



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA EN SISTEMAS

***“ANÁLISIS DEL LENGUAJE UML OCL 2.0 A TRAVÉS DEL FRAMEWORK ECO-NET
GUIADOS POR MODELOS. CASO PRÁCTICO: EMISIÓN DEL CERTIFICADO DE NO
ADEUDAR A NINGUNA DEPENDENCIA PREVIO A LA OBTENCIÓN DEL TÍTULO
ACADÉMICO EN LA ESPOCH.”***

TESIS DE GRADO

**PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS
INFORMÁTICOS**

Presentado por:

MARÍA GABRIELA CÁRDENAS HINOJOSA

NOEMI ELIZABETH OROZCO PILCO

Riobamba – Ecuador

2011

AGRADECIMIENTO

Muchas han sido las personas que de manera directa o indirecta nos han ayudado en la elaboración de esta tesis. Queremos dejar constancia de todas ellas y agradecerles con sinceridad su participación.

De gran importancia es para nosotros mencionar la inmensa gratitud que debemos a nuestros padres por apoyarnos en todo cuanto hizo falta para que nos sintiéramos tranquilas y con ánimos para seguir adelante.

Debemos agradecer de manera especial y sincera al Ing. Wladimir Castro por aceptarnos para realizar esta tesis bajo su dirección. Su apoyo y confianza en nuestro trabajo y su capacidad para guiar nuestras ideas ha sido un aporte invaluable.

Queremos expresar también nuestro más sincero agradecimiento al Ing. Byron Vaca por su importante aporte y participación activa en el desarrollo de esta tesis.

DEDICATORIA

Dedicamos este proyecto y toda nuestra carrera politécnica a Dios por ser quien ha estado a nuestro lado en todo momento dándonos las fuerzas necesarias para continuar luchando día tras día y seguir adelante rompiendo todas las barreras que se nos presenten.

A nuestros padres por el apoyo incondicional, el sacrificio y esfuerzo que nos brindan constantemente para la culminación del presente trabajo y nuestros estudios.

Gabriela

Elizabeth

FIRMAS RESPONSABLES Y NOTAS

NOMBRES	FIRMAS	FECHA
Ing. Iván Ménes DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA	_____	_____
Dr. Raúl Rosero DIRECTOR DE LA ESCUELA DE INGENIERÍA EN SISTEMAS	_____	_____
Ing. Wladimir Castro DIRECTOR DE TESIS	_____	_____
Ing. Gloria Arcos MIEMBRO	_____	_____
Lcdo. Carlos Rodríguez DIRECTOR DEL CENTRO DE DOCUMENTACIÓN	_____	_____

NOTA: _____

RESPONSABILIDAD DEL AUTOR

“Nosotras María Gabriela Cárdenas Hinojosa y Noemí Elizabeth Orozco Pilco somos responsables de las ideas, doctrinas y resultados expuestos en esta Tesis, y el patrimonio intelectual de la misma pertenecen a la Escuela Superior Politécnica de Chimborazo”.

MARÍA GABRIELA CÁRDENAS HINOJOSA

NOEMI ELIZABETH OROZCO PILCO

ÍNDICE DE ABREVIATURAS

ASP.NET	Active Server Pages / Servidor de páginas activas entorno web
BD	Base de Datos
CASE	Ingeniería de Software Asistida por Computadores
CODEGEAR	Empresa desarrolladora de Software
DBMS	Sistema Administrador de Base de Datos
ECO	Enterprise Core Object
ESPOCH	Escuela Superior Politécnica de Chimborazo
HW	Hardware
IBM	International Business Machines Corporation
IEEE	Instituto de Ingeniería Eléctrica y Electrónica
OCL	Object Constraint Language / Lenguaje de restricción de objetos
ODBC	Open Database Connectivity / Conectividad de apertura de base de datos
SECNA	Aplicación Web para la Emisión de Certificados de No Adeudar a la ESPOCH
SRS	Estándar básico para requerimientos de software
SW	Software
UML	Unified Modeling Language / Lenguaje de modelado unificado
VCL	Visual Class Library / Biblioteca de componentes visuales

ÍNDICE GENERAL

PORTADA	
AGRADECIMIENTO	
DEDICATORIA	
FIRMAS RESPONSABLES Y NOTAS	
RESPONSABILIDAD DEL AUTOR	
ÍNDICE DE ABREVIATURAS	
ÍNDICE GENERAL	
ÍNDICE DE TABLAS	
ÍNDICE DE ILUSTRACIONES	
INTRODUCCIÓN	

CAPITULO I MARCO REFERENCIAL

1.1	PROBLEMATIZACION	14
1.2	JUSTIFICACION DEL PROYECTO DE TESIS	16
1.2.1	JUSTIFICACIÓN TEÓRICA	16
1.2.2	JUSTIFICACIÓN METODOLÓGICA	17
1.2.3	JUSTIFICACIÓN PRÁCTICA	18
1.2.4	JUSTIFICACIÓN CIENTÍFICA	18
1.3	OBJETIVOS	19
1.3.1	OBJETIVO GENERAL	19
1.3.2	OBJETIVOS ESPECÍFICOS	19
1.4	HIPOTESIS	19

CAPITULO II MARCO TEORICO

2.1	LENGUAJE DE MODELADO UNIFICADO (UML)	20
2.1.1	MODELADO VISUAL	20
2.1.2	QUE ES UML	21
2.1.3	DIAGRAMAS DE ESTRUCTURAS ESTÁTICAS	22
2.1.3.1	CLASES	22
2.1.3.2	ATRIBUTOS	22
2.1.3.3	GENERALIZACIÓN	22
2.1.3.4	ASOCIACIONES	23
2.1.4	DIAGRAMAS UML	24
2.1.4.1	DIAGRAMAS DE CASOS DE USOS	24
2.1.4.2	DIAGRAMAS DE CLASES	24
2.1.4.3	DIAGRAMAS DE SECUENCIA	25
2.2	LENGUAJE DE ESPECIFICACIÓN OCL 2.0	26
2.2.1	CONCEPTO DE OCL	26
2.2.2	CARACTERÍSTICAS DEL LENGUAJE OCL	26
2.2.3	EXPRESIONES OCL	28
2.2.4	NAVEGACIONES	30
2.2.5	TIPOS PREDEFINIDOS	31
2.2.6	OPERACIONES	31
2.2.7	COLECCIONES	32
2.2.8	REGLAS DE PRECEDENCIA	32
2.2.9	EL PAQUETE ESTANDAR OCL	33
2.2.10	GRAMATICA	34
2.3	FRAMEWORK ECO.NET	34
2.3.1	DEFINICIONES	34
2.3.2	CARACTERÍSTICAS	35
2.3.3	CONSTRUCCION DE UN MODELO EN ECO	35
2.3.4	MODELAR	35
2.4	COMPARACIÓN DE LA TECNOLOGIA ECO FRENTE A LINQ	38

CAPÍTULO III GUIA DE IMPLEMENTACIÓN DEL FRAMEWORK ECO

3.1	INTRODUCCION	40
3.2	REQUISITO DEL SISTEMA	42
3.3	CONSTRUCCION DE UNA APLICACIÓN CON ECO	43
3.4	CONEXIÓN A LA BASE DE DATOS	46
3.5	CREAR EL MODELO EN ECO	48
3.6	CONFIGURACIÓN DE ASP.NET	53
3.7	HERRAMIENTA DE PROTOTIPOS	54
3.8	GENERAR EL ESQUEMA DE BASE DE DATOS	61
3.9	INTERFAZ GRÁFICA DE USUARIO	64
CAPITULO IV IMPLEMENTACION DEL SISTEMA SECNA		
4.1	METODOLOGIA DE LA IMPLEMENTACION	80
4.1.1	INTRODUCCION	80
4.1.2	METODOLOGÍA XP	81
4.1.2.1	CARACTERÍSTICAS FUNDAMENTALES	82
4.1.2.2	FASES DE LA PROGRAMACIÓN EXTREMA (XP).	83
4.2	ESTUDIO PRELIMINAR	85
4.2.1	INTRODUCCION	86
4.2.2	ANTECEDENTES TECNOLOGICOS DE LA INSTITUCION	86
4.2.2.1	RED INSTITUCIONAL	87
4.2.2.2	ACCESO SATELITAL A INTERNET	87
4.2.2.3	INFRAESTRUCTURA DE SERVIDORES	87
4.2.2.4	RECURSOS SOFTWARE	87
4.2.2.5	RECURSOS HARDWARE	88
4.2.3	DEFINICIÓN DE ALTERNATIVAS DE SOLUCIÓN	90
4.2.4	SELECCIÓN DE FUENTES DE INFORMACION	93
4.3	ESPECIFICACION DE REQUERIMIENTOS	93
4.3.1	AMBITO DEL SISTEMA	93
4.3.2	BENEFICIOS DEL SISTEMA	94
4.3.3	FASE I: PLANIFICACIÓN Y ANÁLISIS DEL PROYECTO	94
4.3.3.1	DEFINICION DE LOS REQUERIMIENTOS DEL USUARIO	95
4.3.3.2	DEFINICION DE LOS CASOS DE USO ESENCIALES Y EL FORMATO EXPANDIDO	96
4.3.4	FASE 2: DISEÑO	105
4.3.4.1	MODELO CONCEPTUAL	105
4.3.4.2	DIAGRAMA DE SECUENCIA	106
4.3.4.3	DIAGRAMA DE COMPONENTES	108
4.3.4.4	DIAGRAMAS DE NODOS	109
4.3.4.5	GLOSARIO DE TÉRMINOS	109
4.3.4.6	ANÁLISIS DE RIESGOS	111
4.3.5	FASE 3: CODIFICACIÓN	113
CAPITULO V ANALISIS Y RESULTADOS		
5.1	ANÁLISIS CUANTITATIVO Y CUALITATIVO	166
5.2	CONCLUSIONES DE LA HIPÓTESIS	170
CONCLUSIONES		
RECOMENDACIONES		
RESUMEN		
SUMMARY		
GLOSARIO		
BIBLIOGRAFIA		
ANEXOS		

ÍNDICE DE TABLAS

Tabla N° I. 1. Justificación Científica	18
Tabla N° II. 1. Comparación tecnológica ECO y LINQ	39
Tabla N° IV. 1. Especificaciones Técnicas	90
Tabla N° IV. 2. Especificaciones de Servidores	91
Tabla N° IV. 3. Eventos externos del sistema	97
Tabla N° V. 1. Escala de Valoración cualitativa	168
Tabla N° V. 2. Tabla de comparación cualitativa	168
Tabla N° V. 3. Tabla de comparación cuantitativa	168
Tabla N° V. 4. Tabla de medición de esfuerzos	169

ÍNDICE DE ILUSTRACIONES

Figura II. 1 Representación de una clase	22
Figura II. 2 Representación de Generalización	23
Figura II. 3 Representación de una asociación	23
Figura II. 4 Representación del diagrama de casos de uso	24
Figura II. 5 Representación del diagrama de clases	25
Figura II. 6 Representación del diagrama de secuencia	25
Figura II. 7 Ejemplo de Invariantes	29
Figura II. 8 Instancia Contextual	30
Figura II. 9 Pre y Pos condición	31
Figura III. 10 Crear una nueva aplicación	44
Figura III. 11 Ventana de selección de un nuevo ítem	43
Figura III. 12 Asistente de proyectos ECO	44
Figura III. 13 Proceso de compilación	44
Figura III. 14 Ventana de Guardar aplicación	45
Figura III. 15 Ventana EcoSpaceUnit	45
Figura III. 16 Selección de sqlConnections	46
Figura III. 17 Conexión de la Base de Datos	48
Figura III. 18 Área de Trabajo	48
Figura III. 19 Crear un modelo UML en ECO	49
Figura III. 20 Añadir atributos a diagrama de clases	50
Figura III. 21 Propiedades del Diagrama UML	50
Figura III. 22 Restricciones OCL	51
Figura III. 23 Expresión OCL para el Objeto persona	51
Figura III. 24 Agregar Generalización	52
Figura III. 25 Multiplicidad de los diagramas de clase	53
Figura III. 26 Propiedades de la asociación	53
Figura III. 27 Representación de los diagramas de clase	54
Figura III. 28 Configuración de ASP.NET	54
Figura III. 29 Opciones de ASP.NET	55
Figura III. 30 Opciones de prototipo	55
Figura III. 31 Iniciar XML Persistence	56
Figura III. 32 Clases y OCL	57
Figura III. 33 Añadir un Usuario	57
Figura III. 34 Añadir Permiso	58
Figura III. 35 Propiedades del Usuario	59
Figura III. 36 Acciones OCL	59
Figura III. 37 Editor OCL del objeto Usuario	59
Figura III. 38 Evaluación del Objeto Usuario	59
Figura III. 39 Selección de paquetes UML	60
Figura III. 40 Paquete CoreClassesUnit.CoreClasses	60
Figura III. 41 Paquetes seleccionados de UML	61
Figura III. 42 Compilación del modelo	61
Figura III. 43 Proceso de generación del esquema de BD	62
Figura III. 44 Generación del esquema de BD	62
Figura III. 45 Representación de BD en SQL Server 2005	63
Figura III. 46 Sentencias SQL	64
Figura III. 47 Diseño de Interfaz de Usuario	65
Figura III. 48 Añadir control DataGridView	65
Figura III. 49 Opciones GridView Tasks	66

Figura III. 50 Insertar Áreas de texto	66
Figura III. 51 Vista en el Explorador	68
Figura III. 52 Añadir un AutoForm	68
Figura III. 53 Renombrar el AutoForm	69
Figura III. 54 Propiedades del AutoForm	69
Figura III. 55 Representación AutoForm	69
Figura III. 56 Agregar User Control	69
Figura III. 57 Renombrar Control Menú	70
Figura III. 58 Área de diseño de Construcción del menú	70
Figura III. 59 Agregar el componente Web Navigator	71
Figura III. 60 Añadir elementos al menú	71
Figura III. 61 Añadir SubItems al menú	72
Figura III. 62 Propiedades de la herramienta menú	72
Figura III. 63 Insertar Tabla al formulario AutoForm	73
Figura III. 64 Agrupar Componentes	73
Figura III. 65 Insertar tabla a la página default.aspx	74
Figura III. 66 Reagrupar Componentes en el Formulario Default	74
Figura III. 67 Insertar código en el Default.aspx	75
Figura III. 68 Código para agregar un nuevo usuario	76
Figura III. 69 Visualización en el navegador	76
Figura III. 70 Agregar un nuevo usuario	78
Figura III. 71 Resultados de la agregar Usuario	78
Figura III. 72 Sentencias SQL de inserción de datos	79
Figura III. 73 Rediseño de la aplicación	79
Figura III. 74 Visualización de Resultados	79
Figura IV. 75 Esquema de la Solución Informática	81
Figura IV. 76 Características fundamentales de XP	84
Figura IV. 77 Metodología XP	86
Figura IV. 78 Arquitectura del Sistema SECNA	93
Figura V. 79 Cuadro estadístico de los parámetros de comparación	169
Figura V. 80 Líneas de código con ECO	170
Figura V. 81 Líneas de código sin ECO	170
Figura V. 82 Gráfico de barras - Líneas de código	170
Figura V. 83 Tiempo de Desarrollo	171

INTRODUCCIÓN

En la actualidad la mayoría de empresas u organizaciones están encaminadas a automatizar a través de sistemas informáticos los procesos manuales, utilizando el medio más difundido que es el Internet. Por lo que desarrollar una aplicación web conlleva a difundir a nivel mundial la información necesaria para obtener beneficios importantes.

La tecnología ECO realmente es equivalente a un desarrollo rápido guiado por modelos; reduce drásticamente la cantidad de código que es necesario teclear manualmente, acelerando el despliegue y mejorando la mantenibilidad de sus aplicaciones. Utiliza un subconjunto del lenguaje UML; emplea OCL, lenguaje de restricción de objetos, para definir expresiones sobre los modelos UML.

Herramientas de desarrollo web existen tanto comerciales como de uso libre. Es por ello que la presente tesis se enfoca al análisis del Lenguaje UML y OCL a través del Framework ECO.NET bajo la firma comercial de CodeGear; la misma que se orienta en el manejo directo de instancias de clases sin introducir líneas de código desde la aplicación misma, optimizando de esta manera el tiempo del desarrollador. Además implementando el sistema informático que automatizará el proceso de emisión de certificados de no adeudar a ninguna dependencia previa la obtención del título académico en la Escuela Superior Politécnica de Chimborazo (SECNA), con la metodología de construcción de aplicaciones web XP (extreme programming).

Con los argumentos expuestos, la presente tesis consta de cinco capítulos que se los describirán a continuación. El primer capítulo contiene una breve descripción y justificación del tema de tesis, así como los objetivos que se desean alcanzar durante el desarrollo y como se lo va a comprobar mediante la formulación de la hipótesis. En el segundo capítulo

se presentarán las definiciones de las tecnologías como: UML, OCL, ECO y LINQ, las mismas que nos permitirán conocer sus principales características y utilidades. En el tercer capítulo se desarrollará una guía de implementación del Framework Eco para el desarrollo de aplicaciones utilizando la tecnología ECO.NET. El capítulo cuatro consta la implementación del sistema SECNA. En el capítulo cinco se analizará y comprobará la hipótesis definida en el capítulo uno y se plasmarán los resultados obtenidos en el desarrollo de la presente tesis. Teniendo como resultados la optimización del proceso de obtención del certificado de no adeudar a ninguna dependencia de la ESPOCH, además que las dependencias que no cuenten con una automatización del proceso de préstamos se les proporciona la opción de a través del sistema SECNA tener un control sobre los mismos.

CAPITULO I

MARCO REFERENCIAL

1.1 PROBLEMATIZACION

Las bases de datos se han constituido como una de las herramientas ampliamente difundidas en la actualidad, utilizadas en cuanto a recuperación y almacenamiento de información en todos los campos.

El uso de sistemas de bases de datos automatizadas, se desarrolló a partir de la necesidad de almacenar grandes cantidades de datos, producidas por las nuevas industrias que creaban mayor cantidad de información.

Existen tecnologías alternativas como ECO que permiten eliminar sentencias SQL optimizado procesos de desarrollo de la aplicación, sin tener relación con el manejador de base de datos

OCL permite a los desarrolladores de software escribir restricciones sobre modelos de objetos. Estas restricciones son particularmente útiles, en la medida en que permiten a

los desarrolladores crear un amplio conjunto de reglas que rigen el aspecto de un objeto individual.

En particular no hay necesidad de preocuparse del diseño de la base de datos que hacen las aplicaciones ECO más objetos relacionales y menos orientados a bases de datos. La construcción del modelo en ECO se hace en un editor visual de UML. El código correspondiente es generado rápidamente en cualquiera de los lenguajes.

Existen tecnologías similares que permiten el manejo transparente de la base de datos como lo es LINQ, XPO de DevExpress, presentando estas tecnologías en comparación a ECO limitaciones en el manejo de los datos, y del motor de base de datos.

Una de las tecnologías que hace competencia a ECO es LINQ, que tiene características similares, pero este solo soporta a la familia de SQL Server, además que no permite mapeo complejos. Trabaja con Visual Studio 2008 bajo los lenguajes C# y VB. Mientras que ECO provee un marco de trabajo corporativo completo y adaptable para el desarrollo de aplicaciones .NET guiadas por modelos para Windows y la web.

La Escuela Superior Politécnica de Chimborazo, institución encargada de formar profesionales capaces de desenvolverse en diferentes aéreas de la carrera, gracias a los conocimientos científicos, tecnológicos y valores humanos adquiridos por cada uno de los estudiantes, previo a la obtención del en cualquier oferta académica con las que cuenta la ESPOCH, se debe cumplir con los requisitos necesarios acordes a la facultad, estando estos tipificados en el reglamento institucional.

Entre los requisitos se encuentra el listado de recolección de firmas de no adeudar en las dependencias de la Institución. Este listado varía de acuerdo a las políticas establecidas en cada Facultad. Todo este proceso es manual, por lo que toma tiempo lograr llenar el documento con las firmas requeridas.

El Departamento de Sistemas y Telemática de la ESPOCH, encargado de desarrollar y mantener los sistemas informáticos administrativos, académicos y de la organización; así como de apoyar los procesos de modernización administrativa, académica y de gestión institucional; requiere automatizar el proceso de emisión del certificado de no adeudar a ninguna dependencia previo a la obtención del título académico, en cada una de las facultades existentes en la Institución. El mismo que facilitará y acortará el tiempo de obtención del requisito.

Existe un caso práctico similar a nuestra propuesta de anteproyecto, que se desarrolló como caso de estudio y no se implantó debido al costo elevado de adquisición de un data warehouse, ya que no se justifica para los pocos datos que se maneja en la institución; el costo de implantación para nuestra propuesta es accesible ya que se desarrollará con herramientas que la institución maneja y de bajo costo.

1.2 JUSTIFICACION DEL PROYECTO DE TESIS

1.2.1 JUSTIFICACIÓN TEÓRICA

ECO es equivalente a un desarrollo rápido guiado por modelos; reduce drásticamente la cantidad de código que es necesario teclear manualmente, acelerando la distribución de la aplicación y mejorando el mantenimiento general de sus aplicaciones.

Utiliza un subconjunto de lenguaje UML, ECO emplea OCL, el lenguaje de restricciones sobre objetos, para definir expresiones sobre los modelos UML.

La tecnología ECO da un paso más en esta línea siendo un novedoso marco de trabajo guiado por el diseño para .net y basado en diagramas UML. Esta técnica de desarrollo es conocida como “arquitectura dirigida por modelos”.

Pero ECO va más allá de ser una simple herramienta de modelado, ECO permite la generación automática de código en virtud del modelo y un esquema de persistencia en virtud del motor de base de datos. ECO no es solo una guía para la codificación, cuando se trabaja con ECO no solo se está diseñando o inspeccionando sino que también se está codificando.

1.2.2 JUSTIFICACIÓN METODOLÓGICA

Se crea este sistema con el afán de brindar solución a los problemas que surgen durante el proceso de recolección de firmas previo a la obtención del título profesional.

Para el desarrollo de la aplicación se cree conveniente realizar un análisis de la Metodología del lenguaje para modelado visual de aplicaciones UML y del Lenguaje de restricción de objetos OCL, la cual reúne características de ciertos autores y las aplica efectivamente al desarrollo de una solución adecuada al planteamiento del problema especificado.

Herramientas:

Como herramientas se ha escogido BDS 2007, ya que incluye un soporte para ECO en aplicaciones ASP.NET. ECO es un motor de persistencia de datos con generación automática de código que utiliza un marco de trabajo basado en modelado UML. ECO provee también un evaluador OCL. Como gestor de base de datos se puede utilizar InterBase, Sql Server, IBM DB2 UDB, y otro gestor compatible con Eco y que los desarrolladores se adapten.

Se escogió trabajar con el gestor de base de datos SQL Server 2005 que ayudará a desarrollar este sistema, este gestor nos brinda muchas facilidades.

1.2.3 JUSTIFICACIÓN PRÁCTICA

Se ha tomado como parte práctica la implementación de un sistema informático de emisión del certificado de no adeudar a ninguna dependencia previa a la obtención del título profesional en la ESPOCH, puesto que se ve la necesidad de automatizar el proceso de recolección de las mismas, para disminuir los tiempos de obtención. Cada una de las secretarías de las escuelas cuenta con los equipos necesarios para implantar el sistema, con la finalidad de lograr que los datos que se obtengan sean fiables y transparentes, así como mejorar la obtención del documento en tiempo real.

1.2.4 JUSTIFICACIÓN CIENTÍFICA

Al momento en la ESPOCH y en la Facultad de Informática y Electrónica se ha realizado la siguiente tesis:

Tabla I.1 Justificación Científica

TEMA	AUTOR	AÑO
ESTUDIO COMPARATIVO DE HERRAMIENTAS ETL PARA AUTOMATIZAR LA EMISION DE CERTIFICADO DE NO ADEUDAMIENTO PARA Y/O GRADUACION EN LA ESPOCH	LOGROÑO HUILCA NUBE PIEDAD NUÑEZ URQUIZO GLADYS CRISTINA	2008.02.10

El tema realizado al nuestro se diferencia por la forma del manejo de la base de datos, ya que se utilizara un lenguaje de modelado UML y de restricción de objetos OCL, el cual se generará el modelo de base de datos mediante la combinación con el framework ECO. Permitiendo de esta manera reducir el esfuerzo del desarrollador. Además que la institución cuenta con parte de las herramientas de desarrollo, siendo los costos de implantación mínimos a diferencia de la tesis ya desarrollada.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Analizar el lenguaje UML OCL 2.0 a través del framework ECO .Net guiados por modelos aplicado a la emisión del certificado de no adeudar a ninguna dependencia previo a la obtención del título académico en la ESPOCH.

1.3.2 OBJETIVOS ESPECÍFICOS

- Estudiar los lenguajes UML y OCL a través del framework ECO.Net
- Estudiar las ventajas y desventajas que tiene la tecnología ECO frente a la tecnología LINQ.
- Proponer una guía implementación de aplicaciones a través del framework ECO.
- Implementar e implantar el sistema emisión del certificado de no adeudar a las dependencias previa la obtención del título profesional en la ESPOCH, que será desarrollado en el Departamento de Sistemas y Telemática.

1.4 HIPOTESIS

La utilización del lenguaje UML OCL 2.0 a través de la tecnología ECO .NET reducirá el esfuerzo del desarrollador y optimizará la construcción de sistemas informáticos.

CAPITULO II

MARCO TEÓRICO

2.1 LENGUAJE DE MODELADO UNIFICADO (UML)

2.1.1 MODELADO VISUAL

Tal como indica su nombre, UML es un lenguaje de modelado. Un modelo es una simplificación de la realidad. El objetivo del modelado de un sistema es capturar las partes esenciales del sistema. Para facilitar este modelado, se realiza una abstracción y se plasma en una notación gráfica. Esto se conoce como **modelado visual**.

El modelado visual permite manejar la complejidad de los sistemas a analizar o diseñar. UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura hardware donde se ejecuten.

Otro objetivo de este modelado visual es que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

- Mayor rigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.

2.1.2 QUE ES UML

“UML como un lenguaje que proporciona un vocabulario y unas reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema.

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción”¹.

“UML (Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group)”².

“UML es un lenguaje para especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. No se trata de un método de desarrollo, es decir, no sirve para determinar cómo diseñar el sistema, sino que simplemente proporciona una representación gráfica para visualizar el diseño”³.

¹Hernández Enrique Orallo[2002]. El Lenguaje Unificado de modelado. [En línea]

²G. Booch, J. Rumbaugh, I. Jacobson. Addison Wesley Iberoamericana, 1999. El Lenguaje Unificado de modelado. [En línea]

³Concepto de las tesis

2.1.3 DIAGRAMAS DE ESTRUCTURAS ESTÁTICAS

2.1.3.1 CLASES

Una clase define los atributos y los métodos de una serie de objetos. Todos los objetos o instancias de esta clase tienen las mismas propiedades o atributos, con un valor diferente (o no) cada uno de ellos.

En UML, las clases están representadas por cajas rectangulares, con el nombre de la clase, y adicionalmente, atributos y operaciones. De todas maneras, nos centramos únicamente en la parte estructural de los esquemas, razón por la que no consideraremos las operaciones.

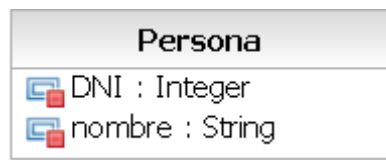


Figura II. 1. Representación de una clase

2.1.3.2 ATRIBUTOS

En UML, los atributos (o propiedades) se muestran al menos con su nombre, y también pueden mostrar su tipo, valor inicial y otras características. En la *figura II.2* la clase persona tiene dos atributos: DNI, de tipo integer, y nombre, de tipo String.

2.1.3.3 GENERALIZACIÓN

Las generalizaciones o herencias permiten que una clase (general o padre) comparta sus atributos y operaciones con una clase derivada de ella (específica o hija). La clase derivada puede alterar/modificar algunos de ellos, así como añadir más atributos y operaciones propias.

En UML, una asociación de generalización entre dos clases, coloca a estas dos (o más clases) en una jerarquía que representa el concepto de herencia.

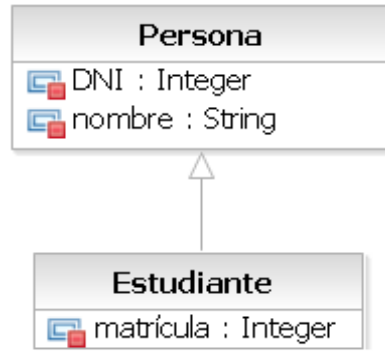


Figura II. 2. Representación de Generalización

2.1.3.4 ASOCIACIONES

Una asociación representa una relación entre N clases, y aporta la semántica común y la estructura de muchos tipos de conexiones entre objetos.

Las asociaciones pueden tener un nombre que especifica el propósito de la asociación y pueden ser unidireccionales o bidireccionales (indicando el tipo de navegabilidad entre ellos). Cada extremo de la asociación también tiene un valor de multiplicidad, que indica cuántos objetos de ese lado de la asociación están relacionados con un objeto del extremo contrario. Además, cada extremo puede tener un nombre de rol, precisamente para aclarar su papel en la asociación.

En UML, las asociaciones se representan por medio de líneas que conectan las clases participantes en la relación, y también pueden mostrar el papel y la multiplicidad de cada uno de los participantes. La multiplicidad se muestra como un rango [mín...máx] de valores no negativos, como se muestra en la *figura II. 3*.

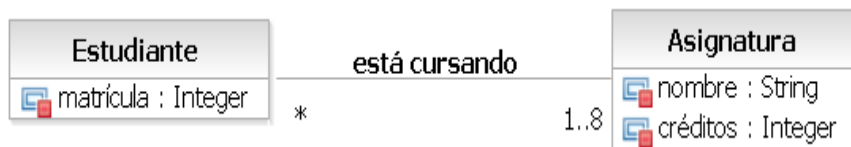


Figura II. 3. Representación de una asociación

2.1.4 DIAGRAMAS UML

Un diagrama es la representación gráfica de un conjunto de elementos con sus relaciones. En concreto, un diagrama ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas

2.1.4.1 DIAGRAMAS DE CASOS DE USOS

El diagrama de casos de usos representa gráficamente los casos de uso que tiene un sistema. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. Es decir, se está diciendo lo que tiene que hacer un sistema y cómo. En la *figura II. 4* se muestra un ejemplo de casos de uso, donde se muestran tres actores (los clientes, los taquilleros y los jefes de taquilla) y las operaciones que pueden realizar (sus roles).

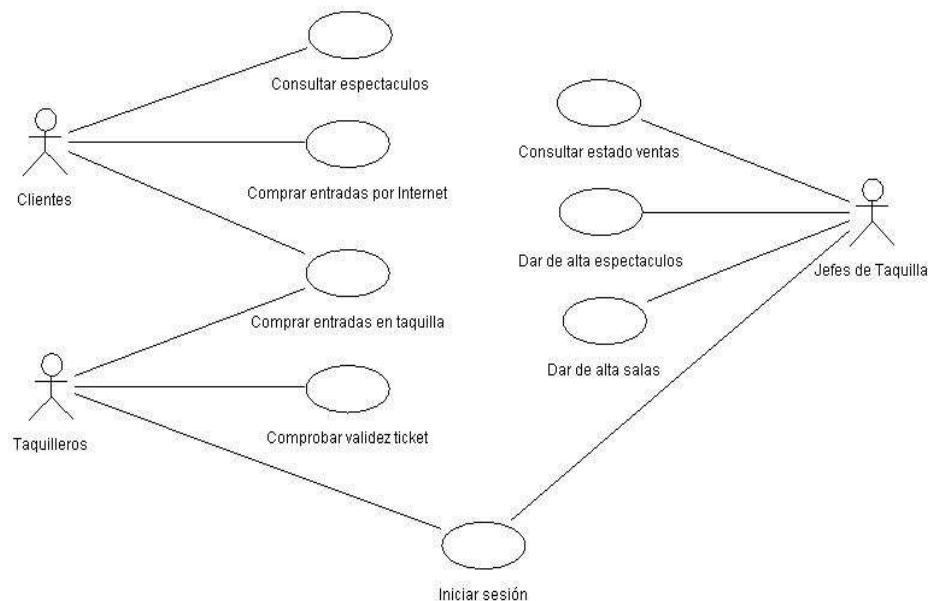


Figura II. 4. Representación del diagrama de casos de uso

2.1.4.2 DIAGRAMAS DE CLASES

El *diagrama de clases* muestra un conjunto de clases, interfaces y sus relaciones. Éste es el diagrama más común a la hora de describir el diseño de los sistemas

orientados a objetos. En la *figura II. 5* se muestran las clases globales, sus atributos y las relaciones de una posible solución al problema de la venta de entradas.

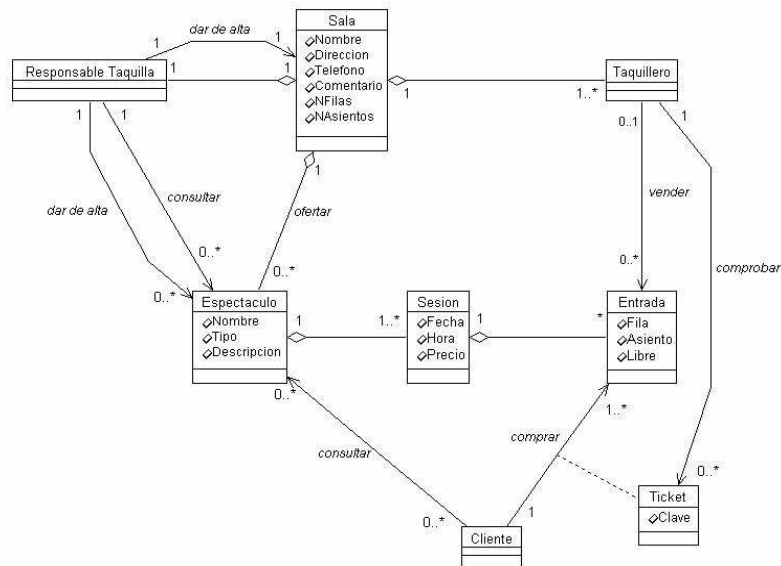


Figura II. 5. Representación del diagrama de clases

2.1.4.3 DIAGRAMAS DE SECUENCIA

En el diagrama de secuencia se muestra la interacción de los objetos que componen un sistema de forma temporal. Siguiendo el ejemplo de venta de entradas, la *figura II.6* muestra la interacción de crear una nueva sala para un espectáculo.

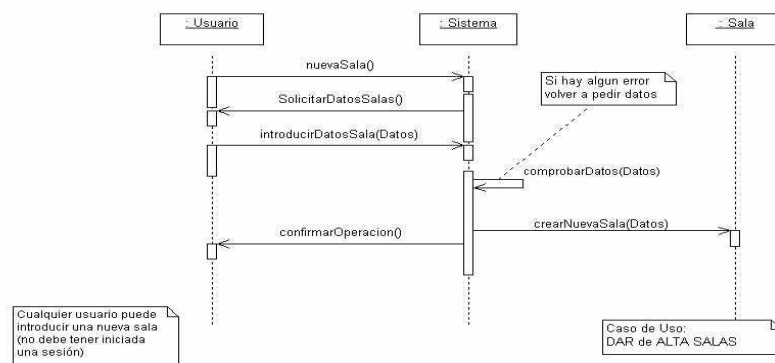


Figura II. 6. Representación del diagrama de secuencia

2.2 LENGUAJE DE ESPECIFICACIÓN OCL 2.0

2.2.1 CONCEPTO DE OCL

“OCL (Object Constraint Language), actualmente en su versión 2.0, fue adoptado en el 2003 por el grupo OMG como parte de UML 2.0. Se trata de un lenguaje para la descripción formal de expresiones en los modelos UML. Tales expresiones pueden representar invariantes, pre condiciones, post condiciones, inicializaciones, reglas de derivación, etc”⁴.

“OCL (Object Constraint Language, lenguaje de restricciones de objetos) es un nuevo lenguaje notacional, un subconjunto del UML estándar industrial, que permite a los desarrolladores de software escribir restricciones sobre modelos de objetos (pre y post condiciones, guardas, invariantes, valores derivados, restricciones sobre operaciones, etc.). Estas restricciones son particularmente útiles, en la medida en que permiten a los desarrolladores crear un amplio conjunto de reglas que rigen el aspecto de un objeto individual”⁵.

2.2.2 CARACTERÍSTICAS DEL LENGUAJE OCL

OCL tiene las características de un lenguaje de expresiones, un lenguaje de modelos y un lenguaje formal:

2.2.2.1 LENGUAJE DE EXPRESIONES

OCL es un lenguaje de expresiones puro. Una expresión OCL garantiza que quedará sin efecto. Esto no puede cambiar nada en el modelo. Esto significa que un estado del sistema nunca cambiará debido a una expresión OCL,

⁴Dolors Costal; María Ribera Sancho; Ernest Teniente Ed. UPC, 2003. Especificación de sistemas software en UML.[En línea]

⁵OCL Juan Casas Cuevas, Mercedes Arenas Fernández. Fac. de Informática Universidad Politécnica de Valencia

incluso una expresión OCL podría usarse para describir tal cambio de estado (p.e. en una post condición). Todos los valores de todos los objetos, incluidos los enlaces, no cambiarán. En cualquier momento en que se evalúa una expresión OCL, simplemente devuelve un valor.

2.2.2.2 LENGUAJE DE MODELOS

OCL es un lenguaje de modelos y no un lenguaje de programación. No se puede escribir un programa lógico o un flujo de control en OCL. Especialmente, no se puede invocar procesos o activar operaciones no de consulta en OCL. Debido a que OCL es en primer lugar un lenguaje de modelos, no se puede asegurar que todo sea directamente ejecutable.

Como lenguaje de modelos, todos los aspectos de implementación están fuera de alcance y no pueden expresarse en OCL. Cada expresión OCL es conceptualmente atómica. El estado de los objetos en el sistema no puede cambiar durante la evaluación.

2.2.2.3 LENGUAJE FORMAL

OCL es un lenguaje formal donde todos los constructores tienen un significado formal definido. La especificación de OCL es parte de la especificación de UML. OCL no tiene la intención de reemplazar los lenguajes formales existentes.

Los lenguajes formales tradicionales se usaban por personas con conocimientos matemáticos, pero dificulta su uso para la mitad de empresas y modeladores de sistemas. OCL ha sido desarrollado para llenar este hueco.

Puesto que en un proyecto hay mucha gente involucrada (usuario, expertos, personas que después se deberán encargar de su mantenimiento, etc.) los

modelos deben ser entendidos por una amplia y variada audiencia. OCL es fácil de aprender y usar por los desarrolladores sin amplios conocimientos matemáticos.

2.2.3 EXPRESIONES OCL

En OCL es posible especificar *expresiones OCL*. Existen dos tipos básicos de expresiones OCL:

- Invariantes
- Pre / post condiciones

Cada expresión OCL está escrita en el contexto de *una instancia* de un tipo específico

2.2.3.1 INVARIANTES

Un *invariante* es una condición que debe ser verdadera para todas las instancias de un tipo específico en cualquier momento. El tipo de la instancia contextual de una expresión OCL, el cual es parte de una invariante, es escrito luego de la palabra reservada *context*. La palabra reservada *inv.* declara que la expresión OCL es una restricción invariante

Ejemplo:

context Habitación

inv. número_de_camas <= 10

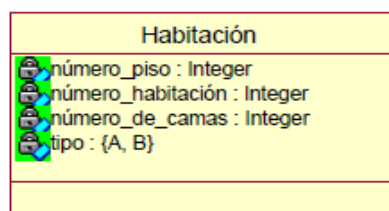


Figura II. 7. Ejemplo de Invariantes

2.2.3.2 INSTANCIA CONTEXTUAL

Cada expresión OCL está escrita en el contexto de *una instancia* de un tipo específico. La palabra reservada *self* es utilizada para referir a la instancia contextual.

context Habitación

inv. self.número_de_camas <= 10

Alternativamente en lugar de *self* es posible definir un *nombre* que represente el rol de *self*

context h: Habitación

inv. h.número_de_camas <= 10

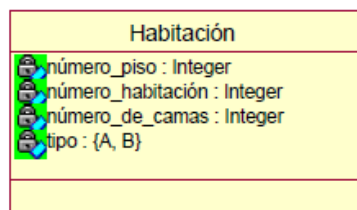


Figura II. 8. Instancia Contextual

2.2.3.3 PRE Y POST CONDICIONES

Una *precondición* / *poscondición* es una condición que debe ser verdadera antes /después de la ejecución de una operación. La instancia contextual *self* es una instancia del tipo que es dueño de la operación.

La declaración del contexto en OCL utiliza la palabra `context`, seguido de la declaración de la operación y el tipo. Las etiquetas `pre` y `post` declaran si se trata de una pre / pos condición.

```
context Baño::usa(): Integer
```

```
post: usos = usos + 1
```

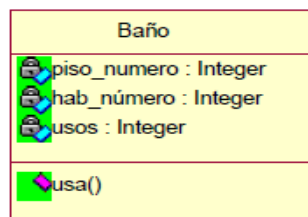


Figura II. 9. Pre y Post Condición

En una pos condición, nos podemos referir a los valores de cada propiedad de un objeto en dos momentos en el tiempo:

- ✓ El valor de la propiedad al comienzo de la operación o método (utilizamos el postfijo `@pre` en expresiones del tipo pos condición)
- ✓ El valor de la propiedad una vez que la operación o método se ha ejecutado.

```
context Empleado::aumentarsalario( cantidad: Real ): Real
```

```
post: self.salario = self.salario@pre + cantidad and result =  
self.salario
```

2.2.4 NAVIGACIONES

- ✓ Una relación es navegada cuando se utiliza en una expresión con el nombre de rol (del extremo opuesto de una relación) que vincula la clase donde se define la expresión con otra clase del diagrama.
- ✓ Una relación es navegada cuando se utiliza en una expresión eventualmente si el nombre de rol se omite, se utiliza el nombre de la clase, a la cual se navega.

- ✓ Las navegaciones no se limitan a una única asociación.
- ✓ Las expresiones OCL pueden estar encadenadas navegando un conjunto de asociaciones.

2.2.5 TIPOS PREDEFINIDOS

En OCL, se definen un número de tipos básicos, los cuales están disponibles para el modelador en todo momento. Estos tipos son valores predefinidos y son independientes de cualquier modelo de objetos.

Tipos Básicos:

- ✓ Boolean: true, false
- ✓ Integer :1, -5, 2, 34, 26524, ...
- ✓ Real: 1.5, 3.14, ...
- ✓ String: 'ser o no ser...'
- ✓ Colecciones: (definidas posteriormente)

Operaciones definidas en los tipos

- ✓ Integer: *, +, -, /, abs(), etc.
- ✓ Real: *, +, -, /, etc.
- ✓ Boolean: and, or, xor, not, implies, if-then-else-endif
- ✓ String: concat(), size(), substring()

Cada expresión OCL se escribe en el contexto de un modelo UML. En UML un *classifier* es una clase, o una interfaz, un tipo de dato. Todos los *classifiers* de un modelo UML son tipos que pueden ser utilizados en las expresiones OCL que estén asociadas a dicho modelo.

2.2.6 OPERACIONES

Sobre las colecciones se pueden realizar varias operaciones: select, collect, forAll, exists, isUnique, size.

- ✓ *Select* Especifica un subconjunto de una colección.
- ✓ *Size* Retorna la cardinalidad de la colección considerada.

2.2.7 COLECCIONES

El tipo *Collection* define un gran número de operaciones predefinidas que permiten al modelador de expresiones OCL manipular colecciones. Las operaciones de colecciones nunca cambian las colecciones originales, sino que guardan el resultado en otra colección.

Colección es un súper tipo abstracto para todos los tipos de colección en OCL. Cada ocurrencia de un objeto en una colección se llama elemento. Si un objeto aparece por segunda vez en una colección, hay dos elementos. Algunas propiedades se pueden definir con el subtipo, el cual significa que hay una post condición adicional o un valor de retorno especial.

La definición de algunas propiedades comunes es diferente para cada subtipo.

OCL distingue tres tipos diferentes de colecciones: *Set*, *Sequence* y *Bag*. *Set* es un conjunto matemático. No contiene elementos duplicados. *Bag* es como un conjunto, que puede contener duplicados. *Sequence* es como un Bag en el que los elementos están ordenados. Para definir una colección, los elementos se encierran entre llaves y están separados por comas. El tipo de la colección se escribe antes de las llaves, como por ejemplo:

Set {1, 2, 5, 88}

Sequence {'mono', 'nuez'}

Bag {1, 3, 4, 3, 5}

2.2.8 REGLAS DE PRECEDENCIA

El orden de precedencia de las operaciones, de mayor a menor prioridad, en OCL es:

- ✓ '@pre
- ✓ punto y operaciones flecha: '.' y '->'
- ✓ operadores unitarios 'not' y menos '-'
- ✓ '*' y '/'
- ✓ '+' y el operador binario '-'
- ✓ 'if-then-else-endif'
- ✓ '<', '>', '<=', '>='
- ✓ '=', '<>'
- ✓ 'and', 'or', y 'xor'
- ✓ 'implies'

Los paréntesis '(' y ')' pueden usarse para variar la precedencia.

2.2.9 EL PAQUETE ESTANDAR OCL

Cada modelo UML que usa como lenguaje de restricción OCL contiene un paquete estándar denominado "UML_OCL". Este paquete es usado por defecto en todos los otros paquetes en el modelo para evaluar expresiones OCL. Este paquete contiene todos los tipos OCL predefinidos y sus características.

Para extender los tipos predefinidos OCL, un modelador debe definir un paquete separado. El paquete estándar de OCL puede ser importado, y cada tipo OCL puede extenderse con nuevas características.

Para especificar que un paquete usó tipos predefinidos OCL de un paquete definido por el usuario en lugar de un paquete estándar, el paquete usado debe definir una dependencia con estereotipo <<Tipo_OCL>> al paquete que define el tipo OCL extendido.

Una restricción, en el paquete OCL definido por el usuario, es que como mínimo todos los tipos predefinidos OCL junto con todas sus características deben estar definidos. El paquete de usuario debe ser una extensión correcta del estándar OCL.

2.2.10 GRAMATICA

La descripción de gramática usa la sintaxis EBNF, donde 'I' significa elección, '?' opcionalidad, y '*' significa cero o más veces, '+' significa una o más veces. En la descripción del nombre, typeName, y string, la sintaxis de los símbolos léxicos se usa el generador de análisis de Java.

2.3 FRAMEWORK ECO.NET

2.3.1 DEFINICIONES

“Enterprise Core Objects es una revolucionaria tecnología de CodeGear (Borland) para el diseño y desarrollo de sistemas hacia el Modelado y control de programas”⁶.

“ECO es equivalente a un desarrollo rápido guiado por modelos; reduce drásticamente la cantidad de código que es necesario teclear manualmente, acelerando la distribución de la aplicación y mejorando el mantenimiento general de sus aplicaciones”⁷.

Utiliza un subconjunto de lenguaje UML, ECO emplea OCL (Object Constraint Language), el lenguaje de restricciones sobre objetos, un estándar del OMG (Object Management Group) para definir expresiones sobre los modelos UML.

La tecnología ECO de CodeGear (Borland) da un paso más en esta línea siendo un novedoso marco de trabajo guiado por el diseño para .net y basado en diagramas UML. Esta técnica de desarrollo es conocida como “arquitectura dirigida por modelos”.

Pero ECO va más allá de ser una simple herramienta de modelado, ECO permite la generación automática de código en virtud del modelo y un esquema de

⁶Norberto Martínez, SmartWare- Guadalajara, México. ECO III y aplicaciones web ASP.NET –Tut. 2. [El Línea]

⁷ Code Gear. All Development Technologies. [El Línea]

persistencia en virtud del motor de base de datos. ECO no es solo una guía para la codificación, cuando se trabaja con ECO no solo se está diseñando o inspeccionando sino que también se está codificando”.

2.3.2 CARACTERÍSTICAS

- Mejoras en el proceso de desarrollo por medio del modelado
- Modelado totalmente orientado a objetos. Mapeo e independencia del modelo relacional de base de datos
- Se consigue drástica reducción de tiempo de desarrollo al evitar la codificación manual de gran parte del código.
- Se consigue reducción de tiempo de desarrollo y de errores al reducir al mínimo la interacción manual con los sistemas de bases de datos.
- Se consigue una separación natural entre la lógica y la presentación.
- Se logran excelentes niveles de reutilización para aplicaciones con interfaces de usuarios multimodales pudiendo acceder a un mismo paquete ECO desde aplicaciones .Net de escritorio o paginas ASP.NET
- Sincronización y actualización automática entre el modelo y el código. El modelo nunca queda obsoleto

2.3.3 CONSTRUCCION DE UN MODELO EN ECO

ECO usa un subconjunto del Lenguaje de Modelado Unificado (UML), como su idioma de modelado. Las partes específicas son el Diagrama de Clase y el Lenguaje de Control de Objetos (Object Constraint Language - OCL).

2.3.4 MODELAR

Los modelos simplifican la comunicación entre diseñadores, usuarios y todos los interesados del sistema. Creando varios tipos de modelos se puede describir un sistema del software complejo en una serie progresivamente más detallada de vistas del dominio del problema.

El problema de usar modelos, de esta manera, es que, cuando se ha refinado a un modelo cada vez más al punto que es completamente inequívoco y ejecutable, ya no se reconoce como una descripción del problema original. Cuando se ha convertido una descripción del problema progresivamente de la descripción del texto al código fuente y esquemas de la base de datos, solo el programador (y el compilador) puede entenderlo. El problema con esto es que la transformación de la lógica de negocio a la lógica de programación tiene que darse cada vez que se modifique el código. Y cuando se visualice el código escrito por otra persona, se necesite hacer la conversión inversa de código fuente a lógica de negocios.

ECO proporciona una manera para quitar la necesidad de realizar manualmente esa transformación. ECO le permite al diseñador modelar y al mismo tiempo, desarrollar la solución de un problema de forma que sea, más estrechamente representada la lógica comercial, y todavía es inequívoco y ejecutable. Esto se refiere a menudo como aumentar el nivel de la abstracción. Haciendo esto, la capacidad de cambiar y de entender software complejo es hecha más fácil y menos costosa.

En ECO las clases que se modela se refieren a menudo como clases del negocio o clases del dominio. En tiempo de ejecución refiere a objetos de negocio u objetos de dominio. La razón que se utiliza estos nombres es porque las clases representan directamente entidades en el dominio del problema del negocio. Por ejemplo, en el caso de la inscripción de la universidad se puede modelar las clases estudiante, profesor y materia.

Ventajas de ECO

Las clases de dominio se definen en un modelo como diagramas de clases y diagramas de estado.

ECO genera tanto la base de datos y el código de la aplicación, liberando tareas vanas.

El esquema de base de datos ECO asegura que los datos existentes no se pierdan, cuando la base de datos se actualiza para que coincida con el nuevo modelo de ahorro de tiempo en cada iteración.

Los desarrolladores pueden construir complejas aplicaciones de negocio o simplemente mucho más rápido y con menos defectos.

ECO está diseñado para el cambio, lo que permite a los desarrolladores implementar rápidamente los cambios impulsados por las necesidades del negocio.

Desventaja de ECO

ECO se puede utilizar con todas las versiones de Microsoft Visual Studio ® 2008, excepto en las versiones Express.

2.4 COMPARACIÓN DE LA TECNOLOGÍA ECO FRENTE A LINQ

Para la respectiva comparación se consideran los siguientes parámetros:

Tabla Nº II. 1. Comparación de tecnología ECO y LINQ

PARÁMETROS	DETALLES	ECO	LINQ
Base de datos	Crear y mantener automáticamente la infraestructura de la BD.	✓	X
	Libre manipulación de los objetos por parte del desarrollador	✓	✓
	Traducción del Lenguaje Integrado de Consultas para ejecución en la BD.	X	✓
	Traducción automática, de las modificaciones a la BD.	✓	✓
	Resuelve automáticamente la persistencia de los objetos en la BD	✓	X
	Generación automática de la base de datos y el código de la aplicación, liberando tareas superficiales.	✓	X
	Facilidad en el diseño de la base de datos	✓	X
	Empleo de INNER JOIN para relaciones entre dos o más tablas	✓	X
	Facilita la implementación de varias arquitecturas para acceso a datos.	✓	✓
Desarrollo	Permite a los desarrolladores utilizar diagramas visuales para construir objetos y relaciones	✓	✓
	Diseño de los diagramas de UML	✓	X
	Restricciones OCL	✓	X
	Permite a los desarrolladores implementar rápidamente los cambios impulsados por las necesidades del negocio.	✓	✓
	Funciona en varios motores de BD.	✓	X
	Permite elevar el nivel de abstracción y claridad del desarrollador.	✓	✓
Seguridad	El esquema de base de datos asegura que los datos existentes no se pierdan.	✓	X

	Seguridad de tipos y chequeo en tiempo de compilación en las expresiones de consulta.	✓	✓
Tiempo	Proporciona una infraestructura en tiempo de ejecución para administrar los datos relacionales como objetos.	✓	✓
	Simplifica el código notablemente para complejas aplicaciones.	✓	X
	Aumenta la capacidad de almacenamiento y se reduce el tiempo de respuesta	✓	✓
	Tratamiento ineficaz de los datos (no se puede modificar, actualizar todos los datos al mismo tiempo)	✓	X
	Cuando la base de datos se actualiza para que coincida con el nuevo modelo, la cual existe un ahorro de tiempo en cada iteración	✓	X
	Menor tiempo en la codificación y generación del esquema de BD.	✓	X

Después de analizar la tabla comparativa se determina que usar la tecnología ECO proporciona un 95.65% confiabilidad, rapidez, seguridad y disminución de tiempos en codificación. Mientras que LINQ es efectiva en un 43.47% en la construcción de aplicaciones.

CAPÍTULO III

GUIA DE IMPLEMENTACIÓN DEL FRAMEWORK ECO

3.1 INTRODUCCION

Es necesario contar con una documentación en la que se indique como utilizar un software de calidad, ya que a medida que crece el nivel de complejidad de un gran proyecto software, se hace necesaria la planificación y diseño del objetivo que quiere alcanzarse.

UML es el lenguaje estándar para el análisis y diseño de sistemas, permitiendo establecer requerimientos y estructuras necesarias previas al proceso de escribir código.

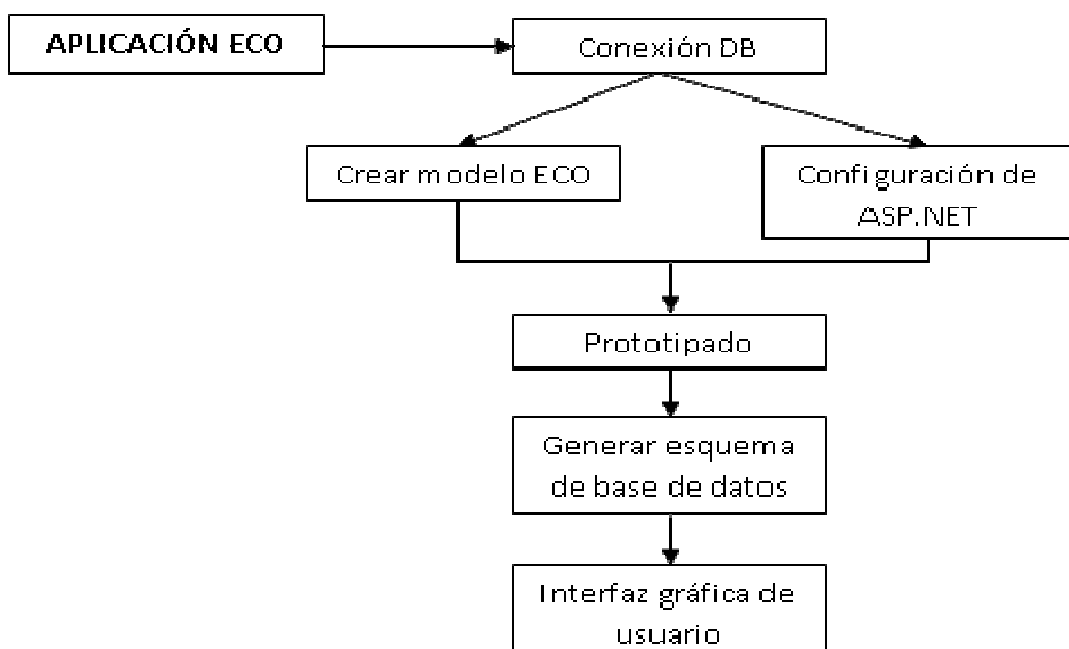
Además de utilizar un subconjunto del lenguaje UML, ECO emplea OCL, el lenguaje de restricciones sobre objetos, un estándar del OMG (Object Management Group) para definir expresiones sobre los modelos UML. Este se utiliza para crear reglas declarativas que calculan o controlan los valores de los atributos de sus objetos.

Aunque UML es un lenguaje, éste posee más características visuales que programáticas, las cuales ayudan a integrantes de un equipo multidisciplinario participar e intercomunicarse fácilmente.

La tecnología ECO de CodeGear da un paso más en esta línea siendo un novedoso marco de trabajo guiado por el diseño para .net y basado en diagramas UML. Esta técnica de desarrollo es conocida como “arquitectura dirigida por modelos”. Además, permite la generación automática de código en virtud del modelo y un esquema de persistencia en virtud del motor de base de datos. ECO no es solo una guía para la codificación, sino que también se está diseñando o inspeccionando la codificación.

En este capítulo, se indicará a un usuario nuevo a que se familiarice con la tecnología ECO, desde su instalación hasta el desarrollo de una aplicación, mediante una guía de implementación que permitirá entender de mejor manera, el manejo de la herramienta de desarrollo RAD Studio 2007 de la empresa desarrolladora CODEGEAR, y el motor de base de datos SQL Server 2005.

El proceso de implementación de una aplicación a través del Framework ECO se lo realiza siguiendo 8 pasos, lo que se muestra a través del siguiente esquema



3.2 REQUISITO DEL SISTEMA

CodeGear es compatible con las siguientes plataformas de Windows:

- Microsoft ® Windows 2000 Professional o Windows 2000 Server (SP4 o superior y actualizaciones de seguridad se recomienda)
- Microsoft Windows XP Professional (SP2)
- Microsoft Windows Vista TM
- Microsoft Windows Server 2003 (SP1)
- Se recomiendan para todas las plataformas el Service Pack más recientes actualizaciones de seguridad
- Microsoft Internet Explorer 6.0 SP1 o posterior

Requisitos: El producto solicita una serie de requisitos previos como el Microsoft. NET Framework 2.0 SDK de Microsoft. NET Framework 2.0 Redistributable Package, Microsoft Visual J # Redistributable Package versión 2.0, Microsoft Data Access Components (MDAC) 2.8, Microsoft XML Core Services (MSXML) 6.0, y el paquete de idioma para Microsoft. NET Framework SDK 2.0. Los requisitos previos se instalan automáticamente en su sistema durante el proceso de instalación si no están presentes.

- Capacidad 3,5 GB de espacio en disco, y el instalador requiere 0,5 GB de espacio temporal adicional.
- Intel Pentium III / M 1.4GHz o Pentium IV 1,4 GHz mínimos (Pentium III / M 1,4 GHz o Pentium IV + + 2 GHz recomendado).
- 512 MB de RAM (1 GB recomendado)
- DVD drive Unidad de DVD
- XGA (1024x768) o monitor de mayor resolución
- Mouse u otro dispositivo señalador.

3.3 CONSTRUCCION DE UNA APLICACIÓN CON ECO

Abrir la aplicación **CodeGear RAD Studio**, elegir la opción **delphi.net**, se muestra la interfaz de la herramienta de desarrollo.

Seleccionar del menú principal **"File|New"**. **"Other"**.

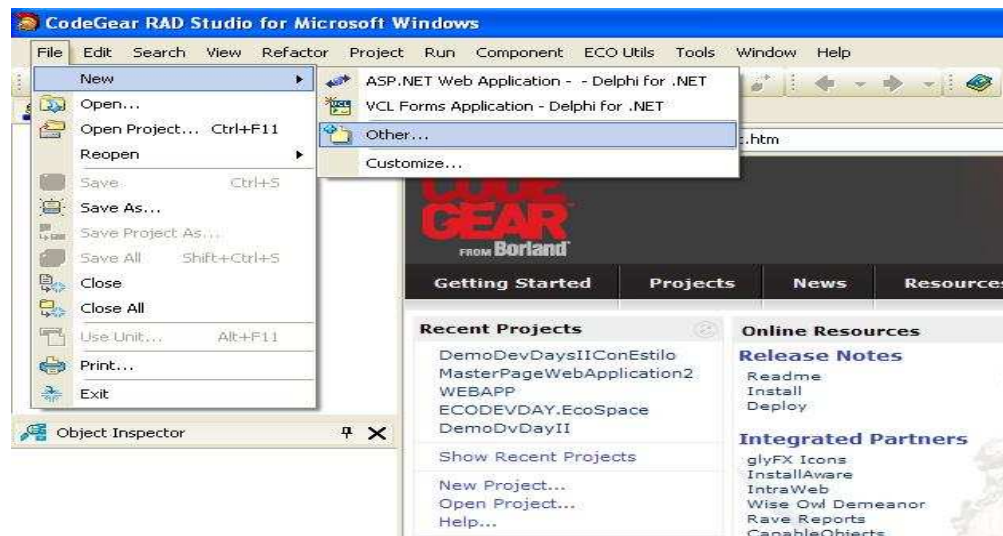


Figura III. 10. Crear una nueva aplicación

En la ventana de dialogo "New Items", dar clic en la opción **"Eco"**, elegir **"Eco Delphi.Net Project Wizard"**, presionar el botón de **"OK"**.

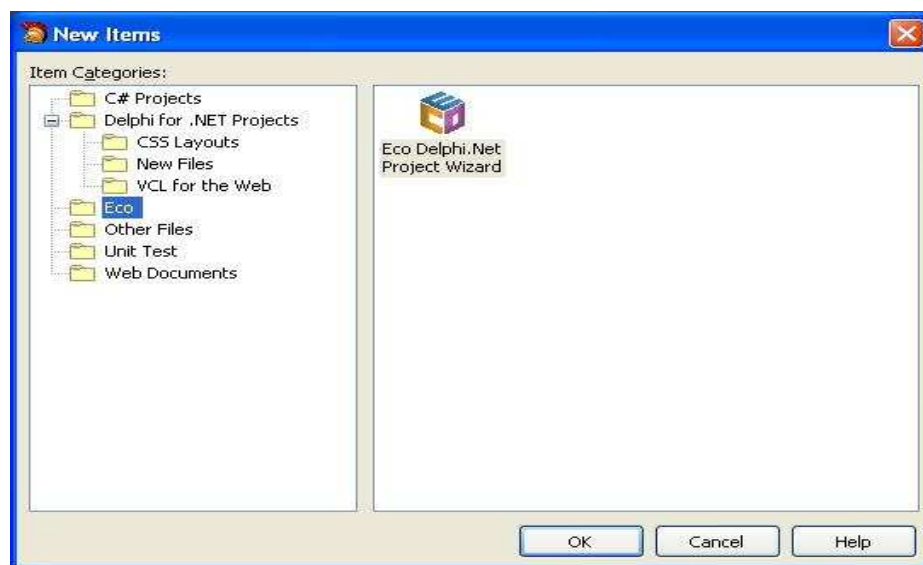


Figura III. 11. Ventana de selección de un nuevo ítem

Se mostrará el dialogo de “**ECO Project Wizard**”.

En el **Presets** del combo box seleccionamos: **Webform Application**. En la opción **name** introducir el nombre de la aplicación. Como ejemplo use el nombre de “**Ecodevday**”.

En **Location** escoger la ruta donde residirá la aplicación “C:\inetpub\wwwroot\Ecodevday”, en el combo box **storage** seleccionar la opción “**SqlConnection**”.

En **User interface** habilitar la opción “**VCL.NET**” y presionar el botón “**OK**”.

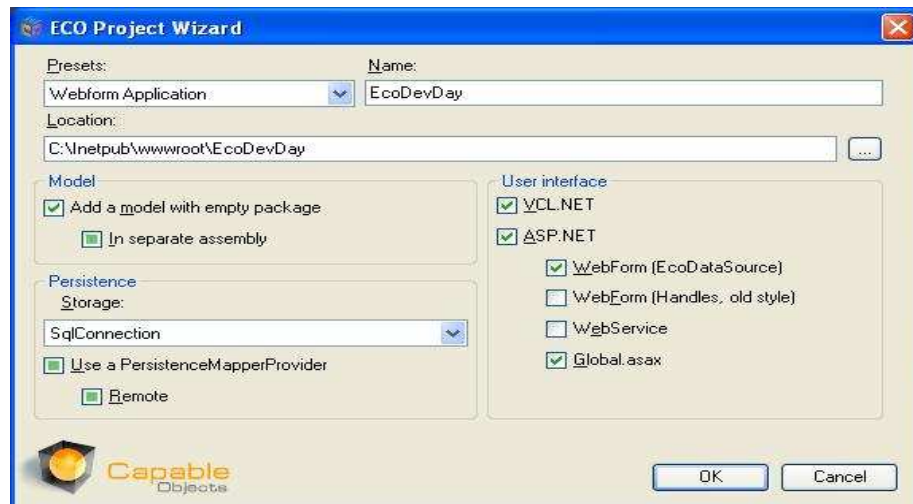


Figura III. 12. Asistente de proyectos ECO

Se compilará automáticamente

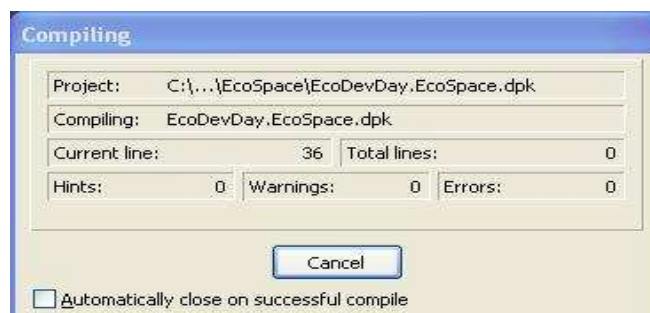


Figura III. 13. Proceso de compilación

Por defecto se guarda en la dirección de la carpeta **EcoSpace** que contiene los objetos en tiempo de ejecución de la aplicación ECO, el EcoSpace contiene las instancias actuales de las clases de su modelo, clic en “Guardar”.

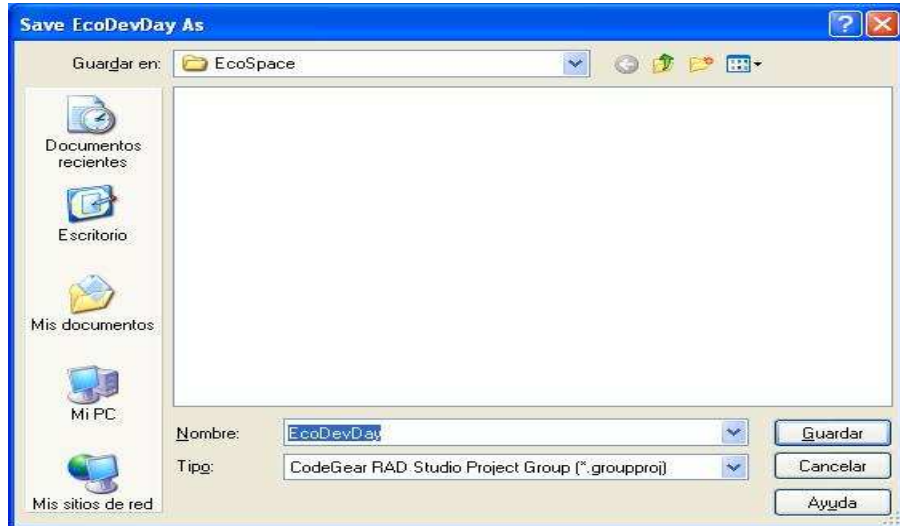


Figura III. 14. Ventana de Guardar aplicación

Automáticamente se mostrará la siguiente ilustración con el **PersistenceMapper** que contiene una referencia a un objeto, que es el encargado de la persistencia del EcoSpace, almacenándolo y recuperándolo de un fichero XML o bien de una base de datos.

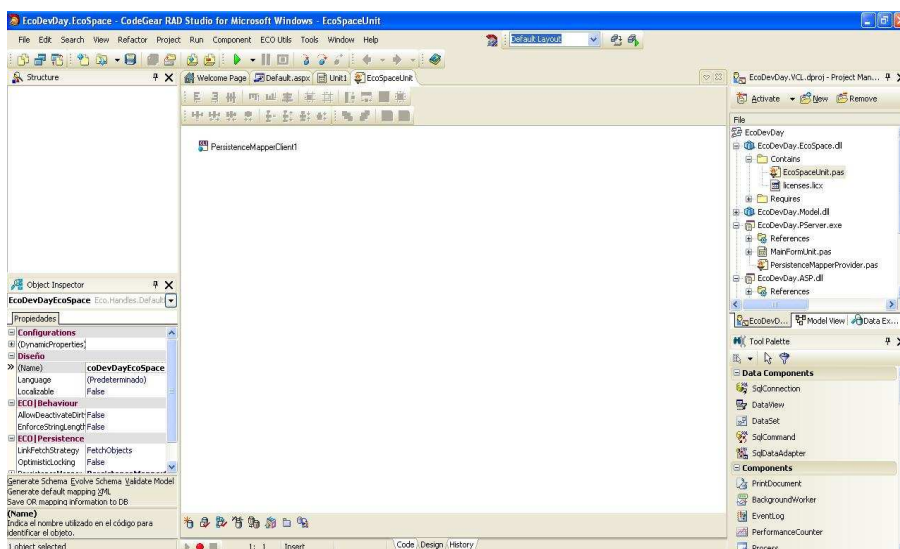


Figura III. 15. Ventana EcoSpaceUnit

3.4 CONEXIÓN A LA BASE DE DATOS

Una vez que se ha creado el nombre de la base de datos, para el ejemplo se desarrollará en SQL Server 2005 con el nombre BD_ECODEVDAY, por lo que se puede volver a RAD Studio y dentro del **ECODEVDAY.PServer.exe**, que se encarga de mantener la persistencia de la base de datos y la conexión de la misma. Dar clic en la unidad **PersistenceMapperProvider.pas**, y a continuación clic en el **sqlConnection1**, que se encuentra el área de diseño.

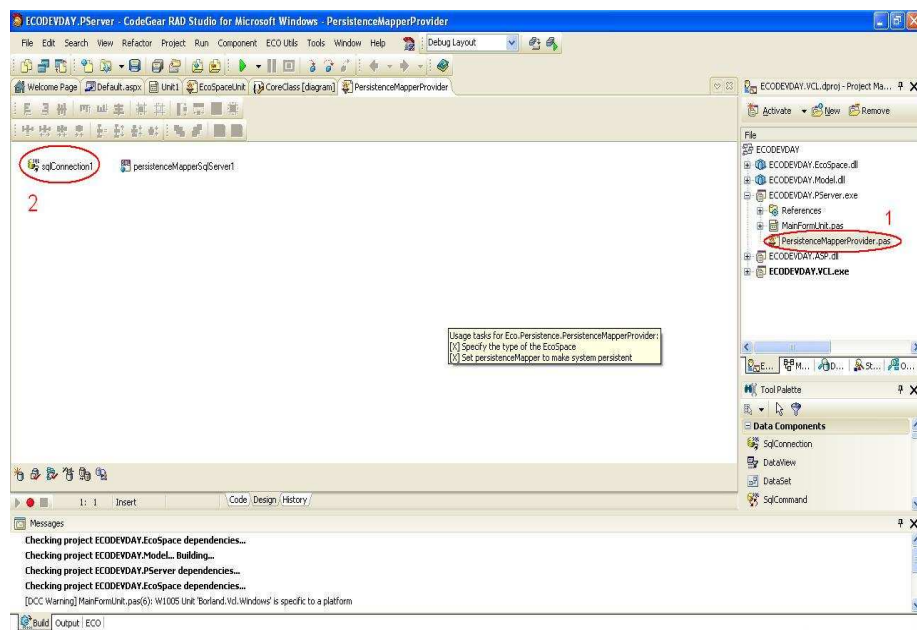


Figura III. 16. Selección de SqlConnections

Una vez marcado el componente **SqlConnection1**. Hacer doble clic en la propiedad **ConnectionString**, se abre la ventana de Propiedades de vínculo de datos. Aquí, seleccionar el nombre del servidor, especifique el nombre de usuario y contraseña, y seleccionar la base de datos en el servidor. Asegúrese de hacer clic en la Prueba de conexión, para comprobar que puede establecer una conexión con la base de datos, Aceptar para continuar con la aplicación.

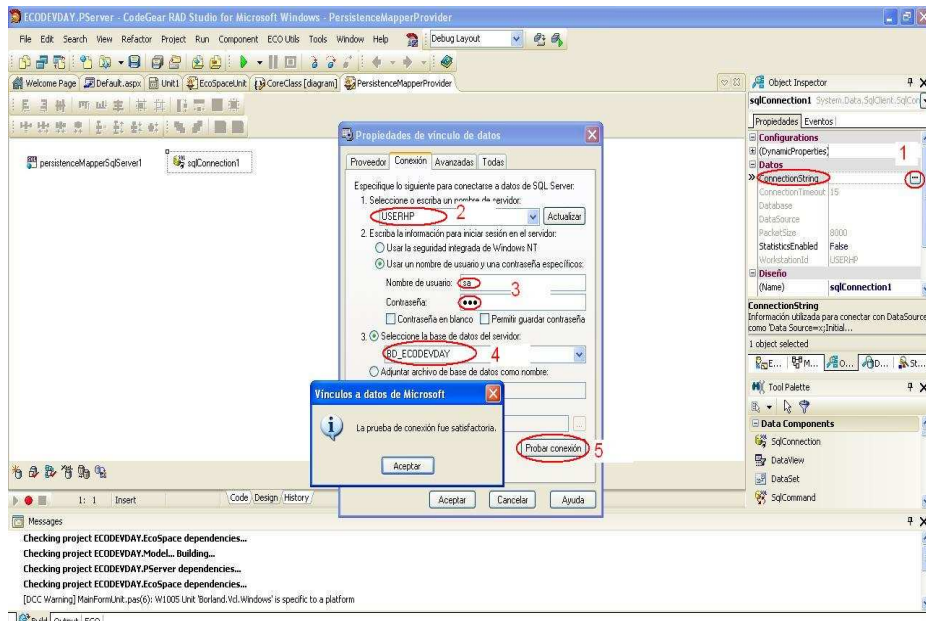


Figura III. 17. Conexión de la Base de Datos

En el Visor de Modelo (**Model View**), hacer doble clic en el paquete de diagrama llamado "**Package_1**". Clic derecho en el nodo de árbol titulado "Package_1" y selecciona "cambiar nombre". Cambie el nombre del paquete a "**CoreClasses**". Esto abrirá al editor del área de modelado. Cuando se abre, la "paleta de Herramientas" cambia para mostrar las herramientas necesarias para crear el modelo.

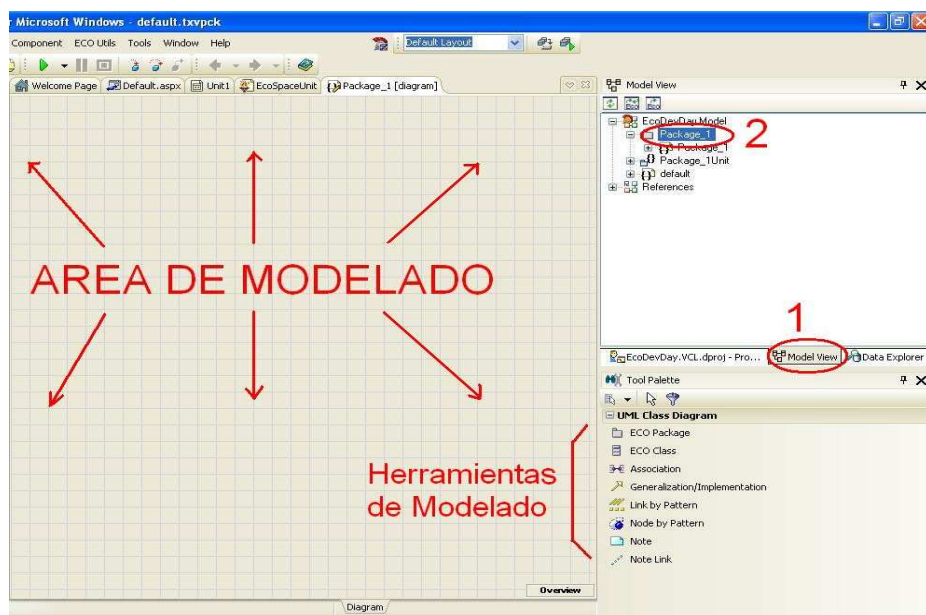


Figura III. 18. Área de Trabajo

3.5 CREAR EL MODELO EN ECO

ECO usa un subconjunto del Lenguaje de Modelado Unificado (UML), como su idioma de modelado. Las partes específicas son el Diagrama de Clase y el Lenguaje de Control de Objetos (Object Constraint Language - OCL).

Iniciar creando una clase. Puede hacerlo de una de la siguiente forma: Seleccione de las herramientas de modelado, el “**ECO Class**” y de clic en el área de modelado.

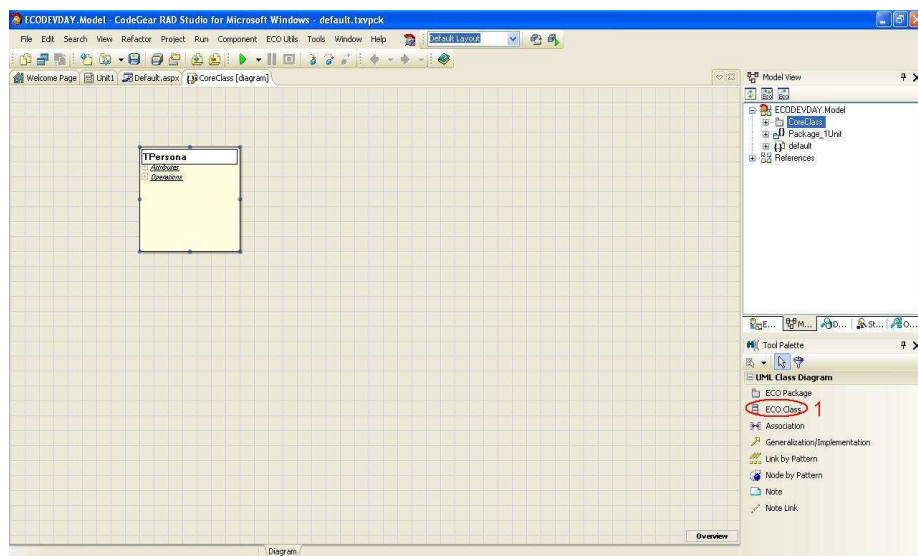


Figura III. 19. Crear un modelo UML en ECO

Una vez creada la clase. Escribir el nombre de la clase “**TPersona**”. Posteriormente clic derecho sobre el área de modelado y selecciona “**Add|Attribute**” del menú desplegable.

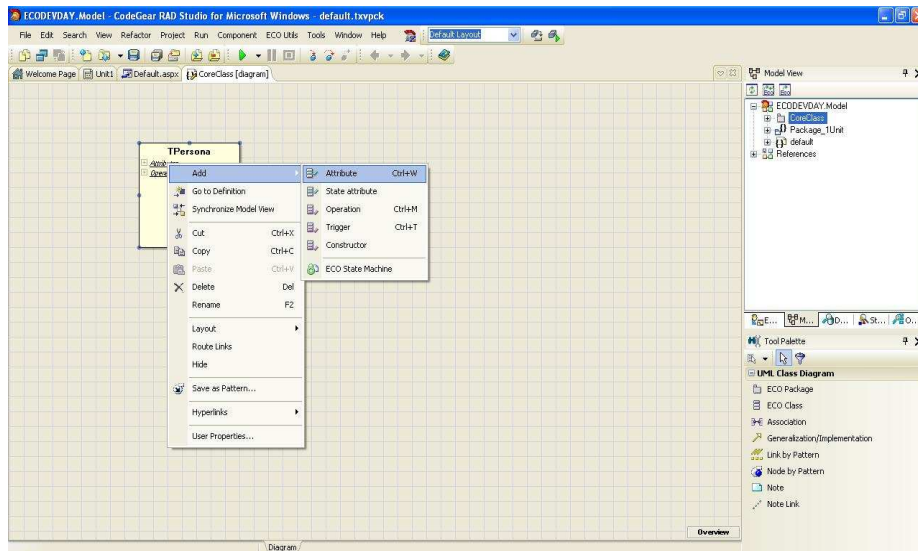


Figura III. 20. Añadir atributos a diagrama de clases

Puede agregar atributos y operaciones para una clase de ECO. Evidentes atributos de una entrada de **TPersona** es el nombre, el apellido la fecha. Por lo tanto, hacer clic derecho en la clase de entrada, y agregar un nuevo atributo llamado **Nombre** de tipo String. Haga clic de nuevo y añadir un segundo atributo llamado **Apellido** de tipo String, un tercer atributo llamado **FechaAlta** de tipo DateTime y finalmente el atributo **/NombreCompleto** de tipo String con un / al inicio para determinar que es un atributo derivado.

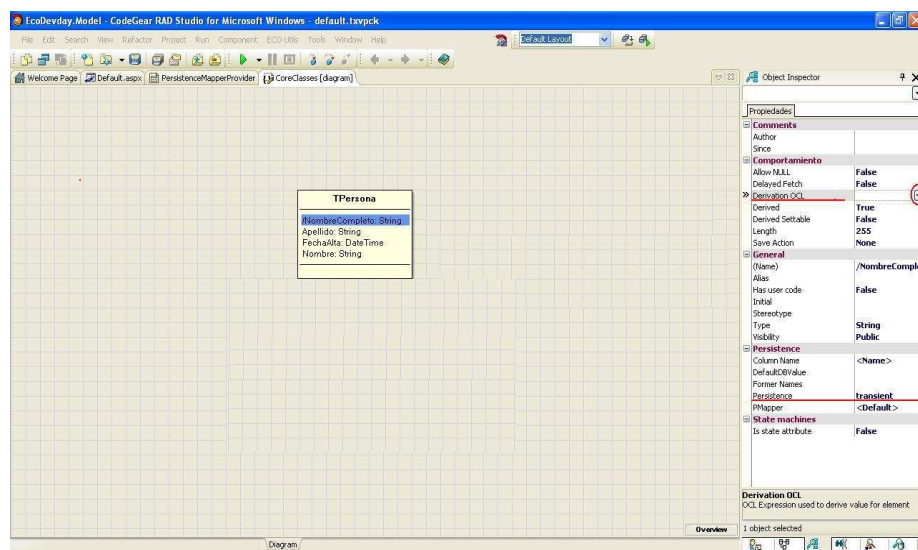


Figura III. 21. Propiedades del Diagrama UML

El atributo **/NombreCompleto** en las propiedades **persistence** está **transient** por que no se va a guardar en la base de datos y es un atributo derivado que se va a calcular de la siguiente manera haciendo clic en la propiedad **derivación OCL** y muestra una ventana de un editor OCL como se observa en la siguiente figura.

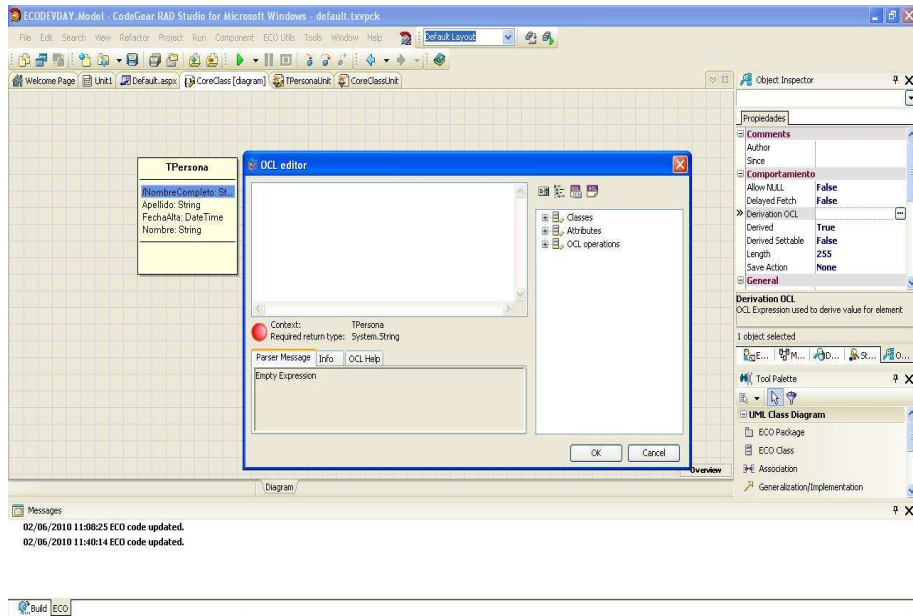


Figura III. 22. Restricciones OCL

Para calcular el nombre completo se debe utilizar la siguiente sintaxis **self.Nombre + ' ' + self.Apellido** en el editor OCL, y clic en OK.

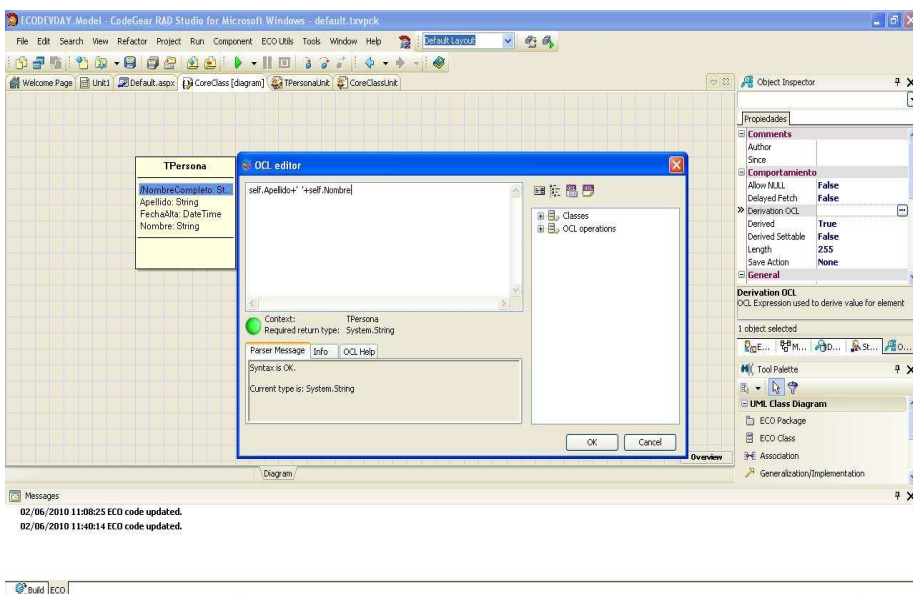


Figura III. 23. Expresión OCL para el Objeto persona

AÑADIR GENERALIZACIÓN

La generalización hereda los métodos y atributos especificados por una clase base, por lo cual una clase derivada además de tener sus propios métodos y atributos, podrá acceder a las características y atributos visibles de su clase base, la generalización se conecta entre dos clases. Para añadir una generalización se debe crear otra clase denominada **TUsuario**, con el atributo **Login** de tipo String. Esto se hace con el icono de la paleta de herramientas que dice "**Generalización / Aplicación**". Haga clic en este icono, luego en la clase TUsuario, y finalmente en la clase TPersona. Tenga en cuenta que el orden importa.

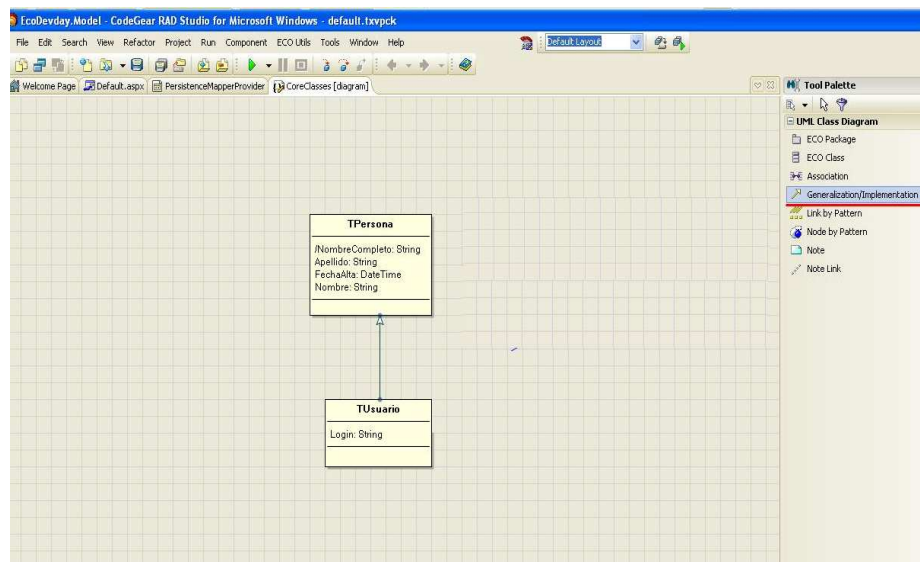


Figura III. 24. Agregar Generalización

AÑADIR ASOCIACIÓN ENTRE CLASES

La asociación, permite relacionar objetos que colaboran entre sí. Cada extremo de la asociación tiene un valor de multiplicidad, que indica cuántos objetos de la asociación están relacionados con un objeto del extremo contrario, para el ejemplo se asociará la clase **TUsuario** con una nueva que se creará denominada **TPermiso** que tiene un atributo **Descripción** de tipo String, seleccione el objeto asociación y relacione la clase TUsuario y la clase TPermiso.

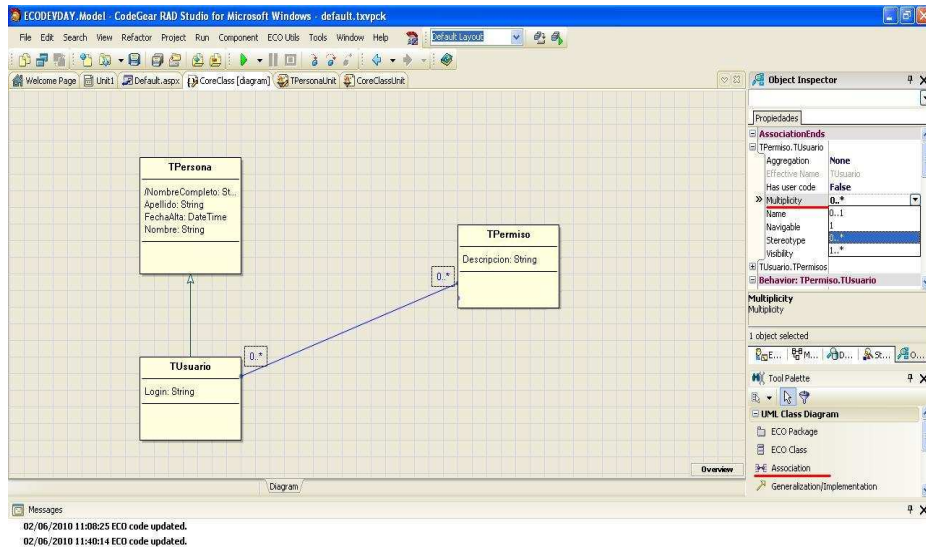


Figura III. 25. Multiplicidad de los diagramas de clase

En el inspector de objetos dar clic en la pestaña **AssociationEnds**, seleccione **TUsuario.Permiso** en la propiedad **multiplicity** escogemos 0..*, de cero a muchos, en **name** dé el nombre de **Usuarios**. De igual manera para **TUsuario.Permisos** la propiedad **multiplicity** 0..*, **name** Permisos.

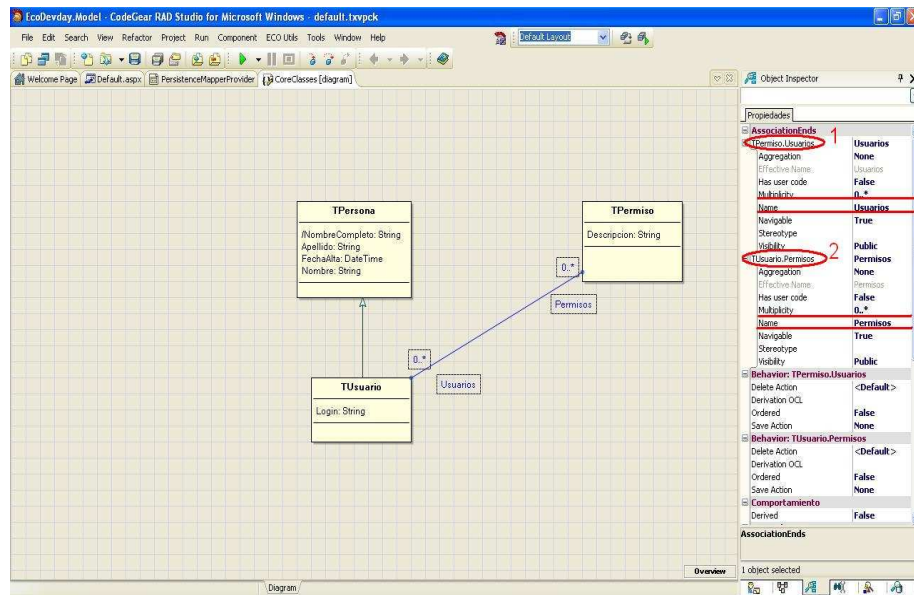


Figura III. 26. Propiedades de la asociación

Finalmente su diagrama de clase debe quedar algo así, como se ilustra en la figura III.27.

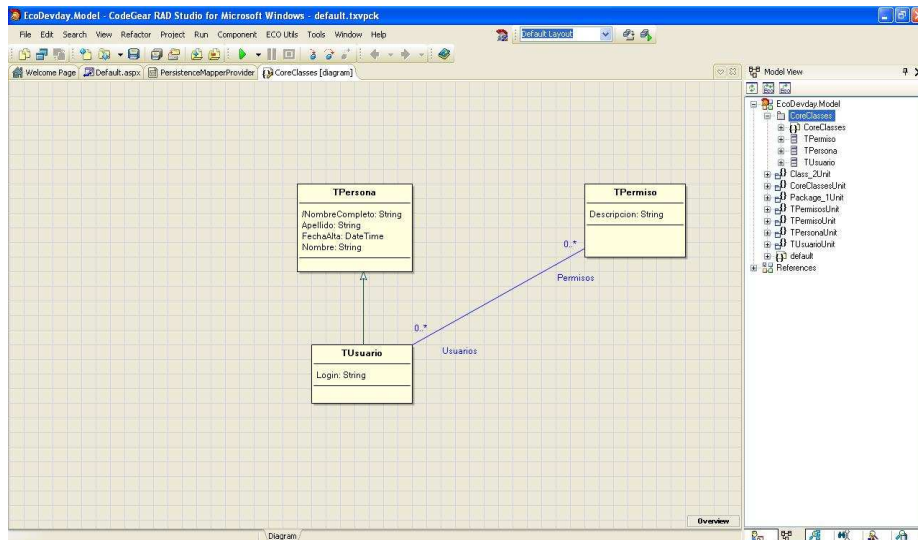


Figura III. 27. Representación de los diagramas de clase

3.6 CONFIGURACIÓN DE ASP.NET

Volver a la pestaña **ECODEVDAY.ASP.dll** del proyecto en el cual está trabajando.

Elija la opción de menú **Project|Options**.

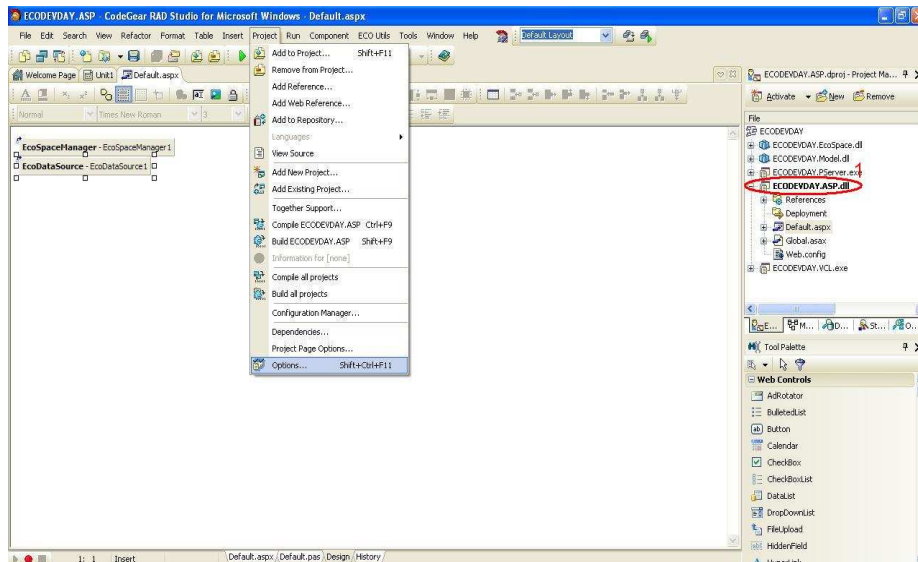


Figura III. 28. Configuración de ASP.NET

En el formulario que aparece elegir el depurador / nodo de **ASP.NET** en la vista en árbol de la izquierda, en **Start page** debe estar default.aspx y seleccionar **Server Cassini Web Server**.

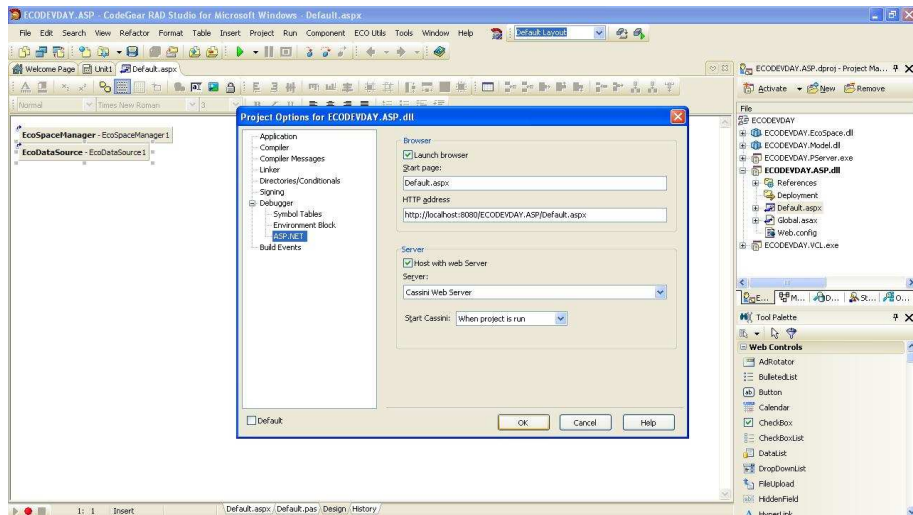


Figura III. 29. Opciones de ASP.NET

3.7 HERRAMIENTA DE PROTOTIPOS

Para iniciar el prototipado en la opción menú seleccionar la herramienta ECO Utils|Quick prototype.

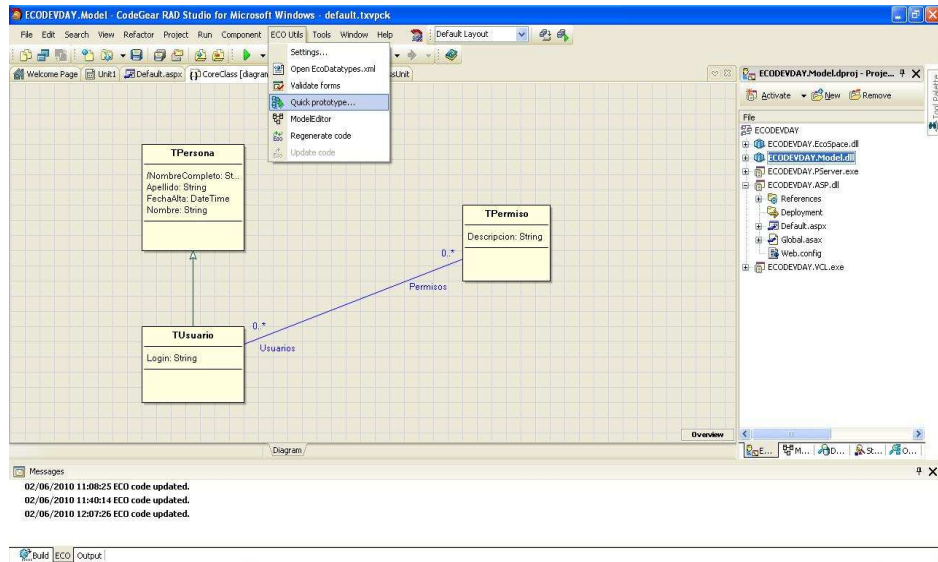


Figura III. 30. Opciones de prototipo

En la siguiente ventana marcar "XML persistence" en la casilla de verificación introducir un nombre del archivo más ruta completa en la que los objetos se mantuvo, haga clic en "Start" para continuar.

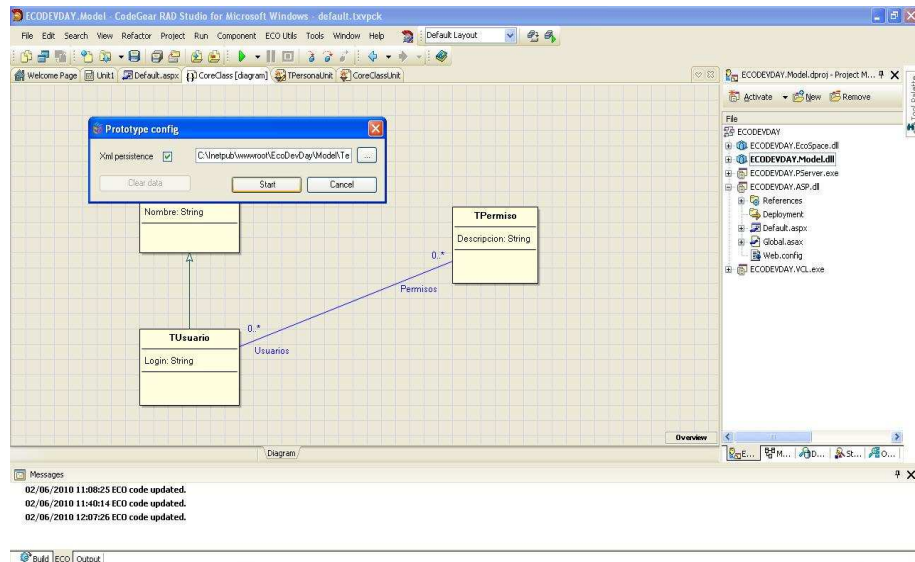


Figura III. 31. Iniciar XML Persistence

Seguidamente aparecerá el formulario del ECO "Quick prototype". El formulario consta de cuatro pestañas principales:

Classes and OCL: Permite ver todas las instancias de una clase específica, es posible evaluar expresiones OCL de su modelo.

Dirty Objects: Muestra una lista de todos los objetos que han sido modificados de alguna manera que requeriría una base de datos actualización.

Undo/Redo: Realiza modificaciones dentro de una transacción para que estos cambios se puede revertir o cometidos como una sola operación.

Log ECO: Especifica las acciones para iniciar la sesión, tales como evaluaciones de OCL, las solicitudes de DB.

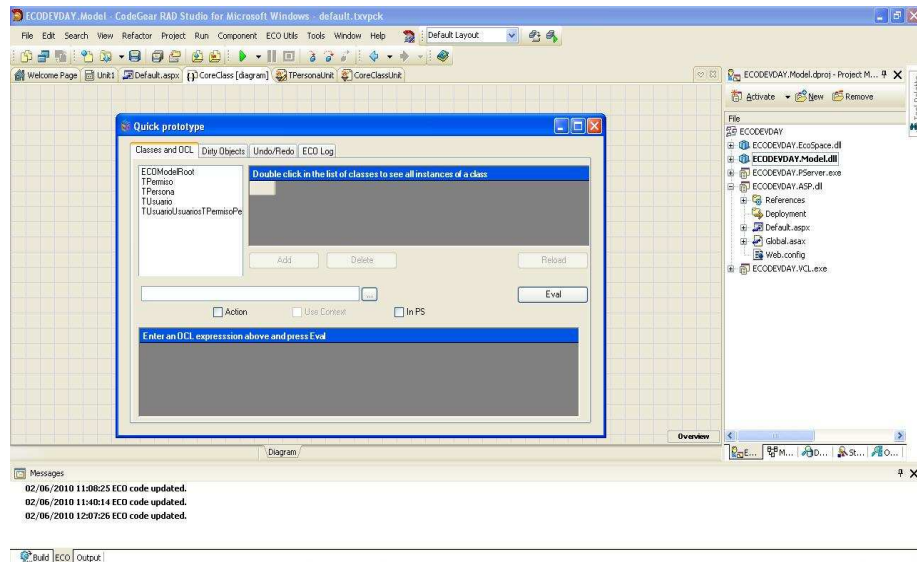


Figura III. 32. Clases y OCL

En la pestaña **Classes and OCL** se muestra todas las clases creadas a partir del modelado UML, a continuación haga doble clic en la clase de **TUsuario** se mostrará los atributos de la clase TUsuario, luego haga clic en el botón "**Add**" de la clase TUsuario que aparece en el formulario "**Quick prototype**". E introduzca la información en cada una de las columnas, Apellido, Nombre y Login el NombreCompleto se genera automáticamente ya que utiliza restricciones OCL.

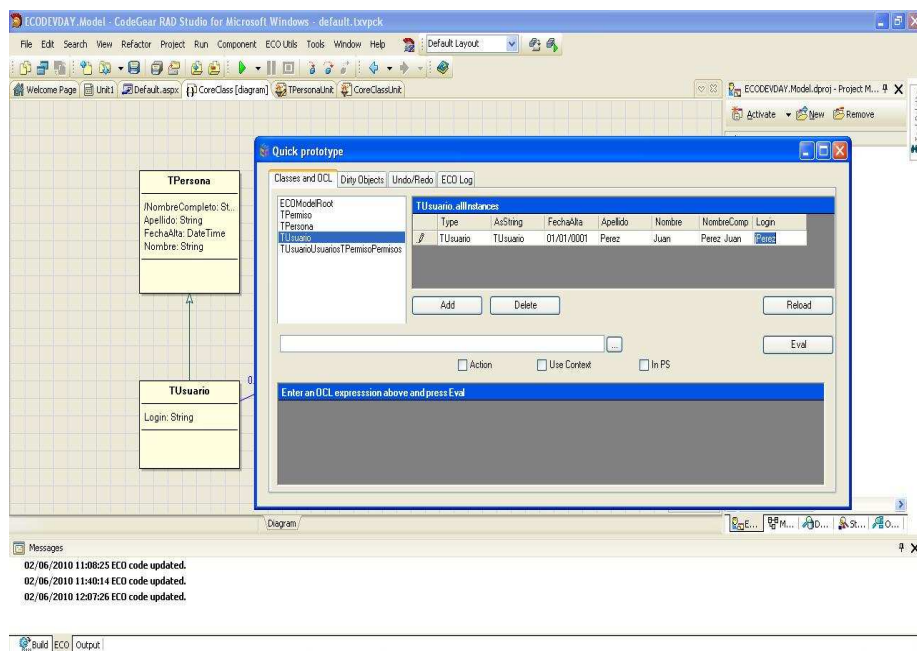


Figura III. 33. Añadir un Usuario

Doble clic en la clase de **TPermiso** en el botón **Add** para añadir una Descripción al permiso para el ejemplo ponga en la descripción Permiso1.

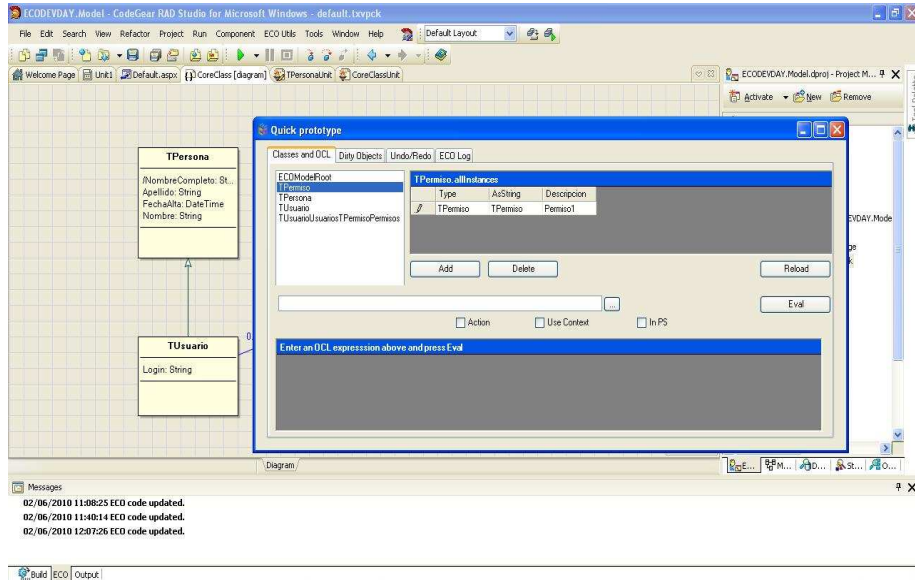


Figura III. 34. Añadir Permiso

Haga doble clic en la clase TUsuario para mostrar todas las instancias del usuario. Crear uno o más usuarios, hacer doble clic en la columna de la izquierda y aparece una instancia del objeto, una forma de generar automáticamente en función de la información del modelo para la clase **TUsuario** será exhibido.

El depurador también está equipado con Autoforms ECO para que pueda abrir el Autoform haga doble clic en la fila. El formulario mostrará las propiedades del objeto, el estado actual se define con los disparadores para la transición a otro estado

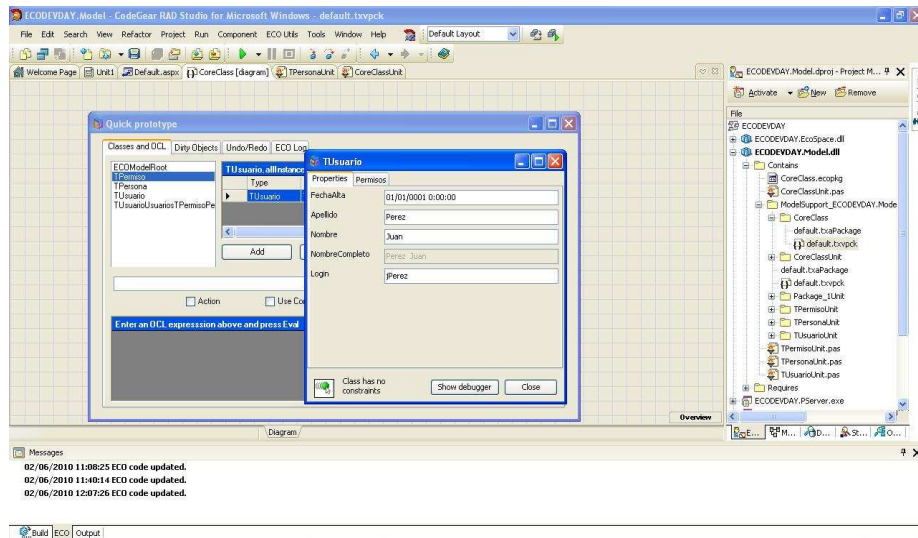


Figura III. 35. Propiedades del Usuario

Antes de iniciar a trabajar con la interfaz de usuario dar clic en el **botón 1** como se muestra en la siguiente ilustración para ejecutar acciones OCL

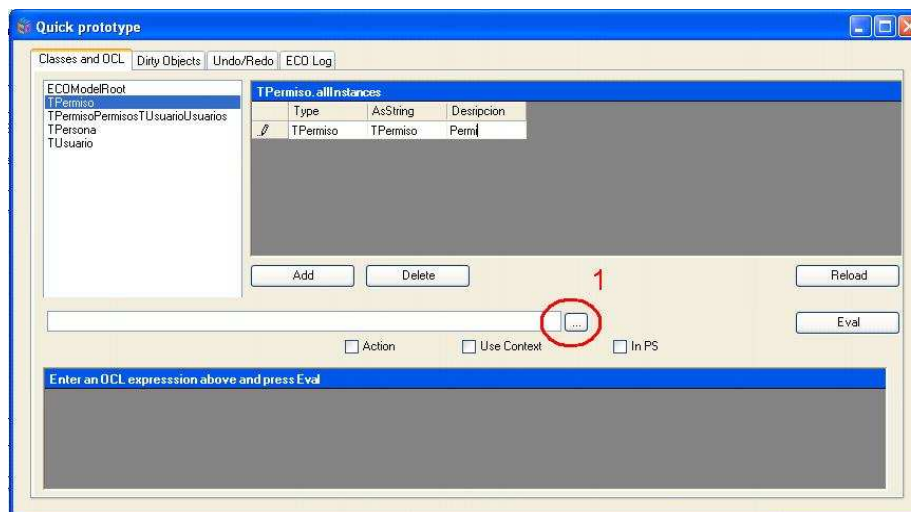


Figura III. 36. Acciones OCL

En el ejemplo pedir todas las instancias de usuario con la siguiente línea de código en el editor OCL **TUusuario.allInstances** y se muestra con un botón verde en Context donde indica que la sentencia es correcta.

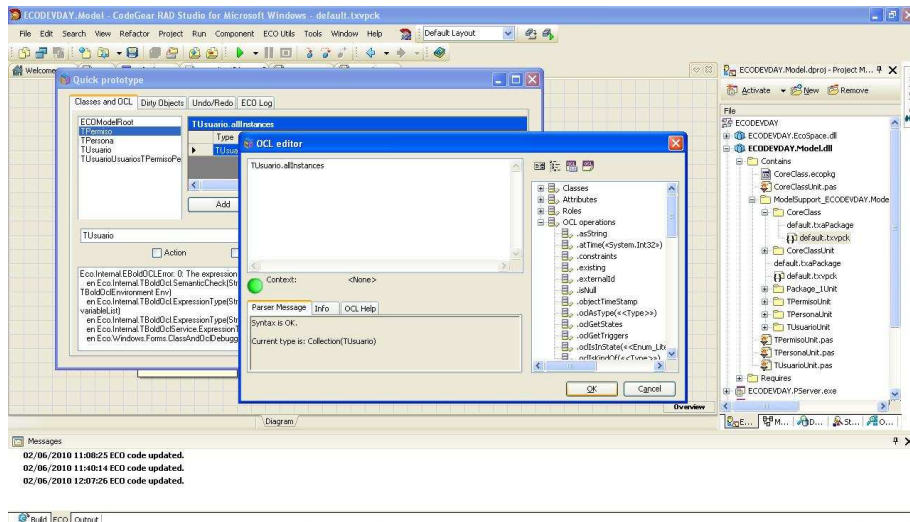


Figura III. 37. Editor OCL del objeto Usuario

A continuación dar clic en Eval y se muestra todas las instancias de la clase usuario.

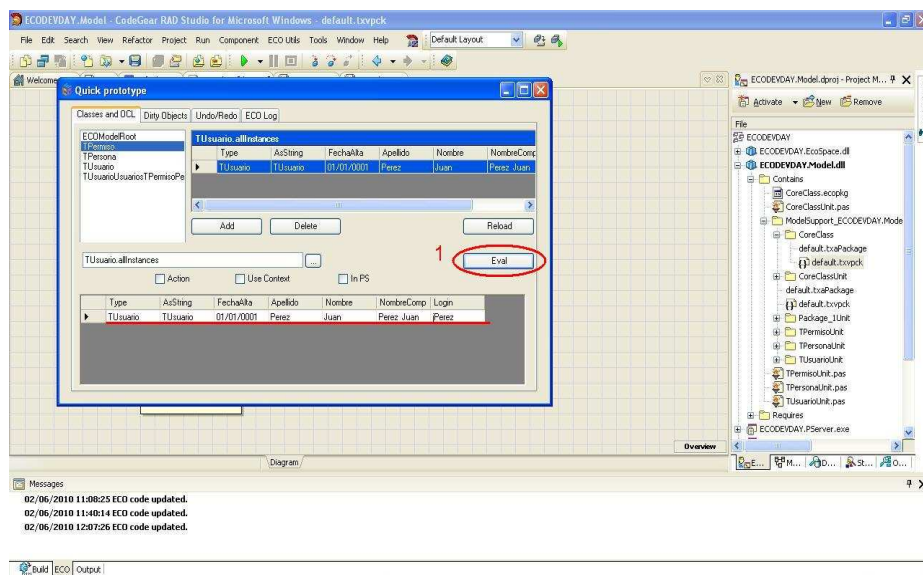


Figura III. 38. Evaluación del Objeto Usuario

Regresar al **ECODEVDAY.EcoSpace.dll**, hacer clic en **EcoSpaceUnit.pas** para determinar los paquetes con los que se va a trabajar, finalmente, clic en **select Packages**, esta herramienta viene integrado las pestañas de diseño y código e historial para facilitar el manejo de la aplicación.

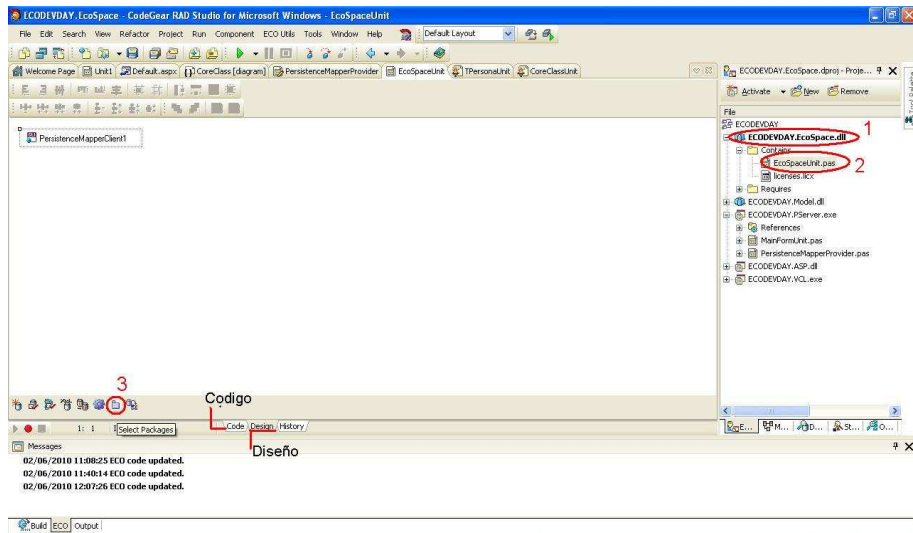


Figura III. 39. Selección de paquetes UML

Seguidamente se muestra el **Select UML Packages** para el ECO Space con dos ventanas donde se encuentra los paquetes UML disponibles, haga clic en Select all packages.

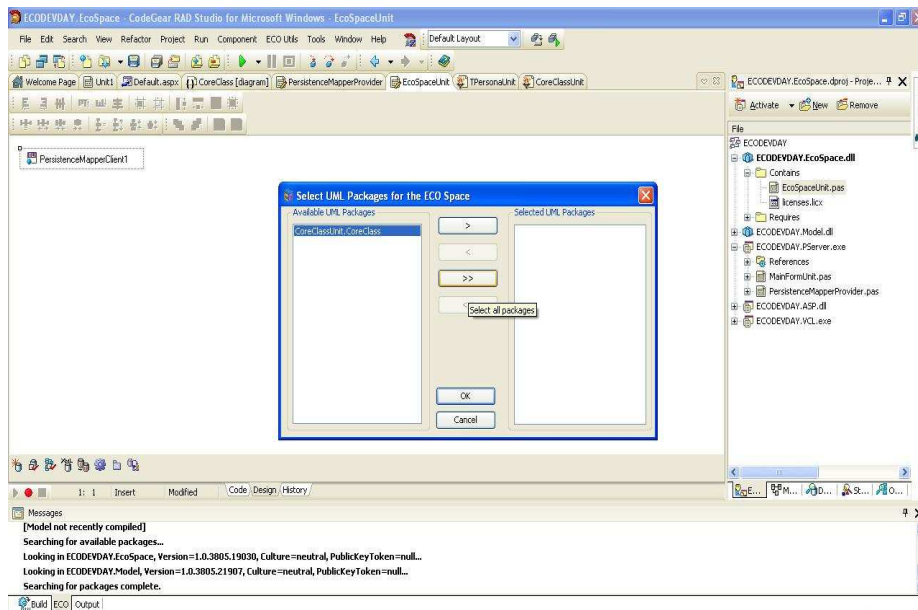


Figura III. 40. Paquete CoreClassesUnit.CoreClasses

Se encuentra **CoreClassesUnit.CoreClasses** dentro de los paquetes seleccionados UML y hacer clic en OK.

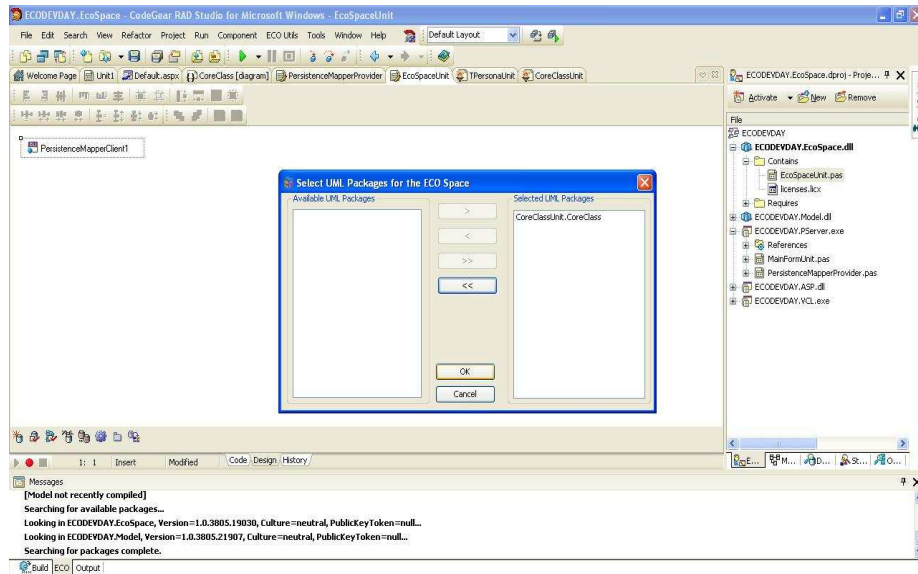


Figura III. 41. Paquetes seleccionados de UML

Compile el modelo para asegurarse de que está sin errores dando clic derecho sobre el **ECODEVDAY.EcoSpace.dll** en **Build** y dar clic en **OK** ya que no existen errores.

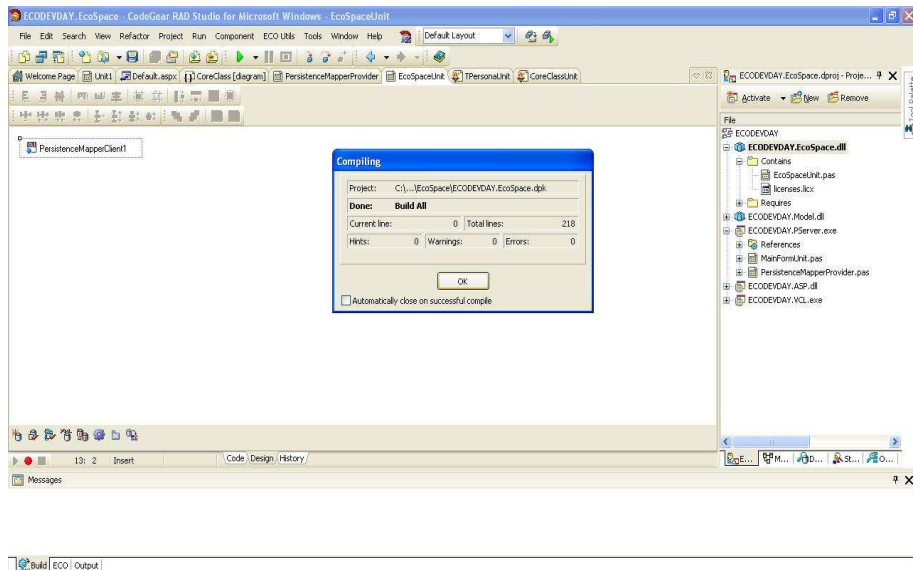


Figura III. 42. Compilación del modelo

3.8 GENERAR EL ESQUEMA DE BASE DE DATOS

El siguiente paso es generar el esquema en la base de datos. Una vez que el componente SqlConnection está configurado, que se lo realiza en la conexión a la

base de datos, Para ello, en la esquina inferior izquierda del área de diseño de la **PersistenceMapperProvider**, haga clic en **"Generate Schema"** botón de la barra de herramientas en la base del diseñador EcoSpace.

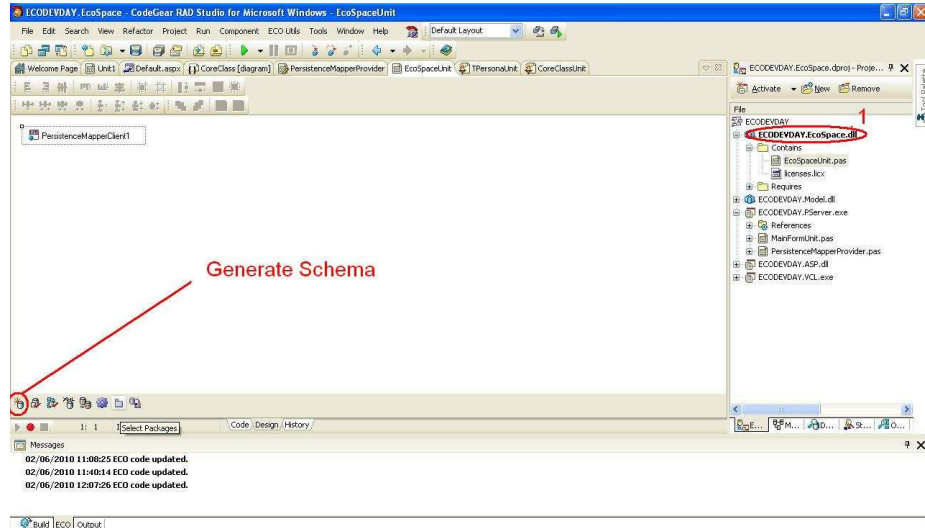


Figura III. 43. Proceso de generación del esquema de BD

Ahora el modelo ha sido generado. Esto creará las tablas necesarias para soportar las clases y sus relaciones establecidas en el modelo. Haga clic en OK para crear los cuadros de base de datos, una vez completado el diseño del modelo garantizará que se hará persistente en la base de datos SQL Server.

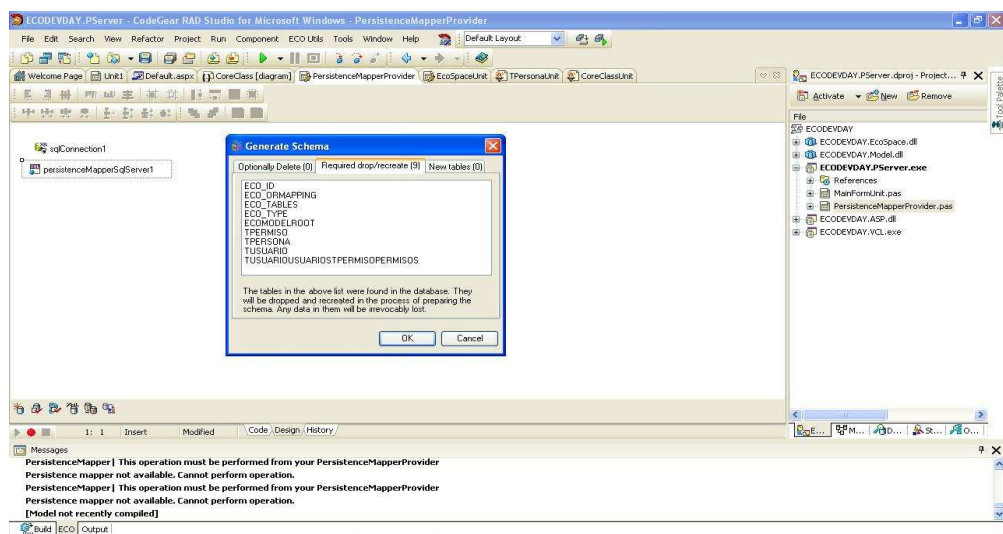


Figura III. 44. Generación del esquema de BD

Se crea una interfaz de usuario base para que se pueda introducir algunos datos. Para ello abra SQL Server 2005 de clic en la Base de Datos **BD_ECODEVDAY**, en la pestaña tablas puede observar las tablas con los diversos nombres que se crearon gracias al modelo generado por el framework ECO.

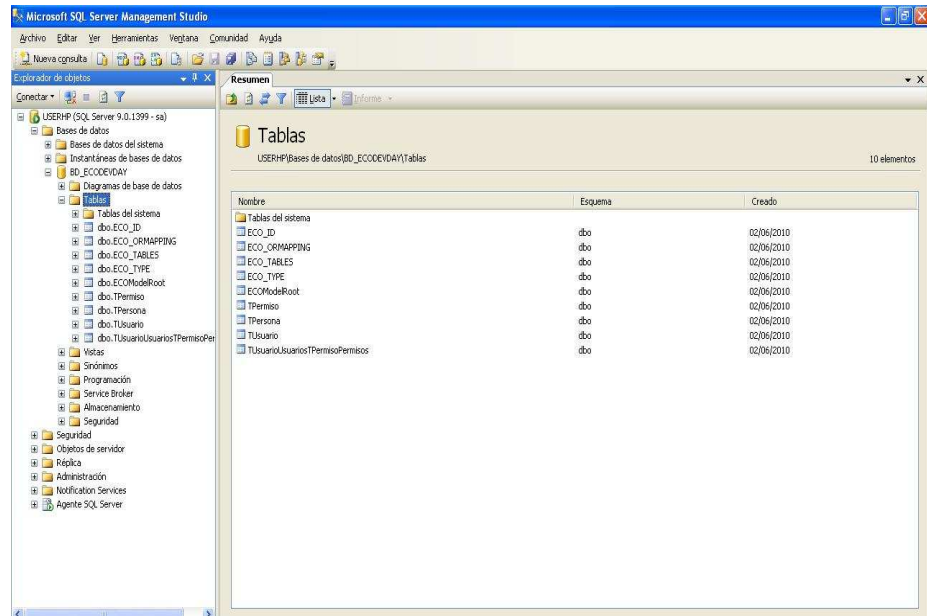


Figura III. 45. Representación de BD en SQL Server 2005

En un analizador de consultas digite ***select * from TPersona*** y ejecute la consulta, como resultado obtiene los campos de la tabla que al momento se encuentran sin datos y el campo nombre completo, no se muestra porque es un atributo no persistente.

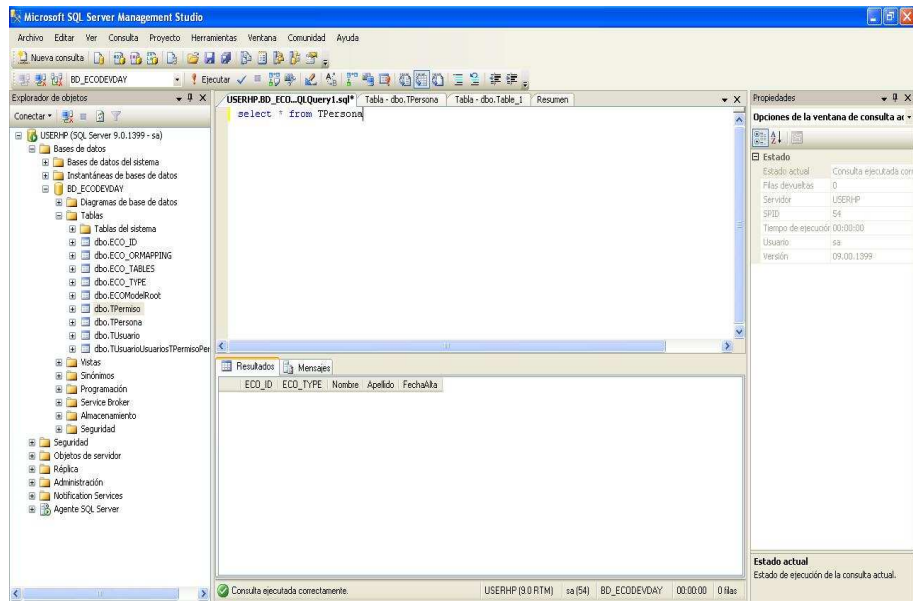


Figura III. 46. Sentencias SQL

3.9 INTERFAZ GRÁFICA DE USUARIO

En la aplicación se dirige a **ECODEVDAY.ASP.DLL**, hacer clic en `Default.aspx`. Existe un **EcoSpaceManager**, Este componente hace que una instancia `EcoSpace` esté disponible durante la prestación de un formulario web, al finalizar el proceso de prestación de este componente asegurará de que la acción adoptada es correcta para liberar la instancia `EcoSpace`. Y un **EcoDataSource** que se encarga de controlar la interacción entre los objetos `EcoSpace` y los controles web de la interfaz de usuario.

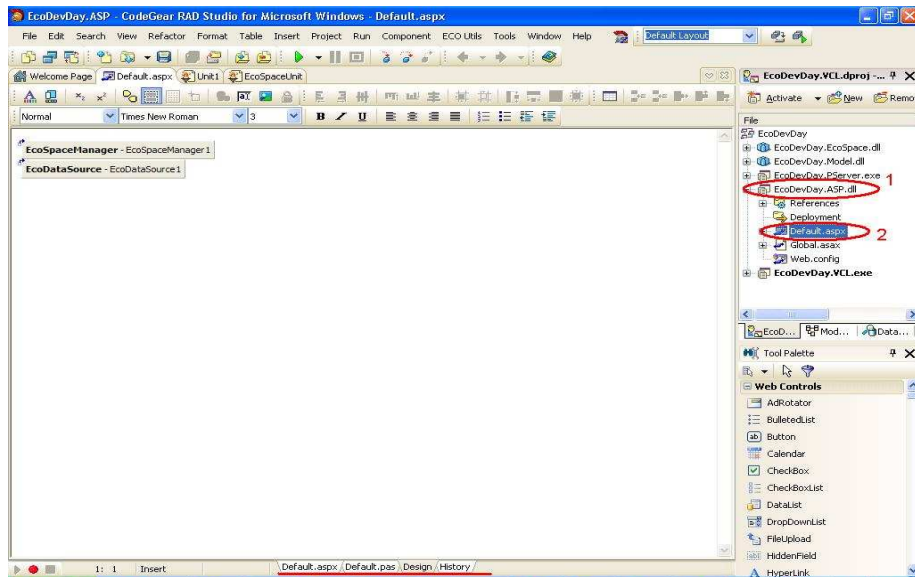


Figura III. 47. Diseño de Interfaz de Usuario

Añadir algunos controles visuales que se encuentra en la paleta de Herramientas, en la categoría de los controles Web, doble clic en el control **DataGrid** y arrastarlos al formulario web del ASP.NET en el flujo de diseño.

Habilitar todas las opciones del GridView Tasks.

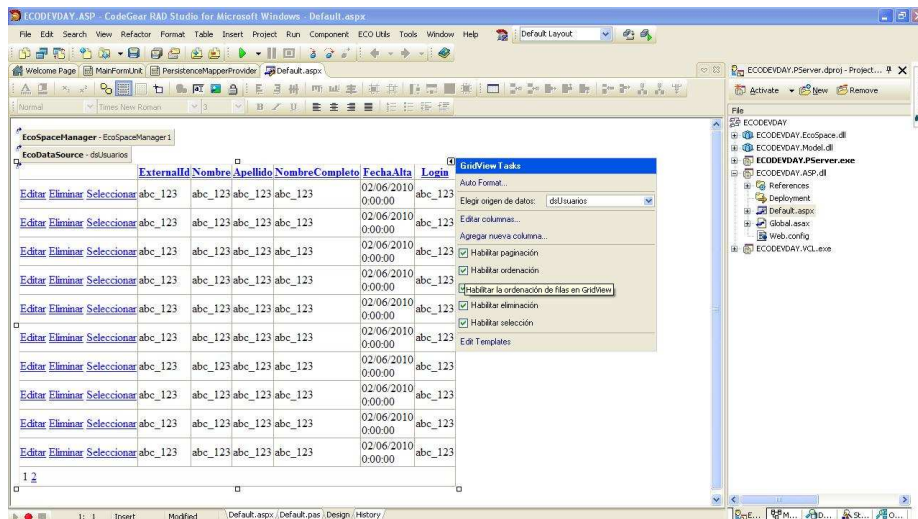


Figura III. 48. Añadir control DataGrid

Se muestra inmediatamente las columnas en la cuadrícula de datos: para el **ExternalId**, los campos del atributo Usuario como el **Nombre**, **Apellido**, **NombreCompleto**, **FechaAlta** y **Login**.

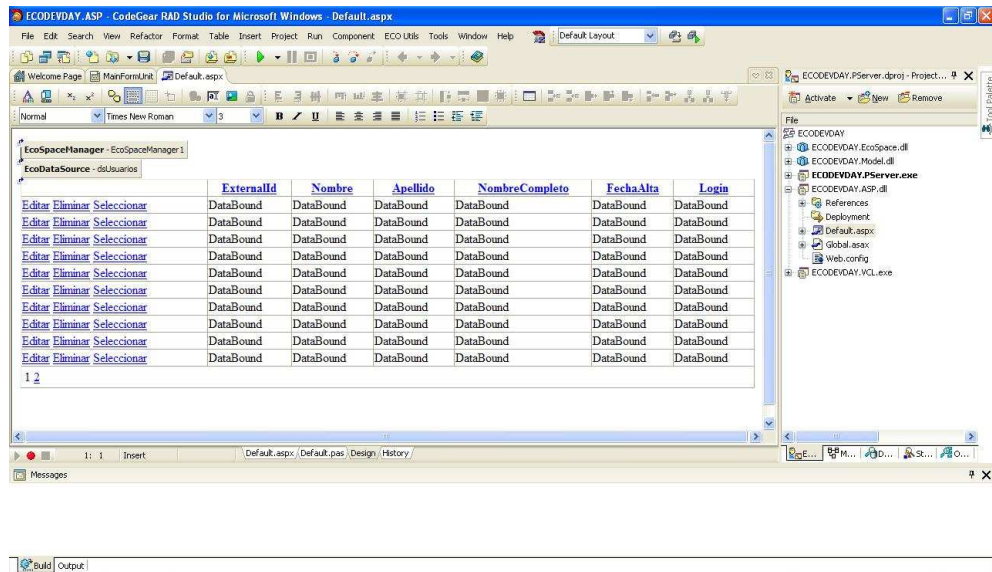


Figura III. 49. Opciones GridView Tasks

En el diseño de esta página colocar un texto, titulado **Aplicación ASP.NET con ECO**, seleccionar el tipo de letra con su tamaño, un encabezado, negrita, cursiva, justificar, es decir dar un estilo al texto.

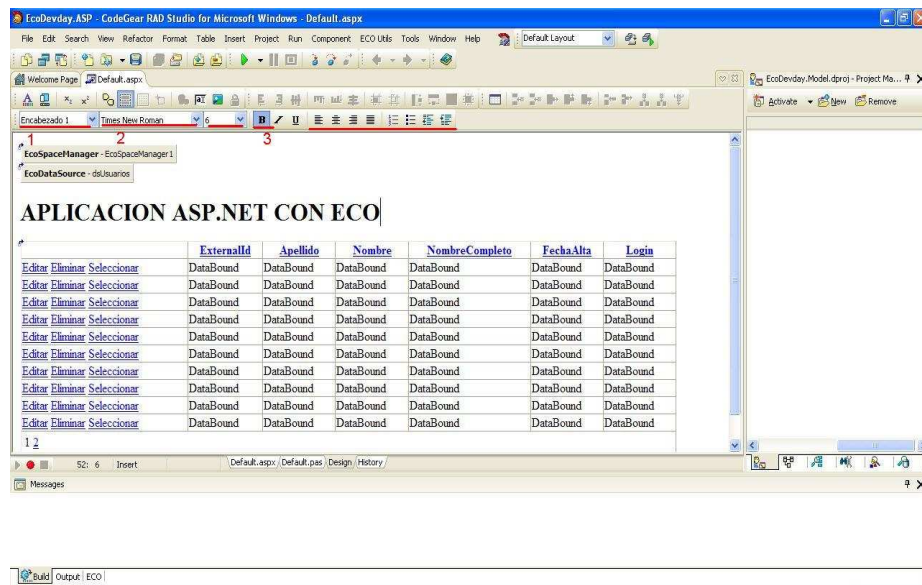


Figura III. 50. Insertar Áreas de texto

Ejecutar la aplicación dando clic derecho en Default.aspx, a continuación View in Browser. Se muestra en el navegador solo el texto que acabó de colocarlo, porque aún no existen datos en la base de datos.



Figura III. 51. Vista en el Explorador

El **AutoForm** permite hacer pruebas rápidas de sus modelos en un entorno web, la creación, actualización, la eliminación de instancias, y vincular / desvincular los objetos a través de sus asociaciones en un navegador. Para ello dirigirse a **File|New|Other**, elegir **New ASP AutoForm** y clic en OK.

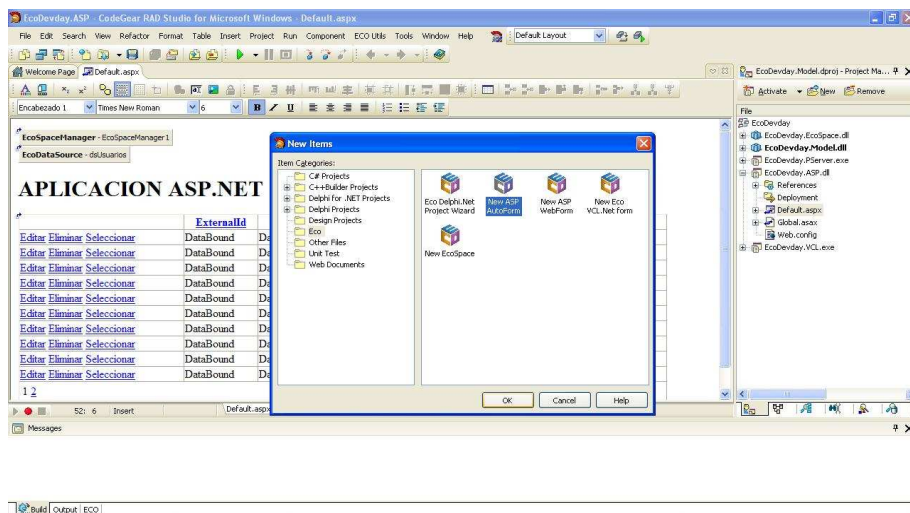


Figura III. 52. Añadir un AutoForm

Acceder a todos los elementos del modelo, antes de tener las páginas de acceso creadas, cambiar el nombre por uAutoForm.

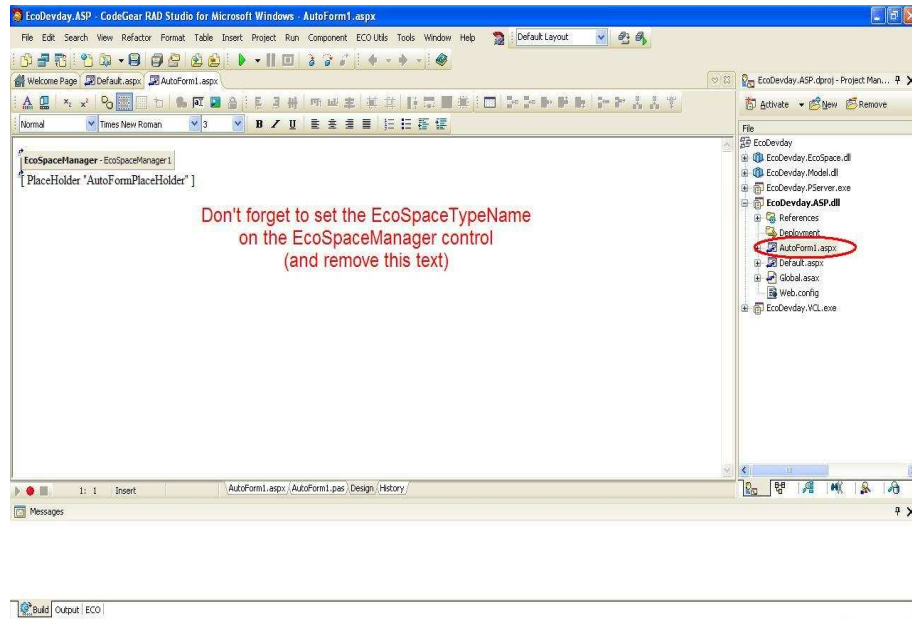


Figura III. 53. Renombrar el AutoForm

En la paleta de propiedades en la opción **EcoSpaceTypeName** poner el nombre **EcoSpaceUnit.EcoDecDay.Eco**. Quitar el texto que se encuentra en rojo.

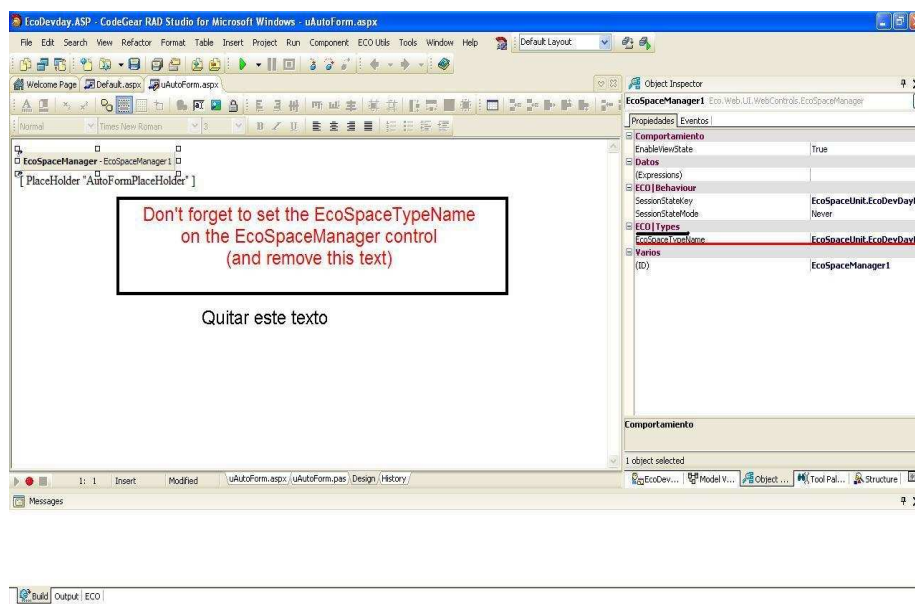


Figura III. 54. Propiedades del AutoForm

Se puede observar el diseño como se muestra en la ilustración, colocar un texto titulado **AutorForm** y dar un estilo.

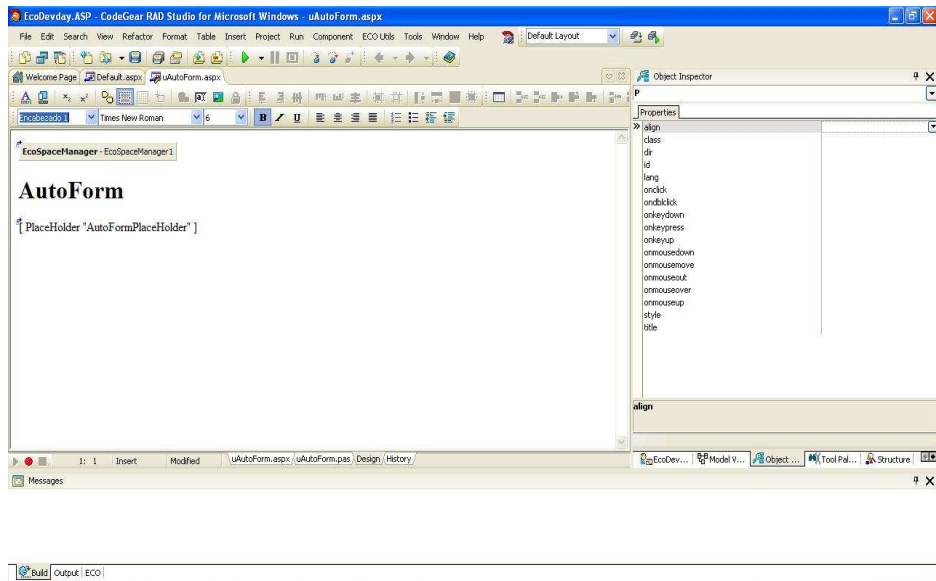


Figura III. 55. Representación AutoForm

Control **Menú**, crear un **ASP.NET User Control**, clic derecho en **ECODEVDAY.ASP.DLL** Add New | Other, seleccionar **ASP.NET User Control**, dars clic en OK.

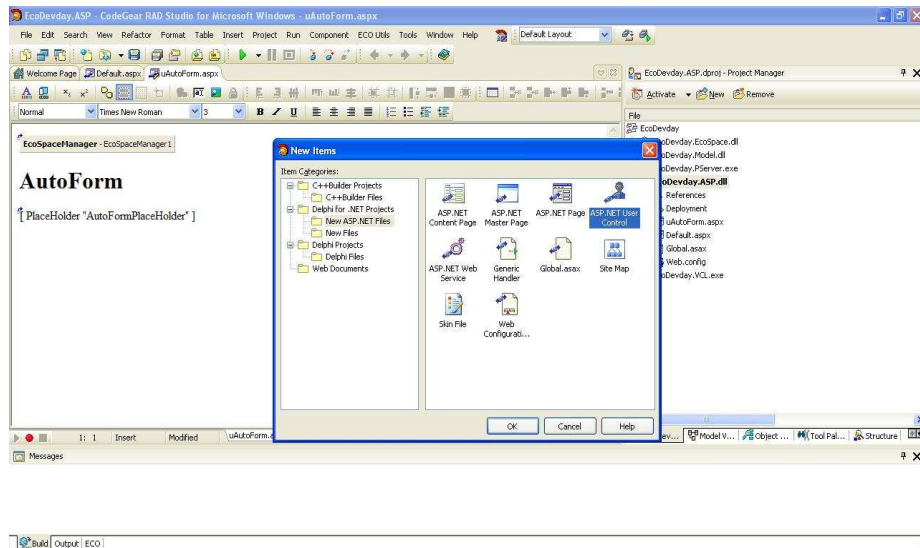


Figura III. 56. Agregar User Control

Asignar un nombre **ucMenu.ascx** y clic en **OK**.

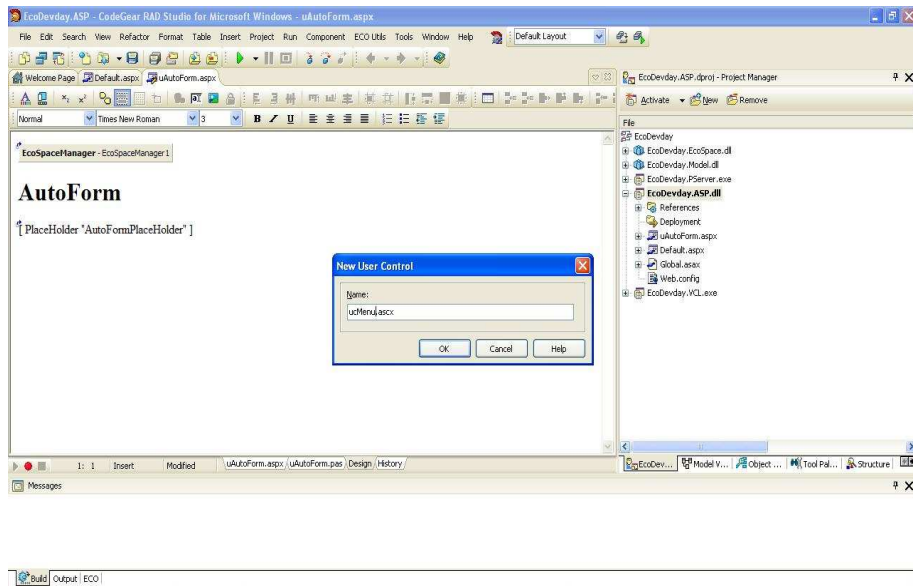


Figura III. 57. Renombrar Control Menú

Aquí es donde puede situar los controles ASP.NET para construir un menú desplegable.

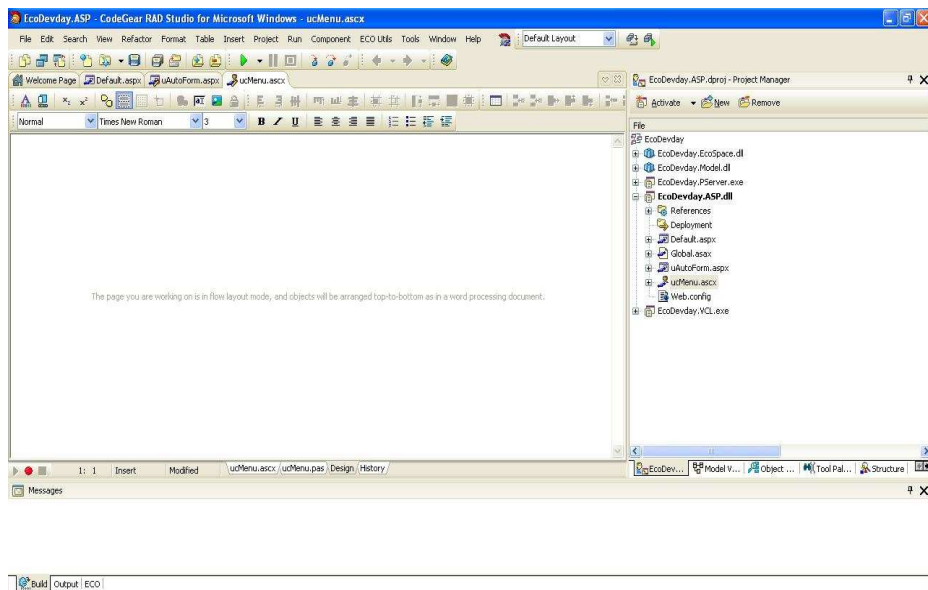


Figura III. 58. Área de diseño de Construcción del menú

El componente **TreeView** es un contenido simbólico y un menú contextual.

De la paleta de herramientas, seleccionar "**Web Navigation | TreeView**", y arrastrar hacia la zona centro izquierda del menú, clic en **editar elementos del menú**.

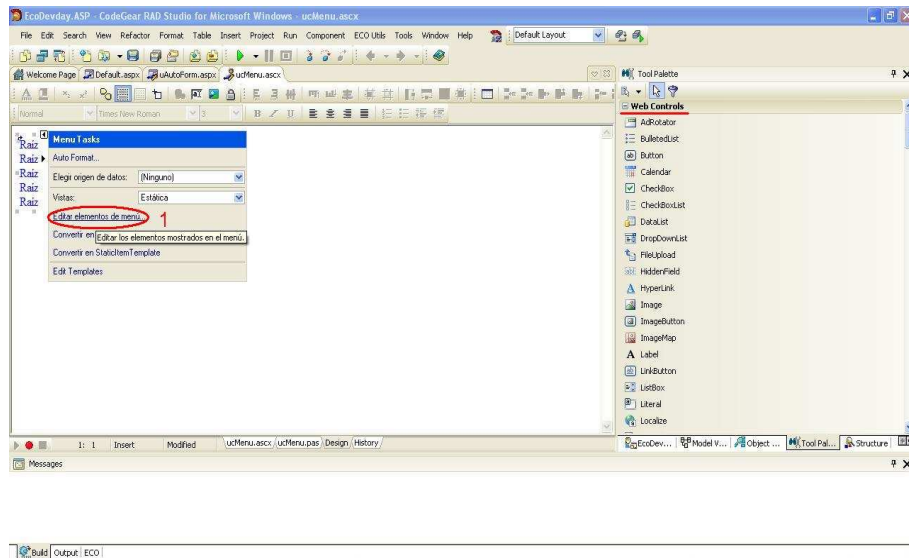


Figura III. 59. Agregar el componente Web Navigator

Se muestra el editor de elementos del menú. Seleccionar de la parte superior izquierda el icono "**Nuevo elemento**" (flecha roja), se crea un nodo raíz, en el panel de la derecha se encuentran las propiedades, llenar la propiedad **Text** con el nombre "**Paginas**".

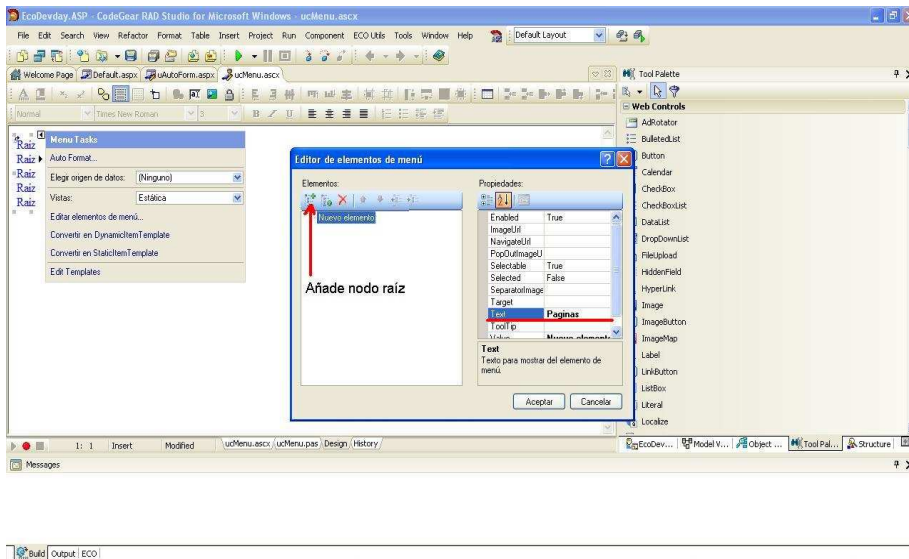


Figura III. 60. Añadir elementos al menú

Un elemento secundario se crea en dando clic en el botón "**Agregar nuevo elemento secundario**" en la propiedad **Text** nombrar como "**AutoForm**" en la propiedad

NavigateUrl, hacer clic en los puntos suspensivos y seleccionar una página aspx, para el ejemplo **uAutoForm.aspx**.

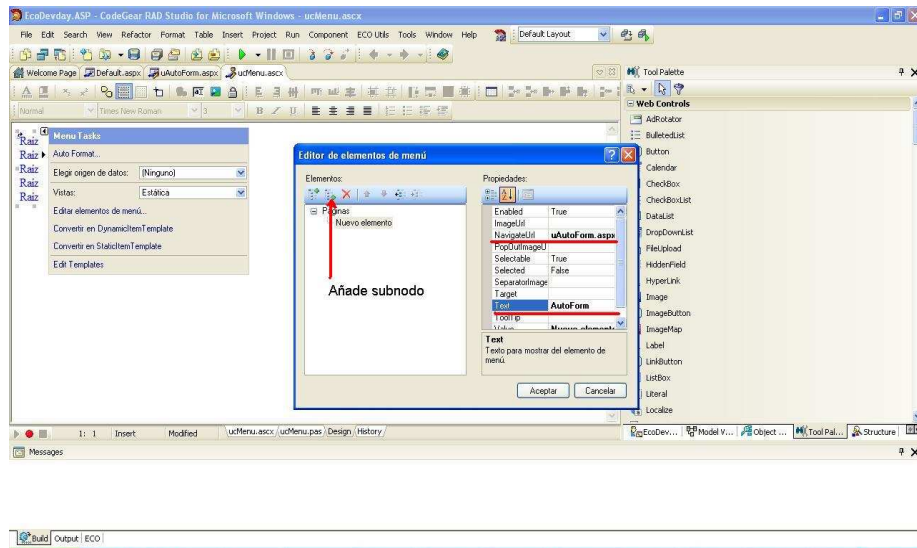


Figura III. 61. Añadir SubItems al menú

Agregar un nuevo elemento secundario, en la propiedad **Text** asignar un denominado **Default** seleccionar el **NavigateUrl**, haga clic en los puntos suspensivos y seleccione una página aspx, elegir la página por defecto **~/Default.aspx**, clic en el botón **Aceptar**.

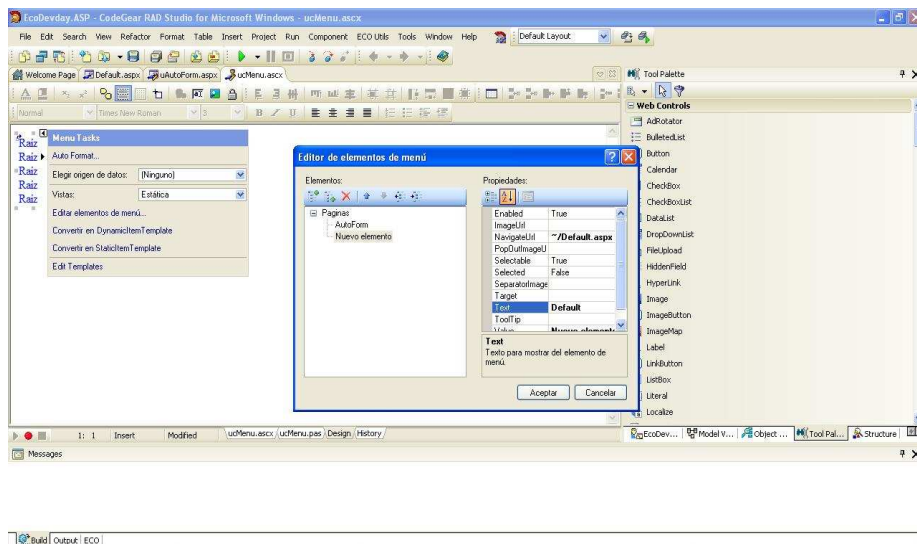


Figura III. 62. Propiedades de la herramienta menú

Abrir el formulario **uAutoForm.aspx**, y en este formulario insertar una tabla, de la paleta de herramientas seleccionar, "**HTML elements|HTML table**", arrastrar y hacer

clic en este formulario, seleccionar "Propiedades": el cuadro de diálogo aparece las propiedades de la tabla, seleccionar 2 filas, 1 columna, clic en **OK**.

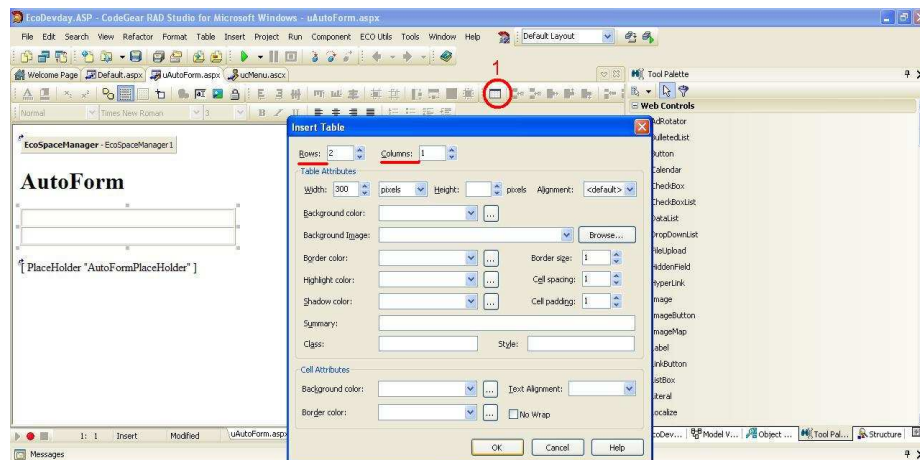


Figura III. 63. Insertar Tabla al formulario AutoForm

En esta tabla colocar los objetos **ucMenu** que acaba de crear, y el objeto **Placeholder AutoFormPlaceholder**.

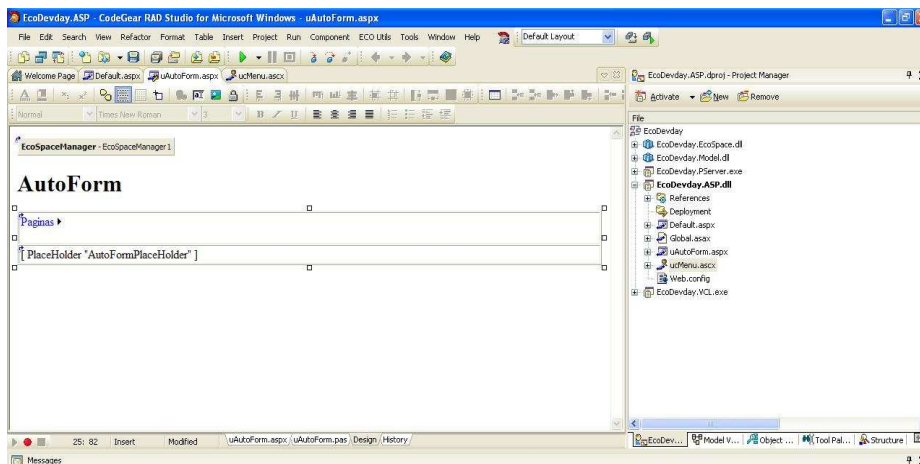


Figura III.64. Agrupar Componentes

En la página **Default.aspx**, insertar una tabla de 3 filas por una columna, en el cual se colocará los objetos **ucMenu**, el **GridView**, y un nuevo objeto **Botón**, clic en **OK**.

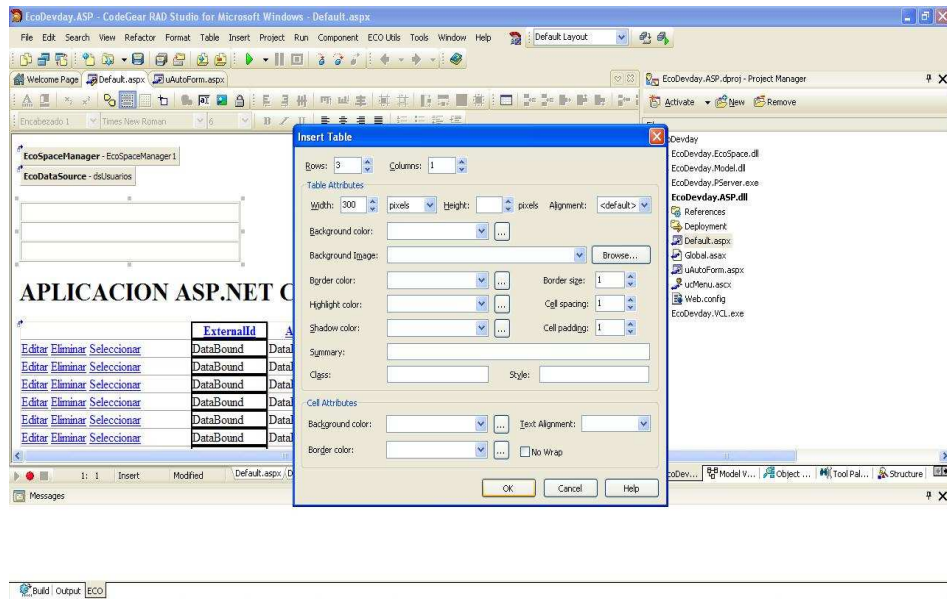


Figura III. 65. Insertar tabla a la página default.aspx

Clic en el botón, en la paleta de propiedades **Object Inspector** asignar un nombre denominar **Nuevo Usuario** en el ID se establecerá como **btnNuevoUsuario**.

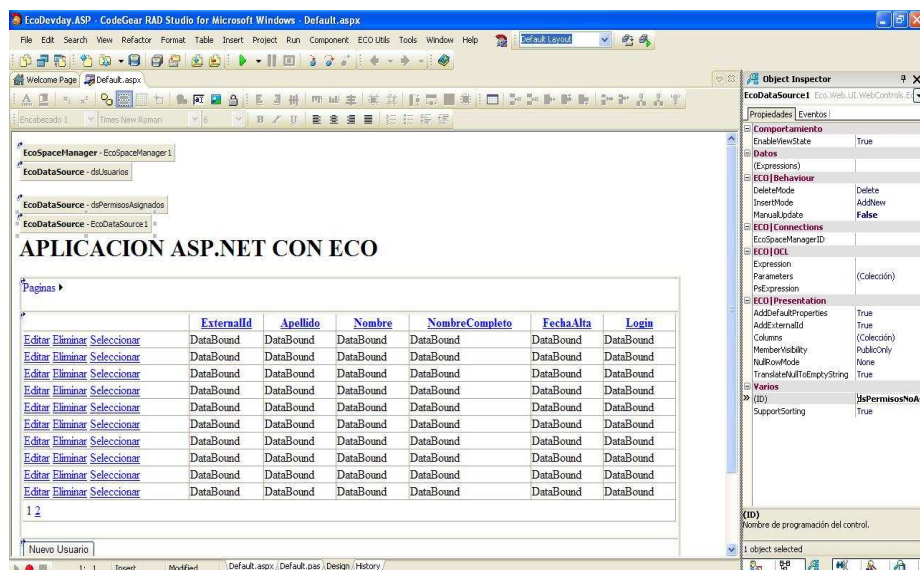


Figura III. 66. Reagrupar Componentes en el Formulario Default

Clic en el botón en los **eventos**, en el evento **click**, hacer doble clic y se muestra el código de la página Default.aspx, el primer paso es crear objetos, denominados **TUsuarioUnit** y **TPermisoUnit** a continuación de **uses**, como se observa en esta figura.

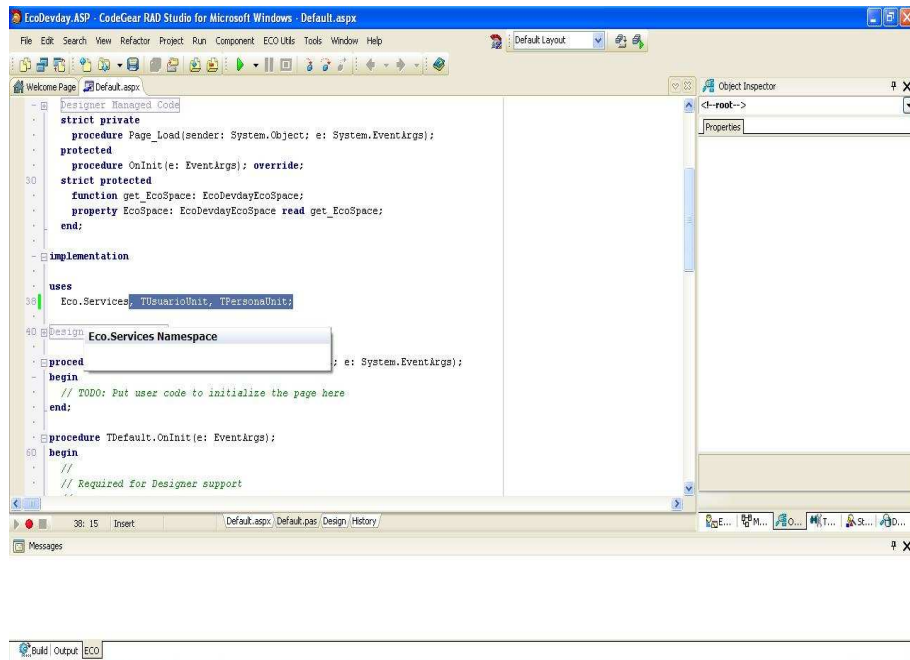


Figura III. 67. Insertar código en el Default.aspx

En el procedimiento botón se crea un usuario, y una variable **u** de tipo **TUsuarios**, crear un objeto **TUsuario**, en este nuevo objeto colocar el nombre, el apellido, el login, y actualizar la base de datos.

```
begin
  u:=TUsuario.Create(EcoSpace);
  u.Nombre:='Coloque nombre aqui';
  u.Apellido:='Coloque apellido aqui';
  u.Login:='Login Aqui';
  EcoSpaceManager1.UpdateDatabase;
  DataBind;
end;
```

El Framework Eco abstrae del motor de base de datos totalmente, de las sentencias SQL.

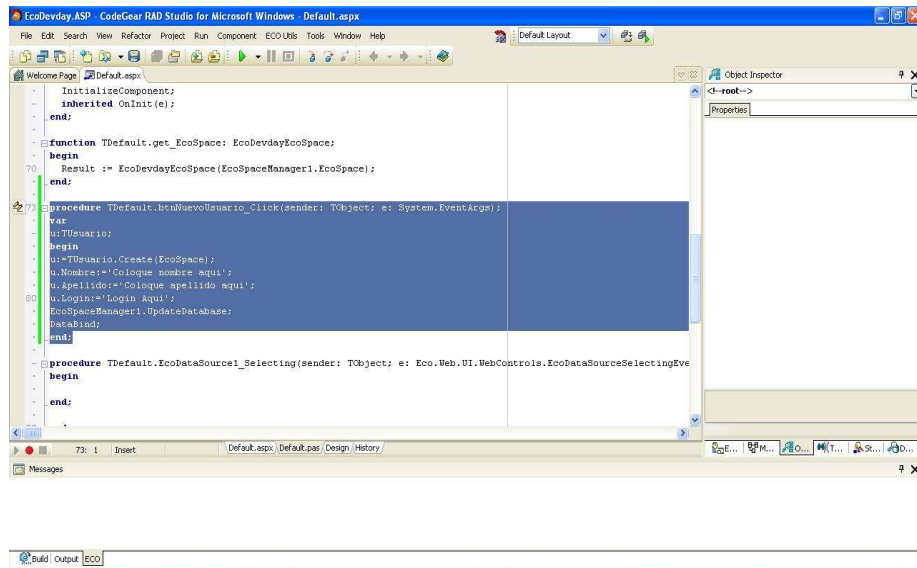


Figura III. 68. Código para agregar un nuevo usuario

En la página **Default.aspx** clic derecho **View in Browser** y se muestra en el navegador el menú con el botón nuevo usuario el cual permitirá crear un nuevo usuario.

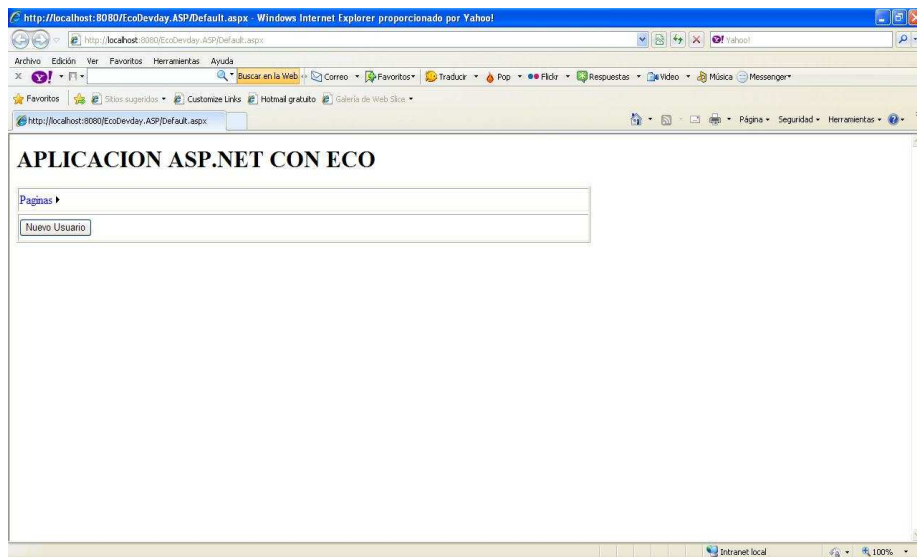


Figura III. 69. Visualización en el navegador

Al dar clic en el botón nuevo usuario se muestra un link para **editar**, **eliminar** y **seleccionar**, hacer clic en **editar** se muestra los campos del usuario el Apellido, Nombre, NombreCompleto este campo no se debe llenar ya que se crea por defecto,

FechaAlta y Login, una vez completados estos campos, existen 2 enlaces **actualizar** y **cancelar** hacer clic en actualizar.



Figura III. 70. Agregar un nuevo usuario

Insertar nuevos datos dando clic en el botón **nuevo usuario**.

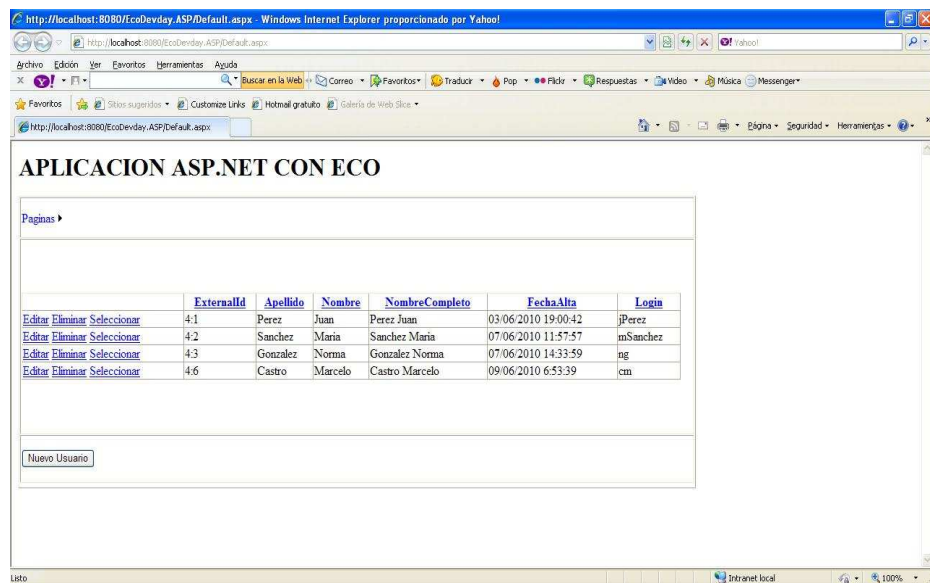


Figura III. 71. Resultados de la agregar Usuario

Abrir la base de datos clic en nueva consulta, realizar la siguiente sentencia SQL.

```
select *  
from TUsuario inner join TPersona  
on TUsuario.ECO_ID = TPersona.ECO_ID
```

Clic en ejecutar y se observa, que los registros se encuentran en la base de datos, mediante la interfaz que acaba de crear se puede eliminar, modificar y actualizar los registros.

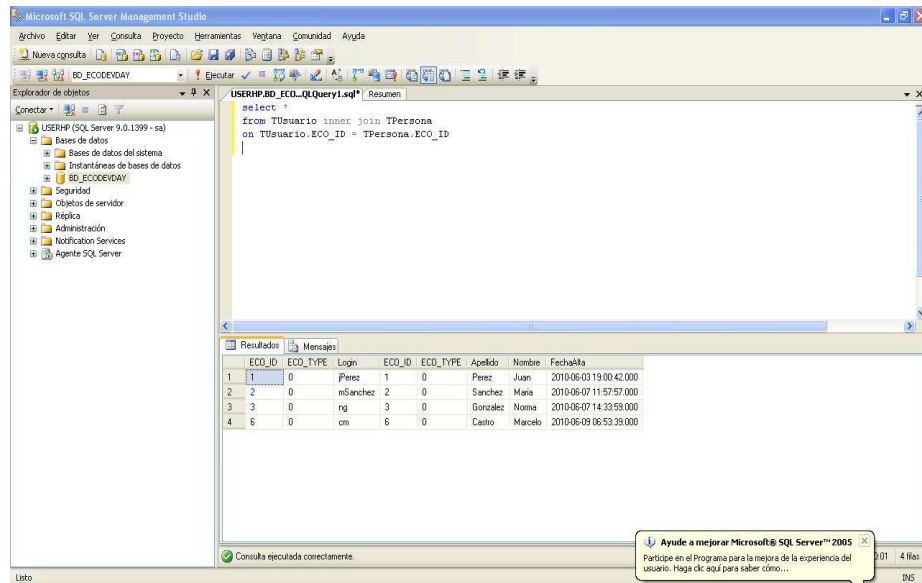


Figura III. 72. Sentencias SQL de inserción de datos

Para que la interfaz sea más atractiva dar un formato al **AutoForm** y seleccionar la opción multicolor.

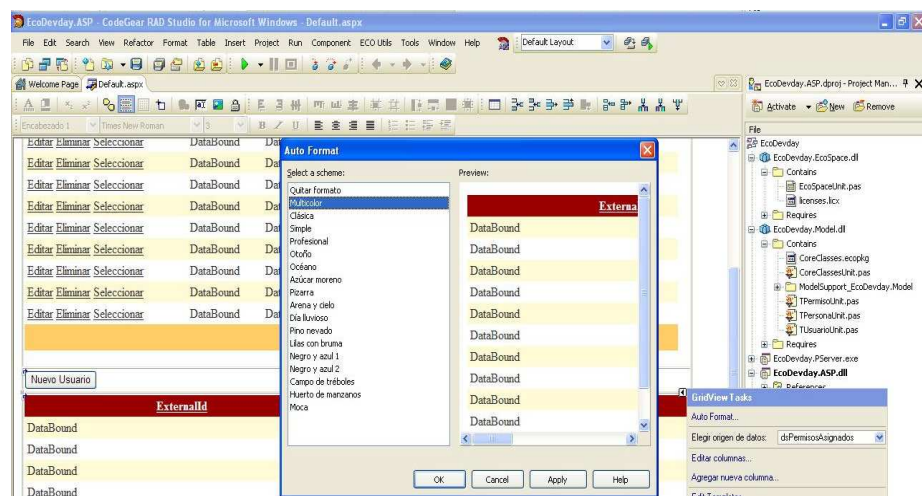


Figura III. 73. Rediseño de la aplicación

Ejecutar la vista en el navegador se muestra múltiples colores por cada registro, como se puede observar en la **figura III.74**.

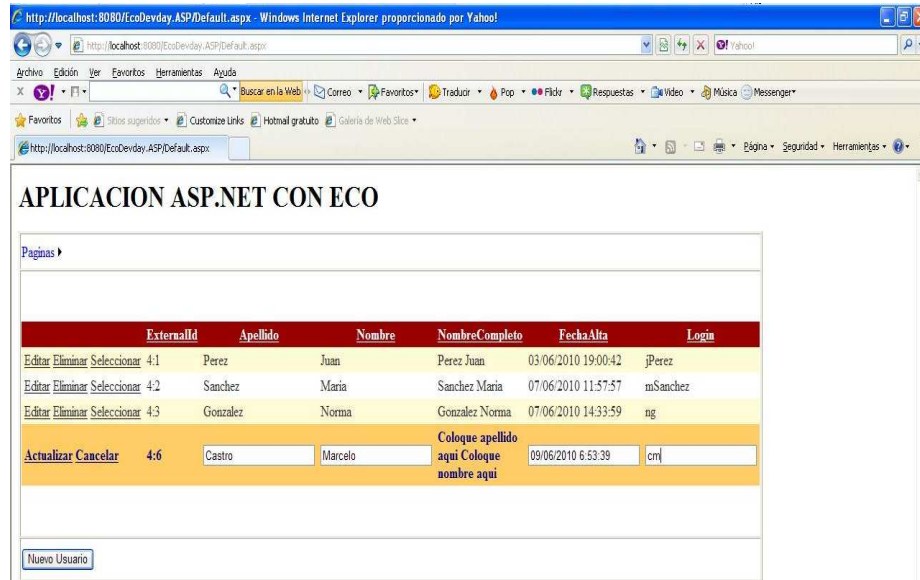


Figura III. 74. Visualización de Resultados

CAPITULO IV

IMPLEMENTACIÓN DEL SISTEMA SECNA

4.1 METODOLOGIA DE LA IMPLEMENTACION

4.1.1 INTRODUCCION

En la construcción y desarrollo de proyectos se aplican métodos y técnicas para resolver los problemas, la informática aporta herramientas y procedimientos sobre los que se apoya la ingeniería de software con el fin de mejorar la calidad de software, aumentar la productividad y trabajo de los ingenieros del software.

Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en cortos lapsos de tiempo. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación.

Las metodologías ágiles, como es el caso de XP, forman parte del movimiento de desarrollo ágil de software, que se basan en la adaptabilidad de cualquier cambio como medio para aumentar las posibilidades de éxito de un proyecto.

Para la implementación del sistema Web se utilizará la metodología ágil XP basada en una serie de valores y de prácticas de buenas maneras que persigue el objetivo de aumentar la productividad a la hora de desarrollar programas.

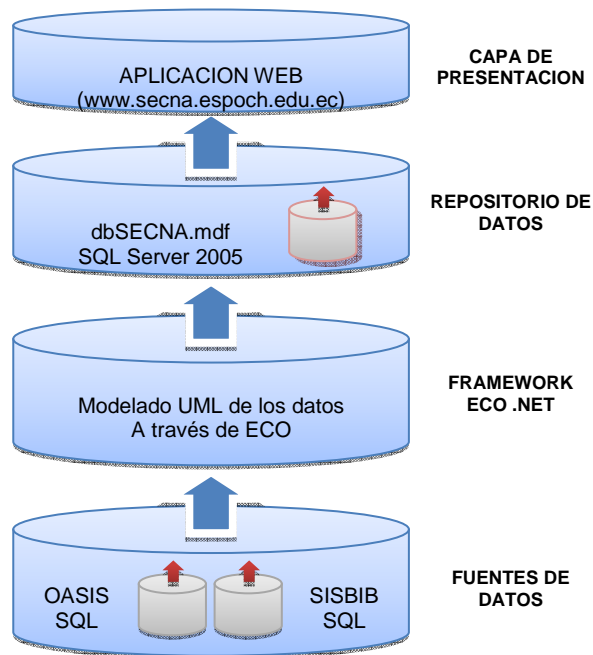


Figura IV. 75. Esquema de la solución informática

4.1.2 METODOLOGÍA XP

La metodología XP (Programación extrema) desarrollada por Kent Beck y probada por muchas compañías, desde 1998 es una de las metodologías más destacadas, muy útil en entornos de desarrollo de software en donde lo principal sea el desarrollo y no la documentación, su principal fortaleza radica en su flexibilidad en los requerimientos, se puede adaptar los cambios de requisitos en cualquier punto de la vida del proyecto, es una de las metodologías basada en la simplicidad, la comunicación y el reciclado continuo de código además contiene

una comunicación fluida entre todos los participantes, facilidad en las soluciones implementadas.

Además es una metodología iterativa, es decir que dado un requerimiento definido se puede realizar el diseño, la codificación y las pruebas.

Está conformado de 4 fases que no se las considera rígidas.

4.1.2.1 CARACTERÍSTICAS FUNDAMENTALES

Las características fundamentales del método son:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en una misma posición.
- Corrección de todos los errores antes de añadir nueva funcionalidad hacer entregas frecuentes.
- Refactorio del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.
- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal
- Simplicidad en el código: La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

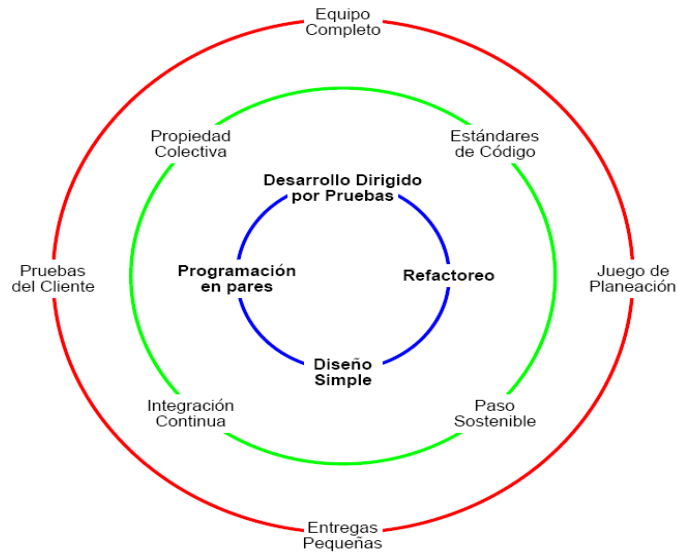


Figura IV. 76. Características Fundamentales de XP

4.1.2.2 FASES DE LA PROGRAMACIÓN EXTREMA (XP).

Se divide en cuatro fases, XP presenta una conjunción de todas las fases de desarrollo en cada momento:

Fase 1: Planificación y Análisis del Proyecto

Casos de Uso: Donde ayuda a proveer el comportamiento que tendrá el sistema frente a los usuarios potenciales, además define los límites del sistema con respecto al entorno.

Conjuntamente a ello la esencia de los casos de uso es describir la funcionalidad del sistema, independientemente de su implementación

Iteraciones: Todo proyecto que siga la metodología XP se ha de dividir en iteraciones de aproximadamente 3 semanas de duración.

Programación en pareja: Incrementa la productividad y la calidad del software desarrollado. El trabajo en pareja involucra a dos programadores trabajando en el mismo equipo; mientras uno codifica haciendo hincapié en la calidad de

la función o método que está implementando, el otro analiza si ese método o función es adecuado y está bien diseñado.

Fase 2: Diseño

Diseño de la Base de Datos: El diseño al más bajo nivel es el de la base de datos, es primordial diseñar desde la primera iteración la base de datos, el cual puede ir cambiando de acuerdo al acoplamiento en las capas superiores del sistema.

Diagrama de Clases: El diseño que se debe seguir en una metodología ágil es orientado a objetos, por ello se define el diseño del sistema a través del diagrama de clases, en donde en las posteriores iteraciones se lo puede ir refinando para un correcto diseño.

Diagrama de componentes: Aunque no es obligatorio el diseño de un diagrama de componentes, y no siempre se lo construya en esta fase es preferible diseñarlo conforme avanza las iteraciones.

Glosarios de términos: Usar glosarios de términos y una correcta especificación de los nombres de métodos y clases ayudará a comprender el diseño y facilitará sus posteriores ampliaciones y la reutilización del código.

Riesgos: Si surgen problemas potenciales durante el diseño, XP sugiere utilizar una pareja de desarrolladores para que investiguen y reduzcan al máximo el riesgo que supone ese problema.

Fase 3: Codificación

Codificación: es aquí en esta fase y en cada iteración donde se plasma lo que se ha planificado y diseñado en las 2 fases anteriores.

Aunque en ciertas ocasiones puede una iteración no producir ningún resultado visible en la codificación, es aquí donde el desarrollador o pareja de desarrolladores debe centrar el máximo de atención por lo que es la parte más importante en las metodología ágiles donde la importancia radica en el producto que se desarrolla que en la documentación de dicho producto.

Fase 4: Pruebas

Cada producto que se libera debe ser probado, las pruebas pueden ser de 2 tipos: las pruebas unitarias y las pruebas del sistema, un componente para finalizar la iteración debe pasar ambos tipos de pruebas, sino lo hace se lo debe enviar para otra iteración, porque su error puede radicarse en cualquiera de las 3 fases anteriores.

Representación gráfica de la Metodología XP

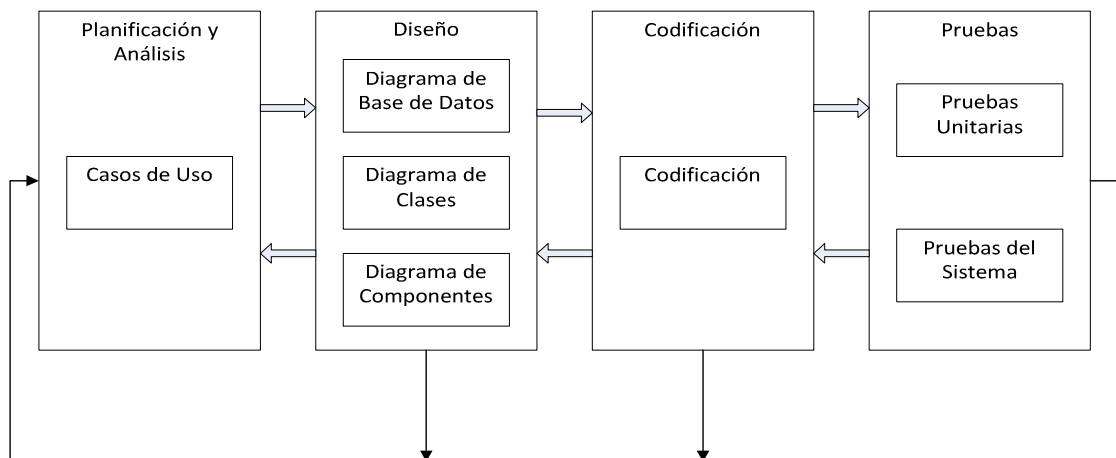


Figura IV. 77. Metodología XP

4.2 ESTUDIO PRELIMINAR

Como estudio previo al desarrollo de la aplicación, se realiza la población de la base de datos, estableciendo la selección de información acordes a las necesidades para modelar el análisis y distinguir que información se necesita extraer para implementar la misma.

4.2.1 INTRODUCCION

El sistema de Control de préstamos y emisión del certificado de no adeudar previo a la obtención del título profesional en la ESPOCH (SECNA) se llevará a cabo en la Escuela Superior Politécnica de Chimborazo a través del Departamento de Sistemas y Telecomunicaciones (DESITEL), el mismo que facilitará un registro de los préstamos de materiales y/o equipos, procesos que en la actualidad se lo realiza de manera manual, por lo que a partir de la creación de este sistema, ayudará a obtener un mejor control sobre los préstamos y a su vez sobre la emisión de los certificados de no adeudar a las diferentes dependencias de la institución optimizando de esta manera el tiempo y los recursos de la ESPOCH.

Para poder definir los requerimientos, se identificará los puntos críticos en los cuales la institución necesita la intervención de la aplicación a desarrollarse, de tal manera que se permita facilitar el control de los préstamos y la emisión de los certificados.

Para llevar a cabo la ejecución del sistema se debe tomar en cuenta la naturaleza de la aplicación que deseamos implementar, el ambiente de trabajo en el que se desenvolverá y los usuarios que van a utilizar.

Por tal motivo se compromete a desarrollar un sistema dentro de la institución con la ayuda del departamento de Sistemas y Telemática, para que la ESPOCH brinde un mejor servicio a sus estudiantes.

4.2.2 ANTECEDENTES TECNOLOGICOS DE LA INSTITUCION

Para la ejecución de la aplicación se requiere una buena infraestructura de soporte que garantice el acceso a la aplicación desde donde se requiera. La

Escuela superior Politécnica de Chimborazo cuenta actualmente con una infraestructura aceptable y que servirá de base para la ejecución del mismo.

Entre los principales recursos tecnológicos que posee la ESPOCH se encuentra:

4.2.2.1 Red institucional

- Un backbone principal de fibra óptica que llega a cada facultad para ser distribuidos a cada escuela mediante un switch para fibra.
- La red utiliza principalmente un cableado de cobre y se utiliza la fibra óptica para cubrir distancias grandes a manera de backbone.
- Se encuentra con equipos de conmutación que aceptan medios de cobre y fibra óptica.

4.2.2.2 Acceso Satelital a Internet

- Conexión satelital a internet
- WLAN, para acceso inalámbrico a Internet.
- Acceso a Internet desde casi todos los nodos conectados a la red.
- Esquemas de Seguridad.
- Se cuenta con un firewall, a nivel de sistema operativo se tiene cuentas de usuario teniendo 2 niveles de usuario (Administrador, cliente).
- Infraestructura de Servidores.

4.2.2.3 Infraestructura de Servidores

- Un servidor web que utiliza la plataforma Linux y Apache.
- Un conjunto de servidores que utilizan la plataforma Windows e IIS, destinado principalmente al soporte del Sistema Académico Institucional.

4.2.2.4 Recursos Software

- Sistema Operativo del Servidores
 - Linux

- Windows XP
- Windows 2003 server
- Sistema Operativo de clientes
 - Linux
 - Windows XP

4.2.2.5 Recursos Hardware

Actualmente la ESPOCH cuenta con un backbone de fibra multimodal la cual parte desde el DESITEL (Departamento de Sistemas y Telemática) hacia los diferentes puntos existentes en la ESPOCH, en lo referente a los equipos activos la institución cuenta con un equipo central el mismo que permite realizar una administración de forma centralizada de la red dentro del DESITEL, cuenta también con una barrera de protección (firewall) y un portal Web donde se encuentran enlaces a los diferentes sistemas que brindan servicios requeridos por la institución. Gracias a las gestiones de las autoridades, los estudiantes y docentes cuentan con Internet Inalámbrico y próximamente Internet2

ESPECIFICACIONES TECNICAS DE LOS DISPOSITIVOS DEL NODO CENTRAL DE LA RED DE LA ESPOCH (DESITEL)

Tabla Nº IV. 1. Especificaciones técnicas

Dispositivo	Marca	Modelo	Número de serie	MAC Addr	IP
--------------------	--------------	---------------	------------------------	-----------------	-----------

Firewall	3COM	Super Stack II	LNCB4052BEC	00-30-1E-05-2B-S	172.30.80.7
Switch 4950	3COM	Super Stack 3	05017FZV8DC2780	000BACD2780	172.30.80.2
Switch 3300 FX	3COM	Super Stack II	080027B2V89896F8	000BACD2780	172.30.80.3
Switch 4228 G	3COM		01037Z1V2P90C77	000D540C7740	-
Switch Baseline 10/100		Super Stack 3	0101/LV5G290073303	-	-
MODEM Satelital P400	PARADISE DATACOM	-	-	-	-
Switch	3COM	-	03067PWZB9078	000D54078B40	172.30.8.26

SERVIDORES

Los servidores que están implementados en la Intranet de la ESPOCH

son:

Tabla Nº IV. 2. Especificaciones de servidores

Servidores	Procesador	Disco Duro	Memoria	CD-ROM	Tarjeta de Red
Servidor Web	2 procesadores INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor DHCP	2 procesadores INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor de Base de Datos	2 procesadores INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor Proxy	2 procesadores INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor DNS Intranet	2 procesadores INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor de Mail	1 procesador INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps
Servidor Académico	1 procesador INTEL CEON 2.4 Hz	80GB	1GB en RAM	52x	Fa 10/100 Mbps

4.2.3 DEFINICIÓN DE ALTERNATIVAS DE SOLUCIÓN

De acuerdo al estudio realizado, se ha tomado en cuenta realizar una aplicación Web llamada SECNA, considerando todos los parámetros definidos anteriormente permitiendo:

- Utilizar una base de datos en SQL Server 2005.
- Hacer uso de una interfaz Web desarrollada en RAD Studio 2007.

ALTERNATIVA

La aplicación web a desarrollarse recoge todas las consideraciones antes mencionadas y se divide en dos módulos importantes:

MODULO DE ADMINISTRACION

En este módulo tiene todo el control de la aplicación web

Funcionalidad

Las tareas que realiza este módulo son:

- El módulo de Administración es el encargado de gestionar el funcionamiento del sistema, es el componente principal en el cual se definen los parámetros mediante los cuales se maneja la aplicación.
- Este módulo realiza la creación de usuarios y sus respectivos permisos para el acceso y manejo de la aplicación
- Realiza el ingreso, la eliminación y la actualización de dependencias, materiales, costos y préstamos.
- Además emite un certificado de no adeudar a la institución.

Ubicación

Este módulo se encontrará implementado en el servidor Web localizado en el Departamento de Sistemas y Telemática de la ESPOCH, y será accesible por medio de la Intranet Institucional.

MODULO OPERATIVO

En este módulo se ha tomado en cuenta a tres clases de usuarios como son el responsable de préstamos, secretaria de carrera y el estudiante, con su funcionalidad respectivamente.

Funcionalidad

Las tareas que realiza este módulo son:

- El responsable de préstamos será capaz de ingresar, eliminar y actualizar los datos de los materiales y realizar un préstamo.
- La secretaria de escuela emitirá el certificado de no adeudar a la institución
- El estudiante podrá consultar sus préstamos pendientes de pago o de entrega del material y su respectiva deuda en caso de tenerla.

Ubicación:

Este módulo se encontrará implementado en el servidor web localizado en el Departamento de Sistemas y Telemática de la ESPOCH, y será accesible por medio de la Intranet Institucional.

DESCRIPCION ARQUITECTURA DE ALTO NIVEL

El esquema de la arquitectura y la interacción la representamos a continuación a través de un diagrama de despliegue en la Figura IV.78.

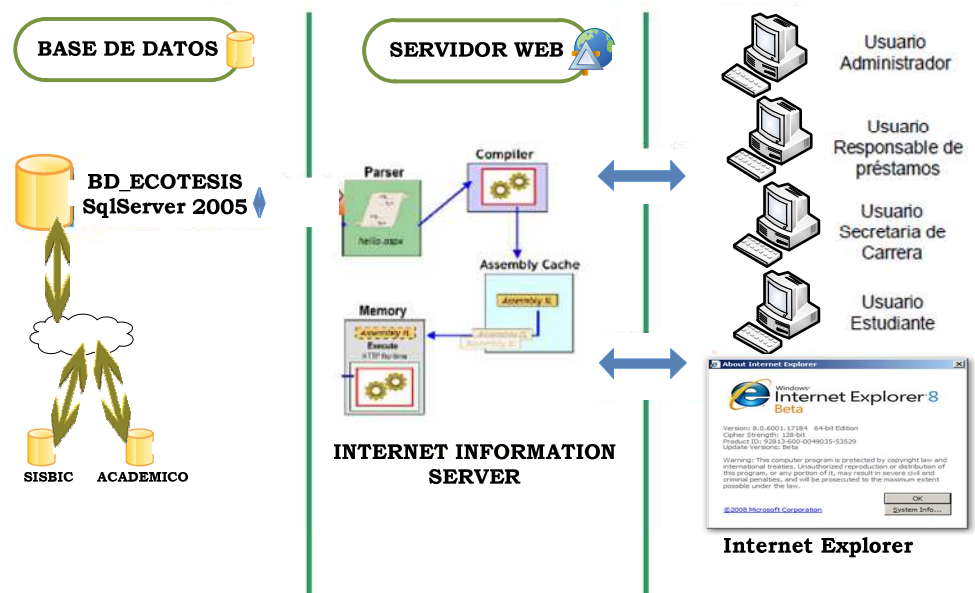


Figura IV. 78. Arquitectura del Sistema SECNA

4.2.4 SELECCIÓN DE FUENTES DE INFORMACION

Para realizar la selección de información se realizó una investigación a fondo de las bases de datos existentes en la ESPOCH y que sean útiles para la aplicación a desarrollarse.

De las bases analizadas se tomaron las siguientes:

- OASIS, pertenece al Sistema Académico, su motor de base de datos es SQL Server 2000.
- BIBLIOTECA que pertenece al Sistema de Bibliotecas llamado SYSBIB y su motor de base de datos es SQL Server 2000

4.3 ESPECIFICACION DE REQUERIMIENTOS

La Especificación de Requerimientos es la tarea de la ingeniería del Software que establece un puente entre la asignación del software a nivel de sistema y el diseño del software. El análisis de requisitos facilita al ingeniero en sistemas la especificación clara y precisa de la función y del rendimiento del software, la descripción de la interfaz con otros elementos del sistema y el establecimiento de las restricciones de diseño que debe considerar el software, y de la misma manera el SRS contiene una descripción completa del comportamiento externo del sistema para los usuarios finales.

4.3.1 AMBITO DEL SISTEMA

La razón que motiva el desarrollo de la aplicación es implementar un sistema para automatizar el proceso de emisión de certificados de no adeudar a ninguna dependencia de la ESPOCH mejorando la forma actual como se lleva este proceso. Para solucionar ésta problemática se ha planteado el desarrollo del sistema "SECNA".

El éxito de esta actividad depende directamente de la colaboración del personal de las entidades involucradas en el mismo, así como de la habilidad para abstraer el ámbito del problema.

4.3.2 BENEFICIOS DEL SISTEMA

- Comodidad para los estudiantes al obtener el certificado de no adeudar en las dependencias de la institución.
- Disminución de tiempo al evitar trámites burocráticos al visitar las dependencias de la ESPOCH.
- Facilidad para administradores o responsables del sistema en las diferentes secretarías, dependencias y departamentos de la institución al momento de generar informes y reportes, así como consultas y registros de préstamos.
- Incrementar la satisfacción de los usuarios.
- Mejorar el servicio prestado a los usuarios.
- Mejorar la imagen de la institución con un Sistema de Emisión de Certificados de no adeudar a la ESPOCH previo a la obtención del título académico.
- Mejorar la toma de decisiones

4.3.3 FASE I: PLANIFICACIÓN Y ANÁLISIS DEL PROYECTO

El objetivo que se desea alcanzar, en éste punto con la aplicación, al utilizar una metodología de desarrollo ágil, es aumentar la productividad y funcionalidad a la hora de elaborar el sistema.

La aplicación se compromete a satisfacer los requerimientos especificados, lo que mejorará el ambiente, calidad, tiempo y optimización de recursos de todos los estudiantes de la ESPOCH.

4.3.3.1 DEFINICION DE LOS REQUERIMIENTOS DEL USUARIO

Dentro del análisis que se ha realizado dentro de la institución, los requerimientos han plasmado en algunos puntos críticos, los cuales se menciona a continuación:

ESTUDIANTES

- Controlar los préstamos realizados y ser más puntuales en las fechas de entrega.

ADMINISTRADOR

- Realizar una migración automática diaria para obtener información de facultades, carreras, escuelas, estudiantes procedentes del Sistema Académico y deudores del Sistema de Bibliotecas (SISBIB)
- Realizar el ingreso, eliminación y actualización de dependencias, materiales o equipos y préstamos.
- Ejecutar la emisión del certificado de no adeudar a la ESPOCH
- Efectuar la administración de usuarios

RESPONSABLES PRESTAMOS

- Realizar la inserción, eliminación y actualización de materiales y préstamos.
- Realizar registro de la devolución del préstamo o material.

SECRETARIA DE ESCUELA

- Realizar la emisión del certificado de no adeudar a la ESPOCH para la obtención del título académico.

4.3.3.2 DEFINICION DE LOS CASOS DE USO ESENCIALES Y EL FORMATO EXPANDIDO

La creación de los casos de uso es una técnica que permite mejorar la comprensión de los requerimientos y se constituyen como historias o casos de utilización de un sistema; no son exactamente los requerimientos ni las especificaciones funcionales, sino que ejemplifica e incluyen fácilmente los requerimientos en las historias que narran.

4.3.3.2.1 IDENTIFICACION DE CASOS DE USO

Como guía para la identificación inicial de casos de uso hemos utilizado el método basado en eventos que nos permitirán identificar los eventos externos al sistema como se observa en la Tabla IV.3

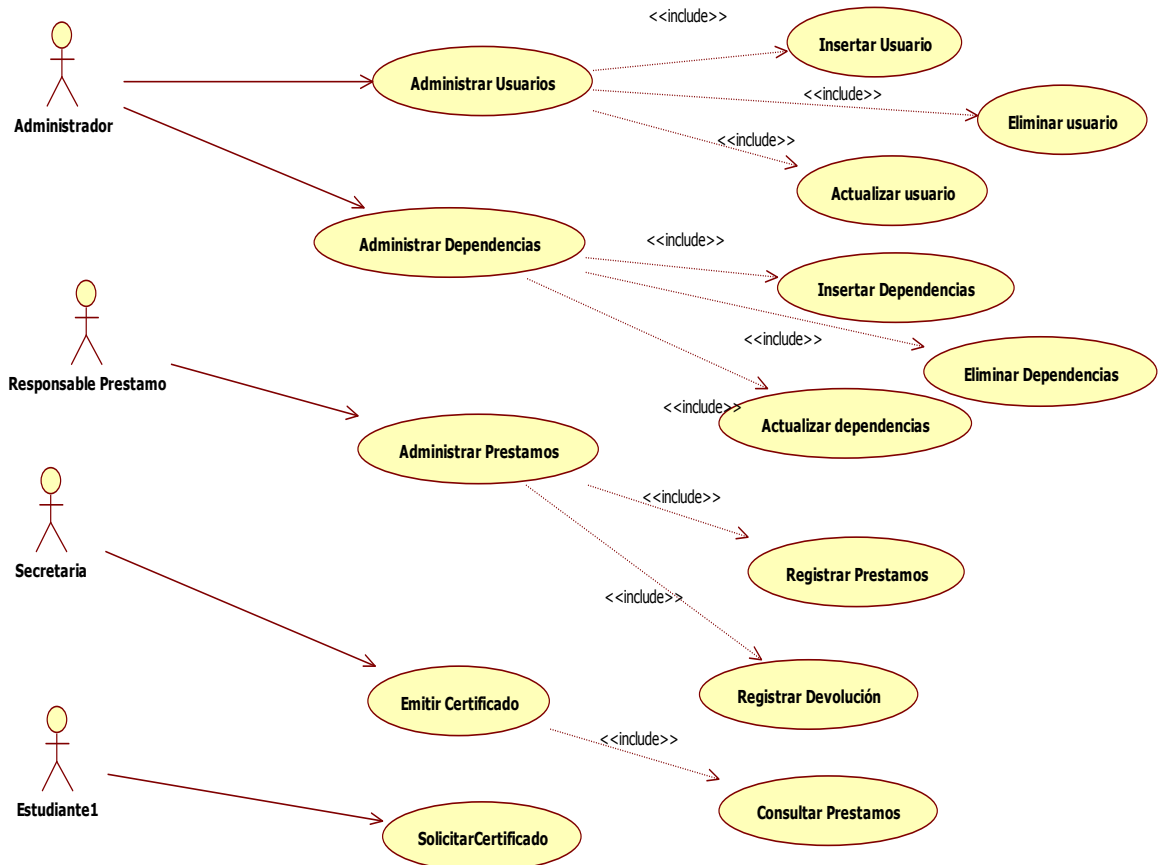
Tabla Nº IV. 3. Eventos externos del sistema

No	Nombre de caso de uso	Eventos	Actores
1	Autenticación del usuario	Ingresar a la página web	Administrador Responsable de prestamos Secretaria de carrera Estudiante
2	Generación de dependencias	Actualizar Eliminar Ingresar	Administrador
3	Dependencias	Listar	Responsable de prestamos
4	Estudiantes	Listar	Administrador Responsable de

			prestamos
5	Generación de materiales y/o equipos	Actualizar Eliminar Ingresar	Administrador Responsable de prestamos
6	Generación de usuarios	Actualizar Eliminar Ingresar	Administrador
7	Generación de certificado	Consultar Imprimir	Administrador Secretaria de carrera
8	Generación de reporte de préstamos	Mostrar información	estudiante

DIAGRAMAS DE CASOS DE USO

Se realizará una representación gráfica del entorno del sistema y su funcionalidad principal.



4.3.3.2.2 DESARROLLO DE CASOS DE USO EN FORMATO EXPANDIDO

Un caso de uso expandido describe un proceso más a fondo que el de alto nivel, es decir, describe paso a paso el curso normal de los eventos.

A continuación se presenta los casos de uso en formato expandido más importantes del sistema actual y propuesto:

SISTEMA ACTUAL

CASO DE USO #1 (CU_CPA)

CASO DE USO	Control de préstamos de materiales y/o equipos
ACTORES	Secretarias de carreras, responsables de préstamos, estudiantes
TIPO	Primario esencial

PROPOSITO	Registro de préstamos de materiales y/o equipos en las diversas dependencias de cada facultad de la ESPOCH.	
DESCRIPCIÓN	El estudiante accede al préstamo de un material y/o equipo.	
REFERENCIAS	Ninguna	
CURSOS TÍPICO DE EVENTOS:		
	ACTOR	SISTEMA
	<p>1.- El estudiante solicita el préstamo de un material y/o equipo de una dependencia de la facultad.</p> <p>4.- Entrega de documentos de identificación.</p> <p>6.- El estudiante hace uso del objeto</p> <p>7.- El estudiante se acerca a entregar el objeto del préstamo.</p> <p>10. Recibe documentos</p>	<p>2. El responsable de la dependencia busca el objeto solicitado.</p> <p>3.- Solicita documentos de identificación.</p> <p>5.- Entrega el objeto de préstamo y registra el préstamo.</p> <p>8.- El responsable recibe el objeto de préstamo.</p> <p>9.- El responsable verifica si la entrega esta dentro del tiempo límite.</p> <p>9.1 SI: Busca los documentos</p> <p>9.2 NO: Calcula el tiempo sobrepasado.</p> <p>11. Devuelve documentos al estudiante</p>
CURSOS ALTERNATIVOS		
LINEA 9.2 NO: Calcula el tiempo sobrepasado, emite una sanción monetaria o retiene los documentos del estudiante.		

SISTEMA PROPUESTO

CASO DE USO #2 (CU_AU)

CASO DE USO	Autenticación del usuario.
ACTORES	Administrador, Responsable de préstamos, secretaria

	de carrera, estudiante.
TIPO	Primario esencial.
PROPOSITO	Ingresar a la página web.
DESCRIPCIÓN	El usuario accede al sitio web para observar información general.
REFERENCIAS	Ninguna
CURSO TIPICO DE EVENTOS:	
ACTOR	SISTEMA
1.- El usuario solicita iniciar sesión. 3.- Ingresa datos solicitados. 5.-Visualiza opciones	2.-El sistema solicita usuario y clave. 4.- Verifica y valida datos <ul style="list-style-type: none"> • Satisfactorio (Accede a la página web) • Insatisfactorio.
CURSOS ALTERNATIVOS:	
LINEA 4: (Insatisfactorio) vuelve a pedir usuario y clave	

CASO DE USO #3 (CU_GAU)

CASO DE USO	Administrar Usuarios
ACTORES	Administrador
TIPO	Primario esencial
PROPOSITO	Describir el Proceso para Administrar los usuarios del Sistema SECNA
DESCRIPCIÓN	La iteración comienza cuando el Administrador Elimina, Modifica o Crea los Datos de los Usuarios que van a ocupar el Sistema.
REFERENCIA	CU_AU
CURSO TIPICO DE EVENTOS:	
ACTOR	SISTEMA
	1.- El sistema le habilitará la opción

<p>2.- Escoge la Acción que desea realizar como: Guardar, Eliminar, Buscar o Modificar Usuarios.</p>	<p>Administrar Usuarios.</p> <p>3.- Realiza la operación descrita por el usuario y muestra un mensaje de confirmación para realizar la acción requerida si es una respuesta afirmativa se realiza.</p> <p>4.- Guarda los cambios realizados por el usuario y emite una confirmación.</p>
--	--

CASO DE USO #4 (CU_GAD)

CASO DE USO	Administrar Dependencias	
ACTORES	Administrador	
TIPO	Primario esencial.	
PROPOSITO	Registrar el Cambio o modificación de Dependencias dentro de la ESPOCH	
DESCRIPCIÓN	La iteración comienza cuando el Usuario Administrador Decide Crear, Modificar, eliminar una dependencia dentro de la ESPOCH	
REFERENCIAS	CU_AU	
CURSO TIPICO DE EVENTOS:		
<p style="text-align: center;">ACTOR</p> <p>1.- Usuario escoge opción administrar dependencias.</p> <p>3.- El usuario Escogerá la opción más adecuada entre las cuales tendrá: Crear, Eliminar, Modificar una dependencia</p> <p>5.- El Usuario Aprobará la opción</p>	<p style="text-align: center;">SISTEMA</p> <p>2.- Se mostrará las opciones para administrar todas las dependencias que se encuentran en la ESPOCH.</p> <p>4.- El sistema Analizará la petición del Súper usuario y emitirá un mensaje de confirmación para la acción requerida.</p> <p>6.- EL sistema ejecutará la solicitud ingresada</p> <p>7.- Emitirá un mensaje de confirmación de que la acción fue realizada</p>	

CASO DE USO #5 (CU_GAI)

CASO DE USO	Administrar Ítems	
ACTORES	Administrador, Empleado	
TIPO	Primario esencial.	
PROPOSITO	Describir el escenario en el que se Administra los ítems en el sistema	
DESCRIPCIÓN	El Administrador o Empleado ingresa con su login y clave, una vez validado y verificado ingresan al sistema, debido a que estos usuarios cuentan con ciertos permisos, estos pueden realizar las acciones de insertar, eliminar, guardar, modificar y buscar un Ítem	
REFERENCIAS	CU_AU	
CURSO TIPICO DE EVENTOS:		
	ACTOR	SISTEMA
	<p>1.- El usuario escoge la opción de administrar ítems.</p> <p>3.- Ingresa a la pantalla de administrar ítems</p> <p>5.- Escoge un de las Opciones como: Guardar, Eliminar, Buscar o Modificar un ítems</p>	<p>2.- El sistema le muestra la interfaz definida para la administración de los Ítems</p> <p>4.- Muestra las opciones definidas para él.</p> <p>6.- Realiza la operación descrita por el usuario y muestra un mensaje de confirmación para realizar la acción requerida si es una respuesta afirmativa se realiza automáticamente la acción caso contrario no se la realiza</p> <p>7.- El sistema guarda automáticamente los cambios</p>

CASO DE USO #8 (CU_GAP)

CASO DE USO	Administrar préstamos
ACTORES	Administrador, responsable de préstamos
TIPO	Primario esencial.

PROPOSITO	Administrar los préstamos.	
DESCRIPCIÓN	El proceso empieza cuando el Administrador o responsable de préstamo realiza una inserción, actualización o eliminación de los préstamos de un estudiante	
REFERENCIAS	CU_AU	
CURSO TIPICO DE EVENTOS:		
ACTOR	SISTEMA	
<p>1.- Ingres a la pantalla de administrar préstamos: Guardar, Eliminar, Buscar o Modificar un préstamo</p> <p>3.- Ingres a la cédula del estudiante y escoge buscar.</p> <p>5.- Elige el ítem y presiona la opción a ejecutar el préstamo.</p> <p>9.- El usuario confirma la operación realizada.</p>	<p>2.- Muestra en pantalla un formulario para buscar el estudiante que va a realizar el préstamo.</p> <p>4.-Busca el estudiante</p> <p>4.1.- Existe: muestra la lista de ítems de la dependencia.</p> <p>4.2.- No existe: emite un mensaje de error</p> <p>6.- Valida opción:</p> <p>6.1.- Editar: muestra todos los datos del préstamo a editar.</p> <p>6.2.- Eliminar: elimina el préstamo del ítem seleccionado.</p> <p>7.- Guarda los datos editados.</p> <p>8.- Emite mensaje</p> <ul style="list-style-type: none"> • Satisfactorio • Insatisfactorio 	
CURSOS ALTERNATIVOS:		
Línea 8 (Insatisfactorio): Si el ingreso es incorrecto		

CASO DE USO #9 (CU_CP)

CASO DE USO	Consultar Préstamos
ACTORES	Invitado Administrador Empleado

TIPO	Primario esencial	
PROPOSITO	Describir el escenario en el que se obtiene los datos del estudiante que realizo un préstamo	
DESCRIPCIÓN	El usuario que puede ser: Invitado, Administrador o Empleado que requieran la consulta de un préstamo realizado por un estudiante en las dependencias de la ESPOCH. Misma consulta que servirá para determinar el estado actual de los estudiantes, que constaran en los registros de la Base de Datos.	
REFERENCIA	CU_AU	
CURSO TIPICO DE EVENTOS:		
	ACTOR	SISTEMA
	<p>1.- Solicita el certificado</p> <p>3.-Proporciona el número de cédula.</p> <p>4.- El usuario ingresa el número de cédula</p>	<p>2.- Muestra la pantalla de ingreso de cedula</p> <p>5.- Valida los datos ingresado.</p> <p>6.- Se ejecuta la petición al sistema académico con los datos del estudiante que solicita el certificado</p> <p>7.- Devuelve los datos de la petición</p> <p>8.- Despliega la información para saber si tiene o no préstamos.</p>

CASO DE USO #10 (CU_ER)

CASO DE USO	Emitir Reportes
ACTORES	Administrador, Invitado, Empleado
TIPO	Primario esencial.
PROPOSITO	Poder Emitir los reportes de las Acciones que el sistema realiza durante un periodo de tiempo
DESCRIPCIÓN	El curso de los eventos comienza cuando los administradores de las dependencias desean obtener un reporte de todos los movimientos de los Ítems pertenecientes a estas

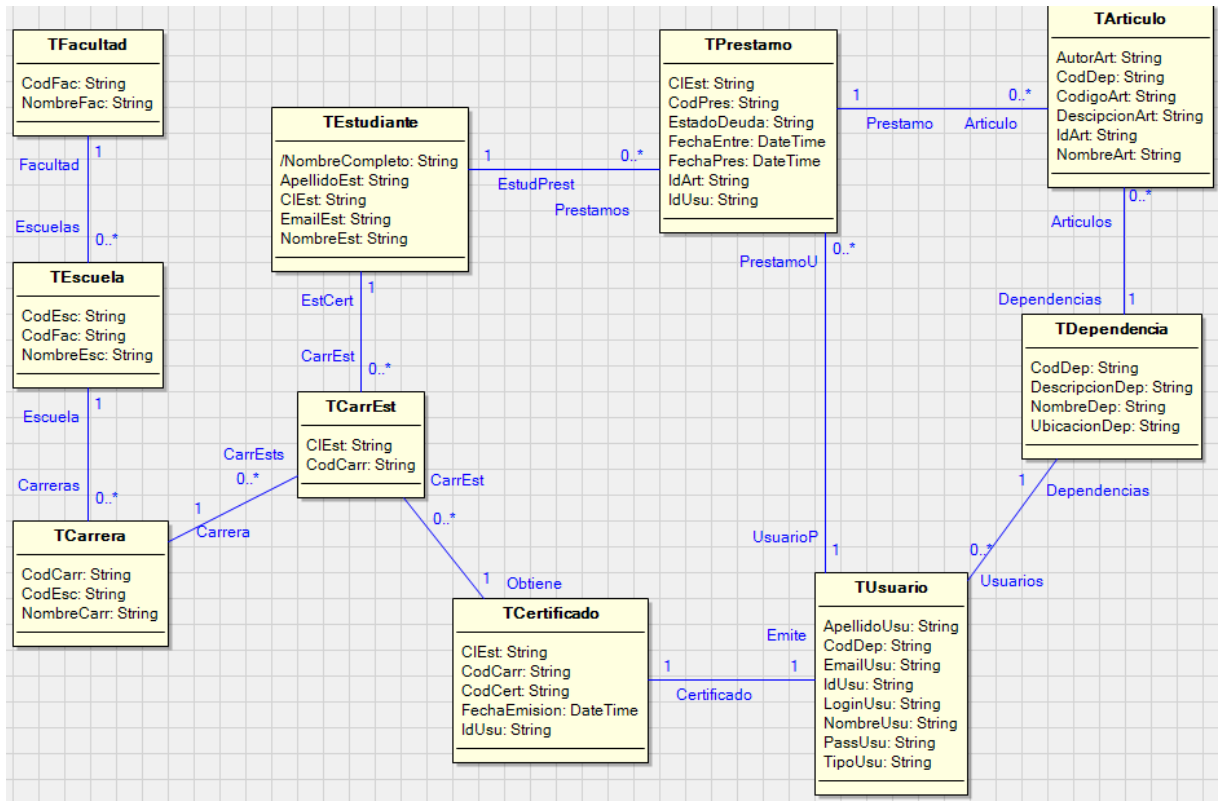
REFERENCIA	CU_AU
CURSO TIPICO DE EVENTOS	
ACTOR	SISTEMA
1.- El Administrador de la dependencia escogerá la opción reportes	2.- El sistema mostrará la pantalla para ingresar los datos del reporte a solicitar
3.- Ingresara los datos correspondientes.	4.- Validara los datos ingresados
	5.- Emitirá los resultados de los Reportes y presentará la opción de imprimir.

4.3.4 FASE 2: DISEÑO

En esta fase se integrarán los módulos, con su interfaz, para así obtener el sistema completo SECNA, además se desarrollarán las pruebas respectivas para determinar la existencia de fallos en el sistema y también se analizará si la interfaz presentada es amigable o necesita cambios.

4.3.4.1 MODELO CONCEPTUAL

A continuación se identifican las clases del sistema de control de préstamos y emisión del certificado de no adeudar a la ESPOCH.



4.3.4.2 DIAGRAMA DE SECUENCIA

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Esta descripción es importante porque puede dar detalle a los casos de uso, aclarándolos al nivel de mensajes de los objetos existentes.

Diagrama de Secuencia para: Solicitar Artículo

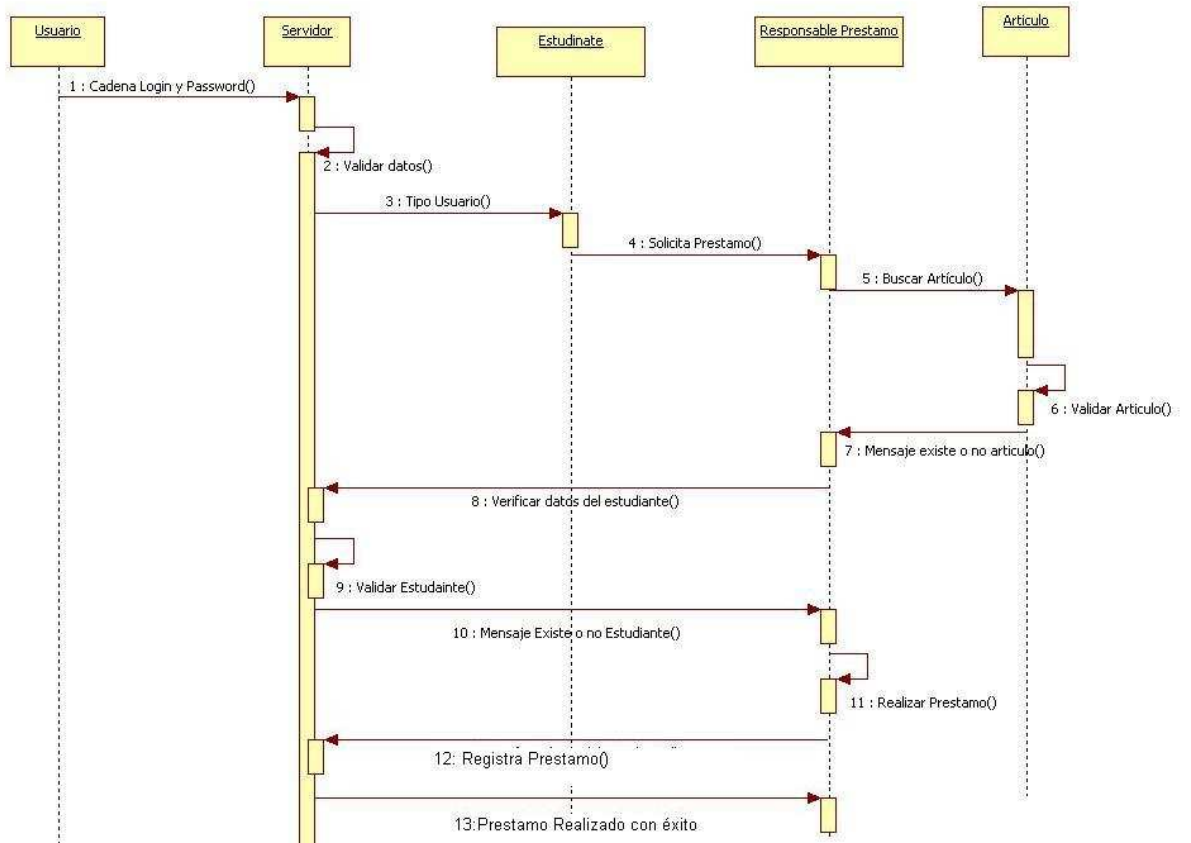
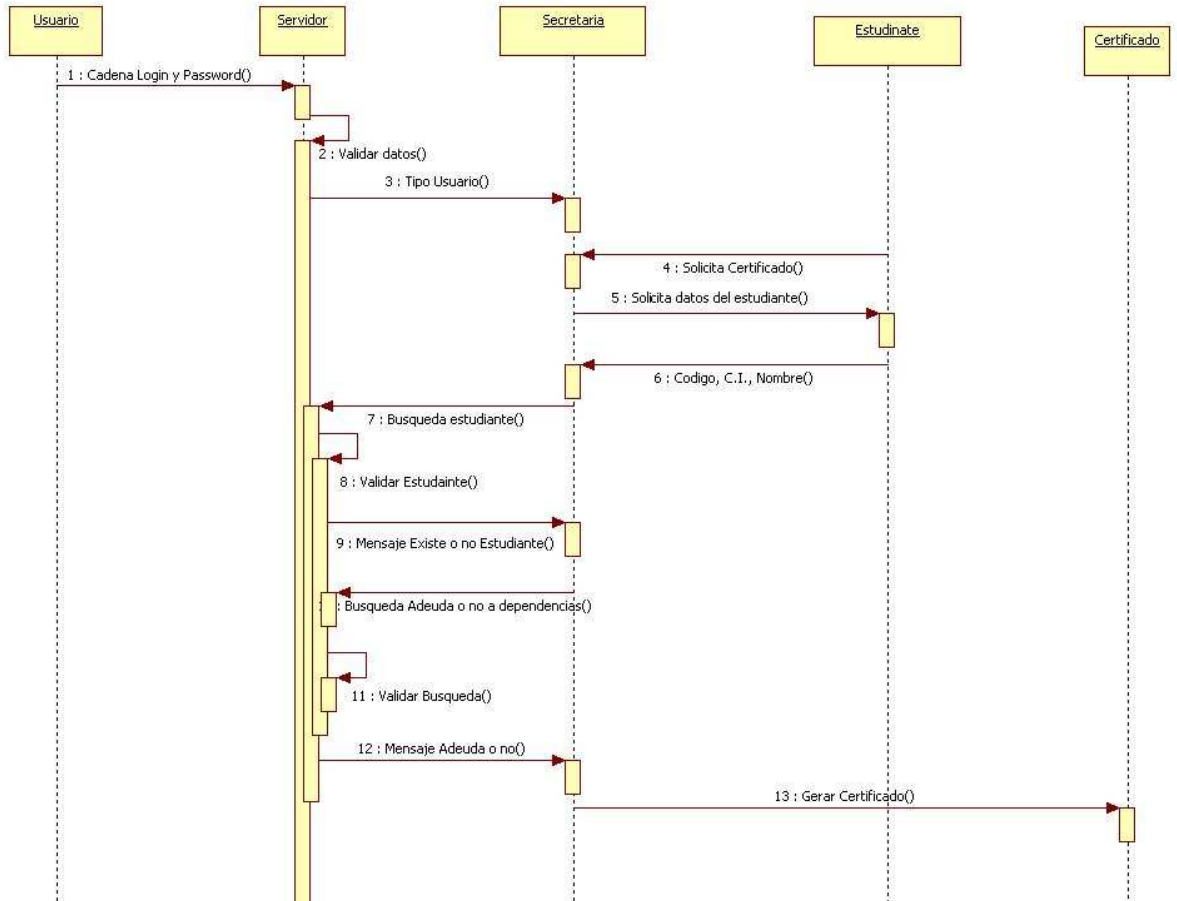
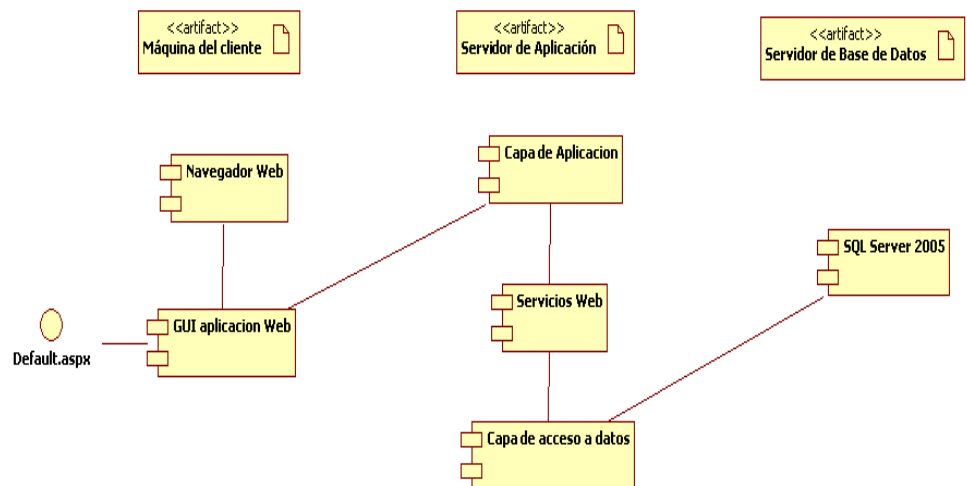


Diagrama de Secuencia para: Solicitar Certificado



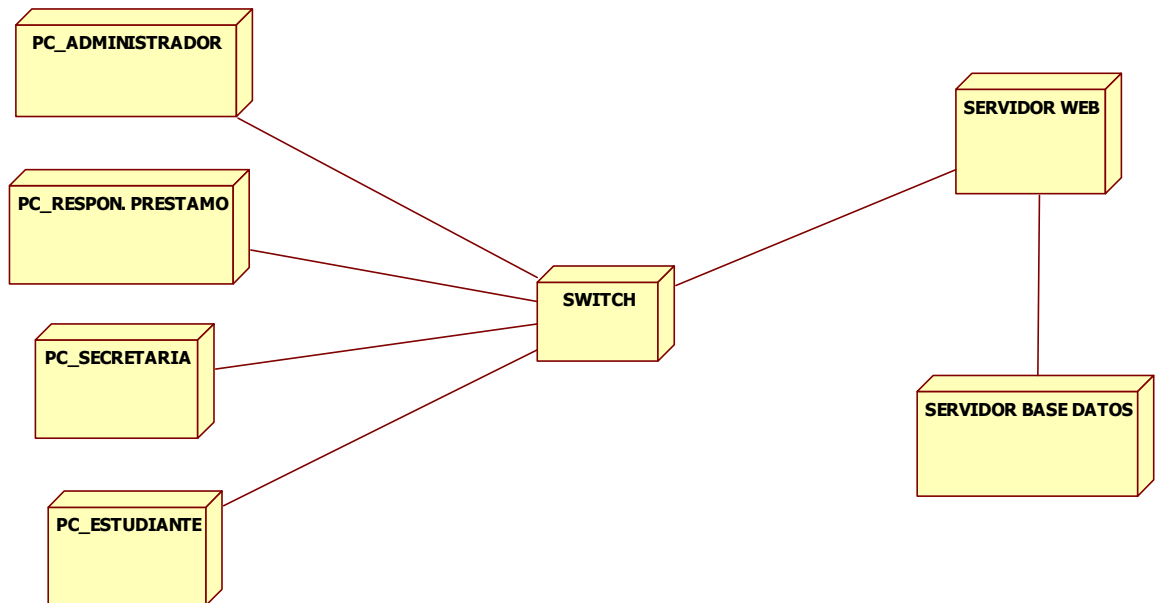
4.3.4.3 DIAGRAMA DE COMPONENTES

El Diagrama de Componentes describe el comportamiento de las clases en el sistema.



4.3.4.4 DIAGRAMAS DE NODOS

En este diagrama se representa los componentes hardware para el buen funcionamiento de la aplicación.



4.3.4.5 GLOSARIO DE TÉRMINOS

Interfaz: esquema a través de la cual el usuario puede comunicarse con la aplicación.

Estándar: normas aprobadas por un comité especializado de estandarización, para documentar procesos, actividades de desarrollo así como también para estandarizar nuestros documentos.

CASE: Ingeniería de Software asistida por computadora; software que se utiliza en cualquiera o en todas las fases de un sistema de información.

SECNA: Sistema de Emisión de certificados de no adeudar

Administrador: Encargado de la dependencia, usuario que puede

administrar ítems certificados y reportes.

Certificado: Documento que certifica que el Estudiante no Adeuda a ninguna dependencia de la ESPOCH, previo a la obtención del título académico.

Responsable del préstamo: Usuario que puede Administrar los ítems y realizar préstamos.

Backup: Copia de seguridad, copia de un programa informático, de un disco o de archivo de datos, realizada para archivar su contenido o para proteger archivos valiosos contra su pérdida en caso de que la copia activa se dañe o quede destruida.

Base de Datos: Conjunto de datos organizados para su almacenamiento en la memoria de un ordenador o computadora. La información se organiza en campos y registros, un campo se refiere a un tipo o atributo de información, y un registro, a toda la información sobre un individuo.

Botón: Es un objeto tangible que realiza un evento tras su activación.

Conexión: Comunicación entre dos entes que tienen características similares de comunicación.

DBA: Es aquella persona que tiene el control central del sistema de base de datos.

DBMS: Sistema manipulador de base de datos, esta diseñado para manejar solo cierto tipo predeterminado de estructura lógica.

SQL: Lenguaje de consulta estructurado, en informática, un lenguaje utilizado en bases de datos para consultar, actualizar y manejar bases de datos relacionales.

Tabla: Entidad que posee campos físicos primarios secundarios

RAD Studio: Herramienta de trabajo que hace posible el desarrollo rápido de aplicaciones (RAD). Es un entorno Integrado de desarrollo (IDE) que combina a varios lenguajes, asistentes y editores que facilitan el desarrollo.

4.3.4.6 ANÁLISIS DE RIESGOS

A través de la experiencia del Técnico Informático del Centro de Documentación, Ing. Eduardo Tenelanda y de las desarrolladoras de la tesis, se han podido identificar los siguientes riesgos y amenazas:

RIESGOS
R1: Incumplimiento de los objetivos
R2: Falta de requerimientos
R3: Falta de comunicación entre los miembros del equipo de trabajo
R4: Problemas de acceso a las bases de datos
R5: Insuficiente información para el desarrollo del sistema
R6: Interfaces mal diseñadas
R7: Insatisfacción de los usuarios
R8: Falta de disponibilidad del personal técnico

Categorización del Riesgo

RIESGOS	PROBABILIDAD			DEVERIDAD		
	ALTA	MEDIA	BAJA	ALTA	MEDIA	BAJA
R1			X	X		
R2		X			X	
R3		X			X	

R4	X			X		
R5		X		X		
R6		X			X	
R7			X	X		
R8		X			X	

Gestión del Riesgo

RIESGO	PLAN DE GESTION
R1	<p>Problema: Sistema no cumple con las expectativas ni con los objetivos planteados.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none"> • Seguir y evaluar continuamente el avance del proyecto. • Interactuar continuamente con los contratistas y los desarrolladores <p>Responsable:</p> <p>Srta. Elizabeth Orozco</p>
R2	<p>Problema: Retraso en la entrega del producto software final.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none"> • Planificar entregas parciales del trabajo desarrollado. • Controlar continuamente el avance del proyecto en base a los requerimientos planteados. <p>Responsable:</p> <p>Srta. Gabriela Cárdenas</p>
R3	<p>Problema: Incapacidad de llegar a consensos.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none"> • Evaluar las capacidades de liderazgo entre los miembros del equipo. • Designar a un miembro que lidere las actividades de todos los miembros. <p>Responsable:</p> <p>Srta. Elizabeth Orozco</p>
R4	<p>Problema: Al no tener las facilidades para acceder a la base de datos, no podremos interactuar a través del web services para la obtención de la información académica.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none"> • Gestionar los permisos necesarios para acceder a la base

	<p>de datos o a una réplica de la misma. Responsable: Srta. Gabriela Cárdenas</p>
R5	<p>Problema: Mal enfoque de la solución del problema</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none">• Planificar reuniones de trabajo para abstraer los verdaderos requerimientos del cliente. <p>Responsable: Srta. Elizabeth Orozco</p>
R6	<p>Problema: La aplicación no es usada.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none">• Integrar a los potenciales usuarios del sistema a la elaboración de interfaces. <p>Responsable: Srta. Elizabeth Orozco</p>
R7	<p>Problema: Desperdicio de los recursos empleados en una aplicación no utilizada.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none">• Identificar y evaluar permanentemente el cumplimiento de los requerimientos de los usuarios. <p>Responsable: Srta. Gabriela Cárdenas</p>
R8	<p>Problema: Atraso en los cronogramas establecidos y en la respectiva entrega de los avances.</p> <p>Actividades a desarrollar:</p> <ul style="list-style-type: none">• Establecer políticas internas de control.• Fomentar el espíritu de responsabilidad. <p>Responsable: Srta. Gabriela Cárdenas</p>

4.3.5 FASE 3: CODIFICACIÓN

CODIGO DE INICIO DE SESION PARA LOS TIPOS DE USUARIOS

unit login;

interface

uses

System.Collections, System.ComponentModel,

System.Data, System.Drawing, System.Web, System.Web.SessionState,

System.Web.UI, System.Web.UI.WebControls,

System.Web.UI.HtmlControls,

System.Web.Security, System.Web.UI.WebControls.WebParts,

System.Configuration,

System.Data.SqlClient;

type

TLogin = class(System.Web.UI.Page)

{ \$REGION 'Designer Managed Code' }

strict private

procedure InitializeComponent;

procedure btnIniciar_Sesion_Click(sender: TObject; e: System.EventArgs);

procedure Login1_Authenticate(sender: TObject; e:
System.Web.UI.WebControls.AuthenticateEventArgs);

{ \$ENDREGION }

strict private

procedure Page_Load(sender: System.Object; e: System.EventArgs);

strict protected

```
Content1: System.Web.UI.WebControls.Content;

btnIniciar_Sesion: System.Web.UI.WebControls.Button;

lblMensaje: System.Web.UI.WebControls.&Label;

txtContraseña: System.Web.UI.WebControls.TextBox;

txtLogin: System.Web.UI.WebControls.TextBox;

Login1: System.Web.UI.WebControls.Login;

protected

    procedure OnInit(e: EventArgs); override;

private

    { Private Declarations }

public

    { Public Declarations }

    procedure Autenticar (login:String; password:String ; var ced, nombre,
tipo,dep: String);

end;

implementation

uses RegistrarDependencia;

{$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify
```

```
/// the contents of this method with the code editor.
```

```
/// </summary>
```

```
procedure TLogin.InitializeComponent;
```

```
begin
```

```
    Include(Self.btnIniciar_Sesion.Click, Self.btnIniciar_Sesion_Click);
```

```
    Include(Self.Login1.Authenticate, Self.Login1_Authenticate);
```

```
    Include(Self.Load, Self.Page_Load);
```

```
end;
```

```
{$ENDREGION}
```

```
procedure TLogin.Page_Load(sender: System.Object; e: System.EventArgs);
```

```
begin
```

```
    // TODO: Put user code to initialize the page here
```

```
end;
```

```
procedure TLogin.OnInit(e: EventArgs);
```

```
begin
```

```
    InitializeComponent;
```

```
    inherited OnInit(e);
```

```
end;
```

```
procedure TLogin.Autenticar(login, password: String; var ced, nombre, tipo,  
dep: String);
```



```
var

conn:string;

objConn: SqlConnection;

objComand: SqlCommand;

pLogin,pclave:SqlParameter; //entrada

pnombre, pcedula, ptipo, pdep:SqlParameter;//salida

begin

    conn:='user id=sa;password="123";data source="ELIZABET-
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';

    objConn:= sqlConnection.Create;

    objConn.ConnectionString:=conn;

    objComand:= SqlCommand.Create;

    objComand.Connection:=objConn;

    objComand.CommandText:='spLogin';

    objComand.CommandType := CommandType.StoredProcedure;

    pLogin := SqlParameter.Create;

    pLogin.set_ParameterName('@Login');

    pLogin.set_DbType(DbType.String);

    pLogin.set_Size(50);

    pLogin.Value:=login;
```

```
objComand.Parameters.Add(pLogin);

pclave := SqlParameter.Create;

pclave.set_ParameterName('@Password');

pclave.set_DbType(DbType.String);

pclave.set_Size(50);

pclave.Value:=password;

objComand.Parameters.Add(pclave);

pcedula := SqlParameter.Create;

pcedula.set_ParameterName('@Cedula');

pcedula.set_DbType(DbType.String);

pcedula.set_Size(11);

pcedula.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pcedula);

pnombre := SqlParameter.Create;

pnombre.set_ParameterName('@Nombre');

pnombre.set_DbType(DbType.String);

pnombre.set_Size(50);

pnombre.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pnombre);

ptipo := SqlParameter.Create;
```

```
    ptipo.set_ParameterName('@Tipo');

ptipo.set_DbType(DbType.String);

    ptipo.set_Size(25);

    ptipo.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(ptipo);

pdep := SqlParameter.Create;

pdep.set_ParameterName('@Dep');

pdep.set_DbType(DbType.String);

pdep.set_Size(25);

pdep.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pdep);

try

    objConn.Open;

    objComand.ExecuteNonQuery;

    nombre:=(pnombre.Value).ToString;

    ced:=(pcedula.Value).ToString;

    tipo:=(ptipo.Value).ToString;

    dep:=(pdep.Value).ToString;

finally

    if (objConn.State = ConnectionState.Open) then
```

```
        objConn.Close;

    end;

end;

procedure TLogin.Login1_Authenticate(sender: TObject; e:
System.Web.UI.WebControls.AuthenticateEventArgs);

var

login,clave:string;

ced,nombre,tipo,dep:string;

begin

login := Login1.UserName;

    clave := login1.Password;

    Autenticar(login,clave,ced,nombre,tipo,dep);

if tipo='ADMINISTRADOR' then

    begin

        Response.Cookies['userName'].Value := nombre;

        Response.Cookies['userName'].Expires :=
DateTime.Now.AddMinutes(1);

        Response.Cookies['tipoUser'].Value := tipo;

        Response.Cookies['tipoUser'].Expires :=
DateTime.Now.AddMinutes(1);

        Session.Add('login', login);
```

```
FormsAuthentication.RedirectFromLoginPage(ced,False);

Response.Redirect('Administrador.aspx');

end

else

    if tipo='RESPONSABLE' then

        begin

            Response.Cookies['userName'].Value := nombre;

            Response.Cookies['userName'].Expires :=
DateTime.Now.AddMinutes(30);

            Response.Cookies['tipoUser'].Value := tipo;

            Response.Cookies['tipoUser'].Expires :=
DateTime.Now.AddMinutes(30);

            Session.Add('login', login);

            FormsAuthentication.RedirectFromLoginPage(ced,False);

            Response.Redirect('Responsable.aspx');

        end

    else

        if tipo = 'SECRETARIA' then

            begin

                Response.Cookies['userName'].Value := nombre;
```

```
        Response.Cookies['userName'].Expires :=  
DateTime.Now.AddMinutes(30);  
  
        Response.Cookies['tipoUser'].Value := tipo;  
  
        Response.Cookies['tipoUser'].Expires :=  
DateTime.Now.AddMinutes(30);  
  
        Response.Cookies['ciUser'].Value := ced;  
  
        Response.Cookies['ciUser'].Expires :=  
DateTime.Now.AddMinutes(30);  
  
        Session.Add('login', login);  
  
        FormsAuthentication.RedirectFromLoginPage(ced,False);  
  
        Response.Redirect('Secretaria.aspx');  
  
    end;  
  
end;  
  
end.
```

CODIGO DE INICIO DE SESION DEL ESTUDIANTE

```
unit LoginEst;  
  
interface  
  
uses  
  
    System.Collections, System.ComponentModel,  
  
    System.Data, System.Drawing, System.Web, System.Web.SessionState,
```

```
System.Web.UI, System.Web.UI.WebControls,  
System.Web.UI.HtmlControls,  
  
System.Web.Security, System.Web.UI.WebControls.WebParts,  
System.Configuration,  
  
System.Data.SqlClient;  
  
type  
  
TLoginEst = class(System.Web.UI.Page)  
  
{$REGION 'Designer Managed Code'}  
  
strict private  
  
    procedure InitializeComponent;  
  
    procedure Login1_Authenticate(sender: TObject; e:  
System.Web.UI.WebControls.AuthenticateEventArgs);  
  
{$ENDREGION}  
  
strict private  
  
    procedure Page_Load(sender: System.Object; e: System.EventArgs);  
  
strict protected  
  
    Content1: System.Web.UI.WebControls.Content;  
  
    Content2: System.Web.UI.WebControls.Content;  
  
    Login1: System.Web.UI.WebControls.Login;  
  
protected  
  
    procedure OnInit(e: EventArgs); override;
```

private

{ Private Declarations }

public

{ Public Declarations }

procedure AutenticarEst(login, password: String; var nombre, apellido:
String);

procedure ConcatenarCl(password: String; var concatpw: String);

end;

implementation

//variables globales sacar datos del est al autenticar

{\$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify

/// the contents of this method with the code editor.

/// </summary>

{\$ENDREGION}

procedure TLoginEst.InitializeComponent;

begin

Include(Self.Login1.Authenticate, Self.Login1_Authenticate);

Include(Self.Load, Self.Page_Load);


```
end;

procedure TLoginEst.OnInit(e: EventArgs);

begin

    // Required for Designer support

    InitializeComponent;

    inherited OnInit(e);

end;

procedure TLoginEst.Page_Load(sender: System.Object; e:
System.EventArgs);

begin

    // TODO: Put user code to initialize the page here

end;

procedure TLoginEst.AutenticarEst(login, password: String; var nombre,
apellido: String);

var

    conn:string;

    objConn: SqlConnection;

    objComand: SqlCommand;

    pLogin,pclave:SqlParameter; //entrada

    pnombre, papellido:SqlParameter;//salida
```

begin

```
conn:='user id=sa;password="123";data source="ELIZABET-  
62FA1C";persist security info=False;initial catalog=BDEIS_ACADEMICO';
```

```
objConn:= sqlConnection.Create;
```

```
objConn.ConnectionString:=conn;
```

```
objComand:= SqlCommand.Create;
```

```
objComand.Connection:=objConn;
```

```
objComand.CommandText:='sploginEst';
```

```
objComand.CommandType := CommandType.StoredProcedure;
```

```
pLogin := SqlParameter.Create;
```

```
pLogin.set_ParameterName('@ced');
```

```
pLogin.set_DbType(DbType.String);
```

```
pLogin.set_Size(11);
```

```
pLogin.Value:=login;
```

```
objComand.Parameters.Add(pLogin);
```

```
pclave := SqlParameter.Create;
```

```
pclave.set_ParameterName('@cedpw');
```

```
pclave.set_DbType(DbType.String);
```

```
pclave.set_Size(11);
```

```
pclave.Value:=password;
```

```
objComand.Parameters.Add(pclave);

    pnombre := SqlParameter.Create;

    pnombre.set_ParameterName('@nom');

    pnombre.set_DbType(DbType.String);

    pnombre.set_Size(50);

    pnombre.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pnombre);

    papellido := SqlParameter.Create;

    papellido.set_ParameterName('@ape');

papellido.set_DbType(DbType.String);

    papellido.set_Size(50);

    papellido.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(papellido);

try

    objConn.Open;

    objComand.ExecuteNonQuery;

finally

    if (objConn.State = ConnectionState.Open) then

objConn.Close;

end;
```

```
nombre:=(pnombre.Value).ToString;

apellido:=(papellido.Value).ToString;

end;

procedure TLoginEst.ConcatenarCI(password: String; var concatpw: String);

var

    ced:string;

    aux1,aux2,aux3:string;

begin

    aux1:=Copy(password,1,9);

    aux2:=Copy(password,10,1);

    aux3:='-';

    ced:=Concat(aux1,aux3,aux2);

concatpw:=ced;

end;

procedure TLoginEst.Login1_Authenticate(sender: TObject; e:
System.Web.UI.WebControls.AuthenticateEventArgs);

var

    log,pw,nom,ape:string;

    concatpw:string;

begin
```

```
log := Login1.UserName;

pw := login1.Password;

ConcatenarCI(pw,concatpw);

AutenticarEst(log,concatpw,nom,ape);

if ((nom<>"") and (ape<>"")) then

begin

    Response.Cookies['userName'].Value := nom;

    Response.Cookies['userName'].Expires :=
DateTime.Now.AddMinutes(10);

    Response.Cookies['apeUser'].Value := ape;

    Response.Cookies['apeUser'].Expires := DateTime.Now.AddMinutes(10);

    Response.Cookies['ciUser'].Value := log;

    Response.Cookies['ciUser'].Expires := DateTime.Now.AddMinutes(10);

    Session.Add('login', log);

    FormsAuthentication.RedirectFromLoginPage(log,False);

    Response.Redirect('Estudiante.aspx');

end;

end;

end.
```

CÓDIGO DE REGISTRAR MODIFICAR Y ELIMINAR USUARIO

```
unit RegistrarDependencia;  
  
interface  
  
uses  
  
    System.Collections, System.ComponentModel,  
  
    System.Data, System.Drawing, System.Web, System.Web.SessionState,  
  
    System.Web.UI, System.Web.UI.WebControls,  
System.Web.UI.HtmlControls,  
  
    System.Web.Security, System.Web.UI.WebControls.WebParts,  
System.Configuration,  
  
Eco.Web.UI.WebControls,  
  
EcoSpaceUnit,  
  
Package_1Unit,  
  
Eco.ObjectRepresentation, Eco.Handles;  
  
type  
  
    TRegistrarDependencia = class(System.Web.UI.Page)  
  
        {$REGION 'Designer Managed Code'}  
  
        strict private  
  
        procedure InitializeComponent;  
  
        procedure btnNuevaDependencia_click(sender: TObject; e:  
System.EventArgs);
```

{ \$ENDREGION }

strict private

procedure Page_Load(sender: System.Object; e: System.EventArgs);

strict protected

Content1: System.Web.UI.WebControls.Content;

Content2: System.Web.UI.WebControls.Content;

EcoSpaceManager1: Eco.Web.UI.WebControls.EcoSpaceManager;

dsDependencia: Eco.Web.UI.WebControls.EcoDataSource;

GridView1: System.Web.UI.WebControls.GridView;

btnNuevaDependencia: System.Web.UI.WebControls.Button;

function get_EcoSpace: TESISecoSpace;

property EcoSpace: TESISecoSpace read get_EcoSpace;

protected

procedure OnInit(e: EventArgs); override;

private

{ Private Declarations }

public

{ Public Declarations }

end;

implementation

uses

Eco.Services, TDependenciaUnit;

{\$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify

/// the contents of this method with the code editor.

/// </summary>

function TRegistrarDependencia.get_EcoSpace: TESISecoSpace;

begin

Result := TESISecoSpace(EcoSpaceManager1.EcoSpace);

end;

procedure TRegistrarDependencia.InitializeComponent;

begin

Include(Self.btnNuevaDependencia.Click, Self.btnNuevaDependencia_click);

Include(Self.Load, Self.Page_Load);

end;

{\$ENDREGION}

procedure TRegistrarDependencia.Page_Load(sender: System.Object; e:

System.EventArgs);

begin


```
// TODO: Put user code to initialize the page here

end;

procedure TRegistrarDependencia.OnInit(e: EventArgs);

begin

    // Required for Designer support

    InitializeComponent;

    inherited OnInit(e);

end;

procedure TRegistrarDependencia.btnNuevaDependencia_click(sender:
TObject; e: System.EventArgs);

var

d: TDependencia;

begin

d:= TDependencia.Create(EcoSpace);

d.Cod_Dependencia:='Ingrese el codigo';

d.Nom_Dependencia:='Ingrese el Nombre';

d.Ubi_Dependencia:='La Ubicacion';

d.Des_Dependencia:='Descripcion';

EcoSpaceManager1.UpdateDatabase;

DataBind;
```

end;

end.

CÓDIGO DE REGISTRAR MODIFICAR Y ELIMINAR DEPENDENCIAS

```
unit RegistrarDependencia;
```

```
interface
```

```
uses
```

```
    System.Collections, System.ComponentModel,
```

```
    System.Data, System.Drawing, System.Web, System.Web.SessionState,
```

```
    System.Web.UI, System.Web.UI.WebControls,
```

```
System.Web.UI.HtmlControls,
```

```
    System.Web.Security, System.Web.UI.WebControls.WebParts,
```

```
System.Configuration,
```

```
    Eco.Web.UI.WebControls,
```

```
    EcoSpaceUnit,
```

```
    Package_1Unit,
```

```
Eco.ObjectRepresentation, Eco.Handles;
```

```
type
```

```
    TRegistrarDependencia = class(System.Web.UI.Page)
```

```
    {$REGION 'Designer Managed Code'}
```

```
    strict private
```

```
procedure InitializeComponent;

procedure btnNuevaDependencia_click(sender: TObject; e:
System.EventArgs);

{$ENDREGION}

strict private

procedure Page_Load(sender: System.Object; e: System.EventArgs);

strict protected

Content1: System.Web.UI.WebControls.Content;

Content2: System.Web.UI.WebControls.Content;

EcoSpaceManager1: Eco.Web.UI.WebControls.EcoSpaceManager;

dsDependencia: Eco.Web.UI.WebControls.EcoDataSource;

GridView1: System.Web.UI.WebControls.GridView;

btnNuevaDependencia: System.Web.UI.WebControls.Button;

function get_EcoSpace: TESISecoSpace;

property EcoSpace: TESISecoSpace read get_EcoSpace;

protected

procedure OnInit(e: EventArgs); override;

private

{ Private Declarations }

public
```

```
{ Public Declarations }

end;

implementation

uses

    Eco.Services, TDependenciaUnit;

{$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify

/// the contents of this method with the code editor.

/// </summary>

function TRegistrarDependencia.get_EcoSpace: TESISEcoSpace;

begin

    Result := TESISEcoSpace(EcoSpaceManager1.EcoSpace);

end;

procedure TRegistrarDependencia.InitializeComponent;

begin

    Include(Self.btnNuevaDependencia.Click, Self.btnNuevaDependencia_click);

    Include(Self.Load, Self.Page_Load);

end;

{$ENDREGION}
```

```
procedure TRegistrarDependencia.Page_Load(sender: System.Object; e:  
System.EventArgs);
```

```
begin
```

```
    // TODO: Put user code to initialize the page here
```

```
end;
```

```
procedure TRegistrarDependencia.OnInit(e: EventArgs);
```

```
begin
```

```
    // Required for Designer support
```

```
    InitializeComponent;
```

```
    inherited OnInit(e);
```

```
end;
```

```
procedure TRegistrarDependencia.btnNuevaDependencia_click(sender:  
TObject; e: System.EventArgs);
```

```
var
```

```
d: TDependencia;
```

```
begin
```

```
d:= TDependencia.Create(EcoSpace);
```

```
d.Cod_Dependencia:='Ingrese el codigo';
```

```
d.Nom_Dependencia:='Ingrese el Nombre';
```

```
d.Ubi_Dependencia:='La Ubicacion';
```

```
d.Des_Dependencia:='Descripcion';
```

```
EcoSpaceManager1.UpdateDatabase;
```

```
DataBind;
```

```
end;
```

```
end.
```

CODIGO PARA GENERAR EL REPORTE DE EMITIR CERTIFICADO

```
unit Reporte_Certificado;
```

```
interface
```

```
uses
```

```
    System.Collections, System.ComponentModel, System.IO,
```

```
    System.Data, System.Drawing, System.Web, System.Web.SessionState,
```

```
    System.Web.UI, System.Web.UI.WebControls,
```

```
    System.Web.UI.HtmlControls,
```

```
    System.Web.Security, System.Web.UI.WebControls.WebParts,
```

```
    System.Configuration,
```

```
    iTextSharp.text, iTextSharp.text.pdf, Root.Reports, System.Data.SqlClient;
```

```
type
```

```
    TReportes = class(System.Web.UI.Page)
```

```
    {$REGION 'Designer Managed Code'}
```

```
    strict private
```

```
        procedure InitializeComponent;
```

```
procedure btnImprimirCertificado_Click(sender: TObject; e:
System.EventArgs);

{$ENDREGION}

strict private

procedure Page_Load(sender: System.Object; e: System.EventArgs);

strict protected

Content1: System.Web.UI.WebControls.Content;

txtCIEstudiante: System.Web.UI.WebControls.TextBox;

btnImprimirCertificado: System.Web.UI.WebControls.Button;

lblMensaje: System.Web.UI.WebControls.Label;

protected

procedure OnInit(e: EventArgs); override;

private

{ Private Declarations }

public

{ Public Declarations }

procedure ConcatenarCI(bci: String; var concatci: String);

procedure BuscarCertificado(ci,cis:string; var ban:integer);

procedure BuscarEstudiante(ci:string; var nombre,apellido:String);

procedure InsertarCertificado(fecha:datetime; ciUS,ciEt:string);
```

```
//procedure GenerateReport;

//Public Function ExportToPDF( rpt : ReportDocument, ByVal
Dependencias : String) :String;

end;

implementation

{$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify

/// the contents of this method with the code editor.

/// </summary>

procedure TReportes.InitializeComponent;

begin

    Include(Self.btnImprimirCertificado.Click, Self.btnImprimirCertificado_Click);

    Include(Self.Load, Self.Page_Load);

end;

{$ENDREGION}

procedure TReportes.Page_Load(sender: System.Object; e:
System.EventArgs);

begin

    // TODO: Put user code to initialize the page here

end;
```



```
procedure TReportes.OnInit(e: EventArgs);

begin

    // Required for Designer support

    InitializeComponent;

    inherited OnInit(e);

end;

procedure TReportes.btnImprimirCertificado_Click(sender: TObject; e:
System.EventArgs);

var

    bci: String;

    cib:String;

    doc:Document;

    f, f1,f2:Font;

    bs:BaseFont;

    jpg,img0 : Image;

    p: Paragraph;

    ban: Integer;

    nombre,apellido, cedula:String;

begin

    bci:= txtCIEstudiente.Text;
```

```
ConcatenarCI(bci,cib);

BuscarCertificado(bci,cib,ban);

if Request.Cookies['userName'] <> Nil then

begin

// lblNameEst.Text :=

Concat(Server.HtmlEncode(Request.Cookies['userName'].Value),

// ' ',Server.HtmlEncode(Request.Cookies['apeUser'].Value));

cedula:= Server.HtmlEncode(Request.Cookies['ciUser'].Value);

if ban= 1 then

begin

BuscarEstudiante(bci,nombre,apellido);

doc:=Document.Create;

PdfWriter.GetInstance(doc,

FileStream.Create(Request.PhysicalApplicationPath +

'\1.pdf', FileMode.Create));

doc.Open;

jpg:= iTextSharp.text.Image.GetInstance(

'C:\inetpub\wwwroot\TESIS_PAG\imagenes\sello_espoch.jpg' );

jpg.SetAbsolutePosition(30,675);

jpg.Alignment:= iTextSharp.text.Image.ALIGN_MIDDLE;

InsertarCertificado(DateTime.Now,bci,cedula);
```

```
doc.Add(jpg);

f := FontFactory.GetFont(FontFactory.HELVETICA, 20,
iTextSharp.text.Font.NORMAL);

f1 := FontFactory.GetFont(FontFactory.HELVETICA, 15,
iTextSharp.text.Font.NORMAL);

f2 := FontFactory.GetFont(FontFactory.HELVETICA, 12,
iTextSharp.text.Font.NORMAL);

p:= Paragraph.Create();

p.Alignment := Element.ALIGN_CENTER;

doc.Add(Paragraph.Create('          ESPOCH',f));

doc.SetMargins(10,10,15,20);

doc.Add(Paragraph.Create('          ESCUELA SUPERIOR
POLITECNICA DE CHIMBORAZO',f1));

doc.Add(Paragraph.Create('          FACULTAD DE
INFORMATICA Y ELECTRÓNICA',f1));

doc.Add(Paragraph.Create('          ESCUELA DE
INGENIERIA EN SISTEMAS',f1));

doc.Add(Paragraph.Create('          Telefax 605609 ext
214 Riobamba-Ecuador',f2));

doc.Add(Paragraph.Create('
_____
,f1));
```

```
doc.Add(
Chunk.Create("",FontFactory.GetFont(FontFactory.COURIER,'1'))) ;

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('CERTIFICADO',f));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('A petición verbal de la parte interesada:',f2));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('Certifico que:',f1));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('El (la) mencionado (a) señor (ita). '+nombre+'
'+apellido+', estudiante de la escuela de Ingeniería en sistemas de la
Escuela Superior Politecnica de Chimborazo, no adeuda valor ni
documento alguno a la institución.',f2));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('Es todo cuanto puedo certificar, conforme
así lo verifican los archivos correspondientes, autorizando al interesado
hacer uso del presente documento en los tramites de Auditoria y/o
Graduación.',f2));

doc.Add(Paragraph.Create('
'));
```

```
doc.Add(Paragraph.Create('
Riobamba: ' + DateTime.Now.ToLongDateString ,f2));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create('Lo Certifico:',f1));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create('
Lcda.
Janeth Mendez',f2));

doc.Add(Paragraph.Create('
SECRETARIA
ESCUELA',f1));

doc.Add(Paragraph.Create('
INGENIERIA
SISTEMAS',f1));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create('
));

doc.Add(Paragraph.Create(' Ing. Cesar Villa
Ing. Rosa Ortega',f2));

doc.Add(Paragraph.Create('DIRECTOR FINANCIERO
TESORERO ESPOCH',f1));

doc.Close;

Response.Redirect('~/.pdf');
```

```
        doc.Free;

    end

else

    if ban=0 then

        begin

            lblMensaje.Visible:= True;

            lblMensaje.Text:= 'El Estudiante Mantiene Prestamos';

        end;

    end

else

    Response.Redirect('Default.aspx');

end;

procedure TReportes.BuscarCertificado(ci,cis:string; var ban:integer);

var

    conn:string;

    objConn: SqlConnection;

    objComand: SqlCommand;

    pced, pcesisbic, pbandera:SqlParameter; //entrada

begin
```

```
conn:='user id=sa;password="123";data source="ELIZABET-  
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';
```

```
objConn:= sqlConnection.Create;
```

```
objConn.ConnectionString:=conn;
```

```
objComand:= SqlCommand.Create;
```

```
objComand.Connection:=objConn;
```

```
objComand.CommandText:='spCertificado';
```

```
objComand.CommandType := CommandType.StoredProcedure;
```

```
pced := SqlParameter.Create;
```

```
pced.set_ParameterName('@ci');
```

```
pced.set_DbType(DbType.String);
```

```
pced.set_Size(11);
```

```
pced.Value:=ci.ToString;
```

```
objComand.Parameters.Add(pced);
```

```
pcesisbic := SqlParameter.Create;
```

```
pcesisbic.set_ParameterName('@cib');
```

```
pcesisbic.set_DbType(DbType.String);
```

```
pcesisbic.set_Size(11);
```

```
pcesisbic.Value:=cis.ToString;
```

```
objComand.Parameters.Add(pcesisbic);
```

```
    pbandera := SqlParameter.Create;

    pbandera.set_ParameterName('@Bandera');

    pbandera.set_DbType(DbType.Int32);

pbandera.set_Size(11);

    pbandera.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pbandera);

    try

        objConn.Open;

        objComand.ExecuteNonQuery;

        ban:= Convert.ToInt32(pbandera.Value );

    finally

        if (objConn.State = ConnectionState.Open) then

            objConn.Close;

        end;

    end;

end;

procedure TReportes.BuscarEstudiante(ci:string; var nombre,apellido:String);

var

    conn:string;

    objConn: SqlConnection;

    objComand: SqlCommand;
```



```
pced, pnombre, papellido:SqlParameter;
```

```
begin
```

```
    conn:='user id=sa;password="123";data source="ELIZABET-  
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';
```

```
    objConn:= sqlConnection.Create;
```

```
    objConn.ConnectionString:=conn;
```

```
    objComand:= SqlCommand.Create;
```

```
    objComand.Connection:=objConn;
```

```
    objComand.CommandText:='spBuscarEstudAcad';
```

```
    objComand.CommandType := CommandType.StoredProcedure;
```

```
    pced := SqlParameter.Create;
```

```
    pced.set_ParameterName('@bcd');
```

```
    pced.set_DbType(DbType.String);
```

```
    pced.set_Size(11);
```

```
    pced.Value:=ci.ToString;
```

```
    objComand.Parameters.Add(pced);
```

```
    pnombre := SqlParameter.Create;
```

```
    pnombre.set_ParameterName('@nombre');
```

```
    pnombre.set_DbType(DbType.String);
```

```
    pnombre.set_Size(50);
```

```
    pnombre.set_Direction(ParameterDirection.Output);

objComand.Parameters.Add(pnombre);

    papellido := SqlParameter.Create;

    papellido.set_ParameterName('@apellido');

papellido.set_DbType(DbType.String);

    papellido.set_Size(50);

    papellido.set_Direction(ParameterDirection.Output);

    objComand.Parameters.Add(papellido);

try

    objConn.Open;

objComand.ExecuteNonQuery;

    nombre:=(pnombre.Value).ToString;

    apellido:=(papellido.Value).ToString;

finally

    if (objConn.State = ConnectionState.Open) then

objConn.Close;

    end;

end;

procedure TReportes.ConcatenarCI(bci: String; var concatci: String);

var
```

```
ced:string;

aux1,aux2:string;

begin

    aux1:=Copy(bci,1,9);

    aux2:=Copy(bci,11,1);

    ced:=Concat(aux1,aux2);

    concatci:=ced;

end;

procedure TReportes.InsertarCertificado(fecha:datetime; ciUS,ciEt:string);

var

    conn:string;

    objConn: SqlConnection;

    objComand: SqlCommand;

    pfecha, pcius, pciest:SqlParameter;

begin

    conn:='user id=sa;password="123";data source="ELIZABET-62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';

    objConn:= sqlConnection.Create;

    objConn.ConnectionString:=conn;
```

```
objComand:= SqlCommand.Create;

objComand.Connection:=objConn;

objComand.CommandText:='spInsertCertificado';

objComand.CommandType := CommandType.StoredProcedure;

pfecha := SqlParameter.Create;

pfecha.set_ParameterName('@dtFechaEmision');

pfecha.set_DbType(DbType.DateTime);

pfecha.set_Size(6);

pfecha.Value:=fecha.Date;

objComand.Parameters.Add(pfecha);

pcius := SqlParameter.Create;

pcius.set_ParameterName('@strCedulaUs');

pcius.set_DbType(DbType.String);

pcius.set_Size(11);

pcius.Value:=ciUS.ToString;

objComand.Parameters.Add(pcius);

pciest := SqlParameter.Create;

pciest.set_ParameterName('@strciEst');

pciest.set_DbType(DbType.String);

pciest.set_Size(11);
```

```
    pciest.Value:=ciEt.ToString;

    objComand.Parameters.Add(pciest);

    try

        objConn.Open;

objComand.ExecuteNonQuery;

    finally

        if (objConn.State = ConnectionState.Open) then

            objConn.Close;

        end;

    end;

end;

end.
```

CODIGO PARA GENERAR REPORTES DE PRESTAMOS POR ESTUDIANTES

```
unit PrestamosRealizados;

interface

uses

    System.Collections, System.ComponentModel, System.IO,

    System.Data, System.Drawing, System.Web, System.Web.SessionState,

    System.Web.UI, System.Web.UI.WebControls,

    System.Web.UI.HtmlControls,
```

System.Web.Security, System.Web.UI.WebControls.WebParts,
System.Configuration,

iTextSharp.text, iTextSharp.text.pdf, System.Data.SqlClient;

type

TPrestamosRealizados = class(System.Web.UI.Page)

{ \$REGION 'Designer Managed Code' }

strict private

procedure InitializeComponent;

{ \$ENDREGION }

strict private

procedure Page_Load(sender: System.Object; e: System.EventArgs);

strict protected

Content1: System.Web.UI.WebControls.Content;

lblNameEst: System.Web.UI.WebControls.&Label;

protected

procedure OnInit(e: EventArgs); override;

private

{ Private Declarations }

public

{ Public Declarations }

```
procedure CedulaUsuarios(nombre,apellido: String; var cedula: String);

procedure PrestamoSISBIC(cedula: String; var cantidad: Integer);

end;

implementation

{$REGION 'Designer Managed Code'}

/// <summary>

/// Required method for Designer support -- do not modify

/// the contents of this method with the code editor.

/// </summary>

procedure TPrestamosRealizados.InitializeComponent;

begin

    Include(Self.Load, Self.Page_Load);

end;

{$ENDREGION}

procedure TPrestamosRealizados.Page_Load(sender: System.Object; e:
System.EventArgs);

var

    doc:Document;

    f, f1,f2,f3:Font;

    fuente:BaseFont;
```

```
jpg: Image;

p: Paragraph;

pdf: PdfWriter;

leer: SqlDataReader;

conexion:string;

objConexion: SqlConnection;

dsprestamos:DataSet;

daPrestamo: SqlDataAdapter;

objcomando: SqlCommand;

cadsql: String;

nombre,apellido, cedula, estbib:string;

pcedula: SqlParameter;

i,cantidad: Integer;

begin

if Request.Cookies['userName'] <> Nil then

begin

lblNameEst.Text :=

Concat(Server.HtmlEncode(Request.Cookies['userName'].Value),

'',Server.HtmlEncode(Request.Cookies['apeUser'].Value));

nombre:= Server.HtmlEncode(Request.Cookies['userName'].Value);
```



```
apellido:=Server.HtmlEncode(Request.Cookies['apeUser'].Value);
```

```
CedulaUsuarios(nombre,apellido,cedula);
```

```
conexion:='user id=sa;password="123";data source="ELIZABET-  
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';
```

```
objConexion := SqlConnection.Create(conexion);
```

```
Try
```

```
objcomando:= SqlCommand.Create;
```

```
objcomando.Connection:= objConexion;
```

```
objcomando.CommandText:= 'spPrestamosECOTESIS';
```

```
objcomando.CommandType:= CommandType.StoredProcedure;
```

```
pcedula:= SqlParameter.Create;
```

```
pcedula.set_ParameterName('@cedula');
```

```
pcedula.set_DbType(DbType.String);
```

```
pcedula.set_Size(11);
```

```
pcedula.Value:= cedula.ToString;
```

```
objcomando.Parameters.Add(pcedula);
```

```
objConexion.Open();
```

```
daPrestamo:=SqlDataAdapter.Create;
```

```
dsprestamos:=DataSet.Create;
```

```
daPrestamo.SelectCommand := objcomando;
```

```
daPrestamo.Fill(dsprestamos);

daPrestamo.Dispose;

objcomando.Dispose;

objConexion.Close;

doc:=Document.Create(PageSize.LEGAL);

PdfWriter.GetInstance(doc,
FileStream.Create(Request.PhysicalApplicationPath +
'\3.pdf', FileMode.Create));

doc.Open;

jpg:= iTextSharp.text.Image.GetInstance(
'C:\inetpub\wwwroot\TESIS_PAG\imagenes\sello_esPOCH.jpg' );

jpg.SetAbsolutePosition(30,850);

jpg.Alignment:= iTextSharp.text.Image.ALIGN_MIDDLE;

doc.Add(jpg);

f := FontFactory.GetFont(FontFactory.HELVETICA, 20,
iTextSharp.text.Font.NORMAL);

f1 := FontFactory.GetFont(FontFactory.HELVETICA, 15,
iTextSharp.text.Font.NORMAL);

f2 := FontFactory.GetFont(FontFactory.HELVETICA, 12,
iTextSharp.text.Font.NORMAL);

f3 := FontFactory.GetFont(FontFactory.HELVETICA, 10,
iTextSharp.text.Font.NORMAL);
```

```
p:= Paragraph.Create();

p.Alignment := Element.ALIGN_CENTER;

doc.Add(Paragraph.Create('          ESPOCH',f));

doc.SetMargins(10,10,15,20);

doc.Add(Paragraph.Create('          ESCUELA SUPERIOR
POLITECNICA DE CHIMBORAZO',f1));

doc.Add(Paragraph.Create('          FACULTAD DE
INFORMATICA Y ELECTRÓNICA',f1));

doc.Add(Paragraph.Create('          ESCUELA DE
INGENIERIA EN SISTEMAS',f1));

doc.Add(Paragraph.Create('          Telefax 605609
ext 214 Riobamba-Ecuador',f2));

doc.Add(Paragraph.Create('
_____',f2
));

doc.Add(Paragraph.Create('          '));

doc.Add(Paragraph.Create('          '));

doc.Add(Paragraph.Create('          LISTADO DE PRESTAMOS DEL
ESTUDIANTE ',f1));

doc.Add(Paragraph.Create('          '));

doc.Add(Paragraph.Create('          '+nombre+'
'+apellido,F3));
```

```
doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('
FECHA PRESTAMO
FECHA
DEVOLUCION
DEPENDENCIA',f2));

doc.Add(Paragraph.Create( '
_____
_____ ',f2));

for l := 0 to dsprestamos.Tables[0].Rows.Count - 1 do

begin

doc.Add(Paragraph.Create('
'+
dsprestamos.Tables[0].Rows[i].Item[0].ToString+
'+dsprestamos.Tables[0].Rows[i].Item[1].ToString+
'+dsprestamos.Tables[0].Rows[i].Item[2].ToString,f3));

end;

PrestamoSISBIC (cedula,cantidad);

if cantidad=1 then

doc.Add(Paragraph.Create('
El estudiante tiene:
'+cantidad.ToString+ ' prestamo (s) en las bibliotecas ',f3))

else

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('
'));

doc.Add(Paragraph.Create('
El estudiante no tiene prestamos en las
bibliotecas ',f3));
```

```
doc.Close;
```

```
    Response.Redirect('~\3.pdf');
```

```
doc.Free;
```

```
    Finally
```

```
    End;
```

```
end
```

```
else
```

```
    Response.Redirect('Default.aspx');
```

```
end;
```

```
procedure TPrestamosRealizados.OnInit(e: EventArgs);
```

```
begin
```

```
    // Required for Designer support
```

```
    InitializeComponent;
```

```
    inherited OnInit(e);
```

```
end;
```

```
procedure TPrestamosRealizados.CedulaUsuarios(nombre,apellido: String;
```

```
var cedula: String);
```

```
var
```

```
    conexion:string;
```

```
    objConexion: SqlConnection;
```

```
objComando: SqlCommand;

nombre, apellido,cedula:SqlParameter;

begin

    conexion:='user id=sa;password="123";data source="ELIZABET-
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';

    objConexion:= sqlConnection.Create;

    objConexion.ConnectionString:=conexion;

    objComando:= SqlCommand.Create;

    objComando.Connection:=objConexion;

    objComando.CommandText:='spPrestamos1';

    objComando.CommandType := CommandType.StoredProcedure;

nombre := SqlParameter.Create;

    nombre.set_ParameterName('@nombre');

nombre.set_DbType(DbType.String);

    nombre.set_Size(25);

    nombre.Value:=nombre.ToString;

    objComando.Parameters.Add(nombre);

apellido := SqlParameter.Create;

    apellido.set_ParameterName('@apellido');

apellido.set_DbType(DbType.String);
```

```
papellido.set_Size(25);

    papellido.Value:=apellido.ToString;

    objComando.Parameters.Add(papellido);

pcedula := SqlParameter.Create;

    pcedula.set_ParameterName('@cedula');

    pcedula.set_DbType(DbType.String);

pcedula.set_Size(11);

    pcedula.set_Direction(ParameterDirection.Output);

    objComando.Parameters.Add(pcedula);

try

    objConexion.Open;

    objComando.ExecuteNonQuery;

    cedula:=(pcedula.Value).ToString;

finally

    if (objConexion.State = ConnectionState.Open) then

        objConexion.Close;

end;

end;

procedure TPrestamosRealizados.PrestamoSISBIC(cedula: String;

var cantidad: Integer);
```

```
var

    conexion:string;

objConexion: SqlConnection;

objComando: SqlCommand;

pcedula, pcantidad:SqlParameter;

begin

    conexion:='user id=sa;password="123";data source="ELIZABET-
62FA1C";persist security info=False;initial catalog=BD_ECOTESIS';

    objConexion:= sqlConnection.Create;

    objConexion.ConnectionString:=conexion;

    objComando:= SqlCommand.Create;

    objComando.Connection:=objConexion;

    objComando.CommandText:='spPrestamoSISBIC';

    objComando.CommandType := CommandType.StoredProcedure;

    pcedula := SqlParameter.Create;

    pcedula.set_ParameterName('@cedula');

    pcedula.set_DbType(DbType.String);

    pcedula.set_Size(11);

    pcedula.Value:=cedula.ToString;

objComando.Parameters.Add(pcedula);
```



```
pcantidad := SqlParameter.Create;

pcantidad.set_ParameterName('@cantidad');

pcantidad.set_DbType(DbType.Int32);

pcantidad.set_Size(11);

pcantidad.set_Direction(ParameterDirection.Output);

objComando.Parameters.Add(pcantidad);

try

    objConexion.Open;

    objComando.ExecuteNonQuery;

cantidad:= Convert.ToInt32(pcantidad.Value);

finally

    if (objConexion.State = ConnectionState.Open) then

        objConexion.Close;

    end;

end;

end.
```

CAPITULO V

ANALISIS Y RESULTADOS

5.1 ANÁLISIS CUANTITATIVO Y CUALITATIVO

ECO está dirigido al .NET Framework y parte de los entornos de desarrollo de CodeGear como pueden ser C# o Delphi. Para probar la Hipótesis “*La utilización del lenguaje UML OCL 2.0 a través de la tecnología ECO .NET reducirá el esfuerzo del desarrollador y optimizará la construcción de sistemas informáticos*”, se realizará un análisis cuantitativo y cualitativo a través del desarrollo de prototipos utilizando el Framework ECO.NET y sin la utilización del mismo. Para ello se definirán parámetros de comparación que permitirán determinar cuál de las dos maneras optimiza la construcción de la aplicación reduciendo el esfuerzo del desarrollador y optimizando la construcción de sistemas.

La calificación de cada parámetro de comparación estará realizado en base a la escala que se mostrará a continuación, con ello se demostrará cuál de las dos maneras de implementación es la más eficiente para el desarrollador.

A las variables que intervienen se les dio un valor de 1, 2 o 3 puntos dependiendo si está de acuerdo totalmente, parcialmente o en desacuerdo. Para determinar la calificación de cada una de las variables empleadas se usó la siguiente tabla de valoraciones

Tabla Nº V. 1. Escala de Valoración Cualitativa

1	2	3
Ninguno	Parcial	Total
No se cumple	Se cumple Parcialmente	Cumple
Inadecuado	Poco Adecuado	Adecuado
No Usa	Frecuentemente	Usa
Externo	Parcial	Implícito

Tabla Nº V. 2. Tabla de comparación cualitativa

PARÁMETROS	APLICACIÓN ECO	APLICACIÓN SIN ECO
Diseño del Modelo Base de Datos	Implícita en la aplicación	Usa un DBMS externo
Conectividad con Base de Datos Definición de la Cadena de Conexión	Adecuado	Poco Adecuado
Modelado UML y Restricciones OCL	Usa	No Usa
Handles para conexión a bloques de memoria u DBMS	Usa	No Usa
Generación de código instantánea a través del modelo	Cumple	No cumple

Tabla Nº V. 3. Tabla de Comparación Cuantitativa

PARÁMETROS	APLICACIÓN ECO	APLICACIÓN SIN ECO
Diseño del Modelo Base de Datos	3	1
Conectividad con Base de Datos	2	1
Modelado UML y Restricciones OCL	3	1
Handles para conexión a bloques de memoria u DBMS	3	1
Generación de código instantánea a través del modelo	3	1

De los parámetros tomados en cuenta para el análisis se obtiene que la optimización en el desarrollo de la aplicación utilizando el framework ECO.Net es de 93.3%, mientras que una aplicación sin ECO apenas se obtuvo un 33.3% de optimización.

Los resultados fueron tabulados e interpretados en un gráfico de columnas que se muestra a continuación.

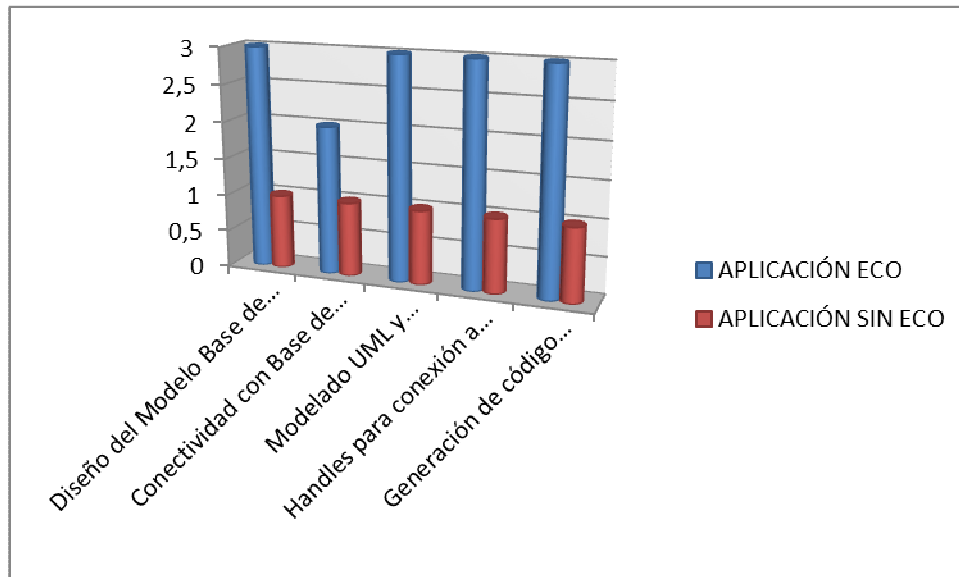


Figura V. 79. Cuadro estadístico de los parámetros de comparación

Para la medición del esfuerzo del programador se ha tomado dos parámetros importantes como son el tiempo de desarrollo de la aplicación y el número de líneas de código que fueron necesarias para la implementación de cada prototipo teniendo como resultado los siguientes datos.

Tabla Nº V. 4. Tabla de Medición del Esfuerzo

Esfuerzo			
	ECO	SIN ECO	% de ahorro
Tiempo de Desarrollo	152 minutos	315 minutos	51.75%
Líneas de Código	524	748	29.95%

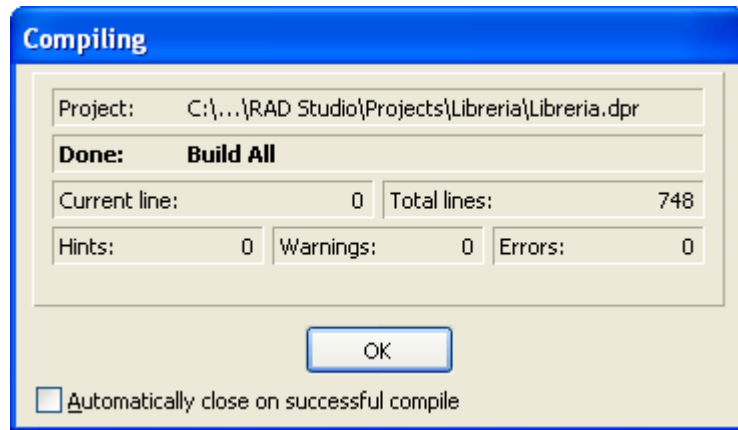


Figura V. 80. Líneas de código con ECO



Figura V. 81. Líneas de código sin ECO

Una vez tabulados los datos arrojados por el desarrollo de las aplicaciones se tienen los siguientes gráficos estadísticos.

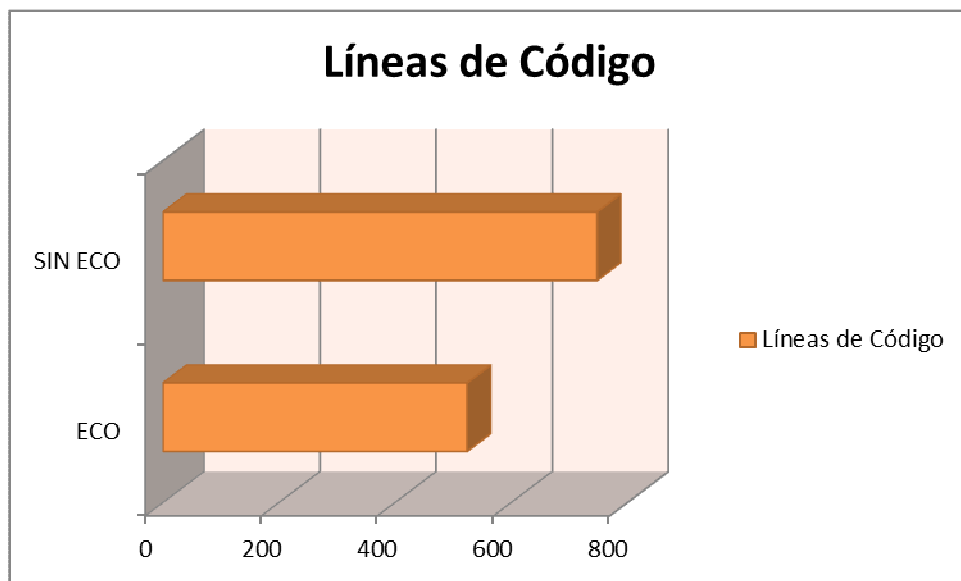


Figura V. 82. Gráfico de barras - Líneas de código

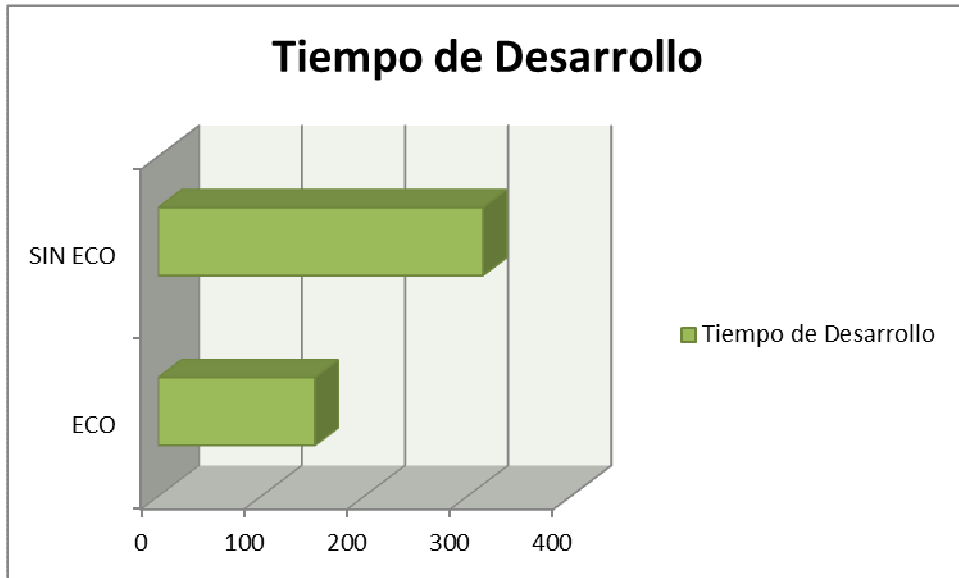


Figura V. 83. Tiempo de Desarrollo

5.2 CONCLUSIONES DE LA HIPÓTESIS

Las aplicaciones desarrolladas con ECO permiten tener un entorno de desarrollo más integrado, ya que contiene varias características implícitas que facilitan el desarrollo, por ejemplo el desarrollador aplica cambios al modelo y estos cambios se realizan automáticamente al código, ahorrándole al programador tiempo y esfuerzo en el desarrollo. Como se muestra en los datos anteriores se pudo comprobar que la tecnología ECO facilita en gran manera la optimización de la construcción de los Sistemas informáticos en un 93.3%, reduciendo el esfuerzo del programador y ahorrando un 51.75% de tiempo de desarrollo y en un 29.95% las líneas de código a programar, por lo tanto la hipótesis queda demostrada.

CONCLUSIONES

- Con el análisis del lenguaje UML y OCL presentes en el framework ECO.Net, permitió facilitar el desarrollo de la aplicación, además de disminuir el tiempo de implementación para los desarrolladores.
- Eco frente a LinQ permite generar el esquema de base de datos en la misma herramienta de desarrollo, es decir codegear, mientras que con LinQ solo se puede realizar consultas a una base de datos ya existente.
- La guía de implementación, permite facilitar la construcción de una aplicación ECO modelando los objetos de la aplicación utilizando UML y construyendo interfaces de usuario para aplicaciones web con controles ASP .NET para usuarios novatos en el conocimiento de la herramienta.
- Con el desarrollo de la aplicación SECNA propuesta, facilitará la obtención del certificado de no adeudar a las dependencias de la ESPOCH, de manera general sin particularizar. Ahorrando tiempo y recursos a los estudiantes.
- Al analizar la hipótesis se pudo comprobar que la tecnología ECO facilita en gran manera la optimización de la construcción de los sistemas informáticos en un 93.3%, reduciendo el esfuerzo del programador y ahorrando el 51.75% de tiempo de desarrollo así como un 29.95% de las líneas de código a programar.

RECOMENDACIONES

- Se recomienda implantar el sistema informático SECNA, ya que beneficiará a los estudiantes y secretarías de carrera, eliminando los recorridos que se realizan por cada una de las dependencias de la ESPOCH, cuyo resultado será la eficiencia y rapidez en la adquisición del certificado.
- Utilizar la guía de implementación para desarrolladores con poco conocimientos de la herramienta ya que esta facilita la utilización de la misma.
- Utilizar la tecnología del Framework ECO, es un marco innovador para crear aplicaciones basadas en modelos de clases, dicho modelo controla aspectos de la aplicación como la persistencia y creación automática del esquema de base de datos, reduciendo la cantidad de código de la aplicación y mejorando el mantenimiento.

RESUMEN

En la investigación se analizó el lenguaje UML OCL 2.0 a través del Framework ECO .net para implementar una solución informática con el objetivo de automatizar la emisión del certificado de no adeudar a ninguna dependencia previo a la obtención del título académico en la Escuela Superior Politécnica de Chimborazo (ESPOCH).

La automatización del proceso de emisión de certificados surge de la necesidad de cada una de las dependencias de la ESPOCH en optimizar tiempos y recursos a los estudiantes, por lo que se utilizó la técnica de encuestas para la recopilación de datos. Escogiendo las herramientas Rad Studio 2007 de Codegear con los componentes UML y OCL que incluyen el Framework ECO, y SQL Server 2005 de Microsoft, las dos herramientas comerciales; desarrollando la aplicación web con ASP.net con vinculación de datos a través de SQL Server. Permitiendo estas herramientas en el proceso de implementación del sistema informático, manipulación directa de instancias sin manipulación de líneas de código. Se aplicó la solución informática sobre una muestra de 20 estudiantes, cuyo tiempo aproximado es de un minuto y su costo de 0.45 centavos para obtener el certificado impreso; actualmente se demora tres días con un costo promedio de 5 dólares, según estudio preliminar efectuado en base a gestión de procesos.

Esta solución informática beneficiará al estudiante y a la secretaria de cada escuela en reducción de tiempo, y en costos económicos a la ESPOCH; razón por la cual recomendamos su implantación cuyo resultado será la eficiencia y rapidez en la adquisición del certificado.

SUMMARY

In the investigation, the language UML OCL 2.0 was analyzed through Framework ECO.Net in order to implement a computer solution with the objective of automating the emission of no debt certificate previous the obtaining of the academic degree in the Escuela Superior Politécnica de Chimborazo (ESPOCH).

The automation of the certifying emission process arises from the necessity of each one of the ESPOCH dependences in order to improve students' time and resources. The survey technique was used for the data compilation; choosing the tools Rad Studio 2007 of Codegear with the UML and OCL components that include framework ECO, and SQL Server 2005 of Microsoft, two commercial tools; developing the web application with ASP.Net with data linking through SQL Server. These tools permit the direct manipulation of instances without manipulation of code lines in the implementation process of the computer system. The computer solution was applied on one sample of 20 students, and the approximate time is one minute, and the cost 0.45 cents in order to obtain the printed certificate. At the moment it takes three day with an average cost of 5 dollars, according to the preliminary study made based on processes administration.

This computer solution will benefit the student and the secretary of each school in reduction of time and the ESPOCH in economic cost; that's why we recommend its implementation, whose, result will be the efficiency and speed in the certificate acquisition.

GLOSARIO

ASP.NET	Active Server Pages / Servidor de páginas activas entorno web
BD	Base de Datos
CASE	Ingeniería de Software Asistida por Computadores
CODEGEAR	Empresa desarrolladora de Software
DBMS	Sistema Administrador de Base de Datos
ECO	Enterprises Core Object
ESPOCH	Escuela Superior Politécnica de Chimborazo
HW	Hardware
IBM	International Business Machines Corporation
IEEE	Instituto de Ingeniería Eléctrica y Electrónica
OCL	Object Constraint Language / Lenguaje de restricción de objetos
ODBC	Open DataBase Connectivity / Conectividad de apertura de base de datos
SECNA	Aplicación Web para la Emisión de Certificados de No Adeudar a la ESPOCH
SRS	Estándar básico para requerimientos de software
SW	Software
UML	Unified Modeling Language / Lenguaje de modelado unificado
VCL	Visual Class Library / Biblioteca de componentes visuales

BIBLIOGRAFIA

REFERENCIAS WEB GENERAL

CODEGEAR RAD STUDIO

Code Gear All Development Technologies.

http://www.unisolutions.com.ar/print/borland_codegear_bds_eco.asp

[6/Abril/2009]

ELLING, Dee. Installation Notes for CodeGear RAD Studio 2007.

<http://edn.embarcadero.com/article/36859>

[6/Abril/2009].

LANUSSE, Andreano. RAD Studio 2007 - La fuerza de Delphi Win32, Delphi .NET y C++ juntos.

<http://edn.embarcadero.com/article/37104>

[16/Marzo/2009].

FRAMEWORK ECO

COLIBRI, Félix John. Tutorial ECO, Asp Net 2.0 Páginas maestras.

http://translate.googleusercontent.com/translate_c?hl=es&ie=UTF-8&sl=en&tl=es&u=http://www.felix-colibri.com/papers/web/asp_net/asp_net_master_pages/asp_net_master_pages.html&prev=t&rurl=translate.google.com.ec&usq=ALkJrhgXy0Kkb6LF0mpTMzBMErTn77560g

[16/Marzo/2010].

MARTÍNEZ, Norberto. SmartWare. Guadalajara - México. ECO III y aplicaciones web ASP.NET - Tutorial 2.

http://www.olegzhukov.com/content/Articles/CodeGearECO/UsingBorlandECOwithDevExpresscontrols/tabid/83/Default.aspx#_Toc153462228

[6/Abril/2009]

VILLEDA, Luis. Creando tu primera aplicación en ECO -Tutorial1.

<http://edn.embarcadero.com/article/34001>

[16/Julio/2009]

REFERENCIAS WEB RELACIONADA AL TEMA

ECO .NET

ASP.NET 2.0 Master-Content Pages and Navigation.

<http://www.drbob42.com/examines/examin99.htm>

[14/Julio/2009]

HODGES, Nick. Blackfish SQL Developer's Guide Overview.

<http://edn.embarcadero.com/article/36770>

[16/Marzo/2010]

VILLEDA, Luis. Trabajando con Asociaciones ECO III - Tutorial3.

<http://edn.embarcadero.com/article/34001>

[16/Marzo/2010]

LINQ

JIMENEZ, Alex. LinQ que es y cómo usarlo.

<http://alexjimenez.wordpress.com/2007/09/10/linq-definicion-como-usarlo/>

[8/Octubre/2009]

OCL

REYNOSO, Luis. 2004. OCL Object Constraint Language.

http://www.eici.ucm.cl/Academicos/R_Villarroel/descargas/ing_sw_1/OCL.pdf

[6/Abril/2009]

UML

BOOCH, J. Rumbaugh, I. Jacobson. Addison Wesley. Iberoamericana 1999. El Lenguaje Unificado de Modelado.

[http://manuales.astalaweb.com/_inicio/Marco.asp?dir=http://www.infomanuales.net/archivo/UML/Desarrollo orientado a objetos con UML 2.zip](http://manuales.astalaweb.com/_inicio/Marco.asp?dir=http://www.infomanuales.net/archivo/UML/Desarrollo%20orientado%20a%20objetos%20con%20UML%202.zip)

[20/Mayo/2009].

COSTAL, Dolores; SANCHO, María Ribera; TENIENTE, Ernest. Ediciones UPC, 2003. Especificación de sistemas software en UML.

[http://manuales.astalaweb.com/_inicio/Marco.asp?dir=http://www.infomanuales.net/archivo/UML/Desarrollo orientado a objetos con UML 2.zip](http://manuales.astalaweb.com/_inicio/Marco.asp?dir=http://www.infomanuales.net/archivo/UML/Desarrollo%20orientado%20a%20objetos%20con%20UML%202.zip)

[20/Mayo/2009]

HERNÁNDEZ, Enrique Orallo, 2002. El Lenguaje Unificado de modelado.

http://manuales.astalaweb.com/_inicio/Marco.asp?dir=http://www.infomanuales.net/archivo/UML/EI%20Lenguaje%20Unificado%20De%20Modelado.zip

[20/Mayo/2009]

ANEXOS

MANUAL DEL SISTEMA

SECNA

Sistema de Emisión de Certificados de No Adeudar

MANUAL DEL SISTEMA

METODOLOGÍA XP

SISTEMA DE EMISIÓN DE CERTIFICADOS DE NO ADEUDAR A LAS DEPENDENCIAS DE LA ESPOCH PREVIO A LA OBTENCION DEL TITULO ACADEMICO

I. Ingeniería de la Información

I.1 Definición del Ámbito del Software

Actualmente la ESPOCH no cuenta con una aplicación que permita obtener el certificado de no adeudar a las dependencias previas a la obtención del título académico, para obtener este certificado se debe cumplir con los requisitos necesarios acordes a la facultad, estando estos tipificados en el reglamento institucional.

Entre los requisitos se encuentra el listado de recolección de firmas de no adeudar en las dependencias de la Institución. Este listado varía de acuerdo a las políticas establecidas en cada Facultad. Todo este proceso es manual, por lo que toma tiempo lograr llenar el documento con las firmas requeridas.

I.2. Antecedentes Tecnológicos

I.2.1. Recursos Hardware

Cantidad: 1 PC	
Procesador:	Intel
Velocidad:	1.7 G.Z
Memoria:	1GB
Disco Duro:	320 GB
CD-ROM:	DVD, CD ROM, CDRW
Teclado:	WIN XP PS/2

Monitor:	LG Digital LCD 15"
-----------------	--------------------

I.2.2. Recursos Software

Características	
Sistema Operativo	Windows XP
Lenguaje de Programación:	BDS 2007
Base de Datos:	SQL SERVER 2007
Antivirus:	Antivir
Otros:	Adobe Reader Internet Explorer 8.0 o Mozilla Firefox install_flash_player Office 2010 Componentes DBS

I.3 Definición del Problema

La ESPOCH institución encargada de formar profesionales capaces de desenvolverse en diferentes áreas de la carrera, gracias a los conocimientos científicos, tecnológicos y valores humanos adquiridos por cada uno de los estudiantes, previo a la obtención del en cualquier oferta académica con las que cuenta la ESPOCH, se debe cumplir con los requisitos necesarios acordes a la facultad, estando estos tipificados en el reglamento institucional, requiere automatizar el proceso de emisión del certificado de no adeudar a ninguna dependencia previo a la obtención del título académico, en cada una de las facultades existentes en la Institución. El mismo que facilitará y acortará el tiempo de obtención del requisito.

I.3.1 Definir Alternativas de Solución

Es por eso que se ha visto la necesidad de realizar un Sistema que permita, la automatización del sistema informático de emisión del certificado de no adeudar a ninguna dependencia previo a la obtención del título profesional en la ESPOCH la misma que disminuirá los tiempos de obtención.

Cada una de las secretarias de las escuelas cuenta con los equipos necesarios para implantar el sistema, con la finalidad de lograr que los datos que se obtengan sean fiables y transparentes, así como mejorar la obtención del documento en tiempo real.

II. Especificación de requerimientos software

Este documento contiene las especificaciones de requisitos software (SRS) para el Sistema Emisión de Certificado de no Adeudar a las Dependencias de la ESPOCH previo a la obtención del título académico. Todo su contenido ha sido elaborado en colaboración con el usuario y responsable del futuro proyecto además se ha reutilizado software que ha sido usando hasta el momento

II.1 Ámbito del sistema

La razón que motiva al desarrollo del sistema es la pérdida de tiempo que se produce cuando el estudiante recorre la institución para la recolección de firmas sin que estas sean validadas en ningún soporte.

El futuro sistema se encargará de generar el certificado de no adeudar a todas las dependencias existentes en la ESPOCH, así como de generar préstamos en las dependencias asociadas al sistema.

II.2 Definición, Acrónimos y Abreviaturas

II.2.1 Definiciones

API: Interfaz de programa de aplicaciones, son las funciones, mensajes, estructuras y tipos de datos, así como las instrucciones para la creación de aplicaciones que ejecutan bajo Windows.

ATRIBUTOS: Indica una o más características de un objeto

DESARROLLADOR: Persona dedicada a la creación del producto software

DINÁMICO: Permite recorrer por todos los botones de comando

ESPOCH: Escuela Superior Politécnica de Chimborazo, institución donde se desarrolla el sistema SECNA.

ESTÁNDAR: Normas aprobadas por un comité especializado de estandarización, que se utiliza para documentar.

INTERFAZ: Lo que es visible para el usuario, y le permite interactuar con el sistema

SRS: Especificación de Requerimientos Software. Documento Técnico que refleja las necesidades de la empresa.

SQL: Structured Query Lenguaje.

II.2.2 Acrónimos

ERS Especificación de Recursos de Software.

IEEE Instituto de Ingenieros Eléctricos y electrónicos

ASP.NET Active Server Pages / Servidor de páginas activas entorno web

BD Base de Datos

CASE Ingeniería de Software Asistida por Computadores

CODEGEAR Empresa desarrolladora de Software

DBMS Sistema Administrador de Base de Datos

ECO Enterprise Core Object

ESPOCH Escuela Superior Politécnica de Chimborazo

- HW** Hardware
- IBM** International Business Machines Corporation
- IEEE** Instituto de Ingeniería Eléctrica y Electrónica
- OCL** Object Constraint Language / Lenguaje de restricción de objetos
- ODBC** Open DataBase Connectivity / Conectividad de apertura de base de datos
- SECNA** Aplicación Web para la Emisión de Certificados de No Adeudar a la ESPOCH
- SRS** Estándar básico para requerimientos de software
- SW** Software
- UML** Unified Modeling Language / Lenguaje de modelado unificado
- VCL** Visual Class Library / Biblioteca de componentes visuales

II.2.3 Referencias

Code Gear All Development Technologies.

[http://www.unisolutions.com.ar/print/borland_codegear_bds_eco.as](http://www.unisolutions.com.ar/print/borland_codegear_bds_eco.asp)

p

[6/Abril/2009]

ELLING, Dee. Installation Notes for CodeGear RAD Studio 2007.

<http://edn.embarcadero.com/article/36859>

[6/Abril/2009].

LANUSSE, Andreano. RAD Studio 2007 - La fuerza de Delphi Win32, Delphi .NET y C++ juntos.

<http://edn.embarcadero.com/article/37104>

[16/Marzo/2009].

II.3 Descripción general

II.3.1 Perspectivas del Producto

El Sistema de Emisión de Certificados de no adeudar a las dependencias de ESPOCH previo a la obtención del título académico es un sistema diseñado específicamente para la ESPOCH Escuela Superior Politécnica de Chimborazo, en base a los requerimientos establecidos es necesario obtener datos de todos los estudiantes de la institución. Por lo tanto necesita tener un medio de comunicación que permita obtener los datos de los estudiantes, encuentran almacenados en la base de datos del Académico el medio de comunicación más difundido es internet.

II.3.2 Características de los Usuarios

El Sistema de Emisión de Certificados de no adeudar a las dependencias de ESPOCH previo a la obtención del título académico, deberá ofrecer una interfaz de usuario intuitivo, de fácil manejo para los usuarios Administrador, Responsable, Secretaria y Estudiante.

II.3.3 Funciones del Producto

El Sistema de Emisión de Certificados de no adeudar a las dependencias de ESPOCH previo a la obtención del título académico, proveerá la posibilidad de ejecutar tareas enfocadas a la gestión, administración, responsable, secretaria y estudiante de mejorar la productividad del estudiante para la obtención del certificado.

II.3.4 Restricciones

El sistema de Control de préstamos y emisión del certificado de no adeudar previo a la obtención del título profesional en la ESPOCH (SECNA) se llevará a cabo en la Escuela Superior Politécnica de

Chimborazo a través del Departamento de Sistemas y Telecomunicaciones (DESITEL), el mismo que facilitará un registro de los préstamos de materiales y/o equipos, procesos que en la actualidad se lo realiza de manera manual, por lo que a partir de la creación de este sistema, ayudará a obtener un mejor control sobre los préstamos y a su vez sobre la emisión de los certificados de no adeudar a las diferentes dependencias de la institución optimizando de esta manera el tiempo y los recursos de la ESPOCH.

El Sistema SECNA será implementado utilizando DBS 2007 como herramienta de desarrollo y SQL Server 2005 como DBMS para el manejo de la base de datos.

II.4 Requisitos específicos

En este apartado, se detallaran los requisitos que deberán ser satisfechos por nuestra aplicación, con la finalidad de cubrir las necesidades expuestas por los usuarios y además solo se registrarán al correcto desempeño de los mismos.

Se describirán los requisitos funcionales del Sistema SECNA.

II.4.1 Requisitos Funcionales

Dentro del análisis que se ha realizado dentro de la institución, los requerimientos han plasmado en algunos puntos críticos, los cuales se menciona a continuación:

ESTUDIANTES

- Controlar los préstamos realizados y ser más puntuales en las fechas de entrega.

ADMINISTRADOR

- Realizar una migración automática diaria para obtener información de facultades, carreras, escuelas, estudiantes procedentes del Sistema Académico y deudores del Sistema de Bibliotecas (SISBIB)
- Realizar el ingreso, eliminación y actualización de dependencias, materiales o equipos y préstamos.
- Ejecutar la emisión del certificado de no adeudar a la ESPOCH
- Efectuar la administración de usuarios

RESPONSABLES PRESTAMOS

- Realizar la inserción, eliminación y actualización de materiales y préstamos.
- Realizar registro de la devolución del préstamo o material.

SECRETARIA DE ESCUELA

- Realizar la emisión del certificado de no adeudar a la ESPOCH para la obtención del título académico.

II.4.2 Requisitos de Interfaces Externas

II.4.2.1 Interfaz de Usuario

Las aplicaciones contarán con una interfaz de usuario fácilmente manejable ya que será orientada a ventanas, y el manejo de programas se realizara a través de teclado y mouse, lo que permitirá solicitar, visualizar y manipular de manera más eficiente, brindando una visualización comprensible al usuario.

II.4.2.2 Interfaz Hardware

Se utilizarán las interfaces hardware instaladas en cada dependencia de la institución.

II.4.2.3 Interfaz Software

Como herramientas para desarrollar se ha escogido BDS 2007, ya que incluye un soporte para el framework ECO en aplicaciones ASP.NET y como motor de base de datos SQL Server 2005.

II.4.2.4 Atributos del Sistema

Los atributos del sistema software son importantes por los siguientes aspectos que tomamos a consideración:

Fiabilidad

El sistema responderá a una petición del usuario en el 95% de los casos, por lo cual se tendrá un alto grado de fiabilidad.

Disponibilidad

El sistema debe estar disponible para ser utilizado por los usuarios el 100% de las veces, así como para los usuarios definidos.

III. Análisis

El sistema de Control de préstamos y emisión del certificado de no adeudar previo a la obtención del título profesional en la ESPOCH (SECNA) se llevará a cabo en la Escuela Superior Politécnica de Chimborazo a través del Departamento de Sistemas y Telecomunicaciones (DESITEL), el mismo que facilitará un registro de los préstamos de materiales y/o equipos, procesos que en la actualidad se realiza de manera manual, por lo que a partir de la creación de este sistema, ayudará a obtener un mejor control sobre los préstamos y a su

vez sobre la emisión de los certificados de no adeudar a las diferentes dependencias de la institución optimizando de esta manera el tiempo y los recursos de la ESPOCH.

Para la ejecución de la aplicación se requiere una buena infraestructura de soporte que garantice el acceso a la aplicación desde donde se requiera. La Escuela superior Politécnica de Chimborazo cuenta actualmente con una infraestructura aceptable y que servirá de base para la ejecución del mismo.

III.1 Definición de casos de uso

Formato Expandido

La creación de los casos de uso es una técnica que permite mejorar la comprensión de los requerimientos y se constituyen como historias o casos de utilización de un sistema; no son exactamente los requerimientos ni las especificaciones funcionales, sino que ejemplifica e incluyen fácilmente los requerimientos en las historias que narran.

Identificación de Casos de Uso

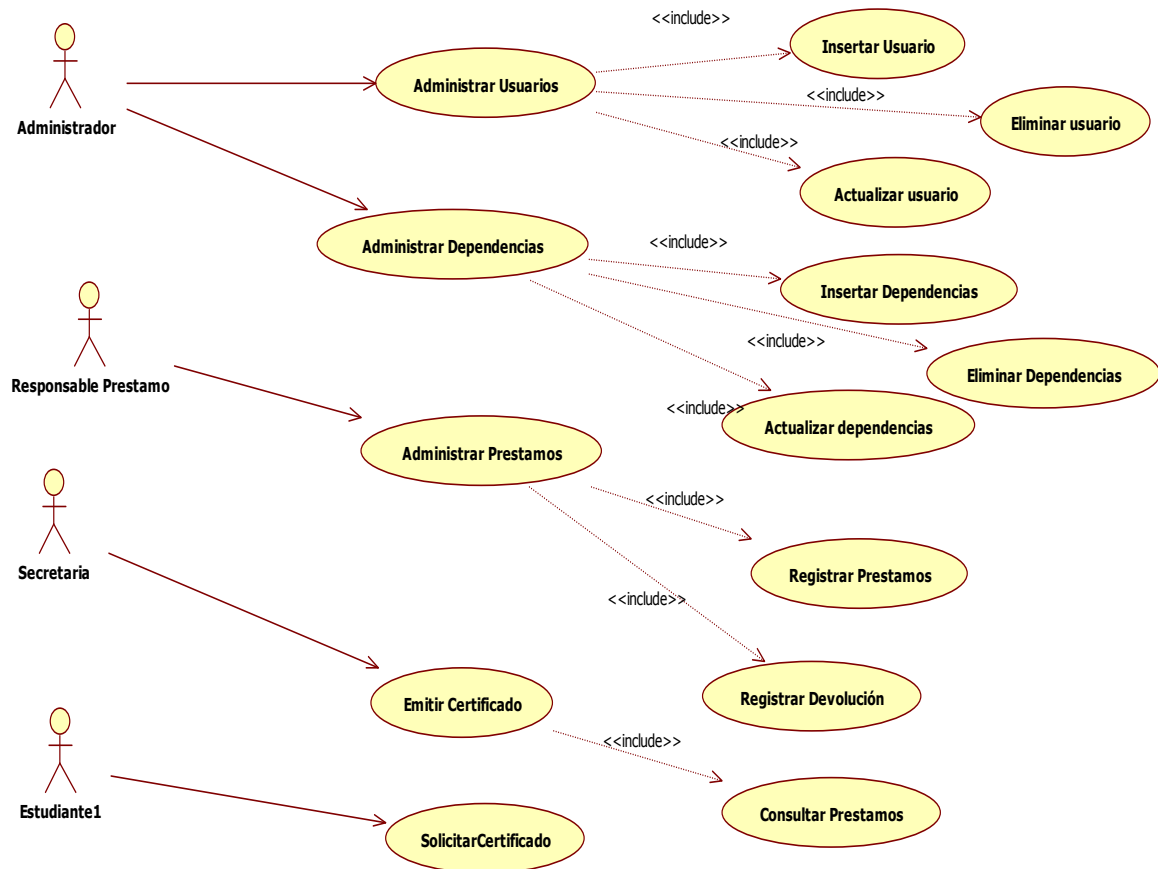
Como guía para la identificación inicial de casos de uso hemos utilizado el método basado en eventos que nos permitirán identificar los eventos externos al sistema

No	Nombre de caso de uso	Eventos	Actores
1	Autenticación del usuario	Ingresa a la página web	Administrador Responsable de prestamos Secretaria de carrera

			Estudiante
2	Generación de dependencias	Actualizar Eliminar Ingresar	Administrador
3	Dependencias	Listar	Responsable de prestamos
4	Estudiantes	Listar	Administrador Responsable de prestamos
5	Generación de materiales y/o equipos	Actualizar Eliminar Ingresar	Administrador Responsable de prestamos
6	Generación de usuarios	Actualizar Eliminar Ingresar	Administrador
7	Generación de certificado	Consultar Imprimir	Administrador Secretaria de carrera
8	Generación de reporte de préstamos	Mostrar información	Estudiante

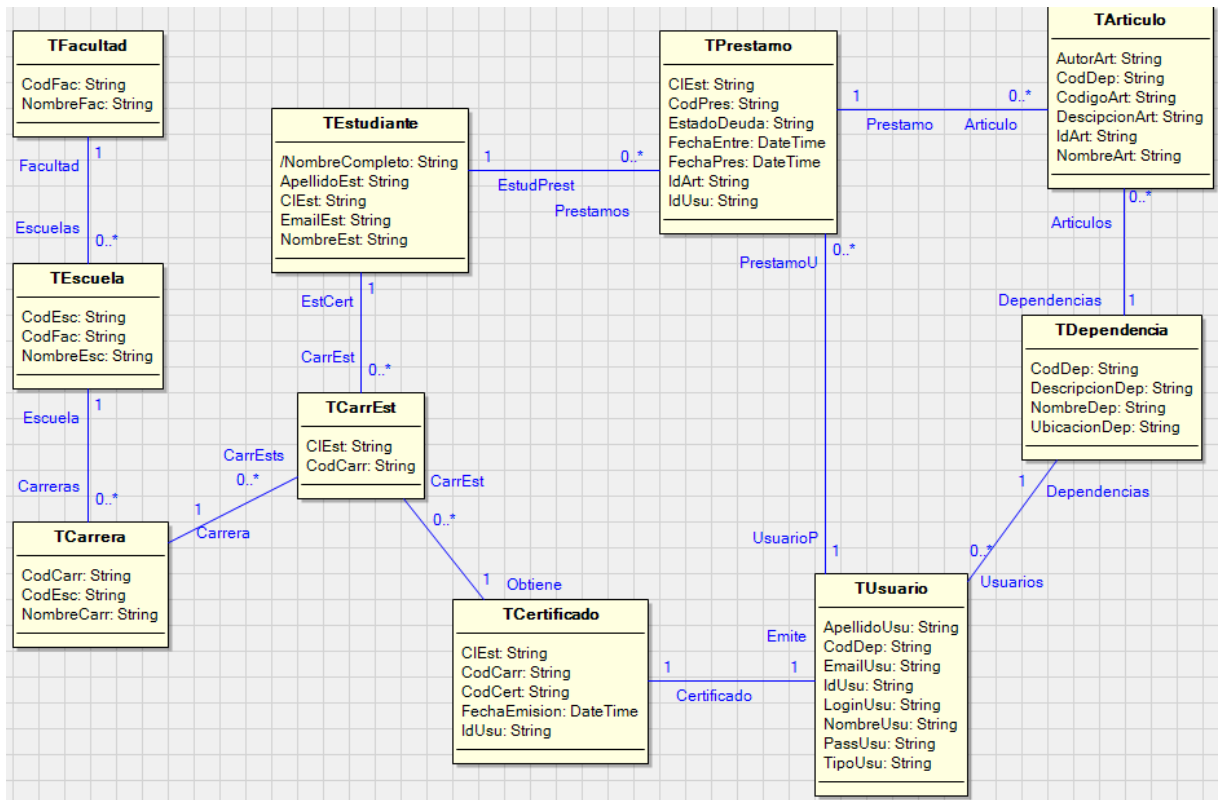
Diagramas de casos de uso

Se realizará una representación gráfica del entorno del sistema y su funcionalidad principal.



Modelo conceptual sistema propuesto

A continuación se identifican las clases del sistema de control de préstamos y emisión del certificado de no adeudar a la ESPOCH.



Definición de Diagramas de Secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Esta descripción es importante porque puede dar detalle a los casos de uso, aclarándolos al nivel de mensajes de los objetos existentes.

Diagrama de Secuencia para: Solicitar Artículo

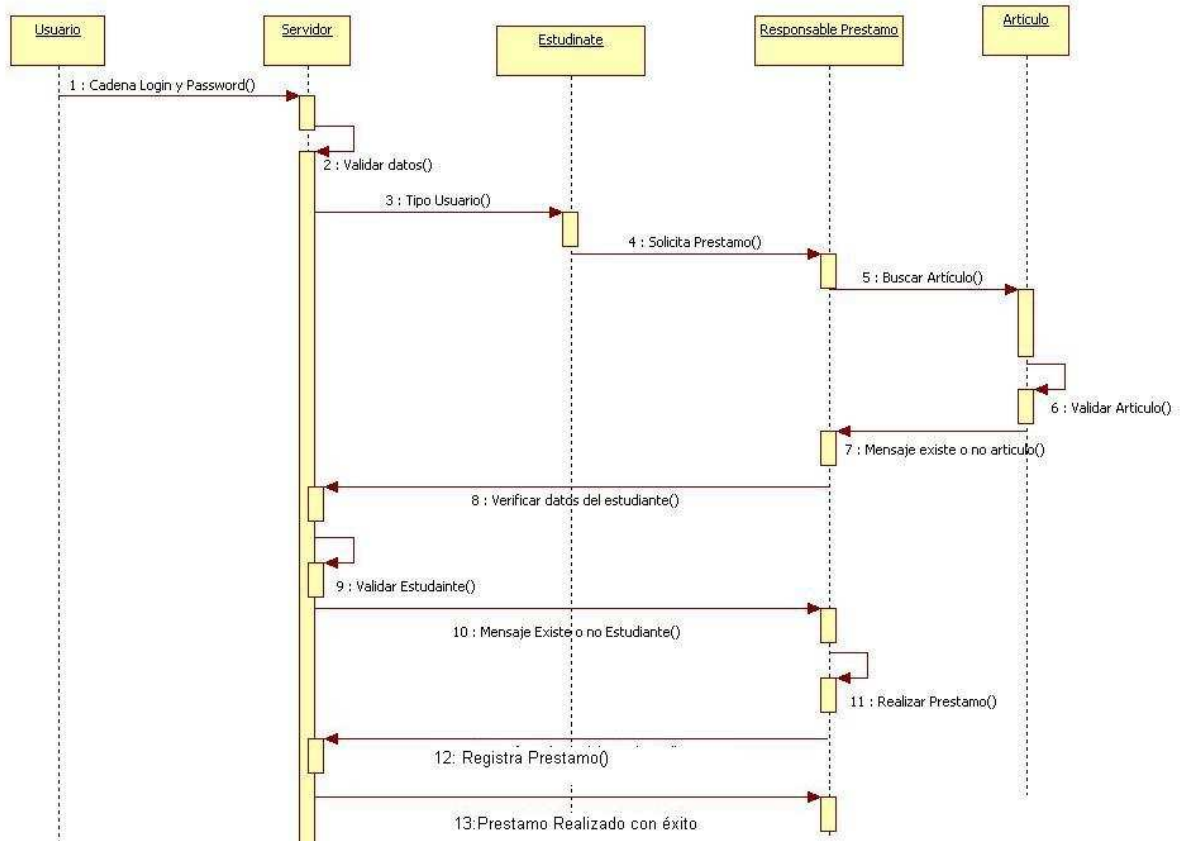


Diagrama de Secuencia para: Solicitar Certificado

Diagrama de Secuencia: Generación de informes

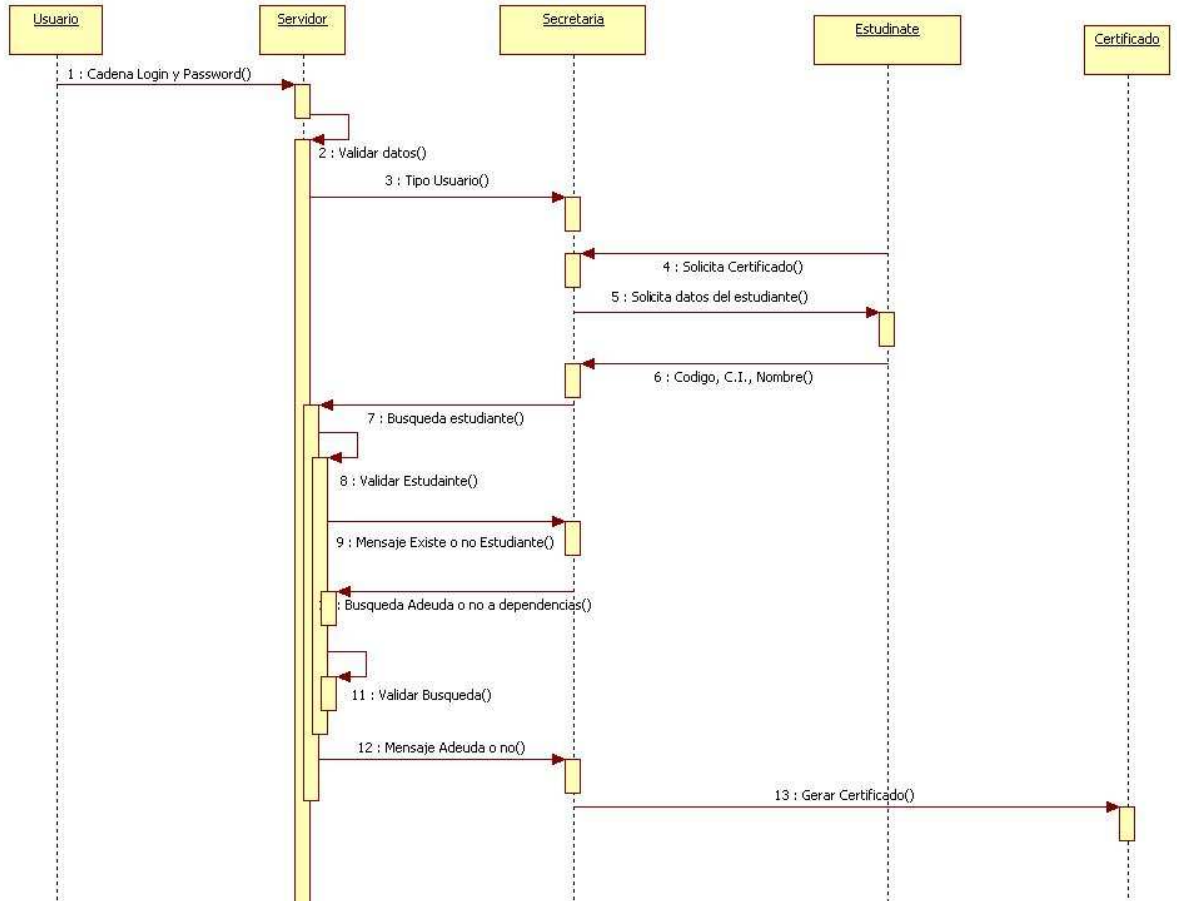
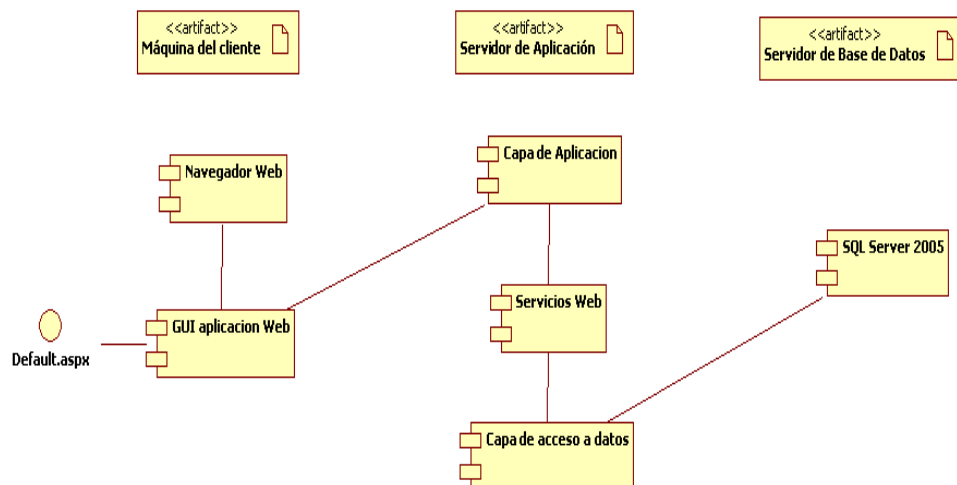


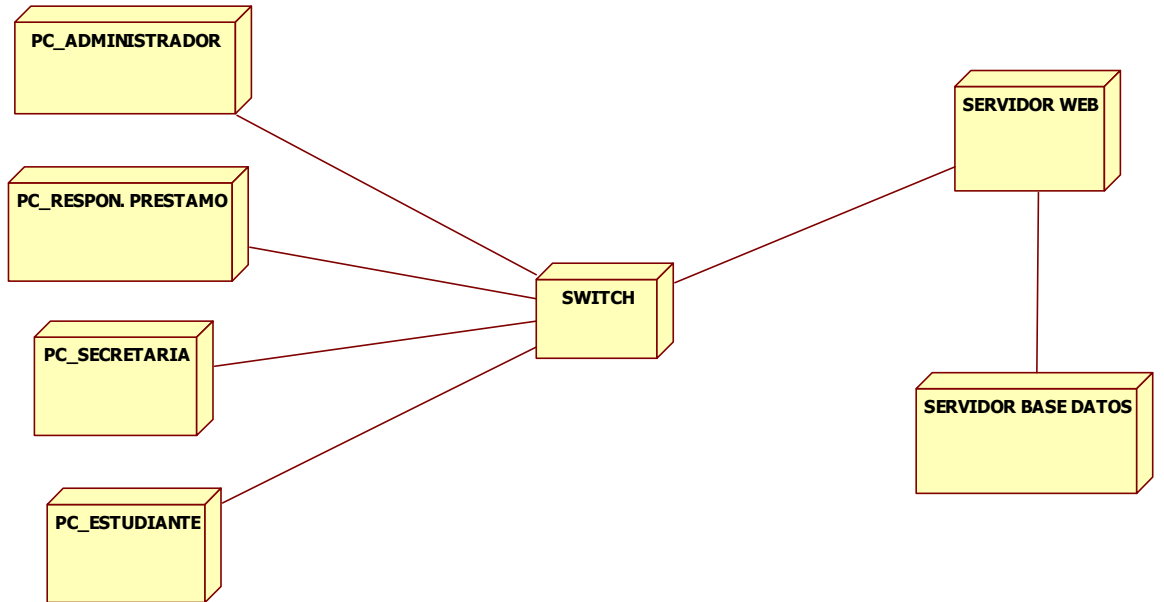
Diagrama de Componentes

El Diagrama de Componentes describe el comportamiento de las clases en el sistema.



Diagramas de Nodos

En este diagrama se representa los componentes hardware para el buen funcionamiento de la aplicación.



MANUAL DE USUARIO

SECNA

Sistema de Emisión de Certificados de No Adeudar

PRESENTACIÓN

En la actualidad la mayoría de empresas u organizaciones están encaminadas a automatizar a través de sistemas informáticos los procesos manuales, utilizando el medio de comunicación más difundido que es el internet. Por lo que desarrollar una aplicación web conlleva a difundir a nivel mundial la información necesaria para obtener beneficios importantes.

La tecnología ECO realmente es equivalente a un desarrollo rápido guiado por modelos; reduce drásticamente la cantidad de código que es necesario teclear manualmente, acelerando el despliegue y mejorando la mantenibilidad de sus aplicaciones. Utiliza un subconjunto del lenguaje UML; emplea OCL, lenguaje de restricción de objetos, para definir expresiones sobre los modelos UML.

INTRODUCCIÓN

SECNA Sistemas de Emisión de Certificados de No Adeudar a las Dependencias de la Escuela Superior Politécnica de Chimborazo, busca optimizar el tiempo de respuestas para los estudiantes que desean obtener el certificado de No Adeudar a las Dependencias de la ESPOCH previo a la obtención del título académico con la finalidad de poder brindar soluciones integrales con proyección a futuro, además garantizar la confiabilidad y seguridad de los datos, ya que la nueva herramienta a aplicarse trabaja con bases de datos que permiten mantener la integridad y persistencia de la información almacenada.

Las bases de datos se han constituido como una de las herramientas ampliamente difundidas en la actualidad, utilizadas en cuanto a recuperación y almacenamiento de información en todos los campos.

Existen tecnologías alternativas como ECO que permiten eliminar sentencias SQL optimizado procesos de desarrollo de la aplicación, sin tener relación con el manejador de base de datos

OCL permite a los desarrolladores de software escribir restricciones sobre modelos de objetos. Estas restricciones son particularmente útiles, en la medida en que permiten a los desarrolladores crear un amplio conjunto de reglas que rigen el aspecto de un objeto individual.

En particular no hay necesidad de preocuparse del diseño de la base de datos que hacen las aplicaciones ECO más objetos relacionales y menos orientados a bases de datos. La construcción del modelo en ECO se hace en un editor visual de UML. El código correspondiente es generado al vuelo en cualquiera de los lenguajes.

El presente manual va dirigida a los Administradores, Responsables de cada Departamento, Secretarias y estudiantes en general como una guía que permita orientar al usuario, con el fin de que pueda acceder y visualizar la información requerida.

Capítulo I

1. REQUISITOS MÍNIMOS PARA SU UTILIZACIÓN

Los requerimientos mínimos de hardware son:



Una computadora Pentium IV 2,40GHz (mínimo)



Un mouse



512Mb de memoria RAM (mínimo) 1 GB
recomendable



Espacio en el disco duro 10 GB.



Monitor Super VGA

Los requerimientos de software son:



Windows XP o superior



Adobe Reader

Internet Explorer 8.0 o Mozilla Firefox

Install_flash_player

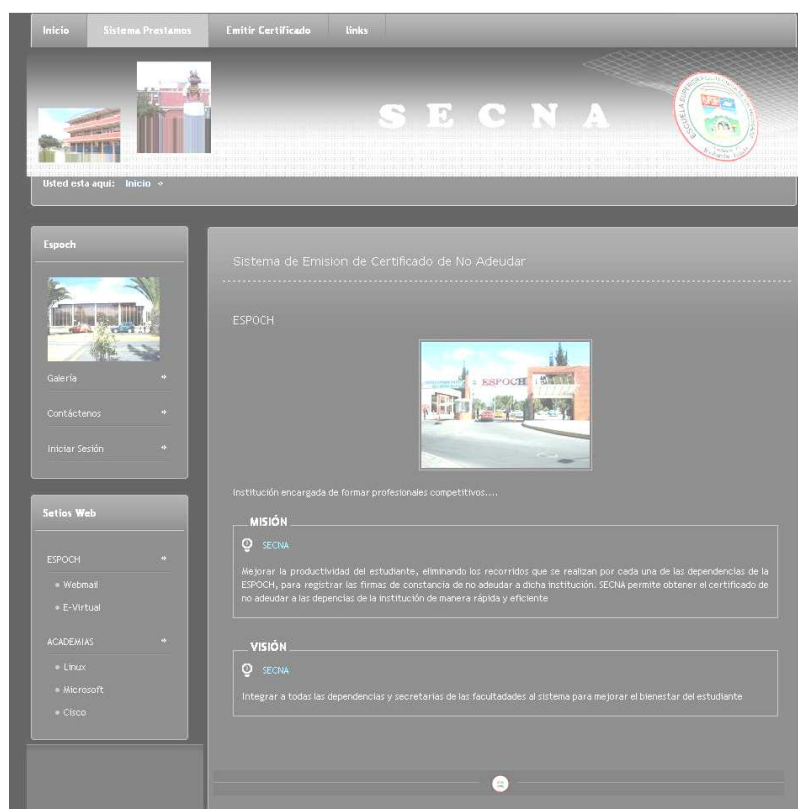
Capítulo II

2. INSTRUCCIONES

Para ingresar a la página principal digite la siguiente URL en el navegador.

<http://www.secna.edu.ec>

A continuación se mostrará la página principal, con un menú superior y otro en la parte izquierda, para acceder a la información de su interés.



El menú superior encontraremos disponible en todas las páginas recorridas en el sistema.



El link **Inicio** presenta un enlace informativo de la misión y visión de la ESPOCH.

El link **Sistema Préstamos** contiene información acerca de los préstamos que realiza la institución.



El link **Emitir Certificado** muestra información acerca del sistema de emitir certificado.

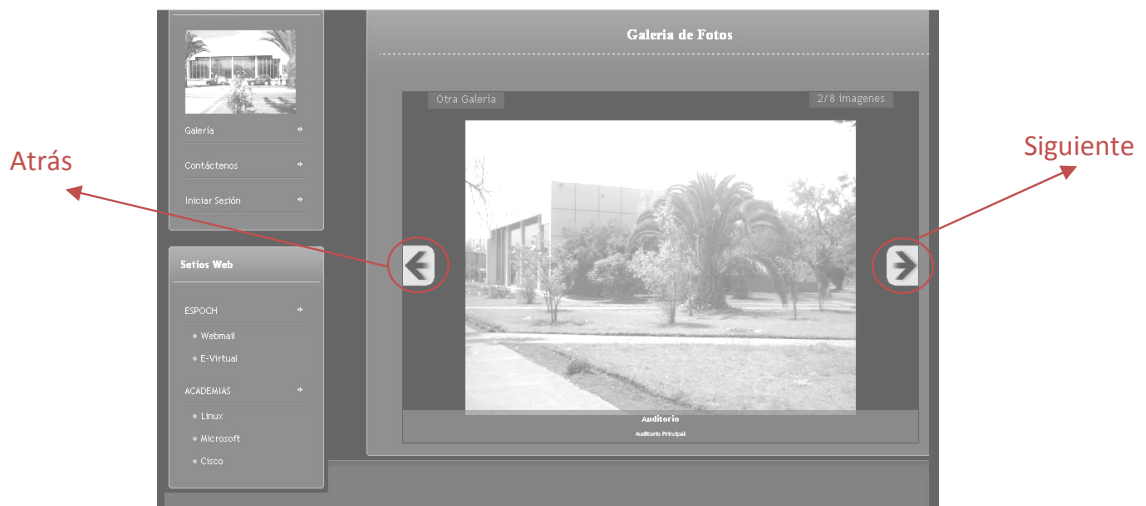


Por último contiene un enlace a la página de la ESPOCH.

En el menú de la izquierda, presenta enlaces para acceder a:



El link **Galería** muestra las fotos de la institución de su estructura física.



El link **Contáctenos** presenta un formulario para enviar un correo electrónico.

A screenshot of a web application's contact form. The main area is titled "Contactenos" and contains a form with the following fields: "Nombre:" (text input), "direccion E-mail:" (text input), "direccion:" (text input), and "Comentario:" (text area). Below the form, there is a "Enviar" button. The sidebar on the left is identical to the previous screenshot, showing the menu and "Sitios Web" section.

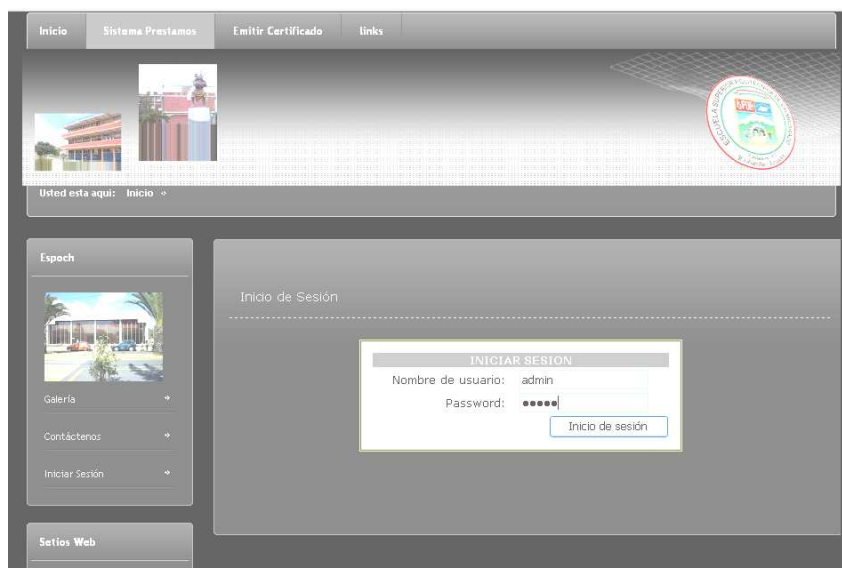
El link **Iniciar Sesión** contiene un inicio de sesión para el administrador, responsable del préstamo, secretaria y estudiante del sitio web.

A screenshot of a web application's login page. The main area is titled "BIENVENIDOS AL SISTEMA SECNA DE LA ESPOCH" and contains a list of roles and their responsibilities: "Administrador SECNA: Responsable del Ingreso de nuevos Usuarios, Dependencias, Reportes ... Todo y mas en el sitio Sitio para Administrador", "Responsable de Préstamos SECNA: Ingreso de nuevos Artículos, Registrar Préstamos,devoluciones ... Todo y mas en el sitio Sitio para Responsables de Préstamo", "Secretaria SECNA: Emitir Certificado de no adeudar a las dependencias de la ESPOCH,Reportes Todo y mas en el sitio Sitio para Secretaria", and "Estudiante: Información de Reportes de Adeudamiento en las dependencias de la ESPOCH..... Todo y mas en el sitio Sitio para Estudiantes". The sidebar on the left is identical to the previous screenshots, showing the menu and "Sitios Web" section.

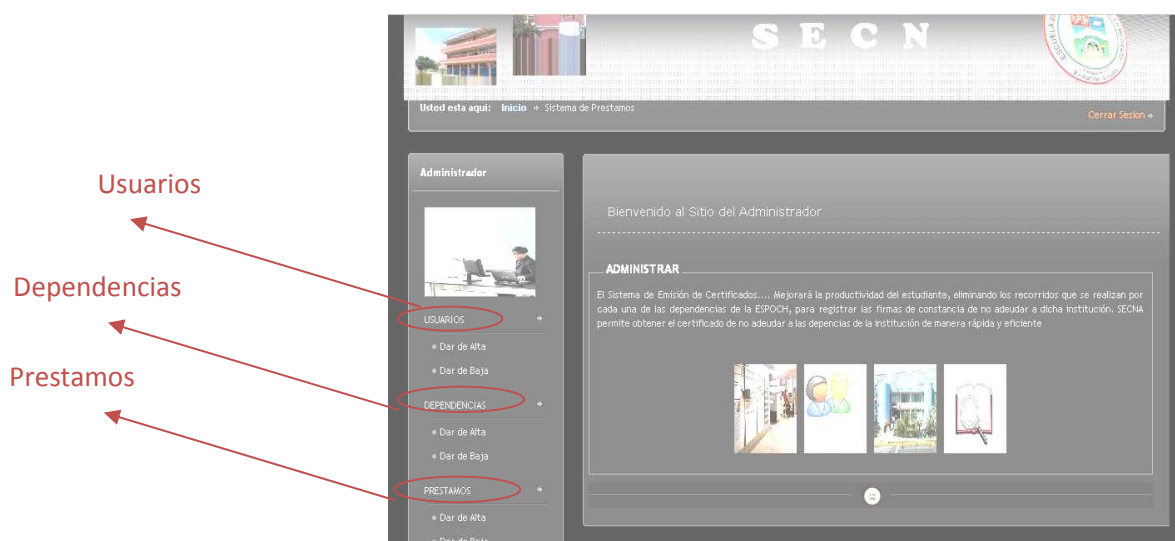
Capítulo III

3.1. Operaciones del Administrador

Al Dar clic en el enlace del Administrador aparece el inicio de sesión, escoger la opción Sitio del Administrador y validarse con la cuenta de administrador



Se presenta el sitio del Administrador, que puede realizar tareas dar de alta o baja a Usuarios, Dependencia, o Préstamos y visualizar los informes.



Dar de Alta Usuarios

Registrar un nuevo usuario dar clic en el enlace **dar de alta** se muestra una pantalla para crear usuarios, se puede crear un nuevo usuario, actualizar los datos o guardar.

Sitio del Administrador
MARIA GABRIELA CARDENAS HINOJOSA

CREACION DE USUARIOS

Usuarios del Sistema

Nombres: Manuel Haro

CI: 060293486-1

Login: mharo

Password:

Tipo Usuario: RESPONSABLE

Dependencia: Laboratorio Investigacion

Guardar

Guardar datos (nuevo/actualizar)

[Atras]

Dar clic en guardar y muestra un mensaje los datos han sido almacenados correctamente.

En el botón **Actualizar** datos seleccionar al usuario, busca al usuario, modificar los datos y guardar.

USUARIOS

Dar de Alta

Dar de Baja

DEPENDENCIAS

Dar de Alta

Dar de Baja

PRESTAMOS

Nuevo Prestamo

Devolución Prestamos

Seleccionar Usuario

Usuario: MERCY JANETH MENDEZ MALDONAD

Usuarios del Sistema

Nombres: MERCY JANETH MENDEZ MALDONAD

CI:

Login: janeth

Password: janeth

Tipo Usuario: SECRETARIA

Dependencia: Secretaria

Guardar

[Atras]

Dar de Baja Usuarios

Seleccionar al usuario que quiere dar de baja, seleccionar al usuario, se muestra los datos del usuario que se dará de baja, dar clic en dar de baja y en el botón guardar datos se mostrará el mensaje los datos han sido registrados correctamente.

Buscar Usuario

Usuario: VICTOR BENITEZ

Datos del Usuario

Nombre:

CI:

Login:

Password:

Tipo Usuario:

Dependencia:

Estado: Inactivo

NO HA SELECCIONADO ES ESTADO O EL USUARIO

Eliminar

Dar de Alta Dependencias

El enlace dar de alta se muestra una pantalla para crear dependencia, actualizar los datos o dar de baja dependencias.

Usted está aquí: Inicio > Sistema de Prestamos

Cerrar Sesión

Administrador

Sitio del Administrador

MARIA GABRIELA CARDENAS HINOJOSA

NUEVA DEPENDENCIA

Datos Dependencias

Nombre:

Departamento: Laboratorio de Mecatrónica

Ubicación: ETS

Horario: 8:00 a 14:00

Estado: Activo

Guardar

Guardar datos (nuevo/actualizar)

Actualizar Dependencia seleccionar la dependencia, se busca dicha dependencia se muestra los datos, modificar los campos deseados, dar clic en **Guardar**.

MARIA GABRIELA CARDENAS HINOJOSA

ADMINISTRACION DE DEPENDENCIAS

Seleccionar Dependencia

Dependencia: Laboratorio de Mecatrónica

Datos Dependencias

Nombre Departamento: Laboratorio de Mecatrónica

Ubicación: EIS

Horario: 8h00 a 14h00

Estado: Efectivo

Guardar

Guardar datos (nuevo/actualizar)

[Atras]

Para **Actualizar** la dependencia seleccionar la dependencia se carga los datos, dar clic en guardar.

MARIA GABRIELA CARDENAS HINOJOSA

ADMINISTRACION DE DEPENDENCIAS

Seleccionar Dependencia

Dependencia: Laboratorio de Mecatrónica

Datos Dependencias

Nombre Departamento: Laboratorio de Mecatrónica

Ubicación: EIS

Horario: 8h00 a 14h00

Estado: Activo

Guardar

[Atras]

Dar de Baja Dependencias

Seleccionar a la dependencia que quiere dar de baja, se muestra los datos de la dependencia, dar clic en estado inactivo y en el botón **Guardar** datos, se mostrará el mensaje los datos se han registrados correctamente.

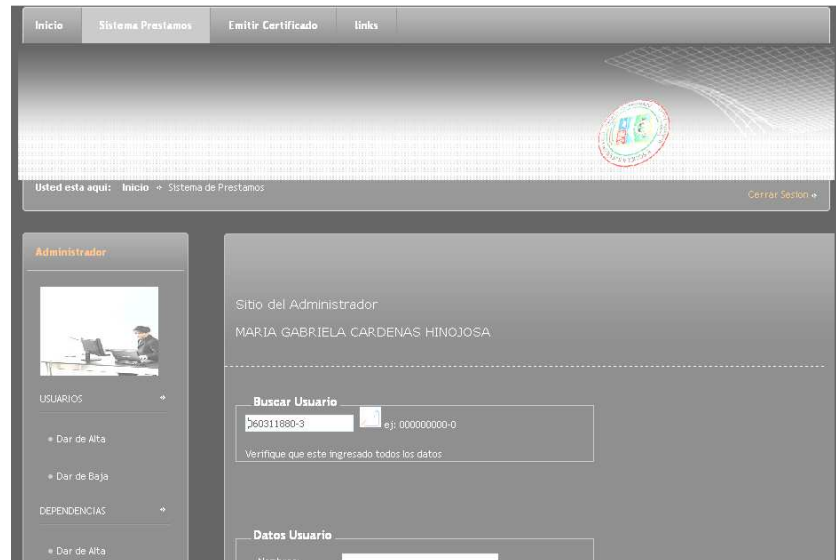
The screenshot shows a web application interface for managing dependencies. On the left is a sidebar menu with categories: USUARIOS, DEPENDENCIAS, and PRESTAMOS. The main content area is titled 'DAR BAJA DEPENDENCIA'. It contains a 'Seleccionar Dependencia' section with a dropdown menu set to 'Laboratorio3'. Below this is a 'Datos Dependencias' section with input fields for 'Nombre', 'Departamento', 'Ubicación', and 'Horario'. The 'Estado' field has a radio button selected for 'Inactivo'. An 'Eliminar' button is positioned to the right of the 'Datos Dependencias' section. At the bottom left of the main area is a '[Atras]' link.

Nuevo Préstamo

Ingresar el número de cédula del estudiante y dar clic en buscar.

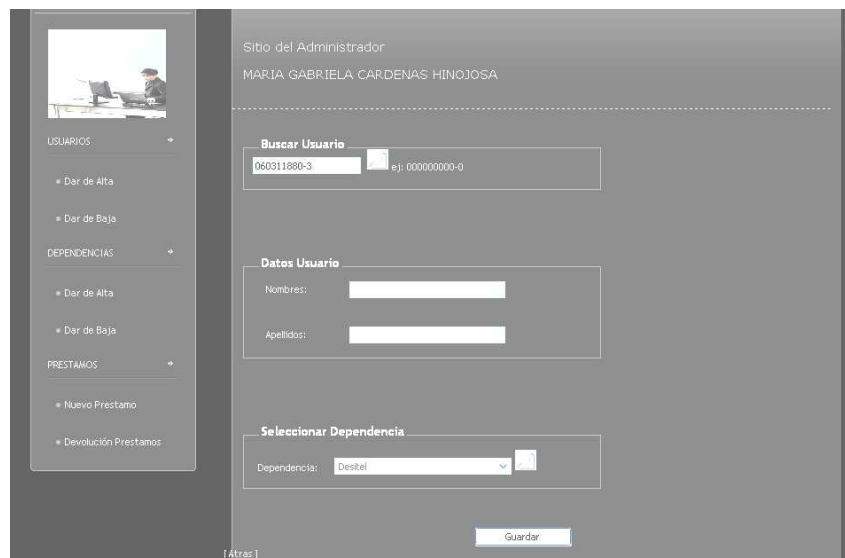
The screenshot shows the 'Sistema Prestamos' web application. The top navigation bar includes 'Inicio', 'Sistema Prestamos', 'Emitir Certificado', and 'links'. The main content area is titled 'Sistema Prestamos' and displays the user's name 'MARIA GABRIELA CARDENAS HINOJOSA'. Below this is a 'Buscar Usuario' section with an input field for a user ID (example: 000000000-0) and a 'Guardar' button. A message below the input field reads 'La cedula no esta registrada...'. At the bottom left of the main area is a '[Atras]' link. The sidebar menu on the left shows the 'USUARIOS' category with a 'Dar de Baja' option selected.

Si el estudiante no existe muestra un mensaje verificar datos de ingreso o el estudiante no existe.



The screenshot shows the 'Sistema Prestamos' web application interface. At the top, there is a navigation menu with 'Inicio', 'Sistema Prestamos', 'Emitir Certificado', and 'links'. Below the menu, there is a header area with a logo and the text 'Usted esta aqui: Inicio > Sistema de Prestamos'. The main content area is titled 'Administrador' and shows the name 'MARIA GABRIELA CARDENAS HINOJOSA'. There is a search bar labeled 'Buscar Usuario' with the input '60311880-3' and a dropdown menu showing 'ej: 000000000-0'. Below the search bar, there is a message 'Verifique que este ingresado todos los datos'. There is also a 'Datos Usuario' section with a 'Nombres:' field.

Si el estudiante existe muestra los datos de dicho estudiante.



The screenshot shows the 'Sistema Prestamos' web application interface. At the top, there is a navigation menu with 'Inicio', 'Sistema Prestamos', 'Emitir Certificado', and 'links'. Below the menu, there is a header area with a logo and the text 'Usted esta aqui: Inicio > Sistema de Prestamos'. The main content area is titled 'Administrador' and shows the name 'MARIA GABRIELA CARDENAS HINOJOSA'. There is a search bar labeled 'Buscar Usuario' with the input '060311880-3' and a dropdown menu showing 'ej: 000000000-0'. Below the search bar, there is a 'Datos Usuario' section with 'Nombres:' and 'Apellidos:' fields. There is also a 'Seleccionar Dependencia' section with a dropdown menu showing 'Dependencia: Desitel'. At the bottom, there is a 'Guardar' button and a '[Atras]' link.

Seleccionar el artículo, si no existe el artículo dar clic en el botón **Nuevo**, llenar los campos solicitados.

PRESTAMOS

- Nuevo Préstamo
- Devolución Préstamos

Seleccionar Dependencia

Dependencia: Laboratorio Investigación

Seleccionar Artículo

Artículo: Libros

Datos Artículo

Nombre:

Descripción:

Stock:

En los datos del préstamo el sistema pone la fecha actual del sistema y muestra la fecha en la cual tiene que entregar el artículo marcamos el estado en prestado y dar clic en **Guardar**

Datos Artículo

Nombre:

Descripción:

Stock:

Datos Préstamo

Fecha Préstamo:

Fecha Devolución:

Estado: Prestado

[Atras]

Copyright © 2010 - EIS ESPOCH
Panamericana Sur Km 1 1/2 Riobamba - Ecuador | Telefono: (03) 2 605907 - 2 605901

Devolución Préstamo

Ingresar el número de cédula del estudiante, dar clic en buscar.



Muestra los datos del estudiante, con todos los prestamos realizados y estado si esta entregado o pendiente.

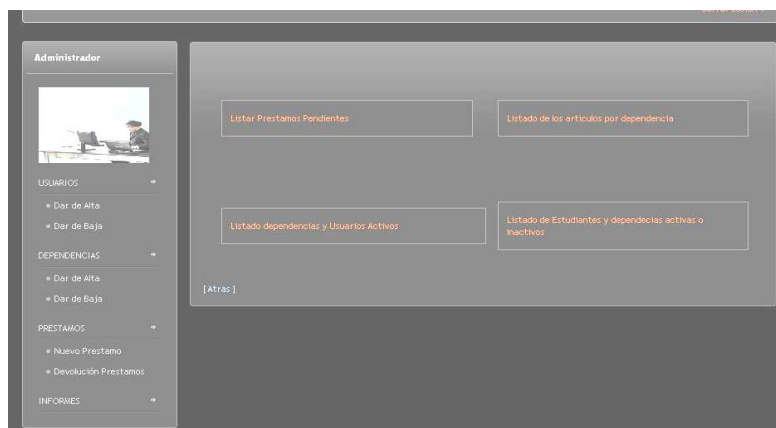


Cambiamos es estado de pendiente a entregado y dar clic en el botón **Guardar**.




Informes

Seleccionar el reporte que quiere visualizar el usuario administrador.



Puede visualizar los préstamos pendientes que existen de los estudiantes.




ESPOCH
 ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
 FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
 ESCUELA DE INGENIERÍA EN SISTEMAS
 Telefax: 005009 ext 214 Robamba-Ecuador

LISTADO DE ESTUDIANTES POR DEPENDENCIA

CEDULA	NOMBRES	DEPENDENCIA	UBICACION
086008007	Xavier Perez	Laboratorio Investigacion	Escuela EIG
00000000-0	Juan Perez	Laboratorio Investigacion	Escuela EIG
05033486-0	ALEX ALBERTO TACUPI LOJULLA	Laboratorio Investigacion	Escuela EIG
09876543-0	Carlos Flores	Laboratorio	EIG

Puede visualizar los artículos por dependencia.



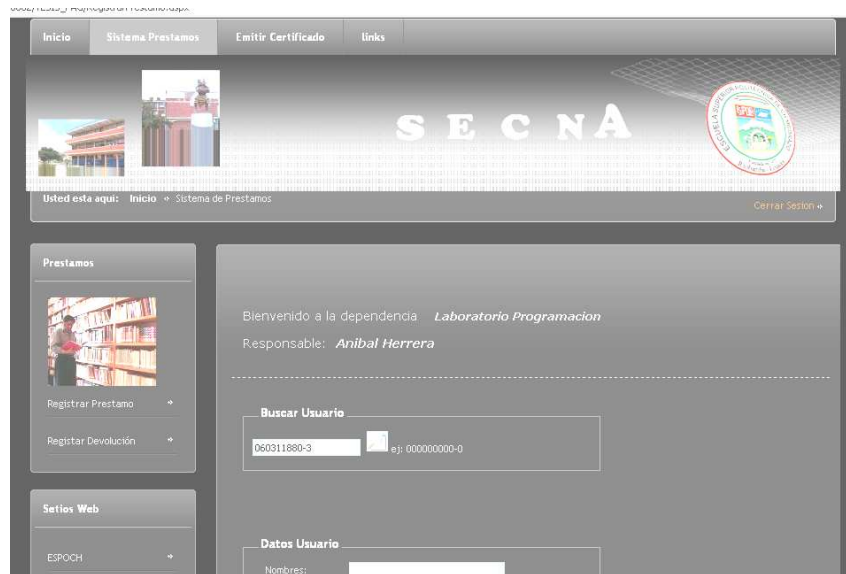
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
 ESCUELA DE INGENIERÍA EN SISTEMAS
 Telefax: 005009 ext 214 Robamba-Ecuador

LISTADO DE ARTÍCULOS POR DEPENDENCIA

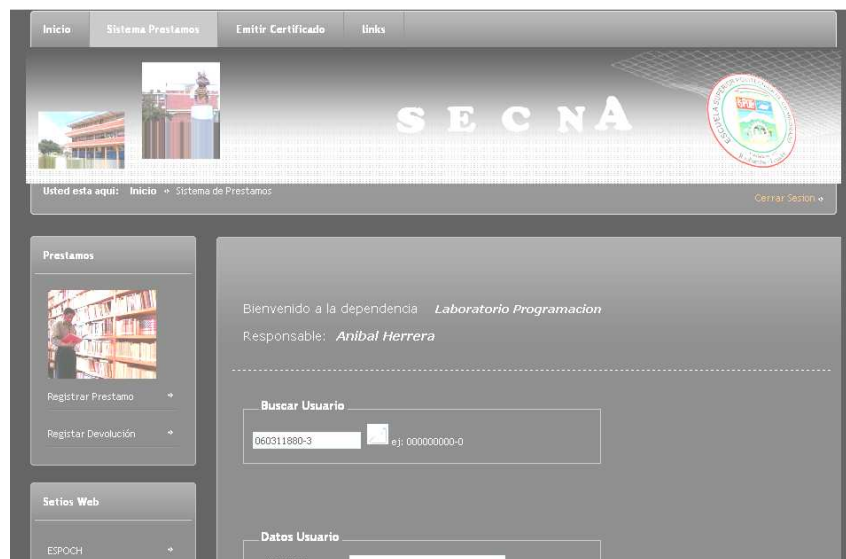
DEPENDENCIA	ARTICULOS	DESCRIPCION	STOCK
Laboratorio Investigacion	Libros	Psico Basica	2
Laboratorio	Libros	Psico Basica	2
Laboratorio Investigacion	Maquinas	Equipo de escritorio	4
Laboratorio	Maquinas	Equipo de escritorio	4
Laboratorio Investigacion	Balones	balones de Basket	6
Laboratorio	Balones	balones de Basket	6

3.2. Operaciones del Responsable

El usuario Responsable es encargado de realizar un nuevo préstamo y la devolución del mismo.



Al dar clic en el link Registrar préstamo, ingresar el número de cédula del estudiante



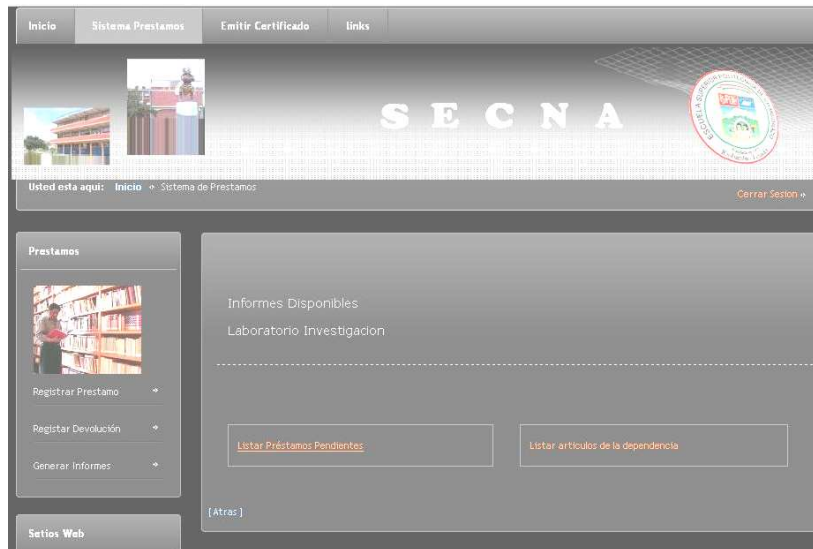
Se muestra dos datos del estudiante

The screenshot shows a web application interface with a dark grey background. On the left, there is a sidebar with a menu titled "Sitios Web" containing sub-items: "ESPOCH" (with sub-items "Webmail" and "E-Virtual"), "ACADEMIAS" (with sub-items "Linux", "Microsoft", and "Cisco"), and "Registrar Prestamo" and "Registrar Devolucion" at the top. The main content area is divided into several sections: "Buscar Usuario" with a search input field containing "060311880-3" and a "Buscar" button; "Datos Usuario" with input fields for "Nombres:" and "Apellidos:"; "Seleccionar Articulo" with a dropdown menu showing "Parlantes" selected, a "Nuevo" button, and a list of items including "Parlantes", "Infocus", "Windows 7", and "Windows XPsp3"; and "Datos Préstamo" with a "Fecha Préstamo:" input field.


Seleccionar el artículo si existe, caso contrario dar clic en el botón Nuevo para registrar un nuevo artículo, en los datos del préstamo el sistema pone la fecha actual y muestra la fecha en la cual tiene que ser entregado el artículo, dar clic en **Guardar**.

The screenshot shows a web application interface for article registration. On the left, there is a sidebar with a menu titled "Sitios Web" containing sub-items: "Microsoft" and "Cisco". The main content area is divided into several sections: "Seleccionar Articulo" with a dropdown menu showing "Parlantes" selected, a "Nuevo" button, and a list of items including "Parlantes", "Infocus", "Windows 7", and "Windows XPsp3"; "Datos Articulo" with input fields for "Nombre:", "Descripcion:", and "Stock:"; and "Datos Préstamo" with input fields for "Fecha Préstamo:" and "Fecha Devolución:", and a radio button for "Estado:" with "Prestado" selected.

Link **Generar Informes** muestra los reportes de los préstamos realizados en dicha dependencia.

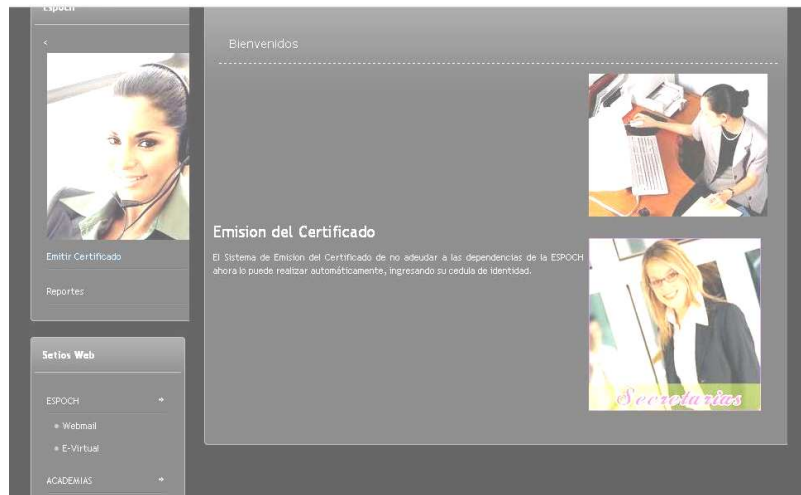


El responsable de la dependencia podrá visualizar los estudiantes que tengan los préstamos pendientes.

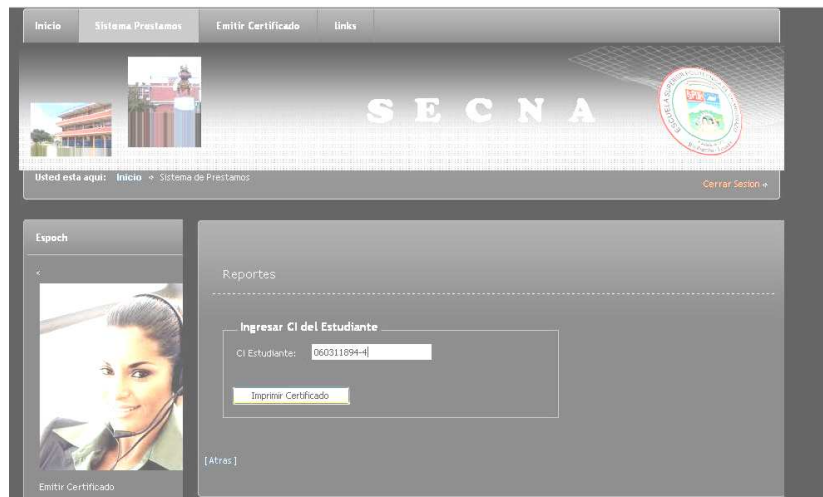
 ESPOCH ESCUELA SUPERIOR POLITECNICA DE CHIMBORAZO FACULTAD DE INFORMATICA Y ELECTRONICA ESCUELA DE INGENIERIA EN SISTEMAS Telefono 025600 ext 214 Riobamba-Ecuador		
LISTADO DE ARTICULOS		
DEPENDENCIA: Laboratorio Investigacion		
USUARIO: ALEX ALBERTO TACURI UQUILLAS		
NOMBRE y APELLIDO	F.PRESTAMO	F.DEVOLUCION
ANGEL PATRICIO FLORES OROZCO	23/01/2011 0:00:00	
RAUL HUMBERTO CUZCO NARANJO	25/01/2011 0:00:00	
RAUL HUMBERTO CUZCO NARANJO	25/01/2011 0:00:00	

3.3. Operaciones de la Secretaria

La secretaria es el usuario encargado de general el certificado



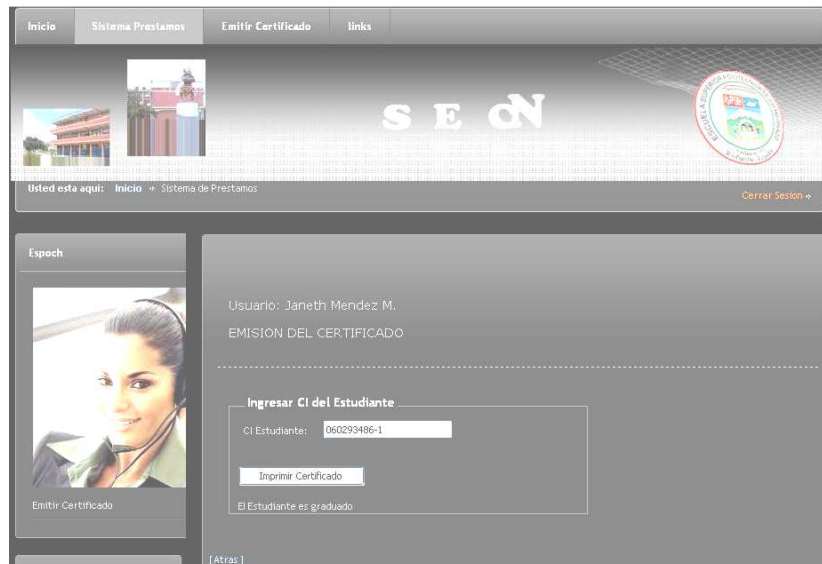
Dar clic en el enlace emitir certificado



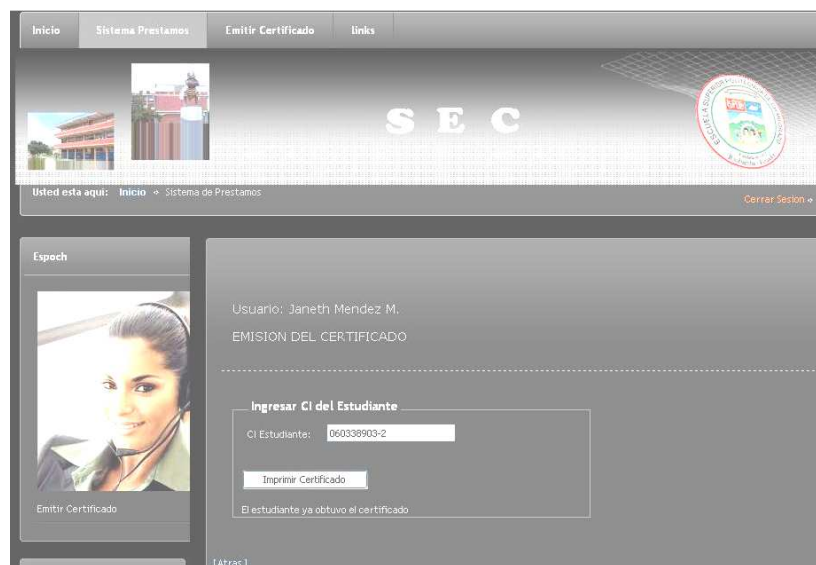
Dar clic en Imprimir Certificado



El estudiante que haya sido incorporado y quiera adquirir el certificado le muestra el siguiente mensaje

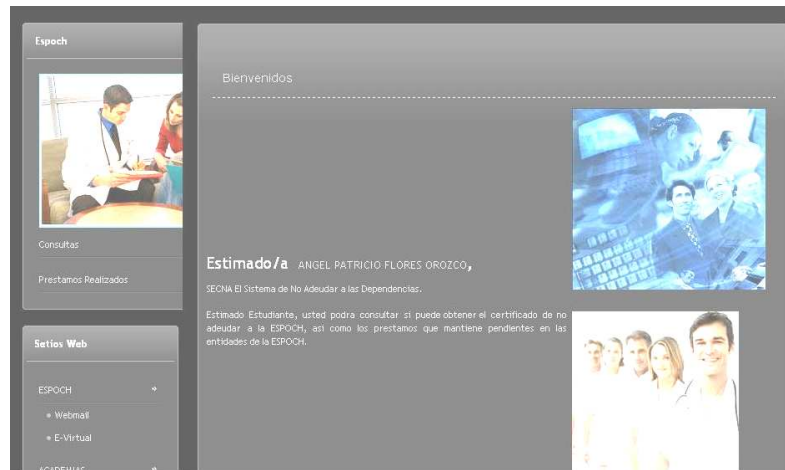


Si el certificado fue emitido se muestra el siguiente mensaje

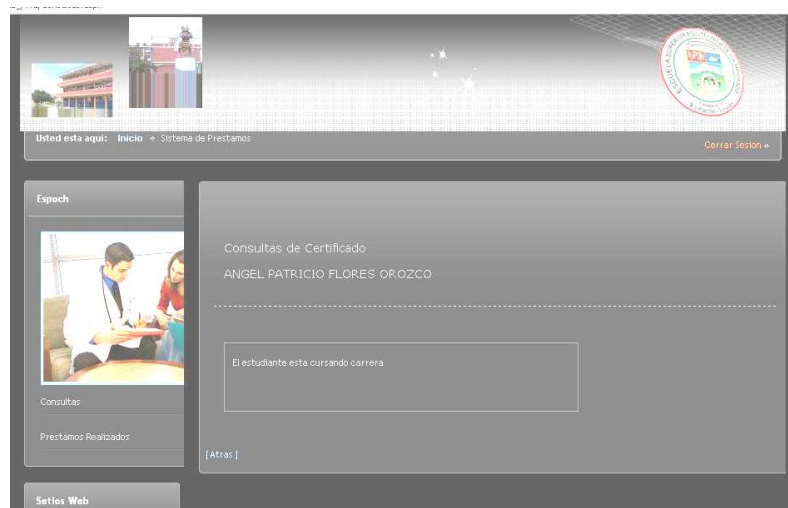


3.4. Operaciones del Estudiante

El usuario estudiante puede consultar sus préstamos pendientes



En el link consultas puede saber si puede o no obtener el certificado



En el link Préstamos realizados muestra los préstamos que ha realizado el estudiante.

ESPOCH
 ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
 FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
 ESCUELA DE INGENIERÍA EN SISTEMAS
 Telefax 055529 ext 214 Riobamba-Ecuador

LISTADO DE PRÉSTAMOS DEL ESTUDIANTE

ANGEL PATRICIO FLORES OROZCO

FECHA PRÉSTAMO	FECHA DEVOLUCION	DEPENDENCIA
15/01/2011 0:00:00	15/01/2011 0:00:00	Laboratorio Investigacion
16/01/2011 0:00:00	17/01/2011 0:00:00	Laboratorio Investigacion

El estudiante no tiene préstamos en las bibliotecas

INSTALACIÓN DE CODEGEAR RAD STUDIO 2007

SECNA

Sistema de Emisión de Certificados de No Adeudar

INSTALACIÓN CODEGEAR RAD STUDIO 2007

Una vez insertado el DVD escoger la primera opción **Install CodeGear RAD Studio**.



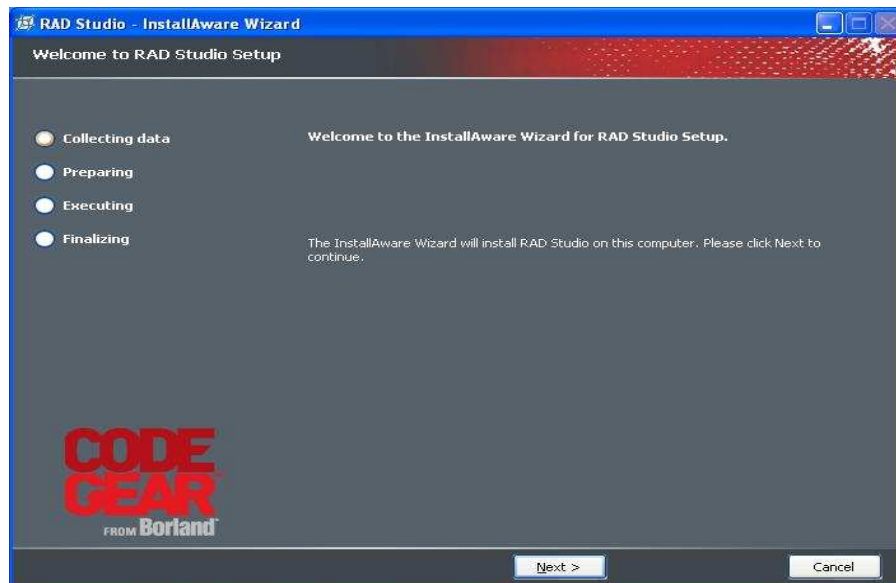
CD de Instalación RAD Studio

Seleccione el idioma como se muestra en la siguiente figura



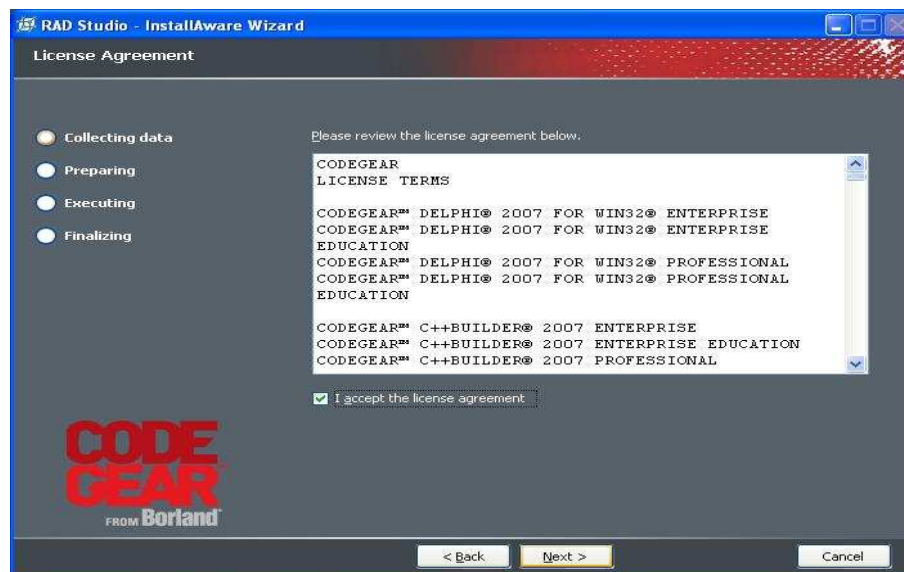
Seleccionar Idioma

A continuación se muestra una pantalla de bienvenida de la instalación Wizard para RAD Studio.



Ventana de bienvenida a la Instalación

Seleccionar la opción aceptar términos de licencia y dar clic en **Next**.

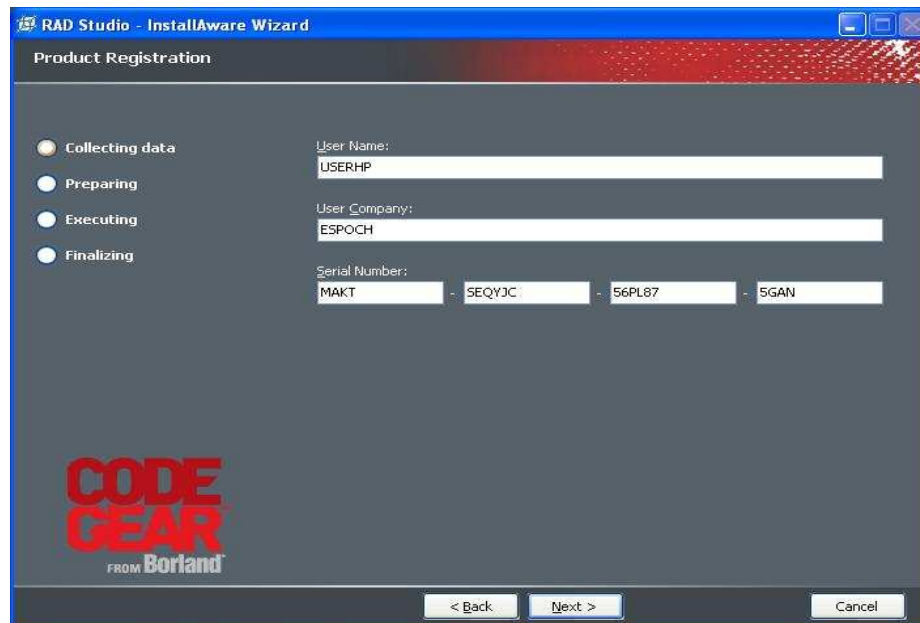


Ventana aceptación de Licencia

Para continuar la instalación necesita un número de serie, lo cual puede usarlo el número de serie que se muestra a continuación, y hacer clic en Next.

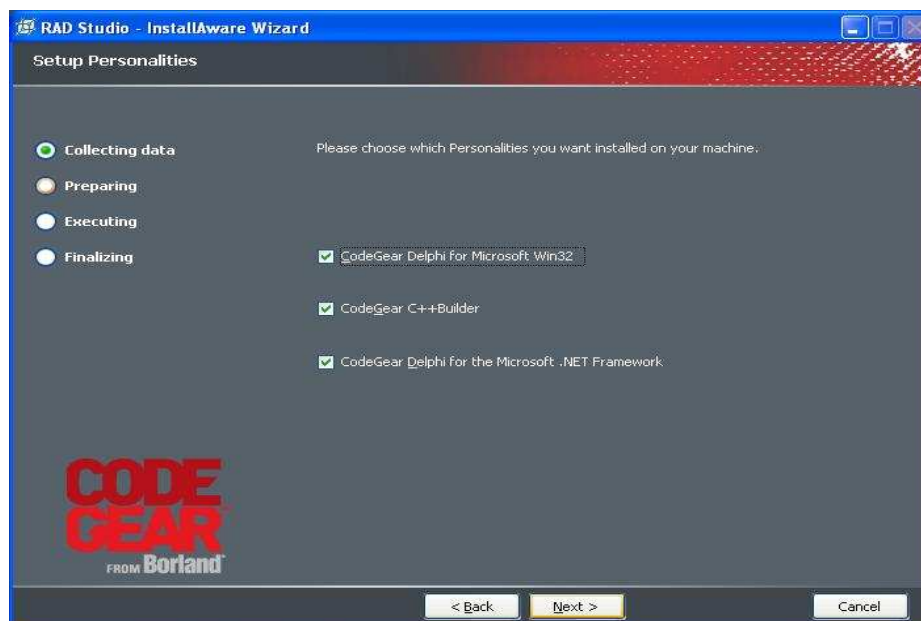
Serial Number:

MAKT-SEQYJC-56PL87-5GAN



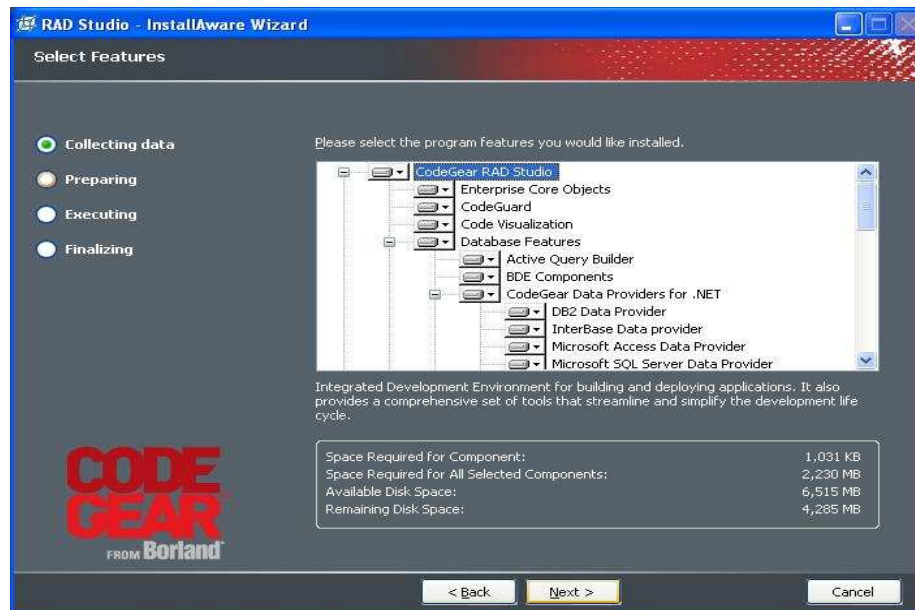
Ventana de registrar serial

Se muestra una selección de opciones que debe elegir para instalar en su máquina, nuestro caso se seleccionará las tres opciones como se muestra en la siguiente figura y damos clic en **Next**.



Seleccionar opciones de instalación

Seleccionar el programa que se desea instalar, elegir la aplicación CodeGear RAD Studio, clic en **Next** para continuar con la instalación.



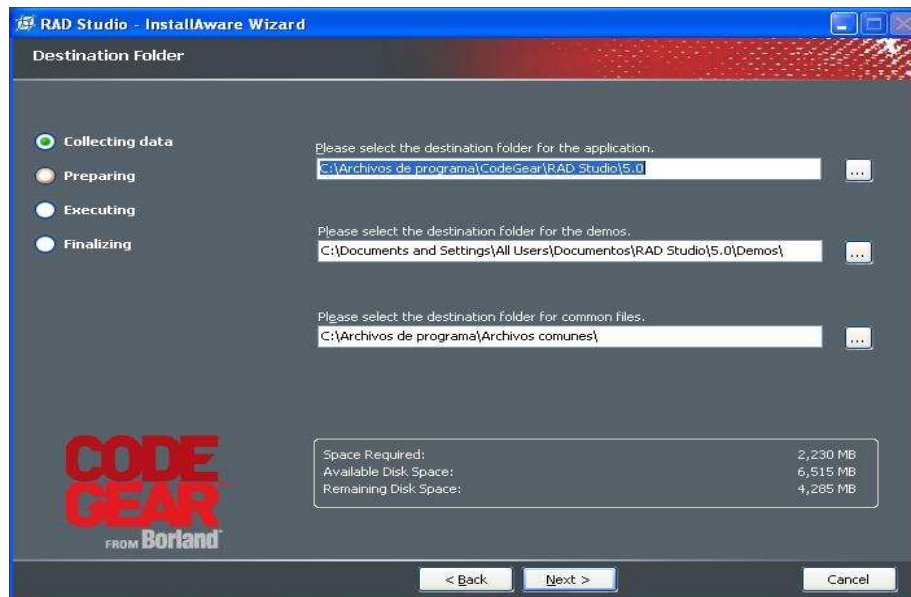
Seleccionar Ubicación de instalación

Seleccionar las opciones de componentes extras que se quiere instalar, **Rave Reports** y **Microsoft ASP.NET**, dar clic en **Next**.



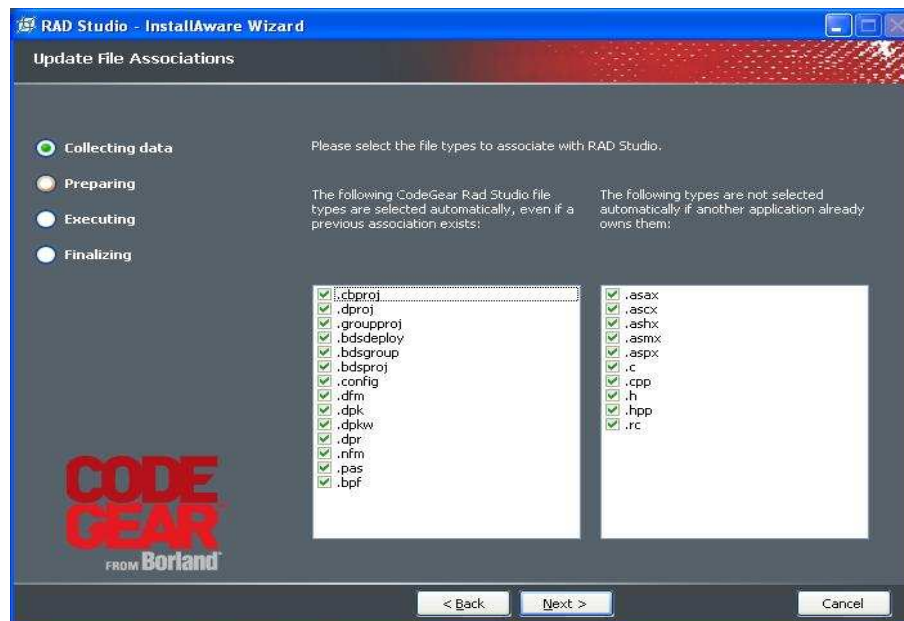
Seleccionar opciones extra de instalación

Elegir las carpetas de destino de la aplicación, de los demos y de los archivos comunes, **Next** para continuar la instalación.



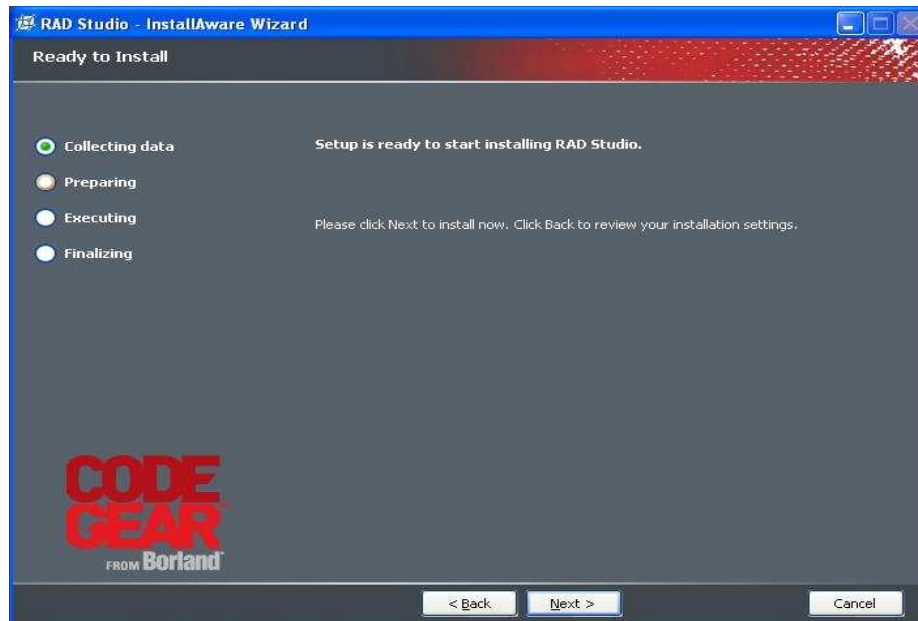
Ventana de designación de carpetas

Elegir los tipos de archivos asociados con RAD Studio, dar clic en **Next**.



Seleccionar tipos de archivos

Clic en **Next** para iniciar la instalación.



Inicio de Instalación

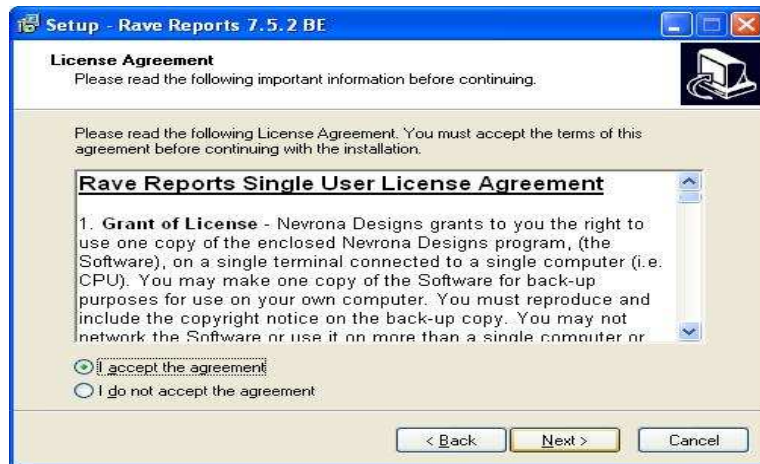
La siguiente pantalla muestra el proceso de instalación.



Proceso de Instalación

A continuación se muestra el asistente de instalación de **Rave Reports**, para continuar la instalación de CodeGear Studio.

Seleccionar **aceptar términos de licencia**.



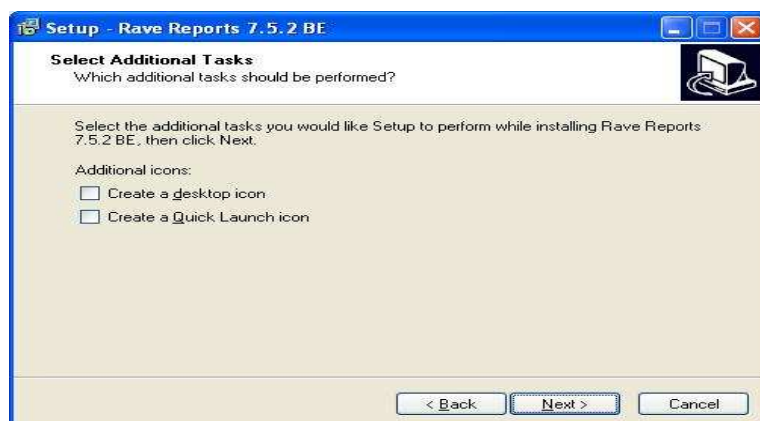
Instalación de Rave Reports 7.5.2 BE

Clic en **Next** para continuar instalando o clic en **Browser** si desea elegir una carpeta diferente en donde residen las aplicaciones.



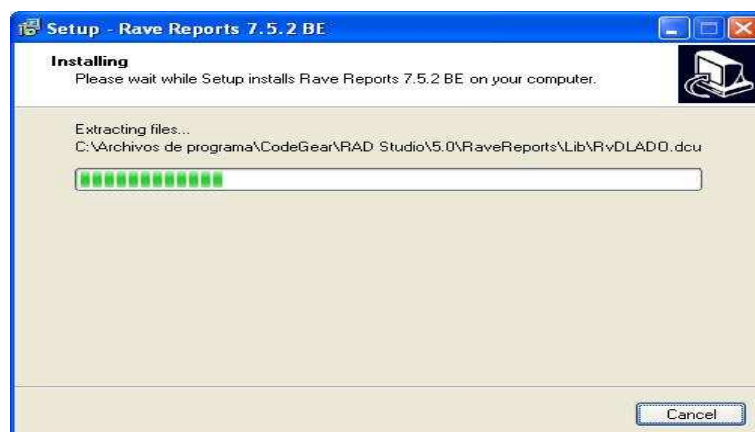
Seleccionar carpeta de inicio

Next para continuar instalando o seleccionar tareas adicionales como crear un ícono en el escritorio o crear un ícono rápido en la lista de programas.



Seleccionar tareas adicionales

Proceso de instalación **Rave Reports**.



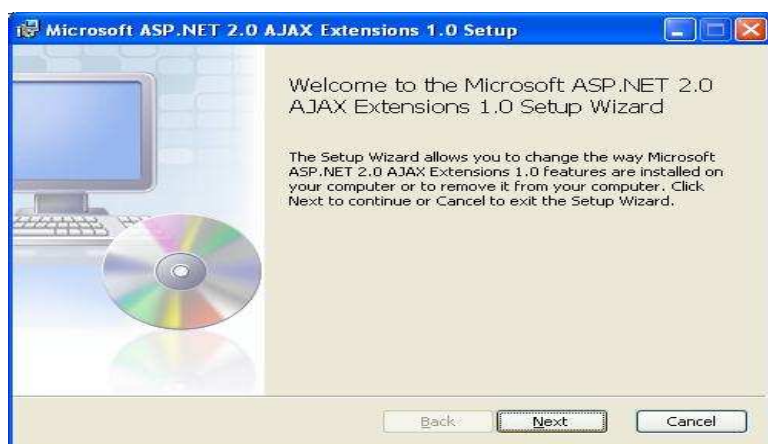
Ventana de proceso de instalación

Clic en **Finish** para culminar el proceso de Rave Reports.



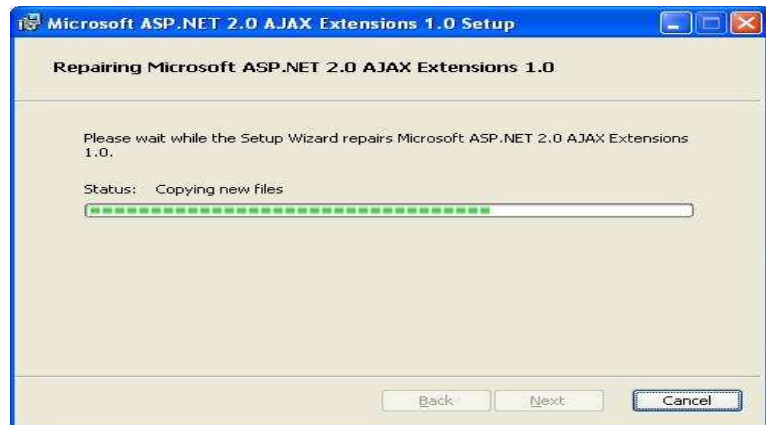
Finalizar la instalación

Seguidamente se visualiza el asistente de **Microsoft ASP.NET**, clic en **Next**.



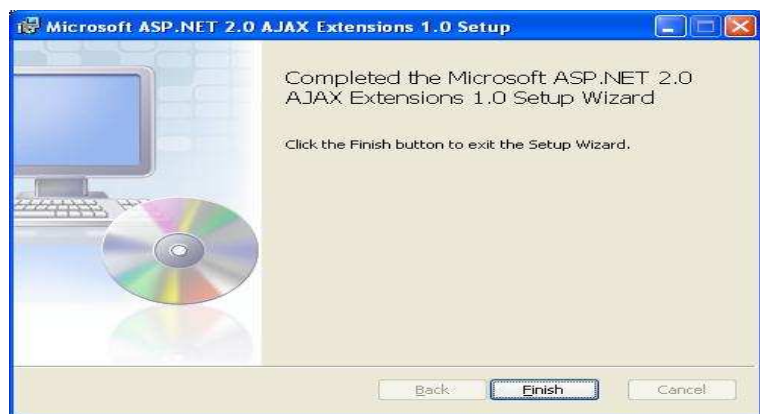
Asistente de instalación de ASP.NET

Se muestra el proceso de instalación de **Microsoft ASP.NET 2.0 Extensión AJAX 1.0**.



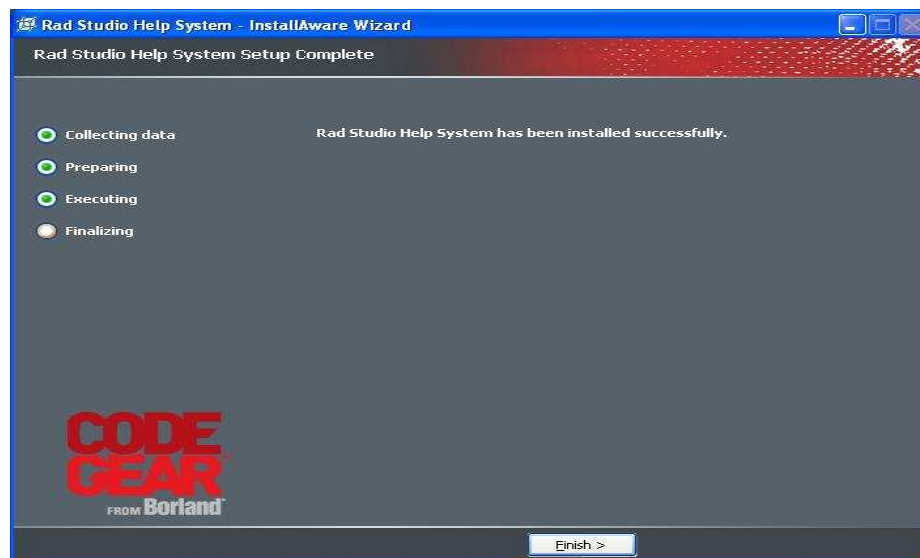
Proceso de instalación ASP.NET

Clic en **Finish** para completar el asistente de instalación.



Finalizar el asistente de instalación

Finalmente clic en **Finish** para completar el proceso de instalación

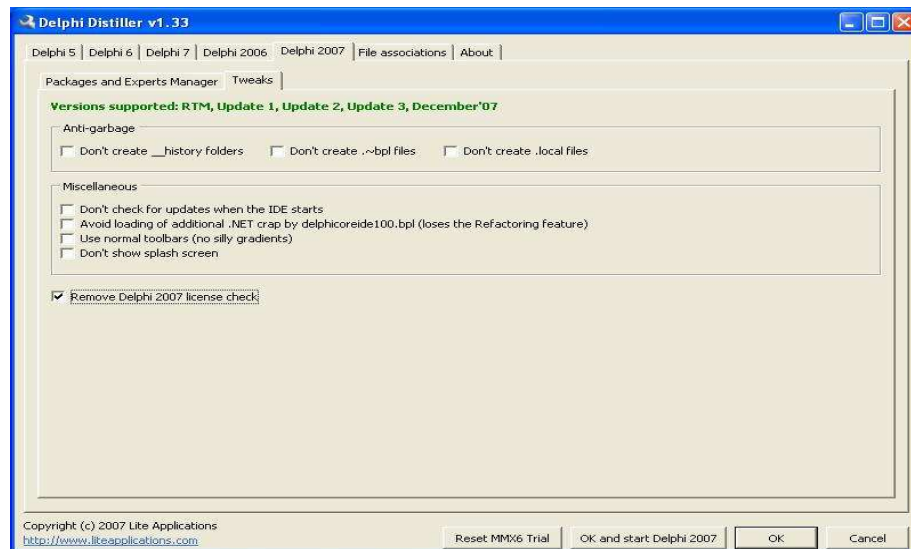


Finalizar Instalación RAD Studio

Después de instalar **CODEGEAR RAD Studio 2007** y antes de ejecutarlo por primera vez, correr **Delphi Distiller** para desactivar el chequeo de activación.

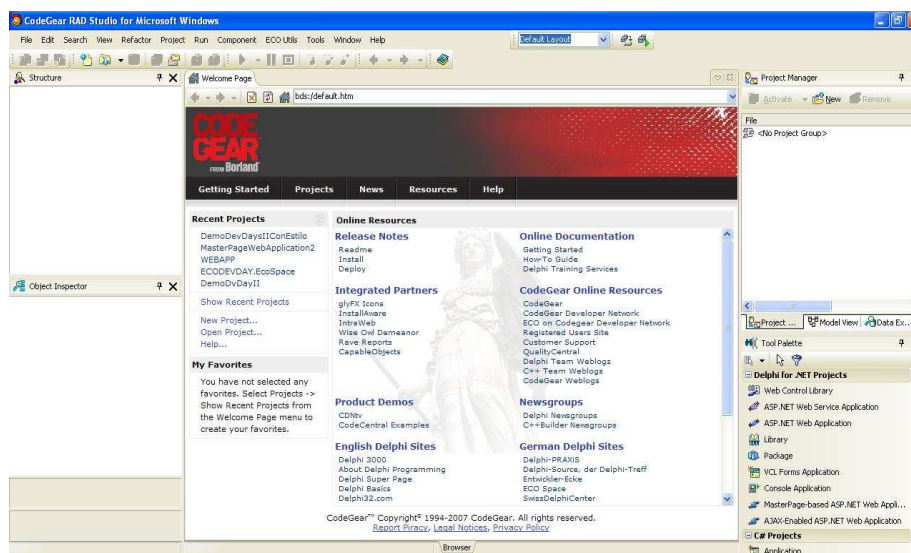
Delphi Distiller v1.33 released

Presionar **Ctrl+Alt+L** o la tecla **altgr +L** para que aparezca la opción **Removedelphi 2007 licensecheck** como se muestra en la siguiente ilustración.



Remove Licencia RAD Studio

Abrir la aplicación **CodeGear RAD Studio**, elegir la opción **delphi.net**, se muestra la interfaz de la herramienta de desarrollo.



Ventana inicial de RAD Studio