

INSTITUTO TECNOLÓGICO POPULAR IGUALITARIO ANDRÉS IBÁÑEZ



**SISTEMA DE INFORMACIÓN PARA LA GESTION ACADEMICA DEL INSTITUTO TECNOLOGICO ANDRES IBAÑES**

Alumno: Jhasmany Jhunnior Fernandez Ortega

**Carrera**:

Sistemas Informáticos

**Tutor**:

Ing. Darwin Cuellar Antelo

**Año**: 2017

**Santa Cruz – Bolivia**

Tabla de contenido

[Parte 1 ASPECTOS GENERALES 6](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177519)

[1.1 ANTECEDENTES 7](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177520)

[1.1.1 RESEÑA HISTÓRICA 7](#_Toc500177521)

[1.1.2. DESCRIPCION DEL PROBLEMA 11](#_Toc500177522)

[1.1.3 JUSTIFICACION 12](#_Toc500177523)

[1.2 OBJETIVOS 13](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177524)

[1.2.1 OBJETIVO GENERAL 13](#_Toc500177525)

[1.2.2 OBJETIVOS ESPECIFICOS 13](#_Toc500177526)

[1.3 ALCANCE DEL PROYECTO 14](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177527)

[1.3.1 Módulo de Usuarios: 14](#_Toc500177528)

[1.3.2 MODULO DOCENTE: 15](#_Toc500177529)

[1.3.3 MODULO ESTUDIANTE 15](#_Toc500177530)

[1.3.4 MODULO CARRERA 16](#_Toc500177531)

[1.3.5 MODULO MATERIA 16](#_Toc500177532)

[1.4 ELEMENTOS DE UN SISTEMA BASADO EN COMPUTADORAS 18](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177533)

[1.4.1 HARDWARE 18](#_Toc500177534)

[1.4.1.1 MEDIOS DE COMUNICACIÓN 18](#_Toc500177535)

[1.4.1.2 OTROS DISPOSITIVOS 18](#_Toc500177536)

[1.4.2 SOFTWARE 18](#_Toc500177537)

[1.4.3 GENTE/ USUARIO 19](#_Toc500177538)

[1.4.4 DOCUMENTO 20](#_Toc500177539)

[Parte 2 MARCO TEORICO 24](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177540)

[2.1 ESTRATEGIA PARA EL DESARROLLO DE SOFTWARE 25](#_Toc500177541)

[2.2METODOLOGIA PARA EL DESARROLLO DE SOFTWARE 25](#_Toc500177542)

[2.2.1CARACTERISTICAS DEL PUDS 25](#_Toc500177543)

[2.1.2 CARACTERISTICAS DEL UML 28](#_Toc500177544)

[2.1.3 HERRAMIENTAS 37](#_Toc500177545)

[2.1.4 PROGRAMACION 3 CAPAS 39](#_Toc500177546)

[2.1.5 ESTANDARES DE PROGRAMACION 44](#_Toc500177547)

[Parte 2 CAPTURA DE REQUISITO 46](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177548)

[3.1 INTRODUCCION 47](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177549)

[3.2 LISTA DE REQUERIMIENTOS 48](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177550)

[3.3 MODELO DE NEGOCIO 49](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177551)

[3.3.1 DIAGRAMA DE ACTIVIDAD 49](#_Toc500177552)

[Realizar Inscripción 49](#_Toc500177553)

[Planificar Curso 50](#_Toc500177554)

[Crear currícula o malla académica de la materia 51](#_Toc500177555)

[3.4. MODELO DE CASO DE USO 52](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177556)

[3.4.1. ACTORES Y LISTA DE CASOS DE USO: 52](#_Toc500177557)

[3.4.2. CASO DE USO: (Descripción De Caso Uso) 53](#_Toc500177558)

[3.5 DIAGRAMA GENERAL DE CASOS DE USO 66](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177559)

[PARTE 4 ANALISIS 67](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177560)

[4.1 INTRODUCCION 68](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177561)

[4.2 DIAGRAMA DE PAQUETES 69](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177562)

[4.3 DIAGRAMA DE CLASES (interfaz, control, entidad) 70](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177563)

[4.3 DIAGRAMA DE CLASES (interfaz, control, entidad) 70](#_Toc500177564)

[Parte 5 DISEÑO 77](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177565)

[5.1 INTRODUCCION 78](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177566)

[5.2 ARQUITECTURA DEL SISTEMA 79](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177567)

[5.3 DISEÑO DE CASOS DE USO 80](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177568)

[5.4 DISEÑO DE CLASE 82](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177569)

[5.5 DISEÑO LOGICO DE LA BASE DE DATOS 83](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177570)

[5.6 DISEÑO FISICO DE LA BASE DE DATOS 84](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177571)

[5.7 INTERFAZ 85](file:///C:\Users\user\Documents\proyecto\ITPIAI\ITPIAI%20OFICIAL.docx#_Toc500177572)

[BIBLIOGRAFIA 88](#_Toc500177573)

**INTRODUCCIÒN**

La educación es la más alta función del estado, por tanto, merece especial atención del gobierno, los establecimientos de estudios quienes tienen necesidades durante el proceso de sus actividades académicas, cuyo fin es formar a los educandos que son el presente y el futuro de nuestra Patria.

Y una de las necesidades es contar con herramientas adecuadas para manejar los procesos académicos de tal forma que estas beneficien tanto a los administrativos de la unidad educativa a los docentes como a los alumnos Principales artífices del proceso educativo.

Es por ello que el desarrollo de Software es Mejora el trabajo en las áreas que van avanzando a pasos agigantados, si bien es cierto que hoy la sociedad puede disfrutar de una gran cantidad de software con muchísimas funciones, esta nunca se percata de la existencia del desarrollo del software como tal, para la sociedad el software hoy en día es importante para poder subsistir, sin importar de donde provenga.

La tecnología ha cambiado la forma en que las personas se comunican y ha creado nuevas posibilidades de transmitir cultura, información, educación y conocimiento. La pedagogía, con sus métodos de enseñanza, aprendizaje e investigación, ha evolucionado debido a la tecnología, a tal punto que ahora estos métodos son más variados y dinámicos.

Para esto se ha realizado la implementación de un sistema de información que garantice el control de estos servicios brindados, facilitando el registro de los equipos y la obtención de datos de los mismos, para así tener una información más eficiente de los servicios realizados.

**Parte 1**

# Parte 1 ASPECTOS GENERALES

## 1.1 ANTECEDENTES

1.1.1 RESEÑA HISTÓRICA

En el 2007, los habitantes de la zona del Plan 3000 se reúnen en una asamblea convocada por el presidente de la Unión de juntas vecinales del Distrito 8 de la ciudad de Santa Cruz, con el único objetivo de considerar la apertura de una Institución de Educación Superior porque sus hijos e hijas no podían acceder a este estudio por la distancia y el factor económico.

Participación de la reunión varios vecinos, entre ellos, obreros gremiales, artesanos, chóferes juntas vecinales y otras fuerzas sociales, en primera instancia solicitaban al Gobierno, la apertura de una Universidad, su pedido fue negado; sin embargo a cambio les ofrecieron la apertura de un Tecnológico, al que se le da el nombre de: Instituto Tecnológico Popular Igualitario “Andrés Ibáñez", nombre en honor al inmortal y primer igualitario y socialista latino Dr. Andrés Ibáñez.

En fecha 20 de mayo del 2008, se logra la Resolución Ministerial Nº 304/08 autorizando la creación, apertura y legal funcionamiento del Instituto Tecnológico Popular Igualitario “Andrés Ibáñez" de carácter público, con domicilio en la Avenida Che Guevara, frente a la plaza 18 de marzo zona del Plan 3000 de la ciudad de Santa Cruz, Colegio “Saint Andrew”. El señor Edwin Hernán Grimaldos Durán de Castro, en su libre y espontánea voluntad, brinda en calidad de préstamo, gratuito y por tiempo indefinido al Ministerio de Educación y Cultura su inmueble, ubicado en la U.V. 147, Mza. 30, lugar donde funcionaba el “Colegio Saint Andrew”, para el funcionamiento del Instituto Tecnológico Popular Igualitario “Andrés Ibáñez". Desde la gestión 2011 a la actualidad, el instituto funciona en la Unidad Educativa Fiscal “Luis Antonio Añez”. El 3 de marzo de la presente gestión se recibió la nueva infraestructura construida con el programa “Bolivia Cambia, Evo Cumple”, en el Plan 3000 en la OTB Suarez Pompeyus, UV 165 Manzana M28, con una superficie de terreno de 8.304,09 m2 y una superficie construida de 2.515,75 m2.

Funciona en turno de la noche, actualmente desde horas 19:00 a 22:30, de lunes a viernes. En la presente gestión, el 3 de marzo,

La creación del instituto fue consolidado con 11 ítems para docentes. En la gestión 2012, se hace responsable de la Rectoría el Ing. Raúl Guzmán, consiguiendo el ítem para el cargo y como Coordinadora Académica se encontraba la Ing. Zulma Dorado Barboza, durante esa gestión se consiguen 15 ítems más para Docentes y un ítem para Portero. En Mayo de la gestión 2013 egresan 6 estudiantes de la Carrera de Agronomía, ninguno de ellos defendió alguna modalidad de graduación. Después de varias solicitudes, en la gestión 2013 se consolida el ítem para la Dirección Académica, en Agosto del mismo año, mediante Convocatoria, asume el cargo la Lic. Fridda Elena Rodríguez.

Desde su fundación, el Instituto no había graduado a ninguno de los egresados de las gestiones anteriores, fue en la gestión académica de la Lic. Rodríguez que se realizan todos los Trámites ante la Dirección Departamental de Educación y Sub Dirección de Educación Superior de Formación Profesional, para que los estudiantes realicen us Proyectos de Grado así como la defensa de los mismos. Se elaboraron los primeros libros históricos de notas de las gestiones pasadas para regularizar, ya que hasta entonces la Dirección no contaba con ninguno de ellos.

En la gestión 2013, se gradúan los primeros 17 estudiantes a través de la modalidad de graduación defensa de Proyecto de Grado, en las Carreras de Alimentos y Comunicación Popular y Alternativa. En la gestión 2014 gradúan 37 estudiantes de las Carreras de Agronomía, Alimentos y Comunicación Popular y Alternativa.

**1.1.1.1 SITUACION ACTUAL**

Instituto “ITIPAI” en su actualidad no cuenta con ningún tipo de software sistematizado, que lleve el control de todas las informaciones del estudiante, ya que los datos, documentos y planillas de control están transcritas de forma manual.

Y unas de las complicaciones que se tiene en el manejo de estas planillas transcritas manualmente, es que es difícil llevar un control eficaz de todas estas informaciones.

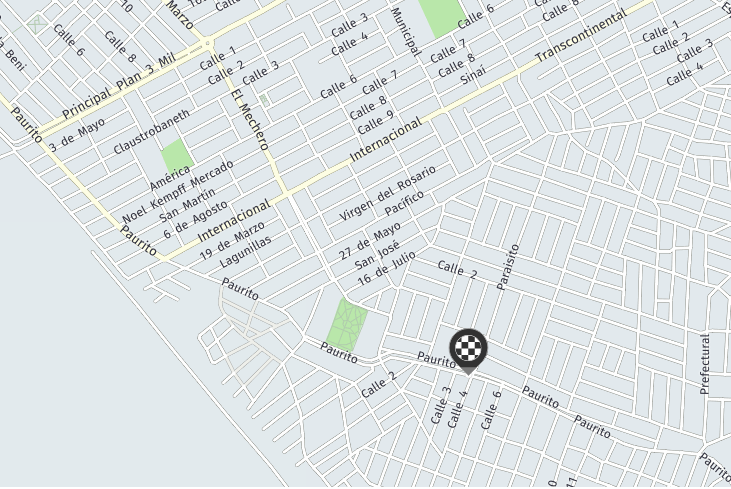
**1.1.1.2 MISIÓN**

A través de la educación superior Tecnológica, ser un agente de desarrollo de la sociedad, atendiendo sus necesidades y expectativas, con la participación entusiasta de nuestra comunidad, unida y en constante búsqueda de la excelencia.

**1.1.1.3 CARRERAS**

* Agronomía
* Comunicación Popular y Alternativa
* Sistemas de Regulación y Control Automáticos
* Informática Industrial
* Industria de Alimentos
* Sistemas Informáticos
* Sistemas de Telecomunicación

**1.1.1.4 MAPA**



**1.1.1.5 FOTOGRAFIA**



1.1.2. DESCRIPCION DEL PROBLEMA

La institución ITPIAI en estos últimos años no cuenta con un software de registro y pago de matrícula, en los últimos años ha experimentado la aglomeración de documentos y registros. Con la acumulación de información en el transcurso de los tiempos.

Con todo lo mencionado anteriormente se identifica los siguientes problemas:

* El llenado de todos los registros de información de los estudiantes se hace de forma manual, lo que ocasiona demora en laatención del cliente.
* La aglomeración de bastante folder de registros y estas son propensas a perderse o dañarse.
* En caso de algún reclamo o consulta, se realiza una búsqueda entre la documentación existente, lo cual implica pérdida de tiempo, que ocasiona molestia al inscrito que espera una respuesta inmediata a la petición.
* La forma de trabajo con los reportes de notas toma un tiempo importante elaborarlas.
* En caso de necesitar las notas que tiene un alumno se debe de buscar el Kardex, y esperar que este actualizada.
* La actualización de los datos en cuestión de notas y registro de estudiante se hacen de forma manual y presencial, lo que conlleva a una pérdida de tiempo en las actividades normales.
* Los datos al no tener un respaldo fijo de copias de seguridad, tienden a ser irrecuperables.
* En caso de perder documentos no se notaría la diferencia hasta llegar a necesitarlos, y es muy posible que la información ya no esté al alcance cercano.
* Luego de haber iniciado la gestión académica, se procede a la elaboración del horario para cada curso, es decir, distribuir la carga horaria de los docentes, de acuerdo a su ítem, en los diferentes cursos de la Unidad Educativa.
* La inscripción y el control de notas de los alumnos, exige mucho tiempo y esfuerzo debido a que toda la información se maneja de forma manual entonces viendo estas deficiencias el factor más importante es poder contar con un sistema para mejorar el manejo de la información para el mejor control académico de todo lo referente a los alumnos de la institución, así como también para informar con datos correctos de cada alumno.

**1.1.2.1 SITUACION DESEADA**

Se desea desarrollar un Sistema de control de Información para el área administrativo y docentes, que controle de manera sistemática los procesos de información requeridas, y dar solución a los problemas citados anteriormente para tener una organización eficaz y poder brindar un mejor servicio al estudiante.

1.1.3 JUSTIFICACION

La razón por la cual se desarrolla este sistema es para cubrir las necesidades de la institución ITPIAI, ya que no cuenta con un sistema que controle los trabajos más realizados dentro de ella, así como también el registro de sus estudiantes, docente y pagos.

Todos los Sistemas de Información utilizan herramientas necesarias para ayudar en el trabajo dentro de una institución que requiere procesar información de la forma más rápida brindando un mejor servicio a sus clientes.

Así mismo este sistema puede dar como beneficio un manejo adecuado y ordenado de las planillas de registros, así como también tiene facilidad de manejo del sistema, esto optimizara el desarrollo de la información de la institución.

## 1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

Desarrollar un sistema de información para la gestión académica del instituto tecnológico Andrés Ibáñez.

1.2.2 OBJETIVOS ESPECIFICOS

* Realizar una recolección de información necesaria, para capturar los requisitos funcionales que se tendrán en cuenta.
* Recabar los formatos de boletines, libros centralizadores de notas, cuadro de filiación, control de asistencia y otros documentos los cuales servirán como requisitos iniciales para la elaboración del proyecto.
* Analizar las políticas de negocio que siguen los documentos para tener entendimiento del ámbito de trabajo.
* Elaborar con toda la información obtenida un análisis de los requisitos a través de desarrollo de modelos utilizando Casos de Uso.
* Establecer las clases necesarias las cuales cubran todos los datos recopilados y respaldados por los anexos que nos fueron confiados.
* Diseñar las interfaces, la base de datos y todo lo que indica el proceso de desarrollo de la manera más correcta posible.
* Implementar lo que se ha diseñado en la aplicación Visual Estudio .NET Basic y SGBD SQL Server.
* Realizar las pruebas necesarias para justificar el buen funcionamiento del sistema y/o encontrar posibles fallas y luego eliminarlas.

Teniendo en cuenta los objetivos de forma clara, se pretenderá alcanzar las mismas, con el fin de tener un software eficiente y de calidad.

## 1.3 ALCANCE DEL PROYECTO

Lograr el desarrollo de un software para la “Sistematización de datos, que controle la información de los estudiantes, docentes, pagos de matrícula, registro de inscripción, control de materias, promedio de notas, control de usuarios activos en el sistema de la institución ITPIAI.

Para ello describiremos los distintos módulos que existen en el proyecto:

1.3.1 Módulo de Usuarios:

En este módulo tendremos y administraremos el acceso a la aplicación, dependiendo el tipo de usuario, además de llevar una bitácora de lo que el usuario realiza dentro del sistema, se controlara los datos personales de cada usuario y sus teléfonos de cada usuario.

Sus funciones principales **son**:

**Gestionar Bitácora:** se manipulará el registro de las acciones del usuario donde los datos que más tomaremos en cuenta son.

* Fecha de la acción.
* acción que realizo.
* tipo de acción que se realizó.
* Usuario.

**Gestionar Usuario:** aquí se administrará el acceso de los distintos usuarios, con los datos personales que se le atribuyen que son.

* User\_name
* Password
* carnet de identidad
* nombre completo
* sexo
* domicilio
* correo electrónico
* nacionalidad
* fecha de nacimiento
* números telefónicos.

1.3.2 MODULO DOCENTE:

En este módulo se manipulará los datos del docente y sus estudios de especialidad que ha llegado a obtener.

Sus funciones principales **son:**

**Gestionar Docente:** Se administrará la información que se tiene del docente.

**Gestionar Especialidad:** Se realizará el registro, modificación y cancelación de las especialidades de un docente, para que se le adjunte, según sea necesario.

1.3.3 MODULO ESTUDIANTE

En este módulo se administrará el Kardex del estudiante con su respectivo avance, además las materias inscritas, el pago que realiza para poder estar habilitado en sistema.

Sus funciones principales **son:**

**Gestionar Kardex:** se administra los datos del Kardex como ser:

* Serie de título bachiller
* Numero de título de bachiller
* Año emisión de título de bachiller
* Fecha de Kardex

**Gestionar Estudiante:** se manipulará los datos del estudiante y el nivel en el que se encuentra de su carrera.

**Gestionar Inscripción:** Se registrará, modificará y cancelará la inscripción del alumno, con sus materias respectivas.

**Gestionar Pago Matricula:** se registrará los pagos hechos por el estudiante registrando los datos: monto a pagar, saldo y tipo de pago.

**Gestionar Nivel:** será para tener en cuenta el nivel que tiene el estudiante

1.3.4 MODULO CARRERA

En este módulo se tendrá en cuenta las diferentes carreras que, si dan en la institución, teniendo en cuenta su modalidad, si es anual o semestral, y creando la malla curricular con las diferentes materias que tendrá.

Sus funciones principales **son:**

**Gestionar Carrera:** se registrará los datos de las diferentes materias que se tendrán en la institución, así como actualizar y cancelar dichas materias, se tendrá en cuenta los siguientes datos:

* Nombre
* Modalidad
* Duración

**Gestionar currículo:** se manipulará los datos de las materias que se tendrá para esta malla curricular y para ello se manipularan los datos:

* Fecha
* Estado
* carreras

1.3.5 MODULO MATERIA

Para este módulo además de los datos de la materia se gestionará el programa de la materia para las diferentes gestiones, como ser el aula, los días y horarios.

**Gestionar Materias:** se registrará, modificará y cancelará las diferentes materias y se tendrá en cuenta los siguientes datos:

* Sigla
* Nombre
* Carga horaria

**Administrar Gestiones:** se registrará, modificará y cancelará las diferentes gestiones anuales y semestrales, para ello se tendrá en cuenta los siguientes datos:

* Fecha inicio
* Fecha final
* Numero de gestión

**Gestionar Programación de Materias:** se tendrá en cuenta la información para realizar la programación en cuanto a los días, aula y horarios que se pasaran en cada materia de las gestiones.

* Docente
* Fecha inicio
* Fecha final
* Gestión a la que pertenece
* Materias inscritas

**Gestionar periodo:** se registrará, modificará y cancelará los diferentes periodos de las materias con los siguientes datos:

* Aula
* Hora
* día

**Gestionar Aula:** se registrará, modificará y cancelará las aulas con los siguientes datos:

* capacidad
* estado
* nombre

**Gestionar Hora:** se tendrá en cuenta todos los distintos horarios para los periodos:

* hora inicio
* hora final

## 1.4 ELEMENTOS DE UN SISTEMA BASADO EN COMPUTADORAS

1.4.1 HARDWARE

El cliente accederá al sistema mediante una computadora de escritorio con las especificaciones mínimas siguientes:

* Procesador: Intel® Core(TM)2 Duo CPU E7400 @2.80GHz 2.79GHz
* Memoria instalada (RAM): 2,00 GB
* Un monitor en el cual se visualizaran los resultados que arroje la aplicación
* Disco duro con espacio mínimo disponible 16GB

1.4.1.1 MEDIOS DE COMUNICACIÓN

* Switch, (para la conexión de red)
* Wii-Fi (para la comunicación inalámbrica de la información)

1.4.1.2 OTROS DISPOSITIVOS

* Impresora
* Escáner

1.4.2 SOFTWARE

* SQL server 2017
* Lenguaje de programación C# de Visual Studio .Net 2017 Community
* Sistema operativo WINDOWS 7 y WINDOWS 8 o superior

**1.4.2.1 OTRO SOFTWARE ADICIONAL**

Se utilizaran las siguientes herramientas adicionales para la visualización de los reportes:

* Adobe Reader
* Access
* Excel

1.4.3 GENTE/ USUARIO

**1.4.3.1 SUPER – USUARIO**

Para llevar un buen control de seguridad en el sistema, se ha decidido que se crearán roles con privilegios específicos, dentro de éstos roles se agruparán usuarios, que heredarán los permisos de acuerdo al rol que pertenece.

**Administrador del sistema**

El cual podrá solicitar reportes, modificar datos, etc. En otras palabras un control total del sistema

**1.4.3.2 USUARIO**

**Docente:** Este tendrá acceso y uso de las notas, además de la lista de sus alumnos, colocar faltas y datos referentes a sus materias dadas.

**Secretaria:** Sera una persona que tiene acceso a la administración de la administración de una forma regulada

**Estudiante:** Es la persona que cursa las materias, donde tendrá acceso a las mismas y podrá ver las notas que tiene hasta el momento

**Administrador:** Es el encargado de gestionar las currículos, materias, programas y todo lo que se pueda orientar a la administración de las materias a cursar.

Dentro del administrador existirán cargos para poder distribuir el trabajo y responsabilidades.

1.4.4 DOCUMENTO

**1.4.4.1 Para el usuario:**

**Manual de usuario**

En este documento se incluye instrucciones sobre el manejo del sistema, además de una lista de preguntas frecuentes entre otras cosas.

• Manual de usuario digital: Este documento se incluye en la aplicación web y contiene instrucciones sobre el manejo del sistema y ayuda online. A obtener:

• Entrevistas: Los informes resultantes de las diversas entrevistas realizadas al personal de la empresa.

• Políticas de la empresa: Serán necesarias para poder definir las políticas de negocio que deberá efectuar el sistema, así como para identificar aquellas que se manejaran de forma manual.

**Estrategia para el desarrollo del proyecto**

**• Método del ciclo de vida de desarrollo de software**

Este es un método que debe cumplirse, sin pararse por alto ninguno de las siguientes etapas:

a) Investigación y estudio preliminar.

b) Determinación de los requerimientos de los sistemas.

c) Diseño del sistema.

d) Desarrollo del software.

e) Prueba del sistema.

f) Implantación y evaluación del nuevo sistema.

**Metodología para el desarrollo del proyecto**

Proceso Unificado de Desarrollo de Software (PUDS)

El Proceso Unificado es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos.

**Es un proceso ORIENTADO A OBJETOS.**

Características. El proceso es:

• Guiado por casos de uso.

• Centrado en la arquitectura.

• Con un ciclo de vida iterativo e incremental.

El Proceso Unificado usa el Lenguaje de Modelado Unificado (UML) en la preparación de todos los planos del sistema. De hecho, UML es una parte integral del Proceso Unificado, fueron desarrollados a la par. Organiza el desarrollo de software en cuatro fases, cada una de ellas con la ejecución de una o más iteraciones de desarrollo de software: creación, elaboración, construcción y las directrices (Transición).

El Proceso Unificado consiste en una serie de disciplinas o flujos de trabajo que van desde los requisitos hasta las pruebas. Los flujos de trabajo desarrollan modelos desde el modelo de casos de uso hasta el modelo de pruebas.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Disciplina** | **Modelos** | | | | |
| **Requisitos** Modelo de Caso de Uso | | | | | |
| **Análisis** Modelo de Análisis | | | | | |
| **Diseño** | Modelo de | Diseño | – | Modelo | de Despliegue |
|  |  |  |  |  |  |
| **Implementación** Modelo de Implementación | | | | | |
| **Prueba** Modelo de Prueba | | | | | |

**Fases dentro del Ciclo de Vida del Proceso Unificado de Sistemas**

**Iniciación:**

Describir producto final /análisis del negocio. Identificar riesgos más importantes.

Establecer planificación inicial del proyecto. Decidir si se continúa.

**Elaboración:**

Establecer plan y arquitectura estable.

**Construcción:**

Desarrollar el producto.

**Transición:**

Proporcionar sistema a usuarios.

Cada fase se divide en iteraciones, cada iteración mini proyecto (encascada) que ejecuta flujos de trabajo produce un incremento en producto tal y como estaba se reduce el riesgo se puede perder sólo lo realizado en esa iteración.

**Flujo de Trabajo**

**Requisitos:**

* Identificar requisitos del sistema.
* Construir un modelo del mismo.
* Modelo de caso de uso.
* Modelo del dominio (o negocio)

**Análisis:**

* Especificar requisitos.
* Construir modelo del análisis.

**Diseño:**

 Encontrar la forma del sistema (solución).

 Construir modelo del diseño, Implementación

 Codificar el diseño (solución).

 Construir modelo de implementación.

**Pruebas:**

 Verificarla implementación.

 Construir modelo de pruebas.

 El computador deberá constar con el sistema operativo Windows 10

 El Sistema de Información se desarrollara en una plataforma de escritorio para Windows con .NET es el entorno de elaboración por el que optamos para trabajar este sistema.

 Y utilizando Enterprise Architect visualizaremos todos los Flujos de trabajo.

**Parte 2**

# Parte 2 MARCO TEORICO

## 2.1 ESTRATEGIA PARA EL DESARROLLO DE SOFTWARE

Como estrategia del desarrollo de software se utilizara el Proceso Unificado de desarrollo de software (PUDS) puesto que provee un enfoque disciplinado en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

Con el PUDS se podrá:

* Las iteraciones y construcciones proporcionan reducen la proporción de las tareas, grupos de trabajo y permiten un constante control de riesgos y realimentaciones.
* Permite un lenguaje común como lo es UML, convirtiendo al desarrollo de software una disciplina de ingeniería en vez de meramente escribir código.
* Al emplear las tecnologías de componentes, se puede emplear la reutilización, reduciendo el tiempo y los costes de desarrollo

## 2.2METODOLOGIA PARA EL DESARROLLO DE SOFTWARE

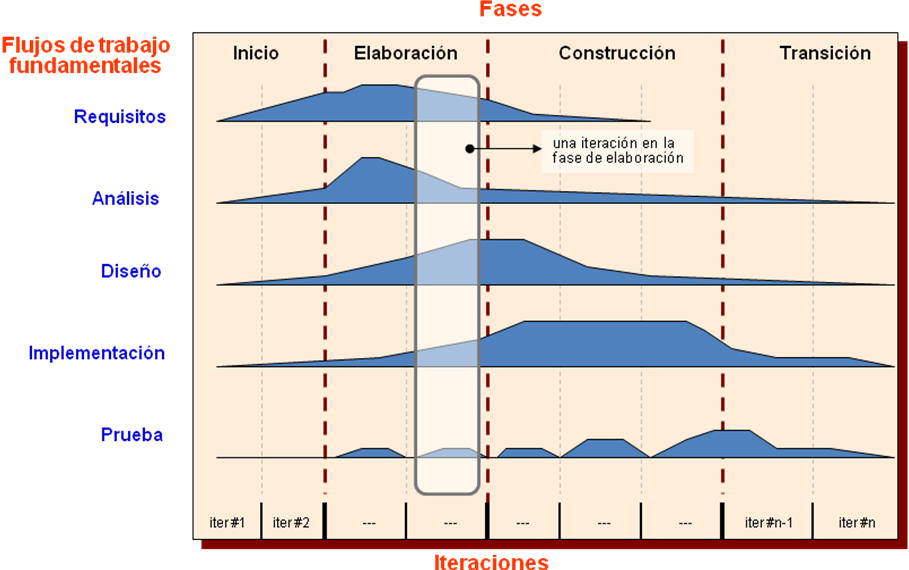
2.2.1CARACTERISTICAS DEL PUDS

Entre las principales características del Proceso Unificado de Desarrollo de Software están:

* Iterativo e Incremental
* Dirigido por los casos de uso
* Centrado en la arquitectura
* Enfocado en los riesgos

 Resumiendo las características del Proceso Unificado están capturados en tres conceptos clave: dirigido por casos de uso (use-case drive), centrado en la arquitectura (architecture-centric), iterativo e incremental. Esto es lo que hace único al Proceso Unificado

**Flujo de trabajo:**

****

* **Los requisitos**

Es el proceso de averiguar, normalmente en circunstancias difíciles, lo que se debe construir.

* **Análisis**

Analizamos los requisitos que se describieron en la captura de requisitos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema entero.

* **Diseño**

En el diseño modelamos el sistema y encontramos su forma para que soporte todos los requisitos incluyendo los requisitos no funcionales y otras restricciones que se le suponen.

* **Implementación**

En la implementación empezamos con el resultado del diseño e implementamos el sistema en términos de componentes, es decir ficheros de código fuente, scripts, ficheros de código binarios ejecutable y similar.

* **Prueba**

Verificamos el resultado de la implementación probando cada construcción, incluyendo construcciones intermedias, así como las versiones finales del sistema a ser entregadas a terceros.

**Fases del Proceso Unificado:**



**Fase de** **concepción o inicio** tiene por finalidad definir la visión, los objetivos y el alcance del proyecto, tanto desde el punto de vista funcional como del técnico, obteniéndose como uno de los principales resultados una lista de los casos de uso y una lista de los factores de riesgo del proyecto. El principal esfuerzo está radicado en el Modelamiento del Negocio y el Análisis de Requerimientos. Es la única fase que no necesariamente culmina con una versión ejecutable.

**Fase de** **elaboración** tiene como principal finalidad completar el análisis de los casos de uso y definir la arquitectura del sistema, además se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.

**Fase de** **construcción** está compuesta por un ciclo de varias iteraciones, en las cuales se van incorporando sucesivamente los casos de uso, de acuerdo a los factores de riesgo del proyecto. Este enfoque permite por ejemplo contar en forma temprana con versiones el sistema que satisfacen los principales casos de uso. Los cambios en los requerimientos no se incorporan hasta el inicio de la próxima iteración

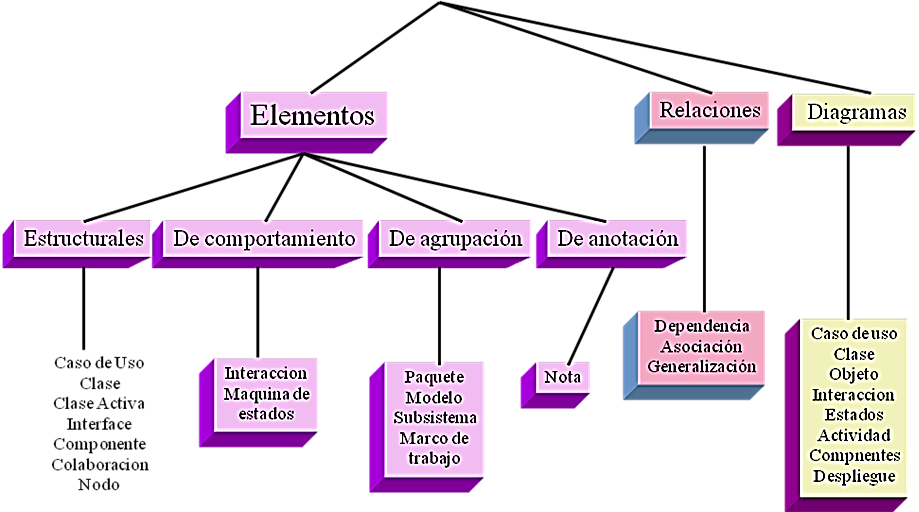
**Fase de** **transición** se inicia con una versión “beta” del sistema y culmina con el sistema en fase de producción.

2.1.2 CARACTERISTICAS DEL UML

El UML ofrece un estándar para describir un "plano" del sistema modelo, incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

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.

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML ofrece nueve diagramas en los cuales modelar sistemas. Algunos diagramas que provee el UML son:

* Diagramas de Casos de Uso para modelar los procesos ’business’.
* Diagramas de Secuencia para modelar el paso de mensajes entre objetos.
* Diagramas de Colaboración para modelar interacciones entre objetos.
* Diagramas de Estado para modelar el comportamiento de los objetos en el sistema.
* Diagramas de Actividad para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
* Diagramas de Clases para modelar la estructura estática de las clases en el sistema.
* Diagramas de Objetos para modelar la estructura estática de los objetos en el sistema.
* Diagramas de Componentes para modelar componentes.
* ****Diagramas de Implementación para modelar la distribución del sistema

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **E**  **L**  **E**  **M**  **E**  **N**  **T**  **O**  **S**  **E**  **S**  **T**  **R**  **U**  **C**  **T**  **U**  **R**  **A**  **L**  **E**  **S** | Clase | | elemento_estructura_clase | Describe un conjunto de objetos que comparten los mismos atributos, métodos, relaciones y semántica. Las clases implementan una o más interfaces. |
| Clase activa | | elemento_estructura_clase_activa | Se trata de una clase, en la que existe procesos o hilos de ejecución concurrentes con otros elementos. Las líneas del contorno son más gruesas que en la clase “normal” |
| Interfaz | | elemento_estructura_interfaz | Agrupación de métodos u operaciones que especifican un servicio de una clase o componente, describiendo su comportamiento, completo o parcial, externamente visible. UML permite emplear un círculo para representar las interfaces, aunque lo más normal es emplear la clase con el nombre en cursiva. |
| Colaboración | | elemento_estructura_colaboracion | Define una interacción entre elementos que cooperan para proporcionar un comportamiento mayor que la suma de los comportamientos de sus elementos. |
| Caso de uso | | elemento_estructura_caso_de_uso | Describe un conjunto de secuencias de acciones que un sistema ejecuta, para producir un resultado observable de interés. Se emplea para estructurar los aspectos de comportamiento de un modelo. |
| Componente | | elemento_estructura_componente | Parte física y por tanto reemplazable de un modelo, que agrupa un conjunto de interfaces, archivos de código fuente, clases, colaboraciones y proporciona la implementación de dichos elementos. |
| Nodo | | elemento_estructura_nodo | Elemento físico que existe en tiempo de ejecución y representa un recurso computacional con capacidad de procesar. |
| **Elementos**  **de**  **comportamiento** | Interacción | elemento_comportamiento_interaccion | | Comprende un conjunto de mensajes que se intercambian entre un conjunto de objetos, para cumplir un objetivo especifico. |
| Máquinas  de  estados | elemento_estructura_estado | | Especifica la secuencia de estados por los que pasa un objeto o una interacción, en respuesta a eventos. |
| **Elementos**  **de**  **agrupación** | Paquete | elemento_agrupacion_paquete | | Se emplea para organizar otros elementos en grupos. |
| **Elementos**  **de**  **notación** | Nota | elemento_anotacion_nota | | Partes explicativa de UML, que puede describir textualmente cualquier aspecto del modelo |

**Relaciones:**

|  |  |  |
| --- | --- | --- |
| Dependencia | relaciones_dependencia | Es una relación entre dos elementos, tal que un cambio en uno puede afectar al otro. |
| Asociación | relaciones_asociacion | Es una relación estructural que resume un conjunto de enlaces que son conexiones entre objetos. |
| Generalización | relaciones_generalizacion | Es una relación en la que el elemento generalizado puede ser substituido por cualquiera de los elementos hijos, ya que comparten su estructura y comportamiento. |
| Realización | relaciones_realizacion | Es una relación que implica que la parte realizante cumple con una serie de especificaciones propuestas por la clase realizada (interfaces). |

**Diagramas:**

|  |  |  |  |
| --- | --- | --- | --- |
| **M**  **O**  **D**  **E**  **L**  **A**  **N**  **E**  **S**  **T**  **R**  **U**  **C**  **T**  **U**  **R**  **A** | Clases | diagramas_clases | Muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones, cubriendo la vista de diseño estática del sistema. |
| Objetos |  | Análogo al diagrama de clases, muestra un conjunto de objetos y sus relaciones, pero a modo de vista instantánea de instancias de una clase en el tiempo. |
| Componentes | diagramas_componentes | Muestra la organización y dependencias de un conjunto de componentes. Cubren la vista de implementación estática de un sistema. Un componente es un módulo de código, de modo que los diagramas de componentes son los análogos físicos a los diagramas de clases. |
| Despliegue | diagramas_despliegue | Muestra la configuración del hardware del sistema, los nodos de proceso y los componentes empleados por éstos. Cubren la vista de despliegue estática de una arquitectura. |
| **M**  **O**  **D**  **E**  **L**  **A**  **N**  **D**  **O**  **C**  **O**  **M**  **P**  **O**  **R**  **T**  **A**  **M**  **I**  **E**  **N**  **T**  **O** | Casos de Uso | diagramas_casos_de_uso | Muestra un conjunto de casos de uso, los actores implicados y sus relaciones. Son diagramas fundamentales en el modelado y organización del sistema. |
| Secuencia | diagramas_secuencia | Son diagramas de interacción, muestran un conjunto de objetos y sus relaciones, así como los mensajes que se intercambian entre ellos. Cubren la vista dinámica del sistema. El diagrama de secuencia resalta la ordenación temporal de los mensajes, mientras que el de colaboración resalta la organización estructural de los objetos, ambos siendo equivalentes o isomorfos. En el diagrama de colaboración de la figura de la izquierda, se puede ver que los elementos gráficos no son cajas rectangulares, como cabría esperar, y en su lugar encontramos sus versiones adornadas. Estas versiones tienen como finalidad evidenciar un rol específico del objeto siendo modelado. En la figura encontramos de izquierda a derecha y de arriba abajo un Actor, una Interfaz, un Control (modela un comportamiento) y una Instancia (modela un objeto de dato). |
| Colaboración | diagramas_colaboracion |
| Estados  Actividades | diagramas_actividadesdiagramas_estados | Muestra una máquina de estados, con sus estados, transiciones, eventos y actividades. Cubren la vista dinámica de un sistema. Modelan comportamientos reactivos en base a eventos.  Tipo especial de diagrama de estados que muestra el flujo de actividades dentro de un sistema. |
|  |  |  |

**Aspecto Estático y Dinámicos:**

Se Dispone de dos tipos diferentes de diagramas los que dan una vista estática del sistema y los que dan una visión dinámica.

Los diagramas estáticos son:

* Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
* Diagrama de objetos: Es un diagrama de instancias de las clases mostradas en el diagrama de clases. Muestra las instancias y como se relacionan entre ellas. Se da una visión de casos reales.
* Diagrama de componentes: Muestran la organización de los componentes del sistema. Un componente se corresponde con una o varias clases, interfaces o colaboraciones.
* Diagrama de despliegue.: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
* Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y relaciones existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.
* Lo diagramas dinámicos son:
* Diagrama de secuencia, Diagrama de colaboración: Muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin pérdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.
* Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
* Diagrama de actividades: Es un caso especial  del diagrama de estados. Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

Como podemos ver el número de diagramas es muy alto, en la mayoría de los casos excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos. En el documento se dará una breve explicación de todos, ampliándose para los más necesarios.

**Diagrama de Clases**

Los diagramas de clases representan un conjunto de elementos del modelo que son estáticos, como las clases y los tipos, sus contenidos y las relaciones que se establecen entre ellos. Algunos de los elementos que se pueden clasificar como estáticos son los siguientes:

**Paquete:** Es el mecanismo de que dispone UML para organizar sus elementos en grupos se representa un grupo de elementos del modelo. Un sistema es un único paquete que contiene el resto del sistema, por lo tanto, un paquete debe poder anidarse, permitiéndose que un paquete contenga otro paquete.

**Clases:** Una clase representa un conjunto de objetos que tienen una estructura, un comportamiento y unas relaciones con propiedades parecidas. Describe un conjunto de objetos que comparte los mismos atributos, operaciones, métodos, relaciones y significado. En UML una clase es una implementación de un tipo. Los componentes de una clase son:

**Atributo:** Se corresponde con las propiedades de una clase o un tipo. Se identifica mediante un nombre. Existen atributos simples y complejos.

Operación. También conocido como método, es un servicio proporcionado por la clase que puede ser solicitado por otras clases y que produce un comportamiento en ellas cuando se realiza.

2.1.3 HERRAMIENTAS

La descripción de los programas que se llegaran a utilizar para el desarrollo del sistema es:

**2.1.3.1 SQL Server Management Studio 2017** (**SSMS**)

Es un entorno integrado para la gestión de cualquier infraestructura de SQL de SQL Server a través de la base de datos SQL. SSMS proporciona herramientas para configurar, supervisar y gestionar instancias de SQL Server disponible dondequiera que se va a implementar SQL Server. Además, SSMS herramientas para la implantación, supervisión y actualización de los componentes de capa de datos que a medida que las bases de datos utilizadas por las aplicaciones y los almacenes de datos, y para crear consultas y scripts.

**Microsoft SQL Server** es un sistema de manejo de bases de datos del modelo relacional, desarrollado por la empresa Microsoft.

El lenguaje de desarrollo utilizado (por línea de comandos o mediante la interfaz gráfica de Management Studio) es Transact-SQL (TSQL), una implementación del estándar ANSI del lenguaje SQL, utilizado para manipular y recuperar datos (DML), crear tablas y definir relaciones entre ellas (DDL).

Dentro de los competidores más destacados de SQL Server están: Oracle, MariaDB, MySQL, PostgreSQL. SQL Server solo está disponible para sistemas operativos Windows de Microsoft.

Puede ser configurado para utilizar varias instancias en el mismo servidor físico, la primera instalación lleva generalmente el nombre del servidor, y las siguientes - nombres específicos (con un guion invertido entre el nombre del servidor y el nombre de la instalación).

**2.1.3.2 Microsoft Visual Studio 2017**

Es un conjunto de herramientas y otras tecnologías de desarrollo de software basado en componentes para crear aplicaciones eficaces y de alto rendimiento, permitiendo a los desarrolladores crear sitios y aplicaciones web, así como otros servicios web en cualquier entorno que soporte la plataforma.

**Visual Studio** En palabras más específicas, Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos los mismos entornos de desarrollo integrado (IDE), que habilita el uso compartido de herramientas y facilita la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones web ASP y Servicios Web XML.

**2.1.3.3 C SHARP (C#)**

C# es un lenguaje elegante, con seguridad de tipos y orientado a objetos, que permite a los desarrolladores crear una gran variedad de aplicaciones seguras y sólidas que se ejecutan en .NET Framework .NET. Puede usar C# para crear aplicaciones cliente de Windows, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos y muchas, muchas más cosas. Visual C# proporciona un editor de código avanzado, prácticos diseñadores de interfaz de usuario, un depurador integrado y muchas otras herramientas que facilitan el desarrollo de aplicaciones basadas en el lenguaje C# y .NET Framework.

2.1.4 PROGRAMACION 3 CAPAS

Cuando se construye software como producto empresarial o comercial, se llevan a cabo varias técnicas de manera que el desarrollo se haga en forma ordenada y así poder asegurar un avance continuo del proyecto, un producto final de calidad, y además realizar posteriores mejoras sea una tarea más fácil. Existen muchas prácticas de programación, dependiendo del tipo de software que se va a desarrollar y de la disciplina o disciplinas de programación que se utilicen en el desarrollo del producto. Una de las más utilizadas se llama la programación por capas, que consiste en dividir el código fuente según su funcionalidad principal.

La programación para lograr sacarle el mayor provecho a la programación por capas se necesita seguir una serie de pasos complejos los cuales primeramente deben ser definidos para cada proyecto en específico, luego deben ser revisados para asegurarse de que el modelo adoptado cumpla con las normas necesarias para que la aplicación sea del agrado del usuario, y por último debe ser implementado por el grupo de desarrollo encargado para tal fin, los cuales siguiendo el modelo propuesto obtienen una herramienta útil para facilitar la labor de programación dividiendo la aplicación en módulos y capas fáciles de pulir.

**2.1.4.1 Características de la Programación en Capas.**

La programación por capas es una técnica de ingeniería de software propia de la programación por objetos, éstos se organizan principalmente en 3 capas: la capa de presentación o frontera, la capa de lógica de negocio o control, y la capa de datos.

Siguiendo el modelo, el desarrollador se asegura avanzar en la programación del proyecto de una forma ordenada, lo cual beneficia en cuanto a reducción de costos por tiempo, debido a que se podrá avanzar de manera más segura en el desarrollo, al ser dividida la aplicación general en varios módulos y capas que pueden ser tratados de manera independiente y hasta en forma paralela.

Por otra parte, otra característica importante de recalcar es la facilidad para las actualizaciones de la aplicación. En este aspecto, la programación en capas juega un papel de suma importancia ya que sigue un estándar conocido en el ambiente de desarrollo de aplicaciones, lo cual da al programador una guía para hacer mejoras a la aplicación sin que esto sea una tarea tediosa y desgastante, siguiendo el estándar establecido para tal fin y dividiendo las tareas en partes específicas para cada capa del proyecto.

Las principales capas que siempre deben estar en este modelo son:

**2.1.4.3 Capa de Presentación o Frontera:**

La presentación del programa ante el usuario, debe manejar interfaces que cumplan con el objetivo principal de este componente, el cual es facilitar al usuario la interacción con la aplicación. Para esto se utilizan patrones predefinidos para cada tipo de aplicación y para cada necesidad del usuario.

La interfaz debe ser amigable y fácil de utilizar, ya que el usuario final es el que se va a encargar de utilizar el sistema y de dar retroalimentación al equipo de desarrollo en caso de que haya algo que mejorar.

Las interfaces deben ser consistentes con la información que se requiere, no se deben utilizar más campos de los necesarios, así como la información requerida tiene que ser especificada de manera clara y concisa, no debe haber más que lo necesario en cada formulario y por último, las interfaces deben satisfacer los requerimientos del usuario, por lo cual no se debe excluir información solicitada por el usuario final y no se debe incluir información no solicitada por el mismo.

Dentro de la parte técnica, la capa de presentación contiene los objetos encargados de comunicar al usuario con el sistema mediante el intercambio de información, capturando y desplegando los datos necesarios para realizar alguna tarea.

En esta capa los datos se procesan de manera superficial por ejemplo, para determinar la validez de su formato o para darles algún orden específico. Esta capa se comunica únicamente con la capa de Reglas de Negocio o Control.

**2.1.4.4 Capa de Lógica de Negocio o Control:**

Es llamada capa de reglas de negocio porque en esta se definen todas las reglas que se deben cumplir para una correcta ejecución del programa. Es aquí donde se encuentra toda la lógica del programa, así como las estructuras de datos y objetos encargados para la manipulación de los datos existentes, así como el procesamiento de la información ingresada o solicitada por el usuario en la capa de presentación.

Representa el corazón de la aplicación ya que se comunica con todas las demás capas para poder llevar a cabo las tareas. Por ejemplo, mediante la capa de presentación obtiene la información ingresada por el usuario, y despliega los resultados.

Si la aplicación se comunica con otros sistemas que actúan en conjunto, lo hace mediante esta capa. También se comunica con la capa de datos para obtener información existente o ingresar nuevos datos. Recibe los datos que ingresó el usuario del sistema mediante la capa de presentación, luego los procesa y crea objetos según lo que se necesite hacer con estos datos; esta acción se denomina encapsulamiento.

Al encapsular los datos, el programa asegura mantener la consistencia de los mismos, así como obtener información precisa de las bases de datos e ingresar en las mismas, solamente la información necesaria, asegurando así no tener datos duplicados ni en las bases de datos, ni en los reportes solicitados por el usuario.

**2.1.4.5 Capa de Datos:**

Es la encargada de realizar transacciones con bases de datos y con otros sistemas para obtener o ingresar información al sistema. El manejo de los datos debe realizarse de forma tal que haya consistencia en los mismos, de tal forma los datos que se ingresan así como los que se extraen de las bases de datos, deben ser consistentes y precisos.

Es en esta capa donde se definen las consultas a realizar en la base de datos, tanto las consultas simples como las consultas complejas parla generación de reportes más específicos.

Esta capa envía la información directamente a la capa de reglas de negocio para que sea procesada e ingresada en objetos según se necesite, esta acción se denomina encapsulamiento.

**2.1.4.6 Ventajas y Desventajas**

La programación en capas no es una técnica rígida que debe implementarse solamente de una forma, sino que los desarrolladores de proyectos tienen múltiples maneras de implementarla según las tecnologías y tendencias que se utilicen.

La satisfacción de los requerimientos del usuario es la base para escoger el modelo de implementación a seguir. La tendencia a utilizar el modelo de programación en capas es grande cuando se trata principalmente de aplicaciones empresariales donde se deben manejar gran cantidad de subsistemas y módulos, así como generar reportes lo suficientemente complejos como para necesitar un orden estricto a la hora de desarrollar el proyecto.

Dentro del concepto de programación en capas, existen dos términos esenciales para el mejor entendimiento de los conceptos relativos a esta metodología, es aquí donde radica la importancia de la cohesión y el acoplamiento dentro de una aplicación generada mediante este método.

**Cohesión:** Este término es utilizado para describir el comportamiento que deben tener los módulos y objetos de un sistema o subsistema, comportamiento que describe la forma en que deben trabajar los objetos y módulos entre sí, con alta cohesión para que trabajando en conjunto los módulos y objetos puedan alcanzar un solo propósito de manera más eficaz y rápida.

Determina que las operaciones de un objeto deben trabajar en conjunto para alcanzar un propósito común. Es deseable que haya alta cohesión.

**Acoplamiento:** Se refiere al grado de dependencia que existe entre los módulos. Este grado de dependencia debe ser considerablemente bajo ya que el trabajo se divide en módulos para que cada uno tenga un funcionamiento específico y puede ser más factible la implementación por separado de cada uno.

En caso de haber alto acoplamiento entre módulos no se estaría alcanzando el principal objetivo de este modelo, el cual es dividir una tarea grande en varias pequeñas, ya que los módulos actuarían como uno solo al estar altamente acoplados entre sí y se perdería el objetivo primordial de dividir el proyecto.

**2.1.4.7 Ventajas**

Al implementar este modelo de programación, se asegura un trabajo de forma ordenada y separada, debido a que sigue el principio de “divide y vencerás”. Cada capa está dividida según su funcionalidad cuando se quiere modificar el sistema basta con cambiar un objeto o conjunto de objetos de una capa. Esto se llama modularidad.

**2.1.4.8 Desventajas**

Cuando se implementa un modelo de programación en capas, se debe llegar a un balance entre el número de capas y subcapas que componen el programa. Este debe ser el necesario y suficiente para realizar un trabajo específico con eficiencia y ser lo más modular posible. De lo contrario se tiene una serie de desventajas como: pérdida de eficiencia, realización de trabajo innecesario o redundante entre capas, gasto de espacio de la aplicación debido a la expansión de las capas, o bien una alta dependencia entre los objetos y capas que contradice el objetivo principal del modelo.

2.1.5 ESTANDARES DE PROGRAMACION

Un estándar de programación es:

* Una forma de "normalizar" la programación de forma tal que al trabajar en un proyecto, cualquier persona involucrada en el mismo tenga acceso y comprenda el código.

Y nos permite:

* Definir la escritura y organización del código fuente de un programa.
* Facilita a un programador la modificación de tu propio código fuente aunque no estés trabajando en el equipo.
* Definir la forma en que deben ser declaradas las variables, las clases, los comentarios
* Especificar que datos deben incluirse acerca del programador y de los cambios realizados al código fuente, etc.

Por ejemplo utilizar una determinada nomenclatura para la declaración de variables o de clases dependiendo de su tipo es de gran ayuda porque al leer el nombre de la variable ya sabes con qué tipo de datos estas trabajando y no tienes que buscar la declaración de la variable, todo esto dependiendo siempre del entorno y lenguaje de programación que utilices.

Hay que tener en cuenta que la entrega de una aplicación a tiempo (en fecha) pero con un pésimo desarrollo de código no es un buen trabajo.

Pongámoslo así: supongamos que nos vamos a una librería y deseamos comprar un recetario de cocina, pero estamos algo apurados así que le decimos al empleado del lugar que nos traiga lo más pronto posible (en 5 min.). Luego de un rato llega el empleado nos muestra un libro, vistoso y amigable a simple vista así que confiamos así que lo compramos y nos llevamos a casa.

Al llegar… ¡Oh sorpresa! Notamos que el recetario es confuso, no da detalles de los pasos, no explica para que sirve tal ingrediente, con este no podemos diferenciar que es lo primero y que es lo segundo, es decir un enredo total. Con ello uno que puede pensar:

* Ya no iré nunca a esa librería
* No confiare de lo que digan los expertos

Ahora asociemos:

* Librería = empresa
* Recetario = aplicación
* Ingredientes = algún modulo o componente

Por ello es muy, pero muy importante el tener un ORDEN, una FORMA, un ESTILO ya definido, ya que a largo o mediano plazo, tendrás unos inconvenientes tan grandes que no se te será suficiente una amanecida para buscar el problema y/o resolverlo.

Si no atacamos esto, como empresa, quedaría mal ante un cliente y así se va su credibilidad.

**Parte 3**

# Parte 2 CAPTURA DE REQUISITO

## 3.1 INTRODUCCION

El primer paso para cumplir el objetivo general del desarrollo del Sistema de Información para la institución ITPIAI, es realizar la captura de requerimientos, según la metodología utilizada (Proceso Unificado de Desarrollo).

**¿Qué son Requerimientos?**   
Normalmente, un tema de la Ingeniería de Software tiene diferentes significados. De las muchas definiciones que existen para requerimiento, ha continuación se presenta la definición que aparece en el glosario de la IEEE.

(1) Una condición o necesidad de un usuario para resolver un problema o alcanzar un objetivo.

(2) Una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar, especificación u otro documento formal.

(3) Una representación documentada de una condición o capacidad como en (1) o (2).

Los requerimientos puedes dividirse en requerimientos funcionales y requerimientos no funcionales. Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas para producir salidas.  
Los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc.

## 3.2 LISTA DE REQUERIMIENTOS

|  |  |  |  |
| --- | --- | --- | --- |
| **N°** | **NOMBRE** | **DESCRIPCIÓN** | **PRIORIDAD** |
| **1** | Inicio de sesión | Iniciar sesión, con un límite de 3 intentos fallidos consecutivos. | Bajo |
| **2** | Gestionar Registro de Estudiante | Manipular los datos del estudiante para el sistema | Importante |
| **3** | Gestionar Teléfonos | Se podrán registrar distintos teléfonos para una misma persona | bajo |
| **4** | Gestionar Kardex | Administrar al usuario dependiendo de su carrera | Importante |
| **5** | Gestionar Registro de Docentes | Manipular los datos del docente en el sistema | Importante |
| **6** | Gestionar Registro de Secretaria | Administrar la información de la secretaria | medio |
| **7** | Gestionar Especialidad | Especialidades o conocimientos de los docentes | medio |
| **8** | Gestionar carrera | Registrar, modificar y eliminar las carreras del instituto | Importante |
| **9** | Gestionar Registro de Administrador | Manipular los datos que tiene el administrador en el sistema | importante |
| **10** | Registrar Pago Matricula | Para tener el control de las inscripciones y aportes a la institución. | Importante |
| **11** | Consultar Inscripción | Obtener los datos de inscripción generados | bajo |
| **12** | Registrar Notas y Asistencia | El docente registrara la puntuación y asistencia de los estudiantes en su materia | Importante |
| **13** | Gestionar Gestión | Registro de gestión para habilitar la inscripción | Importante |
| **14** | Administrar Aulas | administrar las aulas para y su ubicación | Importante |
| **15** | Administrar Horario | Manipulación de aula, horario y día por periodo | Importante |
| **16** | Gestionar Materia | Registro, modificar y eliminar datos de materias. | importante |
| **17** | Gestionar Currícula | Administrador de las materias por niveles para las gestiones | importante |

## 3.3 MODELO DE NEGOCIO

En el modelo de negocio se describen los procesos de negocio de la empresa, a través de un diagrama de actividades para comprender el contexto del sistema.

3.3.1 DIAGRAMA DE ACTIVIDAD

Realizar Inscripción



Planificar Curso

****

Crear currícula o malla académica de la materia

****

## 3.4. MODELO DE CASO DE USO

3.4.1. ACTORES Y LISTA DE CASOS DE USO:

Se identifican los siguientes actores.

**1. Estudiante.** Es la persona, institución a la cual se brindará el registro correspondiente.

**2. Docente.** Es la persona, institución a la cual se brindará el registro correspondiente y se facilitará el acceso al sistema para el llenado de notas a sus estudiantes.

**3. Secretaria.** Esta encargada de realizar todos los registros y verificación de datos de los estudiantes.

**4. Administrador.** Este encargado de asignar el perfil de usuario a la secretaria y docente, así también puede realizar devoluciones de dinero y verificar datos de los estudiantes.



3.4.2. CASO DE USO: (Descripción De Caso Uso)

Caso de usos identificados en la etapa de requisitos se describe a continuación:

**CU1 INICIO DE SESION**



|  |  |
| --- | --- |
| Caso de uso | INICIO DE SESION |
| Propósito | Permitir acceso a los usuarios |
| Descripción | Poder permitir el acceso y colocarlo en el formulario que le corresponde para su tipo de usuario |
| Actores | usuario |
| Actor iniciador | usuario |
| Precondición | Estar registrado en el sistema |
| Proceso | 1. Acceder    1. Ingresar usuario y contraseña    2. Iniciar sesión    3. **Si existe usuario y está activo**       1. **Si contraseña correcta**, acceder al formulario correspondiente al tipo de usuario       2. **Si contraseña incorrecta**, aumentar contador de fallos de usuario    4. **Si no**       1. Denegar acceso 2. Salir    1. Cerrar inicio de sesión |
| Pos condición | Registro del inicio de sesión del usuario en la bitácora |

**CU2 GESTIONAR ESTUDIANTE**

****

|  |  |
| --- | --- |
| Caso de uso | Gestionar Registro de Estudiante |
| Propósito | Manipular el registro de estudiantes |
| Descripción | Se registrara, modificara, eliminara los datos del estudiante, además de la gestión de kardex y teléfonos de contacto |
| Actores | Administrador, secretaria |
| Actor iniciador | usuario |
| Precondición | Ser administrador o secretaria |
| Proceso | 1. **Registrar**    1. Ingresar los datos: numero Carnet, nombre de usuario, contraseña, nombre completo, apellidos, sexo, dirección, email, nacionalidad, fecha nacimiento    2. **Extends CU3**    3. **Extends CU4**    4. Registrar al usuario, los datos personales y al estudiante       1. Si registro de datos personales correcto, registrar teléfonos y asociarlos a la persona.    5. Cargar el registro al formulario 2. **Modificar**    1. Ingresar ci o código itpiai    2. Buscar estudiante    3. **Si existe**       1. Modificar los datos del estudiante       2. **Extends CU3**       3. **Extends CU4**       4. Registrar los cambios 3. **Eliminar**    1. Ingresar ci o código itpiai    2. Buscar estudiante    3. **Si existe**       1. Eliminar registro 4. **Desbloquear** |
| Pos condición | Registro del inicio de sesión del usuario en la bitácora |

**CU3 GESTIONAR TELEFONOS**

****

|  |  |
| --- | --- |
| Caso de uso | Gestionar Teléfonos |
| Propósito | Administrar teléfonos por persona |
| Descripción | Se podrá agregar o borrar teléfonos a personas registradas |
| Actores | Administrador, secretaria |
| Actor iniciador | usuario |
| Precondición | Ser administrador o secretaria |
| Proceso | 1. **Cargar teléfonos de la persona actual** 2. **Agregar**    1. Introducir numero    2. Ligar a la persona que corresponde el teléfono    3. Registrar el teléfono 3. **Borrar**    1. Seleccionar teléfono    2. Borrar registro del teléfono |
| Pos condición | Registro agregado o eliminado en el sistema |

**CU4 GESTIONAR KARDEX**

****

|  |  |
| --- | --- |
| Caso de uso | Gestionar Kardex |
| Propósito | Manipular el kardex del estudiante |
| Descripción | Se guardaran los datos del kardex y no se podrá eliminar |
| Actores | Administrador, secretaria |
| Actor iniciador | usuario |
| Precondición | tener registro de usuario |
| Proceso | 1. Cargar datos del kardex 2. Cargar lista de carreras activas 3. Modificar datos del kardex 4. Registrar cambios |
| Pos condición | Actualización de los datos del kardex |

**CU5 GESTIONAR DOCENTE**

****

|  |  |
| --- | --- |
| Caso de uso | Gestionar Registro de Docente |
| Propósito | Manipular el registro de los docentes |
| Descripción | Se registrara, modificara, eliminara los datos del docente, además de la gestión de especialidades y teléfonos de contacto |
| Actores | Administrador, secretaria |
| Actor iniciador | Administrador, secretaria |
| Precondición | Ser administrador o secretaria |
| Proceso | 1. **Registrar**    1. Ingresar datos: Carnet, nombre de usuario, contraseña, nombre completo, apellidos, sexo, dirección, email, nacionalidad, fecha nacimiento, fecha de ingreso.    2. **Extends CU3**    3. **Extends CU7**    4. Registrar al usuario, datos personales y al docente       1. Si registro de datos personales correcto, registrar teléfonos y especialidades. 2. **Modificar**    1. Ingresar ci    2. Buscar docente    3. **Si existe**       1. Modificar los datos del docente       2. **Extends CU3**       3. **Extends CU7**       4. Guardar los cambios 3. **Eliminar**    1. Ingresar ci    2. Buscar docente    3. **Si existe**       1. Eliminar registro 4. **Desbloquear**    1. Resetear a 0 los intentos fallidos |
| Pos condición | registros agregados o modificados de docentes en el sistema |

**CU6 GESTIONAR SECRETARIA**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Registro de Secretaria |
| Propósito | Manipular el registro de las secretarias |
| Descripción | Se registrara, modificara, eliminara los datos de las secretarias, además de teléfonos de contacto |
| Actores | Administrador, secretaria |
| Actor iniciador | Administrador, secretaria |
| Precondición |  |
| Proceso | 1. **Registrar**    1. Ingresar datos: Carnet, nombre de usuario, contraseña, nombre completo, apellidos, sexo, dirección, email, nacionalidad, fecha nacimiento, fecha de ingreso.    2. **Extends CU3**    3. Registrar al usuario, datos personales y secretaria       1. Si registro de datos personales correcto, registrar teléfonos. 2. **Modificar**    1. Ingresar ci    2. Buscar secretaria    3. **Si existe**       1. Modificar los datos del secretaria       2. **Extends CU3**       3. Guardar los cambios 3. **Eliminar**    1. Ingresar ci    2. Buscar secretaria    3. **Si existe**       1. Eliminar registro 4. **Desbloquear**    1. Restaurar a 0 intentos fallidos |
| Pos condición | registro agregado o modificado de la secretaria en el sistema |

**CU7 GESTIONAR ESPECIALIDAD**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Especialidad |
| Propósito | Administrar especialidades del docente |
| Descripción | Se podrá agregar o borrar especialidades y añadirlos a los docentes registradas |
| Actores | Administrador, secretaria |
| Actor iniciador | Usuario |
| Precondición | Ser administrador o secretaria |
| Proceso | 1. Cargar lista de especialidades que no están registrados al docente 2. Cargar lista de especialidades que están registrados al docente 3. **Agregar**    1. Introducir nombre de especialidad    2. Registrar la especialidad    3. Actualizar lista de especialidades 4. **Borrar**    1. Seleccionar especialidad    2. Desactivar la especialidad para no utilizarse    3. Actualizar lista de especialidades 5. **Incluir**    1. Seleccionar especialidad del docente    2. Incluir registro de especialidad al docente    3. Actualizar lista de especialidades del docente 6. **Quitar**    1. Seleccionar especialidad del docente    2. Quitar registro de especialidad al docente    3. Actualizar lista de especialidad del docente |
| Pos condición | Registro agregado o eliminado en el sistema |

**CU8 GESTIONAR CARRERA**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Carrera |
| Propósito | Administrar Registro o datos de las carreras |
| Descripción | Se registrara, modificara, eliminara los datos de las carreras. |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Precondición |  |
| Proceso | 1. Cargar lista de modalidades 2. **Registrar**    1. Ingresar nombre de la carrera    2. Seleccionar modalidad    3. Seleccionar duración en años de la carrera    4. Registrar la carrera en el sistema    5. Actualizar los campos con el nuevo registro 3. **Modificar**    1. Ingresar nombre o id de la carrera    2. Buscar carrera    3. **Si existe carrera**       1. Cargar datos en los campos       2. Modificar los datos de la carrera       3. Actualizar Registro en el sistema 4. **Eliminar**    1. Ingresar id o nombre de la carrera    2. Buscar carrera    3. **Si existe carrera**       1. Cargar datos en el los campos       2. Preguntar si desea eliminar la carrera       3. Si acepta          1. Eliminar registro |
| Pos condición | registros agregados o modificados de docentes en el sistema |

**CU9 GESTIONAR REGISTRO DE ADMINISTRADOR**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Registro de Administrador |
| Propósito | Manipular el registro de los administradores |
| Descripción | Se registrara, modificara, eliminara los datos de los administradores, además de teléfonos de contacto |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Precondición |  |
| Proceso | 1. **Registrar**    1. Limpiar datos y campos    2. Ingresar datos: Carnet, nombre de usuario, contraseña, nombre completo, apellidos, sexo, dirección, email, nacionalidad, fecha nacimiento, cargo.    3. **Extends CU3**    4. Registrar al usuario, datos personales y administrador       1. Si registro de datos personales correcto, registrar teléfonos y especialidades.    5. Actualizar campos 2. **Modificar**    1. Ingresar ci    2. Buscar administrador    3. **Si existe**       1. Modificar los datos del administrador       2. **Extends CU3**       3. Guardar los cambios 3. **Eliminar**    1. Ingresar ci    2. Buscar administrador    3. **Si existe**       1. Eliminar registro 4. **Desbloquear**    1. Restaurar a 0 intentos fallidos |
| Pos condición | registro agregado o modificado de la secretaria en el sistema |

**CU10 REGISTRAR PAGO DE MATRICULA**



|  |  |
| --- | --- |
| Caso de uso | Registrar Pago De Matricula |
| Propósito | Registrar Pago de la Matricula e inscripción del alumno |
| Descripción | Se registrara el pago de la materia, luego se inscribirá al alumno en la gestión correspondiente |
| Actores | Administrador, Secretaria |
| Actor iniciador | estudiante |
| Precondición | Registro de la gestión, registro de la planificación del curso. |
| Proceso | 1. Buscar estudiante en el sistema Seleccionar modo de pago 2. Ingresar monto a pagar 3. Ingresar monto que se está pagando 4. Registrar pago 5. Generar inscripción del estudiante 6. **Include cu11**    1. Imprimir inscripción en del estudiante |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**CU11 CONSULTAR INSCRIPCION**



|  |  |
| --- | --- |
| Caso de uso | Consultar Inscripción |
| Propósito | Consultar Inscripción generada para el estudiante |
| Descripción | Obtener los datos de la inscripción actual. |
| Actores | Administrador, Secretaria |
| Actor iniciador | Administrador, Secretaria |
| Precondición | Registro existente de la inscripción |
| Proceso | 1. Solicitar registro de inscripción del estudiante 2. Enviar id del estudiante 3. Mostrar en pantalla el registro de inscripción |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**CU12 REGISTRAR NOTAS Y ASISTENCIA**



|  |  |
| --- | --- |
| Caso de uso | Registrar Notas y Asistencia |
| Propósito | Registrar Notas y Asistencias |
| Descripción | El docente registrara las notas y asistencia del alumno en su materia |
| Actores | Docente |
| Actor iniciador | Docente |
| Precondición | Estudiante Registrado en la materia del Docente |
| Proceso | 1. Seleccionar materia 2. Mostrar todos los estudiantes 3. Seleccionar estudiante 4. Ingresar nota y asistencia 5. Registrar acción |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**C13 GESTIONAR GESTION**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Gestion |
| Propósito | Registrar Notas y Asistencias |
| Descripción | El docente registrara las notas y asistencia del alumno en su materia |
| Actores | Docente |
| Actor iniciador | Docente |
| Precondición | Estudiante Registrado en la materia del Docente |
| Proceso | 1. **Registrar o modificar**    1. seleccionar modalidad    2. ingresar número de la gestión, año, fecha inicio y final.    3. Registrar si no existe, si existe modificar 2. **Eliminar**    1. Ingresar datos y guardar para obtener registro    2. Cambiar estado a desactivado |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**CU14 ADMINISTRAR AULAS**



|  |  |
| --- | --- |
| Caso de uso | Administrar Aulas |
| Propósito | Llevar el registro de las aulas que se podrán usar. |
| Descripción | Registrar, borrar las aulas que estarán disponibles para clases |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Proceso | 1. Mostrar aulas registradas 2. **Registrar**     1. Ingresar los datos del aula    2. Verificar los datos    3. Registrarlo en la base de datos 3. **Eliminar**    1. seleccionar aula    2. cambiar su registro a desactivado |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**CU15 ADMINISTRAR HORARIO**



|  |  |
| --- | --- |
| Caso de uso | Administrar Horario |
| Propósito | Planificar el horario de los cursos |
| Descripción | Se tendrá en cuanta las horas, el día y en que aula se pasaran las diferentes materias. |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Proceso | 1. seleccionar carrera en la lista 2. seleccionar materia en la lista 3. seleccionar aula, día, y horario 4. guardar datos seleccionados para el aula. |
| Pos condición | Nuevo registro de pago, nuevo registro de inscripción del estudiante. |

**CU16 GESTIONAR MATERIA**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Materia |
| Propósito | Registro, modificación y cancelación de la materia |
| Descripción | Administrar el registro de las materias activas |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Precondición |  |
| Proceso | 1. **Registrar**    1. Ingresar sigla, nombre y carga horaria    2. Si no existe la materia registrarla    3. Obtener registro    4. Actualizar campo 2. **Modificar**    1. Ingresar sigla o nombre de la carrera    2. Buscar materia    3. **Si existe la materia**       1. Modificar los datos en los campos       2. Actualizar datos en la base de datos       3. Obtener registro       4. Actualizar campos 3. **Eliminar**    1. Ingresar sigla o nombre    2. Buscar materia    3. **Si existe materia**       1. Cargar campos       2. Mostrar mensaje para eliminar el registro       3. Si acepta          1. Eliminar registro |
| Pos condición | Actualización en la tabla de materias de la base de datos. |

**CU17GESTIONAR CURRICULA**



|  |  |
| --- | --- |
| Caso de uso | Gestionar Curricula |
| Propósito | Administrar materias que tendrá la carrera |
| Descripción | Añadir y quitar materias en a una carrera |
| Actores | Administrador |
| Actor iniciador | Administrador |
| Precondición | Materias y carreras registradas |
| Proceso | 1. Mostrar carreras y materias 2. Seleccionar carrera 3. **Añadir**    1. Seleccionar materia    2. Ingresar nivel    3. Añadir materia a la carrera    4. Actualizar lista de materias añadidas 4. **Quitar**    1. Seleccionar materia añadida    2. Quitar materia de la carrera    3. Actualizar lista de materias añadidas |
| Pos condición | Actualización en la tabla de materias de la base de datos. |

## 3.5 DIAGRAMA GENERAL DE CASOS DE USO



**Parte 4**

# PARTE 4 ANALISIS

## 4.1 INTRODUCCION

El objetivo de este flujo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema entero.

El modelo de análisis nos ayuda a refinar los requisitos según las líneas que ya hemos mencionado antes y nos permite razonar sobre los aspectos internos del sistema, incluidos sus recursos compartidos internos. Además, el modelo de análisis nos ofrece un mayor poder expresivo y una mayor formalización. El modelo de análisis proporciona una estructura centrad en el mantenimiento, en aspectos tales como la flexibilidad ante los cambios y la reutilización

**¿Por qué realizamos el análisis?**

Porque es probable que aun quede aspectos sin resolver relativos a los requisitos del sistema, entonces el propósito del análisis es resolverlos analizando los requisitos con mayor profundidad, pero con la gran diferencia de que puede utilizarse el lenguaje de los desarrolladores para describir los resultados

Llevando a cabo el análisis conseguimos una separación de intereses que preparan y simplifica las subsiguientes actividades de diseño e implementación, delimitando los temas que deben resolverse y las decisiones que deben tomarse en esas actividades

## 4.2 DIAGRAMA DE PAQUETES



## 4.3 DIAGRAMA DE CLASES (interfaz, control, entidad)

## 4.3 DIAGRAMA DE CLASES (interfaz, control, entidad)

**CU1 INICIAR SESION**



**CU2 GESTIONAR REGISTRO ESTUDIANTE**



**CU3 GESTIONAR TELEFONO**



**CU4 GESTIONAR KARDEX**



**CU5 GESTIONAR REGISTRO DE DOCENTE**



**CU6 GESTIONAR REGISTRO DE SECRETARIA**



**CU7 GESTIONAR ESPECIALIDAD**

****

**CU8 GESTIONAR CARRERA**



**CU9 GESTIONAR REGISTRO DEL ADMINISTRADOR**



**CU10 REGISTRAR PAGO MATRICULA**



**CU11 CONSULTA INSCRIPCION**



**CU12 REGISTRAR NOTA Y ASISTENCIA**



**CU13 ADMINISTRAR GESTION**



**CU14 ADMINISTRAR AULAS**



**CU15 GESTIONAR HORARIO**



**CU16 GESTIONAR MATERIA**



**CU17 GESTIONAR CURRICULA**



**Parte 5**

# Parte 5 DISEÑO

## 5.1 INTRODUCCION

En el diseño modelamos el sistema y encontramos su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales y otras restricciones.

El modelo de análisis proporciona una comprensión detallada de los requisitos y lo que es más importante, impone una estructura del sistema que debemos esforzarnos por conservar lo más fielmente posible cuando demos forma al sistema

Los propósitos del diseño son:

* Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes y demás.
* Crear una ente utilizables, sistemas operativos, tecnologías de distribución y concurrencia rada apropiada y un punto de partida para actividades de implementación subsiguiente capturando los requisitos o subsistemas individuales, interfaces y clases
* Ser capaces de descompones los trabajo de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo

## 5.2 ARQUITECTURA DEL SISTEMA



## 5.3 DISEÑO DE CASOS DE USO

**GESTIONAR REGISTRO DE ESTUDIANTE**



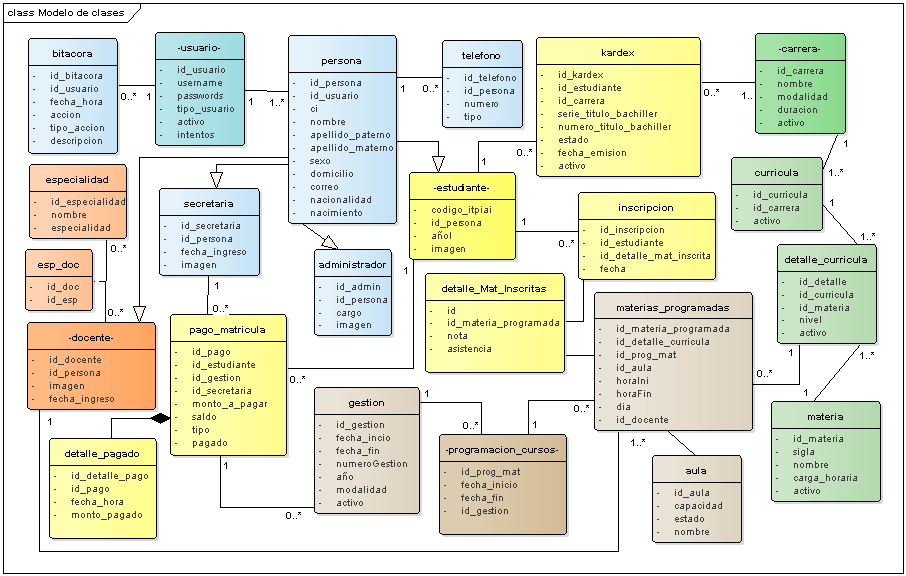
**REALIZAR PAGO DE MATRICULA**



**GESTIONAR CURRICULA**



## 5.4 DISEÑO DE CLASE



## 5.5 DISEÑO LOGICO DE LA BASE DE DATOS

**Usuario:**

PK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id\_usuario | username | passwords | tipo\_usuario | activo | intentos |

**Nivel:**

PK

|  |  |  |
| --- | --- | --- |
| id\_nivel | nombre | modalidad |

**Persona:**

PK FK

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id\_persona | | id\_usuario | ci | nombre | | apellido\_p | apellido\_m | sexo | domicilio |
| Correo | nacionalidad | | nacimiento | |

**Teléfono:**

PK FK

|  |  |  |  |
| --- | --- | --- | --- |
| id\_telefono | id\_persona | numero | tipo |

**Administrador:**

PK FK

|  |  |  |  |
| --- | --- | --- | --- |
| id\_admin | id\_persona | cargo | imagen |

**Docente:**

PK FK

|  |  |  |  |
| --- | --- | --- | --- |
| id\_docente | id\_persona | fecha\_ingreso | imagen |

**Secretaria:**

PK FK

|  |  |  |  |
| --- | --- | --- | --- |
| id\_secretaria | id\_persona | fecha\_ingreso | imagen |

**Estudiante:**

PK FK

|  |  |  |  |
| --- | --- | --- | --- |
| cod\_itpiai | id\_persona | añoIng | imagen |

**Carrera:**

PK

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id\_carrera | nombre | modalidad | duracion | activo |

**Kardex:**

PK FK FK

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| id\_Kardex | cod\_itpiai | id\_carrera | serie\_tit\_b | numero\_tit\_b | fecha\_tit\_b | estado | activo |

**Especialidad:**

PK

|  |  |  |
| --- | --- | --- |
| id\_especialidad | nombre | estado |

**esp\_doc:**

FK FK

|  |  |
| --- | --- |
| id\_docente | id\_especialidad |

**Materia:**

PK

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id\_materia | sigla | nombre | carga\_horaria | activo |

**Gestión:**

PK

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| id\_gestion | numero | año | fechaIni | fechaFin | modalidad | activo |

**Curricula:**

PK FK

|  |  |  |
| --- | --- | --- |
| id\_curricula | id\_carrera | activo |

**detalleCurricula:**

PK

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id\_det | id\_curricula | id\_materia | nivel | activo |

## 5.6 DISEÑO FISICO DE LA BASE DE DATOS

**Usuario:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_usuario | Numérico |  | No |
| username | Alfanumérico | 100 | No |
| passwords | Alfanumérico | 100 | No |
| tipo\_usuario | Alfanumérico | 100 | No |
| Activo | Bit |  | No |
| intentos | Numérico |  | No |

**Nivel:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_nivel | Numérico |  | No |
| Nombre | Alfanumérico | 50 | No |
| modalidad | Alfanumérico | 50 | No |

**Teléfono:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_telefono | Numérico |  | No |
| id\_persona | Numérico |  | No |
| numero | Numérico |  | No |
| tipo | Alfanumérico | 50 | Si |

**Administrador:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_admin | Numérico |  | No |
| id\_persona | Numérico |  | No |
| cargo | Alfanumérico | 100 | No |
| imagen | Alfanumérico | 300 | Si |

**Docente:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_docente | Numérico |  | No |
| id\_persona | Numérico |  | No |
| fecha\_ingreso | Date |  | Si |
| Imagen | Alfanumérico | 300 | Si |

**Secretaria:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_secretaria | Numérico |  | No |
| id\_persona | Numérico |  | No |
| fecha\_ingreso | Date |  | Si |
| imagen | Alfanumérico | 300 | Si |

**Estudiante:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| cod\_itpiai | Numérico |  | No |
| id\_persona | Numérico |  | No |
| añoIng | Date |  | No |
| Imagen | Alfanumérico | 300 | Si |

**Carrera:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_carrera | Numérico |  | No |
| nombre | Alfanumérico | 100 | Si |
| modalidad | Alfanumérico | 100 | Si |
| duracion | Numérico |  | Si |
| activo | Bit |  | Si |

**Kardex:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_Kardex | Numérico |  | No |
| cod\_itpiai | Numérico |  | No |
| id\_carrera | Numérico |  | No |
| serie\_tit\_b | Alfanumérico | 100 | Si |
| numero\_tit\_b | Alfanumérico | 100 | Si |
| fecha\_tit\_b | Date |  | Si |
| estado | Alfanumérico | 100 | Si |
| activo | Bit |  | Si |

**Especialidad:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_especialidad | Numérico |  | No |
| nombre | Alfanumérico | 50 | Si |
| estado | bit |  | Si |

**esp\_doc:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_docente | Numérico |  | No |
| id\_especialidad | Numérico |  | No |

**Materia:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_materia | Numérico |  | No |
| sigla | Alfanumérico | 100 | No |
| nombre | Alfanumérico | 100 | No |
| carga\_horaria | Numérico |  | Si |
| activo | Bit |  | Si |

**Gestión:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_gestion | Numérico |  | No |
| numero | Numérico |  | No |
| año | Numérico |  | No |
| fechaIni | Date |  | Si |
| fechaFin | Date |  | Si |
| modalidad | Alfanumérico | 100 | No |
| activo | Bit |  | No |

**Curricula:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_curricula | Numérico |  | No |
| id\_carrera | Numérico |  | No |
| activo | Numérico |  | No |

**detalleCurricula:**

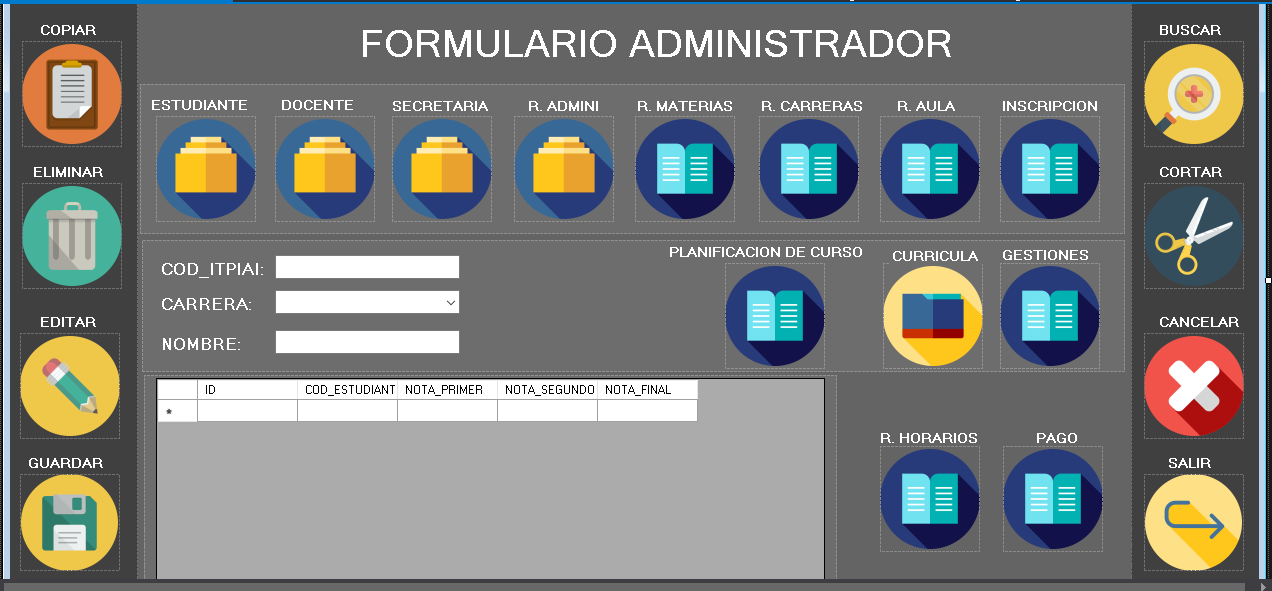
|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos** | **Tipo Datos** | **Amplitud** | **Permite (Nulo)** |
| id\_det | Numérico |  | No |
| id\_curricula | Numérico |  | No |
| id\_materia | Numérico |  | No |
| nivel | Numérico |  | No |
| activo | Bit |  | No |

## 5.7 INTERFAZ

**LOGIN**

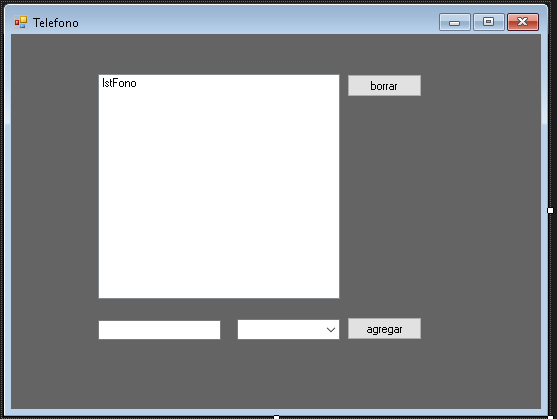


**FORMULARIO ADMINISTRADOR**



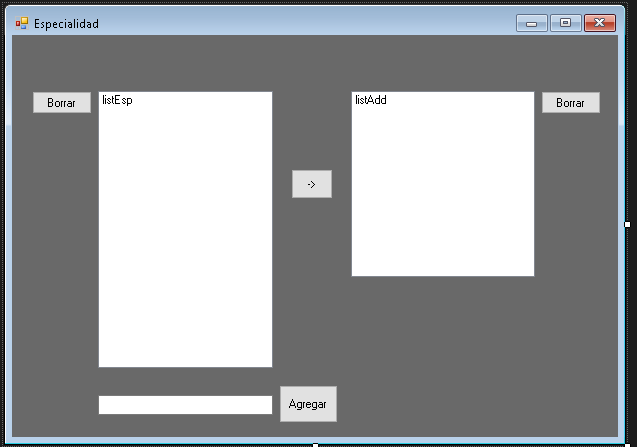


**TELEFONO**





**ESPECIALIDADES DEL DOCENTE**



# BIBLIOGRAFIA

Ivar Jacobson, Grady Booch, James Rumbaugh N. (2001). **El proceso Unificado de desarrollo de software.**.

*Unified Modeling Language Version 2.0:****UML 2.0 Infraestructure Specification and   UML 2.0 Superestructure Specification****",*2005.