar esfuerzos y reducir el tiempo de desarrollo, no reinventar la rueda en cada desarrollo nuevo, esa es la idea fundamental del desarrollo basado en componentes. Colossal es un repositorio de componentes software que reúne, cataloga y permite reutilizar cualquier clase de componentes software, desde clases y paquetes hasta microservicios.

Actualmente se ha desarrollado algunos repositorios de código, por ejemplo, GitHub es uno de los más importantes y usados a nivel mundial; GitHub permite

alojar y revisar código, gestionar proyectos y construir software junto a otros desarrolladores de forma colaborativa [7]. GitHub no cataloga ni permite el uso de componentes ya implementados, sólo permite llevar el control de versiones del código fuente de las aplicaciones. La idea fundamental de Colossal es guardar y catalogar el componente listo para ser utilizado, de tal forma que el desarrollador lo pueda incorporar fácilmente en su aplicación. Colossal no es una herramienta de control de versiones ni de desarrollo de código colaborativo.

A continuación, en la sección 2 se presenta la metodología de este trabajo, luego en la sección 3 se presenta Colossal, en la sección 4 se presentan y discuten los resultados obtenidos, por último, en la sección 5 se dan las conclusiones.

2. Metodología

Este trabajo de investigación se fundamenta en las metodologias ágiles para el desarrollo de software y en la investigación tecnológica aplicada, que busca la construcción de productos software de forma iterativa e incremental. La complejidad que implica el desarrollo de software ha sido abordada con el uso de las metodologías ágiles y las prácticas ágiles que nacieron a partir del manifiesto ágil y sus principios en contraste a las formas tradicionales [6]. Los beneficios más importantes que se obtienen con el uso y adopción de las prácticas ágiles son: el incremento de la productividad del equipo de desarrollo, se mejora la visibilidad del producto software, se adquiere la habilidad para manejar y controlar el cambio en los requerimientos y en las prioridades del proyecto y se hacen entregas más rápidas del proyecto [8], [9].

En la figura 1 se presenta el modelo de investigación llevado a cabo en este trabajo. Primero se realizó la fundamentación en el desarrollo de aplicaciones basadas en componentes, en las características de los repositorios y/o bibliotecas de componentes reutilizables para luego realizar la planificación de Colossal. Segundo se definió la arquitectura del repositorio, se basó en el modelo – vista – controlador y se definió el modelo de datos. Tercero, se realizó la codificiación y desplieuge del repositorio de componentes Colossal y por último se realizó una evaluación por parte de estudiantes y docentes del programa de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander para verificar su funcionamiento y determinar su grado de aceptación.

Para el desarrollo de Colossal se aplicó la metodología ágil XP por sus características [10]:

- Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.
- Se aplica de manera dinámica durante el ciclo de vida del software.
- Es capaz de adaptarse a los cambios de requisitos.
- Los individuos e interacciones son más importantes que los procesos y herramientas.

Colossal se desarrolló de forma iterativa en 4 fases: (1) Planeación, (2) Diseño, (3) Codificación y (4) pruebas. Luego al terminar la implementación se realizó una evaluación con un grupo de estudiantes y profesores que probaron el repositorio y respondieron un cuestionario.

El desarrollo se dividió en tres iteraciones, las cuales se fueron adaptando a medida que el desarrollo avanzaba. En la iteración (1) se incluyeron las historias de usuario: Consulta de componentes, Ver Detalles de Componentes, Subir Componentes y Gestionar peticiones de subida de componentes. En la iteración (2): Actualizar Componentes, Gestionar Contenido del repositorio y Añadir Componentes a listas personalizadas; y en la iteración (3) Reporte de anomalias de los componentes y generación de reportes.

Una de las principales ventajas de trabajar con la metodología XP es la posibilidad de adaptar el proyecto a los cambios y requerimientos que surjan conforme este avanza.

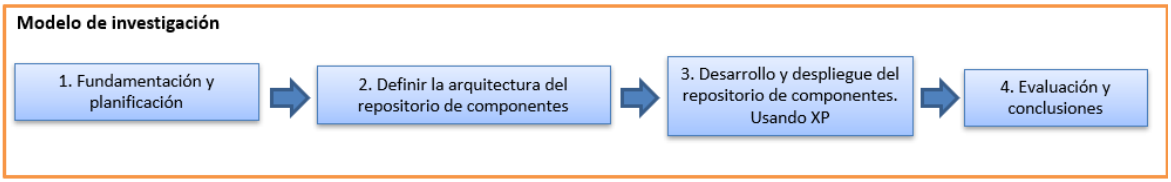

Figura 1. Modelo de investigación. Fuente: elaboración propia.

A continuación se describen las principales funcionalidades de Colossal.

3. Colossal – Repositorio de componentes software reutilizables

Colossal se inició como una propuesta ante la necesidad de fortalecer los procesos académicos del programa de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander (UFPS), tomando como soporte las ventajas que puede ofrecer la ingenieria del software basada en componentes, tanto en entornos empresariales como académicos. En la universidad industrial de Santander - UIS, por ejemplo, se llevó a cabo el desarrollo de un repositorio de componentes que dio como resultado el fortalecimiento de los procesos de desarrollo de software empresarial en la División de Servicios de Información, logrando organizar y reutilizar el software que allí se construye [11].
Ante la ausencia de herramientas propias que fomenten y fortalezcan la reutilización de código en el desarrollo de aplicaicones web en la UFPS, se planteó el desarrollo de un repositorio de componentes de software reutilizables para su implementación como parte del plan de mejora de la plataforma Sandbox - UFPS, el cual es una plataforma como servicio (PaaS) que ofrece herramientas para el despliegue de aplicaciones y servicios de red que dan soporte al desarrollo de aplicaciones web y a las prácticas de las asignaturas de la carrera [12].

Durante la fase de planificación se definieron los roles del sistema, los cuales son: 1) el administrador (se encarga de aprobar los componentes compartidos y configurar el repositorio), 2) el usuario estandar (estudiante o docente visitante del repositorio quien consulta, descarga y usa los componentes); posteriormente se definieron las historias de usuario y funcionalidades que deben contener, las cuales se pueden apreciar en los diagramas de casos de uso mostrado en la figura 2 y 3.

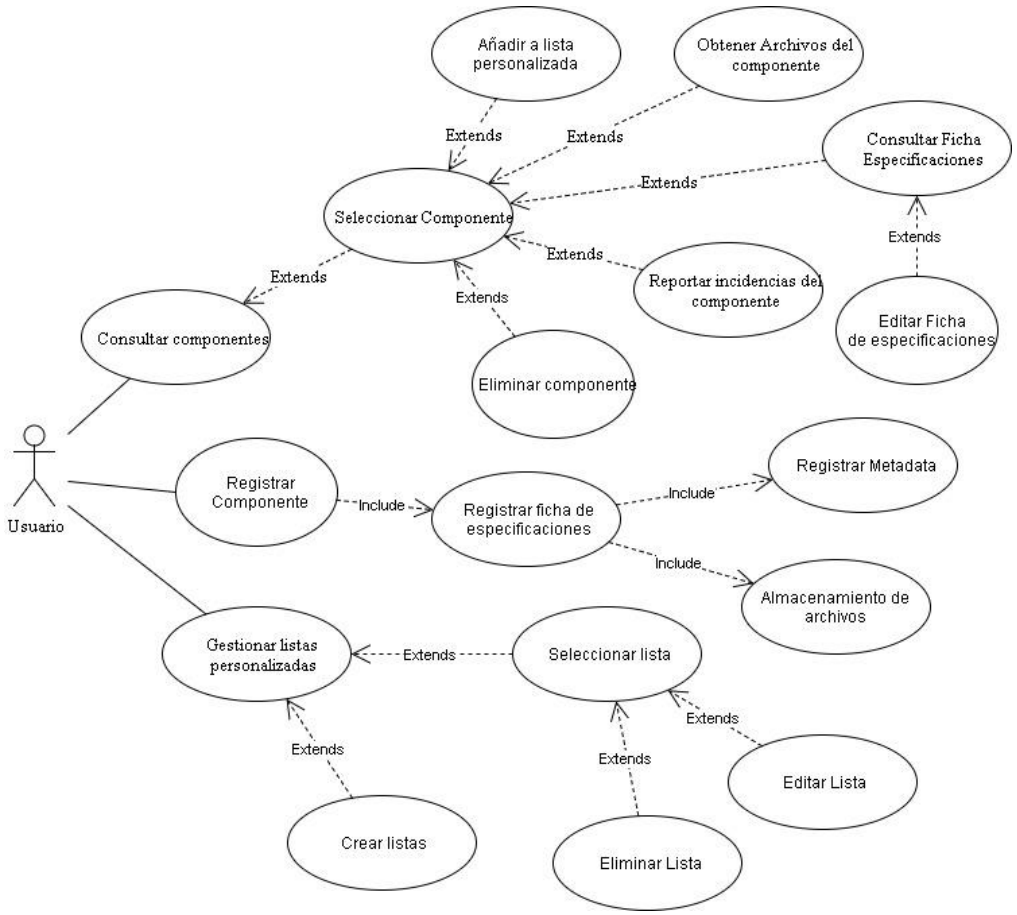

Figura 2. Funcionalidades, casos de uso para el usuario estándar. Fuente: adaptado de [11].

Las principales funcionalidades de Colossal se describen a continuacion:

- Consultar componentes: estos pueden ser consultados por varios criterios de búsqueda, (por tags, categorías, nombre, descripción, servicios ofrecidos y las tecnologías empleadas).

  Cuando el usuario estándar selecciona un componente este puede acceder a revisar la ficha de especificaciones, así como descargar los archivos adjuntos y añadirlo a una lista personalizada. Si el usuario detecta fallos o alguna anomalía puede generar un reporte de incidencias para que un administrador lo revise. En caso de ser el autor, este podrá actualizar la información correspondiente a la ficha de especificaciones y en caso de desearlo eliminar el componente.

- El usuario puede registrar componentes para lo cual debe ingresar los datos correspondientes a la ficha de especificaciones (Registro de metadata, y los archivos que correspondiente al componente), al finalizar la operación se genera una petición de subida que será revisada y evaluada por un administrador. Se ofrece la posibilidad de gestionar listas de componentes, las cuales el usuario puede crear, editar y eliminar a su gusto, estas listas existen con el fin de brindar la comodidad de acceder rápidamente a los componentes o guardarlos para posteriores usos.

El rol de administrador cuenta con todas las funcionalidades anteriormente mencionadas, no obstante, es el encargado de administrar y gestionar el contenido de la plataforma. El usuario administrador adicionalmete cuenta con las siguientes funcionalidades dentro del sistema, estas funcionalidades se pueden apreciar en la figura 3.

- Gestión de peticiones de subida: el administrador consulta una serie de peticiones de subida que son generados por los usuarios cuando se registra un componente nuevo o se actualiza uno que ha sido rechazado, posteriormente estos determinan si el componente cumple con los criterios apropiados para ser aprobados y por ende desplegados dentro del repositorio, en caso contrario serán rechazados. también puede eliminar o recuperar componentes según sea necesario.

- Gestión de incidencias: el administrador recibe una serie de incidencias relacionadas a los componentes que son reportadas por los distintos usuarios sobre un componente, estos evaluaran la situación según los criterios establecidos.

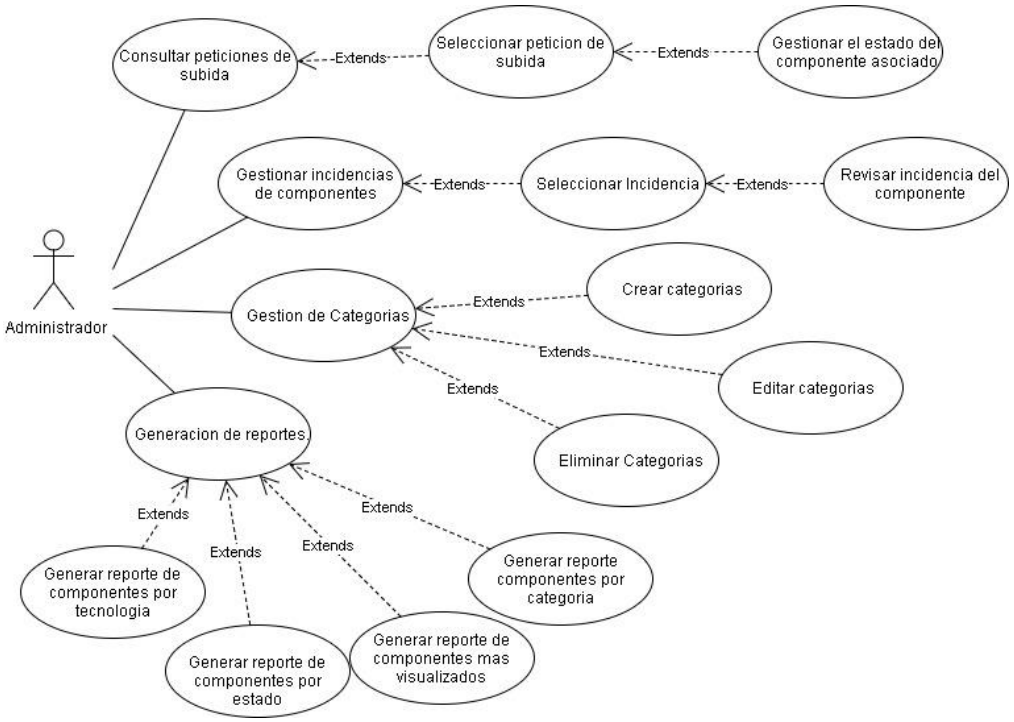

Figura 3. Funcionalidades, casos de uso para el Administrador. Fuente: Adaptado de [11].

- Gestión de categorías: Las categorías son creadas, editadas o eliminadas por los administradores, estas formaran parte de la ficha técnica de los componentes y establecen una clasificación y criterio de búsqueda dentro del repositorio.

- Generación de reportes: el administrador puede generar reportes, los cuales son: (1) Calculo de la cantidad de componentes por categoría mostrándose en una gráfica de barras. (2) Mostrar los componentes más visualizados mostrándose en una gráfica de barras. (3) Calculo de la cantidad de componentes por estado mostrándose en una gráfica de torta. (4) Calculo de la cantidad de componentes por tecnología mostrándose en una gráfica de barras.

3.1. Arquitectura de Colossal – Repositorio de componentes software reutilizables

La arquitectura de software implica definir una solución estructurada que satisfaga todos los requisitos técnicos y operacionales y, a la vez, optimizar los atributos comunes de calidad como rendimiento, seguridad y capacidad de administración [13].

El sistema se desarrolló bajo un patrón arquitectónico Modelo-Vista-Controlador (MVC). Este patrón separa la parte gráfica de una aplicación, de los procesos lógicos y de los datos de esta. Fue desarrollado a finales de los años 70 por Trygve M. H. Reenskau.

El patrón MVC está compuesto por los siguientes elementos: La vista: Es la Interfaz gráfica, es con lo que el usuario de la aplicación interactúa y es donde se reciben las "ordenes" y se presentan los resultados de los procesos al usuario. El controlador: Es el modificador del modelo, es decir, es el que modifica los valores de las variables, objetos, datos en general, lo hace de acuerdo con lo solicitado por el usuario a través de la interfaz gráfica (vista). El modelo: Es la representación específica de la información con la que se está operando en el instante de la ejecución de la aplicación, representa las variables, objetos, datos en general que se están modificando de acuerdo a lo que el usuario solicite [14].

La aplicación de este patrón en la arquitectura de Colossal se puede apreciar en la figura 4.

En la capa del Modelo se encuentran los siguientes paquetes: 1) Seguridad el cual contiene los filtros y el manejo de sesión. 2) Negocio que reúne la lógica de la aplicación. 3) Los objetos de acceso a datos – DAO y 4) las entidades las cuales realizan un mapeo objeto relacional con la base de datos.

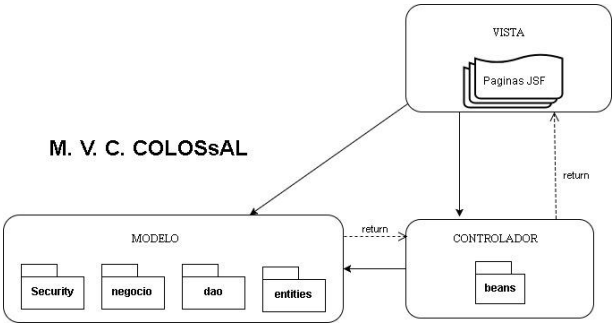

Figura 4. Arquitectura de Colossal – Repositorio de componentes software reutilizables. Fuente: elaboración propia.

En la capa del controlador aparecen las clases java beans (managed beans) que permiten gestionar las peticiones de las páginas y actualizar su estado. La vista corresponde a páginas JSF las cuales se encargan de mostrar, recoger, validar y convertir datos al usuario en cajas de texto y tablas.

La relación entre las capas del modelo arquitectónico se realiza de la siguiente manera: En la vista, las paginas JSF referencian a las entidades obteniendo sus datos para ser mostrados en una tabla o caja de texto. También hace uso de los controladores que se encargan de cumplir distintas funciones y preparan los datos para ser procesados en el negocio mediante la clase Colossal.java, la cual se conecta con los DAO para realizar las distintas operaciones en la base de datos.

Se usó de la tecnología Java en su séptima (7) versión empleando la plataforma Java Enterprise Edition. Para el desarrollo del modelo se emplea el framework Java Persistence Api (JPA) que ofrece un paradigma completamente orientado a componentes al trabajar la base de datos por medio de entidades, utilizando un mapeo objeto - relacional. La gestión y construcción del proyecto se realizó empleando la herramienta Maven [15]. Los controladores encargados de manejar la lógica del negocio al procesar la información suministrada por los clientes y representar los datos son soportados bajo el uso de Enterprise Java Beans (EJB) en su versión 3.

En el desarrollo de las interfaces de usuario se empleó el framework Java Server Faces (JSF) a través de la librería Primefaces en su versión 6.0 que incluye por defecto JQuery para el manejo de peticiones Ajax y el funcionamiento de algunos componentes. Se integró el framework Bootstrap que contiene plantillas tipográficas, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript

opcionales adicionales para potenciar el aspecto responsivo y estético de Colossal.

3.2. Descripción de las funcionalidades de Colossal – Repositorio de componentes software

A continuación, se describen los aspectos fundamentales que pueden encontrarse dentro del repositorio de componentes, los cuales se dividen en dos perfiles: 1) Vista para todos los usuarios. 2) Vista para los administradores.

La vista inicial es la de iniciar sesión que como su nombre indica es la que permite a los usuarios entrar al sistema digitando su usuario y contraseña. Figura 5.

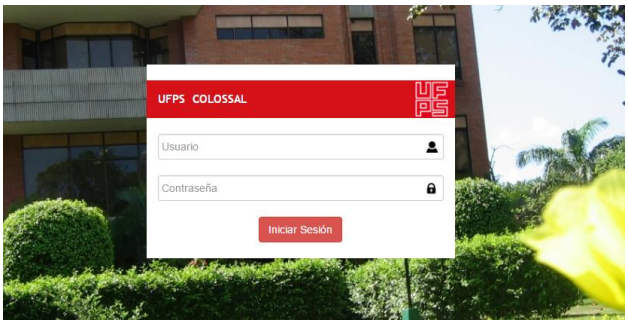

Figura 5. Inicio de sesión de Colossal – Repositorio de componentes software. Fuente: elaboración propia

Luego de iniciar sesión se presenta la vista inicial del sistema donde se pueden encontrar los diferentes componentes software que han sido registrados y que pueden ser descargardos para ser utilizados en los proyectos de desarrollo.

Al inicio del sistema destacan 3 apartados que se enumeran en la figura 6 y se explican a continuación.

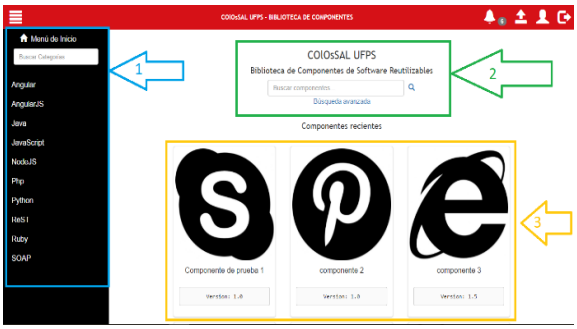

Figura 6. Vista inicial de Colossal – Repositorio de componentes software. Fuente: elaboración propia.

1. La sección 1 corresponde a las categorías. En ellas se despliegan las categorías que existen dentro del sistema, permitiéndole al usuario dar clic sobre alguna y direccionarlo hacia el respectivo panel donde se muestran los componentes asociados a esta. En este apartado se encuentra un campo de búsqueda que permite filtrar las categorías que se listan en caso de ser requerido.
2. En la sección 2 se encuentra el buscador que permite al usuario buscar un componente por su nombre, descripción o tecnología. También se encuentra el enlace que redirige a la vista de la búsqueda avanzada.
3. La sección 3 corresponde a los componentes recientes, en ella se muestran los últimos 6 componentes que han sido publicados en el repositorio, además de la opción "ver más componentes" que permite mostrar todos los componentes publicados en el repositorio en la vista de búsqueda.

La cabecera se muestra en todas las vistas del sistema a excepción del login. En esta encontramos 3 secciones, en el lado izquierdo está el botón que permite mostrar y esconder una barra lateral, en el centro encontramos un texto a manera de título y finalmente en el lado derecho hay una serie de opciones a manera de menú. Como se aprecia en la figura 7.

Figura 7. Opciones del menú principal de Colossal – Repositorio de componentes software. Fuente: elaboración propia.

A continuación, se muestran las funciones de cada icono de acuerdo a la numeración que se dispuso en la figura 7.

1. La campana de notificaciones permite desplegar el panel de notificaciones.
2. El icono de subida de componentes que brinda acceso a la respectiva vista.
3. Este icono permite acceder al perfil del usuario.
4. Acceso al panel administrativo. (esta opción solo se muestra si se es un administrador, en caso contrario no se visualiza).
5. Icono que al ser pulsado cierra la sesión y redirige al inicio del sistema.

Los usuarios cuentan con un panel donde pueden observar los componentes que ellos han subido al repositorio agrupados por estado (Aprobado, Pendiente, Eliminado), además de que pueden gestionar las listas personalizadas (crear, editar y eliminarlas).

4. Evaluación de Colossal – Repositorio de componentes software

Para evaluar el desarrollo y verificar que cumple con su propósito se realizó un estudio cuantitativo descriptivo donde se evalúa el nivel de aceptación de Colossal por parte de un grupo de estudiantes y docentes activos del programa de ingeniería de sistemas de la universidad Francisco de Paula Santander.

Se tomó como muestra los estudiantes que cursan la materia de Programación Web del segundo semestre de 2017 (18 estudiantes activos) y se les aplico un cuestionario. La forma de aplicación del cuestionario fue auto administrada. Estaba compuesto por 6 preguntas cerradas, algunas dicotómicas y otras con más de 2 posibilidades de respuesta. El objetivo de aplicar la prueba entre los estudiantes es determinar el nivel de conocimiento que tienen estos acerca de temas relacionados con la ingeniería del software basada en componentes (ISBC) y medir el nivel de aceptación de Colossal.

El procedimiento llevado a cabo para aplicar el cuestionario fue el siguiente:

1. Se presentó información correspondiente a la ingeniería del software basada en componentes, así como la estructura, proceso desarrollo y detalles generales del proyecto.
2. Se mostró la funcionalidad general del repositorio de componentes a los presentes.
3. Se atendieron preguntas correspondientes a los temas presentados, así como atención ante las dudas y recomendaciones presentadas.
4. Se aplicó el cuestionario.

Los resultados obtenidos fueron los siguientes:

Pregunta 1: ¿Tiene conocimientos previos sobre la ingeniería de software basada en componentes?

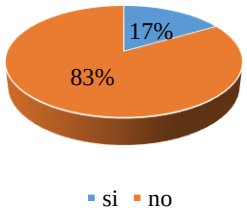

Figura 8. Nivel de conocimiento de los estudiantes de Ingeniería de Sistemas de la UFPS en la ISBC Fuente: elaboración propia.

En la figura 8 se puede apreciar que los estudiantes encuestados el 83% respondió que no conocían la ISBC y el 17% que sí la conocían. Se puede evidenciar que los estudiantes antes de la presentación los temas y el proyecto no tenían conocimientos formales de esta rama de la ingeniería del software.

Pregunta 2: ¿Conoce el concepto de componente software reutilizable?

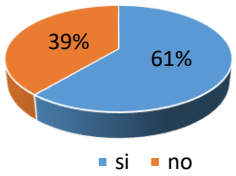

Figura 9. Conocimiento del concepto de componente software de los estudiantes de Ingeniería de Sistemas de la UFPS en la ISBC. Fuente: elaboración propia.

Según la figura 9, del total de encuestados el 39% respondió que no apropiaron el concepto de componente y el 61% que sí. Gracias a la presentación que se socializo con los estudiantes, se puede evidenciar que el 61% de los mismos apropiaron el concepto de un componente de software reutilizable.

Pregunta 3: ¿Tiene conocimiento sobre los beneficios que ofrece la reutilización de código en los procesos de producción y desarrollo de software? Ver Figura 10.

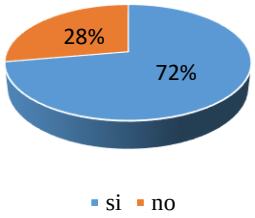

Figura 10. Conocimiento de los beneficios de la reutilización de los estudiantes de Ingeniería de Sistemas de la UFPS en la ISBC.

Según los encuestados el 28% respondió que no conocen los beneficios de la reutilización de software y el 72% que sí. El 72% de los estudiantes comprende los beneficios que pueden obtener al aplicar ingeniería del software basada en componentes en sus procesos de desarrollo gracias a la presentación realizada en el aula.

Pregunta 4: ¿Con que frecuenta hace uso de herramientas o tecnologías que le permitan efectuar reutilización de software? Ver figura 11.

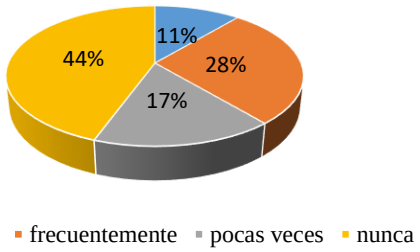

Figura 11. Uso de herramientas o tecnologías que le permitan efectuar reutilización por parte de los estudiantes de Ingeniería de Sistemas de la UFPS en la ISBC. Fuente: elaboración propia.

En cuanto al uso de herramientas tecnológicas que permitan reutlizar componentes, del total de los encuestados el 44% respondió nunca usan alguna herrmienta, mientras que el 17% pocas veces, el 28% frecuentemente y el 11% siempre. Se evidencia que la mayoría de estudiantes no hace uso de herramientas que faciliten la reutilización de código y que una minoría suele aplicar estas en sus procesos de desarrollo, esto puede deberse a una deficiencia en los procesos académicos relacionados a temas afines a la ingeniería del software basada en componentes o conceptos de reutilización de código.

Pregunta 5: ¿Encuentra a Colossal repositorio de componentes software una herramienta útil que permite fortalecer sus procesos de aprendizaje y formación como ingeniero de sistemas? Ver figura 12.

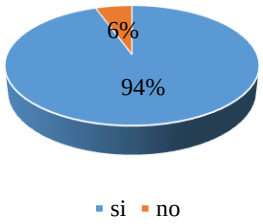

Figura 12. Colossal repositorio de componentes software una herramienta útil. Fuente: elaboración propia.

Según los encuestados el 6% respondió que no encuentra en Colossal una herramienta útil y el 94% que sí. La mayoría de los estudiantes respondió de manera positiva ante el uso de Colossal porque entendieron los beneficios que este les puede traer en sus proyectos académicos, lo que evidencia un interés en la aplicación de temas relativos a la ISBC y una alta aceptación ante herramientas que fomenten la reutilización de código, no obstante, una minoría correspondiente al 6% de los estudiantes se mostró escéptico de su utilización.

Pregunta 6: ¿Qué tan intuitiva le parece la navegabilidad del repositorio de componentes? Ver figura 13.

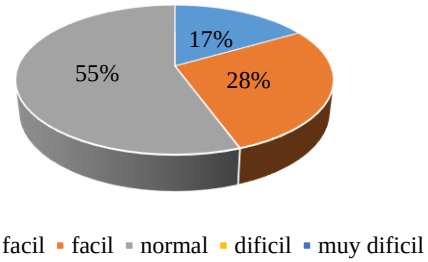

Figura 13. Colossal es una herramienta intuitiva y navegable. Fuente: elaboración propia.

Del total de los encuestados el 55% respondió normal, el 28% fácil, el 17% muy fácil y un 0% respondió difícil y muy difícil. Los estudiantes encontraron buena la navegabilidad y usabilidad del repositorio de componentes de software reutilizables, demostrando que puede tener beneficios en el desarrollo de aplicaciones web en el programa de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander.

En conclusión y según los datos recopilados se puede afirmar que los estudiantes no contaban con conocimientos solidos sobre la ISBC, esta área de la ingeniería del software no se imparte como una asignatura de la carrera, sin embargo, posterior a la presentación en el aula estos se informaron acerca de los beneficios y ventajas que se pueden obtener al ser aplicada en sus proyectos académicos o personales. La mayoría de los estudiantes respondió de manera positiva ante Colossal repositorio de componentes software y se evidenció una buena aceptación del funcionamiento y uso del mismo.

5. Conclusiones

Los repositorios de componentes software permiten mejorar el proceso de desarrollo de software, ya que facilita la reutilización de código. Brindando la capacidad de catalogar, especificar, consultar, descargar y reutilizar componentes de software para disponer de sus funcionalidades en proyectos de investigación, académicos y científicos, construyéndolos más rápido, de mayor calidad e invirtiendo una menor cantidad de recursos.

La ISBC facilita los tiempos y costos de producción, no obstante, tiene sus desventajas como lo es la dificultad en la detección de fallos y la aplicación de pruebas; pues cada componente de un sistema se comporta como otro individualmente. Diversos autores han tratado de dar soluciones a este tipo de inconvenientes al ofrecer modelos de detección de fallos y de pruebas que pueden ser aplicadas, sin embargo, hay que tomar en cuenta el dominio donde se aplica y la arquitectura sobre la cual se basa el desarrollo del proyecto.

El desarrollo de sistemas basados en componentes brinda una gran ventaja al ofrecer flexibilidad en la integración de diferentes tecnologías. Esto puede ser útil cuando un módulo en específico puede no satisfacer las necesidades de un requerimiento. En el caso particular de este proyecto se vio reflejado en la necesidad de integrar varios componentes para la gestión visual del sistema.

En el ámbito académico la ISBC puede fortalecer y potenciar las habilidades de los estudiantes si estos cuentan con las herramientas adecuadas. Por ello se hace necesario que las instituciones universitarias establezcan un proceso de adoctrinamiento en temas y conceptos relacionados con esta rama de la ingeniería del software.

En base a los datos recopilados en la encuesta aplicada; la mayoría de los estudiantes respondió de manera positiva ante Colossal repositorio de componentes, mostrándose entusiasta, evidenciando una alta aceptación del su funcionamiento y uso.

Referencias

[1] I. Crnkovic, S. Larsson, and M. R. V. Chaudron, "Component-based development process and component lifecycle," *J. Comput. Inf. Technol.*, 2005. doi: 10.1109/ITI.2005.1491195

[2] J. Bosch, C. Szyperski, and W. Weck, "Component-oriented programming," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2002.

[3] R. S. Pressman, *Ingeniería del Software un enfoque práctico.*, Septima. Mexico: McGraw Hill Interamericana editores, 2010.

[4] F. H. Vera-Rivera and F. A. Rojas Morales, "Propuesta de aplicación de la Ingeniería del Software Basada en Componentes en el desarrollo de software empresarial," *Rev. Iteckne*, vol. 7, no. 2, pp. 128–135, 2010.

[5] S. Newman, *Building Microservices*. O'Reilly Media, Inc., 2015.

[6] F. H. Vera-Rivera, "A development process of enterprise applications with microservices," *J. Phys. Conf. Ser.*, vol. 1126, no. 17, p. 012017, Nov. 2018.

[7] GitHub Inc, "The world's leading software development platform · GitHub," 2020. [En línea]. Disponible en: https://github.com/.

[8] K. Beck *et al.*, "Manifiesto por el Desarrollo Ágil de Software," 2001. [En línea]. Disponible en: http://agilemanifesto.org/iso/es/manifesto.html.

[9] "13 Anual State of Agile Report," CollabNet , 2019. [En línea]. Disponible en: https://stateofagile.com/?_ga=2.35492769.1692634280.1581014735-1505261659.1581014735#ufh-c-473508-state-of-agile-report

[10] K. Beck, *Extreme Programming Explained: Embrace Change*. Addison Wesley, 2000.

[11] R. Delgado Rojas, F. A. Rojas Morales, and F. H. Vera-Rivera, "Diseño e implementación de un repositorio de componentes software para soportar el desarrollo de software empresarial – caso : División de Servicios de Información de la Universidad Industrial de Santander," *Rev. Iteckne*, vol. 8, no. 2, pp. 223–233, 2011.

[12] F. H. Vera-Rivera, B. R. Perez-Gutierrez, and F. J. Torres-Bermudez, "Sandbox UFPS - cloud development platform for server management, creation and deployment of web applications of academic use," *Res. Comput. Sci.*, vol. 101, pp. 65–75, 2015.

[13] Microsoft - MSDN, "Información general sobre la arquitectura de software - MSDN | Microsoft Docs," *MSDN - Microsoft*, 2011. [En línea]. Disponible en: https://docs.microsoft.com/es-es/previous-versions/msdn10/Hh144976(v=MSDN.10).

[14] C. A. López S., "Cómo mantener el patrón modelo vista controlador en una aplicación orientada a la WEB," *INVENTUM*, vol. 4, no. 7, pp. 72–78, Jul. 2009.

[15] Apache Software foundation, "Maven – Welcome to Apache Maven," *Apache Maven*, 2020. [En línea]. Disponible en: https://maven.apache.org/.