UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 4

Componente para la integración del Juez en Línea Caribeño y la plataforma educativa Zera 2.0

Trabajo de Diploma

para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Arletys González Madera
Reysel Pérez Avilés

Tutores: Ing. Orlando Grabiel Toledano López
Ing. Yonny Mondelo Hernández

"Año 58 de la Revolución"
Ciudad de la Habana, junio de 2016

Pensamiento

"Estudia, mientras otros están durmiendo; planea, mientras otros están jugando; trabaja mientras otros estén haciendo el vago; y sueña mientras otros desean."

William Arthur Ward

Declaración de Autoría

Declaramos ser autores del presente trabajo de diploma y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Arletys González Madera

_____

Firma del autor

Reysel Pérez Avilés

_____

Firma del autor

Ing. Yonny Mondelo Hernández

_____

Firma del tutor

Ing. Orlando Grabiel Toledano López

_____

Firma del tutor

Dedicatoria

Dedico todo mi esfuerzo durante estos largos años a la persona que más AMO en mi vida, al ángel que día a día no descansa velando por mi vida, a la mujer más hermosa de esta tierra, mi más preciado tesoro, a mi mamá Maritza. Solamente te estoy devolviendo lo que me distes en un inicio.

A mi padrastro José Luis, por ser como un padre para mí, por estar siempre ahí a pesar de la distancia que nos ha separado durante estos 5 años.

A mi hermana Arlenys, que la quiero mucho y es mi otro tesoro, con la que siempre he peleado y además, compartido momentos especiales de mi vida.

A mi novio Osmel, por darme amor verdadero, ese que hoy en día escasea mucho y vale más que el propio oro, por ser mi confianza, mi apoyo y siempre estar a mi lado a pesar de todos los obstáculos que se nos presentaron en el transcurso del camino.

A toda mi familia, en especial a mi abuela Cira Martha, a mi tía María Elena, a mi tío Leonel , a mis primos, a mi suegra, a mi cuñada y a Fidelina, que de una forma u otra ayudaron para que este sueño se hiciera realidad y siempre creyeron que yo podía lograrlo.

A la memoria de mi abuelo y de mi tía por el cariño, el amor y la educación que siempre me dieron, que aunque no me acompañen en esta importante etapa de mi vida, siempre los tengo presente.

A todas las personas que se sienten felices por este logro y que creyeron en mí.

Arletys

Dedicatoria

A mis padres, Luis y Ania, como reconocimiento a su amor, consagración y esmero durante mi vida y especialmente en mis años de estudiante, ya que con su apoyo, lo que fuese un sueño, hoy es una realidad.

A mis tíos, a quines les decimos cariñosamente: "Ten" y "Chombi", por siempre haberme apoyado.

A mi primo, a quien le llamo "Abelito", así como a esa magnífica mujer, "Martica", quienes siempre han estado al pendiente de mí.

A esas grandes personas que fueron "Chucha" y "Lupe", quienes me trataron como si fuese uno más de sus hijos.

A mi querida hermanita Yesenia, que ni tan chiquita, aunque para mi siempre lo seguirá siendo.

Al resto de mi familia, que han visto que no importa la cantidad de obstáculos que se interpongan en nuestro camino, lo que realmente importa es la resistencia y los deseos de lograr cumplir las metas que nos proponemos en la vida.

A Lisy, ella que sabe cuan importante a sido para mí, quien me motivó a cambiar la forma de ver la vida y me enseñó el significado del amor, gracias por haber estado ahí para mí.

A todas aquellas personas que de una forma u otra me apoyaron en todo momento y creyeron en mí.

Reysel

Agradecimientos

A mi mamá, por ser un ejemplo para mí, por ser madre y padre, por ser ese motor impulsor que me guió durante todo este largo camino, por siempre haber intentado que yo no tuviese otra preocupación que atender a no ser mis estudios, por apoyarme, guiarme, quererme mucho y mantenerse ahí para mí. Por todo esto y más te agradezco de todo corazón el que estés conmigo a mi lado, sin tí, este momento no hubiese sido posible.

A mi novio Osmel, por soportarme durante todos estos años, por ser tan comprensivo, por saber cómo llegar a mí, por estar presente y guiarme en los momentos más difíciles por los que he atravesado, por apoyarme para que mi sueño se hiciera realidad.

A mi padrastro y a mi hermana por quererme mucho, por brindarme su amor incondicional y por soportar mis peleas, sobre todo.

A toda mi familia que a pesar de la distancia todo este tiempo me ha brindado su apoyo, comprensión y fundamentalmente no han dejado de creer y confiar en mí, y especialmente a mi abuela, a mis tíos, a mis primos, a mi suegra y mi cuñada que se han convertido en mi otra familia.

A mi abuelo Juan y a mi tía "niña", que aunque no están entre nosotros en el día de hoy, se sentirían orgulloso de mi, al ver que el sueño de su nieta y su sobrina se hizo realidad.

A mi compañero de tesis Reysel, por su paciencia, sé que trabajar con una persona como yo es todo un reto.

A mi tutor Orlando y al profesor Adrián, por haber batallado junto a nosotros esta última etapa de la carrera, la cual ha sido dura, pero ha valido la pena y por habernos dado siempre, entre su tremendo cúmulo de trabajo, la guía necesaria.

A mi amiga Rachelys, por ser mi compañera de las mil pruebas o las mil batallas, por siempre acompañarme para estudiar y brindarme su amistad sincera aceptándome tal como soy, con mis defectos y virtudes.

A Yaritza, a quien conozco desde el IPI, gracias por prestarme tus oído, por regalarme tu corazón cuando me supiste comprender, cuando hiciste que reconociera mis errores y que de

Agradecimientos

ellos aprendiera, gracias por abrirme los ojos y evitar que me equivocara.
"Gracias por su amistad a las dos. Sépase que en cualquier lugar del mundo donde estén y aun cuando pasen muchos años, siempre podrán contar conmigo"

A mi amigo desde la infancia Luis Ángel y a su mamá que siempre estuvieron al tanto del desarrollo de mi tesis, apoyándome y demostrándome que no hay nada imposible para alcanzar mis sueños, y que sé que hoy se sienten felices por este logro.

A todas las personas que no dudaron en ayudarnos en el desarrollo del trabajo, y a aquellas que han estado presente durante estos 5 años, creando recuerdos, estudiando y festejando; en especial a todos los amigos que compartieron una larga temporada conmigo en Artemisa, y a los amigos del IPI.

A todos los educadores que contribuyeron en mi formación, y especialmente aquellos que supieron marcar la diferencia y realmente influyeron muy positivamente en mi formación y fueron un ejemplo como Ismael Nodarse, Ángel Alberto Vázquez, Maritza Calaña y Dania Domínguez.

A la Revolución Cubana y en especial al Comandante en Jefe Fidel Castro Ruz por darme la posibilidad de formarme y convertirme en la profesional que soy hoy.

A todas las personas que de una forma u otra dieron un poco de su tiempo y ayuda para lograr culminar esta obra.

A aquellas personas que me desean lo peor e hicieron hasta lo imposible para que no pudiera estar aquí en el día de hoy, pues a ustedes les digo que:
Quien me lastima me hace FUERTE,
Quien me critica me hace IMPORTANTE,
Quien me envidia me hace VALIOSO,
Es hasta divertido saber que ustedes que me desean lo peor...tienen que conformarse que siempre me ocurra LO MEJOR.
Gracias a ustedes me esforcé el doble para llegar a donde estoy hoy.

Arletys

Agradecimientos

A mi mamá, que es todo para mí, por ser quien impulsa mi vida, por cuidarme, guiarme, quererme mucho y estar ahí siempre para mí. Por todo eso y más, sabes que eres mi mundo y no sé que hubiese sido mi vida sin tí.

A mi padre, quien ha sido un guía en mi camino, me ha apoyado en todo momento sea cual sea la decisión que haya tomado y de quien he aprendido tanto. Te quiero, papá.

A mi hermana por quererme mucho a pesar de nuestras peleas, sabes que te quiero y nunca te dejaría sola, siempre estás conmigo.

A mis tíos, a los que cariñosamente les llamo: "Ten" y "Chombi", quienes siempre me han brindado su cariño y apoyo, así como al resto de la familia a quien quiero tanto.

A Lizy, esa chica que supo robarse mi corazón, que me hizo fuerte y de una forma u otra ha sido la razón de esas nuevas metas por seguir, siempre tendrás ese lugar especial en mi corazón aún el día en que ya no estés.

A mi compañera de tesis Arletys.

A todos mis amigos por haber estado ahí en todo momento, por haber sido mis compañeros y brindarme su amistad sincera, siempre los recordaré.

A mi tutor Orlando, así como a: Adrián, Agustín, Arley y Miguel, por haberme brindado su ayuda durante esta última etapa de la carrera, aún cuando se encontraban inmersos en un mar de trabajo, muchas gracias.

A todas aquellas pesonas que de una forma u otra influyeron en mi vida universitaria de tal forma que este es el resultado, en especial a mis amigos: Alexander, Camilo, César, Marcos, Osvel, Rafael, Rosbel y a todos los que compartieron esta etapa de mi vida.

A los profesores que me han guiado a lo largo de la carrerra, por haberme dado las armas necesarias para enfrentar el futuro, en especial a Marcel Puentes Rojas, quien fue el que me encaminó esos primeros años convirtiendose en un extraordinario guía, muchas gracias a todos.

Reysel

Resumen

El vertiginoso avance de las Tecnologías de la Información y las Comunicaciones, ha tenido un impacto significativo en el proceso de enseñanza-aprendizaje, incorporando nuevas herramientas que permiten una mejor comunicación e interacción entre alumnos y docentes, entre las que se destacan las plataformas de gestión de aprendizaje. La plataforma educativa Zera 2.0, actualmente en desarrollo es un fiel ejemplo de lo anteriormente mencionado. Esta presentaba la necesidad de que los profesores de programación contaran con un espacio donde pudieran seguir el aprendizaje de sus estudiantes haciendo uso del COJ y aportar sus conocimientos, por tal motivo se realiza el presente trabajo. Se propone un componente (Conexcoj) interoperable con la plataforma de gestión de aprendizaje Zera 2.0 que permite dar seguimiento al aprendizaje de la disciplina de programación desde la plataforma. Para ello fue necesario analizar elementos teóricos y principales tendencias de integración de sistemas. Luego, se definieron la metodología, herramientas y tecnologías que fueron empleadas en el desarrollo de las funcionalidades propuestas. Posteriormente, se realizaron los artefactos propuestos por los flujos de trabajo modelado de negocio, requisitos, análisis, diseño, implementación y prueba. Finalmente el sistema fue validado a través de pruebas unitarias, de integración y de sistemas, evidenciándose que existe una correspondencia satisfactoria entre el objetivo y los resultados obtenidos, así como una alta satisfacción de los usuarios finales.

Palabras claves: docente, plataforma educativa, evaluación, funcionalidades, integración, proceso de enseñanza-aprendizaje (PEA).

Índice

Índice

Índice

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) están presentes formando parte de la cultura tecnológica que nos rodea y con la que debemos convivir. Las TIC por sus características han cobrado un auge extraordinario al revolucionar la forma de vivir, de comunicarse y hasta de resolver los problemas que se presentan en prácticamente todas las esferas de la sociedad, fundamentalmente en la esfera de la educación. (1)

El desarrollo y perfeccionamiento del proceso de enseñanza y aprendizaje ocupa un lugar prioritario en la Universidad de las Ciencias Informáticas (UCI). En esta universidad se busca aprovechar al máximo las TIC en los procesos fundamentales que en ella se desarrollan, convirtiéndolas en un componente indisoluble en la forma de impartir, evaluar e impulsar los conocimientos tanto en el pregrado como en el postgrado.

Una de las herramientas empleadas en el proceso de enseñanza y aprendizaje son los Sistemas de Gestión de Aprendizaje o LMS (*Learning Management System*, por sus siglas en inglés) que son software o herramientas informáticas, que se basan en los principios del aprendizaje colaborativo. Un LMS tiene entre sus principales funciones: gestionar usuarios, recursos, materiales y actividades de formación, administrar el acceso, controlar y dar seguimiento al proceso de aprendizaje, realizar evaluaciones, entre otras. (2)

Actualmente para dar soporte al proceso de enseñanza - aprendizaje existen diversos LMS que le brindan gran número de facilidades tales como Moodle, Sakai y Canvas. La Facultad 4 de la UCI está a la vanguardia en el desarrollo de plataformas educativas para la gestión del aprendizaje, y más específicamente el Centro de Tecnologías para la Formación (FORTES), se encuentra desarrollando la Plataforma Educativa Zera 2.0, la cual es capaz de adaptarse a los procesos del negocio de alguna institución educacional.

Zera 2.0 es una plataforma de aprendizaje formal que posibilita, entre otros aspectos, el diseño de cursos y llevar un seguimiento detallado de las acciones del educando y sus avances. Esta plataforma sirve de apoyo al profesor, el cual tendrá a su disposición un grupo de opciones para gestionar el aprendizaje de sus estudiantes y aumentar su

preparación profesional. Además le permitirá al mismo realizar sus responsabilidades como educador y evaluador. (3)

Una de las estrategias tomadas a nivel mundial en el área educativa con el objetivo de incidir en el aprendizaje de la disciplina de programación ha sido el desarrollo de los concursos de programación como el ACM *International Collegiate Programming Contest* (ACM-ICPC)[1] surgido en los años 70. (4)

Como soporte a la preparación de este tipo de concursos han surgido un conjunto de herramientas informáticas, entre ellas los jueces en línea de programación creados fundamentalmente para entornos académicos. La naturaleza de estos consiste en proveer problemas de variadas complejidades, y a su vez, evaluar de forma automática las soluciones que los usuarios encuentran a dichos problemas. (4)

La programación de competencia como disciplina formativa y educacional es un campo que se ha explotado durante los últimos años en la UCI, lo cual ha traído consigo el desarrollo incremental y el auge del movimiento ACM-ICPC en Cuba, dando origen de esta forma a la creación y mantenimiento del Juez en Línea Caribeño o COJ (*Caribbean Online Judge*, por sus siglas en inglés); sistema que está disponible en *Internet*[2] desde el 5 de junio de 2010, el cual ha sido fundamental en el desarrollo de múltiples concursos de programación en la UCI, Cuba y el Caribe. (4)

El COJ es utilizado para disímiles finalidades, tales como: herramienta educativa para instrucción informal, herramienta de autoaprendizaje y autosuperación, sistema de manejo de competencias y entretenimiento. Provee además un espacio donde las personas de

---

[1] Es el concurso más antiguo, más grande y de mayor prestigio en el mundo de la programación**.** Competencia de programación de múltiples niveles, basado en equipo con sede en la Universidad de Baylor. El concurso fomenta la creatividad, el trabajo en equipo y la innovación en la creación de nuevos programas de software, y permite a los estudiantes poner a prueba su capacidad para actuar bajo presión. (5)

2 http://coj.uci.cu

todo el mundo pueden intercambiar experiencias y conocimientos, compartir sus problemas y soluciones, probar y mejorar sus habilidades en programación. Esto les permite a los distintos usuarios entrenar para los concursos y olimpiadas de programación como la ACM-ICPC, la IOI[3] y Copas universitarias coma Pascal o la Void.

Desde hace un tiempo los profesores de la asignatura de programación de la Facultad 4 de la UCI se han trazado la meta de evaluar el desempeño de sus estudiantes en el COJ desde la plataforma Zera 2.0. Para ello, los han estado incentivando para que participen en el COJ y de esta forma, adquieran o potencien sus habilidades en la disciplina. Al hacer esto, los educadores aumentan el esfuerzo que realizan, dado que no cuentan con una vía que le permita llevar a cabo el seguimiento a sus estudiantes a través de la revisión de las actividades orientadas por él, para el estudio individual de cada uno, desde la plataforma hacienda uso del sistema de programación competitiva COJ. En otros términos, la plataforma no permite entre otros aspectos: revisar y gestionar las actividades con los problemas de programación que son asignados a los estudiantes, visualizar el estado de los estudiantes así como también el estado de realización de las actividades que le son orientadas, y no brinda la posibilidad de buscar un problema en el COJ para conformar una actividad que posteriormente será asignada. De esta forma queda evidenciado que hasta este momento del desarrollo no se han aprovechado las potencialidades que brinda el COJ desde la plataforma de teleformación Zera 2.0, es decir, no se ha usado de este juez en línea el vasto banco de problemas con que cuenta, el cual soporta la resolución de estos en varios lenguajes de programación y una vez compilados se devuelve una evaluación o resultado, proceso que minimizaría el trabajo del profesor en el momento de evaluar las

3 La Olimpiada Internacional de Informática, (en inglés International Olympiad in Informatics, abreviado IOI), El IOI es una de las cinco olimpiadas científicas internacionales. El objetivo principal de la IOI es estimular el interés por la informática y la tecnología de la información. Otro objetivo importante es reunir a los alumnos con un talento excepcional de varios países y para que compartan experiencias científicas y culturales. En resumen esta olimpiada potencia el aprendizaje de la informática en alumnos de nivel medio superior (pre-universitario).(6)

soluciones de sus estudiantes. La realización de todo el proceso en el propio COJ resultaría abrumadora pues aun cuando el educador accede a la aplicación no tiene forma de controlar y dar seguimiento a los estudiantes de su grupo. Esto es debido a que en el COJ las personas se registran bajo un nombre de usuario el cual no necesariamente pertenece al dominio de la UCI. Todas estas deficiencias impiden maximizar el éxito en el aprendizaje individual y/o colectivo de cada educando.

Siendo esta la principal problemática se plantea como **problema a resolver**: ¿Cómo garantizar la interacción y el intercambio de información entre la plataforma educativa Zera 2.0 y el Juez en Línea Caribeño para dar seguimiento al aprendizaje de la disciplina de programación desde la plataforma?

A partir del problema expuesto se puede inferir como **objeto de estudio**, la integración entre sistemas y como **campo de acción**, la integración de la plataforma educativa Zera 2.0 con el Juez en Línea Caribeño.

En esta investigación se establece como **objetivo general**: Desarrollar un componente que integre el COJ con la plataforma educativa Zera 2.0 para dar seguimiento al aprendizaje de la disciplina de programación desde la plataforma.

Para dar solución a la interrogante planteada en el problema, la investigación se sustenta en la siguiente **hipótesis**: La implementación de un componente que integre el COJ con la plataforma educativa Zera 2.0 permitirá dar seguimiento al aprendizaje de la disciplina de programación desde la plataforma.

A partir del objetivo general definido, se derivan los siguientes **objetivos específicos:**

- Analizar elementos teóricos y principales tendencias del desarrollo web en la actualidad en el campo de la integración de sistemas.
- Analizar las diferentes funcionalidades y características que tendrá el componente para la integración.

- Definir la metodología, herramientas y tecnologías a utilizar en el desarrollo del componente.
- Implementar las funcionalidades del componente para la integración del COJ y la plataforma educativa Zera 2.0.

Para dar cumplimiento a los objetivos específicos se proponen las siguientes **tareas de investigación:**

- Estudio de los principales conceptos y tecnologías utilizadas en el desarrollo de componentes de integración de sistemas.
- Análisis del estado actual de las tendencias de integración entre plataformas de gestión de aprendizaje y otros sistemas e-learning.
- Análisis y definición de la metodología de desarrollo a utilizar, así como las herramientas y tecnologías de desarrollo que cumplan los requisitos para ser utilizadas en la solución.
- Elaborar el modelo de dominio o modelo conceptual del sistema.
- Definir los requerimientos del sistema solicitados por el cliente: requisitos funcionales y no funcionales.
- Realizar el modelado de análisis del sistema que se desea desarrollar.
- Definir patrones de diseño a utilizar en el desarrollo de la propuesta de solución.
- Realizar el modelado de diseño de todas las funcionalidades identificadas.
- Implementación de los requerimientos solicitados por el cliente.
- Validación de la propuesta de solución a través de pruebas de software.

Para lograr los objetivos se definen como **resultados esperados**:

- Un componente mediante el cual se podrá integrar el COJ en la plataforma educativa Zera 2.0, aprovechando las potencialidades que brinda este juez en línea.
- Documentación del proceso y del producto, entendiéndose como producto al componente en sí, de manera que pueda continuarse con el mantenimiento de este, así como el desarrollo de futuras versiones.

Introducción

Para la ejecución de la investigación fueron empleados **métodos teóricos** y **empíricos**, con el propósito de dar respuesta a la problemática planteada. Estos se relacionan a continuación:

Métodos teóricos:

- ***Analítico – sintético:*** Permitió estudiar y realizar el análisis de documentos y bibliografías que sustenten tanto de forma teórica como práctica los elementos asociados a la integración de la plataforma educativa Zera 2.0 con el sistema de programación competitiva COJ, identificando los elementos más importantes y necesarios para dar solución al problema planteado.
- ***Análisis histórico – Lógico:*** utilizado para analizar la evolución histórica de soluciones similares y las tendencias más recientes sobre la integración entre sistemas, para concluir qué aspectos son necesarios en el desarrollo de la solución que se propone.
- ***Modelación:*** Es utilizado para realizar los diagramas necesarios que modelarán la solución.

Métodos empíricos:

- ***Observación:*** Se utilizó para obtener información de las necesidades existentes a lo largo del desarrollo del proyecto y para observar las facilidades que brinda la gestión de las actividades asignadas por los profesores a sus estudiantes en el COJ desde la plataforma Zera 2.0. Permite una comparación de los resultados obtenidos por diferentes vías, contribuyendo alcanzar una mayor precisión en la información recogida.

La presente investigación está conformada por tres capítulos, quedando estructurados de la siguiente manera:

Capítulo 1. Fundamentación Teórica: En este capítulo se describe la fundamentación teórica de la investigación, así como también se incluye un estudio del estado del arte del

tema. Por tanto, se exponen los elementos teóricos utilizados en la investigación, describiendo además las tecnologías, metodologías, herramientas y los lenguajes de programación utilizados en el desarrollo de la solución, así como los principales conceptos involucrados, para una mejor comprensión.

Capítulo 2. Análisis y Diseño: En este capítulo se describe el proceso de análisis y diseño, o sea, se exponen los elementos que permiten describir la propuesta de solución para lograr un entendimiento claro de las funcionalidades a desarrollar. También se detalla la arquitectura de la solución y sus principales características.

Capítulo 3. Implementación y Pruebas: En este apartado se describen las fases de implementación y pruebas. Se realiza la implementación de todas las funcionalidades identificadas, logrando un componente que satisface las principales necesidades del cliente. Se detallan además las pruebas realizadas al sistema, una vez que concluye la implementación, para asegurar que este cumple con las especificaciones requeridas, para de esa manera asegurar la calidad y eficiencia de la solución.

Seguidamente se encuentran las referencias bibliográficas que está formada por documentos y artículos de interés que fueron consultados por los autores de la investigación y se concluye con los anexos donde se agregan algunos artefactos generados en el transcurso del desarrollo de la solución.

Capítulo 1: Fundamentación teórica

Para la fundamentación del presente trabajo es necesario establecer ciertos conceptos básicos fundamentales, inherentes al ambiente de la programación de competencia, los Jueces en Línea, el desarrollo de componentes y la plataforma educativa Zera 2.0. También es necesario un estudio del estado del arte a nivel mundial sobre soluciones similares. Además, en este capítulo se incluirán las tecnologías, metodologías, herramientas y lenguajes de programación utilizados en el desarrollo de la solución, así como una explicación de las razones que llevan a su selección.

1.1 Conceptos asociados al dominio del problema

Para la correcta comprensión del problema y su solución es necesario establecer y analizar los principales conceptos y definiciones asociadas al dominio del problema.

Aprendizaje o educación formal: Aprendizaje ofrecido normalmente por un centro de educación o formación, con carácter estructurado y que concluye con una certificación. El aprendizaje formal es intencional desde la perspectiva del alumno. (7)

Aprendizaje o educación informal: Aprendizaje que se obtiene en las actividades de la vida cotidiana relacionadas con el trabajo, la familia o el ocio. No está estructurado y normalmente no conduce a una certificación. El aprendizaje informal puede ser intencional, pero en la mayoría de los casos, no lo es (es fortuito o aleatorio). (7)

Juez en línea: Es la evolución de sistemas similares llamados evaluadores automáticos. Es una aplicación web para entornos generalmente académicos, que incluye varios problemas de distintas materias para ser resueltos mediante técnicas de programación, además, evalúa automáticamente las soluciones de sus usuarios en los disímiles lenguajes de programación disponibles. (8) Un Juez en línea usualmente ofrece además otros servicios comunitarios y de consulta a sus usuarios, y suele fungir como herramienta para la realización de competencias de programación.

1.1.1 Plataforma educativa

Según Diéguez Rodríguez y Barrio Sáenz, “Es una herramienta física, virtual o una combinación de ambas, que brinda la capacidad de interactuar con uno o varios usuarios con fines pedagógicos. Se considera además, que contribuye en la evolución de los procesos de aprendizaje y enseñanza, complementando o presentando alternativas a las prácticas de educación tradicional”. (9)

Las plataformas son Entornos Virtuales de Educación y Aprendizaje (EVEA), que surgen y se desarrollan con la intención de dar soporte a la docencia, se apoyan en sistemas informáticos basados habitualmente en protocolos de la web con herramientas adaptadas a las necesidades educacionales requeridas. (10)

Plataformas de gestión de aprendizaje (PGA): También conocidas como LMS, son software o aplicaciones web que proveen las funciones administrativas y de seguimiento necesarias para posibilitar y controlar el acceso a los contenidos, implementar recursos de comunicaciones y llevar a cabo el seguimiento de quienes utilizan la herramienta. En general, facilitan la interacción entre los docentes y los estudiantes, aportan herramientas para la gestión de contenidos académicos y permiten el seguimiento y la evaluación. (11)

Plataforma educativa Zera 2.0: Es una plataforma de gestión de aprendizaje, actualmente en desarrollo, donde un centro educativo, institución o empresa, gestiona recursos educativos proporcionados por unos docentes y organiza el acceso a esos recursos por los estudiantes, y además permite la comunicación entre todos los implicados (alumnado y profesorado), contribuyendo de esta manera a la evolución de los procesos de enseñanza y aprendizaje. Es decir, Zera 2.0 se encarga, entre otros aspectos, de presentar los cursos a los usuarios y del seguimiento de la actividad de los estudiantes.

Después de analizados los conceptos anteriores se puede afirmar que las plataformas educativas son un entorno virtual que acogen variadas herramientas para fines docentes. Su función es permitir la creación y gestión de cursos que permitan elevar el nivel de conocimiento de sus usuarios.

1.1.2 Componente de software

Una de las piedras angulares en el desarrollo de software en entornos de desarrollo rápido de aplicaciones (*RAD*, por sus siglas en inglés) es la programación basada en componentes. Con el desarrollo basado en componentes se consigue un desarrollo muy ágil y más simple, además permite que se puedan alcanzar altos niveles de reutilización y mantenimiento del código.

Se pueden encontrar varias definiciones de un componente en las diversas literaturas existentes, la mayoría de las cuales no dan una definición intuitiva de un componente, sino que se centran en los aspectos generales de un componente. Por ejemplo, en un Modelo de Objetos de Componentes (COM) en un resumen técnico de Microsoft, *"*un **componente** se define como una pieza de software compilada, que ofrece un servicio*"* (12). Esta definición es demasiado amplia, ya que, por ejemplo, incluso las bibliotecas compiladas se pueden definir de esta manera. A continuación se aclarará el concepto de componente teniendo en cuenta las diferentes definiciones que se encuentran en las literaturas.

Diferentes definiciones de componentes

- Un componente es una parte reemplazable, casi independiente y no trivial de un sistema que cumple una función clara en el contexto de una arquitectura bien definida (13).
- Szyperski define un componente de software como una unidad de composición con interfaces especificadas contractualmente y dependencias explícitas en un único contexto. Se puede implementar de forma independiente y está sujeto a la composición por terceras partes (14).
- D'Souza y Wills definen un componente como una parte reutilizable de software que se desarrolló de forma independiente y puede combinarse con otros componentes para construir unidades más grandes. Puede ser adaptado, pero no puede ser modificado (15).

Después de analizar estas definiciones y de acuerdo al componente que se desea

desarrollar en el presente trabajo se puede concluir que un componente es una pieza de software reutilizable con bajos niveles de dependencia de otros componentes o librerías, que puede ser reconfigurado de acuerdo a las necesidades de la aplicación que se desea desarrollar y suple una necesidad evidente de la arquitectura definida.

1.2 Análisis de soluciones similares

En la UCI actualmente se desarrolla la plataforma educativa de gestión de aprendizaje Zera 2.0, la cual se quiere integrar con el Juez en Línea Caribeño para aprovechar al máximo las potencialidades que este brinda. En estos momentos no se ha desarrollado componente alguno, que facilite íntegramente este proceso. Con el objetivo de obtener conocimiento y hallar una solución al problema existente fue necesario realizar un estudio sobre las tendencias actuales sobre integración de sistemas que existen. Algunas de las soluciones relevantes con respecto a la temática en cuestión son analizadas a continuación.

1.2.1 Ejemplos de soluciones similares en el mundo

El desarrollo de las TIC, ha impulsado el surgimiento de diferentes iniciativas encaminadas a lograr la integración entre las herramientas de apoyo a los procesos de enseñanza-aprendizaje. En este ámbito, existen disímiles plataformas educativas que implementan las principales tecnologías, estándares de integración entre sistemas y protocolos de comunicación utilizados a nivel mundial. Algunas de las soluciones relevantes son analizadas a continuación.

EDUJUDGE: Integrando el Juez-Online en *e-learning* efectivo

El objetivo principal del proyecto EduJudge es integrar el Juez Online de la UVA en un entorno educativo efectivo, con el fin de satisfacer la demanda de los usuarios de un mayor carácter pedagógico. De esta forma se facilita el uso del Juez Online en los cursos oficiales del campo de las matemáticas y la programación de la enseñanza secundaria y universitaria. (16)

Capítulo 1
Fundamentación Teórica

Durante este proyecto, de más de dos años de duración, el sistema EduJudge ha sido desarrollado y probado en entornos reales. Este sistema consta de tres principales componentes:

- **Juez Online de la UVA** es un servidor de evaluación que automáticamente evalúa la solución enviada a un problema propuesto, y proporciona retroalimentación al estudiante. (16)
- **QUESTOURnament** permite el desarrollo de concursos en la plataforma Moodle, durante los cuales los estudiantes compiten entre sí tratando de resolver un conjunto de desafíos en un límite de tiempo. La tarea de los estudiantes no se limita sólo a responder desafíos, sino que también pueden proponer nuevos desafíos y evaluar las respuestas de sus compañeros y ser recompensados por ello. De manera general el módulo QUESTOURnament posibilita hacer un seguimiento continuo del trabajo personal del alumno, además de proporcionar un registro de su participación. Por otra parte, también ofrece a los profesores la posibilidad de definir sus propios instrumentos de evaluación (pruebas objetivas, libres…) en función del tipo de competencia a evaluar. (17)
- **CrimsonHex** es un repositorio especializado en problemas de programación, aunque soporta cualquier tipo de objeto de aprendizaje, que permite mejorar la accesibilidad y el uso de problemas de algoritmia y programación. (16) De manera general este repositorio cuenta con una colección adecuada de los problemas de programación validados, que fueron almacenados previamente como objetos de aprendizaje (OA). Además contiene una gran variedad de problemas de programación y algorítmicos que se pueden obtener en busca de diferentes características, tales como palabras claves, autor, tipo, nivel de instrucción, nivel de dificultad y el lenguaje. (18)

EduJudge integra estos tres componentes en la plataforma open-source para gestión y desarrollo de procesos de aprendizaje Moodle, dando lugar a un servicio que permite obtener una evaluación automática de las soluciones a los problemas de programación propuestos, cargar, buscar y recuperar objetos de aprendizaje del repositorio dotado con

una colección interesante de problemas de programación de alta calidad, crear cuestionarios online a través de la herramienta de Moodle Quiz, que incluya cuestiones del tipo EduJudge e involucrar a los estudiantes en un entorno interactivo, competitivo y motivante a través de QUESTOURnament. (16)

Sphere Online Exercises (SPOX[4])

Fue anunciada por el *Sphere Interest Project* con el objetivo de extender las prestaciones de SPOJ, es una plataforma dedicada a la enseñanza de la programación, desarrollada sobre el mismo motor del juez en línea original. Esta plataforma adjunta es gratis y permite disponer del conjunto de problemas del SPOJ. Además permite crear y gestionar fácilmente sus cursos, establecer cuáles de los estudiantes está autorizado a participar en sus cursos, así como el acceso por los mismos a miles de tareas de la plataforma spoj.pl. En la misma, las clases pueden incluir tanto grupos de problemas como las conferencias y con ayuda de la tecnología del motor Spoj es posible la clasificación automática de los programas para estudiantes.

SPOX no sólo comprueba la exactitud de las soluciones de los estudiantes, también verifica la calidad de su trabajo y es totalmente objetiva. Entre las deficiencias que se encuentran al analizar esta plataforma se encuentran que la misma implica contar con acceso internet y la limitación de contar solo con los problemas ya existentes. Además no implementa ningún mecanismo para la detección de plagio ni la orientación a los estudiantes como mismo ocurre con el juez en línea SPOJ. (8)

YoungCoder[5] una innovación en e-learning

Es otra solución relevante en el proceso de enseñanza y aprendizaje de la Programación,

[4] *http://spox.spoj.pl/store/*

[5] www.youngcoder.eu

específicamente de los lenguajes C++ y Java, que oferta tutoriales y varios problemas de programación, incluyendo un juez en línea para evaluar las soluciones. Este es un sistema de gestión de contenidos que permite seleccionar los materiales que se adapten al programa de estudios de los estudiantes. Un maestro puede subir y editar sus propios materiales y añadirlos a una lección preparada previamente. El sistema puede ser empleado por profesores para enseñar a sus estudiantes o bien por los mismos estudiantes que de manera individual les guste aprender algoritmos o lenguajes de programación desde el principio. La plataforma está equipada con un conjunto de problemas de programación que los usuarios tienen que resolver de forma individual. Todas las soluciones son enviadas a un servidor para ser revisadas por el juez en línea, donde este proporciona retroalimentación y un resultado, unos segundos más tarde. El sistema comprueba si un determinado programa es ejecutable y verifica la calidad de la solución. La verificación automática de soluciones posibilita la preparación de exámenes o concursos, así como también la publicación de los resultados obtenidos en estos. (19) A pesar de sus múltiples ventajas tiene el importante inconveniente de que no es gratis, no incluye detección de plagio ni orientación a los estudiantes. (8)

1.2.2 Ejemplos de soluciones similares en Cuba

En el contexto nacional, como en el resto del mundo, también se han desarrollado hasta la fecha algunos proyectos que tributan a la integración de sistemas. Algunos de los más destacados por sus logros y resultados obtenidos en tal sentido se muestran a continuación.

Sistemas de interacción por correo a los jueces en línea de programación: COJMail v1.0

Fue desarrollado en la UCI en el año 2011. Sistema capaz de establecer comunicación y consumir los servicios del Juez en Línea Caribeño a través de correo electrónico. Tuvo como objetivo lograr que desde las instituciones con redes de poco ancho de banda o limitados en su acceso pudiesen interactuar a través de correo electrónico con el COJ

obteniendo resultados semejantes a si estuviesen accediendo al mismo vía web. (4)

El principal problema de esta primera versión del COJMail es que no es compatible con la arquitectura actual que presenta el COJ, razón por la cual no está en condiciones de realizar prestaciones para este juez en línea. Desde el punto de vista funcional, el COJMail 1.0 no brinda soporte a los servicios de registrar usuario, comparar los resultados de los usuarios en el Archivo 24 horas, concursos reales y virtuales. Además, en caso de que el usuario especifique el micro lenguaje de forma incorrecta, el COJMail 1.0 envía la ayuda general del sistema, cuando lo ideal sería enviar una ayuda contextual que informe adecuadamente al usuario sobre los errores exactos que cometió escribiendo el micro lenguaje. (4)

Estrategia de interoperabilidad para la transferencia de datos entre sistemas ERP en Cuba (en lo adelante EITDESERPC)

En la investigación llevada a cabo por (Nogales, y otros, 2011) proponen un proceso de interoperabilidad que garantice el intercambio de información entre los sistemas integrales de gestión actualmente desplegados en Cuba y el sistema integral de gestión CEDRUX, como escenario de aplicación de la propuesta.

Desde el punto de vista técnico, la solución se da en varios aspectos, como la construcción del ambiente de configuración para la gestión de los procesos a interoperar, donde se adicionan, modifican, eliminan, y se define la estructura que finalmente puede tener el estándar de intercambio de información. También se observa el desarrollo de un componente donde se gestiona el trabajo con XML y demás estándares que se utilicen, (IDABC, 2009), el cual es usado por los desarrolladores para organizar los elementos que forman las taxonomías. Otra característica de la solución es la creación de un ambiente de interoperabilidad, que cuenta con las funciones Exportar e Importar información, donde el usuario interactúa directamente con la plataforma y ordena la interoperabilidad del proceso que desee. (20)

1.2.3 Resumen sobre análisis de las herramientas similares existentes

La comparación realizada por el equipo de desarrollo, entre los sistemas integrados analizados a nivel internacional y nacional quedó registrada en el Anexo 1 y permitió constatar que las que más se asemejan a lo que se quiere implementar son EduJudge y SPOX, pero no es recomendable su uso debido a que no se adaptan a las necesidades de la institución a la cual va dirigida el presente trabajo de diploma. Por tanto, se hace necesario la implementación de un componente que permita satisfacer las necesidades de la institución, además de tener en cuenta el desempeño de herramientas de aprendizaje informal como los jueces en línea, en una plataforma educativa de aprendizaje formal.

1.3 Metodología de desarrollo de software

Las metodologías de desarrollo de software abarcan todo el ciclo de vida del mismo, y se definen como *"*un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software*"* (Castellanos, 2009). (21) El objetivo de estas metodologías es guiar a un equipo de proyecto durante la creación de un nuevo software. Estas se clasifican en dos grandes grupos:

Metodologías tradicionales: orientadas al control de los procesos; estableciendo rigurosamente las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán.

Metodologías ágiles: Las orientadas a la interacción con el cliente y el desarrollo incremental del software; mostrando versiones parcialmente funcionales del software al cliente en determinados intervalos de tiempo, para que pueda evaluar y sugerir cambios en el producto.

En el Anexo 2 se muestran claramente las diferencias entre metodologías ágiles y tradicionales, que no se refieren solo al proceso en sí, sino también al contexto de equipo y organización que es más favorable a cada uno de estas filosofías de procesos de desarrollo de software. (22)

Capítulo 1 Fundamentación Teórica

Con el empleo de las metodologías se realiza el seguimiento de uno o varios modelos del ciclo de vida del proceso de desarrollo de los sistemas, facilitando en gran medida el éxito en las entregas y la calidad del producto. Se elige una determinada metodología en dependencia de las características del entorno o proyecto en cuestión, pues no existe una metodología universal para todos los casos. (21)

Características de la metodología a utilizar

Existen varias metodologías fundamentalmente con enfoque ágil utilizadas en el mundo para el proceso de desarrollo de software. Algunas de ellas son Programación Extrema o *Extreme Programming* (XP), *Microsoft Solution Framework* (MSF) y Proceso Unificado Ágil o *Agile Unified Process* (AUP).

Desde hace algún tiempo la UCI decidió hacer una variación de la metodología AUP (en lo adelante AUP-UCI), de forma tal que se adaptara al ciclo de vida definido para la actividad productiva de la institución. La adaptación de AUP logra estandarizar el proceso de desarrollo de software, dando cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3. A continuación, se relacionan algunas de las variaciones que sufre la metodología AUP.

- De las 4 fases que propone AUP (Inicio, Elaboración, Construcción y Transición), AUP-UCI mantiene la fase de Inicio, pero modificando el objetivo de la misma, unifica las restantes 3 fases de AUP en una sola llamada Ejecución y agrega la fase de Cierre.
- La metodología AUP abarca siete disciplina o flujos de trabajos, cuatro ingenieriles y tres de apoyo: Modelado, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de Proyectos y Ambiente. Por otra parte, AUP-UCI cuenta con 8 disciplinas, pero a un nivel más atómico que el definido en AUP. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran como disciplinas. Se mantiene la disciplina Implementación, en el caso de Prueba se desagrega en 3

disciplinas: Pruebas Internas, de Liberación y Aceptación y la disciplina Despliegue se considera opcional. Las restantes 3 disciplinas de AUP asociadas a la parte de gestión para la variación UCI serían Gestión de la Configuración, Planeación de proyecto y Monitoreo y control de proyecto.

- AUP propone 9 roles, en tanto AUP-UCI define 11 manteniendo algunos de los propuestos por AUP y unificando o agregando otros. Además, para modelar el sistema en los proyectos surgen 4 escenarios para la disciplina Requisitos.

Fundamentación de la metodología a utilizar

Con el objetivo de lograr la estandarización de la documentación almacenada en los centros productivos de la universidad, el equipo de trabajo decide utilizar para el desarrollo del componente la metodología AUP-UCI, pues esta metodología facilita el trabajo en proyectos de pequeña envergadura y proporciona un ambiente de desarrollo de software iterativo e incremental. En AUP-UCI sólo se utilizan los artefactos que son imprescindibles y realmente necesarios para la realización del producto. Además, describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio adoptando muchas de las técnicas ágiles de XP y conceptos que aún se mantienen válidos en RUP. Esta metodología aplica técnicas ágiles incluyendo: desarrollo dirigido por pruebas, modelado ágil, gestión de cambios ágil y refactorización de base de datos para mejorar la productividad. (23) Y además, fue concebida y está preparada para cambios durante el proyecto. De igual forma AUP-UCI es la metodología empleada por los miembros del equipo de desarrollo del proyecto P-MOOC al cual va dirigido el presente trabajo de diploma.

1.4 Herramientas y tecnologías asociadas al desarrollo del componente

A continuación se da una relación de las principales herramientas y tecnologías que se emplearán durante el desarrollo del componente así como la fundamentación de la elección realizada.

1.4.1 Sistema Gestor de Bases de Datos

Un **Sistema Gestor de Base de Datos (SGBD)** es un sistema de software que permite la definición de base de datos, el almacenamiento, manipulación y consulta de los datos que pertenecen a la misma. En estos sistemas se proporciona un conjunto coordinado de programas, procedimientos y lenguajes, que permiten a los usuarios realizar las tareas habituales con los datos y les garantiza la seguridad de los mismos. (24)

De acuerdo al componente que se desea desarrollar en el presente trabajo se decide utilizar el siguiente concepto encontrado en otra fuente bibliográfica, partiendo de este se seleccionará qué SGBD utilizar:

El **Sistema Gestor de Bases de Datos o SGBD**, también llamado DBMS (Data Base Management System) se define como una colección de datos relacionados entre sí, estructurados y organizados, y un conjunto de programas que acceden y gestionan esos datos. La colección de esos datos se denomina Base de Datos o BD, (las siglas serían DB por sus siglas en inglés Data Base). (25)

Fundamentación del SGBD a utilizar

En el desarrollo de aplicaciones web son empleados varios sistemas gestores de base de dato, tal es el caso de Microsoft SQL Server, MySQL y PostgreSQL.

PostgreSQL

Gestor de base de datos desarrollado por *PostgreSQL Global Development Group* (PGDG), es multiplataforma y posee una licencia libre permisiva. Permite no solo la implementación de bases de datos relacionales, también simula las multidimensionales. Se puede manejar mediante un cliente de entorno gráfico PGAdmin o mediante un cliente en forma de terminal de texto llamado PSQL, el cual es muy efectivo para servidores con pocos recursos de hardware. (26) Además, puede ser usado perfectamente por sistemas donde exista alta concurrencia de usuarios accediendo y donde se manejen grandes volúmenes de datos.

Principales aspectos

- Incluye varios métodos para el manejo de índices.
- Es multiplataforma y permite a los usuarios definir su propio tipo de datos, así como la ejecución de consultas complejas, consultas sobre vistas, subconsultas y uniones de gran tamaño.
- Utiliza nativamente el lenguaje procedural PL/PGSQL para implementar funciones, además de soportar otros, tales como: C, C++, Java PL, Java web y PL/PHP.
- Utiliza un método de almacenamiento llamado Control de Concurrencia para Múltiples Versiones (MVCC) que permite la eficiencia en ambientes donde se manejan grandes volúmenes de datos.

Para el desarrollo del trabajo de diploma se decidió el uso del gestor PostgreSQL en su versión 9.0, por sus características mencionadas con anterioridad y debido a que se caracteriza por su estabilidad, potencia, robustez, facilidad de administración e implementación de estándares, además de poseer una gran documentación en español en su sitio web oficial y foros de internet. Es el motor de base de datos de código abierto más potente del momento. También es un software con licencia libre, es utilizado en el proyecto al cual va dirigido el presente trabajo de diploma y es objeto de estudio en la asignatura de Bases de Datos lo que le permite al equipo de desarrollo tener conocimiento y dominio del uso de dicha herramienta.

1.4.2 Lenguajes de Programación

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que consiguen ser llevados a cabo por la computadora. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión o modo de comunicación humana. (27)

Fundamentación de los lenguajes de programación a utilizar

Para el desarrollo de aplicaciones web existe un gran número de lenguajes de

programación, divididos en dos grupos, lenguajes del lado del cliente y lenguajes del lado del servidor.

Lenguajes de programación del lado del cliente

Los lenguajes de programación del lado del cliente son aquellos que se utilizan para la creación de páginas web y son totalmente independientes del servidor. Son muy utilizados en la capa de presentación o las vistas de los usuarios. (28)

Para lograr que el componente tenga un aspecto uniforme se construirán un grupo de plantillas utilizando los lenguajes **HTML 5**, **CSS 3** y **JavaScript**. Estas plantillas constituyen temas de diseño que permiten definir la estructura, formato de las páginas y establecen el maquetado de las vistas de la aplicación permitiendo una mejor organización del componente y crear vistas más agradables al usuario. (29)

JavaScript: Lenguaje de programación utilizado principalmente para crear páginas web dinámicas, que son las que incorporan efectos tales como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario, entre otras. JavaScript es un lenguaje interpretado, por lo que no es necesario compilar programas para ejecutarlo. (30)

CSS v3: Lenguaje de hojas de estilos, creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Es la mejor forma de separar los contenidos y su presentación, y es indispensable para crear páginas web complejas. (31)

HTML v5: Es un lenguaje de composición de documentos que suele ser utilizado por la World Wide Web para publicar información en la web. Define la sintaxis y coloca instrucciones especiales que no muestra el navegador; aunque sí le indica cómo desplegar el contenido del documento que incluya textos, imágenes y otros medios soportados. (32) HTML es un lenguaje muy fácil de comprender y muy utilizado para la presentación de información. En resumen, permite definir la estructura y el contenido de las páginas, permitiendo combinar textos, imágenes, sonidos, vídeos y enlaces a otras páginas.

Lenguajes de programación del lado del servidor

Los lenguajes de programación del lado del servidor los reconocen, ejecuta e interpreta el propio servidor y se envían al cliente en un formato comprensible para él. Son utilizados en la capa de negocio, la cual se encarga de la implementación de los servicios o funcionalidades que el sistema proveerá al usuario. Se usan para acceder a recursos del servidor como base de datos y generación de contenido dinámico para las páginas. (28)

Para el desarrollo de la solución se utilizará como lenguaje de programación del lado del servidor el mismo que es utilizado en el proyecto al cual va dirigido el presente trabajo de diploma, PHP.

PHP v5.3x que es seguro y confiable debido a que el código fuente es invisible al navegador y al cliente. También es un software totalmente libre, gratuito, abierto y multiplataforma; tiene la capacidad de expandir su potencial con la utilización de gran cantidad de módulos y posee una amplia documentación en su página oficial. Además, el equipo de trabajo cuenta con conocimientos sobre estos lenguajes y ha desarrollado aplicaciones escritas en los mismos.

1.4.3 *Framework* de programación

Los marcos de trabajo o frameworks definen una arquitectura adaptada a las particularidades de un determinado dominio de aplicación, precisa de forma abstracta una serie de componentes, sus interfaces, establecimiento de las reglas y mecanismos de interacción entre ellos. Una alternativa es verlos como el esqueleto de una aplicación que debe ser adaptado por el programador según sus necesidades concretas. (33)

La utilización de marcos de trabajo proporciona grandes ventajas, ya que son el grado más alto de reutilización dentro del desarrollo de software, mejoran la calidad, están intrínsecamente unidos a los componentes, proporcionan la funcionalidad de los mismos y permiten la composición entre ellos de forma consistente.

Fundamentación del *framework* de programación a utilizar

Entre los marcos de trabajos más empleados se encuentran *JQuery, Doctrine, Bootstrap* y *Symfony.* Para el desarrollo de la solución serán utilizados los marcos de trabajo *JQuery* v2.0 y *JQuery UI* v1.10.0 para el trabajo con el lenguaje *JavaScript, Bootstrap* v3.0 para el trabajo con CSS y como framework PHP se utilizará *Symfony* v2.7. La elección de estos marcos de trabajo viene dada por sus características y porque son los utilizados en el proyecto al cual va dirigido el presente trabajo de diploma.

***Symfony* v2.7**

Es un framework diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón modelo vista controlador. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja, automatiza las tareas más comunes y permite al desarrollador, dedicarse por completo a los aspectos específicos de cada aplicación. Es multiplataforma y compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Symfony además es:

- Fácil de instalar y configurar en la mayoría de plataformas y con la garantía de que funciona correctamente en los sistemas Windows y Unix estándares.
- Independiente del sistema gestor de bases de datos y sencillo de usar en la mayoría de los casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo. (34)

***Bootstrap* v3.0**

Es un framework que podemos utilizar como base para crear sitios o aplicaciones web, está diseñado pensando en ofrecer la mejor experiencia de usuario tanto a usuarios de computadoras, como a teléfonos inteligentes y tabletas. Estas son algunas de sus ventajas:

- Permite ahorrar tiempo (No tenemos que empezar una página desde cero, sino que podemos pararnos sobre el código que nos aporta y empezar a desarrollar desde ahí.)
- Utiliza componentes y servicios creados por la comunidad web, maneja un conjunto de buenas prácticas que perdurarán en el tiempo y, emplea HTML5 y CSS3.
- Es una herramienta sencilla y ágil para construir sitios web e interfaces y se organiza por módulos independientes y reutilizables. (35)

***JQuery* v2.0**

Al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio. Entre sus características se destacan:

- Selección de elementos DOM[6] y soporte de extensiones.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3.
- Eventos, efectos y animaciones, animaciones personalizadas, manipulación de la hoja de estilos CSS.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, y funciones para rutinas comunes.
- Compatible con los navegadores Mozilla Firefox 2.0x, Internet Explorer 6x, Safari 3x, Opera 10.6x y Google Chrome 8x. (36)
- Permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX.

---

[6] Modelo de Objetos del Documento o Modelo en Objetos para la Representación de Documentos, siglas en inglés de Document Object Model.

Es la misma jQuery que se conoce más algunas características adicionales, pero la diferencia es que la biblioteca es nueva, construida desde cero. Fue construido con los navegadores actuales y futuros en la mente sin el apoyo de los navegadores más antiguos. (37)

***JQuery* UI v1.10.0**

Es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Además, la licencia open source de jQuery permite que la librería siempre cuente con soporte constante y rápido, publicándose actualizaciones de manera constante. La comunidad jQuery es activa y sumamente trabajadora. En resumen este *framework* fue seleccionado porque entre otras cosas: es flexible y rápido para el desarrollo web, es *open source*, tiene plugins y una excelente comunidad de soporte, además los bugs son resueltos rápidamente y posee excelente integración con AJAX. (38)

1.4.4 Entorno de desarrollo integrado

Un entorno de desarrollo integrado (IDE por sus siglas en inglés) es una herramienta informática que aporta funcionalidades al desarrollador durante todas las etapas del ciclo de vida del desarrollo de software, desde el análisis y diseño a la distribución del producto y su mantenimiento, de ahí el término "integrado". En resumen un IDE es una interfaz visual que permite al desarrollador trabajar con comodidad, se encarga de dialogar internamente con el compilador, el enlazador y demás herramientas. (39)

Fundamentación del IDE a utilizar

Existen muchos IDE de múltiples lenguajes tales como Eclipse, ActiveState Komodo, IntelliJ IDEA, MyEclipse, Oracle JDeveloper, NetBeans, PhpStorm, Codenvy y Microsoft Visual Studio.

PhpStorm

PhpStorm es un IDE de programación desarrollado por *JetBrains*, empresa de desarrollo de software cuyas herramientas están dirigidos hacia los desarrolladores de software y administradores de proyectos. Este IDE tiene como propósito ayudar a mejorar la calidad de nuestro código. Es una herramienta rápida teniendo en cuenta su tamaño, soporta muchos idiomas y marcos a través de plugins, además, es multiplataforma. Es compatible con *Windows*, *Linux* y *Mac OS X*. PhpStorm se destaca por la ejecución del código en la misma interfaz del IDE. Este editor permite además la gestión de proyectos fácilmente, proporciona un fácil autocompletado de código, soporta el trabajo con PHP 5.5 y posee una sintaxis abreviada.(40)

Para el desarrollo de la solución se hará uso del IDE PhpStorm v8.0 debido a que es compatible con los principales marcos de trabajo, o sea, es perfecto para trabajar con Symfony, Drupal, WordPress, Zend Framework, Laravel, Magento, CakePHP y Yii. Además soporta diferentes sintaxis del lenguaje PHP, pero al mismo tiempo también es compatible con HTML, CSS y Java script. Por lo tanto, proporciona un entorno impresionante para una amplia gama de lenguajes de *scripting* que un desarrollador PHP puede necesitar durante el desarrollo de una aplicación web estándar. A su vez, proporciona un soporte completo para todas estas secuencias de comandos con la terminación inteligente de código, la inspección de código, la detección de errores y depuración inteligente para su aplicación web. (40) Es importante destacar que la elección de este entorno de desarrollo está sustentada fundamentalmente en que este es el utilizado en el proyecto al cual va dirigido la presente investigación, además de que los miembros del equipo de desarrollo tienen experiencia en el trabajo con este IDE y también se cuenta con la licencia para su uso.

1.4.5 Servidor WEB

Un servidor web es un programa que implementa el protocolo HTTP[7] , que está diseñado para transferir hipertextos, páginas web o páginas HTML. Es un programa que se ejecuta continuamente en un ordenador, se mantiene a la espera de peticiones por parte de un

7 Hypertext Transfer Protocol, en español Protocolo de Transferencia de Hipertexto

cliente (navegador web) y responde o ejecuta estas peticiones ya sea mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje, si detectó algún error.

Fundamentación del servidor web a utilizar

Existen varios servidores web pero Apache2 es uno de los más utilizados para el trabajo con PHP.

Servidor HTTP Apache v2.4.7

Es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Apache es usado primariamente para enviar páginas web estáticas y dinámicas en la *World Wide Web*. Además de ser altamente configurable, cuenta con base de datos de autenticación y su arquitectura es modular, gracias a lo cual se han desarrollado diversas extensiones entre las que se destaca PHP. Según estadísticas de Netcraft[8] el mayor por ciento de los servidores web actuales son servidores Apache. (41)

Para el desarrollo de la solución planteada el equipo de trabajo decide utilizar Apache v 2.4.7 debido a que es modular, multiplataforma, extensible, de código abierto, uso gratuito, muy robusto, destaca por su seguridad y rendimiento, y también es popular, o sea, es fácil de conseguir ayuda y soporte. (42) Además porque este servidor está recomendado para Symfony y es el usado por los miembros del proyecto al cual va dirigido este trabajo de diploma.

1.4.6 Lenguaje de modelado

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 este. (43) Al estar estandarizado permite a los desarrolladores documentar el software en cuanto a

8 Empresa dedicada a la realización de encuestas y estudios sobre el tráfico en internet.

funcionalidades, procesos de negocios y conceptos asociados. Entre estos lenguajes se destacan el BPM y el UML.

Fundamentación del lenguaje de modelado seleccionado

UML: (*Unified Modeling Language* o Lenguaje unificado de modelado), posee notaciones estandarizadas que permiten la construcción de diagramas y artefactos tanto en el modelado de análisis y diseño del sistema. Esta versión actual simplifica en gran medida la representación de los conceptos, facilitando su uso y la comunicación entre los involucrados en el proyecto. UML de manera general permite visualizar, especificar, construir y documentar un sistema; además de ofrecer un estándar para describir el modelo, incluye aspectos conceptuales como procesos de negocio, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación y esquemas de bases de datos.

Para la modelación de la solución se propone el uso de UML en su versión 2.0 por sus características y ya que es el lenguaje usado para la modelación de los procesos de negocio utilizado en el proyecto al cual va dirigido el presente trabajo de diploma.

1.4.7 Herramienta CASE

Para el uso de un lenguaje de modelado existen varias herramientas CASE[9] (Ingeniería de Software Asistida por Computadoras), estas son aplicaciones informáticas que traen grandes impactos al proyecto al reducir de forma significativa costes y tiempo de producción al trabajar directamente en cada fase del ciclo de vida de desarrollo del software. Su naturaleza consiste en conseguir una mejora en la productividad y en la calidad del producto final, reducir mantenimiento de los sistemas informáticos y facilitar el uso de las distintas metodologías propias de la ingeniería del software. (44)

Análisis de la herramienta CASE Visual Paradigm

9 Acrónimo de Computer Aided Software Engineering

Visual Paradigm: Herramienta CASE que utiliza UML como lenguaje de modelado. Provee una estructura de proyecto jerarquizada donde se pueden modelar las diferentes perspectivas del sistema y sus artefactos en cada fase. Incluye herramientas de animación para simular el funcionamiento del modelado que se construye. Permite por otra parte: el análisis de impacto, administración de tareas, diseño colaborativo y generación de código en múltiples lenguajes a partir de clases. (45)

Fundamentación de la herramienta CASE seleccionada

A partir del estudio realizado se decide que para el desarrollo de la solución se debe emplear Visual Paradigm en su versión 8 debido a que es multiplataforma, cumple con las políticas de migración a software libre en Cuba, proporciona excelentes facilidades de interoperabilidad con otras aplicaciones, soporta el ciclo de vida completo del desarrollo de software, lo que facilita la captura de requisitos, análisis, diseño e implementación, y también la UCI tiene licencia para su uso. También permite dibujar todos los tipos de diagramas de clases y generar código a partir de los diagramas, posee generador de informes en formato PDF y HTML, además de una interfaz agradable e intuitiva. A su vez es la herramienta CASE que es objeto de estudio de la asignatura Ingeniería de software impartida en la carrera, por lo que los miembros del equipo de desarrollo tienen experiencia en su manejo y utilización.

1.5 Patrones de arquitectura

Antes de hablar de patrones primero se debe centrar el análisis en el concepto de arquitectura de software, a partir de este se tendrá una visión más clara, para llegar a la definición de patrón de arquitectura y cómo emplearla en el desarrollo de la solución.

La **arquitectura de software** de un sistema de programa o computación es la estructura de las estructuras del sistema, la cual comprende los componentes del software, las propiedades de esos componentes visibles externamente y las relaciones entre ellos. (46)

El proceso de arquitectura de software toma los requisitos de los clientes, los analiza y produce un diseño para obtener un software que satisfará sus necesidades. De manera

general se puede decir que este proceso constituye un mapeo entre lo que un software debe lograr y los detalles del código o implementación del mismo. Al obtener la arquitectura correcta se garantizará que los requisitos y los resultados coincidan de manera eficaz.

Patrón de arquitectura: Descripción de un problema particular y recurrente de diseño, que aparece en contextos de diseño específico y presenta un esquema genérico demostrado con éxito para su solución. El esquema de solución se especifica mediante la descripción de los componentes que la constituyen, sus responsabilidades y desarrollos, así como también la forma de cómo estos colaboran entre sí. (47)

Ejemplos de patrones arquitectónicos incluyen los siguientes: programación por capas, arquitectura en pizarra, arquitectura dirigida por eventos, arquitectura orientada a servicios y Modelo Vista Controlador.

Fundamentación del patrón seleccionado

Para el desarrollo de la solución planteada el equipo de trabajo seleccionó el patrón Modelo Vista Controlador (MVC) que es el utilizado por el framework symfony seleccionado para el desarrollo de las funcionalidades. Además, la elección se fundamenta en lo siguiente:

- Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos:

  Vista: Es implementada en Java Script o HTML y reside del lado del cliente en tiempo de ejecución. Permite mostrar la información del modelo en un formato adecuado que posibilite que se dé la interacción. Además de poseer un registro acerca del controlador asociado y brinda el servicio de actualización que puede ser usado tanto por el controlador como por el modelo. (Buschmann, y otros, 2007) Es la que permite de manera general, mostrar la información al usuario y realizar las correspondientes validaciones una vez terminado el producto.

  Modelo: Contiene los datos y las funcionalidades esenciales. Es la representación

específica de la información con que opera el sistema, es decir la representación de los datos y reglas del negocio. Es el encargado de manejar un registro de las vistas y de los controladores que existen en el sistema asegurando la integridad y disponibilidad de los datos (Buschmann, y otros, 2007).

Controlador: Funciona como intermediario entre la vista y el modelo de datos y se encarga de todo el comportamiento lógico del sistema. Es el responsable, entre otros aspectos, de recibir los eventos de entrada desde la vista. Responde a los eventos provocados por el usuario que implican cambios en el modelo y la vista, dando una correcta gestión a las entradas de usuario. (Buschmann, y otros, 2007)

El Controlador y el Modelo juntos residen del lado del servidor**.**

- Entre sus características se destacan la sencillez para crear distintas representaciones de los mismos datos, la facilidad para la realización de pruebas unitarias de los componentes y para desarrollar prototipos rápidos, y la reutilización de los componentes. (48)

En el Anexo 3 se describe claramente el funcionamiento del patrón de arquitectura MVC.

1.6 Conclusiones del capítulo

Con el estudio realizado se puede constatar que la Facultad 4 de la UCI en su estado actual no suple las necesidades de los profesores de contar con un componente que integre el COJ y la plataforma educativa Zera 2.0 para facilitar íntegramente el acceso de los mismos a todos los contenidos relacionados con su asignatura, así como la gestión de las actividades realizadas por sus estudiantes en el COJ. Para llevar a cabo la solución del problema se propone el desarrollo de un componente, con el objetivo de satisfacer las especificaciones requeridas por el cliente. Para desarrollar dicha propuesta se definió el uso de la metodología AUP la cual guiará el proceso de desarrollo de software y para su construcción se han elegido los framework *JQuery* v2.0, *JQuery UI* v1.10.0*, Bootstrap* v3.0 y como framework PHP se utilizará *Symfony* v2.7 sobre el entorno de desarrollo PhpStorm v8.0, el cual trabajará en paralelo con el servidor Apache v2.4.7. Como gestor de base de

datos se empleará PostgreSQL en su versión 9.0 y como herramienta de modelado Visual Paradigm 8.0 utilizando el lenguaje de modelado UML en su versión 2.0 y basándose en el patrón Modelo Vista Controlador. Por otra parte, para lograr que el componente tenga un aspecto uniforme se construirán un grupo de plantillas utilizando los lenguajes HTML5, CSS3 y JavaScript, y para el desarrollo de la solución se empleará PHP en una versión superior a 5.3.

Capítulo 2: Análisis y diseño

En el desarrollo de un software, es de vital importancia definir los procesos que intervienen en este para lograr así un mejor entendimiento del sistema a desarrollar entre clientes y desarrolladores. En este capítulo se exponen como artefactos que permiten describir la propuesta de solución: modelo de dominio, la especificación de requisitos funcionales y no funcionales, las historias de usuario que incluyen a su vez los prototipos de interfaz de usuario, el modelo de análisis (donde se identifican las clases del análisis y se realizan los diagramas de clases y de colaboración) y el modelo de diseño (encargado de realizar los diagramas de clases y de secuencia con estereotipos web). Otro aspecto que se define son los patrones de diseño a utilizar, así como el modelo de entidad relación de la base de datos y el diagrama de despliegue.

2.1 Modelo de dominio

Es utilizado como base para analizar los requisitos del usuario en el flujo de la ingeniería del software (13) ya que permite observar los distintos conceptos que forman parte del dominio de interés, y modela la interacción de estos conceptos con la finalidad de obtener un entendimiento del negocio.

2.1.1 Conceptos del dominio

La correcta definición del modelo contribuirá a lograr una mejor comunicación entre usuarios, desarrolladores y clientes, mediante el establecimiento de un vocabulario común que permita entender el funcionamiento del sistema.

Plataforma educativa Zera 2.0: plataforma educativa nombrada destinada a apoyar el proceso de enseñanza-aprendizaje y que les permite a los distintos usuarios del sistema intervenir en un ambiente dinámico e interactivo.

Aplicación externa: herramienta externa a la plataforma creada con el fin de apoyar los procesos de enseñanza-aprendizaje, en este caso el COJ que proveerá los problemas

para la confección de las actividades.

Usuario: persona que interactúa con la plataforma. Existen dos tipos de usuarios: estudiantes y profesores. Ambos consumen recursos y servicios de la plataforma, aunque el profesor es el que actúa como administrador del sistema por lo que se de las actividades de gestión y del seguimiento de la actividad del estudiante.

Actividad: conjunto de operaciones o tareas que son asignadas por el profesor a un usuario y posteriormente este las realiza en un período de tiempo dado.

Actividad *single-method*: es un tipo de actividad donde el usuario debe escribir un método de java que previamente fue descrito. Es decir, no se debe escribir una clase java completa, sino los métodos que son descritos en el enunciado del problema que fue antes creado por el profesor.

2.1.2 Diagrama del modelo de dominio

El diagrama de la Figura 1 muestra la relación que existe entre los conceptos del dominio identificados y descritos previamente, para un mayor entendimiento del problema.

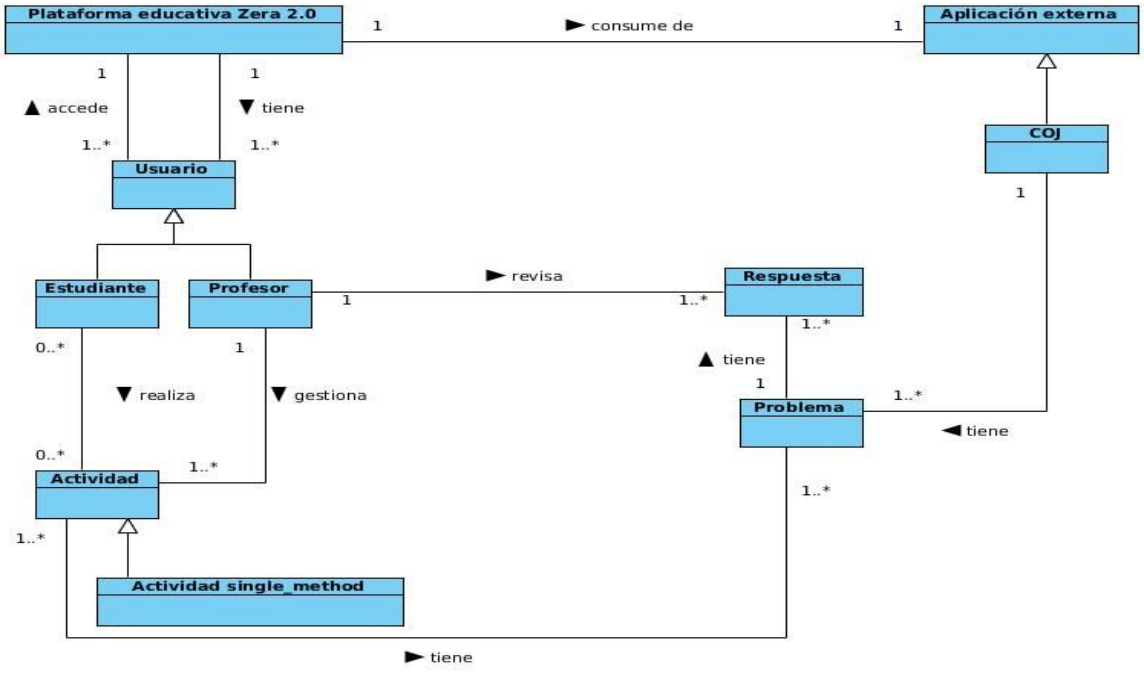

Figura 1. Modelo de dominio.

2.2 Descripción del sistema propuesto

El desarrollo del componente propuesto tiene como objetivo la interacción e intercambio de información entre la plataforma educativa Zera 2.0 y una aplicación externa como el COJ. Las funcionalidades propuestas se localizarán en el componente Conexcoj, componente encargado de agrupar la información y los contenidos específicos para que los profesores de programación puedan asignar actividades a los estudiantes de su grupo haciendo uso del COJ y luego puedan revisarlas y evaluar su desempeño. El acceso al mismo se realiza al ingresar a los hiperentornos con el rol estudiante o profesor, y se tiene acceso al componente a través del menú de un curso específico. Este componente les permitirá a los usuarios contar con una plataforma con recursos educativos de muy alta calidad para evaluar el aprendizaje en la disciplina de programación.

2.3 Requerimientos del sistema

La identificación de los requisitos de software permite definir las condiciones o capacidades que debe cumplir el sistema para satisfacer las necesidades del cliente. Estos se dividen en dos grupos: requisitos funcionales (en lo adelante RF) y requisitos no funcionales (en lo adelante RNF).

2.3.1 Requisitos funcionales

Los requisitos funcionales constituyen las capacidades o condiciones que el sistema debe cumplir. Estos deben ser comprensibles por los clientes, usuarios y desarrolladores, deben tener una sola interpretación y estar definidos en forma medible y verificable. (49) La identificación de requisitos funcionales permite satisfacer las necesidades reales del cliente logrando de esta forma un entendimiento entre los miembros del equipo de desarrollo y éste.

Para el correcto funcionamiento de la solución propuesta se espera que el sistema permita:

RF1. Autenticar usuario: Permite al usuario autenticarse en el sistema y acceder a la información almacenada según los permisos que le son asignados.

RF2. Ver el estado de un usuario: Permite al profesor ver el estado en que se encuentran los usuarios de su grupo mediante el vínculo mostrar estado.

RF3. Gestionar actividad:

RF3.1. Crear actividad: Permite crear una nueva actividad que podrá ser asignada, realizada y evaluada posteriormente. La misma estará conformada por problemas del COJ.

RF3.2. Modificar una actividad: Se brinda al usuario la posibilidad de modificar una actividad que será realizada por otro usuario determinado a través del vínculo editar. Debe existir al menos una actividad previamente creada.

RF3.3. Eliminar actividad: Se brinda al usuario la posibilidad de eliminar una actividad determinada, permitiéndose confirmar la eliminación. Debe existir al menos una actividad previamente creada.

RF3.4. Mostrar los datos de una actividad: Muestra los datos de la actividad que podrá ser realizada por los usuarios. Debe existir al menos una actividad previamente creada.

RF3.5. Crear actividad de tipo *single-method*: Permite crear una nueva actividad de tipo *single-method* que podrá ser asignada, realizada y evaluada posteriormente.

RF4. Buscar problemas en el COJ con opciones de filtrado: Se brinda al usuario la posibilidad de buscar a través de filtros uno o varios problemas en el COJ que conformarán las actividades que serán orientadas para el estudio individual de cada estudiante.

RF5. Enviar soluciones a tareas asignadas: Permite a los usuarios enviar la solución correspondiente a la actividad que le fue orientada. Tendrá dos intentos para el envío de la respuesta.

RF6. Buscar el estado de las respuestas enviadas con opciones de filtrado: Se brinda al usuario la posibilidad de buscar a través de filtros el estado de las respuestas enviadas por los estudiantes a las actividades que le fueron orientadas para el estudio individual.

RF7. Consultar problemas recomendados para un usuario: Permite a los usuarios consultar y realizar (de manera opcional) los problemas que el COJ le recomienda si ha tenido un buen desempeño en la realización de la actividad que le fue orientada.

RF8. Visualizar estado de las respuestas enviadas: Se brinda al estudiante la posibilidad de ver el estado en que se encuentran las respuestas que enviaron a las actividades que le fueron orientadas para el estudio individual.

2.3.2 Requisitos no funcionales

Requerimientos que permiten definir las cualidades o propiedades que el software debe tener, con el objetivo de crear un producto final que sea fácil de utilizar, rápido y confiable. A continuación, se listan los requisitos no funcionales identificados:

Software:

- El sistema debe visualizarse y ejecutarse correctamente en un navegador web moderno, especialmente en Firefox (v20.x en adelante) y Google Chrome (v26.x en adelante), que son dos de los navegadores que mejor funcionan con HTML5 y CSS3.

Hardware:

- El sistema debe ser instalado en un entorno con sistema operativo con: Distribución de CentOS[10] última versión estable.
- Los usuarios finales deberán contar con: Procesador Intel Pentium 4 o superior, 512 MB de memoria física o superior y 20 GB de HDD[11].
- Para un buen funcionamiento el servidor de aplicación web donde se ejecute el sistema debe contar con las siguientes características: Procesador Intel Core i3 o superior, Memoria física de 4Gb y 500Gb de disco duro.
- Servidor de bases de datos relacional PostgreSQL 9.4 con memoria física: 16 GB,

---

10 Acrónimo de Community ENTerprise Operating System , es un sistema operativo de código abierto cuyo objetivo es ofrecer al usuario un software de "clase empresarial" gratuito. Se define como robusto, estable y fácil de instalar y utilizar.

11 Dispositivo de Disco Duro, siglas en inglés de hard disk drive.

disco duro: 100 GB y Procesador Intel Core i3 o superior.

Apariencia o interfaz externa:

- El diseño de interfaz debe ser sencillo y fácil de usar, con una iconografía que represente la acción a realizar. Será formal, serio y con una navegación sugerente.
- El sistema proporcionará claridad y correcta organización de la información, permitiendo la interpretación correcta e inequívoca de ésta.

Seguridad:

- El sistema debe contar con diferentes niveles de acceso a la información almacenada para garantizar la protección de información de accesos no autorizados.
- Mantener el sistema disponible evitando que los mecanismos de seguridad impidan el acceso a la información requerida por los usuarios autorizados.

Usabilidad:

- El sistema podrá ser usado por personas que posean conocimientos básicos de informática y del manejo de ordenadores.
- Se debe mantener informado al usuario del resultado de las acciones realizadas.

Accesibilidad:

- El sistema debe estar disponible desde cualquier estación de trabajo conectada a la red y permitir su administración de forma remota.

2.4 Historias de usuario

Las historias de usuario (en lo adelante HU) son una técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. (50) Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla, en un tiempo estimado (en días) de desarrollo que

lo define el propio equipo del proyecto. (51)

En esencia, las HU son las ideas del cliente organizadas y agrupadas de acuerdo a su funcionalidad. A su vez, también se tiene en cuenta un orden tal que permita al mismo priorizar sus necesidades y al equipo de trabajo definir las que resultan críticas o claves en el momento de desarrollo de la solución. En la claridad de su descripción radica el éxito del proyecto.

A continuación, se muestra la historia de usuario del caso de uso Gestionar actividad, las restantes se describen en el Anexo 4 del presente documento:

Tabla 1. Historia de usuario del RF Gestionar actividad.

Historia de Usuario	
Número: 3	**Nombre del requisito:** Gestionar actividad
Programador: Reysel Pérez Avilés	**Iteración Asignada:** 1ra
Prioridad: Alta	**Tiempo Estimado:** 20 días
Riesgo en Desarrollo: Alto	**Tiempo Real:** N/A
Descripción: **1- Objetivo:** Permitir crear, modificar, eliminar o mostrar una actividad, así como también crear una actividad de tipo single-method en el sistema. **2- Acciones para lograr el objetivo (precondiciones y datos):** -Para crear o modificar una actividad se debe tener en cuenta los siguientes datos: título y descripción. -Para modificar, eliminar o mostrar una actividad debe existir al menos una previamente creada en el sistema. -Para crear cualquier tipo de actividad no se pueden dejar campos en blanco, hay que llenar	

todos los datos del formulario.

-El usuario debe estar autenticado en el sistema con el rol profesor.

3- Comportamientos válidos y no válidos(flujo central y alternos):

Título: campo de texto que admite caracteres alfabéticos y tiene un máximo de hasta 50 caracteres.

Descripción: campo de texto que permite cualquier carácter.

4- Flujo de la acción a realizar:

-En caso de que el usuario seleccione la opción de Crear una nueva actividad, el sistema le proveerá un formulario para que inserte los datos que se necesiten, permitiendo completar la operación o cancelarla. Cuando el usuario incluye correctamente los datos y selecciona la opción Crear, se crea una nueva actividad y el sistema muestra un mensaje de información. Si los datos están incompletos o incorrectos se señalarán los campos en cuestión dando la posibilidad al usuario de realizar nuevamente la acción en cuestión. Si selecciona la opción Cancelar regresará a la vista previa.

-Si el usuario selecciona la opción de Modificar una actividad, el sistema mostrará los datos que pueden ser editables dentro de ésta, y una vez que el usuario modifica de forma correcta los datos necesarios y selecciona la opción Actualizar, se guardan las modificaciones y el sistema muestra un mensaje de información de que la actividad fue modificada de forma correcta. Si los datos están incompletos o incorrectos se señalarán los campos en cuestión dando la posibilidad al usuario de realizar nuevamente la acción en cuestión. Si selecciona la opción Cancelar regresará a la vista previa.

- Si el usuario selecciona la opción Eliminar una actividad del sistema haciendo clic en el botón eliminar de dicha actividad, el sistema muestra un mensaje de confirmación para Aceptar o no la acción que se está realizando. En caso de seleccionar la opción Aceptar se actualiza el listado de actividades y el sistema muestra un mensaje de información.

- Para ver una actividad inicialmente se muestra al usuario un listado con las actividades que han sido incluidas en el sistema. Una vez seleccionada una actividad, podrá ver sus datos seleccionando la opción “Ver”.

- En caso de crear una actividad de tipo single-method al profesor se le muestra una pantalla diferente a la de los estudiantes pues él podrá ver la clase completa y el resto solo visualizará la sección donde corresponde el método a implementar. O sea este se encargará de entrar la entrada y salida estándar, así como la especificación de entrada para que el estudiante introduzca el método sin problema alguno y con las características que se le presentan con anterioridad.
Observaciones:
Prototipo de interfaz: 

2.5 Modelo de análisis

El modelo de análisis es la primera representación técnica de un sistema. Utiliza textos y diagramas para representar los requisitos del software, sus funciones y comportamiento, permitiendo así examinar los requisitos desde diferentes puntos de vista aumentando la probabilidad de encontrar errores y de que se descubran nuevas debilidades a tiempo. De manera general, el objetivo de este flujo de trabajo es generar una arquitectura de objetos que sirva como base para el diseño posterior del sistema. (52)

A continuación, se describe brevemente la clasificación de las clases del análisis que pueden ser identificadas en el dominio de un problema, representadas a través de tres

estereotipos básicos: interfaz, control y entidad.

Entidad: Modela la información que posee larga vida y que es a menudo persistente, en otras palabras, representa la información reflejada en el caso de uso.

Interfaz: Es la encargada de modelar la interacción entre el sistema y sus actores, lo que implica recibir y representar informaciones y peticiones de usuarios y de sistemas externos.

Control: Coordina la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso. (2)

Figura 2. Estereotipos para diagramas de clases del análisis.

2.5.1 Diagramas de clases del análisis

Los diagramas de clases del análisis (DCA en lo adelante) representan la relación entre las clases que intervienen en los casos de uso. Son utilizados para comprender de forma general la estructura del sistema y sirve como entrada para la etapa de diseño. (2)

A continuación se presenta el DCA de la historia de usuario Gestionar actividad. Para el estudio de los demás diagramas remitirse al Anexo 5.

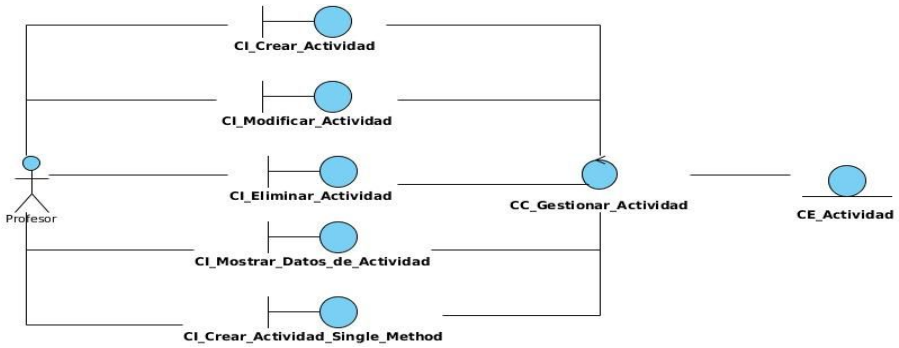

Figura 3. DCA_HU_Gestionar actividad.

2.5.2 Diagramas de colaboración del análisis

Los diagramas de colaboración del análisis (DC en lo adelante) son una forma de representar la interacción entre objetos. Además, muestran como las instancias específicas de las clases trabajan juntas para conseguir un objetivo común. (53)

La representación del DC de la sección Crear Actividad para la historia de usuario de Gestionar actividad se muestra a continuación. Los restantes diagramas se encuentran en el Anexo 6.

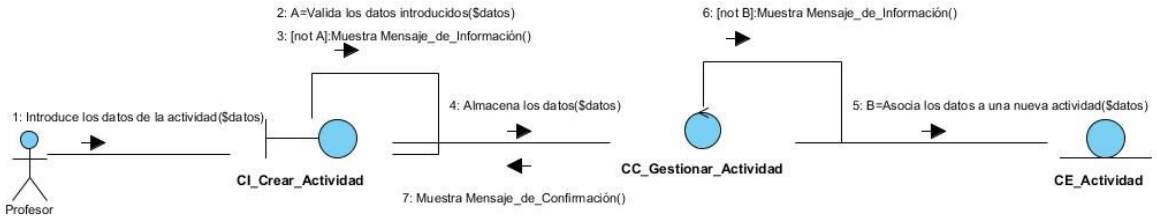

Figura 4. DC de la sección Crear Actividad del RF Gestionar actividad.

2.6 Patrones de diseño

Los patrones de diseño brindan solución a problemas comunes que pueden ser encontrados durante el diseño, perfeccionando los componentes de un sistema de software y sus relaciones. (54)

2.6.1 Patrones GRASP

Teniendo en cuenta las tecnologías y herramientas de desarrollo seleccionadas y el patrón de arquitectura antes elegido, se decide emplear para estructurar el diseño del sistema Patrones Generales de Software para Asignar Responsabilidades, más conocidos como patrones GRASP (General Responsibility Asignment Software Patterns, por sus siglas en inglés), los cuales serán muy útiles para lograr un software con alto grado de usabilidad.

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. (55) Dentro de los patrones GRASP utilizados para el diseño del sistema y que incluye por defecto en su

arquitectura el framework de desarrollo seleccionado se destacan los siguientes:

Experto: consiste en asignar una responsabilidad al experto en información, en otras palabras, se asigna la responsabilidad a la clase que cuenta con la información necesaria para cumplirla. (55) Se evidencia en las clases que extienden de la clase ***Entidad***, las cuales son expertas en su propia información, tales como ***Usuario*** *y* ***Actividad.***

Creador: permite crear objetos de una clase determinada. Es utilizado en la mayoría de las clases controladoras para crear instancias de formularios y entidades. (54) El empleo de este patrón será muy útil para aquellas clases encargadas de registrar datos de entidades que manejará el sistema, y este se evidencia en las clases: ***UserController*** y ***ActivityController,*** por otra parte, para la vista del usuario se encarga la clase ***DefaultController*****.**

Alta cohesión: consiste en asignar una responsabilidad de manera que la cohesión continúe siendo alta. (56) Este patrón se evidencia en las clases controladoras, en las se definieron una serie de funcionalidades que se relacionan entre sí, de modo que cada una de estas clases solo contenga los métodos correspondientes a su área funcional, evitando de esta manera que existan clases sobrecargadas de trabajo. (2) Esto hace posible que el software sea flexible a cambios sustanciales con efecto mínimo. Este patrón se evidencia en las clases ***UserController y ActivityController,*** aunque de forma general la alta cohesión se aplica en todo el sistema***.***

Controlador: se basa en asignar la responsabilidad de todos los eventos realizados a una clase específica que constituye el único punto de entrada para cada evento. (54) En la solución propuesta este patrón está evidenciado en las clases controladoras encargadas de realizar las operaciones de gestión sobre las entidades y en la clase ***ApicojController***; que permite el acceso a la información almacenada en la aplicación externa (COJ).

2.7 Modelo de diseño

El modelo de diseño, es una abstracción de la implementación del sistema que describe la realización física de los casos de uso, centrado en cómo los requisitos funcionales y no

funcionales impactan en el desarrollo de la aplicación. Representa todas las clases del diseño, subsistemas, paquetes, colaboraciones y las relaciones entre ellos, constituyendo la entrada principal a las actividades de la fase de implementación. (54)

2.7.1 Diagramas de clases del diseño

Los diagramas de clases del diseño (DCD en lo adelante) son una representación más concreta y detallada que los diagramas de clases del análisis, aunque también representan la parte estática del sistema conteniendo las clases y sus relaciones. Son empleados para representar las relaciones que se establecen entre las clases.

A continuación se describen de manera general el significado de los principales elementos presentes en los diagramas de clases del diseño para lograr una mejor comprensión de éstos. Es importante mencionar que los paquetes modelo, controlador y vista se usan para distribuir las clases según el patrón Modelo-Vista-Controlador que propone Symfony.

Paquete Vista: Contiene las clases que muestran la información al usuario como resultado de ejecutar una acción determinada.

Paquete Controlador: Contiene las clases encargadas de relacionar la lógica del negocio con la presentación.

Paquete Modelo: Contiene las entidades generadas en correspondencia con las tablas de la base de datos que almacenan toda la información que maneja la aplicación.

Paquete JavaScript: Contiene los clases JavaScript necesarias para las plantillas.
Paquete CSS: Contiene las clases CCS necesarias en las plantillas.

Paquete Componente Symfony: Contiene los componentes y configuraciones propias del framework.

Capítulo 2
Análisis y Diseño

Se presenta a continuación el diagrama de clases del diseño de la historia de usuario Gestionar Actividad. Para el estudio de los demás diagramas de clase del diseño remitirse al Anexo 7.

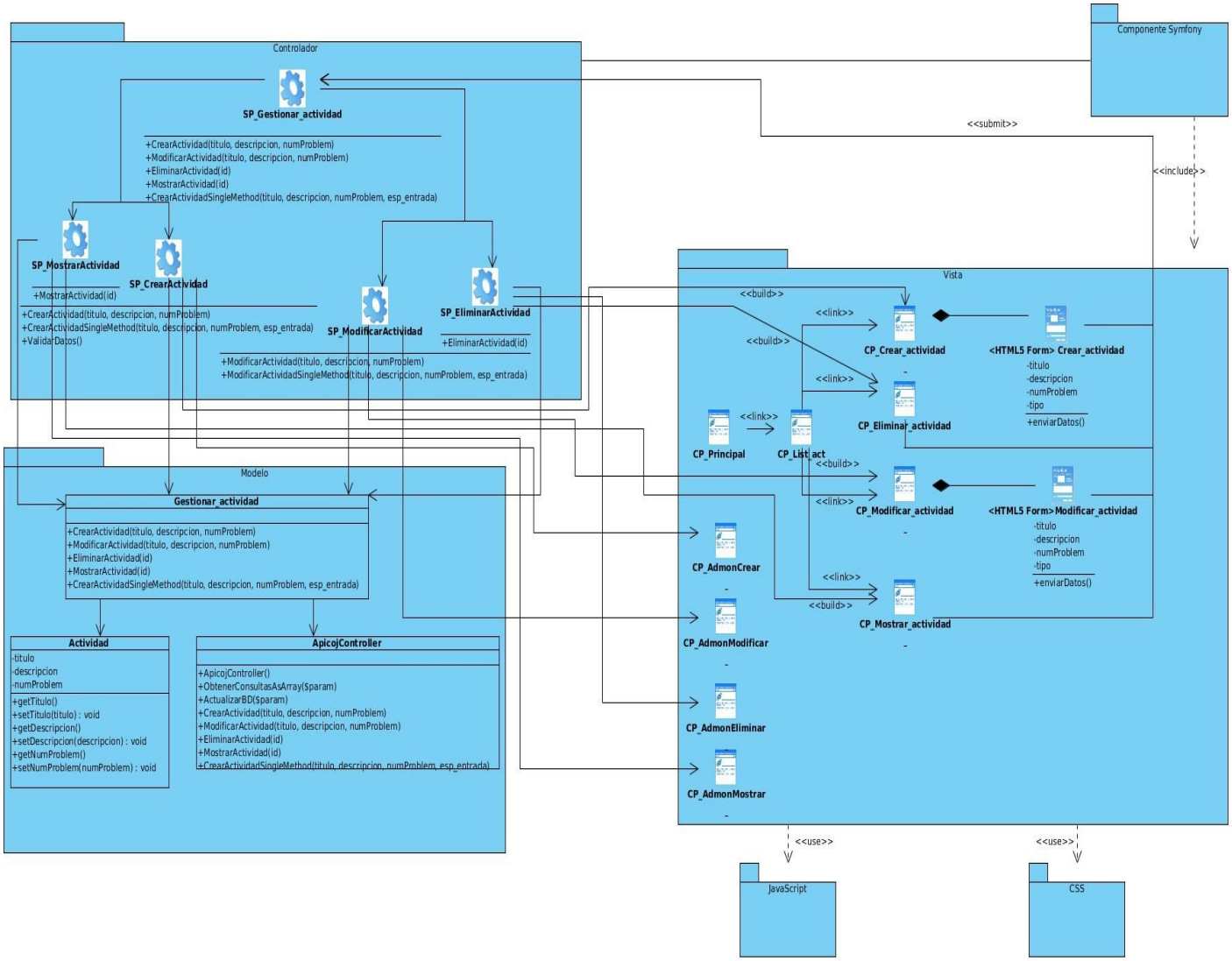

Figura 5. DCD_HU_Gestionar actividad.

2.7.2 Diagramas de secuencia del diseño

Los diagramas de secuencia del diseño (DSD en lo adelante) muestran una iteración ordenada según la secuencia temporal de los eventos, es decir muestran los objetos que interactúan y los mensajes que se intercambian ordenados según la secuencia de tiempo en que se realiza cada uno de ellos, quedando establecido para el eje vertical la representación del tiempo y en el horizontal es donde se colocan los objetos y actores participantes en la interacción. (57) En resumen, el diseño de este artefacto permitirá representar la interacción de los objetos que intervienen en una funcionalidad determinada, mediante la transferencia de mensajes o métodos.

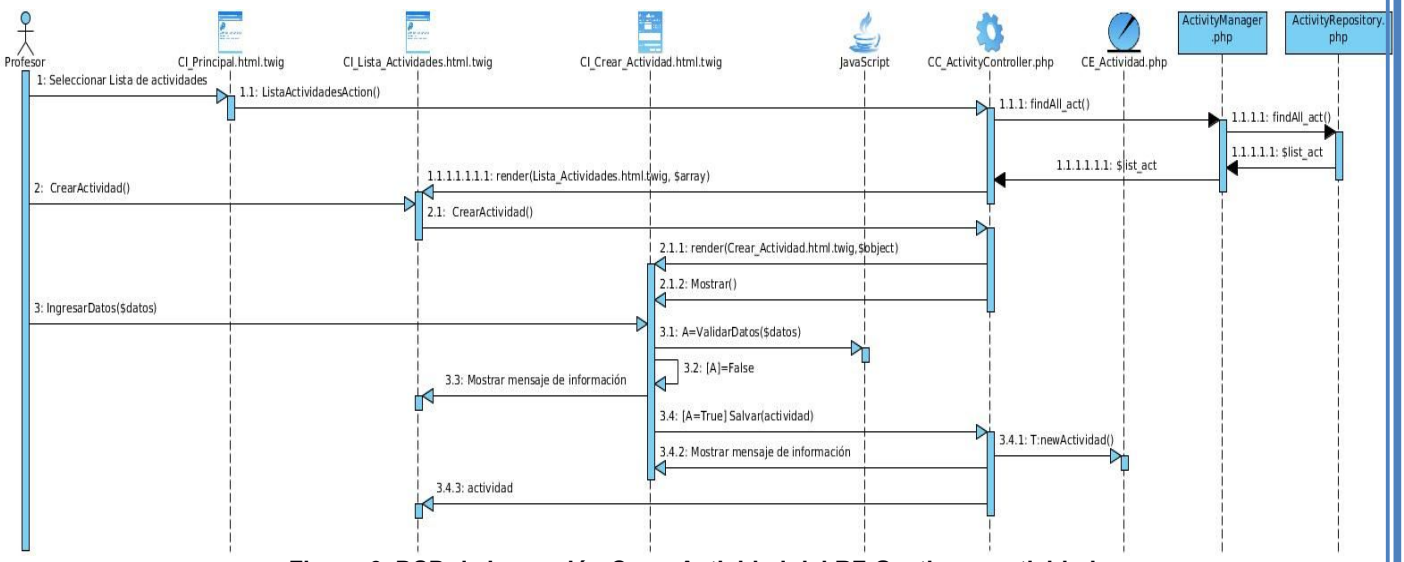

Figura 6. DSD de la sección Crear Actividad del RF Gestionar actividad.

Para el estudio de los demás diagramas de secuencia del diseño remitirse al Anexo 8.

2.7.3 Diagrama de despliegue

El diagrama de despliegue es un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista del despliegue de los artefactos del software en los destinos de distribución. (58)

El diagrama de despliegue que se muestra a continuación representa la distribución física

del sistema a través de nodos. Es decir, modela de manera detallada los nodos computacionales que intervienen en el funcionamiento del sistema, las conexiones que existen entre estos y los protocolos de comunicación que serán utilizados.

Figura 7. Diagrama de despliegue del sistema.

PC Cliente: PC desde donde se podrá visualizar e interactuar con la plataforma a través de un navegador web (previamente instalado en la máquina).

Servidor Web: Servidor Apache donde se encuentra desplegado el componente, al que se conectan los clientes por medio de sus estaciones de trabajo.

Servidor de BD: Servidor PostgreSQL en el que se encuentran almacenados todos los datos persistentes del sistema.

Motor de evaluación: Sistema de evaluación encargado de calificar las soluciones enviadas desde la interfaz por los usuarios a las tareas que le son orientadas.

Los protocolos utilizados son:

- **HTTP:** Protocolo de comunicación estándar-básico que se utiliza en las arquitecturas web. Se utiliza para la comunicación establecida entre el servidor web y las estaciones de los usuarios.
- **TCP/IP:** Protocolo mediante el cual se realiza la comunicación entre el servidor web y el servidor de bases de datos; además de la comunicación entre el servidor web y el motor de evaluación.

2.7.4 Modelo de datos

El modelo de datos es una serie de conceptos que puede utilizarse para describir un conjunto de datos y las operaciones para manipularlos. Está compuesto por las entidades que conformarán las tablas de la base de datos que serán utilizadas por las funcionalidades a implementar. (59)

Descripción de las tablas creadas

- **pkt_conexcoj.tb_user_coj:** Es la tabla que posee la relación del usuario de la plataforma con el usuario del COJ.
- **pkt_conexcoj.tb_activitycoj** y **pkt_conexcoj.tb_activitycoj_single_method:** Son las tablas donde se almacenan los datos de la actividad generada por el profesor, ya sea de tipo *single method* o no, solo lo que es orientado por el profesor porque el resto de los datos que se necesitan para responder la actividad se encuentran en la tabla **pkt_conexcoj.tb_cojproblem.**
- **pkt_conexcoj.tb_cojproblem:** Es la tabla que tiene los datos de los problemas del COJ que se han usado en las actividades, por si el COJ deja de funcionar en determinado momento el usuario pueda ver la actividad.
- **pkt_conexcoj.r_user_activitycoj** y **pkt_conexcoj.r_user_activitysinglemethod:** Son las tablas que portan la relación que existe entre un usuario y una actividad, así como la respuesta a la misma.

En la figura que a continuación se presenta, se muestra la relación entre alguna de las tablas del esquema de PMOOC y las tablas del componente que fueron incorporadas a este.

Capítulo 2
Análisis y Diseño

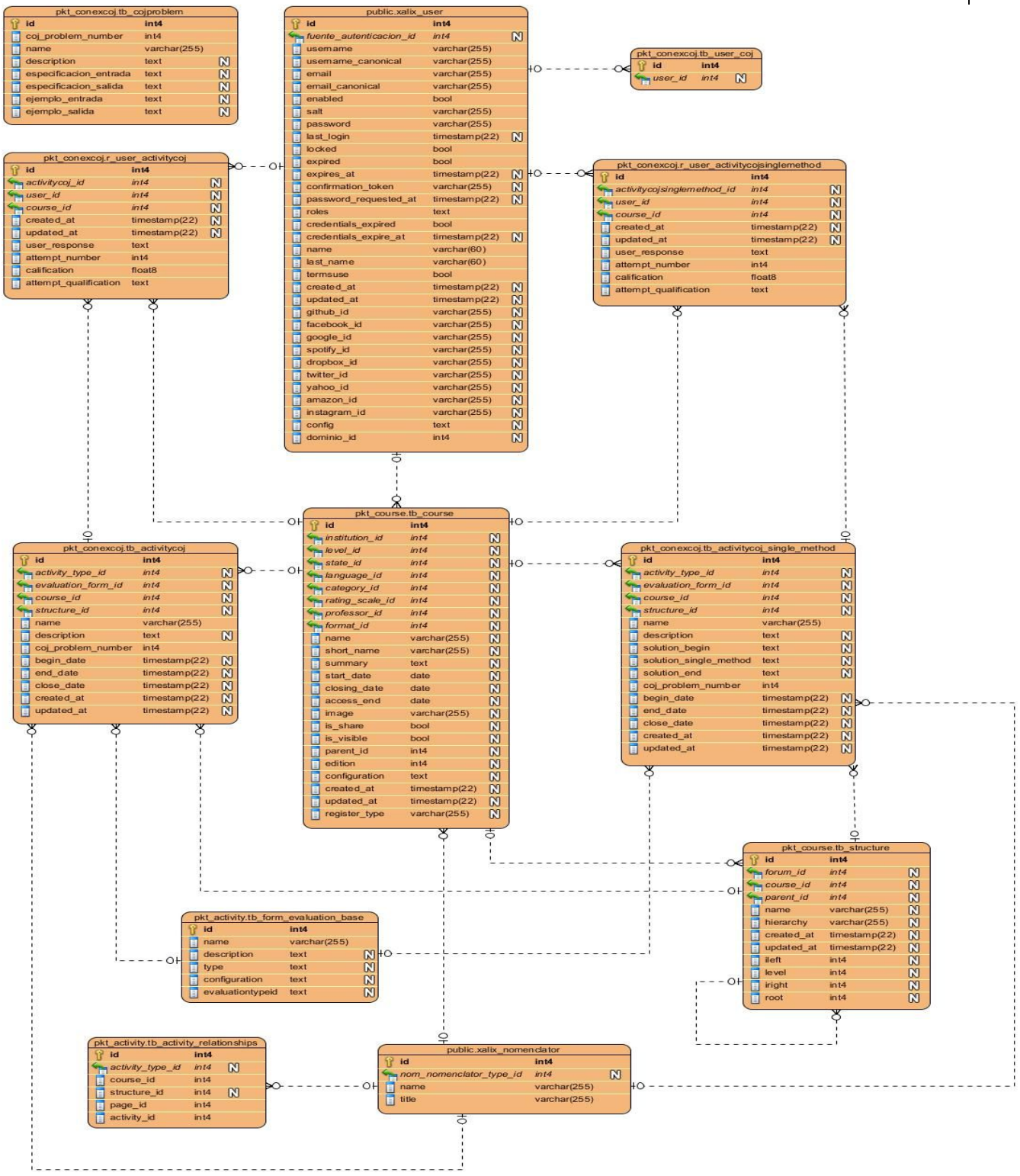

Figura 8. Modelo de datos del sistema.

2.8 Conclusiones del capítulo

La identificación de los principales conceptos asociados a la solución, reflejados en el modelo de dominio posibilitó definir los requisitos funcionales y no funcionales que sustentan la propuesta de solución representados a través de las HU. El desarrollo del análisis y diseño del sistema, permitió definir las bases necesarias para la implementación y posibilitó que se obtuvieran de una forma más concreta y detallada las relaciones que se establecen entre clases. Además, los patrones de diseño que se emplean en el desarrollo de las funcionalidades quedaron bien definidos y argumentados. También se realizó el modelo de base datos donde quedaron plasmadas todas las relaciones de las tablas que serán objetos de persistencia en la BD para las entidades que se manejan en el sistema, y el diagrama de despliegue. Así, se dejan creadas las bases para comenzar con la construcción de la propuesta, velando por el cumplimiento de los requerimientos identificados.

Capítulo 3: Implementación y Pruebas

En este capítulo se documenta el proceso de implementación de los elementos identificados durante la realización del diseño. Para ello se modela el diagrama de componentes. Además se incluyen los resultados de las pruebas y las validaciones realizadas al componente, permitiendo de esta manera que los componentes desarrollados cumplan con los requisitos establecidos y puedan ser integrados a un sistema ejecutable.

3.1 Modelo de implementación

El modelo de implementación describe cómo los elementos del diseño se implementan en componentes. El propósito principal de este flujo de trabajo es desarrollar la arquitectura y el sistema como un todo. Describe también la organización de los componentes según los mecanismos de estructuración y modularización disponibles en el entorno de desarrollo, el lenguaje de programación utilizado, y la dependencia entre componentes. Como parte del modelo de implementación se obtienen los diagramas de componentes que a continuación se presentan. (60)

3.1.1 Diagrama de componentes

El diagrama de componentes representa la estructura física de la implementación. Este diagrama permite visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces. (61) En resumen, muestra los elementos del diseño de un sistema de software y se usa para modelar la estructura del software, incluyendo las dependencias entre los componentes: de software, de código binario y los ejecutables.

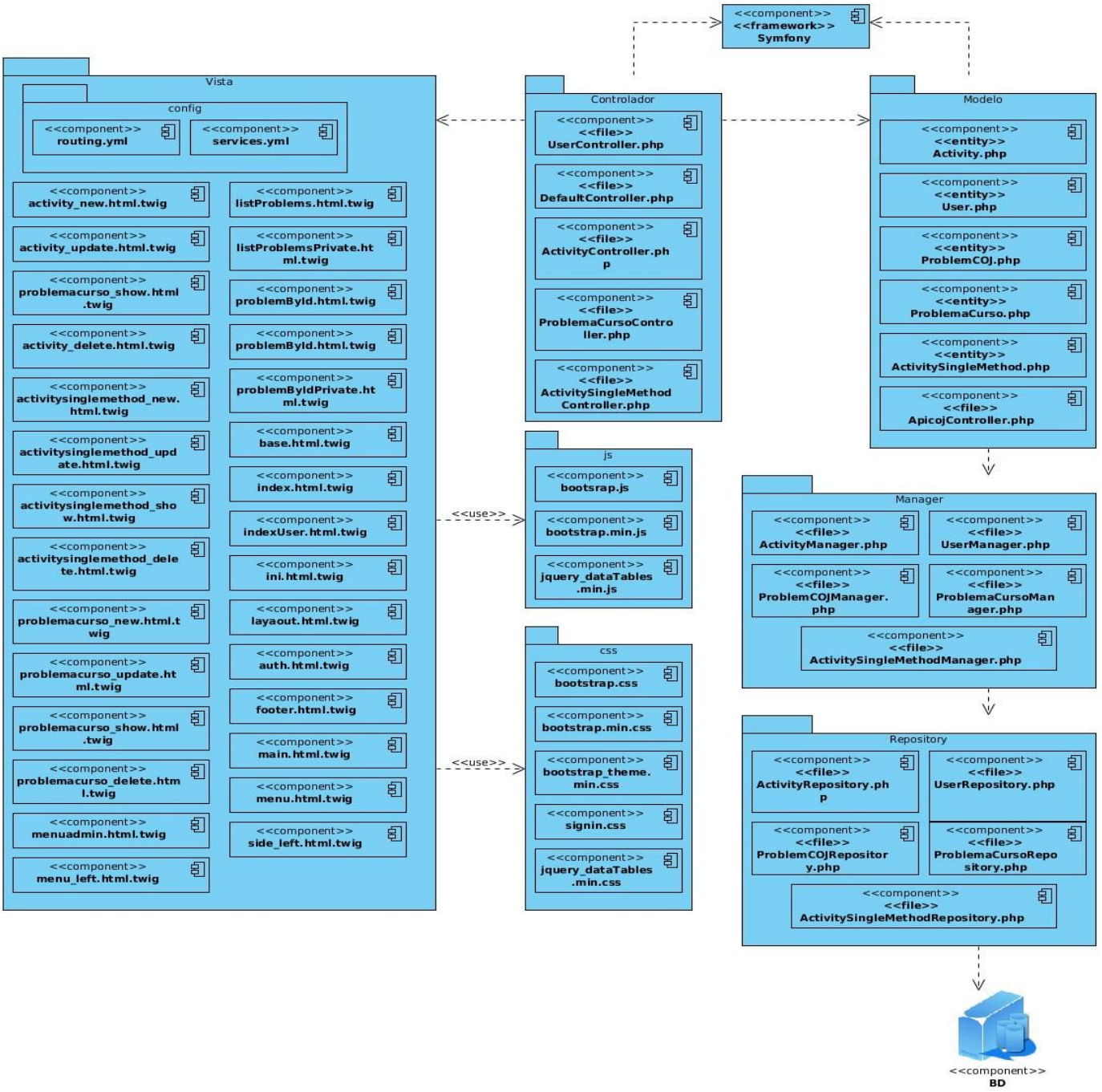

Figura 9. Diagrama de componentes del sistema.

De manera general, el diagrama de componente antes mostrado permite modelar la vista

de implementación del sistema que da inicio a la construcción de la aplicación. Modela además la asignación de clases y otros elementos que facilitan la toma de decisiones respecto a las tareas de implementación.

3.2 Pruebas de software

Las pruebas son un elemento importante dentro del ciclo de vida de un proyecto que proporciona una medida de la calidad del software. Estas son actividades en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. (10)

Las pruebas realizadas a todo producto de software constituyen un elemento crucial para verificar la calidad e integridad del mismo, pues permiten a los desarrolladores identificar, documentar y corregir un conjunto de no conformidades antes de que el producto sea liberado, garantizando de esta forma que el producto final funcione de acuerdo a los fines para los cuales fue diseñado e implemente de manera correcta los requerimientos identificados.

3.2.1 Niveles de prueba

Los niveles de prueba son diferentes formas de verificar y validar un producto de software. A continuación se distinguen los siguientes niveles de prueba empleados para la identificación de los errores.

Prueba unitaria: Es una forma de probar que un módulo de código funcione correctamente. Ello permite garantizar que cada uno de estos funcione de manera eficiente por separado. (62) En resumen, estas pruebas son las que realizan los desarrolladores de software en el código de su componente con el objetivo de detectar errores en los datos, la lógica o en los algoritmos.

Prueba de integración: Es ejecutada para garantizar que en el modelo de implementación, los componentes operen correctamente cuando son combinados para

ejecutar un determinado requisito funcional. (2) Es realizada también por el desarrollador de software con el propósito de detectar errores de interfaces y relaciones entre componentes. Consiste fundamentalmente en validar las conexiones e integración entre dos o más componentes de software haciendo uso de la técnica de caja blanca.

Prueba de sistema: Son las pruebas que se realizan 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. (2) De manera general, este nivel de prueba es preparado y ejecutado por un grupo independiente al desarrollador, y consiste en validar que el software cumpla con los requerimientos especificados por el cliente. Esta es ejecutada utilizando la técnica de caja negra con el objetivo de detectar fallas en el cubrimiento de los requerimientos.

Para la ejecución de las actividades de prueba serán aplicadas por el propio equipo de desarrollo las pruebas unitarias que permitirán verificar y validar la correcta implementación de las funcionalidades que brinda el componente, mediante el uso del framework de pruebas PHPUnit. Se ejecutarán las pruebas de integración para validar la conexión de la plataforma educativa Zera 2.0 con la herramienta de programación competitiva COJ mediante el componente Conexcoj, y también serán realizadas las pruebas de sistema una vez concluido e integrado el componente para garantizar que implemente satisfactoriamente los requerimientos identificados en un inicio.

3.2.2 Métodos de prueba

Un método de prueba es un enfoque sistemático, independiente del nivel en que se enmarque la prueba, que ayuda a encontrar buenos conjuntos de casos de prueba para detectar diferentes tipos de errores. (63) Se puede probar cualquier producto de ingeniería de dos formas: conociendo la función específica para la cual fue diseñado el mismo y conociendo el funcionamiento del producto. Al primer enfoque se le denomina prueba de caja negra y al segundo, prueba de caja blanca. (13)

Pruebas de caja blanca: Es denominada a veces prueba de caja de cristal y es donde se

comprueban los componentes internos. Según Pressman, estas pruebas aseguran que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada. Se basan en un examen detallado de los procedimientos y caminos lógicos del sistema para determinar si el estado real coincide con el esperado. (13)

Pruebas de caja negra: También son denominadas pruebas funcionales o de comportamiento y se centran en los requisitos funcionales del software. Según lo definido por Pressman en el 2002, se llevan a cabo sobre la interfaz del software buscando errores en cada una de las funcionalidades. Con la aplicación de estas se trata de demostrar que las funciones del software son completamente operativas, que las entradas se manejan de forma adecuada y que se produce el resultado esperado. (13)

Para validar la propuesta de solución se emplearán el método de caja blanca que permitirá determinar si el estado real coincide con el esperado o mencionado, y el método de caja negra para comprobar la validez en las respuestas de las funcionalidades ante las acciones del usuario y la calidad de las salidas en dependencia de las entradas.

3.2.3 Diseño de casos de prueba

El diseño de casos de prueba consiste en probar el sistema, incluyendo los datos de entrada y los resultados esperados. Éstos se derivan de las historias de usuario y su objetivo fundamental es encontrar la mayor cantidad de defectos en las funcionalidades implementadas y mostrar que el sistema satisface las necesidades del cliente.

A continuación, se presenta el diseño del caso de prueba perteneciente a la historia de usuario Gestionar Actividad, el resto de los artefactos de este tipo se encuentran en el Anexo 9.

Tabla 2. Diseño de casos de prueba de la HU Gestionar actividad de la SC1 Crear actividad.

CP Gestionar actividad

Descripción general

Permitir crear, eliminar, editar o mostrar los datos de una actividad que puede ser de tipo *single-method* también.

Condiciones de ejecución

El usuario debe estar autenticado en el sistema con el rol Profesor.

Para modificar, eliminar o mostrar una actividad debe existir al menos una previamente creada en el sistema.

SC1 Crear actividad

Escenario	Descripción	T	D	Ee	EE	Se	Respuesta del sistema	Flujo central
EC1.1 Opción Crear actividad.	Selecciona la opción de crear una nueva actividad.						Muestra un formulario para introducir los siguientes datos de la actividad: • Título • Descripción Y de forma opcional: • Tipo de actividad Permite: • Crear la actividad con los nuevos datos. • Cancelar la operación en	Principal/Lista Actividades/Crear Actividad

							cualquier momento.	
EC1.2 Seleccionar tipo de actividad.	Habilita la opción Actividad Single-Method.						Muestra un formulario para introducir los criterios: • Entrada estándar(datos a leer) • Especificación de entrada • Salida estándar(datos a imprimir) Regresa al EC1.1	Principal/Lista Actividades/Crear Actividad
EC1.3 Opción de Crear.	Introduce y/o selecciona los datos de la actividad y ejecuta la opción crear actividad.	V	V	V	V	V	Valida los datos. Crea una nueva actividad. Muestra el listado de actividades y un mensaje de información.	Principal/Lista Actividades/Crear Actividad/Crear
EC1.4 Opción de Cancelar.	Selecciona la opción de Cancelar.						Elimina los datos creados. Regresa al listado de actividades y muestra un mensaje de información.	Principal/Lista Actividades/Crear Actividad/Cancelar
EC1.5	Existen datos	I	V	V	V	V	Muestra un	Principal/Lista

Datos incompletos	incompletos.	V	I	V	V	V	mensaje de información. Muestra un indicador sobre los campos vacíos. Regresa al EC1.1.	Actividades/Crear Actividad/Crear
		V	V	I	V	V		
		V	V	V	I	V		
		V	V	V	V	I		
EC1.6 Datos incorrectos.	Existen datos incorrectos.	I	V	V	V	V	Muestra un mensaje de información. Muestra un indicador sobre los campos incorrectos. Regresa al EC1.1.	Principal/Lista Actividades/Crear Actividad/Crear
		V	I	V	V	V		
		V	V	I	V	V		
		V	V	V	I	V		
		V	V	V	V	I		

Descripción de las variables

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Título (T)	Campo de texto	No	Campo de carácter que representa el nombre de la actividad que está siendo creada o modificada. **Ejemplo:** Actividad 1
2	Descripción (D)	Campo de texto	No	Campo de carácter que representa la descripción de la actividad que se crea o modifica. Puede estar compuesto por la descripción de problemas del COJ.

3	Entrada estándar (Ee)	Campo de texto	No	Campo de carácter que permite entrar parte de la respuesta a la actividad *single-method* que está siendo creado o modificada. Permite entrar el inicio de la clase en java, o sea permite al usuario declarar e inicializar variables, construir el constructor de la clase, entre otros aspectos.
4	Especificación de entrada (EE)	Campo de texto	No	Campo de carácter que permite al usuario entrar los criterios que los estudiantes deben de tener en cuenta a la hora de hacer el método que responde a la actividad en cuestión. Permite especificar nombre del método, nombre de las variables de entrada y salida, etc.
5	Salida estándar (Se)	Campo de texto	No	Campo de carácter que permite entrar parte de la respuesta a la actividad *single-method* que está siendo creado o modificada. Permite entrar la parte final de la clase en java, o sea permite al usuario entre otros aspectos retornar el valor resultante.

3.2.4 Resultados obtenidos

Resultados de las pruebas unitarias

En el proceso de desarrollo del componente se realizaron las pruebas unitarias que permitieron ir comprobando la correcta implementación de las funcionalidades que brinda

el sistema. Para llevar a cabo estas pruebas se empleó el método de caja blanca. Además fueron realizadas por el equipo de desarrollo que aprovechó las ventajas que brinda el framework de pruebas PHPUnit. Estas no se planificaron, ni fueron registrados sus resultados porque se efectuaron en la medida que se desarrollaban las funcionalidades.

Resultados de las pruebas de integración

Durante la implementación de las funcionalidades se realizaron las pruebas de integración donde para evaluar la solución desarrollada se concibieron dos iteraciones para probar el componente de forma íntegra. Estas pruebas validaron la integración de la plataforma educativa Zera 2.0 con la herramienta de programación competitiva COJ mediante el componente Conexcoj, donde se encontraron fallos en las respuestas cuando la operación depende de los servicios prestados por el COJ para el intercambio y transferencia de información. Además, en el resto de este tipo de pruebas los errores detectados están estrechamente relacionados con la comunicación a través de las interfaces. En resumen las pruebas de integración validaron que las funcionalidades establecidas y demás elementos del sistema operen satisfactoriamente, ajustándose a los requisitos especificados.

Resultados de las pruebas de sistema

En las principales pruebas pertenecientes al nivel de sistema que se realizaron se empleó el método de caja negra sobre las interfaces gráficas del sistema. La aplicación de este método tiene como objetivo que el producto final cuente con el menor número de errores posibles. Es decir, se centra en el cumplimiento de las funcionalidades del componente que se obtiene.

Las pruebas realizadas arrojaron los siguientes resultados:

Tabla 3. Resultados obtenidos de las pruebas por iteración.

Iteraciones	Cantidad de casos de prueba	No conformidades detectadas

		Alta	Media	Baja	Total
1	8	11	28	18	57
2	8	6	12	3	21
3	8	2	4	1	7

En la tabla anterior se muestra la ejecución de tres iteraciones, donde en cada iteración fue resuelta cada una de las no conformidades identificadas, contribuyendo al mejoramiento de la propuesta para obtener un producto de calidad libre de errores y satisfacer las necesidades de los usuarios finales.

3.3 Conclusiones del capítulo

En esta etapa del desarrollo quedan implementadas en su totalidad las funcionalidades definidas al inicio de la investigación. El modelo de implementación, conformado por el diagrama de componente obtenido, describió cómo se organizan y cómo se implementan en términos de componentes los elementos del modelo del diseño. Además las pruebas efectuadas al producto permitieron corregir errores para mejorar la calidad de la solución y garantizar la aceptación final por parte del cliente.

Conclusiones Generales

Al término de esta investigación se arriban a las siguientes conclusiones:

- El estudio realizado sobre los sistemas integrados orientados a apoyar el proceso de enseñanza-aprendizaje, sirvió de base para la posterior implementación de un componente completamente funcional que permite dar seguimiento al aprendizaje de la disciplina de programación desde la plataforma educativa Zera 2.0 haciendo uso del COJ.
- La definición de la metodología, tecnologías y herramientas para la implementación del componente, permite afirmar que el uso de tecnologías y herramientas libres, y de metodologías ágiles, aseguran la independencia tecnológica del componente desarrollado, evitan la elaboración de documentación innecesaria y disminuyen el tiempo de desarrollo de la solución.
- Las pruebas aplicadas al sistema mediante la técnica de caja negra permitieron la validación del correcto funcionamiento del componente propuesto, detectándose un total de 85 no conformidades en tres iteraciones, las cuales fueron resueltas al finalizar cada una.
- Las opiniones de los desarrolladores y especialistas consultados, permiten afirmar que el componente desarrollado como parte de esta investigación, mejora las prestaciones de la plataforma actual y fortalece el intercambio de información entre esta y el COJ.

Recomendaciones

Para la continuidad de la presente investigación se recomienda:

- Incorporar un sistema estadístico que posibilite realizar un seguimiento del aprendizaje de programación a partir del cumplimiento de las actividades.
- Utilizar algún algoritmo para la detección de plagio en las respuestas de los estudiantes.
- Construir un manual de usuario para el correcto uso del componente y seguir trabajando en el diseño de las interfaces de las funcionalidades implementadas para lograr un mayor equilibrio y presentación de la información.

Referencias Bibliográficas

1. **Guarín Hernández, Angélica María.** TICS - Tecnologías de Información y Comunicación - Monografias.com. *TICS - Tecnologías de Información y Comunicación.* [En línea] [Citado el: 4 de noviembre de 2015.] http://www.monografias.com/trabajos89/tics-tecnologias-informacion-y-comunicacion/tics-tecnologias-informacion-y-comunicacion.shtml.

2. **Santiesteban Pérez, Irina Ivis y Medina Ramirez, Miguel.** *Desarrollo de funcionalidades que faciliten al docente su preparación y el control del aprendizaje de los estudiantes en la Plataforma Educativa Zera.* Universidad de las Ciencias Informáticas. 2010. Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

3. **Wikipedia: La enciclopedia libre.** *Plataforma educativa.* [En línea] [Citado el: 1 de febrero de 2016.] https://es.wikipedia.org/wiki/Plataforma_educativa.

4. **Cabrera Mallea, Nellis Margarita y Horta Fleitas, Michael.** *Alternativa para el uso del Juez en Línea Caribeño en entornos con dificultades de conectividad.* Universidad de las Ciencias Informáticas. Cuba : s.n., 2013. Trabajo de diploma.

5. **Baylor of University.** The ACM-ICPC International Collegiate Programming Contest. *The ACM-ICPC International Collegiate Programming Contest.* [En línea] International Collegiate Programming Contest, 2 de diciembre de 2014. [Citado el: 26 de enero de 2016.] https://icpc.baylor.edu/. ISBN 00:CC:F5:BC:3A:F7:B4:92:7B:89:EE:F8:A5:87:23:84:41.

6. **International Olympiad in Informatics.** International Olympiad in Informatics. *International Olympiad in Informatics.* [En línea] 2008. [Citado el: 26 de enero de 2016.] http://www.ioinformatics.org/history.shtml. ISSN 2335-8955.

7. **Edurec Blog.** *Tipos de educación (Formal, No Formal e Informal).* [En línea] [Citado el: 11 de febrero de 2016.] http://edurecblog.com/2009/05/13/tipos-de-educacion-fomal-no-formal-e-informal/.

Referencia Bibliográfica

8. **Vázquez, Msc. Tomás Orlando Junco.** *UN JUEZ EN LÍNEA AJUSTADO A LAS NECESIDADES DE LA DOCENCIA.* La Habana : s.n., 2012.

9. **Rodríguez, Diéguez y Saénz, Barrio.** *"Tecnología Educativa y Nuevas tecnologías aplicadas a la educación".* Alcoy : Marfil.

10. **Alemán Llano, Marinés y Pérez Cáceres, Yoandy.** *Desarrollo de un componente que facilite la evaluación del aprendizaje en la Plataforma Educativa ZERA.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2011. Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

11. **Agudelo, Mónica María.** Plataformas educativas. *PLATAFORMAS EDUCATIVAS.* [En línea] 2006. [Citado el: 4 de marzo de 2016.] http://aprendeenlinea.udea.edu.co/banco/html/plataformaseducativas.

12. **Microsoft.** The Component Model Specification. Redmond, 1996, Vol. 99.

13. **Pressman, Roger S.** *Ingeniería de software: Un enfoque práctico.* Quinta edición. s.l. : McGraw Hill, 2002.

14. **Szyperski, C.** *Component Software-Beyond Object-Oriented Programming.* s.l. : Addison-Wesley, 1998.

15. **D'Souza, D. y Wills, A. C.** *Objects, Components and Frameworks: The Catalysis.* s.l. : Addison-Wesley, 1998.

16. **Verdú, Elena, Carabias, David S. y Lorenzo, Rubén M.** *EDUJUDGE: Integrando el Juez-Online en e-learning efectivo.* 2011.

17. **EdUVaLab Universidad de Valladolid.** EdUVaLab. *Módulo QUESTOURnament para Moodle.* [En línea] [Citado el: 8 de febrero de 2016.] http://www.eduvalab.uva.es/proyectos/m-dulo-questournament-para-moodle.

18. **EdUVaLab University of Valladolid.** EdUVaLab. *EDUJudge Project.* [En línea] [Citado el: 8 de febrero de 2016.] http://eduvalab.uva.es/en/projects/edujudge-project.

Referencia Bibliográfica

19. **Youngcoder.** *Youngcoder Platform* . [En línea] [Citado el: 16 de febrero de 2016.] http://www.youngcoder.eu/main.php/platform.

20. **Cutiño Sánchez, Pedro Rafael y Martínez Reina, Gianni.** *SOLUCIÓN DE INTEROPERABILIDAD DEL SISTEMA AUTOMATIZADO PARA LA SUPERACIÓN PEDAGÓGICA EN LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2014. Tesis de diploma.

21. **Barzanallana, Rafael.** Universidad de Murcia. *Metodologías de desarrollo de software.* [En línea] 2 de junio de 2005. [Citado el: 2 de diciembre de 2015.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

22. **Canó, José H., Letelier, Patricio y Panadés, María del Carmen.** *Metodologías Ágiles en el Desarrollo de Software.* Valencia : Universidad Politécnica de Valencia, 2003. ISSN 46022.

23. **Ambler, Scott W.** *The Agile Unified Process (AUP).* [En línea] 1997. [Citado el: 2 de diciembre de 2015.] http://www.ambysoft.com/unifiedprocess/agileUP.html.

24. **Sánchez, Jorge.** *Apuntes completos.Sistemas Gestores de Bases de Datos.* [En línea] 2009. [Citado el: 3 de diciembre de 2015.] http://www.jorgesanchez.net./.

25. **McGraw-Hill Interamericana de España S.A.U. Recursos Educativos AJR.** *Sistemas gestores de bases de datos.* [En línea] [Citado el: 2 de diciembre de 2015.] http://www.mcgraw- hill.es/bcv/guide/capitulo/8448148797.pdf..

26. **PostgreSQL.** Sobre PostgreSQL. *PostgreSQL-es: Portal en español sobre PostgreSQL.* [En línea] 2 de octubre de 2010. [Citado el: 7 de diciembre de 2015.] http://www.postgresql.org.es/sobre_postgresql#intro.

27. **Wilson, Leslie B.** *Comparative Programming Languajes.* [trad.] Addison Wesley. Segunda edición. 1993. pág. 75. ISBN 0-201-56885-3.

28. **Google Inc.** Documentos de Google. *Lenguajes De Programación Del Lado Servidor.* [En línea] [Citado el: 27 de enero de 2016.]

Referencia Bibliográfica

https://docs.google.com/presentation/d/1ZQgIsW4KCyIfH8NyiqH9ueanM83Ym7bjitaxyQKS0hs/edit?pli=1#slide=id.i56.

29. **Grönroos, Marko.** *Book of Vaadin.* Séptima edición. s.l. : Creative Commons, 2014. ISSN CC-BY-ND.

30. **Pérez Eguíluz, Javier.** *Introducción a JavaScript.* 2008.

31. **LIBROSWEB.** *Introducción a CSS. Capitulo 1. Introducción.* [En línea] [Citado el: 27 de enero de 2016.] https://librosweb.es/libro/css/capitulo_1.html. ISBN 437758756273955.

32. **Musciano, Chuck y Kennedy, Bill.** *HTML. La Guía Completa.* [trad.] Yazmín Pérez Parra y Efrén Alatorre Miguel. Segunda edición. 1999. págs. 1-11. ISBN 1-56592-235.

33. **Moreno Navarro, Juan José.** *Arquitecturas software.Curso de software basado en componentes.*

34. **Potencier, Fabien.** Symfony2. *Symfony2.* [En línea] 23 de agosto de 2011. [Citado el: 4 de febrero de 2016.] http://symfony.com/.

35. **CENIA.** ¡Redacta!: Portada. *Bootstrap.* [En línea] 2011. [Citado el: 15 de diciembre de 2015.] http://blog/1/tag/51/bootstrap.

36. **Wikipedia, la enciclopedia libre.** *JQuery.* [En línea] [Citado el: 4 de febrero de 2016.] https://es.wikipedia.org/wiki/JQuery.

37. **macProVideo.com .** macProVideo.com Hub. *Comentario: jQuery 2.0 Nuevas Características.* [En línea] 2016. [Citado el: 4 de febrero de 2016.] http://www.macprovideo.com/es/hub/review-2/review-jquery-20-new-features.

38. **Capacity Academy.** Blog Corporativo Capacity. *jQuery: Qué es, Orígenes, Ventajas y Desventajas.* [En línea] [Citado el: 4 de febrero de 2016.] http://blog.capacityacademy.com/2013/03/16/jquery-que-es-origenes-ventajas-desventajas/.

39. **Novara, Pablo.** Zinjai. *Fundamentos de programación. Introducción a las Herramientas*

Referencia Bibliográfica

de Desarrollo. [En línea] 19 de abril de 2010. [Citado el: 7 de diciembre de 2015.] http://zinjai.sourceforge.net/Anexo1.pdf.

40. **JetBrains s.r.o. .** JetBrains PhpStorm. *PhpStorm IDE.* [En línea] 2000. [Citado el: 8 de febrero de 2016.] http://www.jetbrains.com/phpstorm/.

41. **Netcraft Ltd.** Netcraft. *February 2009 Web Server Survey.* [En línea] 26 de febrero de 2009. [Citado el: 29 de marzo de 2016.] http://news.netcraft.com/archives/2009/.

42. **LinkedIn Corporation.** *Ventajas y desventajas de los servidores apache y IIS.* [En línea] [Citado el: 8 de febrero de 2016.] http://es.slideshare.net/Anthony_mejias/ventajas-y-desventajas-de-los-servidores-apache-y-iis.

43. **González Cornejo, José Enrique.** *¿ Qué es UML? El lenguaje de modelado Unificado.* 2008.

44. **UCLM.** Herramientas CASE . *¿Cómo incorporarlas con éxito en nuestra organización? UCLM, Universidad de Castilla-La Mancha.* [En línea] [Citado el: 5 de diciembre de 2015.] www.uclm.es/ab/educacion/ensayos/pdf/revista10/10_17.pdf.

45. Full-Featured UML Software Design Tool. *Visual Paradigm. Features List.* [En línea] [Citado el: 5 de diciembre de 2014.] http://www.visual-paradigm.com/features/.

46. **S. Pressman, Roger.** Ingeniería de Software. *Un enfoque práctico.* España : Mc Graw Hill, 2005.

47. **Camacho, Erika y Cardesco, Favio.** *Arquitectura de software.* 2004. pág. 21.

48. **BlogSpot.** *PATRONES DE DISEÑO: MODELO VISTA CONTROLADOR.* [En línea] 7 de julio de 2013. [Citado el: 8 de febrero de 2016.] http://thelozu.blogspot.com/2013/07/modelo-vista-controlador.html.

49. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo del Software.* [trad.] Salvador Sánchez y otros. Madrid : Addison Wesley, 2000.

Referencia Bibliográfica

págs. pág. 110, pág. 257-258, pág. 218, pág. 288. y pág. 260. ISBN: 84-7829-036-2.

50. **Letelier, Patricio y Panadés, María del Carmen.** *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* No. 26, Buenos Aires : s.n., 15 de abril de 2006, CyTA, Vol. 5. ISSN 1666-1680.

51. **Beck, K.** *"Extreme Programming Explained. Embrace Change".* s.l. : Pearson Education, 1999. Traducido al español como: "Una explicación de la programación extrema. Aceptar el cambio", Addison Wesley, 2000.

52. **Kremer, Harold.** Mundo Kramer's Blog. *MODELO DE ANÁLISIS.* [Online] mayo 26, 2011. [Cited: abril 18, 2016.] https://mundokramer.wordpress.com/2011/05/20/modelo-de-analisis-software/.

53. **LinkedIn Corporation's Copyright Agent .** SlideShare iOS. *Diagramas de colaboración.* [Online] mayo 21, 2011. [Cited: abril 18, 2016.] http://es.slideshare.net/d-draem/diagramas-de-colaboracion-8052167.

54. **Sú Rodríguez, Gisela and Carrasco Oliva, Argel Jesús.** *Desarrollo de componentes para la interoperabilidad de la plataforma educativa Zera.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2012. Trabajo de diploma.

55. **Larman, Craig.** *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* Segunda edición. pp. 85-88.

56. **Academia Cartagena99.** *Patrones de diseño.* [Online] 2011. [Cited: abril 18, 2016.] http://www.cartagena99.com/recursos/alumnos/apuntes/Patrones%20de%20Diseno.pdf.

57. **Ferré Grau, Xavier and Sánchez Segura, María Isabel.** *Desarrollo orientado a objetos con UML.* Facultad de Informática, Universidad Politécnica de Madrid. Madrid : s.n., 2004.

58. **Melchor, Alex.** Prezi. *Tutorial Diagramas de Despliegue.* [Online] noviembre 22, 2012. [Cited: mayo 17, 2016.] https://prezi.com/e_gpb7xev_im/tutorial-diagramas-de-despliegue/.

Referencia Bibliográfica

59. **Atlantic International University.** AIU - Atlantic International University. *MODELOS DE DATOS.* [Online] [Cited: abril 25, 2016.] http://www.aiu.edu/cursos/base%20de%20datos/pdf%20leccion%202/lecci%C3%B3n%202.pdf.

60. **López Sanz, Marcos.** *Proceso Unificado: Implementación.* [Online] [Cited: abril 25, 2016.] http://www.kybele.etsii.urjc.es/docencia/IS_LADE/2010-2011/Material/[IS-LADE-2010-11]Tema4g.PUD.Implementacion.pdf.

61. **Microsoft.** *Diagramas de componentes de UML: Referencia.* [Online] 2016. [Cited: abril 26, 2016.] https://msdn.microsoft.com/es-es/library/dd409390.aspx.

62. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El lenguaje Unificado de Modelado. Manual de Referencia. .* s.l. : Addison Wesley.

63. **Ramírez, Jaime.** *Unidad de Programación. Métodos de Prueba del Software.*