Facultad de Ciencias y Tecnologías Computacionales

Módulo de Idiomas para el Sistema Automatizado de Gestión Académica (Akademos) de la Universidad de las Ciencias Informáticas.

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

Autor(es): José Antonio Ávalos Vielza
Manuel Alejandro Fernández Rojo
Tutor(es): Ing. Nayilet Martín Soler.
Dr. C.Yoan Martinez Márquez.

La Habana, diciembre del 2022

Año 63 de la Revolución

DECLARACIÓN DE AUTORÍA

Los autores del trabajo de diploma con título ***"Módulo de Idiomas para el Sistema Automatizado de Gestión Académica (Akademos) de la Universidad de las Ciencias Informáticas"*** conceden a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declaran como únicos autores de su contenido. Para que así conste firman la presente a los ____ días del mes de _____ del año _____.

José Antonio Ávalos Vielza	**Manuel Alejandro Fernández Rojo**
_____ Firma del Autor	_____ Firma del Autor
Yoan Martinez Márquez	**Nayilet Martín Soler**
_____ Firma del Tutor	_____ Firma del Tutor

AGRADECIMIENTOS

José Antonio Ávalos Vielza

A mis seres queridos:

Quisiera tomarme un momento para agradecer a todas las personas que han formado parte activa en mi vida. A mis padres y abuelos por educarme y enseñarme el valor de las cosas, a diferenciar entre lo importante y lo efímero de este mundo, a buscar personas especiales y no dejarlas ir. A Camilo y su familia, que sin yo pertenecer a ella me acogieron con más amor del que puedo expresar en palabras, y también me enseñaron que si se puede tener una familia sin compartir el mismo grupo sanguíneo.

A mis compañeros:

Ustedes que hicieron de estos años recuerdos memorables e irrepetibles, nunca olvidaré las clases, locuras y dificultades que pasamos juntos. Los momentos en el comedor, esas cosas no se olvidan.

A mis tutores por sus consejos, y toda la ayuda que nos brindaron.

Por último, a mi compañero de tesis, Manuel, por el apoyo durante todo este proceso. Manolo we did it!!!!!!!!!!

A todos ellos: Gracias.

Manuel Alejandro Fernández Rojo

Quiero agradecerles a mis padres Dulce María y Manuel, por traerme al mundo y convertirme en la persona que soy, por apoyarme en todas mis decisiones, y estar siempre para mí cuando lo he necesitado. Además, son parte esencial e incondicional de cada meta que me propongo y donde siempre encuentro ayuda oportuna, consejo sabio y sobre todo amor infinito.

A mis tíos Pedro, Tulio y tía Leidi por estar siempre a mi lado y apoyarme, por convertirse en otros padres para mí gracias a ellos he podido realizar este sueño.

A mis hermanos Victor y Yoneisi, que siempre me dan consejos y están ahí siempre que los necesito. Ellos que en el día a día con su presencia, respaldo y cariño me impulsan para seguir adelante, además de saber que mis logros también son los suyos.

A mis amigos de toda la vida Andy, Roland y Yunior, que han sido como hermanos para mí y que adonde quiera

que vayan, nuestra amistad siempre existirá.

A mi padrastro Juan y mi madrastra Yisel, que siempre me han apoyado incondicionalmente en mis estudios.

A mi compañero de tesis José, por acompañarme siempre hasta el final. De verdad que gracias.

A todas mis amistades de la UCI que vienen desde 1ro y a los que he conocido con el tiempo, gracias por las risas, las fiestas, las noches de estudio, por dejarme formar parte de sus vidas.

A mis tutores, gracias por su apoyo en el desarrollo del trabajo, por su paciencia, sus consejos y todo el tiempo que dedicaron en ayudarnos.

Por último, a mis compañeros de apartamento, Ariel, Enmanuel y Yosley.

DEDICATORIA

José Antonio Ávalos Vielza

Quisiera dedicar esta tesis a Yusimi, Débora y Armando, las personas que tanto tiempo han invertido en mi educación y han hecho de mi la persona que soy.

A mis compañeros de carrera que me han acompañado en todo momento con su presencia y alegría; sus bromas y ocurrencias, pero sobre todo eso, su sincera amistad.

Manuel Alejandro Fernández Rojo

Quiero dedicar esta tesis a mis padres Manuel y Dulce María porque ellos han dado razón a mi vida, por sus consejos, su apoyo incondicional y su paciencia, todo lo que soy es gracias a ellos.

A mis hermanos Victor y Yoneisi que más que hermanos son mis verdaderos amigos.

A toda mi familia que es lo mejor y más valioso que tengo en la vida.

En especial, a mi abuela Aleida, que desde el cielo me apoya cada día.

RESUMEN

La presente investigación tiene como objetivo el desarrollo de un sistema que permita gestionar la información asociada a los exámenes estandarizados de idiomas en la Universidad de las Ciencias Informáticas. Actualmente, este proceso se realiza en el Centro de Idiomas (CENID) de dicha institución. Este centro se encarga de facilitar a la comunidad universitaria y extra-universitaria, el aprendizaje y uso de competencias comunicativas en idiomas, a través de servicios especializados de formación, evaluación, certificación, traducción e interpretación.

El CENID ofrece una serie de servicios a la comunidad universitaria, entre los que se encuentran los cursos de idiomas, el laboratorio de auto-acceso y la traducción a diferentes idiomas a través de un Grupo de Traducción de este centro. Este centro necesita hoy en día contar con un sistema que permita gestionar todo el proceso de participación de todos los estudiantes de la UCI en los exámenes de certificación, así como la otorgación del requisito de graduación a los estudiantes que aprueben de manera satisfactoria estos exámenes.

Para darle solución a esto se desarrolló un Módulo de Idiomas en el Sistema Automatizado de Gestión Académica (Akademos) de dicha universidad. Para el desarrollo de la aplicación se utilizó la metodología de Desarrollo AUP en su variante UCI (AUP-UCI) y el entorno tecnológico definido por el Departamento de Desarrollo de la Dirección de Informatización. Dentro de este entorno se emplearon los lenguajes de programación html, css, java, php y como herramientas PostgreSQL, para la gestión de la base de datos.

Además, se utilizó como entorno de desarrollo el software PHPStorm y como herramienta de modelado Visual Paradigm. Por otra parte, para la descripción de los requisitos se confeccionaron las historias de usuario respectivas de cada uno de estos. Por último, se le realizó al sistema las pruebas necesarias para validar su correcto funcionamiento y para verificar que cumpla con los requisitos establecidos por el cliente.

PALABRAS CLAVE

Exámenes Estandarizados, Centro de Idiomas, Módulo de Idiomas, Akademos, Metodología AUP-UCI.

ABSTRACT

The objective of this research is the development of a system that allows managing the information associated with the standardized language exams at the University of Informatics Sciences. Currently, this process is carried out at the Language Center (CENID) of said institution. This center is in charge of facilitating the university and extra-university community, the learning and use of communication skills in languages, through specialized training, evaluation, certification, translation and interpretation services.

CENID offers a series of services to the university community, among which are language courses, the self-access laboratory and translation into different languages through a Translation Group at this center. This center currently needs to have a system that allows managing the entire participation process of all UCI students in the certification exams, as well as granting the graduation requirement to students who satisfactorily pass these exams.

To solve this, a Language Module was developed in the Automated Academic Management System (Akademos) of said university. For the development of the application, the AUP Development methodology was used in its UCI variant (AUP-UCI) and the technological environment defined by the Development Department of the Computerization Department. Within this environment, the programming languages html, css, java, php and PostgreSQL were used as tools for database management.

In addition, the PHPStorm software was used as a development environment and Visual Paradigm as a modeling tool. On the other hand, for the description of the requirements, the respective user stories of each of these were made. Finally, the necessary tests were carried

out on the system to validate its correct operation and to verify that it complies with the requirements established by the client.

KEYWORDS

Standardized Exams, Language Center, Language Module, Akademos, AUP-UCI Methodology.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

OPINIÓN DEL(OS) TUTOR(ES)

AVAL DEL CLIENTE

INTRODUCCIÓN

En el mundo de hoy, las lenguas o idiomas forman parte clave de nuestra cultura moderna, puesto que ayudan a ampliar el conocimiento e interactuar con personas de diferentes partes del mundo. Cada día se emplea más en casi todas las áreas del conocimiento y desarrollo humano. El inglés es considerado el idioma más usado en el mundo, por lo que muchas instituciones educativas lo integran en su currículo. Al hablar del inglés como idioma extranjero se hace referencia al aprendizaje de un idioma diferente al de la lengua materna, y que además no es el que se emplea en la vida cotidiana del estudiante o el medio en el cual desarrolla sus actividades [CITATION Mar17 \l 3082].

El proceso de aprendizaje del inglés como lengua extranjera generalmente se da dentro del aula, lugar en el cual se realizan diferentes actividades de tipo controladas. A pesar de que este proceso se da en su mayor parte en el ámbito educativo, los estudiantes pueden alcanzar un alto grado de desarrollo del idioma Inglés [CITATION Mar17 \l 3082]. Indudablemente, el inglés es un idioma que está generando grandes oportunidades no solo para nuestras vidas como profesionales, sino también como agentes transmisores de una cultura mundial orientada hacia la búsqueda del bien común.

El impacto de la investigación como factor transformador de nuestras sociedades tampoco puede desconocerse, y es aquí donde una lengua mundial, como el inglés, fortalece las posibilidades del desarrollo científico, cultural, económico y humanístico[CITATION NIÑ13 \l 3082]. El aprendizaje de la lengua inglesa constituye, hoy, un reclamo y una política de la Educación Superior Cubana en la formación de un profesional capaz de comunicarse eficazmente en una lengua extranjera en los ámbitos cotidiano, académico, y profesional.

Resulta evidente que el dominio de al menos un idioma extranjero de relieve internacional se convierte en una competencia clave del profesional en la actualidad, surgida de la necesidad de comunicación e intercambio entre los países en un mundo cada vez más globalizado [CITATION Mor08 \l 3082]. En Cuba la política de perfeccionamiento de la enseñanza del idioma inglés está orientada a que se pueda facilitar un proceso de formación integral de

nuestros estudiantes, como parte del ejercicio futuro de la profesión [CITATION Gue18 \l 3082].

Las características más significativas de la estrategia son la creación de una estructura para la gestión de la formación de los profesionales, los llamados Centros de idiomas; la necesidad de contar con la tecnología suficiente, de ahí el surgimiento de Centros de auto-acceso; así como la incorporación de las tendencias internacionales a la metodología de la enseñanza del inglés. Precisamente, ante la escasa preparación de los niveles precedentes, la estrategia plantea que la enseñanza debe partir, de ser necesario, de un punto equivalente a cero [CITATION Cen16 \l 3082].

Además, se han realizado exámenes iniciales de colocación, que permiten ubicar a los estudiantes en grupos, por niveles de dominio, y también convocatorias de exámenes de certificación, para avalar si el alumno ha cumplido el requisito de evaluación. En un grupo importante de las universidades se implementa la política, teniendo en cuenta la evaluación de sus condiciones. En algunas, ha habido un proceso de aplicación más efectivo y avanzado en la ejecución, como la Universidad de las Ciencias Informáticas, las de Villa Clara, Cienfuegos, Holguín y Santiago de Cuba [CITATION Gue18 \l 3082].

En la Universidad de las Ciencias Informáticas (UCI), la política se ha implementado en todas sus carreras, teniendo como soporte el Centro de Idiomas (CENID). El Centro de Idiomas (CENID) es un centro de enseñanza y certificación de lenguas, líder en la comunidad educativa, que se distingue por su profesionalismo en la ejecución de programas, servicios y productos innovadores de alta calidad, con un impacto en la internacionalización de la Universidad de las Ciencias Informáticas, en función de la preparación de profesionales comprometidos con su país y su tiempo, en un mundo multilingüe y multicultural.

Su misión es facilitar, a la comunidad universitaria y extra-universitaria, el aprendizaje y uso de competencias comunicativas en idiomas, a través de servicios especializados de formación, evaluación, certificación, traducción e interpretación [CITATION Cen161 \l 3082]. El CENID ofrece una serie de servicios a la comunidad universitaria, entre los que se

encuentran los **cursos**, el **laboratorio de auto-acceso** y la traducción a diferentes idiomas a través de un **Grupo de Traducción** de este centro.

El Centro de Idiomas de la UCI ofrece **cursos** presenciales y semipresenciales en varios de los niveles del Marco Común Europeo de Referencia para las lenguas (MCER) tanto para el pregrado como para el postgrado y trabajadores en general, según la disponibilidad de profesores y las necesidades específicas de la institución. Los cursos que allí se ofrecen son Inglés Nivel A1, Inglés Nivel A2, Inglés Nivel B1 e Inglés Nivel B2 [CITATION Cen161 \l 3082]. El Marco Común Europeo de Referencia para las lenguas (MCER) es el estándar internacional que define la competencia lingüística.

Se utiliza en todo el mundo para **definir las destrezas lingüísticas** de los estudiantes y dada su estructura sencilla, los exámenes son fáciles de entender tanto para profesores de inglés como para estudiantes [CITATION Cam22 \l 3082]. El **laboratorio de Auto-acceso del Centro de Idiomas** le ofrece al usuario distintas posibilidades como apoyo para que, de modo independiente y a su propio ritmo, organice su aprendizaje y elija los materiales apropiados para mejorar sus habilidades y reducir sus debilidades en el uso de un idioma extranjero [CITATION Cen162 \l 3082].

El Grupo de Traducción del CENID se distingue por la calidad en los servicios especializados de traducción e interpretación que ofrece y que tienen un impacto en la internacionalización de la Universidad de las Ciencias Informáticas (UCI). Su **misión** es facilitar a la comunidad universitaria servicios especializados de traducción de documentos y aplicaciones informáticas como soporte a los procesos fundamentales de la UCI y a su internacionalización comercial, científica y académica. Además, ofrece servicios especializados de interpretación para conferencias, eventos científicos y protocolares e intercambios con visitantes extranjeros [CITATION Cen163 \l 3082].

Por otro lado, agrupa 18 profesionales de la rama lingüística, pedagógica e informática que participan como colaboradores para:

- Realizar servicios de traducción e interpretación en idioma inglés, ruso y francés.

- Traducir materiales impresos y digitales relacionados con temas de la especialidad en ciencias informáticas con vistas a la promoción y comercialización de los productos y servicios informáticos de la UCI.

- Servir como intérpretes en conferencias, eventos científicos, culturales, académicos y protocolares o cualquier otro intercambio con visitantes extranjeros [CITATION Cen163 \l 3082].

A pesar de los satisfactorios resultados obtenidos, los procesos que se llevan a cabo se realizan de forma semi-informatizada, presentando las siguientes limitaciones:

- La recopilación de los datos de los usuarios, así como el almacenamiento de sus registros a través del uso de herramientas ofimáticas, retrasan y dificultan el manejo de la información debido a la magnitud de los datos que se procesan y los posibles riesgos de errores humanos en el tratamiento de documentos, como errores ortográficos e información incorrecta, duplicada o ausente.
- La generación de reportes (sobre el estado de los usuarios para instituciones superiores) y certificados (para la entrega de reconocimientos del dominio del idioma) carecen de un control de estandarización, posibilitando la invalidación de los mismos y el gasto innecesario de recursos tales como: materiales de oficina y esfuerzo humano.
- Los datos almacenados se encuentran descentralizados ralentizando la disponibilidad de información actualizada.

En correlación con la situación problemática antes expuesta se deriva como **problema de investigación** lo siguiente: ¿Cómo gestionar la información asociada a los exámenes estandarizados de idiomas en la Universidad de las Ciencias Informáticas? La presente investigación centra su **objeto de estudio** en: la gestión de la información asociada a los exámenes estandarizados de idiomas. El **campo de acción** se enmarca en la gestión de la información asociada a los exámenes estandarizados de idiomas en la Universidad de las Ciencias Informáticas. Se propone como **objetivo general** desarrollar el Módulo de Idiomas para la gestión de la información asociada a los exámenes estandarizados de idiomas en el

Sistema Automatizado de Gestión Académica (Akademos) de la Universidad de las Ciencias Informáticas.

Este sistema será implementado para el Centro de Idiomas (CENID) de la universidad para informatizar la gestión de los cursos de idiomas en la UCI y la gestión de la información relacionada con los procesos que allí se desarrollan. Este nuevo sistema es necesario desarrollarlo, ya que algunos de estos procesos todavía se desarrollan de forma manual.

Se definen los siguientes **objetivos específicos**:

1. Caracterizar los fundamentos teóricos-metodológicos de la investigación asociados a la gestión de cursos académicos.
2. Diseñar la propuesta de solución a partir del proceso de desarrollo de software.
3. Desarrollar el Módulo de Idiomas para el Sistema Automatizado de Gestión Académica (Akademos).
4. Validar la solución desarrollada.

Para garantizar el cumplimiento de los objetivos específicos trazados se establecen las siguientes **tareas de investigación**:

1. Caracterización de los sistemas informáticos utilizados para la gestión de cursos académicos.

2. Caracterización de las herramientas y el proceso de desarrollo de software a emplear en la propuesta de solución.

3. Identificación de los requisitos funcionales y no funcionales de la propuesta de solución.

4. Implementación de las funcionalidades del módulo Idiomas.

5. Caracterización de las pruebas a realizar a la solución.

6. Validación de la propuesta de solución.

Durante el desarrollo de la investigación se emplean los siguientes **métodos científicos:**

Los **métodos teóricos** permitieron estudiar en profundidad la gestión de cursos académicos. Dentro de los métodos teóricos existentes se estudiaron los siguientes:

Analítico-Sintético: Permite el análisis y la realización de la síntesis de la bibliografía, lo que favorece la investigación, permitiendo conocer con profundidad los aspectos fundamentales relacionados con la gestión de cursos académicos.

Modelación: Permite la modelación de los diagramas que facilitan la implementación del Módulo para la gestión de cursos de idioma en la Universidad de las Ciencias Informáticas a desarrollar.

Los **métodos empíricos** estudiados posibilitaron la revelación de las características relevantes del objeto, a través del análisis preliminar de la información relacionada con la gestión de cursos académicos. Los mismos aportan los datos que son procesados para corroborar la magnitud y solución del problema. Entre los métodos empíricos existentes se estudiaron los siguientes:

La observación: Permite estudiar más de cerca el objeto de la investigación, las acciones, causas y consecuencias. Se pudo observar cómo funciona la gestión de cursos académicos en la Universidad de las Ciencias Informáticas y los principales problemas asociados a este.

La entrevista: Permite analizar y valorar la situación existente sobre cómo se gestionan los cursos académicos en la UCI con los expertos del CENID. De esta manera se obtienen los requisitos funcionales y no funcionales necesarios para la elaboración del nuevo sistema a desarrollar.

Con el siguiente trabajo de diploma se esperan alcanzar los siguientes **posibles resultados**:

1. Documentación del sistema.
2. Módulo de Idiomas para el Sistema Automatizado de Gestión Académica (Akademos) de la Universidad de las Ciencias Informáticas.

El documento está estructurado por los siguientes capítulos:

Capítulo 1. Fundamentación teórica: en este capítulo se presentan los elementos teóricos que sirven de base a la investigación. Además, se describen los conceptos fundamentales asociados al dominio del problema y se realizó el análisis de sistemas nacionales e internacionales, que permiten apoyar el desarrollo de la investigación para lograr adaptar el Sistema de Gestión de Idiomas a las necesidades de los clientes. También se detalla cómo funciona el proceso de gestión de cursos académicos en la UCI y se exponen las tecnologías, metodologías y herramientas a utilizar en la implementación de la propuesta de solución.

Capítulo 2. Descripción de la solución informática: este capítulo se dedica a detallar un poco más el funcionamiento del proceso de aplicación de los exámenes estandarizados de idiomas en la Universidad de las Ciencias Informáticas (UCI). Además, se efectúa una descripción más detallada del funcionamiento del negocio explicando las reglas del negocio. También se definen las técnicas para obtener los requisitos funcionales y no funcionales. Más adelante se describen la arquitectura y los patrones de diseño empleados. Por último, se elaboran el Diagrama de Clases del Diseño y el Modelo de Base de Datos.

Capítulo 3. Implementación y validación de la solución propuesta: Se elabora el Diagrama de Componentes y Diagrama de Despliegue. Además, se especifican los estándares de codificación empleados. Asimismo, se realizan y describen las pruebas de software definidas para garantizar su correcto funcionamiento.

CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL OBJETO DE ESTUDIO

El presente capítulo comprende el estudio de varios conceptos y características generales asociados a la gestión de cursos académicos. Se caracteriza el proceso de desarrollo de software, lenguajes de programación y tecnologías que se utilizan para el desarrollo de software en la web. Las tecnologías usadas en la actualidad, también serán objeto de comparación especificando las ventajas y desventajas, así como sus características; concluyendo con una breve explicación del porqué de la selección realizada para el desarrollo definitivo de la aplicación.

◦ I.1 Conceptos asociados al objeto de estudio.

La **gestión** es la acción y el efecto de gestionar y administrar. De una forma más específica, una gestión es una diligencia, entendida como un trámite necesario para conseguir algo o resolver un asunto, habitualmente de carácter administrativo o que conlleva documentación. Es también un conjunto de acciones u operaciones relacionadas con la administración y dirección de una organización. Este concepto se emplea para hablar de proyectos o en general de cualquier tipo de actividad que requiera procesos de planificación, desarrollo, implementación y control [CITATION Sig22 \l 3082].

El término **gestión** también es utilizado para referirse a todo un conjunto de acciones o diligencias que permiten la realización de cualquier actividad o el cumplimiento de un deseo. Dicho de otra manera, al hablar de una gestión se hace referencia a todos aquellos trámites que se deben realizar con la finalidad de resolver una situación o de materializar un proyecto. En el entorno empresarial o comercial, este concepto se asocia con la gestión administrativa de un negocio. Es un grupo de acciones que deben ser ejecutadas para lograr un fin determinado, por ejemplo, un trámite, proyecto o la resolución de un conflicto. Existen varios sinónimos de gestión, como dirección, administración, mando o coordinación[CITATION Aur22 \l 3082].

Este término también es aplicado para el conjunto de acciones que se ejecutan en una organización para evitar errores y es denominado gestión de calidad. Por otra parte, la gestión de riesgos es una perspectiva para el manejo de una amenaza ya existente.

La finalidad de la **Gestión de la información** es ofrecer mecanismos que permitieran a la organización adquirir, producir y transmitir, al menor coste posible, datos e informaciones con una calidad, exactitud y actualidad suficientes para servir a los objetivos de la organización [CITATION Aur22 \l 3082]. En términos perfectamente entendibles, sería conseguir la información adecuada, para la persona que lo necesita, en el momento que lo necesita, al mejor precio posible para tomar la mejor de las decisiones.

Por otra parte, un **Sistema de Gestión de Cursos (CMS)** es una herramienta de software que proporciona una plataforma en línea para alojar cursos, así como para interactuar con estos cursos. Un CMS se construye para la formación profesional con el fin de proporcionar una estructura para gestionar fácilmente el contenido de la formación. Algunos de los mejores sistemas ayudan a obtener mejores resultados de aprendizaje a través de las características incorporadas. Y los mejores también ayudan a que la experiencia sea lo más fácil posible con una plataforma intuitiva, permitiendo que cualquiera pueda realizar fácilmente la formación más efectiva [CITATION Mat22 \l 3082].

Además, un sistema de gestión de cursos, del inglés **Course Management System (CMS)** es un conjunto de herramientas software que permiten a los educadores cargar toda la información sobre sus cursos. Es una forma de programar y rastrear fácilmente todo el contenido que ofrece el curso desde un solo lugar unificado. El profesor puede controlar cada detalle de los cursos de formación, lo que da la oportunidad a los educadores de generar material de curso online y publicarlo en una página web sin entender ningún lenguaje de programación [CITATION Mat22 \l 3082].

Los CMS se pueden utilizar tanto en el mundo educativo como en el empresarial. Nos vamos a concentrar en Sistemas de gestión de cursos implementados en educación. La implementación de herramientas como CMS ofrece varias ventajas a los educadores:

- Asignar instructores y recursos: los profesores pueden programar cosas como instructores, aulas y recursos para los cursos, para asegurarse de poder ejecutar la clase de una manera eficaz y eficiente.
- Toda la información del curso está almacenada en el mismo lugar: en lugar de trabajar desde muchos documentos o plataformas diferentes para mantener bajo control todos los detalles del curso, los educadores pueden cargar toda la información sobre su curso en un sistema específico. Los profesores pueden controlar fácilmente todos los aspectos y editarlos cuando sea necesario.
- Posibilidad de integrarlo con una página web [CITATION EdT22 \l 3082].

Crear el contenido de aprendizaje es solo una parte de lo que debe hacer un sistema de gestión de curso (CMS). Un CMS debe gestionar a los alumnos de diversas maneras. La gestión de los alumnos incluye:

1. Tener acceso a la información sobre el alumnado del curso.
2. Capacidad para realizar grupos de alumnos.
3. Y muchos más, por ejemplo: aplicar diversas escalas en las calificaciones de los alumnos, seguimiento y registros de los accesos de los usuarios y poder subir archivos externos para el uso dentro del curso [CITATION EdT22 \l 3082].

Además, se deben tener en cuenta una serie de elementos para la confección de un CMS. Entre los elementos que no deben faltar en este están los siguientes:

1) **Participantes:** se puede ver la actividad de todos los participantes del curso. Los alumnos generan un perfil personal que puede incluir una imagen, lo que ayuda a establecer lazos sociales en la comunidad de aprendizaje.
2) **Grupos:** asignar a los alumnos a un grupo es una práctica común en la educación. CMS permite al profesor del curso crear fácilmente categorías del grupo, y determinar cómo los miembros se relacionarán entre los demás grupos y en las diferentes actividades.

3) **Administración:** el panel de control de la administración permite con un solo clic todas las funciones importantes de la gestión del curso. Los profesores y los estudiantes pueden ser inscritos o eliminados manualmente. La configuración de copia de seguridad y la restauración de un curso se obtiene en una sola pantalla.

4) **Calendario:** mantener un calendario de acontecimientos es importante para el alumno y el profesor del curso. Los acontecimientos se pueden crear en diversas categorías, incluyendo:

   a) Los acontecimientos globales aparecen en todos los cursos

   b) Los acontecimientos del curso los fija el profesor.

   c) Los acontecimientos de un grupo solo los ve el grupo.

   d) Los acontecimientos próximos aparecen en la página principal del curso, avisando al alumno. Las alarmas son de colores por categoría.

5) **Escalas:** los profesores pueden definir escalas que se utilizarán para calificar foros, tareas y diarios. Las escalas estándares pueden asignar un valor de 1-100% en cada actividad (o ninguna calificación), o indicar si el estudiante demuestra una de las tres características en la actividad.

6) **Calificaciones:** la opción de las calificaciones ofrece una visualización de todas las calificaciones de los foros, tareas, diarios, cuestionarios, lecciones y del taller. La escala de calificación aplicada en una actividad de aprendizaje se muestra, junto con un total acumulado, en una sola página. Esto reduce el tiempo de calificación.

7) **Registros:** los registros establecen claramente donde está un estudiante respecto al curso. Localiza fácilmente la fecha y el acceso específico de una actividad del curso, un alumno, de un módulo.

8) **Archivos:** todos los recursos del curso están localizados dentro del área de los archivos del CMS. Los recursos están disponibles al usar el editor de HTML, permitiendo incluirlos fácilmente en el contenido en una actividad.

9) **Ayuda:** un gran número de ayudas están accesibles en su contexto. Los cursos incluyen un foro exclusivo de profesores, donde pueden colaborar en tareas y compartir ideas.

10) **Conexión:** los alumnos encuentran fácil navegar en las páginas del curso con su navegador; los enlaces están siempre presentes. La conexión se produce en una pantalla familiar.

11) **Claves de inscripción:** los profesores pueden exigir una palabra clave para permitir la inscripción en un curso. Estas permiten un proceso individual de conexión. Los cursos que requieren una clave para la inscripción se indican en la descripción de las categorías de cursos.

12) **Experiencia del alumno:** los alumnos pueden conectar en cualquier momento, desde donde quieran para usar el curso, y pueden especificar la zona horaria y el idioma que desean utilizar.

13) **Notificación vía email:** si los alumnos se “suscriben” a los foros, los nuevos mensajes serán enviados vía correo electrónico. Además, los profesores pueden elegir la notificación en las charlas privadas [CITATION Lon16 \l 3082].

Académico es un término que tiene su origen en latín “academicus”, que a su vez se refiere al término academia que proviene del griego “akademeia”. Lo académico también puede tener que ver con el término griego “akademos”, un héroe griego. Según la leyenda, “El Bosque de Akademos” es el terreno donde Platón estableció su primera escuela filosófica. Dada su etimología, se dice entonces qué académico es el término que se refiere a la academia o algo que le pertenece, cuya preocupación es aplicar los fundamentos del arte oficial [CITATION Ser14 \l 3082].

El académico puede cubrir una gama de otros términos:

1. **Cursos académicos:** son cursos con diplomas reconocidos, es decir, legalmente registrados. Los profesionales que poseen un título académico, que ya han cumplido con todos los requisitos curriculares del mismo, ya pueden ejercer legalmente su profesión,

ya que a partir de entonces se espera que este profesional ya posea todas las competencias necesarias para ello.

2. **Maestrías académicas:** después de la educación superior, los estudiantes pueden continuar sus estudios eligiendo un título de posgrado, ya sea en sentido amplio (cursos de especialización, perfeccionamiento o extensión) o en sentido estricto (maestría profesional, maestría académica o doctorado). La maestría profesional está más enfocada al mercado laboral con un currículo menos teórico. La finalización del curso tiene lugar con la realización de una monografía.

- El máster académico es una especie de título de postgrado centrado en la enseñanza y la investigación. El individuo que se convierte en maestro generalmente se incorpora a la enseñanza en un cierto campo del conocimiento. Para obtener un nuevo título, el estudiante de maestría debe presentar una tesis. Sólo después de esta presentación podrá ejercer su nueva función.

3. **Trabajo académico:** Podemos decir que el trabajo académico es un texto final en el que se presenta una investigación sobre un área particular del conocimiento. Generalmente, los trabajos académicos se escriben en forma de tesis, disertación, artículo científico, entre otros, ya sea en sentido amplio o estricto[CITATION EST12 \l 3082].

Módulo es una unidad o artículo autónomo, como un conjunto de componentes electrónicos y cableado asociado o un segmento de software de computadora, que en sí mismo realiza una tarea definida y puede vincularse con otras unidades similares para formar un sistema más grande. Por otro lado, en educación es un curso de estudio corto, especialmente de una materia vocacional o técnica, que junto con otros cursos completados puede contar para una calificación particular. También es una de las partes separadas de un curso impartido en un colegio o universidad [CITATION Gar19 \l 3082].

Las definiciones antes evidenciadas están estrechamente vinculadas entre ellas y a su vez a los principios que debe cumplir un sistema de gestión de cursos para proteger la confidencialidad, integridad y disponibilidad de la información.

I.2 Soluciones informáticas para la gestión de cursos de idiomas

Con el objetivo de aumentar el nivel de eficiencia de los procesos de gestión de los cursos de idiomas en el Centro de Idiomas, se realizó el análisis de sistemas nacionales e internacionales, que permiten apoyar el desarrollo de la investigación para lograr adaptar el Sistema de Gestión de Idiomas a las necesidades de los clientes.

OfiELE (El software ideal para Escuelas de idiomas y Academias):

Es el nuevo software para academias y escuelas de enseñanza desarrollado por Ofimática para la gestión de Academias de Idiomas y Escuelas en general. La aplicación es fruto de la colaboración con varias escuelas y academias y de los más de 36 años de experiencia con la garantía de más de 18.000 instalaciones hechas en toda España y Latinoamérica. Desarrollado con filosofía Cloud Computing (en la “nube”). El software puede estar instalado en los ordenadores locales de su Escuela o en un Servidor de Internet (propio o, más recomendable, Datacenter externo), así se podrá trabajar desde cualquier lugar y ordenador con una simple conexión a Internet (teletrabajo) [CITATION Ofi22 \l 3082].

También presenta distribución por módulos y gestión multipuesto y multiusuario por medio de contraseñas y concesión de permisos a los usuarios [CITATION Ofi22 \l 3082]. No todos tendrán permiso para acceder a toda la información. Ofrece una solución completa e integral. Sin módulos ni versiones inferiores ni superiores. Presenta una gestión especial centralizada en el Alumno, con amplia toma de información, contando con una agenda personal para cada usuario. Por otro lado, posee gestión de Biblioteca, Videoteca y Libros para la clase. Además de la posibilidad de crear tantos documentos como se quiera e interrelacionarlos con los datos de la ficha del alumno y/o cursos [CITATION ofi22 \l 3082].

7Speaking:

Basado en el modelo 70-20-10, la fórmula 7Speaking combina innovaciones pedagógicas y tecnológicas: social learning, e-CLIL, gamificación y adaptive learning. Con excelentes

resultados: cuando los organismos convencionales tardan 200 horas para que un empleado pase del nivel B1 al nivel B2, 7Speaking tarda solo 50 horas [CITATION 7Sp22 \l 3082]. Para la adquisición de competencias, la investigación en ciencias cognitivas ha demostrado la eficacia del modelo 70-20-10:

1. Aplicación práctica de los conocimientos (70%).
2. Formación informal con compañeros y expertos (20%).
3. Formación formal (10%).

Para aprender un idioma, este enfoque se traduce por una inmersión en la lengua en cuyo seno la motivación del alumno debe ser estimulada a lo largo de su aprendizaje. Las actividades que propone fueron concebidas persiguiendo estas dos prioridades: inmersión 70-20-10 + motivación del alumno. Los mejores resultados obtenidos por el dispositivo 7Speaking encuentran su origen en el ecosistema pedagógico único en el que se desenvuelve el alumno. Entre las actividades que propone están las siguientes:

1) **Cursos basados en la actualidad:** formando e informando al mismo tiempo.
2) **Professional skills:** formación de manera simultánea en inglés y en liderazgo, en coaching, en gestión y en otras formaciones profesionales.
3) **Comunidades de expertos por sector:** a través del Social Learning, bajo la premisa “porque siempre se aprende solo, pero nunca sin los demás”.
4) **Conference Call:** participando en reuniones internacionales con un pequeño número de participantes y un profesor sobre un tema profesional definido.
5) **Chat Texto/Vídeo:** chateando en vídeo o por escrito con sus compañeros o con un profesor nativo sobre el debate del día o en modo de conversación libre.
6) **Talleres:** descubriendo, profundizando y practicando 5 talleres para asimilar lo fundamental.
7) **Microlearning - Clases por correo electrónico:** recibiendo diariamente en el correo electrónico las lecciones basadas en las noticias del día.

8) **Seguimiento y Tutoría:** al inscribirse, cada estudiante tiene una sesión de acompañamiento por teléfono. Desde entonces y hasta el final de su formación, los asesores pedagógicos le acompañan periódicamente por teléfono [CITATION 7Sp22 \l 3082].

Por último, 7Speaking aporta una serie de soluciones. Entre estas están las siguientes:

a) Generalizar el acceso a la formación de idiomas para todos los empleados.

b) Mapeo del nivel y necesidades lingüísticas de toda la compañía.

c) Permitir que los empleados elegidos sean operacionales 4 veces más rápido en su trabajo.

d) Generar la aceptación del alumno, lograr tasas de participación óptimas [CITATION 7Sp22 \l 3082].

SIGENU:

El Sistema de Gestión de la Nueva Universidad (SIGENU) es un sistema que se ha desarrollado con el fin de ser una herramienta que permita la gestión de toda la información académica vinculada con la educación superior en Cuba. En correspondencia con su carácter nacional y la gran diversidad de sistemas de enseñanza superior con que cuenta la universidad cubana, este sistema ha sido concebido de manera tal que sea capaz de brindar gran seguridad e integridad de la información. A la vez, ser tan flexible que permita ser adaptado a todos los centros de educación superior del país con sus diversas particularidades y distintas maneras de realizar determinados procedimientos.

El sistema SIGENU está compuesto por cuatro elementos fundamentales:

1) **Base de Datos:** Este es el lugar en el cual se almacena toda la información registrada.

2) **Servidor de Aplicaciones:** Este elemento es el encargado de actuar como intermediario entre la Base de Datos y las aplicaciones clientes, o sea, es quien hace posible que la información registrada en la base sea visualizada y actualizada a través de las

aplicaciones que son manipuladas por el usuario. Este elemento es quien permite dar un servicio a través de la red de manera que la información pueda ser accedida remotamente por las aplicaciones ubicadas en varios lugares.

3) **Aplicación Cliente Secretaría:** Es la aplicación que constituye el elemento que esencialmente permite la inserción y actualización de toda la información que se registre en el sistema. Además, permite obtener un conjunto importante de reportes muy usados cotidianamente en el mundo de la educación superior. Consta de los siguientes módulos: Codificadores, Matrícula, Control de estudiantes, Plan de Estudio, Evaluaciones y Reportes. La aplicación cliente se encuentra disponible para escritorio (desktop) y web.

4) **Aplicación Cliente de Administración:** Es la aplicación que permite la inserción y actualización de los usuarios y todas las funcionalidades que deban ser ejecutadas por los administradores para monitorear el correcto funcionamiento del sistema y su seguridad. Naturalmente, a esta aplicación únicamente tendrán acceso los encargados de la administración del SIGENU [CITATION Uni15 \l 3082].

Akademos:

La eficiencia en el proceso de gestión académica en la UCI (Universidad de las Ciencias Informáticas) se logra a través del Sistema Automatizado para la Gestión Académica (Akademos), un sistema automatizado para la gestión académica desarrollado por un equipo de trabajo de la Dirección de Informatización. El mismo es un sistema Web distribuido, desarrollado en la plataforma .NET, que utiliza SQL Server 2000 para el almacenamiento de los datos, IIS (Internet Information Services) como servidor Web y Servicios WEB XML para el intercambio con otras aplicaciones.

Sus funcionalidades se agrupan en siete módulos, de los cuales el Plan de Estudio es la entidad fundamental, pues rige todos los subprocesos [CITATION Cos07 \l 3082].

Figura 1: Distribución por módulos (Fuente: Elaboración Propia).

Módulo de plan de estudio: Permite la gestión de diferentes especialidades o carreras al definir los planes de estudio.

Módulo de matrícula: Se encarga del control de los datos de los estudiantes y la gestión de los movimientos a que son sometidos.

Módulo de expediente: Actúa como un repositorio digital de los documentos y la información histórica de los estudiantes, disponible en todo momento.

Módulo de registro: Permite el control del desarrollo de un período académico con el registro de las evaluaciones y la asistencia.

Módulo de profesor: Mantiene un control de la plantilla de profesores.

Módulo de reportes: Obtiene reportes ordenados de la información almacenada en el sistema.

Módulo de estudiantes: Ofrece el historial académico de los años cursados por los estudiantes.

Con respecto a la seguridad, Akademos implementa varios niveles de acceso para restringir las acciones que puede realizar un usuario determinado. Lleva un control sobre las acciones

que realizan los usuarios en el sistema y cuenta con una aplicación de monitoreo de las incidencias en tiempo real.

Resultados de Akademos:

Los avances obtenidos con su despliegue en la UCI son palpables, por ejemplo, en todo momento, el sistema está disponible para la consulta y actualización de la información, ya sea por directivos, profesores o estudiantes. Esto agiliza la gestión, y elimina la necesidad de intermediarios entre la información y los que la generan o necesitan.

Aporte Social de Akademos:

Akademos posibilita un mejor control e incidencia en el aprendizaje de los estudiantes, pues con la información que los profesores introducen en tiempo real, es posible determinar el estado de los estudiantes tanto de forma horizontal como vertical. Además, sus reportes pueden servir para efectuar investigaciones sociales importantes, dada la variedad en términos de proveniencia geográfica, social y académica de la masa estudiantil en la UCI.

Valoración Económica de Akademos:

Aun cuando existan restricciones legales en cuanto a la documentación dura que se maneja en la gestión académica, la posibilidad de que toda la información que se gestiona en Akademos pueda consultarse en línea hace que se incurra en un ahorro de materiales de oficina. Otro aspecto consiste en el ahorro por compra de un sistema de este tipo en el extranjero. Uno de los sistemas con objetivos similares estudiados durante el desarrollo de Akademos fue “Ágora”, que se comercializa en varias variantes [CITATION Cos07 \l 3082].

Las herramientas descritas ofrecen funcionalidades que se acoplan a las tareas del negocio; sin embargo, carecen de integración de los datos con aplicaciones externas y requieren de pago de licencias. Además, no apoyan la política y los lineamientos de informatización del país sobre el uso del software libre para garantizar la soberanía tecnológica.

Debido a lo antes expuesto, se escoge como referente para el desarrollo de este trabajo el sistema Akademos, ya que este sistema es el que está implementado en nuestra

universidad. Además, este presenta una serie de elementos que se ajustan a la solución a desarrollar, las cuales son:

- ✔ El sistema informático Akademos ha posibilitado la participación de directivos, docentes y estudiantes para agilizar los mecanismos de la gestión académica y disminuir la ocurrencia de errores.
- ✔ La flexibilidad en la configuración del sistema permite implantar Akademos en cualquier centro de estudios, a la par que facilita su adaptación a los cambios que se necesite introducir.
- ✔ La documentación generada por la gestión académica es almacenada en el sistema, viabilizando la construcción de la historia de los centros de estudios [CITATION Cos07 \l 3082].

I.3 El proceso de gestión de cursos académicos en la UCI

El proceso de gestión de cursos académicos en la Universidad de las Ciencias Informáticas comienza con la entrada de los aspirantes, que en este caso serían los posibles estudiantes que obtendría el centro. Para ello, la UCI cuenta con programas académicos de alta calidad en diferentes modalidades, los que están disponibles para los aspirantes nacionales y extranjeros que quieran estudiar en este centro de altos estudios. Para esto la Universidad cuenta con requisitos necesarios, los cuales se detallarán a continuación.

Para Estudiantes Cubanos:

Cuando los estudiantes de nuevo ingreso hagan la matrícula oficial deberán presentar la siguiente documentación:

1. Carné de Identidad (Actualizado y en buen estado).
2. Original y fotocopia del título de graduado del nivel medio superior o la certificación de estudios terminados.
3. Los varones que ingresen al curso diurno deben acreditar su situación con respecto al cumplimiento del Servicio Militar Activo (SMA) presentando algunos de los siguientes documentos, según el caso:

a) Si fue declarado apto FAR, el original de la boleta de licenciado del Servicio Militar que expide la unidad militar donde cumplió el servicio, correctamente firmada y acuñada; o

b) Si fue declarado apto con recomendaciones médicas o declarado no apto FAR: el original del modelo SIES-4 de la evaluación del cumplimiento de la tarea socialmente útil (acuñada y firmada por la Comisión de Ingreso Provincial); el Anexo 1 con sello de timbre (que expide el Comité Militar Municipal), así como una fotocopia de la Resolución emitida por el Comité Provincial Militar (se presentará también el original para su cotejo) [CITATION Uni20 \l 3082].

Para Estudiantes Extranjeros:

De acuerdo con las disposiciones legales aprobadas por el Ministerio de Educación Superior (MES) de Cuba:

1. Los ciudadanos extranjeros residentes permanentes en el territorio nacional podrán ingresar a nuestra Universidad según los mismos requisitos vigentes para los cubanos.
2. Los ciudadanos extranjeros que deseen matricular una carrera universitaria como estudiantes autofinanciados deben solicitar a la Dirección de Ingreso y Ubicación Laboral del MES el autorizo de matrícula.
3. En los casos que sean financiados por un gobierno, la embajada de su país presenta la solicitud al Departamento de Exportaciones o Servicios Académicos del MES.
4. Los que sean aprobados recibirán la respuesta por esa misma vía y para matricular deberán presentar los siguientes documentos debidamente legalizados por las oficinas consulares de Cuba en el exterior:

   a) Título de graduado de preuniversitario, bachillerato o nivel equivalente y el certificado de notas.

   b) Certificado de salud debidamente legalizado que incluya la declaración de que no portan enfermedades transmisibles, así como impedimentos físicos o mentales invalidantes para el ejercicio de la profesión a que aspira.

c) Documento que certifique que no posee antecedentes penales.

d) Certificación de Nacimiento.

5. Los ciudadanos extranjeros no hispanohablantes deben demostrar su dominio del idioma español. De no cumplirse este requisito, el aspirante debe matricular la etapa preparatoria (en la propia universidad) antes de iniciar el primer año de la carrera [CITATION Uni20 \l 3082].

Luego de conocer el proceso de adquisición de aspirantes, se pasará a detallar los servicios académicos que ofrece la entidad. Estos se dividen en tres modalidades de formación: Pregrado, Posgrado y Educación a Distancia. Las actividades de Pregrado se especializan en la carrera de Ingeniería en Ciencias Informáticas. Además, se estudian las carreras de Ingeniería en Bioinformática e Ingeniería en Ciberseguridad. El programa de ciclo corto de Administración de Redes y Seguridad Informática complementa la oferta del centro. También se imparten cursos, pasantías y otras modalidades de menor duración vinculadas a las Ciencias de la Computación.

En el Posgrado se ofrecen oportunidades de superación a través de cursos, pasantías, entrenamientos, diplomados, maestrías y doctorados. También se desarrollan eventos científicos que sirven como punto de encuentro para el intercambio de conocimientos y experiencias profesionales. Una última, pero excelente opción de posgrado, son sus Escuelas Internacionales de Verano e Invierno [CITATION Uni22 \l 3082]. Para finalizar, la modalidad de Educación a Distancia se desarrolla a través del Centro Nacional de Educación a Distancia (CENED), perteneciente a esta casa de altos estudios.

El Centro Nacional de Educación a Distancia (CENED), con sede en la Universidad de las Ciencias Informáticas, tiene la misión de contribuir al desarrollo y la excelencia de la educación a distancia en Cuba, incrementando su competitividad mediante la difusión, la mejora continua y la aplicación creadora de las tecnologías de la información y la comunicación. Los servicios académicos que se ofertan en forma presencial y no presencial a todo el país y hacia el exterior son:

1. Asesorías para el diseño de programas en la modalidad a distancia virtual y para el diseño, producción y evaluación de cursos y recursos educativos de apoyo a ese tipo de formación.

2. Asesorías para el diseño y ejecución de proyectos de investigación científica, trabajos de desarrollo y procesos de innovación relacionados con la aplicación de las tecnologías al proceso de formación de pregrado y posgrado.

3. Cursos y entrenamientos de posgrado y capacitación sobre diversas aristas de desarrollo de la modalidad a distancia virtual, entre ellas:

   a) Diseño, gestión y evaluación de cursos virtuales.

   b) Diseño, producción y evaluación de recursos educativos digitales.

   c) Organización tutorial y evaluación del aprendizaje a distancia.

   d) Tecnologías y métodos de formación en red.

   e) Especificaciones y estándares sobre e-learning y TIC.

   f) Minería de datos educacionales y analíticas de aprendizaje.

   g) Administración de plataformas educativas.

4. Maestría en Educación Virtual [CITATION Uni22 \l 3082].

I.4 Tecnologías informáticas para la informatización de la gestión de cursos de idiomas

A continuación se exponen las tecnologías, metodologías y herramientas a utilizar en la implementación de la propuesta de solución. Para ello fue necesario realizar un estudio meticuloso de las actuales tendencias y tecnologías disponibles para el desarrollo de software. Los resultados de dicho estudio, que comprenden, entre otras cosas, los pros y los contras de cada elemento analizado, se reflejan brevemente a continuación.

HTML:

Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language: es el componente más básico de la Web. Define el significado y la estructura del contenido web.

Además de HTML, generalmente se usan otras tecnologías para describir la apariencia/presentación de una página web (CSS) o la funcionalidad/comportamiento (JavaScript). "Hipertexto" hace referencia a los enlaces que conectan páginas web entre sí, ya sea dentro de un único sitio web o entre sitios web. Los enlaces son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a las páginas creadas por otras personas, te conviertes en un participante activo en la «World Wide Web» (Red Informática Mundial) [CITATION Dur22 \l 3082].

Un elemento HTML se distingue de otro texto en un documento mediante “etiquetas”, que consisten en el nombre del elemento rodeado por “<” y “>”. El nombre de un elemento dentro de una etiqueta no distingue entre mayúsculas y minúsculas. Es decir, se puede escribir en mayúsculas, minúsculas o una mezcla. Por ejemplo, la etiqueta <title> se puede escribir como <Title>, <TITLE> o de cualquier otra forma [28]. El Lenguaje de Marcado de Hipertexto (HTML) es el código que se utiliza para estructurar y desplegar una página web y sus contenidos. Por ejemplo, sus contenidos podrían ser párrafos, una lista con viñetas, o imágenes y tablas de datos [CITATION Dur22 \l 3082].

HTML no es un lenguaje de programación; es un lenguaje de marcado que define la estructura de tu contenido. HTML consiste en una serie de elementos que usarás para encerrar diferentes partes del contenido para que se vean o comporten de una determinada manera. Las etiquetas de encierre pueden hacer de una palabra o una imagen un hipervínculo a otro sitio, se pueden cambiar palabras en cursiva, agrandar o achicar la letra, entre otras [CITATION Dur22 \l 3082].

CSS:

Hojas de Estilo en Cascada (del inglés Cascading Style Sheets): es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (en-US) (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). Es el código que utilizas para dar estilo a tu página web. CSS describe cómo debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios. CSS es uno de los lenguajes base de la Open Web y posee una especificación estandarizada por parte del W3C [CITATION CAB21 \l 3082].

World Wide Web Consortium (W3C - El Consorcio de la World Wide Web) es un consorcio internacional en el que las organizaciones miembros, el personal a tiempo completo y el público en general trabajan juntos para desarrollar normas y directrices web diseñadas para garantizar el crecimiento a largo plazo de la web. El objetivo del W3C es que la Web conecte a la humanidad de manera que el acceso al conocimiento sea más eficiente y equitativo [CITATION Ort20 \l 3082].

Anteriormente, el desarrollo de varias partes de las especificaciones de CSS era realizado de manera sincrónica, lo que permitía el versionado de las recomendaciones. Probablemente, habrás escuchado acerca de CSS1, CSS2.1, CSS3. Sin embargo, CSS4 nunca se ha lanzado como una versión oficial. Desde CSS3, el alcance de las especificaciones se incrementó de forma significativa y el progreso de los diferentes módulos de CSS comenzó a mostrar varias diferencias, lo que hizo más efectivo desarrollar y publicar recomendaciones separadas por módulos. En vez de versionar las especificaciones de CSS, el W3C actualmente realiza una captura de las últimas especificaciones estables de CSS [CITATION CAB21 \l 3082].

JavaScript (JS):

Es un lenguaje ligero, interpretado y orientado a objetos con funciones de primera clase, y mejor conocido como el lenguaje de programación para las páginas Web, pero también se utiliza en muchos entornos que no son de navegador. Es un lenguaje de scripts que es dinámico, multiparadigma, basado en prototipos y admite estilos de programación orientados a objetos, imperativos y funcionales. JavaScript se ejecuta en el lado del cliente de la web, y se puede emplear para estilizar/programar cómo se comportan las páginas web cuando ocurre un evento. JavaScript es un potente lenguaje de scripts y fácil de aprender, ampliamente empleado para controlar el comportamiento de las páginas web [CITATION Rom19 \l 3082].

Contrariamente a la creencia popular, JavaScript no es "Java interpretado". En pocas palabras, JavaScript es un lenguaje de scripts dinámico que admite la construcción de objetos basada en prototipos. Intencionalmente, la sintaxis básica es similar a Java y C + + para reducir la cantidad de conceptos nuevos necesarios para aprender el lenguaje.

Construcciones del lenguaje, como las declaraciones if, los bucles for y while, y switch y los bloques try...catch funcionan igual que en esos lenguajes (o casi) [CITATION Mal15 \l 3082].

PHP:

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular, especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. En lugar de usar muchos comandos para mostrar HTML (como en C o en Perl), las páginas de PHP contienen HTML con código incrustado que hace "algo" (en este caso, mostrar "¡Hola, soy un script de PHP!). El código de PHP está encerrado entre las etiquetas especiales de comienzo y final <?php y ?> que permiten entrar y salir del "modo PHP". Lo que distingue a PHP de algo del lado del cliente como JavaScript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente [CITATION Pic16 \l 3082].

Lo mejor de utilizar PHP es su extrema simplicidad para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales[CITATION Pic16 \l 3082]. PHP puede emplearse en todos los sistemas operativos principales, incluyendo Linux, muchas variantes de Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, macOS, RISC OS y probablemente otros más [CITATION Fos18 \l 3082].

PHP admite la mayoría de servidores web de hoy en día, incluyendo Apache, IIS, y muchos otros. Esto incluye cualquier servidor web que pueda usar el binario de PHP FastCGI, como lighttpd y nginx. PHP funciona tanto como módulo como procesador de CGI. De modo que con PHP, se tiene la libertad de elegir el sistema operativo y el servidor web. Además, se tiene la posibilidad de utilizar programación por procedimientos o programación orientada a objetos (POO), o una mezcla de ambas. Una de las características más potentes y destacables de PHP es su soporte para un amplio abanico de bases de datos [CITATION Muñ13 \l 3082].

SQL:

Structured Query Language (Lenguaje de Consulta Estructurado) es un lenguaje estándar para almacenar, manipular y recuperar datos en bases de datos. SQL es un lenguaje de computación para trabajar con conjuntos de datos y las relaciones entre ellos. Los programas

de bases de datos relacionales, como Microsoft Office Access, usan SQL para trabajar con datos. A diferencia de muchos lenguajes de computación, SQL no es difícil de leer y entender, incluso para un usuario inexperto [CITATION Kro02 \l 3082].

Entre las funciones que presenta SQL están las siguientes:

a) Puede ejecutar consultas contra una base de datos.

b) Puede recuperar datos de una base de datos.

c) Puede insertar, actualizar y eliminar registros en una base de datos.

d) Puede crear nuevas bases de datos.

e) Puede generar nuevas tablas en una base de datos.

f) Puede producir procedimientos almacenados en una base de datos.

g) Puede originar vistas en una base de datos.

h) Puede establecer permisos en tablas, procedimientos y vistas [CITATION Kro02 \l 3082].

UML:

El lenguaje de modelado unificado (UML) es un estándar para la representación visual de objetos, estados y procesos dentro de un sistema. Por un lado, el lenguaje de modelado puede servir de modelo para un proyecto y garantizar así una arquitectura de información estructurada; por el otro, ayuda a los desarrolladores a presentar la descripción del sistema de una manera que sea comprensible para quienes están fuera del campo. UML se utiliza principalmente en el desarrollo de software orientado a objetos. Los diagramas UML se utilizan para representar los siguientes componentes del sistema:

1. Objetos individuales (elementos básicos).

2. Clases (combina elementos con las mismas propiedades).

3. Relaciones entre objetos (jerarquía y comportamiento/comunicación entre objetos).

4. Actividad (combinación compleja de acciones/módulos de comportamiento).
5. Interacciones entre objetos e interfaces [CITATION Bur18 \l 3082].

El Lenguaje Unificado de Modelado (UML) desempeña un rol importante no solo en el desarrollo de software, sino también en los sistemas que no tienen software en muchas industrias, ya que es una forma de mostrar visualmente el comportamiento y la estructura de un sistema o proceso. El UML ayuda a mostrar errores potenciales en las estructuras de aplicaciones, el comportamiento del sistema y otros procesos empresariales.

Algunas ventajas del uso de UML son las siguientes:

1) Simplifica las complejidades
2) Mantiene abiertas las líneas de comunicación
3) Automatiza la producción de software y los procesos
4) Ayuda a resolver los problemas arquitectónicos constantes
5) Aumenta la calidad del trabajo
6) Reduce los costos y el tiempo de comercialización [CITATION Bur18 \l 3082].

Visual Paradigm:

Es una herramienta de diseño UML y herramienta CASE del inglés (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora) diseñada para ayudar al desarrollo de software. Soporta los principales estándares de la industria, tales como el Lenguaje de Modelado Unificado (UML), y la notación de Modelado de Procesos de Negocio (BPMN). Ofrece un completo conjunto de herramientas de equipos de desarrollo de software necesario para la captura de requisitos, planificación de software, planificación de controles, modelado de clases y modelado de datos. Además, permite modelar todos los tipos de diagramas de clases, código inverso y generar código desde los diagramas [CITATION Pac16 \l 3082].

CodeIgniter:

Es un marco de desarrollo de aplicaciones, un conjunto de herramientas, para personas que crean sitios web con PHP. Su objetivo es permitirle desarrollar proyectos mucho más rápido de lo que podría si estuviera escribiendo código desde cero, al proporcionar un amplio conjunto de bibliotecas para tareas comúnmente necesarias, así como una interfaz simple y una estructura lógica para acceder a estas bibliotecas. CodeIgniter le permite concentrarse creativamente en su proyecto al minimizar la cantidad de código necesario para una tarea determinada.

En la medida de lo posible, CodeIgniter se ha mantenido lo más flexible posible, lo que le permite trabajar de la manera que desee, sin verse obligado a trabajar de una manera determinada. El marco puede tener partes centrales fácilmente ampliables o reemplazadas por completo para que el sistema funcione de la manera que lo necesita. En resumen, CodeIgniter es el marco maleable que intenta proporcionar las herramientas que necesita mientras se mantiene fuera del camino [CITATION Bem19 \l 3082].

Algunas características de este marco de trabajo son las siguientes:

1. Es un marco que ocupa poco espacio.
2. Proporciona un rendimiento excepcional.
3. Es un marco que requiere una configuración casi nula.
4. Es un marco que no requiere que use la línea de comando.
5. Es un marco que no requiere que se adhiera a reglas de codificación restrictivas.
6. Con el uso de este no es necesario aprender un lenguaje de plantillas (aunque un analizador de plantillas está disponible opcionalmente por este, si lo desea).
7. Evita la complejidad, favoreciendo las soluciones simples.
8. Si necesita una documentación clara y completa, esta es una buena opción para utilizar [CITATION Bem19 \l 3082].

jQuery:

Es una biblioteca de JavaScript rápida, pequeña y rica en funciones. Hace que cosas como el recorrido y la manipulación de documentos HTML, el manejo de eventos, la animación y Ajax sean mucho más simples, con una API fácil de usar que funciona en una multitud de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript [CITATION Gal22 \l 3082].

El propósito de jQuery es hacer mucho más fácil el uso de JavaScript en su sitio web. jQuery toma muchas tareas comunes que requieren muchas líneas de código JavaScript para llevarlas a cabo y las envuelve en métodos a los que puede llamar con una sola línea de código. jQuery también simplifica muchas de las cosas complicadas de JavaScript, como las llamadas AJAX y la manipulación DOM.

La biblioteca jQuery contiene las siguientes características:

a) Manipulación de HTML/DOM.

b) Manipulación de CSS.

c) Métodos de eventos HTML.

d) Efectos y animaciones.

e) AJAX.

f) Utilidades [CITATION Gal22 \l 3082].

Git:

Es un sistema de control de versiones distribuido, gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia. Git es fácil de aprender y ocupa poco espacio con un rendimiento ultrarrápido. Supera a las herramientas de SCM como Subversion, CVS, Perforce y ClearCase con características como sucursales locales económicas, áreas de preparación convenientes y múltiples flujos de trabajo [CITATION Gar15 \l 3082].

Pencil:

Es una herramienta de creación de prototipos de GUI de código abierto que está disponible para TODAS las plataformas. Pencil está diseñado con el propósito de proporcionar una herramienta de creación de prototipos de GUI gratuita y de código abierto que las personas pueden instalar y usar fácilmente para crear maquetas en plataformas de escritorio populares. La última versión estable de Pencil es la 3.1.0, que contiene correcciones de estabilidad y muchas características nuevas [CITATION Gar15 \l 3082]. Este proporciona varias colecciones de formas integradas para dibujar diferentes tipos de interfaz de usuario, desde plataformas de escritorio hasta plataformas móviles.

A partir de 2.0.2, Pencil se envía con las plantillas de interfaz de usuario de iOS y Android preinstaladas. Esto hace que sea aún más fácil comenzar a producir prototipos de aplicaciones con una instalación simple. Las funciones de dibujo populares también se implementan en Pencil para simplificar las operaciones de dibujo [CITATION Par20 \l 3082].

Nginx:

NGINX es un software de código abierto para servicios web, proxy inverso, almacenamiento en caché, equilibrio de carga, transmisión de medios y más. Comenzó como un servidor web diseñado para el máximo rendimiento y estabilidad. Además de sus capacidades de servidor HTTP, NGINX también puede funcionar como un servidor proxy para correo electrónico (IMAP, POP3 y SMTP) y un proxy inverso y equilibrador de carga para servidores HTTP, TCP y UDP.

Igor Sysoev (creador de NGINX), originalmente escribió NGINX para resolver el problema C10K, un término acuñado en 1999 para describir la dificultad que experimentaban los servidores web existentes para manejar grandes cantidades (10K) de conexiones simultáneas (C). Con su arquitectura asincrónica basada en eventos, NGINX revolucionó la forma en que los servidores funcionan en contextos de alto rendimiento y se convirtió en el servidor web más rápido disponible.

Después de abrir el código fuente del proyecto en 2004 y ver su uso crecer exponencialmente, Sysoev cofundó NGINX, Inc. para apoyar el desarrollo continuo de NGINX y comercializar NGINX Plus como un producto comercial con características

adicionales diseñadas para clientes empresariales. NGINX, Inc. se convirtió en parte de F5, Inc. en 2019. Actualmente, NGINX y NGINX Plus pueden manejar cientos de miles de conexiones simultáneas y potenciar más sitios de Internet más concurridos que cualquier otro servidor [CITATION Pal \l 3082].

PhpStorm:

PhpStorm es un entorno de desarrollo integrado muy completo que proporciona un editor para PHP, HTML y JavaScript con análisis de código en tiempo real, prevención de errores y refactorizaciones automatizadas para código Java y PHP. Es perfectamente compatible con otros marcos de trabajo como Symfony, Laravel, Drupal, WordPress, Zend Framework o Magento, entre otras muchas. Entre sus características principales se encuentran la del autocompletado de código, nombres de variables o palabras clave de PHP, además de ofrecer soporte de estilo de codificación, soporte PHPDoc, Code Sniffer o PHAR.

El sistema de detección de código duplicado hace la vida más sencilla y los refactorizadores permiten cambiar nombres, introducir variables, constantes, campos o mover miembros estáticos de la forma más cómoda. Con PhpStorm se puede tomar el control de la base del código gracias a los cientos de inspecciones que se encargan de verificar el código mientras se escribe, analizando el proyecto en su conjunto. Así, se puede escribir un código limpio, muy fácil de mantener, sin errores [CITATION MIr16 \l 3082]. Es compatible con PHP 5.3/8.1, ofrece depuración sin configuración y un editor extendido para HTML, CSS y JavaScript.

PostgreSQL:

Es un potente sistema de base de datos relacional de objetos de código abierto que utiliza y amplía el lenguaje SQL combinado con muchas características que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas. Se ha ganado una sólida reputación por su arquitectura comprobada, confiabilidad, integridad de datos, conjunto sólido de funciones, extensibilidad y la dedicación de la comunidad de código abierto detrás del software para ofrecer soluciones innovadoras y de alto rendimiento. PostgreSQL se ejecuta

en todos los principales sistemas operativos, cumple con ACID desde 2001 y tiene potentes complementos como el popular extensor de base de datos geoespacial PostGIS.

PostgreSQL viene con muchas funciones destinadas a ayudar a los desarrolladores a crear aplicaciones, a los administradores a proteger la integridad de los datos y crear entornos tolerantes a fallas, y ayudarlo a administrar sus datos sin importar cuán grande o pequeño sea el conjunto de datos. Además de ser gratuito y de código abierto, PostgreSQL es altamente extensible. Por ejemplo, puede definir sus propios tipos de datos, generar funciones personalizadas e incluso escribir código de diferentes lenguajes de programación sin volver a compilar su base de datos [CITATION Qui20 \l 3082].

pgAdmin:

Es la principal herramienta de gestión de código abierto para Postgres. Está diseñado para monitorear y administrar múltiples servidores de bases de datos PostgreSQL y EDB Advanced Server, tanto locales como remotos, a través de una sola interfaz gráfica que permite la fácil creación y administración de objetos de bases de datos, así como una serie de otras herramientas para administrar sus bases de datos. Este se puede instalar en dos modos: modo de escritorio y servidor.

El modo de escritorio se instala como una aplicación independiente que utiliza el mismo usuario del sistema operativo, mientras que se puede acceder al modo de servidor a través de la red, lo que permite que lo utilicen varios usuarios. Las implementaciones de ambos modos siguen un enfoque de arquitectura de 3 niveles [CITATION Bla21 \l 3082].

Firefox Developer Edition:

Mozilla Firefox es un navegador web rápido y completo que es fácil de usar. Developer Edition agrega herramientas de depuración e IDE, así como un tema único para el canal beta estándar de Firefox. Es un navegador hecho para desarrolladores. Proporciona todas las herramientas para desarrolladores más recientes de la versión beta, además de funcionalidades experimentales como el editor multilínea de consola y el inspector de WebSocket. Además de un perfil y ruta separados para que puedas ejecutarlo fácilmente en paralelo a la versión oficial o Beta de Firefox. Ofrece preferencias personalizadas para

desarrolladores web: el navegador y la depuración remota están activados de forma predeterminada, al igual que el tema oscuro y el botón de la barra de herramientas para desarrolladores [CITATION Ren11 \l 3082].

Metodología de desarrollo de software AUP versión UCI:

La Universidad de las Ciencias Informáticas (UCI) desarrolló una versión de la metodología de desarrollo de software AUP (Proceso Ágil Unificado), con el fin de crear una metodología que se adapte al ciclo de vida definido por la actividad productiva de la universidad. Esta versión decide mantener para el ciclo de vida de los proyectos la fase de Inicio, pero modificando el objetivo de la misma y se unifican las restantes fases de la metodología de desarrollo de software AUP en una sola, nombrada Ejecución y agregándole también una nueva fase llamada Cierre [CITATION Mor19 \l 3082]. A continuación, se muestra una tabla con las fases de la metodología AUP-UCI:

Tabla 1: Fases de la variación de AUP para la UCI (Fuente: Elaboración Propia).

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases (Variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
Elaboración Construcción Transición	Ejecución	En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
	Cierre	En esta fase se analizan tanto los resultados del proyecto como la ejecución y se realizan las actividades formales de cierre del proyecto.

La metodología de software AUP-UCI a partir de que el modelado de negocio propone tres variantes a utilizar en los proyectos, como son: CUN (Casos de uso del negocio), DPN (Descripción de proceso de negocio) o MC (Modelo conceptual) y existen tres formas de encapsular los requisitos, los cuales son: CUS (Casos de uso del sistema), HU (Historias de usuario), DRP (Descripción de requisitos por proceso), surgen cuatro escenarios para modelar el sistema en los proyectos, los cuales son:

- ✔ Escenario No 1: Proyectos que modelen el negocio con CUN solo pueden modelar el sistema con CUS.
- ✔ Escenario No 2: Proyectos que modelen el negocio con MC solo pueden modelar el sistema con CUS.
- ✔ Escenario No 3: Proyectos que modelen el negocio con DPN solo pueden modelar el sistema con DRP.
- ✔ Escenario No 4: Proyectos que no modelen negocio solo pueden modelar el sistema con HU [CITATION Sán15 \l 3082].

A partir del análisis e investigación realizada, se utilizó la variante #2 (DPN) y el escenario #4 (HU) de la metodología AUP-UCI, debido a que dicha metodología es apropiada para proyectos pequeños, permitiendo disminuir las probabilidades de fracaso, por ser un producto de fácil uso utilizando cualquier herramienta. Con la adaptación de AUP que se propone para la actividad productiva de la UCI se logra estandarizar el proceso de desarrollo de software. Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos. Además, se selecciona AUP-UCI como metodología de desarrollo de software porque es la metodología establecida para la actividad productiva en la UCI.

Esta aplica técnicas ágiles, incluyendo el desarrollo dirigido por modelado ágil, gestión de cambios ágil y pruebas, lo que permite enfrentarse de forma eficaz, a los requisitos cambiantes que tienen un alto riesgo técnico. Expone un conjunto de actividades orientadas a visualizar, especificar, construir y documentar los artefactos necesarios para el desarrollo de software con la calidad que el cliente necesita. La documentación que se obtiene describe los distintos procesos y facilita el entendimiento del sistema por parte del equipo de

desarrollo, y sirve de referencia para posteriores trabajos. Propicia el trabajo en equipo con objetivos bien definidos, permitiendo seguir de manera clara el avance de las tareas.

Conclusiones del capítulo:

Con el estudio y análisis de los sistemas de gestión de cursos académicos, se logró fundamentar las bases teóricas de la investigación y se analizaron los conceptos básicos asociados al Centro de Idiomas para un mayor entendimiento. Además, se pudieron adquirir los conocimientos previos sobre las características principales de estos sistemas, aunque no cumplen con todas las condiciones requeridas. La caracterización de las herramientas, tecnologías, metodología y lenguajes de programación permitió conocer sus beneficios, así como formar las bases propicias para crear una propuesta de solución, que, a su vez, cumpla con el objetivo de la investigación.

CAPÍTULO II: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

En el presente capítulo se presentan las características de la propuesta de solución; para ello se definen las reglas de negocio. Además, se especifican las técnicas de obtención de requisitos, permitiendo precisar los requisitos funcionales y no funcionales. Se describe la arquitectura con el patrón arquitectónico utilizado, así como los patrones de diseño y de base de datos. Se representan los diagramas de clases del diseño y el modelo de base de datos.

II.1 Descripción del proceso de gestión de cursos académicos

El presente epígrafe se dedica a detallar un poco más el funcionamiento del proceso de gestión de información asociada a los exámenes estandarizados de idiomas. Esto se realiza en consecuencia con la descripción de dicho proceso en términos textuales que se realizó en el capítulo 1. Para detallar este proceso se especifica cómo se hace este en el Centro de Idiomas (CENID) de la Universidad de las Ciencias Informáticas (UCI). Por último, se efectuará una descripción más detallada del funcionamiento del negocio explicando sus reglas.

Descripción de las reglas del negocio:

Una regla de negocio es una regulación que define o restringe acciones dentro de las operaciones de una organización. Son declaraciones que guían el comportamiento y determinan dónde, cuándo, por qué y cómo llevar a cabo las tareas empresariales. Las empresas pueden administrar estas reglas de manera formal o informal, por escrito o digitalmente.

No existe una directriz establecida para aplicar las reglas comerciales, por lo que pueden diferir de una empresa a otra. La prioridad de muchos gerentes es desarrollar estrategias y métodos que puedan mejorar el desempeño de la empresa. Los ejecutivos suelen utilizar las reglas comerciales para establecer los estándares de la empresa, ayudar en la resolución de conflictos y reducir los errores [CITATION Cue19 \l 3082].

El Sistema de Gestión de Idiomas gestiona todas las convocatorias de exámenes de diagnóstico, de colocación y de certificación de los exámenes por habilidades que se

realizan en el Centro de Idiomas. Además, se gestionan los cursos de superación del idioma Inglés por niveles. El sistema cuenta con los módulos de Exámenes y Superación. El Sistema de Gestión de Idiomas se crea con el objetivo de responder a la necesidad de informatización de los procesos del Centro de Idiomas para un correcto funcionamiento y centralización de la información. Tiene como objetivos específicos:

1. Permitir el correcto acceso al sistema de las personas involucradas.

2. Lograr la gestión de los elementos de configuración general del sistema.

3. Gestionar la estructura de las Organizaciones.

4. Efectuar la gestión del accionar de los militantes en la interacción con el sistema.

5. Notificar a los usuarios las acciones importantes efectuadas en el sistema.

6. Permitir la gestión del ingreso de los militantes en la Organización.

7. Permitir la gestión de la plantilla y sus trámites en la Organización.

8. Permitir determinar la ubicación física del expediente del militante en el archivo.

Para una mejor organización de los cursos en el Centro de Idiomas se establecieron algunas reglas. Las reglas son las siguientes:

Generales:

- Los cursos son impartidos durante el transcurso de un semestre.
- Los cursos pueden ser cerrados (destinado a un grupo exclusivo de usuarios) o abiertos (cualquier usuario puede hacer solicitud del mismo).

Matrícula:

- La matrícula de los estudiantes a los cursos es administrada por las respectivas facultades a las que pertenecen.
- Si el usuario no ha realizado anteriormente un examen de colocación, solo puede matricularse en el curso del nivel más bajo.

- El usuario no puede matricular en un curso de mayor nivel del que este haya demostrado dominio en los exámenes de colocación y certificación.
- El usuario no puede matricularse en un curso de nivel inferior al que haya demostrado dominio anteriormente en un examen de certificación o en un curso de nivelación.

Evaluaciones:

- Los usuarios serán evaluados según la cantidad de evaluaciones y frecuencias que definen los actores para cada curso.
- Los actores determinarán la aprobación o no de los usuarios durante cualquier momento del transcurso del curso.

II.2 Requisitos, análisis y diseño del módulo de idioma

Este epígrafe consta principalmente de los artefactos resultantes de la Ingeniería de Requisitos desarrollada al sistema. Además, presenta todo lo relacionado con el modelado resultante del análisis y diseño de la solución propuesta, por lo que contiene la descripción de la arquitectura. Para ello se especifican claramente los requisitos del cliente, ya sean los requisitos funcionales y los requisitos no funcionales; así como la forma en que fueron obtenidos estos. También, en este epígrafe, se describen detalladamente los requisitos funcionales con la realización de sus respectivas historias de usuario. Por otra parte, para el modelado del diseño se confecciona el diagrama de clases del diseño y por último se detallan los patrones de diseño que se utilizaron en la confección del sistema.

Requisitos del Cliente:

Los requisitos claramente definidos son señales esenciales en el camino que conduce a un proyecto exitoso. Establecen un acuerdo formal entre un cliente y un proveedor de que ambos están trabajando para alcanzar el mismo objetivo. Los requisitos detallados y de alta calidad también ayudan a mitigar los riesgos financieros y a mantener el proyecto dentro del cronograma [CITATION Din10 \l 3082].

Los requisitos de la solución describen las características específicas que debe tener un producto para satisfacer las necesidades de las partes interesadas y del negocio mismo. Se dividen en dos grandes grupos:

- Los requisitos funcionales definen lo que debe hacer un producto, cuáles son sus características y funciones.
- Los requisitos no funcionales describen las propiedades generales de un sistema. También se conocen como atributos de calidad [CITATION Mai15 \l 3082].

Para saber exactamente lo que el cliente necesita resulta de vital importancia capturar los requisitos, así como las cualidades o propiedades que estos deben tener. Para el desarrollo de la propuesta de solución se identificaron requisitos funcionales y no funcionales. Los requisitos funcionales se obtuvieron mediante técnicas que favorecen una comunicación más estrecha entre el cliente y el equipo de desarrollo. A continuación se caracterizan estas técnicas.

Definición de las técnicas de obtención de requisitos:

El proceso de obtención de requisitos, cuya finalidad es llevar a la luz los requisitos, no solo es un proceso técnico, sino también un proceso social que envuelve a diferentes personas, lo que conlleva dificultades añadidas a su realización. En la identificación de los requisitos de la propuesta de solución se utilizaron las siguientes técnicas:

- Entrevista: esta herramienta permitió analizar en detalle las necesidades del cliente y posibilitó el estudio de las expectativas del mismo respecto a la propuesta de solución, descubriendo sus motivaciones y actitudes.
- Prototipado: la elaboración de los prototipos permitió mostrar las funcionalidades de la propuesta de solución para su validación por parte de los interesados. Este tipo de técnica sirve para eliminar dudas sobre lo que desea el cliente y además para desarrollar la interfaz más amigable.

Requisitos Funcionales:

Los requisitos funcionales son características del producto que los desarrolladores deben implementar para permitir que los usuarios logren sus objetivos. Definen el comportamiento básico del sistema bajo condiciones específicas.

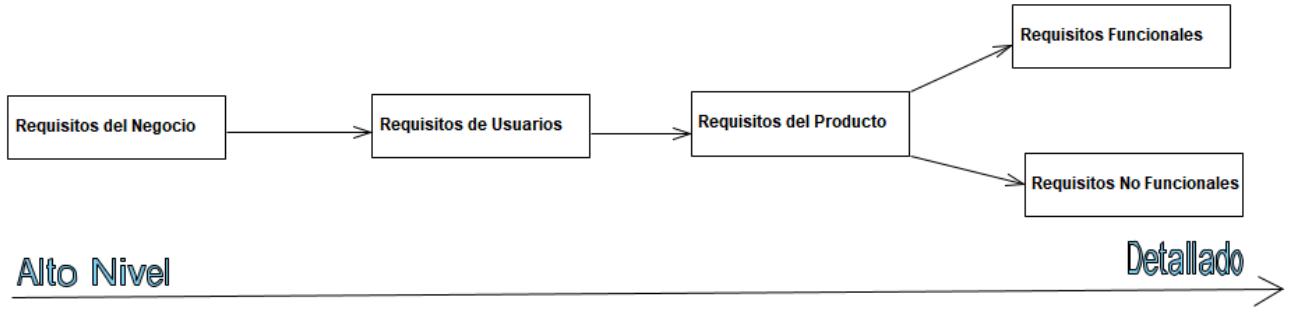

Figura 2: Gestión de Productos (Fuente: Elaboración Propia).

Los requisitos funcionales no deben confundirse con otros tipos de requisitos en la gestión de productos:

- Requisitos del Negocio: Los requisitos del negocio describen las necesidades comerciales de alto nivel, como crear una participación de mercado, reducir la rotación de clientes o mejorar el valor de por vida de los clientes.
- Requisitos de Usuarios: Los requisitos de los usuarios cubren los diferentes objetivos que sus usuarios pueden lograr usando el producto y se documentan comúnmente en forma de historias de usuarios, casos de uso y escenarios.
- Requisitos del Producto: Los requisitos del producto describen cómo debe operar el sistema para cumplir con los requisitos del negocio y del usuario. Incluyen requisitos funcionales y requisitos no funcionales [CITATION Ram17 \l 3082].

Los requisitos funcionales son declaraciones de servicios que el sistema debe proporcionar, cómo debe reaccionar y comportarse él mismo ante insumos y situaciones particulares. En

algunos casos, los requisitos funcionales también pueden indicar explícitamente lo que el sistema no debe hacer.

A continuación se presenta una descripción de cómo funciona el negocio, o sea, como es que se realiza actualmente el proceso de la certificación del idioma inglés en la UCI. Para empezar todo se desencadena de una convocatoria. Al inicio del proceso se lanza una convocatoria de exámenes de certificación, la cual debe tener una fecha de inicio y una fecha de fin. La convocatoria se puede realizar tanto a estudiantes como a trabajadores. Esta está dirigida a 2 tipos de evaluación, las cuales pueden ser examen de certificación o examen de diagnóstico (o colocación). La convocatoria se lanza y cuando llega a la fecha de fin se cierra.

Al cerrar la convocatoria se genera un listado de las solicitudes hechas por los estudiantes o profesores para asistir a esas convocatorias, las cuales se pueden aceptar o denegar por parte del director del centro de idiomas, el subdirector del centro de idiomas o por la secretaria docente. Luego se pasa a la próxima fase del proceso que es asignar el tribunal para el cuidado de las 3 habilidades que son Audición (A), Lectura (L) y Escritura (E). También en esta parte se debe asignar el local donde se van a evaluar cada una de estas habilidades, las cuales se evalúan todas el mismo día y se genera un Acta de Comparecencia de los estudiantes.

El siguiente subproceso es asignar el tribunal para calificar a los estudiantes por habilidad. De aquí sale un tribunal para cada habilidad, por lo que son 3 tribunales. Como salida de este subproceso se obtiene una nota por habilidad y se genera un Acta de Evaluación de las Habilidades. Más adelante, se selecciona el tribunal para otorgar el derecho a realizar la prueba oral, el cual tendrá como criterio de aprobación que para el estudiante poder asistir a esta prueba debe tener las siguientes notas por habilidad: $A \geq A1$, $L \geq A2$ y $E \geq A2$.

Como resultado de este subproceso se genera el Acta de Comparecencia a la prueba Oral y se asigna el tribunal para calificar esta prueba y el local donde van a sesionar cada uno de estos tribunales. Al calificar esta habilidad se genera el Acta de Evaluación de la prueba Oral. En otro momento se genera el tribunal para el compendio de notas por habilidad, el cual de

acuerdo a las notas que el estudiante haya obtenido en las habilidades (ALEO), emite una nota final que sería la nota de certificación.

Aquí se confecciona el Acta del Compendio. Para finalizar con todo este proceso se elabora un Hago Contar, el cual es un documento donde se guarda todas las notas de los estudiantes en cada una de las habilidades, así como su nota de certificación. Por último se genera el Diploma de Certificación del Nivel de Dominio del Idioma Inglés que se entrega a los estudiantes que hayan cumplido el requisito de graduación.

Como consecuencia de este proceso surgen los requisitos funcionales del presente trabajo de diploma. A continuación se presentan en una tabla los requisitos funcionales donde se muestran los requisitos funcionales del módulo Idiomas, con un total de 49, de los cuales se identificaron 7 de complejidad alta, 1 media y 41 baja, también se estableció la prioridad para el cliente de cada uno de ellos, clasificando 19 de prioridad alta, 8 media y 22 baja.

Tabla 2: Requisitos funcionales del Módulo Idiomas (Fuente: Elaboración Propia).

Número	Nombre	Prioridad	Complejidad
RF1	Crear convocatoria de examen	Alta	Alta
RF2	Mostrar convocatorias de exámenes	Media	Baja
RF3	Modificar convocatoria de examen	Alta	Alta
RF4	Detalles de la convocatoria de examen	Baja	Baja
RF5	Crear tipo de convocatoria del examen	Baja	Baja
RF6	Mostrar tipo de convocatoria del examen	Media	Baja
RF7	Modificar tipo de convocatoria del examen	Baja	Baja
RF8	Detalles de tipo de convocatoria del examen	Baja	Baja
RF9	Crear estado de la convocatoria	Baja	Baja

RF10	Crear nivel de calificación	Baja	Baja
RF11	Mostrar nivel de calificación	Media	Baja
RF12	Modificar nivel de calificación	Baja	Baja
RF13	Detalles de nivel de calificación	Baja	Baja
RF14	Mostrar estado de la convocatoria	Media	Baja
RF15	Modificar estado de convocatoria.	Baja	Baja
RF16	Detalles de estado de convocatoria.	Baja	Baja
RF17	Crear estado de solicitud	Baja	Baja
RF18	Mostrar estados de solicitud	Media	Baja
RF19	Modificar estado de solicitud	Baja	Baja
RF20	Detalles de estado de solicitud	Baja	Baja
RF21	Crear forma de evaluación	Baja	Baja
RF22	Mostrar forma de evaluación	Media	Baja
RF23	Modificar forma de evaluación	Baja	Baja
RF24	Detalles de forma de evaluación	Baja	Baja
RF25	Crear habilidad	Baja	Baja
RF26	Mostrar habilidad	Media	Baja
RF27	Modificar habilidad	Baja	Baja
RF28	Detalles habilidad	Baja	Baja

RF29	Asignar local por habilidad	Alta	Alta
RF30	Asignar personas a local	Alta	Alta
RF31	Exportar listado de solicitudes	Baja	Media
RF32	Solicitud de convocatoria de exámenes	Alta	Alta
RF33	Asignar tribunal de cuidado de habilidad a local	Alta	Baja
RF34	Asignar tribunal de calificación de estudiantes por habilidad	Alta	Baja
RF35	Generar Acta de Comparecencia	Alta	Baja
RF36	Crear nota por habilidad	Alta	Baja
RF37	Mostrar nota por habilidad	Media	Baja
RF38	Modificar nota por habilidad	Baja	Baja
RF39	Detalles de nota por habilidad	Baja	Baja
RF40	Generar Acta de Evaluación por habilidad	Alta	Baja
RF41	Seleccionar tribunal de Derecho a prueba Oral	Alta	Baja
RF42	Generar Acta de Comparecencia a prueba Oral	Alta	Baja
RF43	Generar tribunal de prueba Oral	Alta	Baja
RF44	Generar Acta de evaluación de prueba Oral	Alta	Baja
RF45	Generar tribunal para Compendio de Notas por habilidad	Alta	Baja
RF46	Generar Acta Compendio	Alta	Baja

RF47	Generar nota de certificación	Alta	Alta
RF48	Generar Hago Contar	Alta	Baja
RF49	Generar Diploma de Certificación del Nivel de Dominio del Idioma Inglés	Alta	Baja

Requisitos No Funcionales:

A continuación se pasará a detallar en qué consisten los requisitos no funcionales. De ahí que es importante que un producto funcione correctamente y tenga características distinguibles. Con requisitos no funcionales, se puede crear un producto con propiedades únicas. Conocer ejemplos de requisitos no funcionales y cómo funcionan en una aplicación puede ayudarle a diseñar un sistema que satisfaga las necesidades de sus usuarios finales.

Un requisito no funcional es un atributo que dicta cómo funciona un sistema. Hace que las aplicaciones o el software funcionen de manera más eficiente e ilustra la calidad del sistema. Los requisitos no funcionales difieren de los requisitos funcionales de las siguientes maneras:

- Obligatorio frente a no obligatorio: a diferencia de los requisitos funcionales, las características no funcionales no son obligatorias para que un sistema funcione. En cambio, estas características pueden ayudar a diferenciar una aplicación de otros productos en el mercado.
- Operaciones básicas frente a características adicionales: los requisitos funcionales abarcan lo que hace un sistema, mientras que los requisitos no funcionales cubren cómo un sistema completa una tarea. Por ejemplo, el deber funcional de una cámara es tomar fotografías. El deber no funcional es tomar fotografías con mayor enfoque y claridad.
- Finalidad prevista frente a expectativas del cliente: mientras que los requisitos funcionales se centran en el objetivo de una aplicación, los requisitos no funcionales se centran en las expectativas de los usuarios, como el rendimiento del producto [CITATION Min21 \l 3082].

Luego de analizar cómo será la confección del módulo Idiomas se detectaron una serie de requisitos no funcionales a tener en cuenta. Los módulos propuestos cuentan con los siguientes requisitos no funcionales:

Usabilidad:

- Solo se mostrarán a los usuarios aquellas acciones o informaciones a las que por su responsabilidad o rol dentro del negocio necesitan acceder, mostrando en la vista mediante iconos y menús el acceso a la misma.
- El tiempo de respuesta del sistema no excederá los 3 segundos de manera general.
- Los términos visibles para el usuario en la aplicación serán los utilizados por los usuarios en la rama abordada con vista a una mayor comprensión por parte de los mismos de la herramienta de trabajo.
- Las vistas del sistema deben indicar en cada momento la acción que se está realizando así como los iconos deben estar representados por una imagen acorde a la acción que se realiza mediante el mismo.
- Para el despliegue del sistema se contará en el servidor de base de datos con Postgres 8.4 o superior bajo el sistema operativo CentOS 7.0 o superior.
- Para el despliegue del sistema estará instalado en el servidor de aplicaciones web con el sistema operativo CentOS 6.2 donde se encuentre instalado PHP v7.2 con las librerías php7-ldap, php7-gd, php7-mcrypt, php7-pgsql, php7-xsl, php7-openssl, Nginx y JDK 6 o superior.
- Para el uso del sistema se requiere una PC cliente con el navegador web Mozilla Firefox 3.6 o superior para el uso de la aplicación web y la máquina virtual de Java JDK 6 para el uso del Sistema de Impresión de Documentos.
- Para el uso del sistema se requiere una PC cliente con los siguientes sistemas operativos: Windows o GNU/Linux.

- El servidor de aplicación y el de base de datos tendrán los siguientes componentes de hardware: Microprocesador de 8 núcleos, 4 GB de memoria RAM, 250 GB disco duro y una fuente de 800 W como mínimo.
- Para la ejecución del sistema se requiere que la PC cliente tenga los siguientes componentes de hardware: Pentium 4 o superior, 512 MB RAM y 1 GB disco duro disponible como mínimo.

Confiabilidad:

- El sistema puede permanecer inactivo durante 10 minutos. Al cumplirse este término se cerrará la sesión teniendo que autenticarse el usuario nuevamente.
- La información que se registra estará protegida de modificaciones no deseadas de acuerdo a la estrategia de seguridad definida.
- El tratamiento de las excepciones permitirá un seguimiento hasta guardar información acerca del lugar donde se produjo el error y de los parámetros de configuración del sistema que lo provocaron. Cuando ocurre una excepción, el sistema mostrará un mensaje explicativo del error ocurrido y permanecerá en el mismo estado sin realizar ninguna otra operación.
- El sistema de autenticación recogerá los datos necesarios de los usuarios y tendrá control de identidades.
- El sistema debe proveer algún mecanismo seguro de encriptación y transferencia de datos.
- El sistema de autenticación permite auditoría de trazas de las acciones de los usuarios, para lo que se proveerá algún mecanismo para la configuración dinámica y centralizada de la gestión de historiales.
- Se debe mantener una seguridad a nivel de usuarios y contraseñas encriptadas para el acceso a la base de datos.

- El servidor de aplicaciones y de base de datos deberá mantener una seguridad mediante un cortafuego para proteger el código y la información.

Soporte:

- El sistema se regirá por un estándar de código debidamente documentado en el expediente de proyecto.
- Todos los productos de trabajo generados en el desarrollo del software se regirán por unas pautas de configuración debidamente documentadas en el expediente de proyecto.

Requisitos de interfaz:

- La comunicación entre el cliente y el servidor de aplicaciones se lleva a cabo a través del protocolo HTTPS.
- El sistema se integrará con el directorio activo de la institución cliente mediante el protocolo LDAP.
- Existirá un componente interno para la validación correcta de las entradas/salida de datos de cada una de las interfaces de la aplicación.
- El sistema deberá permitir la integración hacia y desde sistemas externos existentes en el ambiente de despliegue mediante servicios web por el protocolo SOAP.
- El sistema deberá implementar una solución arquitectónica para permitir la integración segura entre componentes internos.
- La interfaz de usuario se guiará por la vista de arquitectura de presentación.
- La comunicación entre el servidor de aplicaciones y la base de datos se lleva a cabo a través del protocolo TCP/IP.
- Se tendrán centralizados los archivos de configuración. Para ellos se utilizarán archivos XML que permitirán el fácil mantenimiento del sistema si cambia alguna de las configuraciones que posee.

Descripción de Historias de usuarios:

En el desarrollo de software y la gestión de productos, una historia de usuario es una descripción informal en lenguaje natural de una o más características de un sistema de software. También es una herramienta usada en el desarrollo de software Agile para capturar una descripción de una función de software desde la perspectiva del usuario final. Además, describe el tipo de usuario, lo que quiere y por qué. Una historia de usuario ayuda a crear una descripción simplificada de un requisito.

Las historias de usuarios a menudo se registran en fichas, en notas Post-it o en software de gestión de proyectos. Dependiendo del proyecto, las historias de usuario pueden ser escritas por varias partes interesadas, como clientes, usuarios, gerentes o miembros del equipo de desarrollo [CITATION Ven17 \l 3082].

Los requisitos siempre cambian a medida que los equipos y los clientes aprenden más sobre el sistema a medida que avanza el proyecto. No es exactamente realista esperar que los equipos de proyectos trabajen con una lista de requisitos estáticos y luego entreguen un software funcional meses después. Con el enfoque de la historia del usuario, reemplazamos el gran diseño inicial con un enfoque "solo lo suficiente". Las historias de usuarios reducen el tiempo dedicado a escribir documentación exhaustiva al enfatizar las conversaciones centradas en el cliente [CITATION Sol15 \l 3082].

En consecuencia, las historias de usuarios permiten que los equipos entreguen software de calidad más rápidamente, que es lo que prefieren los clientes. Hay bastantes beneficios al adoptar el enfoque de historia de usuario en el desarrollo ágil, como:

1) El formato simple y consistente ahorra tiempo al capturar y priorizar los requisitos, al tiempo que sigue siendo lo suficientemente versátil como para usarse en características grandes y pequeñas por igual.

2) Manténgase expresando el valor comercial mediante la entrega de un producto que el cliente realmente necesita

3) Evite introducir detalles demasiado pronto que impidan las opciones de diseño y bloqueen de manera inapropiada a los desarrolladores en una solución.

4) Evitar la apariencia de falsa integridad y claridad.

5) Llegue a partes lo suficientemente pequeñas que inviten a la negociación y al movimiento en la cartera de pedidos

6) Deje las funciones técnicas al arquitecto, desarrolladores, probadores, entre otros [CITATION Veg20 \l 3082].

Una historia de usuario es un método ligero para capturar rápidamente el "quién", "qué" y "por qué" de un requisito de producto. En términos simples, las historias de usuarios son ideas establecidas de requisitos que expresan lo que necesitan los usuarios. Las historias de usuarios son breves y cada elemento suele contener menos de 10 o 15 palabras cada uno. Las historias de usuario son listas de "cosas por hacer" que lo ayudan a determinar los pasos a lo largo de la ruta del proyecto. Ayudan a garantizar que su proceso, así como el producto resultante, cumpla con sus requisitos [CITATION Mai15 \l 3082]. Seguidamente, se exponen 2 historias de usuario de las descritas en los Anexos.

Tabla 3: HU_Crear_convocatoria (Fuente: Elaboración Propia).

Historia de Usuario	
Número: HU_10	**Usuario: Administrador**
Requisito: Crear convocatoria de examen	
Prioridad del negocio: Alta	**Iteración Asignada:** 1
Riesgo en Desarrollo: Baja	**Tiempo Estimado:** 3 días
Programador: José Antonio Ávalos Vielza	**Tiempo Real:** 3 días
Descripción: El requisito permite Crear una nueva convocatoria de examen. El usuario selecciona del escritorio de Sistema de Gestión Universitaria, el sistema Idioma, el módulo "Idioma", luego en el menú lateral, la agrupación funcional "Convocatorias" y la funcionalidad "Convocatorias de exámenes". En el área de íconos flotantes, selecciona la acción Crear una convocatoria de	

exámenes. Se muestra una vista donde se registran los siguientes datos de la convocatoria: Nombre de la convocatoria, Matrícula, Idioma, Tipo de la convocatoria, Fecha fin de la convocatoria, Fecha inicio de la convocatoria, Destinado a, Fecha inicio de exámenes, Fecha fin de exámenes, Fecha fin de la solicitud, Tipo de examen, Nivel, Carrera, Categoría de estructura, Estructura Facultad, Año académico, Grupo docente, Forma de evaluación y Descripción. Además, se muestra la opción Listar en el área de iconos flotantes.
Observaciones: El usuario debe estar autenticado en el sistema con los permisos necesarios para acceder a la funcionalidad. No debe existir una convocatoria con el mismo nombre. Los campos **Nombre**, **Matrícula**, **Idioma, Tipo de convocatoria, Fecha inicio de la convocatoria, Fecha fin de la convocatoria, Tipo de examen, Forma de evaluación** y **Nivel** son obligatorios.

Prototipo elemental de interfaz gráfica de usuario:

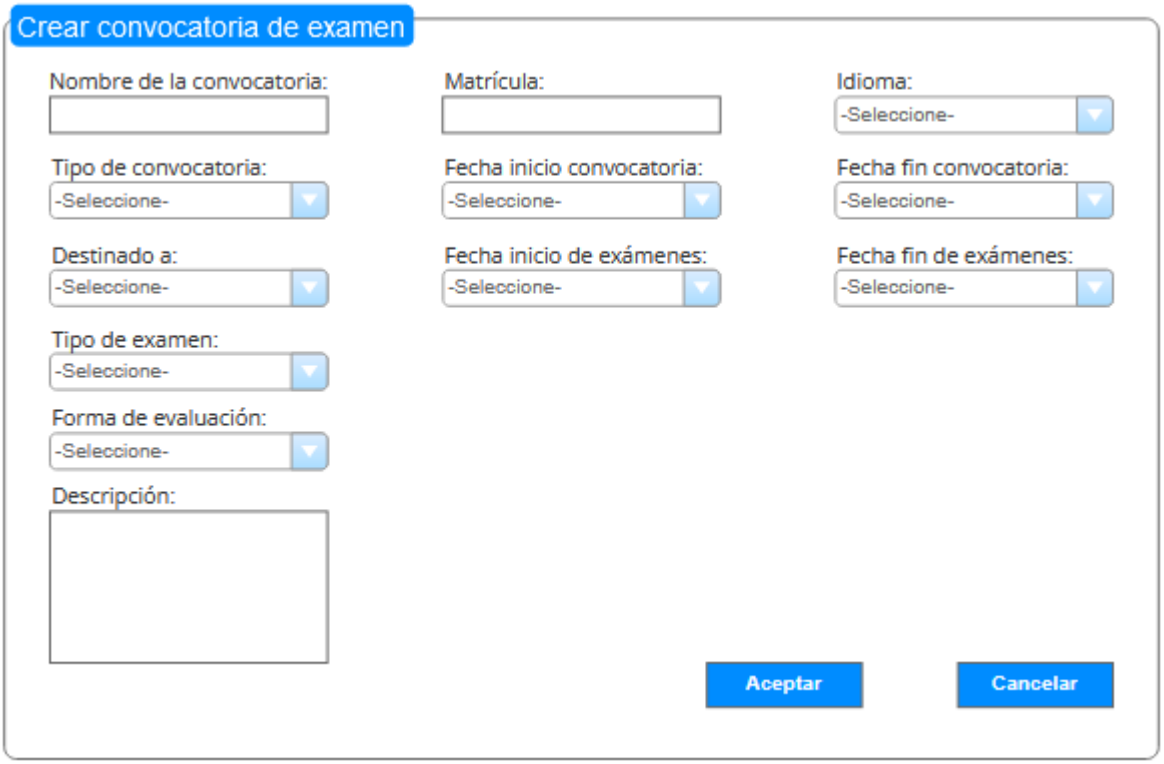

Figura 3: Interfaz gráfica para crear convocatoria de examen (Fuente: Elaboración Propia).

Tabla 4: HU_Crear_habilidad (Fuente: Elaboración Propia).

Historia de Usuario	
Número: HU_18	**Usuario: Administrador**
Requisito: Crear habilidad	
Prioridad del negocio: Baja	**Iteración Asignada:** 1
Riesgo en Desarrollo: Baja	**Tiempo Estimado:** 2 días
Programador: Manuel Alejandro Fernández Rojo	**Tiempo Real:** 2 días
Descripción: El requisito permite crear una habilidad. El administrador selecciona del módulo Gestión de idioma del Sistema de Gestión de Idioma, luego en el menú lateral la agrupación funcional, Configuración, funcionalidad, Habilidad. En el área de iconos flotantes, selecciona la opción Crear	

habilidad. Se muestran los datos a llenar: Nombre, Descripción y Habilitado (Activo o no). Además de la opción: Listar en el área de iconos flotantes.
Observaciones: El usuario debe estar autenticado en el sistema con los permisos necesarios para acceder a la funcionalidad. No debe existir una habilidad con el mismo nombre. El **Nombre** es un campo obligatorio.

Prototipo elemental de interfaz gráfica de usuario:

Figura 4: Interfaz gráfica para crear habilidad (Fuente: Elaboración Propia).

Descripción de la arquitectura y diseño:

La propuesta de solución forma parte de un conjunto de módulos del componente Idiomas, por esta razón su arquitectura queda definida por dicho sistema: la Cliente-Servidor. Asimismo, el patrón arquitectónico empleado es el Modelo-Vista-Controlador, por sus

múltiples ventajas y por haber sido diseñado el marco de trabajo del sistema base para usarlo.

Arquitectura Cliente-Servidor:

Ahora, en estos días, la mayoría de las organizaciones requieren un sistema que sea fácil de procesar, recopilar y trabajar con datos corporativos, y luego mejorar la eficiencia de los procedimientos comerciales y garantizar la supervivencia en los mercados mundiales avanzados. Por lo tanto, la arquitectura del cliente-servidor ofrece el marco perfecto que hoy en día, todas las empresas requieren para enfrentar los desafíos del sector de las Tecnologías de la Información en rápida evolución.

La arquitectura cliente-servidor es una arquitectura de red informática compartida donde varios clientes (sistema remoto) envían muchas solicitudes y finalmente obtienen servicios de la máquina servidor centralizado (sistema host). La máquina del cliente ofrece una interfaz fácil de utilizar que ayuda a los usuarios a activar los servicios de solicitud de la computadora del servidor y, finalmente, a mostrar su salida en el sistema del cliente [CITATION Gon17 \l 3082].

La función principal de la arquitectura cliente-servidor es como un sistema de almacenamiento de datos. En este enfoque, todos los datos y aplicaciones en los dispositivos de almacenamiento se mantienen almacenados en el servidor remoto. Cada vez que un cliente requiere obtener acceso a un archivo o aplicación específica, envía una solicitud al servidor. La arquitectura cliente-servidor también se denomina “Red cliente/servidor” o “Modelo de computación en red”, porque en esta arquitectura todos los servicios y solicitudes se distribuyen en la red. Su funcionalidad es como la de un sistema informático distribuido en el que todos los componentes realizan sus tareas de forma independiente entre sí.

Componentes de la Arquitectura Cliente Servidor:

La arquitectura cliente-servidor contiene tres componentes, las estaciones de trabajo, el servidor y los dispositivos de red, y están conectados entre sí. A continuación se detallan cada uno de ellos:

1) Estación de trabajo: la estación de trabajo también se conoce como “computadora cliente”. Existen diferentes tipos de sistemas operativos, que se instalan en las estaciones de trabajo, como Windows 2000, Windows XP, Windows Vista, Windows 7 y Windows 10. Estos sistemas operativos para estaciones de trabajo son más económicos en comparación con los sistemas operativos de servidor.

2) Servidor: el servidor es un sistema informático de alto rendimiento que contiene la memoria más rápida, más espacio en el disco duro y procesadores de mayor velocidad porque guardan y atienden varias solicitudes que provienen del lado de la estación de trabajo. Un servidor desempeña diferentes tipos de roles, como servidor de correo, servidor de base de datos, servidor de archivos y controlador de dominio al mismo tiempo.

3) Dispositivos de red: con la ayuda de dispositivos de red; las estaciones de trabajo y los servidores están conectados entre sí. Cada dispositivo de red tiene una funcionalidad propia, como el concentrador que se usa para hacer la conexión entre el servidor y varias estaciones de trabajo, el repetidor se usa para mover datos de un dispositivo a otro y los puentes ayudan a aislar todos los segmentos de la red [CITATION Luj02 \l 3082].

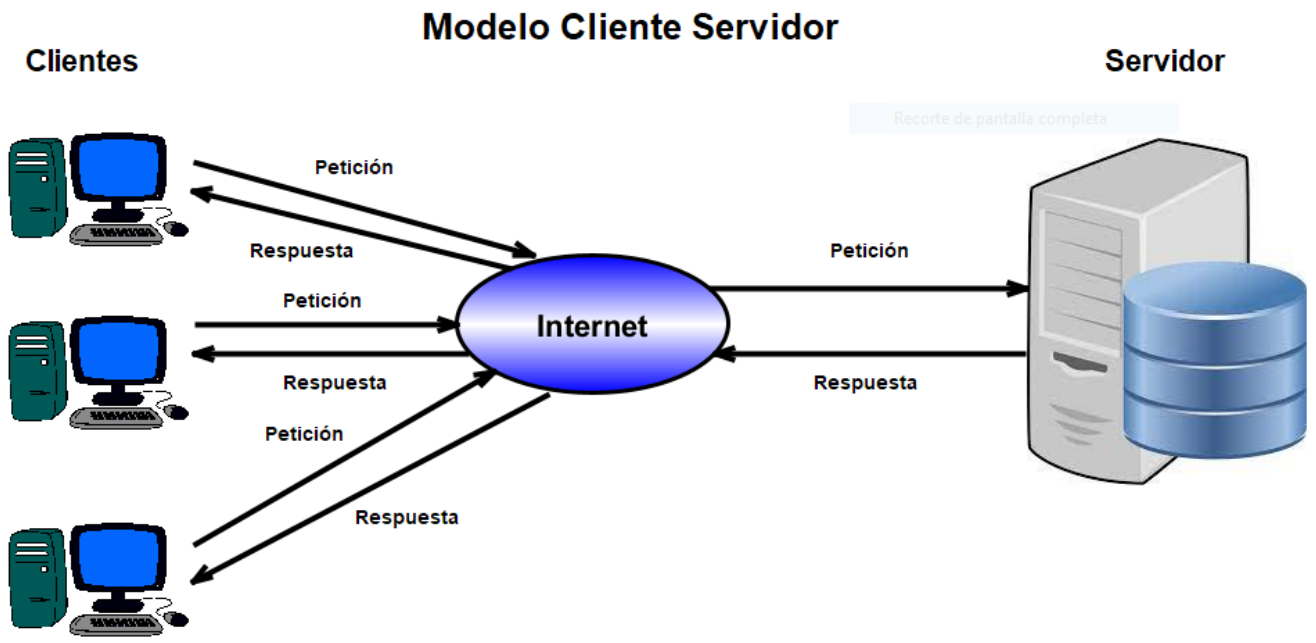

Figura 5: Representación de la arquitectura Cliente - Servidor (Fuente: Elaboración Propia).

Patrón de arquitectura:

Un patrón se puede definir como una idea que es útil en uno o más contextos prácticos para dar una solución reutilizable a un problema. Los patrones arquitectónicos tienen un alcance más amplio. Muchas arquitecturas de software pueden tener el mismo patrón arquitectónico. Estos ayudan a resolver muchos problemas en la ingeniería de software. Un patrón arquitectónico es una colección de aplicación de decisiones de diseño arquitectónico en un contexto dado. Algunos patrones arquitectónicos son básicos, mientras que otros son derivados o una mejora de los patrones arquitectónicos básicos. Un patrón arquitectónico proporciona la solución para diseñar una arquitectura de software.

Para una funcionalidad en particular y la naturaleza del software, los patrones arquitectónicos difieren. Un patrón arquitectónico transmite solo la imagen del sistema, no los detalles al respecto. Aunque tenemos muchos patrones arquitectónicos, elegir el adecuado para el sistema es complicado. No solo depende de la funcionalidad a entregar, sino que también sé considera la tecnología y el uso futuro. También se considera para minimizar el costo de reelaboración porque cada patrón de arquitectura define la forma de construir una

arquitectura que luego define el sistema [58]. Para el desarrollo de este proyecto se utilizará el patrón arquitectónico Modelo Vista Controlador (MVC).

El MVC es un patrón arquitectónico que separa una aplicación en tres componentes lógicos principales: el modelo, la vista y el controlador. Cada uno de estos componentes está diseñado para manejar aspectos de desarrollo específicos de una aplicación. MVC es uno de los marcos de desarrollo web estándar de la industria más empleados para crear proyectos escalables y extensibles. Los siguientes son los componentes de MVC:

1) Modelo: El componente Modelo corresponde a toda la lógica relacionada con los datos con la que trabaja el usuario. Esto puede representar los datos que se transfieren entre los componentes Vista y Controlador o cualquier otro dato relacionado con la lógica empresarial. Por ejemplo, un objeto Cliente recuperará la información del cliente de la base de datos, la manipulará y actualizará los datos en la base de datos o la usará para representar datos.

2) Vista: El componente Vista se utiliza para toda la lógica de la interfaz de usuario de la aplicación. Por ejemplo, la vista Cliente incluirá todos los componentes de la interfaz de usuario, como cuadros de texto, menús desplegables, entre otros, con los que interactúa el usuario final.

3) Controlador: Los controladores actúan como una interfaz entre los componentes Modelo y Vista para procesar toda la lógica comercial y las solicitudes entrantes, manipular datos empleando el componente Modelo e interactuar con las Vistas para generar el resultado final. Por ejemplo, el controlador del Cliente manejará todas las interacciones y entradas de la Vista del Cliente y actualizará la base de datos utilizando el Modelo del Cliente. El mismo controlador se empleará para ver los datos del Cliente [CITATION San20 \l 3082].

Patrones de diseño:

A continuación se presentan los patrones utilizados en el desarrollo de la propuesta de solución. Los patrones de diseño son soluciones habituales a problemas que ocurren con frecuencia en el diseño de software. Son como planos prefabricados que se pueden

personalizar para resolver un problema de diseño recurrente en tu código. No se puede elegir un patrón y copiarlo en el programa como si se tratara de funciones o bibliotecas ya preparadas. El patrón no es una porción específica de código, sino un concepto general para resolver un problema particular. Puedes seguir los detalles del patrón e implementar una solución que encaje con las realidades de tu propio programa.

A menudo los patrones se confunden con algoritmos porque ambos conceptos describen soluciones típicas a problemas conocidos. Mientras que un algoritmo siempre define un grupo claro de acciones para lograr un objetivo, un patrón es una descripción de más alto nivel de una solución. El código del mismo patrón aplicado a dos programas distintos puede ser diferente. Una analogía de un algoritmo sería una receta de cocina: ambos cuentan con pasos claros para alcanzar una meta. Por su parte, un patrón es más similar a un plano, ya que puedes observar cómo son sus resultados y sus funciones [CITATION Rey04 \l 3082].

La mayoría de los patrones se describen con mucha formalidad para que la gente pueda reproducirlos en muchos contextos. Aquí tienes las secciones que suelen estar presentes en la descripción de un patrón:

- El propósito del patrón explica brevemente el problema y la solución.
- La motivación explica en más detalle el problema y la solución que brinda el patrón.
- La estructura de las clases muestra cada una de las partes del patrón y el modo en que se relacionan.
- El ejemplo de código en uno de los lenguajes de programación más populares facilita la asimilación de la idea que se esconde tras el patrón [CITATION Rey04 \l 3082].

Patrones GRASP:

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. Es importante entender y poder aplicar estos principios durante la preparación de un diagrama de interacción, pues un diseñador de software sin mucha experiencia en la tecnología de objetos debe dominarse cuanto antes: constituyen el fundamento de cómo se diseñará el sistema.

GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades). El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos [CITATION Roj15 \l 3082].

Algunos ejemplos de estos patrones son: experto, creador, bajo acoplamiento, alta cohesión y controlador. A continuación se muestran los patrones utilizados en el desarrollo de la propuesta de solución:

- **Experto:** es el encargado de asignar una responsabilidad al experto en información, es decir, la clase que cuenta con la información necesaria para cumplir la responsabilidad. Es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la “intuición” de que los objetos hacen cosas relacionadas con la información que poseen. Este patrón se evidencia en las clases librerías del sistema, las cuales cuentan con la información necesaria para cumplir la responsabilidad sobre los elementos del negocio.

- **Creador:** este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. Por ejemplo: asignarle a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:

✔ B agrega los objetos A.

✔ B contiene los objetos A.

✔ B registra las instancias de los objetos A.

✔ B utiliza específicamente los objetos A.

✔ B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es un Experto respecto a la creación de A).

- ✔ De ahí que B es un creador de los objetos A. Si existe más de una opción, prefiera la clase B que agregue o contenga la clase A.

- **Bajo acoplamiento:** se encarga de asignar una responsabilidad para mantener bajo acoplamiento. Es decir, estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento.

- ✔ El Bajo Acoplamiento soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. No puede considerarse en forma independiente de otros patrones como Experto o Alta Cohesión, sino que más bien ha de incluirse como uno de los principios del diseño que influyen en la decisión de asignar responsabilidades.

- **Alta Cohesión:** se encarga de asignar una responsabilidad de modo que la cohesión siga siendo alta. Una regla práctica de este patrón es la siguiente: una clase con mucha cohesión posee un número relativamente pequeño, con una importante funcionalidad relacionada y poco trabajo por hacer. Colabora con otros objetos para compartir el esfuerzo si la tarea es grande.

- ✔ Una clase con mucha cohesión es útil porque es bastante fácil darle mantenimiento, entenderla y reutilizarla. Su alto grado de funcionalidad, combinada con una reducida cantidad de operaciones, también simplifica el mantenimiento y los mejoramientos. La ventaja que significa una gran funcionalidad también soporta un aumento de la capacidad de reutilización.

- **Controlador:** es el encargado de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Un **evento del sistema** es un evento de alto nivel generado por un actor externo; es un evento de entrada externa. Se asocia a **operaciones del sistema**: las que emite en respuesta a los eventos del sistema. Un **Controlador** es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. El patrón controlador se refleja en las clases controladoras pertenecientes al sistema, que son las clases que

se encargan de obtener datos y enviarlos a las librerías y las vistas [CITATION Roj15 \l 3082].

Patrones GoF:

Los patrones de diseño de GoF se dividen en tres categorías: creacionales, estructurales y de comportamiento. Los patrones de creación se refieren al proceso de creación de objetos. Los patrones estructurales se ocupan de la composición de clases u objetos. Los patrones de comportamiento caracterizan las formas en que las clases u objetos interactúan y distribuyen la responsabilidad [CITATION Ort15 \l 3082].

Los patrones de creación que existen son los siguientes: fábrica abstracta, constructor, método de fábrica, prototipo y único. A continuación se describen cada uno de ellos:

1. **Fábrica Abstracta (Abstract Factory):** se encarga de proporcionar una interfaz para crear familias de personas relacionadas u objetos dependientes sin especificar sus clases concretas.
2. **Constructor (Builder):** se encarga de separar la construcción de un objeto complejo de su representación para que un mismo proceso de construcción pueda crear diferentes representaciones.
3. **Método de Fábrica (Factory Method):** se encarga de definir una interfaz para crear un objeto, pero deja que las subclases decidan qué clase instanciar. Este permite que una clase difiera la creación de instancias a subclases.
4. **Prototipo (Prototype):** especifica los tipos de objetos para crear usando una instancia prototípica, y crea nuevos objetos copiando este prototipo.
5. **Único (Singleton):** se asegura de que una clase solamente tenga una instancia y proporcione un punto global de acceso a él [CITATION Ort15 \l 3082].

Por otro lado, los patrones estructurales existentes son los siguientes: adaptador, puente, compuesto, decorador, fachada, peso mosca y proxy. A continuación se describen cada uno de ellos:

1) **Adaptador (Adapter):** convierte la interfaz de una clase en otra interfaz que esperan los clientes. Permite que las clases trabajen juntas que de otro modo no podrían debido a las interfaces incompatibles.

2) **Puente (Bridge):** permite separar una abstracción de su implementación para que las dos puedan variar independientemente.

3) **Compuesto (Composite):** compone objetos en estructuras de árbol para representar jerarquías de parte-todo. Permite a los clientes tratar objetos individuales y composiciones de objetos uniformemente.

4) **Decorador (Decorator):** adjunta responsabilidades adicionales a un objeto de forma dinámica. Proporciona una alternativa flexible a la creación de subclases para ampliar la funcionalidad.

5) **Fachada (Facade):** proporciona una interfaz unificada a un conjunto de interfaces en un subsistema. Define una interfaz de nivel superior que facilita el uso del subsistema.

6) **Peso Mosca (Flyweight):** utiliza el uso compartido para admitir una gran cantidad de objetos de granularidad de manera eficiente.

7) **Proxy:** proporciona un sustituto o marcador de posición para otro objeto para controlar el acceso a eso [CITATION Ort15 \l 3082].

Por último, los patrones de comportamiento existentes son los siguientes: cadena de responsabilidad, comando, intérprete, iterador, mediador, recuerdo, observador, estado, estrategia, método de plantilla y visitante. A continuación se describen cada uno de ellos:

Cadena de Responsabilidad (Chain of Responsibility): evita vincular el remitente de una solicitud a su receptor por dar a más de un objeto la oportunidad de manejar la solicitud. Encadena la recepción de objetos y pasar la solicitud a lo largo de la cadena hasta que un objeto la maneje.

Comando (Command): encapsula una solicitud como un objeto, lo que le permite parametrizar clientes con diferentes solicitudes, solicitudes de cola o registro, y compatibilidad con deshacer operaciones.

Intérprete (Interpreter): dado un idioma, define una representación para su gramática junto con un intérprete que usa la representación para interpretar oraciones en el idioma.

Iterador (Iterator): proporciona una forma de acceder secuencialmente a los elementos de un objeto agregado sin exponer su representación subyacente.

Mediador (Mediator): define un objeto que encapsula cómo interactúa un conjunto de objetos. Promueve el acoplamiento flexible al evitar que los objetos se refieran entre sí explícitamente, y te permite variar su interacción de forma independiente.

Recuerdo (Memento): sin violar la encapsulación, captura y externaliza el estado interno del objeto para que el objeto pueda ser restaurado a este estado más tarde.

Observador (Observer): define una dependencia de uno a muchos entre objetos de modo que cuando un objeto cambia de estado, todos sus dependientes son notificados y actualizados automáticamente.

Estado (State): permite que un objeto altere su comportamiento cuando cambia su estado interno. El objeto aparecerá para cambiar su clase.

Estrategia (Strategy): define una familia de algoritmos, encapsula cada uno y los hace intercambiables. Permite que el algoritmo varíe independientemente de los clientes que lo usan.

Método de plantilla (Template Method): define el esqueleto de un algoritmo en una operación, aplazando algunos pasos a las subclases. Permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar la estructura del algoritmo.

Visitante (Visitor): representa una operación a realizar sobre los elementos de un objeto o estructura. Permite definir una nueva operación sin cambiar las clases de los elementos sobre los que opera [CITATION Ort15 \l 3082].

Modelado del diseño:

El modelo de diseño es un modelo de objetos que describe la realización de casos de uso y sirve como una abstracción del modelo de implementación y su código fuente. El modelo de

diseño se utiliza como entrada esencial para las actividades de implementación y prueba. Se utiliza para concebir y documentar el diseño del sistema de software. Es un artefacto completo y compuesto que abarca todas las clases de diseño, subsistemas, paquetes, colaboraciones y las relaciones entre ellos. Este establece principalmente la arquitectura, pero también se utiliza como vehículo de análisis durante la fase de elaboración.

Luego se refina mediante decisiones de diseño detalladas durante la fase de construcción. El modelo de diseño se mantiene continuamente consistente con el modelo de casos de uso y el modelo de implementación [CITATION Pér12 \l 3082]. Un buen modelo de diseño tiene las siguientes características:

- Cumple con los requisitos del sistema.
- Es resistente a los cambios en el entorno de implementación.
- Es fácil de mantener en relación con otros posibles modelos de objetos y con la implementación del sistema.
- Está claro cómo implementarlo.
- No incluye información que está mejor documentada en el código del programa.
- Se adapta fácilmente a los cambios en los requisitos [CITATION Pér12 \l 3082].

Diagramas de clases del diseño:

A continuación se muestran el Diagrama de Clases del Diseño correspondiente al caso de uso más relevante del módulo. Además, se logra representar la parte estática del sistema, así como sus clases y las relaciones entre estas.

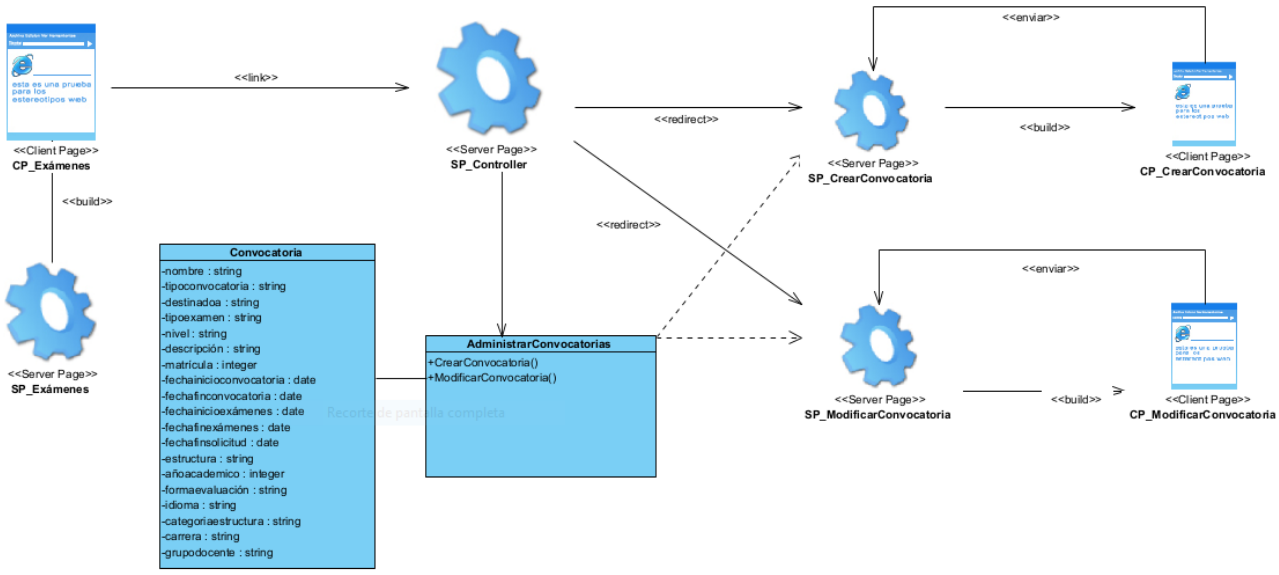

Figura 6: Diagrama de clases del diseño del caso de uso Administrar Convocatorias (Fuente: Elaboración Propia).

◦ **II.3 Diseño e implementación del almacenamiento, procesamiento y transmisión de los datos en el módulo de idioma**

En el presente epígrafe se hace énfasis en la presentación del diseño de los mecanismos para el almacenamiento, procesamiento y transmisión de los datos. Para ello se modela la base de datos del sistema a través de la confección del diagrama Modelo Entidad Relación.

Modelos de datos:

El modelo de datos es el modelado de la descripción de los datos, la semántica de los datos y las restricciones de coherencia de los datos. Proporciona las herramientas conceptuales para describir el diseño de una base de datos en cada nivel de abstracción de datos [CITATION Cap17 \l 3082]. Por lo tanto, se utilizan los siguientes cuatro modelos de datos para comprender la estructura de la base de datos:

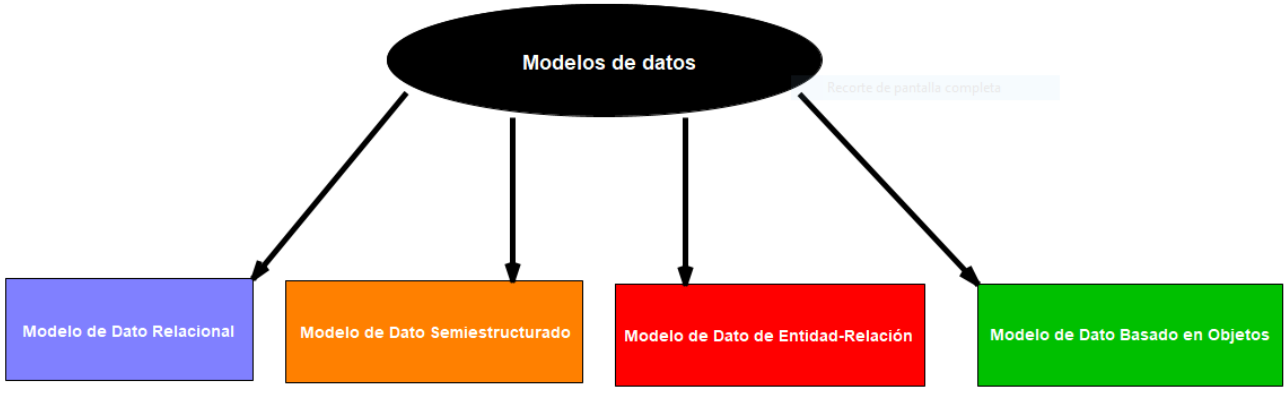

Figura 7: Principales modelos de datos (Fuente: Elaboración Propia).

1) **Modelo de Dato Relacional:** este tipo de modelo diseña los datos en forma de filas y columnas dentro de una tabla. En consecuencia, un modelo relacional usa tablas para representar datos y relaciones intermedias. Las tablas también se denominan relaciones. Este modelo fue descrito inicialmente por Edgar F. Codd, en 1969. El modelo de datos relacionales es el modelo más utilizado principalmente por aplicaciones comerciales de procesamiento de datos.

2) **Modelo de Dato Semiestructurado:** este tipo de modelo de datos es diferente de los otros tres modelos de datos (explicados anteriormente). Permite las especificaciones de datos en lugares donde los elementos de datos individuales del mismo tipo pueden tener diferentes conjuntos de atributos. El lenguaje de marcado extensible, también conocido como XML, se usa ampliamente para representar los datos semiestructurados. Si bien XML fue inicialmente diseñado para incluir la información de marcado al documento de texto, gana importancia debido a su aplicación en el intercambio de datos.

3) **Modelo de Dato de Entidad-Relación:** un modelo ER es la representación lógica de los datos como objetos y las relaciones entre ellos. Estos objetos se conocen como entidades, y la relación es una asociación entre estas entidades. Este modelo fue diseñado por Peter Chen y publicado en artículos de 1976. Fue ampliamente utilizado en el diseño de bases de datos. Un conjunto de atributos describen las entidades. Por ejemplo, nombre_estudiante, id_estudiante describe la entidad ‘estudiante’. Un conjunto del mismo tipo de entidades se

conoce como ‘Conjunto de entidades’, y el conjunto del mismo tipo de relaciones se conoce como ‘Conjunto de relaciones’.

4) **Modelo de Dato Basado en Objetos:** una extensión del modelo ER con nociones de funciones, encapsulación e identidad de objetos también. Este modelo admite un sistema de tipo enriquecido que incluye tipos estructurados y de colección. Así, en la década de 1980, se desarrollaron varios sistemas de bases de datos siguiendo el enfoque orientado a objetos. Aquí, los objetos no son más que los datos que llevan sus propiedades [CITATION Cap17 \l 3082].

Para representar el sistema se hará uso del Modelo Entidad Relación, ya que este permite identificar los objetos de datos y sus relaciones mediante una notación gráfica. A continuación se presentan el modelo de datos físicos con las principales entidades del sistema presentes en los módulos:

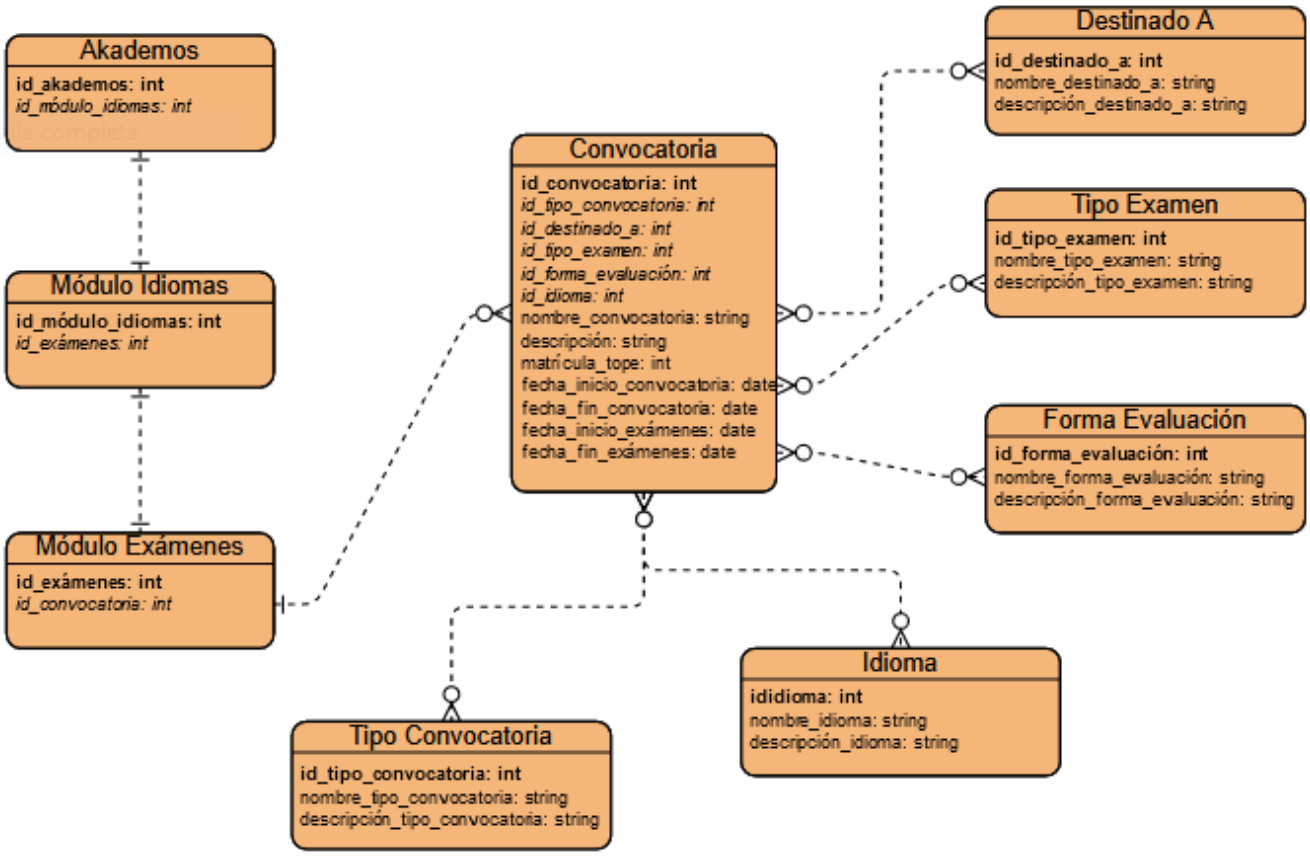

Figura 8: Modelo de datos del módulo Idiomas (Fuente: Elaboración Propia).

- **Conclusiones del capítulo**

En este capítulo se reflejaron los principales conceptos asociados al modelo de dominio. Se definieron los requisitos tanto funcionales como no funcionales que sustentan la propuesta de solución planteada. Durante el flujo de trabajo de Análisis y Diseño propuesto por la metodología de desarrollo AUP-UCI, se analizaron y realizaron las HU como principal artefacto, obteniendo como resultado los diagramas correspondientes al modelo del análisis y del diseño.

La definición de la arquitectura del sistema mediante el patrón arquitectónico MVC permitió delimitar la vista, en la cual se construye las interfaces gráficas de usuarios, la lógica del negocio para la definición de los controladores que gestionan las operaciones de los componentes, y el modelo para la persistencia de la información. Se elaboró y describió el diagrama de clases y los modelos de datos con que contará la solución.

CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo se abordan las fases, implementación y pruebas, del ciclo de vida del software. En la fase de implementación se materializó el software mediante su codificación, mientras que en la de pruebas se realizaron las verificaciones necesarias para garantizar la calidad y funcionalidad de la herramienta.

III.1 Verificación y validación del Módulo de Idiomas

En sentido general, este epígrafe presenta el diseño de los mecanismos utilizados para la verificación y validación de la solución propuesta, su ejecución y los resultados obtenidos. Para ello se confeccionan los Diagramas de Componentes de algunos casos de uso y se elabora, además, el Diagrama de Despliegue correspondiente al sistema.

Diagrama de Componentes:

Los diagramas de componentes constituyen en esta disciplina el modelo de implementación, al describir los componentes a construir, su organización y dependencias. Un componente es una parte física y reemplazable de un sistema que se conforma por un conjunto de interfaces y proporciona la realización de dicho conjunto. Estos diagramas se usan para modelar los elementos físicos que pueden hallarse en un nodo, por lo que empaquetan elementos como clases, colaboraciones e interfaces [CITATION DeA19 \l 3082].

A continuación, se representan el Diagrama de Componentes del Caso de Uso correspondientes al presente trabajo, el cual es un diagrama tipo del Lenguaje Unificado Modelado. Este representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes.

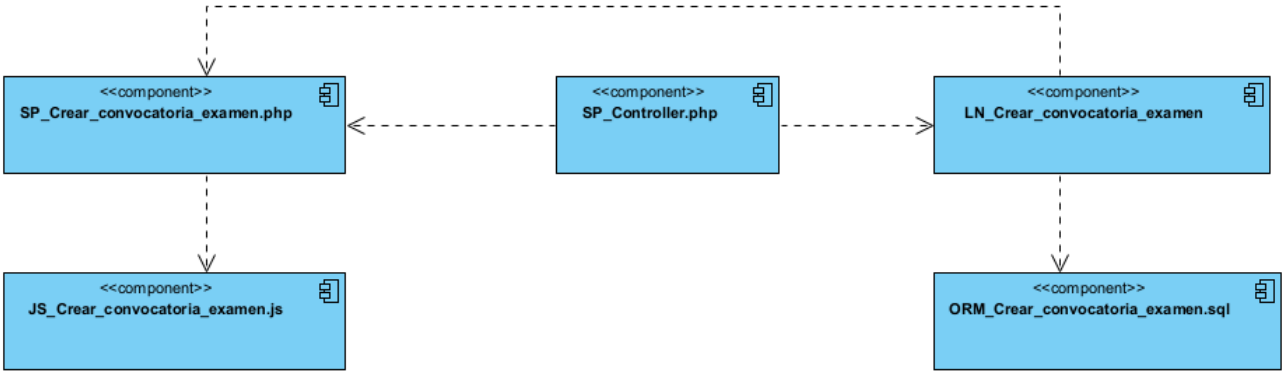

Figura 9: Diagrama de componentes del Caso de Uso Crear convocatoria de examen (Fuente: Elaboración Propia).

Con la realización de este diagrama de componentes se ha podido modelar algunas partes del sistema a implementar. Este diagrama permite, por medio de su estructura, analizar el software, así como sus relaciones entre componentes, así como importar o reutilizar componentes de otro sistema.

Diagrama de Despliegue:

El diagrama de despliegue es otro tipo de UML estructural que representa el hardware en el que se basa el software que se ejecutará para realizar alguna funcionalidad. Los diagramas de despliegue están hechos de diferentes formas UML. Tienen una amplia gama de aplicaciones, entre las que están que se pueden usar para verificar qué componente de hardware se ha desplegado. Además, se puede ilustrar el tiempo de ejecución y procesamiento para el hardware. También se puede proporcionar la vista de topología del sistema de hardware mediante el uso de diagramas de despliegue [CITATION Tit \l 3082].

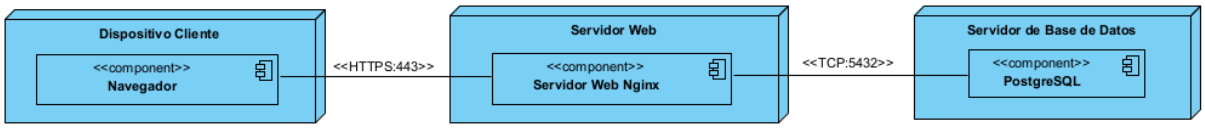

Figura 10: Diagrama de Despliegue (Fuente: Elaboración Propia).

Servidor de Bases de Datos: Dispositivo físico que ejecuta al Sistema Gestor de Base de Datos PostgreSQL que almacena la información persistente del Sistema de Gestión Académica.

Servidor Web: Hardware donde se ejecuta el servidor encargado de dar soporte a la aplicación web.

Dispositivo cliente: Representa cualquier dispositivo que permita al usuario acceder a la aplicación web a través de un navegador.

TCP: Protocolo empleado para establecer la conexión entre el servidor web y el sistema gestor de base de datos empleando el puerto 5432.

HTTPS: Protocolo empleado para establecer a través del puerto 443 la conexión segura entre el dispositivo de acceso cliente y el servidor web.

Con la realización de este diagrama de despliegue se puede visualizar los procesadores/ nodos/dispositivos de hardware del sistema a implementar, los enlaces de comunicación entre ellos y la colocación de los archivos de software en ese hardware. Además, al hacer uso de este se puede entender cómo el sistema se desplegará físicamente en el hardware.

Estándares de codificación:

Para el desarrollo del módulo se aplican los estándares de codificación establecidos por el Departamento de Informatización, con el propósito de estandarizar las nomenclaturas en la implementación de la aplicación web y obtener un producto estable.

Indentación, llaves de apertura y cierre y tamaño de las líneas:

Se debe emplear una indentación sin tabulaciones, con un equivalente a 4 espacios. El uso de las llaves “{}” será en una nueva línea. La longitud de las líneas de código es aproximadamente de 75 a 80 caracteres para mantener la legibilidad del código. En la Figura 14 se muestra un ejemplo de cómo debe quedar el formato del código de acuerdo con el estándar utilizado.

```
function listar_cursos()
{
    echo $this->template->render('cursos/curso/cursos_view');
}
```

Figura 11: Indentación, llaves de apertura y cierre y tamaño de las líneas (Fuente: Elaboración Propia).

Convención de nomenclatura:

Variables: Se rigen por la nomenclatura camelCase. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Se muestra en el método modificarTipologia() un ejemplo donde se emplean las variables:

```
$idTipo = $datos['id'];
$tipologia = $this->tipologia_lib->obtenerTipologiaDadoId($idTipo);
```

Figura 12: Variables (Fuente: Elaboración Propia).

camelCase: Es un estilo de escritura que se aplica a frases o palabras compuestas. El nombre se debe a que las mayúsculas a lo largo de una palabra en camelCase se asemejan a las jorobas de un camello. El término, case se traduce como “caja tipográfica”, que a su vez implica si una letra es mayúscula o minúscula.

Clases: Siempre comienzan con mayúscula, en caso de nombre compuesto las palabras se separan con guion bajo “_” y el resto en minúscula. Se muestra un ejemplo de la nomenclatura de una clase simple y una compuesta:

```
class Grupo_docente extends MY_Controller
    class Tipologia extends MY_Controller {
```

Figura 13: Clases (Fuente: Elaboración Propia).

Funciones: Se rigen por la nomenclatura camelCase. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa. Se muestra un ejemplo de la función (), donde se emplea la nomenclatura funciones:

```
public function obtenerCursoDadoId($id)
{...4 lines }
```

Figura 14: Funciones (Fuente: Elaboración Propia).

Ficheros: Todo siempre en minúscula y en caso de nombres compuestos se usa el guion bajo, seguido del sufijo definido para cada tipo de fichero.

➢ **Vistas:** Intuitivo y relacionado con el formulario y/o vista que representa. Sufijo “_view”.

Ejemplo: detalles_atributo_view.php

➢ **Modelos:** Mismo nombre de la tabla de la base de datos para la cual se crea. Sufijo “_mdl”.

Ejemplo: tb_ddato_almacenado_mdl.php

➢ **Librerías:** Mismo nombre de la clase que representa. Sufijo “_lib”.

Ejemplo: entidad_atributo_valor_lib

➢ **Controladoras:** Mismo nombre de la clase que representa.

Ejemplo: Categoria_entidad.

Estructuras de control:

Las estructuras de control incluyen if, for, foreach, while, switch. Entre las estructuras de control y los paréntesis debe de existir un espacio. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Con esto se aumenta la legibilidad del código y se disminuye la probabilidad de errores lógicos.

Documentación:

Todos los archivos deben de tener la documentación asociada al mismo. Se debe de cumplir con el siguiente bloque al principio de cada clase, como se muestra en la clase Curso_lib:

```php
<?php
/**
*@category Libreria
*@package curso_lib.php
*@author Pedro Ivan Fernandez Gonzalez <pifernandez@estudiantes.uci.cu>
*/
class Curso_lib {
```

Figura 15: Documentación (Fuente: Elaboración Propia).

Buenas prácticas:

Valores booleanos y nulos: los valores booleanos y nulos siempre se escriben con mayúscula, para facilitar la legibilidad del código. Se debe usar un salto de línea antes de las estructuras de control y definición de las funciones.

Etiquetas de apertura y cierre de PHP: el código PHP se debe escribir siempre utilizando las etiquetas **<?php** y **?>** y en ningún caso la versión corta **<?** y **?>**

Uso de comillas: se pueden utilizar tanto las comillas simples ('cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las cadenas de texto. También se recomienda el empleo de comillas dobles cuando el texto puede incluir alguna comilla simple.
Utilizar nombres que revelan intenciones: el nombre de una variable, función o clase debe responder a una serie de cuestiones básicas. Debe indicar por qué existe, qué hace y cómo se emplea. Si un nombre requiere un comentario, significa que no revela su contenido.

III.2 Proceso de pruebas al Módulo de Idiomas en el Sistema de Gestión Académica

Las pruebas son actividades en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos específicos, en orden de observar sus resultados, procesarlos y eventualmente evaluarlos en función de éxito o fracaso del mismo. Mediante esta filosofía se reduce el número de errores no detectados, así como el tiempo entre la introducción de este en el sistema y su detección [CITATION Tor14 \l 3082].

Con la metodología propuesta podemos dividir las pruebas en dos grupos sencillos: Pruebas unitarias y pruebas de aceptación. El objetivo de estas pruebas es que el cliente conozca cuando una HU está lista y bien implementada.

III.2.1 Pruebas Unitarias

Ejecutadas por los mismos desarrolladores, las pruebas unitarias son, en esencia, herramientas utilizadas para comprobar manual o automatizadamente que el código correspondiente a un módulo determinado de un sistema funcione de acuerdo a los requisitos del sistema. Estas pruebas presentan una ventaja única, pues permiten seguir trabajando sobre la base de mejorar el error y así reducir los efectos secundarios de estos mismos [CITATION Rod12 \l 3082].

Pruebas de Caja negra

Las pruebas de caja negra son aquellas que se llevan a cabo a la interfaz del software y permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales del programa.

Su objetivo es realizar pruebas al sistema para revelar un incompleto o incorrecto funcionamiento de este, así como errores que lance la interfaz, o problemas con el rendimiento. Los casos de prueba de Caja Negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.

- Se produce una salida correcta.
- La integridad de la información eterna se mantiene.

Las categorías de los errores encontrados en las pruebas de Caja Negra son:

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

Pruebas de Caja Blanca

Estas pruebas se realizan sobre las funciones internas de un módulo en concreto. El objetivo de la técnica es diseñar casos de prueba para que ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones, tanto en su vertiente verdadera como falsa.

Entre algunas de las técnicas de Caja Blanca podemos citar:

Prueba de Condición: es un método de diseño de casos de prueba que ejercita las condiciones lógicas obtenidas en el módulo de un programa.

Prueba de Flujo de Datos: se seleccionan caminos de prueba de un programa de acuerdo con la ubicación de las definiciones y los usos de las variables del programa.

Prueba de Bucles: es una prueba que se centra en la validez de las construcciones de bucles.

Prueba del Camino Básico: “esta técnica permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener dicho conjunto de caminos independientes se construye el grafo de flujo asociado y se calcula su complejidad ciclomática [CITATION Vie19 \l 3082].”

Los pasos que se siguen al aplicar esta técnica son los siguientes:

- A partir del diseño o del código fuente, se dibuja un grafo de flujo asociado.
- Se calcula la complejidad ciclomática asociada.
- Se determina un conjunto básico de caminos independientes.
- Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

Un camino es aquel para el cual puede efectuarse una traza a través del código desde el comienzo del programa hasta el final del mismo. Hay que tener en cuenta que dos caminos son diferentes si se ejecutan distintas sentencias o las mismas, pero en diferente orden. Para construir el grafo se debe tener en cuenta la notación para las instrucciones.

Caso de Prueba Caja Negra: Técnica de la Partición de Equivalencia.

Escenario: “Crear convocatoria”

Condiciones de ejecución: El usuario debe haber sido autenticado en el sistema. Solo tiene acceso a realizar esta acción el administrador.

Flujo Central: Módulo idioma/Convocatoria/Área de iconos flotantes “crear”.

Tabla 5: Descripción del caso de prueba Crear Convocatoria (Fuente: Elaboración Propia).

Clases válidas	Resultado esperado	Resultado de la prueba	Observaciones
Introducción correcta de los datos: “Nombre de la convocatoria: Examen noviembre 2022”	1)- Después de la introducción de todos los datos al formulario, si el proceso ha sido correcto, en la base	Satisfactorio.	

“Matrícula: 300 Idioma: Inglés” “Tipo de convocatoria: abierta” “Fecha de inicio: 3-11-22” “Fecha de fin: 6-11-22” “Destinado a: estudiantes” “Tipo de examen: colocación” “Forma de examen: MES 2018”	de datos se registra una nueva convocatoria de exámenes. 2)- Se muestra un mensaje indicando que el proceso ha concluido satisfactoriamente.		
Claves inválidas	Resultado esperado	Resultado de la prueba	Observaciones
Introducción incorrecta de los datos: “Nombre de la convocatoria: Examen noviembre 2022” “Matrícula: 300 Idioma: Inglés” “Tipo de convocatoria: abierta” “Fecha de inicio: 6-11-22” “Fecha de fin: 4-11-22”	1)- Se muestra un mensaje indicando que los datos entrados en el formulario no son correctos. 2)- La convocatoria a examen no se registra en el sistema.	Satisfactorio	

“Destinado a: estudiantes” “Tipo de examen: colocación” “Forma de examen: MES 2018”			
Introducción de convocatoria ya existente	1)- Se muestra un mensaje indicando que la convocatoria ya ha sido registrada. 2)- La convocatoria no se registra en la base de datos	Satisfactorio	
Introducción de Campos obligatorios. “Nombre de la convocatoria: Examen noviembre 2022” “Matrícula: 300 Idioma: Inglés” “Tipo de convocatoria: * ” “Fecha de inicio: 3-11-22” “Fecha de fin: 6-11-22” “Destinado a: * ” “Tipo de examen: * ”	1)- Se muestra un mensaje indicando que no se han introducido alguno de los campos obligatorios. 2)- La convocatoria no se registra en la base de datos.	Satisfactorio	

"Forma de examen: MES 2018"			
Evaluación de la prueba	Satisfactoria		

A continuación se enumeran las sentencias de código para la prueba de Camino Básico sobre la función verificarExiste():

```
public function verificarExiste($nombre_convocatoria, $id_convocatoria = '')
{

    $convocatoria = $this->_ci->tb_dconvocatoria_mdl->obtenerDadoAtributos(array('nombre_convocatoria' => $nombre_convocatoria));   1

    if (count($convocatoria) > 0 && empty($id_convocatoria))                                                                       2
        return true;                                                                                                              3
    if (count($convocatoria) > 0 && !empty($id_convocatoria) && $convocatoria[0]->id_convocatoria != $id_convocatoria)             4
        return true;                                                                                                              5


    return false;                                                                                                                 6
}
```

Figura 16: Representación de la función verificarExiste()(Fuente: Elaboración Propia).

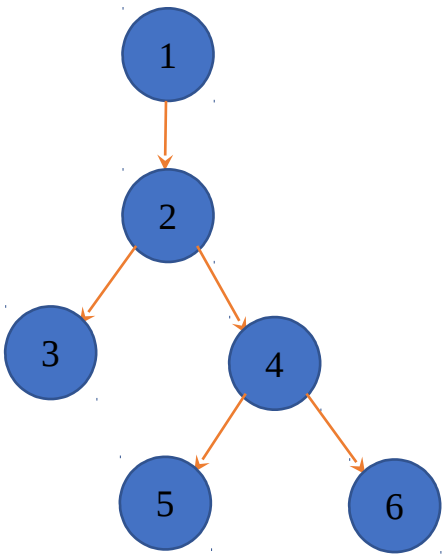

Figura 17: Notación de Grafo de Flujo de la función verificarExiste()(Fuente: Elaboración Propia).

Fórmulas para calcular complejidad ciclomática:

V (G) = (N – A) + 2

V (G) = (6 - 5) + 2

V (G) = 3

Siendo “A” la cantidad total de aristas y “N” la cantidad total de nodos. Se puede usar también:

V (G) = P + 1

V (G) = 2 + 1

V (G) = 3

Siendo “P” la cantidad total de nodos predicados (son los nodos de los cuales parten dos o más aristas).

V (G) = R

V (G) = 3

Siendo “R” la cantidad total de regiones, para cada fórmula “V (G)” representa el valor del cálculo.

El cálculo efectuado mediante las tres fórmulas ha dado el mismo valor, por lo que se puede plantear que la complejidad ciclomática del código es de 3, lo que significa que existen a lo sumo tres posibles caminos por donde el flujo puede circular, este valor representa el límite mínimo del número total de casos de pruebas para el procedimiento tratado. Seguidamente, es necesario representar los caminos básicos por los que puede recorrer el flujo. En la siguiente tabla se muestran los caminos básicos.

Tabla 6: Caminos básicos (Fuente: Elaboración Propia).

Número	Camino Básico
1	1,2,3
2	1,2,4,3
3	1,2,4,5

Después de haber extraído los caminos básicos del flujo, se procede a ejecutar los casos de pruebas para este procedimiento, se debe realizar al menos un caso de prueba por cada camino básico.

Para realizar los casos de pruebas es necesario cumplir con las siguientes exigencias:

- Descripción: se hace la entrada de datos necesaria, validando que ningún parámetro obligatorio pase nulo al procedimiento o no se entre algún dato erróneo.
- Condición de ejecución: se especifica cada parámetro para que cumpla una condición deseada para ver el funcionamiento del procedimiento.
- Entrada: se muestran los parámetros que entran al procedimiento.
- Entrada: se muestran los parámetros que entran al procedimiento.
- Resultados Esperados: se expone el resultado que se espera que devuelva el procedimiento.

Caso de Prueba para el Camino Básico 1 [1, 2, 3]:

Descripción: los datos de entrada pertenecen a una convocatoria existente.

Condición de ejecución: $id_convocatoria = null.

Entrada: $nombre_convocatoria = “Convocatoria noviembre 2022”, $id_convocatoria = null.

Resultados esperados: se muestra un mensaje indicando que “La convocatoria ya existe”.

Caso de Prueba para el Camino Básico 2 [1, 2, 4, 3]:

Descripción: los datos de entrada pertenecen a una convocatoria existente donde $nombre_convocatoria se encuentra en la base de datos.

Condición de ejecución: $nombre_convocatoria = “Convocatoria noviembre 2022”.

Entrada: $nombre_convocatoria = “Convocatoria noviembre 2022”, $id_convocatoria = “031”.

Resultados esperados: se muestra un mensaje indicando que “La convocatoria ya existe”.

Caso de Prueba para el Camino Básico 3 [1, 2, 4, 5]:

Descripción: los datos de entrada no se encuentran en la base de datos.

Condición de ejecución: Los valores de entrada son válidos.

Entrada: $nombre_convocatoria = “Convocatoria enero 2023”, $id_convocatoria = “034”.

Resultados esperados: se muestra un mensaje indicando que “La convocatoria no existe”.

Conclusiones del capítulo

En el proceso de desarrollo de software, las pruebas desempeñan un papel primordial porque proporcionan al cliente conformidad y seguridad ante el producto final. En el capítulo se abordan los métodos de pruebas usados durante el desarrollo de la solución y se describieron algunas de las pruebas de aceptación a las que fue sometida la aplicación, con la finalidad de asegurar que esta cumpla con los requerimientos funcionales. En todos los casos, las mismas devolvieron resultados satisfactorios, asegurando la fiabilidad y eficiencia del módulo Personal y Secretaría.

CONCLUSIONES FINALES

La elaboración del marco teórico a través de la definición de conceptos fundamentales, así como el estudio de herramientas para la confección del Módulo Idioma, permitió establecer un punto de partida para el proceso de desarrollo de la propuesta de solución, logrando identificar posibles funcionalidades que pudieran ser mejoradas en una segunda versión.

Ahora se conocen las ventajas y desventajas del uso de herramientas libres como PHP y los marcos de trabajo CodeIgniter y JQuery, como gestor de bases de datos, PostGreSQL y como IDE PHP-Storm, demostrando la factibilidad de su uso para la implementación de la propuesta de solución.

El análisis de la modelación del Módulo Idioma permitió un mejor entendimiento de los procesos realizados durante la gestión de información asociada a los exámenes estandarizados de idiomas en la Universidad de las Ciencias Informáticas, facilitando en gran medida la implementación de un software funcional que gestionará estos procesos dentro del subsistema Gestión de Idioma.

La implementación del Módulo Idioma permitió resolver las carencias a las que se enfrentaba el personal del Centro de Idiomas de la Universidad de las Ciencias Informáticas, obteniendo así un sistema funcional capaz de gestionar los procesos de obtención y procesamiento de la información académica concerniente a los estudiantes que cursan estudios allí.

La realización de las pruebas unitarias al sistema permitió garantizar que los requerimientos fueron cumplidos y que el sistema es estable.

RECOMENDACIONES

Una vez vencidos los objetivos de esta investigación y tomando en consideración las experiencias obtenidas a lo largo de su desarrollo, se recomienda al Departamento de Informatización:

- Implementar las funcionalidades correspondientes a: reportes y gestión de estudiantes.
- Desarrollar una versión compatible con otros sistemas para otras instituciones.
- Confeccionar un manual de usuario para lograr una mejor comprensión a la hora de trabajar con el sistema.
- Continuar el estudio de los procesos del Centro de idiomas CENID en vista de añadir nuevas funcionalidades.

REFERENCIAS BIBLIOGRÁFICAS

7Speaking. 2022. Las formaciones lingüísticas. [En línea] 2022.

Aplicación web para la gestión de la información especializada en Geociencia. **De Arma Hernández, Arianna y Sablón Fernández, Luis E. 2019.** no 2, s.l. : Ciencia & Futuro, 2019, Vol. vol. 9.

Beltrán, Marcos. 2017. [En línea] 17 de 10 de 2017.

Benites Sandoval, Franklin David. 2019. *Análisis del proceso de administración de ventas en una empresa de productos veterinarios.* 2019.

Blanco Casado, Mario. 2021. *Prototipo de Laboratorio Informático Virtual con Thinclients basados en Raspberry Pi.* 2021.

Burgués, Julián Esteban Gracia. 2018. *Aprende a Modelar Aplicaciones con UML.* 2018.

Cabrera Mercado, Karina María, Barreto Gil, Edgardo y Torres Archibaldo, Jhonatan. 2021. *Desarrollo de página web y gestor de contenidos de la Institución Educativa Flowers Hill Bilingual School de San Andrés Islas.* 2021.

Capacho, José Rafael y Nieto Bernal, Wilson. 2017. *Diseño de bases de datos.* 2017.

Costa, Grisel Infante. 2007. *Akademos, un Sistema Automatizado para la Gestión Académica.* Universidad de las Ciencias Informáticas. 2007.

Cuenca, Joan y Verazzi, Laura. 2019. *Guía fundamental de la comunicación interna. .* s.l. : Editorial UOC, 2019.

Dini, Marco. 2010. *Competitividad, redes de empresas y cooperación empresarial. .* 2010.

DuránCarrillo, Jorge Enrique. 2022. *Gestión colaborativa del conocimiento en cuidado de huertas urbanas basado en sistemas iot.* 2022.

EdTick. 2022. EdTick. [En línea] 2022. https://www.edtick.com/es/glossary/sistema-de-gestion-de-cursos-cms.

El inglés y su importancia en la investigación científica: algunas reflexiones. **Puello, Miryam. 2013.** 1, Sincelejo, Colombia : s.n., 2013, Vol. 5.

Estupiñán, Mireya Cisneros. 2012. *Cómo elaborar trabajos de grado.* s.l. : Ecoe Ediciones, 2012.

Fossati, Matias. 2018. *Introducción a PHP y HTML.* 2018.

Galarza Arias, Juan Carlos. 2022. *Implementación de una aplicación web para el taller Carvy Soluciones Automotrices módulo: gestión de inventario.* 2022.

García Velásquez, Victoria De Lourdes. 2015. *Procedimiento, análisis y desarrollo de un sistema de control de versiones distribuidas, sobre las base de datos del Sistema PROMEINFO.* 2015.

García, Pablo Andrés. 2019. *Sistemas embebidos de tiempo real con aplicaciones en bioingeniería.* 2019.

González Briongos, Ismael David. 2017. *Fundamentos de la virtualización de entornos con VMware: creación de un laboratorio virtualizado doméstico.* 2017.

Guerrero, Lissy Rodríguez . 2018. Inglés en la universidad: replantean política de enseñanza. *Granma.* 31 de 5 de 2018.

Idiomas, Centro de. 2016. [En línea] 2016. https://cenid.uci.cu/es/el-centro-de-idioma.

—. 2016. Cursos. [En línea] 2016. https://cenid.uci.cu/es/cursos.

—. 2016. Laboratorios de autoacceso. [En línea] 2016. https://cenid.uci.cu/es/laboratorios-de-autoacceso.

—. 2016. Traducción. [En línea] 2016. https://cenid.uci.cu/es/traduccion.

Informáticas, Universidad de las Ciencias. 2020. Aspirantes. [En línea] 2020. https://www.uci.cu/estudios/aspirantes.

—. 2022. Postgrado. [En línea] 2022. https://www.uci.cu/estudios/posgrado.

Kroenke, David M. 2002. *Database processing: Fundamentals, design & implementation. .* s.l. : Pearson Educación. 2002.

Longo, Álvaro. 2016. *Desarrollo de un aplicativo web gestor de la comunicación, monitoreo y control de procesos de titulacion en la carrera ingenieria en sistemas computacionales de la universidad de guayaquil.* 2016.

Luján-Mora, Sergio. 2002. *Programación de aplicaciones web: historia, principios básicos y clientes web. Editorial Club Universitario.* 2002.

Maida, Esteban Gabriel y Pacienzia, Julián. 2015. *Metodologías de desarrollo de software.* 2015.

Maldonado, Rosa Isela Zarco. 2015. *Comparativa sintáctica entre los lenguajes de programación java y groovy.* 2015.

Martínez, Aurora. 2022. Concepto Definición. [En línea] 4 de 4 de 2022. https://conceptodefinicion.de/gestion/

Mathieu. 2022. Microlearning Blog. [En línea] 11 de 1 de 2022. https://www.edapp.com/blog/es/los-15-mejores-sistemas-de-gestion-de-cursos-en-2020.

Minchola Rodriguez, Estefany Natali. 2021. *Evaluación de la calidad del Sistema Integrado de Administración Financiera de la Municipalidad Distrital de Calamarca basado en ISO/IEC 25000 durante el año 2021.*

MIranda, Cañizares y Eugenio, Edel. 2016. *Sistema gestión de información para la oficina de atención a la población de la universidad de las ciencias informáticas.* 2016.

Morales Pérez, Juana Idania. 2008. *La evaluación como instrumento de mejora de la calidad del aprendizaje. Propuesta de intervención psicopedagógica para el aprendizaje del idioma inglés.* 2008.

Morales, Daynelis Brito, Borrell, Johan Bravo y Armas, Lijandy Jiménez. 2019. *Aplicación móvil para el análisis de la información captada en SIGEv3.0.* 2019.

Muñoz, Vicente Javier Eslava. 2013. *El nuevo PHP paso a paso.* 2013.

OfiELE. 2022. OfiELE. [En línea] 2022. https://www.ofiele.es/.

—. 2022. Software Gratis. [En línea] 2022. https://www.software-gratis.com/utilidades-ofiele.

Ortiz Chaviano, Yoelvis, Torres Sakipova, Dina Yaksilik y Pérez Alfonso, Damián. 2015. *Complemento para transformar un Árbol de Variantes en una red de Petri.* 2015.

Ortiz Ruiz, Yorka Tatiana. 2020. *Accesibilidad web: legislación y evaluación Chile-España.* 2020.

Pacheco Malagón, Adiel. 2016. *Extrusión de entidades 2D a lo largo de curvas abiertas en el visor 3D de la herramienta AsiXMec 1.0.* 2016.

Palma Pérez, Nurisel. 2019. *Solución informática para la selección de Apache 2 y Nginx durante la migración a código abierto.* 2019.

Parreño, Collado. 2020. *Desarrollo de una Aplicación para la Gestión de tribunales de TFG/TFM en la ETSINF.* 2020.

Pérez, Carlos A. Cánepa. 2012. *Creación e implementación del módulo para la generación del modelo del negocio en la herramienta Case StarUML usando tecnologías de libre disponibilidad.* 2012.

Picón, Oscar Miguel Rangel. 2016. *Sistema de información de proceso de captura de referencia de pacientes de la Secretaría de Salud del estado de Guanajuato.* 2016.

Quispe Mamani, Israel Guillermo. 2020. *Sistema de Geolocalización de vehículos recolectores de basura aplicando Internet de las Cosas.* 2020.

Ramos, Daniel. 2017. *Curso de Ingeniería de Software: 2ª Edición. IT Campus academy.* 2017.

Rengifo Briñez, Johanna Fernanda y Betancurt Pérez, Carlos Alberto. 2011. *Frameworks y herramientas para el desarrollo de aplicaciones orientadas a dispositivos moviles.* 2011.

Reynoso, Carlos Billy. 2004. *Introducción a la Arquitectura de Software.* 2004.

Rodríguez, Anahí Soledad y Soria, Valeria. 2012. *Clasificación de herramientas open source para pruebas de aplicaciones web.* 2012.

Rojas Mantilla, Dariel y Lazo Brito, Rey Manuel. 2015. *Sistema Informático para la Gestión de Currículum Vítae basados en modelos internacionales.* 2015.

Roman Arenaza, Roque Erickson. 2019. *Lenguajes de programación Javascript Java y Javascript. Características. Norma de escritura. Variables y operadores lógicos. Mensajes. Ejercicios. Estructuras condicionales. Funciones y objetos. Aplicaciones.* 2019.

Sanabria Bocanegra, Sebastián Camilo. 2020. *Prototipo software para la entrega de domicilios (Quick Deliver).* 2020.

Sánchez, Tamara Rodríguez. 2015. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2015.

Serraller, Fransisco Clavo. 2014. *El arte contemporáneo.* s.l. : Taurus, 2014.

Significados. 2022. Significados. [En línea] 2022. https://www.significados.com/gestion/.

Solis, Ivan Soria y Tinoco, Enrique E. Condor. 2015. *Compendio de Ingenieria de software. Iván Soria Solís.* 2015.

Tito García, Sandra Rubí. 2018. *UML Introducción al UML, modelando con UML, utilidad del UML, conceptos de USE CASE, objetos, clases y atributos, operaciones, Aplicaciones.* 2018.

Torres, Jose Luis y Jaramillo, Olga. 2014. *Diseño y análisis del puesto de trabajo: herramienta para la gestión del talento humano.* 2014.

Universidad, Sistema de Gestión de la Nueva. 2015. [En línea] 13 de 2 de 2015. https://www.buenastareas.com/ensayos/Manual-De-Usuario-Del-Sigenu/67890988.html.

University, Cambridge. 2022. English Language Assessment. [En línea] 2022. https://www.cambridgeenglish.org/es/exams-and-tests/cefr/

Vega León, Marilin. 2020. *Análisis de la implantación de la metodología SCRUM y la plataforma TFS en la gestión de un proyecto con integración continua en la empresa ANIMSA.* 2020.

Ventura Chero, Julio Alex y Huamán Valqui, José Feliciano. 2017. *Diseño e Implementaciçon de un Sistema Informático Móvil para Mejorar la Atención del Cliente en Restaurantes.* 2017.

Viera Ipaneque, Nazaret Tayler Aldair. 2019. *Implementación de un generador de casos de prueba para procedimientos en lenguaje Java.* 2019.

ANEXOS