Universidad de Cienfuegos "Carlos Rafael Rodríguez"
Facultad de Ingeniería
Carrera de Ingeniería Informática

"Informatización del proceso de confección de planes de impartición y planes calendario docentes en la Universidad de Cienfuegos"

Trabajo de diploma para optar por el título de Ingeniería Informática

Autor:

Leonel Portela Cano

Tutores:

Ing. Boris Pérez Cañedo

Ing. Yuviny Echevarría Cartaya

Cienfuegos, Cuba

Curso 2012 – 2013

Declaración de autoría

Yo, Leonel Portela Cano, declaro que soy el único autor de este trabajo y autorizo al Departamento de Informática de la Facultad de Ingeniería en la Universidad de Cienfuegos “Carlos Rafael Rodríguez”, para que hagan el uso que estimen pertinente con el trabajo de diploma.

Para que así conste firmo (firmamos) la presente a los días del mes de del 2013.

_____

Autor: Leonel Portela Cano

_____

Tutor: Ing. Yuviny Echevarría Cartaya

Ing. Boris Pérez Cañedo

Los abajo firmantes certificamos que el presente trabajo ha sido revisado según acuerdo de la dirección de nuestro centro y el mismo cumple los requisitos que debe tener un trabajo de esta envergadura referente a la temática señalada.

_____ _____

Firma ICT Firma Firma Vicedecano

AGRADECIMIENTOS:

Agradecimientos especiales a mis padres que me brindan su amor y apoyo incondicional en todo momento.

Parte importante y fundamental para realizar esta tesis han sido mis tutores Yuviny Echevarría Cartaya y Boris Pérez Cañedo ellos son excelentes personas que han compartido su tiempo conmigo dándome su ayuda y conocimientos.

A mis amigos que son uno de los tesoros más grandes de la vida y son los que en las buenas acuden si son llamados y en las malas vienen solitos.

Agradezco a todas las personas en general que han contribuido a la realización de esta investigación

DEDICATORIA:

Dedico esta tesis a lo que más quiero en el mundo mi familia…

Resumen:

El siguiente trabajo propone mediante un sistema informático dar solución al problema de confección de los documentos Plan de Impartición y Plan Calendario basándose en la planificación hecha en la Universidad de Cienfuegos, estos documentos actualmente se realizan de forma manual lo cual implica una mala gestión de la información ya que se deben tener en cuenta gran cantidad de datos y recursos a la hora de confeccionarlos, debido a este trabajo manual es posible que no se realice de forma óptima la distribución de los recursos. Para resolver este problema se realizan múltiples tareas entre ellas la recopilación de información necesaria para estructurar los documentos. Para hacer el sistema primeramente se le da entrada a los documentos confeccionados por el Departamento de Planificación Docente para ello se utilizaron diferentes tipos de herramientas y tecnologías informáticas con el fin de que el trabajo sea novedoso y lo más eficiente posible. En el primer capítulo se explica detalladamente en que consiste el problema de confección de los documentos para tener un mejor conocimiento sobre el tema, además se señalan las herramientas a utilizar y los software existentes como base y guía del trabajo. En el capítulo 2 se enfatiza en la metodología Scrum con el objetivo de comprender mejor el problema y contribuir al desarrollo ágil del sistema generando poca documentación dando una solución en un corto período de tiempo. En el capítulo 3 se elaboraron varios diagramas referentes al diseño del sistema, se establecieron estándares de interfaz y se realizó un estudio sobre la factibilidad del sistema, aplicando para ello el método de estimación basado en casos de uso todo esto contribuye al desarrollo correcto de la aplicación.

Índice:

INTRODUCCIÓN:..... 1

CAPÍTULO 1 – FUNDAMENTACIÓN TEÓRICA 7

1.1 - Introducción: 7

1.2 - Conceptos asociados al dominio del problema: 7

1.2.1 - ¿Qué es gestión?..... 7

1.2.2 - Gestión de la información 7

1.2.3 - Planificación..... 8

1.2.4 - Recursos involucrados en el proceso de planificación..... 8

1.2.5 - Documentos involucrados en el proceso de planificación..... 9

1.3 - Contexto Internacional, Nacional y Local: 10

1.3.1 - Internacional 10

1.3.2 - Nacional:.....11

1.4 - Lenguajes y tecnologías utilizadas: 12

1.4.1 - Tendencias, metodologías y tecnologías actuales.....12

1.5 - Conclusiones del Capítulo: 26

CAPÍTULO 2 – DESCRIPCIÓN Y ANÁLIZIS DE LA SOLUCIÓN PROPUESTA.....28

2.2 - Ejemplos de Metodologías Ágiles.....28

2.2.1 - Programación Extrema (Extreme Programming, XP). 28

2.2.2 - *ICONIX.* 30

2.2.3 - Selección de la metodología.....31

2.2.4 - Comparación entre metodologías ágiles y tradicionales. 32

2.3 - Características de Scrum: 33

2.4 - Prácticas de la metodología Scrum. 35
2.4.1 - Revisión de las iteraciones: 35
2.4.2 - Desarrollo incremental: 35
2.4.3 - Desarrollo evolutivo: 35
2.4.4 - Auto-organización: 35
2.4.5 - Colaboración: 36
2.4.6 - Visión general del proceso: 36
2.5 - Equipo Scrum: 36
2.5.1 - Pila del Producto. 36
2.5.2 - Pila del Producto: 37
2.5.3 - Requerimientos no funcionales. 40
2.5.4 - Planeación de los Sprints. 42
2.5.5 - Pila de Sprint 43
2.5.6 - Técnica de estimación de Sprint. 44
2.6 - Conclusiones del Capítulo: 47
CAPÍTULO 3: CONSTRUCCIÓN Y PRUEBA DEL SISTEMA 49
3.1 - Introducción 49
3.2 - Descripción del Objeto de Automatización 49
3.3 - Actores del sistema 50
3.4 - Diagrama de casos de uso del sistema: 50
3.5 - Diagrama de Casos de Uso 51
3.6 - Descripción textual de los actores del sistema: 51
3.7 - Modelo de Base de Datos Diagrama 1 62
3.8 - Principios de diseño del sistema 64

3.8.1 - Estándares en la interfaz de la aplicación. 64
3.8.2 - Concepción general de la ayuda. 64
3.8.3 - Tratamiento de excepciones. 64
3.9 - Estudio de Factibilidad. 65
3.9.1 - Cálculo de Puntos de Casos de Uso sin ajustar. 65
3.9.2 - Descripción tipo de actores y su factor de peso 66
3.9.3 - Especificación de tipos de actores del sistema 66
3.9.4 - Factor de Peso de los Casos de Uso sin ajustar (UUCW). 67
3.9.5 - Descripción de tipos de casos de uso de acuerdo a su complejidad 67
3.9.6 - Especificación de complejidad de los casos de uso del sistema ...67
3.6.7 - Cálculo de Puntos de Casos de Uso ajustados. 69
3.6.8 - Factor de complejidad técnica (TCF). 69
3.6.9 - Factor de ambiente (EF). 70
3.6.10 - Estimación del esfuerzo 72
3.6.11 - Duración: 72
3.6.12 - Criterio de distribución de esfuerzo 72
3.6.13 - Cálculo de costos 73
3.6.14 - Beneficios tangibles e intangibles 73
3.6.15 - Análisis de costos y beneficios. 73
3.10 - Escenarios de Pruebas para el Sistema. 74
3.11 - Conclusiones del Capítulo: 76
CONCLUSIONES GENERALES 77
REFERENCIAS BIBLIOGRÁFICAS: 78
BIBLIOGRAFÍA: 81

ANEXOS: 85

Índice de Tabla:

Tabla 1 Equipo Scrum 36

Tabla 2 Pila del Producto 37

Tabla 3 Planeación de los Sprints.....42

Tabla 4 Velocidad Estimada (sprint 1) 45

Tabla 5 Velocidad Estimada (sprint 2) 46

Tabla 6 Velocidad Estimada (sprint 3) 46

Tabla 7 Velocidad Estimada (sprint 4) 47

Tabla 8 Descripción Actores del Sistema.....51

Tabla 9 Gestionar Tipo de Clase 52

Tabla 10 Gestionar Plan de Impartición 53

Tabla 11 Gestionar Plan Calendario.....54

Tabla 12 Gestionar Profesor.....55

Tabla 13 Gestionar Grupo.....56

Tabla 14 Gestionar Semestre 57

Tabla 15 Gestionar Tema.....58

Tabla 16 Gestionar Actividad.....58

Tabla 17 Gestionar Asignatura 59

Tabla 18 Gestionar Departamento 60

Tabla 19 Descripción tipo de actores y factor de peso 66

Tabla 20 Especificación de tipos de actores del sistema 66

Tabla 21 Descripción de tipos de casos de uso de acuerdo a su complejidad67

Tabla 22 Especificación de complejidad de los casos de uso del sistema 67

Tabla 23 Factor determinante de complejidad técnica del sistema.....69

Tabla 24 Factor externo o ambiente.....71

Tabla 25 Criterio de distribución de esfuerzo.....72

INTRODUCCIÓN:

La tecnología a lo largo de los años se ha desarrollado a niveles exponenciales y ha jugado un papel fundamental dentro de la sociedad facilitando y permitiendo que se realicen trabajos de gran complejidad, el desarrollo tecnológico es de gran importancia para el desarrollo de cualquier país del mundo por ello uno de los principales objetivos es desarrollar la mayor cantidad de industrias posibles para hacerlas más eficientes y brinden beneficios al país, la informatización es muy importante por lo que debe alcanzar cada sector de producción y desarrollo, apoyarse en aplicaciones realizadas por computadoras brinda una mejor gestión y comprensión de la información permitiendo dar soluciones acertadas a distintos tipos de problemas que resultarían muy complejos o de larga duración si se resolvieran de forma manual.

El amplio desarrollo de las TIC ha posibilitado que la síntesis, recuperación y presentación de la información sea de la más variada forma, acercándose cada vez más a las actuales tecnologías. El tratamiento de la información se realiza hoy en día más rápida y oportunamente, de ahí la importancia a escala internacional del crecimiento de estas tecnologías.

Se denominan medios de comunicación a los medios desarrollados a partir de las TIC. Estos medios se encargan del estudio, desarrollo, implementación, almacenamiento y distribución de la información haciendo uso de hardware y software. En general se trata del empleo de las computadoras y las aplicaciones informáticas para la gestión, difusión y localización de los datos necesarios para cualquier actividad humana.

Las TIC no son ninguna panacea ni fórmula mágica, pero pueden mejorar la vida de todos los habitantes del planeta. Se disponen de herramientas para llegar a los Objetivos de Desarrollo del Milenio, de instrumentos que harán avanzar la causa de la libertad y la democracia, y de los medios necesarios para propagar los conocimientos y facilitar la comprensión mutua.

Cuba no se abstiene en lo que a las TIC, su uso y aprovechamiento se refiere, se han pensado, diseñado e implementado los más disímiles mecanismos para contribuir de una forma u otra a un mayor y mejor uso de las mismas. Uno de los puntos más fuertes, en lo que a la fomentación y avance del uso de las TIC se refiere, lo constituyen los procesos de planificación que se desarrollan en todo tipo de centros, empresas e instituciones a lo largo y ancho de la isla.

El Ministerio de Educación Superior es uno de los pilares más fuertes en la organización y planificación. Específicamente en la Universidad de Cienfuegos existe un grupo de personas encargadas de estos sectores estructurados por niveles en todos los departamentos. Estos se encargan de organizar, dirigir y controlar todo tipo de actividades.

Los problemas concernientes a la organización han estado presentes en todos los niveles desde la industria hasta la educación, recientemente se han desarrollados muchos sistemas informáticos con el propósito de resolver estos problemas, estas aplicaciones se han insertado para organizar el trabajo fundamentalmente en el área estudiantil con el objetivo de perfeccionar el proceso docente educativo.

Debido a la diversificación de la enseñanza estudiantil es necesario tener en cuenta varios documentos por lo que uno de los puntos que necesita gran planificación y organización es la elaboración de Planes de Impartición y Planes Calendario, aquí intervienen diferentes factores a tener en cuenta por lo que se debe hacer una adecuada distribución de estos factores para evitar conflictos entre ellos.

El problema de confección de Planes de Impartición y Planes Calendario es un problema que requiere el uso de determinada cantidad de información, el proceso se lleva acabo al inicio de cada semestre cuando el departamento de planificación define los requisitos docentes, según el Plan de Estudios, el fondo de horas declarado, el número de asignaturas y de semanas que lo componen para cada curso de cada facultad. Las facultades a su vez, solicitan a los

departamentos las asignaturas a impartir según Plan de Estudios. El jefe del departamento asigna el profesor a la asignatura y este último conforma el programa de la asignatura y el plan calendario del profesor (P1) de acuerdo a las orientaciones dadas por el Plan de Estudios. Cada P1 contiene una determinada asignatura con su cantidad de horas, semestre en el cual se va a impartir, tipo de clase y tiene delimitados todos los temas que la compone.

El departamento de planificación recibe la información antes mencionada y previa revisión de que cumplan con los requisitos preestablecidos, conforma una plantilla consistente en un arreglo bidimensional de asignaturas por las columnas y semanas por las filas.

El presente informe de resultados está estructurado del siguiente modo, Introducción, Capítulos I, II y III; Conclusiones, Recomendaciones, Bibliografía y Anexos correspondientes.

En el Capítulo I. Fundamentación Teórica: fundamentan y analiza críticamente los principales conceptos vinculados al campo de acción; así como los antecedentes y sistemas informáticos vinculados a la gestión de convenios. Se documenta el estudio realizado sobre las tendencias, tecnologías y herramientas actuales en lo cual se fundamenta la selección de las utilizadas en la solución propuesta.

Capítulo II. Descripción y Construcción de la Solución Propuesta: fundamenta como guía para el desarrollo del software la metodología Scrum. Se completan la pila del producto y los requerimientos no funcionales, se realiza la planeación de cada sprint, así como las técnicas de estimación de estos.

Capítulo III. Historias Técnicas y Construcción de Escenarios de Prueba: En este capítulo se presenta el diagrama de casos de uso del sistema, el modelo lógico y físico de la base de datos, así como los principios seguidos para el desarrollo del sistema y su implementación. Además se realiza el análisis de factibilidad y la creación de escenarios de prueba para la aplicación

Antecedentes:

En los últimos años se ha trabajado mucho en problemas de organización académica en distintas áreas de la educación, se ha tratado la confección de calendarios de exámenes, asignación de estudiantes a grupos y otros problemas. En todos los casos se elaboraron sistemas adaptados a las características de cada institución, estos estudios se han ido desarrollando en distintas universidades entre las cuales se encuentran la Universidad de Holguín, también se han realizado estudios semejantes en la Universidad Central de Las Villas y IPVCE "Vladimir Ilich Lenin" con aplicaciones muy específicas para sus realidades.

Justificación del Estudio:

En el país la enseñanza superior se ha desarrollado considerablemente diversificándose y aumentando la calidad de los graduados esto trae consigo un aumento del número de estudiantes, unido a la diversificación de las actividades docentes, que incluye distintas modalidades tales como presenciales, no presenciales, a distancia, etc., debido a esto es necesario realizar un proceso de organización de las actividades docentes, en extremo dinámico y altamente susceptible a irregularidades e incoherencias que afectan la calidad de la educación. En el proceso de organización y planificación la mala elaboración de documentos como son planes de impartición y Planes calendarios constituyen algunos de los puntos que influyen una planificación con errores y hacen que sea un proceso complejo, lento y propenso a fallas de todo género.

Situación Problémica:

Actualmente los actores del proceso de confección de planes de impartición y planes calendario docentes, a decir: vicedecanos docentes, jefes de departamento, jefes de carreras y profesores no poseen una documentación estandarizada para la elaboración de los documentos bajo sus responsabilidades y/o utilizan formatos y documentos diferentes a los establecidos. Además no existe la coordinación necesaria para efectuar las

actividades de este proceso. Tampoco se controla qué actividad o actividades del proceso están detenidas y quién o quiénes son los responsables de esta situación.

Problema de Investigación:

¿Cómo lograr la informatización del proceso de confección de planes de impartición y planes calendario docentes, en la Universidad de Cienfuegos?

Idea a defender:

La informatización de la gestión del proceso de confección de planes de impartición y planes calendario docentes en la Universidad de Cienfuegos propiciará que las actividades de este proceso se ejecuten de manera coordinada, transparente y de acuerdo con los documentos rectores.

Objeto de Estudio:

El proceso de planificación docente en la Universidad de Cienfuegos.

Campo de Aplicación:

La confección de planes de impartición y planes calendario en la Universidad de Cienfuegos.

Objetivo General:

Desarrollar un sistema informático para la gestión del proceso de confección de planes de impartición y planes calendario docentes en la Universidad de Cienfuegos.

Objetivos específicos:

-Analizar el proceso de confección de planes de impartición y planes calendario docentes en la Universidad de Cienfuegos.

-Diseñar el sistema informático.

-Implementar el sistema informático.

-Evaluar el sistema en escenarios de pruebas.

Tareas Investigativas:

-Recopilar información y recomendaciones con los trabajadores del departamento de planificación sobre el proceso de planificación docente en la Universidad de Cienfuegos.

-Diseño de la base de datos del sistema informático.

-Estimación del esfuerzo de desarrollo del sistema informático.

-Codificación del sistema informático.

-Crear escenarios de prueba para el sistema informático.

Preguntas de Investigación:

¿Cuáles son las características del proceso de planificación docente en la Universidad de Cienfuegos?

¿Cómo elaborar un sistema informático para la organización de la información relacionada con los documentos vinculados al proceso de planificación docente en la Universidad de Cienfuegos?

Aporte Práctico:

-El sistema informático, permitiendo la confección de planes de impartición y planes calendarios en la universidad de Cienfuegos.

CAPÍTULO 1 – FUNDAMENTACIÓN TEÓRICA

1.1 - Introducción:

En este capítulo se tratan conceptos y teorías esenciales para trabajar la temática que describe el contexto donde se desarrolla el problema a resolver, aquí se busca y analiza la información relacionada al objeto de estudio, sistemas existentes asociados al campo de acción, procesos a automatizar y tendencias y tecnologías a utilizar en la elaboración del sistema.

1.2 - Conceptos asociados al dominio del problema:

1.2.1 - ¿Qué es gestión?

Gestión: Organización, utilización y aprovechamiento de determinados dispositivos y recursos. [1]

1.2.2 - Gestión de la información

La Gestión de la Información, (GI), es un conjunto de procesos por los cuales se controla el ciclo de vida de la información, desde su obtención - por creación o captura, hasta su disposición final - archivada o eliminada. Los procesos también comprenden la extracción, combinación, depuración y distribución de la información a los interesados. Los objetivos de la Gestión de la Información es garantizar la integridad, disponibilidad y confidencialidad de la información. Para desarrollar una correcta gestión de la información es necesario tener en cuenta una serie de pasos, entre los que se encuentran los siguientes:[2]

- Determinar la información que se precisa.
- Recoger y analizar la información.
- Registrarla y recuperarla cuando sea necesaria.
- Utilizarla.

- Divulgarla.

1.2.3 - Planificación

¿Qué es planificar?

La planificación es el proceso metódico diseñado para obtener un objetivo determinado. Otras definiciones, más precisas, incluyen "La planificación es un proceso de toma de decisiones para alcanzar un futuro deseado, teniendo en cuenta la situación actual y los factores internos y externos que pueden influir en el logro de los objetivos" (Jiménez, 1982)

Existen diferentes herramientas y técnicas para abordar la planificación de un proyecto, las cuales permiten definir el curso de acción a seguir, que será tomado como base durante la ejecución del mismo.

Si bien la planificación define las acciones a seguir, durante la ejecución puede existir necesidad de cambios respecto de lo definido originalmente, los mismos servirán de punto de partida para un nuevo análisis y una nueva planificación de ser requerido.[3]

1.2.4 - Recursos involucrados en el proceso de planificación

El Proceso de Planificación consiste en la asignación de unos recursos a otros. Estos recursos pueden ser: humanos, docentes, espaciales y de tiempo.

Recursos humanos

La proyección de docentes no difiere del planeamiento de recursos humanos de las grandes organizaciones. Como en éstas, su objetivo es brindar información sobre la cantidad de empleados que se necesitarán, sus aptitudes, la cantidad (y capacidad) existente en la actualidad, su rotación, su estructura y su formación.

Recursos docentes

Dentro de los recursos docentes se encuentran los conceptos de Año Académico, grupo o brigada y estudiante. En este caso se hará referencia al grupo, ya que en la UCF se realiza la planificación por grupo.

Recursos espaciales

Estos tienen como objetivo brindar información acerca de su distribución capacidad y clasificación.

Recursos de tiempo

Estos tienen como objetivo brindar información acerca de una actividad determinada, la cual tiene un tiempo de duración.

1.2.5 - Documentos involucrados en el proceso de planificación

La Planificación está formada por un grupo de documentos que son la base y la guía para realizar una buena organización. Entre estos documentos se encuentran: Plan de Estudio, Orientaciones del curso, Plan de Impartición, Plan Calendario, Balance de Carga y los Horarios Docentes.

Plan de Estudio

Un Plan de Estudio es un “conjunto de enseñanzas y prácticas que, con determinada disposición, han de cursarse para cumplir un ciclo de estudios u obtener un título”.

CARACTERÍSTICAS:

- Los planes de estudios se organizan en *materias*, que pueden distribuirse en *asignaturas*.
- En los planes de estudios puede establecerse una *ordenación tempora*l en el aprendizaje de determinadas materias, mediante:

Orientaciones del Curso

Es el documento con el cuál se definen y orientan la planificación de una serie de actividades que deben realizarse, así como la fecha de inicio y fin del curso y la cantidad de semanas del mismo.

Plan de Impartición

Es el documento por el cual las facultades se rigen para darle a conocer la distribución de los profesores por grupo y asignatura.

Plan Calendario

Es el documento en el cual el profesor se rige para darle a conocer la distribución de actividades de una asignatura.

1.3 - Contexto Internacional, Nacional y Local:

1.3.1 - Internacional

1.3.1.1 - Rediseño de proceso mediante un enfoque de red conversacional- Caso planificación docente en la Universidad Talca:

La Universidad de Talca es una de las dieciséis universidades del Consorcio de Universidades Estatales de Chile, la cual pertenece al Concejo de Rectores de las Universidades de Chilenas. Esta universidad necesita resolver el problema de asignar salas y confeccionar documentos a todas las actividades docentes sin superposiciones. Se trata de determinar, entre el modelo actual y el rediseño, el ahorro de esfuerzo en horas hombre, que permita concluir si es factible implementar una mejora en los subprocesos del Proceso General de Planificación Docente. La intención es aportar una herramienta que permita modelar y proponer un rediseño de la actual estructura para una mejor distribución de los recursos en función de los distintos factores involucrados en el problema. En realidad el desafío está en cómo mejorar el mecanismo actual de asignación de recursos a las actividades, para que estas funcionen apropiadamente.[4]

Muchas Universidades han hechos sus estudios e implementaciones para dar soluciones a sus problemas de organizaciones . La Universidad de Valencia, España, por ejemplo, resalta con una serie de trabajos publicados sobre el tema por el Dr. Enric Crespo (R. Álvarez-Valdés et al. 2005)(Ramón Alvarez-Valdés et al. 2000)(Ramón Alvarez-Valdés et al. 2002)(Ramón Alvarez-Valdés et al. 2001)(Alvarez-Valdés et al. 1997), en los mismos se muestran resultados positivos en problemas de grandes dimensiones.[5]

1.3.2 - Nacional:

1.3.2.1 - Diseño de un sistema para gestionar la información del Proceso de Planificación Docente en las facultades de la Universidad de las Ciencias Informáticas: En la Facultad de Economía de la Universidad de las Ciencias Informáticas (UCI), existe un proyecto de maestría en gestión de información, coordinado por la profesora Elena Font. Como parte de este programa la Ing. Lucia Rodríguez García ha propuesto el trabajo titulado *Diseño de un sistema para gestionar la información del Proceso de Planificación Docente en las facultades de la Universidad de las Ciencias Informáticas.* En este trabajo se propone optimizar el proceso de planificación docente en las facultades de la UCI, diseñando un sistema para gestionar la información.

1.3.2.2 - Sistema Docente Integral (SDI)

Sistema creado en el Instituto Superior Politécnico “José Antonio Echeverría” (CUJAE) por el Grupo de Gestión Universitaria Digital dirigido a aumentar la eficiencia de la gestión universitaria, específicamente el control de los procesos docentes, su planificación y resultados.[6]

“De forma general el objetivo del sistema es registrar, procesar y crear mecanismos de recuperación de la información relacionada con el proceso docente de forma automatizada”.

“El SDI está conformado actualmente por 8 módulos y se trabaja para aumentar sus funcionalidades. Los módulos que están diseñados son: 1) Secretaría de

Facultades, 2)Matrícula, 3)Estadística, 4)Archivo Histórico, 5)Recuperación de Información, 6)Pago de Estipendios, 7)Planificación Docente y 8) Postgrado.

Cada módulo del SDI se desarrolló, con vista a su trabajo en red, sobre una base de datos centralizada que almacena la información generada desde diferentes áreas como: secretarías docentes, economía, Archivo Central, vicerrectorías de investigaciones y postgrado, entre otras. [7]

1.3.2.3 - Sistema de Gestión de la Nueva Universidad (SIGENU)

Este programa, elaborado sobre la base del software libre por especialistas de la Ciudad Universitaria José Antonio Echeverría (CUJAE), nació con el objetivo de lograr un solo producto de automatización de los procesos docentes para todos los centros universitarios.

SIGENU permite registrar los datos de los estudiantes desde el momento de la matrícula hasta que se gradúan o causan baja definitiva, incluyendo bajas temporales, licencia, repitencia, reportes evaluativos, cambio del lugar de residencia, etc.

Según Cedeño este sistema cuenta con dos tipos de aplicaciones: uno de gestión que facilita la captura de la información para el trabajo diario en las secretarías docentes de las universidades y otro que brinda soportes para la toma de decisiones. O sea que de manera ágil y eficiente el mecanismo permite llegar al usuario a la información cuando la necesite, mediante consultas y gráficos dinámicos, con una descripción detallada de la historia del estudiante durante la carrera, cuyos informes permanecen en una base de datos.[6]

1.4 - Lenguajes y tecnologías utilizadas:

1.4.1 - Tendencias, metodologías y tecnologías actuales

Actualmente en el mundo se utilizan un amplio conjunto de técnicas para la construcción de aplicaciones las que tienen sus ventajas y desventajas. La

tendencia fundamental en el mercado de software es la implementación de aplicaciones que interactúan o comparten información en Internet.

A la hora de seleccionar la tecnología y el lenguaje de programación para el desarrollo de la aplicación se debe tener en cuenta los objetivos específicos que se propone la aplicación, el tipo de plataforma informática con que se cuenta en la institución.

Para desarrollar de una forma coherente y ordenada este trabajo se ha utilizado la metodología RUP y el Lenguaje Unificado de Modelado (UML) para el diseño y análisis de software. En los epígrafes a continuación se comenta sobre las diferentes herramientas para el desarrollo de aplicaciones Web, como las tecnologías, lenguajes de programación y sistemas gestores de bases de datos. Posteriormente se explican cuáles fueron los seleccionados para el desarrollo de este trabajo y por qué.

1.4.1.1 - UML "Lenguaje Unificado de Modelado"

Una exigencia de la gran mayoría de instituciones dentro de su Plan Informático estratégico, es que los desarrollos de software bajo una arquitectura en Capas, se formalicen con un lenguaje estándar y unificado. Se necesita que cada una de las partes que comprende el desarrollo de todo software de diseño orientado a objetos, se visualice, especifique y documente con lenguaje común. Para ello se necesitaba un lenguaje que fuese gráfico, a fin de especificar y documentar un sistema de software, de un modo estándar incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema. Este lenguaje unificado que cumple con estos requerimientos, es ciertamente UML, el cual cuenta con una notación estándar y semánticas esenciales para el modelado de un sistema orientado a objetos.[8]

1.4.1.2 - Lenguaje Unificado de Modelado

(LUM o **UML**, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la

actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no especifica en sí mismo qué metodología o proceso usar.

UML no puede compararse con la programación estructurada, pues UML significa Lenguaje Unificado de Modelado, no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que, programación estructurada, es una forma de programar como lo es la orientación a objetos, sin embargo, la programación orientada a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML sólo para lenguajes orientados a objetos.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.[8]

1.4.1.3 - Utilidad de UML

UML es un lenguaje para modelamiento de propósito general evolutivo, ampliamente aplicable, dable de ser soportado por herramientas e industrialmente estandarizado. Se aplica a una multitud de diferentes tipos de sistemas, dominios, y métodos o procesos.

- Como lenguaje de *propósito general*, se enfoca en el corazón de un conjunto de conceptos para la adquisición, compartición y utilización de conocimientos emparejados con mecanismos de extensión.
- Como un lenguaje para modelamiento *ampliamente aplicable*, puede ser aplicado a diferentes tipos de sistemas (software y no - software), dominios (negocios versus software) y métodos o procesos.
- Como un lenguaje para modelamiento *soportable por herramientas*, las herramientas ya están disponibles para soportar la aplicación del lenguaje para especificar, visualizar, construir y documentar sistemas.
- Como un lenguaje para modelamiento *industrialmente estandarizado*, no es un lenguaje cerrado, propiedad de alguien, sino más bien, un lenguaje abierto y totalmente extensible reconocido por la industria.

UML posibilita la captura, comunicación y nivelación de conocimiento estratégico, táctico y operacional para facilitar el incremento de valor, aumentando la calidad, reduciendo costos y reduciendo el tiempo de presentación al mercado; manejando riesgos y siendo proactivo para el posible aumento de complejidad o cambio[9]

1.4.1.4 - NetBeans

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados *módulos*. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

- Administración de las interfaces de usuario (ej. menús y barras de herramientas)
- Administración de las configuraciones del usuario
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato)
- Administración de ventanas
- Framework basado en asistentes (diálogos paso a paso)

El IDE NetBeans es un entorno de desarrollo integrado - una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas.

Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

El NetBeans IDE es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring.

NetBeans IDE 6.5, la cual fue publicada el 19 de noviembre de 2008, extiende las características existentes del Java EE (incluyendo Soporte a Persistencia, EJB 3 y JAX-WS). Adicionalmente, el NetBeans Enterprise Pack soporta el desarrollo de Aplicaciones empresariales con Java EE 5, incluyendo herramientas de desarrollo visuales de SOA, herramientas de esquemas XML, orientación a web servicies (for BPEL), y modelado UML. El NetBeans C/C++ Pack soporta proyectos de C/C++, mientras el PHP Pack, soporta PHP 5.

Modularidad. Todas las funciones del IDE son provistas por módulos. Cada módulo provee una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, permitiéndole al usuario comenzar a trabajar inmediatamente.[10]

1.4.1.5 - Plataforma Java

La **plataforma Java** es el nombre de un entorno o plataforma de computación originaria de Sun Microsystems, capaz de ejecutar aplicaciones desarrolladas usando el lenguaje de programación Java u otros lenguajes que compilen a bytecode y un conjunto de herramientas de desarrollo. En este caso, la plataforma no es un hardware específico o un sistema operativo, sino más bien una máquina virtual encargada de la ejecución de las aplicaciones, y un conjunto de bibliotecas estándar que ofrecen una funcionalidad común.

La plataforma es así llamada la plataforma Java (antes conocida como Plataforma Java 2), e incluye:

- Plataforma Java, Edición Estándar (Java Platform, Standard Edition), o Java SE (antes J2SE)
- Plataforma Java, Edición Empresa (Java Platform, Enterprise Edition), o Java EE (antes J2EE)
- Plataforma Java, Edición Micro (Java Platform, Micro Edition), o Java ME (antes J2ME)

Java Platform, Enterprise Edition o **Java EE** (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4), es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java con arquitectura de N capas distribuidas y que se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma Java EE está definida por una *especificación*. Similar a otras especificaciones del Java Community Process, Java EE es también considerada informalmente como un estándar debido a que los proveedores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son *conformes a Java EE*; estandarizado por The Java Community Process / JCP.

Java EE incluye varias especificaciones de API, tales como JDBC, RMI, e-mail, JMS, Servicios Web, XML, etc y define cómo coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servlets, portlets (siguiendo la especificación de Portlets Java), JavaServer Pages y varias tecnologías de servicios web. Ello permite al desarrollador crear una Aplicación de Empresa portable entre plataformas y escalable, a la vez que integrable con tecnologías anteriores. Otros beneficios añadidos son, por ejemplo, que el servidor de

aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componentes desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de en tareas de mantenimiento de bajo nivel.[11][12]

1.4.1.6 - Apache Tomcat

Apache Tomcat (también llamado **Jakarta Tomcat** o simplemente **Tomcat**) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de javaserver.page (JSP) de Sun Microsystems

Tomcat es mantenido y desarrollado por miembros de la Apache Software Foundation y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la *Apache Software Licence*. Las primeras distribuciones de Tomcat fueron las versiones 3.0.x. Las versiones más recientes son las 7.x, que implementan las especificaciones de Servlet 3.0 y de JSP 2.2. A partir de la versión 4.0, Jakarta Tomcat utiliza el contenedor de servlets Catalina.

Tomcat es un servidor web con soporte de servlets y JSPs. Tomcat no es un servidor de aplicaciones, como JBoss o JOnAS. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad.

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java.

Características del producto:

Tomcat 3.x (distribución inicial)

- Implementado a partir de las especificaciones Servlet 2.2 y JSP 1.1
- Recarga de servlets
- Funciones básicas HTTP

Tomcat 4.x

- Implementado a partir de las especificaciones Servlet 2.3 y JSP 1.2
- Contenedor de servlets rediseñado como Catalina
- Motor JSP rediseñado con Jasper
- Conector Coyote
- Java Management Extensions (JMX), JSP Y administración basada en Struts

Tomcat 5.x

- Implementado a partir de las especificaciones Servlet 2.4 y JSP 2.0
- Recolección de basura reducida
- Capa envolvente nativa para Windows y Unix para la integración de las plataformas
- Análisis rápido JSP

Tomcat 6.x

- Implementado de Servlet 2.5 y JSP 2.1
- Soporte para *Unified Expression Language* 2.1
- Diseñado para funcionar en Java SE 5.0 y posteriores
- Soporte para Comet a través de la interfaz CometProcessor

Tomcat 7.x

- Implementado de Servlet 3.0 JSP 2.2 y EL 2.2
- Mejoras para detectar y prevenir "fugas de memoria" en las aplicaciones web
- Limpieza interna de código
- Soporte para la inclusión de contenidos externos directamente en una aplicación web[13]

1.4.1.7 - Maven:

Maven es una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002. Es similar en funcionalidad a Apache Ant (y en menor medida a PEAR de PHP y CPAN de Perl), pero tiene un modelo de configuración de construcción más simple, basado en un formato XML. Estuvo integrado inicialmente dentro del proyecto Jakarta pero ahora ya es un proyecto de nivel superior de la Apache Software Foundation.

Maven utiliza un Project Object Model (POM) para describir el proyecto de software a construir, sus dependencias de otros módulos y componentes externos, y el orden de construcción de los elementos. Viene con objetivos predefinidos para realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado.

Una característica clave de Maven es que está listo para usar en red. El motor incluido en su núcleo puede dinámicamente descargar plugins de un repositorio, el mismo repositorio que provee acceso a muchas versiones de diferentes proyectos Open Source en Java, de Apache y otras organizaciones y desarrolladores. Este repositorio y su sucesor reorganizado, el repositorio Maven 2, pugnan por ser el mecanismo *de facto* de distribución de aplicaciones en Java, pero su adopción ha sido muy lenta. Maven provee soporte no sólo para obtener archivos de su repositorio, sino también para subir artefactos al

repositorio al final de la construcción de la aplicación, dejándola al acceso de todos los usuarios. Una caché local de artefactos actúa como la primera fuente para sincronizar la salida de los proyectos a un sistema local.

Maven está construido usando una arquitectura basada en plugins que permite que utilice cualquier aplicación controlable a través de la entrada estándar. En teoría, esto podría permitir a cualquiera escribir plugins para su interfaz con herramientas como compiladores, herramientas de pruebas unitarias, etcétera, para cualquier otro lenguaje. En realidad, el soporte y uso de lenguajes distintos de Java es mínimo. Actualmente existe un plugin para .Net Framework y es mantenido, y un plugin nativo para C/C++ fue alguna vez mantenido por Maven.

Convención sobre configuración

La filosofía general de Maven es la estandarización de las construcciones generadas por seguir el principio de Convención sobre Configuración, a fin de utilizar existentes modelos en la producción de software. Esta estrategia necesariamente restringe ampliamente la variabilidad, lo que se refleja en la exhortación de Maven a adherirse a su modelo de proyecto Mientras que Maven es adecuado para nuevos proyectos, los proyectos complejos ya existentes pueden ser simplemente no adaptables para que utilicen Maven. La falta de restricciones de la convención de capas del proyecto fue hecha de alguna manera más configurable con el lanzamiento de Maven 2.

Ciclo de vida

Las partes del ciclo de vida principal del proyecto Maven son:

1. Compile
2. Test
3. Package
4. Install
5. deploy

La idea es que, para cualquier meta, todas las metas anteriores han sido completadas satisfactoriamente. Por ejemplo, cuando se ejecuta mvn install Maven verificará si mvn package ha sido ejecutado exitosamente (el archivo jar existe en target/), en cuyo caso no será ejecutado otra vez.[14]

1.4.1.8 - AndroMDA

AndroMDA (pronunciado "Andromeda") es un programa informático de tipo framework, de generación extensible de código, que se adhiere al paradigma de la arquitectura dirigida por modelos. Los modelos de las herramientas de UML son transformados en componentes desplegables para su plataforma preferida (J2EE, Spring Framework, .NET). Al contrario de otros entornos de desarrollo MDA (Model Driven Architecture), incluye un conjunto de cartuchos enfocados a los kits de desarrollo actuales como son Apache Axis, jBPM, Apache Struts, JSF, Spring e Hibernate. También incluye un Kit para desarrollar tus propios cartuchos generadores de código o personalizar los existentes - el cartucho Meta. Utilizándolo, puedes construir tu propio generador de código empleando una herramienta de UML.

Es un proyecto de código abierto que está bajo la Licencia BSD. Debido a que su generador de código soporta plataformas actuales, se ha convertido en la principal herramienta de código abierto de MDA para el desarrollo de aplicaciones empresariales.

El proyecto AndroMDA tiene una comunidad alrededor del mundo que ayuda a su liberación rápida con una retroalimentación notable por parte de los usuarios. Este proyecto sigue el modelo bazar ya que sigue la principal premisa del mismo que sólo puede darse en un entorno de libertad, cooperación, comunidad y disponiendo del código abierto. La comunidad de desarrollo de este proyecto promueve el concienciar a la gente de que el software es un servicio y no un bien adquirible.[15][12]

1.4.1.9 - Visual Parading

Visual Paradigm es una herramienta CASE: Ingeniería de Software Asistida por Computadora. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación.

Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta privada disponible en varias ediciones, cada una destinada a unas necesidades: Enterprise, Professional, Community, Standard, Modeler y Personal. Existe una alternativa libre y gratuita de este software, la versión Visual Paradigm UML 6.4 Community Edition (ojo, la versión Community Edition, ya que existe la Enterprise, Professional, etc). Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos.[15]

1.4.1.10 - PostgreSQL

Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyada por organizaciones comerciales. Dicha comunidad es denominada el PGDG (*PostgreSQL Global Development Group*).

Características:

Algunas de sus principales características son, entre otras:

Amplia variedad de tipos nativos

PostgreSQL provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas).
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arrays.

Adicionalmente los usuarios pueden crear sus propios tipos de datos, los que pueden ser por completo indexables gracias a la infraestructura GiST de PostgreSQL. Algunos ejemplos son los tipos de datos GIS creados por el proyecto PostGIS

Otras características

- Claves ajenas también denominadas Llaves ajenas o Claves Foráneas (*foreign keys*).
- Disparadores (*triggers*): Un disparador o *trigger* se define como una acción específica que se realiza de acuerdo a un evento, cuando éste ocurra dentro de la base de datos. En PostgreSQL esto significa la ejecución de un procedimiento almacenado basado en una determinada acción sobre una tabla específica. Ahora todos los disparadores se definen por seis características:
    - El nombre del disparador o *trigger*
    - El momento en que el disparador debe arrancar
    - El evento del disparador deberá activarse sobre...
    - La tabla donde el disparador se activará
    - La frecuencia de la ejecución
    - La función que podría ser llamada

Entonces combinando estas seis características, PostgreSQL le permitirá crear una amplia funcionalidad a través de su sistema de activación de disparadores (*triggers*).

- Vistas.
- Integridad transaccional.
- Herencia de tablas.
- Tipos de datos y operaciones geométricas.
- Soporte para transacciones distribuidas. Permite a PostgreSQL integrase en un sistema distribuido formado por varios recursos (p.ej, una base de datos PostgreSQL, otra Oracle, una cola de mensajes IBM MQ JMS y un ERP SAP) gestionado por un servidor de aplicaciones donde el éxito ("commit") de la transacción globlal es el resultado del éxito de las transacciones locales.

Funciones

Bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional.[9]

1.5 - Conclusiones del Capítulo:

En el capítulo se han expuesto teorías y conceptos fundamentales para la comprensión del problema, en él se describe el proceso de confección de planes de impartición y planes calendario en la Universidad de Cienfuegos, se resaltó como es necesario tener un mejor control y organización de la información en la Universidad de Cienfuegos con el fin de dar una solución lo más aceptada posible que permita la gestión de los documentos relacionados con la confección de planes de Impartición y planes calendario , se trataron las distintas

tecnologías y lenguajes a tener en cuenta para desarrollar el trabajo asiendo que este sea lo más preciso posible.

CAPÍTULO 2 – DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

2.1 - Introducción:

En el capítulo se dará a conocer como se desarrollara el sistema y todos los elementos que conformarán este proceso. Se abordará la metodología utilizada y las ventajas que proporciona para el desarrollo de la investigación. Primeramente se seleccionara el equipo de trabajo, se definirán los roles que jugará cada miembro y se asignará las tareas que debe desempeñar cada uno, como parte importante se puntualizan las funcionalidades del sistema apoyados en la Pila del Producto siendo este un artefacto generado por la metodología Scrum, siendo esta la metodología seleccionada, una vez definidos los requisitos funcionales se definirán los requisitos no funcionales del sistema, también se realizarán los Sprint y se realizará la Pila del Sprint que consiste en ubicar las funcionalidades en cada etapa o Sprint, a la Pila del Sprint se le agregan las funcionalidades basándose en las técnicas de estimación de los Sprint, se utilizará la técnica llamada cálculos de velocidad, esta técnica nos permite saber cuánto podemos hacer en el próximo Sprint basándonos en lo que se ha hecho en el anterior. Todos estos elementos servirán como base para el desarrollo del sistema.

2.2 - Ejemplos de Metodologías Ágiles.

2.2.1 - Programación Extrema (Extreme Programming, XP).

Es la más destacada de los procesos ágiles de desarrollo de software formulada por Kent Beck. La programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.

Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Las características fundamentales del método son:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone mayor calidad del código escrito de esta manera ,el código es revisado y discutido mientras se escribe, es más importante que la posible pérdida de productividad inmediata.
- Frecuente interacción del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- Refactorización del código, es decir, rescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender

cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que en más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.[16]

2.2.2 - *ICONIX.*

El proceso de ICONIX maneja casos de uso, como el RUP, pero le falta mucho para llegar al nivel del RUP. También es relativamente pequeño y firme, como XP, pero no desecha el análisis y diseño que hace XP. Este proceso también hace uso aerodinámico del UML mientras guarda un enfoque afilado en el seguimiento de requisitos.

Y, el proceso se queda igual a la visión original de Jacobson del manejo de casos de uso, esto produce un resultado concreto, específico y casos de uso fácilmente entendible, que un equipo de un proyecto puede usar para conducir el esfuerzo hacia un desarrollo real. La Figura 6 muestra el cuadro del proceso. El diagrama retrata la esencia del enfoque aerodinámico al desarrollo del software, que incluye un juego mínimo de diagramas de UML y algunas valiosas técnicas que se toman de los casos del uso para codificar rápida y eficazmente. El enfoque es flexible y abierto; siempre se puede seleccionar de los otros aspectos del UML para complementar los materiales básicos.[17]

2.2.3 - Selección de la metodología.

La selección de una metodología específica para el desarrollo de una herramienta informática ya sea ágil o tradicional depende de varios factores a tener en cuenta, entre estos se encuentran los estatutos de desarrollos definidos en la empresa o equipo de trabajo, tiempo requerido para el de desarrollo del producto, tamaño del equipo de trabajo, estabilidad de los requisitos así como el nivel de integración del cliente con el equipo de desarrollo. Actualmente la industria del software ha alcanzado un nivel de desarrollo inmenso, que ha avanzado a pasos agigantados debido al avance de las tecnologías informáticas y de la comunicación a nivel mundial, esto requiere la realización de productos de software utilizables que respondan a los estándares de calidad establecidos para la comercialización y distribución de los mismos, que materialicen los intereses y necesidades de los usuarios finales en el menor tiempo posible a fin de lograr la satisfacción de los clientes y ganar en cuanto a demanda y competitividad en el mercado actual. En respuesta a estas necesidades es que han surgido las metodologías ágiles, las cuales responden para una mayor productividad a entornos de trabajo con un equipo de desarrollo pequeño y en el cual el cliente mantenga una relación con el equipo durante el proceso de desarrollo. Posibilitan centrarse más en el desarrollo de productos altamente calificados y de alta calidad mediante el cumplimiento de metas, planificación y cumplimiento de iteraciones, en vez de enfocarse en la documentación extensiva.

Tomando en consideración lo antes expuesto las características del equipo de desarrollo del software y el tiempo requerido para el desarrollo del mismo se decide utilizar SCRUM como metodología a seguir para la implementación del sistema por ser una metodología de desarrollo muy simple, que trabaja de forma iterativa, de manera que en cada iteración se vaya mejorando la arquitectura del proyecto y se logre acercarse más al resultado esperado, permite la adaptación continua a las circunstancias de la evolución del proyecto ya que toma en

cuenta los cambios que puedan ir ocurriendo a medida que se vaya desarrollando para de esta manera mejorar u optimizar la arquitectura del mismo a la vez que se va construyendo.

2.2.4 - Comparación entre metodologías ágiles y tradicionales.

A continuación se muestra una tabla comparativa con las principales características de éstas metodologías. Esta información aportará criterios útiles para la selección.[18]

Metodología	Características	Fases	Herramientas Case
RUP	•Dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. •Utiliza UML como lenguaje de modelado. •Amplia documentación. •Es una metodología tradicional.	•Inicio. •Elaboración. •Construcción. •Transmisión.	Racional Rose. Visual Paradigm.
MSF	•Es una serie de modelos que pueden adaptarse a cualquier proyecto de tecnología de información.	•Visión y Alcances. •Planificación. •Desarrollo. •Estabilización.	No Mencionado.

	•Es una Metodología tradicional.		
XP	•Es una metodología ágil. •Realimentación continúa entre el cliente y el equipo de desarrollo. •Adecuada para proyectos con requisitos imprecisos, muy cambiantes, y donde existe un alto riesgo técnico.	•Exploración. •Planificación de la entrega. •Iteraciones. •Producción. •Mantenimiento y Muerte del Proyecto.	Sprintometer.
SCRUM	•Metodología ágil. •Especialmente indicada para proyectos con un rápido cambio de requisitos.	•Planeamiento. •Arquitectura o diseño de alto nivel.	Sprintometer.

2.3 - Características de Scrum:

Scrum es un modelo de referencia que define un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto. Los roles principales en Scrum son el

ScrumMaster, que mantiene los procesos y trabaja de forma similar al director de proyecto, el *ProductOwner*, que representa a los *stakeholders* (interesados externos o internos), y el *Team* que incluye a los desarrolladores.

Durante cada *sprint*, un periodo entre una y cuatro semanas (la magnitud es definida por el equipo), el equipo crea un incremento de software *potencialmente entregable* (utilizable). El conjunto de características que forma parte de cada sprint viene del *Product Backlog*, que es un conjunto de requisitos de alto nivel priorizados que definen el trabajo a realizar. Los elementos del *Product Backlog* que forman parte del sprint se determinan durante la reunión de *Sprint Planning*. Durante esta reunión, el *Product Owner* identifica los elementos del *Product Backlog* que quiere ver completados y los hace del conocimiento del equipo. Entonces, el equipo determina la cantidad de ese trabajo que puede comprometerse a completar durante el siguiente sprint. Durante el sprint, nadie puede cambiar el Sprint Backlog, lo que significa que los requisitos están congelados durante el sprint.

Scrum permite la creación de equipos autoorganizados impulsando la co-localización de todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto.

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan (a menudo llamado *requirements churn*), y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, Scrum adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido, y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes.

Existen varias implementaciones de sistemas para gestionar el proceso de Scrum, que van desde notas amarillas "post-it" y pizarras hasta paquetes de software. Una de las mayores ventajas de Scrum es que es muy fácil de aprender, y requiere muy poco esfuerzo para comenzarse a utilizar.[19]

2.4 - Prácticas de la metodología Scrum.

2.4.1 - Revisión de las iteraciones:

Al finalizar cada iteración (sprint) se lleva a cabo una revisión con todas las personas implicadas en el proyecto. Es por tanto la duración del sprint, el periodo máximo que se tarda en reconducir una desviación en el proyecto o en las circunstancias del producto.

2.4.2 - Desarrollo incremental:

Las personas implicadas no trabajan con diseños o abstracciones. El desarrollo incremental implica que al final de cada iteración se dispone de una parte de producto operativa, que se puede inspeccionar y evaluar. [20]

2.4.3 - Desarrollo evolutivo:

Los modelos de gestión ágil se emplean para trabajar en entornos de incertidumbre e inestabilidad de requisitos. Intentar predecir en las fases iniciales cómo será el resultado final, y sobre dicha predicción desarrollar el diseño y la arquitectura del producto no es realista, porque las circunstancias obligarán a remodelarlo muchas veces.

¿Para qué predecir los estados finales de la arquitectura o del diseño si van a estar cambiando? Scrum considera a la inestabilidad como una premisa, y se adoptan técnicas de trabajo para permitir la evolución sin degradar la calidad de la arquitectura que también evoluciona durante el desarrollo.

Durante el desarrollo se genera el diseño y la arquitectura final de forma evolutiva. Scrum no los considera como productos que deban realizarse en la primera “fase” del proyecto. (El desarrollo ágil no es un desarrollo en fases). [20]

2.4.4 - Auto-organización:

En la ejecución de un proyecto son muchos los factores impredecibles en todas las áreas y niveles. La gestión predictiva confía la responsabilidad de su

resolución al gestor de proyectos. En Scrum los equipos son auto-organizados (no auto-dirigidos), con margen de decisión suficiente para tomar las decisiones que consideren oportunas. [19]

2.4.5 - Colaboración:

Las prácticas y el entorno de trabajo ágiles facilitan la colaboración del equipo. Ésta es necesaria, porque para que funcione la auto organización como un control eficaz cada miembro del equipo debe colaborar de forma abierta con los demás, según sus capacidades y no según su rol o su puesto.[20]

2.4.6 - Visión general del proceso:

Scrum denomina “sprint” a cada iteración de desarrollo y según las características del proyecto y las circunstancias del sprint puede determinarse una duración desde una hasta dos meses, aunque no suele ser recomendable hacerlos de más de un mes

2.5 - Equipo Scrum:

Tabla 1 Equipo Scrum

ScrumMaster	Propietario del Producto	Equipo
Boris Pérez Cañedo	Yuviny Echevarría Cartaya	Leonel Portela Cano Yuvini Echevarria Cartaya

2.5.1 - Pila del Producto.

La Pila de Producto, o Product Backlog, es un artefacto del marco de trabajo para la gestión agile de proyectos de desarrollo de software, SCRUM. Y que es, en líneas generales, una lista ordenada u priorizada de las tareas que componen un proyecto de aplicación, la división en tareas implica una organización del

trabajo y una primera visión del alcance del proyecto. Es decir, qué es lo que se quiere obtener después de semanas o meses de trabajo.

La Pila de Producto, según Scrum, es propiedad del Dueño del Producto, teniendo en cuenta que es un artefacto vivo y que va a sufrir modificaciones tanto de prioridad como de alcance durante todo el proyecto, se construye una primera Pila de Producto que comprenda la descripción de las funcionalidades esperadas a alto nivel.[21]

La estrategia empleada por la gestión ágil para estimar el esfuerzo puede ser:

-No empeñarse en estimaciones precisas.

-Estimar con la técnica "juicio de expertos".
-En el proyecto se ha escogido la primera opción ya que no existe un personal calificado como experto para llevar a cabo la estimación.

Para establecer la prioridad de cada funcionalidad se han escogido valores en una escala de 1 a 5 donde el 1 significa máxima prioridad y 5 mínima prioridad. Para la estimación del esfuerzo inicial se usará una escala de 1 a 10 donde 10 significa mayor esfuerzo y 1 poco esfuerzo.

2.5.2 - Pila del Producto:

Tabla 2 Pila del Producto

Elemento	Prioridad	Descripción	Estimación Inicial (Puntos de Historia)
Autenticar Usuario	**1**	Se visualiza un formulario que contiene los campos usuario y contraseña en caso de llenarlos con los datos correctos se entra al sistema	**2**

Salir del Sistema	**5**	Se visualiza la opción de salir del sistema en caso de seleccionarla se abandona el programa.	**1**
Gestionar Profesor (Buscar, Crear, Actualizar, Eliminar Profesor)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Profesor mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**
Gestionar Grupo (Buscar, Crear, Modificar, Eliminar Grupo)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Profesor mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**
Gestionar Semestre (Buscar, Crear, Modificar, Eliminar Semestre)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Profesor mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**

Gestionar Plan de Impartición	**1**	Se procede a Buscar, Crear, Actualizar y Eliminar Planes de Impartición, en caso de haber problemas con la gestión se mostrara un mensaje de error.	**10**
Gestionar Tipo de Clase (Buscar, Crear, Actualizar, Eliminar Tipo de Clase)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Tipo de Clase mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**
Gestionar Tema (Buscar, Crear, Actualizar, Eliminar Tema)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Tema mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**
Gestionar Actividad (Buscar, Crear, Actualizar, Eliminar Actividad)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos de la Actividad visualizando formularios con los campos correspondientes	**6**

		en caso de fallo se mostrará un mensaje de error.	
Gestionar Asignatura (Buscar, Crear, Actualizar, Eliminar Tema)	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos de la asignatura mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**
Gestionar Plan Calendario	**1**	Se procede a Buscar, Crear, Actualizar y Eliminar Planes Calendario, en caso de haber problemas con la gestión se mostrara un mensaje de error.	**10**
Gestionar Departamento	**2**	Se procede a Buscar, Crear, Actualizar y Eliminar los datos del Departamento mediante formularios con todos sus datos en caso de fallo se mostrará un mensaje de error.	**6**

2.5.3 - Requerimientos no funcionales.

Requerimientos de Interfaz.

-Utilizar en la interfaz gráfica y los mensajes de la aplicación en idioma Español.

-La ejecución de la aplicación y la introducción de datos deben ser posibles mediante el uso del teclado y/o el Mouse.

-Controlar en la medida de lo posible que los mensajes de error sean emitidos desde la aplicación y no por el sistema operativo.

-La interfaz debe ser diseñada respetando los parámetros de diseño (colores, tipografía, logos).

-Todas las salidas del programa respetarán el formato de los documentos oficiales utilizados hasta el momento.

Requerimientos de Usabilidad.

-Los usuarios del sistema serán los diferentes especialistas vinculados a las acciones referentes al proceso de planificación que se lleva a cabo en la Universidad de Cienfuegos.

Requerimientos de Soporte.

-Los servicios de instalación y mantenimiento del sistema deberán realizarse por personal calificado, teniendo en cuenta las configuraciones necesarias para su correcto funcionamiento. El sistema debe propiciar su mejoramiento y la anexión de otras opciones que se le incorporen en un futuro.

Requerimientos Legales.

-La herramienta propuesta responde a los intereses de la oficina de Planificación de la Universidad de Cienfuegos.

Requerimientos de Confiabilidad.

-Mantener un registro histórico al pasar de un año a otro (almacenando hasta 5 años de información).

Requerimientos de software. La aplicación debe poderse ejecutar en entornos Windows y/o Linux (Multiplataforma).

Requerimiento de Hardware. Se requiere de una máquina que funcione como servidor de aplicaciones y de base de datos (Pentium 4 1 Ghz 256 Mb Ram 40 Gb disco duro).

2.5.4 - Planeación de los Sprints.

-Para la planeación de los Sprint se deben desarrollar los siguientes aspectos:

-Una meta de Sprint.

-Una lista de miembros (y su nivel de dedicación, si no es del 100%).

-Una Pila de Sprint (lista de historias incluidas en el Sprint)

-Una fecha concreta para la Demo del Sprint.

-Un lugar y momento definidos para el Scrum Diario.

Tabla 3 Planeación de los Sprints

Sprint	**Duración (días)**	**Participantes**	**Scrum Diario**		**Factor de Dedicación**
			Lugar	**Hora**	
Sprint 1	15	Leonel Portela Cano Yuvini Echevarria Cartaya	Oficina de Yuvini	8:45 am	0.7
Sprint 2	20	Leonel Portela Cano Yuvini Echevarria Cartaya	Oficina de Yuvini	8:45 am	0.7
Sprint 3	30	Leonel Portela Cano Yuvini Echevarria Cartaya	Oficina de Yuvini	8:45 am	0.7
Sprint 4	30	Leonel Portela Cano Yuvini Echevarria Cartaya	Oficina de Yuvini	8:45 am	0.7

2.5.5 - Pila de Sprint

Sprint	Pila de Sprint	Metas	Fecha de entrega
Sprint 1	**1-Autenticar Usuario** **3-Salir del Sistema**	Lograr autenticar usuarios con motivo de seguridad para el sitio.	5-3-2013
Sprint 2	**1-Gestionar Profesor** **2-Gestionar Grupo** **3-Gestionar Semestre** **4-Gestionar Departamento**	Se quiere poder gestionar toda la información referente a los Profesores, Grupos, Semestres permitiendo poder entrar datos al sistema modificándolos o eliminándolos según se estime, una vez realizado esto apoyándose en la información previamente almacenada se confeccionará el Plan de Impartición.	28-3-2013
Sprint 3	**1-Gestionar Plan de Impartición** **2-Gestionar Tipo de Clase.** **3-Gestionar Tema**	Se quiere poder gestionar toda la información referente al Plan de Impartición, los Tipos de Clases existentes y los diferentes Temas, permitiendo buscar actualizar y eliminar todos sus datos.	20-4-2013

Sprint 4	**1-Gestionar Actividad** **2-Gestionar Asignatura** **3-Gestionar Plan Calendario**	Trabajar con todos los datos de las Actividades, Asignaturas y gestionar el Plan Calendario creando, actualizando y eliminando sus datos según se estime conveniente.	22-5-2013

2.5.6 - Técnica de estimación de Sprint.

Existen varias técnicas para la estimación de los Sprint, con estas lo que se persigue es incluir en cada Sprint una serie de "Historias de Usuario" o sea elementos de la Pila del Producto que el equipo pueda desarrollar en el tiempo planificado para el Sprint.[19]

Las técnicas son: Ojo de buen cubero. Cálculos de velocidad.

Ojo de buen cubero: Esta técnica se basa específicamente en indagar con los miembros del equipo cuanto serían capaces de hacer en un Sprint o sea cuantas "Historias de Usuario" son capaces de realizar completamente en el Sprint.

Analizando la prioridad de cada elemento de la Pila del Producto se logran incluir los elementos de la pila del Sprint. Se dice que esta técnica funciona bien para equipos pequeños y Sprint corto.

Cálculos de Velocidad: Es una técnica que se emplea para ubicar los elementos de la Pila del Producto en la de Sprint basado en el nivel de dedicación que tenga cada miembro del equipo en la realización de sus tareas. Se dice que es recomendable para equipos donde las personas no se conocen o son nuevas, establecer un Factor de Dedicación de un 70%. El cálculo de la velocidad es un proceso que se realiza cada vez que se pretende comenzar un Sprint para el cual se tomaran los datos del Sprint anterior como son Factor de dedicación del último Sprint y Velocidad Real.

Factor de Dedicación del Último Sprint. (1) FACTOR DE DEDICACIÓN = (VELOCIDAD REAL)

(DIAS-HOMBRE DISPONIBLES)

La velocidad real es la suma de las estimaciones iniciales (Puntos de historia), que se completaron en el último Sprint.

Para calcular la velocidad se utilizaría la siguiente fórmula.

(2) VELOCIDAD ESTIMADA = (DÍAS-HOMBRE DISPONIBLES) X (FACTOR DE DEDICACIÓN).

Para confeccionar los Sprint se toma la técnica del cálculo de velocidad la cual nos ayuda a confeccionar los sprints de una forma más ajustada a la realidad basándose en el trabajo del sprint previo.

Sprint 1.

Aplicando fórmulas (1) y (2):

FACTOR DE DEDICACIÓN (sprint 0) = 0.7

VELOCIDAD ESTIMADA (sprint 1) = 15 x 0,7 (Puntos de Historia).

Tabla 4 Velocidad Estimada (sprint 1)

Participantes	**Días-Hombres (disponibles)**	**Velocidad**		**Factor de dedicación**
		Estimada	Real	Total**:**0.2
Leonel Portela	10	10.5	3	0.1
Boris Pérez Cañedo	5			0.1

Sprint 2.

Aplicando fórmulas (1) y (2):

FACTOR DE DEDICACIÓN (sprint 1) = 3 = 0.2 =0.1

15 2

VELOCIDAD ESTIMADA (sprint 2) = 20 x 0.1 (Puntos de Historia).

Tabla 5 Velocidad Estimada (sprint 2)

Participantes	**Días-Hombres (disponibles)**	**Velocidad**		**Factor de dedicación**
		Estimada	Real	Total**:**1.05
Leonel Portela	15	2	21	0.52
Boris Pérez Cañedo	5			0.52

Sprint 3.

Aplicando fórmulas (1) y (2):

FACTOR DE DEDICACIÓN (sprint 2) = 21 =1,05 =0.52

20 2

VELOCIDADESTIMADA (sprint 3) = 30 x 0.52 (Puntos de Historia).

Tabla 6 Velocidad Estimada (sprint 3)

Participantes	**Días-Hombres (disponibles)**	**Velocidad**		**Factor de dedicación**
		Estimada	Real	Total**:**1.43
Leonel Portela	25	15.6	43	0.71

Boris Pérez Cañedo	5			0.71

Sprint 4.

Aplicando fórmulas (1) y (2):

FACTOR DE DEDICACIÓN (sprint 3) = 43 =1.43 =0.71

30 2

VELOCIDADESTIMADA (sprint 4) = 30 x 0.71 (Puntos de Historia).

Tabla 7 Velocidad Estimada (sprint 4)

Participantes	**Días-Hombres (disponibles)**	**Velocidad**		**Factor de dedicación**
		Estimada	Real	Total**:**2.36
Leonel Portela	25	21.3	71	1.18
Boris Pérez Cañedo	5			1.18

Aplicando fórmulas (1) y (2):

FACTOR DE DEDICACIÓN (sprint 4) = 71 =2.36 =1.18

30 2

2.6 - Conclusiones del Capítulo:

En el capítulo se han detallado partes esenciales del sistema a desarrollar resaltando sus funcionalidades y dando a conocer cómo va a estar estructurado su ciclo de realización destacando las personas que intervendrán en él y que papel desempeñaran. La metodología Scrum cumplió con las expectativas inmediatas de todo el equipo, primero reducir la documentación asociada al

sistema y luego realizar la implementación del sistema en un corto período de tiempo. Mediante esta metodología se pudo comprender mejor el problema a resolver y contribuye al desarrollo ágil del sistema.

CAPÍTULO 3: CONSTRUCCIÓN Y PRUEBA DEL SISTEMA

3.1 - Introducción

A la hora de crear un sistema uno de los principales elementos a tener en cuenta es el diseño, cuyo propósito primario es decidir cómo será implementado. Durante el mismo, se toman decisiones importantes que contribuyen al cumplimiento sus requerimientos funcionales, en el capítulo se exponen técnicas que ayudarán a los miembros del equipo a comprender mejor el trabajo que debe hacerse durante la implementación del sistema. Se introducen una serie de diagramas que sirven de ayuda y guía en la implementación del modelo de sistema, como son: el diagrama de casos de uso del sistema, el modelo de la base de datos entre otros, se hace referencia a los principios de diseño del sistema, como son: estándares de diseño de interfaz y de programación, tratamiento de errores que se tienen en cuenta a la hora de realizar la aplicación.

3.2 - Descripción del Objeto de Automatización

A partir de esta investigación se obtiene un producto de software que facilita la gestión para la confección de los documentos planes de impartición y planes calendarios relacionados con el proceso de planificación docente en la Universidad de Cienfuegos “Carlos Rafael Rodríguez” y que responda a los objetivos propuestos.

El producto brindará al usuario las facilidades para crear, actualizar y eliminar los distintos documentos. De esta manera se mejora la confiabilidad y rapidez en el proceso de gestión de la información.

3.3 - Actores del sistema

Un actor no es más que un conjunto de roles que los usuarios de Casos de Uso

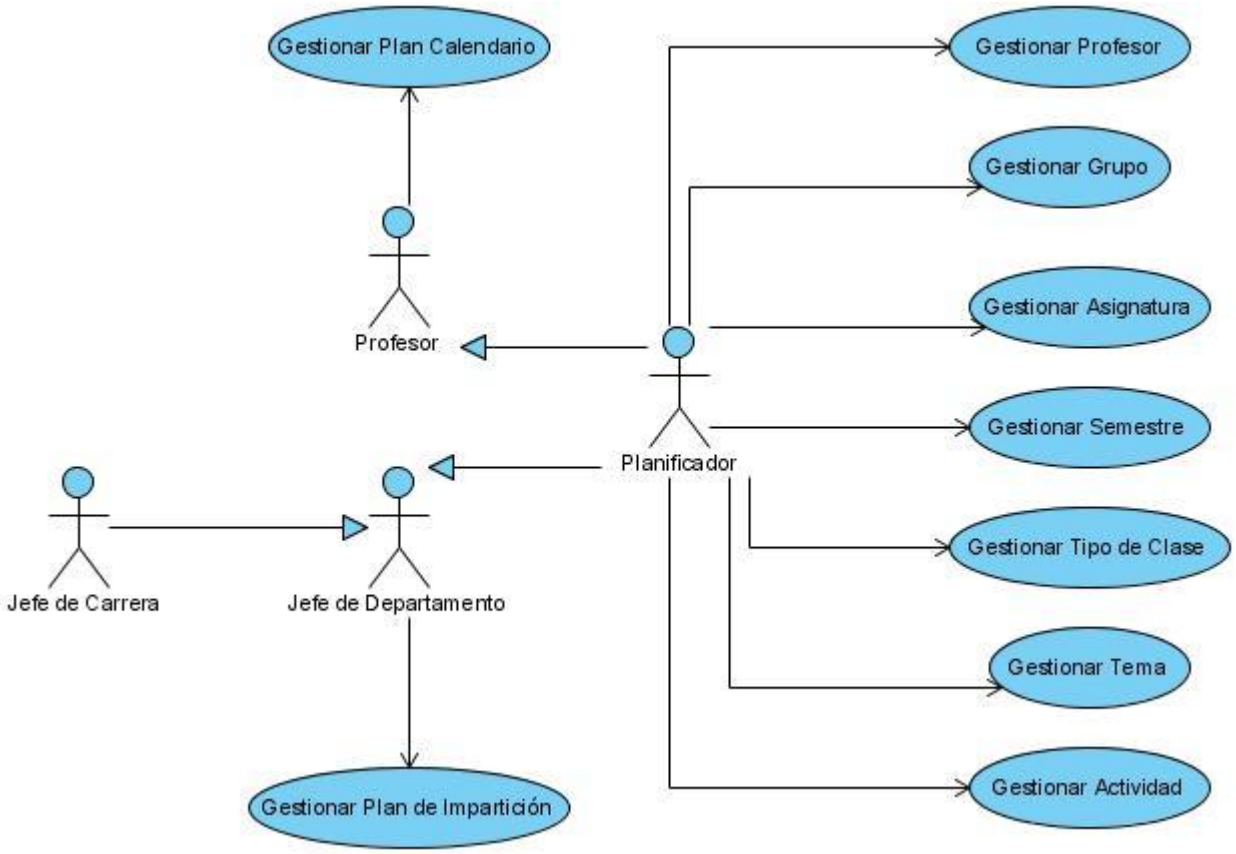
Desempeñan cuando interaccionan con estos Casos de Uso. Los actores representan terceros fuera del sistema que colaboran con el mismo. Una vez que hemos identificado los actores del sistema, tenemos identificado el entorno externo del sistema.[4]

3.4 - Diagrama de casos de uso del sistema:

Cada forma en que los actores usan el sistema se representa con un Caso de Uso. Los Casos de Uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, un Caso de Uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia

Los diagramas de casos de uso sirven para especificar la funcionalidad y el comportamiento de un sistema mediante su interacción con los usuarios y otros sistemas. O lo que es igual, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema.

3.5 - Diagrama de Casos de Uso



3.6 - Descripción textual de los actores del sistema:

Tabla 8 Descripción Actores del Sistema

Actor del Sistema	Justificación
Administrador(Planificador)	Es la persona que administra el sistema tiene acceso a toda la aplicación incluyendo los casos de uso que administran los profesores y jefe de departamento a la hora de elaborar los planes de impartición y planes calendario

Jefe de Departamento	Este usuario tendrá acceso solamente al caso de uso gestionar plan de impartición ya que él tiene acceso a crear este documento
Profesor	Este usuario tendrá acceso solamente al caso de uso gestionar plan de impartición ya que él es único que tiene acceso a crear este documento
Jefe de Carrera	Este usuario tendrá acceso solamente al caso de uso gestionar plan de impartición ya que él tiene acceso a crear este documento

Tabla 9 Gestionar Tipo de Clase

Caso de uso	**Gestionar Tipo de Clase**
Actores:	Planificador
Propósito:	Actualizar la información referente a los tipos de clase.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los tipos de clase, y finalmente el sistema: Busca, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Tipo de Clase
Precondiciones:	Se necesita que hayan tipos de clase en el sistema: Si acción es Buscar, se busca la información de los tipos de clase. Si acción es Actualizar, actualiza la información de los tipos de clase. Si acción es Eliminar, elimina la información del tipo de clase.

Postcondiciones:	Se actualiza la información de los tipos de clase: Si acción es Crear, se inserta la información del tipo de clase. Si acción es Actualizar, se actualiza la información del tipo de clase. Si acción es eliminar, se elimina la información del tipo de clase.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Tipo de Clase.

Tabla 10 Gestionar Plan de Impartición

Caso de uso	**Plan de Impartición**
Actores:	Jefe de Departamento
Propósito:	Actualizar la información de los planes de impartición.
Resumen: El caso de uso inicia cuando el Jefe de Departamento o Jefe de Carrera decide actualizar los planes de impartición, y finalmente el sistema: Busca, Crea, Actualiza o Elimina la información de los mismos; culminando de este modo el caso de uso.	
Referencias:	Gestionar Plan de Impartición
Precondiciones:	Se necesita que hayan planes de impartición: Si acción es Buscar, se busca la información de los planes de impartición. Si acción es Actualizar, actualiza la información de los planes de impartición. Si acción es Eliminar, elimina la información del plan de impartición.

Postcondiciones:	Se actualiza la información de los planes de impartición: Si acción es Crear, se crea la información del plan de impartición. Si acción es Actualizar, actualiza la información de los planes de impartición
Requisitos Especiales:	El plan de impartición creado se podrá modificar mientras no se de por terminado.
Prototipo:	Anexo

Tabla 2 – Descripción del caso del sistema: Gestionar Plan de Impartición.

Tabla 11 Gestionar Plan Calendario

Caso de uso	**Gestionar Plan Calendario**
Actores:	Profesor
Propósito:	Actualizar la información de los planes calendario (P-1).
Resumen: El caso de uso inicia cuando el jefe de departamento de la facultad decide asignar el P-1 de una asignatura a un profesor para que este lo confeccione, luego de ser confeccionado por el profesor el jefe de departamento es el responsable de la revisión del mismo, rechazando o aprobando a este; en caso de ser aprobado termina el caso de uso, si es rechazado el profesor tiene que realizar los cambios señalados para que pueda ser aprobado y de esta manera terminar el caso de uso.	
Referencias:	R-25
Precondiciones:	Se necesita que hayan planes calendario: Si acción es Buscar, se busca la información de los planes calendario. Si acción es Actualizar, actualiza la información de los planes calendario. Si acción es Eliminar, elimina la información del plan calendario.
Postcondiciones:	Se actualiza la información de los planes de impartición:

	Si acción es Crear, se crea la información del plan calendario. Si acción es Actualizar, actualiza la información del plan calendario. Si acción es aprobar, se actualiza la información del plan calendario.
Requisitos Especiales:	El plan de calendario creado se podrá modificar mientras no sea aprobado.
Prototipo:	Anexo

Tabla 3.25 – Descripción del caso de uso del sistema: Gestionar Plan Calendario.

Tabla 12 Gestionar Profesor

Caso de uso	**Gestionar Profesor**
Actores:	Planificador
Propósito:	Actualizar la información referente a los profesores.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los profesores, y finalmente el sistema: Busca, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Profesor
Precondiciones:	Se necesita que hayan profesores en el sistema: Si acción es Buscar, se busca la información de los profesores Si acción es Actualizar, actualiza la información de los profesores. Si acción es Eliminar, elimina la información de los profesores.
Postcondiciones:	Se actualiza la información de los profesores: Si acción es Crear, se inserta la información de los profesores.

	Si acción es Actualizar, se actualiza la información de los profesores. Si acción es eliminar, se elimina la información de los profesores.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Profesor.

Tabla 13 Gestionar Grupo

Caso de uso	**Gestionar Grupo**
Actores:	Planificador
Propósito:	Actualizar la información referente a los grupos.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los grupos, y finalmente el sistema: Buscar, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Grupos
Precondiciones:	Se necesita que hayan grupos en el sistema: Si acción es Buscar, se busca la información de los grupos. Si acción es Actualizar, actualiza la información de los grupos. Si acción es Eliminar, elimina la información de los grupos.
Postcondiciones:	Se actualiza la información de los grupos: Si acción es Crear, se inserta la información de los grupos. Si acción es Actualizar, se actualiza la información de los grupos. Si acción es eliminar, se elimina la información de los grupos.
Requisitos Especiales:	-

Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Grupo.

Tabla 14 Gestionar Semestre

Caso de uso	**Gestionar Semestre**
Actores:	Planificador
Propósito:	Actualizar la información referente a los semestres.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los semestres, y finalmente el sistema: Buscar, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Semestre
Precondiciones:	Se necesita que hayan semestres en el sistema: Si acción es Buscar, se busca la información de los semestres Si acción es Actualizar, actualiza la información de los semestres. Si acción es Eliminar, elimina la información de los semestres.
Postcondiciones:	Se actualiza la información de los semestres: Si acción es Crear, se inserta la información de los semestres. Si acción es Actualizar, se actualiza la información de los semestres. Si acción es eliminar, se elimina la información de los semestres.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Semestre.

Tabla 15 Gestionar Tema

Caso de uso	**Gestionar Tema**
Actores:	Planificador
Propósito:	Actualizar la información referente a los temas.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los temas, y finalmente el sistema: Busca, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Tema
Precondiciones:	Se necesita que hayan temas en el sistema: Si acción es Buscar, se busca la información de los temas. Si acción es Actualizar, actualiza la información de los temas. Si acción es Eliminar, elimina la información de los temas.
Postcondiciones:	Se actualiza la información de los temas: Si acción es Crear, se inserta la información de los temas. Si acción es Actualizar, se actualiza la información de los temas. Si acción es eliminar, se elimina la información de los temas
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Temas.

Tabla 16 Gestionar Actividad

Caso de uso	**Gestionar Actividad**
Actores:	Planificador

Propósito:	Actualizar la información referente a las actividades.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de las actividades, y finalmente el sistema: Buscar, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Actividad
Precondiciones:	Se necesita que hayan actividades en el sistema: Si acción es Buscar, se busca la información de las actividades. Si acción es Actualizar, actualiza la información de las actividades. Si acción es Eliminar, elimina la información de las actividades.
Postcondiciones:	Se actualiza la información de las actividades: Si acción es Crear, se inserta la información de las actividades. Si acción es Actualizar, se actualiza la información de las actividades. Si acción es eliminar, se elimina la información de las actividades.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Actividad.

Tabla 17 Gestionar Asignatura

Caso de uso	**Gestionar Asignatura**
Actores:	Planificador
Propósito:	Actualizar la información referente a las asignaturas.
Resumen:	

El caso de uso inicia cuando el planificador decide actualizar la información de las asignaturas, y finalmente el sistema: Buscar, Crea, Actualiza o Elimina la información; culminando de este modo el caso de uso.	
Referencias:	Gestionar Asignatura
Precondiciones:	Se necesita que hayan asignaturas en el sistema: Si acción es Buscar, se busca la información de las asignaturas Si acción es Actualizar, actualiza la información de las asignaturas. Si acción es Eliminar, elimina la información de las asignaturas.
Postcondiciones:	Se actualiza la información de a las asignaturas: Si acción es Crear, se inserta la información de las asignaturas. Si acción es Actualizar, se actualiza la información de las asignaturas. Si acción es eliminar, se elimina la información de las asignaturas.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Asignatura.

Tabla 18 Gestionar Departamento

Caso de uso	**Gestionar Departamento**
Actores:	Planificador
Propósito:	Actualizar la información referente a los departamentos.
Resumen: El caso de uso inicia cuando el planificador decide actualizar la información de los departamentos, y finalmente el sistema: Buscar, Crea, Actualiza o Elimina la información; culminando de este modo	

el caso de uso.

Referencias:	Gestionar Departamento
Precondiciones:	Se necesita que hayan departamentos en el sistema: Si acción es Buscar, se busca la los departamentos Si acción es Actualizar, actualiza la los departamentos. Si acción es Eliminar, elimina la información de los departamentos.
Postcondiciones:	Se actualiza la información de los departamentos: Si acción es Crear, se inserta la información de los departamentos. Si acción es Actualizar, se actualiza la información de los departamentos. Si acción es eliminar, se elimina la información de los departamentos.
Requisitos Especiales:	-
Prototipo:	Anexo

Tabla 1 – Descripción del caso del sistema: Gestionar Departamento

3.7 - Modelo de Base de Datos Diagrama 1

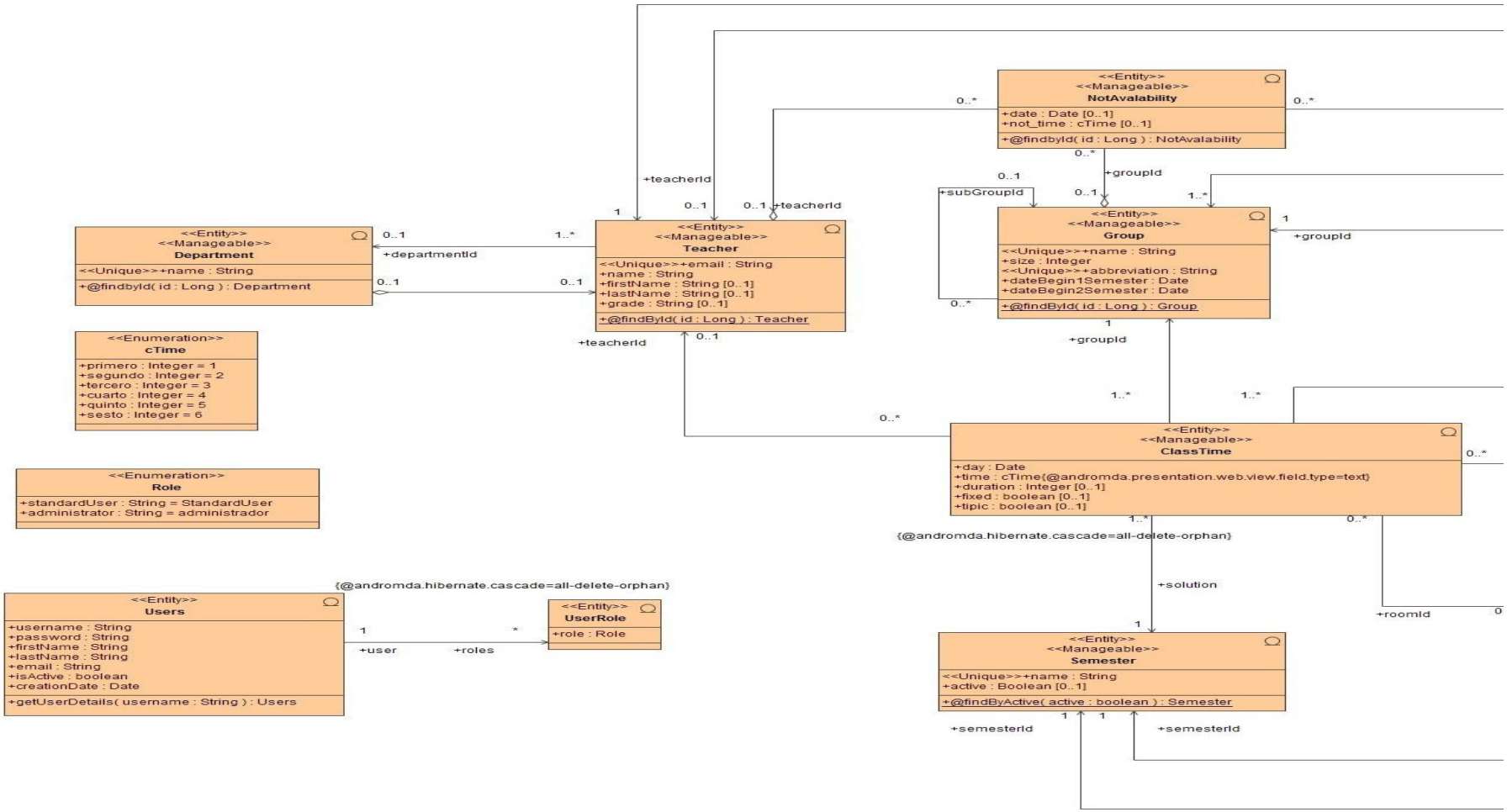
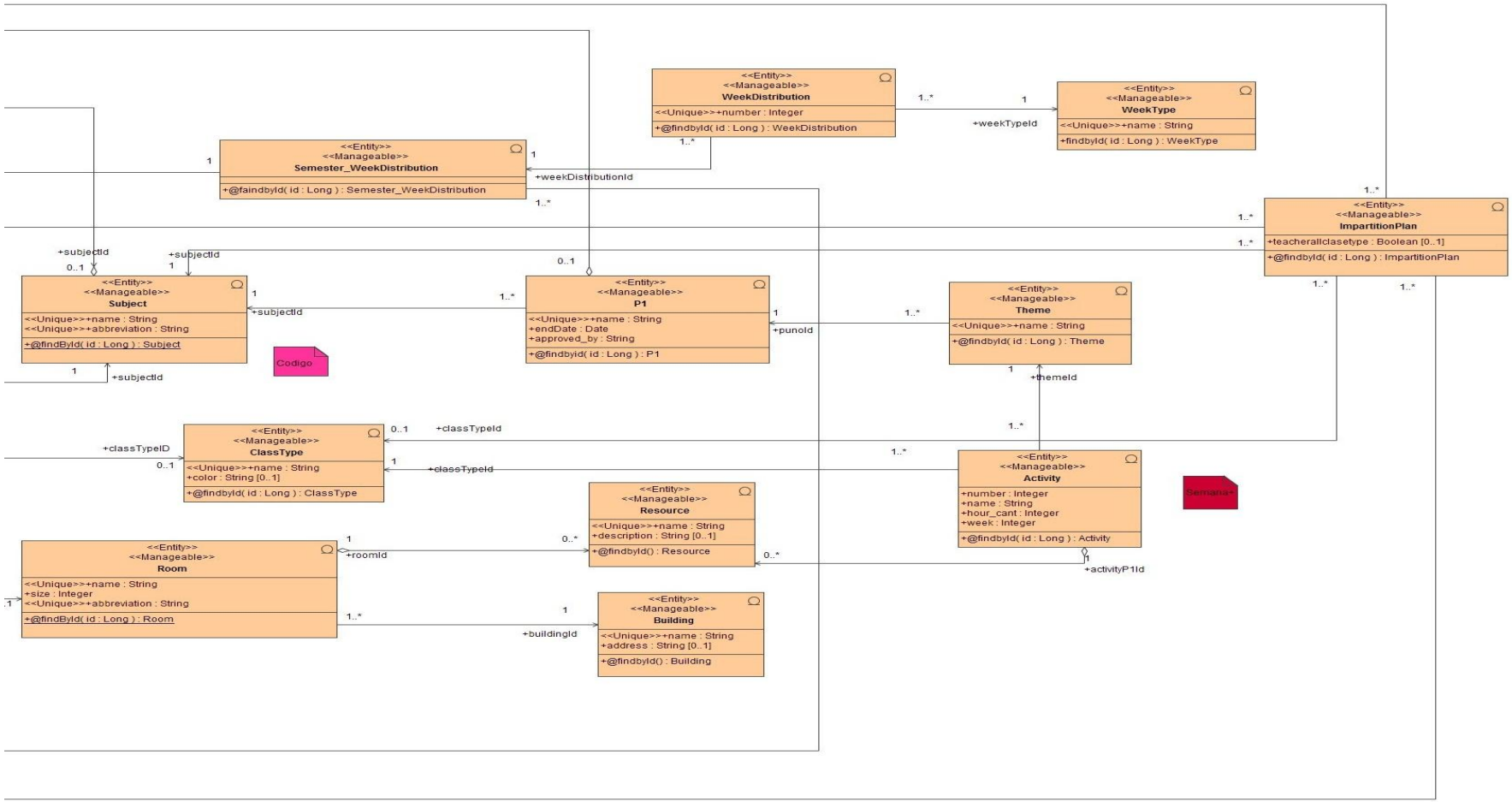


Diagrama 2:


<<Entity>>
<<Manageable>>
WeekDistribution
<<Unique>>+number : Integer
+@findbyId(id : Long) : WeekDistribution
<<Entity>>
<<Manageable>>
WeekType
<<Unique>>+name : String
+findbyId(id : Long) : WeekType
+weekTypeId
<<Entity>>
<<Manageable>>
Semester_WeekDistribution
+@faindbyId(id : Long) : Semester_WeekDistribution
+weekDistributionId
<<Entity>>
<<Manageable>>
ImpartitionPlan
+teacherallclasetype : Boolean [0..1]
+@findbyId(id : Long) : ImpartitionPlan
+subjectId
+subjectId
<<Entity>>
<<Manageable>>
Subject
<<Unique>>+name : String
<<Unique>>+abbreviation : String
+@findById(id : Long) : Subject
+subjectId
Codigo
<<Entity>>
<<Manageable>>
P1
<<Unique>>+name : String
+endDate : Date
+approved_by : String
+@findbyId(id : Long) : P1
+punoId
<<Entity>>
<<Manageable>>
Theme
<<Unique>>+name : String
+@findbyId(id : Long) : Theme
+themeId
+classTypeID
<<Entity>>
<<Manageable>>
ClassType
<<Unique>>+name : String
+color : String [0..1]
+@findbyId(id : Long) : ClassType
+classTypeId
+classTypeId
<<Entity>>
<<Manageable>>
Activity
+number : Integer
+name : String
+hour_cant : Integer
+week : Integer
+@findbyId(id : Long) : Activity
+activityP1Id
Semana
<<Entity>>
<<Manageable>>
Resource
<<Unique>>+name : String
+description : String [0..1]
+@findbyId() : Resource
<<Entity>>
<<Manageable>>
Room
<<Unique>>+name : String
+size : Integer
<<Unique>>+abbreviation : String
+@findById(id : Long) : Room
+roomId
<<Entity>>
<<Manageable>>
Building
<<Unique>>+name : String
+address : String [0..1]
+@findbyId() : Building
+buildingId

3.8 - Principios de diseño del sistema.

El diseño de sistemas se define como el proceso de aplicar ciertas técnicas y principios con el propósito de definir un dispositivo, un proceso o un sistema, con suficientes detalles como para permitir su interpretación y realización física.[22]

3.8.1 - Estándares en la interfaz de la aplicación.

La interfaz de entrada/salida del sistema corre a cargo de la herramienta que se utilizó para desarrollar el sistema (AndroMDA). Esto permitió usar en una plataforma web el estándar de controles típico de Windows en las aplicaciones de escritorio y las interacciones se basan en selecciones de tipo menú y en acciones físicas sobre elementos de código visual botones, acompañadas de logotipos y características propias de la Universidad.

3.8.2 - Concepción general de la ayuda.

La ayuda constituye una parte imprescindible en todo sistema. En el menú principal aparece una opción Ayuda que explica de forma detallada cómo funciona el sistema, tratando de aclarar los puntos que podría causar duda al usuario. Cada una de las opciones del sistema, así como las consideraciones que se asumen en la ejecución de ellas, está propiamente documentada para evitar cualquier tipo de confusión por parte del usuario. Cada aspecto de la ayuda ha sido diseñado con el objetivo de expresar explícitamente cómo y en qué orden debe operar el usuario.

3.8.3 - Tratamiento de excepciones.

El diseño de la interfaz ha estado dirigido a evitar errores, teniendo en cuenta paralelamente la creación de interfaces útiles y amigables. Se ha garantizado una validación intrínseca de los datos, procurando facilitar la corrección de errores lógicos tanto en la introducción como en cualquier otro momento del tratamiento de la información. Ante la ocurrencia de errores en el sistema se genera una excepción, la ejecución normal se detiene y se transfiere el control a

la zona de tratamiento de excepciones. Las excepciones internas se generan automáticamente por el sistema.

3.9 - Estudio de Factibilidad.

El estudio de factibilidad es análisis que se realiza antes del equipo comprometerse con la realización del software, tiene como fin realizar una comparación entre los costos y beneficios del sistema informático a desarrollar, con el único propósito de determinar si es fiable o no la realización del mismo.

3.9.1 - Cálculo de Puntos de Casos de Uso sin ajustar.

Se calcula a partir de la siguiente ecuación:

$$UUCP = UAW + UUCW$$

Dónde:

UUCP: Puntos de Casos de Uso sin ajustar.

UAW: Factor de Peso de los Actores sin ajustar.

UUCW: Factor de Peso de los Casos de Uso sin ajustar.

Factor de Peso de los Actores sin ajustar (UAW).

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema.[23]

Los criterios se muestran en la siguiente tabla:

3.9.2 - Descripción tipo de actores y su factor de peso

Tabla 19 Descripción tipo de actores y factor de peso

Tipo de actor	Descripción	Factor de peso
Simple	Otro sistema interactúa con el sistema a desarrollar mediante una interfaz de programación(API)	1
Medio	Otro sistema interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2
Complejo	Una persona interactúa con el sistema mediante una interfaz gráfica.	3

3.9.3 - Especificación de tipos de actores del sistema

Tabla 20 Especificación de tipos de actores del sistema

Actor	**Tipo de actor**	**Factor de peso**
Administrador(Planificador)	Complejo	3
Jefe de Departamento	Complejo	3
Profesor	Complejo	3

Dado que se cuenta con tres actores a interactuar con el sistema los cuales son personas que interactúan con el sistema tenemos que:

UAW=3x3=9

3.9.4 - Factor de Peso de los Casos de Uso sin ajustar (UUCW).

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia.[23]

3.9.5 - Descripción de tipos de casos de uso de acuerdo a su complejidad

Tabla 21 Descripción de tipos de casos de uso de acuerdo a su complejidad

Tipo de caso de uso	Descripción	Factor de peso
Simple	El caso de uso contiene de una a tres transacciones	5
Medio	El caso de uso contiene de 4 a 7 transacciones.	10
Complejo	El caso de uso contiene más de 8 transacciones	15

3.9.6 - Especificación de complejidad de los casos de uso del sistema

Tabla 22 Especificación de complejidad de los casos de uso del sistema

Caso de uso	Complejidad
Gestionar Profesor.	Simple
Gestionar Grupo.	Simple
Gestionar Semestre.	Simple
Gestionar Plan de Impartición.	Medio
Gestionar acciones Tipo de Clase.	Simple
Gestionar Tema.	Simple
Gestionar Actividad.	Simple
Gestionar Asignatura.	Simple
Gestionar Plan Calendario.	Medio
Gestionar Departamento	Simple

Tipo de caso de uso	Cantidad de casos de uso	Factor de peso por caso de uso	Peso total
Simple	8	5	**40**
Complejo	**2**	**10**	**20**

De acuerdo a la tabla anterior se cuenta con 8 casos de uso de tipo Simple (peso 5) y 2 de tipo Medio (peso 10) con lo cual el factor de peso de los casos de uso sin ajustar resulta:

UUCW = 8 x 5 + 2 x 10

UUCW =60

Finalmente, los Puntos de Casos de Uso sin ajustar resultan

UUCP = UAW + UUCW

UUCP =9+ 60

UUC =69

3.6.7 - Cálculo de Puntos de Casos de Uso ajustados.

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se d**e**be ajustar éste valor mediante la siguiente ecuación:

$$UCP = UUCP \times TCF \times EF$$

Donde,

UCP: Puntos de Casos de Uso ajustados.

UUCP: Puntos de Casos de Uso sin ajustar.

TCF: Factor de complejidad técnica.

EF: Factor de ambiente.

3.6.8 - Factor de complejidad técnica (TCF).

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante.

Factor determinante de complejidad técnica del sistema

Tabla 23 Factor determinante de complejidad técnica del sistema

Factor	Descripción	Peso	Importancia	Peso total

T1	Sistema distribuido	2	0	0
T2	Objetivos de performance o tiempo de respuesta	1	3	3
T3	Eficiencia del usuario final	1	5	5
T4	Procesamiento interno complejo	1	4	4
T5	El código debe ser reutilizable	1	5	5
T6	Facilidad de instalación	0.5	5	2.5
T7	Facilidad de uso	0.5	5	2.5
T8	Portabilidad	2	5	10
T9	Facilidad de cambio	1	5	5
T10	Concurrencia	1	3	3
T11	Incluye objetos especiales de seguridad	1	2	2
T12	Provee acceso directo a terceras partes	1	0	0
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1	1	1

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

$$TCF = 0.6 + 0.01 \times \sum (Peso_i \times Valor\ asignado_i)$$

Dados los valores asignados en la tabla anterior y realizando el cálculo obtuvimos:

TCF=0.6 +0.01 x43

TCF=1,03

3.6.9 - Factor de ambiente (EF).

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se

contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5.

Factor externo o ambiente.

Tabla 24 Factor externo o ambiente

Factor Ambiente	Descripción	Peso	Valor asignado	Peso Total
E1	Familiaridad con el modelo de proyecto utilizado	1,5	3	**4,5**
E2	Experiencia en la aplicación	0,5	4	**2**
E3	Experiencia en la orientación a objetos	1	5	**5**
E4	Capacidad del analista líder	0,5	4	**2**
E5	Motivación	1	4	**3**
E6	Estabilidad de los requerimientos	2	4	**8**
E7	Personal a tiempo parcial	-1	0	**0**
E8	**Dificultad del lenguaje de programación**	**-1**	**2**	**-2**

El Factor de ambiente se calcula mediante la siguiente ecuación:

$$EF = 1.4 - 0.03 \times \sum (Peso_i \times Valor\ asignado_i)$$

Apoyándose en la tabla anterior y realizando los cálculos pertinentes:

EF=1.4 - 0.03x22.5

EF=0,725

Los puntos de casos de uso ajustados resultan:

UCP = UUCP x TCF x EF

UCP= 209x 1.03 x 0.725

UCP=156.07075

3.6.10 - Estimación del esfuerzo

CF: Factor de Conversión

CF= 20 Horas-Hombre

El esfuerzo en horas /hombre está dado por:

E= UCP * CF

E= 156.07075* 20 = 3121.415 horas-hombre.

3.6.11 - Duración:

Trabajando 30 días al mes y 12 horas diarias como promedio, se tiene que:

Duración (días)= Total de Horas /Hombre entre 12 horas al día = 3121.415 /12
Duración (días)=261

Duración (meses)=Total de días / 30 días por mes = 261 / 30

Duración (meses)=8.7 ≈ 9

3.6.12 - Criterio de distribución de esfuerzo

Tabla 25 Criterio de distribución de esfuerzo

Actividad	Porcentaje	Personal disponible por fase	Esfuerzo (H/H)
Análisis	10%	3	312,1415
Diseño	20%	3	624,283
Programación	55%	1	1716,77825
Prueba	10%	1	312,1415
Sobrecarga	5%	1	156,07075
Total de Horas	100%		3121,415

3.6.13 - Cálculo de costos

Tomando como salario promedio mensual $400.00

Costo = 9 meses x $400.00 = $3600.00

3.6.14 - Beneficios tangibles e intangibles

Con el desarrollo de este sistema informático se logra disminuir en gran manera el tiempo requerido para el proceso de confección de los documentos planes de impartición y planes calendario en la Universidad de Cienfuegos, así como la presentación de los mismos con los datos correctamente actualizados. De esta manera se da cumplimiento a los objetivos trazados ya que el desarrollo del sistema informático está enfocado a los mismos.

3.6.15 - Análisis de costos y beneficios.

Anteriormente el proceso de confección de los documentos planes de impartición y planes calendario en la Universidad de Cienfuegos se realizaba de

forma manual consumiendo un nivel de tiempo enorme e impidiendo el conocimiento en totalidad de la información requerida de los mismos debido al gran número de información relacionada con el tema. Con la ayuda del sistema informático propuesto se logra eliminar los problemas antes descritos. Teniendo en cuenta el análisis del costo del sistema y los beneficios que brinda el mismo al cumplir con los objetivos planteados, se concluye que la aplicación es factible.

3.10 - Escenarios de Pruebas para el Sistema.

Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto software. Son utilizadas para identificar posibles fallos de implementación, calidad, o usabilidad de un programa de ordenador. Básicamente es una fase en el desarrollo de software consistente en probar las aplicaciones construidas.

Los escenarios de prueba son las pruebas que se realizan, desde una perspectiva, para determinar lo rápido que realiza una tarea un sistema en condiciones particulares de trabajo. También puede servir para validar y verificar otros atributos de la calidad del sistema, tales como la escalabilidad, fiabilidad y uso de los recursos. Los escenarios de prueba son un subconjunto de la ingeniería de pruebas, una práctica informática que se esfuerza por mejorar el rendimiento, englobándose en el diseño y la arquitectura de un sistema.

La aplicación que se quiere lograr con esta investigación debe ser capaz de resolver gran parte de la creación manual de los documentos plan de impartición y plan calendario en la UCF facilitando el proceso de planificación ayudando al personal que trabaja para esto por lo que es necesario realizarle pruebas al sistema para asegurar su funcionamiento. Un posible escenario de prueba seria entregarle el software antes de su salida oficial a las personas encargadas de la planificación, el planificador es el principal responsable de gestionar todo lo referente para confeccionar los documentos y tiene pleno control sobre la

aplicación las personas que realizan los documentos son: el jefe de cada departamento siendo el encargado de realizar el plan de impartición y los profesores confeccionan los planes calendarios (P1), esta prueba es clave a la hora de detectar errores o fallas y deja claro en que escala se encuentran los distintos conceptos como estabilidad, eficiencia y seguridad siendo muy importantes en la calidad de un producto bien desarrollado.

Administrador (Planificador): Es la persona que administra el sistema tiene acceso a toda la aplicación incluyendo los casos de uso que administran los profesores y jefe de departamento a la hora de elaborar los planes de impartición y planes calendario, una vez que el planificador entra a la aplicación con su respectiva contraseña administrativa tendrá acceso a controlar todo lo referente a los Profesores, Grupos, Semestres, Departamentos, Tipos de Clase, Temas, Actividades, Asignatura, Planes Calendario, Planes de Impartición, el planificador tendrá el acceso a los documentos pero no será el responsable de confeccionarlos ya que es trabajo de otras personas, el sistema permitirá al administrador poder manejar todos los datos de las entidades antes mencionadas facilitándole buscar, crear, actualizar, eliminar cualquiera de sus atributos permitiéndole almacenar información de manera centralizada de fácil consulta y manejo.

Jefe de Departamento: Este usuario tendrá acceso solamente al caso de uso gestionar plan de impartición ya que él es único que tiene acceso a crear este documento, una vez que entre en el sistema con su contraseña podrá confeccionar el plan de impartición basándose en la relación que tiene cada profesor con las asignaturas que impartirá el sistema permitirá obtener información previamente almacenada por el planificador de profesores y asignaturas para establecer la relación, esta relación tendrá en cuenta además del nombre del profesor y sus asignaturas el tipo de clase, el grupo donde se dará una determinada asignatura así como el semestre , todos estos datos son necesarios a la hora de confeccionar el plan calendario

Profesor: Este usuario tendrá acceso solamente al caso de uso gestionar plan de impartición ya que él es único que tiene acceso a crear este documento, el profesor al igual que el jefe de departamento tendrá una contraseña propia que le permitirá acceder al sistema con los privilegios necesarios para crear el plan calendario una vez dentro tendrá que tener en cuenta varios aspectos para completar satisfactoriamente la gestión como quien es la persona que aprobara el plan una vez terminado así como el profesor que lo confeccionó, la asignatura y una fecha límite de entrega para tener control de quien es la persona que el que falta por entregar el P1.

3.11 - Conclusiones del Capítulo:

En el presente capítulo se elaboró el diagrama de casos de uso del sistema y el diagrama de base de datos generado por la herramienta utilizada para elaborar el sistema. Además se definieron los estándares en la interfaz de la aplicación, la concepción general de la ayuda y el tratamiento de excepciones. También se realizó un estudio sobre la factibilidad del sistema, aplicando para ello el método de estimación basado en casos de uso.

El desarrollo de la validación del sistema mostró resultados favorables a partir de las pruebas, donde el ahorro de tiempo y la confiabilidad son los mayores beneficios que aporta el sistema.

CONCLUSIONES GENERALES

Teniendo en cuenta los objetivos planteados, se arriban a las siguientes conclusiones:

- Teniendo en cuenta el proceso de gestión de los documentos oficiales para realizar el pan de impartición y el plan calendario se definieron las actividades a automatizar.
- Se analizó el proceso de confección de planes de impartición y planes calendario docentes comprendiendo su funcionamiento.
- Se implementó un sistema informático, que da respuesta a la problemática descrita.
- La vinculación de las tecnologías informáticas a la gestión de los documentos posibilita que esta se desarrolle de un modo rápido y confiable.
- Se desarrollo un sistema informático para la gestión de planes de impartición y planes calendario proporcionando una herramienta que mejorará el flujo informativo en la Universidad de Cienfuegos.
- Se informatizó el proceso de confección de planes de impartición y planes calendario docente propiciando que las actividades de este proceso se ejecuten de manera coordinada, transparente y de acuerdo con los documentos rectores.

REFERENCIAS BIBLIOGRÁFICAS:

[1] «Qué es la gestión de procesos», 31-abr-2013. [Online]. Available: http://es.answers.com/Q/Qu%C3%A9_es_la_gesti%C3%B3n_de_procesos. [Accessed: 31-abr-2013].

[2] «Gestión de la información - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/Gesti%C3%B3n_de_la_informaci%C3%B3n. [Accessed: 31-abr-2013].

[3] «Que Es Planificar», 31-abr-2013. [Online]. Available: http://www.slideshare.net/jcfdezmxestra/que-es-planificar. [Accessed: 31-abr-2013].

[4] Jacobson I, *El Proceso Unificado De Desarrollo De Software*. Addison-Wesley Iberoamericana España, S.A.

[5] Yuviny Echevarría Cartaya, «Sistema Informático para la confección de Horarios Docentes en la Facultad de Informática de la Universidad de Cienfuegos», Universidad de Cienfuegos «Carlos Rafael Rodríguez», Cienfuegos.

[6] López Corzo E, «La Nueva Universidad en Informática 2009».

[7] Yisel Alfaro Abreus Orelvis Vázquez Ruiz, «Sistema de Gestión Académica». .

[8] «Lenguaje Unificado de Modelado - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: https://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado. [Accessed: 31-abr-2013].

[9] Idiel Suárez Álvarez, «SISTEMA PARA LA GESTIÓN DE LOS RECURSOS Y DOCUMENTOS DE LA PLANIFICACIÓN DOCENTE», Universidad de Cienfuegos «Carlos Rafael Rodríguez», Cienfuegos.

[10] «NetBeans - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/NetBeans. [Accessed: 31-abr-2013].

[11] «Plataforma Java - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/Plataforma_Java. [Accessed: 31-abr-2013].

[12] Yuviny Echevarría Cartaya, «Sistema Informático para la confección de Horarios Docentes en la Facultad de Informática de la Universidad de Cienfuegos», Universidad de Cienfuegos «Carlos Rafael Rodríguez», Cienfuegos.

[13] «Apache Tomcat - Welcome!», 31-abr-2013. [Online]. Available: http://tomcat.apache.org/. [Accessed: 31-abr-2013].

[14] «Maven - Welcome to Apache Maven», 31-abr-2013. [Online]. Available: http://maven.apache.org/. [Accessed: 31-abr-2013].

[15] «AndroMDA Model Driven Architecture Framework - AndroMDA - Homepage», 31-abr-2013. [Online]. Available: http://www.andromda.org/index.html. [Accessed: 31-abr-2013].

[16] «Programación extrema - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema. [Accessed: 31-abr-2013].

[17] «ICONIX - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/ICONIX. [Accessed: 31-abr-2013].

[18] «Metodología ágil versus tradicional: no tan diferentes en sus principios | PensandoEnSOA.com - Andrés Hevia», 31-abr-2013. [Online]. Available: http://pensandoensoa.com/2012/05/19/metodologia-agil-versus-tradicional/. [Accessed: 31-abr-2013].

[19] Jesús Suárez Abraham, «Sistema Informático para la Gestión de Contratos en la Oficina del Centro Internacional de la Habana de la Universidad de Cienfuegos».

[20] «Scrum - Wikipedia, la enciclopedia libre», 31-abr-2013. [Online]. Available: http://es.wikipedia.org/wiki/Scrum. [Accessed: 31-abr-2013].

[21] «Pilas de Producto, hablando de Scrum. Artefacto de gestión de proyectos.», 31-abr-2013. [Online]. Available: http://www.genbetadev.com/metodologias-de-programacion/pilas-de-producto-hablando-de-scrum. [Accessed: 31-abr-2013].

[22] «Principios a seguir en el diseño de un sistema. «□Blog de Jose Cuéllar», 31-abr-2013. [Online]. Available: http://www.josecuellar.net/arquitectura-de-software/principios-a-seguir-en-el-diseno-de-un-sistema/. [Accessed: 31-abr-2013].

[23] «Estimación de software basada en puntos de casos de uso. Ejemplo práctico - Monografias.com», 31-abr-2013. [Online]. Available: http://www.monografias.com/trabajos87/estimacion-software-basada-puntos/estimacion-software-basada-puntos.shtml. [Accessed: 31-abr-2013].

BIBLIOGRAFÍA:

[1] Alfaro Abreus, Yisel; Vázquez Ruiz, Orelvis «Sistema de Gestión Académica». .

[2] «AndroMDA Model Driven Architecture Framework - AndroMDA - Homepage», 31-may-2013. [Online]. Available: http://www.andromda.org/index.html. [Accessed: 31-may-2013].

[3] «Apache Tomcat - Welcome!», 31-may-2013. [Online]. Available: http://tomcat.apache.org/. [Accessed: 31-may-2013].

[4] «Diagrama de casos de uso - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Diagrama_de_casos_de_uso. [Accessed: 31-may-2013].

[5] «ejemplos | proyectos Ágiles», 31-may-2013. [Online]. Available: http://www.proyectosagiles.org/taxonomy/term/10. [Accessed: 31-may-2013].

[6] Jacobson I, *El Proceso Unificado De Desarrollo De Software*. Addison-Wesley Iberoamericana España, S.A.

[7] Echevarría Cartaya, Yuviny, «Sistema Informático para la confección de Horarios Docentes en la Facultad de Informática de la Universidad de Cienfuegos», Universidad de Cienfuegos «Carlos Rafael Rodríguez», Cienfuegos.

[8] «Estimacion basada en puntos de casos de uso», 31-may-2013. [Online]. Available: http://www.slideshare.net/dianitadance/estimacion-basada-en-puntos-de-casos-de-uso. [Accessed: 31-may-2013].

[9] «Estimación de software basada en puntos de casos de uso. Ejemplo práctico - Monografias.com», 31-may-2013. [Online]. Available: http://www.monografias.com/trabajos87/estimacion-software-basada-puntos/estimacion-software-basada-puntos.shtml. [Accessed: 31-may-2013].

[10] «Estimación y Planificación ágil», 31-may-2013. [Online]. Available: http://www.slideshare.net/seba_soft/presenta-estimacion-planificacion. [Accessed: 31-may-2013].

[11] «Gestión de la información - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Gesti%C3%B3n_de_la_informaci%C3%B3n. [Accessed: 31-may-2013].

[12] «ICONIX - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/ICONIX. [Accessed: 31-may-2013].

[13] «Ingeniería de software orientada a objetos con UML, Java en Internet - Alfredo Weitzenfeld - Google Libros», 31-may-2013. [Online]. Available: http://books.google.com.cu/books?id=MOviEp0ApQcC&pg=PA200&lpg=PA200&dq=ingenieria+software+Actores+del+sistema&source=bl&ots=OWAHf9Onit&sig=2kv2Adgv9FTANZzvKCjbcQ5ZA44&hl=es&sa=X&ei=FuWoUY-cEuPO0gGs7YHQBA&ved=0CCgQ6AEwAA. [Accessed: 31-may-2013].

[14] López Corzo E, «La Nueva Universidad en Informática 2009».

[15] «Lenguaje Unificado de Modelado - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: https://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado. [Accessed: 31-may-2013].

[16] «Maven - Welcome to Apache Maven», 31-may-2013. [Online]. Available: http://maven.apache.org/. [Accessed: 31-may-2013].

[17] «Metodología ágil versus tradicional: no tan diferentes en sus principios | PensandoEnSOA.com - Andrés Hevia», 31-may-2013. [Online]. Available: http://pensandoensoa.com/2012/05/19/metodologia-agil-versus-tradicional/. [Accessed: 31-may-2013].

[18] «NetBeans - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/NetBeans. [Accessed: 31-may-2013].

[19] «Pilas de Producto, hablando de Scrum. Artefacto de gestión de proyectos.», 31-may-2013. [Online]. Available: http://www.genbetadev.com/metodologias-de-programacion/pilas-de-producto-hablando-de-scrum. [Accessed: 31-may-2013].

[20] «Planear una iteración», 31-may-2013. [Online]. Available: http://msdn.microsoft.com/es-es/library/ee191595.aspx. [Accessed: 31-may-2013].

[21] «Plataforma Java - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Plataforma_Java. [Accessed: 31-may-2013].

[22] «Principios a seguir en el diseño de un sistema. «Blog de Jose Cuéllar», 31-may-2013. [Online]. Available: http://www.josecuellar.net/arquitectura-de-software/principios-a-seguir-en-el-diseno-de-un-sistema/. [Accessed: 31-may-2013].

[23] «Programación extrema - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema. [Accessed: 31-may-2013].

[24] «Puntos de caso de uso - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Puntos_de_caso_de_uso. [Accessed: 31-may-2013].

[25] «Qué es la gestión de procesos», 31-may-2013. [Online]. Available: http://es.answers.com/Q/Qu%C3%A9_es_la_gesti%C3%B3n_de_procesos. [Accessed: 31-may-2013].

[26] «Que Es Planificar», 31-may-2013. [Online]. Available: http://www.slideshare.net/jcfdezmxestra/que-es-planificar. [Accessed: 31-may-2013].

[27] «Scrum - Wikipedia, la enciclopedia libre», 31-may-2013. [Online]. Available: http://es.wikipedia.org/wiki/Scrum. [Accessed: 31-may-2013].

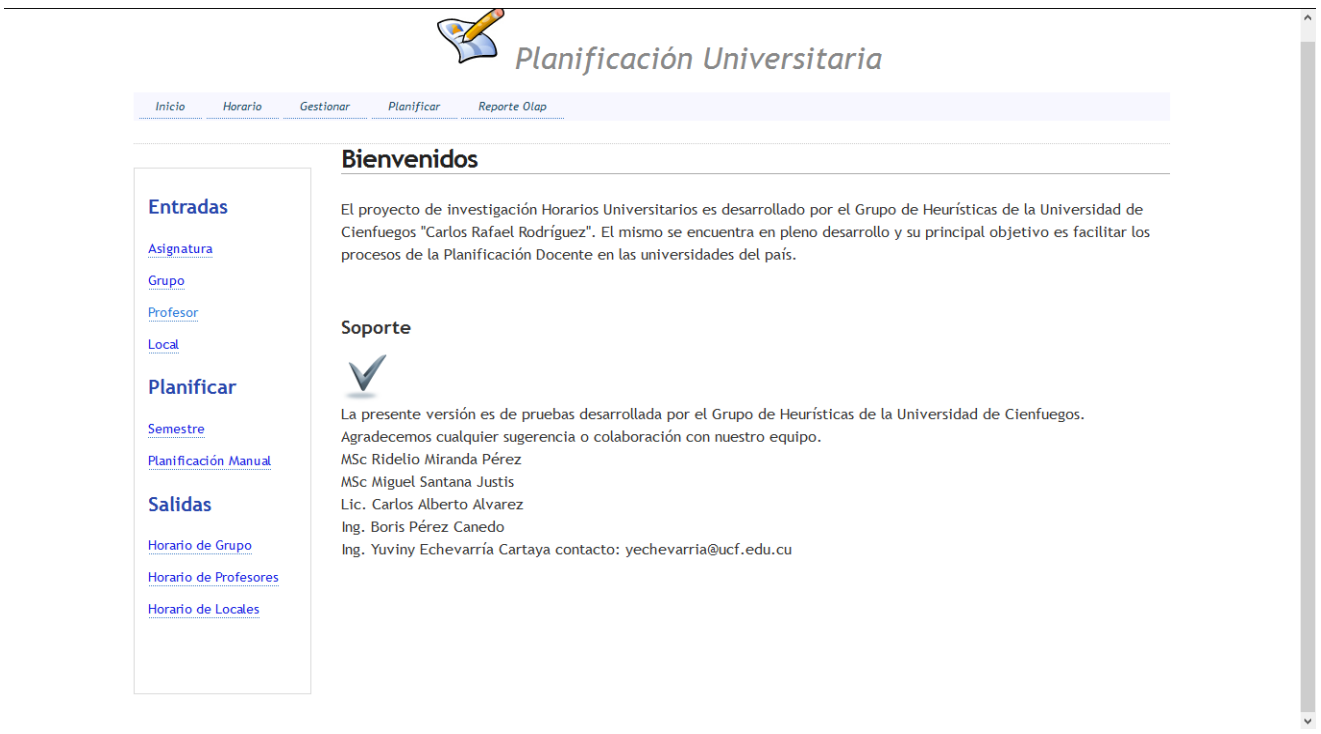
[28] Suárez Abraham, Jesús «Sistema Informático para la Gestión de Contratos en la Oficina del Centro Internacional de la Habana de la Universidad de Cienfuegos».

[29] Suárez Álvarez, Idiel «SISTEMA PARA LA GESTIÓN DE LOS RECURSOS Y DOCUMENTOS DE LA PLANIFICACIÓN DOCENTE», Universidad de Cienfuegos «Carlos Rafael Rodríguez», Cienfuegos.

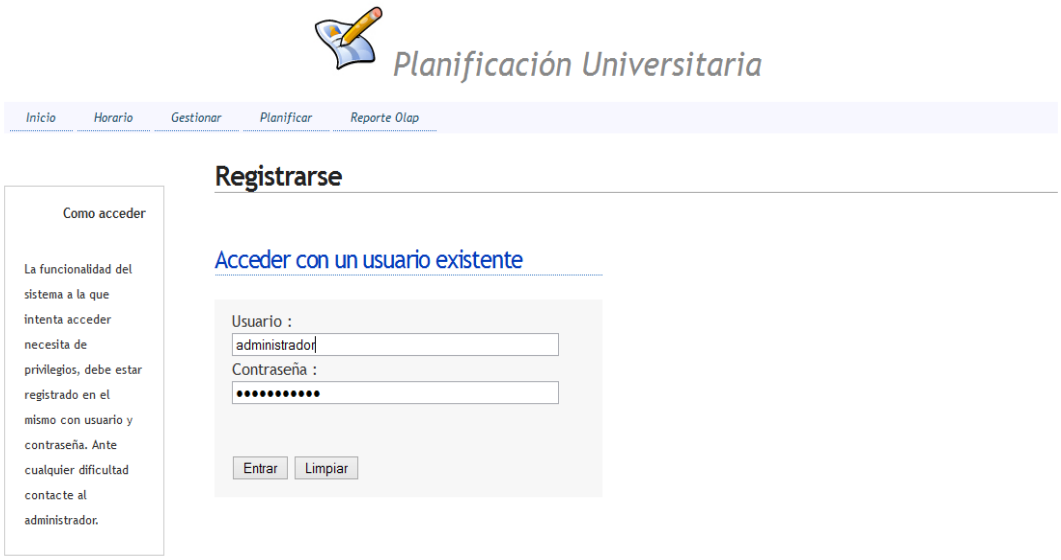
[30] «Una experiencia práctica de Scrum a través del aprendizaje basado e...», 31-may-2013. [Online]. Available: http://www.slideshare.net/sayazyi/una-experiencia-prctica-de-scrum-a-travs-del-aprendizaje-basado-en-proyectos-mediado-por-tic-en-un-equipo-distribuido. [Accessed: 31-may-2013].

[31] «www.navegapolis.net - Midiendo velocidad, trabajo y tiempo en gestión ágil», 31-may-2013. [Online]. Available: http://www.navegapolis.net/content/view/781/58/. [Accessed: 31-may-2013].

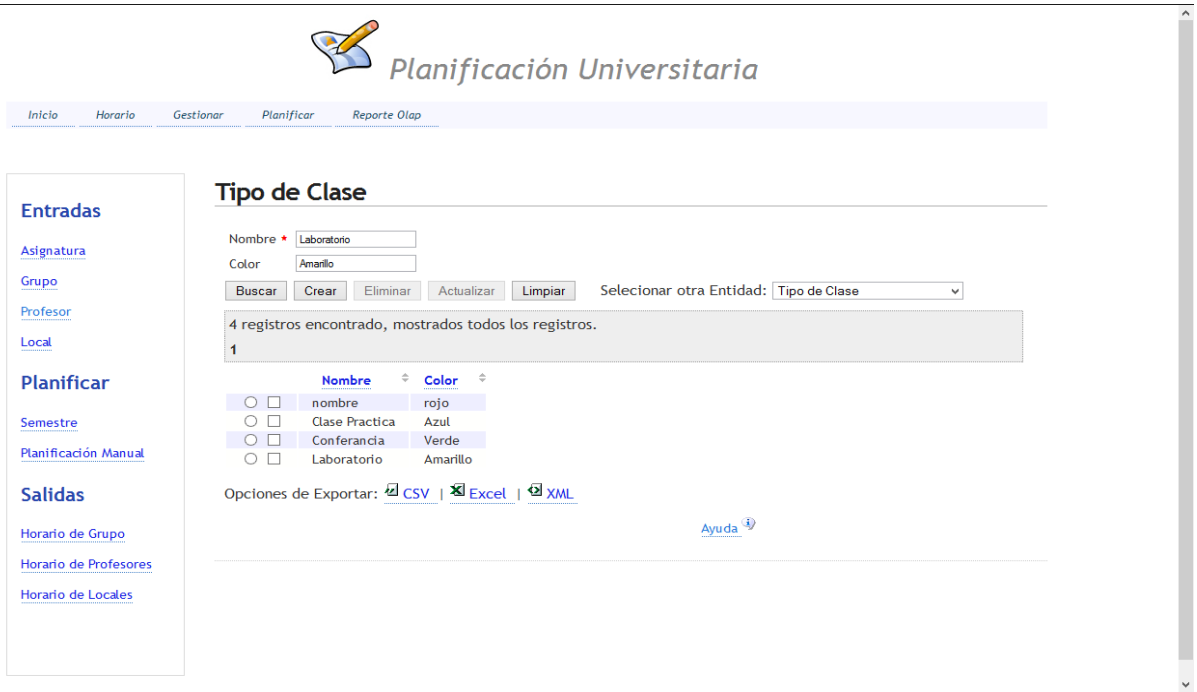
ANEXOS:



Anexo 1 – Inicio



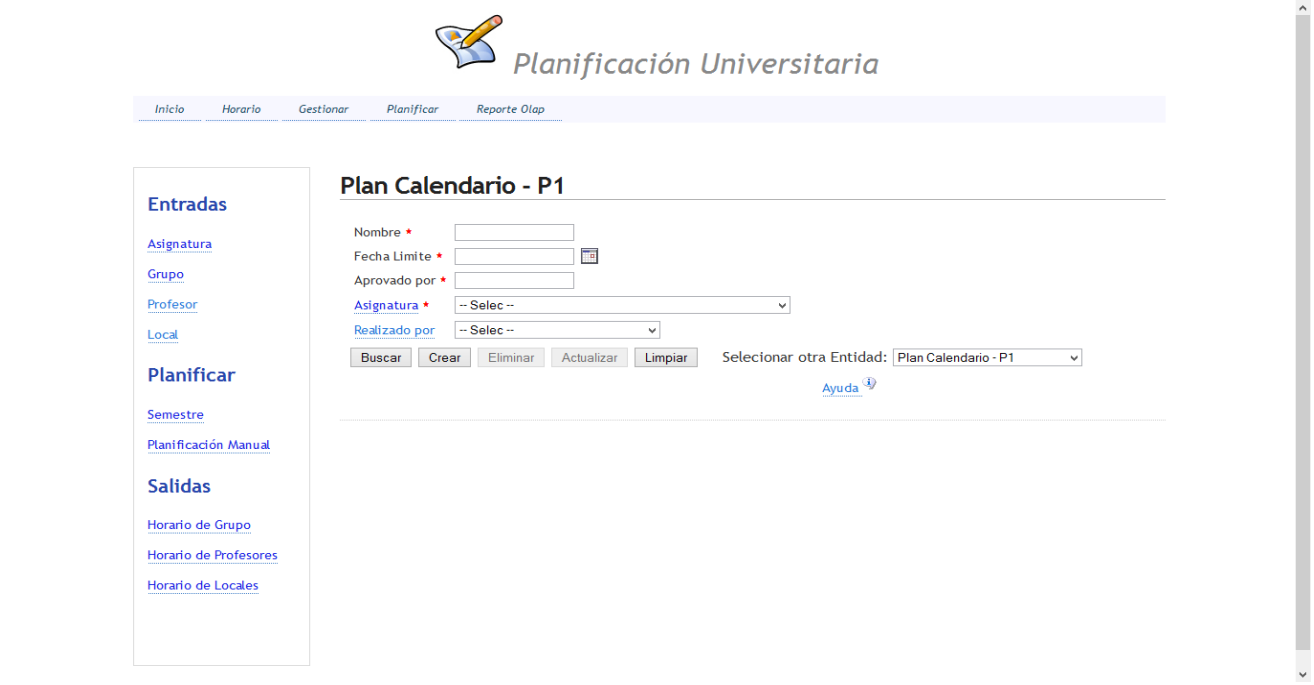
Anexo 2 – Autentificarse en el Sistema



Anexo 3 – Gestionar Tipo de Clase

Anexo 4 – Gestionar Tema

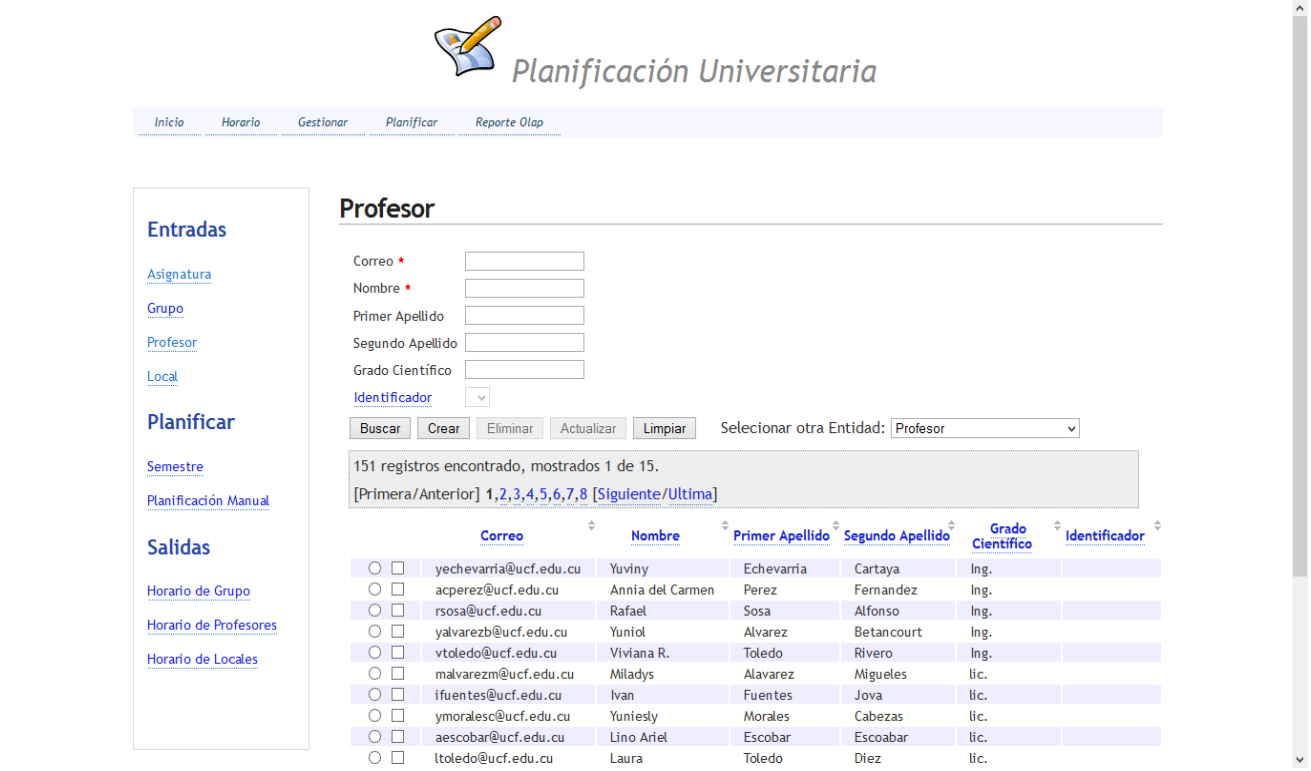
Anexo 5 – Gestionar Actividad



Anexo 6 – Gestionar Plan Calendario

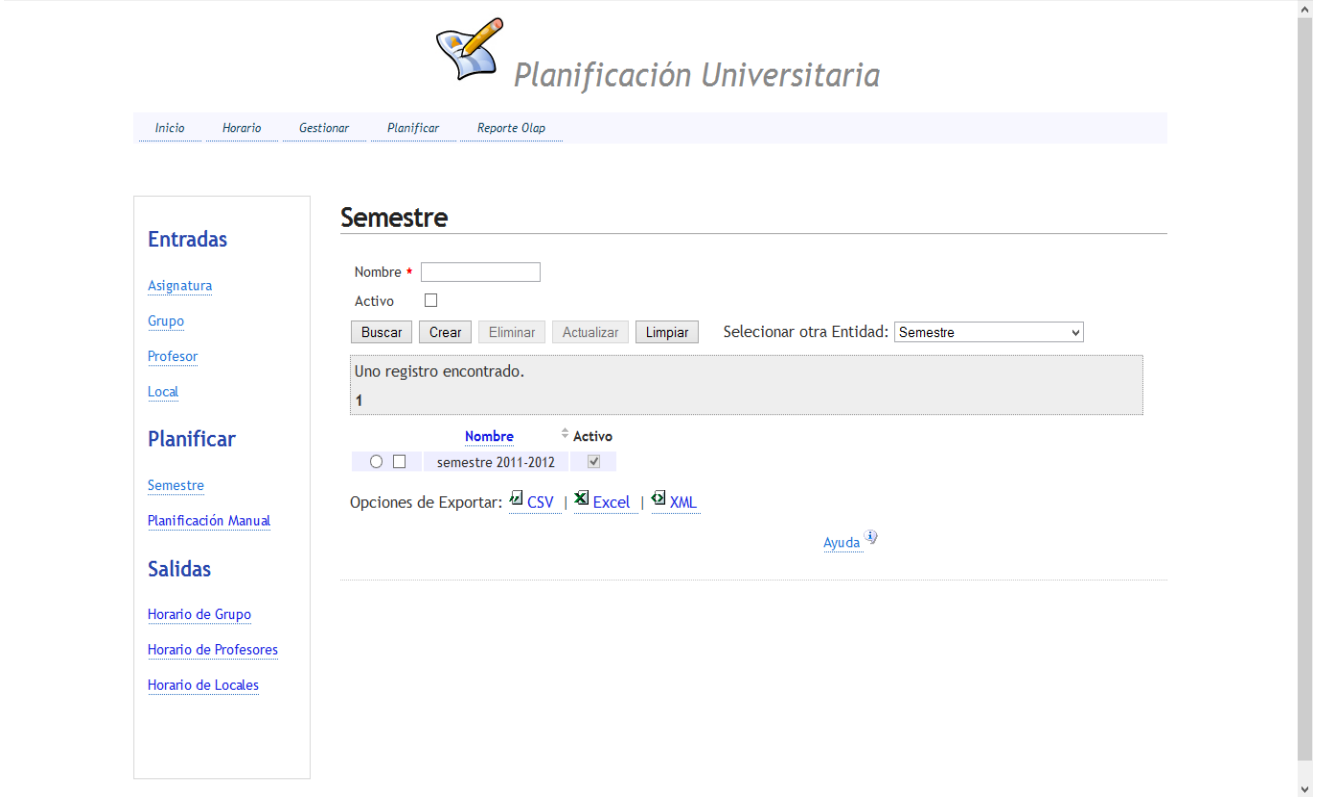
Anexo 7 – Gestionar Asignatura

Anexo 8 – Gestionar plan de Impartición

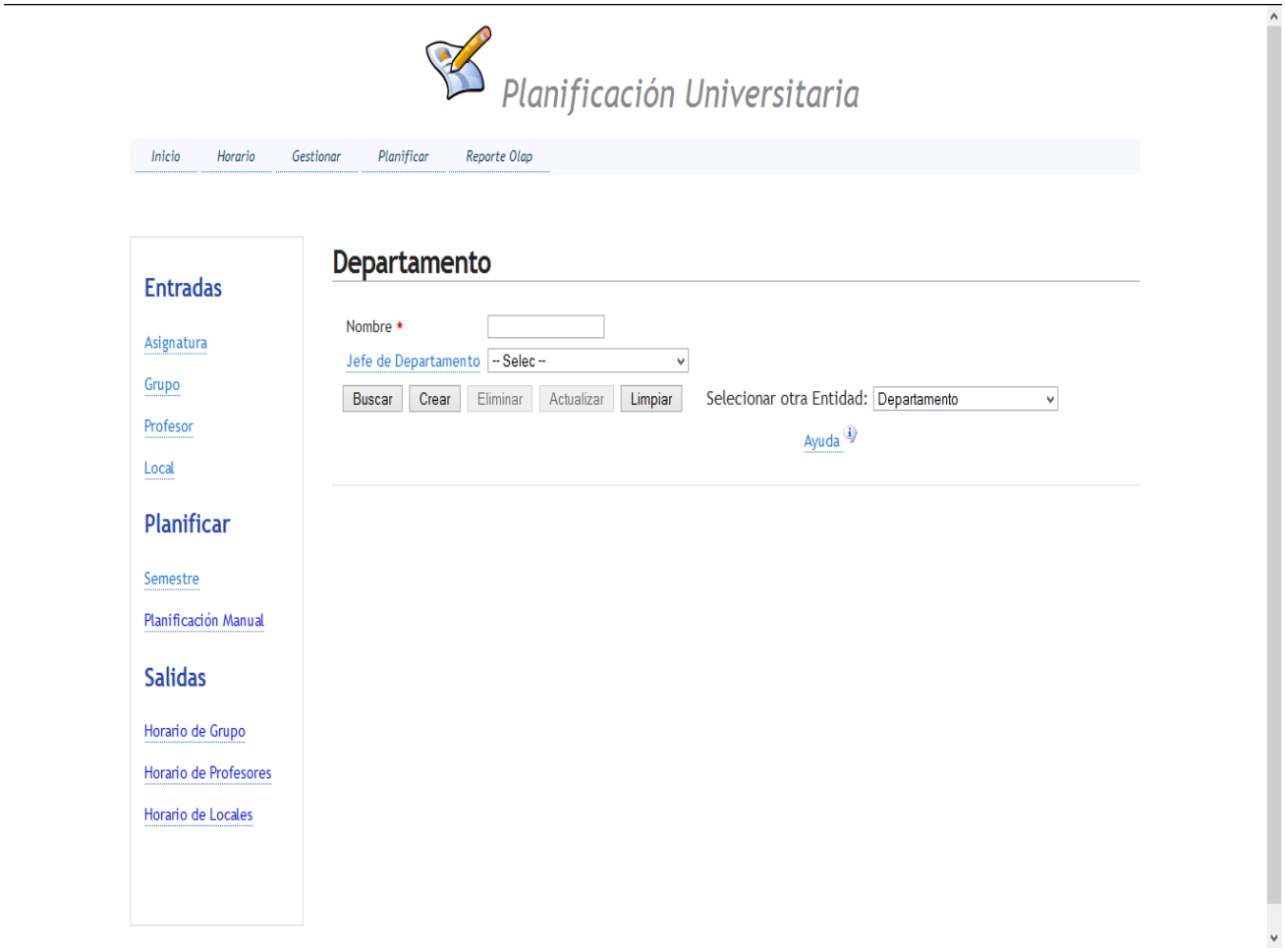


Anexo 9 – Gestionar Profesor

Anexo 10 – Gestionar Grupo



Anexo 11 – Gestionar Semestre



Anexo 12 – Gestionar Departamento