



Universidad de las Ciencias Informáticas

Módulo de administración para el “Sistema informático de apoyo al aprendizaje de la asignatura Teleinformática”

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

Autor: Melissa Mujica Suarez.

Tutor: Dargel Veloz Morales.

Co-tutor: Osmel Mojena Dubet.

La Habana junio 2019

“Año 61 de la Revolución”

Declaración de autoría

Declaro por este medio que yo **Melissa Mujica Suárez**, con carné de identidad **96092709696** soy el autor principal del trabajo titulado “Módulo de administración para el Sistema informático de apoyo al aprendizaje de la asignatura Teleinformática” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de dicho trabajo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

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

Firma del autor

Melissa Mujica Suárez

Firma de tutor

Ing. Dargel Veloz Morales.

Firma de co-tutor

Ing. Osmel Mojena Dubet.

Dedicatoria

Dedico este logro a mis padres Aymé y Raúl, y a mi abuelita bella Margot que han hecho hasta lo imposible por que pueda hoy hacerme ingeniera.

Agradecimientos

A mi mamá Aymé y mi abuela Margot por ser mis compañeras de camino desde siempre y confiar en que este día llegaría.

A mi papá Raúl quien me decía que la universidad es una carrera de resistencia, quien hizo de todo para que yo cumpliera mis metas y hoy no puede estar aquí.

A mi hermano, Marlon que va a comenzar esta carrera de resistencia, espero llegue a la meta como su hermanita linda.

A mi abuelo Raúl que yo sé que está conmigo siempre y ha visto en lo que me he convertido hoy, aunque nos dejara desde hace mucho tiempo.

A mi querido compañero de universidad, de 5 años de paciencia, Carlos Manuel, por soportarme mis momentos de matraquilla; que sería de tí sin mí.

A toda familia que está ahí cuando se necesita Adria, Gloria, Esther.

A mis compañeros de grupo, amigos para toda la vida, a los que no están, a los que no resistieron, a los que se fueron, Yenisley, Duriet, mis compañeras de estudio estos años, Rafa, Grisel, Karla, Alejandro ese niño que se hizo conmigo los 5 años todos los trabajos de la carrera que se sentó 4 años en la misma mesa, en fin, son muchos.

A mi tutor Dargel y Co-tutor Osmel, por ayudarme, aconsejarme, y estar hoy aquí y ser parte de esto. A mi oponente Julio, por corregir documentos, muchas gracias a todos los profesores que he tenido en la carrera, Gracias universidad, porque aquí conocí gente muy importante, sobre todo aprendí mucho de la vida.

*Agradezco a todos los que de una forma u otra pusieron un granito de arena en este largo camino que hoy llega a su fin. **Muchas gracias.***

Resumen

Las tecnologías de la educación suponen una oportunidad para facilitar el acceso a la educación. Inmerso en el desarrollo de estas tecnologías, no se identifica como es debido el uso de los medios tecnológicos en nuestras aulas para mejorar la calidad de las clases e incluso facilitar el acceso a la información en tiempo real. La aplicación SAATI es un Sistema de Apoyo al Aprendizaje de la asignatura Teleinformática el cual pretende motivar el estudio entre los estudiantes de esta generación y aprovechar las variantes competitivas y tecnológicas existentes. SAATI no cuenta con un mecanismo que permita a los profesores y administradores gestionar la información necesaria, no realiza un control de acceso de los usuarios en el sistema lo que provoca que estos no tengan definidas las acciones que puedan realizar y con estos conjuntos de permisos definidos aportaría parte de la seguridad y fiabilidad de la información necesaria. Esta investigación tiene como objetivo desarrollar un módulo de administración para SAATI que garantice un entorno dinámico y permita la adecuada gestión de contenidos en correspondencia con los usuarios. Definidas las herramientas informáticas con características similares a la solución propuesta, los métodos y tecnologías más factibles a utilizar para lograr un correcto diseño del módulo a través del análisis de las metodologías existentes, obtuvo como resultado el módulo de administración para SAATI que permitirá a un profesor gestionar y administrar los contenidos en el sistema y servirá de apoyo para el aprendizaje de la asignatura de teleinformática.

Palabras claves: administración, control de acceso, educación, permisos.

Índice General

Introducción	10
Capítulo 1: Fundamentación teórica de la investigación	14
1.1.1. Administración en WebCT	14
1.1.2. Administración en Claroline	14
1.1.3. Administración en Moodle.....	15
1.1.4. Administración en Duolingo	16
1.1.5. Administración en ClassDojo	17
1.1.6. Administración en ZERA.....	17
1.1.7. Definición de parámetros de selección	18
1.1.8. Resultados del análisis de las soluciones existentes	18
1.2. Tecnologías y herramientas a utilizar	19
1.2.1. Lenguaje de modelado	20
1.2.2. Herramienta CASE	20
1.2.3. Lenguaje de programación	21
1.2.4. Entorno de desarrollo integrado.....	23
1.2.5. Servidor web Apache.....	24
1.2.7. Marcos de trabajo	25
1.3. Metodología de desarrollo	25
Conclusiones del Capítulo	26
Capítulo 2: Características y diseño del módulo.....	27
2.1. Análisis de la propuesta de solución	27
2.2. Modelo de dominio.....	28
2.3. Requisitos funcionales (RF)	30
2.4. Requisitos no funcionales (RNF).....	31
2.5. Historias de Usuario (HU)	31
2.6. Arquitectura de software	33
2.7. Patrones de diseño	34
2.7.1. Patrones GRASP	35
2.7.2. Patrones GoF.....	36
2.8. Diagrama de clases del diseño	36
2.9. Modelo de datos.....	38
2.10. Modelo de despliegue	38

Conclusiones del Capítulo	39
Capítulo 3: Implementación y pruebas al módulo	40
3.1 Diagrama de componente	40
3.2 Estándares de codificación.....	40
3.3 Pruebas funcionales.....	43
3.4. Pruebas de rendimiento	46
3.5. Pruebas de integración.....	48
3.6. Pruebas de aceptación.....	49
Resultados de las pruebas realizadas.....	49
Conclusiones generales.....	51
Referencia Bibliográficas	52
Anexos.....	56
Glosario de términos.....	79

Índice de ilustraciones

Ilustración 1: Modelo de dominio del sistema. (Elaboración propia)	28
Ilustración 2: Flujo de trabajo de la arquitectura. (Condri, 2012).....	33
Ilustración 3: Fragmento de la arquitectura(MTV) en el sistema. (Elaboracion propia).....	34
Ilustración 4: Ejemplo de patrón Experto. (Elaboración propia).....	35
Ilustración 5: Ejemplo de patrón Creador. (Elaboración propia.)	35
Ilustración 6: Ejemplo de patrón Controlador. (Elaboración propia).....	36
Ilustración 7: Ejemplo de patrón Decorador. (Elaboración propia).....	36
Ilustración 8: Diagrama de clases "Gestionar pregunta". (Elaboración propia).....	37
Ilustración 9: Modelo de base de dato del sistema. (Elaboración propia).....	38
Ilustración 10: Diagrama de despliegue. (Elaboración propia).....	39
Ilustración 11: Diagrama de componente. (Elaboración propia)	40
Ilustración 12: Resultados de las pruebas funcionales. (Elaboración propia).	46

Índice de tablas

Tabla 1: Requisitos funcionales. (Elaboración propia.).....	30
---	----

Tabla 2: Historia de usuario "Crear pregunta". (Elaboración propia.)	32
Tabla 3: Estándares de codificación. (Elaboración propia.).....	41
Tabla 4: Variables de entrada para los casos de prueba. (Elaboración propia).....	44
Tabla 5: Caso de prueba "Añadir Pregunta"(parte 1). (Elaboración propia).....	44
Tabla 6: Caso de prueba "Añadir Pregunta"(parte 2). (Elaboración propia).....	45
Tabla 7: Resultados de las pruebas de carga y estrés.....	48
Tabla 8: Historia de usuario "Crear roles del sistema". (Elaboración propia).....	56
Tabla 9: Historia de usuario "Listar roles del sistema". (Elaboración propia).....	56
Tabla 10: Historia de usuario "Actualizar roles de sistema". (Elaboración propia).....	57
Tabla 11: Historia de usuario "Eliminar roles del sistema". (Elaboración propia).....	57
Tabla 12: Historia de usuario "Crear permisos". (Elaboración propia).....	58
Tabla 13: Historia de usuario "Listar permisos". (Elaboración propia).....	58
Tabla 14: Historia de usuario "Actualizar permisos". (Elaboración propia).....	59
Tabla 15: Historia de usuario "Eliminar permisos". (Elaboración propia).....	59
Tabla 16: Historia de usuario "Adicionar usuarios". (Elaboración propia).....	60
Tabla 17: Historia de usuario "Listar usuarios". (Elaboración propia).....	60
Tabla 18: Historia de usuario "Actualizar usuarios". (Elaboración propia).....	61
Tabla 19: Historia de usuario "Eliminar usuarios". (Elaboración propia).....	61
Tabla 20: Historia de usuario "Crear mundo". (Elaboración propia).....	61
Tabla 21: Historia de usuario "Listar mundos". (Elaboración propia).....	62
Tabla 22: Historia de usuario "Actualizar mundo". (Elaboración propia).....	62
Tabla 23: Historia de usuario "Eliminar mundo". (Elaboración propia).....	63
Tabla 24: Historia de usuario "Crear nivel". (Elaboración propia).....	63
Tabla 25: Historia de usuario "Listar nivel". (Elaboración propia).....	64
Tabla 26: Historia de usuario "Actualizar nivel". (Elaboración propia).....	64
Tabla 27: Historia de usuario "Eliminar nivel". (Elaboración propia).....	65
Tabla 28: Historia de usuario "Listar opciones". (Elaboración propia).....	65
Tabla 29: Historia de usuario "Eliminar opciones". (Elaboración propia).....	66
Tabla 30: Historia de usuario "Crear estados". (Elaboración propia).....	66

Tabla 31: Historia de usuario "Listar estados". (Elaboración propia).	67
Tabla 32: Historia de usuario "Actualizar estados". (Elaboración propia).	67
Tabla 33: Historia de usuario "Eliminar estados". (Elaboración propia).	68
Tabla 34: Historia de usuario "Listar pregunta". (Elaboración propia).	68
Tabla 35: Historia de usuario "Actualizar preguntas". (Elaboración propia).	69
Tabla 36: Historia de usuario "Eliminar preguntas". (Elaboración propia).	69
Tabla 37: Historia de usuario "Crear notificaciones". (Elaboración propia).	70
Tabla 38: Historia de usuario "Listar notificaciones". (Elaboración propia).	70
Tabla 39: Historia de usuario "Actualizar notificaciones". (Elaboración propia).	71
Tabla 40: Historia de usuario "Eliminar notificaciones". (Elaboración propia).	71
Tabla 41: Historia de usuario "Crear grupo de estudiantes". (Elaboración propia).	72
Tabla 42: Historia de usuario "Listar grupos de estudiantes". (Elaboración propia).	72
Tabla 43: Historia de usuario "Actualizar grupos de estudiantes". (Elaboración propia).	73
Tabla 44: Historia de usuario "Eliminar grupos de estudiantes". (Elaboración propia).	73
Tabla 45: Historia de usuario "Listar acciones administrativas". (Elaboración propia).	74
Tabla 46: Historia de usuario "Actualizar cambios generales en el sistema". (Elaboración propia).	74
Tabla 47: Historia de usuario "Recuperar contraseña olvidada o caducada". (Elaboración propia).	74
Tabla 48: Historia de usuario "Crear comodines". (Elaboración propia).	75
Tabla 49: Historia de usuario "Listar comodines". (Elaboración propia).	75
Tabla 50: Historia de usuario "Eliminar comodines". (Elaboración propia).	76
Tabla 51: Historia de usuario "Actualizar comodines". (Elaboración propia).	76
Tabla 52: Historia de usuario "Crear opciones". (Elaboración propia).	76

Introducción

Las Tecnologías de la Información y las Comunicaciones (TIC) son el conjunto de herramientas aplicadas principalmente a la gestión de la información, que abarcan un abanico muy amplio de soluciones (Corrales, 2009). Las TIC pueden complementar, enriquecer y transformar la educación. En los últimos años han jugado un papel clave en el desarrollo de nuevas políticas y proyectos educativos, ya que han incidido en la forma de acceder al conocimiento, incorporar nuevos roles al profesor, nuevos contenidos, programas y formas de evaluar. Ellas han supuesto una oportunidad para facilitar el acceso a la educación, bien sea presencial o a distancia, pero al mismo tiempo supone un conjunto de desafíos. En este sentido, se han desarrollado servicios y plataformas de protocolo de Internet destinados a la educación, los cuales han permitido repensar los modelos de enseñanza y aprendizaje. Entre ellos podemos nombrar las plataformas de *e-learning* las cuales son aplicaciones web que integra un conjunto de herramientas (Significados, 2010).

En Cuba desde hace varios años diversas instituciones educativas comenzaron a utilizar plataformas (*e-Learning*), las cuales facilitan la enseñanza y el aprendizaje a través de *Internet/Intranet*, como apoyo a la educación tradicional y para impartir algunos cursos a distancia. También se desarrollaron herramientas para el uso del aprendizaje electrónico en los procesos docentes educativos (UIV(Internet), 2018).

Un ejemplo de esto es la Universidad de las Ciencias Informáticas (UCI), institución que intenta mantener el ritmo de los avances tecnológicos en la sociedad cubana. En su plan de estudio está el impartir la asignatura de Teleinformática en el 3er año de la carrera, la cual permite formar a los futuros ingenieros en un vasto campo sobre el que se basan las redes de computadoras en la actualidad, además de lograr un conocimiento de los estándares de los protocolos de interconexión, metodología de implementación de redes heterogéneas y las mejores prácticas para su diseño y planificación. Por ser puramente técnica cuesta más trabajo a los estudiantes poder acceder e incorporar ese conocimiento. Se torna difícil que los estudiantes puedan comprender los ejemplos prácticos con la manera tradicional que se utiliza en la universidad, carece de mecanismos propios que logren motivar a los estudiantes a aprender de manera autónoma. Por otra parte, los medios tecnológicos, dígase ordenadores personales o teléfonos inteligentes, no se utilizan como es debido en nuestras aulas para mejorar la calidad de las clases e incluso facilitar el acceso a información en tiempo real.

La Facultad 1 de la UCI se encuentra desarrollando SAATI Sistema informático de apoyo al aprendizaje de la asignatura Teleinformática, el cual tiene entre sus objetivos lograr motivar al estudio entre los

estudiantes de esta generación y aprovechar las variantes competitivas existentes que propician un mayor rendimiento evaluativo. En la misma interactúan usuarios de manera simultánea, los cuales generan tráfico constante, por lo cual se hace necesario un mecanismo para controlar el acceso de cada uno, así como la organización del contenido que va a ser consumido de manera tal que la aplicación garantice seguridad y confiabilidad de la información y una mejor retroalimentación entre estudiantes y profesores.

SAATI no permite a los profesores y administradores gestionar la información y contenidos que necesita el sistema, por lo que no realiza el seguimiento ni gestión de los procesos que el sistema requiere; no permite la gestión de los contenidos puestos a disposición para los estudiantes, lo que provoca que los contenidos sean mostrados de forma estática, proceso que debe realizarse de forma dinámica y en constante evolución; no realiza una correcta gestión de roles y permisos que permita el control de acceso al sistema, provoca que los usuarios no tengan definidas las acciones que puedan realizar y con estos conjuntos de permisos definidos aportaría seguridad y fiabilidad de la información, accediendo así a la información que necesita y correcta funcionalidad de acciones en el sistema.

Debido a la necesidad de dar solución a la situación planteada se identifica como **problema de investigación**: ¿Cómo garantizar la seguridad y la gestión de los contenidos para el Sistema informático de apoyo al aprendizaje de la asignatura Teleinformática (SAATI)?

Ante la anterior situación investigativa se plantea como **Objeto de estudio**: la gestión de información y control de acceso en sistemas que emplean gamificación, enmarcado en el **campo de acción** gestión de la contenidos y control de acceso de SAATI.

Para dar solución al problema de investigación planteado se define como **objetivo general** desarrollar un módulo de administración para SAATI que garantice un entorno dinámico y permita la adecuada gestión de contenidos en correspondencia con los usuarios.

En ese mismo sentido los **Objetivos específicos**:

- Identificar el marco teórico conceptual con respecto a los componentes empleados en la aplicación, las herramientas y tecnologías asociadas.
- Realizar el análisis y diseño de las funcionalidades del módulo de gestión y seguridad de la información para SAATI.

- Implementar las funcionalidades del módulo de gestión y seguridad de la información para SAATI de acuerdo con la estructura de diseño definida.
- Probar la solución propuesta para comprobar el cumplimiento de los requerimientos de la misma.

Luego de haber realizado y desarrollado el diseño teórico, se formulan las siguientes **preguntas científicas**:

1. ¿Cómo realizar un análisis y diseño para un módulo de administración para SAATI para lograr un buen resultado y aceptación?
2. ¿Cómo gestionar la información y el control de acceso en la aplicación a desarrollar?
3. ¿Cómo llevar a cabo el proceso de desarrollo del módulo de administración para SAATI?
4. ¿Qué resultados se obtendrán de validar el módulo de administración para SAATI?

Se plantean las siguientes **Tareas de la investigación** para el cumplimiento de los objetivos:

1. Identificación del marco teórico conceptual con respecto a la aplicación.
2. Realización del estudio del arte sobre el desarrollo de sistemas para la creación del módulo de administración.
3. Descripción de las tecnologías de desarrollo a utilizar para la realización del módulo de administración.
4. Levantamiento de la información requerida para la comprensión y selección de patrones de diseño y arquitectura.
5. Implementación de la solución propuesta.
6. Diseño y aplicación de pruebas necesarias para obtener resultado completo y satisfactorio.

En este trabajo se emplearon como **Método de investigación** los métodos empíricos y teórico.

Para lo cual los **métodos teóricos** son:

- Histórico-Lógico: que es utilizado en el análisis de los sistemas homólogos, de manera que permita buscar elementos que los caractericen y aspectos para fundamentar la propuesta de solución a la problemática planteada.

- Analítico-Sintético: con el fin de descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución propuesta.

Luego los **métodos empíricos** utilizados son:

- Encuestas: realizadas para validar la aceptación del sistema.

El presente trabajo de diploma está compuesto por el análisis y diseño metodológico de la investigación, y tres capítulos que se describe brevemente de que abordan a continuación:

Capítulo 1: Fundamentación teórica de la investigación: Se hace referencia a los elementos teóricos que constituyen la base de la investigación realizada. Se describen los conceptos esenciales para comprender el problema, así como las tecnologías y herramientas a utilizar en el diseño e implementación de la aplicación.

Capítulo 2: Características y diseño del módulo: Se describe la propuesta de solución a desarrollar para resolver el problema planteado y se especifican los requisitos funcionales y no funcionales. Por otra parte, se describen las historias de usuario correspondientes a los requisitos funcionales, la arquitectura y los patrones de diseño que se utilizarán durante la implementación.

Capítulo 3: Implementación y pruebas al módulo: Se documentan los artefactos asociados a la implementación de la propuesta de solución y se realizan los casos de prueba para lograr el desarrollo de un software con la calidad requerida. Además, se establecen los estándares de codificación que se tuvieron en cuenta para el desarrollo del sistema y relaciones lógicas a través del diagrama de componentes.

Capítulo 1: Fundamentación teórica de la investigación

En este capítulo se realiza la fundamentación teórica de la investigación. Se realiza un estudio de las diferentes herramientas que son útiles para la implementación del módulo. Se selecciona la metodología a utilizar en el ciclo de vida del software. De igual manera se hace una revisión y elección de lenguajes de programación propuestos para el desarrollo del sistema y se analizan soluciones similares.

1.1. Análisis de administración de contenidos en sistemas gamificados

1.1.1. Administración en WebCT

WebCT, desarrollada para buscar una aptitud hacia la tecnología, un control y un nuevo estilo para una correcta enseñanza, constituye una herramienta eficaz, flexible, orientada a cursos y muy utilizada por sus probados resultados en la educación a distancia (ED). Su popularidad se basa en (Rodríguez, 2004):

- ✓ La facilidad de su uso debido a la sencillez de su interfaz no permite al usuario equivocaciones y una buena navegación (flexibilidad).
- ✓ El diseño uniforme de la interfaz dentro del sistema.
- ✓ La interacción del alumno con el sistema.
- ✓ La orientación hacia el estudiante, se adapta perfectamente a sus necesidades y le posibilita descargar cursos, entre otras facilidades.
- ✓ WebCT prepara al profesor para que domine la plataforma.
- ✓ La interacción en el aula virtual entre profesor y alumno.

Entre sus aspectos negativos, se encuentra el alto precio de su uso. La mayoría de los países que lo emplean son desarrollados y muchas de las universidades que lo utilizan son privadas. Otra de sus deficiencias para las necesidades propias del sector de la salud en el país es su generalidad, que estandariza la metodología y el modelo de los cursos y obliga, debido a intereses comerciales, a que la tecnología sea adaptable a la enseñanza, tanto del idioma en Japón como de la medicina en Puerto Rico (Rodríguez, 2004).

1.1.2. Administración en Claroline

Es una plataforma de aprendizaje (o LMS: *Learning Management System*) y *groupware* de código abierto (GPL). Permite a cientos de instituciones de todo el mundo (universidades, colegios, asociaciones, empresas ...) de crear y administrar cursos y espacios de colaboración en línea. Permite su descarga e

instalación libre y gratuita bajo cualquier sistema operativo y en variadas lenguas a través del uso de cualquier navegador web, siendo compatible con LINUX, MAC, Windows, No tiene límites de usuarios, basado en herramientas y lenguajes libres como PHP y MySQL e integra estándares actuales como SCORM (Modelo Referencial para Objetos de Contenidos Compartidos) e IMS (Gestión de Sistemas de Información)/QTI (Especificación de Interoperabilidad de Preguntas y Pruebas de IMS) para intercambiar contenidos. La filosofía de *Claroline* se basa en lograr la autonomía pedagógica del docente, por tanto, su éxito depende del buen uso pedagógico que se le dé a esta herramienta (Naranjo, 2014).

Permite la publicación de documentos en diferentes formatos (.doc, .odt, ppt, pdf), administrar y modelar foros de discusión, administrar listados de enlaces de referencia, crear grupos de estudios, confeccionar ejercicios de práctica, estructurar y administrar una agenda de eventos, y tareas, gestionar el envío por parte de los estudiantes de documentos, tareas, trabajos, gestionar y almacenar los chat entre estudiantes y docente, y realizar por parte del docente un seguimiento completo y evaluación de los estudiantes, entre otras (Naranjo, 2014).

Sus principales desventajas están relacionadas con los pocos módulos y *plugins* con que cuenta para descargar, la lentitud de su chat y la dificultad a la hora de abrir sus archivos (Naranjo, 2014).

1.1.3. Administración en Moodle

Aula virtual por excelencia utilizada en múltiples ámbitos. Es un paquete de software para la creación de cursos y sitios Web basados en Internet para dar soporte a un marco de educación social constructivista.

Constituye un sistema de gestión de contenidos educativos, que fue creado como una herramienta de código abierto gratuito que presenta un amplio abanico de posibilidades, desde su utilización como repositorio de objetos de aprendizajes hasta la creación de un espacio virtual adecuado para el desarrollo de cursos a distancia, a través de la red con interactividad entre estudiantes y profesores, da privilegios al trabajo autónomo y colaborativo. Moodle es considerada hoy en día, una de las plataformas *e-learning* más utilizadas en las instituciones educativas para fomentar la interacción entre estudiantes y docentes (Naranjo, 2014).

Ofrece un conjunto de herramientas; foros, diálogos, cuestionarios, consultas, encuestas, tareas, chat, glosarios, lecciones entre otros. Permite a los docentes crear EVA en los que desarrollar cursos on-line o utilizarlos de apoyo como complemento a la enseñanza tradicional y proporciona al profesor algunas

herramientas estadísticas para llevar el registro y seguimiento de los estudiantes, así como el historial de cada estudiante en particular, (participación, mensajes enviados, entradas) (Naranjo, 2014)

Responde a los estándares internacionales SCORM por lo que permite exportar e importar contenidos de otras plataformas. Dispone de una excelente documentación de apoyo en línea y comunidades de usuarios a través del sitio <http://www.modlle.org> donde los usuarios de Moodle (administradores del sistema, profesores, investigadores, diseñadores entre otros) comparten información y resuelven dudas. Cada usuario puede seleccionar su zona horaria. Todas estas características planteadas, hacen de Moodle una plataforma única (Naranjo, 2014).

Sus limitaciones están dadas fundamentalmente por el servidor y el ancho de banda. Se plantea que, por cada 50 usuarios, debe disponerse de 1 Gb de memoria RAM. No dispone de algunas herramientas pedagógicas como crucigramas, juegos de roles y otros; muestra los mismos contenidos a todos los alumnos por tanto no permite el trabajo personalizado con los estudiantes de acuerdo a su desempeño y dificultades. También presenta desventajas asociadas a su seguridad, dependiendo de su alojamiento, las políticas de seguridad y la infraestructura tecnológica. Al ser programado en PHP requiere de cuidados especiales al ser configurada para muchos usuarios (Naranjo, 2014).

1.1.4. Administración en Duolingo

Es una herramienta de colaboración masiva online que combina un espacio de aprendizaje de idiomas gratis con una plataforma de traducción de texto. El servicio está diseñado para que los estudiantes puedan aprender una lengua, mientras ayuda a traducir páginas web y documentos. Los jóvenes principiantes comienzan con frases básicas y sencillas, mientras que los usuarios avanzados reciben textos más complejos.

Mientras los estudiantes aprenden un idioma, ganan puntos “sobre habilidades” cuando las lecciones se han completado o el contenido de la web está 100% traducido. Lecciones asociadas con una habilidad se completan sucesivamente cuando se han completado una serie de traducciones.

Cada curso en DuoLingo está conformado por módulos, los cuales están agrupados para formar habilidades. Permite realizar varios intentos a los contenidos de aprendizaje.

1.1.5. Administración en ClassDojo

ClassDojo es una plataforma de gestión del aula en la que participan tanto estudiantes como profesores y además de padres. Hoy en día es una de las herramientas de gamificación educativa que más éxito está teniendo, su objetivo es permitir al profesor administrar un aula escolar en base a puntuaciones sobre el comportamiento de los chavales, ofreciendo una interfaz abierta, muy atractiva y sencilla de usar y que, además, es personalizable a las necesidades de cada clase, materia y profesor. ClassDojo está centrada en un alumno que necesita más seguimiento paterno, en edades tempranas, hay comunicación entre profesores, padres y dirección pero el uso de los alumnos está centrado en añadir fotos y videos en sus carpetas, tiene bastantes problemas con el acceso, el seguimiento de los alumnos solo se puede ver semanalmente, no permite el cambio de contraseña, fallos con la visualización de los mensajes y el idioma en el que está definida es el inglés, por tanto si un padre no entiende no podrá interactuar con la aplicación (Valle, 2018).

Es ideal para evaluar de una forma objetiva el porcentaje dentro de los criterios de calificación referidos al comportamiento, actitud, o como queramos llamarlo. Es adaptable para cualquier plataforma, incluye el control de asistencia de los alumnos. Las familias se ponen en contacto con profesores, donde los padres también tienen un rol activo en la plataforma. El docente puede personalizar todos estos comportamientos como considere necesario.

1.1.6. Administración en ZERA

En la Universidad de las Ciencias Informáticas se desarrolló un LMS denominado Plataforma Educativa ZERA que puede ser utilizada en Cuba y en otros países (Manso Guerra, y otros, 2016). XAUCE ZERA 2.0 es una plataforma educativa innovadora, flexible, fácil de usar, interactiva y adaptable capaz de guiar el proceso de enseñanza-aprendizaje de cualquier nivel de enseñanza a partir de un conjunto poderoso de herramientas. Es accesible desde cualquier dispositivo móvil y soporta contenidos interactivos enriquecidos con elementos psicológicos y didácticos, así como recursos multimedia a partir de plantillas previamente definidas que se entremezclan con documentos, presentaciones y otros materiales junto a foros de discusión, cuestionarios y disímiles tipologías de ejercicios que pueden ser evaluados de forma automática, por coevaluación o directamente por el profesor. También da soporte a los Cursos Masivos Abiertos en Línea (MOOC) y asegura que no haya limitaciones lingüísticas para aprender en línea dando soporte a varios idiomas. Brinda la posibilidad de crear cursos agrupados por categorías (Xauce, 2016).

Entre las características generales y administrativas de esta plataforma se encuentra (Xauce, 2016):

- ✓ Interfaz moderna, fácil de usar.
- ✓ Actividades y herramientas colaborativas.
- ✓ Monitoreo del progreso.
- ✓ Evaluación de las actividades.
- ✓ Gestión conveniente de archivos.
- ✓ Editor de texto simple e intuitivo.
- ✓ Plantillas personalizadas para contenidos.
- ✓ Notificaciones en tiempo real.
- ✓ Editor de texto simple e intuitivo.
- ✓ Diseño personalizado del banner y la información del pie de página de la plataforma.
- ✓ Capacidad de ver contenido en español y varios idiomas.
- ✓ Capacidad multiorganizativa.
- ✓ Gestione permisos y roles de usuario.
- ✓ Autenticación segura.

1.1.7. Definición de parámetros de selección

Se identifican como elementos valorados para la correcta administración de SAATI los siguientes aspectos:

- ✓ Gestión del control de acceso del sistema.
- ✓ Gestión de elementos de gamificación empleados (progreso, premios, desafíos, entre otras).
- ✓ Administración y organización de los contenidos puestos a disposición.
- ✓ Gestión de roles y permisos.
- ✓ Rendimiento del sistema con respecto a la conectividad de los usuarios en la misma.

1.1.8. Resultados del análisis de las soluciones existentes

Las plataformas antes descritas son muy utilizadas en el mundo por universidades, y otras instituciones para la confección de entornos de aprendizaje que fomenten la enseñanza en las mismas, sin embargo, las funciones y características que algunas de estas plataformas brindan como la publicación de documentos en diferentes formatos, la administración y modelación de foros de discusión y el uso de

tecnología propietaria, no será utilizadas en nuestra aplicación ya que son más de las necesarias. De esta manera se dificulta el realizar la selección de una plataforma en específico y así dar respuesta al objetivo inicial planteado: desarrollar un módulo de administración para SAATI que facilite un entorno dinámico y permita la correcta administración para una adecuada visualización de contenidos en correspondencia con los usuarios.

Por lo tanto, el desarrollo de una plataforma propia, siguiendo los estándares y requisitos que requieren las mismas respecto a entornos de aprendizajes desarrollados anteriormente, el no comprometer a la institución con el alto costo de una licencia en caso de una plataforma privativa, es garantía de mantener una coherencia entre esta aplicación y el modelo educativo de la universidad que la desarrolla, otras de las razones son:

- ✓ Que responde a situaciones educativas y carencias concretas detectadas en la institución.
- ✓ Tener independencia total, se puede reajustar, y adoptar en cualquier momento que sea necesario.
- ✓ Investigar sobre el tema, en el proceso de desarrollo se crea así un personal experto.

Es válido destacar que la utilización de elementos de gamificación se evidencia, tal es el caso de utilizar los recursos de los juegos para trasladar estas experiencias a otros ámbitos. Aunque el desarrollo de un modelo propio de plataforma requiere de un esfuerzo significativo de creación, esta opción posibilitaría obtener un sistema completamente adaptado al entorno tecnológico y educacional en el país, con un bajo costo de creación, soporte y mantenimiento, acorde con los patrones pedagógicos actuales de Cuba, en busca de una formación integral del alumno, orientadora y personalizada en todos los sentidos. Permitiría a los docentes de la UCI identificarse en un escenario donde podrá, entre otras cosas, estudiar a la vez que se entretiene, responder a preguntas que luego podrá estudiar, evaluarse, y aprender con fluidez.

1.2. Tecnologías y herramientas a utilizar

Una herramienta es cualquier programa o instrucción que facilita una tarea. Las herramientas son aplicaciones, programas o meramente instrucciones que ofrecen la posibilidad de realizar determinadas funcionalidades con disímiles propósitos. De otro modo, las tecnologías son el conjunto de conocimientos técnicos, ordenados científicamente que permiten diseñar y crear bienes y servicios (Definición de Herramienta, 2016).

A continuación, se muestran todas las herramientas y tecnologías utilizadas en el desarrollo de la solución propuesta, estas fueron seleccionadas por políticas del Sistema informático de apoyo al aprendizaje de la asignatura Teleinformática al que pertenece el módulo que se pretende desarrollar. Se fundamentan también otras características de importancia.

1.2.1. Lenguaje de modelado

El modelado constituye una simplificación de la realidad donde se define lo esencial para la construcción del software con los objetivos de comunicar la estructura de un sistema complejo, especificar el comportamiento deseado del sistema, comprender mejor lo que se está desarrollando y descubrir oportunidades de simplificación y reutilización (Rumbaugh, et al., 2001)

Para llevar a cabo el diseño del módulo se propone utilizar UML, en su versión 2.0, por las siguientes características:

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

1.2.2. Herramienta CASE

Para modelar el módulo se hará uso de una herramienta de Ingeniería de Software Asistida por Computadoras (CASE) por sus siglas en inglés. Las herramientas CASE son aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas ayudan en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el diseño de proyectos, cálculo de costes, documentación o detección de errores entre otras (Sommerville, 2011).

Visual Paradigm en su versión 8.0, se selecciona como herramienta CASE de modelado profesional, que utiliza *UML* para la completa representación de las etapas por las que transita un producto de software. Este permite la realización de una amplia gama de diagramas como: casos de uso, de

actividades, de despliegue, entre otros, así como la generación de código fuente desde los mismos y la documentación asociada al proceso que esté siendo modelado (Paradigm, 2013).

Además, esta herramienta brinda las siguientes ventajas:

- ✓ Dibujo: Facilita el modelado de UML, proporciona herramientas específicas para ello. Esto también permite la estandarización de la documentación, ya que se ajusta al estándar soportado por la herramienta.
- ✓ Corrección sintáctica: Controla que el modelado sea correcto
- ✓ Coherencia entre diagramas: Dispone de un repositorio común, es posible visualizar el mismo elemento en varios diagramas, esto evita las duplicidades.
- ✓ Integración con otras aplicaciones: Permite integrarse con otras aplicaciones como herramientas ofimáticas, lo cual aumenta la productividad.
- ✓ Trabajo multiusuario: Permite el trabajo en grupo, proporciona herramientas de compartición de trabajo.
- ✓ Reutilización: Dispone de una herramienta centralizada donde se encuentran los modelos utilizados para otros proyectos.
- ✓ Generación de código: Permite generar código de forma automática, esto reduce los tiempos de desarrollo y evita errores en la codificación del *software*.
- ✓ Generación de informes: Permite generar diversos informes a partir de la información introducida en la herramienta.

1.2.3. Lenguaje de programación

Un lenguaje de programación es aquella estructura que, con una cierta base sintáctica y semántica, imparte distintas instrucciones a un programa de computadora. El lenguaje de programación tiene la capacidad de especificar, de forma precisa, cuáles son los datos que debe trabajar un equipo informático, de qué modo deben ser conservados o transferidos dichos datos y qué instrucciones debe poner en marcha la computadora ante ciertas circunstancias (Merino, et al., 2012).

Python

Python en su versión 3.5, es un lenguaje de programación poderoso y fácil de aprender. Cuenta

con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos. Su elegante sintaxis y tipos de datos dinámicos junto con su naturaleza interpretada, hacen de este un lenguaje ideal para *scripting* y desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas (Sevilla, 2016).

A continuación, se listan algunas de las características más sobresalientes del lenguaje Python (Ramirez, 2010):

- ✓ Es orientado a objetos, pero soporta también los estilos de programación procedural y funcional.
- ✓ Corre en múltiples plataformas, incluyendo Windows, Mac OS y Linux.
- ✓ Su sintaxis y semántica es sencilla y consistente.
- ✓ Utiliza tipos dinámicos.
- ✓ Es adecuado tanto para programar scripts como aplicaciones de gran tamaño.
- ✓ Es muy modular.
- ✓ Cuenta con administración automática de memoria Incluye una poderosa y extensa biblioteca de clases.
- ✓ Cuenta con una gran comunidad que se dedica a promover su desarrollo y adopción.
- ✓ Por su naturaleza interactiva, resulta ideal para llevar a cabo programación experimental y desarrollo rápido.

HTML 5: es el Lenguaje para el Formato de Documentos de Hipertexto. Es un sistema para definir tipos de documentos estructurados y lenguajes de marcas para representar dichos documentos. HTML no es un lenguaje de programación como C++, sino un sistema de etiquetas que no se compila por lo que si existe algún error este no lo señalará, sino que representará en el navegador lo que interprete del código erróneo.

Estas etiquetas permiten representar textos con diversos estilos, así como aplicar diferentes efectos a los párrafos que permitan que el entorno del sitio *web* sea agradable para el usuario. Además, se pueden incorporar a las páginas multimedia como sonidos, imágenes y videos. También con HTML se pueden crear vínculos a otras páginas y pueden incluir los *scripts* que permiten más interactividad en la navegación, como los de *JavaScript*. El propio *World Wide Web Consortium (W3C)* define el lenguaje HTML como "un lenguaje reconocido universalmente y que permite publicar información de forma global".

Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje que se utiliza en muchas aplicaciones electrónicas como buscadores, tiendas *online* y banca electrónica (Gauchat, 2014).

CSS 3.0: es un lenguaje de hojas de estilo creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML, es decir páginas *web*. CSS es la mejor forma de separar los contenidos y su presentación, es imprescindible para crear páginas *web* complejas. Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML bien definidos y con significado completo. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Cuando se crea una página *web* con HTML se enmarcan los contenidos como son: párrafo, titular, texto destacado, tabla, lista de elementos, entre otros. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, posición de cada elemento dentro de la página, entre otros (Pérez, 2014).

JavaScript: técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Se utiliza principalmente para crear páginas *web* dinámicas, una página *web* dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Pérez, 2014).

Se utiliza principalmente en un navegador *web*, permitiendo el desarrollo de interfaces de usuario mejoradas y páginas *web* dinámicas. Puede incluirse en cualquier documento y es compatible con HTML. Puede ser utilizado por profesionales y por quienes se inician en el desarrollo y diseño de sitios *web*. No requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.

1.2.4. Entorno de desarrollo integrado

Entorno de desarrollo IDE (por sus siglas en inglés *Integrated Development Environment*), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

PyCharm es una multiplataforma utilizado para desarrollar en el lenguaje de programación Python. Es un IDE que posee un juego completo de herramientas para el desarrollo productivo. Para llevar a cabo la implementación del módulo se hará uso de PyCharm 2017, en su versión 2.2. Completamiento de código de manera inteligente y señalamiento de errores con reparación de los mismos de forma automatizada. Integración con marcos de trabajo de desarrollo web modernos como *Django*. Depurador integrado y herramientas de pruebas. Soporte para bases de datos e integración con sistemas de control de versiones (JetBrains, 2016)

1.2.5. Servidor web Apache

Los servidores orientan su operación sobre el protocolo HTTP. El protocolo de transferencia de hipertexto - HTTP es un protocolo de la capa de aplicación del modelo TCP/IP (Transmission control protocol/internet protocol) y es el corazón de la Web y define la forma como los clientes Web solicitan objetos a un servidor y cómo los servidores los transfieren a los clientes.

Apache en su versión 2.4 se caracteriza por ser estable, multiplataforma, modular y altamente configurable, lo cual significa que se puede adaptar para satisfacer diferentes necesidades. Además, dispone de componentes de seguridad, ser de código abierto y gratuito (Montoya, y otros, 2013).

Apache es una aplicación gratuita que convierte un ordenador en un servidor web. Es de código abierto, flexible, rápido y eficiente. *Apache* permite negociar protocolos *HTTP* entre una máquina que haría de servidor web y los otros ordenadores que deseen ver un determinado sitio web. Es fácil encontrar documentación para profundizar en las funcionalidades que ofrece. Se caracteriza por su gran escalabilidad, seguridad y rendimiento. Tiene soporte para varios lenguajes como *Perl*, *Python* y *PHP*, lo que permite desarrollar aplicaciones web de gran calidad (Apache Software Foundation, 2011).

1.2.6. Sistema gestor de base de datos

Un Sistema gestor de base de datos (SGBD) es el *software* que permite la utilización o actualización de los datos almacenados en una o varias bases de datos, por uno o varios usuarios desde diferentes puntos de vista. El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular los datos, de forma que no le sea necesario conocer el modo de almacenamiento ni el método de acceso empleado.

PostgreSQL en su versión 9.4.1, es un Sistema Gestor de Bases de Datos (SGBD) objeto-relacional que se encuentra distribuido bajo la licencia de software libre BSD. Es el sistema de gestión de bases

de datos de código abierto más potente del mercado. Cuenta con una amplia comunidad de usuarios y programadores que colaboran activamente. Se centra en que el software sea robusto, de calidad, fácil de mantener y se destaca por su estabilidad, potencia, robustez y facilidad de administración. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios, accediendo paralelamente al sistema (Martinez, 2010).

1.2.7. Marcos de trabajo

Se seleccionó *Django* como *framework* de desarrollo web del lado del servidor, pues, es construido sobre *Python*, fomenta el desarrollo ágil permitiendo la construcción de aplicaciones de alta calidad en un breve período de tiempo. Utiliza una modificación de la arquitectura Modelo – Vista – Controlador (MVC) llamada Modelo – Plantilla – Vista (*MTV*, por sus siglas en inglés). *Django* impulsa el desarrollo de código limpio y promueve la utilización de buenas prácticas de desarrollo web como el principio *DRY*. Entre algunos casos de éxito se encuentran: *Instagram*, *Pinterest* y *The New York Times* (Condri, 2012).

1.3. Metodología de desarrollo

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. En el que se indica paso a paso todas las actividades a realizar para lograr el producto informático deseado, indica además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además, detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla.

En la Ingeniería de Software, una metodología de desarrollo de software es la encargada de la separación de este proceso en distintas fases o etapas, que contienen actividades enfocadas a una mejor planificación y administración del mismo. Puede incluir la definición previa de una serie de artefactos que son creados y completados con el equipo que desarrolla o mantiene la aplicación (Sommerville, 2011).

Para el desarrollo de la solución propuesta se seleccionó Proceso Unificado Ágil (AUP, por sus siglas en inglés), versión UCI, debido a que logra estandarizar el proceso de desarrollo de software, para dar cumplimiento a las buenas prácticas que define CMMI-DEV (Modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software) v1.3. Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos.

AUP-UCI es una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. En aras de aumentar la calidad del *software* que se produce esta metodología se apoya en el Modelo CMMI-DEV v1.3. (por sus siglas en inglés *Capability Maturity Model Integration Development*, Integración de Modelos de Madurez de Capacidades para Desarrollo). El mismo constituye una guía para aplicar las mejores prácticas y obtener un producto o servicio con calidad en una entidad desarrolladora. Las fases de la metodología son inicio, ejecución y cierre (Sanchez, 2014).

Se aplica el escenario No.4 que evalúa el negocio a informatizar, como resultado obtiene un negocio muy bien definido y recomiendo en proyectos no muy extensos. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos.

Conclusiones del Capítulo

Las relaciones existentes entre los principales conceptos asociados al dominio de la presente investigación, y el análisis de los sistemas homólogos, permitieron una mayor comprensión de la propuesta de solución. Se realizó un estudio del marco teórico en el que se fundamenta el desarrollo del módulo de gestión propuesto, además de las herramientas y tecnologías que se utilizarán, seleccionado así Visual Paradigm para modelar el sistema, el uso de la metodología de desarrollo de software AUP en su versión UCI, PostgreSQL como gestor de bases de datos, el lenguaje de programación Python del lado del servidor y del lado del cliente JavaScript, HTML y CSS. También se utilizó PyCharm para la gestión de código fuente.

Capítulo 2: Características y diseño del módulo

En el presente capítulo se plasman las características del sistema a través de los requisitos funcionales y no funcionales, las historias de usuarios que especifican cada requisito funcional. Se describe la propuesta de solución donde se incluye la arquitectura del *software* para conocer la manera en que se estructura, los patrones de diseño a utilizar, el diagrama de clases, el diagrama de paquetes y por último el diagrama de despliegue.

2.1. Análisis de la propuesta de solución

Como solución se propone el desarrollo de un Módulo de Administración para el control y gestión de la información de SAATI en la Universidad de las Ciencias Informáticas, para garantizar con dicho módulo parte de la seguridad y organización en los contenidos generados por la aplicación.

El presente módulo garantiza:

- ✓ Una correcta administración de los datos que se generan en el transcurso del uso de la aplicación, siendo capaz de mostrar, adicionar, actualizar y eliminar información de forma segura.
- ✓ Responder a la necesidad de informar a los usuarios del sistema, ante la ejecución de acciones en la aplicación, que afecten información sensible relacionada con estos.
- ✓ También mantendrá un control oportuno de las principales actividades que realizan los usuarios en la plataforma, permitiendo llevar un control sobre la integridad de los datos del sistema.
- ✓ La gestión de los contenidos y analizar la información introducida a fines a la asignatura que serán consumidos por los estudiantes y consultada por los mismos cuando ellos decidan.
- ✓ El profesor también obtiene información de los avances de sus alumnos y el nivel que estos alcancen.
- ✓ Otorgar ventajas en el mismo o bien alentarlos a seguir intentándolo, así como interactuar con otros estudiantes que realizan consultas y aclaran dudas entre ellos.
- ✓ La gestión de roles y permisos, mediante la cual cada rol va a tener determinados permisos a funcionalidades específicas de la aplicación y cada usuario va a tener asignado un rol, garantiza

de esta manera parte de la seguridad del sistema y visualización de los contenidos correspondiente a cada usuario.

- ✓ La integración correcta entre los distintos componentes de SAATI y el módulo de administración.

2.2. Modelo de dominio

El Modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocio al cual el sistema va a servir. Permite comprender y describir las clases más importantes dentro del contexto del sistema. No es una descripción del diseño del software, es una representación visual de las clases conceptuales del mundo real en un dominio de interés (Paradigm, 2013).

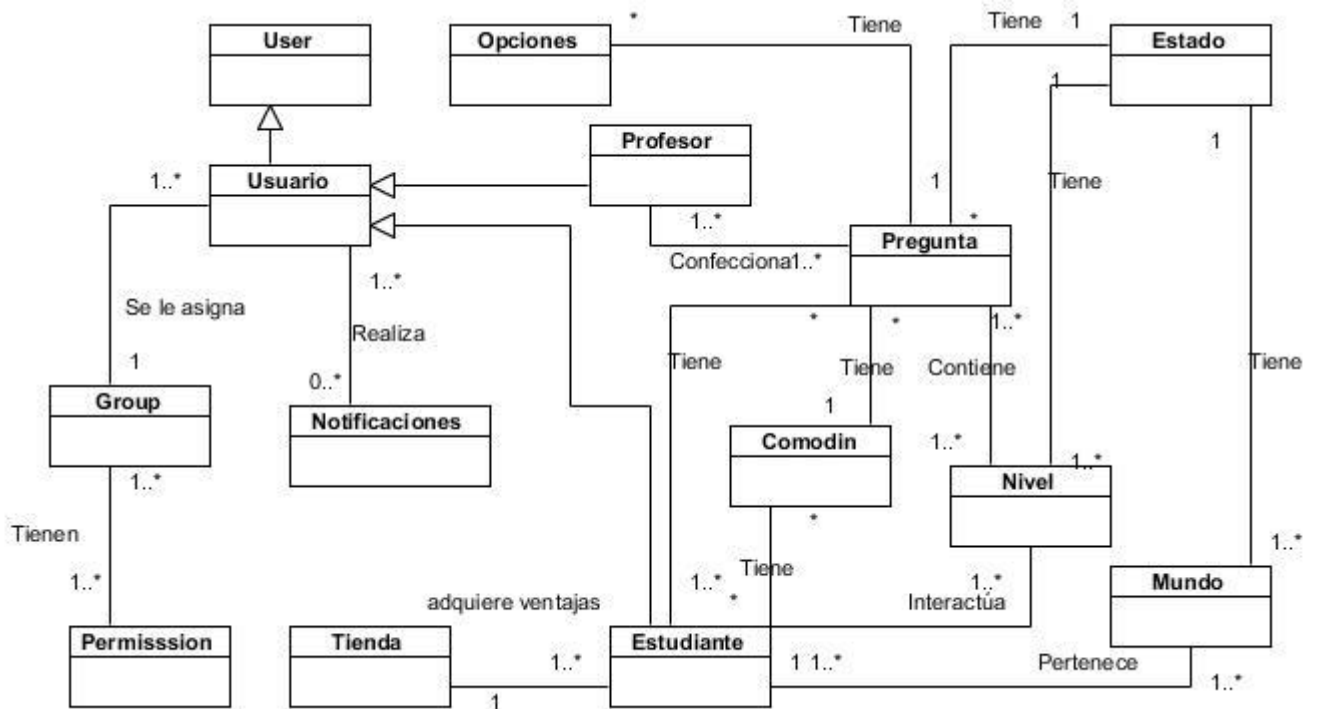


Ilustración 1: Modelo de dominio del sistema. (Elaboración propia)

Estudiante: usuario que usará el módulo de administración, para su estudio individual y realizar tareas y evaluaciones orientadas a responder preguntas para su preparación individual. Cuenta con un perfil el cual otros usuarios observan.

Profesor: usuario que se encarga de brindar la información necesaria para el estudio individual y confección de preguntas para los estudiantes, orientar los diferentes tipos de evaluación. Cuenta con un perfil el cual otros usuarios observan.

Mundo: hace referencia a las temáticas del juego de la asignatura de Teleinformática, a los cuales un estudiante debe pertenecer para jugar en el sistema.

Nivel: son parte de los mundos y organizan el sistema en un conjunto de preguntas para los usuarios estudiantes.

Pregunta: enunciado interrogativo que se emite por los profesores con la intención de que los estudiantes demuestren sus conocimientos en la asignatura de Teleinformática.

Permiso: son asignados por los administradores del sistema con la intención de cada rol pueda realizar determinadas tareas.

Rol: papel que cumple un determinado usuario de la aplicación, cumple con un conjunto de normas y comportamiento que fue otorgado por los administradores del sistema.

Notificaciones: son clasificadas como notificaciones de sistema y notificaciones e-mail; esta primera son emitidas a los usuarios por el sistema, las segundas son mensajes entre los usuarios de la aplicación.

Estado: es la situación o nivel de confección que se encuentra una pregunta, un nivel o un mundo. Determinado por un usuario profesor.

Respuesta: son una selección de una pregunta respondida por el estudiante y almacenada para su posterior consulta.

Tienda: Contiene comodines y puntos que se pueden comprar con el uso de los créditos obtenidos con el uso de la aplicación.

2.3. Requisitos funcionales (RF)

Tabla 1: Requisitos funcionales. (Elaboración propia.)

RF	Funcionalidad	RF	Funcionalidad
RF1	Crear roles del sistema.	RF2	Listar roles del sistema.
RF3	Actualizar roles del sistema.	RF4	Eliminar roles del sistema.
RF5	Crear permisos.	RF6	Listar permisos.
RF7	Actualizar permisos.	RF8	Eliminar permisos.
RF9	Adicionar usuarios.	RF10	Listar usuarios.
RF11	Actualizar usuarios.	RF12	Eliminar usuarios.
RF13	Crear mundo.	RF14	Listar mundo.
RF15	Actualizar mundo.	RF16	Eliminar mundo.
RF17	Crear nivel.	RF18	Listar nivel.
RF19	Actualizar nivel.	RF20	Eliminar nivel.
RF21	Listar opciones.	RF22	Eliminar opciones.
RF23	Crear estados.	RF24	Listar estados.
RF25	Actualizar estados.	RF26	Eliminar estados.
RF27	Crear pregunta.	RF28	Listar pregunta.
RF29	Actualizar pregunta.	RF30	Eliminar pregunta.
RF31	Crear notificaciones.	RF32	Listar notificaciones.
RF33	Actualizar notificaciones.	RF34	Eliminar notificaciones.
RF35	Crear grupos de estudiantes.	RF36	Listar grupos de estudiantes.
RF37	Actualizar grupos de estudiantes.	RF38	Eliminar grupos de estudiantes.
RF39	Listar acciones administrativas.	RF40	Actualizar cambios generales en el sistema.
RF41	Recuperar contraseña olvidada.	RF42	Crear comodines.
RF43	Listar comodines.	RF44	Eliminar comodines.
RF45	Actualizar comodines.	RF46	Crear opciones.
RF47	Actualizar opciones.		

2.4. Requisitos no funcionales (RNF)

Usabilidad:

RNF 1: Se debe mostrar al usuario todos los detalles sobre los errores ocurridos en su petición, sin comprometer la seguridad del sistema.

Seguridad:

RNF 2: Se debe restringir el acceso a los recursos en dependencia de los permisos del usuario autenticado.

RNF 3: El sistema debe permitir que los usuarios interactúen entre ellos de manera concurrente.

Rendimiento:

RNF 5: Ante los errores que puedan ocasionarse en el sistema, no se deben mostrar detalles de información que puedan comprometer la seguridad e integridad de los datos.

RNF 6: La velocidad de procesamiento de la información y los tiempos de respuestas deben ser menores a los 5 segundos.

Soporte:

RNF 7: El portal será actualizado por los administradores y editores del sistema diariamente. También será garantizado su mantenimiento, teniendo semanalmente un encuentro para analizar las inconformidades y deficiencias emitidas por los usuarios

RNF 8: El sistema debe dar la posibilidad de ser mejorado, así como de incorporarle nuevas funcionalidades en caso de ser necesarias.

2.5. Historias de Usuario (HU)

Según la metodología AUP-UCI una de las formas de encapsular los requisitos es mediante historias de usuarios.

Las HU se representan mediante tablas las cuales contienen las siguientes secciones:

- ✓ Número: número que identifica el requisito funcional.

- ✓ Nombre del requisito: Nombre que identifica la HU.
- ✓ Programador: programador de la funcionalidad escrita en la HU.
- ✓ Usuario: Usuario encargado del desarrollo y funcionamiento del requisito descrito en la HU.
- ✓ Riesgo en desarrollo: El riesgo que puede tener durante la fase de desarrollo: alto, medio o bajo según el criterio de los desarrolladores.
- ✓ Prioridad en el negocio: Esta característica es dada por el cliente con los valores: alta, media o baja en dependencia de la importancia en que desean ser implementadas.
- ✓ Puntos estimados: Tiempo estimado en semanas que se le asignará.
- ✓ Descripción: Breve descripción del proceso que define la HU.
- ✓ Observaciones: Alguna acotación importante de señalar acerca de la HU.

A continuación, se describen algunas historias de usuarios las restantes se encuentran en el anexos 1:

Tabla 2: Historia de usuario "Crear pregunta". (Elaboración propia.)

Historia de Usuario	
Número: HU_27	Nombre del requisito: Crear pregunta.
Programador: Melissa Mujica Suárez	Usuario: Profesor
Riesgo en Desarrollo: alta	Prioridad en el negocio: alta
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: un profesor visualiza un botón de <i>confeccionar una pregunta</i> , luego de seleccionado el botón, se abre un formulario el cual es rellenado por dicho profesor y guardado luego en base de dato. Las preguntas pueden ser creadas bajo la modalidad de ser sorpresa, y les proporcionan puntos extra a los estudiantes, esto especificado en el formulario.	
Prototipo:	

Añadir Pregunta

<p>Tipo de pregunta <input type="text"/></p> <p>Categoría a la que pertenece <input type="text"/></p> <p>Puntos que vale <input type="text"/></p> <p>Mundo que pertenece <input type="text"/></p>	<p>Texto de pregunta <input type="text"/></p> <p>Texto de respuesta <input type="text"/></p> <p>Estado de confección <input type="text"/></p> <p>Imagen <input type="text"/></p>
---	--

2.6. Arquitectura de software

La arquitectura de *software* es la estructura de un sistema o bien la forma en que va a estar organizado este; incluye los elementos que lo conforman, sus propiedades visibles desde el exterior y las relaciones que existen entre ellos (Pressman, 2010).

El *framework* de desarrollo web Django emplea una modificación del estilo arquitectónico Modelo-Vista- Controlador (MVC), llamada *Model-Template-View* (MTV), que sería Modelo-Plantilla-Vista, esta forma de trabajar permite que sea pragmático (Montero, 2012).

Se asumirá el estilo arquitectónico MTV de Django del cual se explica su funcionamiento a continuación y se podrá observar en la ilustración:

1. El navegador web envía una solicitud
2. La vista interactúa con el modelo para obtener los datos
3. La vista llama a la plantilla
4. La plantilla muestra la respuesta a la solicitud del navegador

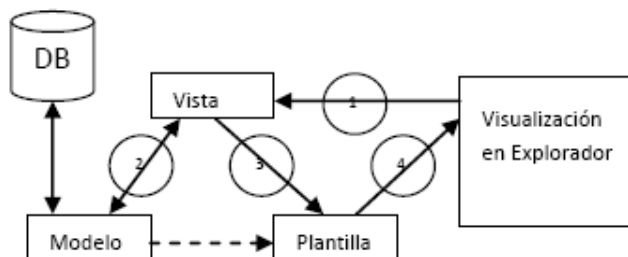


Ilustración 2: Flujo de trabajo de la arquitectura. (Condri, 2012)

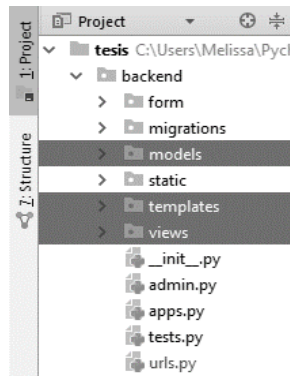


Ilustración 3: Fragmento de la arquitectura(MTV) en el sistema. (Elaboracion propia).

Modelo: Contiene toda la información sobre los datos. Cada una de las entidades de la base de datos se encuentra en el modelo en forma de clases de Python, y sus atributos se almacenan en variables con ciertos parámetros. También estos archivos poseen métodos, lo que permite indicar y controlar el comportamiento de los datos.

Vista: Es la capa de la lógica de negocios, contiene la lógica que accede al modelo y la delega a la plantilla apropiada. Esta capa sirve de “puente” entre el modelo y la plantilla, se presenta en forma de funciones de Python y su función principal es determinar qué datos serán visualizados en las plantillas.

Plantilla: Recibe los datos de la vista y luego los organiza para la presentación al navegador web. Básicamente es una página HTML (*HyperText Markup Language*) con algunas etiquetas extras que son propias del Django.

La aplicación presenta la siguiente estructura en el marco de trabajo según la arquitectura MTV:

models.py: Contiene la clase *Client*, responsable de brindar información de los clientes ligeros.

views.py: Contiene todas las *views* encargadas de acceder a la clase del *models.py* y llamar a las plantillas contenidas en la carpeta *templates*.

templates: Es la carpeta que contiene todas las plantillas utilizadas en la aplicación y llamadas desde las *views*.

2.7. Patrones de diseño

Un patrón de diseño es una buena práctica documentada de la solución de un problema que ha sido aplicado satisfactoriamente en múltiples entornos. Es una solución recurrente a un problema común observado o descubierto durante el estudio o construcción de numerosas aplicaciones. Su principal

objetivo es incrementar la calidad del *software* en términos de reusabilidad, mantenimiento y extensibilidad (Larman, 2010).

2.7.1. Patrones GRASP

Los patrones GRASP constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. Este enfoque para la comprensión y utilización de los principios de diseño se basa en los patrones de asignación de responsabilidades (Larman, 2010).

El **Experto en Información** se utiliza con frecuencia en la asignación de responsabilidades; es un principio de guía básico que se utiliza continuamente en el diseño de objetos. El Experto no pretende ser una idea oscura o extravagante; expresa la "intuición" común de que los objetos hacen las cosas relacionadas con la información que tienen.

```
def cant_preguntas (self):  
    return self.preguntas_set.count()
```

Ilustración 4: Ejemplo de patrón Experto. (Elaboración propia).

El patrón **Creador** guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. La intención básica del patrón Creador es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Elijiéndolo como el creador se favorece el bajo acoplamiento.

```
@login_required(login_url='/')  
@admin_required(login_url=ERROR_403_URL)  
@active_required(login_url=settings.ADMIN_LOOK_URL)  
@permission_required(perm='change_pregunta')  
def activate_pregunta(request, pk):  
    pregunta = get_object_or_404(Pregunta, pk=pk, is_delete=True)  
    pregunta.is_delete = False  
    pregunta.delete_at = None  
    pregunta.delete_by = None  
    pregunta.save()  
    messages.success(request, 'pregunta activada')  
    return HttpResponseRedirect(reversed('modules:Pregunta'))
```

Ilustración 5: Ejemplo de patrón Creador. (Elaboración propia.)

Un **Controlador** es un objeto que no pertenece a la interfaz de usuario, responsable de recibir o manejar un evento del sistema. Un Controlador define el método para la operación del sistema.

```

@login_required(login_url=settings.ADMIN_LOGIN_URL)
@admin_required(login_url=ERROR_403_URL)
@active_required(login_url=settings.ADMIN_LOOK_URL)
@permission_required(perm='change_pregunta')

class add_pregunta(CreateView):
    model = Pregunta
    form_class = preguntaForm
    template_name = "pregunta/add_pregunta.html"
    success_url = reverse_lazy('listaPregunta')

```

Ilustración 6: Ejemplo de patrón Controlador. (Elaboración propia).

2.7.2. Patrones GoF

Representan patrones que dan soluciones técnicas basadas en programación orientada a objeto (POO) que favorecen la reutilización del código (Guerrero, y otros, 2013).

Decorador: Patrón estructural que extiende la funcionalidad de un objeto dinámicamente de manera tal que es transparente a sus clientes, utiliza una instancia de una subclase de la clase original que delega las operaciones al objeto original. Como ejemplo de este patrón se evidencia la utilización del decorador de Django.

```

@login_required(login_url=settings.ADMIN_LOGIN_URL)
@admin_required(login_url=ERROR_403_URL)
@active_required(login_url=settings.ADMIN_LOOK_URL)
@permission_required(perm='change_pregunta')

class add_pregunta(CreateView):

```

Ilustración 7: Ejemplo de patrón Decorador. (Elaboración propia)

2.8. Diagrama de clases del diseño

Los diagramas de clases describen la estructura estática de un sistema. Las cosas que existen y que nos rodean se agrupan naturalmente en categorías. Una clase es una categoría o grupo de cosas que tienen atributos (propiedades) y acciones similares. Incluye la siguiente información (Larman, 2010):

- ✓ Clases, asociaciones y atributos.
- ✓ Interfaces, con sus operaciones y constantes.
- ✓ Métodos.
- ✓ Navegabilidad.

✓ Dependencias.

A diferencia del Modelo Conceptual, un Diagrama de Clases de Diseño muestra definiciones de entidades software más que conceptos del mundo real.

Los estereotipos usados son:

<<Server page>> Representa la página web que contiene código que se ejecuta en el servidor.

<<Client Page>> Una instancia de Página Cliente es una página web, con formato HTML. Cada página cliente es construida por una sola página de servidor.

<<Form>> Colección de elementos de entrada que son parte de una página cliente. Sus atributos son los elementos de entrada del formulario.

<<Submit>> Es la relación que se crea siempre entre una página servidor y un formulario, a través de esta relación el formulario manda los valores de sus campos al servidor, para ser procesados por la página servidor.

<<Build>> Representa una asociación especial que relaciona las páginas cliente con las páginas servidor, de forma general se expresa como que las páginas que se encuentran en el servidor construyen las páginas en el cliente.

<<Redirect>> Indica que la página de servidor además de construir una página cliente puede redireccionar el procesamiento a otra página.

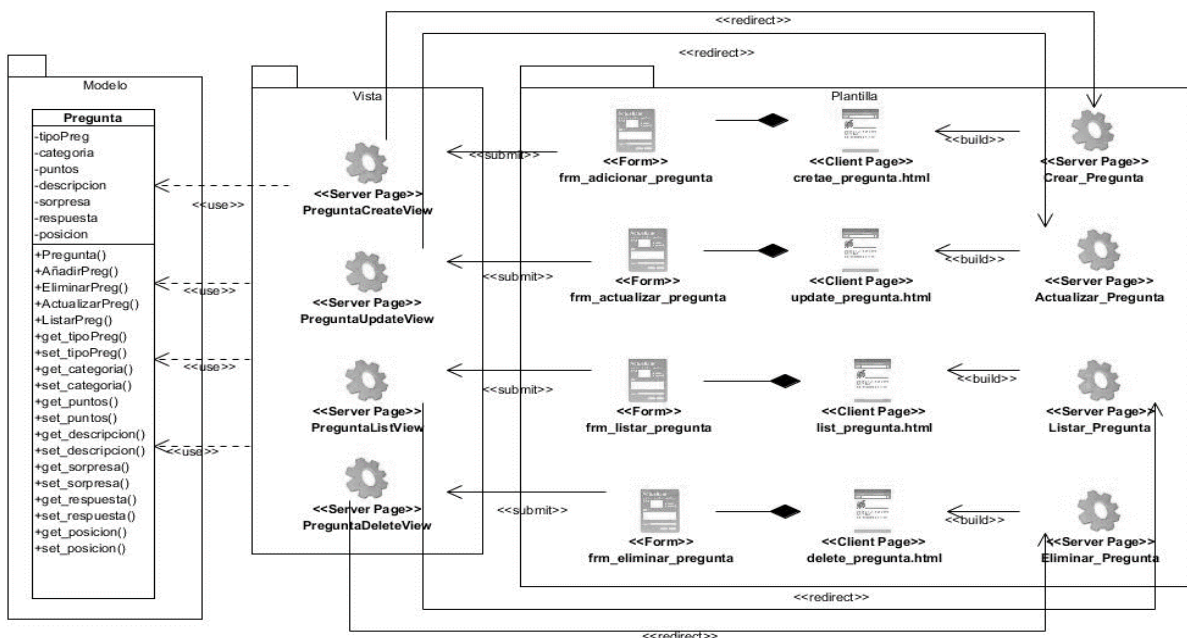


Ilustración 8: Diagrama de clases "Gestionar pregunta". (Elaboración propia).

2.9. Modelo de datos

Un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos o podríamos decir que es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, la estructura de una base de datos (Redondo, 2017).

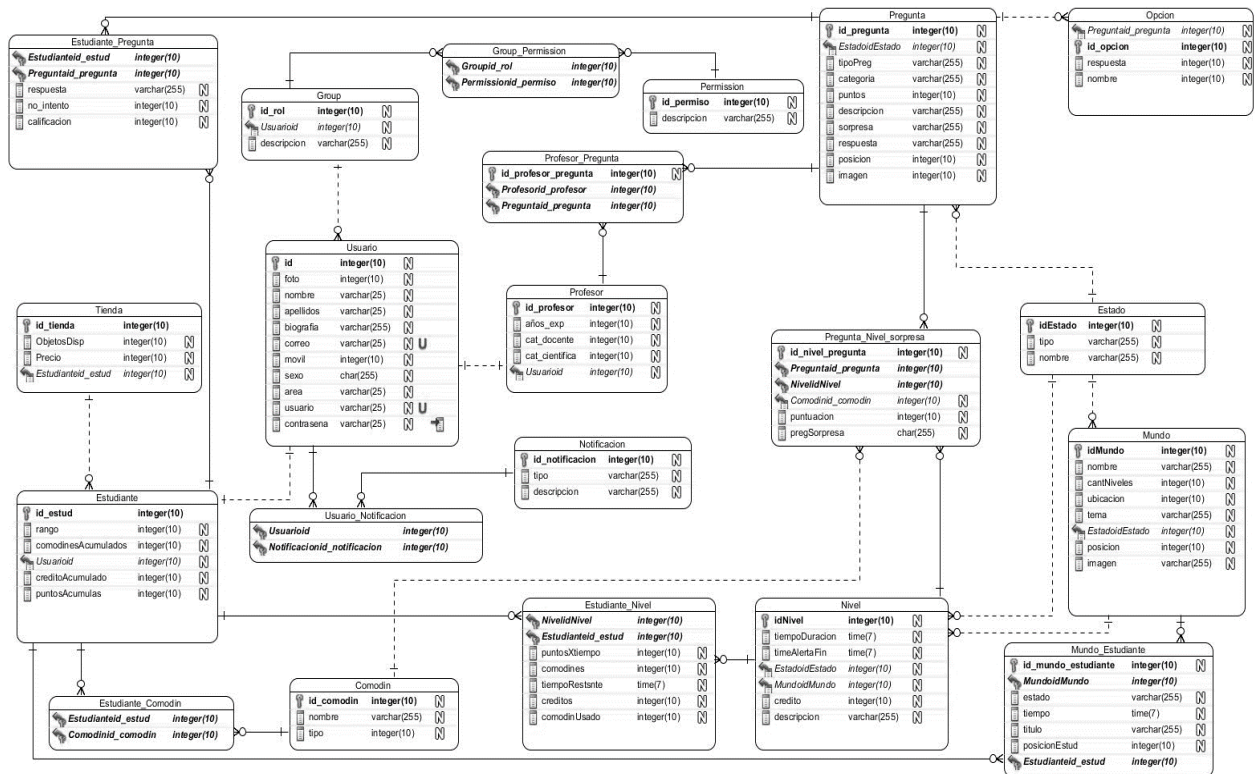


Ilustración 9: Modelo de base de dato del sistema. (Elaboración propia).

2.10. Modelo de despliegue

Un diagrama de despliegue modela la arquitectura de un sistema en tiempo de ejecución; muestra la configuración de los elementos de *hardware* (nodos) y muestra cómo los elementos y artefactos del *software* se trazan en esos nodos. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que generalmente tiene memoria y a menudo, capacidad de procesamiento. Representan un procesador o un dispositivo sobre el que se pueden desplegar los componentes. La relación entre nodo y componente que despliega puede mostrarse con una relación de dependencia (Krajewski, y otros, 2000).



Ilustración 10: Diagrama de despliegue. (Elaboración propia).

El nodo Cliente PC se refiere a las estaciones de trabajo que realizan las peticiones al servidor de aplicaciones, mediante un navegador web utiliza el protocolo de comunicación *HTTPS* Protocolo de Transferencia de Hipertexto Seguro conocido (por sus siglas en inglés *HTTPS*), es el protocolo utilizado entre los navegadores de los clientes y el servidor web. Este elemento en la arquitectura representa un tipo de comunicación segura y confiable entre clientes y servidor, ya que garantiza que cualquier dato introducido será cifrado. El *Servidor Web* es el encargado de brindar la interfaz de la plataforma para que los usuarios puedan hacer uso de esta, almacena todo el código fuente del sistema y se comunica por medio de los protocolos *TCP* con el servidor de bases de datos. El sistema gestor de bases de datos almacena toda la información que brinda la plataforma hospedada en el servidor de aplicaciones. La información es obtenida o modificada en dependencia del nivel de privilegio del usuario que realiza la petición.

Conclusiones del Capítulo

El desarrollo de las historias de usuario, los requisitos funcionales y no funcionales, la elaboración de los diagramas de clases del diseño y el diagrama de paquetes, posibilitaron un mejor diseño y comprensión del funcionamiento de la propuesta de solución. La metodología de desarrollo seleccionada y el estilo arquitectónico, constituyeron un guía fundamental para la construcción de la propuesta de solución.

Capítulo 3: Implementación y pruebas al módulo

En el presente capítulo se muestra el diagrama de componentes como resultado del diseño desarrollado. La fase de implementación comprende la materialización, en forma de código, de los artefactos y las descripciones con el objetivo de conformar el producto final requerido por el cliente. Además, en este capítulo se presenta la validación del sistema, con el objetivo de corroborar la correspondencia entre el producto y los requisitos definidos anteriormente.

3.1 Diagrama de componente

El diagrama de componentes tiene como propósito mostrar las relaciones estructurales entre los componentes de un sistema, sus dependencias y localización. Las dependencias en este tipo de diagrama indican que un componente utiliza los servicios ofrecidos por otro componente. Los componentes son considerados unidades autónomas encapsuladas dentro de un sistema que pueden ser librerías, ejecutables, binarios y códigos fuente (Bell, 2004).

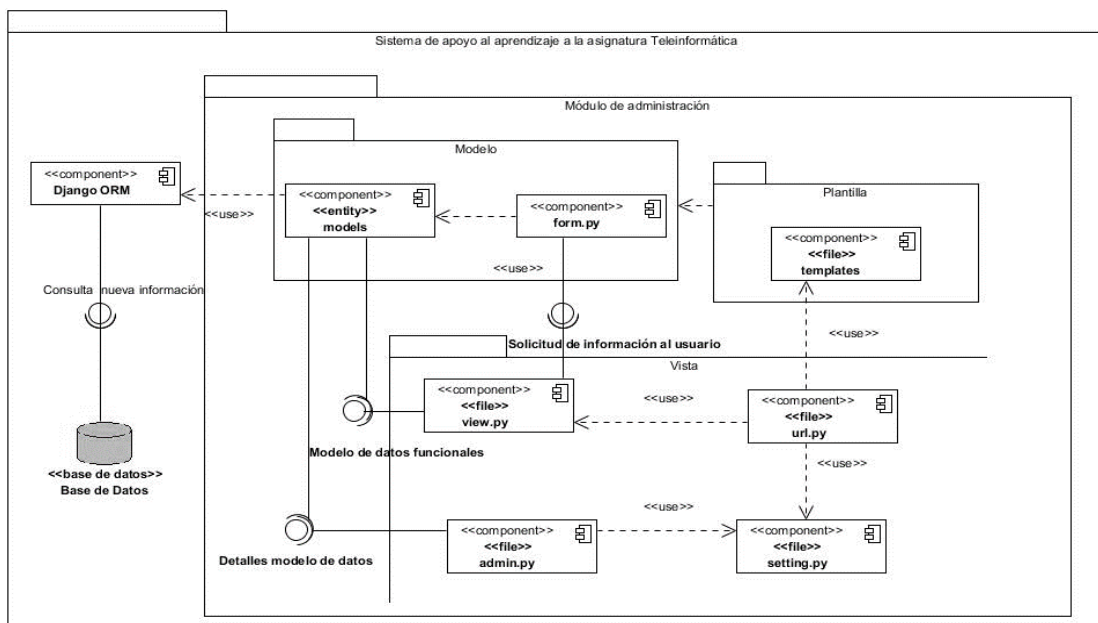


Ilustración 11: Diagrama de componente. (Elaboración propia)

3.2 Estándares de codificación

Para la codificación de la propuesta de solución se utilizará el estándar de codificación para Python basado en la guía de estilo del código Python por Guido Van Rossum, Barry Warsaw. En este documento

se listan distintas convenciones utilizadas en el código Python comprendido en la librería estándar de la distribución principal de Python (Colaboradores Python, 2013).

Tabla 3: Estándares de codificación. (Elaboración propia.)

Alineación
Las líneas de continuación deben alinearse verticalmente con el carácter que se ha utilizado (paréntesis, llaves, corchetes). Utilizar un sangrado de una tabulación para cada línea con excepción de la primera. El sangrado se realizará solamente con tabulaciones, no deben utilizarse nunca los cuatro (4) espacios.
Líneas en blanco
Separar las funciones de alto nivel y definiciones de clases con dos (2) líneas en blanco. Las definiciones de métodos dentro de una clase deben separarse por una (1) línea en blanco. Se pueden utilizar líneas en blanco, escasamente, para separar secciones lógicas.
Máxima longitud de las líneas
Todas las líneas deben estar limitadas a un máximo de setenta y nueve (79) caracteres. Dentro de paréntesis, corchetes o llaves se puede utilizar la continuación implícita para cortar las líneas largas. En cualquier circunstancia se puede utilizar el carácter “\” para cortar las líneas largas.
Codificaciones
Utilizar la codificación UTF-8. Se pueden incluir cadenas que no correspondan a esta codificación utilizarían “\x”, “\u” o “\U”.
Importación
Las importaciones deben estar en líneas separadas. Siempre deben colocarse al comienzo del archivo. Deben quedar agrupadas de la siguiente forma: Importaciones de la librería estándar. Importaciones terceras relacionadas. Importaciones locales de la aplicación / librerías. Cada grupo de importaciones debe estar separado por una línea en blanco.

Espacios en blanco en expresiones y sentencias

Evitar utilizar espacios en blanco en las siguientes situaciones:

- Inmediatamente dentro de paréntesis, corchetes y llaves.
- Inmediatamente antes de una coma, un punto y coma o dos puntos.
- Inmediatamente antes del paréntesis que comienza la lista de argumentos en la llamada a una función.
- Inmediatamente antes de un corchete que empieza una indexación.
- Más de un espacio alrededor de un operador de asignación (u otro) para alinearlos con otro.

Deben rodearse con exactamente un espacio los siguientes operadores binarios:

- Asignación (=).
- Asignación de aumentación (+=, -=, etc.).
- Comparación (==, <, >, >=, <=, !=, <>, in, not in, is, is not).
- Expresiones lógicas (and, or, not).

Si se utilizan operadores con prioridad diferente se aconseja rodear con espacios a los operadores de menor prioridad.

No utilizar espacios alrededor del igual (=) cuando es utilizado para indicar un argumento de una función o un parámetro con un valor por defecto.

Comentarios

Los comentarios deben ser oraciones completas.

Si un comentario es una frase u oración su primera palabra debe comenzar con mayúscula a menos que sea un identificador que comience con minúscula.

Nunca cambiar las minúsculas y mayúsculas en los identificadores de clases, objetos, funciones, etc.

Si un comentario es corto el punto final puede omitirse.

Comentarios en bloque

Deben tener un sangrado al mismo nivel que el código a comentar.

Cada línea de un comentario en bloque comienza con un numeral (#) y un espacio en blanco.

Comentarios en la misma línea

Se recomienda utilizarlos escasamente.

Se debe definir comenzar por un numeral (#) seguido de un espacio en blanco.

Deben ubicarse en la misma línea que se desea comentar.

Cadenas de documentación
Deben quedar documentados todos los módulos, funciones, clases y métodos públicos. Para definir una cadena de documentación debe quedar encerrada dentro de ("""). Los (""") que finalizan una cadena de documentación deben quedar en una línea a no ser que la cadena sea de una sola línea.
Convenciones de nombramiento
Nunca se deben utilizar como simple caracteres para nombres de variables los caracteres ele minúscula "l", o mayúscula "O", ele mayúscula "L" ya que en algunas fuentes son indistinguibles de los números uno (1) y cero (0). Los módulos deben tener un nombre corto y en minúscula. Los nombres de clases deben utilizar la convención "CapWords" (palabras que comienzan con mayúsculas). Los nombres de las excepciones deben estar escrito también en la convención "CapWords" utilizan el sufijo "Error". Los nombres de las funciones deben estar escritos en minúscula separando las palabras con un guión bajo "_". Las constantes deben quedar escritas con letras mayúsculas separando las palabras por un guión bajo (_).

3.3 Pruebas funcionales

Se denominan pruebas funcionales o *Functional Testing*, a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplen con las funciones específicas para los cuales han sido creados, es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales (Guerrero, 2017).

A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, ya que los *testers* o analistas de pruebas no enfocan su atención a como se generan las respuestas del sistema, básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas (Serna, 2015).

A continuación, el caso de prueba correspondiente a la funcionalidad "Añadir pregunta":

Tabla 4: Variables de entrada para los casos de prueba. (Elaboración propia)

No.	Nombre del campo	Clasificación	Valor	Descripción
1	Tipo de pregunta	Campo selección	No nulo	Puede ser una selección de tres que aparecen.
2	Categoría a la que pertenece	Campo selección	No nulo	Puede ser una de las opciones que muestra el campo de seleccionar
3	Puntos que vale	Campo número	Opcional	Puede ser cualquier número entero positivo
4	Texto de pregunta	Campo texto	No nulo	Puede ser cualquier combinación de caracteres alfanuméricos incluyendo el espacio
5	Texto de la respuesta	Campo texto	No nulo	Puede ser cualquier combinación de caracteres alfanuméricos incluyendo el espacio
6	Estado de confección	Campo selección	No nulo	Puede ser una de las opciones que muestra el campo de seleccionar
7	Imagen	Campo de input	Opcional	Puede ser una introducción de un imagen.

Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

Tabla 5: Caso de prueba “Añadir Pregunta”(parte 1). (Elaboración propia).

Escenario	Descripción	V1	V2	V3	Respuesta	Flujo central
EC 1.1: Añadir pregunta de forma correcta	Se registran los datos de una pregunta de forma correcta	V Selecciona el tipo de pregunta	V Selecciona la categoría	V Puntos que va a aportar la pregunta	Se inserta correctamente una pregunta	El profesor inserta los datos en el formulario y da 'clic' en Guardar

EC 1.2: Añadir pregunta dejando campos vacíos	Existen campos que pueden estar vacíos al añadir una pregunta	V Selecciona el tipo de pregunta	- Vacío	N/A Puntos que va a aportar la pregunta	Muestra el mensaje "Rellene los campos obligatorios"	El profesor deja vacíos algunos campos obligatorios
EC 1.3: Añadir pregunta de forma incorrecta	No se adicionan preguntas con datos incorrectos	V Selecciona el tipo de pregunta	I Introduce números	V Puntos que va a aportar la pregunta	Muestra el mensaje "El campo no acepta ese dato"	El profesor inserta algún tipo de dato que no corresponde

Tabla 6: Caso de prueba "Añadir Pregunta"(parte 2). (Elaboración propia).

Escenario	Descripción	V4	V5	V6	V7	Respuesta	Flujo central
EC 1.1: Añadir pregunta de forma correcta	Se registran los datos de una pregunta de forma correcta	V Introduce texto de la pregunta	V Introduce texto de la respuesta	V Selección estado de confección inicial	V Inserta imagen	Se inserta correctamente una pregunta	El profesor inserta los datos en el formulario y da 'clic' en Guardar
EC 1.2: Añadir pregunta dejando campos vacíos	Existen campos que no pueden estar vacíos al añadir una pregunta	V Introduce texto de la pregunta	- Vacío	- Vacío	N/A Inserta imagen	Muestra el mensaje "Rellene los campos obligatorios"	El profesor deja vacíos algunos campos obligatorios
EC 1.3: Añadir pregunta de forma incorrecta	No se adicionan preguntas con datos incorrectos	V Introduce texto de la pregunta	V Introduce texto de la respuesta	V Selección estado de confección inicial	V Inserta imagen	Muestra el mensaje "El campo no acepta ese dato"	El profesor inserta algún tipo de dato que no corresponde

Resultados de las pruebas funcionales

Las pruebas funcionales se realizaron en cuatro iteraciones donde se aplicaron los casos de prueba diseñados, la primera iteración arrojó diecisiete (17) no conformidades, de las cuales se solucionaron (7), la segunda iteración arrojó diez (10) no conformidades, fueron solucionadas (5), la tercera arrojó cinco (5) no conformidades, todas estas resueltas y la cuarta iteración no arrojo ninguna (0) no conformidad. A continuación, se muestran los resultados de las pruebas en cada iteración.

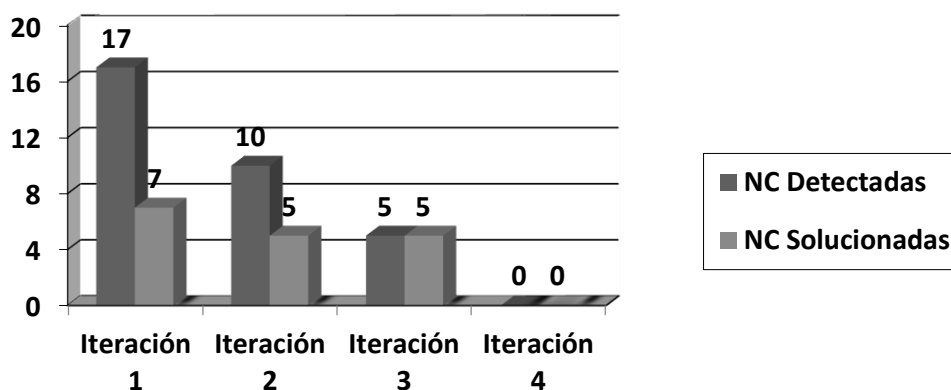


Ilustración 12: Resultados de las pruebas funcionales. (Elaboración propia).

El gráfico de la ilustración anterior muestra el comportamiento de las no conformidades encontradas durante el proceso de prueba, donde se observa que en la primera iteración se detectaron diecisiete (17) no conformidades, de ellas cuatro (4) errores de interfaz, nueve (9) validaciones incorrectas y cuatro (4) errores de excepción. En la segunda iteración se detectaron dos (2) nuevas no conformidades, las cuales se resolvieron satisfactoriamente además de las dos (2) pendientes de la iteración anterior. Luego de una tercera iteración se resolvieron todas las no conformidades, por lo que el módulo se encuentra listo para su funcionamiento.

3.4. Pruebas de rendimiento

Las pruebas de rendimiento tienen que diseñarse (también conocidas como pruebas de carga y estrés) para asegurar que el sistema pueda procesar la carga esperada. Esto normalmente implica planificar una serie de pruebas en las que la carga se va a incrementar regularmente hasta que el rendimiento del sistema se haga inaceptable.

Pruebas de carga

La carga de trabajo se refiere a la capacidad máxima que tiene un servidor web, para atender a un conjunto de usuarios de manera simultánea. Por ello, las actividades de esta etapa tienen relación con comprobar, de manera anticipada, el funcionamiento que tendrá el servidor del Sitio Web cuando esté en plena operación (Díaz, y otros, 2014).

Las pruebas en este caso consisten en simular una carga de trabajo similar y superior a la que tendrá cuando el sitio esté funcionando, con el fin de detectar si el software instalado cumple con los requerimientos de muchos usuarios simultáneos y también si el hardware es capaz de soportar la cantidad de visitas esperadas (Serna, 2015).

Pruebas de estrés

Enfocada a evaluar cómo el sistema responde bajo condiciones anormales. (Extrema sobrecarga, insuficiente memoria, servicios y hardware no disponible, recursos compartidos no disponible (Perez, 2015).

Resultados de las pruebas de carga y estrés

Las pruebas se desarrollaron con el apoyo de la herramienta Apache Meter 2.12, en la que se simuló el entorno donde debe interactuar el módulo para obtener la información más correcta acerca del comportamiento y resultados en general. Las pruebas son realizadas en una computadora con características como sistema operativo Windows 10 y arquitectura de 64bits, como sistema gestor de base de datos se usó PostgreSQL 9.4, como servidor web Apache v2.4 y se tomó como muestra máxima entre 1000 y 2000 estudiantes. Las características del hardware utilizado fue un microprocesador Intel Core(TM) i3 – 5120M y memoria RAM de 4GB.

Las siguientes variables miden los resultados obtenidos en las pruebas:

- ✓ Muestra: Cantidad de peticiones realizadas para cada URL.
- ✓ Media: Tiempo promedio en milisegundos en el que se obtienen los resultados.
- ✓ Mediana: Tiempo en milisegundos en el que se obtuvo el resultado que ocupa la posición central.
- ✓ Min: Tiempo mínimo que demora un hilo en acceder a una página.
- ✓ Max: Tiempo máximo que demora un hilo en acceder a una página.

- ✓ Línea 90%: Máximo tiempo utilizado por el 90% de la muestra, al resto de la misma le llevo más tiempo.
- ✓ % Error: Por ciento de error de las páginas que no se llegaron a cargar de manera satisfactoria.
- ✓ Rendimiento (Rend): El rendimiento se mide en cantidad de solicitudes por segundo.
- ✓ KB/s: velocidad de carga de las páginas.

Tabla 7: Resultados de las pruebas de carga y estrés

Usuarios	Muestra	Media	Mediana	Línea 90%	Min	Max	%Error	Rend.	KB/s
700	1000	1249	1422	1724	1	2666	0	90	1000
800	1500	4176	5200	3110	5	5002	1	80	1500
1000	2000	9770	1020	5142	4	3231	1.2	75	2000

La tabla resume los resultados de aplicar las pruebas de carga y estrés con ayuda de la herramienta Apache JMeter, en la misma se muestra que la aplicación con un total de 700 usuarios, el sistema responde en un tiempo promedio de 1.5 segundos, con una cantidad de 90 solicitudes por segundo y 0% de errores. Para comprobar en la carga esperada en condiciones altas se tomó un total de 1000 usuarios, respondiendo el sistema en un tiempo promedio de 2 segundos con una cantidad de 75 solicitudes por segundo y 1.2% de errores.

3.5. Pruebas de integración

Este tipo de pruebas consiste en la comprobación de que los elementos del software que interactúan entre sí, funcionan de manera correcta. Es una forma de comprobar la correcta interrelación de los distintos componentes del sistema. En el caso de la solución desarrollada es la verificación de una correcta interoperabilidad entre el módulo desarrollado y entre la aplicación de apoyo al aprendizaje a la asignatura de Teleinformática.

Resultados de las pruebas de integración

El módulo se integra a SAATI de manera continua a través del uso de Git, pensando en la eficiencia y la confiabilidad del mantenimiento de las versiones de aplicaciones, se utilizó una rama para trabajar en

cada proyecto, archivando y coordinando el trabajo que varias personas realizan y una rama master en la que se integra el código final. Además, se hace uso de codecomunidades o Gitlab que almacena los códigos utilizando Git. Por lo tanto, se concluye que existe una correcta integración entre los componentes internos del sistema sin presentar no conformidades, debido a que el módulo de administración y la aplicación fueron implementadas sobre un mismo proyecto.

3.6. Pruebas de aceptación

Según la metodología AUP variación UCI, es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

Con el objetivo de evaluar la solución implementada se realizó la técnica de ladov a SAATI y a su módulo de administración. Esta técnica evalúa el nivel de satisfacción del usuario, de esta forma se conoce si los componentes realizados cumplen las expectativas de los clientes; está conformada por cinco preguntas: 3 cerradas y 2 abiertas (Francés, 2019).

La técnica de ladov constituye una vía indirecta para el estudio de la satisfacción, debido a que los criterios que se utilizan se fundamentan en las relaciones que se establecen entre las preguntas cerradas (preguntas 1, 2 y 3) que se intercalan dentro de un cuestionario (ver Anexo 2) y cuya relación el sujeto desconoce (Francés, 2019).

Por otra parte, se le entregó la aplicación al cliente, quien en colaboración de un grupo de especialistas de la asignatura Teleinformática, emitió un criterio satisfactorio a través de una carta de aceptación. Se valida además que el sistema satisface las necesidades de los usuarios, y cumple con los requisitos especificados por el mismo en un inicio (Francés, 2019).

Resultados de las pruebas realizadas

El sistema le permitirá a un profesor gestionar con mínimos esfuerzos y administrar los contenidos en el sistema conociendo el porcentaje de recursos de *hardware* que están consumiendo en el servidor. Además sirve de apoyo para el aprendizaje de la asignatura de teleinformática y también informatizar algunos de sus procesos lo puedan hacer mediante la utilización del sistema. Es una de las propuestas que aborda el tema de la gamificación, a través del desarrollo de competencias a través de una tabla de posiciones centralizada, entre otras, lo cual motiva al estudiante para enfrentarse a contenidos de alta complejidad. Por lo que su uso en las aulas y la docencia en general, brindan satisfactorios resultados.

Conclusiones del capítulo

La realización del diagrama de componentes, así como la especificación de los estándares de codificación permitieron visualizar con facilidad la estructura general de la solución y permitieron implementar códigos completamente legibles para el buen entendimiento de otros programadores y facilitar así su recodificación. Por último, la detección temprana de errores, mediante la aplicación de pruebas permitió validar que el sistema funciona como inicialmente fue diseñado, que no se incurrió en la violación de ningún requisito y la implementación fue correcta.

Conclusiones generales

Con la finalización de esta investigación, la cual tuvo como base el desarrollo de un módulo de administración para SAATI, para ello se dio cumplimiento satisfactorio a una serie de objetivos específicos, por lo que se alcanzaron las siguientes conclusiones:

- ✓ Se desarrolló un módulo de administración para SAATI que garantiza un entorno dinámico y permite la adecuada gestión de contenidos en correspondencia con los usuarios.
- ✓ El sistema le permite a un profesor gestionar con mínimos esfuerzos y administrar los contenidos en el sistema conociendo el seguimiento de los estudiantes, y el desarrollo de la gestión de los cursos y contenidos. Permite a un administrador monitorear las actividades de los usuarios y analizar ante cualquier fallo en el módulo de administración.
- ✓ El módulo implementado ofrece una interfaz que facilita el uso a los profesores y administradores gestionar el sistema y dota a los estudiantes de una herramienta para el estudio individual de los contenidos de la asignatura.
- ✓ Se realizaron pruebas a la solución implementada y se comprobó el cumplimiento satisfactorio de los requerimientos de la misma validándose también la aceptación de los usuarios que utilizarán la misma.

Referencia Bibliográficas

Apache Software Foundation. 2011. [En línea] The Apache HTTP Server Project., 20 de mayo de 2011. [Citado el: 2016 de diciembre de 6.] <http://httpd.apache.org/>.

Badzanalla, R. 2016. *Ingeniería del software. Sistemas Informáticos. Nivel de madurez software. Informática Aplicada a la Gestión Pública.* 2016.

Bell, D. 2004. UML basics: The component diagram. [En línea] 2004. <https://www.ibm.com/developerworks/rational/library/dec04/bell/>.

Benvenuto, A. 2003. Las tecnologías de la información y las comunicaciones (TIC) en la docencia universitaria. [En línea] 2003. <http://www.redalyc.org/resumen.oa?id=29901210>.

Campo, M. G. 2016. *Desarrollo e Implementación de una aplicación web para la enseñanza de Gestión de Riesgos utilizando el framework deGamification como técnica de e-learning.* Samborondón : s.n., 2016.

Colaboradores Python. 2013. Guía de estilo para el código Python-PEP. [En línea] 2013. [Citado el: 20 de diciembre de 2018.] www.recuriospython.com/pep8es.pdf.

Condri, J. L. 2012. *Python-Django. Framework de desarrollo web para perfeccionistas.* 2012.

Contreras Espinosa, R., Eguia Gómez, J. L. y Solano Albaje, L. 2016. Investigación-acción como metodología para el diseño de un serious game. s.l. : RIED: Revista Iberoamericana de Educación a Distancia, 2016. Vol. 19, 2.

Corrales, A. R. 2009. *La integración de las Tecnologías de la Información y Comunicación (TIC) en el area de la Educación Física.* s.l. : Hekademos: revista educativa digital, 2009.

Definición de Herramienta. 2016. Significado y definición de Herramienta. [En línea] 26 de October de 2016. [Citado el: 20 de october de 2018.] <https://www.mastermagazine.info/termino/5234.php>.

Díaz, J. A. y Dosting, A. 2014. *Gestión de empresas de desarrollo de software a partir de aplicaciones de tipo código abierto.* 2014.

Diseño web adaptativo para la plataforma educativa Zera. **Manso Guerra, Y., Cañizares Gonzales, R. y Febles, J. 2016.** 2, s.l. : Revista Cubana de Ciencias Informáticas, 2016, Vol. 10.

Espeso, P. 2016. ClassDojo: ¿qué es y cómo empezar a usarlo en clase? [En línea] Educacion 3.0, 3 de marzo de 2016. [Citado el: 2018 de noviembre de 14.] <http://www.clasdojo.com>.

Estado actual de los sistemas e-learning. Teoría de la Educación. **Peñalvo, J. F. Garcia. 2005.** 2, s.l. : Educación y Cultura en la Sociedad de la Información, 2005, Vol. 6.

Fernández Pampillón, A. C. 209. Las plataformas e-learning para la enseñanza y el. [aut. libro] C. Lopez Alonso y M.Matesanz del Barrio. *Las plataformas de aprendizaje. Del mito a la realidad.* Madrid : Biblioteca nueva, 209.

Francés, León R. A. 2019. *Sistema Informatico de apoyo a la asignatura de Teleinformática.* La Habana : s.n., 2019.

Galindo, R. 2013. *Guión Visual Paradigm for UML.* 2013.

Garcia Matos, R. M. 1999. *Diseño de almacenes de datos.* 1999.

Garcia, F. y Mancebo, J. 2015. *BPMS-Game: Herramienta para la Gamificación de Procesos de Negocio.* 2015.

Gauchat, J. D. 2014. *El gran libro de HTML5.* s.l. : Editorial Tomera Ltda, 2014.

Guerrera, Y. A. 2010. *Conceptualizacion de una red educativa que integre de forma colaborativa las aplicaciones e-learning de la UCI.* 2010.

Guerrero, C. A., Suárez, J. M. y Gutiérrez, L. E. 2013. *Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web.* s.l. : Información tecnológica, 2013.

Guerrero, L.P. 2017. *Tecnología Investigación y Academia. Gestión en Proyectos de Software,*. 2017.

Huidobro. 2007. Tecnologías de información y comunicación. [En línea] 2007. <http://www.monografias.com/trabajos37/tecnologias-comunicacion/tecnologias-comunicacion.shtml>.

JetBrains, Team. 2016. MPS: Meta Programming System. [En línea] 2016. [Citado el: 2019 de enero de 12.] [https://www.jetbrains.com/pycharm/..](https://www.jetbrains.com/pycharm/)

Krajewski, L. J. y Ritzman, L. P. 2000. *Administración de operaciones: estrategia y análisis.* s.l. : Pearson educación, 2000.

Larman, Craig. 2010. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* s.l. : Prentice Hall, 2010. (Condri, 2012).

Las plataformas de aprendizajes, una. **Naranjo, F. A. y López, R. M. 2014.** 2, s.l. : Revista Cubana de Informática Médica, 2014, Vol. 6.

Lopez, P. y F., Ruaiz. 2011. *Lenguaje Unificado de Modelado - UML.* 2011.

Manso Guerra, Y, Cañizares Gonzalez, R. y Febles, J. 2016. *Diseño web adaptativo para la plataforma educativa ZERA.* s.l. : Revista Cubana de Ciencias, 2016. Vol. 10.

Martinez, R. 2010. PostgreSQL. [En línea] 2010. [Citado el: 9 de diciembre de 2018.] http://postgresql.org.es/sobre_postgresql.

Merino, M. Pérez, J. 2012. Definición de. [En línea] 2012. <https://definicion.de/lenguaje-de-programacion/>).

Merino, M. y Perez, J. 2012. Lenguaje de Programación. [En línea] 2012. <https://definicion.de/lenguaje-de-programacion/>.

Montero, S. I. 2012. Maestros del web. *Curso de Django para perfeccionistas.* [En línea] 1 de abril de 2012. [Citado el: 27 de septiembre de 2018.] <http://www.maestrosdelweb.com/guias/#guias-django>.

Montoya, C. E. Gomez, Uribe, Christian A. C. y Rodriguez, L. E. Sepúlveda. 2013. *Seguridad en la configuración del servidor web Apache.* 2013.

Olivares Lastra, R. 2010. Pruebas Alfa-Beta. [En línea] Septiembre de 2010. <https://rfolivares.wordpress.com/2010/09/08/pruebas-alfa-y-beta/>.

Paradigm, Visual. 2013. *Visual Paradigm for UML tool for software application development.* 2013.

Perez, D. E., y Paumier, A. A. 2015. *Testing como Práctica para Evaluar la Eficiencia en Aplicaciones Web.* s.l. : Revista Latinoamericana de Ingeniería de Software, 2015.

Pérez, J.E. 2014. Introducción a CSS. *Introducción a.* [En línea] 2014. [Citado el: 20 de enero de 2019.] <http://librosWeb.es/>.

Pressman, R. 2010. *Software Engineering: A practitioner's Approach. Seventh Edition.* New York. Estados Unidos : McGraw-Hill, 2010.

PyCharm, TEAM. 2016. [En línea] 26 de October de 2016. <https://www.jetbrains.com/pycharm/>.

- Ramirez, A. Ortiz. 2010.** *Python como primer lenguaje de programación.* Campus Estado de México : Publicación interna del Tecnológico de Monterrey, 2010.
- Redondo, J. M. Gonzales. 2017.** *Estudio con modelos de datos para la automatización.* 2017.
- Rodriguez, A. 2004.** *Medcampus: un proyecto de plataforma para la educación a distancia.* s.l. : Acimed, 2004.
- Rodriguez, J. S. 2009.** *Plataformas de enseñanza virtual para entornos educativos. Pixel-Bit.* s.l. : Revista de Medios y Educación, 2009.
- Rumbaugh, J. y Jacobson, I. 2001.** *El lenguaje Unificado de Modelado. Manual de Referencia.* 2001.
- Sanchez, T. Rodriguez. 2014.** *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana : s.n., 2014.
- Serna, E., Bedoya, i. A., y López, K. 2015.** *Nivel de madurez y grado de satisfacción de las herramientas libres para pruebas funcionales.* s.l. : Scientia et technica, 2015.
- Sevilla, V. D., Fernández, M. A., y Díaz., M. J. 2016.** *Introducción práctica a la programación con Python.* 2016.
- Significados. 2010.** "TIC (Tecnologías de la información y la comunicación)". [En línea] Significados.com, 2010. [Citado el: 23 de marzo de 2019.] <https://www.significados.com/tic/>.
- Sommerville, Ian. 2011.** *Technology & Engineering. Integrated requirements engineering.* s.l. : ISBN 9na edicion , 2011. 84-7829-074-5.
- UIV(Internet). 2018.** Universidad Internacional de Valencia. [En línea] 2018. <http://www.viu.es/caracteristicas-tipos-y-plataformas-mas-utilizadas-para-estudiar-adistancia/>.
- Valle, D. Pareja. 2018.** *Diseño e implementación de una aplicación móvil docente.* s.l. : Tesis Doctoral, 2018.
- Xauce, Zera. 2016.** ZERA. [En línea] Universidad de las Ciencias Informaticas, 20 de enero de 2016. [Citado el: 20 de enero de 2019.] <https://eva.uci.cu/es/aboutAs>.

Anexos

Anexo 1: Historias de usuario.

Tabla 8: Historia de usuario "Crear roles del sistema". (Elaboración propia).

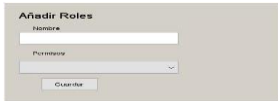
Historia de Usuario	
Número: HU_1	Nombre del requisito: Crear roles del sistema.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: un administrador crea roles de forma directa en la base de dato de la aplicación, estos roles serán asignados a usuarios registrados de la aplicación.	
Prototipo: 	

Tabla 9: Historia de usuario "Listar roles del sistema". (Elaboración propia).

Historia de Usuario	
Número: HU_2	Nombre del requisito: Listar roles del sistema.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores del sistema obtienen una lista de todos los roles que han sido creados y definidos por él y otros administradores, mediante la solicitud a la base de dato y respuesta rápida del sistema.	

Prototipo:

Lista de roles existentes	
Nombre	Permiso

Tabla 10: Historia de usuario "Actualizar roles de sistema". (Elaboración propia).

Historia de Usuario	
Número: HU_3	Nombre del requisito: Actualizar roles del sistema.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: el sistema se encarga de actualizar los roles asignados o creados por los administradores.	
Prototipo:	

Tabla 11: Historia de usuario "Eliminar roles del sistema". (Elaboración propia).

Historia de Usuario	
Número: HU_4	Nombre del requisito: Eliminar roles del sistema.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores eliminan de la base de datos los roles que ya no serán necesarios utilizar en el negocio de la aplicación.	

Prototipo:

¿Seguro desea eliminar el rol "...?"

Eliminar

Tabla 12: Historia de usuario "Crear permisos". (Elaboración propia).


Historia de Usuario	
Número: HU_5	Nombre del requisito: Crear permisos.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación muestra una vista a los administradores que les permite asignar permisos necesarios para determinados roles de la aplicación. Estos permisos son los que luego serán asignados a los usuarios.	
Prototipo:	
	

Tabla 13: Historia de usuario "Listar permisos". (Elaboración propia).

Historia de Usuario	
Número: HU_6	Nombre del requisito: Listar permisos.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación muestra una vista a los administradores con una lista de todos los permisos que fueron creado y a que roles fueron asignados, con el clic de un botón intuitivo a la acción.	

Prototipo:

Lista de permisos existentes		
Nombre	Tipo	Permiso
<div style="border: 1px solid #ccc; height: 20px; width: 100%;"></div>		

Tabla 14: Historia de usuario "Actualizar permisos". (Elaboración propia).


Historia de Usuario	
Número: HU_7	Nombre del requisito: Actualizar permisos.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación muestra una vista a los administradores que les permite encargarse de actualizar algún cambio necesario en el sistema con los roles que asignan, crean o eliminan, los cuales	
Prototipo:	
	

Tabla 15: Historia de usuario "Eliminar permisos". (Elaboración propia).

Historia de Usuario	
Número: HU_8	Nombre del requisito: Eliminar permisos.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación muestra una vista a los administradores que les permite eliminar permisos innecesarios.	

Prototipo:

¿Seguro desea eliminar el permiso "...?"

Eliminar

Tabla 16: Historia de usuario "Adicionar usuarios". (Elaboración propia).

Historia de Usuario	
Número: HU_9	Nombre del requisito: Adicionar usuarios.
Programador: Melissa Mujica Suárez	Usuario: usuario
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: una persona solicita registrarse en el sistema, luego se le muestra un formulario en el que tiene que rellenar los datos personales que se le solicita como necesarios (nombre, apellido, e-mail, usuario, contraseña, y biografía), luego confirmar salvándolo para comprobar los datos y evitar coincidencias en aspectos que son importantes y asegurar que sea humano. El sistema envía un código de confirmación a la dirección de correo especificada por el usuario, y este utiliza el código para terminar el formulario, después de comprobado guarda y se adiciona un usuario nuevo a la base de datos del sistema.	
Prototipo:	

Tabla 17: Historia de usuario "Listar usuarios". (Elaboración propia).

Historia de Usuario	
Número: HU_10	Nombre del requisito: Listar usuarios.
Programador: Melissa Mujica Suárez	Usuario: usuario
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los usuarios de la aplicación pueden obtener una lista de los usuarios de sus mismos roles, excepto los administradores no ven la lista de los administradores, esto con seleccionar un botón que sugiere esta función.	

Prototipo:

Tabla 18: Historia de usuario "Actualizar usuarios". (Elaboración propia).

Historia de Usuario	
Número: HU_11	Nombre del requisito: Actualizar usuarios.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores actualizan los usuarios que se eliminen o sufran algún cambios con la evolución del sistema.	
Prototipo:	

Tabla 19: Historia de usuario "Eliminar usuarios". (Elaboración propia).

Historia de Usuario	
Número: HU_12	Nombre del requisito: Eliminar usuarios.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores eliminaran usuarios en el sistema solicitados por los mismos por cualquier situación.	
Prototipo:	

Tabla 20: Historia de usuario "Crear mundo". (Elaboración propia).

Historia de Usuario	
Número: HU_13	Nombre del requisito: Crear mundo.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: un mundo es creado por el administrador, el profesor es quien edita su contenido, para lo cual el mismo debe llenar un formulario en el que complete todos los campos necesarios, luego de	

salvado y creado el mundo automáticamente, tendrá el estado de borrador, hasta que se le otorgue el estado de activo para ser visto por los estudiantes. Los mundos hacen referencia a temas de la asignatura de Teleinformática.

Prototipo:



Tabla 21: Historia de usuario "Listar mundos". (Elaboración propia).

Historia de Usuario	
Número: HU_14	Nombre del requisito: Listar mundo.
Programador: Melissa Mujica Suárez	Usuario: estudiante, profesores y administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los estudiantes observan una lista de los mundos disponibles bloqueados al inicio y luego se van a desbloquear a medida que avancen el juego. Los profesores tienen una lista de los mundos para editarlos mientras su estado no sea activo. Los mundos hacen referencia a temas de la asignatura de Teleinformática.	
Prototipo:	

Tabla 22: Historia de usuario "Actualizar mundo". (Elaboración propia).

Historia de Usuario	
Número: HU_15	Nombre del requisito: Actualizar mundo.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: el sistema actualiza los cambios realizados a los mundos por cualquier usuario de la aplicación. Los estudiantes van a desbloquear los mundos e interactúan en el mismo, y los profesores	

editan los mundos antes de ser lanzada la aplicación. Los mundos hacen referencia a temas de la asignatura de Teleinformática.

Prototipo:



Tabla 23: Historia de usuario "Eliminar mundo". (Elaboración propia).

Historia de Usuario	
Número: HU_16	Nombre del requisito: Eliminar mundo.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los profesores jefes de la asignatura pueden eliminar un mundo que ya no se quiera tener en el sistema solicitándolo a los administradores del sistema. Los mundos hacen referencia a temas de la asignatura de Teleinformática.	
Prototipo:	
<p>¿Seguro desea eliminar el mundo "..."? <input type="button" value="Eliminar"/></p>	

Tabla 24: Historia de usuario "Crear nivel". (Elaboración propia).

Historia de Usuario	
Número: HU_17	Nombre del requisito: Crear nivel.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los profesores son los que crean los niveles, solicitan al sistema agregar un nuevo nivel a un mundo ya creado, esta solicitud es hecha mediante en relleno de un formulario y la asignación de preguntas que responderán los estudiantes en el transcurso del juego. Los niveles hacen referencia a preguntas que tienen que resolver los estudiantes.	

Prototipo:

Prototipo de la interfaz de usuario para 'Añadir Nivel'. El formulario incluye los siguientes campos:

- Créditos:** Campo de entrada con un valor predeterminado de 0.
- Nombre del nivel:** Campo de texto.
- Estado de confección:** Menú desplegable.
- Tiempo de duración:** Menú desplegable.
- Orden en el mundo:** Campo de entrada con un valor predeterminado de 0.
- Mundo en el que está el nivel:** Menú desplegable.
- Descripción:** Campo de texto.
- Guardar:** Botón de acción.

Tabla 25: Historia de usuario "Listar nivel". (Elaboración propia).

Historia de Usuario											
Número: HU_18	Nombre del requisito: Listar nivel.										
Programador: Melissa Mujica Suárez	Usuario: administrador, profesor, estudiante										
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto										
Puntos estimados: 0.25	Puntos reales: 0.25										
<p>Descripción: los estudiantes cuentan con una lista de niveles del mundo en el que estén en ese momento, los cuales estarán bloqueados, y se desbloquean con la culminación de cada nivel anterior. Los profesores también tienen otra lista de los niveles que deben gestionar para el uso de los usuarios, y ven el estado en que se encuentra (<i>Borrador, Agendado, Listo, Activo</i>). Los niveles hacen referencia a preguntas que tiene que resolver los estudiantes.</p>											
<p>Prototipo:</p> <p>Prototipo de la interfaz de usuario para 'Lista de niveles existente'. Muestra una tabla con las siguientes columnas:</p> <table border="1"> <thead> <tr> <th>Nombre</th> <th>Créditos</th> <th>Mundo</th> <th>Descripción</th> <th>Estado</th> </tr> </thead> <tbody> <tr> <td colspan="5">[Contenido oculto por un elemento de desplazamiento]</td> </tr> </tbody> </table>		Nombre	Créditos	Mundo	Descripción	Estado	[Contenido oculto por un elemento de desplazamiento]				
Nombre	Créditos	Mundo	Descripción	Estado							
[Contenido oculto por un elemento de desplazamiento]											

Tabla 26: Historia de usuario "Actualizar nivel". (Elaboración propia).

Historia de Usuario	
Número: HU_19	Nombre del requisito: Actualizar nivel.
Programador: Melissa Mujica Suárez	Usuario: administrador, profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: la aplicación permite a los profesores actualizar modificaciones de los niveles aun en confección. Los niveles hacen referencia a preguntas que tiene que resolver los estudiantes.</p>	

Prototipo:



Tabla 27: Historia de usuario "Eliminar nivel". (Elaboración propia).

Historia de Usuario	
Número: HU_20	Nombre del requisito: Eliminar nivel.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite que los profesores jefes de asignatura eliminen los niveles que no son necesarios en el juego, para otra versión del juego, también por cualquier inconveniente con los mismos. Los niveles hacen referencia a preguntas que tiene que resolver los estudiantes.	
Prototipo: ¿Seguro desea eliminar el nivel "..."? <input type="button" value="Eliminar"/>	

Tabla 28: Historia de usuario "Listar opciones". (Elaboración propia).

Historia de Usuario	
Número: HU_21	Nombre del requisito: Listar opciones.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite a los usuarios con permisos obtener una lista de las opciones en base de dato.	

Prototipo:

Lista de opciones existente		
Pregunta	Nombre	Respuesta
< >		

Tabla 29: Historia de usuario "Eliminar opciones". (Elaboración propia).

Historia de Usuario	
Número: HU_22	Nombre del requisito: Eliminar opciones.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: las opciones son eliminadas por los profesores jefes de la asignatura o administradores del sistema.	
Prototipo:	
<p>¿Seguro desea eliminar la opción "..."? <input type="button" value="Eliminar"/></p>	

Tabla 30: Historia de usuario "Crear estados". (Elaboración propia).

Historia de Usuario	
Número: HU_23	Nombre del requisito: Crear estados.
Programador: Melissa Mujica Suárez	Usuario: administrador y profesor
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: un estado es confeccionado por los administradores, quienes dan las especificaciones a cada cual y otorgan para que clases responden esos estados. Los profesores son los que realizan el proceso de confección del sistema, van a cambiar los estados de cada clase en ese proceso a medida que los necesite. Un estado determina el avance de confección que tienen las clases mundo, nivel y pregunta.	

Prototipo:

Tabla 31: Historia de usuario "Listar estados". (Elaboración propia).

Historia de Usuario	
Número: HU_24	Nombre del requisito: Listar estados.
Programador: Melissa Mujica Suárez	Usuario: administrador y profesor
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: la aplicación permite listar los estados confeccionados a los administradores del sistema, y mostrar a los profesores el estado en el que está un mundo, nivel, o pregunta específica. Un estado determina el avance de confección que tienen las clases mundo, nivel y pregunta.</p>	
<p>Prototipo:</p>	

Tabla 32: Historia de usuario "Actualizar estados". (Elaboración propia).

Historia de Usuario	
Número: HU_25	Nombre del requisito: Actualizar estados.
Programador: Melissa Mujica Suárez	Usuario: administrador y profesor
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: los estados son actualizados por los administradores, si se confeccionara o eliminara un estado; y los profesores que actualizan el estado de confección que llevan las clases que los necesitan. Un estado determina el avance de confección que tienen las clases mundo, nivel y pregunta.</p>	

Prototipo:

Añadir Estados

Nombre

Tipo al que está asociado

Guardar

Tabla 33: Historia de usuario "Eliminar estados". (Elaboración propia).

Historia de Usuario	
Número: HU_26	Nombre del requisito: Eliminar estados.
Programador: Melissa Mujica Suárez	Usuario: administrador y profesor
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores eliminan los estados que no se usaran en la confección de mundos, niveles y preguntas. Un estado determina el avance de confección que tienen las clases mundo, nivel y pregunta.	
Prototipo:	
<p>¿Seguro desea eliminar el estado "..."? <input type="button" value="Eliminar"/></p>	

Tabla 34: Historia de usuario "Listar pregunta". (Elaboración propia).

Historia de Usuario	
Número: HU_28	Nombre del requisito: Listar preguntas.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los profesores tienen una lista de preguntas que han sido confeccionadas por sus colegas para evitar repeticiones agrupadas pro temáticas. Los estudiantes tienen una lista de las preguntas con las que han interactuado y sus respuestas; en el juego los estudiantes contestan de una en una las preguntas y pueden listar las anteriores e intentar de nuevo responder. Los administradores	

son los que seleccionan las preguntas que van en cada nivel, a partir de las temáticas que le fueron asignadas a la pregunta, y esta lista de todas las preguntas son las que visualizan los administradores.

Prototipo:



Tabla 35: Historia de usuario "Actualizar preguntas". (Elaboración propia).

Historia de Usuario	
Número: HU_29	Nombre del requisito: Actualizar preguntas.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los profesores actualizan las preguntas mientras las estén en confección, modifican su contenido.	
Prototipo:	

Tabla 36: Historia de usuario "Eliminar preguntas". (Elaboración propia).

Historia de Usuario	
Número: HU_30	Nombre del requisito: Eliminar preguntas
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: los administradores eliminan preguntas, que dejaron repetidas los profesores o que no se usaran más.	

Prototipo:

¿Seguro que desea eliminar la pregunta " ... " ?

Eliminar

Tabla 37: Historia de usuario "Crear notificaciones". (Elaboración propia).


Historia de Usuario	
Número: HU_31	Nombre del requisito: Crear notificaciones.
Programador: Melissa Mujica Suárez	Usuario: estudiante, profesor y administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: las notificaciones son clasificadas de tipo e-mail y de tipo sistema. Las notificaciones de e-mail son realizadas entre los usuarios de la aplicación, para comunicación entre ellos, el usuario solicita enviar correo, <i>Nuevo mensaje es redactado y enviado</i> luego al destinatario. Las notificaciones del sistema, son las que emite el mismo, en caso de errores cometido por los usuarios, advertencias de tiempo, entre otras, están definidas ya y fueron creadas por los administradores.</p>	
<p>Prototipo:</p> 	

Tabla 38: Historia de usuario "Listar notificaciones". (Elaboración propia).

Historia de Usuario	
Número: HU_32	Nombre del requisito: Listar notificaciones.
Programador: Melissa Mujica Suárez	Usuario: estudiante, profesor y administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: los administradores tienen una lista de las notificaciones de sistema creadas por ellos, y los usuarios tienen una lista de los mensajes que han sido enviados y recibidos en el uso de la aplicación.</p>	

Prototipo:

Tabla 39: Historia de usuario "Actualizar notificaciones". (Elaboración propia).

Historia de Usuario	
Número: HU_33	Nombre del requisito: Actualizar notificaciones.
Programador: Melissa Mujica Suárez	Usuario: estudiante, profesor y administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite la actualización de cambios de notificaciones del sistema realizados por los administradores y cambios realizados por los usuarios.	
Prototipo:	

Tabla 40: Historia de usuario "Eliminar notificaciones". (Elaboración propia).

Historia de Usuario	
Número: HU_34	Nombre del requisito: Eliminar notificaciones.
Programador: Melissa Mujica Suárez	Usuario: estudiante, profesor y administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite eliminar notificaciones del sistema a los administradores, por cambios que pueda surgir en el negocio del juego y eliminar e-mails a los usuarios de la aplicación que ya no quieran ver en sus vistas.	

Prototipo:

¿Seguro desea eliminar esta notificación "..."?

Eliminar

Tabla 41: Historia de usuario "Crear grupo de estudiantes". (Elaboración propia).


Historia de Usuario	
Número: HU_35	Nombre del requisito: Crear grupos de estudiantes.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: los profesores crean un grupo de estudiantes a los que ellos evaluarían en el transcurso del curso, estos grupos terminan su confección con la confirmación del estudiante de querer pertenecer a ese grupo, ya que el mismo puede escoger uno entre varios profesores que quieran que el estudiante sea parte de su grupo en el juego. Los estudiantes solo pueden pertenecer a un grupo.</p>	
<p>Prototipo:</p> 	

Tabla 42: Historia de usuario "Listar grupos de estudiantes". (Elaboración propia).

Historia de Usuario	
Número: HU_36	Nombre del requisito: Listar grupos de estudiantes.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
<p>Descripción: la aplicación permite listar los grupos que están creados. Solo será visto todos los grupos por los administradores y el profesor ver todos sus grupos.</p>	

Prototipo:

Tabla 43: Historia de usuario "Actualizar grupos de estudiantes". (Elaboración propia).

Historia de Usuario	
Número: HU_37	Nombre del requisito: Actualizar grupos de estudiantes.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite actualizar los grupos de los profesores, y actualizar el cambio de grupo de un estudiante.	
Prototipo:	

Tabla 44: Historia de usuario "Eliminar grupos de estudiantes". (Elaboración propia).

Historia de Usuario	
Número: HU_38	Nombre del requisito: Eliminar grupos de estudiantes.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite a los profesores eliminar su grupo de la aplicación o estudiantes en el grupo oprimiendo botones asociados a esta acción.	

Prototipo:

Lista de grupos existente	
Nombre	Estudiantes
< >	

Tabla 45: Historia de usuario "Listar acciones administrativas". (Elaboración propia).

Historia de Usuario	
Número: HU_39	Nombre del requisito: Listar acciones administrativas.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: el sistema permite listar las acciones administrativas realizadas en la aplicación.	
Prototipo:	

Tabla 46: Historia de usuario "Actualizar cambios generales en el sistema". (Elaboración propia).

Historia de Usuario	
Número: HU_40	Nombre del requisito: Actualizar cambios generales en el sistema.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: medio	Prioridad en el negocio: medio
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: el sistema permite actualizar cualquier cambio que se necesite como administrador o como el profesor.	
Prototipo:	

Tabla 47: Historia de usuario "Recuperar contraseña olvidada o caducada". (Elaboración propia).

Historia de Usuario	
Número: HU_41	Nombre del requisito: Recuperar contraseña olvidada o caducada.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25

Descripción: la aplicación permite a los usuarios recuperar su contraseña en caso de olvidarse o caducarse esto se realiza mediante el correo electrónico que los usuarios introducen en su perfil al crearse un usuario en la aplicación.

Prototipo:

Tabla 48: Historia de usuario "Crear comodines". (Elaboración propia).

Historia de Usuario	
Número: HU_42	Nombre del requisito: Crear comodines.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite a los administradores crear comodines que luego serán usados por los estudiantes en el transcurso de los niveles, aleatoriamente y acumulativos. Un comodín tiene varias clasificaciones e irán adquiriéndolos a medida de que avancen en el juego.	
Prototipo:	

Tabla 49: Historia de usuario "Listar comodines". (Elaboración propia).

Historia de Usuario	
Número: HU_43	Nombre del requisito: Listar comodines.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite a los usuarios listar los comodines que tienen sin usar en la <i>Tienda</i> y los administradores los existentes sin usar y los que están creados. Un comodín, luego de ser adquirido por el estudiante lo utilizara en la tienda para adquirir otras ventajas.	

Prototipo:

Lista de comodines existentes		
Nombre	Tipo	Estudiantes
[Empty table body]		


Tabla 50: Historia de usuario "Eliminar comodines". (Elaboración propia).

Historia de Usuario	
Número: HU_44	Nombre del requisito: Eliminar comodines.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite eliminar de la vista de los usuarios los comodines que uso, y adquirir otros en el juego.	
Prototipo:	
<p>¿Seguro desea eliminar el comodin "...?"</p> <p><input type="button" value="Eliminar"/></p>	

Tabla 51: Historia de usuario "Actualizar comodines". (Elaboración propia).

Historia de Usuario	
Número: HU_45	Nombre del requisito: Actualizar comodines.
Programador: Melissa Mujica Suárez	Usuario: administrador
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación permite a los administradores actualizar los comodines que los estudiantes ya han utilizados.	
Prototipo:	

*Tabla 52: Historia de usuario "Crear opciones". (Elaboración propia).***Historia de Usuario**

Número: HU_46	Nombre del requisito: Crear opciones.
Programador: Melissa Mujica Suárez	Usuario: profesor
Riesgo en Desarrollo: alto	Prioridad en el negocio: alto
Puntos estimados: 0.25	Puntos reales: 0.25
Descripción: la aplicación muestra las opciones de posibles respuestas a las preguntas elaboradas por el profesor, estas opciones son posibles respuestas que otorgan créditos a los estudiantes en dependencia de lo respondido. Las opciones son creadas por los profesores, y asignadas a preguntas en una misma temática. Las bonificaciones son dadas mediante puntos que se adquieren en el curso de juego.	
Prototipo:	
	

Anexo 2: Cuestionario para evaluar la satisfacción de SAATI.

Lee con cuidado y responde:

- 1. ¿Considera usted que es posible obtener mejores resultados en la asignatura Teleinformática sin una adecuada motivación de los estudiantes?**

Si____ No____ No sé____
- 2. ¿Utilizaría usted la aplicación propuesta a la hora de estudiar y aprender Teleinformática?**

Si____ No____ No sé____
- 3. ¿Satisface sus necesidades, según el rol que desempeña, en la docencia con respecto a la asignatura Teleinformática, el sistema propuesto?**

Si____ No____ No sé____
- 4. ¿Qué es lo que más te ayuda a estudiar Teleinformática?**
- 5. ¿Valora que te parece la aplicación propuesta?**

Anexo 3: Carta de aceptación del cliente.

ACTA DE ACEPTACIÓN

En cumplimiento del **Convenio de colaboración** establecido entre el **Departamento de Programación de la Facultad 1** y el estudiante **Rafael Angel Francés León** de la Facultad 1, ambos de la Universidad de las Ciencias Informáticas, se hace entrega de los productos que se relacionan a continuación:

- Aplicación web "**SAATI. Sistema Informático de apoyo al aprendizaje de la asignatura Teleinformática**".

La Parte Cliente, luego de haber revisado los productos de trabajo determina que se **Aceptan**.

Entrega

Recibe

Nombre y apellidos: **Rafael Angel Francés León**

Nombre y apellidos: **Ailyn Gutiérrez Ferrera**

Cargo: **Estudiante**

Cargo: **Jefe de Departamento**



Glosario de términos

Gamificación: es el empleo de mecánicas de juego en entornos y aplicaciones no lúdicas con el fin de potenciar la motivación, la concentración, el esfuerzo, la fidelización y otros valores positivos comunes a todos los juegos. Gamificar un entorno, consiste en convertir tareas monótonas o aburridas en otras más atractivas para el usuario.

E-learning (electronic learning): Término que cubre un amplio grupo de aplicaciones y procesos, tales como aprendizaje basado en Web, aprendizaje basado en computadora, aulas virtuales y colaboración digital. Incluye entrega de contenidos vía Internet, intranet/extranet, audio y videgrabaciones, transmisiones satelitales, TV interactiva, CD-ROM y más.

Entorno virtual de aprendizaje: es un espacio educativo en la web, un conjunto de herramientas informáticas que posibilita la interacción didáctica de manera que el alumno pueda llevar a cabo las labores propias de la docencia, como leer documentación, realizar ejercicios, formular preguntas, entre otras.