Universidad de las Ciencias Informática

Facultad 4

Reconstrucción del módulo Sistemas educativos de la Plataforma Educativa Zera

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

Autores

Yolaida Hernández Chavez

Diosbel Pérez Guevara

Tutores

Ing. Yuleisy González Pérez

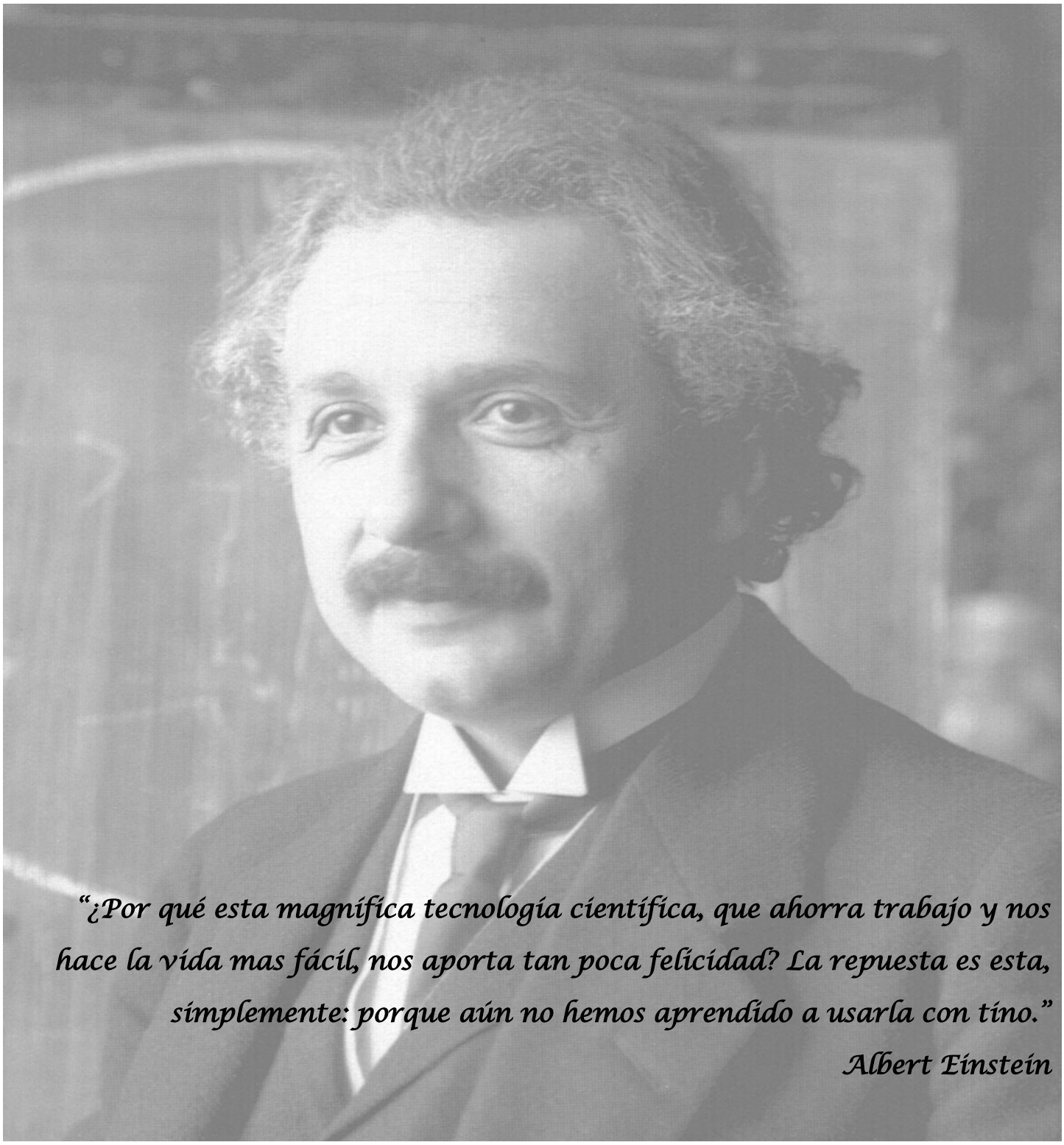
Ing. Daniel Rodríguez Soberats

Cotutores

Msc. Roberto López Dosagües

Ing. Yaismel Miranda Pons

La Habana. Cuba



"¿Por qué esta magnífica tecnología científica, que ahorra trabajo y nos hace la vida mas fácil, nos aporta tan poca felicidad? La repuesta es esta, simplemente: porque aún no hemos aprendido a usarla con tino."

Albert Einstein

Declaración de autoría

Declaramos que somos los autores de este trabajo y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas, así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los _____ días del mes de _____ del año 2012.

Autores

Yolaida Hernández Chavez

Diosbel Pérez Guevara

Tutores

Ing. Yuleisy González Pérez

Ing. Daniel Rodríguez Soberats

Cotutores

Msc. Roberto López Dosagües

Ing. Yaismel Miranda Pons

Agradecimientos

Yolaida Hernández Chavez

Agradezco especialmente a mi papá por ser el principal protagonista de que yo me hiciera Ingeniera Informática. Cuando yo no había pensado que estudiar, ya tú me exigías una carrera universitaria. Papito, gracias por estar siempre conmigo, por tu apoyo incondicional durante estos 5 años, por tus sabios consejos, tu firmeza a la hora de exigirme buenos resultados, por tu amor y cariño, tu comprensión, tus regaños, gracias por demostrarme que yo si podía, sinceramente, sin ti hoy tu hijita no lo hubiese logrado. Lo único que siempre pediste fue que me graduara y por eso hoy me siento muy orgullosa de poder darte esa felicidad. Te quiero mucho papito, gracias por existir. ***Alexis Hernández Martínez***

A mi mamá, por ser una mujer maravillosa, por tener muchos valores como madre que la hacen unica y especial, desde que nos trajo al mundo ha sido una madre abnegada, dedicada a su hogar y a sus hijas. Mima, muchas gracias por estar en todo momento conmigo, por estar siempre pendiente de mi y por hacerme sentir querida y protegida. Te quiero mucho, deseo que siempre estés a mi lado. ***Yolaída Chavez Ivonnet***

A mis hermanas, hermas las quiero con todo mi corazón y sería capaz de hacer cualquier cosa por ustedes. Gracias, porque a pesar de ser tan jóvenes, me he sentido apoyada y motivada en estos 5 años de carrera por el cariño y amor brindado a pesar de estar separadas todo este tiempo. ***Lourdes Leonor Hernández Chavez y Rita Beatríz Hernández Arcaya.***

A mis abuelitos, gracias por ese cariño y protección que siempre tienen los abuelos, por ser tan lindos y tiernos, por estar orgullosos de mi aún sin haberme graduado y por pensar siempre que sí lo iba a lograr. ***Leonor Martínez y Santiago Hernández & Lourdes Ivonnet y René Chavez.***

A mis tíos. ***Amarilis Hernández, El negro, Yaumara Chavez, Tony Hernández, René Chavez.***

A mis primos. ***Zoilita, Oscar y David.***

A mi madrastra por estar siempre con nosotras. ***Damaris Arcaya***

A mi hermanastra. ***Amor de los Ángeles.***

A mi compañero de tesis, mil gracias por permitirme realizar la tesis contigo, has sido un excelente compañero, nunca hubo ni un sí, ni un no al tomar cualquier decisión porque todo era negociable, inclusive hasta la hora de combinarnos la ropa, gracias por esa paciencia, comprensión y humildad que te caracteriza, eres muy lindo y bueno, gracias mi vida, no fue tarea fácil lograrlo, pero juntos salimos adelante. ***Diosbel Pérez Guevara***

A mis amigos más cercanos, los que han estado en las buenas y en las malas, con los que he compartido mis secretos, mis alegrías, mis tristezas y a los que quiero incondicionalmente, a ustedes les agradezco estos años vividos que serán inolvidables. ***Mayliuvis Esquijarrosa, Ancel Nuñez y Alejandro Coterón.***

A mis amigos de forma general. ***Yureidis, Kenier, Migue, Irena, Joel, Yamila, Miroslava, Yusdel, Roilan, Irina, Miguel Mdna, Yuniel, Mayi, Ary, Ayme, Deylis, Cuni, Lirizandra, Martín, Pedri, Carlos Luis, Lionis etc.***

A mis amigos de Venezuela, a una persona que quiero mucho y que me cuidó y me adoptó como si fuera su hija durante toda mi estancia allá, ***Odalis López****, muchas gracias, dios te bendiga y espero volvernos a ver. A* ***Jany****, loquita muchas gracias por ser tan buena, por quererme y malcriarme tanto, espero verte pronto, aquí tienes a una hermana para lo que necesites. A* ***Neri, Dayron, Aida, Mariela, Jandry, Thamy****, etc, a todos de manera general mis agradecimientos por su apoyo y creer en mi estando tan lejos.*

A nuestra tutora, gracias por ser tan peleona, exigente, preocupada, hiperactiva a la hora de intervenir por nosotros en cualquier situación, la verdad a ningún estudiante le gusta que le exijan ni le pongan metas, pero Yula, de no haber sido porque estas cosas te caracterizan, hoy nosotros no lo hubiésemos logrado, este resultado también es tuyo, gracias por dedicarle a la par de nosotros, horas extras a esta tesis. ***Yuleisy González Pérez.***

A nuestro tutor, gracias Dany porque a pesar de ser bastante calladito, eres buen profe y siempre estuviste ahí cuando la aplicación mostraba su lado oscuro. ***Daniel Rodríguez Soberat.***

A mis cotutores, gracias por brindarnos su ayuda siempre que lo necesitamos, sus consejos, recomendaciones, preguntas, todo fue de gran beneficio. **Yaismel y *Dosagüe.***

Diosbel Pérez Guevara

Agradecer especialmente a mis padres por darme un ejemplo a seguir, por haber creído siempre en mí y apoyarme en todas mis decisiones y momentos, por ser esas personas a las que debo mi vida y mi forma de ser. Mil gracias por ser no solo mis padres sino, que fueron y son mis mejores amigos. Los quiero muchísimo y me siento súper orgulloso de ser su hijo. ***Marlene Guevara Guerra y Daniel Pérez García***

Agradecer a mi hermanita que aunque nos pasamos la vida fajados la quiero muchísimo y no sabes cuánto me dolió no poder estar el año pasado en tus 15 años, te quiero mucho mi hermanita. Ahora solo falta que me invites a tu discusión de tesis, porque sé que serás una excelente profesional. ***Darlene Pérez Guevara***

A mis abuelos y sobre todo a mis abuelas Nancy y Nena, la primera que me ha malcriado como a nadie y de quien siempre seré su niño chiquito. Te quiero mucho "mamí" y la segunda porque aunque ya no esté presente me dio toda la educación que tengo y estoy seguro de que se sentiría muy orgullosa de este momento. ***Nancy y Nena***

Agradecer a mi novia por soportarme y estar siempre a mi lado durante estos 4 años, por todo el cariño y el apoyo que me ha dado, por creer en mí y sobre todo por haber sabido esperarme cuando nos alejamos 1 año. Te quiero mucho mi nene. ***Gretel A Segura Rodríguez***

A todos mis amigos tanto los que logré en esta Universidad como en Ciego, los primones porque en la UCI pasamos 5 años juntos y tenemos muchísimas historias, a mis amigos de Ciego que aunque nos separamos hace 6 años hemos sabido mantener la amistad. ***Addrel,***

Aballe, Lomba, Carlos, Edua, Pepe, Roge

Agradecer a mi compañera de tesis por todo el trabajo realizado y toda la ayuda que me ha brindado y sobretodo porque supo crecerse y demostrar que podía lograrlo cuando al inicio nos parecía imposible terminar, muchas gracias y te deseo que sigas siendo esa persona súper alegre y divertida que eres. ***Yolaida Hernández Chavez***

A los tutores porque nos supieron ayudar y apoyar en este corto tiempo de tesis, en especial a Yula que hubiese sido mi compañera de tesis el año pasado y este año casi que ha hecho otra tesis con nosotros, por estar siempre pendiente en todo momento del avance de esta investigación, de verdad muchas gracias. ***Yuleisy Gonzalez Pérez y Daniel Rodríguez Soberats***

Agradecer a los cotutores que nos ayudaron y apoyaron todo el tiempo, en especial a ***Yaísmel*** *que ha sido un buen amigo y compañero y siempre está dispuesto a ayudar.*

Agradecer por último a todas las personas que a lo largo de estos 6 años han estado a mi lado y con las que he compartido la vida en esta increíble Universidad, mis compañeros de grupo, profesores y todos aquellos que de una forma u otra han compartido conmigo.

Nosotros

*Agradecer al **tribunal** y al **oponente** porque en la predefensa nos exigieron y nos señalaron todos los errores y nos permitieron crear un documento mucho mejor elaborado y mucho más profesional.*

*Agradecer por último a esta **Universidad**, a esta **Facultad** que me ha permitido formarme como profesional y como persona y sobre todo al **Comandante en Jefe Fidel Castro** por haber luchado por esta revolución y crear este increíble centro de estudios.*

*Agradecer de manera especial a nuestro **decano** Pedro Luis Basulto porque además de ser nuestro máximo líder en la facultad, es amigo, es compañero, gracias por celebrar en nuestras alegrías y por ayudarnos en los momentos de desesperación.*

Dedicatoria

De Yolaida Hernández Chavez:

*Dedico esta tesis a mis **padres** de manera general por haberme traído al mundo, por formarme y educarme desde que era una niña, por inculcarme que todo en la vida lleva un sacrificio y que tras este sacrificio siempre hay una recompensa, es aquí la recompensa de la que tanto hablaron, a ustedes muchas gracias, los amoooo; pero **principalmente** esta tesis va dedicada a mi **hermana**, lurdita, tu más que nadie sabe lo alegre que me sentí el día que entraste a la universidad y estábamos estudiando lo mismo pero en diferentes años, hoy nos estuviésemos graduando juntas, cosa que anhelaba con la vida y no pudo ser. Aunque no lo creas, que tú no te graduaras, me impulsó mucho a llegar a este momento, porque sentí que este título tenía que lucharlo para las dos, lo logré mi hermana, hoy te estas graduando conmigo, te adorooo.*

De Diosbel Pérez Guevara

Dedico esta tesis a mis padres porque son y serán mis ejemplos a seguir, porque me han dado la educación y la formación, por enseñarme a pensar. Gracias por traerme al mundo y espero se sientan orgullosos de mí. A mis abuelas Nancy y Nena por educarme y malcriarme siempre, aunque mi abuela Nena no este entre nosotros se que estaría muy orgullosa de mí. A mi hermanita y espero que pronto seas una profesional también. Los quiero muchoooo.

Resumen

Con el desarrollo continuo de las Tecnologías de la Información y las Comunicaciones (TIC), surgen soluciones informáticas del tipo Sistemas de Gestión del Aprendizaje (LMS, por sus siglas en inglés). Estos sistemas presentan como objetivo fundamental contribuir al proceso de enseñanza-aprendizaje. La Universidad de las Ciencias Informáticas (UCI) y específicamente el Centro de Tecnologías para la Formación (FORTES), unido a este desarrollo de las TIC y los LMS, se dio a la tarea de implementar la Plataforma Educativa Zera. El presente trabajo de diploma tiene como principal objetivo erradicar las deficiencias en los procesos Sistema curricular, Programas de estudio y Periodos evaluativos que componen el módulo Sistemas educativos. Como solución se aplica una reconstrucción al módulo, permitiendo que los procesos anteriormente mencionados cumplan con las funcionalidades identificadas a partir de las nuevas solicitudes realizadas por el cliente.

Se realizó un estudio de plataformas y *software* educativos que gestionan algunos de los procesos que se encuentran deficientes en el módulo. Además, fueron analizadas las tecnologías y herramientas utilizadas en la versión anterior del módulo. Se desarrollaron los flujos de trabajo propuestos por la metodología de desarrollo de *software* RUP, obteniendo en cada caso los artefactos propuestos por la metodología. Se realizaron pruebas aplicando el método de caja negra con el objetivo de examinar exhaustivamente el sistema. Finalmente se logró desarrollar un módulo que entre sus principales funcionalidades permite crear copias de un sistema educativo en diferentes instituciones; realizar ediciones a los programas de estudio, informándole a los Administradores locales de cada una de las instituciones sobre algún cambio en el sistema educativo central, brindándole además, la posibilidad de aceptar estas actualizaciones.

Palabras Clave: copias, ediciones, módulo, personalizaciones, reconstrucción, sistema educativo

Índice de contenido

Introducción

Las TIC han evolucionado de manera vertiginosa en los últimos años, gracias a su especial capacidad de interconexión a través del uso de la red. La aplicación y desarrollo de estas tecnologías en el campo de la educación ha tenido gran impacto en la organización del proceso de enseñanza-aprendizaje. Con la aplicación de las TIC surgen los llamados LMS. Estos constituyen un apoyo a la enseñanza no presencial, pues permiten administrar, distribuir y controlar los contenidos. Funcionan con tecnología web y facilitan la interacción entre docentes y estudiantes, aportando herramientas que permiten la gestión de algunos de los procesos que se encuentran en la estructura de un sistema educativo.

Los sistemas educativos abarcan tanto a la institución como a todos los medios sociales que influyen en la educación. Es un instrumento curricular donde se organizan las actividades de enseñanza-aprendizaje que permite orientar al docente en su práctica con respecto a los objetivos a lograr.**(1)**

La necesidad de abarcar la mayor parte de las posibilidades y bondades del ámbito educativo por mediación de la web, de gestionar en un mismo entorno todos los procesos que componen un sistema educativo, el no contar con una estructura capitular en cada uno de los cursos que se crean, además, de la inexistente característica en plataformas de ser multi-instituciones, es decir, que existan una o varias instituciones que utilicen los mismos recursos de manera simultánea, son motivos por los cuales la UCI y específicamente FORTES, se dan la tarea de implementar la Plataforma Educativa Zera.

La Plataforma Educativa Zera cuenta con varios módulos, entre ellos se encuentra el módulo Sistemas educativos. Este tiene la responsabilidad de brindar soporte a la gestión de un grupo de procesos, entre los que se encuentran:

- Sistema curricular: sección que abarca el conjunto de materias pertenecientes a la institución.
- Periodos lectivos: espacio de tiempo en el cual estarán vigentes las materias para la institución.
- Programas de estudio: son creados a partir del sistema curricular. Es la adecuación del

macro índice al plan de estudios del sistema educativo y la institución que adquiere el software.

- Sistema evaluativo: comprende dos conceptos que definen la forma de evaluar y el periodo en el cual se realizarán estas evaluaciones en la escuela, así como la posibilidad de ponderar el(los) periodo(s) evaluativo(s).
  - Escala de calificación: mide la capacidad de una persona para producir algo o realizar una operación, es definida como un rango que a su vez podría tener subrangos.
  - Periodos evaluativos: periodo de tiempo en el cual se realizarán las evaluaciones de los programas de estudio que tengan asociados los estudiantes.
  - Ponderación de los Periodos evaluativos: valor que se le asigna a cada periodo evaluativo asociado a un programa de estudio, donde se refleja la importancia de un periodo evaluativo sobre otro.

Independientemente de las funcionalidades que brinda el módulo Sistemas educativos, existen limitaciones en algunos de sus procesos, incumpliendo con parte de los requisitos solicitados por el cliente, entre los que se destacan:

- Cuando hay creado un programa de estudio debe existir al menos un periodo evaluativo asignado a este, actualmente si se tiene un solo periodo evaluativo puede ser eliminado, no cumpliendo así con las necesidades del cliente.
- La ***edición*** de los procesos sistema curricular y programas de estudio presenta problemas, debido a que se crean dos árboles con la información a editar (el primero con los capítulos, temas y subtemas que posee y el segundo con todos los capítulos, temas y subtemas que hay en la plataforma), lo que provoca la ejecución de consultas innecesarias a la base de datos.
- Existen problemas en cuanto a las ***copias*** que realizan las instituciones educativas de los sistemas educativos creados en la plataforma central. Siempre que una institución requiera de uno de estos, se generará una copia tanto en dicha institución como en la plataforma, lo que trae consigo que el módulo actual evidencie ineficiencias en su desempeño (ejecutando grandes consultas a la base

de datos), provocando una disminución en el rendimiento de la Plataforma Educativa Zera.

- La ***personalización*** de un Sistema educativo (Sistema curricular, Programas de estudio, Periodos lectivos o Sistema evaluativo) en una institución central, no puede extenderse a las demás instituciones que utilicen este Sistema educativo de forma automática, si dichas instituciones desean el mismo cambio hay que hacerlo manualmente en cada una de estas, implicando un gasto de tiempo y esfuerzo para la persona encargada de realizar esta acción.

Teniendo en cuenta las deficiencias señaladas donde se evidencia la ineficiencia y poca práctica del módulo Sistemas educativos actual se hace necesario desarrollar una reconstrucción del mismo, realizando nuevamente el análisis y diseño utilizando técnicas de Ingeniería Inversa y para la etapa de reimplementación, herramientas de Ingeniería Directa, de tal manera que los cambios se orienten a su solución, con mayores niveles de facilidad en cuanto a reutilización para obtener un producto con una mejor funcionalidad, mejor desempeño y fiabilidad, así como una mejor facilidad de mantenimiento. **(2)**

En base a la situación antes expuesta, el **problema de la investigación** se resume en la siguiente interrogante: ¿Cómo perfeccionar los procesos que conforman el módulo Sistemas educativos de la Plataforma Educativa Zera?

Por lo que se define como **objetivo general:** Reconstruir el módulo Sistemas educativos de la Plataforma Educativa Zera, para el perfeccionamiento de los procesos que lo conforman.

La presente investigación tiene como **objeto de estudio**: los procesos de gestión de los sistemas educativos en plataformas y *software* educativos y el **campo de acción** se enmarca en: la reconstrucción del módulo Sistemas educativos de la Plataforma Educativa Zera.

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

1. Elaborar el estado del arte de los sistemas informáticos para la gestión de los sistemas educativos.
2. Definir el análisis y diseño del módulo en aras de corregir los errores detectados.
3. Implementar el módulo teniendo en cuenta el análisis y diseño realizado.
4. Demostrar mediante pruebas el funcionamiento del módulo.

Por lo anteriormente planteado se define la siguiente **idea a defender**: si se reconstruye el módulo Sistemas educativos de la Plataforma Educativa Zera, se erradicarán las deficiencias que presentan algunas de sus funcionalidades.

Métodos de investigación utilizados:

Desde la construcción del módulo Sistemas educativos de la Plataforma Educativa Zera fueron estudiados, analizados y escogidos varios métodos de investigación para un mejor entendimiento del objetivo que se perseguía a lo largo de todo el desarrollo del módulo. No fueron cumplidos en su totalidad los requerimientos trazados, por tanto, se decide reutilizar los mismos para la reconstrucción del módulo.

Métodos Teóricos:

El método ***analítico-sintético*** se utilizó para el análisis de la teoría y extracción de los principales conceptos a incluir en el marco teórico. También para llevar a cabo el estudio y análisis de la información relacionada con sistemas y plataformas educativas. Posibilitó descubrir características generales y las relaciones esenciales entre ellas con la Plataforma Educativa Zera, haciendo énfasis en el módulo Sistemas educativos.

El método ***deducción*** fue utilizado a partir de conocimientos y hechos generales de sistemas y plataformas estudiadas, para inferir casos particulares por un razonamiento lógico, teniendo en cuenta que dentro del módulo Sistemas educativos existen funcionalidades con características únicas, no reflejadas en los sistemas estudiados.

El método ***histórico-lógico*** se utilizó para conocer la evolución y desarrollo del proceso de gestión de sistemas educativos, en *software* y plataformas que gestionan contenidos, así como la forma en que han

ido evolucionando durante los últimos años.

Métodos Empíricos:

Se utilizó la ***observación científica*** donde se fue ***selectivo*** y preciso a la hora de centrarse en el objetivo de dicha reconstrucción, ***sistémico*** pues se recogieron datos relevantes durante la investigación del módulo realizando exámenes al mismo con el objetivo de descubrir las deficiencias y determinar la efectividad de las herramientas. Todo ello posibilitó ser ***objetivo*** en cuanto a la recogida de las deficiencias detectadas para que fuera llevada a cabo una reconstrucción del módulo Sistemas educativos.

El contenido del presente trabajo cuenta con la siguiente **Estructura Capitular**:

Capítulo 1: Fundamentación Teórica.

En el trabajo se decide utilizar las tecnologías, metodología, lenguajes de programación y modelado anteriormente investigados y escogidos por el grupo de desarrollo del módulo Sistemas educativos de la Plataforma Educativa Zera. Presenta además, una descripción del estado del arte acerca del problema y las soluciones existentes en plataformas y sistemas similares, nacionales e internacionales.

Capítulo 2: Características del Sistema.

En este capítulo se realiza la Modelación del Negocio y se lleva a cabo el levantamiento de requisitos. Se obtienen como artefactos fundamentales el Diagrama de Casos de Uso del sistema y las descripciones de cada uno de los casos de uso que lo conforman.

Capítulo 3: Análisis y diseño del módulo Sistemas educativos.

En este capítulo se realiza el análisis y diseño del módulo según las deficiencias detectadas, así como los Diagramas de Clases del Diseño, los cuales brindan una visión clara de cómo debe quedar el funcionamiento del módulo luego de terminada la reconstrucción.

Capítulo 4: Implementación y pruebas del módulo Sistemas educativos.

Este capítulo abarca todo lo relacionado con la reimplementación del módulo. Reflejando prácticas de programación, patrones arquitectónicos, patrones de diseño y los estándares de codificación. Además, describe la estrategia trazada para desarrollar las pruebas al módulo mediante la técnica de prueba seleccionada.

Capítulo 1: *Fundamentación Teórica*

1.1 Introducción

La base teórica del presente trabajo está sustentada por las tecnologías, metodología, lenguajes de programación y modelado anteriormente investigados y escogidos por el grupo que desarrolló el módulo Sistemas educativos de la Plataforma Educativa Zera, teniendo en cuenta que se ajustan a la reconstrucción del mismo, variando exclusivamente en las actualizaciones de las nuevas versiones. Este capítulo presenta además, una descripción del estado del arte acerca de los problemas que presenta el módulo Sistemas educativos y de las soluciones existentes en plataformas y sistemas similares, a nivel nacional e internacional.

1.2 Sistemas Similares

El elevado desarrollo de las TIC ha permitido la incorporación de herramientas que apoyan el proceso de enseñanza-aprendizaje. Mediante la integración de estas herramientas en plataformas que presenten un alto grado de flexibilidad, se permitirá adaptar dichos procesos a las necesidades de las instituciones en las que sean utilizadas. A continuación se muestra el análisis realizado a varias plataformas así como a algunos sistemas de gestión de matrícula, con el objetivo de encontrar y definir como se gestionan los procesos que componen un sistema educativo en estas plataformas.

1.2.1 Sistemas a nivel internacional

Moodle: presenta varias similitudes con la Plataforma Educativa Zera al ser un LMS, está diseñado de manera modular, permite gran flexibilidad para agregar y quitar funcionalidades en muchos niveles. Cuenta con escalas de calificación personalizadas, los profesores pueden definir sus propias escalas que permiten dar seguimiento al desempeño de un estudiante en la plataforma y evaluar la participación de estos en los foros, en las tareas y ejercicios realizados.**(4)**

Edu 2.0: al igual que la Plataforma Educativa Zera, es un sistema de gestión docente dirigido principalmente a la educación semipresencial. Incluye los usuarios más importantes de una escuela, está ideado, por tanto, para ser utilizado por los profesores que trabajan habitualmente de forma no presencial y que desean incluir elementos digitales en línea. Esta plataforma crea planes de estudio personalizados, tiene periodos de calificación los cuales están divididos por sistemas de ponderación, presenta además, periodos académicos.**(5)**

DoKeos: es un sistema de aprendizaje virtual basado en la web, intuitivo y fácil de usar por parte de todos los usuarios. Brinda una amplia gama de herramientas. Facilita la creación y organización de contenidos interactivos y ejercicios. Ofrece un entorno virtual que integra herramientas de creación de contenidos, así como herramientas para la creación de actividades, herramientas colaborativas, herramientas de seguimiento e informes sobre el desempeño de los alumnos en el curso.**(6)**

Brinda a los profesores la posibilidad de calificar a sus estudiantes. Permite la gestión y creación de cursos.

Sakai: es una plataforma con un ambiente de aprendizaje y colaboración que añade herramientas para el desarrollo de contenidos. Brinda la posibilidad de generar evaluaciones con diferentes tipos de preguntas, posibilita además, la gestión de programas de estudio con una estructura de curso.**(7)**

1.2.2 Sistemas a nivel nacional

aprenDIST: es una plataforma de educación a distancia que ha comenzado a revolucionar el proceso de educación existente. A medida que aumenta su explotación, los impactos de aprenDIST se van extendiendo hacia otras entidades fuera del centro. Es una plataforma flexible y ajustable a toda una variedad de cursos, por lo que el usuario de este sistema puede dominarlo con solo tener algún conocimiento de navegación sobre la Web. Entre sus características se encuentra la posibilidad de realizar matrículas en línea y permite a los profesores tener un seguimiento completo de la trayectoria de un estudiante en un curso determinado.**(8)**

Akademos: es un sistema de gestión de matrícula que se utiliza en la UCI. Cuenta con varias funcionalidades y módulos que lo hacen necesario en la institución. Uno de los módulos por los cuales

está conformado es el Plan de Estudio, en dependencia del periodo lectivo le muestra al usuario las materias que cursa. Este módulo se define como una sucesión de periodos de tiempo llamados niveles y momentos, los cuales mantienen un orden de presencia, lo que permite generar la estructura estática en años y semestres de un centro de estudios. El problema fundamental del módulo consiste en que se ajusta mucho al negocio de la UCI en su implementación, lo cual unido a su configuración estática ofrece poca flexibilidad para su despliegue en otros centros de estudios. Otras de sus funciones es que permite la configuración de las evaluaciones de las asignaturas, estas pueden ser frecuentes, parciales o finales. El docente puede incluir la evaluación parcial y total del estudiante así como sus cortes evaluativos. **(9)**

1.2.3 Resumen de los sistemas similares estudiados

Las plataformas estudiadas presentan funcionalidades que las convierten en sistemas muy estables, gestionan algunos de los procesos existentes en la estructura de un sistema educativo, pero esa gestión no llega a ser muy abarcadora. Se considera que ninguna de ellas se ajusta a las necesidades que existen en la Plataforma Educativa Zera, debido a que en su mayoría no definen una estructura capitular en el proceso de gestión de los cursos o planes de estudio, a diferencia de lo que ocurre en la Plataforma Educativa Zera que el programa de estudio se deriva de un sistema curricular con una estructura capitular bien definida. Ninguno de estos sistemas presenta la característica de ser multi-institucional, permitiendo que varias instituciones puedan hacer uso de un mismo sistema educativo de forma simultánea. De este estudio se obtiene una visión de cómo manejar algunos de los procesos que se encuentran inmersos dentro de la estructura de los sistemas educativos, por lo que se facilitó el trabajo a la hora de tomar ideas para la reconstrucción del módulo Sistemas educativos en la Plataforma Educativa Zera.

1.3 Metodologías de desarrollo de software, lenguajes de modelado y herramientas para el modelado

El éxito de un proyecto de software involucra la utilización de metodologías de desarrollo, entiéndase por estas, conjunto de pasos y procedimientos a seguir que permiten estructurar, planear y controlar el proceso de desarrollo de *software*.**(10)**

Con la presente investigación se pretende analizar las metodologías, herramientas *CASE (Computer Aided Software Engineering)* y lenguajes de desarrollo de *software* ya estudiados y escogidos por los creadores de este módulo, con el objetivo de reafirmar la selección de estas para ser utilizadas nuevamente en la reconstrucción. Se muestran a continuación los estudios realizados:

1.3.1 Metodologías de desarrollo de software

eXtreme Programming

La Programación Extrema (XP) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de *software*, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, en una comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Se adecua especialmente para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico. **(11)**

Microsoft Solution Framework (MSF)

La metodología *Microsoft Solution Framework (MSF)* proporciona un sistema de modelos, principios y pautas que permiten que todos los elementos de un proyecto, tales como personas, procesos y herramientas puedan ser manejados con éxito. Esta metodología se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. Entre sus principales características se encuentra la adaptabilidad, escalabilidad y flexibilidad. Es válido resaltar que esta metodología puede ser utilizada para desarrollar soluciones basadas sobre cualquier tecnología.**(12)**

Rational Unified Process (RUP)

El Proceso Unificado de Rational *(RUP)* constituye la metodología más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Su ciclo de vida se caracteriza por ser iterativo e incremental, centrado en la arquitectura y dirigido por casos de uso.

Esta metodología define “quién” (trabajadores) debe hacer “qué” (artefactos), “cuándo” (flujo de trabajo y fases) y “cómo” (actividades) debe hacerlo.

El proceso de desarrollo del *software* llevado a cabo utilizando RUP, se divide en cuatro fases: Inicio, en la que se definen la visión, los objetivos y el alcance del proyecto; Elaboración, la cual tiene como principal finalidad completar el análisis de los casos de uso y definir la arquitectura del sistema; Construcción, fase que se encarga de la elaboración de un producto totalmente operativo y eficiente; Transición, en la cual se realiza el despliegue del producto y se capacita a los usuarios finales.

RUP consta de nueve flujos de trabajo: Modelado del Negocio, Análisis de Requisitos, Análisis y Diseño, Implementación, Prueba, Distribución, Gestión de Configuración y Cambios, Gestión del Proyecto y Gestión del Entorno. **(10)**

Selección de la Metodología

Luego del estudio realizado las diferentes metodologías existentes se llega a la conclusión de que a pesar que las metodologías XP y MSF brindan ventajas y beneficios para el desarrollo de un *software*, no se ajustan a las necesidades que hoy presenta la Plataforma Educativa Zera. XP propone por la dinámica del desarrollo un especialista del negocio trabajando conjuntamente con el cliente y en la presente solución el especialista realiza las entrevistas con el cliente a largo plazo, por tanto, esta metodología no es la más idónea a utilizar. Por otro lado MSF, no se ajusta a las necesidades del equipo de desarrollo, no se tiene pleno dominio de la misma, aspecto importante a tener en cuenta, pues se vería afectado el desarrollo de la aplicación en cuanto a tiempo y esfuerzo.

Por todo lo antes expuesto se decide escoger RUP como metodología de desarrollo para la reconstrucción del módulo Sistemas educativos: esta metodología posee las mejores prácticas probadas en la industria. Incorpora las enseñanzas de líderes, así como la experiencia de una buena cantidad de proyectos. Basa su trabajo principalmente en la documentación del *software* y expone un conjunto de actividades orientadas a visualizar, especificar, construir y documentar los artefactos necesarios para lograr un desarrollo con la calidad necesaria, es muy utilizada en proyectos de gran escala.

1.3.2 Lenguajes de modelado

A través de la evolución de la informática y el desarrollo de *software*, se han desarrollado varios modelos

de proceso de *software*, es por ello que existen diversas propuestas de lenguajes. En los últimos años, su construcción se ha convertido en una herramienta habitual en distintos ámbitos, tales como el de la ingeniería de requisitos y el del modelado de procesos en organizaciones. Un lenguaje de modelado es un conjunto de elementos o notaciones que se disponen de modo que ayuden a modelar parte de una aplicación. No todos cubren los mismos aspectos de un sistema, suelen estar ligados a metodologías de desarrollo y/o paradigmas para avanzar de una especificación inicial a un plan de implementación que debe conocer todo el equipo de desarrolladores.**(14)**

A continuación se muestra una breve descripción de importantes propuestas de lenguajes de modelado, dando así a conocer sus características principales.

Lenguaje Unificado de Modelado

Unified Modeling Language por sus siglas en inglés (UML), es un lenguaje capaz de abstraer cualquier tipo de sistema (informático o no), mediante los diagramas, esto es, mediante representaciones gráficas que contienen toda la información relevante del sistema. Para lograrlo, utiliza distintos tipos de diagramas, como los Diagramas de Implementación, Diagramas de Comportamiento o Interacción, Diagramas de Casos de Uso y Diagramas de Clases.

Aún siendo independiente del proceso de desarrollo, UML cuenta con su propia metodología de desarrollo, la cual está basada en componentes. Este proceso utiliza UML para expresar gráficamente todos los esquemas de un sistema de *software*. **(14)**

Las principales funciones de UML son:

- **Visualizar**: permite expresar de una forma gráfica un sistema de forma que cualquier persona lo pueda entender.
- **Especificar**: permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir**: a partir de los modelos especificados se pueden construir los sistemas diseñados.
- **Documentar**: los propios elementos gráficos se pueden utilizar como documentación del sistema desarrollado, sirviendo para su futura revisión.

Está compuesto por tres clases de bloques de construcción. El primero de ellos son los Elementos, es decir, las abstracciones de cosas reales o ficticias por ejemplo (objetos y acciones); luego se encuentran las Relaciones entre los elementos (la cual puede ser de dependencia, asociación, generalización, etc); finalmente se encuentran los Diagramas, que son colecciones de elementos con sus relaciones.

UML representa para los desarrolladores de aplicaciones y sistemas una serie de ventajas, entre las que se destacan:

- Produce un aumento en la calidad del desarrollo.
- Reduce los costos del proyecto.
- Permite especificar la estructura y el comportamiento del sistema y comunicarlo a todos los integrantes del proyecto.
- Permite dimensionar mejor los riesgos de un proyecto, tener un mejor rendimiento antes de construir el sistema. .
- Posibilita realizar una verificación y validación del modelo realizado.

ApEM-L

Lenguaje para la modelación orientada a objetos de aplicaciones educativas y multimedia (ApEM-L). Es una extensión de UML y no modifica la semántica de UML, sino que trabaja en estereotipos restrictivos, por lo que a su vez produce modificaciones descriptivas y decorativas en la representación de los componentes del lenguaje base. En ApEM-L están incluidos los elementos más significativos de OMMMM-L (Lenguaje para Modelado de Aplicaciones Multimedia), para lograr de esta forma una extensión consistente para la modelación de aplicaciones educativas. Es un lenguaje orientado a la descripción de las vistas del *software* educativo. Además, responde a una mejor modelación de aplicaciones educativas teniendo en cuenta que en él se incorporan nuevos diagramas y estereotipos que permiten una mejor comprensión del funcionamiento de aplicaciones con estas características.**(15)**

OMMM-L

OMMM-L para el modelado se usa como metodología RUP, en su extensión lanza una propuesta para la integración de especificaciones de sistemas multimedia basados en el paradigma orientado a objetos. Se

encuentra sustentado en cuatro vistas fundamentales, donde cada una se asocia a un tipo de diagrama en particular.

OMMM–L no es un lenguaje nuevo, sino una extensión del UML, por lo que no es necesario aprenderlo, sino interpretar las características extendidas, centradas a la lógica de funcionamiento de una multimedia, que es por lo general sencilla. Muestra análisis similares a otras metodologías y no se especializa en una clasificación de producto, sino que generaliza a través del uso de la semántica original de UML. Hereda de RUP el ciclo de vida basado en iteraciones y el flujo de trabajo iterativo e incremental, centrado en casos de uso y en la arquitectura. **(16)**

Selección del Lenguaje de modelado

Al concluir el estudio de varios lenguajes de modelado, se acordó seleccionar UML. Este lenguaje tiene como exigencia en la mayoría de instituciones dentro de su Plan Informático estratégico, el desarrollo de *software* bajo una arquitectura en Capas, formalizadas por un lenguaje estándar y unificado. Se requiere que cada una de las partes que comprende el desarrollo de todo *software* de diseño orientado a objetos, se visualice, especifique y documente con un lenguaje común. Para la realización de este trabajo se requirió un lenguaje que fuese gráfico, con el objetivo de especificar y documentar el sistema de *software* de un modo estándar. Incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema. Este lenguaje unificado que cumple con los requerimientos, es ciertamente UML, el cual cuenta con notaciones estándar y semánticas esenciales para el modelado de un sistema orientado a objetos. Presenta gran aceptación a nivel mundial, además, se ajusta completamente a la metodología antes señalada.

1.3.3 Herramientas para el modelado

Se puede definir a las Herramientas *CASE (Computer Aided Software Engineering)*, como la Ingeniería de Software Asistida por Computadora que tiene un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de *software* y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un *software*.**(17)**

Los estados en el ciclo de vida de desarrollo de un *software* son: Investigación Preliminar, Análisis, Diseño, Implementación e Instalación. Es también el conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases. La realización de un nuevo *software* o la reconstrucción de uno ya existente requiere que las tareas sean organizadas y completadas en forma correcta y eficiente. Las herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de un *software*.

Dia

Dia, más que una herramienta de modelado UML, es un programa de dibujo. Permite hacer diagramas de flujo, diagramas de redes, entre otros. Todo mediante librerías de símbolos y un buen conjunto de herramientas. Por lo que hace referencia a UML tiene buen soporte, con una librería surtida y con la ventaja de la libertad de "movimientos", la posición que tienen las interconexiones de los objetos y el control de la escala del dibujo lo hace muy manejable. **(18)**

ArgoUML

Presenta una interfaz muy versátil a la hora de dibujar el modelo, gozando también de una excelente generación de código a partir de los diagramas. Tiene como ventaja que está escrito en Java por lo que se puede ejecutar en cualquier plataforma que tenga una máquina virtual de Java, es fácilmente extensible. **(19)**

Visual Paradigm

Visual Paradigm es una herramienta que utiliza como lenguaje de modelado UML. Soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El *software* de modelado UML ayuda a una más rápida construcción de aplicaciones de mayor calidad y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.**(20)**

Entre sus características más importantes se destacan:

- Soporte UML hasta la versión 2.2.

- Modelado de Caso de uso.
- Generación de reportes.
- Ingeniería inversa de código *Java*, *.NET*, *PHP5*, *Python*.
- Ingeniería inversa - Código a modelo, código a diagrama.
- Integración con Visio - Dibujo de diagramas UML con plantillas (*stencils*) de MS Visio.

Selección de la herramienta para el modelado

Como herramienta para el modelado se escogió *Visual Paradigm* pues fue la utilizada por el equipo de desarrollo del módulo Sistemas educativos. Además, esta herramienta presenta una completa integración con el lenguaje de modelado y con la metodología de desarrollo antes mencionada. Brinda grandes facilidades para la creación de los diferentes diagramas y cuenta con la posibilidad de realizar un control de versiones durante todo el ciclo de trabajo. A pesar de que Dia y ArgoUML son herramientas de modelado que presentan varias ventajas y facilidades para su uso, tienen como inconveniente en el caso de Dia, por ejemplo: soporta una generación de código siempre que existan programas externos que "parsean" el archivo que Dia genera. Resulta muy importante para el caso de que exista un buen modelado, que el código se ha de generar solo, pero siempre y cuando todo está correcto y completo. Por su parte *ArgoUML* al ser el lenguaje *Java* interpretado, esta característica lo convierte en una aplicación muy pesada y únicamente usable en máquinas con buenas prestaciones y de alto rendimiento.

1.4 Lenguajes y tecnologías de programación

1.4.1 Lenguajes y tecnologías del lado del cliente

- **XHTML:** Lenguaje de Marcado de Hipertexto Extensible es una versión más estricta y limpia de HTML que nace precisamente con el objetivo de reemplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en XML. XHTML extiende HTML 4.0 combinando la sintaxis de HTML (diseñado para mostrar datos), con la de XML (diseñado para

describir los datos).**(21)**

- **CSS:** Hojas de Estilo en Cascada *(Cascading Style Sheets)* son un lenguaje formal utilizado para definir la presentación de un documento estructurado escrito en HTML o XML y XHTML. El W3C *(World Wide Web Consortium)* es el encargado de formular la especificación de las hojas de estilo que servirá de estándar para los agentes de usuario o navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.**(22)**
- **JavaScript:** se utiliza principalmente para crear páginas web enriquecidas del lado del cliente. Es un lenguaje interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.**(23)**
- **Ajax:** Acrónimo de *Asynchronous* JavaScript And XML *(JavaScript asíncrono y XML)*, es una técnica de desarrollo web para crear aplicaciones interactivas o RIA *(Rich Internet Applications)*. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones. **(22)**

Lenguajes del lado del servidor

- **Python:** lenguaje de programación interpretado, independiente de plataforma y orientado a objetos. El mismo se encuentra preparado para realizar cualquier tipo de programa, desde aplicaciones *Windows* a servidores de red o incluso, páginas web. Es un lenguaje interpretado, pues no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad.

  En los últimos años el lenguaje se ha hecho muy popular, gracias a varias razones como: **(24)**

  - La cantidad de librerías que contiene, tipos de datos y funciones incorporadas en el propio lenguaje que ayudan a realizar muchas tareas habituales sin necesidad de tener que

programarlas desde cero.

- La sencillez y velocidad con la que se crean los programas. Un programa en Python puede tener de 3 a 5 líneas de código menos que su equivalente en Java o C.
- Es gratuito, incluso para propósitos empresariales.

- **PHP:** es un lenguaje de código abierto muy popular especialmente adecuado para desarrollo web y que puede ser incrustado en HTML.**(25)**

  En junio de 2003 se liberó la primera versión beta de PHP 5, con más mejoras sobre el motor Zend y otras importantes características como el soporte de datos XML y adaptación al protocolo IP versión 6.**(28)**

  Al ser un lenguaje libre dispone de unas cuantas características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas como son:

  - Soporte para diferentes tipos de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL *Server*, *Sybase* mSQL, Informix, entre otras.
  - Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de *Acrobat Reader*) hasta analizar código XML.
  - Ofrece una solución simple y universal para las paginaciones dinámicas en la Web de fácil programación.
  - Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
  - Soportado por una fuerte comunidad de desarrolladores, como producto de código abierto, PHP cuenta con la ayuda de un grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.

De acuerdo con los autores de la creación del módulo Sistemas educativos, se decide utilizar del lado del servidor, PHP. El mismo posee características que lo hacen portador de una mayor flexibilidad en cuanto al trabajo con los distintos gestores de base de datos. Posee además, librerías que facilitan su uso a los desarrolladores. Se debe señalar que dicho lenguaje es actualmente el que se emplea con mayor

aceptación entre los desarrolladores web.

1.4.2 Marcos de Desarrollo

Es necesario la utilización de un marco de desarrollo *(framework)*, debido a que este tipo de tecnología aporta facilidades a la hora de llevar a cabo la implementación de una determinada aplicación.

Un *framework*, en el desarrollo de software es una estructura de soporte definida en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. **(22)**

Son librerías de código que contienen procesos o rutinas ya listos para usar. Los programadores utilizan *framework* para no tener que desarrollar ellos mismos las tareas más básicas, puesto que en el propio *framework* ya hay implementaciones que están probadas, funcionan y no se necesita volver a programar. **(27)**

1.4.2.1 *Frameworks para la capa de presentación*

La capa de presentación no es más que la imagen de una aplicación web, donde su diseño y flexibilidad debe ser bastante fuerte con el objetivo de causar gran impacto en el usuario final.

- **Blueprint:** es un marco de desarrollo para CSS, que tiene como objetivo reducir el tiempo de desarrollo. Brinda una base sólida para construir su proyecto, cuenta con una red de fácil uso, la tipografía sensible, plugins útiles e incluso, una hoja de estilo para la impresión.**(28)** Es un complemento para los desarrolladores web que aumenta la productividad en las tareas de diseño y maquetación de páginas. Contiene una serie de librerías de CSS que brindan código útil para maquetar una página web y aplicar otros tipos de estilos tipográficos o de impresión, algo que comúnmente se conoce como *Framework* CSS.
- **960 Grid:** es un *framework* de CSS que facilita el proceso de implementación de aplicaciones web. Se creó por la necesidad de tener un “estándar” en el ancho de los sitios web. En la actualidad

existe solo un pequeño porcentaje de usuarios que utilizan resolución de 800 x 600 pixeles (px), mientras la mayoría ya utiliza resoluciones de 1024 x 768 px o mayores. Es por eso que un grupo de personas decidió crear este sistema de maquetado basado en 960px de ancho, con configuraciones de 12 y 16 columnas para poder combinar entre sí y así crear un diseño de una forma simple y rápida.**(29)**

- **JQuery:** es una biblioteca de *JavaScript* rápida y concisa que simplifica la forma de presentar e interactuar en un documento HTML. Permite el manejo de eventos, animación y las interacciones Ajax para el desarrollo web rápido. Está diseñado para cambiar la forma en que se escribe JavaScript.**(32)**

  Entre la principales características de este *framework* se tiene:

  - Permite acceder al documento HTML *(DOM, Document Object Model)*.
  - Permite modificar el contenido y la apariencia de las páginas.
  - Posibilita el manejo de eventos con los elementos contenidos en el documento.
  - Permite crear efectos visuales y modificar los estilos CSS.
  - Trabajo fácil con la tecnología *Ajax*.
  - Posee un sistema de *plugins* que permite que sea muy extensible y utilizable.
- **Prototype:** es un *framework* que facilita el desarrollo de aplicaciones web con *JavaScript* y *AJAX*. Su autor original es *Sam Stephenson*, aunque las últimas versiones incorporan código e ideas de muchos otros programadores. A pesar de que incluye decenas de utilidades, la librería es compacta y está programada de forma muy eficiente.**(31)**

El *framework JQuery* se adapta a los requisitos necesarios para lograr la reconstrucción del módulo, es por eso que fue seleccionado para el trabajo con *JavaScript*, debido a la variedad de *plugins* que presenta, los cuales facilitan y hacen más adaptable el trabajo. Se selecciona *Blueprint* para el trabajo con CSS ya que a diferencia de 960 *Grid*, este permite definir un tamaño fuera de 960 px a las aplicaciones y mantiene un estándar adecuado en todos los navegadores web.

1.4.2.2 *Frameworks para la capa de Lógica del Negocio*

En una aplicación web la capa de lógica del negocio es la encargada de realizar cada una de las peticiones y necesidades del usuario. Esta es la capa abstracta, debido a que la misma no es visualizada por el usuario debido a que trabaja por debajo de la capa de presentación y su objetivo principal es realizar las consultas y los procedimientos para mostrarle al usuario lo que necesita ver.

- **Symfony:** *framework* que separa la lógica del negocio, la lógica del servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. **(32)**

  Principales características:

  - Independiente del sistema gestor de bases de datos.
  - Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
  - Sigue la mayoría de las mejores prácticas y patrones de diseño para la web.
  - 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.
  - Código fácil de leer que incluye comentarios de *phpDocumentor* y que permite un mantenimiento muy sencillo.
  - Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.
- **CodeIgniter:** contiene una serie de librerías que se utilizan para el desarrollo de aplicaciones web y además, propone una manera de desarrollarlas que se debe seguir para obtener provecho de la aplicación. Marca una manera específica de codificar las páginas web y clasificar sus diferentes secuencias de comandos, que sirve para que el código esté organizado y sea más fácil de crear y

mantener. **(33)**

Principales características:

- Versatilidad.
- Compatibilidad PHP4 y PHP5.
- Facilidad de instalación.
- Flexibilidad.
- Ligereza.

Luego de realizar el estudio de estos dos *framework* de desarrollo para aplicaciones basadas en la tecnología web, se selecciona *Symfony Framework* para llevar a cabo la reconstrucción del módulo. Este presenta características que lo convierten en un *framework* mejor que *CodeIgniter* a la hora de desarrollar proyectos de gran escala, como la generación de código automáticamente, la facilidad de aprendizaje, el trabajo con los formularios y las validaciones. Provee mecanismos de defensa contra ataques a los formularios, la base de datos y al sistema. Aporta *sub-framework* para el trabajo con el enrutamiento y las pruebas y agrega un alto nivel de seguridad a las aplicaciones.

1.4.2.3 *Capa de Acceso a Datos*

La capa de acceso a datos es la encargada de llevar a cabo el trabajo directo con la base de datos, o sea, es la que gestiona y realiza todas las consultas a esta.

- **Doctrine:** es un Mapeador de Objetos Relacionales (ORM) para PHP5 que se encuentra en la parte superior de una Capa de Abstracción de Base de Datos (DBAL) de gran alcance. Una de sus principales características es la posibilidad de escribir consultas de base de datos en un lenguaje orientado a objetos de propiedad SQL llamada Doctrine Query Lenguaje (DQL), inspirado en Hibernates HQL. Esto proporciona a los desarrolladores una alternativa a SQL que mantiene la flexibilidad sin necesidad de duplicar código innecesario. **(35)**
- **Propel:** es un ORM para PHP5. Permite el acceso a la base de datos a través de objetos previendo de esta forma una Interfaz de Programación de Aplicaciones (API) para manejar y

almacenar los datos. Brinda a los programadores la posibilidad de trabajar con la base de datos al mismo tiempo que lo hace con las clases y objetos de PHP. **(35)**

Teniendo en cuenta las características de Doctrine y Propel queda seleccionado Doctrine como ORM a utilizar en la reconstrucción del módulo. El mismo posee una documentación más abarcadora y es más fácil de emplear que Propel. Además, Doctrine es el que más aceptación y mayor soporte está teniendo en la actualidad.

1.4.3 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es un *software* que tiene como objetivo proporcionar una interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Están compuestos por un lenguaje de definición de datos, uno de manipulación de datos y uno de consulta.

- **PostgreSQL:** es un SGBD objeto-relacional, distribuido bajo la licencia PostgreSQL, similar a la licencia BSD, que solo requiere que el código fuente mantenga su información de derecho de autor (*copyright*) y licencia. Es de código abierto y sus últimas versiones pueden ser comparadas fácilmente con otras bases de datos comerciales.

  PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en lugar de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. **(36)**

  Principales características:

  - Integridad referencial.
  - Juegos de caracteres internacionales.
  - Múltiples métodos de autenticación.
  - Acceso encriptado vía SSL.
  - Completa documentación.
  - Disponible para Linux, UNIX y *Windows* 32/64bit.

Algunos de los límites de PostgreSQL son:

Limite	Valor
Máximo tamaño de base de dato	Ilimitado
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

Tabla 1: Límites de PostgreSQL.

- **MYSQL:** es un SGBD relacional, licenciado bajo la GPL de GNU. Su diseño multihilos le permite manipular bases de datos de gran tamaño, con un orden de 6,000 tablas y alrededor de 50 millones de registros y hasta 32 índices por tabla. Fue creada por la empresa sueca MySQL AB, que mantiene el derecho de autor del código fuente del servidor SQL, así como también de la marca. Aunque MySQL es un *software* libre, MySQLAB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un *software* propietario, ya que de no ser así se vulneraría la licencia GPL.

Analizadas las ventajas de cada uno de estos SGBD se confirma realizar la reconstrucción del módulo con PostgreSQL. Presenta características que lo hacen sobresalir como SGBD por encima de MySQL como es la manipulación de multiprocesos en lugar de multihilos, además de ser de libre acceso y utilización.

1.4.4 Entorno de Desarrollo

Entorno de desarrollo integrado, en inglés, *Integrated Development Environment* (IDE): se refiere a los programas o interfaces visuales de programación que se utilizan para interactuar con el compilador y/o depurador de un lenguaje de programación determinado de forma amigable para el desarrollador, brindándole al mismo una serie de posibilidades que le permitan hacer un mejor uso de sus conocimientos y explotar el lenguaje de programación en el cual está trabajando.

- **NetBeans IDE 7.0.1:** herramienta para que los programadores puedan escribir, compilar, depurar y

ejecutar programas. Está escrito en Java pero puede ser utilizado en cualquier otro lenguaje de programación. Posee un número importante de módulos para extenderlo permitiendo ampliar y facilitar su utilización. Es un producto libre y gratuito sin restricciones de uso. **(37)**

Principales características:

- Creación de Proyectos PHP.
- Integración con *Symfony* y *ZenFramework*.
- Editor de Código Fuente.
- Depuración de PHP.
- Integración con Sistemas de Control de Versiones.

- **Eclipse:** es una plataforma de programación, usada para crear IDE. Es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto.

  Al ser una plataforma IDE, pueden existir centenares de plugins para cada lenguaje. Permite trabajar con lenguajes como son: Java, C/C++, PHP, Cobol, entre otros. Como se podrá observar si se utiliza por ejemplo el paquete de PHP llamado PDT, el mismo brinda soporte con lenguajes para web como HTML y CSS.

De los IDE estudiados, *NetBeans* 7.0.1 es el idóneo para llevar a cabo la reconstrucción del módulo Sistemas educativos de la Plataforma Educativa Zera, ya que presenta posibilidades de integración con *frameworks* como *Symfony* de PHP y *JQuery* de JavaScript, ambos ya seleccionados en la investigación. Además, es un IDE totalmente libre y presenta una comunidad de desarrollo que lo hace ser una buena elección para los desarrolladores de aplicaciones en lenguajes como Java y PHP.

1.5 Conclusiones parciales

Tomando como punto de partida el estudio realizado previamente sobre los sistemas similares y las principales deficiencias detectadas en el módulo Sistemas educativos de la Plataforma Educativa Zera, se decide realizar la reconstrucción del módulo basado en tecnologías web. Se hará uso del lenguaje PHP

versión 5.3, donde se integrarán los *framework JQuery* 1.6.4, *Blueprint* 1.0 y *Symfony* 1.4, en aras de ganar en velocidad de elaboración e incorporar buenas prácticas de desarrollo con el uso de patrones. De igual forma para esta reconstrucción se determina hacer uso de las herramientas *NetBeans* 7.0.1 y *Visual Paradigm* 8.0 debido a las facilidades que presentan y al conocimiento que posee el equipo de desarrollo sobre las mismas. Todo el proceso será controlado y orientado por la metodología RUP, la cual constituye una guía de cómo se debe desarrollar un software y como lenguaje de modelado se utilizará UML en su versión 2.0.

Capítulo 2: *Características del sistema*

2.1 Introducción

Un elemento de gran importancia en el desarrollo de software es describir las principales funciones y el flujo actual de los procesos relacionados en el campo de acción en el que se enmarca el desarrollo.
Se ha de incluir además, la elaboración del Modelo de Dominio. Se tiene como propósito exponer el contexto en que se desarrolla el problema que da origen a esta investigación. Plantear los requisitos funcionales y no funcionales que requiere el sistema. Definiéndose los actores y las relaciones entre ellos, así como los casos de uso del sistema con sus descripciones textuales.

2.2 Descripción de los procesos de negocio

Una vez analizado el módulo Sistemas educativos que se encuentra inmerso dentro de la Plataforma Educativa Zera, se comprueba que los procesos existentes no se hallan bien definidos, lo que provoca que no responda a cabalidad los requerimientos definidos por el cliente. Se identificaron los principales conceptos relacionados con este negocio y se decide ajustar el Modelo de Dominio ya existente para este módulo a las nuevas modificaciones, las cuales serán expuestas en un marco conceptual. Este modelo permitirá de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio de los procesos actuales del negocio; lo cual ayudará a los usuarios, clientes, desarrolladores e interesados, a emplear un vocabulario común para entender el contexto en que se sitúa el módulo. Además, se capturarán los requisitos necesarios que harán factible dicha reconstrucción.
Para ello se identificaron los siguientes conceptos:

- **Administrador central:** usuario que crea la entidad Sistema educativo y junto con esta todos los procesos que la comprenden, puede realizar todas las funciones que realiza el administrador local.
- **Administrador local:** usuario que realiza las tareas de administración en la plataforma que se encuentra instalada en la escuela o institución, así como en algunos de los procesos presentes en el módulo y puede consultar todos los datos del Sistema educativo y de los procesos que lo

componen.

- **Docente:** usuario encargado de educar a los estudiantes en el proceso de enseñanza-aprendizaje. Realiza modificaciones en el Periodo evaluativo para sus estudiantes siempre y cuando el Administrador central lo autorice.
- **Sistema educativo:** es un concepto insertado en el contexto de las prácticas pedagógicas, de la organización política, social y económica de un país. Abarca no solo a la institución, sino a todos los medios sociales que influyen en la educación. Es un instrumento curricular donde se organizan las actividades de enseñanza-aprendizaje, que permite orientar al docente en su práctica con respecto a los objetivos a lograr, las conductas que deben manifestar los estudiantes, las actividades y contenidos a desarrollar, así como las estrategias y recursos a emplear con este fin. De acuerdo a las necesidades e intereses de cada país, puede organizarse en más o menos niveles y a medida que se avanza en estos la complejidad del conocimiento aumenta progresivamente. **(1)**

  Su estructura y organización interna está fuertemente vinculada a otros conceptos. El módulo Sistemas educativos de la Plataforma Educativa Zera está conformado por varios conceptos como son:

- **Sistema curricular:** contiene todos los contenidos educativos que imparte una institución.
  Podrá estar asociado a uno o varios Sistemas educativos.
- **Periodo lectivo:** es el periodo de tiempo por el cual el estudiante va a tener acceso a la materia. Estará asociado a uno o varios Programas de estudio.
- **Programa de estudio:** es creado a partir del Sistema curricular, es la adecuación del macro índice al plan de estudios del Sistema educativo y la institución que adquiere el software.
- **Sistema evaluativo:** comprende dos conceptos que definen la forma de evaluar y el tiempo en el cual se realizarán estas evaluaciones en la escuela, ellos son:
- **Escala de calificación:** Mide la capacidad de una persona para producir algo o realizar una operación, es definida como un rango y puede contar con varios subrangos.

- **Periodo evaluativo:** es el periodo de tiempo en el cual los estudiantes tendrán evaluaciones de los Programas de estudio a los que estén asociados.
- **Ponderación de los Periodos evaluativos:** es el valor en porciento (%) que se le otorga a cada periodo evaluativo para diferenciar la importancia de uno respecto al otro en un mismo programa de estudio. La ponderación que se selecciona debe sumar el 100% entre todos los pertenecientes a un mismo Programa de estudio.

2.3 Descripción de las funcionalidades propuestas

Teniendo en cuenta deficiencias que fueron detectadas en el módulo Sistemas educativos de la Plataforma Educativa Zera, se requiere una reconstrucción de este módulo. Para ello, se proponen las siguientes mejoras a realizar:

- Para solucionar el problema existente de la ***edición*** y las consecuencias que provoca editar cualquiera de los procesos ya sea dentro del módulo o entre instituciones, se pretende que al actualizar un Sistema educativo (Programa de estudio) que se encuentra en la plataforma central, se notifique a los responsables de la o las instituciones que lo estén usando mediante la mensajería (interna) si desea o no aceptar el cambio. En caso de aceptar, se ejecuta la modificación integrándose con la versión que se encuentra en el servidor local de la institución. Siendo capaces de asimilar estos cambios, añadiendo los mismos como elemento final, hasta tanto la institución decida personalizarlo.
- Para mejorar el problema de las ***copias*** que realizan las instituciones educativas de los Sistemas educativos que ya están creados, se propone agregar nuevos campos a las tablas que permitan reconocer aquellos Sistemas educativos que son copias y a su vez poder conocer a partir de que Sistema educativo se crearon.
- Para el caso en que sea ***eliminado*** el único Periodo evaluativo que se encuentra asignado a un Programa de estudio, como este último debe tener que tener siempre al menos uno

asignado, se creará un Periodo evaluativo por defecto que contendrá:

- Nombre del Programa de estudio.
- Fecha de inicio y fin del Periodo lectivo.
- Contenidos del Programa de estudio.
- Ponderaciones de las cinco actividades que conforman un Periodo evaluativo con un 20% cada una.

- Prescindir de funcionalidades como es el caso de la acción ***Eliminar*** que presentan los procesos Sistema curricular, Programas de estudio, Sistema evaluativo y la Escala de calificación, ya que solo pueden ser eliminados los Sistemas educativos y los Periodos evaluativos por el administrador central y los Periodos evaluativos pueden ser eliminado también por el administrador local.
- Perfeccionar el proceso de ***inserción*** de los Periodos evaluativos, informándole a los usuarios aquellos contenidos que fueron previamente seleccionados. Además, agregarle al pie del árbol del Programa de estudio un mensaje de ayuda que le permita al usuario entender qué sucede cuando se seleccionan los contenidos que conformarán un determinado Periodo evaluativo.

2.4 Requerimientos del Software

Los requerimientos de software son las condiciones o capacidades que debe cumplir el sistema, en este caso el módulo Sistemas educativos de la Plataforma educativa Zera, para satisfacer el contrato establecido.

2.4.1 Requerimientos Funcionales

Los requerimientos funcionales (RF) son capacidades o funciones que el sistema debe cumplir.

- **RF 1: Gestionar Sistema educativo:**

- **RF 1.1 Incluir un nuevo Sistema educativo:** se le brinda al usuario con rol de administrador central, la posibilidad de introducir o seleccionar los siguientes datos para la creación de un Sistema educativo:
  - Nombre.
  - Sistema curricular.
  - Periodos lectivos.
  - Programas de estudio.
  - Sistema evaluativo.
  - Escala de calificación.
  - Periodos evaluativos.
  - Ponderación de los Periodos evaluativos.
- **RF 1.2 Ver datos del Sistema educativo:** para esta sección debe existir al menos un sistema educativo creado. Se busca uno para ver sus datos y se selecciona la opción ver. Se muestran los datos del sistema educativo seleccionado.
- **RF 1.3 Modificar datos del Sistema educativo:** debe existir al menos un sistema educativo creado para que pueda ser seleccionado. Se selecciona la opción editar.
- **RF 1.4 Eliminar Sistema educativo:** debe existir al menos un sistema educativo creado. Se selecciona uno y la opción Eliminar.
- **RF 1.5 Personalizar un Sistema educativo:** para dar cumplimiento a este requisito debe existir al menos un sistema educativo creado. Se selecciona uno y la opción Personalizar, la cual permitirá ajustar el sistema educativo a las necesidades de la institución educativa para la cual se está personalizando.
- **RF 1.6 Seleccionar un Sistema educativo:** se muestra un listado de los sistemas educativos que ya han sido creados, del mismo se puede seleccionar uno, este será el sistema educativo que tendrá la institución. La asignación de un sistema educativo a una institución educativa, generará una copia del elemento asignado en la base de datos de la plataforma central.

- **RF 1.7 Actualizar Sistemas educativos:** para actualizar un sistema educativo, debe estar seleccionado previamente y el actor debe tener el permiso de actualizar el elemento. Si lo actualiza, se notificará del cambio a las instituciones que estén utilizando el mismo y si las instituciones deciden aceptarlo se integrará el cambio con la versión de la plataforma central.
- **RF 2: Consultar Sistema educativo:**
- **RF 2.1 Mostrar un listado de los Sistemas educativos:** es necesario que al menos exista un sistema educativo creado para poder cumplir este requerimiento.
- **RF 2.2 Buscar un Sistema educativo dado criterios de búsqueda:** para ello debe existir al menos un sistema educativo creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.

- **RF 3: Gestionar Periodos lectivos:**
  - **RF 3.1 Seleccionar Periodos lectivos:** se muestra un listado de los periodos lectivos que ya han sido creados, del mismo se pueden seleccionar varios, estos serán los periodos de tiempo en los cuales estarán vigentes las materias que conforman el Sistema curricular.
  - **RF 3.2 Incluir un nuevo Periodo lectivo**: se selecciona la opción Incluir. Se le permite al administrador central introducir o seleccionar los siguientes datos:
    - Nombre.
    - Tipo de Periodo lectivo.
    - Anual (se define en un rango de 10 a 14 meses).
    - Semestral (se define en un rango de 4 a 8 meses).
    - Fecha de inicio.
    - Fecha de fin.
  - **RF 3.3 Ver datos del Periodo lectivo:** debe existir al menos un periodo lectivo creado. Se escoge uno y se selecciona la opción Ver.
  - **RF 3.4 Modificar los datos del Periodo lectivo:** debe existir al menos un periodo lectivo creado. Se selecciona el mismo y la opción Editar. Se permite la modificación de todos sus

datos por parte del administrador central. Además, se le permite al administrador local modificar el rango de fechas del periodo en aproximadamente (2) meses.

- **RF 4: Consultar Periodos lectivos:** posibilitar al usuario realizar búsquedas de un periodo lectivo de acuerdo a diferentes criterios, así como mostrar todos los existentes.
  - **RF 4.1 Mostrar un listado de los Periodos lectivos:** para ello debe existir al menos un periodo lectivo creado.
  - **RF 4.2 Buscar un Periodo lectivo dado criterios de búsqueda:** para ello debe existir al menos un periodo lectivo creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.
- **RF 5: Gestionar Programas de estudio:**
  - **RF 5.1 Incluir un nuevo Programa de estudio:** el usuario encargado de esta función es el administrador central. La inclusión de un programa de estudio incluye el establecimiento de los contenidos del mismo, el orden del índice que lo compone y la inclusión de los siguientes datos:
    - Ubicación de la asignatura en el plan de estudios.
    - Enfoque dado al contenido.
    - Base del programa.
    - Marco de referencia.
    - Objetivos de unidad: conocimientos, habilidades, valores y actitudes que se propone adquirir en la unidad de estudio.
    - Elementos de instrumentación.
    - Estrategias didácticas: sugerencias de actividades a desarrollar.
    - Sugerencias de evaluación: estrategias propuestas de evaluación del conocimiento adquirido.
    - Bibliografía: presentada por unidad, son los datos bibliográficos de referencia.
    - Retícula: modelo gráfico de la relación entre objetivos y trayectoria propuesta para su

enseñanza.

• Equivalencias del software con el programa de estudio.

- **RF 5.2 Ver datos del Programa de estudio:** debe estar creado al menos un programa de estudio. Se selecciona uno y la opción Ver.
- **RF 5.3 Modificar datos del Programa de estudio:** debe estar creado al menos un programa de estudio. Se selecciona uno y la opción Editar. En caso de ejecutar el cambio desde la plataforma central, el mismo podrá ser replicado hacia las diferentes instituciones que hagan uso del mismo.

- **RF 6: Consultar Programas de estudio:**
  - **RF 6.1 Mostrar un listado de los Programas de estudios:** debe haber al menos un programa de estudio creado.
  - **RF 6.2 Buscar un Programa de estudio dado criterios de búsqueda:** en esta sección debe existir al menos un programa de estudio creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.
- **RF 7: Gestionar Sistema evaluativo:**
  - **RF 7.1 Incluir un nuevo Sistema evaluativo:** se selecciona la opción Incluir. Se seleccionan o introducen varios datos entre los que se encuentran:
    • Nombre.
    • Descripción.
    • Escala de calificación.
    • Periodos evaluativos.
    • Ponderación de los Periodos evaluativos.
  - **RF 7.2 Ver datos del Sistema evaluativo:** debe estar creado al menos un sistema evaluativo. Se selecciona uno y la opción Ver.
  - **RF 7.3 Modificar datos del Sistema evaluativo:** debe estar creado al menos un sistema evaluativo. Se selecciona uno y la opción Editar.

- **RF 8: Consultar Sistema evaluativo:**
  - **RF 8.1 Mostrar un listado de Sistemas evaluativos:** debe haber al menos un sistema evaluativo creado.
  - **RF 8.2 Buscar un Sistema evaluativo dado criterios de búsqueda:** para ello debe existir al menos un sistema evaluativo creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.
- **RF 9: Gestionar Escala de calificación:**
  - **RF 9.1 Incluir una nueva Escala de calificación:** se selecciona la opción de incluir nuevo. Se le brinda al usuario con rol administrador central la posibilidad de introducir o seleccionar los siguientes datos durante la creación de una escala de calificación:
    • Nombre.
    • Tipo:
    • Cualitativa.
    • Cuantitativa.
    • Descripción.
    • Máximo de la escala.
    • Subrangos, al menos uno. Los subrangos permiten la creación de las escalas.
    Ejemplo: en una escala de 100, el subrango de 98 - 100 se traduce en Sobresaliente, de 90 - 97 se traduce en Excelente, y así sucesivamente.
  - **RF 9.2 Ver datos de la Escala de calificación:** debe existir al menos una escala de calificación. Se selecciona una y la opción Ver.
  - **RF 9.3 Modificar datos de la Escala de calificación:** se ha de contar al menos con una escala de calificación. Se selecciona una y la opción Editar.
- **RF 10: Consultar Escala de calificación:**
  - **RF 10.1 Mostrar un listado de Escalas de calificación:** debe haber al menos una escala de calificación creada.

- **RF 10.2 Buscar una Escala de calificación dado criterios de búsqueda:** debe aparecer al menos una escala de calificación creada. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.

➢ **RF 11: Gestionar Periodos evaluativos:**

- **RF 11.1 Incluir un nuevo Periodo evaluativo:** selecciona la opción Incluir. Se permite introducir o seleccionar. Para la creación de los periodos evaluativos, se tienen en cuenta los siguientes datos:
  - Nombre.
  - Programa de estudio.
  - Descripción.
  - Fecha de inicio: la fecha de inicio no podrá ser menor a la del periodo lectivo definido en el programa de estudio seleccionado.
  - Fecha de fin: la fecha de fin no podrá ser mayor a la del periodo lectivo definido en el programa de estudio seleccionado.
  - Actividades (para cada una de las actividades se fija una ponderación, que podrá ser modificada por el administrador central, local o docente en caso de estar autorizado):
  - Actividad en la plataforma.
  - Actividad en el aula.
  - Reactivo en la plataforma.
  - Reactivo en el aula.
  - Evidencias.
- **RF 11.2 Ver datos del Periodo evaluativo:** debe existir al menos un periodo evaluativo creado. Se selecciona uno y la opción Ver.
- **RF 11.3 Modificar datos del Periodo evaluativo:** debe existir al menos un periodo evaluativo creado. Se selecciona uno y la opción Editar.
- **RF 11.4 Eliminar Periodo evaluativo:** debe existir al menos un periodo evaluativo creado. Se

selecciona uno y la opción Eliminar.

- **RF 12: Consultar Periodos evaluativos:**
  - **RF 12.1 Mostrar un listado de Periodos evaluativos:** debe haber al menos un periodo evaluativo creado.
  - **RF 12.2 Buscar un Periodo evaluativo dado criterios de búsqueda:** debe existir al menos un periodo evaluativo creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.
- **RF 13: Configurar Ponderación de los Periodos evaluativos:** debe haber al menos un periodo evaluativo creado. Cada periodo evaluativo es ponderado (asignación de un porciento (%)), permitiendo asignar el peso en dependencia del tiempo y/o contenido a evaluar este (el peso es asignado por el administrador central, el local o el docente, en caso de estar autorizado).
- **RF 14: Gestionar Sistema curricular:**
  - **RF 14.1 Incluir Sistema curricular:** en la creación de un sistema curricular se le brinda al usuario la posibilidad de introducir o seleccionar los siguientes datos:
    - Nombre.
    - Índice (tres (3) niveles del contenido) que conformará el sistema curricular.
    - Modificar el orden de los elementos del índice.
    - Editar el nombre a los índices.
  - **RF 14.2 Ver Sistema curricular:** es necesario que haya al menos un sistema curricular creado. Se escoge uno de estos. Se selecciona la opción Ver y los datos se mostrarán.
  - **RF 14.3 Modificar Sistema curricular:** debe existir al menos un sistema curricular creado para que se pueda seleccionar. Se permite al rol administrador central modificar los datos de un sistema curricular seleccionado. Las modificaciones realizadas serán replicadas a los sistemas educativos que lo utilizan mediante un mensaje.
- **RF 15: Consultar Sistema curricular:**
  - **RF 15.1: Mostrar un listado de los Sistemas curriculares:** para ello debe existir al menos un

sistema curricular creado.

- **RF 15.2: Buscar un Sistema curricular dado criterios de búsqueda:** para ello debe existir al menos un sistema curricular creado. Se seleccionan los criterios de búsqueda por los cuales se desea filtrar. Se selecciona la opción Buscar.

2.4.2 Requerimientos No Funcionales

Los requerimientos no funcionales (RNF) son propiedades o cualidades que el producto debe tener.

- **RNF de *Software*:**
  - Las computadoras deben tener instalado el navegador Mozilla Firefox 3.x o superior, Internet Explorer 7 o superior, Chrome, Opera o Safari.
- **RNF de *Hardware*:**
  - Procesador Pentium 233 MHz (recomendado 500 MHz o mayor).
  - 1 GB de RAM o superior.
  - 120 GB de espacio en disco duro.
  - Soporte de video que admita resolución de al menos 800x600 y 24 bits.
  - Dispositivo de red de al menos 10 Mbits.
- **RNF de Diseño e Implementación:**
  - Lenguaje de programación: PHP 5.3.
  - El marco de trabajo base de desarrollo que se utilizará es: *Symfony* 1.4.*.
  - Como IDE se empleará NetBeans 7.0.1
  - Como servidor Web se explotará Apache 2.
  - El SGBD deberá ser PostgreSQL 9.1.
- **RNF de Apariencia o Interfaz Externa:**
  - El diseño de las interfaces debe ser sugerente al usuario, permitiendo mayor entendimiento y facilidad del uso del sistema.

- El sistema proporcionará claridad y correcta organización de la información, permitiendo la interpretación inequívoca de ésta.
- Debe contener un diseño sencillo, con pocas imágenes y gráficos para acelerar la velocidad de respuesta del sistema.

- **RNF de Seguridad y Confiabilidad:**
  - Restringir el acceso a usuarios no autorizados para que no puedan realizar acciones sobre las cuales no tenga permiso en la plataforma.
  - Seguridad de acceso y administración de usuarios: otorgamiento de privilegios y roles, asignación de perfiles. Los niveles de acceso están determinados por los diferentes roles válidos dentro de la plataforma.
  - Garantizar que la información sea editada únicamente por el personal esté autorizado y posea los permisos necesarios para ello.
  - Verificación sobre acciones irreversibles (mensajes de eliminación).
- **RNF de Usabilidad:**
  - El módulo podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general.
  - Se deben mostrar las rutas de acceso según la navegación que tenga el usuario.
  - Las rutas de acceso deben tener vínculos a las secciones que muestran.
  - Los elementos gráficos como los íconos deberán contar con un mensaje flotante que señalen el tipo de recurso al que se refiere.
- **RNF de Portabilidad:**
  - La Plataforma Educativa Zera podrá ser utilizada bajo cualquier Sistema Operativo.
- **RNF Funcionalidad:**
  - Reducir al mínimo el tiempo en que carga la Plataforma Educativa Zera.
- **RNF Rendimiento:**

- Tiempos de respuestas rápidas al igual que la velocidad de procesamiento de la información.

➢ **RNF Legales, Derecho de Autor y otros:**

- Una vez terminado el producto, el sistema debe ser sometido a una evaluación y certificación por parte del cliente del mismo.

➢ **RNF de Disponibilidad:**

Los usuarios del sistema deben tener acceso (según sus permisos) en todo momento a la información solicitada.

2.5 Modelo de Casos de Uso del Sistema

Los actores identificados son los responsables de inicializar e interactuar con los casos de uso del sistema.

2.5.1 Descripción de los actores del Sistema

Actores	Descripción
Administrador central	Usuario que gestiona, personaliza y asigna el Sistema educativo a las instituciones, gestionando y consultando también todos los procesos que lo conforman.
Administrador local	Usuario que puede acceder a todos los contenidos de las materias autorizadas. Puede configurar el Programa de estudio de cada materia, estas modificaciones podrán ser: dejar visibles, ocultar y cambiar el orden de los subtemas, temas y capítulos. También puede gestionar los Periodos evaluativos, así como consultar todos los datos que

	conforman el módulo Sistemas educativos. Actualiza en la institución el Sistema educativo correspondiente a la misma en caso de que haya sido modificado a nivel central.
Docente	Usuario que puede gestionar los Periodos evaluativos siempre y cuando el Administrador central le otorgue el permiso necesario para realizar esta acción..

Tabla 2: Descripción de los actores del sistema.

2.5.2 Patrones de casos de uso

Debido al papel clave que juegan los casos de uso (CU) en el modelado de un sistema, se hace necesaria su estructuración. Para ello es necesario auxiliarse de patrones de CU. Estos patrones son un par problema/solución que han demostrado ser la solución adoptada en la comunidad de desarrollo de software. Existen varios tipos de patrones de CU, a continuación se mostrarán los que se reflejan en el diagrama de CU del módulo.

CRUD (Creating, Reading, Updating, Deleting, Creación, Lectura, Actualización y Eliminación por sus siglas en inglés).**(38)**
Se basa en la fusión de casos de uso simples para formar una unidad conceptual.

- **Completo:** este patrón consta de un CU llamado Información CRUD o Gestionar información. Modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio y estos a su vez son cortos y simples.
- **Parcial:** este patrón consta de tres CU llamados, Incluir Información, Ver Información y Modificar Información. Este patrón se refleja en las relaciones actor con los CU Sistema curricular,

Programas de estudio, Escala de calificación y Periodos lectivos.

Múltiples actores

- **Rol común:** puede suceder que varios actores jueguen el mismo rol sobre el CU. Este rol es representado por otro actor, heredado por los actores que comparten este rol. Es aplicable cuando, desde el punto de vista del CU, solo exista una entidad externa interactuando con cada una de las instancias del CU.**(39)**

Este patrón se utilizó para unir las acciones comunes del Administración central y el administrador local.

Otros patrones aplicados son:

- **Relación de extensión:** este patrón es alternativo, extiende las subsecuencia de acciones compartiendo el CU. **(38)** Se aplicará a los CU Gestionar Sistema curricular, Gestionar Programas de estudio, Gestionar Periodos lectivos, Gestionar Sistema evaluativo y Gestionar Escala de calificación, que extenderán de sus CU Consultar.
- **Relación de inclusión:** consta de dos (2) CU. El primero llamado subsecuencia común, modela una secuencia de acciones que aparecerán en múltiples CU en el modelo. **(38)** Su aplicación estará dada en el CU Configurar Ponderación de los Periodos evaluativos, incluido en el CU Gestionar Periodo evaluativo.

2.5.3 Diagrama de Casos de Uso del Sistema

El actor Docente no se modela en el Diagrama de Casos de Uso del Sistema teniendo en cuenta que el mismo no siempre podrá realizar la acción de Gestionar los Periodos evaluativos. Esta acción solo podrá ser realizada por él cuando el Administrador central le otorgue el permiso necesario.

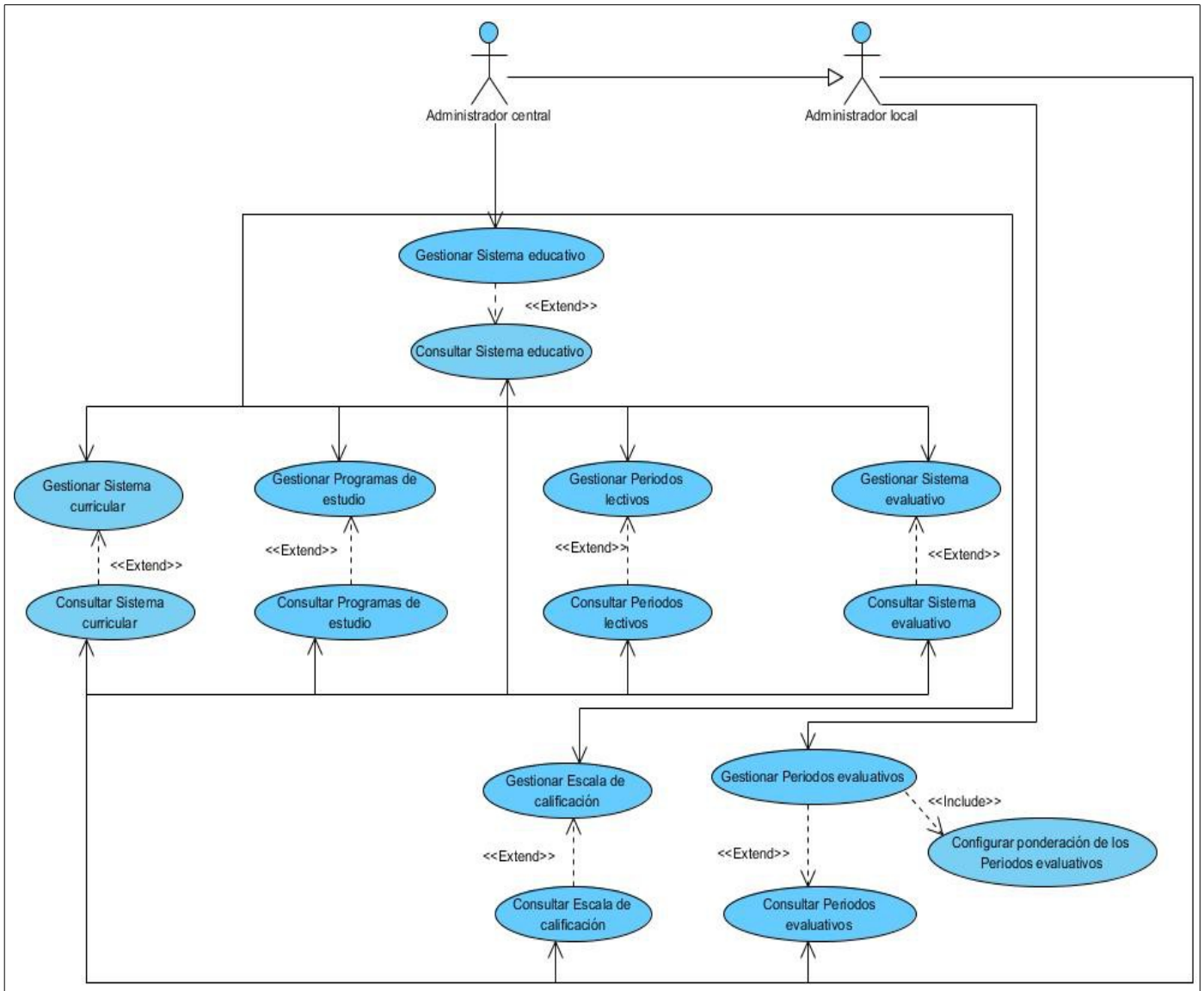


Ilustración 1: Diagrama de casos de uso del Sistema

2.5.4 Descripción de los casos de uso del sistema

Nombre del CU	Gestionar Programas de estudio
Objetivo	Incluir, ver, modificar un Programa de estudio.
Actor	Administrador central (Inicia): Incluye, ve y modifica Programas de estudio.
Precondiciones	Debe haberse generado el escritorio de trabajo del usuario autenticado. Debe haberse realizado una consulta previa de los Programas de estudio. Para ver un Programa de estudio, debe estar seleccionado previamente, y el actor debe tener permiso de acceder al contenido, ya sea porque es el autor, un rol superior en jerarquía, o porque se le ha asignado temporalmente. Para incluir un Programa de estudio, el actor debe tener el permiso de incluir el elemento. Para modificar un Programa de estudio, debe estar seleccionado previamente y el actor debe ser el responsable temporal de la acción.
Poscondiciones	Se incluyó, vio y modificó un Programa de estudio por el actor.
Resumen del caso de uso	El caso de uso se inicia cuando el Administrador central selecciona la opción que le permite realizar una acción sobre un Programa de estudio. El actor puede incluir, ver y modificar un Programa de estudio. En caso de que seleccione la opción de incluir un Programa de estudio, el sistema dará la posibilidad de insertar los datos que se necesitan para llenar esta plantilla. Si el actor elige la opción de ver un Programa de estudio el sistema mostrará el contenido del Programa de estudio en cuestión. Si el actor elige la opción de modificar un Programa de estudio, el sistema mostrará los datos que pueden ser editables dentro del Programa de estudio, y una vez realizados los cambios, guardará las modificaciones.
Referencia	**RF 5:** RF 5.1, RF 5.2, RF 5.3

Tabla 3: Descripción del CU Gestionar Programa de estudio.

2.6 Conclusiones parciales

Con el estudio realizado de los principales conceptos del Dominio se logra desarrollar un análisis crítico de los procesos que conforman un sistema educativo. Se presenta un listado de RF y RNF, donde se recogen las principales necesidades y deficiencias detectadas en el módulo a diseñar. Estas necesidades fueron traducidas a un conjunto de CU, las cuales representan las principales funcionalidades del mismo. Teniendo en cuenta el estudio realizado hasta este momento, se le puede dar comienzo a la reconstrucción del módulo partiendo de las principales deficiencias detectadas y cumpliendo con cada requerimiento identificado.

Capítulo 3: *Análisis y diseño del módulo Sistemas educativos*

3.1 Introducción

La realización del análisis y diseño del módulo permitirá describir en términos de componentes, las clases y subsistemas identificados anteriormente en el análisis y diseño del módulo Sistemas educativos, realizado en el trabajo de diploma "Análisis y Diseño de un módulo para la gestión de los Sistemas educativos en la Plataforma Educativa Zera" que antecede a este trabajo de diploma. Teniendo en cuenta las deficiencias detectadas, las cuales se encuentran corregidas en el presente trabajo. Se toma el diagrama de CU del módulo como punto de inicio.

Se describen los artefactos generados a partir del seguimiento de las actividades del flujo de trabajo Análisis y Diseño, para el CU Gestionar Programas de estudio tomado como muestra. El resto de los artefactos desarrollados se encuentran detallados en los anexos. Los requisitos refinados a través del Modelo de Análisis y un plano de la reimplementación a través del Modelo de Diseño se plasman en este capítulo, reflejándose las buenas prácticas de programación utilizadas, así como los patrones arquitectónicos, patrones de diseño y los estándares de codificación que guían el desarrollo de esta etapa de la investigación.

3.2 Modelo de Análisis

En el Modelo de Análisis se identifican una serie de clases y relaciones. Como finalidad tiene refinar los CU detalladamente. Brinda un conocimiento razonable del módulo y ayuda a estructurar los requisitos. Permite entender mejor los aspectos internos, pues sus diagramas ofrecen una mayor formalización y poder expresivo. Es un Modelo de Objetos que describe la realización de CU.**(40)**

Seguidamente se explica de forma breve cada uno de los estereotipos presentes en los Diagramas de Clases del Análisis (DCA).

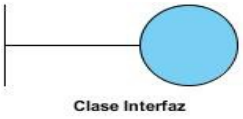
Clases	Descripción	Prototipos
Clase interfaz	Modelan la interfaz del sistema y manejan la comunicación entre el entorno y el interior del mismo.	
Clase entidad	Representan la información manejada en el CU. Además, modelan la información y el comportamiento asociado que generalmente es de larga duración.	
Clase controladora	Coordinan los eventos necesarios para la realización o especificación del CU, o sea, son las que ejecutan el CU.	

Tabla 4: Descripción de las Clases de Análisis del Sistema (DCA).

3.2.1 Diagramas de Clases del Análisis

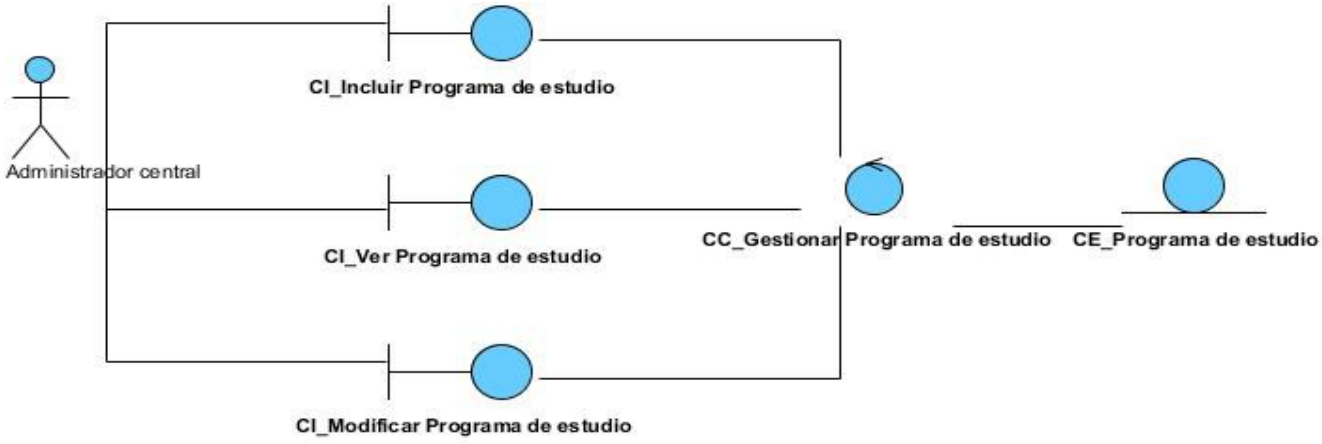


Ilustración 2: DCA Gestionar Programa de estudio.

3.2.2 Diagramas de Colaboración

Ilustración 3: DC_CU Gestionar Programas de estudio, Sección Ver.

3.3 Arquitectura del módulo

Para el desarrollo de software existen buenas prácticas en el diseño arquitectural, especialmente en cuanto a la arquitectura lógica a gran escala, estas prácticas se han escrito en forma de patrones. El *framework* Symfony utiliza en su implementación una serie de patrones que clasifican y describen formas de solucionar problemas específicos y comunes a la hora de desarrollar aplicaciones utilizando este *framework*. En lo adelante se explicarán algunos de los patrones utilizados directamente en la solución.

3.3.1 Patrón arquitectónico Modelo – Vista – Controlador

- **Modelo Vista Controlador (MVC):** divide una aplicación interactiva en tres componentes. El modelo representa la información con la que trabaja la aplicación, es decir su lógica de negocio. La vista transforma el modelo en una página web que permita al usuario interactuar con ella. El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.**(41)**

  Symfony utiliza lo mejor de la arquitectura MVC y la realiza de modo que el desarrollo de

aplicaciones sea rápido y sencillo. En el controlador se encuentran las acciones (*Actions*), las cuales son el núcleo de la aplicación, pues contienen toda la lógica de la aplicación. Estas acciones utilizan el modelo que a su vez utiliza una capa de abstracción de base de datos en este caso el ORM Doctrine y precisan las variables para la vista, la cual tiene las plantillas necesarias para mostrar la información al usuario y como mínimo una plantilla global (*layout*) que encapsula el código HTML común de las demás plantillas.

- **Front Controller (Controlador Frontal)**: este elemento provee un controlador centralizado para gestionar las peticiones web a la aplicación. El controlador proporciona un punto de entrada único que controla y gestiona las peticiones web realizadas por los clientes, remitiendo a su vez cada petición al gestor de peticiones adecuado. Son los puntos ideales para implementar servicios de seguridad, tratamiento de errores y la gestión del control para la generación de contenidos. En la propuesta de solución presente se utiliza este patrón y se evidencia cuando el *framework* de desarrollo genera un controlador por cada aplicación o subsistema, que en este caso son (*index.php, administration.php y bachelor.php*) de las respectivas aplicaciones o subsistemas (*learning, administration y bachelor*) donde se utiliza la propuesta de solución.

3.3.2 Patrones de diseño

Patrones GOF

- **Singleton (Instancia única):** este patrón garantiza la existencia de una instancia única para una clase y la creación de un mecanismo de acceso global a dicha instancia. Como ejemplo de su utilización se cuenta con la clase *sfContext* que guarda una referencia a todos los objetos del núcleo de *Symfony*, mecanismo mediante el cual se interactúa con los objetos únicos.
- **Decorator (Envoltorio):** el objetivo de este patrón es extender la funcionalidad de un objeto dinámicamente. Un ejemplo de utilización de este patrón se pone en práctica en el módulo *evaluationPeriod*, *studyProgram* en el momento de cargar dinámicamente el árbol con las materias y al crear el *layout_educative_system_wizard.*

- **Observer (Observador):** este patrón permite definir una dependencia entre objetos de forma que cuando un objeto cambia de estado, los objetos que dependen de este son notificados y actualizados.

Patrones GRASP

- **Experto:** la responsabilidad de realizar una labor determinada la posee la clase que tiene o puede tener datos involucrados. Es uno de los patrones que más se utiliza cuando se trabaja con *Symfony*, con la inclusión de la librería Doctrine para mapear la Base de Datos.
- **Creador**: es utilizado para establecer la responsabilidad de quién es el encargado de crear los objetos de una clase determinada. Se pone en práctica en la mayoría de las acciones de los módulos *educativeSystem*, *studyProgram*, *evaluativeSystem* y *evaluationPeriod*, creando objetos de entidades, de formularios y de clases de abstracción a la base de datos.
- **Alta Cohesión**: cada elemento del diseño realiza una labor única dentro del sistema, no desempeñada por el resto de los elemento. *Symfony* permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Las clases *Actions* están formadas por varias funcionalidades que están estrechamente relacionadas, siendo la misma responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las properties.

3.4 Modelo de Diseño

Es un modelo de objetos que describe la realización física de los CU, centrándose en cómo los RF y los RNF tienen impacto en la aplicación a desarrollar. Facilita la abstracción de la reconstrucción del módulo y es de ese modo, el artefacto fundamental de entrada de las actividades de implementación.

3.5 Modelo de Despliegue

El Modelo de Despliegue define la arquitectura física del sistema. Detalla los nodos físicos y las

asociaciones de comunicación que existen entre ellos. Del mismo modo, queda especificado qué *hardware*, sistemas operativos, software de interfaces y soporte conformarán el nuevo sistema.

A continuación se presenta el Diagrama de Despliegue propuesto para el sistema:

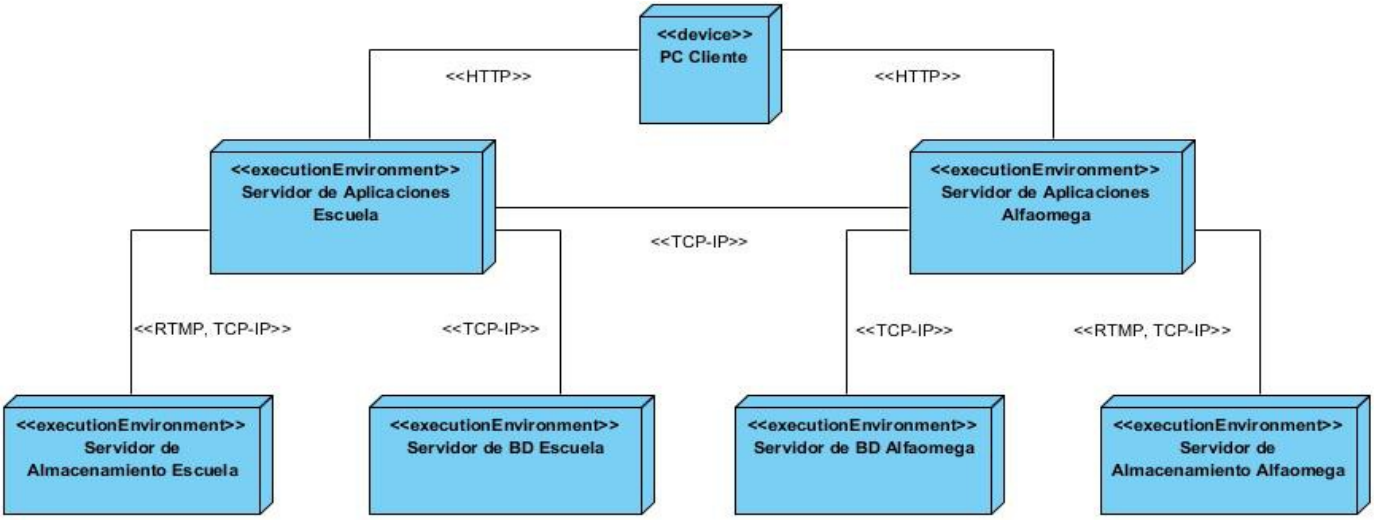


Ilustración 4: Modelo de Despliegue.

3.6 Conclusiones parciales

Este capítulo muestra una presentación de rediseño del módulo Sistemas educativos luego de corregir las deficiencias detectadas. Se describen los patrones arquitectónicos y de diseños utilizados y a utilizar en el flujo de implementación que dará comienzo una vez terminado el presente apartado. También se crea el Diagrama de CU del Sistema, se describen los CU del Sistema, se realiza el Diagrama de Clases del Análisis y el Diagrama de Colaboración, se realiza el Diagrama de Clases del Diseño, se crea el Modelo de Base de Datos y se describen sus tablas. Sentando de esta forma las bases necesarias para comenzar el flujo de implementación y prueba.

Capítulo 4: *Implementación y pruebas del módulo Sistemas educativos*

4.1 Introducción

Durante el flujo de trabajo Análisis y Diseño son generados los artefactos que se utilizan como entrada fundamental al flujo Implementación. La reimplementación del módulo Sistemas educativos de la Plataforma Educativa Zera, tiene como propósito fundamental: definir la organización del código, reutilizar clases y objetos en forma de componentes, probar los componentes desarrollados e integrarlos a un sistema ejecutable. Para llevar a cabo la prueba de estos componentes, es necesario desarrollar un control estricto de la calidad del producto resultante, para lo cual es necesario que el producto funcione como está descrito y que la validación de requisitos se desarrolle correctamente.

4.2 Modelo de Implementación

La reimplementación del módulo comienza con los resultados obtenidos en el diseño. Los Diagramas de Despliegue y Componentes conforman lo que se conoce como un modelo de implementación.
El Modelo de Implementación representa la composición física de la implementación en términos de subsistemas de implementación y elementos de implementación (directorios y archivos, incluyendo código fuente, datos y archivos ejecutables). Identifica los componentes físicos de la implementación para que puedan comprenderse y gestionarse mejor. Define las principales unidades de integración alrededor de las cuales se organizan los equipos, así como las unidades que se pueden versionar, desplegar y reemplazar separadamente.**(42)**

4.2.1 Diagrama de Componentes

Un Diagrama de Componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos. Describen los elementos físicos del módulo y sus relaciones. Los

componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables o paquetes. Estos prevalecen en el campo de la arquitectura de software pero pueden ser utilizados para modelar y documentar cualquier arquitectura de sistema.

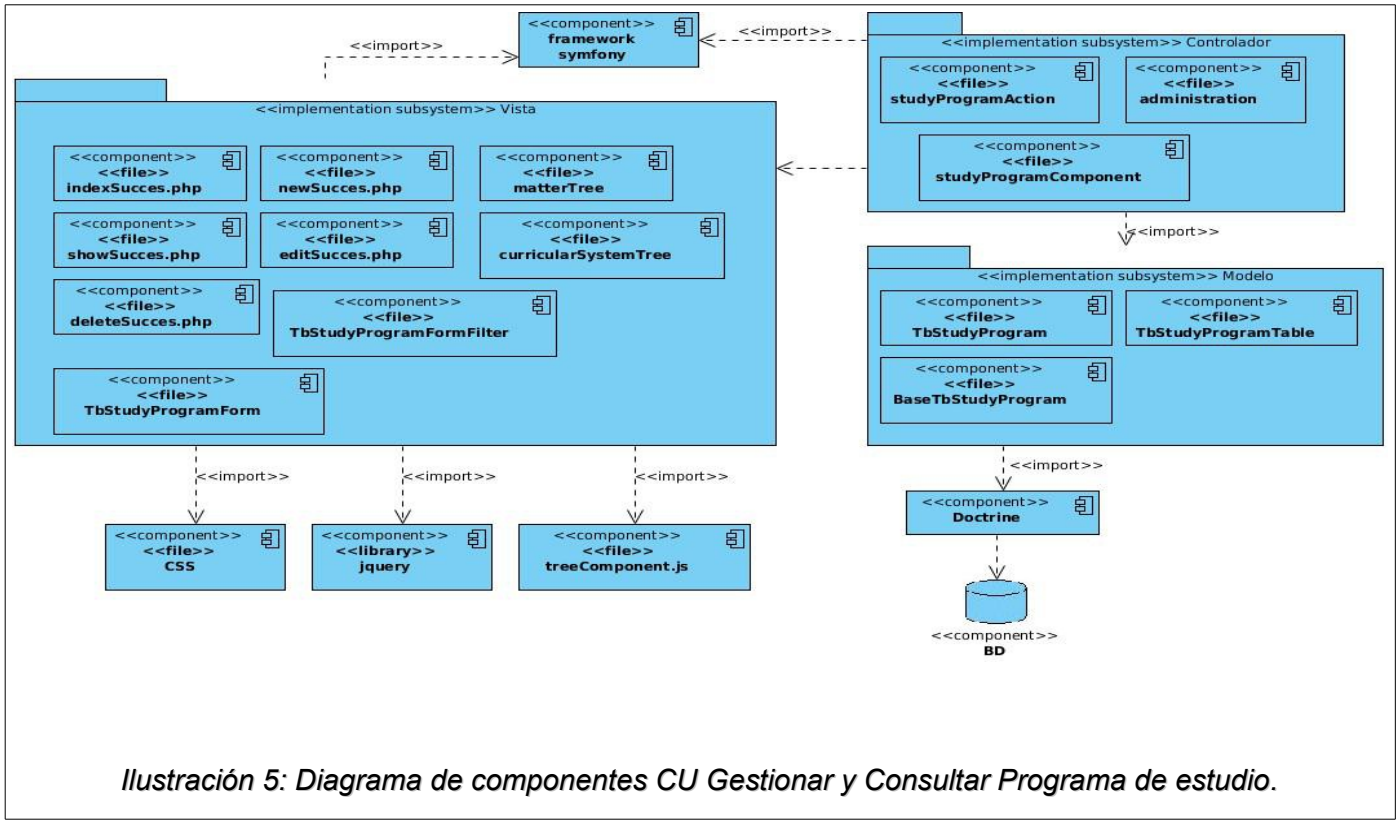


Ilustración 5: Diagrama de componentes CU Gestionar y Consultar Programa de estudio.

4.3 Pruebas de software

Las pruebas se centran principalmente en la evaluación o la valoración de la calidad del producto, hecho que se lleva a cabo mediante las prácticas:

- Buscar y documentar los defectos en la calidad del software.
- Validar que el producto de software funciona según lo diseñado.

- Validar que los requisitos se han implementado de forma adecuada.

Existe una interesante diferencia entre Probar y las otras disciplinas de RUP - básicamente, las tareas de Probar son encontrar y exponer los puntos flacos del producto de software. Es interesante porque, para obtener mayores beneficios, necesita una filosofía general diferente de la que se utiliza en las disciplinas de Requisitos, Análisis & diseño e Implementación. Una sutil diferencia es que estas tres disciplinas se centran en la completitud, mientras que la Prueba se centra en la falta de esta.**(42)**

4.3.1 Niveles de Prueba

El proceso de realización de las pruebas está compuesto por una serie de niveles entre los que se encuentran: el nivel de pruebas unitarias, el nivel de pruebas de integración, el nivel de pruebas del sistema y nivel de pruebas de aceptación. Tras la culminación de la etapa de implementación del sistema este fue sometido a algunos de estos niveles con el objetivo de detectar los errores existentes. A continuación se exponen los niveles utilizados.

- **Pruebas Unitarias:** las pruebas de unidad se concentran en la lógica del procesamiento interno y en las estructuras de datos de los límites de un componente. Se le aplican a cada módulo de un software de manera independiente. Su objetivo es verificar que el módulo, entendido como una unidad funcional de un programa independiente, está correctamente codificado.**(2)**

  El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas.
- **Pruebas de Integración:** es una técnica sistemática para construir la arquitectura del software mientras, al mismo tiempo, se aplican las pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar componentes a los que se le aplicó una prueba de unidad y construir una estructura de programa que determine el diseño.**(2)**

  Se refiere a la (s) prueba (s) de todos los elementos unitarios que componen un proceso, realizadas en conjunto de una sola vez. Consiste en efectuar pruebas para verificar que un gran conjunto de partes del software funcionan unidos.

- **Pruebas del Sistema**: tienen como propósito fundamental ejercitar profundamente el sistema desarrollado, con el objetivo de verificar que se hayan integrado correctamente todos los elementos del sistema y que realizan correctamente las funciones descritas.**(2)**

  Este tipo de pruebas estudia el producto completo para analizar defectos globales o para estudiar aspectos específicos de su comportamiento, tales como seguridad o rendimiento.

4.3.2 Técnicas de prueba

La metodología RUP propone 2 técnicas de pruebas fundamentales con el objetivo de validar el software, estas son: Pruebas de Caja Blanca y las Pruebas de Caja Negra. A continuación se describen ambas técnicas, realizándose mayor énfasis en la técnica de Caja Negra, teniendo en cuenta que será la utilizada en la comprobación de la presente solución.

- **Pruebas de Caja Blanca:** denominada a veces prueba de caja de cristal, es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante los métodos de prueba de Caja Blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada módulo, ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa; ejecuten todos los ciclos en sus límites y con sus límites operacionales y ejerciten las estructuras internas de datos para asegurar su validez.**(2)**
- **Pruebas de Caja Negra:** las pruebas de Caja Negra, también denominadas pruebas de comportamiento, se concentran en los requisitos funcionales del *software*. Es decir, permiten al ingeniero de *software* derivar un conjunto de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa.

  El objetivo es demostrar que las funciones del *software* son operativas, que las entradas se aceptan de forma adecuada, que se produzca un resultado correcto y que la integridad de la información externa se mantiene, saber qué es lo que hace el *software* pero sin entrar en detalles de código, es decir, qué es lo que hace, y no cómo lo hace (no se ve el código). Estas pruebas se

centran principalmente en los requisitos funcionales del *software* y permiten encontrar: **(2)**

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

4.3.3 Resultados obtenidos

Durante la etapa de desarrollo del módulo Sistemas educativos de la Plataforma Educativa Zera se realizaron un conjunto de pruebas con el objetivo de evaluar el correcto desempeño de las funcionalidades propuestas. Prestando especial atención en las pruebas de integración, con vistas a demostrar la correcta integración del módulo con el resto de los módulos de la Plataforma.

Las pruebas realizas se muestran a continuación, pertenecen al nivel de sistema, utilizando el método de Caja Negra:

- **Pruebas Internas:** son realizadas por el equipo de calidad interna del proyecto (en este caso las analistas), con el objetivo de entregar todo la documentación generada con la menor cantidad de errores posibles, centrándose en la revisión exhaustiva de la misma.
- **Pruebas Cruzadas:** fueron realizadas por el equipo de desarrollo del proyecto (analistas y desarrolladores de los diferentes módulos) en aras de detectar la mayor cantidad de errores en cuanto a validaciones, pautas de diseño y formato de los campos, entre otras.
- **Pruebas de Liberación:** fueron realizadas en conjunto con el equipo de calidad interno del proyecto (en este caso los analistas), encargados de validar que el software cuenta con la calidad requerida para ser liberado y por personal perteneciente al Proyecto de Calidad del Centro FORTES.

A continuación se presentan los resultados arrojados por las diferentes pruebas aplicadas:

Pruebas Internas:

Módulo	Casos de uso	Iteración	NC	Cerrada (s)	No procede
Sistemas educativos	15	1ra	19	19	0
		2da	5	5	0
De estas NC obtenidas:					
Alta		Media		Baja	
0		9		15	

Tabla 5: Resultados de las Pruebas Internas.

Pruebas Cruzadas:

Módulo	Casos de uso	Iteración	NC	Cerrada (s)	No procede
Sistemas educativos	15	1ra	7	5	2
		2da	5	4	1
De estas NC obtenidas:					
Alta		Media		Baja	
0		2		10	

Tabla 6: Resultados de las Pruebas Cruzadas.

Pruebas de Liberación:

Módulo	Casos de uso	Iteración	NC	Cerrada (s)	No procede
Sistemas educativos	15	1ra	2	1	1
		2da	1	0	1
De estas NC obtenidas:					
Alta		Media		Baja	
0		0		3	

Tabla 7: Resultados de las Pruebas de Liberación.

4.4 Conclusiones parciales

En este capítulo se modelaron los diagramas correspondientes a la reimplementación del módulo Sistemas educativos: el Diagrama de Despliegue el cual representa la distribución física por nodos de la aplicación y el Diagrama de Componentes que refleja la dependencia de los mismos con la base de datos. Además, las pruebas realizadas validaron que las funcionalidades desarrolladas satisfacen los requisitos especificados, dejando listo el módulo con vista a su utilización.

Conclusiones Generales

Al concluir el presente trabajo de diploma se le ha dado cumplimiento al objetivo general y los objetivos específicos definidos en un inicio de la investigación, dando cumplimiento a la reconstrucción propuesta para el módulo Sistemas educativos de la Plataforma Educativa Zera. Para iniciar la investigación fue necesario indagar en algunos sistemas y software educativos sobre las deficiencias detectadas en el módulo, no encontrando así, sistemas ni software con características tan específicas como las que posee la Plataforma Educativa Zera. Se identificaron y explicaron los principales conceptos relacionados con el trabajo para lograr un mejor entendimiento del mismo, como es el caso de los Sistemas educativos, Sistema curricular, Periodos lectivos, Programas de estudio y los Sistemas evaluativos, dentro de este último la Escala de calificación, los Periodos evaluativos y la Ponderación de los mismos. Se realizó un análisis minucioso de las metodologías, lenguajes de modelado y herramientas que fueron estudiadas anteriormente en la construcción del módulo, quedando definidos los RF y los RNF del software en cuestión. Posteriormente se realizó todo el análisis, diseño, implementación y prueba del módulo Sistemas educativos. A continuación se muestra a grandes rasgos las pruebas más fehacientes del cumplimiento de los objetivos a grandes rasgos:

- Se obtuvo toda la documentación generada durante el proceso de reconstrucción del módulo, como cumplimiento de las actividades planificadas.
- Se le dio solución a los problemas de ediciones, con la utilización de un solo árbol para lograr una mejor gestión del proceso por parte de los usuarios.
- Se logró que después de actualizar un programa de estudio en la plataforma central, les llegara un aviso a las instituciones que estuvieran utilizando el mismo programa de estudio y en caso de aceptar, se ejecute la modificación integrándose esta con la versión que se encuentra en la plataforma local.
- Nunca va a encontrarse un programa de estudio sin un periodo evaluativo asignado.
- Se demostró a partir de las diferentes iteraciones de pruebas practicadas al módulo, que el mismo

satisface los RF y los RNF obtenidos durante el flujo de trabajo de Requerimientos.

Recomendaciones

- La eliminación de los procesos que provoquen dependencia innecesaria como es el caso de los sistemas curriculares, con esta eliminación no será necesario para consultar un programa de estudio ir a los sistemas curriculares, seleccionar una Materia, para luego remitirse al programa de estudio correspondiente, sino que se irá directamente a los programas de estudio donde existirá una lista de estos, correspondientes a cada asignatura impartida en la institución.
- Crear programas de estudio individuales sin tener que estar asociados a un sistema educativo en particular.
- Mejorar el proceso actual de gestión de los sistemas educativos, ya que se hace engorroso y complejo para el usuario final la cantidad de pasos que conlleva crear uno de estos.
- Mejorar el problema de las copias que realizan las instituciones educativas de los sistemas educativos que ya están creados, se propone diseñar y guardar estas copias o personalizaciones en un espacio diseñado en la base de datos de la aplicación y que sea capaz de adaptarse a los cambios que puedan sufrir estas copias, independientemente de cuál sea el cambio.
- Lograr la personalizar de los demás procesos que componen el módulo Sistemas educativos y extender estas actualizaciones a las instituciones que utilicen el sistema educativo que se encuentra en la plataforma central, debido a que esta acción se consiguió únicamente para los programas de estudio.
- Tener en cuenta el trabajo presentado para proyectos futuros, siempre y cuando se cumpla con las normas de confidencialidad establecidas.

Glosario de Términos

Learning Management System (LMS): un LMS es una aplicación de software instalado en un servidor, que se emplea para administrar, distribuir y controlar las actividades de formación no presencial o e-Learning(Educación a distancia) de una institución u organización.**(43)**

Plataforma Educativa: las plataformas educativas son herramientas que permiten interactuar con varios usuarios a la vez con fines pedagógicos. Usadas para organizar e implantar cursos en línea o actividades educativas en general. Software que permite a un profesor, crear un espacio en la web donde sea capaz de colgar todos los materiales que crea son de interés para los usuarios que recibirán el curso, enlazar tantos otros, incluir foros, wikis, recibir tareas de sus alumnos, desarrollar cuestionarios, promover debates, chats, obtener estadísticas de evaluación y uso, etc.**(44)**

módulo: porción funcional del software, que compone a un sistema informático. Se le denomina al conjunto de elementos fundamentales que componen el producto.

Metodologías: las metodologías de desarrollo de software son utilizadas para obtener un software de alta calidad, en el tiempo planificado y que satisfaga las necesidades del cliente. En la actualidad existen muchas metodologías de desarrollo, todas creadas para solucionar problemas tanto generales como específicos de algunos tipos de proyectos de desarrollo.**(45)**

Caso de Prueba: conjunto de condiciones o variables bajo las cuáles el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio.

Gestión académica: la gestión académica permite asesorar, programar y gestionar hacia el mejoramiento de la calidad académica, en coherencia con la producción de procesos educativos y la identificación de áreas críticas de funcionamiento y el alineamiento de los planes de desarrollo en concordancia al Plan Estratégico Institucional en lo académico. Para ello se trazan planes con el objetivo de mejorar y fortalecer los aspectos provechosos, así como se realizan seguimientos a los planes de estudio para detectar deficiencias.

Referencias Bibliográficas

1. DEFINICIÓN ABC. Definición de Sistema educativo y concepto. In: *Definición ABC una guía única en la red* [online]. 2007. [Accessed 12 January 2011]. Available from: http://www.definicionabc.com/social/sistema-educativo.php.

2. PRESSMAN, Roger. *Ingeniería del Software. Un enfoque Práctico.* [online]. S.l.: s.n., [no date]. Available from: eva.uci.cu.

3. SICILIA, Miguel Angel. ¿Qué es Reingeniería del Software? In: [online]. 9 2009. [Accessed 25 March 2012]. Available from: http://cnx.org/content/m17438/latest/. Breve descripción de Reingeniería del Software y beneficios de aplicarlo.

4. Entornos Virtuales de Aprendizaje - Moodle. In: [online]. [Accessed 7 October 2011]. Available from: http://www.wiziq.com/tutorial/3237-Entornos-Virtuales-de-Aprendizaje-Moodle.

5. Edu 2.0 Una Nueva Herramienta Para Las Escuelas y gratis.. In: [online]. [Accessed 18 October 2011]. Available from: http://ayudasparadocentes.blogspot.com/2009/09/una-nueva-herramienta-para-nosotros.html.

6. PECQUET, Emmanuel. *Manual del docente Dokeos 1.8* [online]. April 2007. S.l.: s.n. Available from: http://www.dokeos.com/es/documentacion/manuales-en-pdf.

7. HERNÁNDEZ LÓPEZ, Nitza M. TecnoInfo - Una comunidad de aprendizaje en línea: Sakai – Plataforma educativa de código abierto. In: [online]. 2010. [Accessed 13 January 2011]. Available from: http://tecnoinfo6300.blogspot.com/2010/10/sakai-plataforma-educativa-de-codigo.html.

8. SEPÚLVEDA PEÑA, Juan Carlos. *Plataforma de Teleformación aprenDIST* [online]. 2005. S.l.: s.n. Available from: http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBUQFjAA&url=http%3A%2F

%2Fwww.forumcyt.cu%2FUserFiles%2Fforum%2FTextos%2F0301877.pdf&ei=zRkuTYrAKo-u8AbDvpXCCQ&usg=AFQjCNGLXObkIrDUwdZ9koI6pII1IX_Apw.

9. TAMAYO PALMA, D.E and AVILA PORTALES, Y. *Análisis y Diseño de los Procesos de gestión de carreras para el sistema Akademos v2.0*. S.l.: s.n., 2008.

10. ARAUJO, Yuriana. Metodologia RUP. In: [online]. May 2010. [Accessed 14 January 2011]. Available from: http://www.scribd.com/doc/31440864/Metodologia-RUP.

11. LETELIER, Patricio and PENADÉS, Mª Carmen. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). In: *www.cyta.com.ar/ta0502/v5n2a1.htm* [online]. 15 April 2006. [Accessed 10 April 2012]. Available from: http://www.cyta.com.ar/ta0502/v5n2a1.htm.

12. MSF : La metodología aplicada. In: [online]. [Accessed 11 April 2012]. Available from: http://www.willydev.net/descargas/Articulos/General/MSF.aspx.

13. CIUDAD RICARDO, Febe Angel and HERRERA MARTÍNEZ, Yosnel. Una aplicación del Lenguaje de Modelado Orientado a Objetos para Aplicaciones Educativas y ultimedia (ApEM –L) en su versión 1.5. In: .

14. Cientec. In: [online]. [Accessed 11 April 2012]. Available from: http://www.cientec.com/analisis/ana-uml.html.

15. ApEM-L - EcuRed. In: [online]. [Accessed 11 April 2012]. Available from: http://www.ecured.cu/index.php/ApEM-L.

16. EcuRed. OMMMA-L. In: [online]. 2011. [Accessed 20 May 2011]. Available from: http://www.ecured.cu/index.php/OMMMA-L.

17. MENÉNDEZ ALONSO, Evelyn. Herramientas CASE proceso desarrollo Software. In: [online]. 2009.

[Accessed 18 May 2011]. Available from: http://www.plusformacion.com/Recursos/r/Herramientas-CASE-para-proceso-desarrollo-Software.

18. Modelando con UML en Linux - Dia - Wikilearning. In: [online]. [Accessed 11 April 2012]. Available from: http://www.wikilearning.com/articulo/modelando_con_uml_en_linux/330-3.

19. Modelando con UML en Linux - ArgoUML - Wikilearning. In: [online]. [Accessed 11 April 2012]. Available from: http://www.wikilearning.com/articulo/modelando_con_uml_en_linux/330-4.

20. ALFARO, Félix Murillo. *Herramientas Case*. S.l.: s.n., 1999.

21. WEB, Consortium World Wide. Guía Breve de XHTML. In: [online]. [Accessed 18 December 2010]. Available from: http://www.w3c.es/divulgacion/guiasbreves/XHTML.

22. CODEBOX. CodeBox : Glosario. In: [online]. 2008. [Accessed 18 December 2010]. Available from: http://www.codebox.es/glosario.

23. PÉREZ, Javier Eguíluz. Introducción a JavaScript | LibrosWeb.es. In: [online]. [Accessed 10 January 2011]. Available from: http://www.librosweb.es/javascript/.

24. ALVAREZ, Miguel. Qué es Python. In: [online]. [Accessed 9 October 2011]. Available from: http://www.desarrolloweb.com/articulos/1325.php.

25. ALVAREZ, Miguel Angel. Qué es PHP. In: [online]. [Accessed 13 January 2011]. Available from: http://www.desarrolloweb.com/articulos/392.php.

26. Lenguajes de Programación. In: [online]. [Accessed 9 October 2011]. Available from: http://www.frt.utn.edu.ar/sistemas/paradigmas/lenguajes.htm#_PHP.

27. ALVAREZ, Miguel Angel. *Introducción a jQuery*. 2010. S.l.: s.n.

28. BLUEPRINT.ORG. Blueprint. In: [online]. [Accessed 14 January 2011]. Available from: http://www.blueprintcss.org/.

29. BERNARDINO, Jepser. 960 Grid, un framework para CSS /// Jepser Bernardino. In: [online]. [Accessed 9 October 2011]. Available from: http://jepserbernardino.com/idea/960-grid-un-framework-para-css/.

30. JQUERY.COM. jQuery: The Write Less, Do More, JavaScript Library. In: [online]. 2010. [Accessed 14 January 2011]. Available from: http://jquery.com/.

31. PÉREZ, Javier Eguíluz. Introducción a AJAX | LibrosWeb.es. In: *Libros Web* [online]. [Accessed 18 December 2010]. Available from: http://www.librosweb.es/ajax/index.html.

32. POTENCIER, Fabien and ZANINOTTO, François. *Symfony la guía definitiva*. S.l.: s.n., 2008.

33. ALVAREZ, Miguel Angel. *Manual de CodeIgniter*. S.l.: s.n., 2010.

34. DOCTRINE-PROJECT.ORG. Doctrine - Object Relational Mapper Documentation (2.0). In: [online]. 2010. [Accessed 15 January 2011]. Available from: http://www.doctrine-project.org/projects/orm/2.0/docs/en.

35. PROPELORM.ORG. Propel ORM. In: [online]. 2010. [Accessed 15 January 2011]. Available from: http://www.propelorm.org/.

36. PostgreSQL: Dossier de Prensa de PostgreSQL 9.0. In: [online]. [Accessed 30 May 2012]. Available from: http://www.postgresql.org/about/press/presskit90/es/.

37. NETBEANS.ORG. Bienvenido a NetBeans y www.netbeans.org, Portal del IDE Java de Código Abierto. In: [online]. 2011. [Accessed 14 January 2011]. Available from: http://netbeans.org/index_es.html.

38. EcuRed. Patrones de Casos de Uso. In: [online]. 2011. [Accessed 18 May 2011]. Available from: http://www.ecured.cu/index.php/Patrones_de_Casos_de_Uso.

39. Departamento Docente Central de IWS UCI. *Patrones de Caso de Uso* [online]. 2011. S.l.: s.n. Available from: http://eva.uci.cu/mod/resource/view.php?inpopup=true&id=34917.

40. *Introducción a la Disciplina Análisis y Diseño* [online]. 2011. S.l.: s.n. Available from: http://eva.uci.cu/mod/resource/view.php?id=35381.

41. POTENCIER, Fabien and ZANINOTTO, François. *Symfony la guía definitiva*. S.l.: s.n., 2008.

42. Rational Software Architect. In: [online]. 2006. [Accessed 21 May 2012]. Available from: file:///media/Datos/usuarios/Diosbel/Documentacion/RUP/RUP Espa_ol/index.htm.

43. Sistemas de Gestión de Aprendizaje o Learning Management System (LMS). In: [online]. [Accessed 7 October 2011]. Available from: http://www.scholarium.co/index.php?option=com_content&view=article&id=62&Itemid=112.

44. AGUDELO B, Mónica María. Plataformas educativas. In: [online]. October 2011. [Accessed 7 October 2011]. Available from: http://aprendeenlinea.udea.edu.co/banco/html/plataformaseducativas/.

45. Metodologías de desarrollo de software. ¿Cuál es el camino? - Monografias.com. In: [online]. [Accessed 9 April 2012]. Available from: http://www.monografias.com/trabajos60/metodologias-desarrollo-software/metodologias-desarrollo-software.shtml.

Bibliografías Consultadas

DTMI: , Diseño teórico y metodológico de la investigación,

EPUS: Ivar Jacobson, Grady Booch, James Rumbaugh, El Proceso Unificado deSoftware,

EBDI: Facultad de Infromática U.V, ,

SAHFAE: , Sistema de acciones para las habilidades fundamentales de la actividad de estudio ,

EICRA: Rolando Alfredo Hernández León, Sayda Coello González, El proceso de investigación científica,

LMCIP: Dr. Julio Cerezal Mezquita, Dr. Jorge Fiallo Rodríguez, Los métodos científicos en las investigaciones pedagógicas, 2002

UOMIC: Dr. Cs. Carlos Alvarez de Zayas, Universidad de oriente, metodología de la investigación científica,

MIEPP: Gastón Pérez Rodríguez, Gilberto García Batista, Irma Nocedo de León, Miriam Lucy García Inza, Metodología de la Investigación educacional, Primera Parte, 1996

MICS: M en C. Roberto Hernández Sampieri, Metodología de la investigación,

RSA: , Rational Software Architect,

PS: Dr. Macario Polo Usaola, Pruebas del Software,