INSTITUTO TECNOLOGICO
"CORDILLERA"

CARRERA DE ANÁLISIS EN SISTEMAS

"MEJORAMIENTO DE LOS PROCESOS DE GESTIÓN ACADÉMICA MEDIANTE UNA APLICACIÓN INFORMÁTICA EN EL CENTRO EDUCATIVO JASON MILLER DE LA CIUDAD DE QUITO"

Proyecto de investigación previo a la obtención del título de Tecnólogo en Análisis de Sistemas.

Autora: Lady Senaida Moreno Narváez

Tutor: Ing.Johnny Coronel

Quito, Abril 2015

Aprobación del tutor

En mi calidad de tutor del trabajo sobre el tema: **"MEJORAMIENTO DE LOS PROCESOS DE GESTIÓN ACADÉMICA MEDIANTE UNA APLICACIÓN INFORMÁTICA EN EL CENTRO EDUCATIVO JASON MILLER DE LA CIUDAD DE QUITO"** presentado por el ciudadano: **Lady Senaida Moreno Narváez**, estudiante de la Escuela de Análisis de Sistemas, considero que dicho informe reúne los requisitos y méritos suficientes para ser sometido a la evaluación por parte del Tribunal de Grado, que el Honorable Consejo de Escuela designe, para su correspondiente estudio y calificación.

Ing. Johnny Coronel	Ing. Patricia Garzón
TUTOR	LECTOR

Declaratoria

Declaro que la investigación es absolutamente original, auténtica, personal, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes, las ideas doctrinas, resultados y conclusiones a los que he llegado son de mi absoluta y entera responsabilidad.

_____

Lady Senaida Moreno Narváez

CC: 040155706-1

Contrato de cesión de derechos a la institución

Comparecen a la celebración del presente contrato de cesión y transferencia de derechos de propiedad intelectual, por una parte, el estudiante LADY SENAIDA MORENO NARVÀEZ**,** por sus propios y personales derechos, a quien en lo posterior se le denominará el "CEDENTE"; y, por otra parte, el INSTITUTO SUPERIOR TECNOLÓGICO CORDILLERA, representado por su Rector el Ingeniero Ernesto Flores Córdova, a quien en lo posterior se lo denominará el "CESIONARIO". Los comparecientes son mayores de edad, domiciliados en esta ciudad de Quito Distrito Metropolitano, hábiles y capaces para contraer derechos y obligaciones, quienes acuerdan al tenor de las siguientes cláusulas:

PRIMERA: ANTECEDENTE.- a) El Cedente dentro del pensum de estudio en la carrera de Análisis y Sistemas que imparte el Instituto Superior Tecnológico Cordillera, y con el objeto de obtener el título de Tecnólogo Analista de Sistemas, el estudiante participa en el proyecto de grado denominado MEJORAMIENTO DE LOS PROCESOS DE GESTIÓN ACADÉMICA MEDIANTE UNA APLICACIÓN INFORMÁTICA EN EL CENTRO EDUCATIVO JASON MILLER DE LA CIUDAD DE QUITO" El cual incluye la investigación para la implementación de un sistema académico en el CENTRO EDUCATIVO JASON MILLER, para lo cual ha implementado los conocimientos adquiridos en su calidad de alumno. **b)** Por iniciativa y responsabilidad del Instituto Superior Tecnológico Cordillera se desarrolla la investigación para el estudio de factibilidad, motivo por el cual se regula de forma clara la cesión de los derechos de autor que genera la obra literaria y que es producto del proyecto de grado, el mismo que culminado es de plena aplicación técnica, administrativa y de reproducción.

SEGUNDA: CESIÓN Y TRANSFERENCIA.- Con el antecedente indicado, el Cedente libre y voluntariamente cede y transfiere de manera perpetua y gratuita todos los derechos patrimoniales del estudio de factibilidad descrito en la cláusula anterior a favor del Cesionario, sin reservarse para sí ningún privilegio especial (código fuente, código objeto, diagramas de flujo, planos, manuales de uso). El Cesionario podrá explotar la investigación para el estudio de factibilidad por cualquier medio o procedimiento tal cual lo establece el Artículo 20 de la Ley de

Propiedad Intelectual, esto es, realizar, autorizar o prohibir, entre otros: a) La reproducción de la investigación para el estudio de factibilidad por cualquier forma o procedimiento; b) La comunicación pública de la investigación para el estudio de factibilidad; c) La distribución pública de ejemplares o copias, la comercialización, arrendamiento o alquiler de la investigación para el estudio de factibilidad; d) Cualquier transformación o modificación de la investigación para el estudio de factibilidad; e) La protección y registro en el IEPI la investigación para el estudio de factibilidad a nombre del Cesionario; f) Ejercer la protección jurídica la investigación para el estudio de factibilidad; g) Los demás derechos establecidos en la Ley de Propiedad Intelectual y otros cuerpos legales que normen sobre la cesión de derechos de autor y derechos patrimoniales.

TERCERA: OBLIGACIÓN DEL CEDENTE.- El cedente no podrá transferir a ningún tercero los derechos que conforman la estructura, secuencia y organización de la investigación para el estudio de factibilidad que es objeto del presente contrato, como tampoco emplearlo o utilizarlo a título personal, ya que siempre se deberá guardar la exclusividad de la investigación para el estudio de factibilidad a favor del Cesionario.

CUARTA: CUANTIA.- La cesión objeto del presente contrato, se realiza a título gratuito y por ende el Cesionario ni sus administradores deben cancelar valor alguno o regalías por este contrato y por los derechos que se derivan del mismo.

QUINTA: PLAZO.- La vigencia del presente contrato es indefinida.

SEXTA: DOMICILIO, JURISDICCIÓN Y COMPETENCIA.- Las partes fijan como su domicilio la ciudad de Quito. Toda controversia o diferencia derivada de éste, será resuelta directamente entre las partes y, si esto no fuere factible, se solicitará la asistencia de un Mediador del Centro de Arbitraje y Mediación de la Cámara de Comercio de Quito. En el evento que el conflicto no fuere resuelto mediante este procedimiento, en el plazo de diez días calendario desde su inicio, pudiendo prorrogarse por mutuo acuerdo este plazo, las partes someterán sus controversias a la resolución de un árbitro, que se sujetará a lo dispuesto en la Ley de Arbitraje y Mediación, al Reglamento del Centro de Arbitraje y Mediación de la Cámara de comercio de Quito, y a las siguientes normas: a) El árbitro será

seleccionado conforme a lo establecido en la Ley de Arbitraje y Mediación; b) Las partes renuncian a la jurisdicción ordinaria, se obligan a acatar el laudo arbitral y se comprometen a no interponer ningún tipo de recurso en contra del laudo arbitral; c) Para la ejecución de medidas cautelares, el árbitro está facultado para solicitar el auxilio de los funcionarios públicos, judiciales, policiales y administrativos, sin que sea necesario recurrir a juez ordinario alguno; d) El procedimiento será confidencial y en derecho; e) El lugar de arbitraje serán las instalaciones del centro de arbitraje y mediación de la Cámara de Comercio de Quito; f) El idioma del arbitraje será el español; y, g) La reconvención, caso de haberla, seguirá los mismos procedimientos antes indicados para el juicio principal.

SÉPTIMA: ACEPTACIÓN.- Las partes contratantes aceptan el contenido del presente contrato, por ser hecho en seguridad de sus respectivos intereses.

En aceptación firman a los 27 días del mes de Marzo del dos mil quince.

f)_____	f)_____
C.C. Nº040155706-1	Instituto Superior Tecnológico Cordillera
CEDENTE	**CESIONARIO**

Agradecimiento

A mi madre que con su ayuda incondicional, paciencia, amor y además su apoyo a cada minuto es mi sendero a seguir.

Mis más sinceros agradecimientos al Centro Educativo Jason Miller por abrirme sus puertas y especialmente al Ing. Johnny Coronel que sin su apoyo no se hubiera podido culminar el presente trabajo.

También quiero agradecer a la Ing. Patricia Garzón y Al Ing. Hugo Heredia por permitirme culminar un sueño más.

A Dios por permitirme un día más de vida y darme fuerza y fortaleza para seguir adelante.

A todos quienes de una u otra manera colaboraron desinteresadamente para la realización del proyecto mis más sinceros agradecimientos.

Dedicatoria

A mi padre que dios lo tenga en su gloria y especialmente a mi madre, que con su esfuerzo, sacrificio y amor incondicional deposito en mí su confianza y así ha hecho posible este sueño.

A mi futuro esposo, quien con su amor incomparable, ha cambiado mi vida, y siempre ha estado conmigo a pesar de las adversidades.

A mis hermanos gracias por el apoyo y amor que me han brindado a cada momento de desarrollo de mi tesis y en mi vida, de verdad mil gracias.

Índice general

Contenido

Índice de tablas

Índice de figuras

Índice de manuales

Resumen ejecutivo

El centro educativo Jason Miller fue creado para brindar su servicio educativo a la comunidad estudiantil, con el fin primordial de dar sus servicios de enseñanza y aprendizaje para el correcto desarrollo de educación tanto en niños desde los 3 meses hasta niños de 12 años. El principal objetivo del proyecto es crear la solución informática que le permita al centro Educativo brindar una atención de calidad a los padres de familia al momento de ellos ingresar a sus hijos a esta institución.

Dentro del capítulo I se da a conocer el contexto del porque se realiza la aplicación la justificación previa y su análisis de fuerzas T que permitirá dar las pautas para el desarrollo de dicho sistema.

Luego el capítulo II se describe los requerimientos que son los que nos van a permitir conocer las reglas del negocio, estos requerimientos se los obtiene a partir de la entrevista, seguidamente de la matriz de requerimientos que describe cada uno de ellos en funcionales y no funcionales. En el capítulo III se presentan el árbol de problemas y de objetivos que son los que proporcionan un instrumento para su impacto de desarrollo, continuando con el capítulo IV que nos permite definir la matriz de análisis de alternativas y de impacto de objetivos para continuar con los modelos lógicos y físicos de la base de datos del sistema.

En el capítulo V se presenta la documentación del sistema con la especificación de los estándares utilizados para la programación y especificación de pruebas realizadas (unidad, carga). Finalmente el capítulo VI y VII nos presentan los recursos, presupuestos, cronograma, conclusiones y recomendaciones a tomar dentro del sistema.

Abstract

Jason Miller Education Center was created to provide educational services to the student community, the primary purpose of giving their services for teaching and learning for the proper development of education in children from 3 months to children 12 years. The main objective of the project is to create a software solution that allows the Educational center providing quality care to parents when they enter their children to this institution.

In Chapter I is given to know the context of that application is made prior justification and analysis of forces T will allow provide guidelines for the development of such a system.

Then chapter II requirements that are going to allow us to know the rules of business, these requirements are those obtained from the interview, then requirements matrix that describes each functional and nonfunctional described . In Chapter III the problem tree and objectives are those that provide a tool for development impact, continuing with Chapter IV which allows us to define the matrix analysis of alternatives and impact objectives to continue with present logical and physical models of the database system.

In Chapter V system documentation is presented to the specification of standards used for programming and specification tests (unit load).

Finally Chapter VI and VII show us the resources, budgets, schedule, conclusions and make recommendations within the system.

Capítulo I: Antecedentes

1.01 Contexto

El centro educativo Jason Miller localizada en la ciudad de Quito, provincia de pichincha, cantón Quito, fue creado para brindar su servicio educativo a la comunidad estudiantil, con el fin primordial de dar sus servicios de enseñanza y aprendizaje para el correcto desarrollo de educación tanto en niños desde 1 año hasta niños de 12 años.

Jason Miller es un centro que cuenta con ingresos económicos, los mismos les permiten atender a los niños con la calidad de atención que se merecen, esta institución actualmente no dispone de un sistema que proporcione un método ayuda en sus actividades en el área administrativa ya el espacio físico en las instalaciones y el personal no es capacitado, esto ha provocado una gran incertidumbre en este departamento debido a que ellos pierden gran parte de su tiempo en verificar las calificaciones de los estudiantes al momento de las matrículas así como también ha provocado que el Centro Educativo Jason Miller no tenga competitividad con otras instituciones educativas tanto a nivel cantonal, provincial y nacional; en lo cual el gobierno nacional requiere que las instituciones educativas sean las forjadoras de los profesionales del mañana es así que en Ecuador el ministerio de educación se encarga de que cada institución brinde educación de calidad.

Al no contar con un sistema informático capaz de disminuir el tiempo, agilidad y espacio en las actividades realizadas en el área administrativa, muchas veces los problemas que se han producido en la institución ha envuelto grandes problemas, uno de los importantes es tener a los padres de familia insatisfechos, ya que ellos como tutores de sus hijos son los clientes directos que tiene la Unidad Educativa y

quienes al asistir al Plantel pierden mucho tiempo por la falta de agilidad al emitir calificaciones o información referente a sus hijos.

La institución educativa no automatiza sus procesos y se encuentra en un desfase de las políticas de aprovechamiento de las tecnologías de la información, actualmente no tiene incorporada una herramienta de apoyo en el área administrativas ya que las mismas solo se las realiza en hojas de cálculo de Excel y otras llevadas a mano, es muy importante para la comunidad institucional contar con una herramientas de trabajo capaz de ayudar en las diferentes tareas establecidas por los directivos de la institución.

Debido a la complejidad y el costo de un sistema automatizado a nivel de educación se incrementan la necesidad de contar con un sistema capaz de brindar soporte de las labores académicas dentro de la institución. La implementación del sistema agilitara tiempo, costo y capacidad en el cumplimiento de las demandas requeridas por los usuarios en lo que se refiere a aplicaciones informáticas, siendo el campo educativo uno de los más requeridos en el mercado del software.

Al no automatizar los procesos informático requeridos el centro Educativo continuara con estos problemas seguirían trabajando normalmente y perdiendo credibilidad e imagen ante la sociedad lo que ocasionaría una gran ausencia de estudiantes que no querrán estudiar en la unidad educativa, tanto por la falta de tecnología como también por el prestigio de ellos como alumnos de la institución

1.02 Justificación

Las unidades educativas en el Ecuador son el pilar fundamental de la educación en todas las comunidades, ya que brindan apoyo al conocimiento de los estudiantes en el nivel inferior, medio y superior.

La implementación de un sistema académico para el centro educativo"Jason Miller" de la ciudad de Quito, para adaptar a todas las necesidades y las realidades de la institución. Constantemente las instituciones de educación investigan sobre cuáles son las necesidades más frecuentes con las que cuenta, ya que cada una de las instituciones trabaja bajo un régimen diferente.

Esto da paso para la realización de nuestra la aplicación la que tendrá como objetivo principal proceder a la creación de un software informático orientado a la Web que tiene por finalidad presentar la solución informática dirigida a la problemática presente actualmente en el centro de Educativo; la cual permitirá que los profesores ingresen las notas bajo diferentes modalidades directamente al sistema, llevar datos completos de alumnos. También generará todo tipo de informes.

Otro de los puntos importantes es que permitirá que tanto los padres de familia como la persona encargada de manejar el sistema no desperdicien tiempo valioso al momento de realizar la matriculación y los procesos requeridos de los estudiantes.

En la parte técnica se puede identificar los problemas que implica no tener un sistema informático web para apoyo en el área administrativa, pero más allá de eso las grandes ventajas de tenerlo.

Dicha solución posibilitara la administración de información vinculada, tanto de los alumnos, docentes y familiares de la Institución, desde el apoyo en la gestión escolar y académica que permitirá automatizar gran parte de las tareas rutinarias y administrativas del personal de la Institución, satisfaciendo las necesidades.

1.03 Definición del problema central (matriz T)

Mediante esta matriz podremos identificar de forma más rápida y detallada sobre la situación actual de la Empresa en base al problema establecido, causas por las que nos permitirán la implementación de mejora y sus desventajas.

Tabla 1

Matriz de análisis de fuerza " T "

Análisis de fuerza " T"					
Situación empeorada	**Situación actual**				**Situación mejorada**
Menor oportunidad de competencia en nivel educativo dentro de la institución al momento de prestar servicios.	Control de procesos académicos llevados a cabo con computador pero en programas que no son adecuados para llevar estas tareas.				Optimización de procesos para mejorar la atención que prestan a los estudiantes, padres de familia y profesores.
Fuerzas impulsadoras	**I**	**Pc**	**I**	**Pc**	**Fuerzas bloqueadoras**
Generación de registros en computador con programas descargados de internet para el proceso de ingreso de notas.	2	4	3	4	*Falta de implementación de recursos en el área administrativa.*
Mejoramiento para llevar registros de datos como son asistencias de alumnos y profesores en computador.	2	4	4	5	Falta de control de registros con una herramienta tecnología avanzada.
Investigación de necesidades de los padres de familia, docentes y alumnos al momento de agilizar algún trámite administrativo.	2	4	3	4	Escasa colaboración por parte de los directivos de la institución y/ o padres de familia.
Innovación tecnológica necesaria para generar respuestas rápidas.	3	4	3	4	Falta de compresión sobre la visión de las familias en el proceso de implementación de sistemas de control de procesos.

NOTA: la matriz de fuerzas "T" nos permite dar una visión detallada sobre la situación actual de la Institución

Capítulo II. Análisis de Involucrados

2.01 Requerimientos

2.01.01 Descripción del sistema actual

El centro educativo Jason Miller no trabaja con ninguna clase de sistema por lo que el control de procesos académicos los hacen manualmente; esta institución se fundó con el objetivo de alcanzar la formación integral de los niños potencializando sus capacidades y destrezas para formar seres competitivos, capaces de enfrentar los retos impuestos en la sociedad; tomando como activador fundamental el desarrollo de proyectos enfocado en el aspecto lúdico y la expresión artística.

La institución no cuenta con un sistema capaz de brindar soporte a las actividades realizadas por los directivos y docentes de la institución, la misma tiene muchas necesidades en cuanto al aspecto de matriculación pero no se cuenta con un software específico para apoyo, inicialmente vienen trabajando con software que se encuentra en el mercado y que es principalmente pagado.

El principal trabajo en cuanto a encontrar un sistema apropiado para matricular, registro de notas han causado el déficit de acceso a los dato de los alumno y falta de organización de la información; trayendo como consecuencia que la atención a los estudiantes, padres de familia y profesores se a deficiente; lo cual conlleva la pérdida de tiempo, pérdida económica y de datos.

Aquí describiremos algunos de los procesos que se realizan en la institución.

- PROCESO DE CONTROL DE ESTUDIANTES

El sistema registrara los datos de los estudiantes para el momento del ingreso de datos.

- PROCESO DE CONTROL DE DOCENTES

Se llevara un registro de datos de los docentes para el momento del ingreso de datos.

- PROCESO DE CONTROL DE ADMINISTRATIVOS.

Se llevara un registro de datos del personal administrativo al momento del ingreso de datos.

- PROCESO DE CONTROL DE ASIGNATURAS.

Se llevara un registro de las asignaturas impartidas a diario en la institución.

- PROCESO DE CONTROL DE PERIODOS LECTIVOS

  Esta tabla almacena la fecha de inicio y fin de año lectivo.

- PROCESO DE CONTROL DE CONTROL DE JORNADAS

Se llevara un control estricto de las jornadas de trabajo de los docentes.

- PROCESO DE CONTROL DE CURSOS/NIVELES/PARALELOS

Se llevara control de los cursos, niveles y paralelos para que al momento de la matriculación las aulas queden con el número adecuado de niños.

- PROCESO DE CONTROL DE INSCRIPCIONES

Para llevar a cabo el proceso de matriculación se deberá realizar una inscripción previa.

- PROCESO DE MATRICULAS

La Institución tiene muchos niños de edad desde los 3 meses hasta los 12 años lo cual consiste en llevar un adecuado control de cuantos alumnos (as) se encuentran matriculados dentro de la misma.

- PROCESO DE CONTROL DE DOCUMENTOS ENTREGADOS POR LOS ESTUDIANTES.

El sistema permitirá llevar a cabo la actividad de controlar todos los documentos entregados por los estudiantes al momento que los administrativos, docentes o padres de familia así lo deseen.

- PROCESO DE CONTROL DE HORARIOS

Permite tener un horario de cada profesor para impartir las asignaturas a cada curso.

- PROCESO DE CONTROL DE NOTAS POR PERIODO (DIARIO, BIMESTRAL, QUIMESTRAL)

Este proceso permite mantener un registro de nota acorde a su periodo de evaluación. Las notas se las llevara de acuerdo a los porcentajes destinados por el ministerio de educación.

- PROCESO DE EVALUACION A LOS ESTUDIANTES.

A medida que los niños realizan sus actividades se proporcionara un campo que guardara las evaluaciones realizadas por los estudiantes.

- PROCESO DE REPORTES DE LAS AVALUACIONES.

Se obtendrá mensualmente reportes de cómo han ido evaluando y como ha sido su desarrollando en cuanto al manejo de las actividades planteadas.

- GAMA COMPLETA DE CONSULTAS CON OPCION A REPORTES A TODAS LAS TABLAS Y PROCESOS FACTIBLES.

Permite que se realice consulta de todos los reportes generados en el trascurso del proceso académico.

2.01.02 Visión y alcance

Brindar una educación de calidad y calidez, mejorar las condiciones de escolaridad, el acceso y la cobertura de la educación, y desarrollar un modelo educativo que responda a las necesidades locales.

La enseñanza debe tener en cuenta no solo la piscología de cada alumno, sino también las teorías de aprendizaje. Sin embargo, la educación en general y la informática Educativa en particular, carecen aún de estima en influyentes núcleos de la población, creando entonces serios problemas educativos que resultan difíciles de resolver y que finalmente condicionan el desarrollo global de la sociedad. La mejora

en el aprendizaje resulta ser uno de los anhelos más importantes de todos los docentes; de ahí que la enseñanza individualizada y el aumento de productividad de los mismos son los problemas críticos; el aprendizaje se logra mejor cuando es activo, es decir cuando cada estudiante crea sus conocimientos en un ambiente dinámico de descubrimiento.

La sociedad y la escuela Jason Miller pueden valerse de la tecnología para mejorar su respuesta en cuanto a tiempo y agilidad de la información. En el ámbito educativo los recursos tecnológicos viene a compensar estas dificultades, nuestro sistema servirá de apoyo a los docentes, estudiantes y padres de familia, ellos fueron también los que impulsaron la búsqueda de nuevos métodos de evaluación, insatisfechos con las descripciones y explicaciones que obtenían a partir de los sistemas tradicionales.

2.01.03 Entrevista

La matriz de entrevistas nos permite por medio de preguntas conocer cuáles son las necesidades más comunes dentro de la Institución para así poder determinar los procesos a seguir.

Véase en Anexo
A.01

2.01.04 Matriz de requerimientos

Tabla 2

Matriz de requerimientos

Matriz de requerimientos funcionales						
Identificador	**Descripción**	**Fuente**	**Prioridad**	**Tipo**	**Estado**	**Usuarios involucrados**
Requerimientos funcionales						
RF001	Se llevara control de los estudiantes y docentes.	Director	Alta	Sistema	Aprobado	-docentes -estudiantes secretaria
RF002	Se llevara a cabo el control de asignaturas impartidas diariamente.	Director	Alta	Sistema	En revisión	-alumnos -docentes -secretaria
RF003	Se llevara un control de notas diarias, mensuales de los estudiantes para así sacar los promedios quimestrales.	Director	Alta	Sistema	En revisión	-alumnos -docentes secretaria -padres de familia
RF004	Se llevara un control de jornadas de trabajo de los docentes ya que los turnos serán acorde al área donde se encuentren.	Docente Director	Alta	Sistema	En revisión	-docentes -alumnos Administrativos.
RF005	La aplicación deberá contener el campo de control de cursos, niveles y paralelos.	Director	Alta	Sistema	En revisión	-docentes Administrativos. -alumnos
RF006	El sistema deberá contener un campo que permitirá realizar inscripciones previas, para así poder tener la cantidad de niños acordes a los cupos.	**Director**	**Alta**	**Sistema**	**En revisión**	Administrativos. -docentes
RF007	Al momento de la matricula se llevara a cabo la recepción de documentos de los estudiantes los cuales serán cargados en su hoja de matrícula.	**Director**	**Alta**	**Sistema**	**En revisión**	Administrativos. -docentes -padres de familia.

Tabla 3

Matriz de requerimientos no funcionales

MATRIZ DE REQUERIMIENTOS NO FUNCIONALES						
IDENTIFICADOR	**DESCRIPCIÓN**	**FUENTE**	**PRIORIDAD**	**TIPO**	**ESTADO**	**USUARIOS INVOLUCRADOS**
Requerimientos no funcionales						
NRF001	La aplicación deberá ser compatible con cualquiera navegador.	Director	Media	Usuario	En revisión	- administrativos
NRF 002	Respaldar la base de datos	Director	Media	Usuario	En revisión	- administrativo - programador
NRF03	Los datos de los usuarios no deberán repetirse en la base de datos	Director	Media	Usuario	En revisión	-alumnos -docentes -director - administrativo
NRF04	Se plasmara en impresión las notas generadas bimestralmente por los estudiantes	Docente	Media	Usuario	En revisión	- alumnos -docentes -director - administrativo
NRF05	Permitir hacer seguimiento y monitoreo de actividades	Administrativo	Media	Usuario	En revisión	- administrativo -docente

NOTA: En la matriz de requerimientos se detalla los requerimientos funcionales y no funcionales para así poder desarrollar nuestros diagramas de casos de uso de sistema.

2.01.05 Descripción detallada

Tabla 4

Requerimiento funcional 001

Se llevara registro de la base de datos	alumnos y docentes en	ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12//2014	FECHA DE ACTUALIZACIÓN	
Identificador	**RF001**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código de usuario Código del alumno Código del docente Nombre del alumno Nombre del docente		
Descripción	Una vez ingresado el código o nombre del alumno o el docente la aplicación inmediatamente permitirá su registro.		
Datos de Salida	Registro de docentes y estudiantes		
Resultados esperados	Los resultados esperados con este requerimientos llevar un control estricto de los estudiantes y docentes.		
Origen	DIRECTOR		
Dirigido a	Docentes Estudiantes Secretaria		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	1.-Para realizar el requerimiento primero debe tener un registro de alumnos y docentes. 2.-Una vez ingresado al sistema los docentes y estudiantes ingresaran registro en la institución.		
Poscondiciones	1.- Si los usuarios (docentes, estudiantes) no tienen ningún registro en la institución no se podrá llevar control alguno.		
Criterios de Aceptación	Los datos consultados no se alteran en la base de datos y el reporte resultante debe poder imprimirse.		

Tabla 5

Requerimiento funcional 002

Se llevara a cabo el control de asignaturas impartidas diariamente.		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**RF002**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código asignatura Nombre de la asignatura		
Descripción	Permite ingresar dentro del sistema una materia o asignatura.		
Datos de Salida	Interactuar en las distintas asignaturas que se destinan para trabajar.		
Resultados esperados	Los resultados esperados con este requerimiento es ingresar a las asignaturas designadas de acuerdo a cada curso, nivel, paralelo.		
Origen	DIRECTOR		
Dirigido a	Secretaria		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	1.-Para realizar el requerimiento primero debe tener las asignaturas con las que se va a trabajar 2.-Los datos de la materia se almacenan dentro de la base de datos.		
Poscondiciones	1.- Si no se puede controlar la asignatura permitir recuperarla mediante un código.		
Criterios de Aceptación	Permitir que los usuarios registren las asignaturas dependiendo de los niveles.		

Tabla 6

Requerimiento funcional 003

Se llevara un control de notas diarias, mensuales de los estudiantes para así sacar los promedios quimestrales.		**ESTADO**	**ANÁLISIS**
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**RF003**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código docente Seleccionar el nivel, curso, paralelo. Selección de estudiante. Seleccionar alumnos matriculados		
Descripción	Una vez ingresado el código del docente se validara los datos luego se escogerá el nivel y el alumno para llenar notas de alumnos matriculados.		
Datos de Salida	Ingreso de notas.		
Resultados esperados	Los resultados esperados con este requerimiento es ingresarlas notas solo a alumnos matriculados.		
Origen	DIRECTOR		
Dirigido a	Alumnos Docentes Secretaria Jefe administrativo		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	1.-Para realizar el requerimiento primero debe tener registro de los alumnos en lavase de datos. 2.-Una vez ingresado al sistema el docentes podrá ingresar notas diarias, mensuales, quimestrales.		
Poscondiciones	1.- Si los usuarios docentes no se acuerda de la clave podrán recuperarla.		
Criterios de Aceptación	Permitir que los docentes cambien las notas con la debida autorización del jefe administrativo.		

Tabla 7

Requerimiento funcional 004

Se llevara un control de jornadas de trabajo de los docentes ya que los turnos serán acorde al área donde se encuentren.		**ESTADO**	**ANÁLISIS**
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	

Identificador	**RF004**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código del docente. Selección de jornada. Código del alumno		
Descripción	Los directivos generaran un control de las jornadas para cada uno de los docentes.		
Datos de Salida	Generación de jornadas. Generación de horarios dentro de las jornadas para alumnos y docentes.		
Resultados esperados	Tener un control adecuado de las jornadas del trabajo tanto de los alumnos como de los docentes.		
Origen	DIRECTOR		
Dirigido a	Alumnos Docentes Jede administrativo Secretaria		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Para realizar este proceso tendremos en cuenta que cada docente tiene distintas horas de trabajo.		
Poscondiciones	Si el docente no cuenta con jornada de trabajo se le ingresar nuevamente.		
Criterios de Aceptación	Permitir que el docente genere registro de su jornada de trabajo.		

Tabla 8

Requerimiento funcional 005

La aplicación deberá contener el campo de control de cursos, niveles y paralelos para así tener conocimiento de cuantos alumnos quedaría en cada uno.		**ESTADO**	**ANÁLISIS**
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	

Identificador	**RF005**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código del curso/nivel/paralelo Código del alumno		
Descripción	Los directivos generaran un control de las cursos/ niveles/ paralelos para que cada estudiantes estén ingresados en sus respectivos niveles.		
Datos de Salida	Generación de cursos/niveles/paralelos. Listados de alumnos registrados en cada curso/nivel/paralelo.		
Resultados esperados	Mantener un registro de todos los estudiantes ingresados en cada nivel para así tener conocimiento de cuantos niños faltan y si hay cupos disponibles.		
Origen	DIRECTOR		
Dirigido a	Alumnos Docentes Secretaria Jefe administrativo		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Para realizar este proceso debemos tener en cuenta el ingreso desde la matriculación para así generar cupos.		
Poscondiciones	Si no hay cupo disponible y se tiene de 1 a 3 niños adicionales permitir el ingreso.		
Criterios de Aceptación	Permitir que los directivos generen listados de los alumnos en cada nivel.		

Tabla 9

Requerimiento funcional 006

El sistema deberá contener un campo que permitirá realizar inscripciones previas, para así poder tener la cantidad de niños acordes a los cupos acordados.		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	Lady Moreno
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**RF006**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código de inscripción Código alumno Asignación de alumno a grupos. Registro de materias Registro del docente		
Descripción	Los padres de familia al momento de la inscripción ya tienen un cupo designado el cual ya está determinado el docente y el curso.		
Datos de Salida	Registro de estudiantes.		
Resultados esperados	Con este requerimiento se espera generar las listas los estudiantes inscritos en la institución para así saber cuentos cupos disponibles en cada nivel existen.		
Origen	DIRECTOR		
Dirigido a	Alumnos Docentes Secretaria Jefe administrativo		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Para la realización de este requerimiento se debe tener en cuenta la edad de los niños para el ingreso de cada nivel.		
Poscondiciones	1.-Para ejecutar este requerimiento el alumno debe estar ingresado en el sistema. 2.-Una vez ingresado al sistema la secretaria debe llenar los campos para la realización de la inscripción.		
Criterios de Aceptación	Permitir que el padre de familia cancele la inscripción en 5 días vigentes.		

Tabla 10

Requerimiento funcional 007

Después de la inscripción se realizara la matricula que llevara a cabo la recepción de documentos de los estudiantes los cuales serán cargados en su hoja de matrícula		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	
FECHA DE CREACIÓN	04/12/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**RF005**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Código de usuario Nombre del estudiante		
Descripción	El personal administrativo que realiza la matricula deberá cargar en archivos adjunto los documentos escaneados de los estudiantes.		
Datos de Salida	Registro de documentos en la hoja de matrícula de cada estudiante.		
Resultados esperados	Registro de documentos en la hoja de matrícula de los estudiantes. .		
Origen	DIRECTOR		
Dirigido a	Jefe administrativo Secretaria Padres de familia Estudiantes		
Prioridad	5		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	1.-Para realizar este proceso tendremos que implementar un campo para la carga de los escaneados de los documentos de cada alumno.		
Poscondiciones	1.- El personal administrativo ingresara a guardar en carpetas los registros de cada alumno.		
Criterios de Aceptación	Los alumnos, docentes, personal administrativo y padres de familia tendrán acceso a la información de los alumnos.		

Tabla 11

Requerimiento no funcional 001

La aplicación deberá ser compatible con cualquier navegador.		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	
FECHA DE CREACIÓN	15/06/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**NRF001**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Conexión a internet Código de usuarios		
Descripción	La aplicación será capaz de funcionar con cualquier navegador ya que es un sistema que está orientado a la web para brindar mayor comodidad al usuario.		
Datos de Salida	Acceso al sistema.		
Resultados esperados	Brindar más comodidad al usuario permitiéndole utilizar la aplicación en el navegador de su preferencia.		
Origen	DIRECTOR		
Dirigido a	Programador		
Prioridad	3		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Este proceso se lo realizara implementado un lenguaje optimo que permitirá que la aplicación se ejecute en cualquier navegador que el usuario requiera.		
Poscondiciones	El sistema se ejecutara de excelente forma en los diferentes tipos de navegadores.		
Criterios de Aceptación	El usuario tendrá mayor comodidad de utilizar la aplicación ya sea en Internet Explore, Mozilla Firefox, Google Chrome etc.		

Tabla 12

Requerimiento no funcional 002

El sistema no deberá demorarse más de 3 minutos.		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	
FECHA DE CREACIÓN	15/06/2014	FECHA DE ACTUALIZACIÓN	
Identificador	**NRF002**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Acceso al sistema Código de usuarios Asignaturas Ingreso de base de datos Procesos de gestión del sistema		
Descripción	La aplicación será capaz de realizar cualquier acción en menos de 1 minuto ya que tendremos una excelente conexión de datos con el servidor de Software.		
Datos de Salida	Acceso al sistema y utilización de la misma		
Resultados esperados	Brindar rapidez al sistema y utilización de la misma		
Origen	DIRECTOR		
Dirigido a	Programador		
Prioridad	3		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Este proceso se lo realizara utilizando los códigos de programación necesarios para que el software realice la compilación de forma rápida.		
Poscondiciones	La aplicación se ejecuta en forma correcta ya que su tiempo de compilación será mínimo.		
Criterios de Aceptación	El usuario realizara sus actividades sin preocuparse por el tiempo de espera.		

Tabla 13

Requerimiento no funcional 003

Los datos de los usuarios no deberán repetirse en la base de datos		ESTADO	ANÁLISIS
CREADO POR	Lady Moreno	ACTUALIZADO POR	
FECHA DE CREACIÓN	15/06/2014	FECHA DE ACTUALIZACIÓN	

Identificador	NRF003		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Ingreso al sistema Toma de datos de usuario		
Descripción	El usuario no debe ser registrado dos veces y tampoco sus datos pueden repetirse.		
Datos de Salida	El usuario podrá realizar sus actividades sin que su información sea repetida.		
Resultados esperados	Tener un perfecto control de usuarios ingresados en la base de datos repetidos.		
Origen	DIRECTOR		
Dirigido a	Secretaria Programador		
Prioridad	4		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	Para realizar este requerimiento se hará a programación respectiva en la aplicación para que se detecte datos repetidos y no los vuelva a ingresar.		
Poscondiciones	El usuario tendrá registrado sus datos exclusivamente de uso único en el sistema.		
Criterios de Aceptación	La aplicación permitirá que el usuario registrado no tenga datos repetidos para mayor comodidad tanto de usuario como administrador.		

Tabla 14

Requerimiento no funcional 004

Se plasmara en impresión las notas generadas bimestralmente por los estudiantes		**ESTADO**	**ANÀLISIS**
CREADO POR	Lady Moreno	ACTUALIZADO POR	
FECHA DE CREACIÒN	15/06/2014	FECHA DE ACTUALIZACIÒN	
Identificador	**NRF004**		
Tipo de Requerimiento	Critico	Tipo de Requerimiento	Funcional
Datos de Entrada	Registro de notas Registro de usuarios		
Descripción	Los padres de familia tendrá la opción de tener las notas de sus hijos impresas, de igual manera el docente puede realizar esto para llevar un control de notas de los estudiantes.		
Datos de Salida	Notas impresas		
Resultados esperados	Tener un control de las notas de los estudiantes matriculados.		
Origen	DIRECTOR		
Dirigido a	Docentes Alumnos Secretaria Jefe administrativo		
Prioridad	3		
Requerimientos Asociados	Ninguno		
ESPECIFICACION			
Precondiciones	1.-Para realizar este proceso daremos la opción de que el software permita la impresión de las notas de los alumnos. 2.-Se configurara una impresora que lleve a cabo este proceso.		
Poscondiciones	Los usuarios podrán realizar las impresiones que sean necesarias.		
Criterios de Aceptación	Los docentes llevaran un registro documentado de las notas y desempeño diario de los alumnos.		

2.02 Mapeo de involucrados

Fig 1: Mapeo de Involucrados

El análisis de los involucrados es un instrumento que permite identificar a aquellas personas y organizaciones interesadas en el problema central de un proyecto

2.03 Matriz de involucrados

La matriz de involucrados nos permite identificar a aquellas personas e instituciones interesadas en el éxito del proyecto así como las que contribuyen y son afectadas con el éxito del mismo.

A.02

Capitulo III: Problemas y Objetivos

3.01. Árbol de problemas

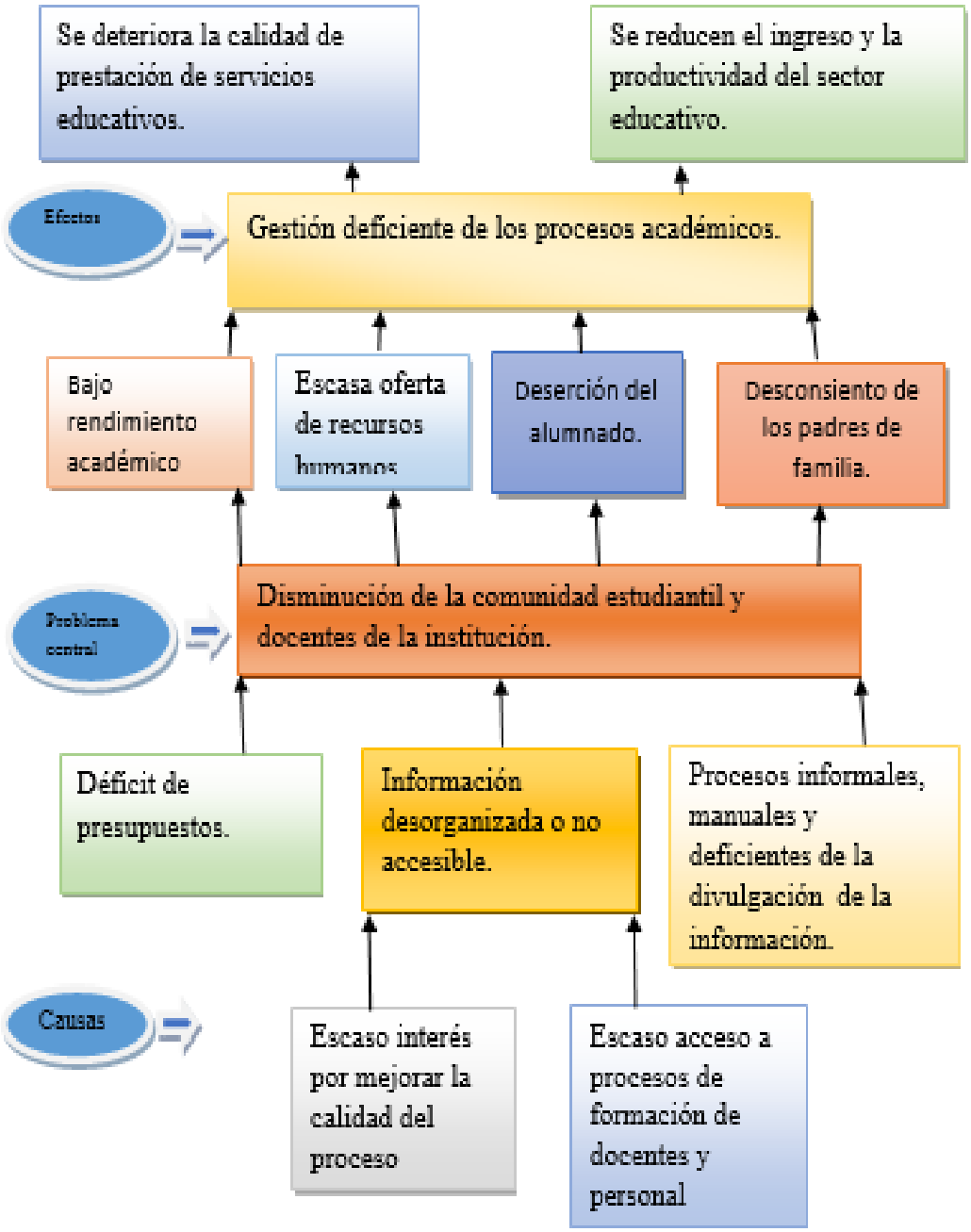


Fig 2: *Árbol de problemas*

El árbol de problemas detalla la problemática que se va a resolver, donde se verá las condiciones negativas percibidas por los involucrados.

3.02 Árbol de objetivos

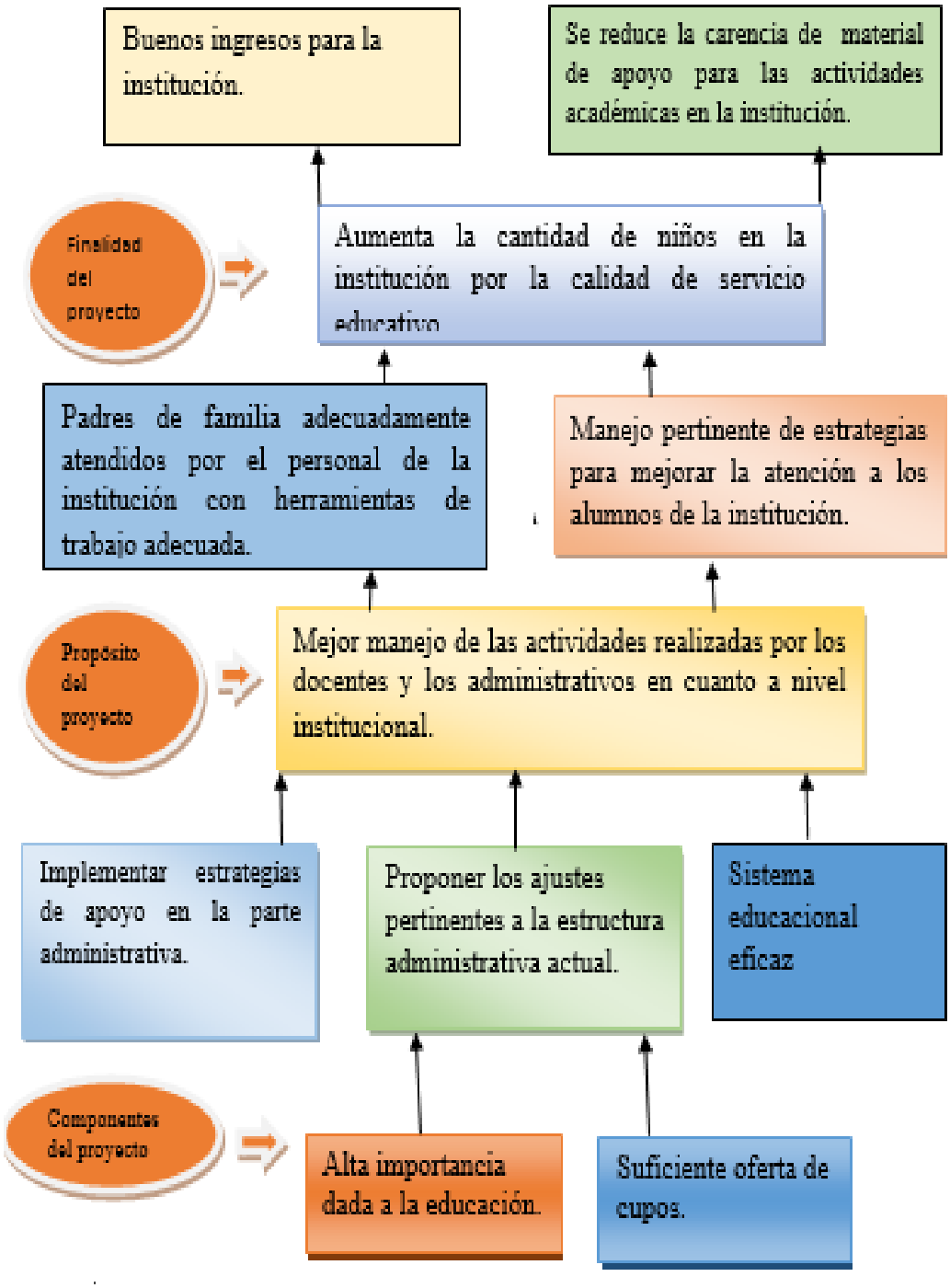


Fig 3: Árbol de objetivos

En el árbol de objetivos, los problemas encontrados en el árbol de problemas se convierten en soluciones o en objetivos.

3.03 Diagramas de casos de uso

El diagrama de casos de uso enseña el conjunto de actividades que se realizaron y los actores involucrados, este diagrama nos ayudara a representar el funcionamiento y la organización del sistema.

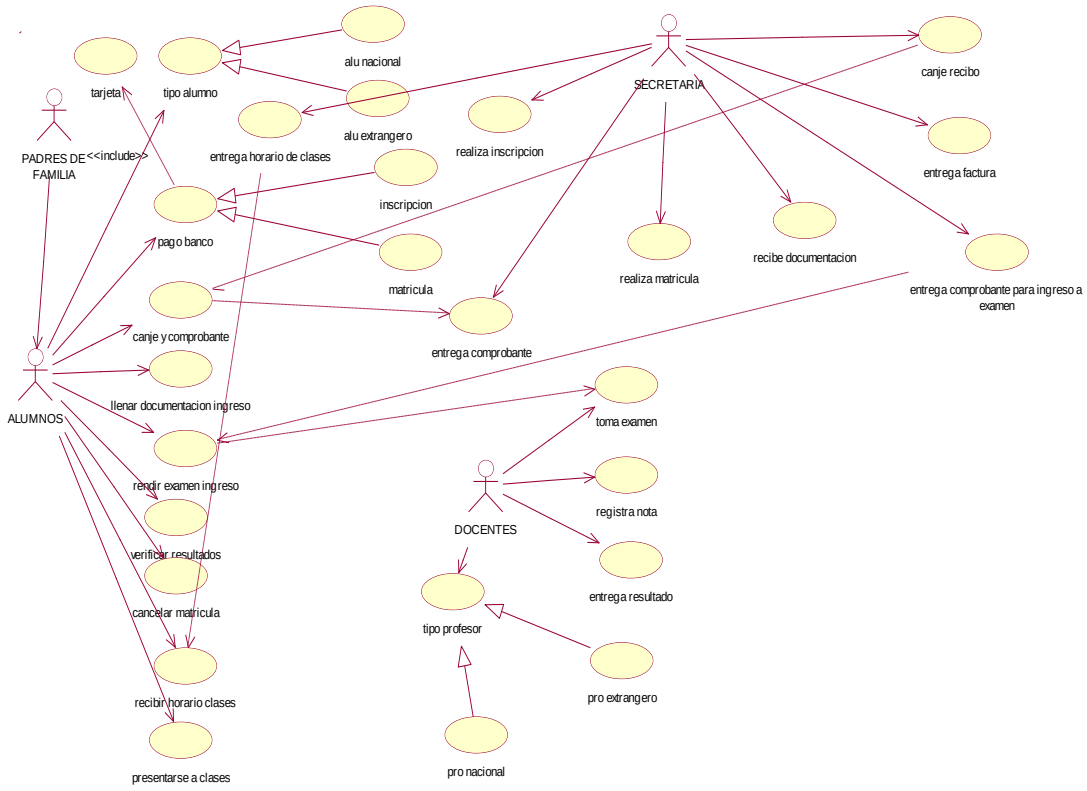


Fig 4: Diagramas de casos de uso

El diagrama de caso de uso general: representa la situación actual en la empresa.

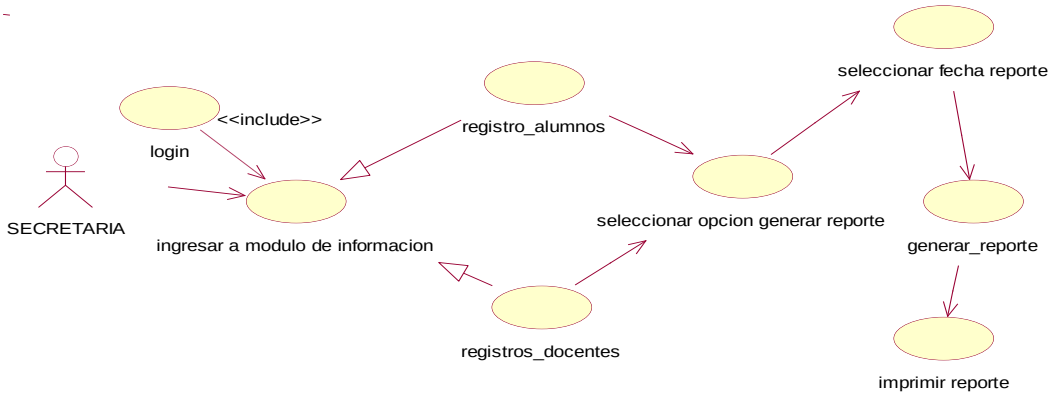


Fig 5: El diagrama caso de uso 001

El diagrama caso de uso 001 representa: el registro de alumnos y docentes en la base de datos.

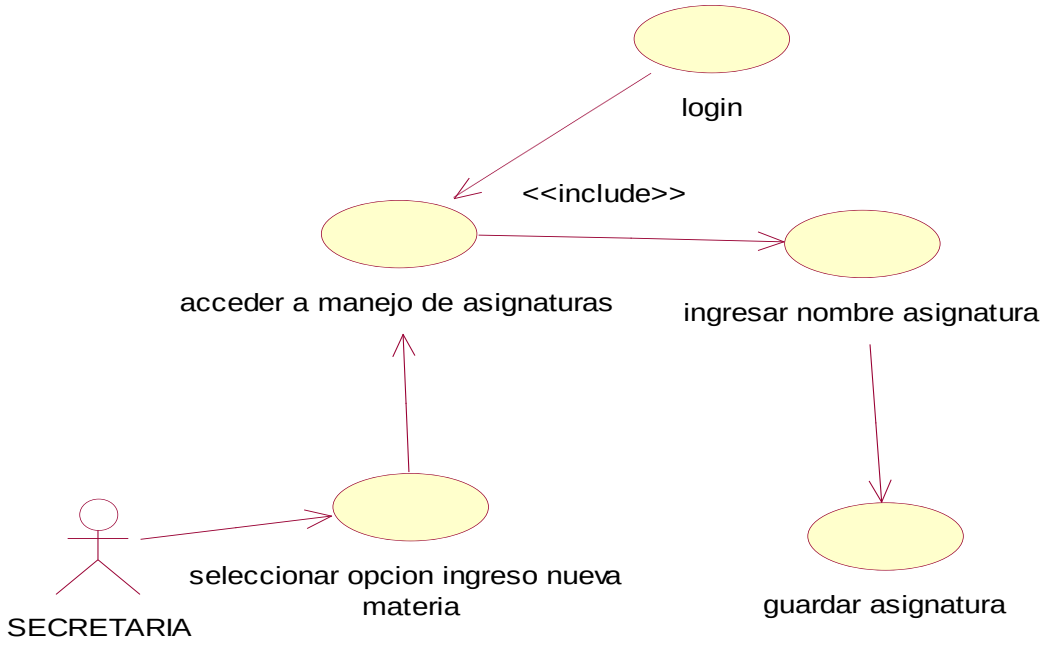


Fig 6: El diagrama caso de uso 002

El diagrama caso de uso 002 representa: el control de asignaturas en las base de datos

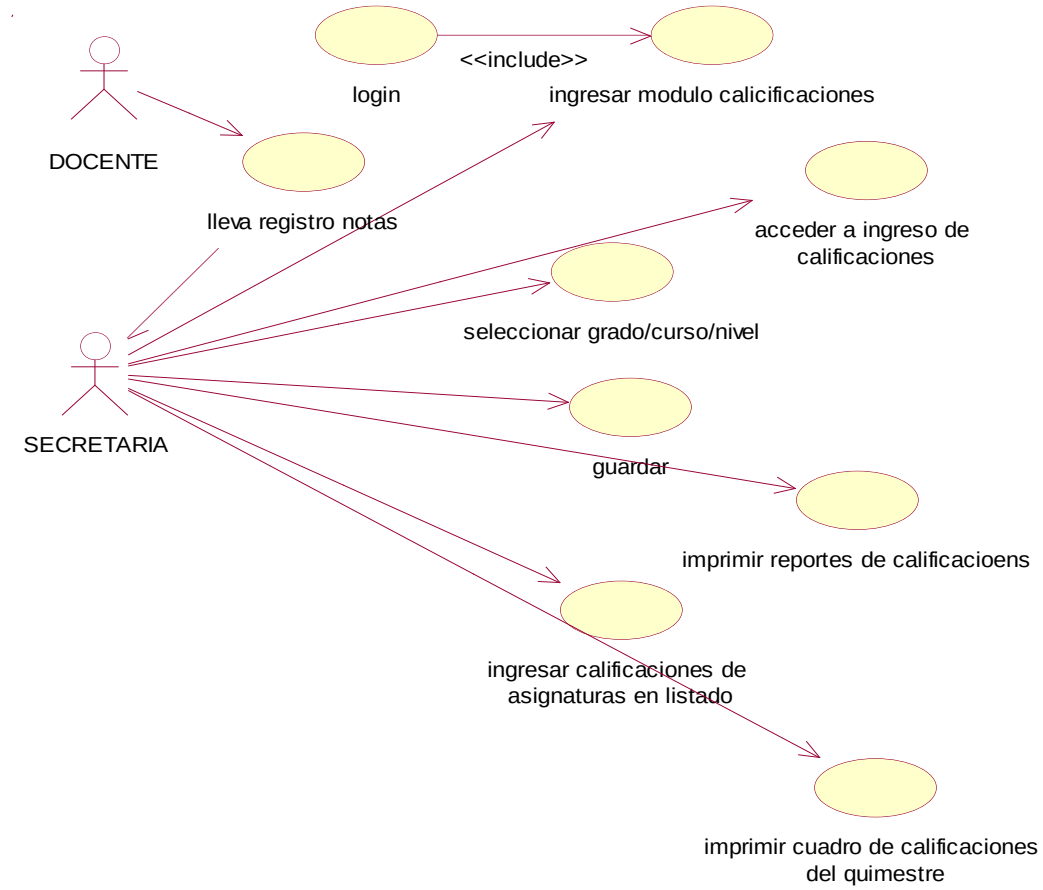


Fig 7: El diagrama caso de uso 003

El diagrama caso de uso 003: nos da a conocer el registro de las notas acorde a cada nivel.

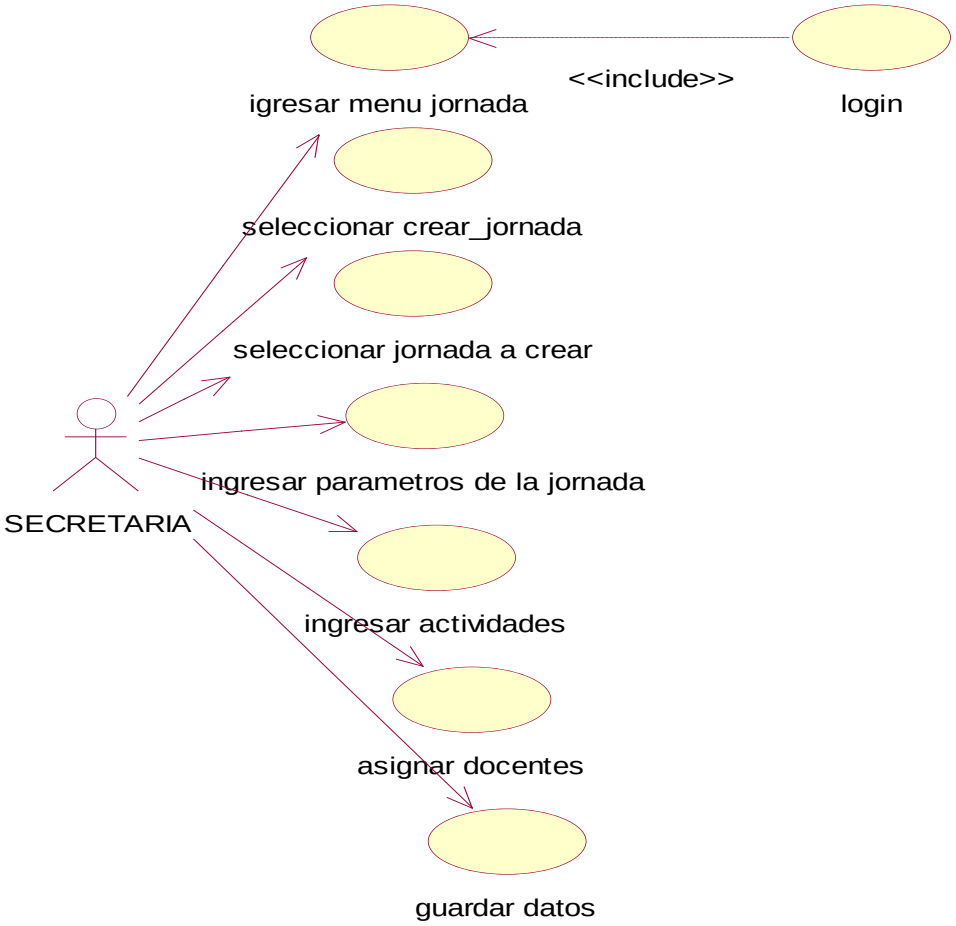


Fig 8: El diagrama caso de uso 004

El diagrama caso de uso 004 representa: a los procesos que se va a utilizar en el registro de jornada de los docentes de la institución.

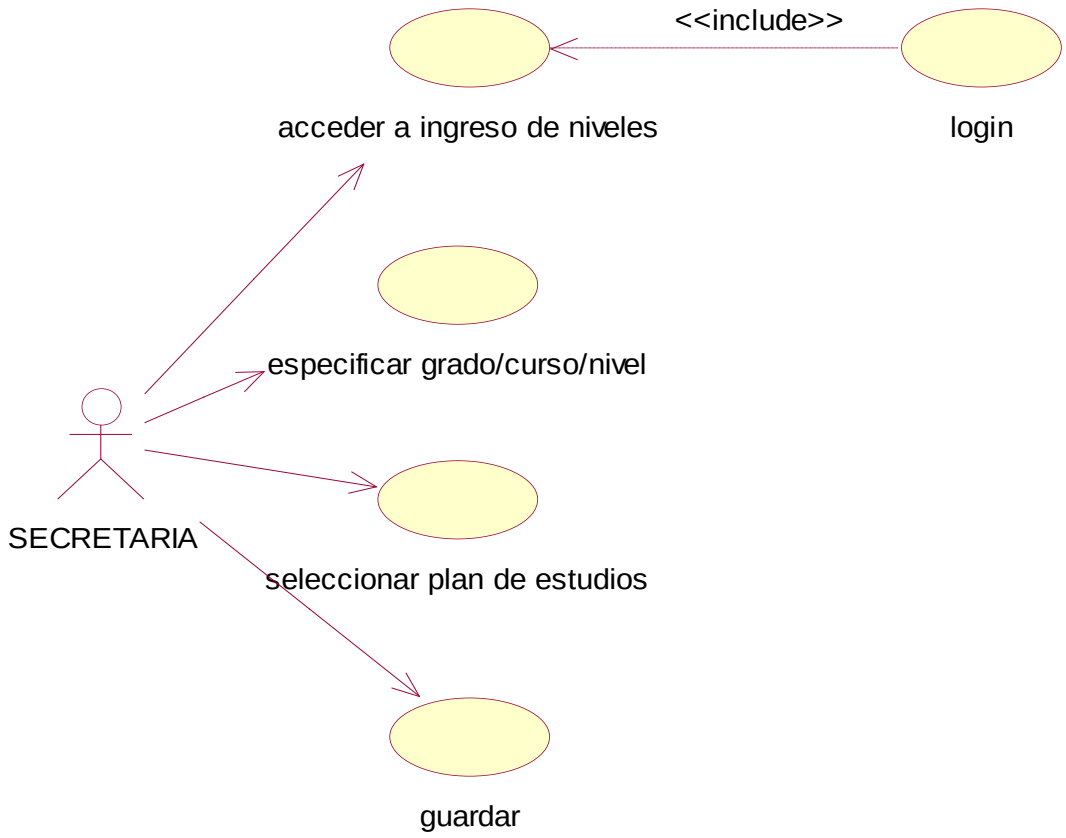


Fig 9: El diagrama caso de uso 005

El diagrama caso de uso 005: nos da a conocer que el sistema será un sistema donde se manejara selección de cursos/niveles/grados.

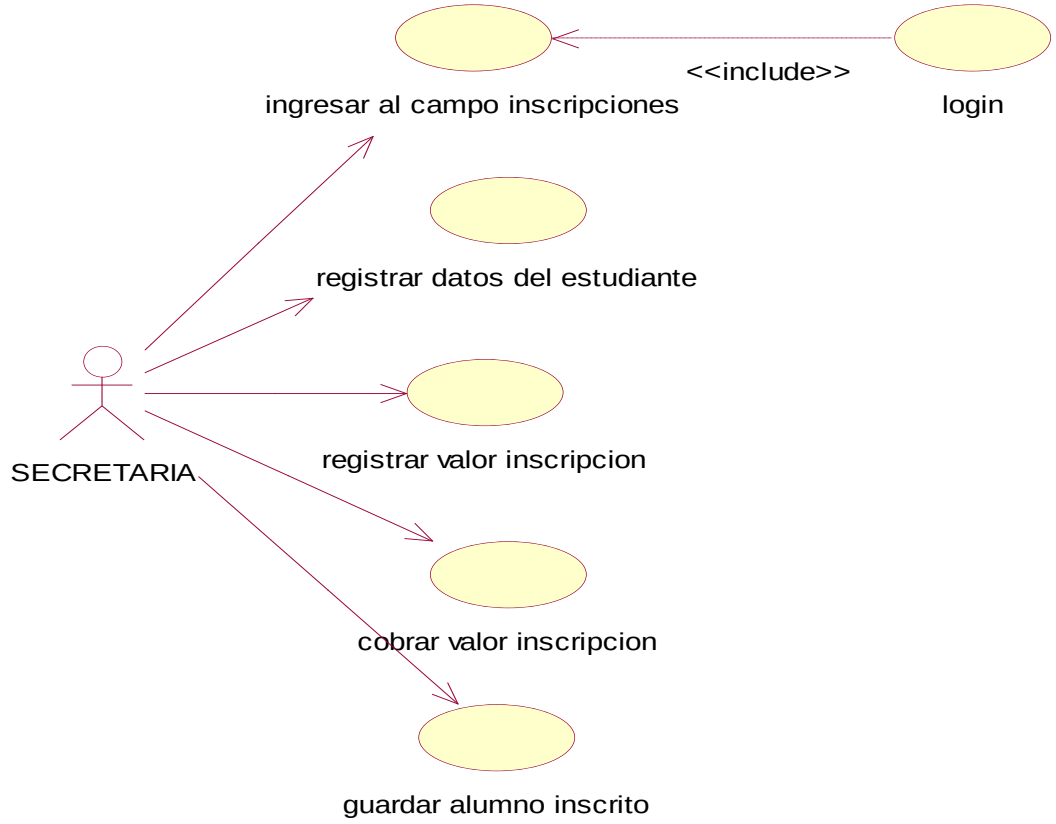


Fig 10: El diagrama caso de uso 006

El diagrama caso de uso 006: nos da a conocer que el sistema contara con registro de inscripciones previas a la matrícula para la obtención de los cupos.

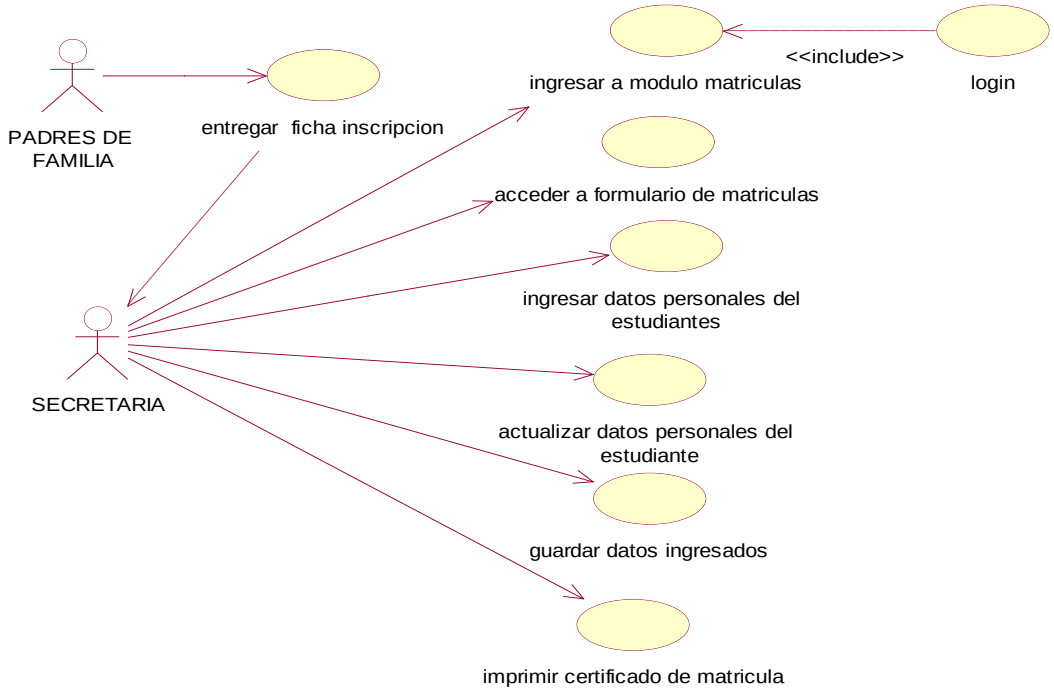


Fig 11: El diagrama caso de uso 008

El diagrama caso de uso 008: nos da a conocer que el sistema realizara matriculación de los estudiantes que ya este inscritos en el sistema con la previa presentación a la matrícula.

3.04 Diagramas de caso de uso de realización

En nuestro diagrama de casos de uso visualizaremos a nuestros actores, las actividades y los procesos que realizaran cada uno de ellos.

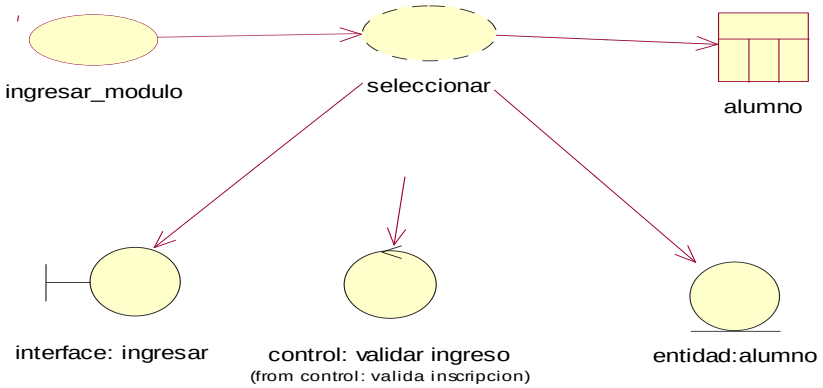


***Fig 12*:** Caso de uso de realización 001

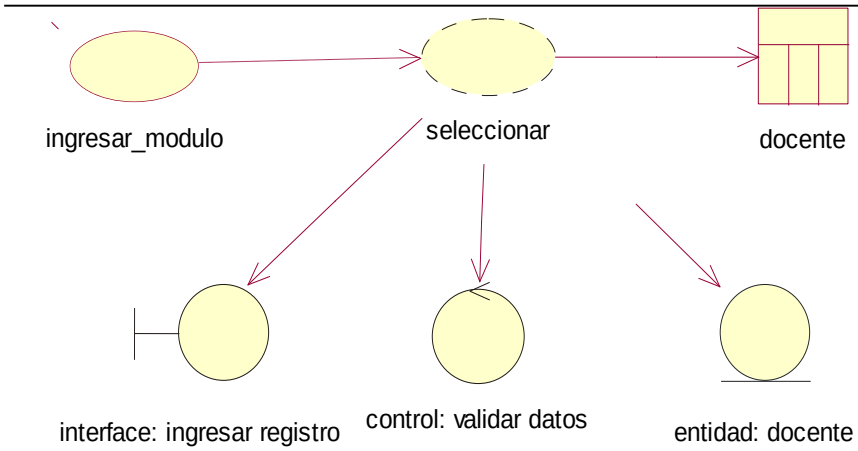


Fig 13: Caso de uso de realización 001

Tabla 15

Registro alumnos y docentes

Nombre:	**REGISTRAR ALUMNOS Y DOCENTES**
Identificador:	UC001
Responsabilidades:	Realizar el registro, actualización y eliminación de alumnos, docentes.
Tipo:	SISTEMA
Referencias casos de uso	UC001
Referencias requisitos	RF001
PRECONDICIONES	
De instancia **Se necesita una interface para que se registren los alumnos y docentes.** **Se necesita una instancia para que el administrador realice un parámetro de búsqueda.** **Se ubicara un control para que se verifique en la base de datos la información guardada.** **Se necesitara un control para que se valide el ingreso de datos en el sistema.** **Se requerirá una instancia para el tipo de datos.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Validar campo de cédula.** **Identificar el tipo de docente.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Ingreso en datos personales de alumnos y docentes. **Validación de campos de alumnos y docentes.**	

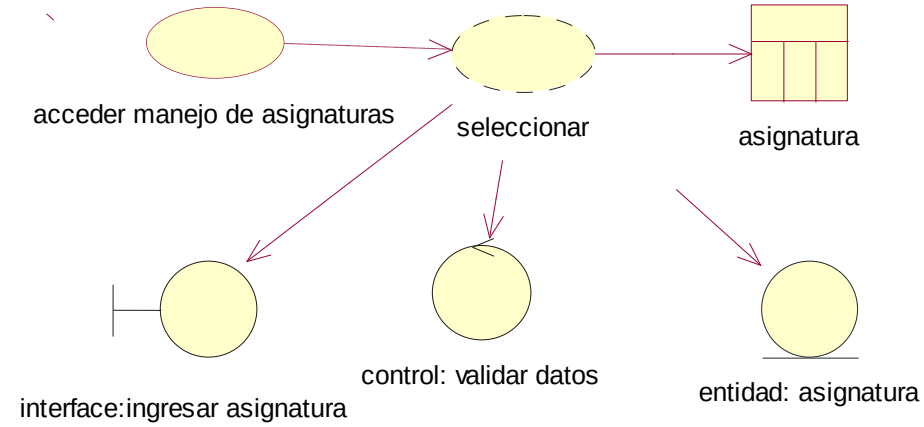


Fig 14: Caso de uso de realización 002.

Tabla 16

Control de asignaturas

Nombre:	**CONTROL DE ASIGNATURAS.**
Identificador:	UC002
Responsabilidades:	Realizar el control de asignaturas.
Tipo:	SISTEMA
Referencias casos de uso	UC003
Referencias requisitos	RF003
PRECONDICIONES	
De instancia **Se necesitará una interface para que el ingreso de las asignaturas.** **Se ubicará un control para que verifique las asignaturas ingresadas en la base de datos para generar la información solicitada.** **Se necesita una entidad para ir guardando la información.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Permitir que el usuario ingrese asignaturas.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Ingreso a asignaturas. **Controlar las actividades dentro de cada asignatura.**	

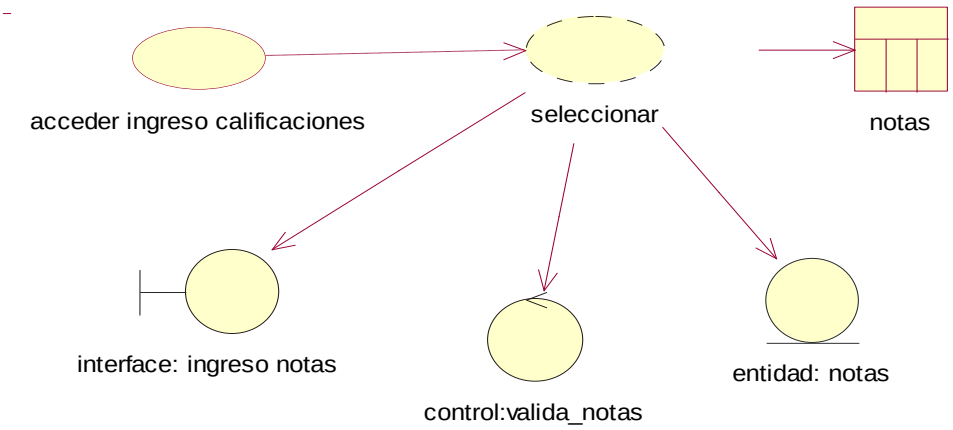


Fig 15: Caso de uso de realización 003

Tabla 17

Ingreso de notas en el sistema

Nombre:	**INGRESO DE NOTAS EN EL SISTEMA**
Identificador:	UC003
Responsabilidades:	Se realizará el ingreso de las notas de cada asignatura en el sistema.
Tipo:	SISTEMA
Referencias casos de uso	UC003
Referencias requisitos	RF003
PRECONDICIONES	
De instancia **Se necesitará un interface para que el docente la carga de las notas de cada alumno por materia.** **Se colocará un control para ser validada la información que está solicitando para el ingreso.** **Se necesitará una entidad de notas para ir generando y guardando los reportes.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Permitir que el docente cambie notas si así lo desea por algún error suscitado al momento del registro.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Reporte de notas semanales, mensuales y quimestrales.	

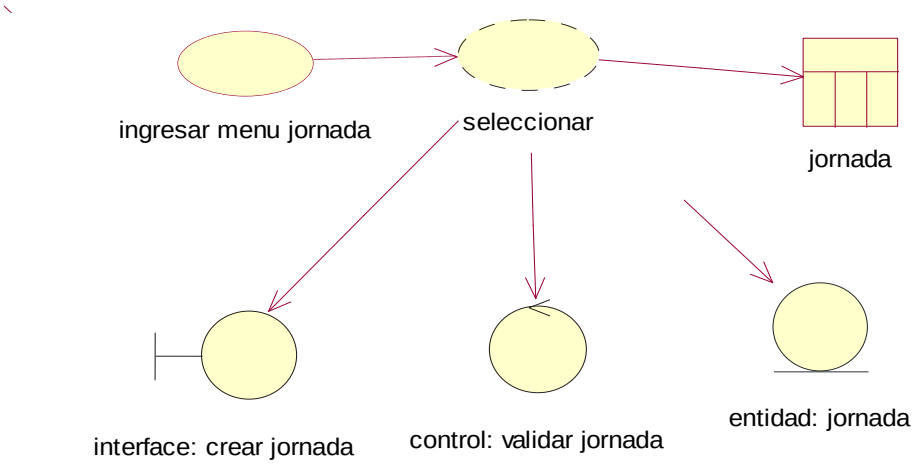


Fig 16: Caso de uso de realización 004**.**

Tabla 18

Manejo de jornada de trabajo del docente

Nombre:	**MANEJO DE JORNADA DE TRABAJO DEL DOCENTE**
Identificador:	UC004
Responsabilidades:	Se realizará el ingreso de jornadas de trabajo de los docentes acorde a su área.
Tipo:	SISTEMA
Referencias casos de uso	UC004
Referencias requisitos	RF004
PRECONDICIONES	
De instancia **Se necesitará una interface jornada para poder crear un registro de docentes a cargo de ciertas áreas.** **Se necesita una entidad jornada para ir guardando la información.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Permitir cambiar jornadas de trabajo.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Registro de jornada de trabajo.	

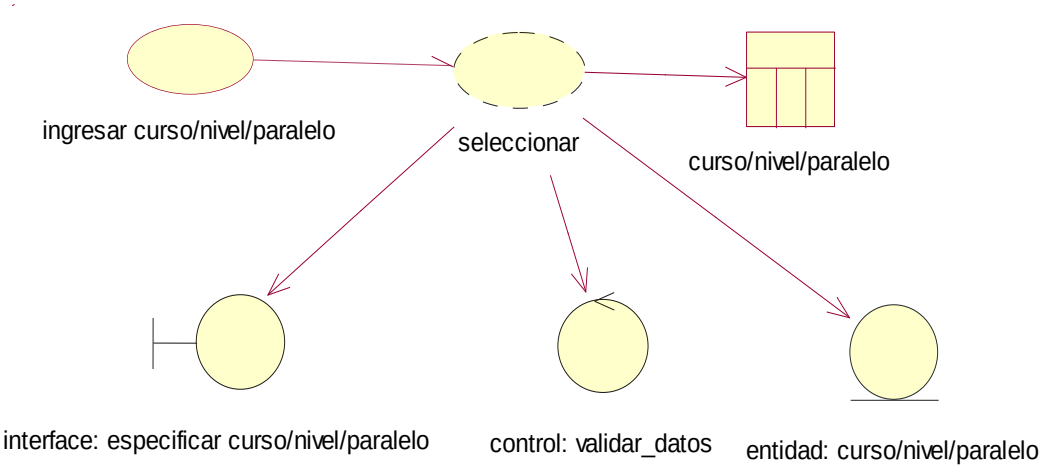


Fig 17: *Caso de uso de realización 005.*

Tabla 19

Registro *de alumno en curso/nivel/paralelo*

Nombre:	**REGISTRO DE ALUMNO EN CURSO/NIVEL/PARALELO**
Identificador:	UC005
Responsabilidades:	Generar un interface que me permita colocar a cada estudiante en su respectivo curso/nivel/paralelo.
Tipo:	SISTEMA
Referencias casos de uso	UC005
Referencias requisitos	RF005
PRECONDICIONES	
De instancia **Se necesitará una interface para que el docente y el alumno puedan trabajar de acuerdo a sus necesidades.** **Se ubicará un control para que se generen actividades acordes a la materia.** **Se necesita una entidad para ir generando al información requerida** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Implementación de imágenes con sonidos** **De relación** **No tiene**	
SALIDAS PANTALLA	
Ambiente de trabajo adecuado acorde a las necesidades de los alumnos con capacidades especiales	

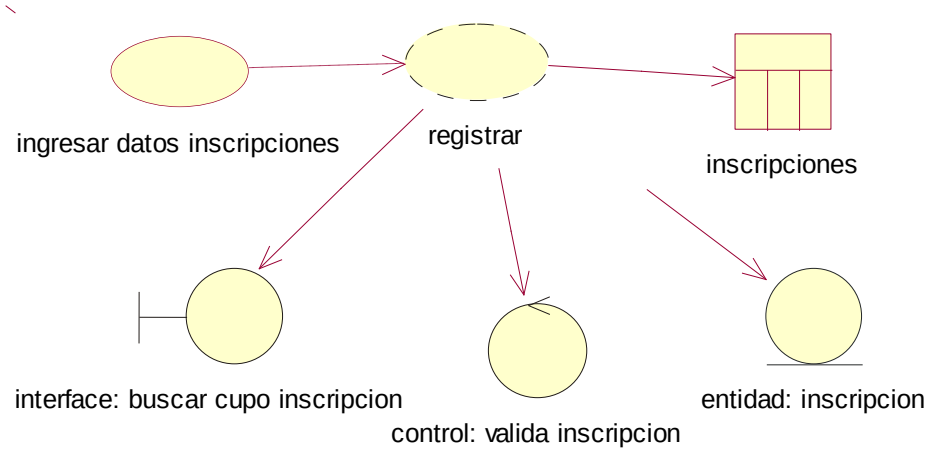


Fig 18: Caso de uso de realización 006

Tabla 20

Registro de inscripciones de alumnos

Nombre:	**REGISTRO DE INSCRIPCIONES DE ALUMNOS**
Identificador:	UC006
Responsabilidades:	Generar un interface que me permita realizar inscripciones previas a la matriculación para la toma de cupos.
Tipo:	SISTEMA
Referencias casos de uso	UC006
Referencias requisitos	RF006
PRECONDICIONES	
De instancia **Se necesitará una interface buscar cupo de inscripción del alumno.** **Se ubicará un control para que se genere la validación de la inscripción.** **Se necesita una entidad para generar la inscripción.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Permitir realizar inscripción luego de terminados los cupos con previa anticipación al personal administrativo.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Una vez realizada la inscripción el alumno ya tiene un cupo solo deberá acercarse el día de la matrícula para terminar con el proceso de ingresos.	

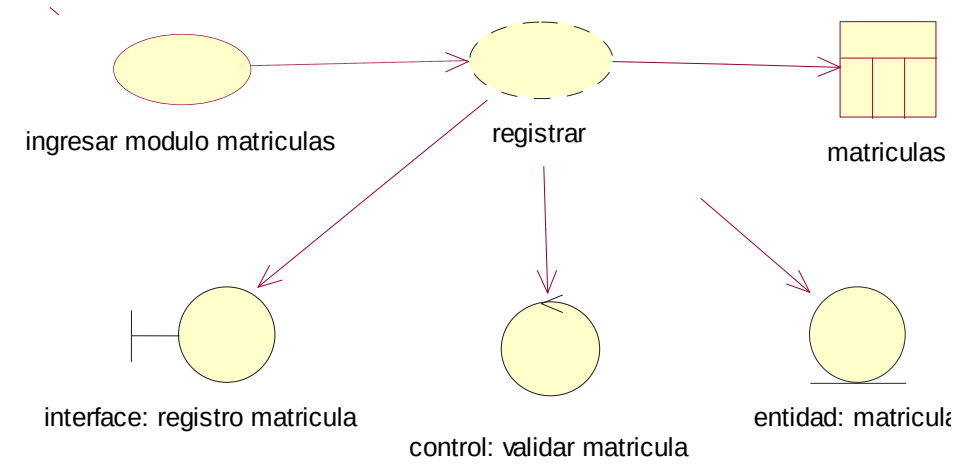


Fig 19: Caso de uso de realización 007.

Tabla 21

Matriculación de los estudiantes

Nombre:	**MATRICULACIÒN DE LOS ESTUDIANTES.**
Identificador:	UC008
Responsabilidades:	Ingresar una interface que me permita registrar a los alumnos para las matriculas
Tipo:	SISTEMA
Referencias casos de uso	UC008
Referencias requisitos	RF008
PRECONDICIONES	
De instancia **Se necesitará una interface para el registro de la matrícula.** **Se ubicará un control validar matricula.** **Se necesita una entidad para generar el reporte del alumno matriculado.** **De relación** **No tiene**	
POSCONDICIONES	
De instancia **Permitir matricular previa autorización de administrativos y revisión de cupos.** **De relación** **No tiene**	
SALIDAS PANTALLA	
Se generará la lista de alumnos matriculados en el nuevo año lectivo.	

3.05 Diagramas de secuencia del sistema

Los diagramas se secuencia van a determinar las actividades que realiza cada actor en los casos de uso.

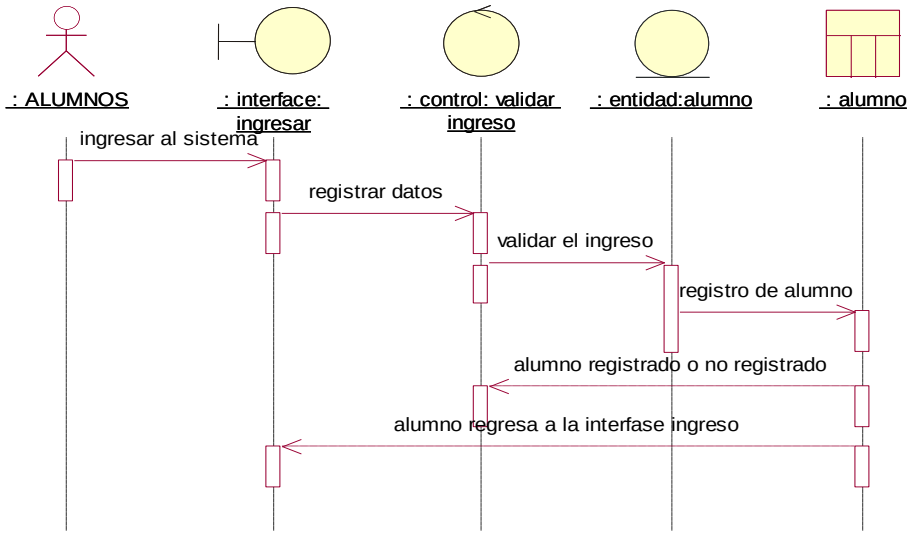


Fig 20: Diagrama de secuencia 001

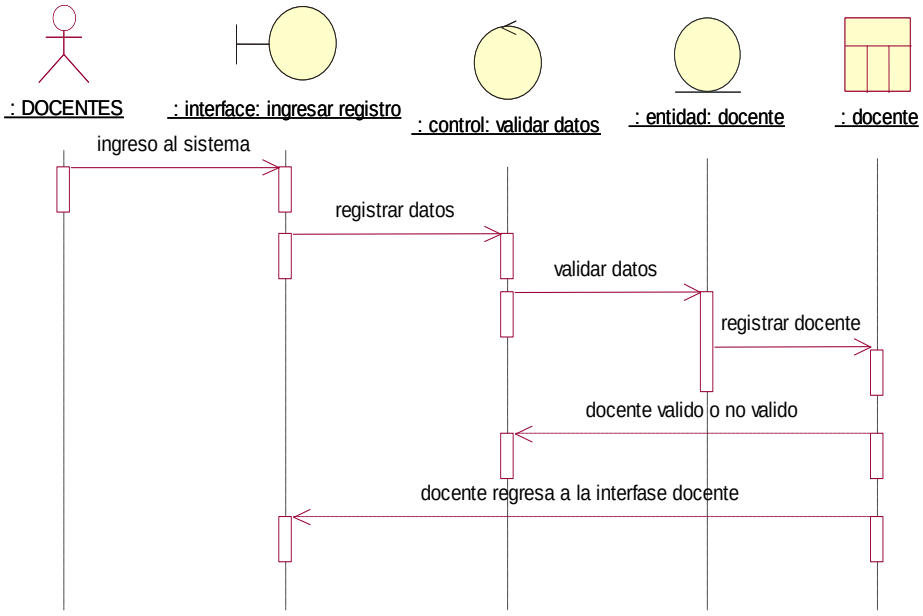


Fig 21 : Diagrama de secuencia 001

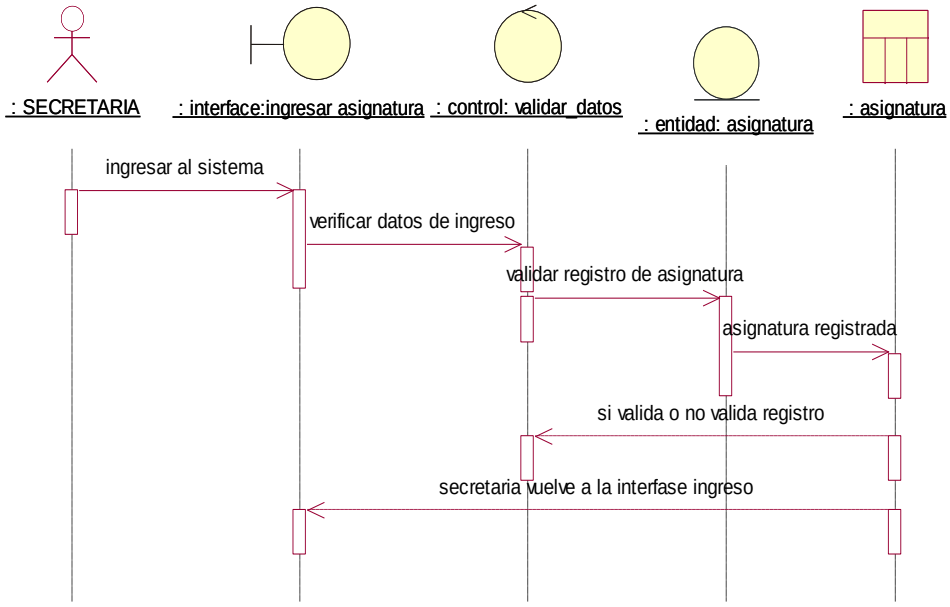


Fig 22: Diagrama de secuencia 002.

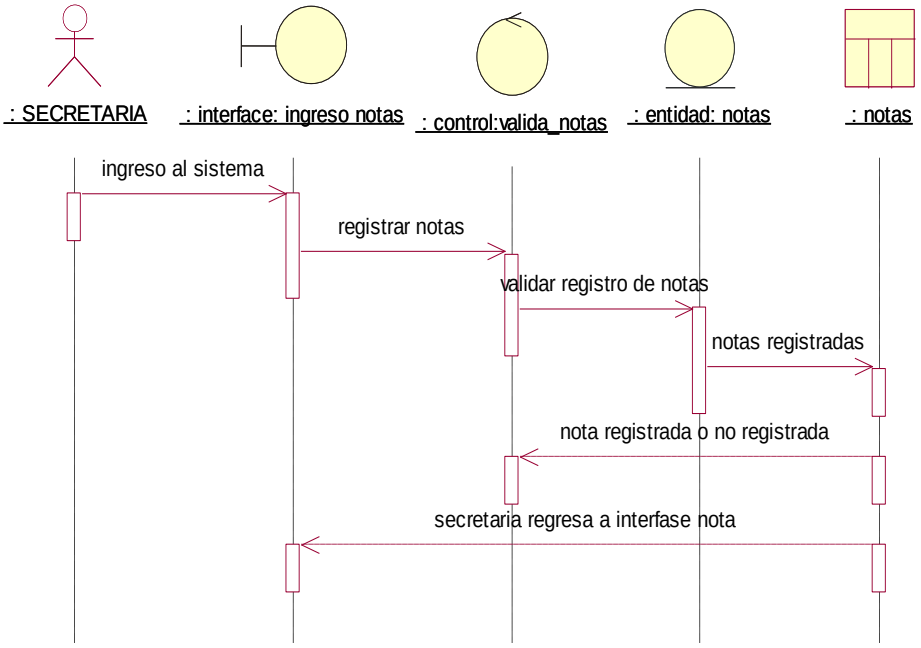


Fig 23: Diagrama de secuencia 003

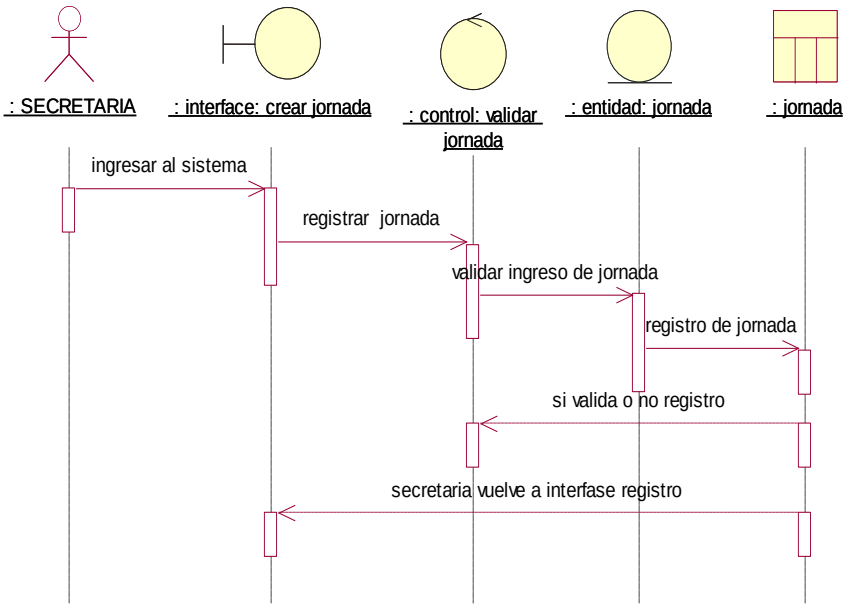


Fig 24: Diagrama de secuencia 005

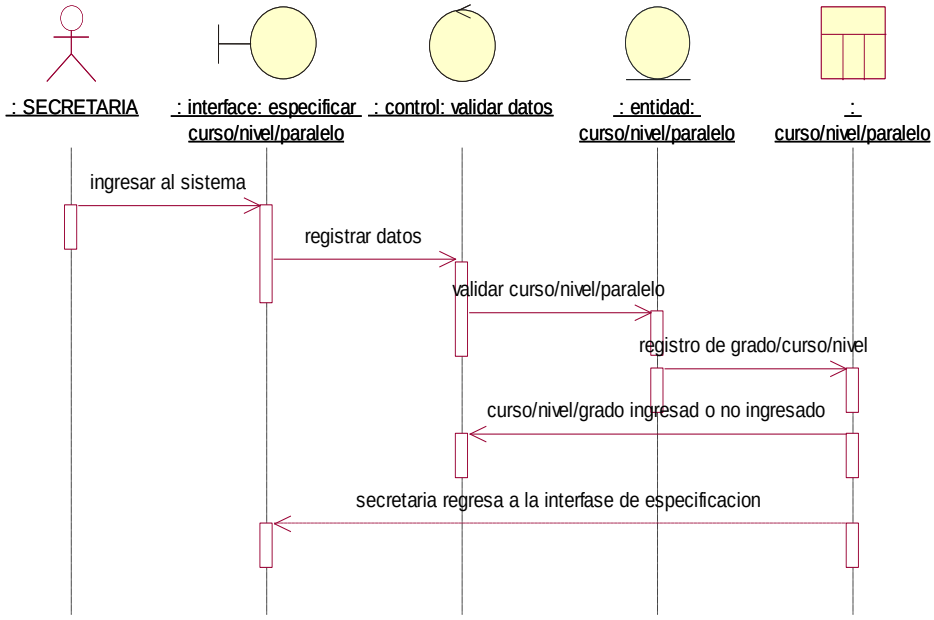


Fig 25: Diagrama de secuencia 006.

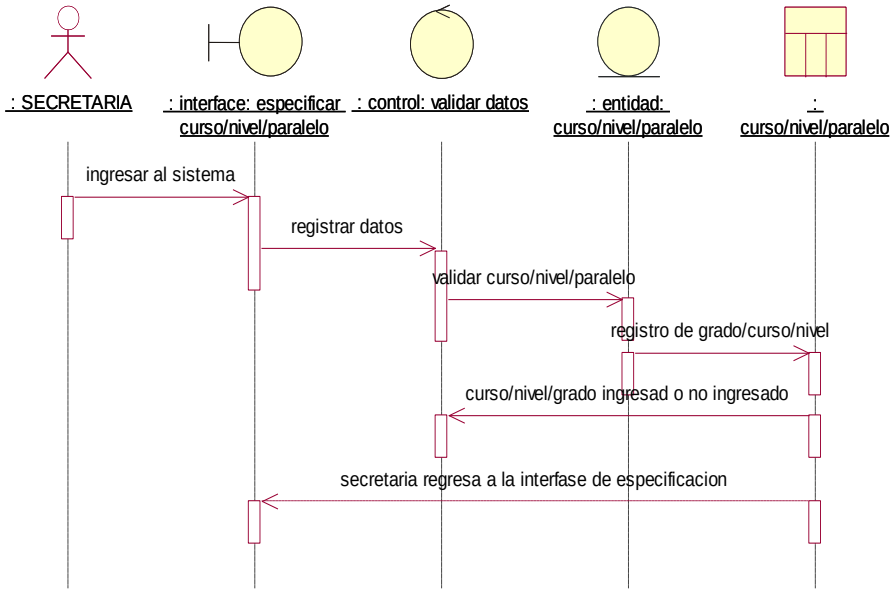


Fig 26: Diagrama de secuencia 007

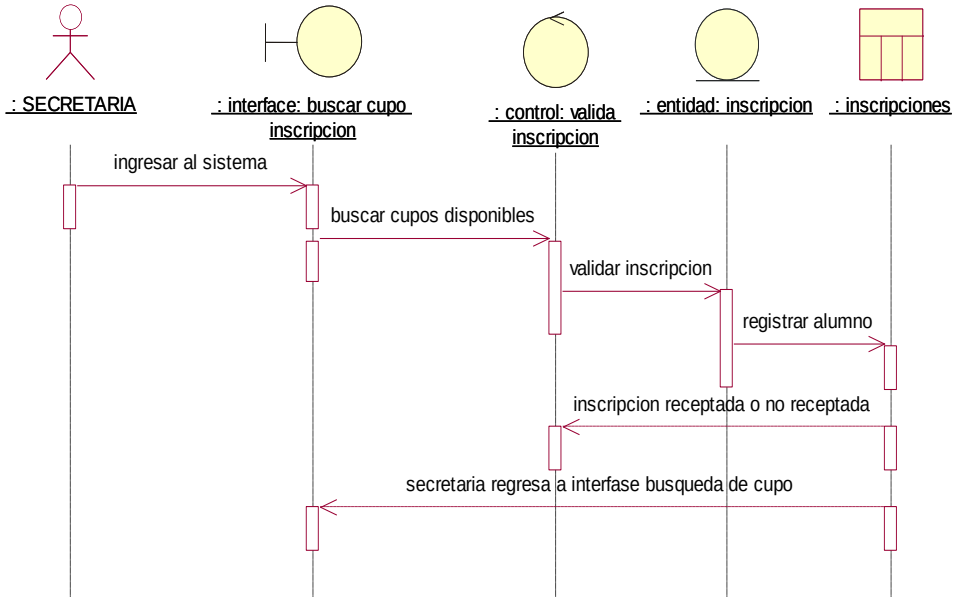


Fig 27: Diagrama de secuencia 008

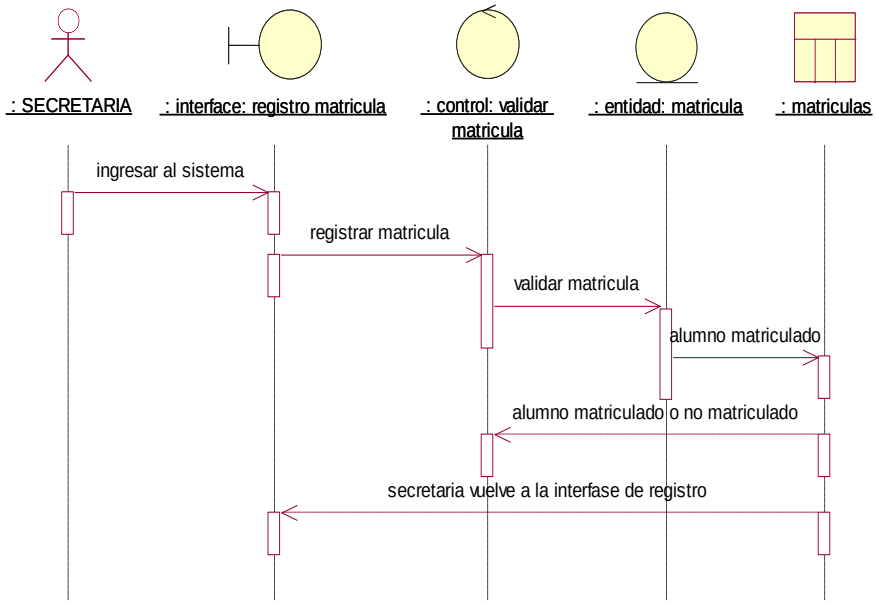


Fig 28: Diagrama de secuencia 009

3.06 Especificación de los casos de uso

Tabla 22

Caso *de uso 001*

Casos de uso	Se llevará registro de alumnos y docentes en la base de datos
Identificador	UC001
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activa cuando se procede al registro y autentificación de un usuario.	Si el usuario se encuentra registrado o no registrado el sistema nos emitirá un aviso de autentificación de registro o usuario se encuentra registrado.
2.- El usuario podrá registrar la siguiente información: -Nombres y Apellidos -Teléfonos -Domicilio -Edad	Una vez ingresado esta información se podrá seleccionar: -Nivel -Asignatura -Docente
CURSOS ALTERNATIVOS	
El usuario que no se encuentre registrado no podrá tener acceso al sistema y para eso deberá proceder a pedir su registro a la persona encargada para que pueda tener acceso al sistema.	

Tabla 23

Caso de uso 002

Casos de uso	**Los usuarios llevarán un control de asignaturas impartidas diariamente.**
Identificador	UC002
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activa cuando el usuario se encuentra ingresando las asignaturas.	El usuario después de realizar el ingreso de la asignatura podrá llevar un control diario de las mismas.
2.- El usuario podrá imprimir el registro de los estudiantes a las diferentes asignaturas. -Nombre -Alias	El usuario podrá realizar e imprimir las asignaturas una vez seleccionado el alumno:
CURSOS ALTERNATIVOS	
El usuario no podrá imprimir las asignaturas sin no está registrado en la mismas sus datos personales y la nota.	

Tabla 24

Caso de uso 003

Casos de uso	**Se llevara un control de notas diarias, mensuales de los estudiantes para así sacar los promedios quimestrales.**
Identificador	UC003
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activara cuando el docente ingrese al sistema**.**	El usuario podrá realizar el ingreso de las notas de acuerdo al nivel/curso/paralelo.
2.- El usuario podrá realizar ingreso de las notas de los estudiantes dependiendo de la asignatura	El usuario una vez haya seleccionado el alumno deberá ingresar las notas.
CURSOS ALTERNATIVOS	
El usuario no podrá alterar los registros una vez ya guardado el registro final.	

Tabla 25

Caso de uso 004

Casos de uso	**Se llevará un control de jornadas de trabajo de los docentes ya que los turnos serán acorde al área donde se encuentren.**
Identificador	UC004
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activara al momento del ingreso del usuario.	El usuario podrá seleccionar jornadas solo en el tiempo establecido del docente.
2.-El usuario registra las jornadas con el registro de asignaturas adjunto.	El usuario realizará los registros y cambios una vez al mes.
CURSOS ALTERNATIVOS	
El usuario no generara registro de jornadas si no selecciona al docente.	

Tabla 26

Caso de uso 005

Casos de uso	**La aplicación deberá contener el campo de control de cursos, niveles y paralelos para así tener conocimiento de cuantos alumnos quedaría en cada uno.**
Identificador	UC005
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activara cuando el usuario seleccione curso/nivel/paralelo.	El usuario podrá mantener registros del control de curso/nivel/paralelo.
2.-El usuario podrá seleccionar los curso/nivel/paralelo para cada niño en cada jornada.	El usuario una vez seleccionado el curso/nivel/paralelo podrá realizar las búsquedas:
CURSOS ALTERNATIVOS	
El usuario solo podrá realizar el ingreso más no la eliminación de algún curso/nivel/paralelo.	

Tabla 27

Caso de uso 006

Casos de uso	**El sistema deberá contener un campo que permitirá realizar inscripciones previas, para así poder tener la cantidad de niños acordes a los cupos acordados**
Identificador	UC006
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activará cuando el usuario ingrese al sistema.	El usuario podrá registrar a los alumnos solo inscribiéndoles con un cupo y aula asignada.
2.-El usuario podrá inscribir a los alumnos solo con el cupo registrado.	El usuario una vez ingresado las inscripciones generara un reporte de cupos sobrantes.
CURSOS ALTERNATIVOS	
El usuario solo podrá realizar el ingreso más no la eliminación de alguna inscripción una vez generado el reporte.	

Tabla 28

Caso de uso 008

Casos de uso	**Después de la inscripción se realizará la matricula que llevara a cabo la recepción de documentos de los estudiantes los cuales serán cargados en su hoja de matrícula.**
Identificador	UC008
CURSO TIPICO DE EVENTOS	
USUARIO	SISTEMA
1.-El caso de uso se activará cuando el usuario ingrese al sistema de matrículas.	El usuario realizará la matriculación de los estudiantes inscritos.
2.-El usuario podrá generar un listado de alumnos matriculados al día.	El usuario una vez ingresado al registro podrá sacar un reporte de cuantos cupos de matrículas quedan disponibles.
CURSOS ALTERNATIVOS	
El usuario no podrá ingresar más alumnos a matriculas si ya no contiene cupos disponibles.	

Capitulo IV: Análisis de alternativas

4.01 Matriz de análisis de alternativas

La matriz de alternativas permite establecer la base para determinar las distintas estrategias alternativas que podrían contribuir al cambio de la situación actual a la situación futura deseada. Esas estrategias deberán ser evaluadas a través de diversos criterios, que dependerán del problema de desarrollo.

Tabla 29

Matriz de Análisis de Alternativas

MATRIZ DE ANALISIS DE ALTERNATIVAS							
OBJETIVOS	Impacto sobre el propósito	Factibilidad Técnica	Factibilidad Financiera	Factibilidad Social	Factibilidad Política	Total	Categorías
1. Mejor manejo de las actividades realizadas por los administrativos	4	3	3	3	2	15	ALTA
2. Fomentar la minga solidaria en beneficio de la comunidad.	4	3	3	3	2	15	ALTA
3. Docentes con más herramientas de apoyo.	4	3	3	2	2	14	MEDIA ALTA
4. Alumnos físicamente integrados a un aula de trabajo.	4	2	3	2	3	14	MEDIA ALTA
5. Profesores han sido capacitados en el manejo de las tecnologías de la información.	4	3	3	2	4	16	ALTA
TOTAL	20	14	15	12	13	74	

NOTA: la matriz de alternativas nos ayuda a establecer las estrategias para llevar a cabo la situación deseada.

Categoría alta: Determina que los puntos tratados dentro de la matriz de requerimientos es de mucha necesidad para la realización de las actividades en la institución.

Categoría media: Nos indica que puntos a tratar dentro de los requerimientos son necesarios para la institución.

Categoría baja: Los puntos a tratar no son exactamente necesarios para el sistema, estos son requerimientos no funcionales.

4.02 Matriz de impacto de objetivos

Tabla 30

Matriz de impactos de Objetivos

Objetivos	**Factibilidad de Lograse**(Alta.Med-Baj) (4 - 2 - 1)	**Impacto en Género**(Alta.Med-Baj) (4 - 2 - 1)	**Impacto Ambiental**(Alta.Med-Baj)(4 - 2 - 1)	**Relevancia**(Alta.Med-Baj)(4 - 2 - 1)	**Sostenibilidad**(Alta.Med-Baj)(4 - 2 - 1)	**Total**
Aumenta la calidad de educación en la institución con los beneficios adecuados para los estudiantes.	-Los beneficios que se adquieren son mayores que los costos que producen. -Se cuenta con apoyo de la institución. -Existe tecnología adecuada y está disponible para su realización. **13 puntos**	1. Incrementa la participación del padre y los docentes en la institución educativa. 2.Incrementa el nivel Educativo de los alumnos dentro de la institución 3. Fortalece la aplicación de los derechos de la mujer. **12 puntos**	-Contribuye a proteger el entorno físico. -Mejora el entorno social. -Mejora el entorno cultural. -Protege el uso de los recursos. -Favorece la educación ambiental **20 puntos**	-Responde a las expectativas de los beneficiarios -Es una prioridad sentida por los beneficiarios. -Es una ayuda muy aceptada por los beneficiarios. -Los beneficios son deseados por los beneficiarios **16 puntos**	-Fortalece la participación de los beneficiarios y población local. -Fortalece el conocimiento de los estudiantes sobre las tecnologías y sus beneficios. -La población está en posibilidades de aportar medios **16 puntos**	77

La matriz de impactos de objetivos nos permite determinar los objetivos a alcanzar dentro del sistema de información. Con la utilización de este sistema se podrá

controlar más adecuadamente el área Administrativa de la escuela por ende tenemos que fijar los objetivos que queremos alcanzar.

4.03 Estándares para el diseño de clases

Un diagrama de clases está compuesto por los siguientes elementos:

Principales conceptos de base de datos:

Estándares para el diseño de clases

Base de datos: Conjunto relacionado entre si y que está estructurado de forma que tal que puede accederse a ellos automáticamente e independientemente de los programas que los gestionan

Tablas: Es un conjunto de información del mismo tipo. Por ejemplo, en un local de ventas de vehículos, la tabla tendrá información relativa a todos los autos que disponga en el local, en otra tabla contendrá la información de los compradores y proveedores.

Clase: Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio (una Casa, un Auto, una Cuenta Corriente, etc.).

En UML, una clase es representada por un rectángulo que posee tres divisiones:

En donde:

- **Superior**: Contiene el nombre de la Clase
- **Intermedio:** Contiene los atributos (o variables de instancia) que caracterizan a la Clase (pueden ser private, protected o public).

- **Inferior:** Contiene los métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno (dependiendo de la visibilidad: private, protected o public).

Registro: Se denomina registro a la unidad elemental de información de una tabla. En el ejemplo anterior, un registro viene hacer la información del vehículo, con su placa, año, marca,etc.

Atributo: Un atributo es una característica de una entidad. El valor especifico de un atributo, conocido como *elemento de datos*, se puede encontrar con los campos de registro que describe una entidad. Como ya se planteó, un conjunto de campos de un objeto específico representa un registro. Una clave es un campo o grupo de campos en un registro que se utiliza para identificar a este.

Clave Principal: Una clave principal es un campo (o conjunto de campos) que presenta valores únicos en una tabla. Estos valores se pueden usar para hacer referencia a registros enteros, ya que cada registro tiene un valor distinto para la clave. ***Una tabla solo puede tener una clave principal***

Clave Foránea: Una clave foránea en una base de datos relacional es una clave que se usa en una tabla secundaria y que coincide con la clave primaria en una tabla primaria relacionada. Las claves foráneas pueden tener valores duplicados (multiplicidad) en la tabla secundaria, mientras que para las claves primarias eso no es posible. El uso apropiado de claves foráneas permite exigir la integridad referencial

Método: Un **método** es una subrutina cuyo código es definido en una clase y puede pertenecer tanto a una clase, como es el caso de los **métodos de clase o estáticos**,

como a un objeto, como es el caso de los **métodos de instancia**. Un método u operación es la implementación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen es una función que le indica a las instancias de la clase que hagan algo.

Sin tipo derivado

El nombre del elemento **EntitySet** del tipo.

Columna de cada propiedad escalar, incluidas todas las propiedades escalares de las propiedades de tipo complejo.

Columna o columnas que corresponden a la propiedad o propiedades de la clave de entidad.

Tipo derivado

Concatenación del nombre de elemento de **EntitySet** de tipo base y del nombre de tipo.

Es decir, empezamos con la abreviatura de la tabla y a continuación con su nombre, con letra en mayúscula y en Singular

Una columna para cada propiedad escalar no heredada (incluidas todas las propiedades escalares de propiedades de tipo complejo) y otra para cada propiedad clave heredada.

Columna o columnas que corresponden a la propiedad o propiedades de la clave de entidad heredadas.

La clave principal debe ser única en cada tabla y en cada clase que se vaya creando

La clave principal de la tabla secundaria también es una clave externa que hace referencia a la clave principal de su tabla primaria.

Se pueden crear claves externas adicionales. Para obtener más información, vea la sección "Asociaciones y claves externas" más adelante en este tema.

Asociaciones y claves externas

En la siguiente tabla se describen como se realizan las asociaciones. Tenga en cuenta que se crea una restricción de clave externa para todas las asociaciones.

Asociaciones y claves externas

Uno a cero o uno (1:0..1) – O bien – Uno a varios (1:*)

Si no se define ninguna restricción referencial en la asociación, se agregan columnas a la tabla que corresponden al tipo de entidad 0..1 o * extremo de la asociación. Las columnas agregadas tienen restricciones de clave externa que hacen referencia a la clave principal de la tabla que corresponde al tipo de entidad en el otro extremo de la asociación. Las columnas de tabla agregadas se asignan a la asociación, no al tipo de entidad. El nombre de cada columna agregada es la concatenación del nombre de propiedad de navegación y del nombre de propiedad de clave.

Uno a uno (1:1)

Si no se define ninguna restricción referencial en la asociación, se agregan columnas a una de las tablas que corresponden a los tipos de entidad de los extremos de la asociación. La tabla a la que se agregan las columnas se elige arbitrariamente. Las columnas agregadas tienen restricciones de clave externa que hacen referencia a la

clave principal de la tabla que corresponde al tipo de entidad en el otro extremo de la asociación. Las columnas agregadas se asignan a la asociación, no al tipo de entidad. El nombre de cada columna agregada es la concatenación del nombre de propiedad de navegación y del nombre de propiedad de clave.

Varios a varios (*:*)

Se crea una tabla combinada o tabla de rompimiento como comúnmente se lo conoce. Por cada propiedad de clave de cada tipo de entidad, se agrega una columna a la tabla. Las columnas tienen restricciones de clave externa que hacen referencia a las claves principales de las tablas creadas basándose en los tipos de entidad en los extremos de la asociación. La clave primaria de la tabla creada será una clave principal compuesta formada por todas las columnas de la tabla.

4.04 Diagrama de clases

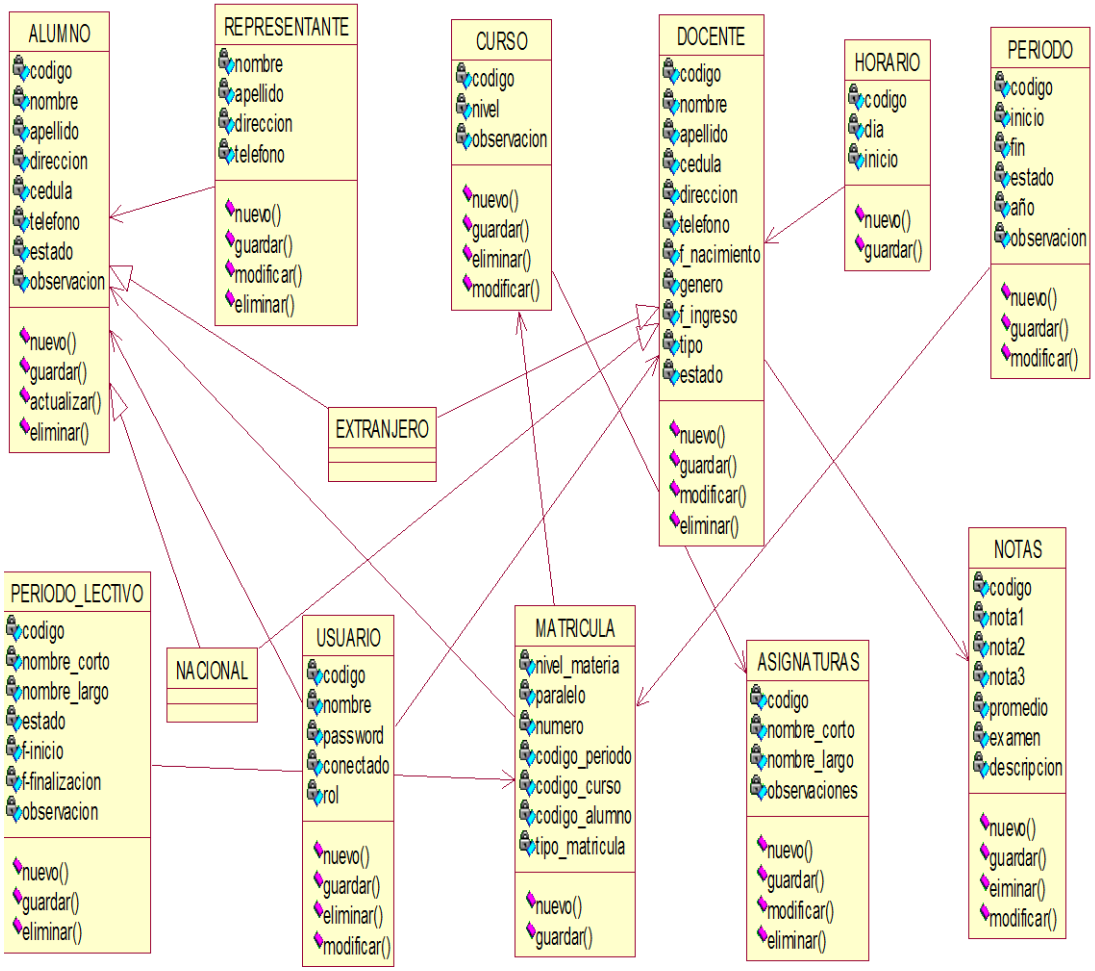


Fig 29: Diagrama de clase

Es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, orientados a objetos.

4.05 Diagrama lógico - físico

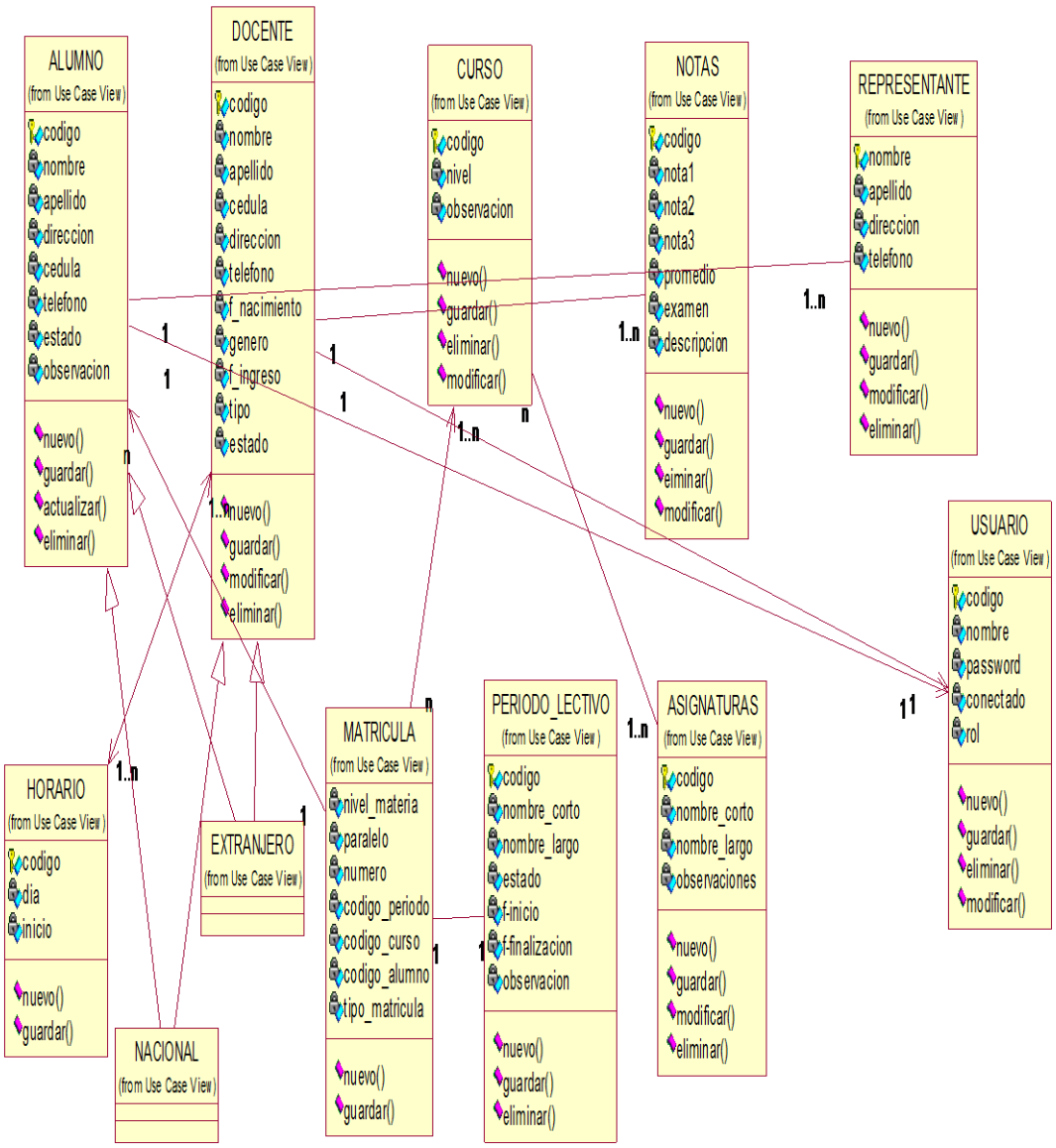


Fig 30: Diagrama Lógico- Físico

4.06 Diagrama de componentes

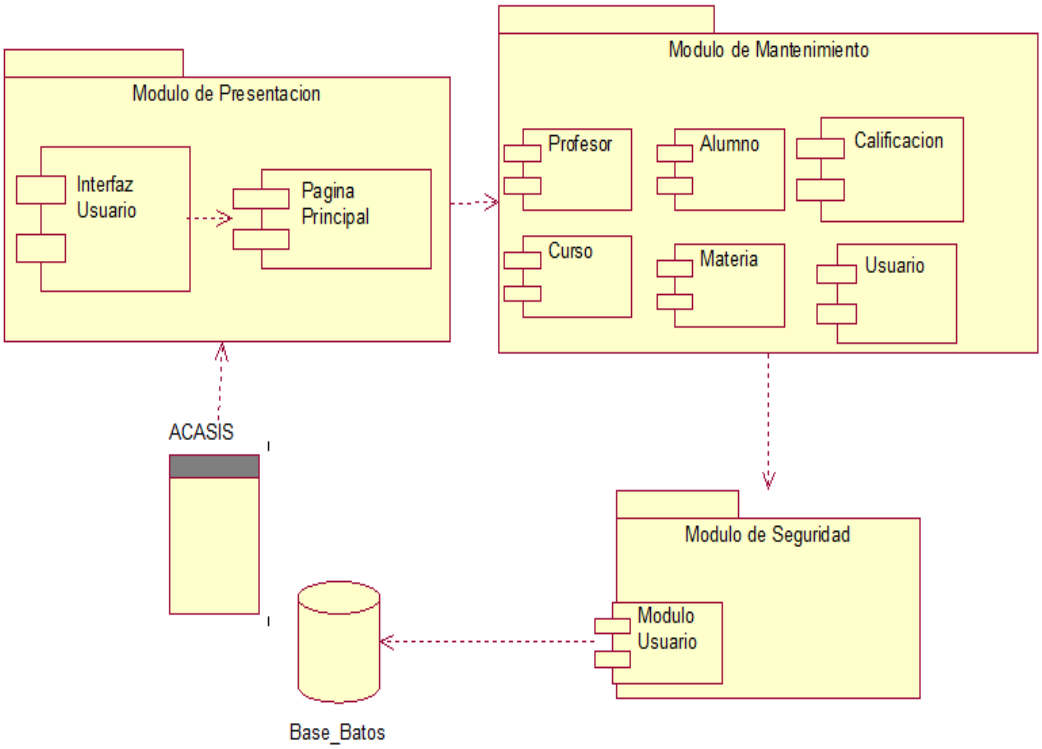


Fig 31: Diagrama de Componentes

Representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes

4.07 Diagrama de Estrategias

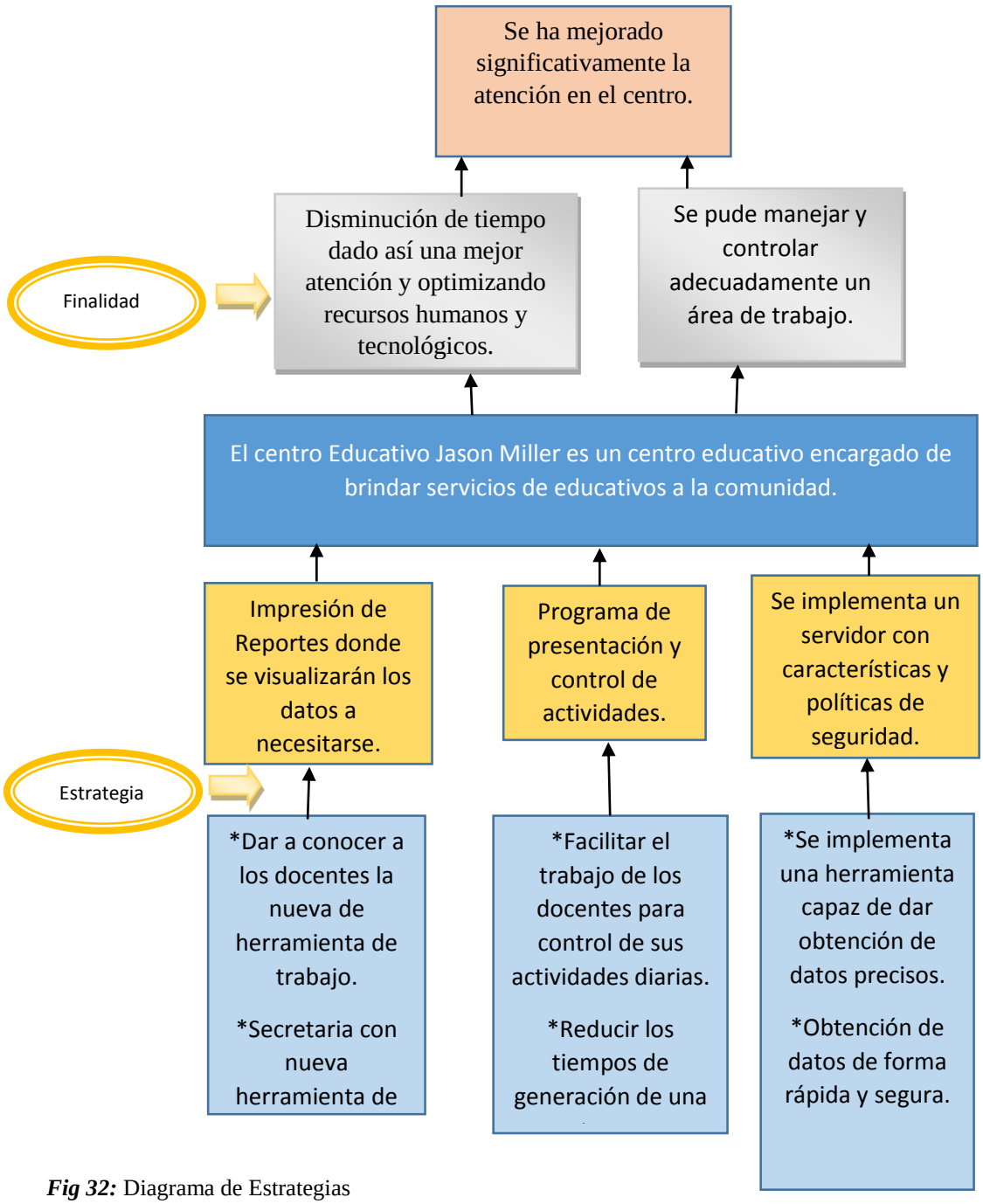


Fig 32: Diagrama de Estrategias

El diagrama de estrategias tiene como finalidad definir la estructura y alcance de las estrategias con los objetivos que en forma vinculada pueden determinar los límites específicos para el proyecto.

4.08 Matriz del marco lógico

La matriz del marco lógico nos permite saber en síntesis que es lo que se desea lograr en el proyecto, como se alcanzara el propósito trazado y sus componentes, como se mide el éxito de los objetivos, resultados y que recursos se necesitan para la ejecución del proyecto.

Tabla 31

Matriz de Marco Lógico

Resumen Narrativo de los objetivos	Indicadores	Medios de verificación	Supuestos
FIN Se incrementa el uso de tecnología para para el desarrollo de tareas de los docentes.	El número de niños beneficiados serán aún mayor. Se disminuye la cantidad de material para la toma de notas.	Software implementado en la empresa para soporte de las actividades académicas Resultados de encuestas a los docentes y padres de familia.	Es un sistema rápido y eficiente.
PROPOSITO EL sistema implementado es confiable y mejora la eficiencia de docentes en cuanto mejora de sus tareas.	El efecto logrado al final del proyecto será aceptable para los usuarios. Los docentes disminuyen el escás de tiempo para realizar las tareas asignadas con el apoyo de las nuevas tecnologías.	-El informe consolidado de gestión del proyecto. - Encuestas realizadas a los docentes y usuarios.	Continuidad del servidor actualización y ejecución simultanea del mismo
COMPONENTES Solución en sistemas académico y que da respuesta, a una necesidad relacionada con los **b**eneficios.	Mejora la eficiencia y eficacia de los docentes al momento de realizar una actividad.	Los docentes estarían registrados y esto permitirá generar actividades aleatorias para un mejor servicio de las actividades y reducir los tiempos.	Sistema que reduce el tiempo en ejecución de actividades académicas.
Actividades del proyecto: 1.1 Capacitar personal. 1.2 Mejorar la herramienta de trabajo. 1.3 Mejorar el ambiente social laboral. 2.1 Registrar todos los alumnos y docentes.	**Presupuesto:** La Escuela cuenta con los equipos necesarios y con los requerimientos suficientes para realizar la implementación del sistema.		- El departamento de administración entregara todo los componentes que facilitaran las actividades para coordinar las clases en el nuevo sistema.

4.09 Vistas arquitectónicas

4.01.01 Vista lógica

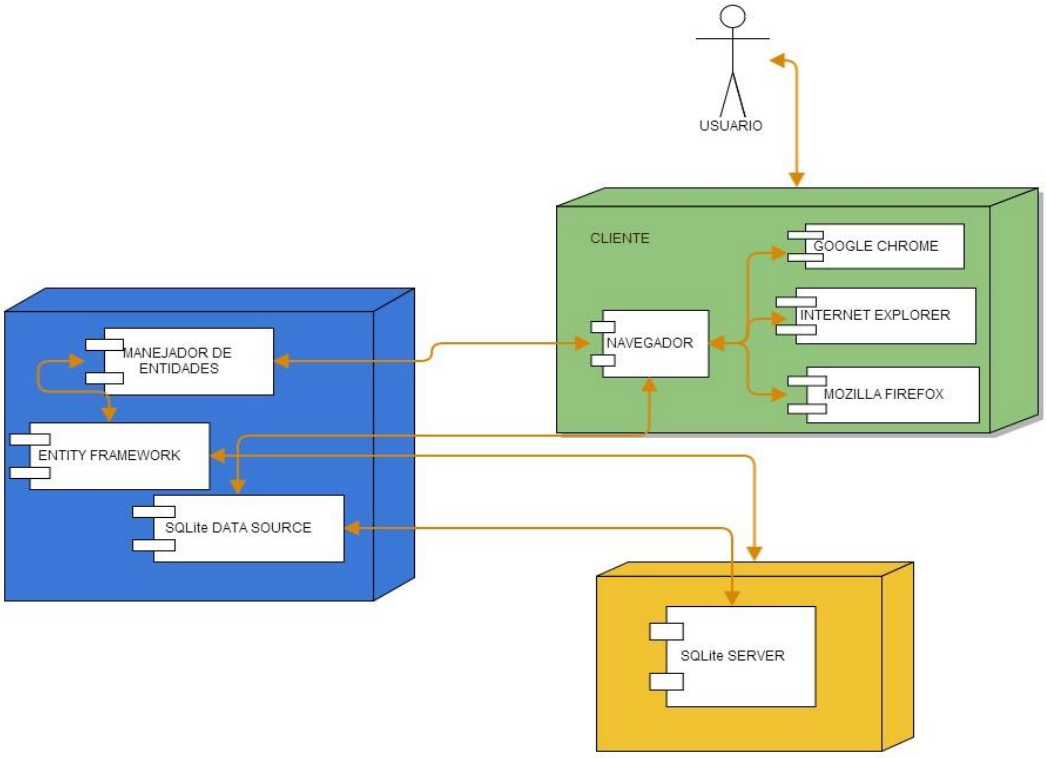


Fig 33: Vista Lógica

4.01.02 Vista física

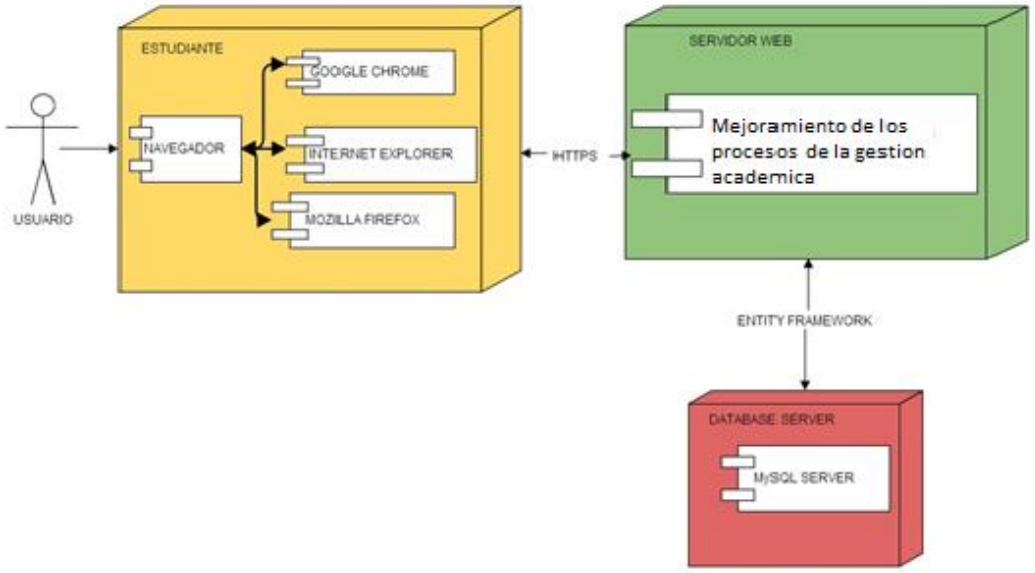


Fig 34: Vista Física

4.01.03 Vista de desarrollo

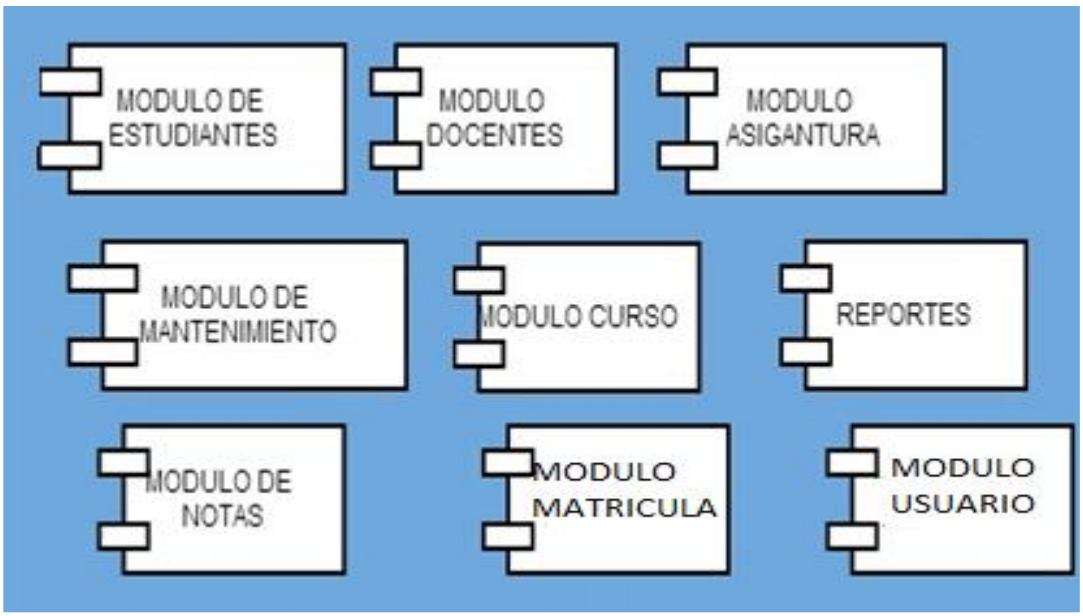


Fig 35: Vista de Desarrollo

4.01.04 Vista de procesos

Se tratan los aspectos de concurrencia y distribución, integridad del sistema, y tolerancia a fallos.

Se especifica en cuál hilo de control se ejecuta efectivamente una operación de una clase identificada en la vista lógica.

Puede ser descrita como un conjunto de redes lógicas de procesos que son ejecutados de forma independiente, y distribuidos a lo largo de varios recursos de hardware conectados mediante un bus o a una red de datos.

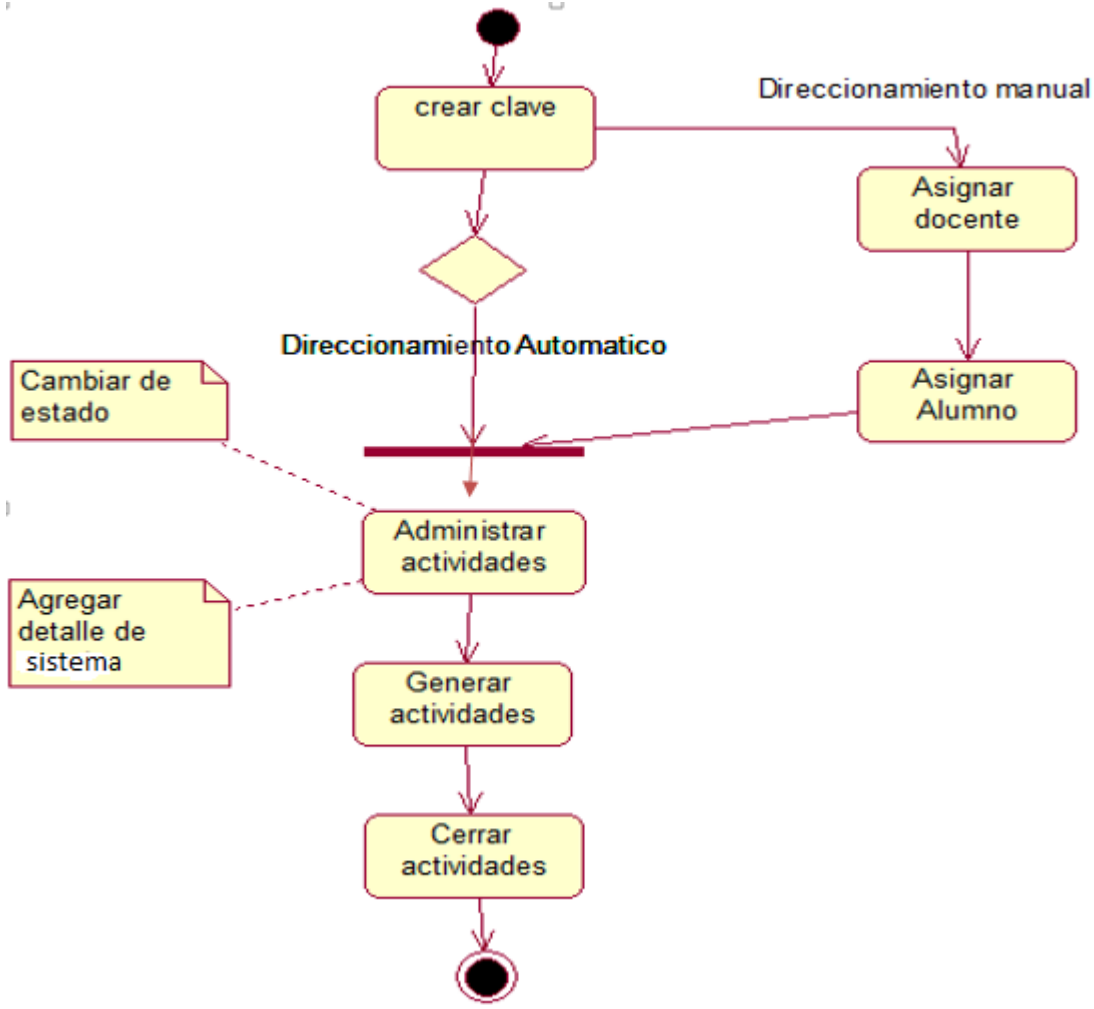


Fig 36: Vista de Procesos

Capítulo V: Propuesta

5.01. Especificación de estándares de programación

Estos estándares deben considerarse como guías en las etapas de diseño de los sistemas.

Las técnicas efectivas de manejo y control de proyectos combinados con una participación activa de los usuarios y la utilización de metodologías estructuradas de desarrollo de sistemas, pueden minimizar riesgos de incumplimiento de fechas de actividades importantes, de gastos excesivos en relación a los costos estimados e insatisfacciones de los usuarios de los sistemas.
Reglamentar la forma en que se implementará el código fuente del proyecto, pasando, por las variable, controles, ficheros, archivos y todo aquello que esté implicado en el código.

Mejorar y uniformizar a través de las reglas que se proponen, el estilo de programación que tiene cada programador.

- Los nombres de variables serán monemotécnicos con lo que se podrá saber el tipo de dato de cada variable con sólo ver el nombre de la variable.
- Los nombres de variables serán sugestivos, de tal forma que se podrá saber el uso y finalidad de dicha variable o función fácilmente con solo ver el nombre de la variable.
- La decisión de poner un nombre a una variable o función será mecánica y automática, puesto que seguirá las reglas definidas por nuestro estándar.
- Permite el uso de herramientas automáticas de verificación de nomenclaturas.

Define la estructura y la organización del código fuente de un programa. Además el seguir un estándar de programación, facilita al momento de su modificación del código y también, ayuda al programador para realizar una modificación de código aunque no esté trabajando en su propio equipo.(GUZMAN VALDEZ, 2005)

A continuación se realizará la representación de los estándares para los controles:

Estándares de programación

Tipo de control Prefijo Ejemplo

Label Lbl Lbl_Ejemplo

Descripción: Representa una etiqueta estándar de Windows.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto.

Button Btn Btn_Ejemplo

Obtiene la distancia, en píxeles, que existe entre el borde inferior del control y el borde superior del área cliente de su contenedor.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

DropDownList Cmb Cmb_Ejemplo

Se utiliza para mostrar datos en un cuadro combinado desplegable.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Image Img Img_Ejemplo

Obtiene o establece la imagen que se muestra en un control Label.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Tipo de control

Prefijo **Ejemplo** **Descripción**

ImageButton Imb Imb_Ejemplo

Permite al usuario hacer clic en él para ejecutar una acción

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

LinkButton Lnk Lnk_Ejemplo

Muestra un control de botón de tipo hipervínculo en una página Web.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Panel Pnl Pnl_Ejemplo

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

RadioButton	Rbn	Rbn_Ejemplo

Permite al usuario seleccionar una única opción de un grupo de opciones cuando está emparejado con otros controles de RadioButton

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Table	Tbl	Tbl_Ejemplo

Tabla a otra, puede copiar sólo la definición de la columna o copiar la definición y los datos.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

TextBox	Txt	Txt_Ejemplo

El control TextBox o Caja de texto se utiliza para Ingresar y/o visualizar Texto.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

GridView	Gdv	Gdv_Ejemplo

Proporciona una forma eficaz y flexible de mostrar datos en formato de tabla.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Formulario	Frm	Frm_Ejemplo

Proporciona una forma eficaz y flexible de mostrar datos en formato de tabla.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

CrystalReports	Rpt	Rpt_Ejemplo

Crear contenido interactivo con calidad de presentación al entorno de Windows.

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

Clase	Cls	Cls_Ejemplo

Un objeto puede ser una porción de una aplicación, como un control o un formulario

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto

CheckBox	Chk	Chk_Ejemplo

Se utiliza habitualmente para presentar al usuario una selección de tipo Sí/No o Verdadero/Falso

La primera letra inicia con mayúscula y las restantes serán la abreviatura del objeto.

Arquitectura de Desarrollo

Para el desarrollo de las nuevas aplicaciones de desarrollo se utilizará una arquitectura de 3 capas.

Capa Presentación: Provee la interfaz del usuario. Las tecnologías que se utilizan en .Net son winforms para aplicaciones de cliente inteligente y ASP.Net para aplicaciones web.

Capa Lógica de Negocios: implementa la funcionalidad de la lógica de negocio. Esta capa está compuesta por componentes desarrollados en cualquier lenguaje de .NET. Generalmente se utiliza COM+ (Enterprise Services) para brindar el soporte transaccional, escalabilidad, etc.

Capa Acceso a Datos: provee el acceso a los repositorios de información persistente y/o sistemas externos. La tecnología utilizada para desarrollar esta capa es ADO.NET, es común utilizar storedprocedure y XML.
(Estandares de Programaciòn)

5.02 Diseño de interfaces de usuario

Ingreso al sistema

Numeración	Representación	Descripción
1	TextBox	Nombre de la institución
2	TextBox	Usuario
3	TextBox	Contraseña
4	Button	Inicio de Sesión

Fig 37: Ingreso al sistema

Pantalla de Bienvenida

Numeración	Representación	Descripción
A	TextBox	Nombre de la institución
B	TextBox	Menú a visualizarse
3	TextBox	Breve reseña de la Institución

Fig 38: Pantalla de Bienvenida

Inscripción del Alumno

Numeración	Representación	Descripción
A	TextBox	Cedula del Alumno
B	TextBox	Nombre del Alumno
C	TextBox	Apellido del Alumno
D	DropDownList	Fecha de Nacimiento del Alumno
E	DropDownList	Nacionalidad del Alumno
F	DropDownList	Genero del Alumno
G	TextBox	Nombre del Padre
H	TextBox	Apellido del Padre
I	DropDownList	Nivel de Educación Padre
J	DropDownList	Ocupación del Padre
K	DropDownList	Área
L	TextBox	Teléfono del Padre
M	TextBox	Celular del Padre
N	TextBox	Dirección del Padre
O	TextBox	Nombre de la Madre
P	TextBox	Apellido de la Madre
Q	DropDownList	Nivel de educación de la Madre
R	DropDownList	Ocupación de la Madre
S	DropDownList	Área de la Madre
T	TextBox	Teléfono de la Madre
U	TextBox	Celular de la Madre
V	TextBox	Dirección de la Madre
W	ImageButton	Nuevo Alumno

X	ImageButton	Guardar Alumno
Y	ImageButton	Cancelar Inscripción

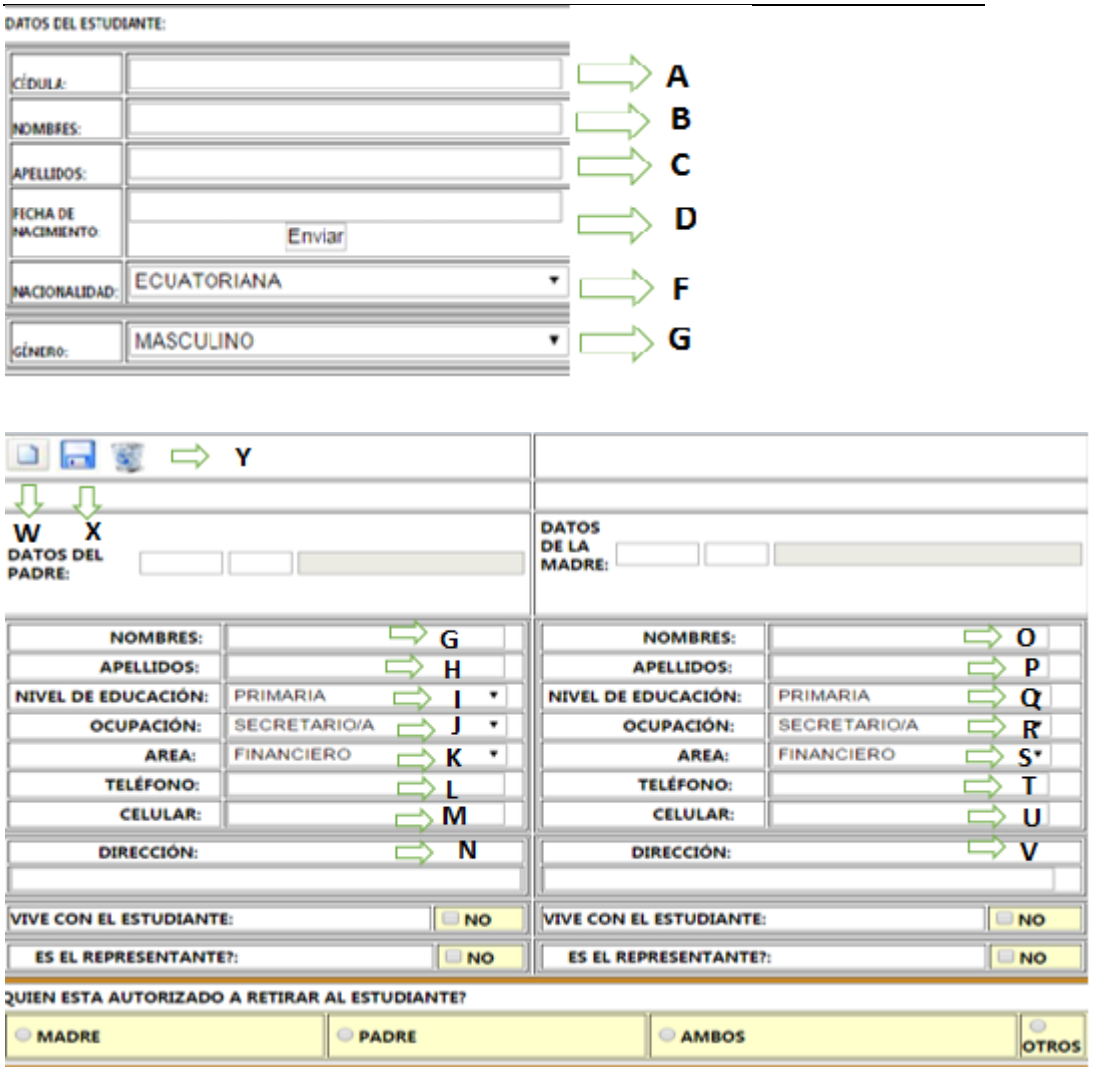


Fig 39*: *Inscripción del Alumno

Matrícula del Alumno

Numeración	Representación	Descripción
A	Label	Nombre del formulario matrícula
B	DropDownList	Tipo de Matrícula
C	TextBox	Número de la Matrícula
D	TextBox	Nombre del Alumno
E	TextBox	Curso
F	TextBox	Representante

G	TextBox	Parentesco
H	Button	Nueva Matrícula
I	Button	Matricular Alumno
J	Button	Cancelar Matrícula
K	Button	Salir, regresa al menú Principal

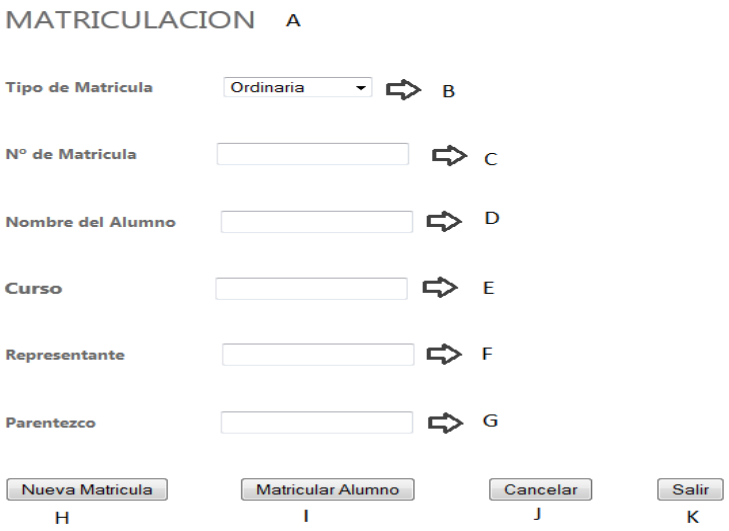


Fig 40: Matrícula del alumno

Registro Nuevo Docente

Numeración	Representación	Descripción
A	DropDownList	Todos
B	RadioButton	Opciones de Búsqueda Docente
C	TextBox	Cédula del Docente
D	TextBox	Nombre del Docente
E	TextBox	Apellido del Docente
F	TextBox	Dirección del Docente
G	CheckBox	Habilitado Docente
H	TextBox	Teléfono del Docente
I	TextBox	Celular del Docente
J	TextBox	E-mail del Docente
K	TextBox	Fecha de Nacimiento del Docente
L	CheckBox	Conectado
M	ImageButton	Nuevo Docente
N	ImageButton	Guardar Docente

O	ImageButton	Eliminar Docente Registro
P	CheckBox	Generar automáticamente usuario

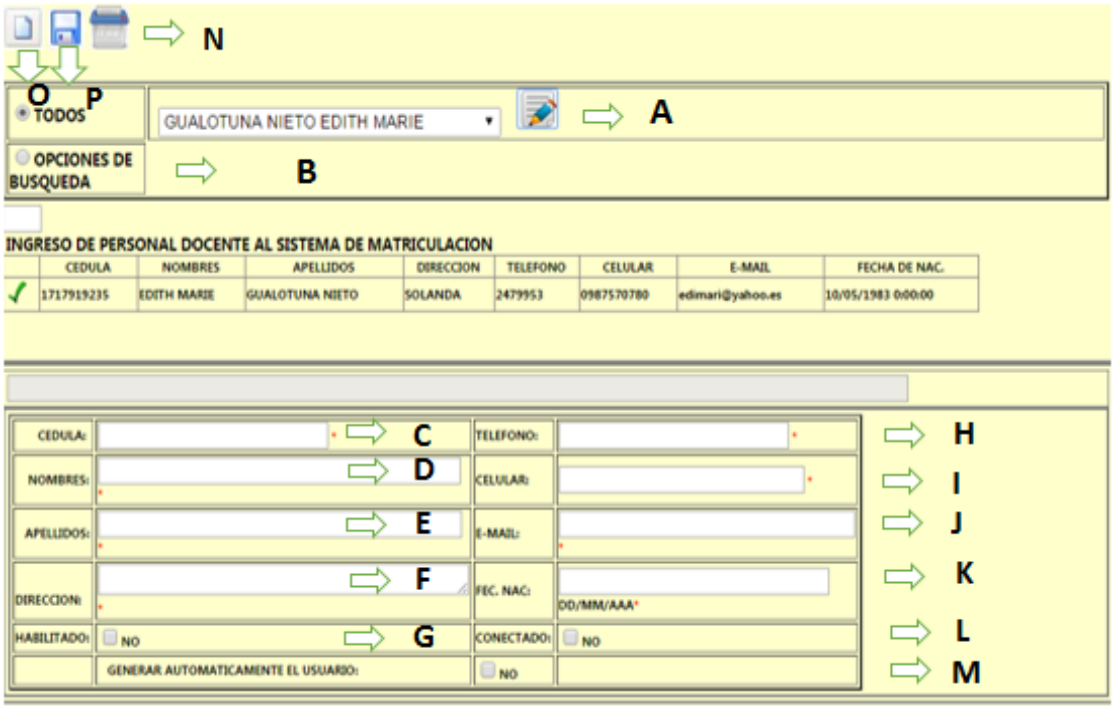


Fig 41: Registro del Docente

REGISTRO AÑO LECTIVO

Numeración	Representación	Descripción
A	CheckBox	Todo
B	CheckBox	Opciones
C	TextBox	Código Año Lectivo
D	TextBox	Nombre Largo del año Lectivo
E	TextBox	Nombre corto del año Lectivo
F	CheckBox	Estado año lectivo
G	TextBox	Fecha de Inicio
H	TextBox	Fecha de Culminación
I	CheckBox	Estado Año lectivo
J	ImageButton	Nuevo Año lectivo
K	ImageButton	Guardar Año lectivo
L	ImageButton	Eliminar Año lectivo

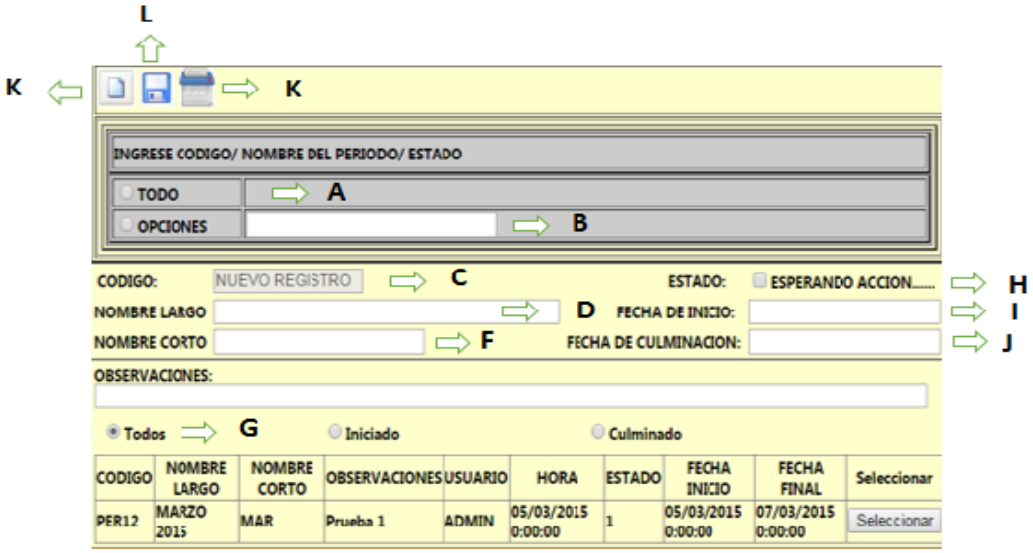


Fig 42: Registro de Año lectivo

5.03 Especificación de pruebas de unidad

Consiste en contrastar las respuestas de una implementación del software a series de datos de prueba y examinar las respuestas del software y su comportamiento operacional, para comprobar que se desempeñe conforme a lo requerido. Llevar a cabo las pruebas es una técnica dinámica de la verificación y validación ya que requiere disponer de un prototipo ejecutable del sistema.(DRAKE, 2011)

Tabla 32

Especificación de pruebas de unidad 001

Ingreso al sistema

Identificador de la prueba	EPU001
Método a probar	Validación de los campos a llenar al momento de ingreso al sistema
Objetivo de la prueba	Validar los campos de usuario y contraseña para no tener problemas con claves de usuarios.
Datos de entrada	Nombre de usuario y contraseña
Resultado esperado	Verificar que el usuario y la contraseña ingresados sea el correcto caso contrario el sistema genere error automáticamente.
Comentarios	El sistema permite verificar que al no llenar los campos de usuario y contraseña que son obligatorios no se permite ingreso alguno al sistema.

Tabla 33

Especificación de pruebas de unidad 002

Encriptación de contraseña

Identificador de la prueba	**EPU002**
Método a probar	Encriptación de la contraseña
Objetivo de la prueba	Encriptar la contraseña del usuario al momento de ingresar al sistema
Datos de entrada	**Contraseña**
Resultado esperado	**Contraseña Encriptada**
Comentarios	**Se encripto correctamente la contraseña del usuario.**

Tabla 34

Especificación de pruebas de unidad 003

Validación de cédula del estudiante

Identificador de la prueba	**EPU003**
Método a probar	Validación campo de ingreso de cédula estudiante.
Objetivo de la prueba	Ingresar cédula
Datos de entrada	**Ingresar cédula**
Resultado esperado	**Cédula ingresada correctamente**
Comentarios	**Requerimiento validado correctamente.**

Tabla 35

Especificación de pruebas de unidad 004

Validación de la edad de los estudiantes

Identificador de la prueba	**EPU004**
Método a probar	Validación de la edad
Objetivo de la prueba	Ingresar correctamente la edad del estudiante máximo hasta 13 años.
Datos de entrada	**Edad Correcta**
Resultado esperado	**Verificar si la edad ingresado se lo hace de la manera correcta**
Comentarios	**Sin Comentarios**

Tabla 36

Especificación de pruebas de unidad 005

Validación fecha de nacimiento

Identificador de la prueba	EPU005
Método a probar	Validación de las fechas de Nacimiento
Objetivo de la prueba	Ingresar correctamente la fecha de nacimiento
Datos de entrada	Fecha de Nacimiento
Resultado esperado	Verificar si la fecha de nacimiento ingresada es la correcta.
Comentarios	Sin Comentarios

Tabla 37

Especificación de pruebas de unidad 006

Validación inicio del período de año lectivo

Identificador de la prueba	EPU006
Método a probar	Validación de la fecha de Inicio de Período
Objetivo de la prueba	Ingresar fecha de inicio del período lectivo
Datos de entrada	
Fecha de inicio del período	
Resultado esperado	
Verificación del inicio del año lectivo	
Comentarios	
Fecha de inicio de año lectivo ingresada correctamente.	

Tabla 38

Especificación de pruebas de unidad 007

Finalización del año lectivo

Identificador de la prueba	EPU007
Método a probar	Ingresar fecha de finalización del período escolar
Objetivo de la prueba	Ingresar fecha de finalización
Datos de entrada	Fecha de finalización del año lectivo
Resultado esperado	Verificación fecha de finalización
Comentarios	No guarda y no valida la fecha de finalización ingresada

Tabla 39

Especificaciones de pruebas de unidad

Generar registro de alumno matriculado

Identificador de la prueba	**EPU008**
Método a probar	Crear el número de matrícula
Objetivo de la prueba	Generar el número de matrícula
Datos de entrada	**Ninguno, se genera de manera automática**
Resultado esperado	**El número de la matrícula se genera de manera automática y secuencial**
Comentarios	**No guarda registro de alumno matriculado**

Tabla 40

Especificación de pruebas de unidad

Ingreso correcto de notas

Identificador de la prueba	**EPU009**
Método a probar	Ingreso de las calificaciones del Alumno
Objetivo de la prueba	Verificar el ingres correcta de las calificaciones
Datos de entrada	**Calificaciones**
Resultado esperado	**Las calificaciones se lo hace de acuerdo al formato**
Comentarios	**Sin Comentarios**

Tabla 41

Generar el número de matricula

Identificador de la prueba	**EPU011**
Método a probar	Crear el número de matrícula
Objetivo de la prueba	Generar el número de matrícula
Datos de entrada	**Ninguno, se genera de manera automática**
Resultado esperado	**El número de la matrícula se genera de manera automática y secuencial**
Comentarios	**Sin Comentarios**

Tabla 42

Ingreso de usuarios del sistema

Identificador de la prueba	**EPU010**
Método a probar	Verificar nombre de usuario y contraseña.
Objetivo de la prueba	Habilitar a usuario ingresado con un rol específico.
Datos de entrada	**Nombre de usuario.** **Contraseña.**
Resultado esperado	**Usuario ingresado**
Comentarios	**Ingreso correcto del usuario al sistema.**

5.04 Especificaciones de pruebas de aceptación

Tabla 43

Ingreso de Alumnos a la base de datos.

Identificador de la prueba	**EPA001**
Caso de uso	CU001
Tipo de usuario	Administrador
Objetivo de la prueba	Registrar los alumnos correctamente
Secuencia de eventos	**Registro, validación, ingreso**
Resultados esperados	**Registro de alumnos en la base de datos.**
Comentarios	**Ninguno**

Estado: Aceptado

Tabla 44

Ingreso de asignaturas

Identificador de la prueba	**EPA002**
Caso de uso	CU003
Tipo de usuario	Administrador
Objetivo de la prueba	Verificar el ingreso correcto de las asignaturas designadas en la base de datos.
Secuencia de eventos	**Registro, validación, ingreso , guardado**
Resultados esperados	**Guardado de las asignaturas de los alumnos en el sistema**
Comentarios	**El sistema negará el ingreso si ya existe esa asignatura en el sistema**

Estado: Aceptado.

Tabla 45

Ingreso de Docentes al Sistema

Identificador de la prueba	EPA003
Caso de uso	CU001
Tipo de usuario	Administrador
Objetivo de la prueba	Verificar el ingreso correcto de los docentes en la base de datos.
Secuencia de eventos	Registro, validación, ingreso , guardado
Resultados esperados	Ingreso de los Docentes en el sistema.
Comentarios	El sistema negará el ingreso si ya existe el docente en el sistema.
Estado: Aceptado.	

5.05 Especificación de prueba de carga

El objetivo de esta prueba es determinar el "throughput" (volumen de trabajo o de información neto) necesario para que el sistema funcione en hora punta.(Prueba de carga, 2015)

Tabla 46

Especificación de prueba de carga 001

Pruebas de carga 001	
Identificado de la Prueba:	EPC001
Tipo de Prueba:	Funcionalidad del sistema con ingreso de 6 usuarios iniciando sección y navegando.
Objetivo de la Prueba:	Someter el software a diferentes cantidades de usuarios virtuales y en vivo,
Descripción	
Se requiere un sitio web tenga una capacidad para soportar 100 usuarios simultáneos.	
Resultado Esperado	
Verificar el límite de usuarios que pueden estar en línea simultáneamente	
Comentarios	
En revisión, ya que no se culminó el ingreso masivo de usuarios.	

Tabla 47

Especificación de pruebas de carga 002

Identificado de la Prueba:	EPC002
Tipo de Prueba:	Funcionalidad del sistema con ingreso de 25 usuarios sin ninguna actividad posterior.
Objetivo de la Prueba:	Someter el software a 25 usuarios ingresando sin actividad alguna.
Descripción	
Se requiere un sitio web tenga una capacidad para soportar 100 usuarios simultáneos.	
Resultado Esperado	
Verificar que pasa cuando se ingresan 25 usuarios sin actividad alguna dentro del sistema.	
Comentarios	
En revisión, se colapsó el sistema al ingreso número 20.	

5.06 Configuración del ambiente mínima/ideal

Esquema general de funcionamiento

La extensión de la microinformática y de las redes de ámbito mundial que interconectan recursos informáticos de todo tipo, ha hecho que los peligros que sufre la información almacenada en los diversos sistemas crezcan considerablemente y se diversifiquen, y que las medidas adoptadas internamente sean insuficientes.

En los últimos años no sólo la prensa especializada en informática, sino todos los medios de difusión han hecho eco del futuro de las autopistas de la información, cuyo embrión está representado por la red Internet. Que con el gran crecimiento que ha tenido permite mayores formas de ataque a la seguridad en red, incluyendo los virus, Caballos de Troya y penetración de las redes internas.

Los Servidores Web suministran páginas Web a los navegadores (como por ejemplo, Netscape Navigator, Internet Explorer de Microsoft) que lo solicitan. En términos más técnicos, los servidores Web soportan el Protocolo de Transferencia de Hypertexto conocido como HTTP (HyperText Transfer Protocol), el estándar de Internet para comunicaciones Web. Usando HTTP, un servidor Web envía páginas

Web en HTML y CGI, así como otros tipos de scripts a los navegadores o browsers cuando éstos lo requieren. Cuando un usuario hace clic sobre un enlace (link) a una página Web, se envía una solicitud al servidor Web para localizar los datos nombrados por ese enlace. El servidor Web recibe esta solicitud y suministra los datos que le han sido solicitados (una página HTML, un script interactivo, una página Web generada dinámicamente desde una base de datos,...) o bien devuelve un mensaje de error.(PEÑA AYALA, 2006)

Seguridad

La seguridad en redes de telecomunicaciones está fundamentada en tres elementos:

La Integridad.- Se refiere a que el contenido y el significado de la información no se alteren al viajar por una red, no obstante el número y tipo de equipos que se encuentren involucrados; la infraestructura utilizada debe ser transparente para el usuario.

La Confiabilidad.- Implica que el servicio debe estar disponible en todo momento.

La Confidencialidad.- Es quizá la parte más estratégica del negocio, ya que contribuye a impedir que personas no autorizadas lean y conozcan la información que se transmite.

Al estimar tomamos en cuenta no solo del procedimiento técnico a utilizar en el proyecto, sino que se toma en cuenta los recursos, costos y planificación. El Tamaño del proyecto es otro factor importante que puede afectar la precisión de las estimaciones.

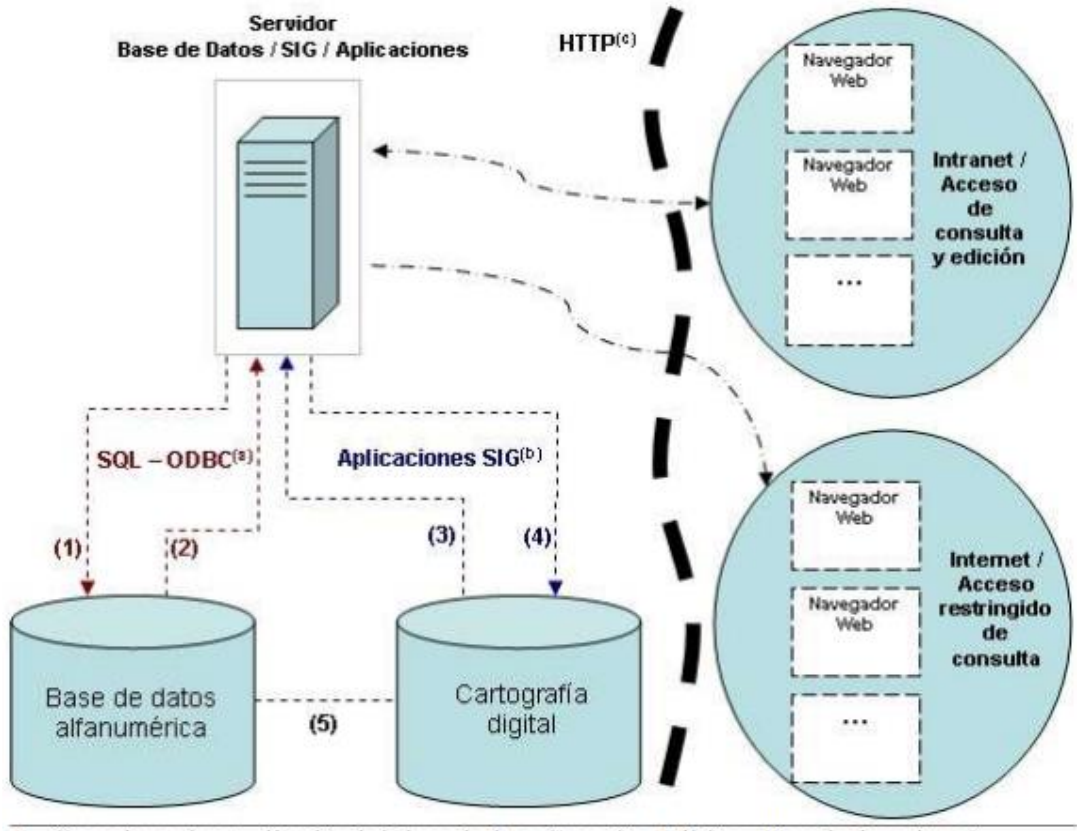


Fig 43: Esquema General de funcionamiento del sistema de Información.

Análisis de sistemas de información

Los análisis y sistemas de computación son un conjunto o disposición de procedimientos o programas relacionados de manera que juntos forman una sola unidad. Un conjunto de hechos, principios y reglas clasificadas y dispuestas de manera ordenada mostrando un plan lógico en la unión de las partes. Un método, plan o procedimiento de clasificación para hacer algo. Estas actividades se llevan a cabo teniendo en cuenta ciertos principios:

1) Debe presentarse y entenderse el dominio de la información de un problema

2) Defina las funciones que debe realizar el Software.

3) Represente el comportamiento del software a consecuencias de acontecimientos externos.

4) Divida en forma jerárquica los modelos que representan la información, funciones y comportamiento.

El proceso debe partir desde la información esencial hasta el detalle de la Implementación.

La función del Análisis puede ser dar soporte a las actividades de un negocio, o desarrollar un producto que pueda venderse para generar beneficios. Para conseguir este objetivo, un Sistema basado en computadoras hace uso de seis (6) elementos fundamentales:

- Software
- Hardware
- Herramientas del Sistema.
- Base de Datos
- Documentación, Manuales, formularios.

Procedimientos, o pasos que definen el uso específico de cada uno de los elementos o componentes del Sistema y las reglas de su manejo y mantenimiento.

Un Análisis de Sistema se lleva a cabo teniendo en cuenta los siguientes objetivos en mente:

Identifique las necesidades del cliente

Evalúe que conceptos tiene el cliente del sistema para establecer su viabilidad.

Realice un Análisis Técnico y económico.

Asigne funciones al Hardware, Software, personal, base de datos, y otros elementos del Sistema.

Establezca las restricciones de presupuestos y planificación temporal.

Cree una definición del sistema que forme el fundamento de todo el trabajo de Ingeniería.

Para lograr estos objetivos se requiere tener un gran conocimiento y dominio del Hardware y el Software, así como de la Ingeniería humana (Manejo y Administración de personal), y administración de base de datos.

En fin una serie de herramientas que se explicarán con detalle en el siguiente informe. (Kendal&Kendall, 2010).

Capítulo VI: Aspectos Administrativos

6.01. Recursos

La Ingeniería de Sistemas representa la principal herramienta de trabajo de los desarrolladores de sistemas de información. Está representa por una metodología compuesta por un conjunto de etapas que se realizan secuencialmente para dar vida a una aplicación en forma evolutiva. Cada etapa se integra por un conjunto de acciones encaminadas para obtener productos específicos, como: especificaciones, diagramas, formatos, código, pruebas y documentos diversos. Para la realización de todo sistema de información, se requiere de una serie de recursos, que aplicándolos de la manera correcta, obtendremos resultados óptimos y favorables en el desarrollo de cualquier sistema de información.

Un sistema de información es un conjunto de elementos interrelacionados con el propósito de prestar atención a las demandas de información de una organización, para elevar el nivel de conocimientos que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones. (McLeod Mr, 2000)

Los recursos necesarios para el desarrollo del proyecto generalmente se clasifican en cuatro tipos:

6.01.1 Recursos Humanos:

Humanos. Está compuesto por dos grupos:

El técnico, que posee los conocimientos especializados en el desarrollo de sistemas, siendo estos los: Administradores, Líderes de Proyecto, Analistas, Programadores, Operadores y Capturistas.

El usuario, representado por las personas interesadas en el manejo de información vía cómputo, como apoyo al mejor desempeño de sus actividades, siendo estos los: Funcionarios, Contadores, Ingenieros, Empleados, Público, etc

Para poner en marcha cualquier proyecto planificado con anterioridad, se debe disponer de personas adecuadas y capacitadas para realizar las actividades y tareas previstas.

6.01.2 Recursos Físicos:

Los recursos físicos tradicionalmente, comprenden varios ítems como terrenos, edificios, maquinaria, equipos, infraestructura, bibliografía, documentación, medios de transporte, etc. Sin embargo, este tipo de recursos no siempre deben ser adquiridos, pero sí puede ser cubiertos o suplidos con lo que se tiene. Cuando hay convencimiento y suficiente motivación para emprender una misión, es importante fomentar la movilización de recursos en donde todos ponen lo que puedan.

6.01.3 Recursos Técnicos:

En caso de que el proyecto contemple este tipo de componente, es necesario establecer las alternativas técnicas elegidas y las tecnologías a utilizar. Cuando un proyecto contempla la adopción de innovaciones tecnológicas, es bueno tener presente, que muy probablemente, la adopción de la innovación no se va a producir en un su totalidad. El proceso de transferencia de tecnología es de doble vía, es decir, la propuesta generalmente presentada por el grupo de agentes de desarrollo, al encontrarse con la tecnología tradicionalmente implementada en las comunidades, entra en un procesos de diálogo en donde ambas se transforman para evolucionar a una tercera propuesta producto de su conjunción.

6.01.4 Recursos Financieros:

Financieros. Es el aspecto económico que permite la adquisición, contratación y mantenimiento de los demás recursos que integran un sistema de información.

Una vez que se ha determinado la necesidad de los recursos financieros y la forma de obtenerlos, es pertinente saber que se aplicaran tales recursos. A través de la asignación se estiman las cantidades de dinero que se erogarán en compras de maquinaria y equipo, instalaciones, compras, de materia prima, material de empaque, mano de obra directa e indirecta, así como de los distintos costos de distribución.

Esta asignación se hace por periodos bien definidos en los distintos departamentos de la empresa.(GARCIA GIL).

Consiste en una estimación de los fondos que se pueden obtener, indicando las diferentes fuentes con que se podrán contar: presupuesto ordinario, subvenciones, pago del servicio por los usuarios, ingresos o beneficios, créditos, etc. Es necesario también establecer un calendario financiero, en donde se indica cada actividad en determinado momento del proyecto y cuáles son los recursos financieros necesarios para llevarlas a cabo.

6.02. Presupuesto

6.02.1 Costo del sistema actual

- Gastos Materiales Directos (MD)

	Mensual	6 Meses
Suministro de Oficina	$30,00	$180,00

- Gastos Materiales Indirectos (MI)

	Mensual	6 Meses
Luz	$12,00	$72,00

- Gastos Equipos de Cómputo (GC)

	Mensual	6 Meses
Depreciación (1,67%)		
1 computador ($750,00)	$10,86	$65,13
1 impresora ($120,00)	$2,01	$12,03

Total: $77,15

- Gastos Personales (GP)

	Mensual	6 Meses
Seminario	$741,44	$741,44
Semestre Instituto	$77,00	$522,00

- Gastos Varios (GV)

	Mensual	6 Meses
Varios	$30,00	$180,00

- Gastos Directos (GD)

GD= MD+MI+GC+GP+GV

GD=$180,00 + $72,00 + $77,15 + $1.063,44 + 180,00GD=$ 1572,61

6.03. Cronograma

Véase en anexo A.03

Capítulo VII: Conclusiones y Recomendaciones

7.01. Conclusiones

- El proyecto que realizamos ha contribuido de manera muy importante para identificar y resaltar los puntos que hay que cubrir y considerar para llevar a cabo una implementación exitosa de los sistemas de información apropiada para el Centro Educativo.

- Los procesos utilizados manualmente, nos dan un desfase en tiempo ya que al momento de guardar información se surge perdidas de la misma, por lo cual los sistemas automatizados nos permiten el ahorro de tiempo en un gran porcentaje, facilitando con exactitud reportes de toda índole, provocando una gran satisfacción a las personas que manejan el sistema; y en un futuro a los padres de familia.

- El Centro Educativo Jason Miller gracias a nuestro sistema de información obtendrá mayor agilidad en sus procesos lo cual les permitirá atender con rapidez los requerimientos del personal y padres de familia del centro.

- Gracias al apoyo y la información brindada por la Directora de la institución se puedo llevar a cabo el desarrollo de este sistema el cual tiene como fin primordial de culminar el presente proyecto.

7.02 Recomendaciones

- Es necesario proporcionar apoyo profesional en la utilización de sistemas de Información, así como el uso de estrategias motivacionales que animen a los Directivos de cada institución particular a adoptar nuevos métodos de manejo de información.

- Realizar respaldos de la Base de Datos para que en un futuro no se tenga inconvenientes y pérdida de información lo cual facilitara la recuperación de cualquier pérdida de información.

- Utilizar menos hojas de papel para la impresión del documento de las matriculas, reduciendo de esta manera los costos de impresión y apoyando el medio ambiente.

- El Centro Educativo Jason Miller deberá dotar más herramientas tecnológicas ya que las instituciones de Educación hoy en día implementan mucho material tecnológico adecuado a las necesidades que actualmente exige la educación Ecuatoriana.

Anexos

A.01 *Matriz de entrevistas*

IDENTIFICADOR: 001DIRECTORA DEL CENTRO EDUCATIVO JASON MILLER		
Preguntas	Objetivo	Análisis Posterior
Actualmente la institución como realiza el proceso de matriculación.	Conocer cómo trabaja actualmente la institución educativa al momento de la matriculación.	-La matrícula se la realiza en hojas de cálculo de Excel. -Trabajan en base a generación de reportes generados a base de lo que genero la hoja de cálculo. -No cuenta con ninguna clase de sistema de apoyo.
La institución cuenta con un software de soporte para las actividades en el área administrativa.	Establecer como está estructurada el área de trabajo para así plantear las necesidades requeridas.	-Existen muchas necesidades en lo que se refiere a soporte en el área administrativa ya que no cuentan con software específico. -Trabajan con hojas de cálculo de Excel para todos los registros. -Utilizan lo más necesario para apoyo.

Cuál es la problemática que busca solucionar a través del sistema de control administrativo que se va implementar.	Determinar la problemática a solucionar con el sistema Web que se va a implementar.	-Requieren realizar registro sistematizado de los docentes, alumnos y personal de la institución. -Apoyo en ingresos de las notas de las asignaturas impartidas a diario. -Que tenga versatilidad didáctica para que los programas puedan dar buena respuesta y ayuda al docente, administrativos. -Que cada proceso sea motivador y atractivo capaz de ser de mucha ayuda para el personal de la institución. -Que los usuarios trabajen acorde a su área. -La aplicación deberá ser compatible con cualquier navegador. -Tener un registro de docentes y estudiantes. -llevar un registro de asignaturas. -Los docentes llevaran registro de notas diariamente para poder sacar los promedios parciales acorde a los porcentajes asignados.

En qué área de trabajo hay más necesidades de automatizar procesos.	Determinar los tipos de necesidades que existen dentro de la Institución para así tener una visión más profunda, para la evaluación de las mismas.	-Lo más necesarios a automatizar seria: Control de asignaturas. Control de docentes. Control de estudiantes. Control de matriculación. Control de notas. Control de periodos lectivos. Control de documentación de los estudiantes. Control de curos/niveles/paralelos. Control de inscripciones. Proceso de control de evaluación de los estudiantes.
Para la evaluación de las prácticas en el software como debería ser el registro de las notas.	Analizar los niveles de evaluación requeridos.	-Las evoluciones se las calificará diariamente, mensualmente y quimestralmente para así sacar los promedios acorde a los porcentajes designados.
Que áreas de trabajo acorde a las necesidades de la institución serían las más necesarias para plasmar en nuestro sistema Web a desarrollar.	Generar un listado de áreas de trabajo a plasmar en el sistema de acuerdo a niveles de evaluación.	Se generara un listado de áreas a trabajar.
El software que se va a implementar requiere la impresión de algún tipo de documentación.	Determinar si la institución necesita algún tipo de material impreso.	-Las notas se imprimirán al finalizar cada periodo de evaluación quimestral con los porcentajes acordados a las notas registradas dentro del sistema.

NOTA: La matriz de entrevistas nos permite por medio de preguntas conocer cuáles son las necesidades más comunes dentro de la Institución para así poder determinar los procesos a seguir.

A.02 Matriz de involucrados

Actores involucrados	Intereses sobre el problema central	Problemas percibidos	Recursos, mandatos y capacidades	Intereses sobre el proyecto	Conflictos potenciales
Docentes	Mejorar la forma en que desarrollan su labor.	-Reforma inicialmente jerárquica, sin apoyo del gobierno -Escasez de tiempo Para ingresar las notas de los alumnos manualmente.	Son quienes Implementan las nuevas políticas. Su participación y cooperación es imprescindible.	Gran interés en la reforma, pero con los recursos adecuados	Con los padres, que no están acostumbrados a participar y no apoyan las nuevas prácticas
Secretaria	Mejora en los métodos de trabajo.	-Escasez de métodos de trabajo agiles y más factibles que mejores en rendimiento académico.	Recursos propios para gestión de información. Apoyo de padres de familia Apoyo en asuntos de docentes.	Gran interés en mejorar de las actividades académicas en el centro.	Dificultad en el manejo de la información de todos los alumnos.
Padres de familia	Que sus hijos reciban una buena educación con las herramientas adecuadas.	Falta de Sistemas informáticos. Falta de capacitación de los maestros. Educación irrelevante a sus necesidades Presiones Económicas fuerzan a los directivos a no contar con software adecuado de apoyo.	Capacidad de influir en sus hijos. Pueden negarse a participar en la gestión de la escuela	Facilidad de acceder a la información de su representado	La mala información recibida al momento de obtener las respectivas calificaciones.
Director	Es muy grande el	El problema se	-Recursos Económicos	El saber que se puede mejorar la	El no contar con un sistema en donde

	interés ya que lo que se busca es mejorar la empresa día a día con la aparición de nuevas tecnologías.	percibe más cuando la institución crea la necesidad de materiales de apoyo.	-Recursos Humanos	atención a los niños brindándoles innovación.	se pueda llevar a cabo el manejo adecuado de la información.
Desarrollador	Con la aparición de nuevas tecnologías se crea la necesidad de realizar nuevos y mejorados sistemas de información.	Se percibe una demora en lo que es proceso académico y toma de notas.	-ITSCO -SENRES	El interés es grande ya que me atrae la idea de realizar un software para que se aplique en las nuevas tecnologías que se manejan hoy en día.	Mejorar las velocidades de carga en el sistema

NOTA: La matriz de involucrados nos permite identificar a aquellas personas e instituciones interesadas en el éxito del proyecto así como las que contribuyen y son afectadas con el éxito del mismo.

A.03 *Cronograma*

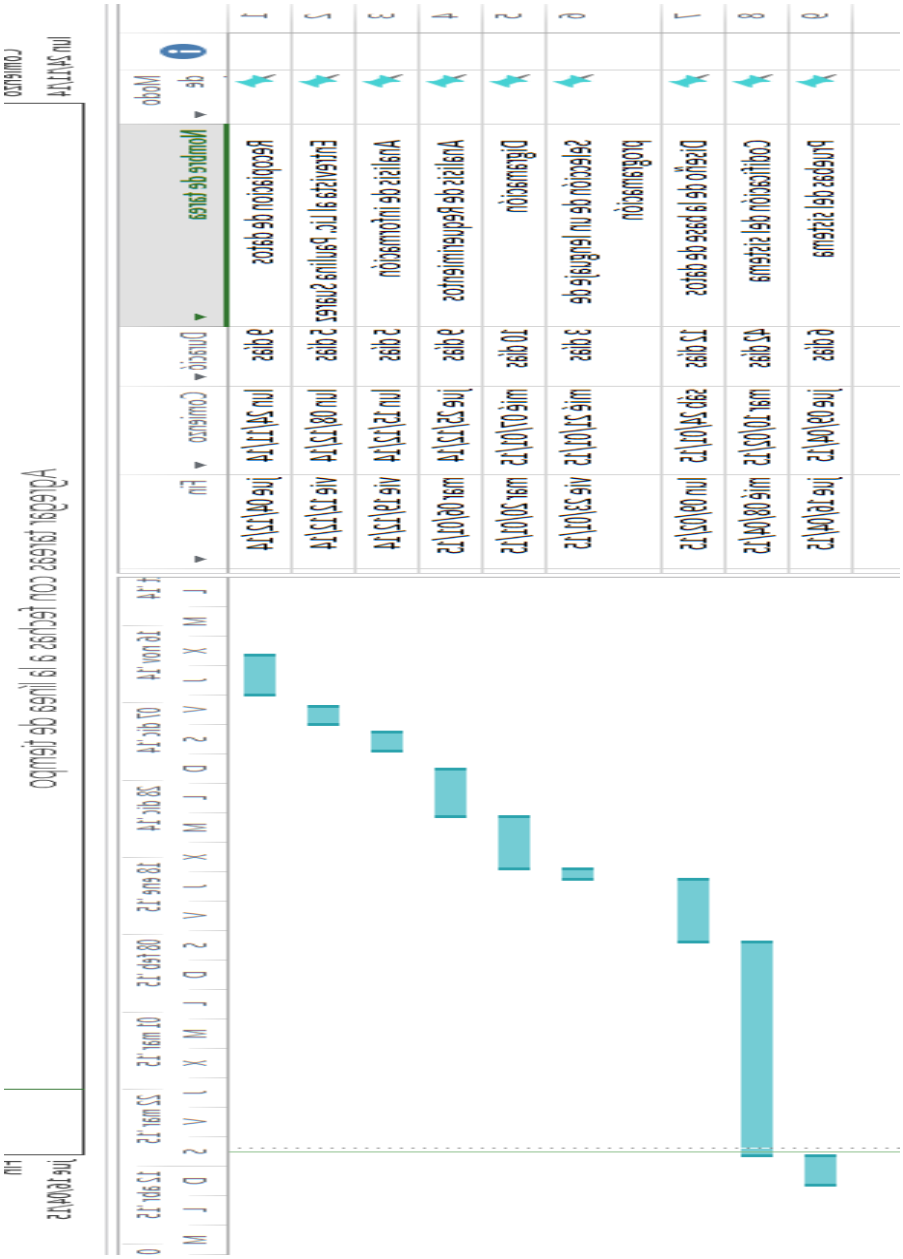


Fig 44: *Cronograma*

A.04 *Manuales*

Manual 1: Instalación de Visual Studio

1.- Puedes descargar el software en el sitio Web de visual Studio.

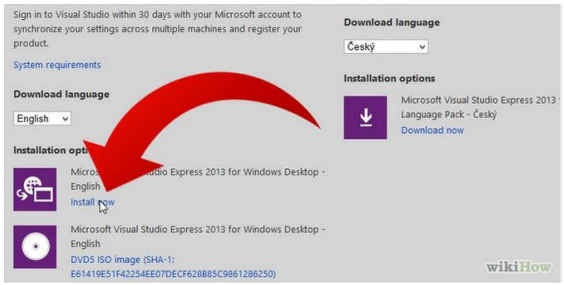


Fig 45: *Inicio de Instalación*

2.- Una vez descargado nos permitirá elegir qué tipo de visual studio es el que se desea para trabajar, en nuestro caso se trabajara con Visual Studio 2013 Preview.

Fig 46: *Elección de Visual Studio*

3.- Una vez inicializada la instalación con todos los pasos a seguir se nos desplegara la ventana de idioma con el que se desea trabajar.

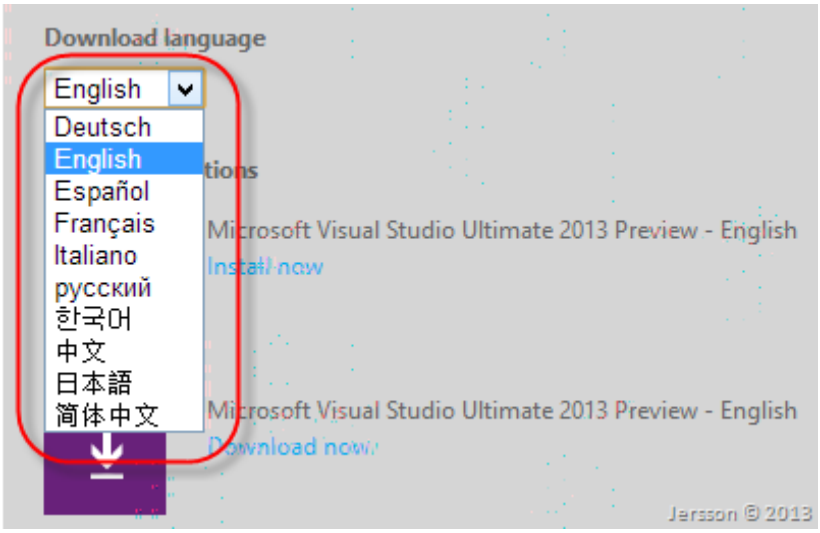


Fig 47: *Elección de Idioma de Instalación*

4.- Ya elegido el idioma con el que se va a trabajar se genera las carpetas contenedoras de nuestro sistema.

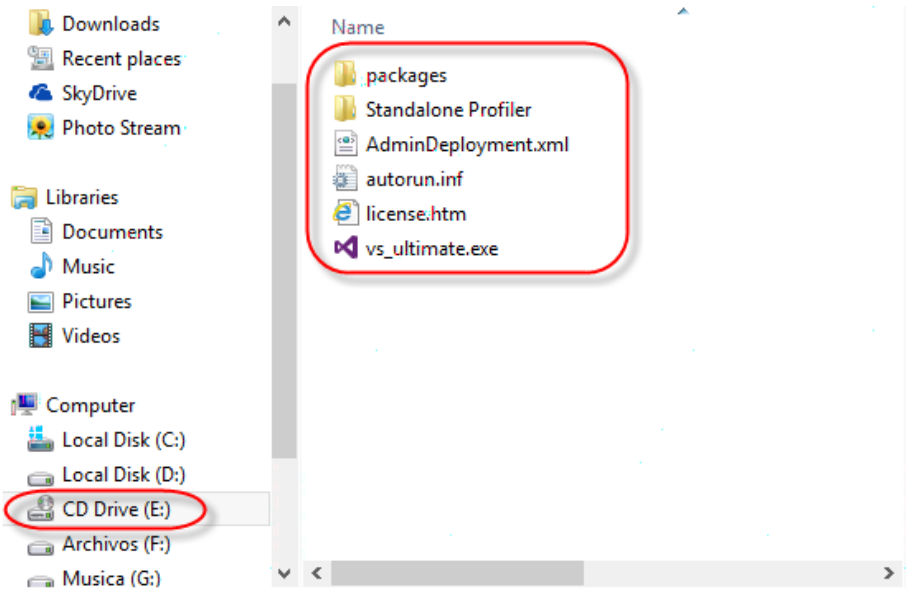


Fig 48: *Generación de carpetas contenedoras*

5.-Hacemos doble clic al instalador en este caso a vs-ultimate.exe y empezamos con la instalación.

Fig 49: *Pagina de Instalación Visual Studio*

6.- Una vez ya iniciado el proceso de instalación se procede a elegir los componentes del sistema.

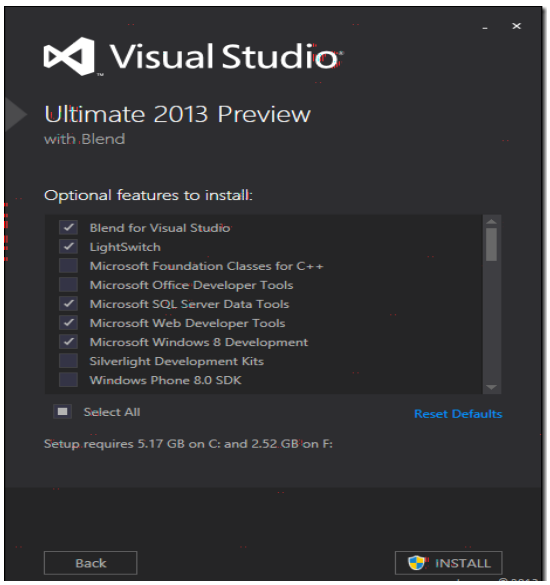


Fig 50: *Pantalla de Instalación*

7.- La instalación se completa con éxito y es necesario reiniciar el equipo.

Fig 51: *Pantalla de aviso de termino de instalación*

8.- Al momento de ingresar a la ventana de inicio aparece el entorno de la nube el cual no pide que se inicie sección.

Fig 52: ***Pantalla de inicio de Sesión***

9.- Si ya se tiene creada una cuenta en Microsoft se abrirá automáticamente si no es necesario crear una.

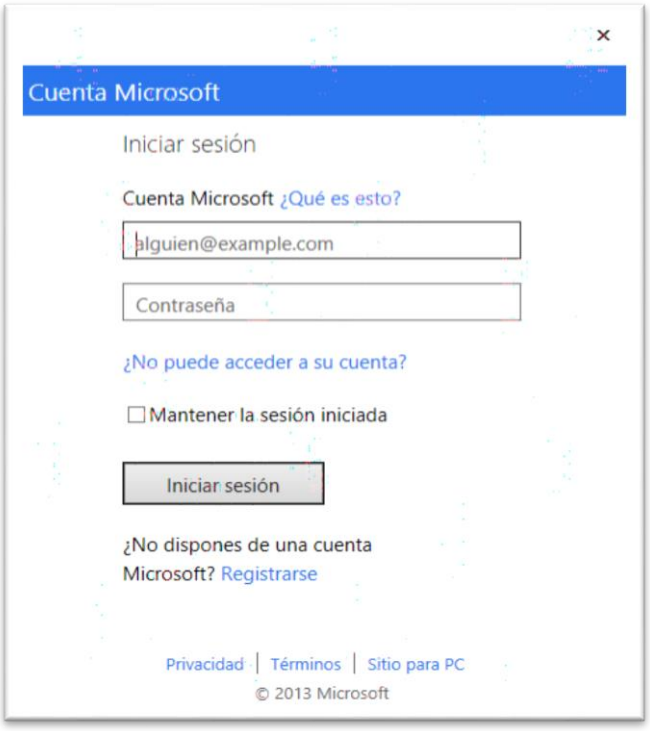


Fig 53: *Ingreso de datos*

10.- Una vez ingresado se nos desplegara la ventana de bienvenida del de Visual Studio 2013.

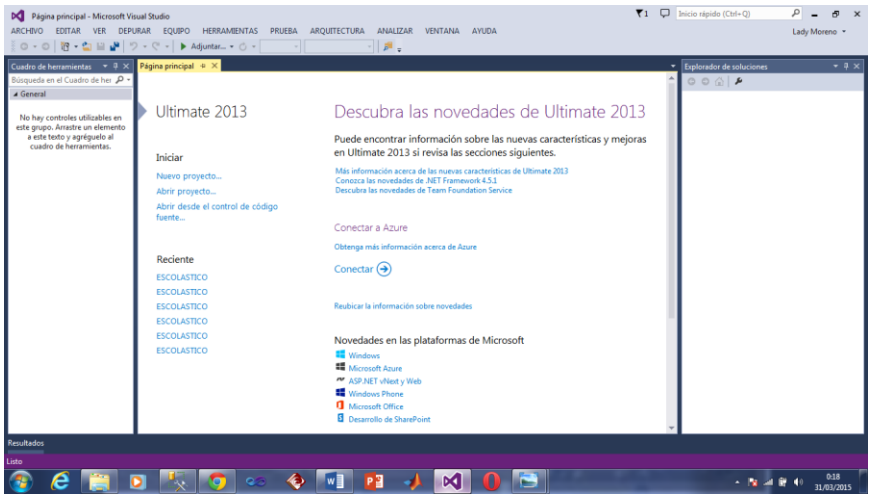


Fig 54: Instalación lista

Manual 2: Instalación de Crystal Report para visual Studio

Comenzaremos instalando un complemento de visual studio, esta herramienta la utilizaremos para la generación de reportes.

1.-Descargar el archivo de la siguiente url:

http://www.sap.com/solution/sme/software/analytics/crystal-visual-studio/index.html

2.-Una vez descargado, descomprimimos el archivo y click en el setup.exe de la aplicación.

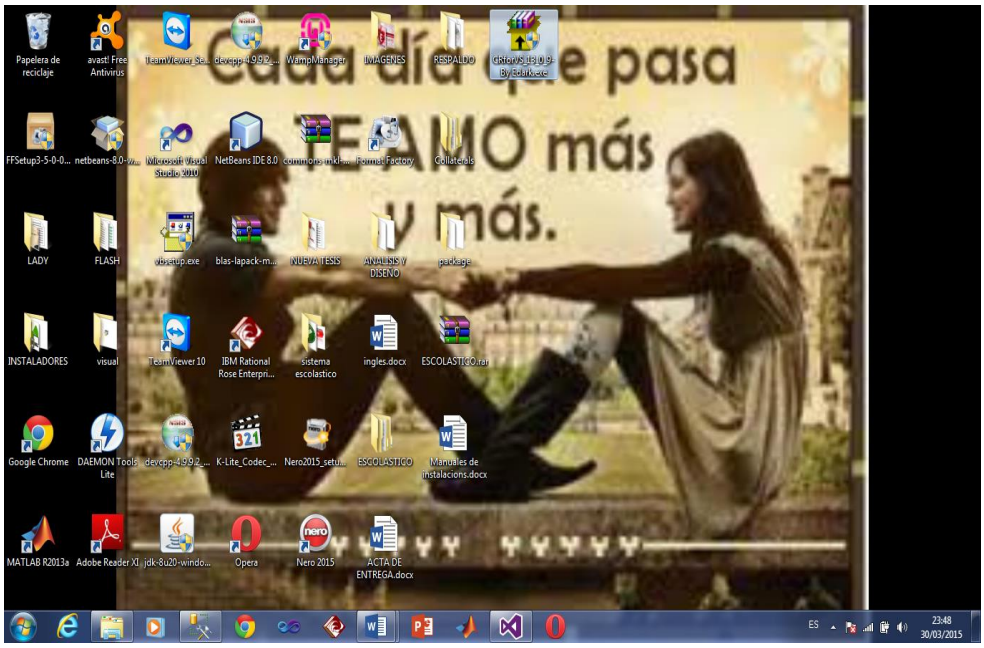


Fig 55: *Carpeta contenedora de Crystal Reports*

3.-Al iniciar la instalación, nos aparece una ventana, que nos indica el comienzo de la instalación del programa, click en siguiente.

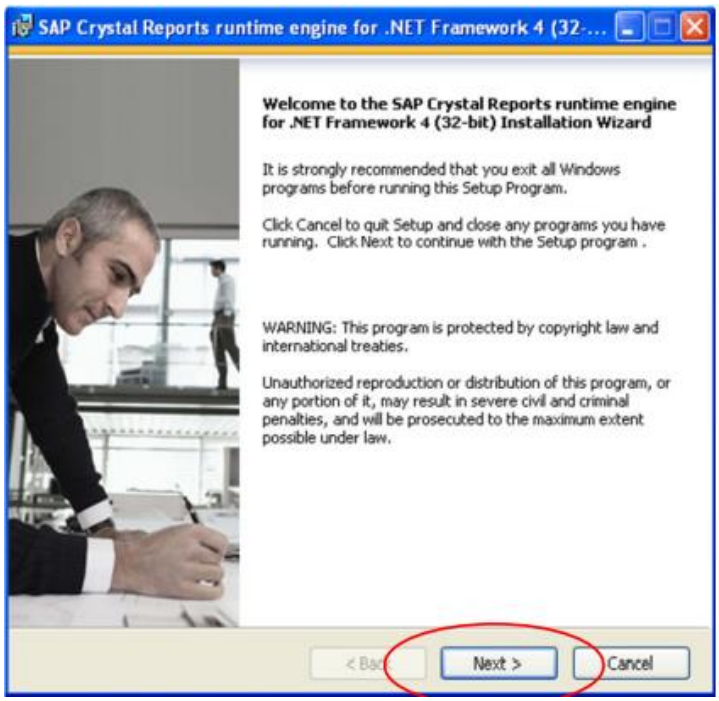


Fig 56: *Pantalla de Bienvenida de Crystal Reports*

4.-Nos despliega las condiciones de uso y pantalla de bienvenida de la instalación, aceptamos y click en siguiente.

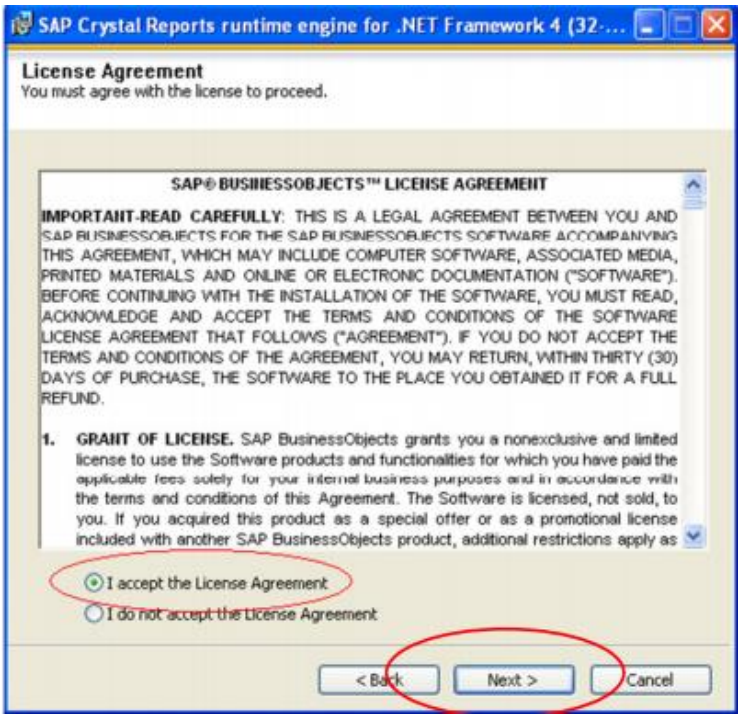


Fig 57: *Pantalla de Licencia*

5.-Aceptamos términos y condiciones del programa, click en siguiente y observaremos la instalación del complemento de visual.

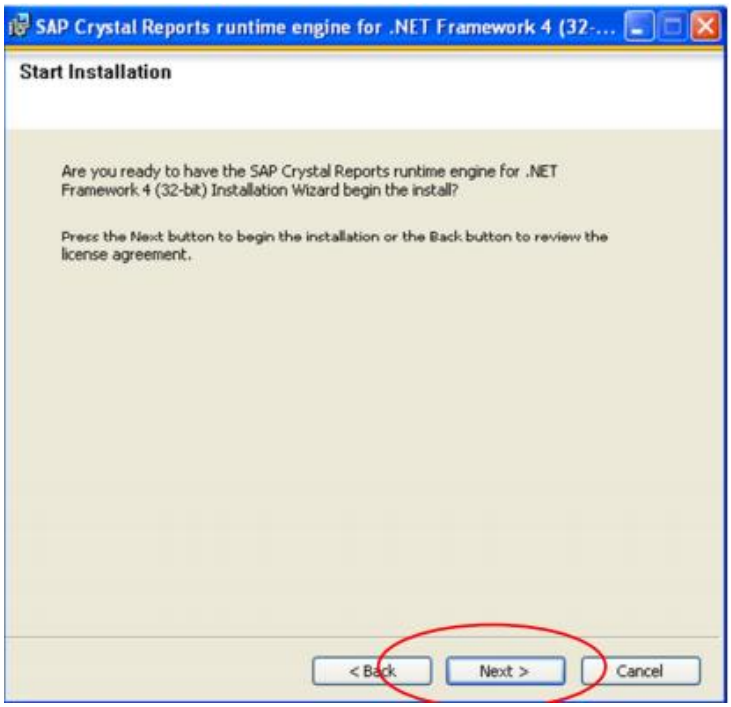


Fig 58: *Star Installation*

6.-Una vez culminada la instalación, click en finalizar y hemos terminado la instalación del complemento de visual studio 2013 el cual es SAP Crystal Reports.

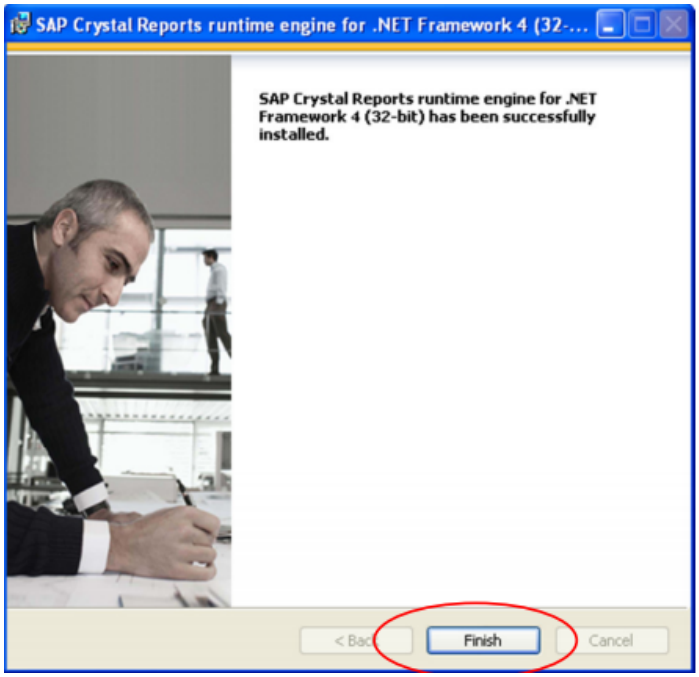


Fig 59: *Finish*

7.- Observamos dentro de visual studio 2013, en el cuadro de herramientas de la siguiente manera que ya cuenta con la instalación de su complemento.

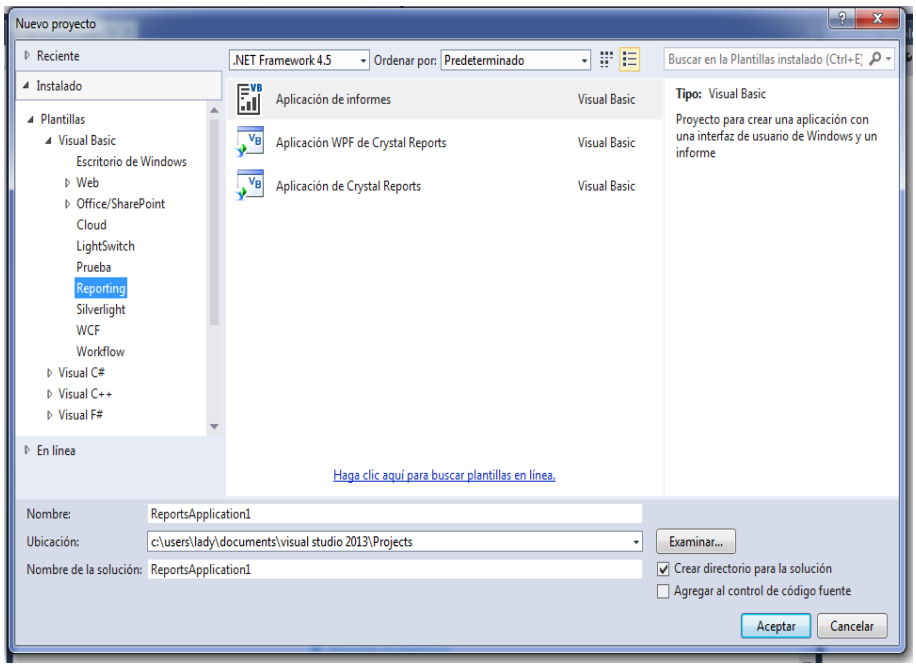


Fig 60: *Pantalla de nuevo proyecto*

Manual 3: Instalación de SQL Server 2008 R2

Para trabajar con base de datos se ha escogido como gestor de nuestra base de datos con SQL SERVER 2008 R2

1.- Descargamos el setup.exe del siguiente url:

http://technet.microsoft.com/en-us/bb851664.aspx

2.-Click derecho en ejecutar como administrador

3.- Nos desplegará una ventana indicando el inicio de la instalación de nuestro gestor de base de datos sql server 2008 R2

Fig 61: *Inicio de Instalación SQL Server*

Podemos revisar los requerimientos mínimos necesarios para la instalación

4.-Click en la opción **Installation,** y seleccionamos New Sql Sever, click en siguiente.

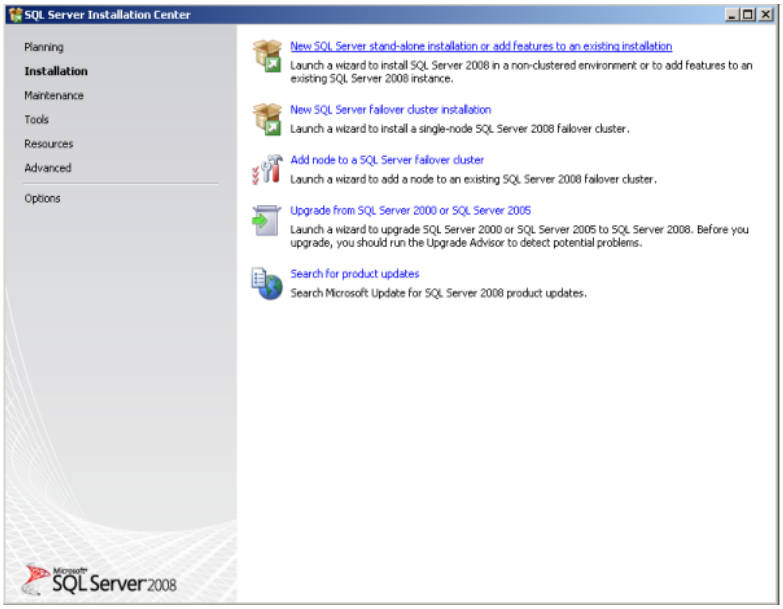


Fig 62: *Carpeta de Instalación*

5.-En el recuadro, realizamos click en siguiente para continuar con el proceso de instalación.

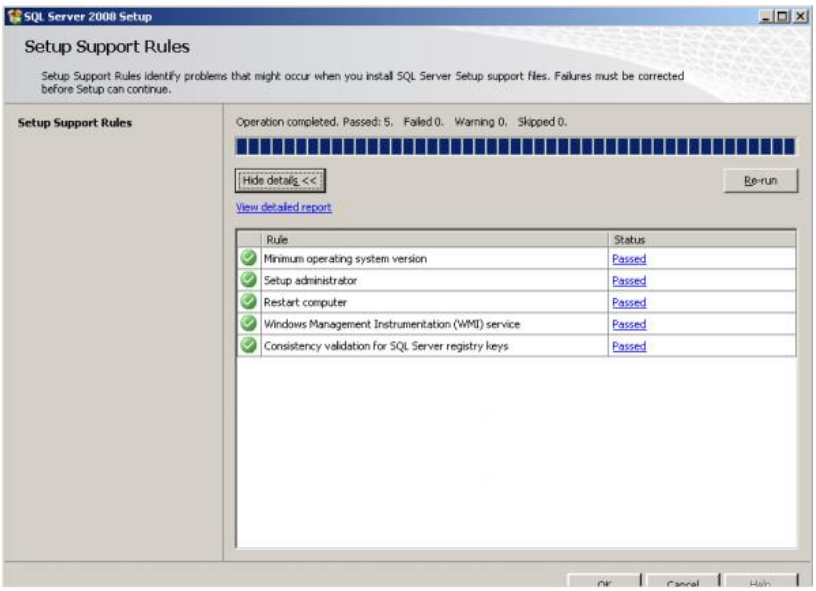


Fig 63: *Septup Support Rules*

6.-Aceptamos los términos y condiciones del sistema que nos indican a continuación y click en siguiente

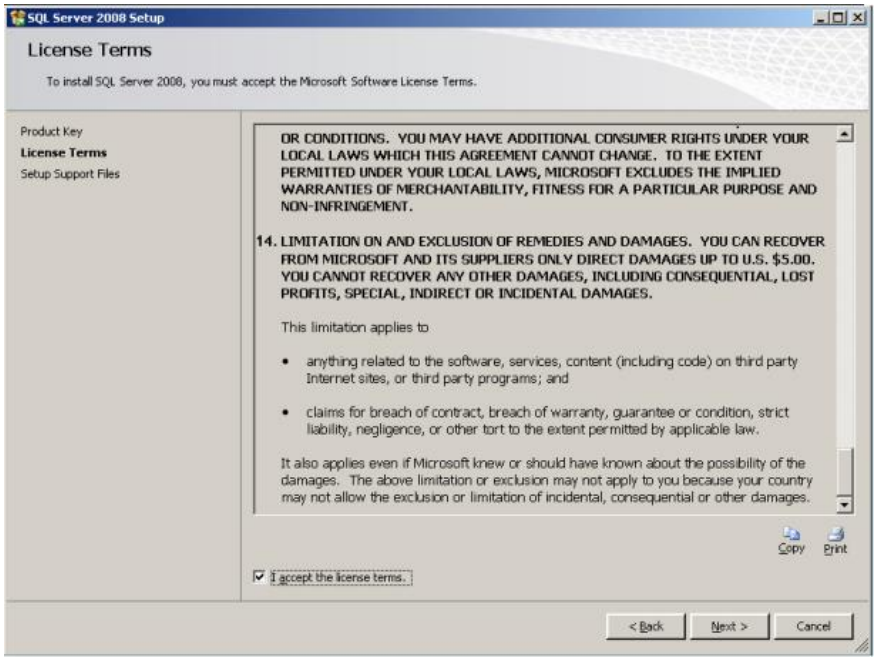


Fig 64: *Términos y Condiciones*

7.-Click en Install.

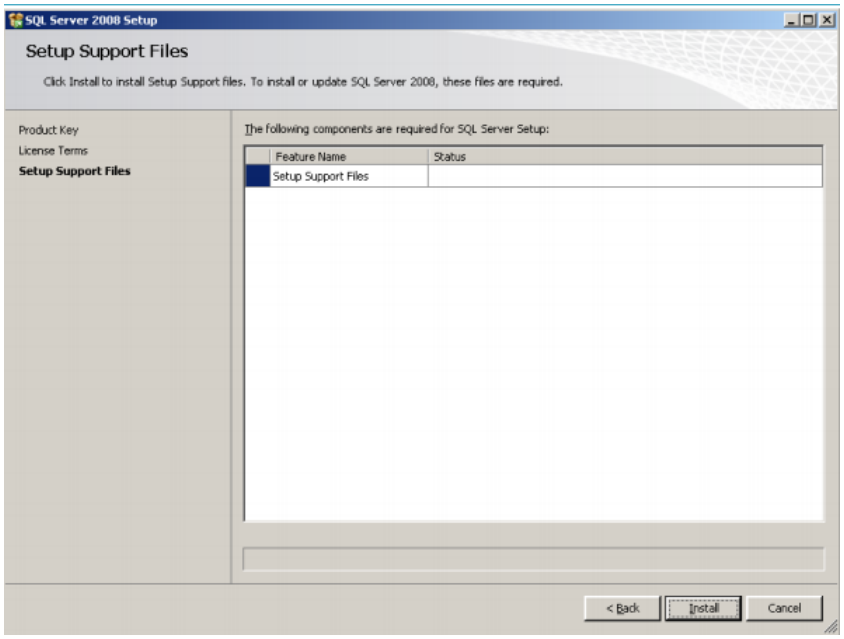


Fig 65: *Instalar*

8.-A continuación seleccionamos las características que vamos a utilizar en nuestro gestor de base de datos.

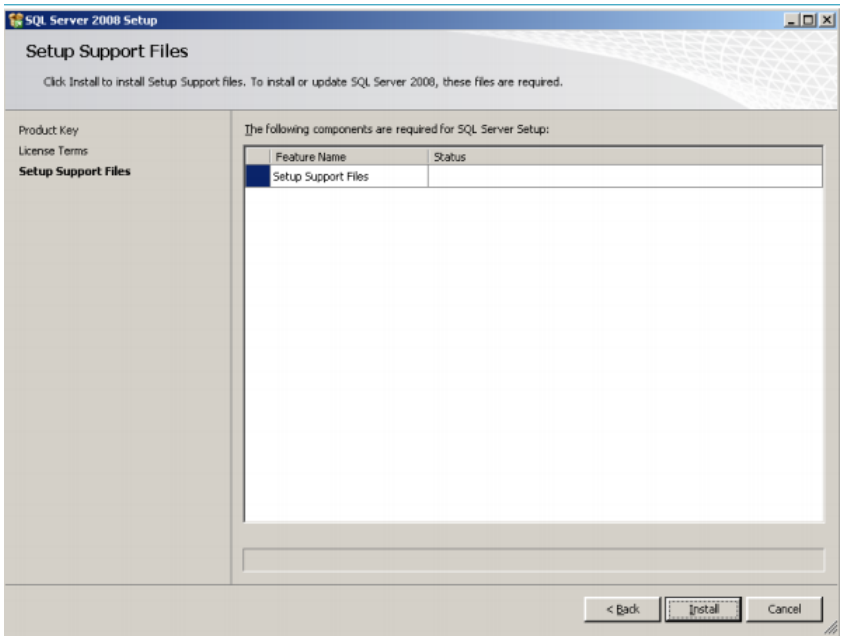


Fig 66: *Selección características*

9.-Realizamos click en las ventanas siguientes hasta llegar a la ventana que indica si el usuario desea ingresar una contraseña propia o usar la de autentificación de Windows, la decisión la tiene el usuario, nosotros usaremos la autentificación de Windows.

Fig 67: ***Configuración de la base de datos***

10.- Click en siguiente en las demás ventanas, y esperamos unos minutos hasta llegar al siguiente recuadro de instalación.

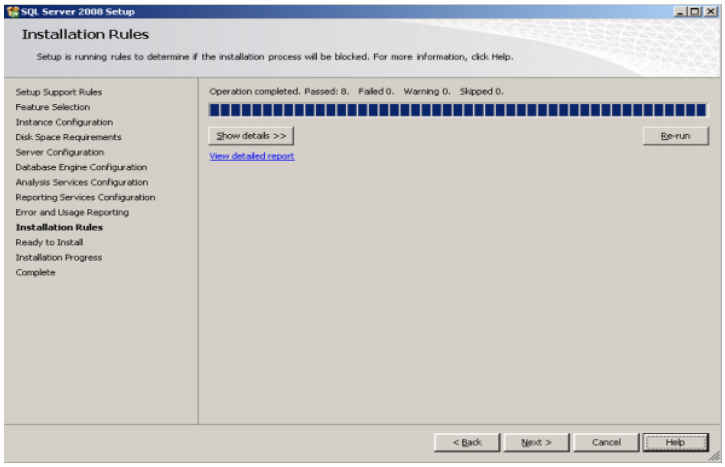


Fig 68: Ventana de proceso de instalación

11.-Click en Next hasta culminar la instalación, al final veremos un recuadro que nos indicará la finalización del a instalación y click en Close.

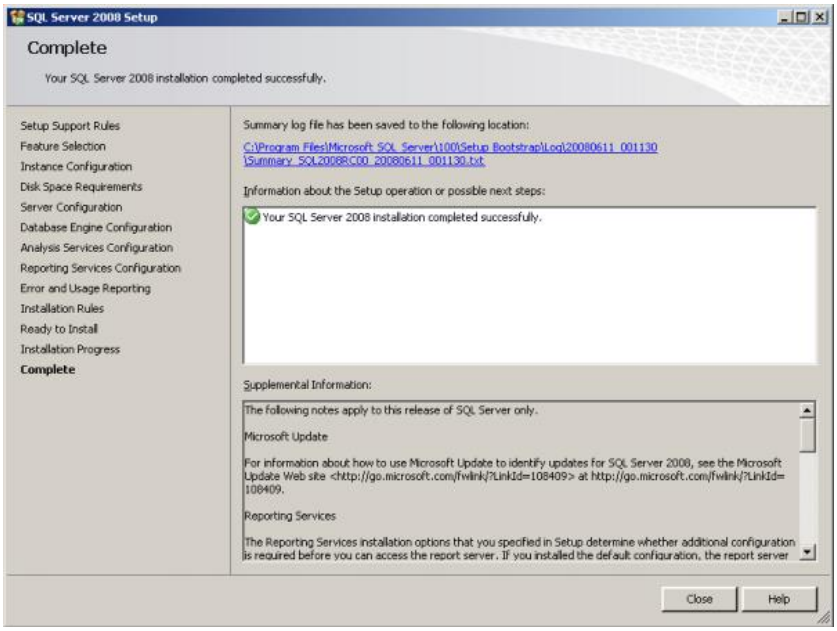


Fig 69: *Instalación completa*

12.-Para ver su correcta instalación, en nuestra barra de herramientas, buscamos sql server management, lo ejecutamos y visualizaremos la siguiente pantalla.

Fig 70: *Pantalla de ejecución*

13.-Visualizamos que se ha instalado correctamente nuestro gestor de base de datos.

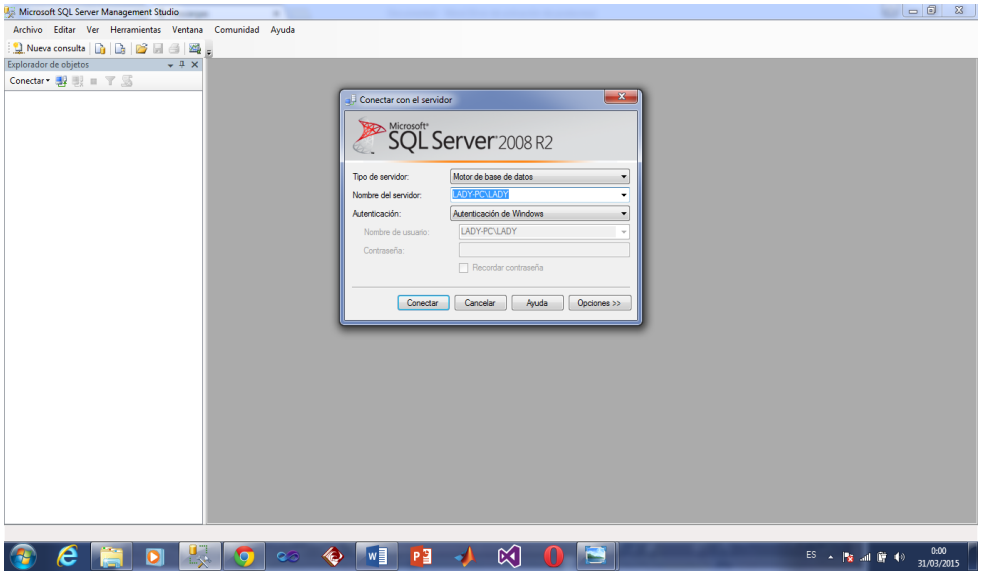


Fig 71: *Pantalla de Instalación correcta*

Hemos finalizado la instalación con éxito, ahora ya podemos crear nuestro modela de base de datos dentro de nuestro Gestor.

Manual 4: Usuario

1.-Ingreso al Sistema

Fig 72: *Pantalla de Inicio*

Nombre de usuario

Contraseña

Clik en ingresar

2.-Ingresamos a la pantalla principal

Fig 73: *Pantalla de Bienvenida*

Misión

Reseña histórica

3.-Seleccionamos los campos a manejar

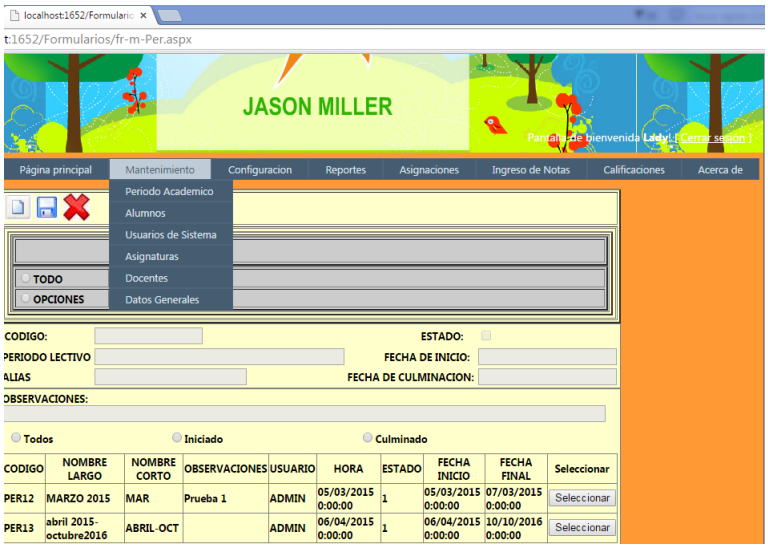


Fig 74: *Tablas de Mantenimientos*

Alumnos

Docentes

Periodo lectivo

Asignaturas

Usuarios del sistema

Datos generales

4.-Primeramente ingresamos al periodo académico.

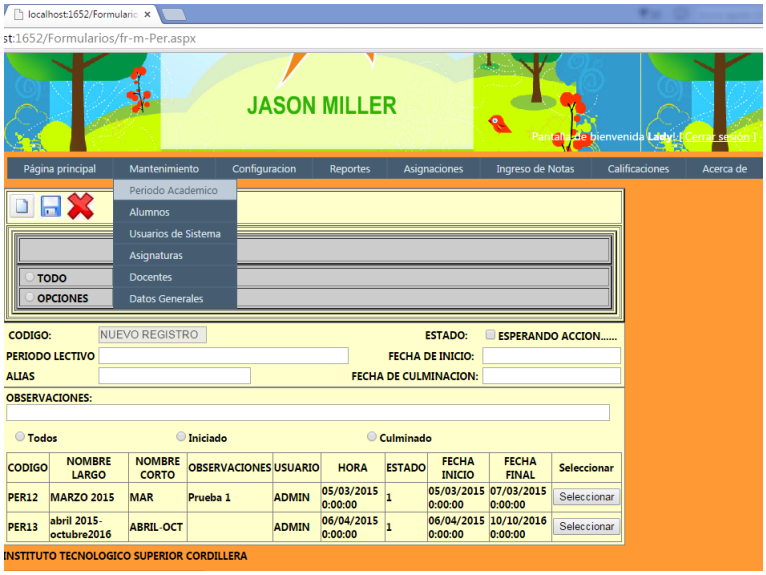


Fig 75: ***Pantalla de periodo lectivo***

Llenamos los campos de periodo

Alias

Estado

Fecha de inicio

Fecha de culminación

5.- Ingresamos a la pantalla de alumnos e ingresamos uno,

Fig 76: *Pantalla principal de alumnos*

Pantalla principal para ingreso de alumnos

6.-Botones de nuevo, guardar y cancelar

Fig 77: *Vista de Botones*

7.- Ingreso de usuarios el Sistema

Fig 78: Pantalla de usuarios

8.-Pantalla principal para usuario del sistema

Fig 79: *Pantalla de usuario con campos llenos*

Ingresamos un nuevo usuario

El código se genera automáticamente

Ingrese nombre de usuario

Ingrese contraseña

Repita la contraseña

Elegimos el rol y activamos para el ingreso

9. Ingresamos al campo asignatura

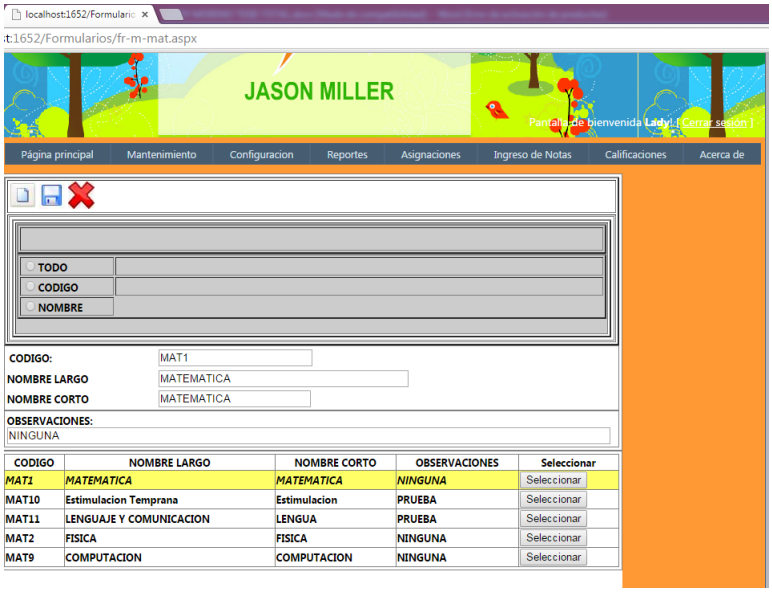


Fig 80: *Ingreso de asignaturas*

Código asignatura

Nombre

Alias

10. Ingresamos al campo de docentes

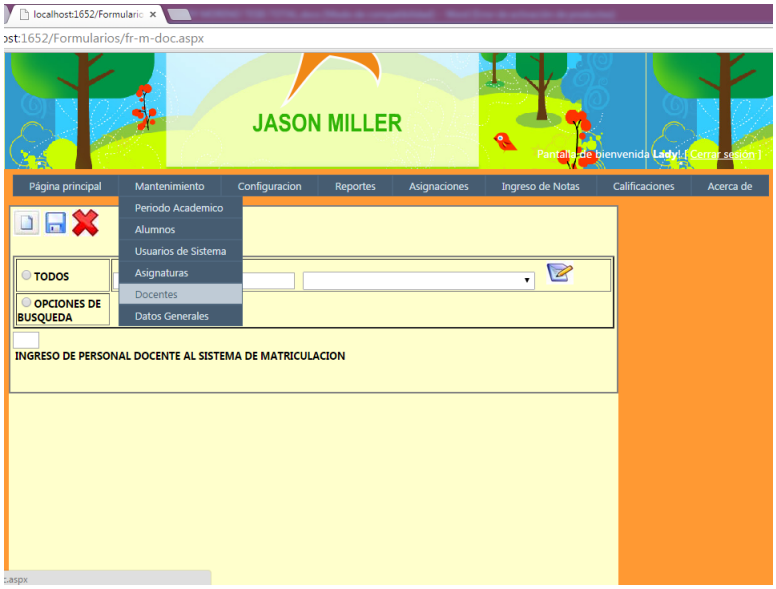


Fig 81: *Ingreso a campo de docentes*

Pantalla principal de docente

Botones nuevo, guardar, eliminar

11.- Ingreso a campo de docente para ingresar un nuevo docente.

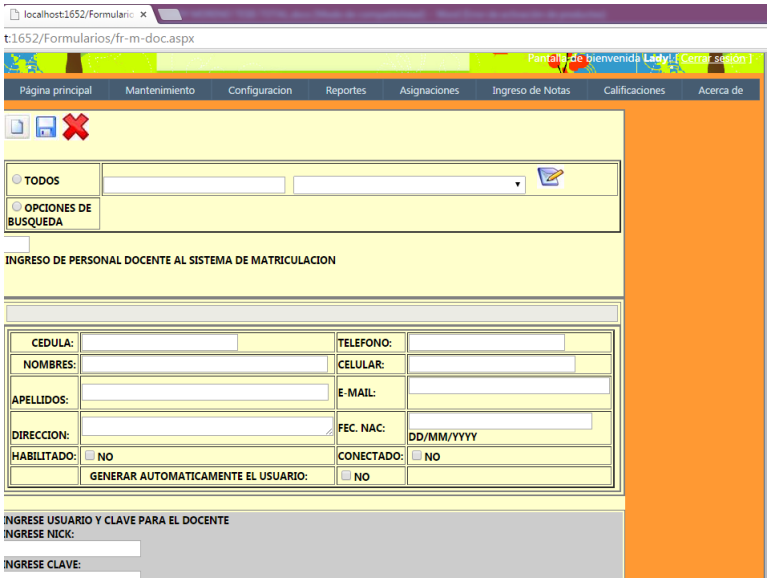


Fig 82: *Campos de docente activados*

Llenamos los campos

Cédula

Nombres

Apellidos

Dirección

Teléfono

Celular

Email

Fecha de nacimiento

Habilitamos al usuario

12.- Ingreso a códigos secuenciales del sistema

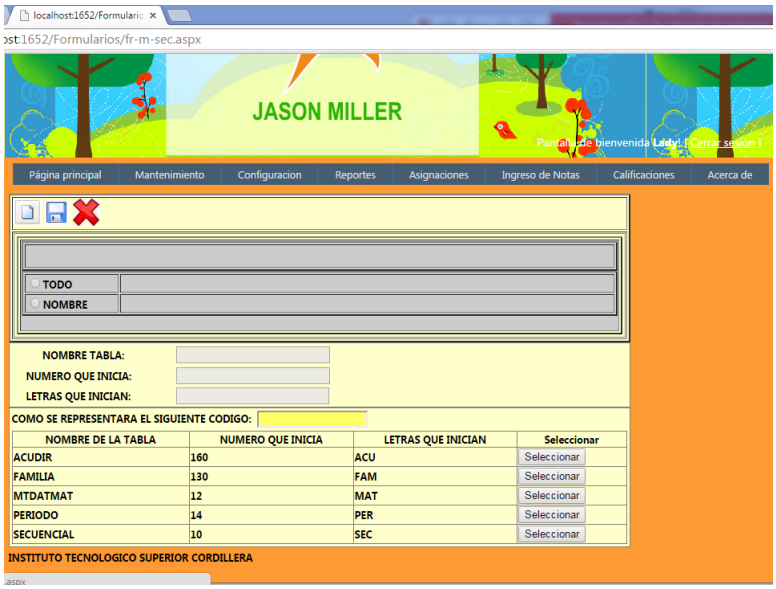


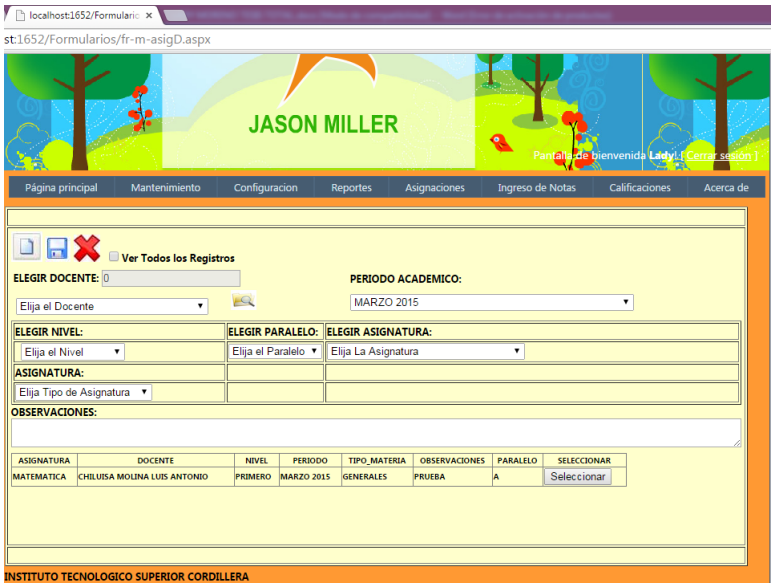
Fig 83: *Códigos secuenciales*

Nombre de tabla

Número que inicia

Letras con que inicia

10. Nivel de asignaciones de docentes y estudiantes



Elegir dicente

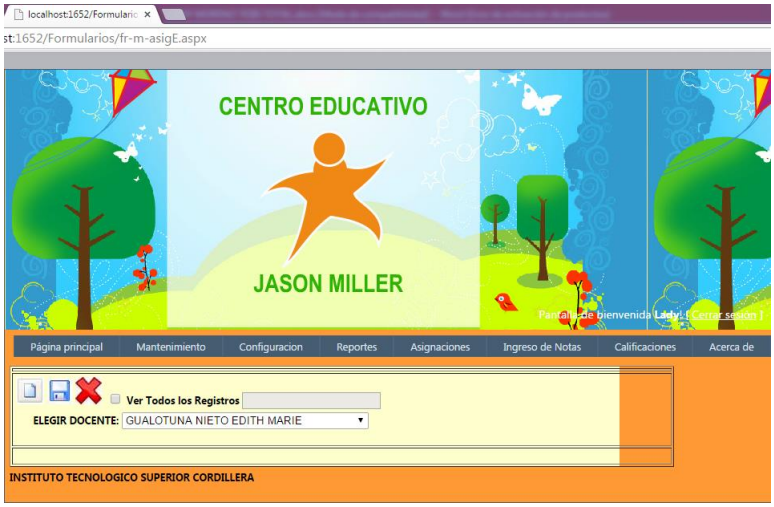
Periodo académico

Elegir nivel

Elegir paralelo

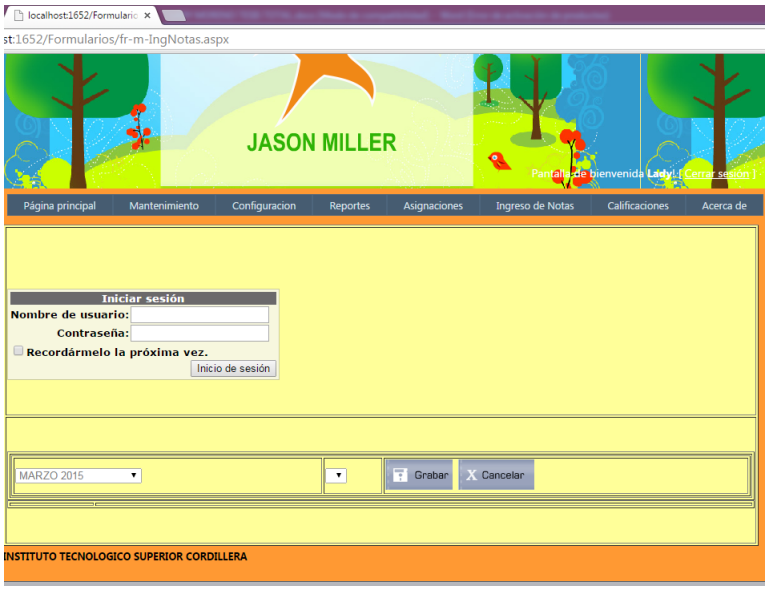
Elegir asignatura

Observaciones

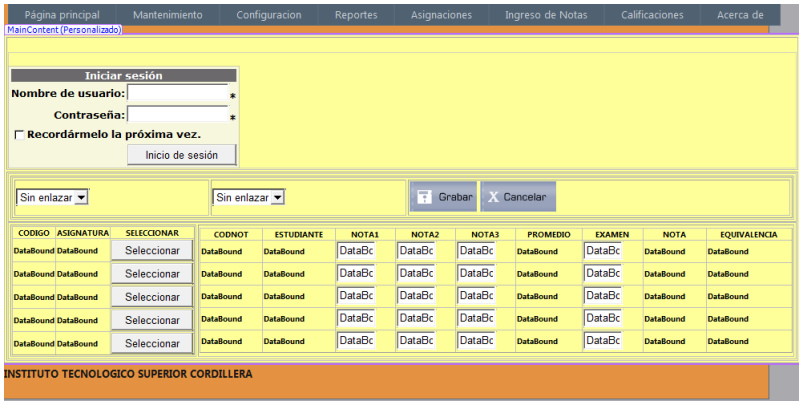


Elegimos docente para ver quien está matriculado con ese docente

11. Ingreso de notas al sistema



El usuario en estos casos los docentes tendrán ingreso a las notas una vez ingresado por medio de su contraseña y usuario.



Llenamos los capos de notas esto no dará el promedio automáticamente y por tanto el parcial.

12. Generación de Calificaciones

Aquí se realiza la generación de calificaciones de los alumnos.

Manual 5: Técnico del Sistema

1.01 Scrip base de datos (SQL SERVER 2008)

```
USE[master]
GO
/****** Object:  Database [Escolastico]    Script Date: 04/07/2015
02:01:07 ******/
CREATEDATABASE[Escolastico]ONPRIMARY
(NAME=N'Escolastico',FILENAME=N'C:\Program Files (x86)\Microsoft SQL
Server\MSSQL10_50.LADY\MSSQL\DATA\Escolastico.mdf',SIZE=
3072KB,MAXSIZE=UNLIMITED,FILEGROWTH= 1024KB)
LOGON
(NAME=N'Escolastico_log',FILENAME=N'C:\Program Files (x86)\Microsoft
SQL Server\MSSQL10_50.LADY\MSSQL\DATA\Escolastico_log.ldf',SIZE=
1024KB,MAXSIZE= 2048GB,FILEGROWTH= 10%)
GO
ALTERDATABASE[Escolastico]SETCOMPATIBILITY_LEVEL= 100
GO
IF (1 =FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC[Escolastico].[dbo].[sp_fulltext_database]@action='enable'
end
GO
ALTERDATABASE[Escolastico]SETANSI_NULL_DEFAULTOFF
GO
ALTERDATABASE[Escolastico]SETANSI_NULLSOFF
GO
ALTERDATABASE[Escolastico]SETANSI_PADDINGOFF
GO
ALTERDATABASE[Escolastico]SETANSI_WARNINGSOFF
GO
ALTERDATABASE[Escolastico]SETARITHABORTOFF
GO
ALTERDATABASE[Escolastico]SETAUTO_CLOSEOFF
GO
ALTERDATABASE[Escolastico]SETAUTO_CREATE_STATISTICSON
GO
ALTERDATABASE[Escolastico]SETAUTO_SHRINKOFF
GO
ALTERDATABASE[Escolastico]SETAUTO_UPDATE_STATISTICSON
GO
ALTERDATABASE[Escolastico]SETCURSOR_CLOSE_ON_COMMITOFF
GO
ALTERDATABASE[Escolastico]SETCURSOR_DEFAULTGLOBAL
GO
ALTERDATABASE[Escolastico]SETCONCAT_NULL_YIELDS_NULLOFF
GO
ALTERDATABASE[Escolastico]SETNUMERIC_ROUNDABORTOFF
GO
ALTERDATABASE[Escolastico]SETQUOTED_IDENTIFIEROFF
GO
ALTERDATABASE[Escolastico]SETRECURSIVE_TRIGGERSOFF
GO
ALTERDATABASE[Escolastico]SETDISABLE_BROKER
```

```
GO
ALTERDATABASE[Escolastico]SETAUTO_UPDATE_STATISTICS_ASYNCOFF
GO
ALTERDATABASE[Escolastico]SETDATE_CORRELATION_OPTIMIZATIONOFF
GO
ALTERDATABASE[Escolastico]SETTRUSTWORTHYOFF
GO
ALTERDATABASE[Escolastico]SETALLOW_SNAPSHOT_ISOLATIONOFF
GO
ALTERDATABASE[Escolastico]SETPARAMETERIZATIONSIMPLE
GO
ALTERDATABASE[Escolastico]SETREAD_COMMITTED_SNAPSHOTOFF
GO
ALTERDATABASE[Escolastico]SETHONOR_BROKER_PRIORITYOFF
GO
ALTERDATABASE[Escolastico]SETREAD_WRITE
GO
ALTERDATABASE[Escolastico]SETRECOVERYSIMPLE
GO
ALTERDATABASE[Escolastico]SETMULTI_USER
GO
ALTERDATABASE[Escolastico]SETPAGE_VERIFYCHECKSUM
GO
ALTERDATABASE[Escolastico]SETDB_CHAININGOFF
GO
USE[Escolastico]
GO
/****** Object:  Table [dbo].[SECUENCIAL]    Script Date: 04/07/2015
02:01:08 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
SETANSI_PADDINGON
GO
CREATETABLE[dbo].[SECUENCIAL](
      [CODPAR][nvarchar](10)NOTNULL,
      [PARSEC][nvarchar](10)NOTNULL,
      [CADPAR][varchar](3)NULL,
PRIMARYKEYCLUSTERED
(
      [CODPAR]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
SETANSI_PADDINGOFF
GO
/****** Object:  StoredProcedure [dbo].[SearchAllTables]    Script
Date: 04/07/2015 02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROC[dbo].[SearchAllTables]
(
@SearchStrnvarchar(100)
```

```
)
AS
BEGIN

CREATETABLE#Results(ColumnNamenvarchar(370),ColumnValuenvarchar(3630
))

SETNOCOUNTON

DECLARE@TableNamenvarchar(256),@ColumnNamenvarchar(128),@SearchStr2n
varchar(110)
SET@TableName=''
SET@SearchStr2=QUOTENAME('%'+@SearchStr+'%','''')

WHILE@TableNameISNOTNULL
BEGIN
SET@ColumnName=''
SET@TableName=
(
SELECTMIN(QUOTENAME(TABLE_SCHEMA)+'.'+QUOTENAME(TABLE_NAME))
FROMINFORMATION_SCHEMA.TABLES
WHERETABLE_TYPE='BASE TABLE'
ANDQUOTENAME(TABLE_SCHEMA)+'.'+QUOTENAME(TABLE_NAME)>@TableName
ANDOBJECTPROPERTY(
OBJECT_ID(
QUOTENAME(TABLE_SCHEMA)+'.'+QUOTENAME(TABLE_NAME)
),'IsMSShipped'
)= 0
)

WHILE (@TableNameISNOTNULL)AND(@ColumnNameISNOTNULL)
BEGIN
SET@ColumnName=
(
SELECTMIN(QUOTENAME(COLUMN_NAME))
FROMINFORMATION_SCHEMA.COLUMNS
WHERETABLE_SCHEMA=PARSENAME(@TableName, 2)
ANDTABLE_NAME=PARSENAME(@TableName, 1)
ANDDATA_TYPEIN('char','varchar','nchar','nvarchar')
ANDQUOTENAME(COLUMN_NAME)>@ColumnName
)

IF@ColumnNameISNOTNULL
BEGIN
INSERTINTO#Results
EXEC
(
'SELECT '''+@TableName+'.'+@ColumnName+''', LEFT('+@ColumnName+',
3630)
                    FROM '+@TableName+' (NOLOCK) '+
' WHERE '+@ColumnName+' LIKE '+@SearchStr2
)
END
END
END

SELECTColumnName,ColumnValueFROM#Results
END
```

```
GO
/****** Object:  Table [dbo].[REPORTE]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[REPORTE](
      [CODREP][nvarchar](70)NOTNULL,
      [PATHREP][nvarchar](200)NOTNULL,
      [NOMREPORT][nvarchar](100)NOTNULL,
PRIMARYKEYCLUSTERED
(
      [CODREP]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATPER]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATPER](
      [COD_SEM][nvarchar](5)NOTNULL,
      [NOML_SEM][nvarchar](50)NULL,
      [NOMC_SEM][nvarchar](20)NULL,
      [OBSER_CAR][nvarchar](1000)NULL,
      [AS_USUARIO][nvarchar](30)NULL,
      [AS_HORA][date]NULL,
      [ESTADO][nvarchar](1)NULL,
      [FECINI][date]NULL,
      [FECCUL][date]NULL,
CONSTRAINT[PK_SEMESTRE]PRIMARYKEYCLUSTERED
(
      [COD_SEM]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATNOT]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATNOT](
      [CODNOT][int]IDENTITY(1,1)NOTNULL,
      [CODIASIG][int]NULL,
      [CODIEST][nvarchar](10)NULL,
      [NOT1NOT][decimal](10, 2)NULL,
      [NOT2NOT][decimal](10, 2)NULL,
      [NOT3NOT][decimal](10, 2)NULL,
      [PRONOT][decimal](10, 2)NULL,
      [EXANOT][decimal](10, 2)NULL,
```

```
	[NOTNOT][decimal](10, 2)NULL,
	[EQUINOT][decimal](10, 2)NULL,
PRIMARYKEYCLUSTERED
(
	[CODNOT]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATMAT]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATMAT](
	[CODMAT][nvarchar](6)NOTNULL,
	[NOMLMAT][nvarchar](70)NULL,
	[NOMCMAT][nvarchar](50)NULL,
	[OBSMAT][nvarchar](1000)NULL,
	[AS_USUARIO][nvarchar](30)NULL,
	[AS_HORA][date]NULL,
PRIMARYKEYCLUSTERED
(
	[CODMAT]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATFAM]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATFAM](
	[CODFAM][nvarchar](7)NOTNULL,
	[NOMFAM][nvarchar](70)NOTNULL,
	[APEFAM][nvarchar](70)NOTNULL,
	[NIVFAM][nvarchar](7)NULL,
	[OCUFAM][nvarchar](7)NULL,
	[AREFAM][nvarchar](7)NULL,
	[TELFAM][nvarchar](10)NOTNULL,
	[CELFAM][nvarchar](10)NOTNULL,
	[DIRFAM][nvarchar](250)NOTNULL,
	[VIVFAM][nvarchar](1)NOTNULL,
	[REPFAM][nvarchar](1)NOTNULL,
	[AUTFAM][nvarchar](1)NOTNULL,
	[CODIEST][nvarchar](10)NULL,
	[TIPFAM][nvarchar](1)NOTNULL,
PRIMARYKEYCLUSTERED
(
	[CODFAM]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
```

```
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATEST]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATEST](
	[CODEST][nvarchar](10)NOTNULL,
	[NOMEST][nvarchar](70)NOTNULL,
	[APEEST][nvarchar](70)NOTNULL,
	[FECNACEST][datetime]NULL,
	[NACEST][nvarchar](7)NULL,
	[POSCEDEST][nvarchar](1)NOTNULL,
	[CEDEST][nvarchar](10)NOTNULL,
	[SEXEST][nvarchar](7)NULL,
	[CMEDEST][nvarchar](1)NOTNULL,
	[CDENEST][nvarchar](1)NOTNULL,
	[CCEDPEST][nvarchar](1)NOTNULL,
	[CCEDMEST][nvarchar](1)NOTNULL,
	[PNACEST][nvarchar](1)NOTNULL,
	[FOTO][nvarchar](100)NOTNULL,
	[PCEDEST][nvarchar](1)NOTNULL,
	[NUMMATEST][nvarchar](10)NOTNULL,
	[CODMINEDU][nvarchar](10)NOTNULL,
	[FECMATEST][datetime]NOTNULL,
PRIMARYKEYCLUSTERED
(
	[CODEST]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[MTDATDOC]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
SETANSI_PADDINGON
GO
CREATETABLE[dbo].[MTDATDOC](
	[COD_DOC][nvarchar](10)NOTNULL,
	[NOM_DOC][nvarchar](25)NULL,
	[APE_DOC][nvarchar](25)NULL,
	[NICK][nvarchar](20)NULL,
	[CLAVE][varbinary](500)NULL,
	[DIR_DOC][nvarchar](35)NULL,
	[TEL_DOC][nvarchar](10)NULL,
	[CEL_DOC][nvarchar](10)NULL,
	[MAIL_DOC][nvarchar](30)NULL,
	[FEC_NAC_DOC][date]NULL,
	[AS_USUARIO][nvarchar](30)NULL,
	[AS_HORA][date]NULL,
	[HABILITADO][nvarchar](1)NULL,
	[CONECTADO][nvarchar](2)NULL,
```

```
PRIMARYKEYCLUSTERED
(
	[COD_DOC]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
SETANSI_PADDINGOFF
GO
/****** Object:  Table [dbo].[MTDATDAT]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
SETANSI_PADDINGOFF
GO
CREATETABLE[dbo].[MTDATDAT](
	[CODDAT][nvarchar](7)NOTNULL,
	[TIPDAT][varchar](2)NOTNULL,
	[NOLDAT][nvarchar](70)NOTNULL,
	[NOCDAT][nvarchar](40)NOTNULL,
	[AS_USUARIO][nvarchar](30)NOTNULL,
	[AS_FECHA][datetime]NOTNULL,
PRIMARYKEYCLUSTERED
(
	[CODDAT]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
SETANSI_PADDINGOFF
GO
/****** Object:  Table [dbo].[MTDATASIG]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATASIG](
	[CODASIG][int]IDENTITY(1,1)NOTNULL,
	[CODIPER][nvarchar](5)NULL,
	[CODIDOC][nvarchar](10)NULL,
	[CODIMAT][nvarchar](6)NULL,
	[NIVMAT][nvarchar](7)NULL,
	[PARMAT][nvarchar](7)NULL,
	[TIPMAT][nvarchar](7)NULL,
	[OBSASIG][nvarchar](500)NULL,
PRIMARYKEYCLUSTERED
(
	[CODASIG]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
```

```
/****** Object:  Table [dbo].[MTDATACU]    Script Date: 04/07/2015
02:01:24 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[MTDATACU](
	[CODACU][nvarchar](7)NOTNULL,
	[NOMACU][nvarchar](70)NOTNULL,
	[APEACU][nvarchar](70)NOTNULL,
	[TELACU][nvarchar](10)NOTNULL,
	[RELACU][nvarchar](7)NULL,
	[CODIEST][nvarchar](10)NULL,
	[TIPDATACU][nvarchar](2)NOTNULL,
PRIMARYKEYCLUSTERED
(
	[CODACU]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  UserDefinedFunction [dbo].[ENCRIPTAR]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEFUNCTION[dbo].[ENCRIPTAR]
(
@claveNVARCHAR(500)
)
RETURNSVARBINARY(500)
AS
BEGIN
	declare@passasvarbinary(500)
	set@pass=encryptbypassphrase('clave',@clave)
	return@pass
END
GO
/****** Object:  UserDefinedFunction [dbo].[DESENCRIPTAR]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEFUNCTION[dbo].[DESENCRIPTAR]
(
@claveVARBINARY(500)
)
RETURNSNVARCHAR(500)
AS
BEGIN
	declare@passasNVARCHAR(500)
	set@pass=decryptbypassphrase('clave',@clave)
	return@pass
END
GO
```

```
/****** Object:  Table [dbo].[AUDITORIA]    Script Date: 04/07/2015
02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATETABLE[dbo].[AUDITORIA](
	[AUDIID][nvarchar](6)NOTNULL,
	[AUDIFECHAHORA][datetime]NULL,
	[AUDIIP][nvarchar](30)NULL,
	[AUDIUSUARIO][nvarchar](50)NULL,
	[AUDIMOVIMIENTO][nvarchar](1)NULL,
	[AUDIMOVTABLA][nvarchar](30)NULL,
PRIMARYKEYCLUSTERED
(
	[AUDIID]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
/****** Object:  Table [dbo].[USUARIO]    Script Date: 04/07/2015
02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
SETANSI_PADDINGOFF
GO
CREATETABLE[dbo].[USUARIO](
	[CODUSU][nvarchar](10)NOTNULL,
	[NOMUSU][nvarchar](50)NULL,
	[PASUSU][varbinary](500)NULL,
	[CONECTADO][nvarchar](2)NULL,
	[ROLUSU][nvarchar](7)NULL,
PRIMARYKEYCLUSTERED
(
	[CODUSU]ASC
)WITH
(PAD_INDEX=OFF,STATISTICS_NORECOMPUTE=OFF,IGNORE_DUP_KEY=OFF,ALLOW_R
OW_LOCKS=ON,ALLOW_PAGE_LOCKS=ON)ON[PRIMARY]
)ON[PRIMARY]
GO
SETANSI_PADDINGOFF
GO
/****** Object:  StoredProcedure [dbo].[spsp_codsec]    Script Date:
04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEProcedure[dbo].[spsp_codsec](--DROP Procedure spnp_codsec
	@i_TABLAS   varchar(10),
	@o_CODSEC   integerOutPut
)As
Declare@p_STRING  varchar(200)
Declare@p_CODSEC  integer
```

```
Select@p_STRING=' Declare Rst_CodSec Cursor For Select
IsNull(Max(CODSEC),0) AS CODSEC From '+@i_TABLAS
Exec (@p_STRING)
OPENRst_CodSec
FETCHNEXTFROMRst_CodSecInto@p_CODSEC
IF@@FETCH_STATUS<>-1
      Select@o_CODSEC=@p_CODSEC+ 1
ELSE
      Select@o_CODSEC= 1
CloseRst_CodSec
DeallocateRst_CodSec
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATNOT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATNOT]
@OPNVARCHAR(1),
@DATONVARCHAR(100)
AS
BEGIN
      IF (@OP=1)
      BEGIN

      SELECTN.CODNOT,N.CODIASIG,N.CODIEST,N.NOT1NOT,N.NOT2NOT,N.NOT3
NOT,N.PRONOT,
            N.EXANOT,N.NOTNOT,N.EQUINOT,
            (SELECT (E.APEEST+'
'+E.NOMEST)FROMMTDATESTEWHEREE.CODEST=N.CODIEST)ASESTUDIANTE
            FROMMTDATNOTN
            WHERECODIASIG=@DATO
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATMAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIEROFF
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATMAT]
@OPNVARCHAR(1),
@DATONVARCHAR(70)
AS
BEGIN
      IF(@OP=1)
      BEGIN

      SELECTCODMAT,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORAFROMMTDA
TMAT
      END
      IF(@OP=2)--POR CODIGO
      BEGIN

      SELECTCODMAT,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORAFROMMTDA
TMAT
```

```
            WHERECODMATLIKE'%'+@DATO+'%'
      END

      IF(@OP=3)--POR NOMBRE DE LA MATERIA
      BEGIN

      SELECTCODMAT,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORAFROMMTDA
TMAT
            WHERENOMLMATLIKE'%'+@DATO+'%'
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATFAM]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATFAM]
@OPNVARCHAR(1),
@CEDESTNVARCHAR(10),
@TIPONVARCHAR(2)
AS
BEGIN
      IF(@OP=1)--BUSQUEDA POR CEDULA Y TIPO
      BEGIN
            select*fromMTDATFAMwhereCODIEST=@CEDESTandTIPFAM=@TIPO
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATEST]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATEST]
@OPNVARCHAR(1),
@DATONVARCHAR(100)
AS
BEGIN
      IF(@OP=1)--BUSQUEDA GENERAL
      BEGIN
                  SELECTCODEST,NOMEST,APEEST,FECNACEST,NACEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE       (CODDAT=E.NACEST))ASNACIONALIDAD,POSCEDEST,CEDEST,SEXEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE
(CODDAT=E.SEXEST))ASGENERO,CMEDEST,CDENEST,CCEDPEST,CCEDMEST,PNACEST
,FOTO,PCEDEST,NUMMATEST,CODMINEDU,
FECMATEST,(APEEST+' '+NOMEST)ASNOMBRES
                  FROMdbo.MTDATESTASE
      END
      IF(@OP=2)--BUSQUEDA POR CEDULA
      BEGIN
                  SELECTCODEST,NOMEST,APEEST,FECNACEST,NACEST,
```

```
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE       (CODDAT=E.NACEST))ASNACIONALIDAD,POSCEDEST,CEDEST,SEXEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE
(CODDAT=E.SEXEST))ASGENERO,CMEDEST,CDENEST,CCEDPEST,CCEDMEST,PNACEST
,FOTO,PCEDEST,NUMMATEST,CODMINEDU,
FECMATEST,(APEEST+' '+NOMEST)ASNOMBRES
                  FROMdbo.MTDATESTASEWHERECEDESTLIKE'%'+@DATO+'%'
      END

            IF(@OP=3)--BUSQUEDA POR CEDULA
      BEGIN
                  SELECTCODEST,NOMEST,APEEST,FECNACEST,NACEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE       (CODDAT=E.NACEST))ASNACIONALIDAD,POSCEDEST,CEDEST,SEXEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE
(CODDAT=E.SEXEST))ASGENERO,CMEDEST,CDENEST,CCEDPEST,CCEDMEST,PNACEST
,FOTO,PCEDEST,NUMMATEST,CODMINEDU,
FECMATEST,(APEEST+' '+NOMEST)ASNOMBRES
                  FROMdbo.MTDATESTASEWHEREAPEESTLIKE'%'+@DATO+'%'
      END
                  IF(@OP=4)--BUSQUEDA POR CODIGO
      BEGIN
                  SELECTCODEST,NOMEST,APEEST,FECNACEST,NACEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE       (CODDAT=E.NACEST))ASNACIONALIDAD,POSCEDEST,CEDEST,SEXEST,
(SELECTNOLDAT
FROMdbo.MTDATDAT
WHERE
(CODDAT=E.SEXEST))ASGENERO,CMEDEST,CDENEST,CCEDPEST,CCEDMEST,PNACEST
,FOTO,PCEDEST,NUMMATEST,CODMINEDU,
FECMATEST,(APEEST+' '+NOMEST)ASNOMBRES
                  FROMdbo.MTDATESTASEWHERECODESTLIKE'%'+@DATO+'%'
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATDOC]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROC[dbo].[SPSB_MTDATDOC]
@OPNVARCHAR(1),
@DATONVARCHAR(100),
@DATOCNVARCHAR(100)
AS
BEGIN
declare@clavevarbinary
set@clave=CONVERT(VARBINARY(MAX),@DATOC)
      IF(@OP=1)--BUSQUEDA GENERAL
      BEGIN
```

```
	SELECTCOD_DOC,NOM_DOC,APE_DOC,NICK,dbo.DESENCRIPTAR(CLAVE),DIR
_DOC,TEL_DOC,CEL_DOC,MAIL_DOC,FEC_NAC_DOC,AS_USUARIO,AS_HORA,HABILIT
ADO,CONECTADO,dbo.DESENCRIPTAR(CLAVE)ASCLAVED,(APE_DOC+'
'+NOM_DOC)ASNOMBRESFROMMTDATDOC
	END
	IF(@OP=2)-- BUSQUEDA POR OPCIONES CODIGO/ APELLIDOS/ NOMBRES/
TELEFONO/ CELULAR/
	BEGIN

	SELECTCOD_DOC,NOM_DOC,APE_DOC,NICK,dbo.DESENCRIPTAR(CLAVE),DIR
_DOC,TEL_DOC,CEL_DOC,MAIL_DOC,FEC_NAC_DOC,AS_USUARIO,AS_HORA,HABILIT
ADO,CONECTADO,dbo.DESENCRIPTAR(CLAVE)ASCLAVED,(APE_DOC+'
'+NOM_DOC)ASNOMBRESFROMMTDATDOC

	WHERECOD_DOCLIKE'%'+@DATO+'%'ORAPE_DOCLIKE'%'+@DATO+'%'ORTEL_D
OCLIKE'%'+@DATO+'%'ORCEL_DOCLIKE'%'+@DATO+'%'ORNOM_DOCLIKE'%'+@DATO+
'%'
	END
	/*IF(@OP=2) -- BUSQUEDA POR OPCIONES CODIGO/ APELLIDOS/
NOMBRES/ TELEFONO/ CELULAR/
	BEGIN
		SELECT COD_DOC, NOM_DOC, APE_DOC, NICK,
dbo.DESENCRIPTAR(CLAVE), DIR_DOC, TEL_DOC, CEL_DOC, MAIL_DOC,
FEC_NAC_DOC, AS_USUARIO, AS_HORA, HABILITADO, CONECTADO,
dbo.DESENCRIPTAR(CLAVE) AS CLAVED , (APE_DOC + ' ' + NOM_DOC) AS
NOMBRES FROM MTDATDOC
		WHERE COD_DOC LIKE '%' + @DATO + '%'  OR APE_DOC LIKE
'%' + @DATO + '%' OR TEL_DOC LIKE '%' + @DATO + '%' OR CEL_DOC LIKE
'%' + @DATO + '%' OR NOM_DOC LIKE '%' + @DATO + '%'
	END*/

	IF(@OP=3)-- BUSQUEDA POR NICK Y CLAVE
	BEGIN

	SELECTCOD_DOC,NOM_DOC,APE_DOC,NICK,dbo.DESENCRIPTAR(CLAVE),DIR
_DOC,TEL_DOC,CEL_DOC,MAIL_DOC,FEC_NAC_DOC,AS_USUARIO,AS_HORA,HABILIT
ADO,CONECTADO,dbo.DESENCRIPTAR(CLAVE)ASCLAVED,(APE_DOC+'
'+NOM_DOC)ASNOMBRESFROMMTDATDOC
		WHERENICK=@DATO--AND CLAVE = dbo.DESENCRIPTAR(@clave)
	END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATDAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATDAT]
@iOPnvarchar(1),
@iDATOnvarchar(50)
AS
BEGIN
	IF@iOP=1
	BEGIN
		SELECT*FROMMTDATDAT
	END
```

```
	IF@iOP=2
	BEGIN
		SELECT*FROMMTDATDAT
		WHERECODDATLIKE'%'+@iDATO+'%'
	END
	IF@iOP=3
	BEGIN
		SELECT*FROMMTDATDAT
		WHERETIPDATLIKE'%'+@iDATO+'%'
	END
	IF@iOP=4
	BEGIN
		SELECT*FROMMTDATDAT
		WHERENOLDATLIKE'%'+@iDATO+'%'
	END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATASIG]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATASIG]
@OPNVARCHAR(1),
@DATONVARCHAR(100),
@DATO1NVARCHAR(100)
AS
BEGIN

	IF (@OP=1)
	BEGIN

	SELECTCODASIG,CODIPER,CODIDOC,CODIMAT,NIVMAT,PARMAT,TIPMAT,

	(SELECTP.NOML_SEMFROMMTDATPERPWHEREP.COD_SEM=N.CODIPER)ASPERIO
DO,
		(SELECT (D.APE_DOC+'
'+D.NOM_DOC)FROMMTDATDOCDWHERED.COD_DOC=N.CODIDOC)ASDOCENTE,

	(SELECTNOMLMATFROMMTDATMATMWHEREM.CODMAT=N.CODIMAT)ASASIGNATUR
A,


	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.NIVMAT)ASNIVEL,

	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.PARMAT)ASPARALELO,


	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.TIPMAT)ASTIPO_MATE
RIA,
		OBSASIG
		FROMMTDATASIGN
		WHERECODIPER=@DATO

	END

	IF (@OP=2)
```

```
	BEGIN

	SELECTCODASIG,CODIPER,CODIDOC,CODIMAT,NIVMAT,PARMAT,TIPMAT,

	(SELECTP.NOML_SEMFROMMTDATPERPWHEREP.COD_SEM=N.CODIPER)ASPERIO
DO,
		(SELECT (D.APE_DOC+'
'+D.NOM_DOC)FROMMTDATDOCDWHERED.COD_DOC=N.CODIDOC)ASDOCENTE,

	(SELECTNOMLMATFROMMTDATMATMWHEREM.CODMAT=N.CODIMAT)ASASIGNATUR
A,


	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.NIVMAT)ASNIVEL,

	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.PARMAT)ASPARALELO,


	(SELECTS.NOLDATFROMMTDATDATSWHERES.CODDAT=N.TIPMAT)ASTIPO_MATE
RIA,
		OBSASIG
		FROMMTDATASIGN
		WHERECODIPER=@DATOANDCODIDOC=@DATO1
	END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATACU]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATACU]
@OPNVARCHAR(1),
@CEDESTNVARCHAR(10),
@TIPONVARCHAR(2)
AS
BEGIN
	IF(@OP=1)--BUSQUEDA POR CEDULA Y TIPO
	BEGIN

	select*frommtdatacuwhereCODIEST=@CEDESTandTIPDATACU=@TIPO
	END

END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_USUARIO]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_USUARIO]
@CODIGONVARCHAR(10),
@NOMBRENVARCHAR(50),
@PASSNVARCHAR(50),
@CONECTADONVARCHAR(2),
@ROLNVARCHAR(6)
```

```
AS
BEGINTRANSACTION
      IFEXISTS(select*fromUSUARIOWHERECODUSU=@CODIGO)
      BEGIN
            UPDATEUSUARIOSET
            NOMUSU=@NOMBRE,
            PASUSU=dbo.encriptar(@PASS),
            CONECTADO=@CONECTADO,
            ROLUSU=@ROL

            WHERECODUSU=@CODIGO
            If@@error!=0
                                          Begin

      RollBackTransaction
                                                      Select'Error
Actualización de Registro'
                                          End
                                                Else
                                                Begin

      Select'Actualización Realizada con Exito'
                                                CommitTransaction
                                          End
      END
      ELSE
      BEGIN
            INSERTINTOUSUARIO(CODUSU,NOMUSU,PASUSU,ROLUSU,CONECTADO)

      VALUES(@CODIGO,@NOMBRE,dbo.encriptar(@PASS),@ROL,@CONECTADO)
                  If@@error!=0
                  Begin
                        RollBackTransaction
                        Select'Ocurrio un Error al Insertar el
Registro'
                  End
                  Else
                  Begin
                        Select'Registro Insertado Correctamente'
                        CommitTransaction
                  End
      END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_SECUENCIAL]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_SECUENCIAL]
@CODPARNVARCHAR(10),
@PARSECNVARCHAR(10),
@CADPARNVARCHAR(3)
AS
BEGINTRANSACTION

IFEXISTS(SELECT 1 FROMSECUENCIALWHERECODPAR=@CODPAR)
            BEGIN
```

```
      UPDATESECUENCIALSET
      PARSEC=@PARSEC,
      CADPAR=@CADPAR
      WHERECODPAR=@CODPAR
            If@@error!=0
                                                Begin

      RollBackTransaction

                                          End
                                          Else
                                          Begin


      CommitTransaction
                                          end
END
            ELSE
            BEGIN

      InsertIntoSECUENCIAL(CODPAR,PARSEC,CADPAR)
                              VALUES(@CODPAR,@PARSEC,@CADPAR)
                              If@@error!=0
                              Begin
                                    RollBackTransaction

                              End
                              Else
                              Begin

                                    CommitTransaction
                              End
            END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_REPORTE]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
createPROC[dbo].[SPSA_REPORTE]
@CODREPNVARCHAR(70),
@PATHREPNVARCHAR(200),
@NOMREPNVARCHAR(100)
AS
BEGINTRANSACTION

IFEXISTS(SELECT*fromREPORTEWHERECODREP=@CODREP)
BEGIN
                  UPDATEREPORTESET
                  PATHREP=@PATHREP,
                        NOMREPORT=@NOMREP

                        WHERECODREP=@CODREP
                        If@@error!=0
                              Begin
                              RollBackTransaction
```

```
                              select'Error Actualización de
Registro'
--                            Select iRetCode = -100,  sErrMsg =
'Error Actualización de Registro'
                        End
                        Else
                        Begin
                              --Select iRetCode = 0, sErrMsg =
                              select'Actualización Realizada con
Exito'
                              CommitTransaction
                        End
END
ELSE
BEGIN
      INSERTINTOREPORTE(CODREP,NOMREPORT,PATHREP)
      VALUES (@CODREP,@NOMREP,@NOMREP)
                        If@@error!=0
                  Begin
                        RollBackTransaction
                        --Select iRetCode = -100,  sErrMsg =
                        select'Ocurrio un Error al Insertar el
Registro'
                  End
                  Else
                  Begin
                        --Select iRetCode = 1, sErrMsg=
                        select'Registro Insertado Correctamente'
                        CommitTransaction
                  End


END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATPER]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATPER]

      @CODSEM     nvarchar(5),      -- Codigo
      @NOMLSEM    nvarchar(50),     -- Nombre Largo
      @NOMCSEM    nvarchar(20),     -- Nombre Corto
      @OBSSEMnvarchar(1000),
      @AS_USUARIO varchar(30),
      @ESTADOnvarchar(1),
      @FECINIDATE,
      @FECCULDATE
AS
BEGINTRANSACTION
declare
@p_parsecNVARCHAR(10),
@p_cadparNVARCHAR(3),
@p_CODSEMNVARCHAR(10)

IFEXISTS(SELECT 1 FROMMTDATPERWHERECOD_SEM=@CODSEM)
```

```
            BEGIN
      UPDATEMTDATPERSET
      NOML_SEM=@NOMLSEM,
      NOMC_SEM=@NOMCSEM,
      OBSER_CAR=@OBSSEM,
      AS_USUARIO=@AS_USUARIO,
      AS_HORA=GETDATE(),
      ESTADO=@ESTADO,
      FECINI=@FECINI,
      FECCUL=@FECCUL
      WHERECOD_SEM=@CODSEM
            If@@error!=0
                                                Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                                    End
                                    Else
                                    Begin
                                                      SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'

      CommitTransaction
                                    end
END
            ELSE
            BEGIN
            select@p_parsec=parsec,@p_cadpar=cadpar
                        fromsecuencialwherecodpar='PERIODO'
                        set@p_CODSEM=@p_cadpar+@p_parsec


      InsertIntoMTDATPER(COD_SEM,NOML_SEM,NOMC_SEM,OBSER_CAR,AS_USUA
RIO,AS_HORA,ESTADO,FECINI,FECCUL)

      VALUES(@p_CODSEM,@NOMLSEM,@NOMCSEM,@OBSSEM,@AS_USUARIO,GETDATE
(),@ESTADO,@FECINI,@FECCUL)

                        updatesecuencialsetparsec=@p_parsec+ 1
wherecodpar='PERIODO'

                              If@@error!=0
                              Begin
                                    RollBackTransaction
                                    SelectiRetCode=-
100,sErrMsg='Error Insercion de Registro'
                              End
                              Else
                              Begin
                                    SelectiRetCode=
1,sErrMsg='Registro Insertado Correctamente'
                                    CommitTransaction
                              End
            END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATNOTAS]    Script
Date: 04/07/2015 02:01:25 ******/
```

```
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATNOTAS]
      @CODNOTint,
      @NOT1NOTdecimal(10,2),
      @NOT2NOTdecimal(10,2),
      @NOT3NOTdecimal(10,2),
      @EXANOTdecimal(10,2),
      @EQUINOTdecimal(10,2)

AS
DECLARE@PRONOTdecimal(10,2)
DECLARE@NOTNOTdecimal(10,2)
SET@PRONOT=(@NOT1NOT+@NOT2NOT+@NOT3NOT)/3
SET@NOTNOT=(@PRONOT+@EXANOT)/2
BEGINTRANSACTION
BEGIN
                                    UPDATEMTDATNOTSET

                                          NOT1NOT=@NOT1NOT,
                                          NOT2NOT=@NOT2NOT,
                                          NOT3NOT=@NOT3NOT,
                                          PRONOT=@PRONOT,
                                          EXANOT=@EXANOT,
                                          NOTNOT=@NOTNOT,
                                          EQUINOT=@EQUINOT

                                    WHERECODNOT=@CODNOT

                                    If@@error!=0
                                                Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                                    End
                                    Else
                                    Begin
                                                      SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'


      CommitTransaction

                                    end
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATNOT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATNOT]
      @CODNOTint,
```

```
      @CODIASIGINT,
      @CODIESTNVARCHAR(10),
      @NOT1NOTdecimal(10,2),
      @NOT2NOTdecimal(10,2),
      @NOT3NOTdecimal(10,2),
      @PRONOTdecimal(10,2),
      @EXANOTdecimal(10,2),
      @NOTNOTdecimal(10,2),
      @EQUINOTdecimal(10,2)

AS
BEGINTRANSACTION
BEGIN
IFNOTEXISTS(SELECT 1 FROMMTDATNOTWHERECODNOT=@CODNOT)

            BEGIN

                              --Insert Into
MTDATMAT(CODSEC,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORA)

      InsertIntoMTDATNOT(CODIASIG,CODIEST,NOT1NOT,NOT2NOT,NOT3NOT,PR
ONOT,EXANOT,NOTNOT,EQUINOT)
                              Values
(@CODIASIG,@CODIEST,@NOT1NOT,@NOT2NOT,@NOT3NOT,@PRONOT,@EXANOT,@NOTN
OT,@EQUINOT)

                              If@@error!=0
                                    Begin
                                    RollBackTransaction
                                    SelectiRetCode=-
100,sErrMsg='Registro Insertado Incorrectamente'
                              End
                              Begin
                                    SelectiRetCode=
1,sErrMsg='Registro Insertado Correctamente'
                                    CommitTransaction
                              End
            END
            ELSE
            BEGIN
                                    UPDATEMTDATNOTSET
                                          CODIASIG=@CODIASIG,
                                          CODIEST=@CODIEST,
                                          NOT1NOT=@NOT1NOT,
                                          NOT2NOT=@NOT2NOT,
                                          NOT3NOT=@NOT3NOT,
                                          PRONOT=@PRONOT,
                                          EXANOT=@EXANOT,
                                          NOTNOT=@NOTNOT,
                                          EQUINOT=@EQUINOT

                                    WHERECODNOT=@CODNOT

                                    If@@error!=0
                                                Begin

      RollBackTransaction
```

```
      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                                    End
                                    Else
                                    Begin
                                                SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'


      CommitTransaction

                                    end
            END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATMAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIEROFF
GO

CREATEPROCEDURE[dbo].[SPSA_MTDATMAT]

      @i_CODMAT   nvarchar(6),      -- Codigo
      @i_NOMLMAT  nvarchar(70),     -- Nombre Largo
      @i_NOMCMAT  nvarchar(50),     -- Nombre Corto
      @i_OBSMATnvarchar(1000),
      @i_AS_USUARIO     varchar(8)
AS
DECLARE
@p_parsecVARCHAR(10),
@p_cadparVARCHAR(3),
@p_valmatVARCHAR(10)
Declare@p_CODSECinteger
BEGINTRANSACTION

BEGIN
IFNOTEXISTS(SELECT 1 FROMMTDATMATWHERECODMAT=@i_CODMAT)

            BEGIN
            --exec spsp_codsec 'MTDATMAT',@p_CODSEC OutPut

      select@p_parsec=parsec,@p_cadpar=CADPARfromSECUENCIALwherecodp
ar='MTDATMAT'
                  set@p_valmat=@p_cadpar+@p_parsec

                              --Insert Into
MTDATMAT(CODSEC,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORA)

      InsertIntoMTDATMAT(CODMAT,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS
_HORA)
                              Values
(@p_valmat,@i_NOMLMAT,@i_NOMCMAT,@i_OBSMAT,@i_AS_USUARIO,GETDATE())

                              updateSECUENCIALsetparsec=@p_parsec+ 1
wherecodpar='MTDATMAT'
```

```
                              If@@error!=0
                                    Begin
                                    RollBackTransaction
                                    SelectiRetCode=-
100,sErrMsg='Error Insercion de Registro'
                              End
                              Begin
                                    SelectiRetCode=
1,sErrMsg='Registro Insertado Correctamente'
                                    CommitTransaction
                              End
            END

            ELSE


            BEGIN
                  UpdateMTDATMATSet
                  CODMAT=     @i_CODMAT,
                  NOMLMAT=    @i_NOMLMAT,
                  NOMCMAT=    @i_NOMCMAT,
                  OBSMAT            =     @i_OBSMAT,
                  AS_USUARIO  =     @i_AS_USUARIO,
                  AS_HORA     =     GETDATE()
                  WhereCODMAT =     @i_CODMAT
                                    If@@error!=0
                                          Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                                    End
                                    Else
                                    Begin
                                                SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'


      CommitTransaction

                                    end
            END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATFAM]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATFAM]
@CODFAMNVARCHAR(7),
@NOMFAMNVARCHAR(70),
@APEFAMNVARCHAR(70),
@NIVFAMNVARCHAR(7),--NIVEL DE EDUCACION
@OCUFAMNVARCHAR(7),--OCUPACION
@AREFAMNVARCHAR(7),--AREA DE OCUPACION
@TELFAMNVARCHAR(10),
```

```
@CELFAMNVARCHAR(10),
@DIRFAMNVARCHAR(250),
@VIVFAMNVARCHAR(1),
@REPFAMNVARCHAR(1),
@AUTFAMNVARCHAR(1),
@CODIESTNVARCHAR(10),
@TIPFAMNVARCHAR(1)

AS
BEGIN
declare
@p_parsecNVARCHAR(10),
@p_cadparNVARCHAR(3),
@p_CODFAMNVARCHAR(10)



                  IFEXISTS(SELECT 1
FROMSECUENCIALWHERECODPAR='FAMILIA')
                  BEGIN

      IFNOTEXISTS(SELECT 1 FROMMTDATFAMWHERECODFAM=@CODFAM)
      BEGIN

      select@p_parsec=parsec,@p_cadpar=cadpar
                        fromsecuencialwherecodpar='FAMILIA'
                        set@p_CODFAM=@p_cadpar+@p_parsec


      INSERTINTOMTDATFAM(CODFAM,NOMFAM,APEFAM,NIVFAM,OCUFAM,AREFAM,T
ELFAM,CELFAM,DIRFAM,VIVFAM,REPFAM,AUTFAM,CODIEST,TIPFAM)

      VALUES(@p_CODFAM,@NOMFAM,@APEFAM,@NIVFAM,@OCUFAM,@AREFAM,@TELF
AM,@CELFAM,@DIRFAM,@VIVFAM,@REPFAM,@AUTFAM,@CODIEST,@TIPFAM)

                        updatesecuencialsetparsec=@p_parsec+ 1
wherecodpar='FAMILIA'

If@@error!=0
                              Begin--3
                                    Select'Existe un error al
insertar'
                              End--3
                              else
                              Begin--3
                                    select'Registro Insertado'
                              end--3
            END--2
            ELSE
            BEGIN--2
                        select'No existe numeración para la tabla
(MTDATFAM)'
            END--2
      END
      ELSE
      BEGIN

                  UPDATEMTDATFAMSET
```

```
                  NOMFAM=@NOMFAM,
                  APEFAM=@APEFAM,
                  NIVFAM=@NIVFAM,
                  OCUFAM=@OCUFAM,
                  AREFAM=@AREFAM,
                  TELFAM=@TELFAM,
                  CELFAM=@CELFAM,
                  DIRFAM=@DIRFAM,
                  VIVFAM=@VIVFAM,
                  REPFAM=@REPFAM,
                  AUTFAM=@AUTFAM,
                  CODIEST=@CODIEST,
                  TIPFAM=@TIPFAM
                  WHERECODFAM=@CODFAM

                  If@@error!=0
                  Begin--3
                        Select'Existe un error al actualizar'
                  End--3
                  else
                  Begin--3
                        select'Registro Actualizado'
                  end--3
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATEST]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
/*****************************************************************
'*****************************************************************
*** */
CREATEPROCEDURE[dbo].[SPSA_MTDATEST]
      @CODESTNVARCHAR(10),
      @NOMESTNVARCHAR(70),
      @APEESTNVARCHAR(70),
      @FECNACESTDATETIME,
      @NACESTNVARCHAR(7),--NACIONALIDAD
      @POSCEDESTNVARCHAR(1),
      @CEDESTNVARCHAR(10),
      @SEXESTNVARCHAR(7),--SEXO DEL ESTUDIANTE
      @CMEDESTNVARCHAR(1),--CERTIFICADO MEDICO
      @CDENESTNVARCHAR(1),--CERTIFICADO DENTAL
      @CCEDPESTNVARCHAR(1),--COPIA DE CEDULA DEL PADRE
      @CCEDMESTNVARCHAR(1),--COPIA DE CEDULA DE LA MADRE
      @PNACESTNVARCHAR(1),--PARTIDA DE NACIMIENTO DEL ESTUDIANTE
      @FOTONVARCHAR(100),
      @PCEDESTNVARCHAR(1),--SI DEJA CEDULA
      @NUMMATESTNVARCHAR(10),-- NUMERO DE MATRICULA
      @CODMINEDUNVARCHAR(10),-- CODIGO MINISTERIO DE EDUCACION
      @FECMATESTDATETIME
      AS
      BEGINTRANSACTION
            IFEXISTS(SELECT 1 FROMMTDATESTWHERECODEST=@CODEST)
                  BEGIN
```

```
                        UPDATEMTDATESTSET
                              CODEST=@CODEST,
                              NOMEST=@NOMEST,
                              APEEST=@APEEST,
                              FECNACEST=@FECNACEST,
                              NACEST=@NACEST,--NACIONALIDAD
                              POSCEDEST=@POSCEDEST,
                              CEDEST=@CEDEST,
                              SEXEST=@SEXEST,--SEXO DEL ESTUDIANTE
                              CMEDEST=@CMEDEST,--CERTIFICADO MEDICO
                              CDENEST=@CDENEST,--CERTIFICADO DENTAL
                              CCEDPEST=@CCEDPEST,--COPIA DE CEDULA
DEL PADRE
                              CCEDMEST=@CCEDMEST,--COPIA DE CEDULA
DE LA MADRE
                              PNACEST=@PNACEST,--PARTIDA DE
NACIMIENTO DEL ESTUDIANTE
                              FOTO=@FOTO,
                              PCEDEST=@PCEDEST,--SI DEJA CEDULA
                              NUMMATEST=@NUMMATEST,-- NUMERO DE
MATRICULA
                              CODMINEDU=@CODMINEDU,-- CODIGO
MINISTERIO DE EDUCACION
                              FECMATEST=@FECMATEST
                        WHERECODEST=@CODEST
                              If@@error!=0
                              Begin
                                    RollBackTransaction
                                    Select'Error al Actualizar la
informacion'
                              End
                              Else
                              Begin
                                    Select'Actualizacion realizada
con Exito'
                                    CommitTransaction
                              End
                  END
                  ELSE
                  BEGIN

      INSERTINTOMTDATEST(CODEST,NOMEST,APEEST,FECNACEST,NACEST,POSCE
DEST,CEDEST,SEXEST,CMEDEST,CDENEST,CCEDPEST,CCEDMEST,PNACEST,FOTO,PC
EDEST,NUMMATEST,CODMINEDU,FECMATEST)

      VALUES(@CODEST,@NOMEST,@APEEST,@FECNACEST,@NACEST,@POSCEDEST,@
CEDEST,@SEXEST,@CMEDEST,@CDENEST,@CCEDPEST,@CCEDMEST,@PNACEST,@FOTO,
@PCEDEST,@NUMMATEST,@CODMINEDU,@FECMATEST)
                        If@@error!=0
                              Begin
                                    RollBackTransaction
                                    Select'Error al Insertar la
informacion'
                              End
                              Else
                              Begin
                                    Select'Insercion Realizada con
Exito'
```

```
                                    CommitTransaction
                                End
                            END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATDOC]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATDOC]
      @COD_DOCnvarchar(10),
      @NOM_DOCnvarchar(25),
      @APE_DOCnvarchar(25),
      @NICKnvarchar(20),
      @CLAVENVARCHAR (500),
      @DIR_DOCnvarchar(35),
      @TEL_DOCnvarchar(10),
      @CEL_DOCnvarchar(10),
      @MAIL_DOCnvarchar(30),
      @FEC_NAC_DOCdate,
      @AS_USUARIOnvarchar(30),
      @HABILITADOnvarchar(1),
      @CONECTADOnvarchar(2)
AS
BEGINTRANSACTION
BEGIN
IFNOTEXISTS(SELECT 1 FROMMTDATDOCWHERECOD_DOC=@COD_DOC)

            BEGIN

                                --Insert Into
MTDATMAT(CODSEC,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORA)

      InsertIntoMTDATDOC(COD_DOC,NOM_DOC,APE_DOC,NICK,CLAVE,DIR_DOC,
TEL_DOC,CEL_DOC,MAIL_DOC,FEC_NAC_DOC,AS_USUARIO,AS_HORA,HABILITADO,C
ONECTADO)
                                Values
(@COD_DOC,@NOM_DOC,@APE_DOC,@NICK,dbo.encriptar(@CLAVE),@DIR_DOC,@TE
L_DOC,@CEL_DOC,@MAIL_DOC,@FEC_NAC_DOC,@AS_USUARIO,GETDATE(),@HABILIT
ADO,@CONECTADO)

                                If@@error!=0
                                      Begin
                                      RollBackTransaction
                                      SelectiRetCode=-
100,sErrMsg='Registro Insertado Incorrectamente'
                                End
                                Begin
                                      SelectiRetCode=
1,sErrMsg='Registro Insertado Correctamente'
                                      CommitTransaction
                                End

            END

            ELSE
```

```
            BEGIN
                  UpdateMTDATDOCSet
                  COD_DOC=@COD_DOC,
                  NOM_DOC=@NOM_DOC,
                  APE_DOC=@APE_DOC,
                  NICK=@NICK,
                  CLAVE=dbo.encriptar(@CLAVE),
                  DIR_DOC=@DIR_DOC,
                  TEL_DOC=@TEL_DOC,
                  CEL_DOC=@CEL_DOC,
                  MAIL_DOC=@MAIL_DOC,
                  FEC_NAC_DOC=@FEC_NAC_DOC,
                  AS_USUARIO  =     @AS_USUARIO,
                  AS_HORA     =     GETDATE(),
                  HABILITADO=@HABILITADO,
                  CONECTADO=@CONECTADO
                  WhereCOD_DOC=@COD_DOC
                              If@@error!=0
                                          Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                              End
                              Else
                              Begin
                                                SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'


      CommitTransaction

                              end
            END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATDAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATDAT]
      @i_CODDAT         nvarchar(6),
      @i_TIPDAT         nvarchar(3),
      @i_NOLDAT         nvarchar(50),
      @i_NOCDAT         nvarchar(30),
      @i_AS_USUARIO     nvarchar(8)
AS
BEGINTRANSACTION
IFEXISTS(SELECT*FROMMTDATDATWHERECODDAT=@i_CODDAT)
BEGIN
                  UPDATEMTDATDATSET
                        TIPDAT=@i_TIPDAT,
                        NOLDAT=@i_NOLDAT,
                        NOCDAT=@i_NOCDAT,
                        AS_USUARIO=@i_AS_USUARIO,
```

```
                        AS_FECHA=GETDATE()
                        WHERECODDAT=@i_CODDATANDTIPDAT=@i_TIPDAT
                        If@@error!=0
                              Begin
                              RollBackTransaction
                              select'Error Actualización de
Registro'
--                            Select iRetCode = -100,  sErrMsg =
'Error Actualización de Registro'
                        End
                        Else
                        Begin
                              --Select iRetCode = 0, sErrMsg =
                              select'Actualización Realizada con
Exito'
                              CommitTransaction
                        End
END
ELSE
BEGIN
      INSERTINTOMTDATDAT(CODDAT,TIPDAT,NOLDAT,NOCDAT,AS_USUARIO,AS_F
ECHA)
      VALUES
(@i_CODDAT,@i_TIPDAT,@i_NOLDAT,@i_NOCDAT,@i_AS_USUARIO,GETDATE())
                        If@@error!=0
                  Begin
                        RollBackTransaction
                        --Select iRetCode = -100,  sErrMsg =
                        select'Ocurrio un Error al Insertar el
Registro'
                  End
                  Else
                  Begin
                        --Select iRetCode = 1, sErrMsg=
                        select'Registro Insertado Correctamente'
                        CommitTransaction
                  End
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATASIG]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATASIG]
      @CODASIGint,
      @CODIPERNVARCHAR(5),
      @CODIDOCNVARCHAR(10),
      @CODIMATNVARCHAR(6),
      @NIVMATNVARCHAR(7),
      @PARMATNVARCHAR(7),
      @TIPMATNVARCHAR(7),
      @OBSASIGNVARCHAR(500)
AS
BEGINTRANSACTION
BEGIN
IFNOTEXISTS(SELECT 1 FROMMTDATASIGWHERECODASIG=@CODASIG)
```

```
            BEGIN

                                --Insert Into
MTDATMAT(CODSEC,NOMLMAT,NOMCMAT,OBSMAT,AS_USUARIO,AS_HORA)

      InsertIntoMTDATASIG(CODIMAT,CODIDOC,NIVMAT,PARMAT,CODIPER,TIPM
AT,OBSASIG)
                                Values
(@CODIMAT,@CODIDOC,@NIVMAT,@PARMAT,@CODIPER,@TIPMAT,@OBSASIG)

                                If@@error!=0
                                      Begin
                                      RollBackTransaction
                                      SelectiRetCode=-
100,sErrMsg='Registro Insertado Incorrectamente'
                                End
                                Begin
                                      SelectiRetCode=
1,sErrMsg='Registro Insertado Correctamente'
                                      CommitTransaction
                                End
            END
            ELSE
            BEGIN
                                      UPDATEMTDATASIGSET

                                            CODIMAT=@CODIMAT,
                                            CODIDOC=@CODIDOC,
                                            NIVMAT=@NIVMAT,
                                            PARMAT=@PARMAT,
                                            CODIPER=@CODIPER,

                                            TIPMAT=@TIPMAT,
                                            OBSASIG=@OBSASIG
                                      WHERECODASIG=@CODASIG

                                      If@@error!=0
                                                  Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error Actualizaciòn de Registro'
                                      End
                                      Else
                                      Begin
                                                        SelectiRetCode=
0,sErrMsg='Registro Modificado Correctamente'


      CommitTransaction

                                      end
            END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_MTDATACU]    Script
Date: 04/07/2015 02:01:25 ******/
```

```
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSA_MTDATACU]
@CODACUNVARCHAR(7),
@NOMACUNVARCHAR(70),
@APEACUNVARCHAR(70),
@TELACUNVARCHAR(10),
@RELACUNVARCHAR(7),--RELACION
@CODIESTNVARCHAR(10),--CODIGO DE ESTUDIANTE
@TIPDATACUNVARCHAR(2)
AS
BEGIN
declare
@p_parsecNVARCHAR(10),
@p_cadparNVARCHAR(3),
@p_CODACUNVARCHAR(10)
                  IFEXISTS(SELECT 1
FROMSECUENCIALWHERECODPAR='ACUDIR')
                  BEGIN

      IFNOTEXISTS(SELECT 1 FROMMTDATACUWHERECODACU=@CODACU)
      BEGIN

      select@p_parsec=parsec,@p_cadpar=cadpar
                        fromsecuencialwherecodpar='ACUDIR'
                        set@p_CODACU=@p_cadpar+@p_parsec


      INSERTINTOMTDATACU(CODACU,NOMACU,APEACU,TELACU,RELACU,CODIEST,
TIPDATACU)

      VALUES(@p_CODACU,@NOMACU,@APEACU,@TELACU,@RELACU,@CODIEST,@TIP
DATACU)

                        updatesecuencialsetparsec=@p_parsec+ 1
wherecodpar='ACUDIR'

If@@error!=0
                              Begin--3
                                    Select'Existe un error al
insertar'
                              End--3
                              else
                              Begin--3
                                    select'Registro Insertado'
                              end--3
            END--2
            ELSE
            BEGIN--2
                        select'No existe numeración para la tabla
(MTDATACU)'
            END--2
      END
      ELSE
      BEGIN
```

```
                  UPDATEMTDATACUSET
                        NOMACU=@NOMACU,
                        APEACU=@APEACU,
                        TELACU=@TELACU,
                        RELACU=@RELACU,--RELACION
                        CODIEST=@CODIEST--CODIGO DE ESTUDIANTE
                  WHERECODACU=@CODACU

                  If@@error!=0
                  Begin--3
                        Select'Existe un error al actualizar'
                  End--3
                  else
                  Begin--3
                        select'Registro Actualizado'
                  end--3
      END
      END
GO
/****** Object:  StoredProcedure [dbo].[SPSA_AUDITORIA]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROC[dbo].[SPSA_AUDITORIA]
@AUDIIDNVARCHAR(6),
@AUDIFECHAHORADATETIME,
@AUDIIPNVARCHAR(30),
@AUDIUSUARIONVARCHAR(50),
@AUDIMOVIMIENTONVARCHAR(1),
@AUDIMOVTABLANVARCHAR(30)
AS
BEGINTRANSACTION
BEGIN
      IFEXISTS(SELECT 1 FROMAUDITORIAWHEREAUDIID=@AUDIID)
      BEGIN
                        UPDATEAUDITORIASET

                        AUDIFECHAHORA=@AUDIFECHAHORA,
                        AUDIIP=@AUDIIP,
                        AUDIUSUARIO=@AUDIUSUARIO,
                        AUDIMOVIMIENTO=@AUDIMOVIMIENTO,
                        AUDIMOVTABLA=@AUDIMOVIMIENTO
                        WHEREAUDIID=@AUDIID
                  If@@error!=0
                        Begin
                        RollBackTransaction
                        select'Error Actualización de
Registro'
--                      Select iRetCode = -100,  sErrMsg =
'Error Actualización de Registro'
                  End
                  Else
                  Begin
                        --Select iRetCode = 0, sErrMsg =
                        select'Actualización Realizada con
Exito'
```

```
                              CommitTransaction
                        End
      END
      ELSE
      BEGIN

      INSERTINTOAUDITORIA(AUDIID,AUDIFECHAHORA,AUDIIP,AUDIUSUARIO,AU
DIMOVIMIENTO,AUDIMOVTABLA)

      VALUES(@AUDIID,@AUDIFECHAHORA,@AUDIIP,@AUDIUSUARIO,@AUDIMOVIMI
ENTO,@AUDIMOVTABLA)
                                    If@@error!=0
                  Begin
                        RollBackTransaction
                        --Select iRetCode = -100,  sErrMsg =
                        select'Ocurrio un Error al Insertar el
Registro'
                  End
                  Else
                  Begin
                        --Select iRetCode = 1, sErrMsg=
                        select'Registro Insertado Correctamente'
                        CommitTransaction
                  End

      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_USUARIO]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_USUARIO]
      @CODUSUNVARCHAR(10)
AS
BEGIN
IFEXISTS(SELECT 1 FROMUSUARIOWHERECODUSU=@CODUSU)
            BEGIN
            DELETEFROMUSUARIOWHERECODUSU=@CODUSU
If@@error!= 0
            Begin--2
                  Select'Existe un error al borrar'
            End--2
            else
            Begin--2
                  select'Registro Eliminado'
            end--2
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_SECUENCIAL]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
```

```
CREATEPROCEDURE[dbo].[SPSD_SECUENCIAL]
@CODPARNVARCHAR(10)
AS
BEGIN
      DELETEFROMSECUENCIALWHERECODPAR=@CODPAR
END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_REPORTE]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_REPORTE]
@CODIGONVARCHAR(70)
AS
BEGIN--0
      IFEXISTS(SELECT 1 FROMREPORTEWHERECODREP=@CODIGO)
      BEGIN--1
            DELETEFROMREPORTEWHERECODREP=@CODIGO;
            If@@error!= 0
            Begin--2
                  SelectiRetCode=-100,sErrMsg='Existe un error al
borrar'
            End--2
            else
            Begin--2
                  selectiRetCode= 0,sErrMsg='Registro Eliminado'
            end--2
      END--1
END--0
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATPER]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATPER]

      @CODSEM     nvarchar(5)
AS
BEGIN
      DELETEFROMMTDATPERWHERECOD_SEM=@CODSEM
END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATNOT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATNOT]
      @CODNOTint

AS
BEGINTRANSACTION
BEGIN
```

```
                  DELETEMTDATNOTWHERECODNOT=@CODNOT
                              If@@error!=0
                                    Begin
                                    RollBackTransaction
                                    SelectiRetCode=-
100,sErrMsg='Hubo un error al eliminar el Registro'
                              End
                              Begin
                                    SelectiRetCode=
1,sErrMsg='Registro Eliminado Correctamente'
                                    CommitTransaction
                              End
                  END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATMAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIEROFF
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATMAT]

      @i_CODMAT   nvarchar(6)
AS


            BEGIN
                  DELETEFROMMTDATMAT

                  WhereCODMAT =     @i_CODMAT
                                    If@@error!=0
                                          Begin


      SelectiRetCode=-100,sErrMsg='Error al Eliminar Registro'
                                    End
                                    Else
                                    Begin
                                                SelectiRetCode=
0,sErrMsg='Registro Eliminado Correctamente'



                                    end
            END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATDOC]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATDOC]
@CODDOCNVARCHAR(10)
AS
BEGIN
      IFEXISTS(SELECT 1 FROMMTDATNOTWHERE@CODDOC=@CODDOC)
```

```
      BEGIN
            select'NO SE PUEDE ELIMINAR POR QUE EL DATO EXISTE EN
MTDATNOT'
      END
      ELSE
      BEGIN
            DELETEFROMMTDATDOCWHERECOD_DOC=@CODDOC
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATDAT]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATDAT]
@CODDATNVARCHAR(7)
AS
BEGIN--0
      IFEXISTS(SELECT 1 FROMMTDATDATWHERECODDAT=@CODDAT)
      BEGIN--1
            DELETEFROMMTDATDATWHERECODDAT=@CODDAT
            If@@error!= 0
            Begin--2
                  Select'Existe un error al borrar'
            End--2
            else
            Begin--2
                  select'Registro Eliminado'
            end--2
      END--1
END--0
GO
/****** Object:  StoredProcedure [dbo].[SPSD_MTDATASIG]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSD_MTDATASIG]
      @CODASIGint
AS
BEGINTRANSACTION
BEGIN
                                    DELETEFROMMTDATASIG

                                    WHERECODASIG=@CODASIG

                                    If@@error!=0
                                          Begin

      RollBackTransaction

      SelectiRetCode=-100,sErrMsg='Error al Eliminar Registro'
                                    End
                                    Else
                                    Begin
```

```
                                                    SelectiRetCode=
0,sErrMsg='Registro Eliminado Correctamente'


      CommitTransaction

                                    end
            END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_USUARIO]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROC[dbo].[SPSB_USUARIO]
@OPNVARCHAR(1),
@DATONVARCHAR(50),
@DATOPNVARCHAR(50)
AS
BEGIN
declare@clavasvarbinary(500)
set@clav=encryptbypassphrase('clave',@DATOP)

IF (@OP=1)--GENERAL
            BEGIN


      selectcodusu,nomusu,dbo.desencriptar(pasusu)ASCLAVE,CONECTADO,
ROLUSU,

      (SELECTNOLDATFROMMTDATDATWHERECODDAT=ROLUSU)ASROLfromusuario
            END
IF (@OP=2)--POR CODIGO
            BEGIN

      selectcodusu,nomusu,PASUSU,dbo.desencriptar(pasusu)ASCLAVE,rol
usufromusuario
            WHERECODUSULIKE'%'+@DATO+'%'
            END
IF (@OP=3)--POR NOMBRE DE USUARIO
            BEGIN

      selectcodusu,nomusu,PASUSU,dbo.desencriptar(pasusu)ASCLAVE,rol
usufromusuario
            WHERENOMUSULIKE'%'+@DATO+'%'
            END
IF (@OP=4)--POR CLAVE Y CONTRASEÑA
            BEGIN

      selectcodusu,nomusu,dbo.desencriptar(pasusu)ASCLAVE,CONECTADO,
ROLUSU,

      (SELECTNOLDATFROMMTDATDATWHERECODDAT=ROLUSU)ASROLfromusuario

      WHERENOMUSU=@DATOANDdbo.desencriptar(pasusu)=@DATOPANDCONECTAD
O='NO'-- PASUSU=@clav
```

```
            UPDATEUSUARIOSET
            conectado='SI'
            WHERENOMUSU=@DATO
            END

IF (@OP=5)--POR NOMBRE DE USUARIO
            BEGIN

      selectcodusu,nomusu,PASUSU,dbo.desencriptar(pasusu)ASCLAVE,rol
usufromusuario
            WHERENOMUSU=@DATO
            UPDATEUSUARIOSET
            conectado='NO'
            WHERENOMUSU=@DATO
            END
IF (@OP=6)--POR CLAVE Y CONTRASEÑA
            BEGIN

      selectcodusu,nomusu,dbo.desencriptar(pasusu)ASCLAVE,CONECTADO,
ROLUSU,

      (SELECTNOLDATFROMMTDATDATWHERECODDAT=ROLUSU)ASROLfromusuario
            WHERENOMUSU=@DATOANDdbo.desencriptar(pasusu)=@DATOP

            END

END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_SECUENCIAL]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_SECUENCIAL]
@OPNVARCHAR(1),
@DATONVARCHAR(10)
AS
BEGIN
      IF(@OP=1)
      BEGIN
            SELECT*FROMSECUENCIAL
      END

      IF(@OP=2)
      BEGIN

      SELECT*FROMSECUENCIALWHERECODPARLIKE'%'+@DATO+'%'ORPARSECLIKE'
%'+@DATO+'%'ORCADPARLIKE'%'+@DATO+'%'
      END

END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_REPORTE]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
```

```
GO
CREATEPROC[dbo].[SPSB_REPORTE]
@OPNVARCHAR(1),
@CODREPORTENVARCHAR(70),
@NOMREPNVARCHAR(200)
AS
BEGIN
      IF(@OP=1)
            BEGIN

      IFEXISTS(SELECT*FROMREPORTEWHERECODREP=@CODREPORTE)
                  BEGIN
                        SELECT*FROMREPORTE
                        WHERECODREP=@CODREPORTE
                  END
                  ELSE
                  BEGIN
                        select'EL REPORTE '+@CODREPORTE+' NO EXISTE'
                  END
            END
      IF(@OP=2)--BUSQUEDA GENERAL
      BEGIN
                  SELECT*FROMREPORTE
      END
      IF(@OP=3)--BUSQUEDA POR CODIGO
      BEGIN
                  SELECT*FROMREPORTE
                              WHERECODREPLIKE'%'+@CODREPORTE+'%'
      END
      IF(@OP=4)--BUSQUEDA POR NOMBRE DE REPORTE
      BEGIN
                  SELECT*FROMREPORTE
                  WHERENOMREPORTLIKE'%'+@NOMREP+'%'
      END

            IF(@OP=5)--BUSQUEDA POR PATH DE REPORTE
      BEGIN
                  SELECT*FROMREPORTE
                  WHEREPATHREPLIKE'%'+@NOMREP+'%'
      END
END
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATPER]    Script
Date: 04/07/2015 02:01:25 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROC[dbo].[SPSB_MTDATPER]
@OPNVARCHAR(1),
@DATONVARCHAR(100)
AS
BEGIN
      IF(@OP=1)
      BEGIN
      selectCOD_SEM,NOML_SEM,NOMC_SEM,OBSER_CAR,AS_USUARIO,AS_HORA,E
STADO,FECINI,FECCULfromMTDATPER
      END
```

```
	IF(@OP=2)
	BEGIN
	selectCOD_SEM,NOML_SEM,NOMC_SEM,OBSER_CAR,AS_USUARIO,AS_HORA,E
STADO,FECINI,FECCULfromMTDATPER
	WHERECOD_SEMLIKE'%'+@DATO+'%'ORNOML_SEMLIKE'%'+@DATO+'%'ORESTA
DOLIKE'%'+@DATO+'%'
	END

	IF(@OP=3)
	BEGIN
	selectCOD_SEM,NOML_SEM,NOMC_SEM,OBSER_CAR,AS_USUARIO,AS_HORA,E
STADO,FECINI,FECCULfromMTDATPER
	WHEREESTADO=@DATO
	END
END
GO
/****** Object:  View [dbo].[VI_MTDATEST]    Script Date: 04/07/2015
02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEVIEW[dbo].[VI_MTDATEST]
AS
SELECTCODEST,NOMEST,APEEST,FECNACEST,
NACEST,(SELECTNOLDATFROMMTDATDATWHERECODDAT=E.NACEST)ASNACIONALIDAD,
POSCEDEST,CEDEST,
SEXEST,(SELECTNOLDATFROMMTDATDATWHERECODDAT=E.SEXEST)ASGENERO,CMEDES
T,CDENEST,CCEDPEST,CCEDMEST,PNACEST,FOTO,PCEDEST,NUMMATEST,CODMINEDU
,FECMATESTFROMMTDATESTE
GO
/****** Object:  View [dbo].[VIUSUARIO]    Script Date: 04/07/2015
02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEVIEW[dbo].[VIUSUARIO]
AS

	selectcodusu,nomusu,dbo.desencriptar(pasusu)ASCLAVE,CONECTADO,
ROLUSU,

	(SELECTNOLDATFROMMTDATDATWHERECODDAT=ROLUSU)ASROLfromusuario
GO
/****** Object:  View [dbo].[VIMTDATASIG]    Script Date: 04/07/2015
02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEVIEW[dbo].[VIMTDATASIG]
AS
selectCODASIG,CODIPER,CODIDOC,CODIMAT,NIVMAT,PARMAT,TIPMAT,
(selectP.NOML_SEMfromMTDATPERPwhereP.COD_SEM=A.CODIPER)ASSEMESTRE,
(SELECT (D.APE_DOC+'
'+D.NOM_DOC)FROMMTDATDOCDWHERED.COD_DOC=A.CODIDOC)ASDOCENTE,
(SELECTM.NOMLMATFROMMTDATMATMWHEREM.CODMAT=A.CODIMAT)ASASIGNATURA,
```

```
(SELECTT.NOLDATFROMMTDATDATTWHERET.CODDAT=A.NIVMAT)ASNIVEL,
(SELECTT.NOLDATFROMMTDATDATTWHERET.CODDAT=A.PARMAT)ASPARALELO,
(SELECTT.NOLDATFROMMTDATDATTWHERET.CODDAT=A.TIPMAT)ASTIPO
fromMTDATASIGA
GO
/****** Object:  View [dbo].[VIDOCENTE]    Script Date: 04/07/2015
02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEVIEW[dbo].[VIDOCENTE]
AS
SELECTCOD_DOC,NOM_DOC,APE_DOC,NICK,dbo.DESENCRIPTAR(CLAVE)ASCLAVE,
DIR_DOC,TEL_DOC,CEL_DOC,MAIL_DOC,FEC_NAC_DOC,AS_USUARIO,AS_HORA,HABI
LITADO,
CONECTADO,dbo.DESENCRIPTAR(CLAVE)ASCLAVED,(APE_DOC+'
'+NOM_DOC)ASNOMBRESFROMMTDATDOC
GO
/****** Object:  View [dbo].[VI_MTDATNOT]    Script Date: 04/07/2015
02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEVIEW[dbo].[VI_MTDATNOT]
AS
SELECT*,
(SELECT (E.APEEST+'
'+E.NOMEST)FROMMTDATESTEWHEREE.CODEST=N.CODIEST)ASESTUDIANTEFROMMTDA
TNOTN,VIMTDATASIGA
WHEREN.CODIASIG=A.CODASIG
GO
/****** Object:  StoredProcedure [dbo].[SPSB_MTDATNOTA]    Script
Date: 04/07/2015 02:01:26 ******/
SETANSI_NULLSON
GO
SETQUOTED_IDENTIFIERON
GO
CREATEPROCEDURE[dbo].[SPSB_MTDATNOTA]
@OPNVARCHAR(1),
@CEDNVARCHAR(10),
@PERNVARCHAR(10)
AS
BEGIN

      IF (@OP=2)--BUSCANDO DESDE VI_MTDATNOT SEGUN CEDULA Y PERIODO
      BEGIN
      SELECT*FROMVI_MTDATNOTWHERECODIEST=@CEDANDCODIPER=@PER
      END
END
GO
```

1.02 Script Del Sistema JASONMTSCOL

Conexión capa datos

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace ENTIDADES
{
classConexion
    {

publicstring conexionC()
        {

StreamReader leerArchivo = newStreamReader("C:\\CONF\\ESCOCONF.txt");
////TOMAR ESA INFORMACION Y DIRIGIRME A LA BASE DE DATOS
string datos;
            datos = leerArchivo.ReadToEnd();
return@"Data source = " + datos + "; Integrated Security=SSPI";
        }

publicstring conexionF()
        {
return@"Data source = LADY-PC\LADY; Initial Catalog = FOTO; Integrated
Security=SSPI";
}

    }
}
```

CAPA DE DATOS DOCENTES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.Common;
using System.Data.SqlClient;
using ENTIDADES;
namespace DATOS
{
publicclassDocentesDALC
    {
privateConexion conection = newConexion();
//PARA INGRESOS Y MODIFICACIONES

publicBoolean insertarDoc(ENTDocentes docentes)
{
SqlConnection cnn = newSqlConnection(conection.conexionC());
```

```
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATDOC", cnn);
            cmd.CommandType = CommandType.StoredProcedure;


            cmd.Parameters.Add("@COD_DOC", SqlDbType.NVarChar).Value =
docentes.iCODDOC;
            cmd.Parameters.Add("@NOM_DOC", SqlDbType.NVarChar).Value =
docentes.iNOMDOC;
            cmd.Parameters.Add("@APE_DOC", SqlDbType.NVarChar).Value =
docentes.iAPEDOC;
            cmd.Parameters.Add("@NICK", SqlDbType.NVarChar).Value =
docentes.iNICK;
            cmd.Parameters.Add("@CLAVE", SqlDbType.NVarChar).Value =
docentes.iCLAVE;
            cmd.Parameters.Add("@DIR_DOC", SqlDbType.NVarChar).Value =
docentes.iDIRDOC;
            cmd.Parameters.Add("@TEL_DOC", SqlDbType.NVarChar).Value =
docentes.iTELDOC;
            cmd.Parameters.Add("@CEL_DOC", SqlDbType.NVarChar).Value =
docentes.iCELDOC;
            cmd.Parameters.Add("@MAIL_DOC", SqlDbType.NVarChar).Value =
docentes.iMAILDOC;
            cmd.Parameters.Add("@FEC_NAC_DOC", SqlDbType.Date).Value =
docentes.iFECNACDOC;
cmd.Parameters.Add("@AS_USUARIO", SqlDbType.NVarChar).Value =
docentes.iAS_USUARIO;
            cmd.Parameters.Add("@HABILITADO", SqlDbType.NVarChar).Value =
docentes.iHABILITADO;
            cmd.Parameters.Add("@CONECTADO", SqlDbType.NVarChar).Value =
docentes.iCONECTADO;
int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;

}
//FUNCION CREADA PARA CONSULTAR DOCENTES POR EL APELLIDO
publicList<ENTDocentes> listarDocentes(ENTDocentes docente)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand SQL = newSqlCommand("SPSB_MTDATDOC", cnn);
            SQL.Parameters.Add("@OP", SqlDbType.NVarChar, 1).Value =
docente.iOP;
            SQL.Parameters.Add("@DATO", SqlDbType.NVarChar, 100).Value =
docente.iDATO;
            SQL.Parameters.Add("@DATOC", SqlDbType.NVarChar, 100).Value =
docente.iDATOC;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTDocentes> coleccion = newList<ENTDocentes>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTDocentes docentes = newENTDocentes()
                {
```

```
                    iCODDOC = lector.GetString(0),

                    iNOMDOC = lector.GetString(1),
                    iAPEDOC = lector.GetString(2),
                    iNICK = lector.GetString(3),
                    iCLAVE = lector.GetString(4),
                    iDIRDOC = lector.GetString(5),
                    iTELDOC = lector.GetString(6),
                    iCELDOC = lector.GetString(7),
                    iMAILDOC = lector.GetString(8),
                    iFECNACDOC = lector.GetDateTime(9),
iAS_USUARIO = lector.GetString(10),
                    iAS_HORA = lector.GetDateTime(11),
                    iHABILITADO = lector.GetString(12),
                    iCONECTADO = lector.GetString(13),
                    iCLAVED = lector.GetString(14),
                    iNOMBRESDOC = lector.GetString(15),
                };
                coleccion.Add(docentes);
            }
return coleccion;
        }
//FUNCION CREADA PARA ELIMINAR DOCENTES
publicBoolean eliminarDoc(ENTDocentes docentes)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSD_MTDATDOC", cnn);
            cmd.CommandType = CommandType.StoredProcedure;


            cmd.Parameters.Add("@CODDOC", SqlDbType.NVarChar).Value =
docentes.iCODDOC;

int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;

        }

    }
}
```

CAPA DATOS ASIGNATURA

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.Common;
using System.Data.SqlClient;
using ENTIDADES;
namespace DATOS
{
publicclassMateriasDALC
```

```
    {
privateConexion conection = newConexion();
//PARA INGRESOS Y MODIFICACIONES

publicBoolean insertarM(string[] materia)
{
SqlConnection cnn = newSqlConnection(conection.conexionC ());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATMAT", cnn);
            cmd.CommandType = CommandType.StoredProcedure;


            cmd.Parameters.Add("@i_CODMAT",SqlDbType.NVarChar ).Value =
materia[0];
            cmd.Parameters.Add("@i_NOMLMAT", SqlDbType.NVarChar).Value =
materia[1];
            cmd.Parameters.Add("@i_NOMCMAT", SqlDbType.NVarChar).Value =
materia[2];
            cmd.Parameters.Add("@i_OBSMAT", SqlDbType.NVarChar).Value =
materia[3];
            cmd.Parameters.Add("@i_AS_USUARIO", SqlDbType.NVarChar).Value =
materia[4];
int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;

        }

//PARA BUSQUEDAS POR NOMBRE

publicList<ENTMateria>  listarAsignaturas(ENTMateria nombre)
{
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand SQL = newSqlCommand("SPSB_MTDATMAT", cnn);
            SQL.Parameters.Add("@OP", SqlDbType.NVarChar, 1).Value =
nombre.IOP;
            SQL.Parameters.Add("@DATO", SqlDbType.NVarChar, 70).Value =
nombre.IDATO;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTMateria> coleccion = newList<ENTMateria>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTMateria  xNombre = newENTMateria()
                {
                    I_CODMAT = lector.GetString(0),
                    I_NOMLMAT  = lector.GetString(1),
                    I_NOMCMAT  = lector.GetString(2),
                    I_OBSMAT  = lector.GetString(3),
                    I_USUARIO = lector.GetString(4),
                    I_FECHA = Convert.ToString(lector.GetDateTime(5))
};
                coleccion.Add(xNombre);
            }
return coleccion;
```

```
        }
//PARA ELIMINAR DATOS DE MATERIAS
publicbool Eliminar(string[]  eliminar)
{
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSD_MTDATMAT", cnn);
            cmd.CommandType = CommandType.StoredProcedure;


            cmd.Parameters.Add("@i_CODMAT", SqlDbType.NVarChar).Value =
eliminar[0];

int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;
}
//PARA BUSQUEDAS POR CODIGO
publicList<ENTMateria> listarXCodigo(ENTMateria codigo)
{
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();

SqlCommand SQL = newSqlCommand("SPSC_LISMATCOD", cnn);
            SQL.Parameters.Add("@i_CODMAT", SqlDbType.NVarChar, 6).Value =
codigo.I_CODMAT;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTMateria> coleccion = newList<ENTMateria>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTMateria xCodigo = newENTMateria()
                {

                    I_CODMAT = lector.GetString(0),
                    I_NOMLMAT = lector.GetString(1),
                    I_NOMCMAT = lector.GetString(2),
I_OBSMAT = lector.GetString(3)
                };
                coleccion.Add(xCodigo);
            }
return coleccion;
        }
//PARA BUSCAR TODOS LOS CODIGOS
publicList<ENTMateria> ListarMaterias()
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());


List<ENTMateria> lstMaterias = newList<ENTMateria>();

try
            {
                cnn.Open();
SqlCommand cmd = newSqlCommand("SPSC_ALLMATERIAS", cnn);
                cmd.CommandType = System.Data.CommandType.StoredProcedure;
```

```
SqlDataReader dr = cmd.ExecuteReader();
dynamic pos1 = dr.GetOrdinal("CODMAT");
dynamic pos2 = dr.GetOrdinal("NOMLMAT");
dynamic pos3 = dr.GetOrdinal("NOMCMAT");
dynamic pos4 = dr.GetOrdinal("OBSMAT");
while (dr.Read())
{
//'Trabajamos con los constructores creados
ENTMateria objENTMaterias = newENTMateria();
//'Llamamos  a los constuctores
var with = objENTMaterias;
with.I_CODMAT = dr.GetValue(pos1);
                    with.I_NOMLMAT  = dr.GetValue(pos2);
                    with.I_NOMCMAT = dr.GetValue(pos3);
                    with.I_OBSMAT = dr.GetValue(pos4);



//'Añadimos a la lista todos los
//'registros encontrados con
//'objBELProducto
                    lstMaterias.Add(objENTMaterias);
                }
            }
catch (Exception)
            {
//si no vas a utilizar el ex para q lo declaras bolsasiqui es una alerta
}
finally
            {
                cnn.Close();
            }
return lstMaterias;
}
    }
}
```

CAPA DATOS PERSONA ACUDIR

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.SqlClient;
using ENTIDADES;

namespace DATOS
{
publicclassMtDatAcuDALC
    {
privateConexion conection = newConexion();
publicbool insertarAcudir(string[] dato)
        {
String msg = "";
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATACU", cnn);
```

```
//cmd.Parameters.Add("@i_BANDERA", SqlDbType.NVarChar,1).Value = dato[0];
            cmd.Parameters.Add("@CODACU", SqlDbType.NVarChar, 7).Value =
dato[0];
            cmd.Parameters.Add("@NOMACU", SqlDbType.NVarChar, 70).Value =
dato[1];
            cmd.Parameters.Add("@APEACU", SqlDbType.NVarChar, 70).Value =
dato[2];
            cmd.Parameters.Add("@TELACU", SqlDbType.NVarChar, 10).Value =
dato[3];
            cmd.Parameters.Add("@RELACU", SqlDbType.NVarChar, 7).Value =
dato[4];
            cmd.Parameters.Add("@CODIEST", SqlDbType.NVarChar, 10).Value =
dato[5];
            cmd.Parameters.Add("@TIPDATACU", SqlDbType.NVarChar, 2).Value =
dato[6];
            cmd.CommandType = CommandType.StoredProcedure;
int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;
//cmd.ExecuteNonQuery();

//IDataReader lector = cmd.ExecuteReader();
//while (lector.Read())
//{
//    {
//        msg = lector.GetString(0);

//    }
//}

//cnn.Close();
//return msg;
        }

publicList<ENTMtDatAcu> listarAcudir(ENTMtDatAcu Acudir)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();

SqlCommand SQL = newSqlCommand("SPSB_MTDATACU", cnn);
SQL.Parameters.Add("@OP", SqlDbType.NVarChar, 1).Value = Acudir.IOP;
            SQL.Parameters.Add("@CEDEST", SqlDbType.NVarChar, 10).Value =
Acudir.ICODIEST;
SQL.Parameters.Add("@TIPO", SqlDbType.NVarChar, 2).Value = Acudir.IDATO;
SQL.CommandType = CommandType.StoredProcedure;

List<ENTMtDatAcu> coleccion = newList<ENTMtDatAcu>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTMtDatAcu periodoA = newENTMtDatAcu()
                {

                    ICODACU = lector.GetString(0),
                    INOMACU = lector.GetString(1),
                    IAPEACU = lector.GetString(2),
                    ITELACU = lector.GetString(3),
```

```
                    IRELACU = lector.GetString(4),
                    ICODIEST = lector.GetString(5),
                    IDATO = lector.GetString(6),

};
                coleccion.Add(periodoA);
            }
return coleccion;
}
    }
}

using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.Common;
using System.Data.SqlClient;
using ENTIDADES;

namespace DATOS
{
publicclassmtDatAsigDALC
    {

privateConexion conection = newConexion();


publicBoolean inserAsignacion(string[] datos)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATASIG", cnn);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@CODASIG", SqlDbType.Int).Value =
Convert.ToInt32(datos[0]);
cmd.Parameters.Add("@CODIPER", SqlDbType.NVarChar, 5).Value = datos[1];
            cmd.Parameters.Add("@CODIDOC", SqlDbType.NVarChar, 10).Value =
datos[2];
            cmd.Parameters.Add("@CODIMAT", SqlDbType.NVarChar, 6).Value =
datos[3];
            cmd.Parameters.Add("@NIVMAT", SqlDbType.NVarChar, 7).Value =
datos[4];
            cmd.Parameters.Add("@PARMAT", SqlDbType.NVarChar, 7).Value =
datos[5];
            cmd.Parameters.Add("@TIPMAT", SqlDbType.NVarChar, 7).Value =
datos[6];
            cmd.Parameters.Add("@OBSASIG", SqlDbType.NVarChar, 500).Value =
datos[7];
int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;

        }

publicBoolean eliminarAsignaciones(string[] datos)
        {
```

```
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSD_MTDATASIG", cnn);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.Add("@CODASIG", SqlDbType.Int).Value =
Convert.ToInt32(datos[0]);

int huboexito = cmd.ExecuteNonQuery();
if (huboexito == 0)
returnfalse;
else
returntrue;

        }




publicList<ENTMtDatAsig> listarAsignaciones(ENTMtDatAsig asig)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();

SqlCommand SQL = newSqlCommand("SPSB_MTDATASIG", cnn);
            SQL.Parameters.Add("@OP", SqlDbType.NVarChar, 1).Value = asig.iOP;
            SQL.Parameters.Add("@DATO", SqlDbType.NVarChar, 100).Value =
asig.iDATO;
            SQL.Parameters.Add("@DATO1", SqlDbType.NVarChar, 100).Value =
asig.iDATO1;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTMtDatAsig> coleccion = newList<ENTMtDatAsig>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTMtDatAsig periodoA = newENTMtDatAsig()
                {

                    iCODASIG = Convert.ToString(lector.GetInt32(0)),
                    iCODIPER = lector.GetString(1),
                    iCODIDOC = lector.GetString(2),
                    iCODIMAT = lector.GetString(3),
                    iNIVMAT = lector.GetString(4),
                    iPARMAT = lector.GetString(5),
                    iTIPMAT = lector.GetString(6),
                    iPERIODO = lector.GetString(7),
                    iDOCENTE = lector.GetString(8),
                    iASIGNATURA = lector.GetString(9),
                    iNIVEL = lector.GetString(10),
                    iPARALELO = lector.GetString(11),
                    iTIPO_MATERIA = lector.GetString(12),
                    iOBSASIG = lector.GetString(13)
};
                coleccion.Add(periodoA);
            }
return coleccion;
        }
    }
```

```
}
```

CAPA DATOS PARA INGRESOS Y MODIFICACIONES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.SqlClient;
using ENTIDADES;


namespace DATOS
{
publicclassmtDatdatDALC
    {
privateConexion conection = newConexion();

publicDataSet IngresarDatosSpdatdat(DataSet dtsFunciones)
{

try
            {
ENTMpdatdat spE = newENTMpdatdat();

DataSet dsRes = newDataSet();
DataTable dtPersonas = dtsFunciones.Tables[0];
DataRow[] filas = dtPersonas.Select();
foreach (DataRow cell in filas)
                {
                    spE.I_CODDAT = cell["CODIGO"].ToString();
                    spE.I_TIPDAT = cell["TIPO"].ToString();
                    spE.I_NOLDAT = cell["NOLDAT"].ToString();
                    spE.I_NOCDAT = cell["NOCDAT"].ToString();
spE.I_USUARIO = cell["USUARIO"].ToString();

dsRes = spE.update(dsRes);
                }
return dsRes;

}
catch (Exception)
            {

throw;
            }
        }

//PARA INGRESOS Y MODIFICACIONES

publicstring insertarSpdatdat(string[] dato)
        {
String msg = "";
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATDAT", cnn);
//cmd.Parameters.Add("@i_BANDERA", SqlDbType.NVarChar,1).Value = dato[0];
            cmd.Parameters.Add("@i_CODDAT", SqlDbType.NVarChar, 6).Value =
dato[0];
```

```
            cmd.Parameters.Add("@i_TIPDAT", SqlDbType.NVarChar, 3).Value =
dato[1];
            cmd.Parameters.Add("@i_NOLDAT", SqlDbType.NVarChar, 50).Value =
dato[2];
            cmd.Parameters.Add("@i_NOCDAT", SqlDbType.NVarChar, 30).Value =
dato[3];
            cmd.Parameters.Add("@i_AS_USUARIO", SqlDbType.NVarChar, 8).Value =
dato[4];

            cmd.CommandType = CommandType.StoredProcedure;
            cmd.ExecuteNonQuery();

IDataReader lector = cmd.ExecuteReader();
while (lector.Read())
            {
                {
                    msg = lector.GetString(0);

                }
            }

            cnn.Close();
return msg;
        }
//PARA ELIMINAR DATOS
publicstring eliminarSpdatdat(string dato)
        {
String msg = "";
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSD_MTDATDAT", cnn);
            cmd.Parameters.Add("@CODDAT", SqlDbType.NVarChar, 6).Value = dato;
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.ExecuteNonQuery();

IDataReader lector = cmd.ExecuteReader();
while (lector.Read())
            {
                {
                    msg = lector.GetString(0);

}
            }

            cnn.Close();
return msg;
        }

//PARA BUSQUEDAS POR CODIGO DE DATO

publicList<ENTMpdatdat> listarDato(ENTMpdatdat consulta)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();

SqlCommand SQL = newSqlCommand("SPSB_MTDATDAT", cnn);
            SQL.Parameters.Add("@iOP", SqlDbType.NVarChar, 1).Value =
consulta.IOP;
```

```
            SQL.Parameters.Add("@iDATO", SqlDbType.NVarChar, 50).Value =
consulta.IDATO;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTMpdatdat> coleccion = newList<ENTMpdatdat>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTMpdatdat consulta1 = newENTMpdatdat()
                {

                    I_CODDAT = lector.GetString(0),
                    I_TIPDAT = lector.GetString(1),
                    I_NOLDAT = lector.GetString(2),
                    I_NOCDAT = lector.GetString(3),
                    I_USUARIO = lector.GetString(4),
                    I_ASHORA = Convert.ToString(lector.GetDateTime(5))
};
                coleccion.Add(consulta1);
            }
return coleccion;
        }


    }
}
```

CAPA DE DATOS USUARIO

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using ENTIDADES;
namespace ENTIDADES
{
publicclassusuarioDALC
    {
privateConexion conection = newConexion();
//public static int crearUsuario(string usuario, string contraseña)
//{
publicstring insertarUsuario(string[] dato)
        {
string msg = "";
int resultado = 0;
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_USUARIO", cnn);
            cmd.Parameters.Add("@CODIGO", SqlDbType.NVarChar, 10).Value =
dato[0];
            cmd.Parameters.Add("@NOMBRE", SqlDbType.NVarChar, 50).Value =
dato[1];
            cmd.Parameters.Add("@PASS", SqlDbType.NVarChar, 50).Value =
dato[2];
            cmd.Parameters.Add("@CONECTADO", SqlDbType.NVarChar, 2).Value =
dato[3];
            cmd.Parameters.Add("@ROL", SqlDbType.NVarChar, 50).Value =
dato[4];
```

```
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.ExecuteNonQuery();

IDataReader lector = cmd.ExecuteReader();
while (lector.Read())
            {
                {
                    msg = lector.GetString(0);

}
            }

            cnn.Close();
return msg;

        }


//PARA CONSULTAR USUARIOS
publicList<ENTUsuario> listarDato(ENTUsuario consulta)
{
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();

SqlCommand SQL = newSqlCommand("SPSB_USUARIO", cnn);
            SQL.Parameters.Add("@OP", SqlDbType.NVarChar, 1).Value =
consulta.IOP;
            SQL.Parameters.Add("@DATO", SqlDbType.NVarChar, 50).Value =
consulta.IDATO;
            SQL.Parameters.Add("@DATOP", SqlDbType.NVarChar, 50).Value =
consulta.ICLAVEDES;
            SQL.CommandType = CommandType.StoredProcedure;

List<ENTUsuario> coleccion = newList<ENTUsuario>();
IDataReader lector = SQL.ExecuteReader();
while (lector.Read())
            {
ENTUsuario consulta1 = newENTUsuario()
                {

                    ICODIGO = lector.GetString(0),
IUSUARIO = lector.GetString(1),
                    ICLAVEDES = lector.GetString(2),
                    ICONECTADO = lector.GetString(3),
IROLUSU = lector.GetString(4),
                    IROL= lector.GetString(5)

};
                coleccion.Add(consulta1);
            }
return coleccion;
}

//PARA ELIMINAR USUARIO
publicstring eliminarUsuario(string[] dato)
        {
string msg = "";
int resultado = 0;
```

```
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSD_USUARIO", cnn);
            cmd.Parameters.Add("@CODUSU", SqlDbType.NVarChar, 10).Value =
dato[0];

            cmd.CommandType = CommandType.StoredProcedure;
            cmd.ExecuteNonQuery();

IDataReader lector = cmd.ExecuteReader();
while (lector.Read())
            {
                {
                    msg = lector.GetString(0);

}
            }

            cnn.Close();
return msg;

        }
    }
}
```

CAPA ENTIDADES CONEXIÓN

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace ENTIDADES
{
classConexion
    {

publicstring conexionC()
        {

StreamReader leerArchivo = newStreamReader("C:\\CONF\\ESCOCONF.txt");
////TOMAR ESA INFORMACION Y DIRIGIRME A LA BASE DE DATOS
string datos;
            datos = leerArchivo.ReadToEnd();
return@"Data source = " + datos + "; Integrated Security=SSPI";
        }

publicstring conexionF()
        {
return@"Data source = LADY-PC\LADY; Initial Catalog = FOTO; Integrated
Security=SSPI";
}

    }
}
```

CAPA ENTIDADES DOCENTES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ENTIDADES
{
publicclassENTDocentes
    {
privatestring OP;
publicstring iOP
        {
get
            { return OP; }
set
            { OP = value; }

        }

privatestring DATO;
publicstring iDATO
        {
get
            { return DATO; }
set
            { DATO = value; }

        }

privatestring DATOC;
publicstring iDATOC
        {
get
            { return DATOC; }
set
            { DATOC = value; }

        }

privatestring CODDOC;
publicstring iCODDOC
        {
get
            { return CODDOC; }
set
            { CODDOC = value; }

        }
privatestring NOMBRESDOC;
publicstring iNOMBRESDOC
        {
get
            { return NOMBRESDOC; }
set
            { NOMBRESDOC = value; }

        }

privatestring NOMDOC;
```

```
publicstring iNOMDOC
        {
get
            { return NOMDOC; }
set
            { NOMDOC = value; }

        }

privatestring APEDOC;
publicstring iAPEDOC
        {
get
            { return APEDOC; }
set
            { APEDOC = value; }

        }

privatestring NICK;
publicstring iNICK
        {
get
            { return NICK; }
set
            { NICK = value; }

        }

privatestring CLAVE;
publicstring iCLAVE
        {
get
            { return CLAVE; }
set
            { CLAVE = value; }

        }

privatestring CLAVED;
publicstring iCLAVED
        {
get
            { return CLAVED; }
set
            { CLAVED = value; }

        }


privatestring DIRDOC;
publicstring iDIRDOC
        {
get
            { return DIRDOC; }
set
            { DIRDOC = value; }

        }
```

```
privatestring TELDOC;
publicstring iTELDOC
        {
get
            { return TELDOC; }
set
            { TELDOC = value; }

        }

privatestring CELDOC;
publicstring iCELDOC
        {
get
            { return CELDOC; }
set
            { CELDOC = value; }

        }

privatestring MAILDOC;
publicstring iMAILDOC
        {
get
            { return MAILDOC; }
set
            { MAILDOC = value; }

        }

privateDateTime FECNACDOC;
publicDateTime iFECNACDOC
        {
get
            { return FECNACDOC; }
set
            { FECNACDOC = value; }

        }

privatestring HABILITADO;
publicstring iHABILITADO
        {
get
            { return HABILITADO; }
set
            { HABILITADO = value; }

        }

privatestring CONECTADO;
publicstring iCONECTADO
        {
get
            { return CONECTADO; }
set
            { CONECTADO = value; }
```

```
        }
privatestring AS_USUARIO;
publicstring iAS_USUARIO
        {
get
            { return AS_USUARIO; }
set
            { AS_USUARIO = value; }

        }

privateDateTime  AS_HORA;
publicDateTime iAS_HORA
        {
get
            { return AS_HORA; }
set
            { AS_HORA = value; }

}

    }
}
```

CAPA ENTIDADES ESTUDIANTES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ENTIDADES
{
publicclassENTMtDatEst
    {
//PARA OP
privatestring iOP;
publicstring IOP
        {
get { return iOP; }
set { iOP = value; }
        }
//PARA CODIGO
privatestring iDATO;
publicstring IDATO
        {
get { return iDATO; }
set { iDATO = value; }
        }

//PARA CODIGO
privatestring iCODEST;
publicstring ICODEST
        {
get { return iCODEST; }
set { iCODEST = value; }
        }
```

```
//PARA NOMBRE
privatestring iNOMEST;
publicstring INOMEST
        {
get { return iNOMEST; }
set { iNOMEST = value; }
}

//PARA APELLIDO
privatestring iAPEEST;
publicstring IAPEEST
        {
get { return iAPEEST; }
set { iAPEEST = value; }
        }

//PARA NOMBRES COMPLETOS
privatestring iNOMBRES;
publicstring INOMBRES
        {
get { return iNOMBRES; }
set { iNOMBRES = value; }
}

//PARA FECHA DE NACIMIENTO
privatestring iFECNACEST;
publicstring IFECNACEST
        {
get { return iFECNACEST; }
set { iFECNACEST = value; }
        }

//PARA CODIGO NACIONALIDAD
privatestring iNACEST;
publicstring INACEST
        {
get { return iNACEST; }
set { iNACEST = value; }
}

//PARA NACIONALIDAD
privatestring iNACIONALIDAD;
publicstring INACIONALIDAD
        {
get { return iNACIONALIDAD; }
set { iNACIONALIDAD = value; }
        }
//PARA POSEE CEDULA O NO
privatestring iPOSCEDEST;
publicstring IPOSCEDEST
        {
get { return iPOSCEDEST; }
set { iPOSCEDEST = value; }
        }

//PARA CEDULA
privatestring iCEDEST;
publicstring ICEDEST
        {
```

```
get { return iCEDEST; }
set { iCEDEST = value; }
        }

//PARA SEXO DEL ESTUDIANTE
privatestring iSEXEST;
publicstring ISEXEST
        {
get { return iSEXEST; }
set { iSEXEST = value; }
}

//PARA SEXO DEL ESTUDIANTE
privatestring iGENERO;
publicstring IGENERO
        {
get { return iGENERO; }
set { iGENERO = value; }
        }

//PARA CERTIFICADO MEDICO
privatestring iCMEDEST;
publicstring ICMEDEST
        {
get { return iCMEDEST; }
set { iCMEDEST = value; }
        }

//PARA CERTIFICADO DENTAL
privatestring iCDENEST;
publicstring ICDENEST
        {
get { return iCDENEST; }
set { iCDENEST = value; }
}

//PARA COPIA DE CEDULA DEL PADRE
privatestring iCCEDPEST;
publicstring ICCEDPEST
        {
get { return iCCEDPEST; }
set { iCCEDPEST = value; }
}

//PARA CEDULA DE LA MADRE
privatestring iCCEDMEST;
publicstring ICCEDMEST
        {
get { return iCCEDMEST; }
set { iCCEDMEST = value; }
        }

//PARA PARTIDA DE NACIMIENTO DEL ESTUDIANTE
privatestring iPNACEST;
publicstring IPNACEST
        {
get { return iPNACEST; }
set { iPNACEST = value; }
}
```

```
//PARA FOTO
privatestring iFOTO;
publicstring IFOTO
        {
get { return iFOTO; }
set { iFOTO = value; }
        }

//PARA VER SI DEJA CEDULA O NO
privatestring iPCEDEST;
publicstring IPCEDEST
{
get { return iPCEDEST; }
set { iPCEDEST = value; }
        }

//PARA CODIGO
privatestring iNUMMATEST;
publicstring INUMMATEST
        {
get { return iNUMMATEST; }
set { iNUMMATEST = value; }
}

//PARA CODIGO
privatestring iCODMINEDU;
publicstring ICODMINEDU
        {
get { return iCODMINEDU; }
set { iCODMINEDU = value; }
        }

//PARA CODIGO
privatestring iFECMATEST;
publicstring IFECMATEST
        {
get { return iFECMATEST; }
set { iFECMATEST = value; }
        }
    }
}
```

CAPA ENTIDADES PARA ASIGNATURA

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ENTIDADES
{
publicclassENTMtDatAsig
    {
privatestring OP;
publicstring iOP
        {
get
```

```
            { return OP; }
set
            { OP = value; }

        }

privatestring DATO;
publicstring iDATO
        {
get
            { return DATO; }
set
            { DATO = value; }

        }
privatestring DATO1;
publicstring iDATO1
        {
get
            { return DATO1; }
set
            { DATO1 = value; }

        }

privatestring CODASIG;
publicstring iCODASIG
        {
get
            { return CODASIG; }
set
            { CODASIG = value; }

        }

privatestring CODIMAT;
publicstring iCODIMAT
        {
get
            { return CODIMAT; }
set
            { CODIMAT = value; }

        }

privatestring CODIDOC;
publicstring iCODIDOC
        {
get
            { return CODIDOC; }
set
            { CODIDOC = value; }

        }

privatestring NIVMAT;
publicstring iNIVMAT
        {
get
```

```
            { return NIVMAT; }
set
            { NIVMAT = value; }

        }

privatestring PARMAT;
publicstring iPARMAT
        {
get
            { return PARMAT; }
set
            { PARMAT = value; }

        }

privatestring CODIPER;
publicstring iCODIPER
        {
get
            { return CODIPER; }
set
            { CODIPER = value; }

        }


privatestring TIPMAT;
publicstring iTIPMAT
        {
get
            { return TIPMAT; }
set
            { TIPMAT = value; }

        }

privatestring ASIGNATURA;
publicstring iASIGNATURA
        {
get
            { return ASIGNATURA; }
set
            { ASIGNATURA = value; }

        }

privatestring DOCENTE;
publicstring iDOCENTE
        {
get
            { return DOCENTE; }
set
            { DOCENTE = value; }

        }

privatestring NIVEL;
publicstring iNIVEL
```

```
        {
get
            { return NIVEL; }
set
            { NIVEL = value; }

        }

privatestring PARALELO;
publicstring iPARALELO
        {
get
            { return PARALELO; }
set
            { PARALELO = value; }

        }

privatestring PERIODO;
publicstring iPERIODO
        {
get
            { return PERIODO; }
set
            { PERIODO = value; }

        }

privatestring TIPO_MATERIA;
publicstring iTIPO_MATERIA
        {
get
            { return TIPO_MATERIA; }
set
            { TIPO_MATERIA = value; }

        }
privatestring OBSASIG;
publicstring iOBSASIG
        {
get
            { return OBSASIG; }
set
            { OBSASIG = value; }

}




    }
}
```

CAPA ENTIDADES DE NOTAS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
```

```
using System.Data.SqlClient;

namespace ENTIDADES
{
publicclassENTMtDatNot
    {
privatestring OP;
publicstring iOP
        {
get
            { return OP; }
set
            { OP = value; }

        }

privatestring DATO;
publicstring iDATO
        {
get
            { return DATO; }
set
            { DATO = value; }

        }

privatestring CODNOT;
publicstring iCODNOT
        {
get
            { return CODNOT; }
set
            { CODNOT = value; }

        }

privatestring CODASIG;
publicstring iCODASIG
        {
get
            { return CODASIG; }
set
            { CODASIG = value; }

        }

privatestring NOT1NOT;
publicstring iNOT1NOT
        {
get
            { return NOT1NOT; }
set
            { NOT1NOT = value; }

        }

privatestring NOT2NOT;
publicstring iNOT2NOT
        {
```

```
get
            { return NOT2NOT; }
set
            { NOT2NOT = value; }

        }

privatestring NOT3NOT;
publicstring iNOT3NOT
        {
get
            { return NOT3NOT; }
set
            { NOT3NOT = value; }

        }

privatestring PRONOT;
publicstring iPRONOT
        {
get
            { return PRONOT; }
set
            { PRONOT = value; }

        }

privatestring EXANOT;
publicstring iEXANOT
        {
get
            { return EXANOT; }
set
            { EXANOT = value; }

        }

privatestring NOTNOT;
publicstring iNOTNOT
        {
get
            { return NOTNOT; }
set
            { NOTNOT = value; }

        }

privatestring CODIASIG;
publicstring iCODIASIG
        {
get
            { return CODIASIG; }
set
            { CODIASIG = value; }

        }

privatestring EQUINOT;
publicstring iEQUINOT
```

```
        {
get
            { return EQUINOT; }
set
            { EQUINOT = value; }

        }

privatestring CODIMAT;
publicstring iCODIMAT
        {
get
            { return CODIMAT; }
set
            { CODIMAT = value; }

        }

privatestring CODIDOC;
publicstring iCODIDOC
        {
get
            { return CODIDOC; }
set
            { CODIDOC = value; }

        }

privatestring NIVMAT;
publicstring iNIVMAT
        {
get
            { return NIVMAT; }
set
            { NIVMAT = value; }

        }

privatestring PARMAT;
publicstring iPARMAT
        {
get
            { return PARMAT; }
set
            { PARMAT = value; }

        }

privatestring CODIPER;
publicstring iCODIPER
        {
get
            { return CODIPER; }
set
            {  CODIPER = value; }

        }

privatestring CODIEST;
```

```
publicstring iCODIEST
        {
get
            { return CODIEST; }
set
            { CODIEST = value; }

        }

privatestring TIPMAT;
publicstring iTIPMAT
        {
get
            { return TIPMAT; }
set
            { TIPMAT = value; }

        }

privatestring SEMESTRE;
publicstring iSEMESTRE
        {
get
            { return SEMESTRE; }
set
            { SEMESTRE = value; }

        }


privatestring ASIGNATURA;
publicstring iASIGNATURA
        {
get
            { return ASIGNATURA; }
set
            { ASIGNATURA = value; }

        }

privatestring DOCENTE;
publicstring iDOCENTE
        {
get
            { return DOCENTE ; }
set
            { DOCENTE = value; }

        }

privatestring NIVEL;
publicstring iNIVEL
        {
get
            { return NIVEL; }
set
            { NIVEL = value; }

        }
```

```
privatestring PARALELO;
publicstring iPARALELO
        {
get
            { return PARALELO ; }
set
            { PARALELO = value; }

        }

privatestring PERIODO ;
publicstring iPERIODO
        {
get
            { return PERIODO; }
set
            { PERIODO = value; }

        }

privatestring ESTUDIANTE;
publicstring iESTUDIANTE
        {
get
            { return ESTUDIANTE; }
set
            { ESTUDIANTE = value; }

        }

privatestring TIPO_MATERIA;
publicstring iTIPO_MATERIA
        {
get
            { return TIPO_MATERIA; }
set
            { TIPO_MATERIA = value; }

        }

privateConexion conection = newConexion();

publicDataSet update(DataSet dataset)
        {
SqlConnection cnn = newSqlConnection(conection.conexionC());
            cnn.Open();
SqlCommand cmd = newSqlCommand("SPSA_MTDATNOTAS", cnn);
//cmd.Parameters.Add("@i_BANDERA", SqlDbType.NVarChar,1).Value = dato[0];
            cmd.Parameters.Add("@CODNOT", SqlDbType.Int).Value =
Convert.ToInt32(iCODNOT);
            cmd.Parameters.Add("@NOT1NOT", SqlDbType.Decimal).Value =
Convert.ToDecimal(iNOT1NOT);
            cmd.Parameters.Add("@NOT2NOT", SqlDbType.Decimal).Value =
Convert.ToDecimal(iNOT2NOT);
            cmd.Parameters.Add("@NOT3NOT", SqlDbType.Decimal).Value =
Convert.ToDecimal(iNOT3NOT);
            cmd.Parameters.Add("@EXANOT", SqlDbType.Decimal).Value =
Convert.ToDecimal(iEXANOT);
```

```
cmd.Parameters.Add("@EQUINOT", SqlDbType.Decimal).Value =
Convert.ToDecimal(iEQUINOT);
cmd.CommandType = CommandType.StoredProcedure;
            cmd.ExecuteNonQuery();
return dataset;
        }

    }
}
```

CAPA ENTIDADES USUARIO

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ENTIDADES
{
publicclassENTUsuario
    {
// PARA @iOP
privatestring iOP;
publicstring IOP
        {
get { return iOP; }
set { iOP = value; }
        }

// PARA @iDATO
privatestring iDATO;
publicstring IDATO
            {
get { return iDATO; }
set { iDATO = value; }
}
//PARA CODIGO
privatestring iCODIGO;
publicstring ICODIGO
            {
get { return iCODIGO; }
set { iCODIGO = value; }
            }
//PARA NOMBRE DE USUARIO
privatestring iUSUARIO;
publicstring IUSUARIO
            {
get { return iUSUARIO; }
set { iUSUARIO = value; }
            }
//PARA PASSWORD
privatestring iCLAVE;
publicstring ICLAVE
            {
get { return iCLAVE; }
set { iCLAVE = value; }
            }
//PARA PASSWORD DECIFRADO
privatestring iCLAVEDES;
publicstring ICLAVEDES
```

```
            {
get { return iCLAVEDES; }
set { iCLAVEDES = value; }
            }

//PARA CODIGO DEL ROL
privatestring iROLUSU;
publicstring IROLUSU
            {
get { return iROLUSU; }
set { iROLUSU = value; }
}

//PARA EL ROL
privatestring iROL;
publicstring IROL
            {
get { return iROL; }
set { iROL = value; }
}


//PARA EL ROL
privatestring iCONECTADO;
publicstring ICONECTADO
            {
get { return iCONECTADO; }
set { iCONECTADO = value; }
            }
    }
}
```

CAPA NEGOCIO AUTENTIFICACION

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.IO;

namespace NEGOCIOS
{
publicclassAutentificacion
    {
publicstaticbool Autenticar(string usuario, string password)
       {

StreamReader leerArchivo = newStreamReader("C:\\CONF\\ESCOCONF.txt");
////TOMAR ESA INFORMACION Y DIRIGIRME A LA BASE DE DATOS
string datos;
           datos = leerArchivo.ReadToEnd();



//consulta a la base de datos
string sql = @"SELECT COUNT(*)
                          FROM VIUSUARIO
                          WHERE NOMUSU = @NOMUSU AND CLAVE = @CLAUSU";
```

```
//instanciar la conexion

            #region
String Conexion;
// cadena de conexion
            Conexion = @"Data source = " + datos + "; Integrated
Security=SSPI";
SqlConnection cnn = newSqlConnection(Conexion);
cnn.Open();
            #endregion


SqlCommand cmd = newSqlCommand(sql, cnn); //ejecutamos la instruccion
            cmd.Parameters.AddWithValue("@NOMUSU", usuario); //enviamos los
parametros
cmd.Parameters.AddWithValue("@CLAUSU", password);

int count = Convert.ToInt32(cmd.ExecuteScalar()); //devuelve la fila afectada

if (count == 0)
returnfalse;
else
returntrue;

        }

publicstaticbool AutenticarDocente(string usuario, string password)
        {

StreamReader leerArchivo = newStreamReader("C:\\CONF\\ESCOCONF.txt");
////TOMAR ESA INFORMACION Y DIRIGIRME A LA BASE DE DATOS
string datos;
            datos = leerArchivo.ReadToEnd();



//consulta a la base de datos
string sql = @"SELECT COUNT(*)
                            FROM VIDOCENTE
                            WHERE NICK = @NOMUSU AND CLAVE = @CLAUSU";

//instanciar la conexion

            #region
String Conexion;
// cadena de conexion
Conexion = @"Data source = " + datos + "; Integrated Security=SSPI";
SqlConnection cnn = newSqlConnection(Conexion);
cnn.Open();
            #endregion


SqlCommand cmd = newSqlCommand(sql, cnn); //ejecutamos la instruccion
            cmd.Parameters.AddWithValue("@NOMUSU", usuario); //enviamos los
parametros
cmd.Parameters.AddWithValue("@CLAUSU", password);
```

```
int count = Convert.ToInt32(cmd.ExecuteScalar()); //devuelve la fila afectada

if (count == 0)
returnfalse;
else
returntrue;

}
    }
}
```

CAPA NEGOCIOS MANEJADOR DOCENTES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using DATOS;
using ENTIDADES;
namespace NEGOCIOS
{
publicclassDocentesManejador
    {
DocentesDALC insDocentes = newDocentesDALC();

publicBoolean insertDocentes(ENTDocentes docentes)
        {

return insDocentes.insertarDoc(docentes);
}

publicList<ENTDocentes> listarDocentes(ENTDocentes apellido)
        {
DocentesDALC objeto = newDocentesDALC();
return objeto.listarDocentes(apellido);
        }


publicBoolean eliminarDocentes(ENTDocentes docentes)
        {

return insDocentes.eliminarDoc(docentes);
        }
    }
}
```

CAPA NEGOCIOS MANEJADOR ESTUDIANTES

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ENTIDADES;
using DATOS;
namespace NEGOCIOS
{
publicclassmanejadorMtdatEst
    {
mtdatestDALC estM = newmtdatestDALC();
publicstring insertarEstudiante(string[] dato)
{
```

```
return estM.insertarEstudiantes(dato);
        }

publicList<ENTMtDatEst> cargarEstudiantes(ENTMtDatEst consulta)
        {

return estM.listarEstudiantes(consulta);
        }

    }
}
```

CAPA NEGOCIOS ASIGNATURAS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using DATOS;
using ENTIDADES;
namespace NEGOCIOS
{
publicclassmanejadorMaterias
    {
publicBoolean insertMaterias(string[] dato)
        {
MateriasDALC insMaterias = newMateriasDALC();
return insMaterias.insertarM(dato);
}

publicList<ENTMateria> listarAsignaturas(ENTMateria nombre)
        {
MateriasDALC objeto = newMateriasDALC();
return objeto.listarAsignaturas(nombre);
        }

publicList<ENTMateria> listarXCodigo(ENTMateria codigo)
        {
MateriasDALC objeto = newMateriasDALC();
return objeto.listarXCodigo(codigo);
        }

publicList<ENTMateria> ListarTodoMaterias()
        {
MateriasDALC objeto = newMateriasDALC();
return objeto.ListarMaterias();

        }
publicbool Eliminar(string[] dato)
        {
MateriasDALC objetoEliminar = newMateriasDALC();
return objetoEliminar.Eliminar(dato);
        }
    }
}
```

CAPA NEGOCIOS NOTAS

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using ENTIDADES;
using DATOS;
using System.Data;
namespace NEGOCIOS
{
publicclassmanejadorNotas
    {
ENTMtDatNot notaE = newENTMtDatNot();
mtDatNotDALC notaD = newmtDatNotDALC();
publicBoolean insertarNotas(string[] datos)
        {
return notaD.inserNotas(datos);

        }

publicBoolean eliminaNotas(string[] datos)
        {
return notaD.eliminarNotas(datos);

        }

publicList<ENTMtDatNot> listaNotas(ENTMtDatNot notas)
        {
return notaD.listarNotas(notas);
        }

publicList<ENTMtDatNot> verNotas(ENTMtDatNot notas)
{
return notaD.verNotas(notas);
        }


publicDataSet IngresarNotas(DataSet dtsFunciones)
        {
mtDatNotDALC notD = newmtDatNotDALC();
return notD.IngresarNotas(dtsFunciones);
        }

    }
}
```

CAPA NEGOCIOS USUARIO

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ENTIDADES;
using ENTIDADES;
namespace NEGOCIOS
{
publicclassmanejadorUsuarios
{
usuarioDALC usuarioD = newusuarioDALC();
publicstring crearUsuario(string[] dato)
{
return usuarioD.insertarUsuario(dato);
        }
```

```
publicstring eliminaUsuario(string[] dato)
        {
return usuarioD.eliminarUsuario(dato);
        }

publicList<ENTUsuario> cargarUsuarios(ENTUsuario consulta)
        {
return usuarioD.listarDato(consulta);
        }

    }
}
```

Bibliografía

DRAKE, J. M. (2011). *INGENIERIA EN SISTEMAS.* Recuperado el LUNES de MARZO de 2015, de http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M7_09_VerificacionValidacion-2011.pdf

Estandares de Programaciòn. (s.f.). Recuperado el lunes de Febrero de 2015, de http://sistemas.mag.go.cr/SoporteTecnico/Est%C3%A1ndares%20de%20Sistemas.pdf

GARCIA GIL, V. L. (s.f.). *Planeacion Administrativa.* Villahermosa, Tabasco. Recuperado el domingo de Marzo de 2015, de http://www.monografias.com/trabajos33/planeacion-administrativa/planeacion-administrativa5.shtml

GUZMAN VALDEZ, L. (2005). *LENGUAJES DE PROGRAMACION.* UNIVERSIDAD AUTONÓMA DEL NORESTE: THOMPSON. Recuperado el JUEVES de FEBRERO de 2015, de http://www.monografias.com/trabajos26/lenguajes-programacion/lenguajes-programacion.shtml#biblio

Kendal&Kendall. (s.f.). *Análisis y Diseño de Sistemas* (3º Edicion ed.). Pearson Educacion. Recuperado el Domingo de Marzo de 2015, de http://www.monografias.com/trabajos94/metodologia-y-analisis-s-i/metodologia-y-analisis-s-i.shtml

McLeod Mr, R. (2000). *Sisttemas de informacion gerencial* (7ª ed.). Ciudad de Granda: Prentice Hall. Obtenido de http://elvex.ugr.es/idbis/db/docs/intro/A%20Sistemas%20de%20Informaci%C3%B3n.pdf

PEÑA AYALA, A. (2006). *Ingenieria de Software* (Vol. PRIMERA EDICION). MEXICO, MEXICO: Printed in México. Obtenido de http://www.wolnm.org/apa/articulos/ingenieria_software.pdf

Prueba de carga. (2015). DOCSETOOLS. Recuperado el martes de MARZO de 2015, de http://docsetools.com/articulos-educativos/article_11477.html