



Universidad de Valladolid

E. T. S. DE INGENIERÍA INFORMÁTICA

Ingeniería Técnica en Informática de Gestión

**Desarrollo y Configuración de una Línea de
Producto Software de Comercio Electrónico**

Alumnas: Esther Hernández Herrera

Mercedes García de Acilu Laá

Tutor: Miguel Ángel Laguna Serrano

Deseamos expresar nuestro más sincero agradecimiento

a nuestro tutor por guiarnos
en todo momento en este proyecto,

a nuestras familias por habernos apoyado durante todos
estos años y por ayudarnos a llegar hasta aquí,

a todos nuestros amigos y compañeros, en especial a O.,
con los que hemos compartido
la mayoría de nuestros mejores momentos y
sin los cuales no hubiésemos podido recorrer este largo camino,

y a Víctor y a Yonatan por estar siempre ahí.

Gracias a todos.

ÍNDICES

A Índice de Contenidos

ÍNDICES	5
<i>A Índice de Contenidos</i>	7
<i>B Índice de Figuras</i>	12
INTRODUCCIÓN	15
1 Presentación y objetivos del proyecto	17
1.1 Introducción	17
1.2 Ámbito del proyecto	17
1.3 Definición y objetivos del proyecto	18
1.4 Organización de la memoria	19
2 Líneas de Producto Software	21
2.1 La reutilización del software	21
2.2 Descripción general	22
2.2.1 Introducción.....	22
2.2.2 Beneficios.....	23
2.2.3 Desarrollo.....	24
2.2.4 Trazabilidad de requisitos.....	25
2.3 Modelo de Características para tratar la variabilidad	25
2.3.1 Definición.....	25
2.3.2 Problemas del modelo de características.....	26
2.4 Trazabilidad	26
2.4.1 Problemas.....	26
2.4.2 Soluciones aportadas para gestionar la variabilidad.....	27
2.5 Concepto de Paquetes. “Package Merge”, solución en el diseño	28
2.6 Clases parciales, solución en la implementación.	29
DESARROLLO DE LA SOLUCIÓN	31
3 Análisis del Sistema	33
3.1 Entrevistas	33
3.2 Modelo de características de una línea de comercio electrónico	33
3.3 Catálogo de Requisitos	35
3.3.1 Requisitos funcionales.....	35
3.3.1.1 Paquete: Registration.....	35
3.3.1.2 Paquete: Credit Card Information.....	37
3.3.1.3 Paquete: Shipping Address Information.....	37

3.3.1.4	Paquete: Billing Address Information	38
3.3.1.5	Paquete: Quick Checkout Profile.....	38
3.3.2	Requisitos no funcionales	39
3.4	Modelo de Casos de Uso	41
3.4.1	Actores	41
3.4.2	Diagrama de Casos de Uso	41
3.4.3	Especificación de Casos de Uso	43
3.4.3.1	Paquete: Registration	43
3.4.3.2	Paquete: Credit Card Information.....	48
3.4.3.3	Paquete: Shipping Address.....	50
3.4.3.4	Paquete: Billing Address.....	51
3.4.3.5	Paquete: Quick Checkout Profile.....	52
3.5	Diagramas de Estados y de Secuencia	54
3.5.1	Paquete: Registration	54
3.5.1.1	CU - 0001: Registrarse	54
3.5.1.2	CU-0002: Identificarse	57
3.5.1.3	CU-0003: Modificar datos.....	59
3.5.1.4	CU-0004: Darse de Baja.....	61
3.5.2	Paquete: Credit Card Information.....	63
3.5.2.1	CU-0005: Insertar Datos Tarjeta	63
3.5.2.2	CU-0006: Pago con Tarjeta	65
3.5.3	Paquete: Shipping Address	67
3.5.3.1	CU-0007: Insertar datos dirección destino	67
3.5.4	Paquete: Billing Address	69
3.5.4.1	CU-0008: Insertar datos dirección de facturación	69
3.5.5	Paquete: Quick CheckOut Profile.....	71
3.5.5.1	CU-0009: Utilizar Perfil Quick CheckOut	71
3.5.5.2	CU-0010: Modificar Perfil Quick CheckOut	73
3.6	Modelo de Dominio.....	74
4	<i>Diseño del Sistema</i>.....	77
4.1	Diagramas de Secuencia.....	77
4.1.1	Paquete: Registration	77
4.1.1.1	CU - 0001: Registrarse	78
4.1.1.2	CU - 0002: Identificarse	79
4.1.1.3	CU - 0003: Modificar datos.....	79
4.1.1.4	CU - 0004: Darse de Baja.....	80
4.1.2	Paquete: Credit Card Information.....	81
4.1.2.1	CU - 0005: Insertar Datos Tarjeta	81
4.1.2.2	CU - 0006: Pago con Tarjeta	82
4.1.3	Paquete: Shipping Address	83
4.1.3.1	CU - 0007: Insertar datos dirección destino	83
4.1.4	Paquete: Billing Address	84

4.1.4.1	CU-0008: Insertar datos dirección de facturación.....	84
4.1.5	Paquete: Quick CheckOut Profile.....	85
4.1.5.1	CU-0009: Utilizar Perfil Quick CheckOut.....	85
4.1.5.2	CU-0010: Modificar Perfil Quick Checkout.....	86
4.2	Modelo de Diseño.....	87
4.3	Diagramas de Clases.....	88
4.3.1	Paquete: Registration.....	88
4.3.1.1	Clases.....	88
4.3.1.2	Especificación.....	88
4.3.2	Paquete CreditCard.....	90
4.3.2.1	Clases.....	90
4.3.2.2	Especificación.....	91
4.3.3	Paquete Shipping Address.....	93
4.3.3.1	Clases.....	93
4.3.3.2	Especificación.....	93
4.3.4	Paquete Billing Address.....	94
4.3.4.1	Clases.....	94
4.3.4.2	Especificación.....	95
4.3.5	Paquete Quick CheckOut Profile.....	96
4.3.5.1	Clases.....	96
4.3.5.2	Especificación.....	96
4.4	Diseño de la Base de Datos.....	97
4.4.1	Gestor Base de Datos: SQL Server.....	97
4.4.1.1	Ventajas.....	97
4.4.1.2	Desventajas.....	97
4.4.2	Modelo relacional.....	98
4.4.3	Descripción de tablas.....	99
5	<i>Implementación de la solución.....</i>	103
5.1	Introducción a ASP.NET y formularios Web.....	103
5.2	Clases parciales.....	103
5.3	Interfaz de usuario.....	104
5.3.1	Master.Page.....	104
5.3.1.1	Introducción.....	104
5.3.1.2	Diseño de Master.Page.....	105
5.3.2	Hoja de estilos CSS.....	106
5.3.2.1	Introducción.....	106
5.3.2.2	Beneficios de utilizar CSS.....	107
5.3.2.3	Diseño de hojas de estilos.....	107
5.4	Solución a la variabilidad en la interfaz de usuario.....	108
5.4.1	Introducción a controles de usuario.....	108
5.4.2	Controles de Usuario dinámicos.....	109

5.4.3	Agregar controles de forma dinámica.....	111
5.5	Registro de usuarios	113
5.6	Envío de correos electrónicos	114
5.6.1	Protocolo SMTP	114
6	Pruebas.....	115
6.1	Introducción	115
6.2	Pruebas de Caja Blanca	115
6.3	Pruebas de Caja Negra.....	115
6.3.1	Partición Equivalente.....	116
6.3.2	Análisis de valores limites	118
6.4	Errores dentro de un ambiente Web.	118
6.5	Pruebas de implantación.....	118
6.6	Pruebas de aplicaciones Web.....	119
6.6.1	Prueba del contenido.....	119
6.6.2	Prueba de función	119
6.6.3	Prueba de la facilidad de uso	119
6.6.4	Prueba de navegabilidad	120
6.6.5	Prueba de compatibilidad	120
6.6.6	Pruebas de seguridad	120
7	Manuales.....	121
7.1	Manual de instalación.....	121
7.1.1	Instalación de Internet Information Server (IIS)	121
7.1.1.1	Activación de Certificado de Seguridad SSL	124
7.1.2	Instalación de SQL Server	125
7.1.3	Instalación del Framework .NET 2.0.....	126
7.1.4	Instalación de la Aplicación.....	127
7.2	Manual de usuario	130
7.2.1	Paquete Registration	130
7.2.2	Paquete Credit Card	140
7.2.3	Paquete Billing Address	146
7.2.4	Paquete Shipping Address	146
7.2.5	Paquete Quick Checkout.....	147
8	Configuración del producto final.....	151
8.1	Introducción	151
8.2	Análisis.....	151
8.2.1	Documento de requisitos	151
8.2.1.1	Requisitos funcionales.....	151
8.2.1.2	Requisitos no funcionales.....	152

8.2.2	Modelo de casos de uso	153
8.2.2.1	Actores.....	153
8.2.2.2	Diagrama de casos de uso.....	153
8.2.2.3	Especificación casos de uso.....	154
8.2.2.4	Diagrama de secuencia	155
8.3	Diseño	156
8.3.1	Diagrama de clases.....	156
8.3.2	Diagramas de secuencia	157
8.3.3	Diseño de la base de datos	157
8.3	Implementación de la solución.....	158
8.3.1	Configuración en Visual Studio. Archivo .csproj.....	158
8.3.2	Proyecto de configuración	159
8.4	Pruebas.....	161
8.4.1	Pruebas de caja blanca	161
8.4.2	Pruebas de caja negra.....	161
8.5	Manual	161
8.6	Conclusiones del capítulo	164

CONCLUSIONES.....	165
--------------------------	------------

9 Conclusiones	167
9.1 Resumen y valoraciones del trabajo.....	167
9.2 Problemas y dificultades.....	168
9.3 Trabajos futuros.....	168

BIBLIOGRAFÍA.....	171
--------------------------	------------

Referencias Bibliográficas.....	173
Fuentes Web.....	175

APÉNDICES.....	177
-----------------------	------------

Apéndice A .NET.....	179
-----------------------------	------------

Apéndice B Herramientas utilizadas	185
---	------------

Apéndice C. Contenido del CD-Rom.....	187
--	------------

B Índice de Figuras

Figura 2-1 Proceso de reutilización del software	21
Figura 2-2 Conceptos básicos de una Línea de Productos Software	22
Figura 2-3 Ingenierías de Dominio y de Aplicación	24
Figura 2-4 Desarrollo de una Línea de Producto Software	25
Figura 2-5 Notación en el modelo de características.....	27
Figura 2-6 Ejemplo del mecanismo de "package merge"	28
Figura 2-7 Clases parciales en distintos paquetes.....	30
Figura 3-1 Modelo de características de comercio electrónico	34
Figura 3-2 Diagrama de paquetes	34
Figura 3-3 Interacción Paquetes – Casos de Uso.....	42
Figura 3-4 Diagrama de Casos de Uso del paquete Registration	43
Figura 3-5 Diagrama de Casos de Uso del paquete Credit Card	48
Figura 3-6 Diagrama de Casos de Uso del paquete Shipping Address	50
Figura 3-7 Diagrama de Casos de Uso del paquete Billing Address.....	51
Figura 3-8 Diagrama de Casos de Uso del paquete Quick Checkout.....	52
Figura 3-9 Diagrama de Estados del CU – 0001 Registrarse	55
Figura 3-10 Diagrama de Secuencia del CU – 0001 Registrarse	56
Figura 3-11 Diagrama de Estados del CU – 0002 Identificarse	57
Figura 3-12 Diagrama de Secuencia del CU – 0002 Identificarse	58
Figura 3-13 Diagrama de Estados del CU – 0003 Modificar Datos.....	59
Figura 3-14 Diagrama de Secuencia del CU – 0003 Modificar Datos	60
Figura 3-15 Diagrama de Estados del CU – 0004 Darse de Baja.....	61
Figura 3-16 Diagrama de Secuencia del CU – 0004 Darse de Baja.....	62
Figura 3-17 Diagrama de Estados del CU – 0005 Insertar Datos Tarjeta	63
Figura 3-18 Diagrama de Secuencia del CU – 0005 Insertar Datos Tarjeta	64
Figura 3-19 Diagrama de Estados del CU – 0006 Pago con Tarjeta.....	65
Figura 3-20 Diagrama de Secuencia del CU – 0006 Pago con Tarjeta	66
Figura 3-21 Diagrama de Estados del CU – 0007 Insertar Datos Envío	67
Figura 3-22 Diagrama de Secuencia del CU – 0007 Insertar Datos Envío	68
Figura 3-23 Diagrama de Estados del CU – 0008 Insertar Datos Facturación.....	69
Figura 3-24 Diagrama de Secuencia del CU – 0008 Insertar Datos Facturación...	70
Figura 3-25 Diagrama de Estados del CU – 0009 Utilizar Perfil Quick Checkout	71
Figura 3-26 Diagrama de Secuencia del CU – 0009 Utilizar Perfil Quick Checkout	72
Figura 3-27 Diagrama de Estados del CU – 0010 Modificar Perfil Quick Checkout	73

Figura 3-28 Diagrama de Secuencia del CU – 0010 Modificar Perfil Quick Checkout	74
Figura 3-29 Modelo de Dominio	75
Figura 4-1 Diagrama de Secuencia del CU – 0001 Registrarse.....	78
Figura 4-2 Diagrama de Secuencia del CU – 0002 Identificarse.....	79
Figura 4-3 Diagrama de Secuencia del CU – 0003 Modificar Datos	79
Figura 4-4 Diagrama de Secuencia del CU – 0004 Darse de Baja	80
Figura 4-5 Diagrama de Secuencia del CU – 0005 Insertar Datos Tarjeta.....	81
Figura 4-7 Diagrama de Secuencia del CU – 0007 Insertar Datos Envío.....	83
Figura 4-8 Diagrama de Secuencia del CU – 0008 Insertar Datos Facturación	84
Figura 4-9 Diagrama de Secuencia del CU – 0009 Utilizar Perfil Quick Checkout	85
Figura 4-10 Diagrama de Secuencia del CU – 0010 Modificar Perfil Quick Checkout	86
Figura 4-11 Modelo de Diseño	87
Figura 4-12 Diagrama de Clases del paquete Registration	88
Figura 4-13 Diagrama de Clases del paquete Credit Card.....	90
Figura 4-14 Diagrama de Clases del paquete Shipping Address	93
Figura 4-15 Diagrama de Clases del paquete Billing Address	94
Figura 5-1 Diseño del sistema dividido en paquetes.....	104
Figura 5-2 Diseño de la Master.Page en el Visual Studio	105
Figura 5-3 Diseño de la Master.Page en el navegador.....	106
Figura 5-4 Ejemplo de la hoja de estilos.....	108
Figura 5-5 Producto final que incluye un único método de pago	109
Figura 5-6 Producto final que incluye varios métodos de pago.....	110
Figura 5-7 Herramienta de configuración de ASP. NET	113
Figura 7-1 Instalación de Internet Information Server (IIS).....	121
Figura 7-2 Selección de componente IIS	122
Figura 7-3 Creación del directorio virtual.....	123
Figura 7-4 Ventana emergente para la comprobación de la instalación	124
Figura 7-6 Página oficial de Microsoft, descarga de SQL Server	126
Figura 7-7 Página oficial de Microsoft, descarga de .NET Framework	127
Figura 7-8 Asistente para la instalación de la LPS_ecommerce	128
Figura 7-9 Selección de la ubicación Web para la LPS_ecommerce	128
Figura 7-10 Instalación de la LPS_ecommerce completada.....	129
Figura 7-11 Configuración del Proyecto para el usuario	130
Figura 7-12 Pantalla inicial, incluyendo el paquete Registration	131
Figura 7-13 Datos personales.....	132

Figura 7-14 Registro completado	133
Figura 7-15 Inicio de sesión	134
Figura 7-16 Opciones de usuario	135
Figura 7-17 Modificar datos personales	136
Figura 7-18 Baja de usuario.....	137
Figura 7-19 Ver producto	138
Figura 7-20 Carrito de la compra con una única forma de pago, Paypal	139
Figura 7-21 Pago con PayPal.....	140
Figura 7-22 Insertar datos de tarjeta	141
Figura 7-23 Carrito de la compra con dos formas de pago, Paypal y Tarjeta de crédito	142
Figura 7-24 Pago con tarjeta de crédito.....	143
Figura 7-25 Confirmación del pago con tarjeta de crédito.....	144
Figura 7-26 Confirmación del pedido realizado.....	145
Figura 7-27 Insertar dirección de facturación.....	146
Figura 7-28 Insertar dirección de envío.....	147
Figura 7-29 Carrito de la compra con opción de pago por método rápido.....	148
Figura 7-30 Modificación de datos del perfil Quick Checkout.....	149
Figura 8-1 Diagrama de casos de uso	153
Figura 8-2 Diagrama de secuencia CU-0015 Seleccionar paquetes.....	155
Figura 8-3 Diagrama de clases de la Configuración.....	156
Figura 8-4 Diagrama de secuencia CU-0015 Seleccionar paquetes.....	157
Figura 8-5 Configuración	162
Figura 8-6 Modificar .csproj.....	163
Figura 8-7 Crear proyecto.....	163
Figura A-1 El Framework de .NET	180
Figura A-2 Biblioteca de clases de .NET	181
Figura A-3 Funcionamiento de .NET Framework.....	182

INTRODUCCIÓN

1 Presentación y objetivos del proyecto

1.1 Introducción

En los últimos años, el desarrollo de software se está convirtiendo en una actividad cada vez más compleja. Esto es debido a que las necesidades de los usuarios han ido creciendo y cada día se demanda más cantidad de software, más eficiente y de mayor calidad. De estos retos nació la idea de la reutilización del software, gracias a la cual se ahorra tiempo y esfuerzo a la hora de emprender nuevos proyectos, ya que éstos se basan en la utilización de soluciones ya conocidas. Se consigue además que aumenten la calidad, fiabilidad y eficiencia de los productos.

El problema al que la reutilización del software se enfrenta ahora, es el que se presenta cuando se quieren desarrollar diferentes versiones de un mismo producto destinadas cada una de ellas a un grupo diferenciado de clientes. Para que el desarrollo de múltiples variaciones de un producto software sea eficiente, hay que ser capaces de solventar los problemas derivados de la evolución y personalización de las diferentes versiones.

Así es como surgen las Líneas de Producto Software (LPS) que permiten la reutilización sistemática en los casos en los que se tienen familias de productos, es decir, aplicaciones similares pero diferenciadas por algunas características. El objetivo es sacar el máximo partido de los elementos comunes, y gestionar de una manera eficaz las variaciones.

Una Línea de Productos Software es un grupo de productos que comparten un conjunto de características comunes, y que individualmente ofrecen características propias que los diferencian del resto de productos de su misma familia. Cada uno de ellos está orientado a satisfacer las necesidades específicas de un segmento de mercado particular. De este modo se promueve la industrialización del desarrollo software.

El Grupo de Investigación en Reutilización y Orientación al Objeto (GIRO) de la Universidad de Valladolid ha propuesto una serie de proyectos en el marco de la reutilización sistemática del software. Más concretamente, se han realizado proyectos sobre Líneas de Producto y herramientas de Ingeniería Dirigida por Modelos (MDE), orientados al desarrollo de Líneas de Producto dentro del dominio de transacciones electrónicas.

Lo que se ha perseguido con el proyecto actual es la idea de que, basándose en un prototipo de un producto software ya existente, se puedan desarrollar una serie de productos similares en la misma línea que el inicial, reutilizando lo ya existente y añadiendo y combinando de diferentes maneras los nuevos componentes.

1.2 Ámbito del proyecto

La forma más habitual de gestionar la variabilidad, es decir, de representar la parte común y la parte variable de una línea de productos software, es mediante modelos de características o “features”. Estos modelos permiten además seleccionar la configuración de cada aplicación concreta dentro de la línea de productos.

Con el reto al que nos enfrentamos ahora, es cómo manejar la trazabilidad entre los modelos de características antes citados y los modelos de diseño, estos últimos generalmente basados en UML. La propuesta consiste en utilizar el mecanismo de combinación de paquetes (“package merge”) de UML 2 como herramienta de representación y configuración de la variabilidad de la línea de productos y reservar los mecanismos clásicos de modelado (la especialización de los modelos estructurales o la relación <<extiende>> de los modelos de casos de uso) para expresar las variantes válidas en tiempo de ejecución de cada aplicación concreta. La estructura de los modelos de características se refleja directamente en las relaciones entre paquetes del modelo arquitectónico, de modo que la trazabilidad de las decisiones de configuración es automática. La forma de implementación que se ha elegido se basa en la utilización de clases parciales, disponibles en lenguajes como C#.

El grupo GIRO realizó una investigación para simplificar el paso de un modelo de desarrollo convencional a otro que aproveche las ventajas de las líneas de producto con buenos resultados, utilizando simplemente modelos estándar de UML 2 y modelos de características.

Como trabajo posterior, se decidió evaluar la escalabilidad de la solución aportada desarrollando Líneas de Producto Software de aplicación industrial, proponiendo el estudio y demostración práctica sobre la posibilidad del desarrollo de una Línea de Producto Software basada en sistemas web.

Por esta razón, el grupo de investigación GIRO decidió que sería interesante desarrollar una línea de producto de comercio electrónico como caso totalmente práctico y con la posibilidad de dar cabida a futuros proyectos de ampliación. Para ello, se propusieron una serie de proyectos orientados cada uno a desarrollar una parte de la línea de productos.

Los componentes básicos que constituyen esta línea de producto han sido desarrollados en un proyecto anterior al actual. Por lo tanto, lo que ahora nos ocupa es investigar y mostrar que dicha Línea de Producto puede seguir ampliándose desarrollando nuevos componentes de manera que al combinarse entre sí darán lugar a una serie de productos diferentes pero con una base común.

1.3 Definición y objetivos del proyecto

La idea de lo que se pretende con estos proyectos es buscar soluciones adecuadas en un entorno web a la hora de implementar una Línea de Productos Software, tomando como ejemplo práctico un portal de transacciones electrónicas vía web. El principal objetivo es demostrar que es factible desarrollar una línea de producto viable en una plataforma on-line a nivel industrial con el desarrollo del prototipo de un portal de ventas.

Basándonos en el proyecto realizado con anterioridad y en el cual se desarrollaba la parte común de la línea de producto (puntos obligatorios del modelo de características), en este PFC se añadirá un conjunto de paquetes opcionales que implementan el registro de un usuario como cliente. También introduciremos una diferenciación entre los tipo de productos, lo que provocará que dependiendo del tipo elegido, se tengan que incluir unas características u otras. Por lo tanto, algunos de los paquetes incluidos mostrarán ciertas dependencias entre sí.

Los objetivos de este PFC son los siguientes:

- ✓ Desarrollo de un prototipo de una Línea de Producto Software de comercio electrónico funcional por sí mismo.
- ✓ Desarrollo del conjunto de paquetes que implementan el registro de los clientes.
- ✓ Desarrollo de nuevas formas de pago, nuevos tipos de productos y otras características que se puedan añadir al proyecto ya existente.
- ✓ Integrar los nuevos paquetes con el núcleo de la Línea de Producto ya desarrollado.
- ✓ Conseguir suficiente número de productos finales diferentes para poder considerar que es una Línea de Producto como tal.

1.4 Organización de la memoria

El presente documento está estructurado en cinco grandes bloques: Introducción, Desarrollo de la Solución, Conclusiones, Bibliografía y Apéndices.

Este capítulo, titulado “*Presentación y objetivos del proyecto*” junto con el siguiente “*Líneas de Producto Software*” en el que se trata en profundidad el concepto de las LPS, forman la Introducción. En ambos capítulos se da una idea general de lo que es el proyecto realizado y de los conceptos teóricos en los que se basa.

El siguiente bloque es el Desarrollo de la Solución, que engloba los capítulos “*Análisis del sistema*”, “*Diseño del sistema*”, “*Implementación de la solución*”, “*Pruebas*” y “*Manuales*”. El último capítulo de este bloque, “*Configuración*” explica con detalle el desarrollo de la aplicación de configuración de la línea de producto. Este capítulo sigue la misma estructura que el resto de la memoria, incorporando su propia introducción, análisis, diseño... e incluso su manual de utilización.

Después del desarrollo, está el capítulo “*Conclusiones*”, en el que se exponen las valoraciones del trabajo y se detallan algunos de los problemas a los que hemos tenido que enfrentarnos, y las soluciones aportadas.

A continuación, “*Bibliografía*”, donde se nombran las referencias bibliográficas, libros, artículos y fuentes web que hemos consultado para documentarnos y buscar ayuda durante todo el proceso de desarrollo.

Para cerrar, hemos incluido el bloque “*Apéndices*”, que consta de tres apartados: el apéndice A “.NET”, en el que hacemos un resumen de la plataforma utilizada; el apéndice B “Herramientas”, donde describimos los programas con los que hemos trabajado; y por último el apéndice C “Contenido del CD-ROM”.

2 Líneas de Producto Software

2.1 La reutilización del software

El término reutilización vio la luz en una conferencia de la OTAN de 1968 sobre Ingeniería del Software . Fue propuesto por M.D. McIlroy como una de las principales y mejores soluciones para aumentar la productividad de los desarrolladores de software, así como para aumentar la fiabilidad de los productos software construidos. La idea básica respondía al principio de aprovechar esfuerzos previos para completar un nuevo desarrollo.

Actualmente, el concepto de reutilización ha evolucionado hacia la idea de que todo el conocimiento y productos derivados de la producción de software son susceptibles de ser reutilizados en la construcción de nuevos sistemas. Este enfoque se basa en la selección e integración de elementos del software que hayan sido intencionadamente diseñados, desarrollados y documentados para servir como materia prima para nuevos productos software. De esta forma aparece el concepto de *asset* o de *componente software reutilizable*, planteado por Karlsson en 1995.

En resumidas cuentas, se podría definir reutilización como “cualquier procedimiento que conduce o ayuda a producir un sistema mediante el nuevo uso de algún elemento procedente de un esfuerzo de desarrollo anterior”, según la definición de P. Freeman de 1987.

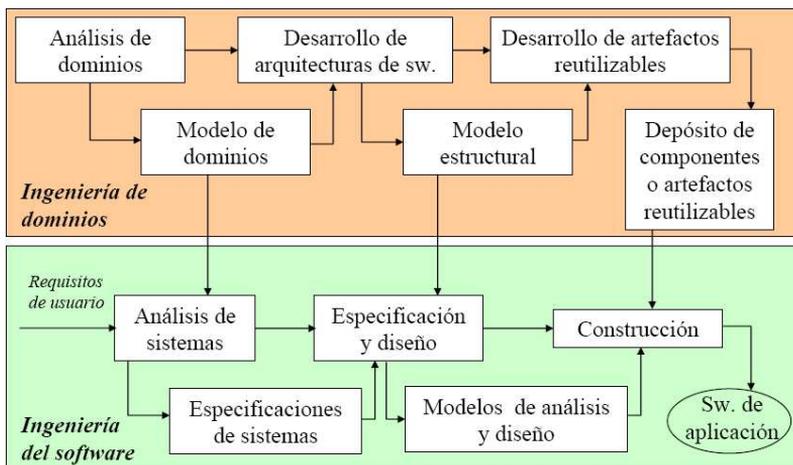


Figura 2-1 Proceso de reutilización del software

Aún hoy en día, la reutilización continúa siendo uno de los campos de mayor investigación dentro de la Ingeniería del Software. Esto se debe a que a pesar de sus avances y de ser la solución más realista para los problemas de productividad en el desarrollo, muchos autores afirman que no se han conseguido avances significativos para la utilización sistemática de la reutilización en el proceso de construcción de software.

INTRODUCCIÓN

Según otros autores, la reutilización externa, es decir, la que se lleva a cabo a la hora de reutilizar componentes software de terceros, ha sido un éxito. Algún ejemplo de este tipo sería la reutilización de sistemas operativos o de servidores de bases de datos.

Sin embargo, la reutilización interna desarrollada por las propias organizaciones no ha experimentado grandes avances, por lo que es un objetivo a conseguir. Para aumentar este tipo de reutilización, se han hecho intentos con las bibliotecas, la orientación a objetos, los componentes y últimamente con las Líneas de Producto Software.

2.2 Descripción general

2.2.1 Introducción

La idea básica de lo que es una Línea de Producto Software está inspirada en los procesos de producción industrializada de los productos físicos, tales como la producción de vehículos o hardware. Consiste en el ensamblaje de partes de software previamente elaboradas.

“Una Línea de Producto Software es un conjunto de sistemas de software que comparten un conjunto común y gestionado de aspectos que satisfacen las necesidades específicas de un segmento de mercado o misión y que son desarrollados a partir de un conjunto común de activos fundamentales de software de una manera preescrita” (Clements y Northrop; Software Products Lines: Practices and Patterns, 2001)

Las Líneas de Producto Software pueden ser descritas en términos de cuatro conceptos básicos, que están representados en la siguiente figura:

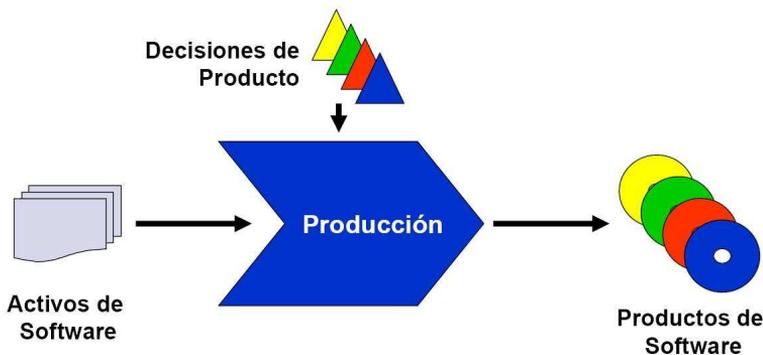


Figura 2-2 Conceptos básicos de una Línea de Productos Software

- Activos software reutilizables (*assets*): colección de componentes software, tales como requisitos, casos de uso, arquitectura o documentación, que pueden ser configurados y combinados de diferentes maneras para dar lugar a los distintos productos de una línea.

- Decisiones de Producto y Modelo de Decisión: cada uno de los productos pertenecientes a una línea se diferencia de los demás gracias a la elección de las características variables y opcionales que hay en el modelo de decisión. Es decir, quedan definidos por las decisiones de producto.
- Mecanismo y procesos de producción: los medios que permiten la configuración de productos a partir de activos software reutilizables. Durante este proceso, las decisiones de producto determinan cuáles serán los componentes que van a utilizarse y cómo configurar los puntos de variabilidad entre ellos.
- Productos de Software: colección de todos los productos que pueden crearse para la línea de productos. El alcance de la línea de productos está determinado por el conjunto de productos software que puede ser producido a partir de los activos reutilizables y del modelo de decisión

2.2.2 Beneficios

El beneficio principal de utilizar la tecnología de LPS es que la entrega de productos se hace de una manera más rápida y económica porque se reducen los costes de ingeniería y con una calidad mucho mayor ya que se reducen las tasas de errores.

Según Charles Krueger existen una serie de argumentos a favor de las Líneas de Producto, que se podrían dividir en beneficios tácticos y estratégicos.

Los beneficios tácticos serían los siguientes:

- Reducción en el tiempo medio de creación y entrega de nuevos productos.
- Reducción en el número medio de defectos por producto.
- Reducción en el esfuerzo medio requerido para desarrollar y mantener los productos.
- Reducción en el coste medio de producción.
- Incremento en el número total de productos que pueden ser desarrollados y mantenidos.

Por otro lado, los beneficios estratégicos de negocios serían:

- Reducción en el tiempo de entrega de nuevos productos.
- Mejoras en el valor competitivo del producto.
- Márgenes mayores de ganancias.
- Mejor calidad de los productos.
- Mejoras en la reputación de la empresa.
- Mayor escalabilidad del modelo de negocios en términos de productos y mercados.
- Mayor agilidad para expandir el negocio a nuevos mercados.
- Reducción de riesgos en la entrega de productos.

2.2.3 Desarrollo

El desarrollo de las Líneas de Producto Software introduce un cambio en la forma de crear software, ya que este trabajo se convierte en un proceso cada vez más industrializado. Lo que se pretende es reutilizar las partes que tienen en común todos los productos y desarrollar de forma eficiente y sistemática nuevos miembros de la familia simplemente con añadirle nuevas funcionalidades.

Las dos actividades clave para poder desarrollar una Línea de Producto son la Ingeniería de Dominio y la Ingeniería de Aplicación. Para exponer la relación existente entre ellas, lo mejor es definir las y relacionarlas entre sí, y al mismo tiempo, con el concepto de reutilización.

La Ingeniería de Dominio y la Ingeniería de Aplicación son complementarias, interactúan en procesos paralelos. El modelo de procesos basado en componentes incorpora explícitamente la reutilización del software en el proceso de desarrollo de aplicaciones. Este modelo considera la reutilización desde las perspectivas de las dos ingenierías:

- Desarrollo de software para la reutilización: el propósito es producir componentes de software reutilizables. A este proceso se le denomina Ingeniería de Dominio.
- Desarrollo de software con reutilización: su propósito es desarrollar software reutilizando componentes existentes. Este proceso se llama Ingeniería de Aplicación.

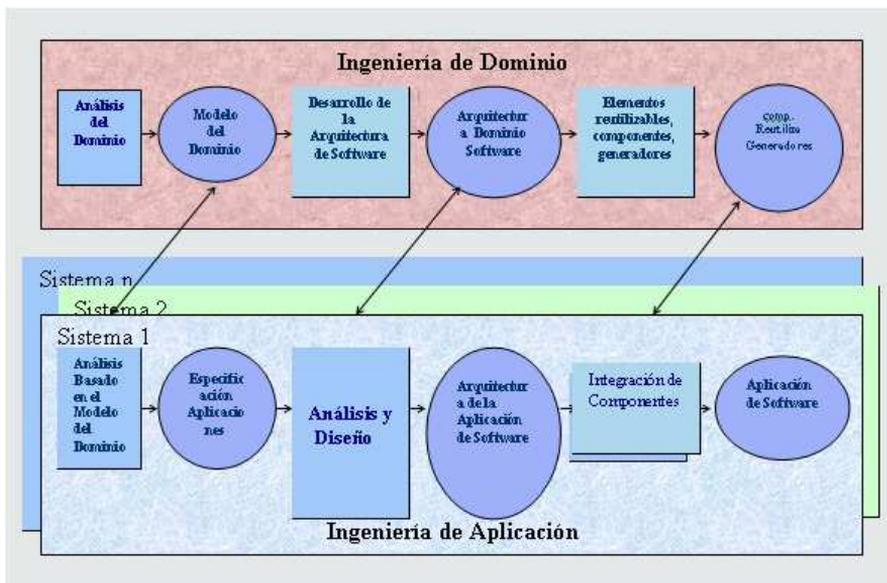


Figura 2-3 Ingenierías de Dominio y de Aplicación

La Ingeniería de Dominio se dirige a la creación sistemática de modelos de dominios, arquitecturas, componentes y artefactos de software reutilizables en el desarrollo de cualquier nuevo producto de una LPS.

La Ingeniería de Aplicación se basa en la reutilización de componentes existentes y en el conocimiento del dominio. Los productos de la LPS son construidos mediante el ensamblaje de activos de software.

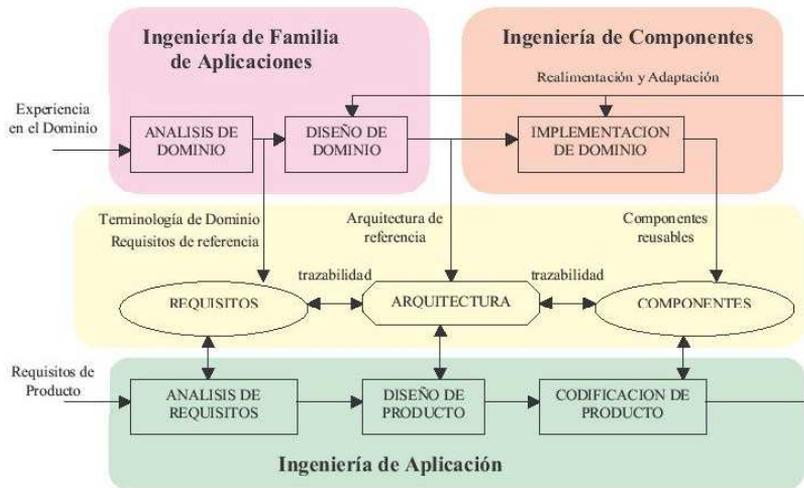


Figura 2-4 Desarrollo de una Línea de Producto Software

2.2.4 Trazabilidad de requisitos

La trazabilidad de requisitos es un reconocido factor de calidad en los procesos de desarrollo de software. Es la habilidad para seguir la vida de un requisito en ambos sentidos, hacia sus orígenes o hacia su implementación, a través de las explicaciones generadas en el proceso de desarrollo. Esta propiedad es clave para la consistencia entre los diferentes modelos y etapas del desarrollo, y en el mantenimiento de las Líneas de Producto Software.

2.3 Modelo de Características para tratar la variabilidad.

2.3.1 Definición

La variabilidad se define, según Svanhberg, como la habilidad de cambio o de personalización de un sistema. Para la definición de requisitos de una línea de productos, hay que prestar especial atención al análisis de la parte común y la parte variable, estableciendo las dependencias que existen entre ellas.

Para ello, los métodos más utilizados son los basados en *features*, como FODA (*Feature Oriented Domain Analysis*). Esta es una metodología desarrollada por el SEI (Software Engineering

Institute) para la aplicación del análisis de dominio, definiendo las etapas del método y los resultados obtenidos en cada una de ellas. El proceso se basa en identificar las características que los usuarios esperan comúnmente en las aplicaciones dentro de un dominio dado. El método FODA soporta el descubrimiento, análisis y la documentación de los aspectos comunes y las diferencias de un dominio.

Este método se basa en la utilización de árboles jerárquicos para organizar las capacidades de la línea de productos. Estas capacidades pueden ser obligatorias (comunes para todos los productos), opcionales o alternativas. Aquellas que son comunes van colocadas en los nodos raíz del árbol, mientras que las variantes representan las ramas. De este modo, en un único modelo se encuentran representadas todas las variaciones posibles de la línea de productos.

2.3.2 Problemas del modelo de características

Actualmente, los modelos de características presentan dos problemas:

- 1) Sólo se centran en aspectos funcionales de la línea, es decir, no han sido diseñados para representar información extra-funcional o de calidad.
- 2) El razonamiento automático propuesto sobre este tipo de modelos es muy limitado y en ningún caso tratan aspectos extra-funcionales.

En el enfoque de desarrollo de línea de productos, cada aplicación concreta se deriva del framework que implementa la arquitectura de la línea de productos. En este proceso, se deben seleccionar aquellas variantes que resultan apropiadas para los requisitos funcionales y no funcionales expresados por los usuarios. Esta actividad es esencialmente una selección de características efectuada por el ingeniero de aplicación, generando un sub-modelo de características, que a su vez (gracias a las relaciones de trazabilidad) generará por derivación toda o la mayor parte del código de la aplicación.

2.4 Trazabilidad

2.4.1 Problemas

La clave reside en la trazabilidad desde las metas/características hasta el código pasando por los modelos de diseño. Sin embargo, esta trazabilidad no es fácil de gestionar por varias razones.

En primer lugar, una característica opcional puede originar varios elementos en un modelo de diseño. Es decir, tenemos que asignar la relación de trazabilidad entre elementos de los dos niveles con una multiplicidad uno a varios, y lo mismo en sentido contrario, lo que complica rápidamente el modelo global haciendo que sea muy poco escalable.

El segundo problema tiene que ver con el hecho de que los mecanismos básicos de modelado de la variabilidad (la especialización en los diagramas de clases o la relación <<extend>> de los casos de uso) se utilizan en muchas ocasiones para expresar dos niveles de variabilidad distinta: el diseño de la arquitectura de la línea de productos (que se corresponde con requisitos opcionales) y el diseño

de una aplicación concreta, que sigue teniendo variaciones en tiempo de ejecución (por ejemplo, dos formas de pago alternativas).

En el ejemplo de una línea de productos de comercio electrónico, el mecanismo de extensión de los casos de uso permite mostrar distintas formas de pago. Ahora bien, en una solución concreta todas las formas de pago pueden ser válidas en tiempo de ejecución y por tanto estarán presentes. Pero es posible que otras aplicaciones de la línea de productos admitan sólo determinadas formas de pago. El problema está en que el mismo mecanismo sirve para mostrar variaciones en tiempo de ejecución y en tiempo de configuración.

Icon	Explanation
● F	Solitary feature <i>F</i> with feature cardinality [1..1] (i.e. mandatory feature)
○ G	Solitary feature <i>G</i> with feature cardinality [0..1] (i.e. optional feature)
○ [0..m] H	Solitary feature <i>H</i> with feature cardinality [0..m], $m > 1$ (i.e. optional cloneable feature)
● [m..n] J	Solitary feature <i>J</i> with feature cardinality [m..n], $m > 0 \wedge n > 1$ (i.e. mandatory cloneable feature)
■ K	Grouped feature <i>K</i> with feature group cardinality [0..1]
■ L	Grouped feature <i>L</i> with feature group cardinality [1..1]
○ M ('value' : T)	Feature <i>M</i> with attribute of type <i>T</i> and value of <i>value</i>
⊖	Feature group with group cardinality <1-1> (i.e. exclusive-or group)
⊕	Feature group with group cardinality <1-k>, $k = \text{group size}$ (i.e. inclusive-or group)
⊖ <i-j>	Feature group with group cardinality <i-j>

Figura 2-5 Notación en el modelo de características

Sin embargo, en el trabajo pionero de Jacobson, no hay diferencia fundamental entre la representación de la variabilidad en ambos niveles: por ejemplo la relación <<extend>> de los casos de uso es uno de los pilares con los que construye su método.

En resumen, este enfoque tiene como problema principal la falta de separación entre la variabilidad global de la línea de productos y la variabilidad residual de cada aplicación concreta.

2.4.2 Soluciones aportadas para gestionar la variabilidad

Las soluciones más recientes propuestas para mostrar la variabilidad de una línea de productos con UML pasan por modificar o anotar los modelos, tanto estructurales como funcionales o incluso dinámicos. De este modo se resuelve el segundo problema, pero raramente se ataca el primero (múltiples dependencias entre elementos UML y características).

La propuesta más reciente de Czarnecki y Antkiewicz, sí permite que cada característica opcional se refleje en una o varias partes de un diagrama, pero de nuevo es necesario introducir elementos auxiliares en el meta-modelo de UML.

INTRODUCCIÓN

Como resultado del estudio de los puntos fuertes y débiles de estas propuestas, podemos establecer un conjunto mínimo de requisitos que debe cumplir una técnica útil de representación, y gestión de la variabilidad en el nivel de diseño de una línea de productos software:

1. Localizar en un solo punto del modelo de diseño todas las variaciones que origina cada característica opcional, de forma que se mantenga una correspondencia uno a uno y se facilite la gestión de la trazabilidad.
2. Separar la variabilidad originada en el nivel de la línea de productos de la variabilidad intrínseca en el nivel de las aplicaciones concretas, eliminando ambigüedades.
3. Mantener inalterado el meta-modelo de UML para eliminar la barrera de entrada a este paradigma para cualquier desarrollador además de permitir el uso de herramientas CASE convencionales.
4. Conectar con los modelos de implementación para acercarnos al ideal de desarrollo sin costuras, propugnado por el paradigma de orientación a objetos pero desechado muchas veces por irrealizable

2.5 Concepto de Paquetes. “Package Merge”, solución en el diseño

Para la correcta gestión de la variabilidad, el grupo GIRO aboga por expresar la variabilidad en los modelos UML utilizando el concepto de combinación de paquetes (o “package merge”), presente en el metamodelo de infraestructura de UML 2 y utilizado de forma exhaustiva en la definición misma de UML 2. El mecanismo “package merge” consiste fundamentalmente en añadir detalles de forma incremental (ver Figura 2, extraída del documento oficial de UML 2).

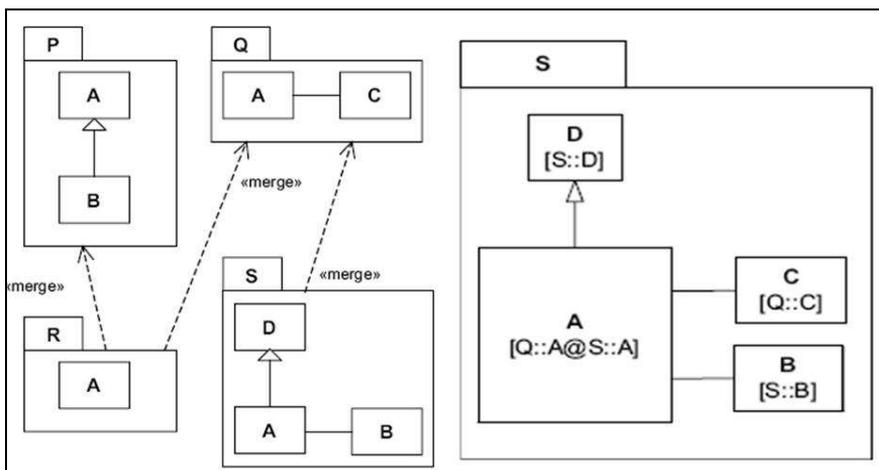


Figura 2-6 Ejemplo del mecanismo de "package merge"

Según la especificación UML 2, se define como una relación entre dos paquetes que indica que los contenidos de ambos se combinan. Es similar a la generalización y se utiliza cuando elementos en distintos paquetes tienen el mismo nombre y representan el mismo concepto, comenzando por una base común.

Dicho concepto se extiende incrementalmente en cada paquete añadido. Seleccionando los paquetes deseados es posible obtener una definición a la medida de entre todas las posibles. Aunque nos interesan sobre todo los diagramas de clases, el mecanismo se puede extender a cualquier diagrama de UML, en particular los de casos de uso y los de secuencia.

Evidentemente, las reglas que establece la especificación UML2 son muy estrictas para evitar inconsistencias. Por ejemplo, no puede haber ciclos, las multiplicidades resultantes son las menos restrictivas de entre las posibles, o las operaciones deben conformar en número, orden y tipo de parámetros.

Los conceptos que maneja UML 2 son fundamentalmente:

- Paquete combinable (primer operando), paquete receptor (segundo operando) y paquete resultado, que conceptualmente es el resultado de la combinación y que en el modelo coincide con el paquete receptor (interpretando que se observa después de ejecutada la operación).
- Elemento combinable (perteneciente al paquete combinable), elemento receptor (perteneciente al paquete receptor y que si presenta coincidencia con un elemento combinable se fusiona con él) y elemento resultado, que conceptualmente es el resultado de la combinación de dos elementos. Aquellos elementos que no coincidan en los dos paquetes que se combinan se incorporan sin modificaciones al paquete resultado.

La filosofía general es que el elemento resultante tiene que ser al menos tan capaz como el original antes de la combinación. Este mecanismo nos permite establecer una trazabilidad clara entre los modelos de características y los artefactos UML.

2.6 Clases parciales, solución en la implementación.

Para extender la trazabilidad hasta los modelos de implementación utilizamos el concepto de clases parciales. La utilización de “mixins” o clases parciales representa una alternativa a la herencia múltiple y una manera de manejar la variabilidad relacionada con aspectos. La intención es mantener la correspondencia uno a uno no sólo entre características y paquetes de diseño sino también con la estructura del código.

En este modelo de desarrollo del grupo GIRO se utiliza el mecanismo de clases parciales de C# para implementar la línea de productos dentro de la estructura de soluciones y paquetes de la plataforma MS Visual Studio 2005. Si el framework que implementa la arquitectura de la línea de productos está organizado en paquetes de clases parciales (un paquete base y tantos paquetes auxiliares como variaciones existen), para derivar una aplicación concreta basta con importar o referenciar los paquetes que se correspondan directamente con la configuración elegida en el modelo de características.

INTRODUCCIÓN

En la Figura 2-7 se aprecia una parte de la línea de producto, implementada utilizando paquetes de C#. Estos paquetes opcionales pueden añadirse o no al proyecto que representa una aplicación concreta de la línea de producto utilizando los ficheros de configuración del compilador. De esta manera se establece la trazabilidad uno-a-uno desde las metas/características hasta el código.

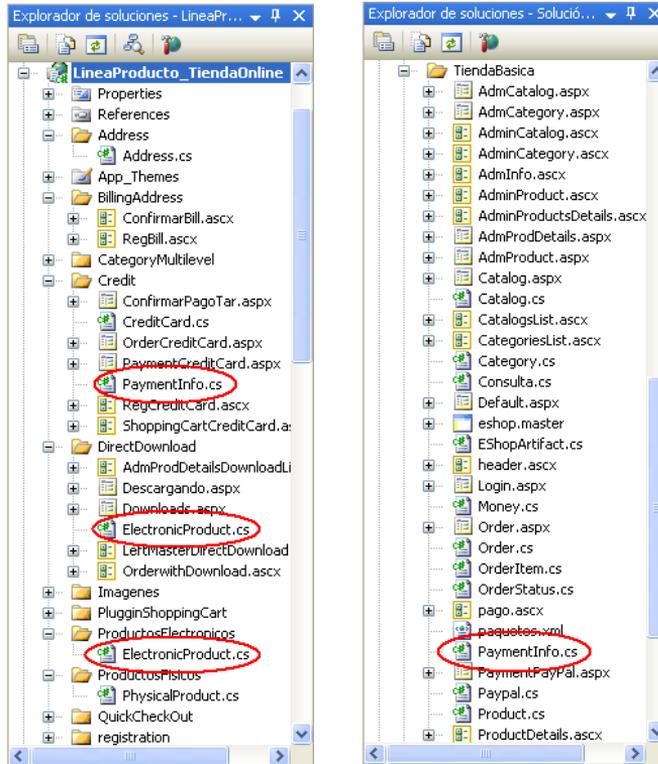


Figura 2-7 Clases parciales en distintos paquetes

DESARROLLO DE LA SOLUCIÓN

3 Análisis del Sistema

En este capítulo nos centraremos básicamente en determinar los objetivos y límites del sistema, así como de caracterizar su estructura y funcionamiento. De lo que se trata es de identificar cuáles van a ser las necesidades del usuario final del sistema, para lo cual se establecen un serie de requisitos.

Esta primera fase del proyecto es de vital importancia, ya que un error en la identificación de los requisitos que debe cumplir el sistema puede acarrear problemas graves a la hora del desarrollo y de la utilización del producto final.

3.1 Entrevistas

La forma de establecer los requisitos que debía cumplir el sistema han sido las entrevistas realizadas con el tutor del proyecto. A partir de estas entrevistas, se identifican los requisitos y se establece una prioridad para los mismos, de acuerdo a las necesidades de los usuarios finales y a las funcionalidades y restricciones del sistema que se va a construir.

3.2 Modelo de características de una línea de comercio electrónico

Este proyecto, al igual que el proyecto a partir del cual estamos trabajando, están basados en el modelo de características propuesto por Sean Quan Lau en 2006, que marca las distintas funcionalidades que puede adoptar la familia de productos..

En él, se detallan las distintas características que un producto final de comercio electrónico puede tener, indicando claramente las que son obligatorias, como por ejemplo un carrito, una estructura de catálogos, productos con una información básica al menos, etc. y las que son obligatorias pero se da a elegir una opción entre varias, como por ejemplo el proceso del pedido que es obligatorio y deberá llevarse a cabo mediante usuario registrado o usuario invitado, pero al menos uno de ellos. También se especifican las características opcionales que se podrán ir añadiendo en distintos paquetes para completar la funcionalidad y así aumentar la variabilidad de la línea.

La Figura 3-1 muestra el modelo de características de la línea de comercio electrónico, donde se encuentran representados tanto los paquetes ya existentes, como las nuevas opciones desarrolladas en el actual proyecto.

La parte que ahora nos ocupa, es la característica opcional que representa el registro de usuario. En el proyecto base, todos los compradores eran usuarios anónimos, pero ahora si el administrador elige esta característica, los usuarios tendrán la opción de registrarse como clientes. A su vez, este registro tiene una serie de características que también pueden ser seleccionadas o no, como son las Direcciones de Envío y Facturación, el registro de una Tarjeta de Crédito o la opción de tener un Método de Pago Rápido..

Otra característica importante es el Tipo de Producto: al menos se deberá escoger una de las dos características, ya que los productos han de ser o bien electrónicos, o bien físicos. En el caso de estos últimos, si se marca esta opción automáticamente se seleccionará también la opción de

DESARROLLO DE LA SOLUCIÓN

Dirección de Envío, ya que este tipo de productos han de ser enviados a una dirección física. Este es un claro ejemplo de las dependencias existentes entre algunos de los paquetes implementados, como veremos en capítulos posteriores.

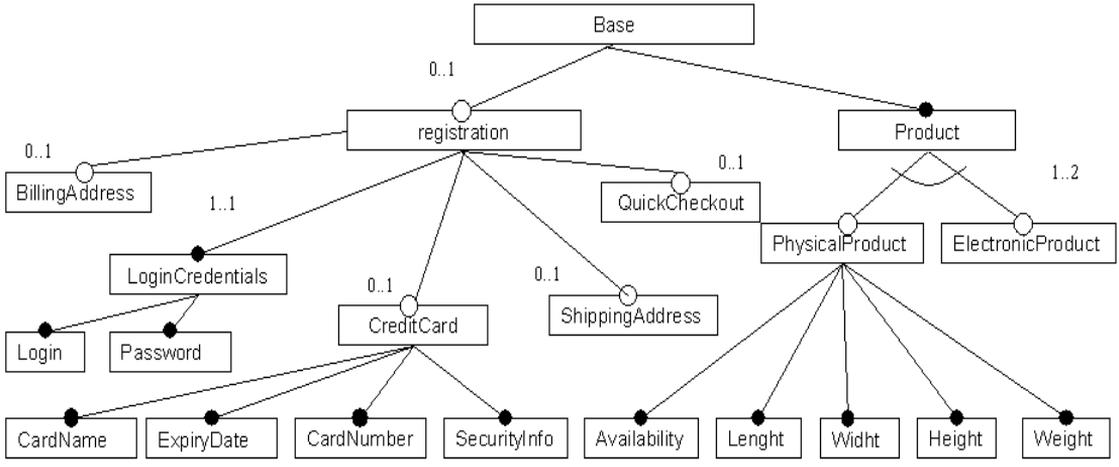


Figura 3-1 Modelo de características de comercio electrónico

Este diagrama de características se traduce en un diagrama de paquetes, en el que cada paquete representa una característica de la línea de producto, como se puede observar en la siguiente figura:

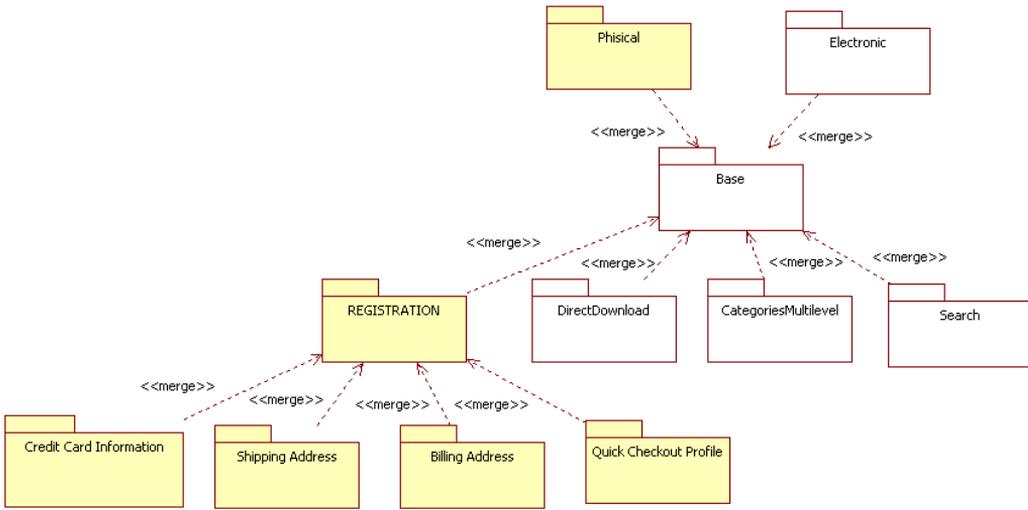


Figura 3-2 Diagrama de paquetes

3.3 Catálogo de Requisitos

Un requisito es una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. También se aplica a las condiciones que debe cumplir o poseer un sistema para satisfacer un contrato, una norma o una especificación. Por tanto, el conjunto de requisitos determinan lo que hará el sistema y definen las restricciones sobre su operación e implementación.

La especificación de los requisitos debe ser completa, no tener definiciones contradictorias, y debe ser expresada con exactitud, ya que si hay ambigüedad se corre el riesgo de que sean interpretados de forma diferente por los usuarios y por los desarrolladores.

3.3.1 Requisitos funcionales

Estos requisitos constituyen la declaración de los servicios que el sistema debe proporcionar, cómo debe reaccionar ante una entrada particular y cómo se debe comportar ante situaciones particulares. En resumen, describen la funcionalidad del sistema y de qué forma va a utilizarlo el usuario.

Como se trata de una Línea de Producto, hemos establecido los requisitos por paquetes, empezando por nuestro paquete principal que sería el que implementa el registro de un cliente y siguiendo por el resto de paquetes que añaden funcionalidad al primero.

3.3.1.1 Paquete: Registration

FRQ-0001	Registro de usuario
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir a un usuario registrarse como cliente o no hacerlo, dependiendo de si ha sido seleccionada esa opción. El registro estará basado en un nombre de usuario y una contraseña</i>
Importancia	vital

FRQ-0002	Identificación de usuario
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>exigir al usuario identificarse, en caso de que tenga que hacerlo para poder acceder. El sistema comprobará si dicho usuario estaba ya registrado</i>
Importancia	vital

DESARROLLO DE LA SOLUCIÓN

FRQ-0003	Baja de usuarios
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir a un usuario registrado darse de baja del sistema.</i>
Importancia	vital

FRQ-0004	Modificar datos
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir al usuario cambiar algunos datos de su perfil como puede ser el nombre, los apellidos o contraseña.</i>
Importancia	importante

FRQ-0005	Comprar productos físicos
Versión	1.0 (16/07/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir la compra de productos físicos solamente si el usuario se ha registrado anteriormente.</i>
Importancia	importante

FRQ-0006	Emisión de factura
Versión	1.0 (16/07/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir la emisión de facturas de compra sólo en el caso de que el usuario se haya registrado previamente</i>
Importancia	importante

3.3.1.2 Paquete: Credit Card Information

FRQ-0007	Registrar tarjeta de crédito
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir a los usuarios poder introducir los datos de la tarjeta de crédito que van a utilizar para realizar una compra</i>
Importancia	vital

FRQ-0008	Validar tarjeta de crédito
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>comprobar si la tarjeta de crédito que ha introducido el cliente es válida o no, a través de la autenticación del número de tarjeta.</i>
Importancia	vital

FRQ-0009	Pagar con tarjeta de crédito
Versión	1.0 (17/07/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir a un usuario registrado realizar la compra de productos a través del método tarjeta de crédito.</i>
Importancia	vital

3.3.1.3 Paquete: Shipping Address Information

FRQ-0010	Introducir dirección de envío
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir a los usuarios introducir la dirección física a la que quieren que se envíe su pedido.</i>
Importancia	vital

DESARROLLO DE LA SOLUCIÓN

3.3.1.4 Paquete: Billing Address Information

FRQ-0011	Introducir dirección de facturación
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir al usuario introducir los datos de la dirección fiscal de facturación de la compra realizada.</i>
Importancia	vital

3.3.1.5 Paquete: Quick Checkout Profile

FRQ-0012	Añadir información al perfil de Quick Checkout
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>poder almacenar información por defecto cuando el cliente realiza un pedido. Esta información incluye la información de pago y, si es necesario, la dirección de envío</i>
Importancia	vital

FRQ-0013	Utilizar datos del perfil de Quick Checkout
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>ser capaz de utilizar la información almacenada en el Quick Check Out Profile para que el cliente pueda realizar una compra utilizando los datos de pedidos anteriores automáticamente</i>
Importancia	vital

FRQ-0014	Actualizar información Quick Checkout
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>modificar los datos almacenados en el perfil Quick Checkout por unos nuevos si el cliente así lo solicita</i>
Importancia	importante

3.3.2 Requisitos no funcionales

Especifican las propiedades del sistema que tienen que ver con el rendimiento, velocidad, uso de memoria, fiabilidad, etc. Imponen condiciones a los requisitos funcionales.

NFR-0001	Sistema final compuesto por paquetes
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>estar compuesto por una serie de paquetes que se podrán añadir o no dependiendo de las peticiones del cliente. Uno de ellos será el núcleo de la aplicación, siendo obligatorio su uso para un normal funcionamiento.</i>
Importancia	vital

NFR-0002	Sistema flexible ante nuevas funcionalidades
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>ser altamente flexible ante nuevas funcionalidades añadidas mediante módulos o paquetes</i>
Importancia	vital

NFR-0003	Paquete base
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>ser capaz de desarrollar un sistema de comercio electrónico con las funcionalidades básicas que este tipo de aplicaciones requiera, mediante un paquete base que deberá estar incluido siempre en el sistema final.</i>
Importancia	vital

NFR-0004	Añadir paquetes opcionales asegurando la integridad del sistema
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>poder añadir o excluir de la aplicación final todas las funcionalidades extra divididas en paquetes asegurando la integridad del mismo</i>
Importancia	Vital

DESARROLLO DE LA SOLUCIÓN

NFR-0005	Integración de paquetes de forma automática
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>ser capaz de reconocer e integrar aquellos paquetes elegidos por el cliente, de forma automática</i>
Importancia	vital

NFR-0006	Paquetes opcionales compatibles
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>asegurar que los distintos paquetes opcionales que se añadan a la aplicación final sean totalmente compatibles entre si</i>
Importancia	vital

NFR-0007	Tiempo de respuesta
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>reaccionar a cualquier evento en tiempo inferior a 1 minuto</i>
Importancia	vital

NFR-0008	Integridad de la información
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>mantener los datos de los clientes bajo una confidencialidad absoluta</i>
Importancia	importante

3.4 Modelo de Casos de Uso

Los casos de uso nos ayudan a describir un uso del sistema y cómo éste interactúa con el usuario. “Un Caso de Uso es una secuencia de acciones realizadas por el sistema, que producen un resultado valioso para un actor en particular”

3.4.1 Actores

Los actores son entidades externas que interaccionan con el sistema participando en los casos de uso. Representan los papeles que distintos usuarios pueden jugar, por lo que un mismo usuario podría representar el papel de distintos actores. También el sistema puede ser un actor.

ACT-0001	Usuario
Versión	1.0 (02/05/2008)
Descripción	Este actor representa a un usuario interesado en el contenido de la tienda
Comentarios	Ninguno

ACT-0002	Usuario Registrado
Versión	1.0 (02/05/2008)
Descripción	Este actor representa a un usuario interesado en algún producto de la tienda y que se ha registrado para poder comprar en ella.
Comentarios	Ninguno

3.4.2 Diagrama de Casos de Uso

Los diagramas de Casos de Uso muestran las relaciones estructurales entre los actores y los casos de uso del sistema, es decir, representan la funcionalidad del sistema.

Los casos de uso representados a continuación están agrupados por paquetes. Para diferenciar los nuevos de los ya existentes, hemos representado los ya creados en el proyecto anterior en blanco y en un tamaño menor.

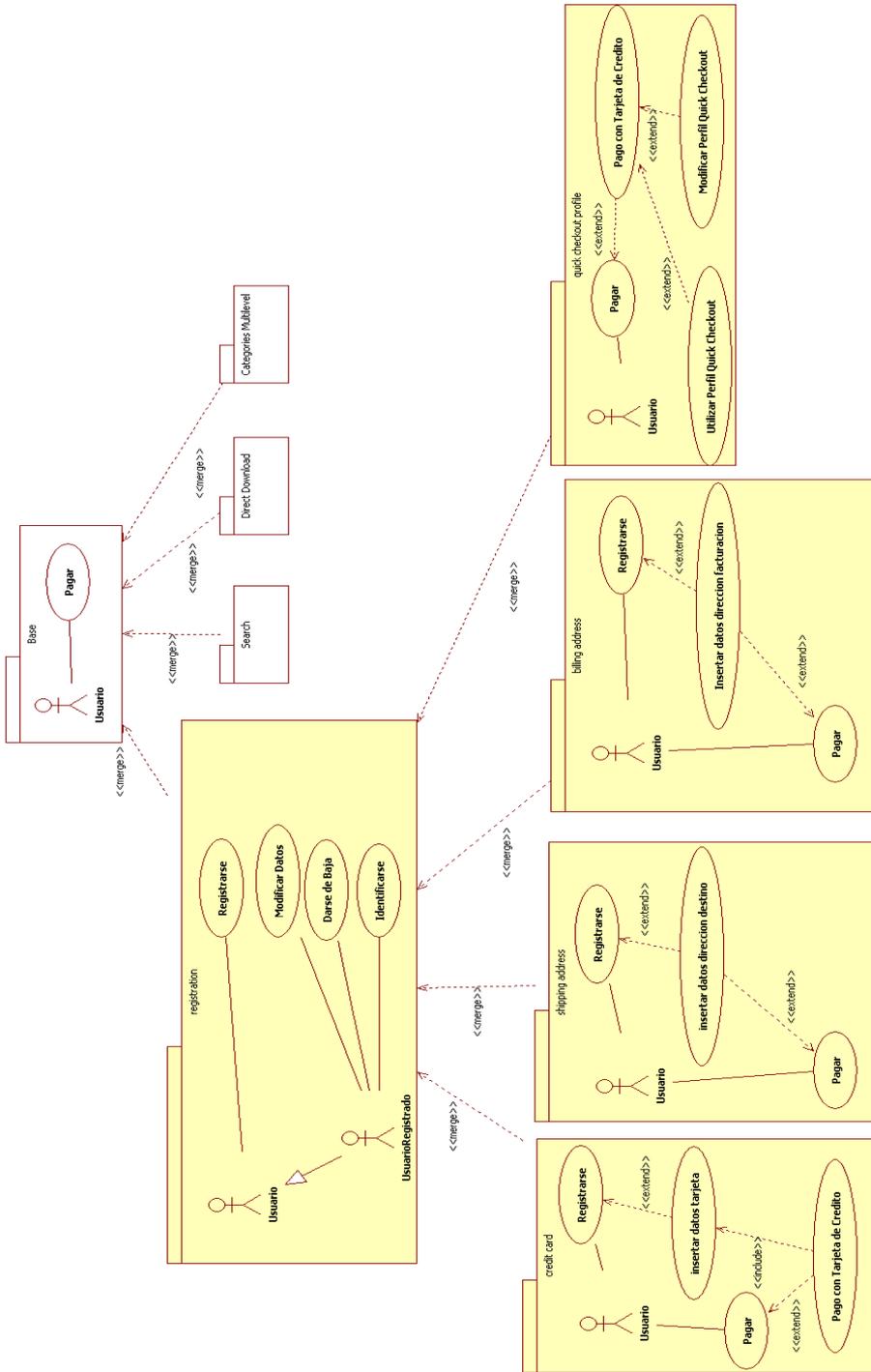


Figura 3-3 Interacción Paquetes – Casos de Uso

3.4.3 Especificación de Casos de Uso

A continuación describiremos los Casos de Uso que representan la utilización del sistema, junto con sus excepciones y agrupados en paquetes. Este documento es la parte más útil de los Casos de Uso, ya que en él se explica detalladamente la forma de interactuar entre el sistema y el usuario.

3.4.3.1 Paquete: Registration

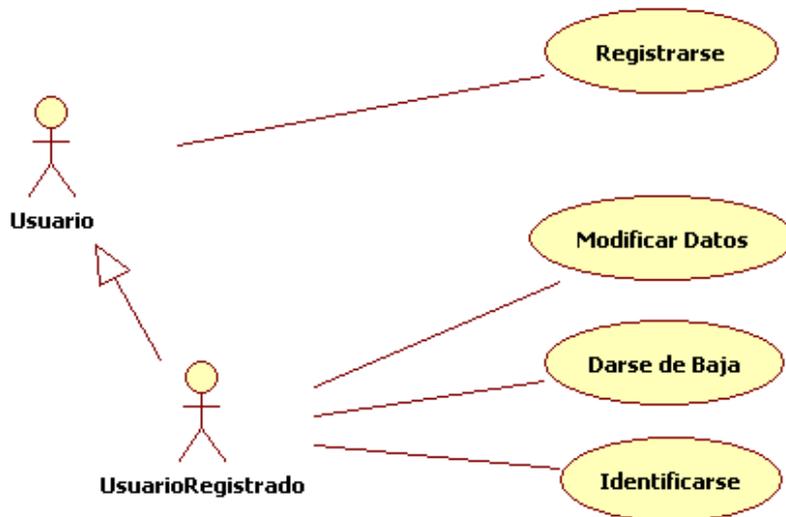


Figura 3-4 Diagrama de Casos de Uso del paquete Registration

UC-0001	Registrarse	
Versión	1.0 (02/05/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario quiera registrarse como cliente y entrar en nuestra base de datos para después autenticarse y poder comprar.</i>	
Precondición	este incluido el paquete Registration y el usuario no esté registrado.	
Secuencia normal	Paso	Acción
	1	El actor <u>Usuario (ACT-0001)</u> solicita registrarse.
	2	El sistema solicita al usuario su nombre, apellidos, nombre de usuario del sistema, email y contraseña.

DESARROLLO DE LA SOLUCIÓN

	3	El actor Usuario (ACT-0001) introduce su nombre, apellidos, nombre de usuario del sistema, email y contraseña en el formulario.
	4	El sistema comprueba que se han rellenado correctamente todos los campos y que el usuario no existe en la base de datos.
	5	El sistema introduce los datos personales del usuario en la base de datos
	6	El sistema muestra un mensaje informativo de que el registro se ha realizado con éxito y manda un email recordando el nombre de usuario y la contraseña.
Postcondición	se añade a la base de datos un nuevo usuario registrado	
Excepciones	Paso	Acción
	1	Si el usuario ha iniciado sesión anteriormente y no la ha cerrado, el sistema no permite el registro, a continuación este caso de uso queda sin efecto
	4	Si los campos no han sido rellenados correctamente, el sistema muestra un mensaje informativo y vuelve al paso 2, a continuación este caso de uso continúa
	4	Si ya existe un usuario con ese mismo nombre de usuario en el sistema, el sistema muestra un mensaje informativo y vuelve al paso 2, a continuación este caso de uso continúa
	4	Si el usuario desea introducir los datos de su tarjeta de crédito, se realiza el caso de uso Insertar datos tarjeta (UC-0005) , a continuación este caso de uso continúa
	4	Si el usuario desea introducir los datos de la dirección fiscal de facturación, se realiza el caso de uso Insertar datos dirección de facturación (UC-0008) , a continuación este caso de uso continúa
	4	Si el usuario desea introducir los datos de la dirección de envío de productos físicos, se realiza el caso de uso Insertar datos dirección destino (UC-0007) , a continuación este caso de uso continúa
	6	Si el servidor de correo no se encuentra disponible en esos momentos, el sistema muestra un mensaje informativo, a continuación este caso de uso queda sin efecto

UC-0002	Identificarse	
Versión	1.0 (30/06/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario desee acceder al sistema</i> .	
Precondición	el usuario este registrado y no se haya iniciado ninguna sesión.	
Secuencia normal	Paso	Acción
	1	El actor Usuario Registrado (ACT-0002) solicita identificarse
	2	El sistema muestra la pagina para que el usuario introduzca su nombre de usuario y su contraseña
	3	El actor Usuario Registrado (ACT-0002) introduce su nombre de usuario y su contraseña
	4	El sistema verifica que los datos introducidos son correctos
	5	El sistema verifica que el usuario pertenece al conjunto de usuarios registrados
	6	El sistema permite el acceso a zonas solo permitidas para usuario registrados y la compra de productos
Postcondición	se inicia una sesión con los datos del usuario	
Excepciones	Paso	Acción
	4	Si los datos introducidos no son correctos, el sistema muestra un mensaje informativo y vuelve al paso 2, a continuación este caso de uso continúa
	5	Si el usuario no esta registrado en la base de datos, el sistema muestra un mensaje informativo donde se pide al usuario que vuelva a intentar introducir los datos y que sino esta registrado que lo haga y vuelve al paso 2, a continuación este caso de uso continúa

DESARROLLO DE LA SOLUCIÓN

UC-0003	Modificar datos	
Versión	1.0 (30/06/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario desee modificar algún dato de su perfil como puede ser el nombre, los apellidos o la contraseña.</i>	
Precondición	ser un usuario registrado y haberse identificado	
Secuencia normal	Paso	Acción
	1	El actor Usuario Registrado (ACT-0002) solicita modificar su perfil
	2	El sistema <i>muestra el formulario donde el usuario podrá cambiar su nombre, apellidos, dirección de correo y contraseña</i>
	3	El actor Usuario Registrado (ACT-0002) <i>modifica los datos que crea necesarios</i>
	4	El sistema <i>comprueba que los datos introducidos son correctos</i>
	5	El sistema <i>modifica los datos en la base de datos</i>
Postcondición	modificar los datos registrados en el perfil de usuario	
Excepciones	Paso	Acción
	4	Si <i>los datos no son correctos</i> , el sistema <i>envía un mensaje informativo y vuelve al paso 2</i> , a continuación este caso de uso continúa

UC-0004	Darse de Baja	
Versión	1.0 (30/06/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario solicite darse de baja en el sistema</i>	
Precondición	estar registrado en el sistema y haberse identificado	
Secuencia normal	Paso	Acción
	1	El actor Usuario Registrado (ACT-0002) <i>solicita darse de baja en el sistema</i>
	2	El sistema <i>pide confirmación al usuario para darse de baja en el sistema</i>
	3	El actor Usuario Registrado (ACT-0002) <i>confirma su deseo de darse de baja</i>
	4	El sistema <i>da de baja al usuario quedando este imposibilitado para acceder de nuevo al sistema y cerrando sus sesión actual.</i>
Postcondición	el usuario queda eliminado y se cierra su sesión	
Excepciones	Paso	Acción
	3	<i>Si el usuario cancela la acción, el sistema devuelve al usuario a la tarea que estaba realizando, a continuación este caso de uso queda sin efecto</i>

3.4.3.2 Paquete: Credit Card Information

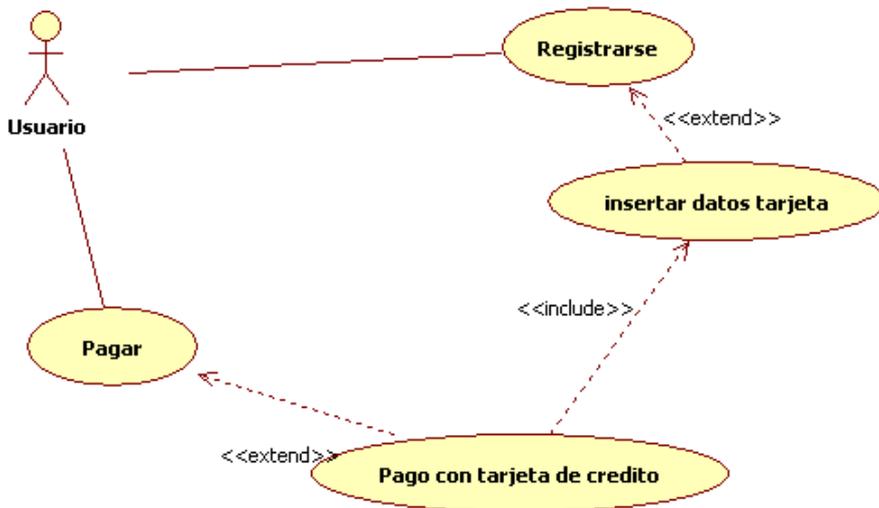


Figura 3-5 Diagrama de Casos de Uso del paquete Credit Card

UC-0005	Insertar datos tarjeta	
Versión	1.0 (02/05/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario registre sus datos personales o cuando desee realizar una compra y elija la opción Pago con Tarjeta</i> o durante la realización de los siguientes casos de uso: [UC-0001] Registrarse , [UC-0006] Pago con Tarjeta de Crédito	
Precondición	el usuario debe estar registrado en el sistema.	
Secuencia normal	Paso	Acción
	1	El sistema <i>solicita al usuario que introduzca el numero, el tipo, fecha de expiración, código de seguridad y nombre del titular de la tarjeta con la que va a realizar la compra.</i>
	2	El actor Usuario Registrado (ACT-0002) <i>introduce el numero, fecha de expiración, código de seguridad, tipo y nombre del titular referentes a su tarjeta de crédito</i>
	3	El sistema <i>verifica que los datos son correctos comprobando el numero de tarjeta .</i>
	4	El sistema <i>almacena los datos para su posterior uso</i>

Postcondición	los datos de la tarjeta de crédito se almacenaran en la base de datos del sistema	
Excepciones	Paso	Acción
	3	Si los datos de tarjeta no son correctos , el sistema lo comunica al usuario con un mensaje informativo y se vuelve al caso 1, a continuación este caso de uso continúa

UC-0006	Pago con Tarjeta de Crédito	
Versión	1.0 (11/07/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario opte por realizar el pago del importe de la compra a través de la tarjeta de crédito.</i>	
Precondición	el usuario quiera pagar el importe, que debe ser no nulo, de uno o mas productos existentes y desee hacerlo a través de la tarjeta de crédito	
Secuencia normal	Paso	Acción
	1	El actor Usuario Registrado (ACT-0002) solicita realizar el pago de su compra usando el método Pago con Tarjeta de Crédito.
	2	Se realiza el caso de uso Insertar datos tarjeta (UC-0005)
	3	El sistema mostrara una pantalla para que el usuario compruebe que tanto los productos que desea adquirir como los datos de la tarjeta son correctos.
	4	El actor Usuario Registrado (ACT-0002) confirma que los datos son correctos
	5	El sistema procesa el pedido y almacena los datos de la compra.
Postcondición	el pago se realiza correctamente y el pedido queda registrado	
Excepciones	Paso	Acción
	2	Si el paquete de la dirección de facturación esta incluido en el sistema , se realiza el caso de uso Insertar datos dirección de facturación (UC-0008) , a continuación este caso de uso continúa
	2	Si el paquete dirección destino esta incluido y el cliente ha adquirido algún producto de tipo físico , se realiza el caso de uso Insertar datos dirección destino (UC-0007) , a continuación este caso de uso continúa

3.4.3.3 Paquete: Shipping Address

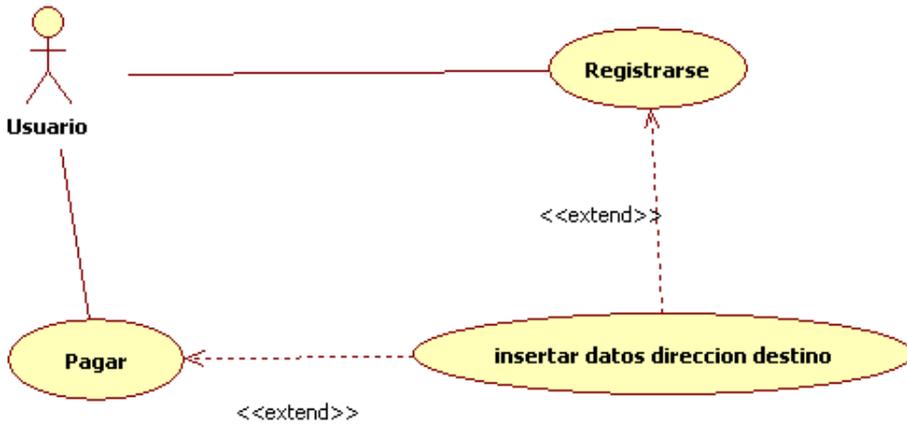


Figura 3-6 Diagrama de Casos de Uso del paquete Shipping Address

UC-0007	Insertar datos dirección destino	
Versión	1.0 (02/05/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>solicite al usuario que introduzca los datos donde se ha de enviar el pedido</i> o durante la realización de los siguientes casos de uso: [UC-0001] Registrarse , [UC-0006] Pago con Tarjeta de Crédito	
Precondición	que el usuario este registrado en el sistema	
Secuencia normal	Paso	Acción
	1	El sistema <i>solicita al usuario que introduzca la dirección, municipio, ciudad, provincia, país y código postal de la dirección física donde desea que se envíen sus pedidos.</i>
	2	El actor Usuario Registrado (ACT-0002) <i>introduce la dirección, municipio, provincia, país y código postal de la dirección donde desea que le lleguen los pedidos</i>
	3	El sistema <i>comprueba que todos los campos han sido rellenados y que el código postal se encuentre dentro del rango establecido.</i>
	4	El sistema <i>almacena los datos de la dirección destino para su posterior uso.</i>

Postcondición	la dirección destino donde el usuario desea que manden sus pedidos queda almacenada	
Excepciones	Paso	Acción
	3	Si <i>algún campo no se ha rellenado</i> , el sistema <i>envía un mensaje informativo al usuario y se vuelve al paso 1</i> , a continuación este caso de uso <i>continúa</i>
	3	Si <i>el código postal no tiene el formato correcto</i> , el sistema <i>muestra un mensaje informativo y se vuelve al paso 1</i> , a continuación este caso de uso <i>continúa</i>

3.4.3.4 Paquete: Billing Address

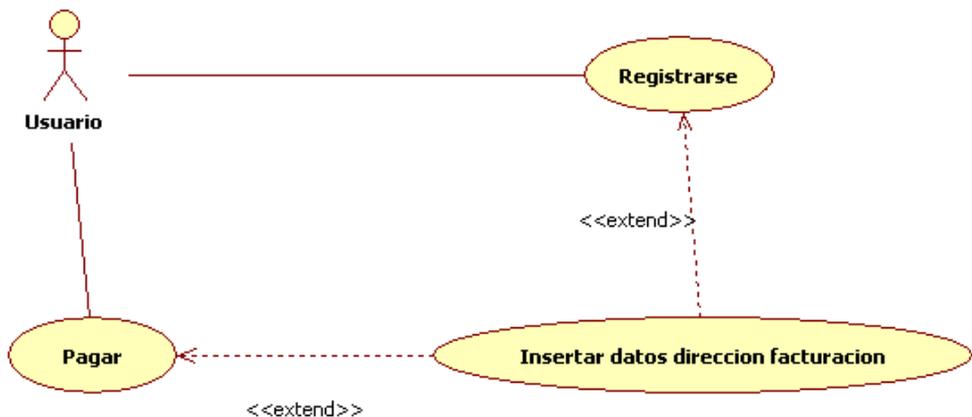


Figura 3-7 Diagrama de Casos de Uso del paquete Billing Address

UC-0008	Insertar datos dirección de facturación
Versión	1.0 (30/06/2008)
Dependencias	Ninguno
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>solicite al usuario los datos de la dirección fiscal donde desea que se facturen sus compras.</i> o durante la realización de los siguientes casos de uso: [UC-0001] Registrarse , [UC-0006] Pago con Tarjeta de Crédito
Precondición	que el usuario este registrado en el sistema.

DESARROLLO DE LA SOLUCIÓN

Secuencia normal	Paso	Acción
	1	El sistema <i>solicita al usuario que introduzca la dirección, municipio, provincia, país y código postal de la dirección fiscal donde desea que queden registrados sus pagos.</i>
	2	El actor <u>Usuario Registrado (ACT-0002)</u> <i>introduce la dirección, municipio, provincia, país y código postal de la dirección de facturación.</i>
	3	El sistema <i>verifica que todos los campos han sido rellenados y que el código postal este dentro del rango establecido</i>
	4	El sistema <i>almacena los datos de la dirección de facturación en la base de datos</i>
Postcondición	la dirección fiscal donde el usuario desea que queden registradas sus facturas queda almacenada.	
Excepciones	Paso	Acción
	3	Si <i>no han sido rellenados todos los campos</i> , el sistema <i>envía un mensaje informativo y vuelve al paso 1</i> , a continuación este caso de uso <i>continúa</i>
	3	Si <i>el código postal no tiene el formato correcto</i> , el sistema <i>muestra un mensaje informativo y vuelve al paso 1</i> , a continuación este caso de uso <i>continúa</i>

3.4.3.5 Paquete: Quick Checkout Profile

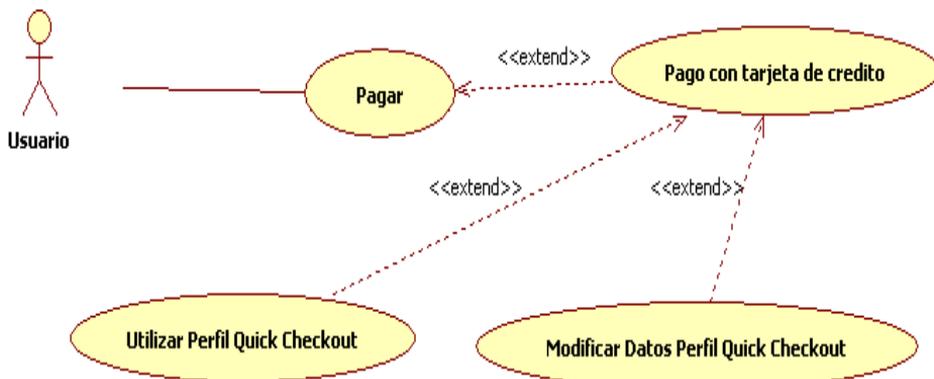


Figura 3-8 Diagrama de Casos de Uso del paquete Quick Checkout

UC-0009	Utilizar perfil Quick Checkout	
Versión	1.0 (01/07/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario seleccione realizar el pago por el método rápido.</i>	
Precondición	el paquete de Quick Checkout este incluido y los datos de la tarjeta y de la dirección de facturación estén almacenados.	
Secuencia normal	Paso	Acción
	1	El actor Usuario Registrado (ACT-0002) <i>selecciona la opción de pagar a través del método Quick Checkout.</i>
	2	<i>El sistema muestra los datos de la tarjeta y dirección de facturación que van a ser utilizados para realizar el pago. También se mostrarán los datos de la dirección de envío en el caso de que exista algún producto físico en su pedido.</i>
	3	El actor Usuario Registrado (ACT-0002) <i>confirma el deseo de realizar el pago con esa tarjeta y que la dirección de facturación y la de envío en el caso de que la haya sean las deseadas.</i>
	4	<i>El sistema muestra un mensaje informativo confirmando que la compra se ha realizado correctamente.</i>
Postcondición	el usuario habrá realizado la compra sin tener que volver a introducir los datos de la tarjeta y la dirección de facturación.	
Excepciones	Paso	Acción
	3	<i>Si desea cancelar la operación de pago por el método de Quick Checkout, el sistema muestra de nuevo las distintas opciones de pago, a continuación este caso de uso queda sin efecto</i>

UC-0010	Modificar perfil Quick Checkout	
Versión	1.0 (01/07/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el usuario tenga que decidir si los nuevos datos introducidos serán utilizados a partir de ahora como perfil Quick Checkout.</i>	
Precondición	el usuario este registrado, exista ya un perfil <i>Quick Checkout</i> y seleccione la opción de modificarlo.	

Secuencia normal	Paso	Acción
	1	El sistema <i>pregunta al usuario si quiere que los nuevos datos introducidos de tarjeta y dirección de facturación así como la dirección de envío, en el caso de que sea necesaria, pasen a formar parte del perfil Quick Checkout.</i>
	2	El actor <u>Usuario Registrado (ACT-0002)</u> <i>acepta la opción de modificar su perfil.</i>
	3	El sistema <i>guarda en la base de datos el nuevo perfil Quick Checkout.</i>
Postcondición	el usuario tiene asociado un nuevo perfil <i>Quick Checkout.</i>	
Excepciones	Paso	Acción
	-	-

3.5 Diagramas de Estados y de Secuencia

Para continuar con el análisis del sistema, vamos a utilizar los siguientes elementos, también agrupados por paquetes:

Diagrama de estados: muestran el comportamiento de un objeto, es decir, el conjunto de estados por los cuales pasa un objeto durante su vida, junto con los cambios que permiten pasar de un estado a otro.

Diagrama de secuencia: muestran las interacciones entre objetos ordenadas en secuencia temporal. Nos enseña los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario.

3.5.1 Paquete: Registration

3.5.1.1 CU - 0001: Registrarse

Este caso de uso se realiza cuando un usuario quiere registrarse como cliente de la tienda electrónica.

Diagrama de estados

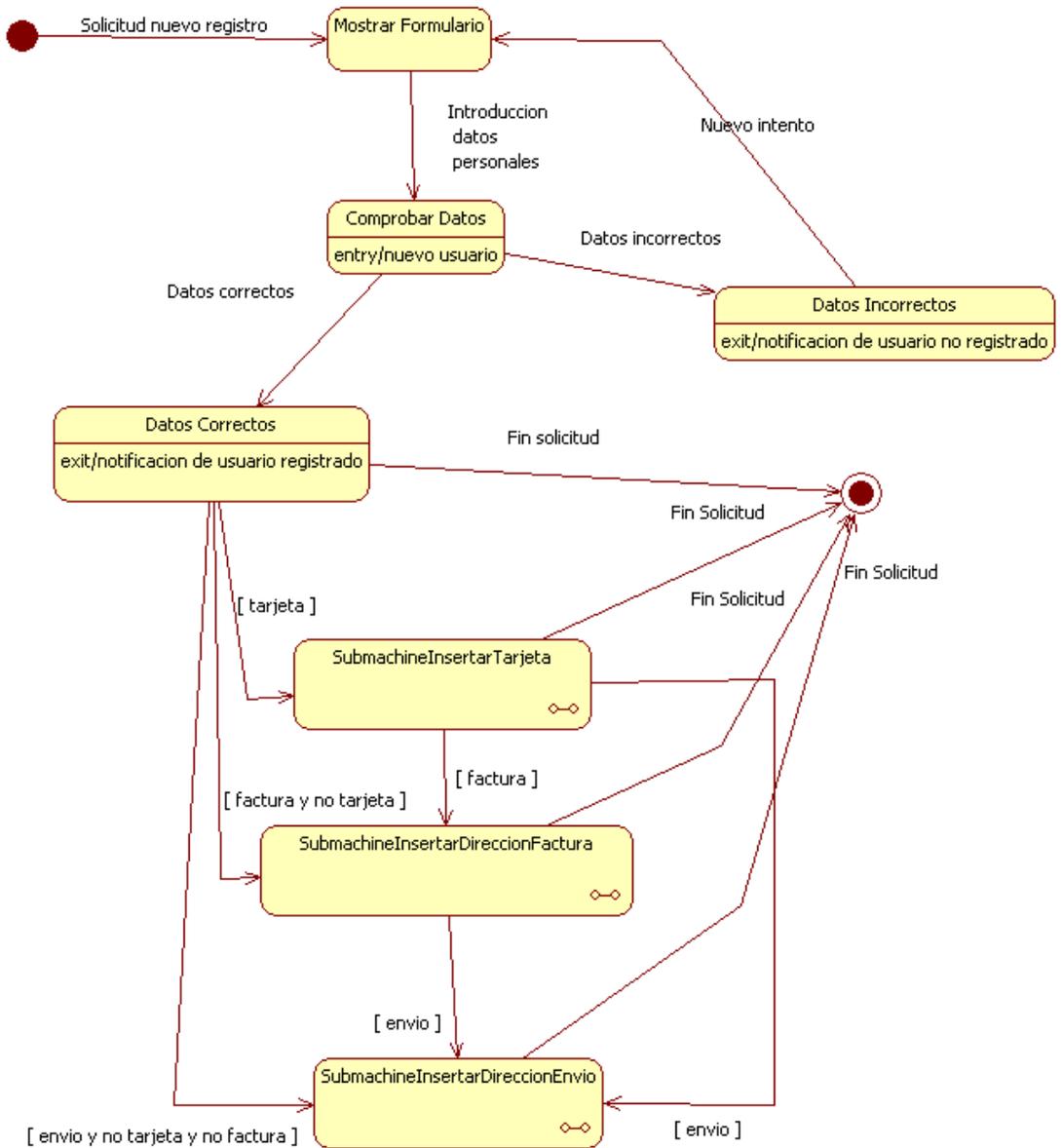


Figura 3-9 Diagrama de Estados del CU – 0001 Registrarse

Diagrama de secuencia

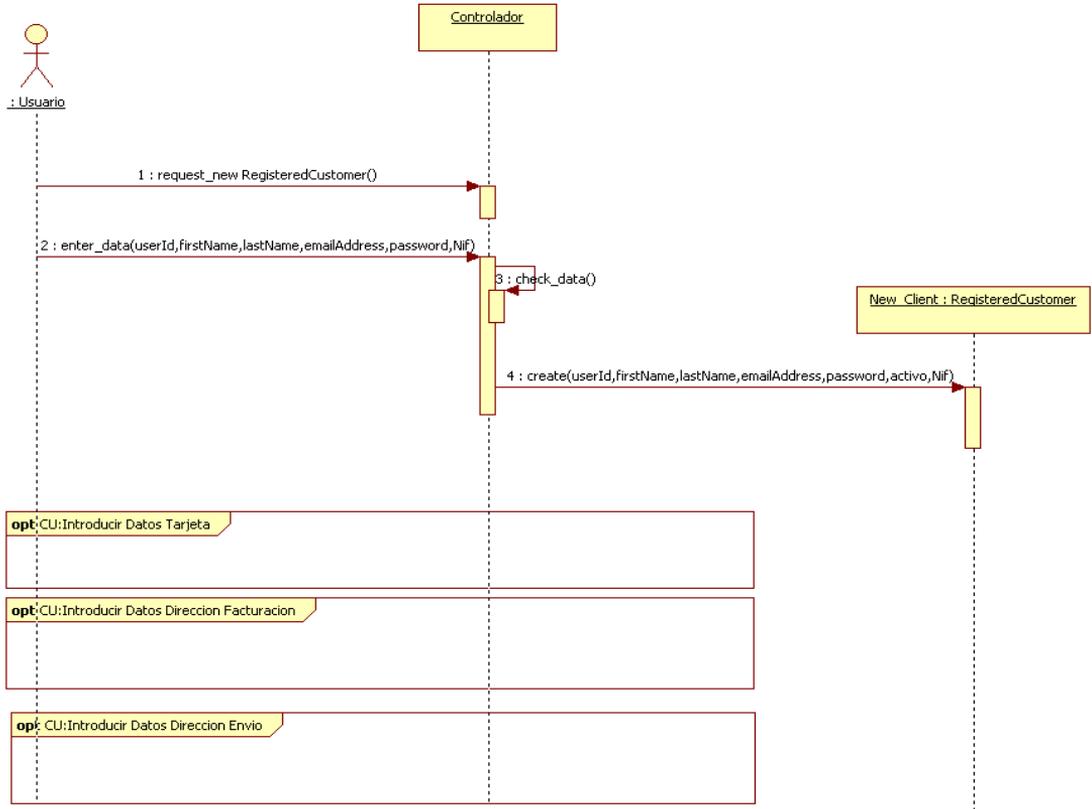


Figura 3-10 Diagrama de Secuencia del CU – 0001 Registrarse

3.5.1.2 CU-0002: Identificarse

Este caso de uso se realiza cuando un usuario registrado quiere identificarse para poder acceder a los servicios que ofrece la tienda electrónica.

Diagrama de estados

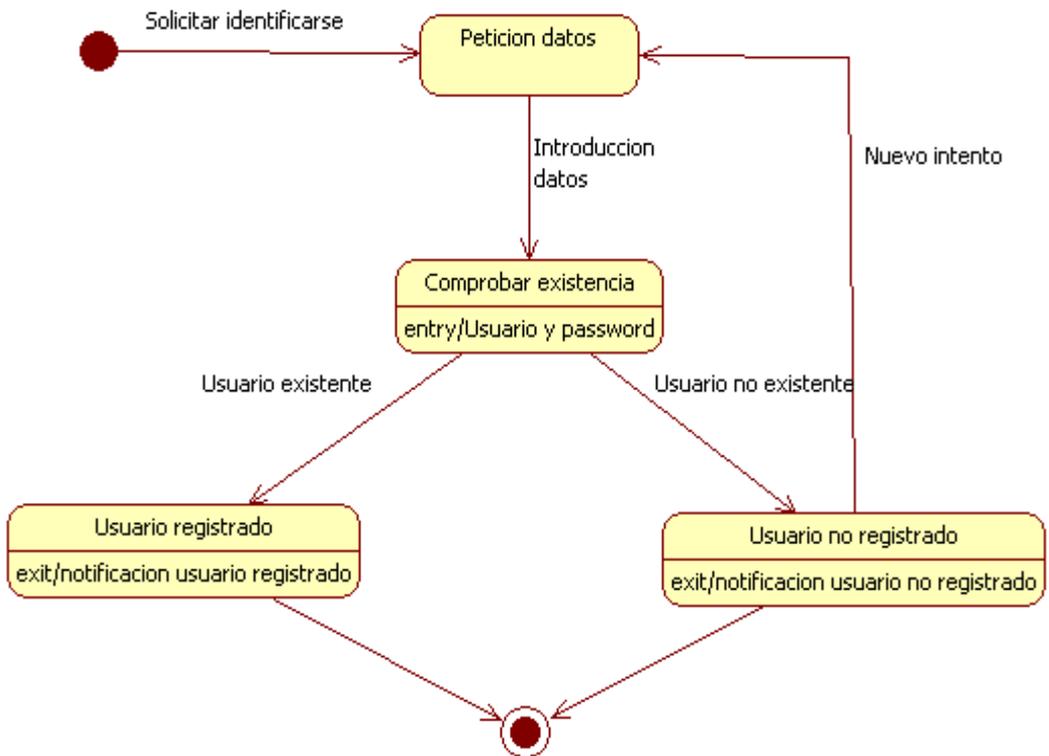


Figura 3-11 Diagrama de Estados del CU – 0002 Identificarse

Diagrama de secuencia

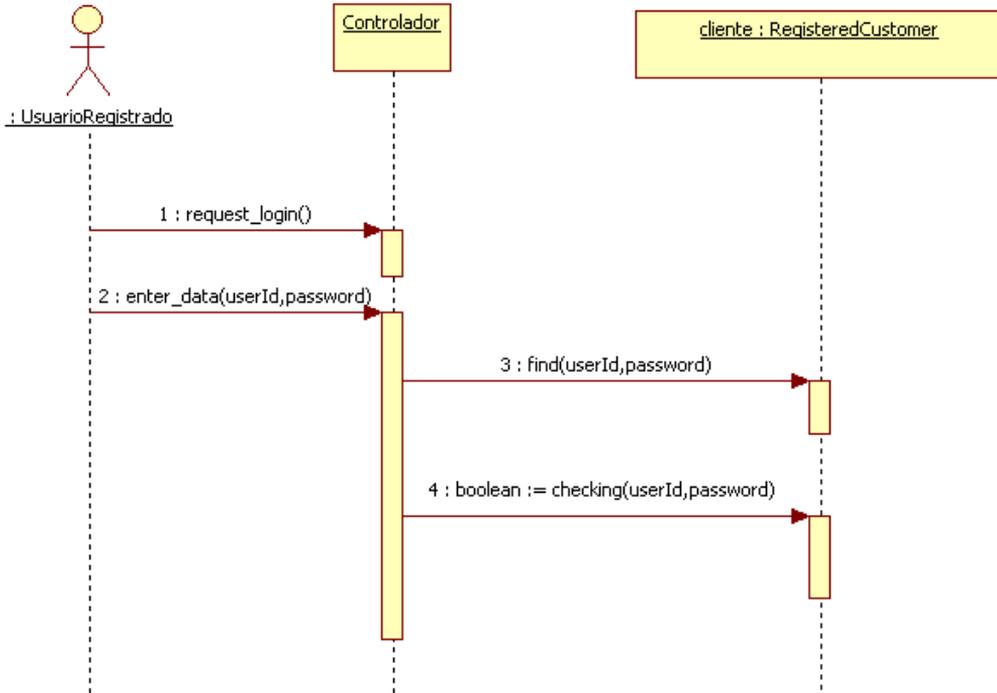


Figura 3-12 Diagrama de Secuencia del CU – 0002 Identificarse

3.5.1.3 CU-0003: Modificar datos

Este caso de uso se realiza cuando un usuario registrado desee modificar algún dato de su perfil como puede ser el nombre, los apellidos o la contraseña.

Diagrama de estados

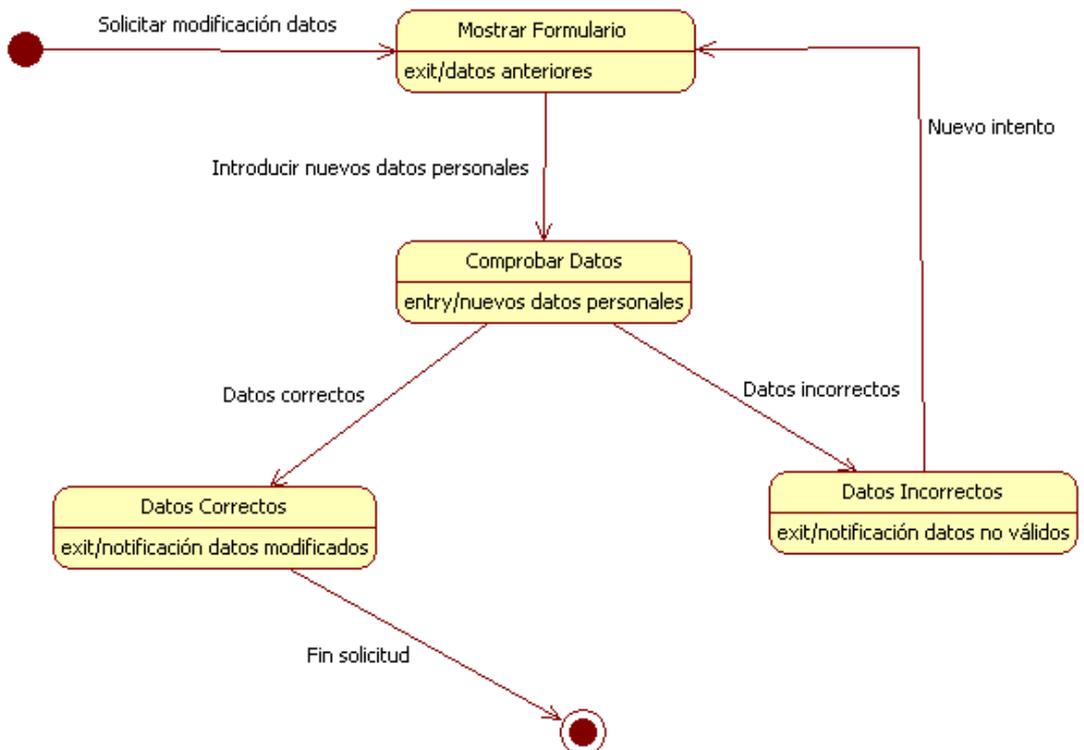


Figura 3-13 Diagrama de Estados del CU – 0003 Modificar Datos

Diagrama de secuencia

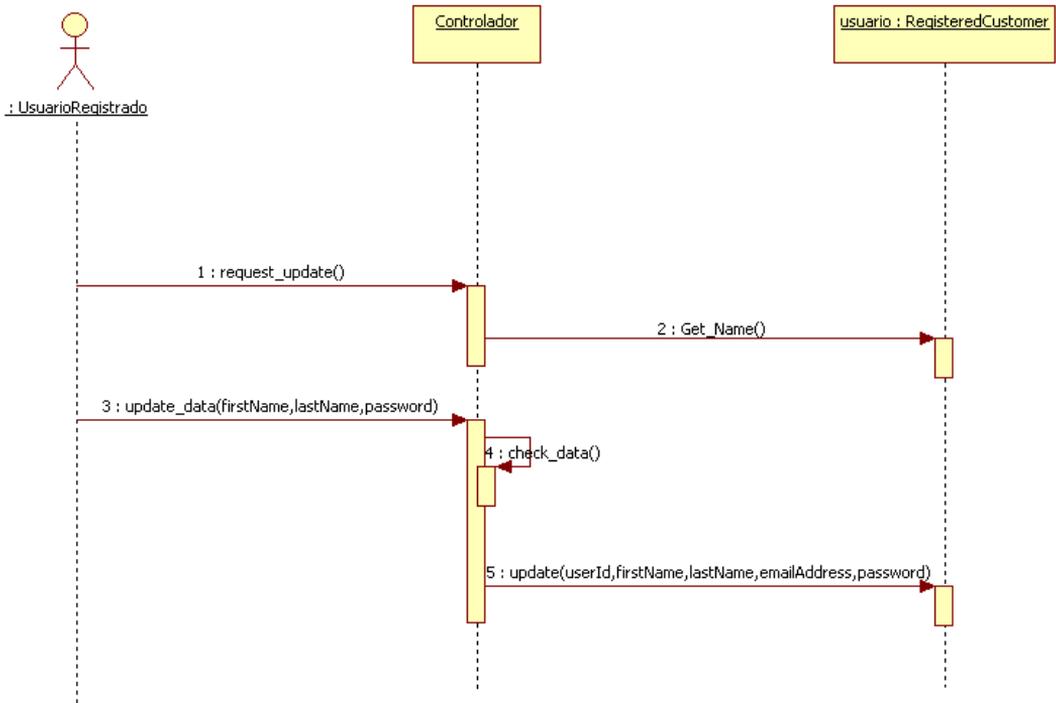


Figura 3-14 Diagrama de Secuencia del CU – 0003 Modificar Datos

3.5.1.4 CU-0004: Darse de Baja

Este caso de uso se realiza cuando un usuario registrado solicite darse de baja en el sistema

Diagrama de estados

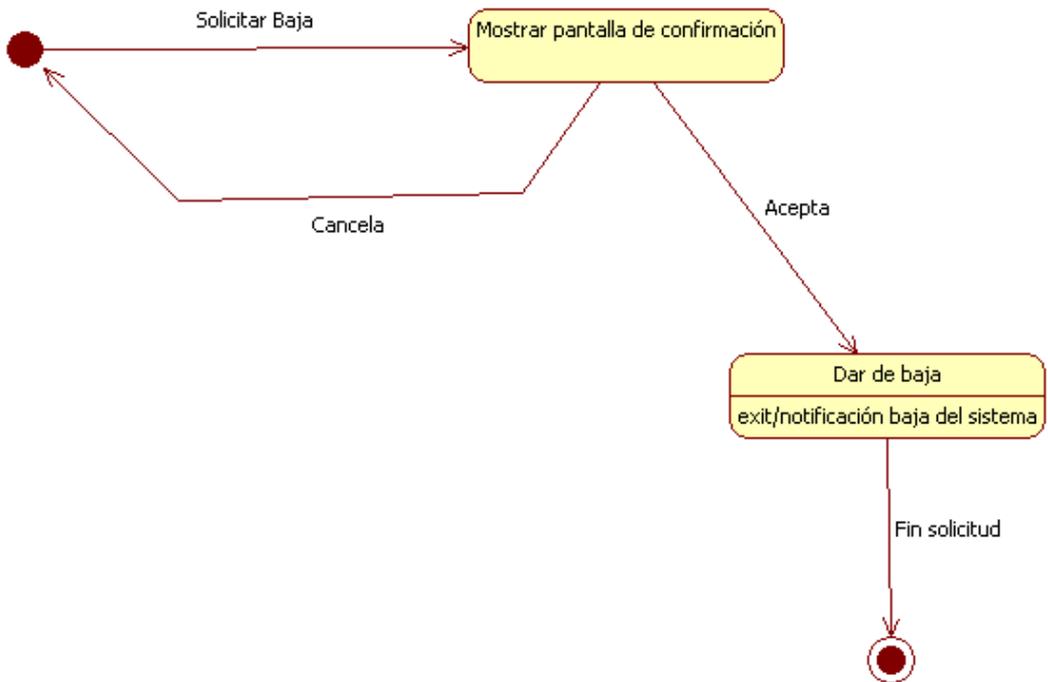


Figura 3-15 Diagrama de Estados del CU – 0004 Darse de Baja

Diagrama de secuencia

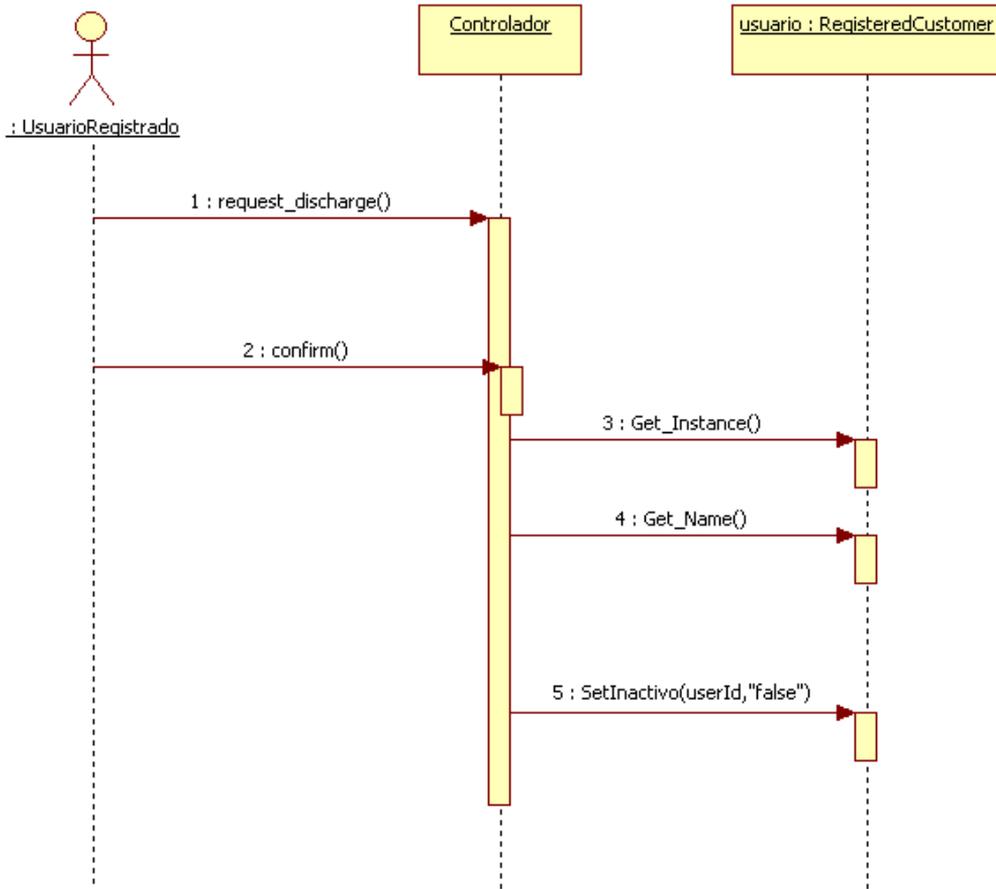


Figura 3-16 Diagrama de Secuencia del CU – 0004 Darse de Baja

3.5.2 Paquete: Credit Card Information

3.5.2.1 CU-0005: Insertar Datos Tarjeta

Este caso de uso se realiza cuando un usuario registre sus datos personales o cuando desee realizar una compra y elija la opción Pago con Tarjeta.

Diagrama de estados

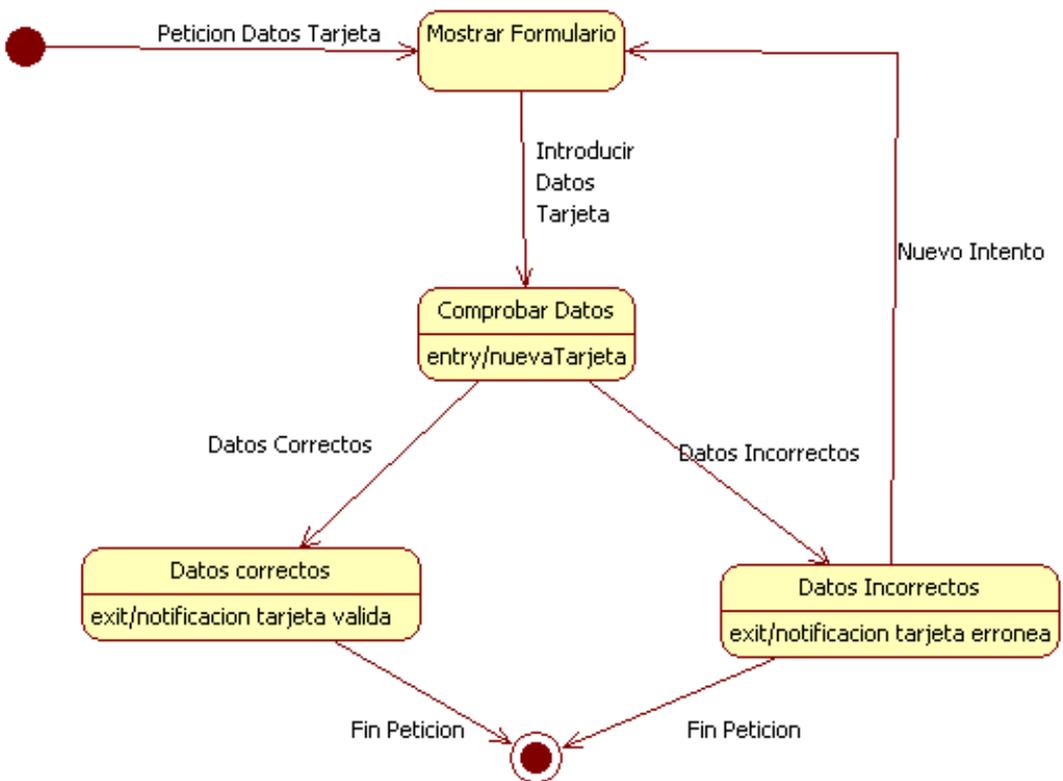


Figura 3-17 Diagrama de Estados del CU – 0005 Insertar Datos Tarjeta

Diagrama de secuencia

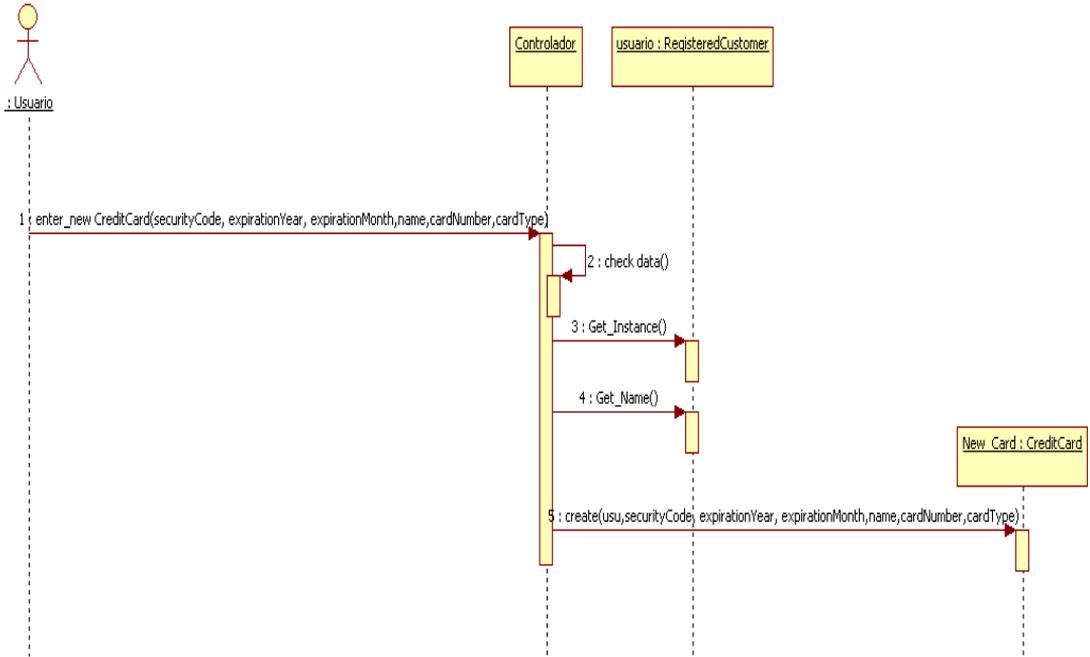


Figura 3-18 Diagrama de Secuencia del CU – 0005 Insertar Datos Tarjeta

3.5.2.2 CU-0006: Pago con Tarjeta

Este caso de uso se realiza cuando un usuario opte por realizar el pago del importe de la compra a través de la tarjeta de crédito.

Diagrama de estados

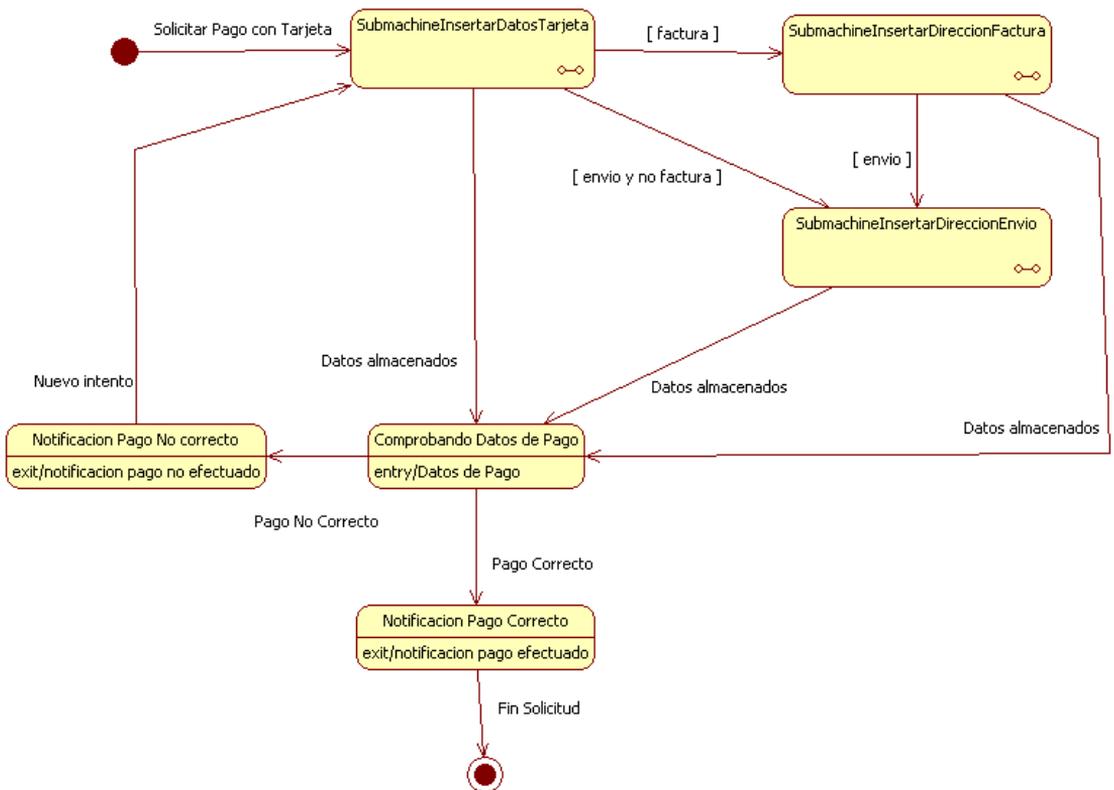


Figura 3-19 Diagrama de Estados del CU – 0006 Pago con Tarjeta

Diagrama de secuencia

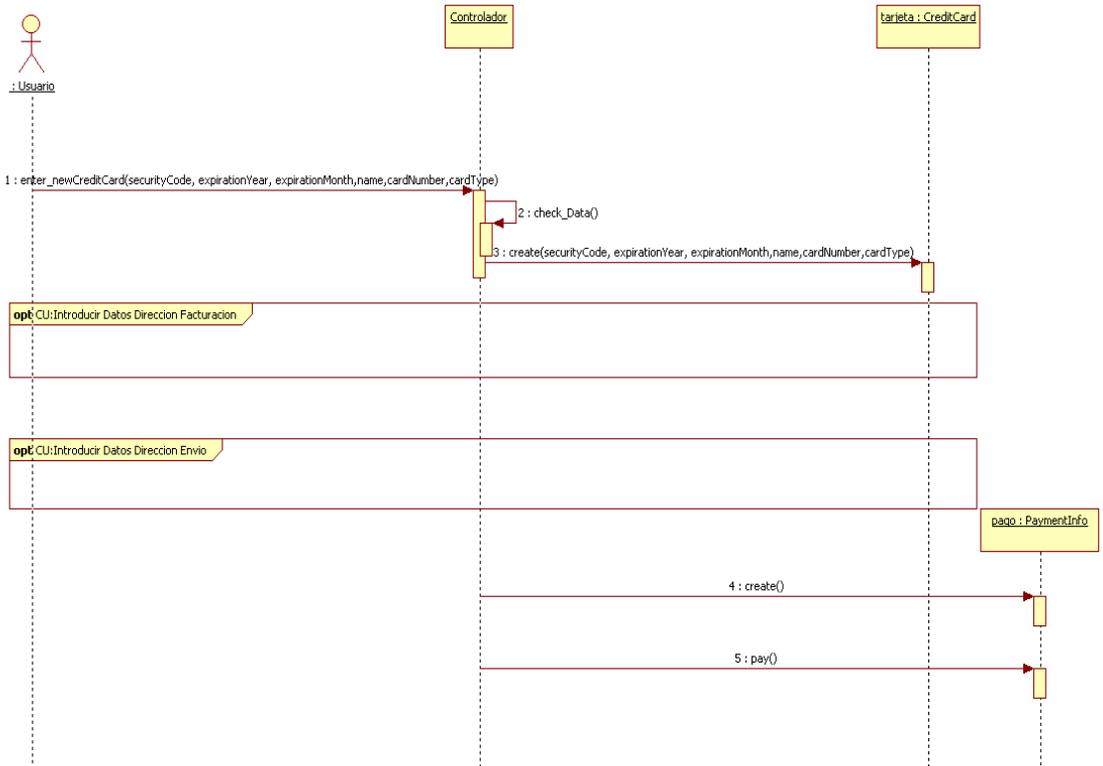


Figura 3-20 Diagrama de Secuencia del CU – 0006 Pago con Tarjeta

3.5.3 Paquete: Shipping Address

3.5.3.1 CU-0007: Insertar datos dirección destino

Este caso de uso se realiza cuando un usuario vaya a comprar un producto físico y el sistema le solicite que introduzca una dirección de envío.

Diagrama de estados

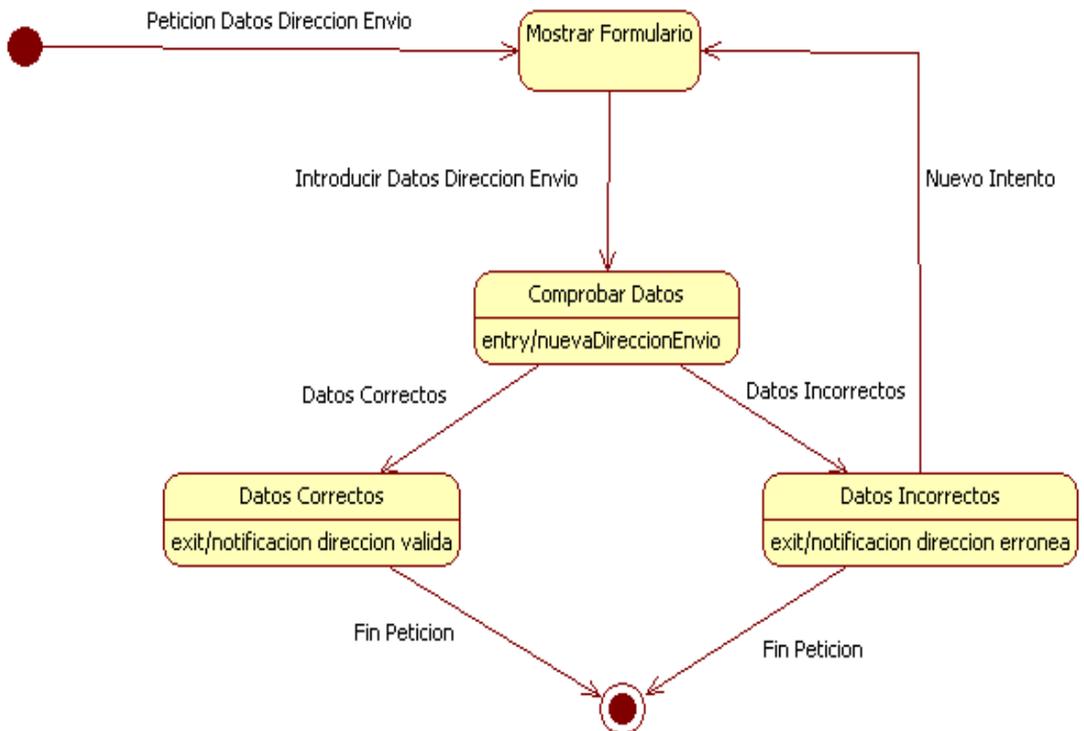


Figura 3-21 Diagrama de Estados del CU – 0007 Insertar Datos Envío

Diagrama de secuencia

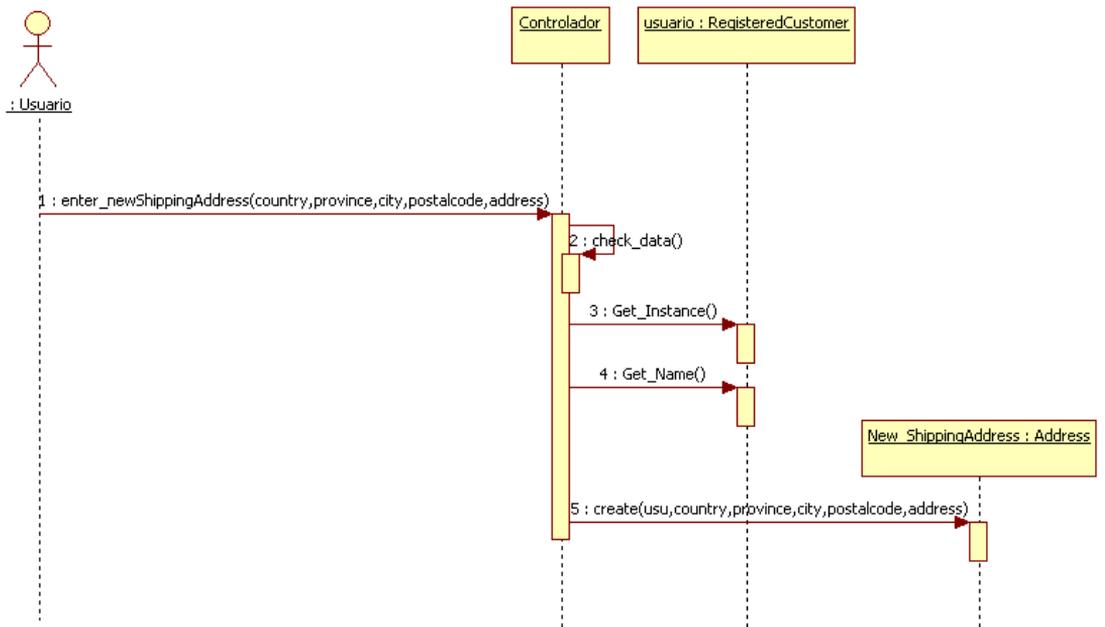


Figura 3-22 Diagrama de Secuencia del CU – 0007 Insertar Datos Envío

3.5.4 Paquete: Billing Address

3.5.4.1 CU-0008: Insertar datos dirección de facturación

Este caso de uso se realiza cuando un usuario vaya a introducir los datos de la dirección fiscal donde quiere que se facturen sus compras.

Diagrama de estados

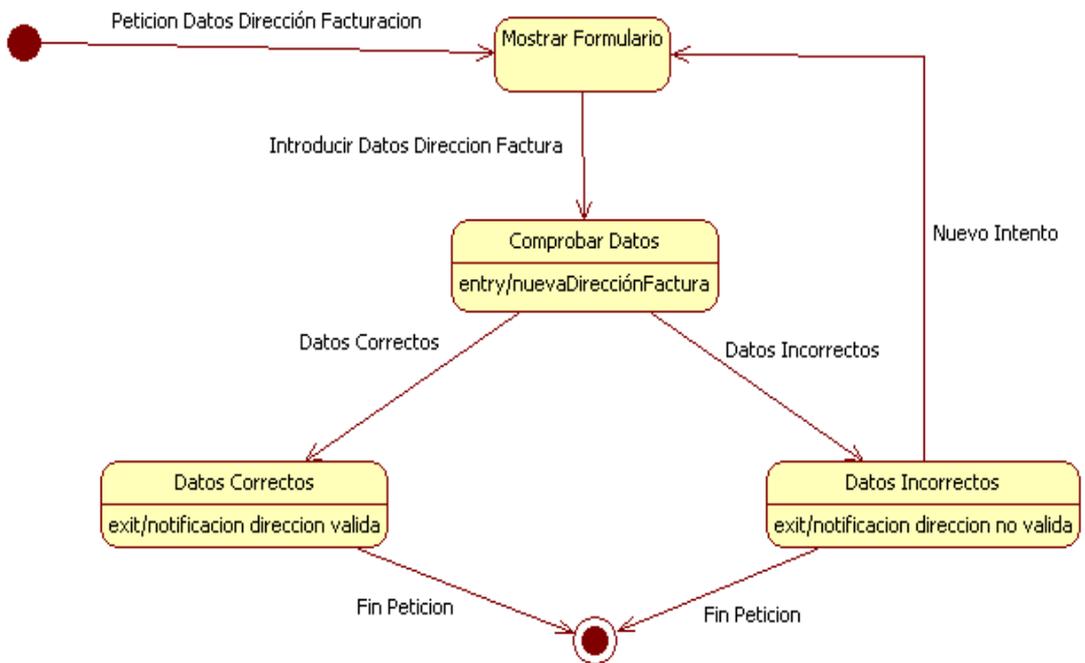


Figura 3-23 Diagrama de Estados del CU – 0008 Insertar Datos Facturación

Diagrama de secuencia

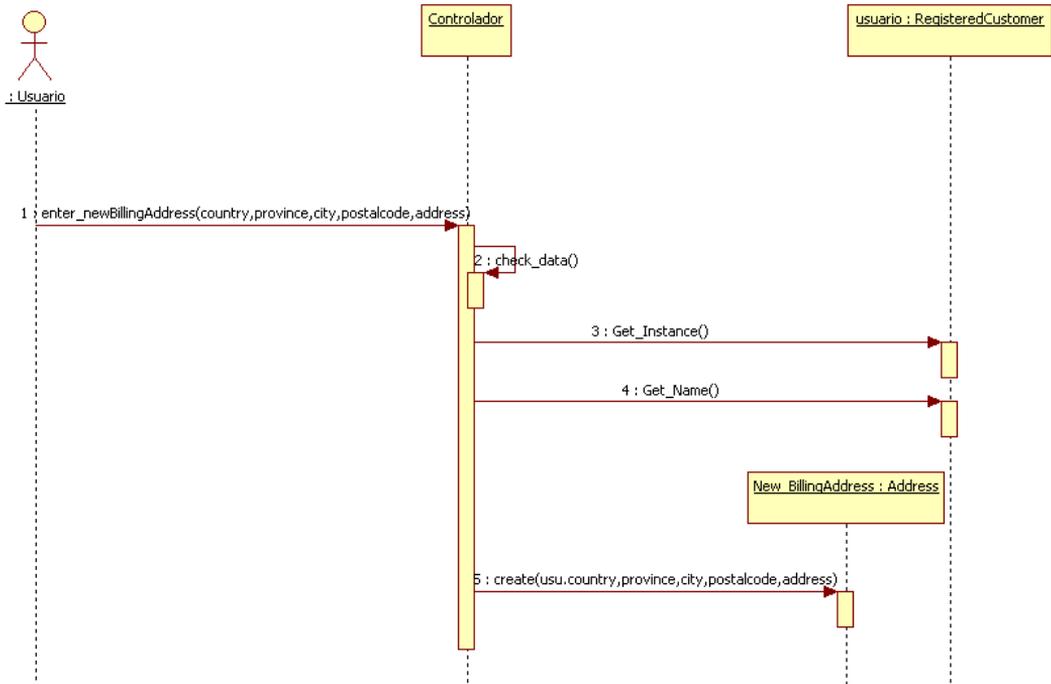


Figura 3-24 Diagrama de Secuencia del CU – 0008 Insertar Datos Facturación

3.5.5 Paquete: Quick CheckOut Profile

3.5.5.1 CU-0009: Utilizar Perfil Quick CheckOut

Este caso de uso se realiza cuando un usuario seleccione realizar el pago por el método rápido.

Diagrama de estados

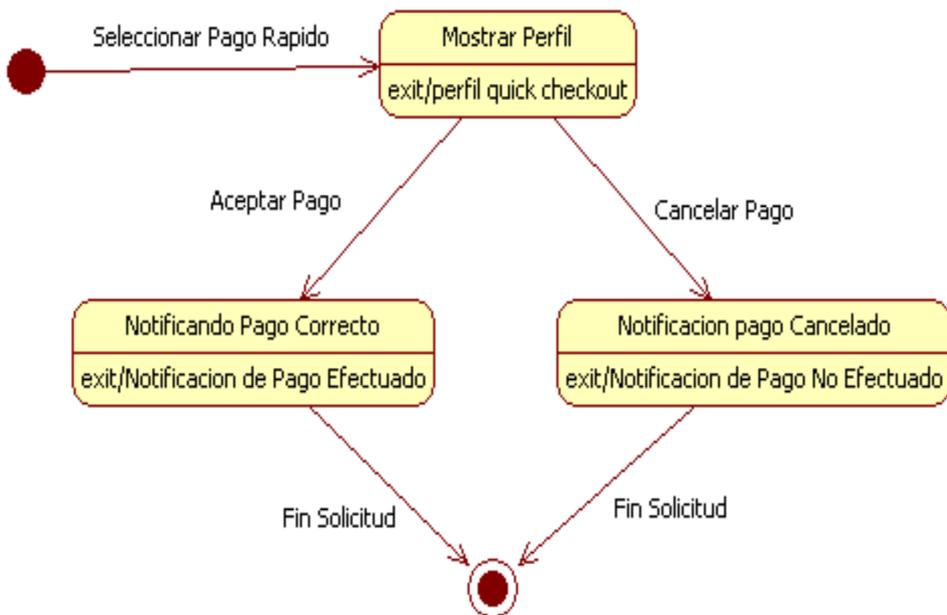


Figura 3-25 Diagrama de Estados del CU – 0009 Utilizar Perfil Quick Checkout

Diagrama de secuencia

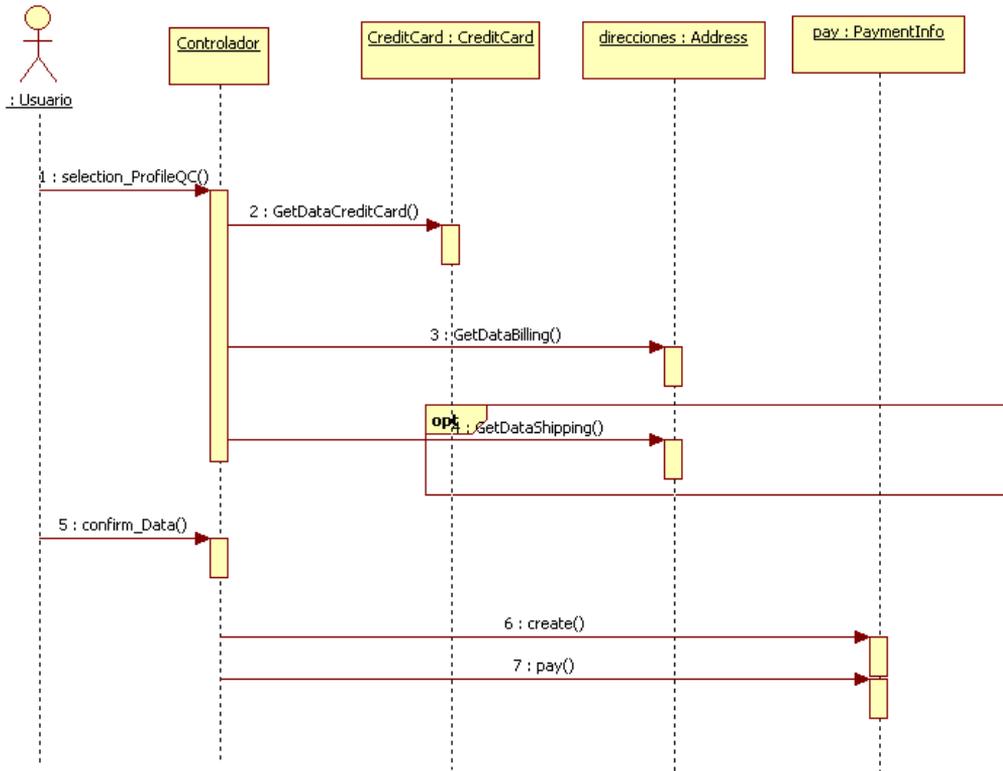


Figura 3-26 Diagrama de Secuencia del CU – 0009 Utilizar Perfil Quick Checkout

3.5.5.2 CU-0010: Modificar Perfil Quick CheckOut

Este caso de uso se realiza cuando un usuario tenga que decidir si los nuevos datos introducidos serán utilizados a partir de ahora como perfil Quick Checkout.

Diagrama de estados

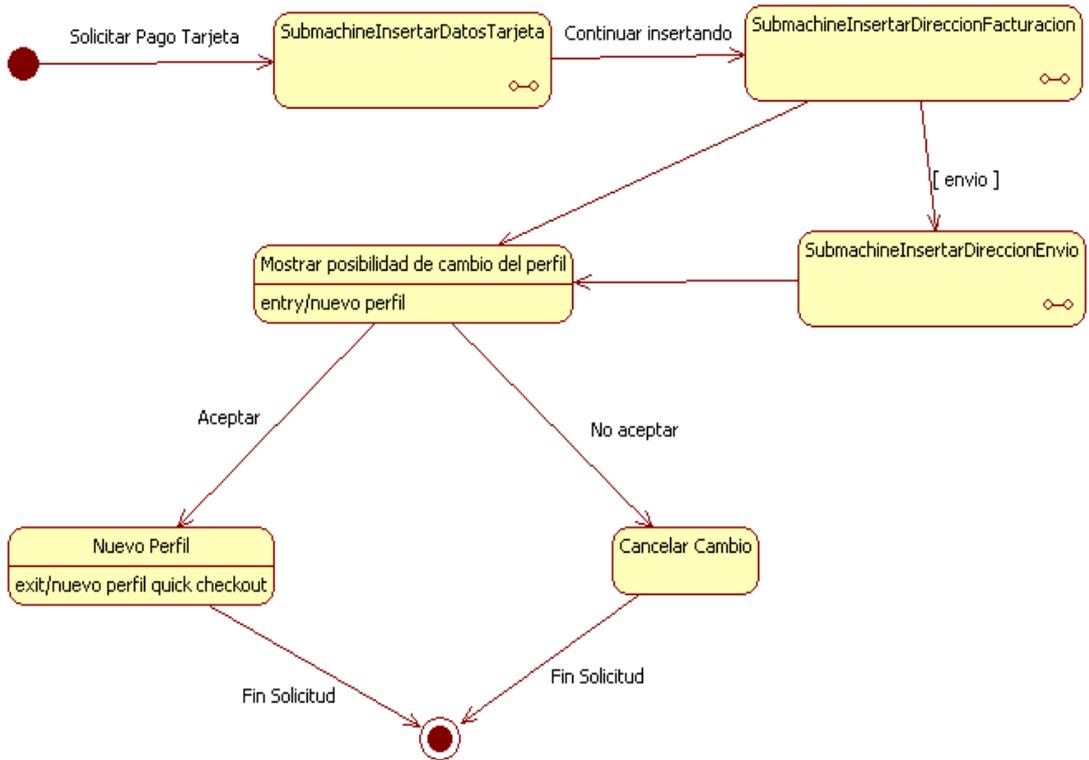


Figura 3-27 Diagrama de Estados del CU – 0010 Modificar Perfil Quick Checkout

Diagrama de secuencia

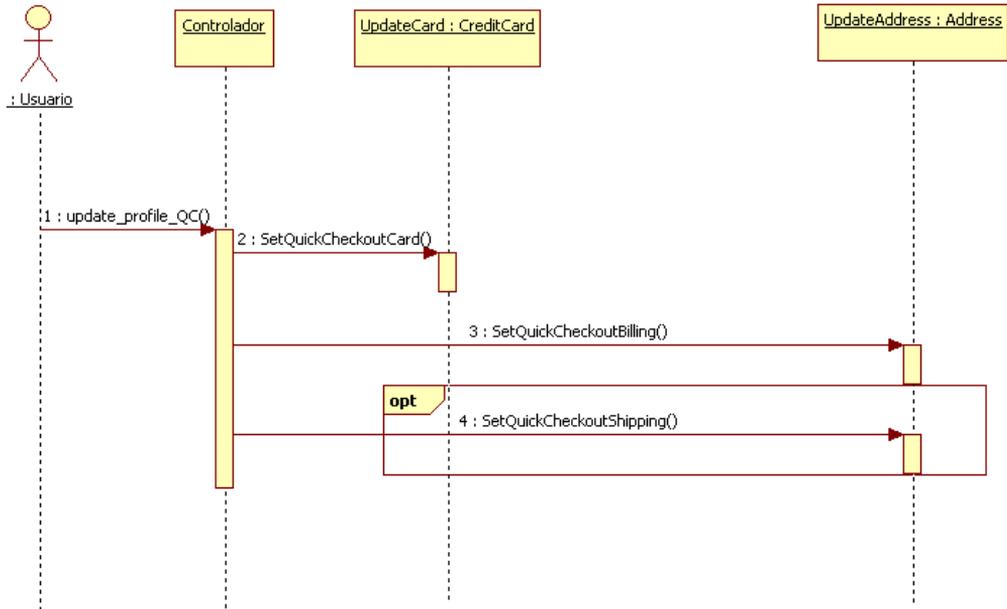


Figura 3-28 Diagrama de Secuencia del CU – 0010 Modificar Perfil Quick Checkout

3.6 Modelo de Dominio

La Figura 3-29 muestra el Modelo de Dominio con los paquetes que se incluyen y las clases que contiene cada paquete, especificando también sus atributos.

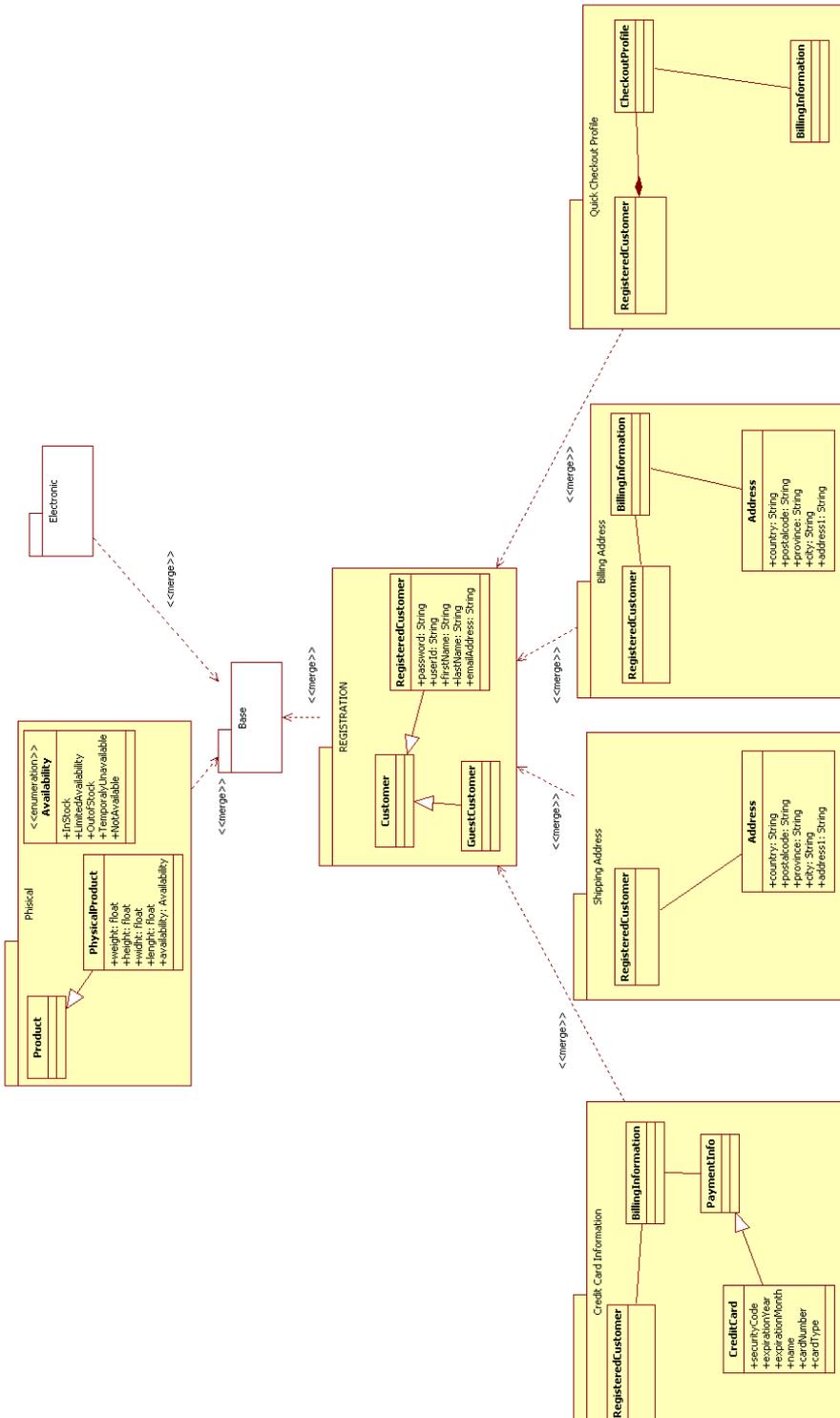


Figura 3-29 Modelo de Dominio

4 Diseño del Sistema

En la fase de Diseño del sistema, se transforma el modelo de dominio de la información, creado durante el análisis, en las estructuras de datos necesarios para implementar el software. El diseño es la manera de materializar con precisión los requisitos del sistema.

4.1 Diagramas de Secuencia

Para realizar el diseño del sistema vamos a utilizar diagramas de secuencia como los de la parte de análisis, pero con la diferencia de que al tratarse de la fase de diseño, estarán más detallados en cuanto a la implementación.

4.1.1 Paquete: Registration

4.1.1.1 CU - 0001: Registrarse

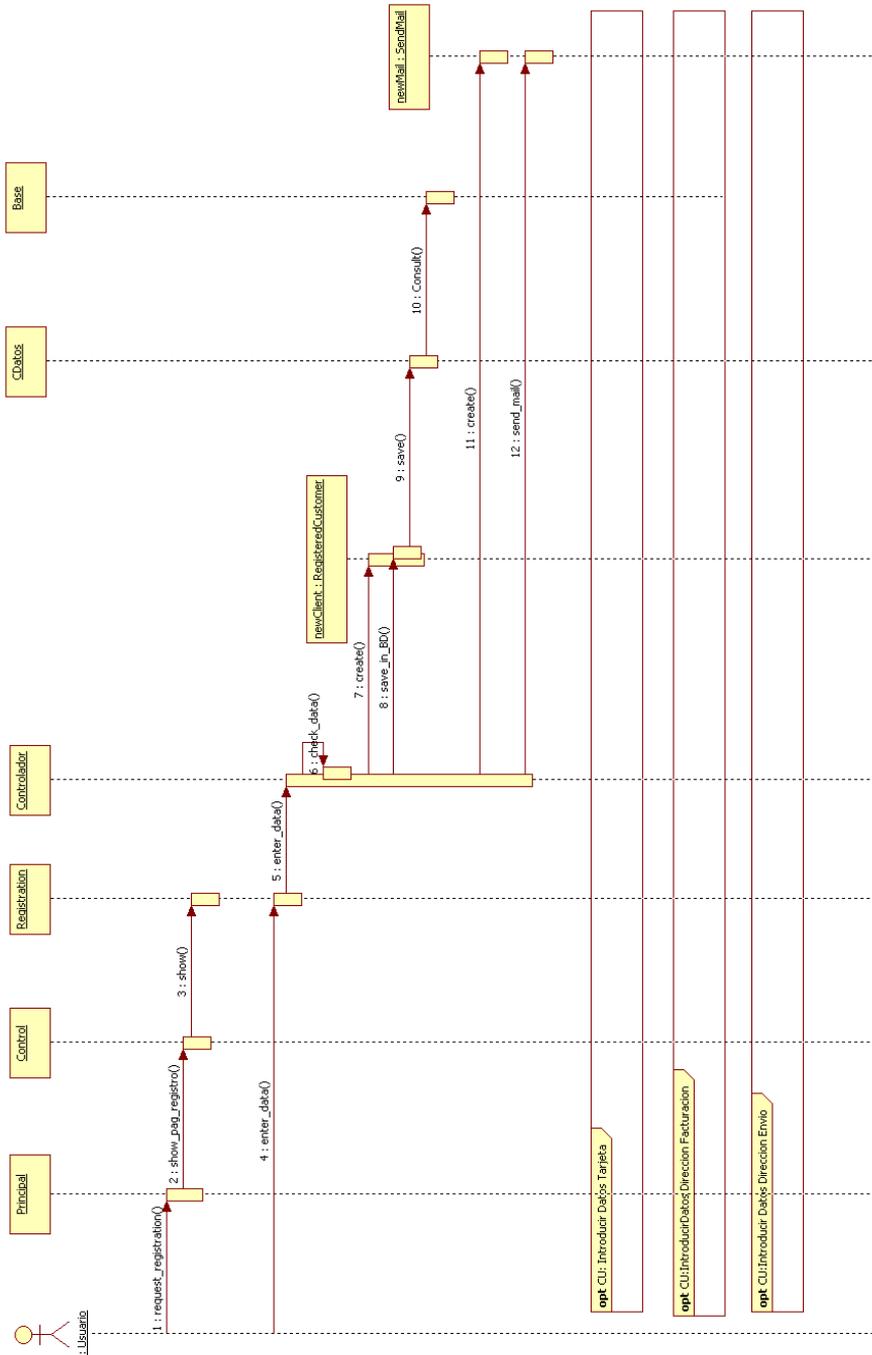


Figura 4-1 Diagrama de Secuencia del CU – 0001 Registrarse

4.1.1.2 CU - 0002: Identificarse

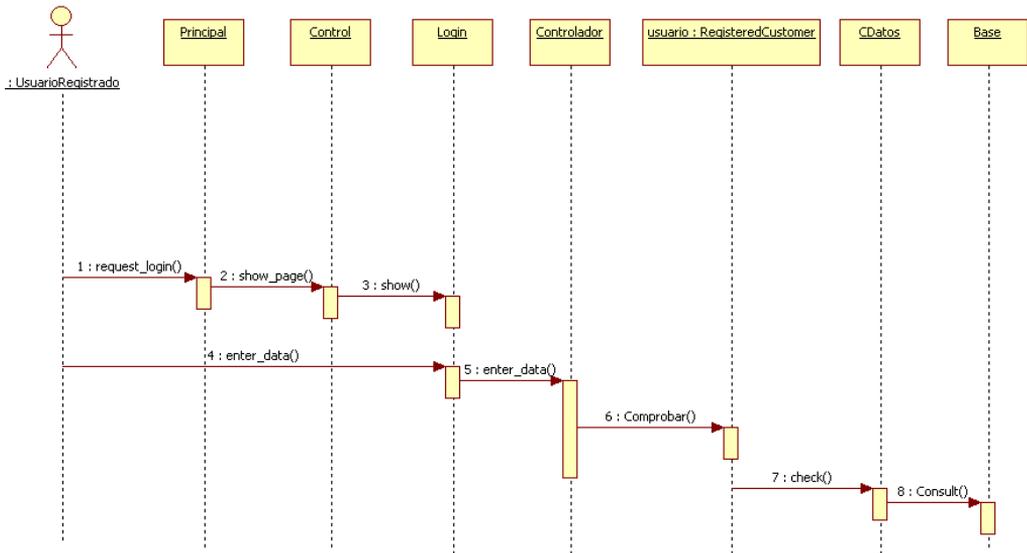


Figura 4-2 Diagrama de Secuencia del CU – 0002 Identificarse

4.1.1.3 CU-0003: Modificar datos

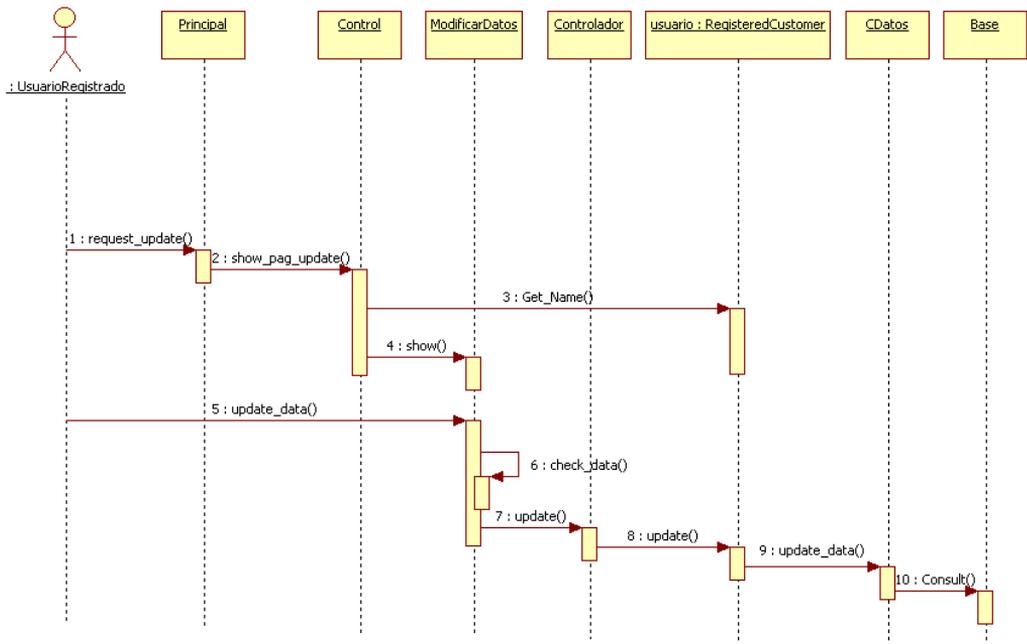


Figura 4-3 Diagrama de Secuencia del CU – 0003 Modificar Datos

4.1.1.4 CU-0004: Darse de Baja

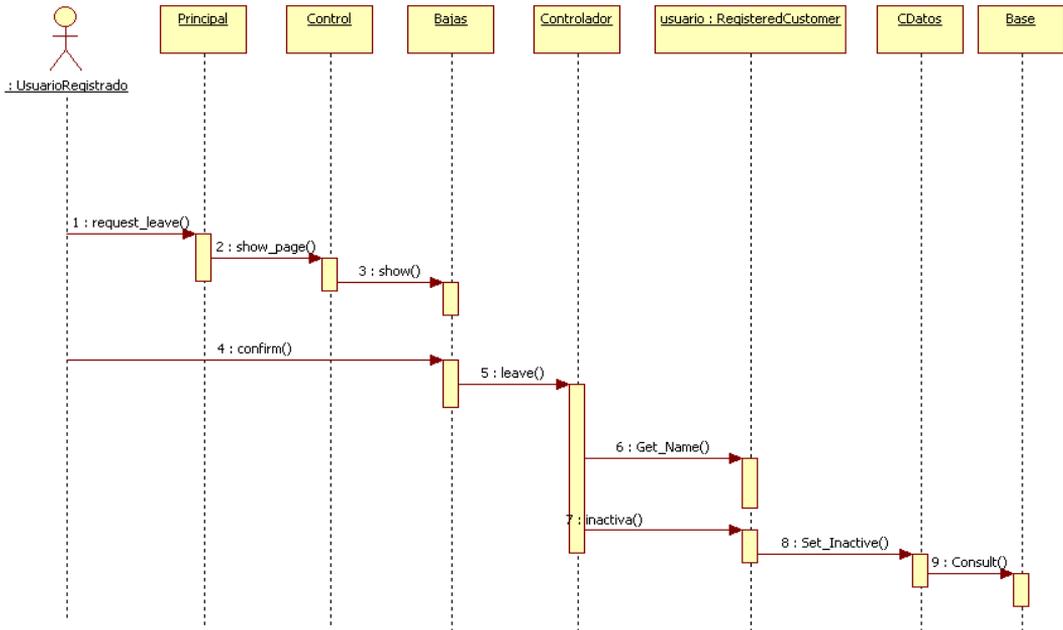


Figura 4-4 Diagrama de Secuencia del CU – 0004 Darse de Baja

4.1.2 Paquete: Credit Card Information

4.1.2.1 CU-0005: Insertar Datos Tarjeta

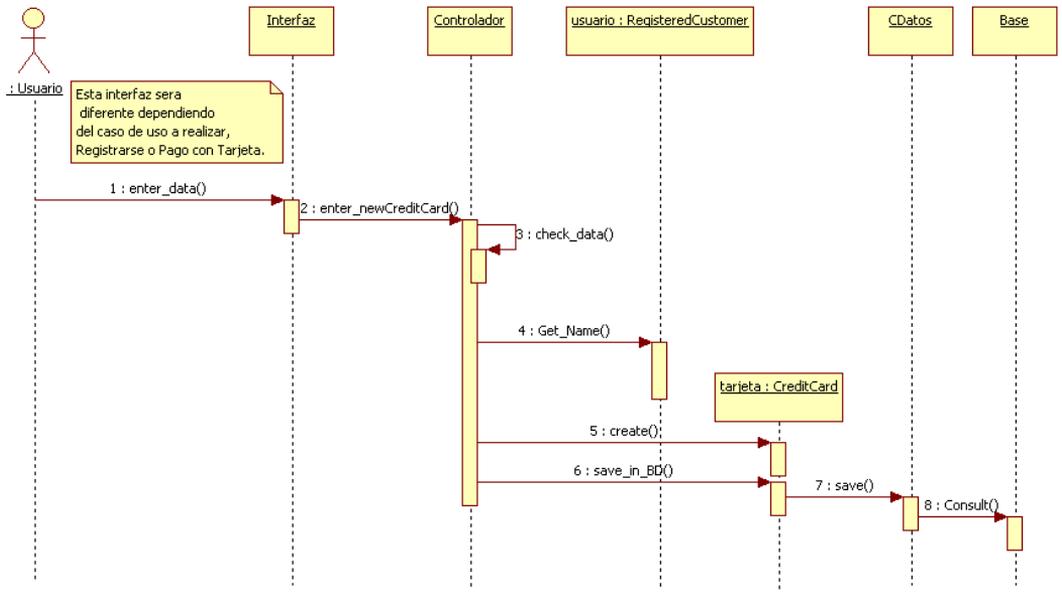


Figura 4-5 Diagrama de Secuencia del CU – 0005 Insertar Datos Tarjeta

4.1.2.2 CU-0006: Pago con Tarjeta

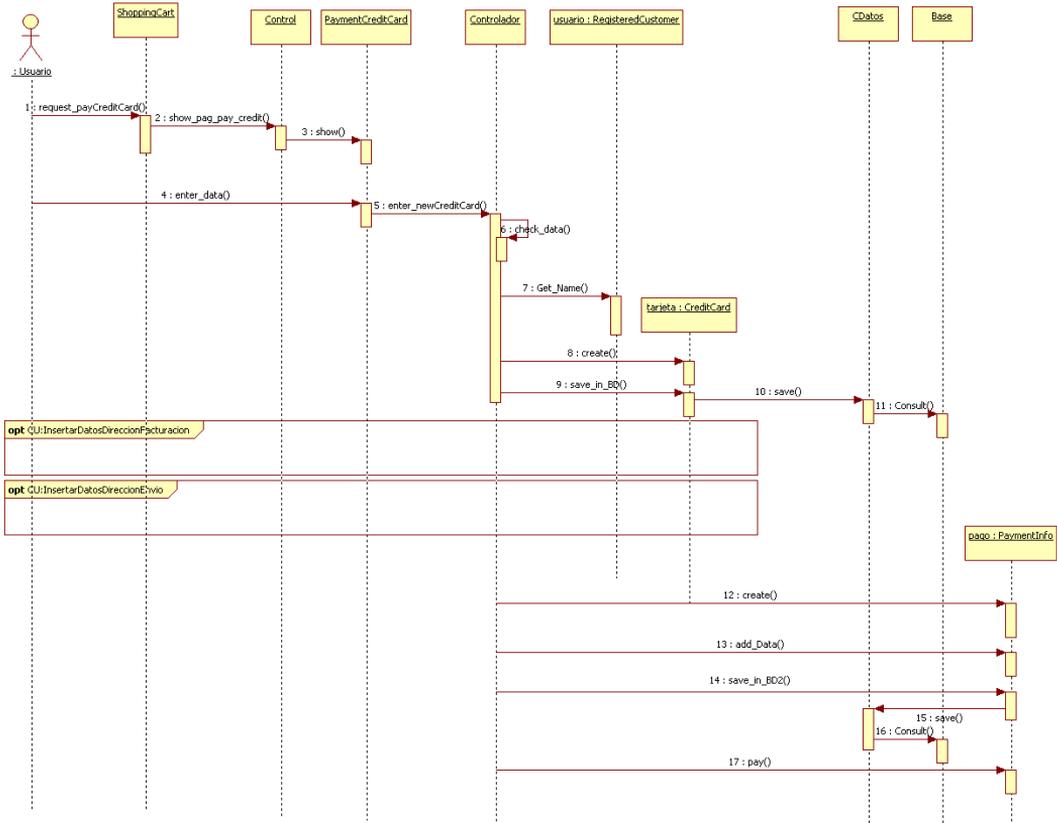


Figura 4-6 Diagrama de Secuencia del CU – 0006 Pagar con Tarjeta

4.1.3 Paquete: Shipping Address

4.1.3.1 CU-0007: Insertar datos dirección destino

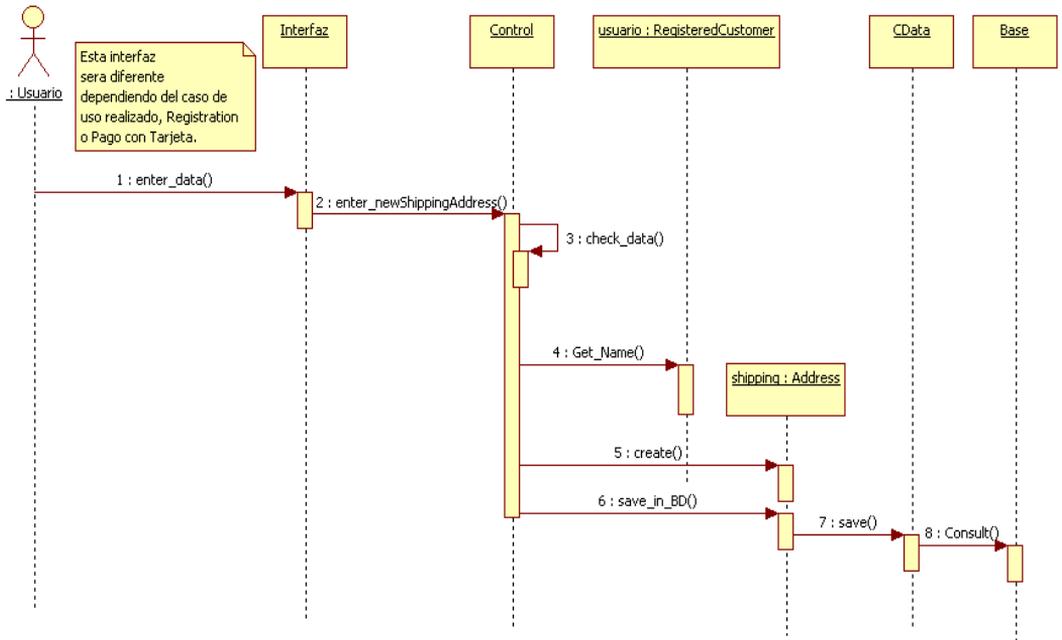


Figura 4-7 Diagrama de Secuencia del CU – 0007 Insertar Datos Envío

4.1.4 Paquete: Billing Address

4.1.4.1 CU-0008: Insertar datos dirección de facturación

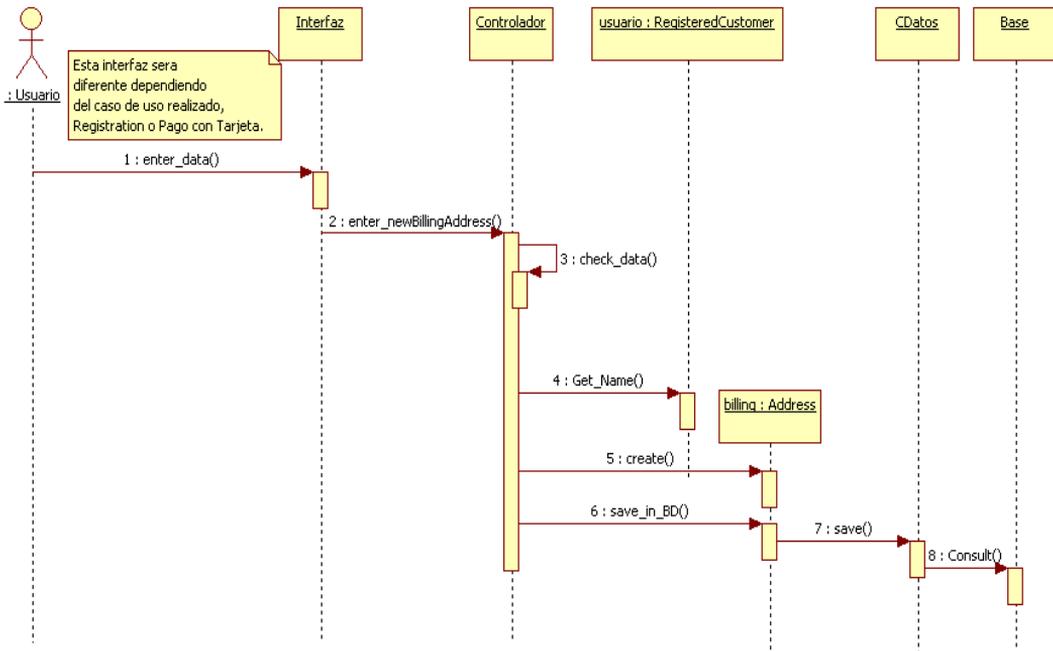


Figura 4-8 Diagrama de Secuencia del CU – 0008 Insertar Datos Facturación

4.1.5 Paquete: Quick CheckOut Profile

4.1.5.1 CU-0009: Utilizar Perfil Quick CheckOut

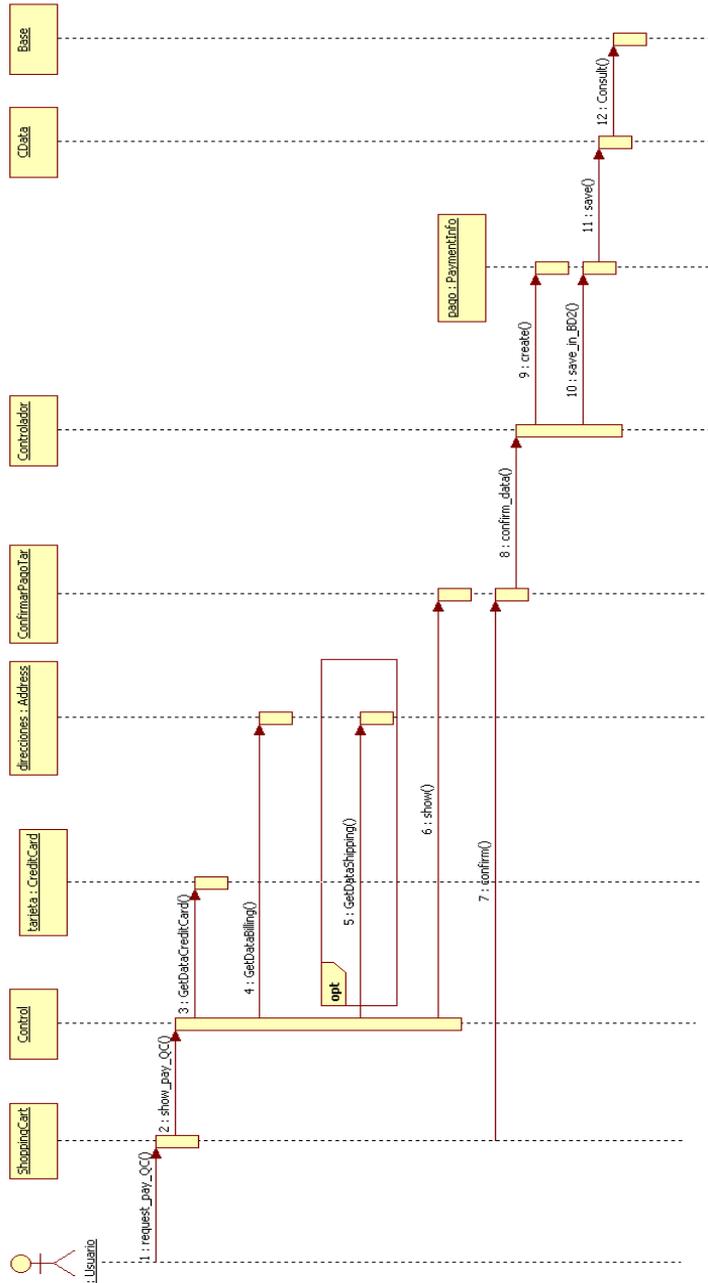


Figura 4-9 Diagrama de Secuencia del CU – 0009 Utilizar Perfil Quick Checkout

4.1.5.2 CU-0010: Modificar Perfil Quick Checkout

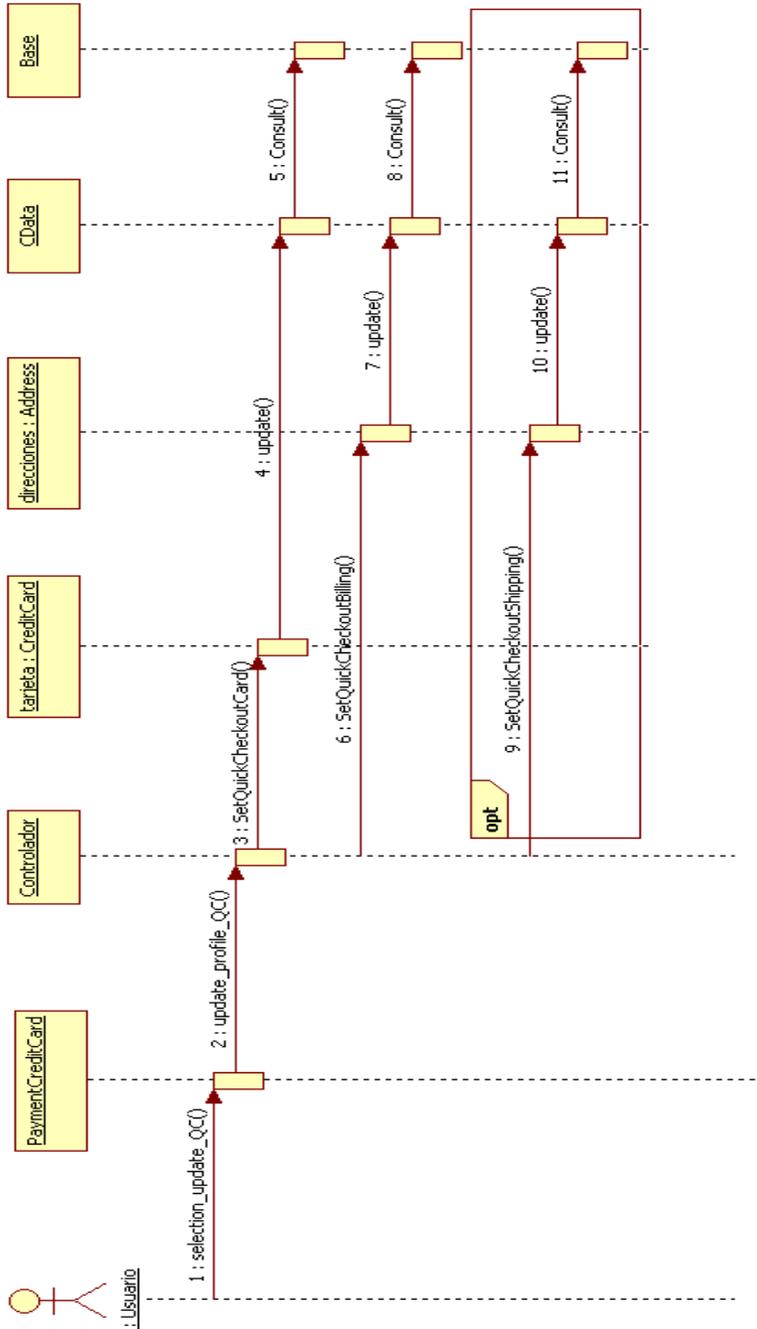


Figura 4-10 Diagrama de Secuencia del CU – 0010 Modificar Perfil Quick Checkout

4.2 Modelo de Diseño

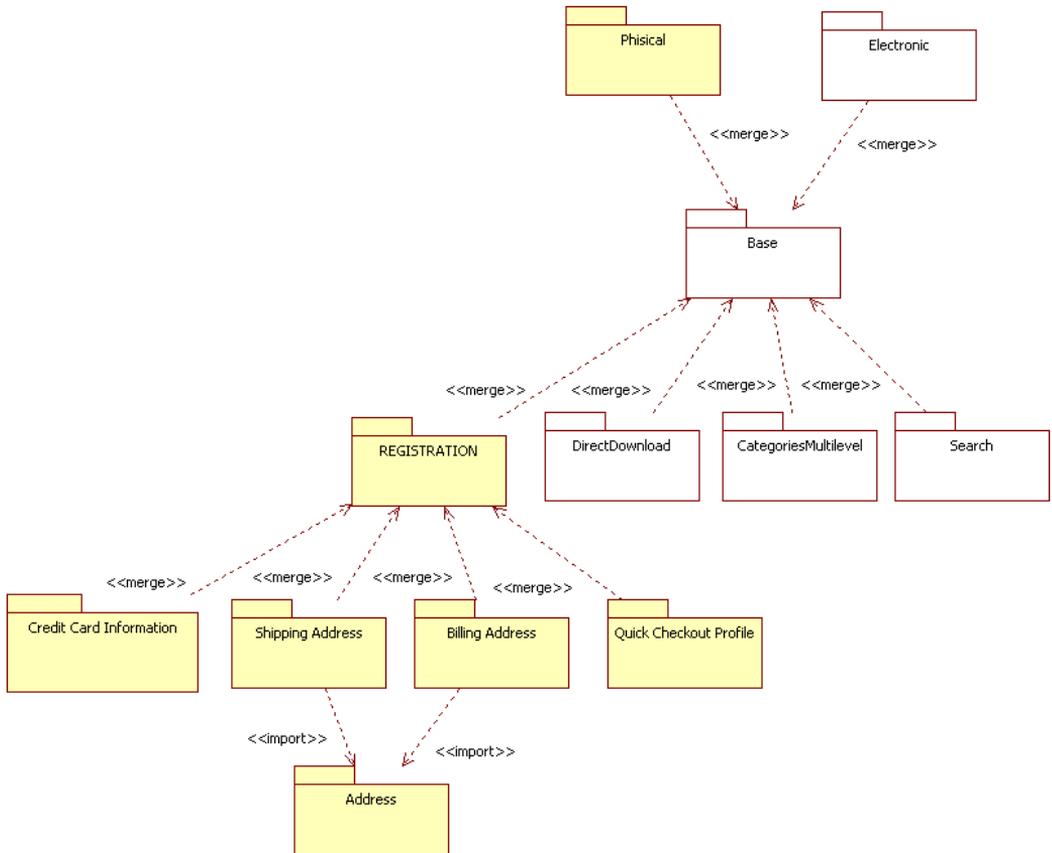


Figura 4-11 Modelo de Diseño

4.3 Diagramas de Clases

Este tipo de diagramas incluyen clases y sus relaciones estáticas, aunque puede contener otros elementos como objetos, enlaces, paquetes, interfaces, etc.

4.3.1 Paquete: Registration

4.3.1.1 Clases

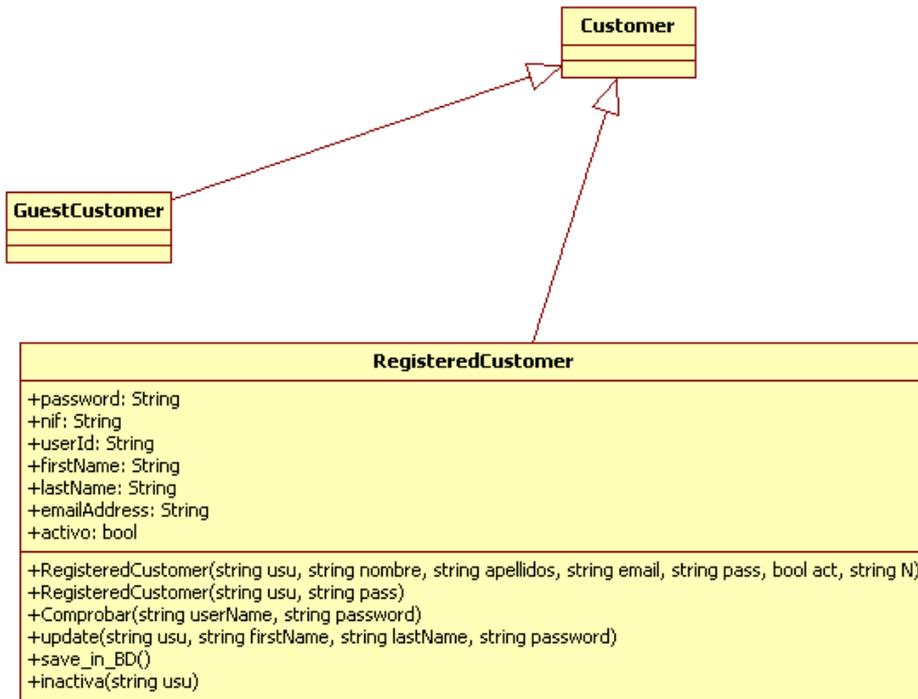


Figura 4-12 Diagrama de Clases del paquete Registration

4.3.1.2 Especificación

Partial class Customer
Es una clase abstracta que representa a todos los usuarios del sistema, tanto los registrados como los no registrados. No contiene atributos comunes porque la clase <code>GuestCustomer</code> no contiene información identificativa.

Partial class GuestCustomer
Es una clase abstracta que representa a los usuarios del sistema que no se han identificado.

Partial class RegisteredCustomer	
Es una clase que representa los atributos de los usuarios que se han registrado en el sistema.	
Atributos	Operaciones
+userId:string Representa el nombre que identifica al usuario en el sistema.	+RegisteredCustomer(string usu, string nombre, string apellidos, string email, string pass, bool act, string N) Método creador de la clase a la hora de crear un nuevo cliente e introducirlo en la base de datos.
+firstName:string Nombre	+RegisteredCustomer(string usu, string pass) Método creador de la clase a la hora de recuperar un usuario existente de la base de datos.
+lastName:string Apellidos	+Comprobar(string userName, string password) Método para comprobar si un usuario esta registrado o no lo esta.
+emailAddress:string Dirección de correo electrónico	+update(string usu, string firstName, string lastName, string password) Método proporcionado para que el usuario pueda cambiar algunos atributos como el nombre o la contraseña.
+password :string Cadena de caracteres que utilizara el usuario para identificarse en el sistema.	+save_in_BD() Utilizado para guardar en la base de datos un cliente.
	+inactiva(string usu) Usado para dar de baja al usuario en el sistema.

4.3.2 Paquete CreditCard

4.3.2.1 Clases

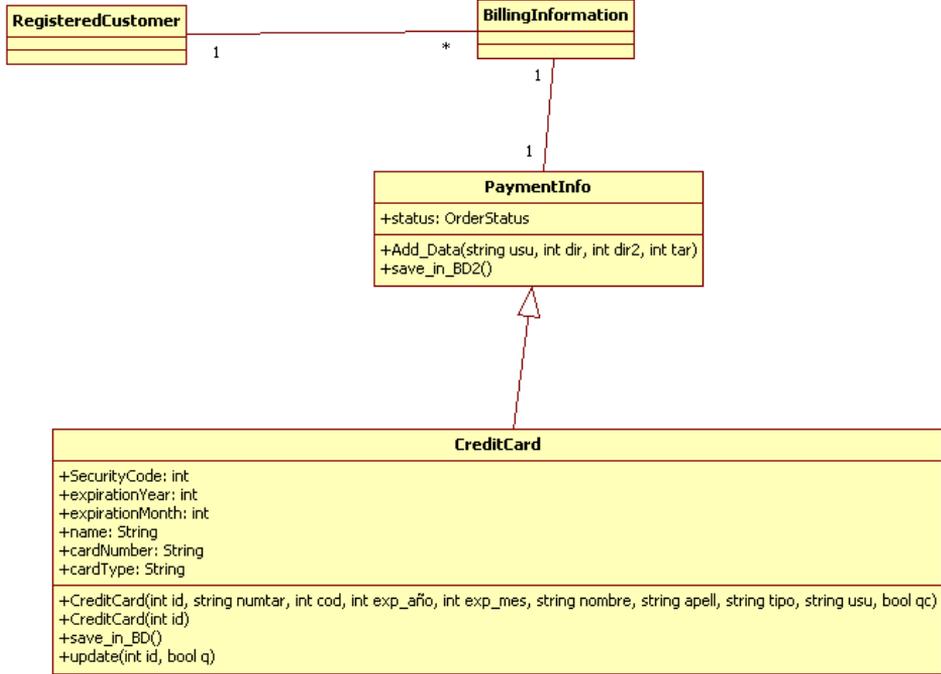


Figura 4-13 Diagrama de Clases del paquete Credit Card

4.3.2.2 Especificación

Partial class CreditCard	
Esta clase representa los atributos de las tarjetas introducidas en el sistema.	
Atributos	Operaciones
+securityCode:int CVC o código de validez de la tarjeta. Es un código de seguridad extra establecido.	+ CreditCard(int id, string numtar, int cod, int exp_año, int exp_mes, string nombre, string apell, string tipo, string usu, bool qc) Método creador de la clase a la hora de crear e introducir una nueva tarjeta en la base de datos.
+expirationYear:int Año de expiración de la tarjeta.	+CreditCard(int id) Método creador de la clase a la hora de identificar una tarjeta una vez creada.
+expirationMonth:int Mes de expiración de la tarjeta.	+ save_in_BD() Utilizado para introducir en la base de datos una nueva tarjeta.
+name:string Nombre del titular de la tarjeta	+update(int id, bool q) Método utilizado para establecer una tarjeta como nueva tarjeta por defecto en el perfil Quick Checkout.
+cardNumber:string Numero de la tarjeta de crédito.	
+cardType:string Tipo de tarjeta: VISA, MasterCard, American Express...	

DESARROLLO DE LA SOLUCIÓN

Partial class PaymentInfo	
Esta clase representa el pago asociado a una compra.	
Atributos	Operaciones
+order:Order Orden de compra a la que esta vinculada este pago.	+PaymentInfo() Método creador de la clase el cual inicializa el atributo estado a "pending".
+status:OrderStatus Estado del pago.	+pay() Método que cambia el estado a "completed", es decir que el pago se ha completado.
+userId:string Usuario registrado que ha realizado la compra	+cancel_payment() Método que cambia el estado del pago a "cancelled", es decir cancelado.
+num_direccion_facturacion:int Identificador de la direccion fiscal donde se ha registrado el pago.	+ Add_Data(string usu, int dir, int dir2, int tar) Usado para añadir datos de interés para realizar el pago con tarjeta como puede ser el usuario, la tarjeta o las direcciones.
+num_direccion_envio:int Identificador de la dirección de envío donde el cliente recibirá el pedido en el caso de que este contenga productos físicos.	+ save_in_BD2() Utilizado para introducir en la base de datos un nuevo pago realizado con tarjeta.
+id_tarj:int Identificador de la tarjeta con la que se ha realizado el pago.	

4.3.3 Paquete Shipping Address

4.3.3.1 Clases

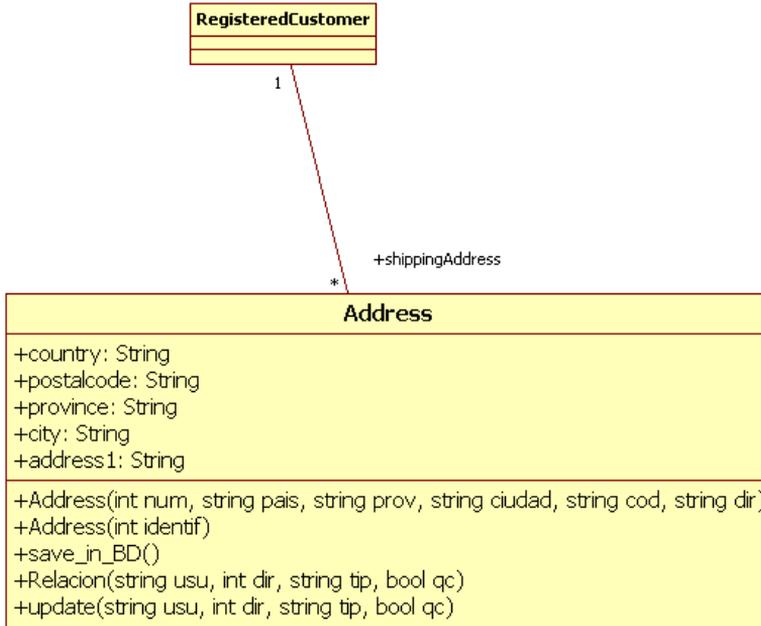


Figura 4-14 Diagrama de Clases del paquete Shipping Address

4.3.3.2 Especificación

Partial class Address	
Clase que representa los atributos de las direcciones de los usuarios del sistema. Estas direcciones pueden ser tanto de envío como de facturación.	
Atributos	Operaciones
+country:string Pais.	+Address (int num, string pais, string prov, string ciudad, string cod, string dir)() Método creador de la clase a la hora de crear una nueva dirección e introducirla en la base de datos.
+postalCode:string Codigo postal.	+Address(int identif) Método creador de la clase a la hora de recuperar una dirección existente de la base de datos.

DESARROLLO DE LA SOLUCIÓN

+province:string Provincia	+save_in_BD() Utilizado para guardar en la base de datos una dirección.
+city:string Municipio	+ Relacion(string usu, int dir, string tip, bool qc) Usado para introducir en la base de datos el tipo de dirección que es ,envió o facturación, y relacionarla con el usuario.
+address:string Dirección.	+ update(string usu, int dir, string tip, bool qc) Utilizado para establecer una dirección como parte del perfil Quick Checkout de un cliente

4.3.4 Paquete Billing Address

4.3.4.1 Clases

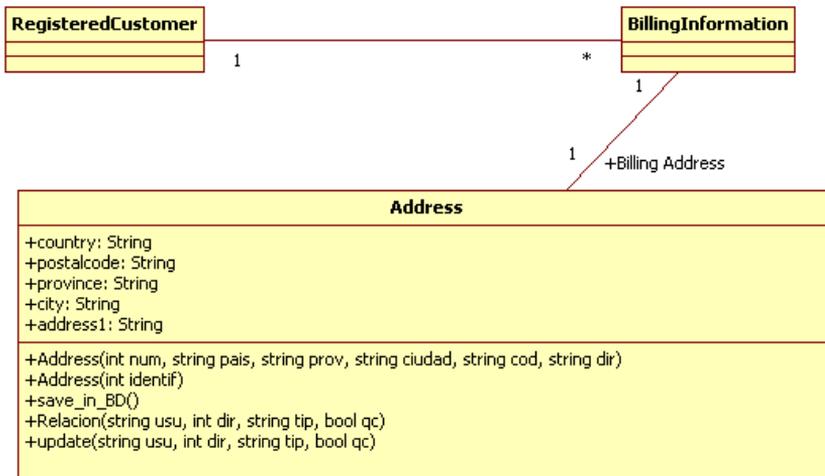


Figura 4-15 Diagrama de Clases del paquete Billing Address

4.3.4.2 Especificación

Partial class Address	
Clase que representa los atributos de las direcciones de los usuarios del sistema. Estas direcciones pueden ser tanto de envío como de facturación.	
Atributos	Operaciones
+country:string Pais.	+Address (int num, string pais, string prov, string ciudad, string cod, string dir)() Método creador de la clase a la hora de crear una nueva dirección e introducirla en la base de datos.
+postalCode:string Codigo postal.	+Address(int identif) Método creador de la clase a la hora de recuperar una dirección existente de la base de datos.
+province:string Provincia	+save_in_BD() Utilizado para guardar en la base de datos una dirección.
+city:string Municipio	+ Relacion(string usu, int dir, string tip, bool qc) Usado para introducir en la base de datos el tipo de dirección que es ,envió o facturación, y relacionarla con el usuario.
+address:string Dirección.	+ update(string usu, int dir, string tip, bool qc) Utilizado para establecer una dirección como parte del perfil Quick Checkout de un cliente

4.3.5 Paquete Quick CheckOut Profile

4.3.5.1 Clases

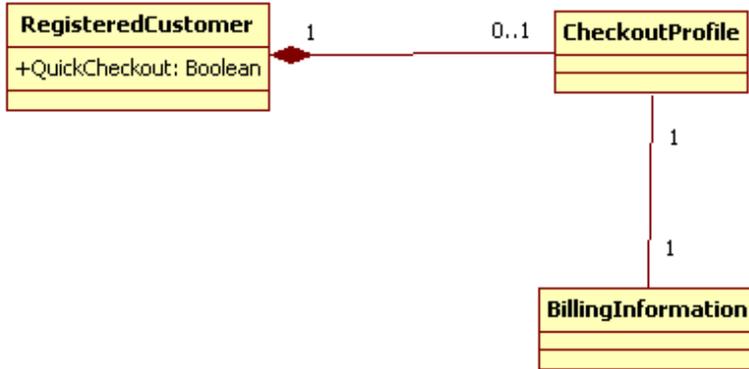


Figura 4-16 Diagrama de Clases del paquete Quick Checkout

4.3.5.2 Especificación

Partial class Checkout Profile
Es una clase abstracta que representa datos asociados a un usuario para poder realizar las compras.

4.4 Diseño de la Base de Datos

4.4.1 Gestor Base de Datos: SQL Server

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (SGBD) basada en el lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle o Sybase ASE.

4.4.1.1 Ventajas

- Soporte de transacciones.
- Gran estabilidad.
- Gran seguridad.
- Escalabilidad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos

4.4.1.2 Desventajas

- Solamente funciona bajo sistemas operativos de Microsoft.
- La instalación y operación requiere del Internet Explorer (IE) 4.0.
- La migración requiere un reinicio de la base de datos. El reinicio de todos los datos en una base de datos es un trabajo serio que invita a la potencial pérdida de datos.
- Ausencia de integridad referencial declarativa en cascada (DRI). La ausencia de una integridad referencial en cascada podría ser la desventaja más grande del Servidor SQL en comparación con las otras bases de datos dentro del mercado NET. Incluso Access ofrece soporte de este estilo. Se pueden utilizar triggers para compensar esta desventaja, aunque en otras bases de datos esta técnica no es necesaria.

4.4.2 Modelo relacional

Nuestra base de datos contiene las tablas ya creadas en el anterior proyecto (algunas de las cuales han tenido que ser modificadas para ajustarlas a nuevas necesidades que han ido surgiendo) y nuevas tablas que se han añadido posteriormente integrándose con las anteriores.

A continuación se muestran las tablas que ya existían anteriormente:

- **Catalog** (id_catalogo, name,description,icon,active)
- **Category** (id_category,id_catalogo, name, descripcion,icon, active)
- **Product** (id_producto,id_catalogo,name,description,icon,active,longDescription,standardDescription,suggestedRetailPrice,price)
- **Prod_agrupa_Cat** (id_categoria,id_producto)
- **OrderItem** (itemId,orderId,id_producto,name, price,status)
- **Orders** (orderId, date,email,status,total)
- **PaymentInfo** (id_payment,orderId,status,price,tipo)
- **Paypal** (id_payment,email)

A la hora de realizar las ampliaciones necesarias para poder implementar los nuevos paquetes, algunas de las tablas anteriores tuvieron que ser ligeramente modificadas, añadiéndose nuevos atributos.

El nuevo modelo relacional sería el siguiente:

- Tablas existentes que han permanecido igual:
 - **Catalog** (id_catalogo, name,description,icon,active)
 - **Category** (id_category,id_catalogo, name, descripcion,icon, active)
 - **Prod_agrupa_Cat** (id_categoria,id_producto)
 - **OrderItem** (itemId,orderId,id_producto,name, price,status)
 - **Orders** (orderId, date,email,status,total)
 - **Paypal** (id_payment,email)

- Tablas existentes que han sido modificadas:
 - **Product** (id_producto, id_catalogo, name, description, icon, active, longDescription, standardDescription, suggestedRetailPrice, price, type)
 - **PaymentInfo** (id_payment, orderId, status, price, tipo, userId, numero_tarjeta, num_dir_fact, num_dir_env)

- Tablas nuevas:
 - **RegisteredCustomer** (userId, firstName, lastName, emailAddress, password, activo, nif)
 - **Address** (num_direccion, address, postalcode, province, country, city)
 - **CreditCard** (id_tarj, cardNumber, securityCode, expirationYear, expirationMonth, name, apellidos, cardType, userId, QC)
 - **Relacion** (userId, num_direccion, tipo, QC)

4.4.3 Descripción de tablas

RegisteredCustomer: Almacena los datos personales de todos los clientes registrados en la tienda, también de los que se hayan dado de baja.

Nombre Atributo	Tipo	Descripción
userId	varchar(50)	Identificador para cada cliente, es único.
firstName	varchar(50)	Nombre del cliente
lastName	varchar(50)	Apellidos del cliente
emailAddress	varchar(50)	Dirección electrónica del cliente
password	varchar(50)	Contraseña
activo	bit	Indica si el cliente sigue activo o si se ha dado de baja del sistema
nif	varchar(50)	Número de identificación fiscal del cliente.

DESARROLLO DE LA SOLUCIÓN

Address: Almacena todas las direcciones que han utilizado los clientes, ya sea como dirección de envío de productos físicos (Shipping address) o como dirección fiscal (Billing address) donde se realiza la facturación de sus pedidos.

Nombre Atributo	Tipo	Descripción
Num_direccion	int	Identificador para cada dirección, es único.
address	varchar(MAX)	Nombre de la calle, número, piso, letra...
postalcode	int	Código postal
province	varchar(50)	Provincia
country	varchar(50)	País
city	varchar(50)	Población

CreditCard: Almacena todas las tarjetas de crédito que hayan sido utilizadas por los clientes registrados para pagar un pedido.

Nombre Atributo	Tipo	Descripción
Id_tarj	int	Identificador para cada dirección, es único.
cardNumber	varchar(50)	Número de la tarjeta de crédito
SecurityCode	int	Código de seguridad de la tarjeta (3 cifras en la parte de atrás)
expirationYear	int	Año de expiración de la tarjeta
expirationMonth	int	Mes de expiración de la tarjeta
name	varchar(50)	Nombre del titular
apellidos	varchar(50)	Apellidos del titular
cardType	varchar(50)	Tipo de tarjeta (Visa, Mastercard...)
userId	varchar(50)	Identificador del cliente que ha usado la tarjeta
QC	bit	Indicador de si la tarjeta esta siendo utilizada en el perfil Quick CheckOut

Relación: Esta tabla relaciona cada dirección con el cliente al que pertenece, y también si dicha dirección es de facturación o de envío.

Nombre Atributo	Tipo	Descripción
userId	varchar(50)	Identificador para cada cliente, es único.
num_direccion	int	Identificador para cada dirección, es único.
tipo	varchar(50)	Tipo de dirección (envío o facturación)
QC	bit	Indicador de si la dirección es la que figura en el perfil Quick CheckOut

Product: Almacena todos los productos de la tienda, indicando a que categoría y catálogo pertenecen. Esta tabla ya existía pero se le ha añadido una variable para poder distinguir entre productos electrónicos y productos físicos, ya que estos últimos no estaban considerados en el anterior proyecto. Además, en proyectos posteriores también se puede considerar la posibilidad de que haya otro tipo de productos, que serían servicios.

Nombre Atributo	Tipo	Descripción
id_producto	int	Identificador para cada producto, es único.
id_catalogo	int	Índice del catálogo al que corresponde
name	varchar(50)	Nombre del producto
description	varchar(100)	Descripción del producto
icon	varchar(100),	Imagen representativa
active	bit	Indica si esta o no activo en la tienda
thumbnail	varchar(MAX)	Miniatura de la imagen
longDescription	varchar(MAX)	Descripción larga
standardDescription	varchar(MAX)	Descripción estándar
suggestedRetailPrice	money	Precio recomendado de venta
price	money	Precio real de venta
size	varchar(50)	Tamaño del producto (electrónico o físico)
weight	varchar(50)	Peso del producto (si es físico)
link	varchar(MAX)	Url del producto para ser descargado
type	varchar(50)	Indica si el producto es electrónico, físico o servicio.

PaymentInfo: Almacena toda la información relativa al pago de la compra. También es una tabla ya implementada anteriormente, pero se le han añadido variables relativas al cliente registrado y a sus direcciones y tarjetas de crédito.

Nombre Atributo	Tipo	Descripción
orderId	varchar(50)	Identificador de la orden.
id_payment	varchar(50)	Identificador del pago de una orden.
Status	varchar (50)	Estado en que se encuentra el pago
Price	money	Precio total de la compra
Tipo	varchar(50)	Tipo de pago escogido
userId	varchar (50)	Identificador del cliente que realiza esta compra.
numero_tarjeta	int	Identificador de la tarjeta utilizada en la compra.
num_dir_fact	Int	Identificador de la dirección fiscal de esta factura.
num_dir_env	int	Identificador de la dirección de envío del producto físico si lo hubiera.

5 Implementación de la solución

El objetivo principal de este proyecto es continuar con el manejo de la variabilidad de productos finales en una misma línea de productos. A continuación explicaremos detenidamente los métodos y mecanismos utilizados para lograrlo.

5.1 Introducción a ASP.NET y formularios Web.

Los formularios Web ASP.NET permiten crear páginas Web programables, dinámicas, rápidas e interactivas. Una página de formulario Web consta de una interfaz de usuario y de una estructura lógica de programación. La interfaz está formada por un archivo que contiene lenguaje marcado como HTML y controles de servidor. Este archivo se denomina “página” y utiliza la extensión .aspx.

La lógica de programación se puede implementar con el archivo ASPX o en un archivo independiente escrito en cualquier lenguaje compatible como puede ser C# o Visual Basic.NET. Este archivo se denomina “code-behind” y lleva la extensión .aspx.cs o .aspx.vb, en función del lenguaje que se utilice.

Cuando se utiliza el modelo “code-behind” como estilo de programación, el desarrollador escribe código para responder a diferentes eventos, como pueden ser la carga de la página o cuando se hace click en algún control. El “code-behind” de ASP.NET es la principal diferencia que existe con asp y que anima a los programadores a construir aplicaciones con una clara separación entre el código y la interfaz. En teoría esto debería permitir a los diseñadores web centrarse en el diseño de la interfaz sin interferir en el trabajo del programador propiamente dicho.

5.2 Clases parciales

Como ya hemos explicado en el apartado 2.6, la manera de dar soporte en la implementación a la variabilidad de la línea de producto, es usando clases parciales de C#.

Las clases parciales (llamadas también tipos parciales) son una característica presente en algunos lenguajes de programación, como C# y VB.Net, que permiten que la declaración de una clase se realice en varios archivos de código fuente, rompiendo así la tradicional regla "una clase, un archivo". Será tarea del compilador tomar las porciones de los distintos archivos y fundirlas en una única entidad.

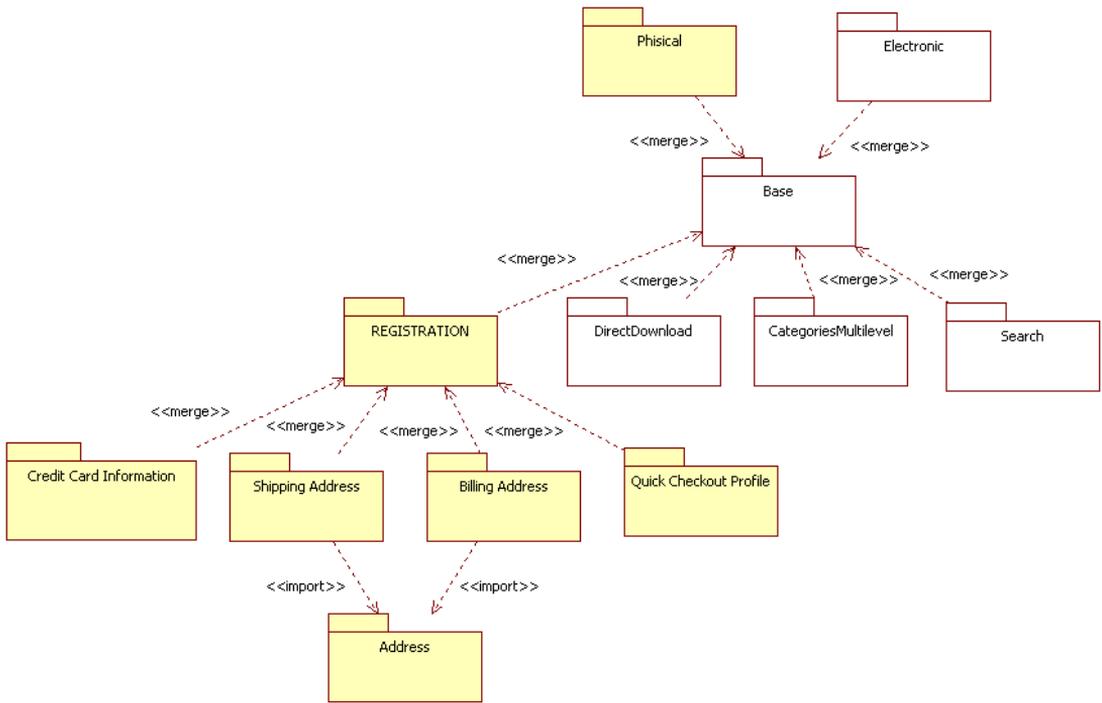


Figura 5-1 Diseño del sistema dividido en paquetes

5.3 Interfaz de usuario

5.3.1 Master.Page

5.3.1.1 Introducción

Una master.page funciona como contenedor de plantillas y página de combinación de las páginas de contenido de la aplicación Web ASP.NET. La característica de las "Master Pages" nos proporciona la habilidad de definir una estructura y unos elementos de interfaz comunes para nuestro sitio, tales como la cabecera de página o la barra de navegación, en una ubicación común denominada "master page", para ser compartidos por varias páginas del sitio. Esto evita la duplicación innecesaria de código para estructuras o comportamientos del sitio que son compartidos.

La principal ventaja de la master.page, es que tan sólo tenemos que crear una o varias plantillas para todo nuestro sistema. Este hecho facilitará enormemente el mantenimiento, ya que solamente con modificar la master.page, los cambios se verán reflejados en todas las páginas que la utilicen. Otra ventaja muy importante es que tendremos soporte en diseño WYSIWYG (lo que ves, es lo que obtienes), en las páginas.aspx, y podremos ver en todo momento cómo será la presentación final.

5.3.1.2 Diseño de Master.Page

Como hemos señalado en el apartado anterior, una master.page es la interfaz común a una aplicación Web. Por esta razón y debido a que nuestro proyecto es una ampliación de un proyecto anterior, hemos querido mantener el diseño realizado por nuestros compañeros.

Este diseño consta de una cabecera en la parte superior donde se alojará la imagen corporativa de la empresa cliente, es decir de la empresa que se dedica al comercio. A la izquierda de la página nos encontramos los distintos menús de funcionalidad que serán accesibles en todo momento; y por último tenemos la parte central, que corresponde al contenido de cada página concreta que albergará la master.page. Este área se limita a tener un contenedor “ContentPlaceHolder” que será la parte que se podrá editar en cada página aspx de forma individual.

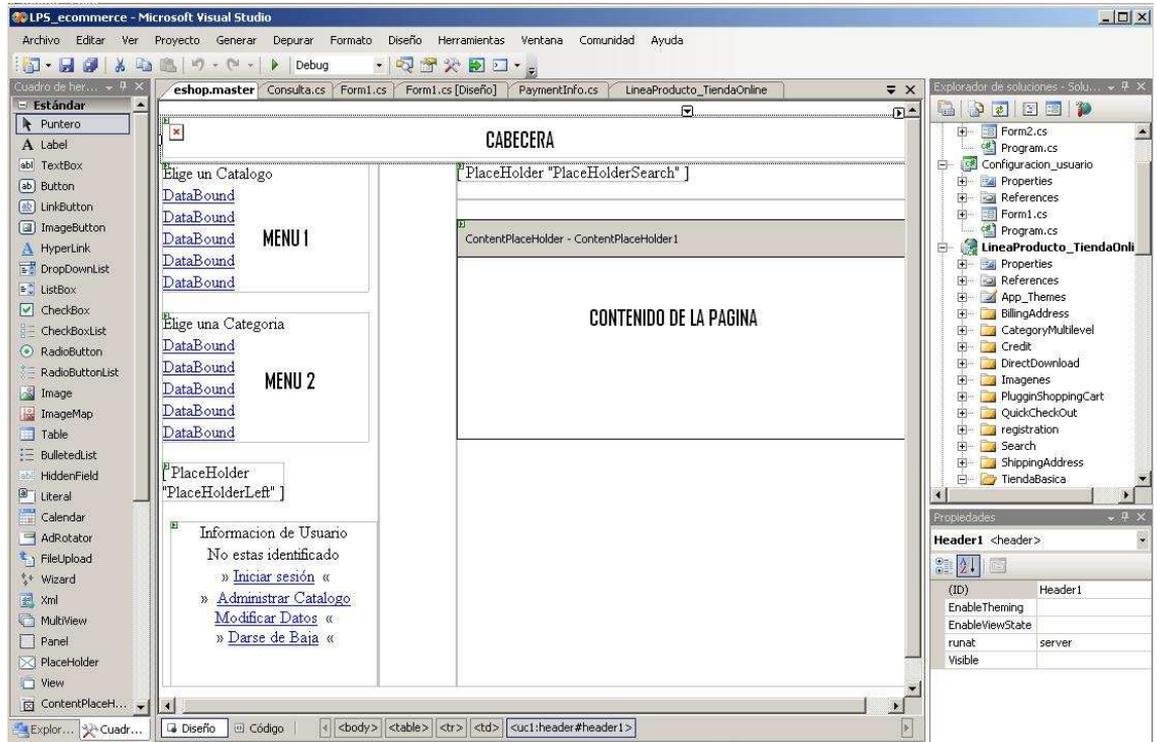


Figura 5-2 Diseño de la Master.Page en el Visual Studio

DESARROLLO DE LA SOLUCIÓN

También podemos observar que esta página presenta dos de los denominados contenedores o “Placeholder”, que nos han sido de gran ayuda a la hora de agregar menús que dan paso a las nuevas funciones que hemos añadido al proyecto inicial.

Como podemos observar, este diseño carece de colores y la letra es por defecto. La razón es que se han utilizado hojas de estilo CSS que por su tecnología no se aplican hasta el momento en que el navegador cargue la página. El resultado se ve reflejado en la siguiente captura:

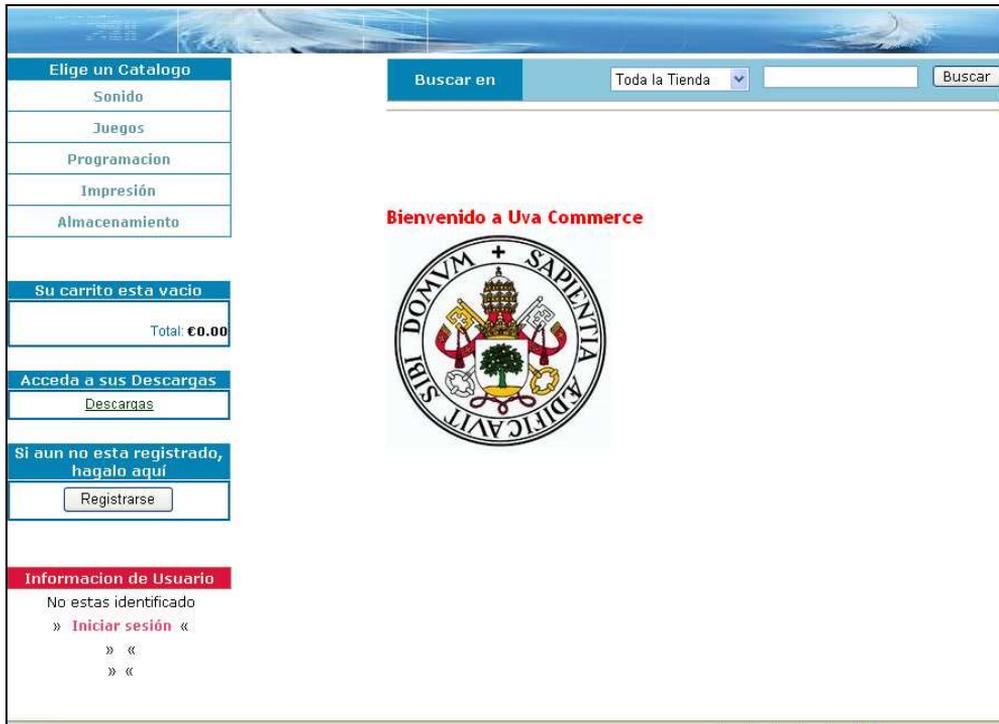


Figura 5-3 Diseño de la Master.Page en el navegador

5.3.2 Hoja de estilos CSS

5.3.2.1 Introducción

Una 'hoja de estilo' es una plantilla, guía o conjunto de instrucciones, que dicta al navegador como debe mostrar el contenido de una o varias páginas Web. Cualquier cambio en la plantilla de estilo se refleja en un cambio inmediato en la apariencia de las páginas relacionadas, sin necesidad de modificar físicamente éstas.

El creador de la página define estilos (tamaño de letra, color, tipo de fuente, márgenes, etc.) que se visualizarán con preferencia a los definidos por defecto en el propio navegador. Si después desea

cambiar la apariencia (nuevos márgenes, etc.) bastará que cambie la definición de estilo. El documento en si no se modifica.

Las CSS (hojas de estilo en cascada) mejoran las posibilidades de diseño y presentación de documentos en la red, facilitando además su mantenimiento, ya se trate de un único archivo HTML, o de grandes sitios, con multitud de páginas.

5.3.2.2 Beneficios de utilizar CSS

De acuerdo a las características de nuestro proyecto, que, repetimos, es una ampliación de otro proyecto anterior y que seguramente precede a otros muchos, el uso de hojas de estilo es idóneo para poder establecer unas reglas y una homogeneidad entre todos los proyectos que van a trabajar en esta línea de productos.

Debido a esto, la interfaz tiene que estar definida de forma clara y precisa desde el primer momento, ya que es una de las partes mas importantes en una aplicación Web y mas aún en un sistema de comercio electrónico. Esta técnica nos ha supuesto un gran ahorro de tiempo ya que al reutilizar la hoja de estilos realizada por nuestros compañeros, tan solo hemos tenido que añadir las reglas que hemos creído necesarias.

Asimismo si futuros desarrolladores decidieran utilizar una nueva hoja de estilos creando nuevas reglas con el mismo nombre que las existentes o añadiendo alguna más, podrían hacerlo fácilmente, señalando este cambio en el archivo web.config.

5.3.2.3 Diseño de hojas de estilos

Vamos a dedicar este apartado a explicar con más detalle cómo es nuestra hoja de estilos en concreto. Como decíamos anteriormente, nosotras hemos reutilizado las reglas existentes creadas por nuestros compañeros y hemos añadido reglas correspondientes a nuestros paquetes siguiendo cierta homogeneidad, para que el diseño de la pagina resulte uniforme.

En .NET, el archivo CSS debe encontrarse ubicado en el directorio App_Themes del proyecto para poder utilizar las hojas de estilo. En el caso de que se quieran utilizar dos o más estilos se deben crear subdirectorios. En nuestro caso se utiliza tan solo un estilo llamado EShopDefault.

También se necesita incluir en el archivo de configuración del sistema “web.config” un tag que haga referencia al tema que se va a aplicar, en nuestro caso sería

```
" <pages theme='EShopDefault' />"
```

La imagen mostrada a continuación contiene un fragmento de la hoja de estilos que hemos incluido en la de nuestros compañeros.

```
.PaymentMethods
{
    font-size: 14px;
    color: #0583b5;
    font-family: Arial;
    font-weight: bold;
}

.RegTitles
{
    font-size: 15px;
    color: #ff0000;
    font-family: Verdana,
Arial;
}

.Registration
{
    font-weight: bold;
    font-size: 16px;
    color: #0583b5;
    font-family: Sans-Serif;
    font-variant: normal;
}

.DatosRegistration
{
    font-weight: bold;
    font-size: 14px;
    font-family: Sans-Serif;
}

.Notificaciones
{
    font-weight: bold;
    font-size: 12px;
    color: #cc0000;
    font-family: Arial;
}
```

Figura 5-4 Ejemplo de la hoja de estilos

5.4 Solución a la variabilidad en la interfaz de usuario

En este apartado procederemos a explicar los mecanismos usados para crear interfaces de usuario variables en los diferentes productos finales de la línea.

5.4.1 Introducción a controles de usuario

Se denominan controles de usuario a los controles personalizados y reutilizables que se pueden crear con las mismas técnicas que se aplican para las páginas Web. Es una página de formulario normal, con la diferencia de que no se incluyen las etiquetas <html>, <head>, <body> y <form> , las cuales derivan de la clase System.Web.UI.UserControl.

Al igual que las páginas Web, estos controles contienen código estático HTML y su correspondiente modelo “code-behind”. Estos controles tienen extensión .aspx.

Una vez diseñado el control de usuario, con todos sus elementos, el paso siguiente es incluirlo en la página web propiamente dicha con extensión .aspx

5.4.2 Controles de Usuario dinámicos

Después de estudiar detenidamente las ventajas y desventajas de este tipo de controles, se llegó a la conclusión de que la inserción de controles de usuario en tiempo de ejecución era la manera más eficiente y a la vez sencilla de realizar una web totalmente configurable por el cliente donde se pudieran añadir o no ciertos formularios.

Es decir, obtendremos la misma página web con más o menos funciones dependiendo de los paquetes que se añadan según las especificaciones del cliente.

Por ejemplo, las figuras mostradas a continuación son dos capturas de pantalla de la página que muestra las distintas formas de pago de dos productos finales distintos.

The screenshot displays a web shopping cart interface. On the left, there is a navigation menu with categories: 'Elige un Catalogo' (Sonido, Juegos, Programacion, Impresión, Almacenamiento), 'Su carrito de la compra contiene' (with a link to 'ver detalles'), 'Acceda a sus Descargas', and 'Si aun no esta registrado, hazlo aquí'. At the bottom left, there is a user information section stating 'No estas identificado' and a link to 'Iniciar sesión'. The main content area features a search bar at the top with a dropdown menu set to ' Toda la Tienda ' and a 'Buscar' button. Below the search bar, it states 'Su Carrito de la Compra contiene:' followed by a table listing the product 'SQLOne 3.0' with a price of '€20.00' and an 'Eliminar' button. The total is shown as 'Total: €20.00' with a 'Continuar Comprando' button. Under the heading 'Métodos de pago', there is a section for 'Pago por Pay-Pal' with an 'Aceptar' button.

Producto	Precio	
SQLOne 3.0	€20.00	Eliminar
Total: €20.00		Continuar Comprando

Métodos de pago

Pago por Pay-Pal

[Aceptar](#)

Figura 5-5 Producto final que incluye un único método de pago



Figura 5-6 Producto final que incluye varios métodos de pago

En la primera de las figuras podemos observar que tan sólo se puede realizar el pago por medio de PayPal, mientras que en la segunda existen tres formas de pago: PayPal, tarjeta de crédito y mediante el método rápido.

Esto ha sido gracias a la inserción de controles de usuario en tiempo de ejecución o de forma dinámica. Estos controles son añadidos automáticamente por el sistema dependiendo de los paquetes seleccionados. Las ventajas proporcionadas al usar esta técnica es el poder reutilizar los controles tantas veces como se desee y la completa independencia entre los paquetes.

Para conseguir añadir los controles dinámicamente utilizamos los denominados Placeholder, que son contenedores de todo tipo de controles, entre ellos los controles de usuario. Del proyecto anterior al nuestro hemos sabido aprovechar algunos de los contenedores proporcionados por ellos. A la vez, hemos dejado dispuestos Placeholder que consideramos serán necesarios para los futuros desarrolladores de esta línea.

Llegados a este punto sólo nos queda por detallar la forma en la que el sistema reconoce los paquetes que proporcionan las funciones requeridas por el cliente. También tendrá que reconocer el lugar donde debe añadir los controles correspondientes a cada paquete.

El mecanismo desarrollado para solucionar el problema ha sido el uso de un fichero XML en el que se encuentran detallados los paquetes disponibles, si se añaden al producto final o no (true o false) y los archivos incluidos en cada uno de ellos.

5.4.3 Agregar controles de forma dinámica

Como decíamos en el apartado anterior el manejo de la variabilidad de productos a nivel de interfaz lo conseguiríamos a través del uso de controles de usuario, y su inserción en las páginas con ayuda de un fichero de configuración XML.

En el fichero XML encontramos la información necesaria en cuanto a los paquetes incluidos (activados), los controles de usuario que interactúan con el paquete base y la ruta física donde se encuentran:

```
<paquetes>
  <paquete>
    <nombre>-- Nombre y ruta del paquete --- </nombre>
    <activado>---- true o false --</activado>
    <archivos>
      <archivo>-----controldeusuariol.ascx asociado -----
</archivo>
      <archivo>-----controldeusuario2.ascx asociado -----
</archivo>
    </archivos>
  </paquete>
</paquetes>
```

El criterio propuesto por nuestros compañeros para conocer en que página se aloja cada control se basa en que el nombre de cada control de usuario tenga como primeras letras el nombre de la página que lo va a contener, seguido del nombre propio del control junto a la extensión propia ascx. Es decir:

Nombre_páginaControldeUsuario.ascx

DESARROLLO DE LA SOLUCIÓN

A continuación se muestra el fragmento del archivo “paquetes.xml” que hemos añadido al ya existente:

```
<?xml version="1.0" encoding="utf-8" ?>
<paquetes>
<paquete>
  <nombre>../registration</nombre>
  <activado>>true</activado>
  <archivos>
    <archivo>LeftMasterRegistration.ascx</archivo>
    <archivo>RegCredentials.ascx</archivo>
  </archivos>
</paquete>
<paquete>
  <nombre>../Credit</nombre>
  <activado>>true</activado>
  <archivos>
    <archivo>ShoppingCartCreditCard.ascx</archivo>
    <archivo>RegCreditCard.ascx</archivo>
  </archivos>
</paquete>
<paquete>
  <nombre>../ShippingAddress</nombre>
  <activado>>true</activado>
  <archivos>
    <archivo>RegShip.ascx</archivo>
    <archivo>ConfirmarShip.ascx</archivo>
  </archivos>
</paquete>
<paquete>
  <nombre>../BillingAddress</nombre>
  <activado>>true</activado>
  <archivos>
    <archivo>RegBill.ascx</archivo>
    <archivo>ConfirmarBill.ascx</archivo>
  </archivos>
</paquete>
<paquete>
  <nombre>../QuickCheckout</nombre>
  <activado>>true</activado>
  <archivos>
    <archivo>ShoppingCartQuick.ascx</archivo>
    <archivo>RegQuick.ascx</archivo>
  </archivos>
</paquete>
</paquetes>
```

Finalmente, hemos de señalar que para el correcto funcionamiento del sistema cada página debe contener un método de lectura del fichero xml en su “code-behind”, para reconocer si tiene que incluir algún control de usuario.

5.5 Registro de usuarios

La tarea de registrarse en una página de comercio electrónico es un paso que a la larga puede ahorrarnos mucho tiempo, ya que esos datos permanecen almacenados y se pueden reutilizar para la realización de futuras compras. Además, esta información puede ser de gran utilidad para los dueños de la tienda a la hora de realizar estrategias de mercado y otros estudios similares.

También cabe señalar que la mayoría de las tiendas online guardan privilegios para los usuarios registrados a los que los no registrados no tienen acceso. En nuestro caso los usuarios registrados pueden pagar sus productos por tres métodos diferentes: PayPal, tarjeta de crédito y el pago por método rápido a diferencia de los no registrados que tan sólo tienen la posibilidad de pagar a través de PayPal.

Hemos manejado el registro de usuarios y el acceso a privilegios solo aptos para usuarios registrados a través de los denominados roles. En nuestro proyecto concretamente, se ha compaginado la sincronización de dos roles: el rol de usuario que se registra en la página y el rol de usuario administrador creado durante el proyecto anterior.

Como decíamos, la clase Roles desarrollada por ASP.NET proporciona la posibilidad de asignar un grupo de usuarios a ciertas funciones y permite controlar el acceso a partes o características diferentes de la aplicación Web. El manejo de Roles y la inserción de usuarios a estos roles se puede realizar mediante la herramienta de configuración de ASP.NET que podemos encontrar pulsando el icono señalado en la siguiente figura que se encuentra en el explorador de soluciones de Visual Studio.

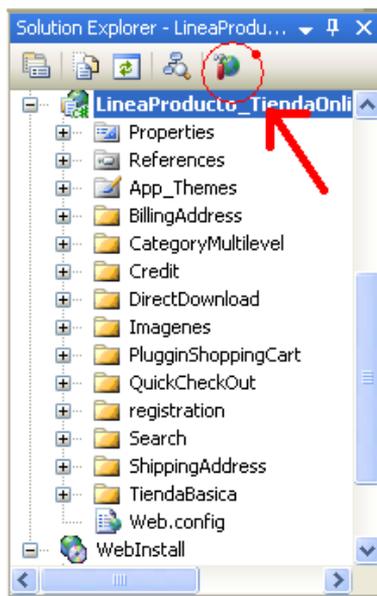


Figura 5-7 Herramienta de configuración de ASP. NET

DESARROLLO DE LA SOLUCIÓN

En el desarrollo, hemos creado dos roles: el rol “administrators” al que solamente pertenece el usuario administrador y el rol “users” al que pertenecen el resto de usuarios registrados del sistema. En nuestro caso añadimos y eliminamos los usuarios registrados a su rol mediante líneas de código utilizando los métodos proporcionados por la clase Roles:

```
Roles.AddUserToRole(nom_usu.Text, "users");  
Roles.RemoveUserFromRole(usu, "users");
```

Por último, hemos de señalar que para que todo esto funcione se ha de añadir al archivo web.config el siguiente tag:

```
<roleManager enabled="true" />
```

5.6 Envío de correos electrónicos

Al igual que ocurre en la mayoría de las páginas de comercio electrónico publicadas actualmente en la red, nuestra aplicación también envía un correo a los usuarios cuando estos se acaban de registrar para recordarles el nombre de usuario y la contraseña con la que deben acceder al sistema a partir de ese momento. Para ello, en primer lugar nos creamos una cuenta en Gmail (<http://www.gmail.com/>), ya que ésta es una de las corporativas que permite el envío de correos usando su servidor. Después creamos una clase SendMail.cs en la que asociamos todas las características necesarias para el envío: tanto las del mensaje que se va a enviar, como las del protocolo SMTP (Protocolo Simple de Transferencia de Correo).

5.6.1 Protocolo SMTP

Simple Mail Transfer Protocol es un protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras o distintos dispositivos (PDA's, teléfonos móviles, etc.).

SMTP se basa en el modelo cliente-servidor, donde un cliente envía un mensaje a uno o varios receptores. La comunicación entre el cliente y el servidor consiste enteramente en líneas de texto compuestas por caracteres ASCII. El tamaño máximo permitido para estas líneas es de 1000 caracteres.

Al igual que a la hora de registrarse, el usuario también recibirá un mail con todos los datos de su compra una vez que se hayan realizado y confirmado el pedido y el pago.

6 Pruebas

6.1 Introducción

Uno de los últimos pasos realizados tras terminar la implementación del código fuente, fue la exhaustiva realización de pruebas para asegurarnos del correcto funcionamiento de la aplicación..

El objetivo de esta fase no es convencerse de que el programa funciona bien sino ejercitarlo con la peor intención a fin de encontrarle fallos. Para ello hemos llevado a cabo una serie de casos de prueba diseñados para determinar si los requisitos establecidos durante el diseño del proyecto se han cumplido total o parcialmente.

El desarrollo de las pruebas se ha llevado a cabo desde dos perspectivas diferentes:

- Las pruebas de caja blanca son aquellas que se realizan sobre las funciones internas de un módulo. Se comprueba así la lógica interna de un programa.
- En las pruebas de caja negra no conocemos la implementación del código, sólo la interfaz. Tan sólo podemos probar dando distintos valores a las entradas y salidas.

6.2 Pruebas de Caja Blanca

Estas pruebas también llamadas estructurales o de caja transparente, se han ido realizando muy frecuentemente y a medida que desarrollábamos el código, de forma que todas las clases y la colaboración entre ellas ha quedado probada.

Los casos de prueba nos han ayudado a:

- Garantizar que se ejercitan por lo menos una vez todos los caminos.
- Ejercitar todas las decisiones lógicas por sus vertientes Cierto y Falso.
- Ejecutar todos sus ciclos en sus límites y límites operacionales.
- Ejecutar las estructuras internas de datos para asegurar su validez.

6.3 Pruebas de Caja Negra

Este tipo de pruebas está enfocado a comprobar que los requisitos funcionales se han cumplido. Es decir, la prueba de caja negra permite obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. No son una alternativa a las pruebas de caja blanca sino que se han de realizar los dos por separado y de forma complementaria, de forma que se puedan detectar tipos de errores diferentes.

Desde un punto de vista parecido la prueba funcional o de caja negra se centra en el estudio de la especificación del software, del análisis de las funciones que debe realizar, de las entradas y de las salidas.

6.3.1 Partición Equivalente

Es el método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

Esta técnica trata cada parámetro como un modelo algebraico donde unos datos son equivalentes a otros. Si logramos partir un rango excesivamente amplio de posibles valores reales a un conjunto reducido de clases de equivalencia, entonces es suficiente probar un caso de cada clase, pues los demás datos de la misma clase son equivalentes.

Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Típicamente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica.

Como usuario del sistema se han realizado las siguientes pruebas:

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Registrarse en el sistema.	El sistema muestra en pantalla el formulario y lleva a cabo la acción.	CORRECTO
Registrarse sin introducir todos los datos	El sistema avisa de que faltan datos	CORRECTO
Identificarse en el sistema con nombre de usuario y contraseña correctos.	El sistema permite la realización de acciones restringidas a usuarios registrados.	CORRECTO
Identificarse en el sistema con nombre de usuario y contraseña no correctos.	El sistema no permite esta acción	CORRECTO
Darse de baja en sistema.	El sistema no permite el acceso con ese nombre de usuario y esa contraseña.	CORRECTO
Modificar datos personales del usuario.	El acceso al sistema se realizara con estos nuevos datos.	CORRECTO
Modificar los datos personales sin introducir todos los datos	El sistema avisa de que faltan datos	CORRECTO

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Registrar datos de una dirección de envío.	El sistema almacenara estos datos.	CORRECTO
Registrar una dirección de envío sin introducir todos los datos	El sistema no permite esta acción	CORRECTO
Comprar productos físicos sin estar registrado.	El sistema no permite esta acción	CORRECTO
Comprar productos físicos y electrónicos.	Se permite validar la operación y si es correcta se muestra la forma de pago.	CORRECTO

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Registrar datos de una dirección de facturación.	El sistema almacenara estos datos.	CORRECTO
Registrar una dirección de facturación sin introducir todos los datos	El sistema no permite esta acción	CORRECTO
Emitir una factura si el usuario no esta registrado	El sistema no permite esta acción	CORRECTO
Emitir factura a usuarios registrados	El sistema envía un e-mail con los datos de la factura	CORRECTO

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Registrar datos referentes a una tarjeta valida.	El sistema almacenara estos datos.	CORRECTO
Introducir datos de una tarjeta no valida.	El sistema no permite esta acción	CORRECTO
Realizar un pago a través de una tarjeta de crédito.	El sistema muestra los datos para su confirmación y tramita el pago.	CORRECTO

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Realizar el pago a través del método rápido.	El sistema mostrara los datos del perfil.	CORRECTO
Establecer unos nuevos datos como perfil para el pago rápido.	El sistema almacenara esos datos dentro del perfil.	CORRECTO
Realizar el pago a través del método rápido si no existe un perfil registrado	El sistema no permite esta acción	CORRECTO

6.3.2 Análisis de valores límites

Esta es una técnica complementaria a la de partición equivalente. Los errores tienden a darse más en los límites del campo de entrada que en el centro. Por ello, se ha desarrollado el análisis de valores límites (AVL) como técnica de prueba. El análisis de valores límite lleva a una elección de casos de prueba que ejerciten los valores límite.

Existen una serie de valores denominados “frontera” que es muy conveniente probar, ya que un gran numero de errores aparecen en torno a los puntos de cambio de clase de equivalencia.. Usualmente se necesitan 2 valores por frontera, uno justo abajo y otro justo encima

En nuestro caso, la mayoría de las acciones realizadas por el sistema tienen presente la introducción de datos por parte del usuario, por ello se ha realizado un minucioso estudio de esta técnica.

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
La contraseña tiene cuatro o menos caracteres	El sistema informa de que la contraseña no es correcta.	CORRECTO
Introducir un numero de tarjeta con menos de 16 dígitos	El sistema informa de que la tarjeta no es correcta.	CORRECTO
Procesar pedido con cero productos	El sistema informa de que no hay productos en la cesta y no deja procesar el pedido	CORRECTO

6.4 Errores dentro de un ambiente Web.

En el caso concreto de esta aplicación, la implementación puede darse en varias configuraciones diferentes y dentro de distintos ambientes, por lo tanto es muy difícil predecir la situación en la que un error va a ocurrir.

Dado que las aplicaciones Web residen dentro de una arquitectura cliente/servidor, el rastreo de los errores puede ser difícil a través de las tres capas arquitectónicas: el cliente, el servidor o la red en sí.

6.5 Pruebas de implantación

El proceso de someter a prueba un aplicación Web es un conjunto de actividades que tienen un objetivo común: descubrir errores en el contenido, la función, la facilidad de uso, la navegabilidad, el desempeño, la capacidad y la seguridad de la aplicación.

Las pruebas examinan una o mas de las siguientes dimensiones de calidad:

- El contenido se evalúa tanto en el ámbito semántico como en el sintáctico.
- La estructura se valora para asegurarse de que se entrega adecuadamente contenido y función de la aplicación.
- La facilidad de uso se prueba para garantizar que a cada categoría de usuario la soporta la interfaz.
- La seguridad se prueba al valorar las vulnerabilidades potenciales e intentar explotar cada una de ellas.

6.6 Pruebas de aplicaciones Web

6.6.1 Prueba del contenido

El contenido de una aplicación Web puede albergar errores que pueden ir desde fallos ortográficos o tipográficos hasta la violación de las leyes de propiedad intelectual.

En el caso de nuestro proyecto, dirigido principalmente al registro de usuario, hemos tenido especial cuidado a la hora de mostrar información procedente de la base de datos para que fuera totalmente legible.

6.6.2 Prueba de función

Las funciones que debe realizar cualquier aplicación vienen enumeradas en los requisitos funcionales que se describen durante el análisis y diseño del proyecto. Por lo tanto, para la realización de este tipo de pruebas, se comprueba uno por uno que todos los requisitos funcionales han sido cubiertos y llevados a la práctica sin problemas.

6.6.3 Prueba de la facilidad de uso

La facilidad de uso es la capacidad de que cualquier usuario pueda realizar las funciones que desarrolla nuestra aplicación sin ningún tipo de dificultad. Es decir, que la navegabilidad a través de nuestras paginas sea sencilla y que el usuario pueda acceder a la sección que desee en cualquier momento.

Una aplicación Web diseñada para realizar transacciones de comercio electrónico está dirigida a cualquier usuario de la red que pueda tener o no conocimientos de informática.

En nuestro caso hemos permitido a varios usuarios que utilizaran la aplicación, y han podido realizar con éxito las tareas de registro, modificación de datos personales, baja en el sistema, y el completo desarrollo de una compra de productos tanto físicos como electrónicos y su posterior pago con tarjeta de crédito.

6.6.4 Prueba de navegabilidad

Como hemos descrito en el apartado anterior es muy importante que el usuario pueda navegar sin problemas de una página a otra del sistema y que pueda acceder en cada momento a la sección que desee.

Este tipo de pruebas es muy complicado ya que cada usuario tomará una ruta diferente dependiendo de sus necesidades y de lo que en ese momento se le está mostrando. El objetivo de este tipo de pruebas es permitir que el usuario realice la acción deseada cuando ésta se le ofrece y más concretamente en nuestro caso, que pueda llevar a cabo una compra satisfactoria.

6.6.5 Prueba de compatibilidad

La compatibilidad de las aplicaciones con los diferentes navegadores, sistemas operativos, tipos de máquinas y otros factores constituye siempre un reto fundamental e importante.

En nuestro caso hemos lanzado la aplicación en dos navegadores diferentes: Internet Explorer y Mozilla. Los resultados en ambos casos han sido totalmente satisfactorios.

6.6.6 Pruebas de seguridad

Estas pruebas están diseñadas para comprobar lo vulnerable que es la aplicación en el ambiente del lado del cliente, las comunicaciones de red que ocurren mientras los datos pasan del cliente al servidor y de vuelta, y en el ambiente del lado del servidor.

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Entrar en una zona privada modificando la url del explorador	El sistema no permite la entrada	CORRECTO
Intentar acceder a una zona privada	El sistema no permite la entrada a usuarios externos	CORRECTO
Introducir caracteres de escape en los campos de texto	La acción se realiza con normalidad.	CORRECTO
Intentar modificar el importe total de la compra	El sistema no permite esta acción	CORRECTO

7 Manuales

7.1 Manual de instalación

7.1.1 Instalación de Internet Information Server (IIS)

Internet Information Server (IIS) es un servidor de páginas Web de la plataforma Microsoft. Esta aplicación Web ha sido realizada en ASP.NET, y para su correcto funcionamiento ha de tener instalado IIS.

Para poder realizar la instalación se necesita el CD de instalación del sistema operativo, en nuestro caso, Windows XP. Internet Information Server se encuentra en este CD, para poder instalarlo se debe seleccionar la opción “Instalar componentes adicionales de Windows”

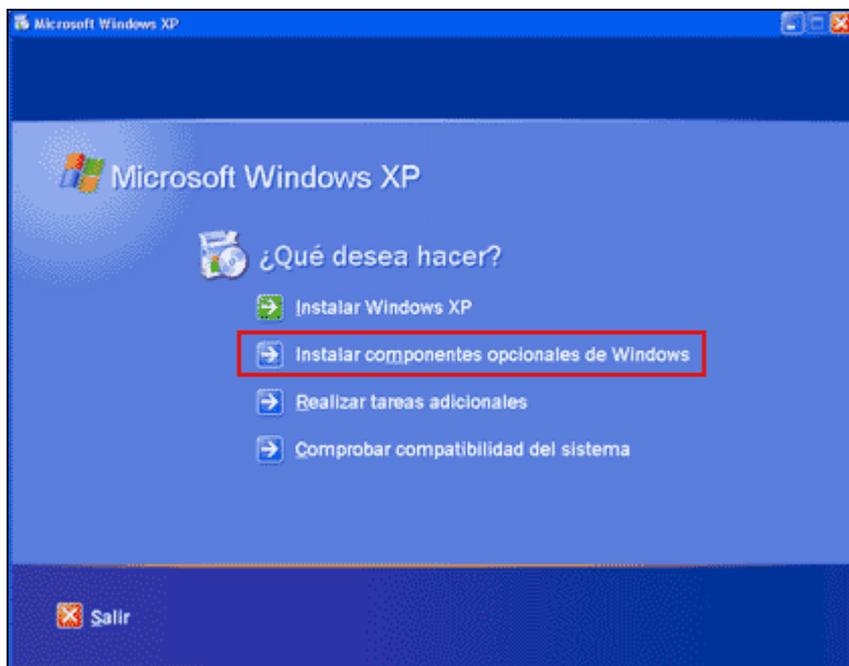


Figura 7-1 Instalación de Internet Information Server (IIS)

Como se puede ver en la imagen a continuación, para poder instalarlo hay que ir a *Inicio > Configuración > Agregar o quitar programas* y seleccionar la pestaña *Agregar o Quitar componentes de Windows*. Aquí aparece una ventana en la cual se pueden seleccionar los componentes de Windows que hay disponibles. Se elige la opción IIS y se pulsa el botón *Siguiente*.

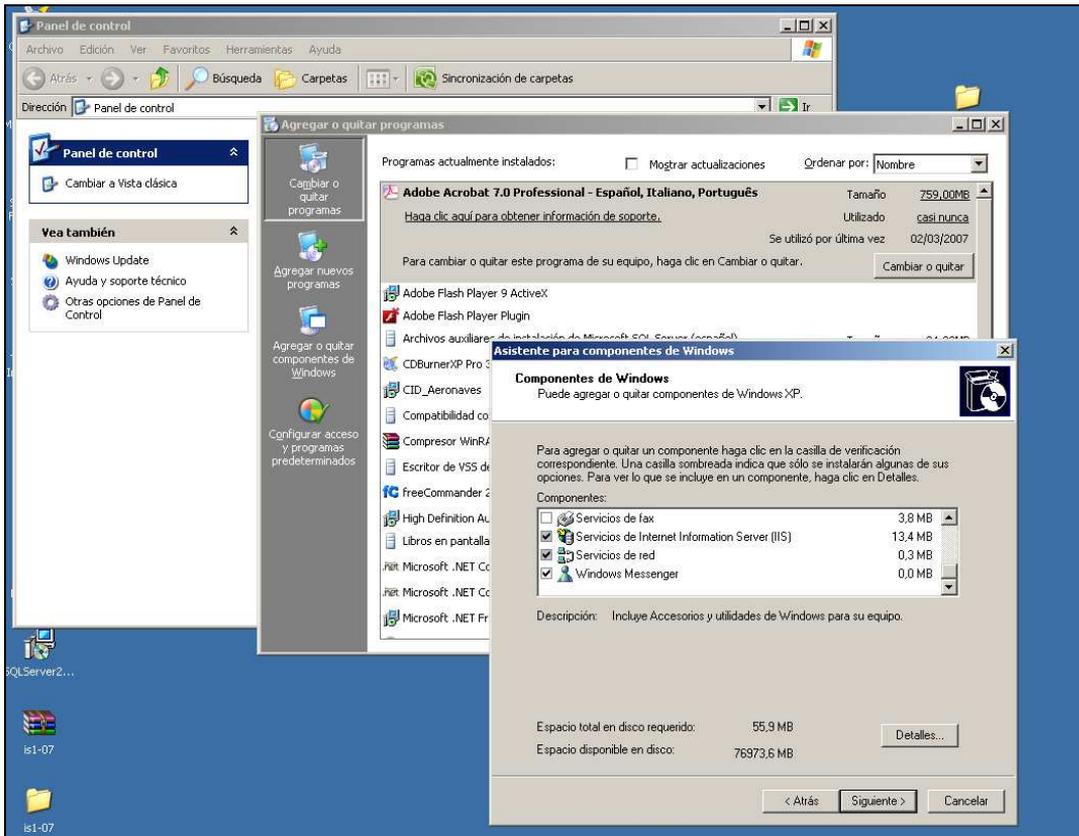


Figura 7-2 Selección de componente IIS

Llegados a este punto, se tiene la plataforma .NET instalada en nuestro sistema. Crearemos ahora una carpeta en “C:\Inetpub\wwwroot” a la que llamaremos *trazabilidad*, y en ella alojaremos todo el contenido. Éste será el directorio particular.

Ahora se debe crear un directorio virtual, es decir, un directorio del servidor que no está dentro del directorio de publicación habitual pero al que se puede acceder a través del servidor Web como si estuviera dentro del directorio “C:\Inetpub\wwwroot”. A este directorio virtual se accede de la forma “http://localhost/directorio_virtual” pero este directorio puede encontrarse en cualquier localización del disco duro e incluso en otro directorio situado en un ordenador remoto de la red.

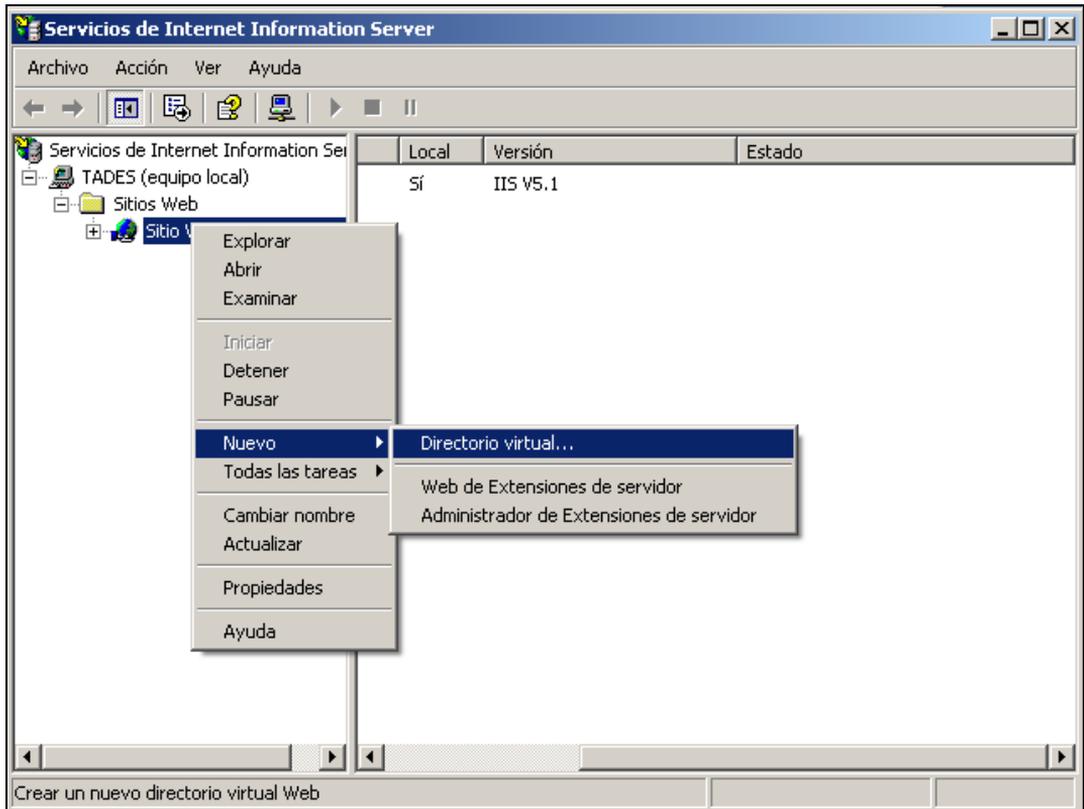


Figura 7-3 Creación del directorio virtual

Para crear dicho directorio virtual hay que acceder al Panel de Control del IIS que está situado en *Inicio > Panel de Control > Herramientas Administrativas > Administrador de Servicios de Internet*. Se solicitará un alias para el directorio virtual, escribimos “*trazabilidad*”. Además se pedirá la ubicación de la carpeta donde se encuentra el proyecto, en nuestro caso “*C:\Inetpub\wwwroot\trazabilidad*”. También se solicita la incorporación de permisos al directorio virtual. Una vez hecho todo esto, se pulsa en *Siguiente* para continuar.

En este punto, se ha llegado al final de la instalación, para comprobar si se ha instalado correctamente, se puede abrir el Internet Explorer y escribir *http://localhost*, y nos mostrará una página web informando de que IIS está instalado. Además si se instaló la documentación de IIS aparecerá en una ventana emergente.

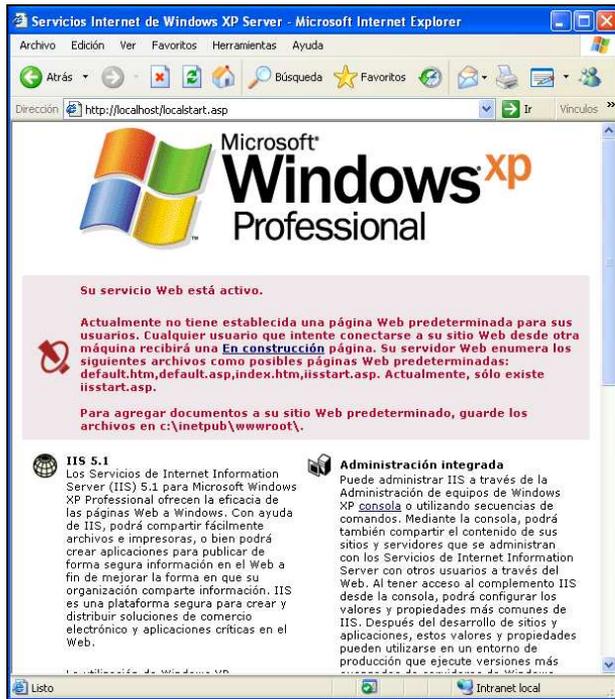


Figura 7-4 Ventana emergente para la comprobación de la instalación

7.1.1.1 Activación de Certificado de Seguridad SSL

SSL es un protocolo que permite la comunicación de nuestro navegador con el servidor a través de un canal seguro. Este canal seguro se consigue encriptando las comunicaciones, y su función es impedir que terceras personas intercepten los datos privados que estamos enviando.

En general, distinguiremos si estamos accediendo a una página web sobre SSL cuando utilicemos el protocolo https en lugar de http. Además, nuestro navegador generalmente recalcará que la página web es segura mediante un icono de un candado, en el que podemos hacer click para ver en detalle la información sobre el certificado de seguridad del sitio web al que accedemos.

Para activar el soporte https para páginas web en el IIS, hay que seguir los siguientes pasos:

1. En la consola de administración del IIS, hay que seleccionar las propiedades del "Sitio Web predeterminado" (Default Web Site) y después, en la pestaña de Seguridad, hacer click en "Certificado de servidor..." (Server Certificate...)
2. Se abre entonces un asistente, en el que se debe seleccionar "Crear un certificado nuevo" (Create a new certificate) y en la siguiente pantalla "Preparar la petición ahora pero enviarla más tarde" (Prepare the request now, but send it later)

3. A continuación se escoge como longitud de clave 2048 bits y se rellenan todos los datos necesarios, obteniendo, al final, una petición de certificado "certreq.txt".
4. Para obtener un certificado para nuestro servidor, tendríamos que contactar con una empresa que los emita o bien, si sólo lo queremos para probar, podemos emitir nuestro propio certificado. En nuestro caso lo hemos obtenido del siguiente enlace:
<http://www.instantssl.com/ssl-certificate-products/free-ssl-certificate.html?gclid=CK-7oIKy6ZECFRMbagodFxsqfg>

A través de esta página se pueden descargar certificados de seguridad gratuitos durante un periodo de prueba de 90 días.

5. Ahora que ya se dispone del certificado digital, se puede volver a abrir la ventana de certificados del IIS seleccionando esta vez la opción de "Procesar la petición pendiente e instalar el certificado".
6. En la siguiente pantalla se debe indicar la ruta del certificado que se acaba de crear (certificado.cer) y se da por finalizado el proceso

En este momento, con el certificado de seguridad funcionando, podemos acceder a nuestras páginas tanto por http como por https. Si se quiere permitir el acceso a una determinada aplicación o página web únicamente a través de https, se siguen los siguientes pasos:

- Ir a las propiedades de la aplicación o página web en la consola de administración del IIS.
- En la pestaña de seguridad pulsar el botón editar en el apartado "Comunicaciones seguras" y marcar la casilla "Requerir canal seguro (SSL)" (Require secure channel (SSL)).

Al acceder a cualquiera de nuestras páginas a través de https, se puede observar que el navegador lanza una alerta que dice que hay problemas con el certificado, ya que este no puede ser comprobado a través de ninguna entidad certificadora. Como esto es un certificado digital únicamente para hacer pruebas no hay nada más que hacer. Si no queremos comprar un certificado digital real, la única opción que nos queda es la de ignorar el mensaje de alerta y continuar.

7.1.2 Instalación de SQL Server

Para la explotación de este proyecto es necesario la instalación del sistema gestor de bases de datos Sql Server. Existen diferentes versiones, en nuestro caso hemos utilizado "SQL Server 2005 Express Edition".

Es posible descargar esta versión gratuitamente desde su página oficial de Microsoft:

<http://www.microsoft.com/spanish/msdn/vstudio/express/SQL/default.aspx>

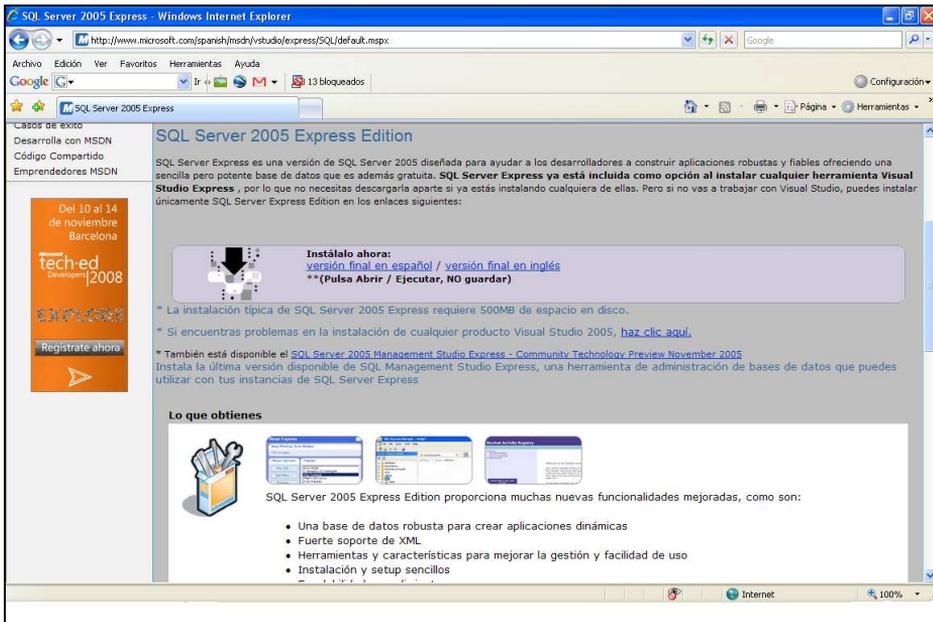


Figura 7-6 Página oficial de Microsoft, descarga de SQL Server

7.1.3 Instalación del Framework .NET 2.0

Microsoft .NET Framework versión 2.0. instala el entorno en tiempo de ejecución y los archivos asociados de .NET Framework necesarios para ejecutar aplicaciones desarrolladas para .NET Framework v2.0.

Para instalarlo, se puede descargar desde su página principal:

<http://www.microsoft.com/downloads/details.aspx?displaylang=es&FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5>

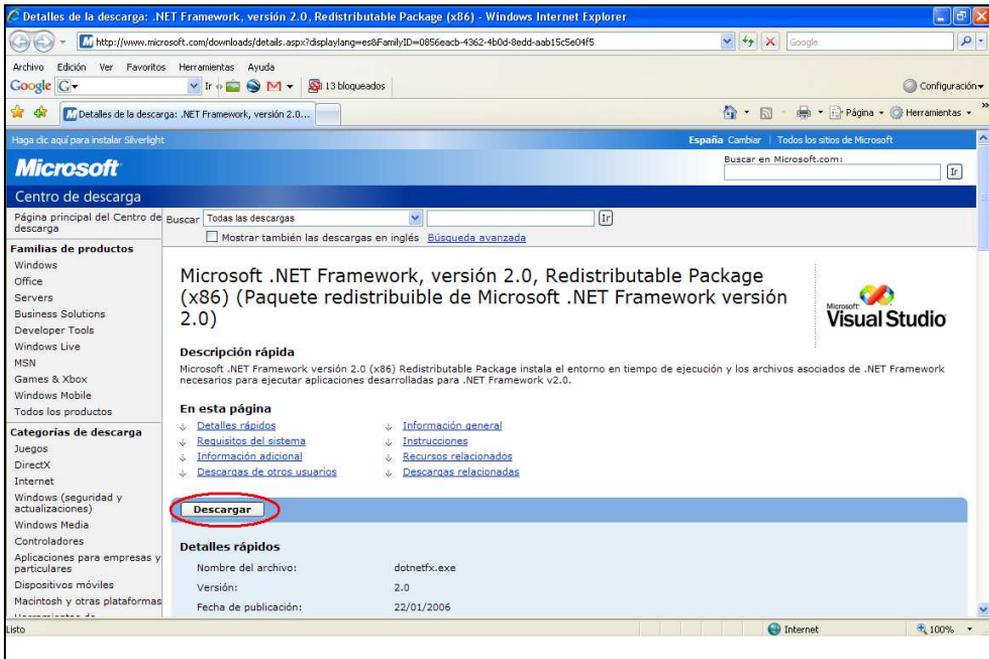


Figura 7-7 Página oficial de Microsoft, descarga de .NET Framework

Después de haber descargado el instalador, se ejecutará y comienza la instalación. Una vez instalado .NET Framework, se necesita instalar la versión de idioma del paquete de idioma de .NET Framework para terminar de configurar el entorno completamente.

7.1.4 Instalación de la Aplicación

A lo largo de la memoria hemos explicado que el producto final obtenido no es único sino que dependiendo de las especificaciones del cliente se pueden obtener diferentes productos software.

En el CD adjunto se han incluido dos versiones diferentes: una de ellas con tan solo los paquetes necesarios para llevar a cabo el registro y el pago con tarjeta y otra que incluye todas las funcionalidades. Seguidamente vamos a explicar la instalación de la aplicación.

En primer lugar debemos introducir el CD y dirigirnos al directorio ProductosFinales. En el encontraremos dos ejecutables: version1.exe y version2.exe. Esta ultima es la que incluye todos los paquetes. Pulsamos la opción que más nos interese y aparecerá esta pantalla:

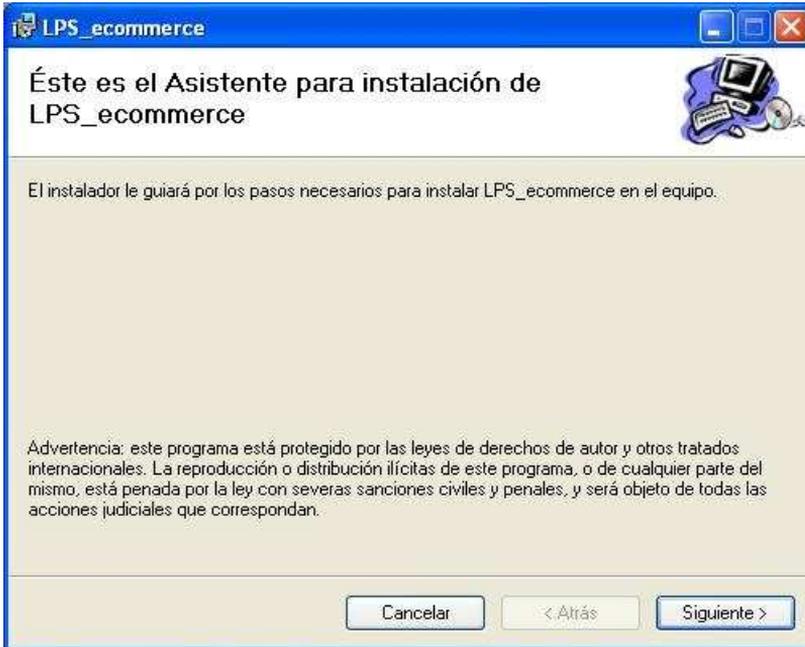


Figura 7-8 Asistente para la instalación de la LPS_ecommerce

Pulsamos Siguiete y a continuación nos aparecerá esta pantalla en la que debemos seleccionar el directorio virtual donde queremos instalar la aplicación. Por defecto este directorio ser WebSetup1.

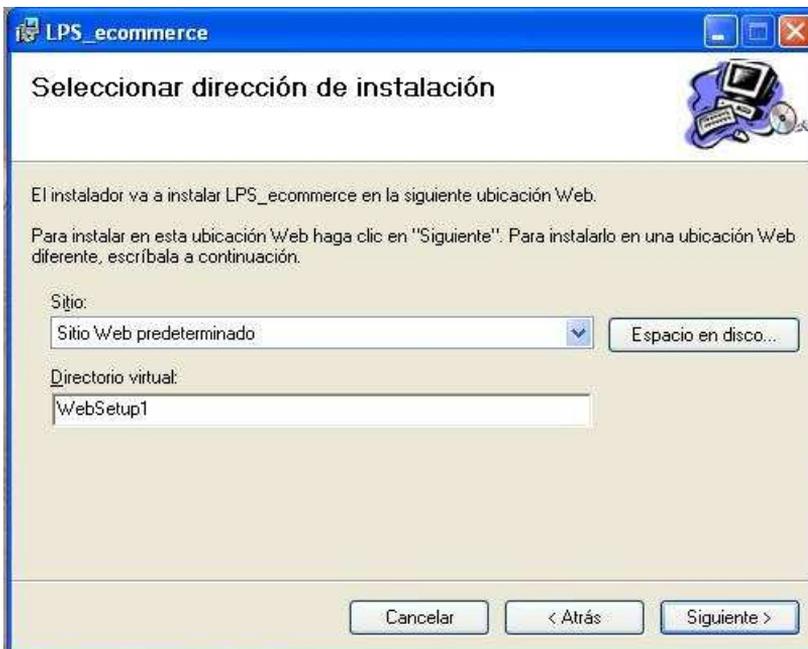


Figura 7-9 Selección de la ubicación Web para la LPS_ecommerce

Pulsamos siguiente para continuar y el sistema nos mostrará la siguiente pantalla indicando que el proyecto de instalación ha sido llevado a cabo con éxito.

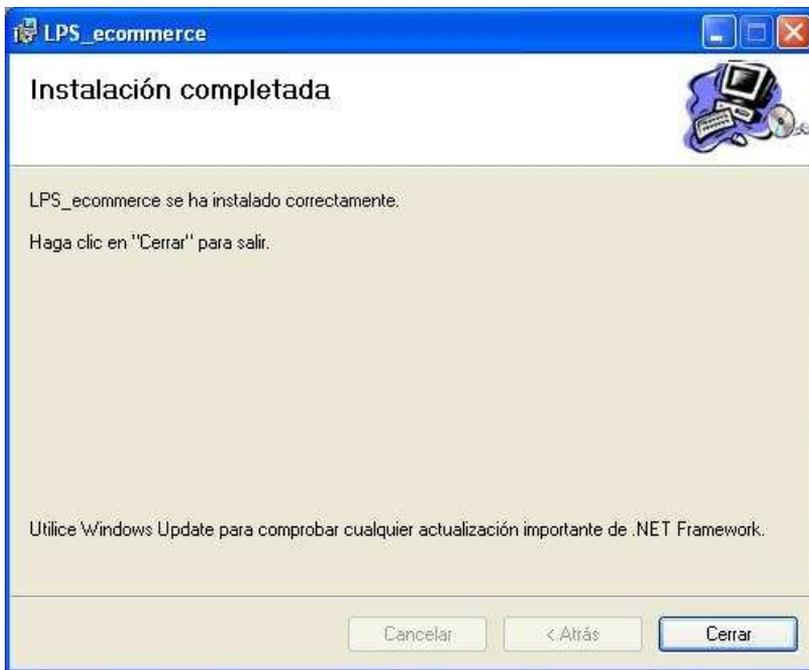


Figura 7-10 Instalación de la LPS_ecommerce completada

Una vez instalada la aplicación del producto final debemos añadir una base de datos que la soporte. Para ello tenemos dos opciones: crearla usando el SQL Server o usar una aplicación de configuración dirigida al usuario donde podrá seleccionar el nombre de una base de datos existente o crear una nueva. También podrá configurar los parámetros del sistema, como puede ser el nombre, la moneda que se va a utilizar, etc. Esta opción es la mas recomendable por su sencillez. Para llevarla a cabo ejecute el fichero Configuracion_usuario.exe que se encuentra en el directorio ProductosFinales del CD adjunto. Aparecerá la siguiente pantalla:



Figura 7-11 Configuración del Proyecto para el usuario

Como podemos ver, debemos especificar la ruta física donde acabamos de instalar la aplicación que por defecto es C:/inetpub/wwwroot/WebSetup1. Una vez configurado todo correctamente pulsamos Aceptar y aparecerá un mensaje informativo indicando que el sistema se ha configurado correctamente. Para probar si la instalación se ha llevado a cabo correctamente, tecleamos en la barra de direcciones del navegador: <http://localhost/WebSetup1/TiendaBasica> y se mostrara la pagina principal.

7.2 Manual de usuario

Al tratarse de una aplicación dirigida a la venta de productos a través de la red, se ha tenido en cuenta en todo momento la sencillez de la interfaz para que de este modo cualquier usuario, con o sin conocimientos de informática, pueda realizar una compra. Aun así en este apartado hemos desarrollado un manual de usuario para facilitar más si cabe el manejo del sistema.

Para explicarlo con la mayor sencillez posible, comenzaremos suponiendo que están incluidos los paquetes desarrollados por nuestros compañeros junto con el paquete base. A este producto le iremos añadiendo uno por uno los desarrollados en el proyecto actual:

7.2.1 Paquete Registration

La pantalla inicial que aparecerá será la siguiente:



Figura 7-12 Pantalla inicial, incluyendo el paquete Registration

En ella encontramos una opción en la parte izquierda, para proceder al registro. Si seleccionamos esta opción el navegador nos redirigirá a la siguiente pantalla:

The screenshot shows a web registration form with a blue header and a white background. On the left side, there is a vertical navigation menu with the following items: 'Elige un Catalogo' (with sub-items: Sonido, Juegos, Programacion, Impresión, Almacenamiento), 'Su carrito esta vacío' (Total: €0.00), 'Acceda a sus Descargas' (with a 'Descargas' link), and 'Si aun no esta registrado, hagalo aqui' (with a 'Registrarse' button). Below the menu is a red box for 'Información de Usuario' containing the text 'No estas identificado', a link '» Iniciar sesión «', and navigation arrows '» «' and '« «'. The main form area has a search bar at the top with 'Buscar en' and a dropdown menu set to 'Toda la Tienda'. The form is titled 'Datos Personales' and contains the following fields: 'Nombre' and 'N.I.F.' (both with asterisks), 'Primer Apellido' and 'Segundo Apellido' (both with asterisks), 'E-mail' and 'Nombre de usuario' (both with asterisks), and 'Password' and 'Confirme Password' (both with asterisks). A note below the password fields says '(entre 5 y 10 caracteres)'. At the bottom of the form is a button labeled 'Enviar Datos Personales'.

Figura 7-13 Datos personales

En esta página procedemos al registro de nuestros datos personales. Una vez rellenados los campos pulsamos el botón que dicta *Enviar Datos Personales* y seguidamente al botón *Continuar*. Esto nos llevará a la siguiente pantalla donde se nos indica que el registro ha sido llevado a cabo correctamente y nos invita a identificarnos en el sistema:

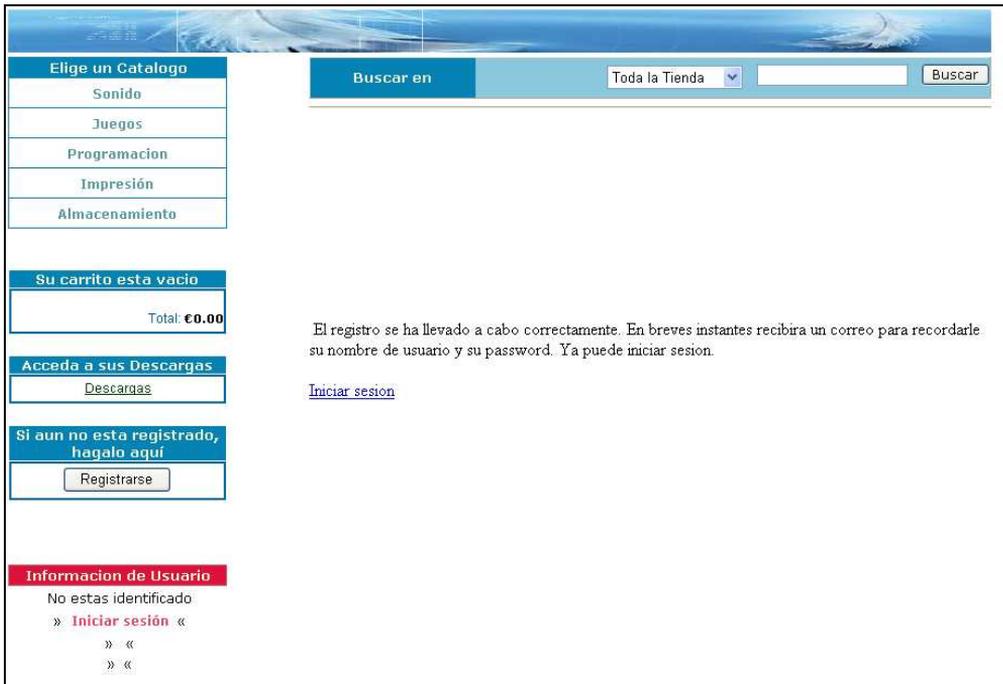


Figura 7-14 Registro completado

En este caso tenemos dos opciones, pulsar el link para identificarnos o continuar con la compra e identificarnos más adelante utilizando el control que encontramos a nuestra izquierda para iniciar sesión. En cualquiera de los dos casos aparecerá la siguiente pantalla:

Elige un Catalogo

- Sonido
- Juegos
- Programacion
- Impresión
- Almacenamiento

Su carrito esta vacio

Total: €0.00

Acceda a sus Descargas

Descargas

Si aun no esta registrado, hagalo aqui

Registrarse

Informacion de Usuario

No estas identificado

» [Iniciar sesión](#) «

» «

Buscar en

Toda la Tienda

Buscar

¿Quien eres?

Nombre de usuario: beita

Contraseña: ●●●●●●

Recordármelo la próxima vez.

Inicio de sesión

Figura 7-15 Inicio de sesión

Una vez identificados, si deseamos modificar los datos personales o darnos de baja del sistema podemos hacerlo pulsando los links que encontramos a nuestra izquierda:

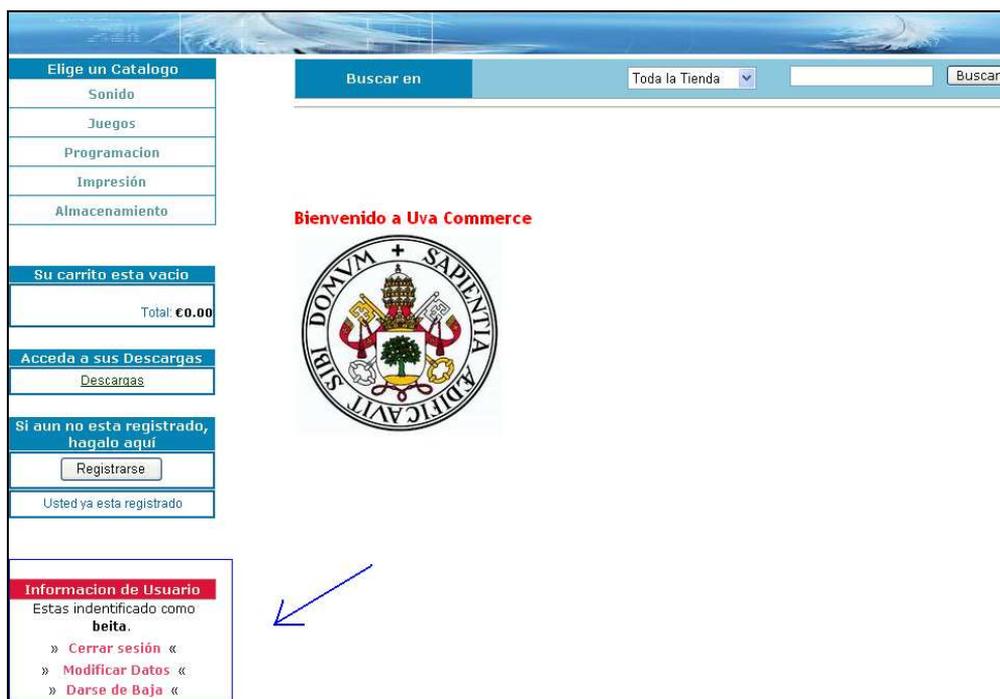


Figura 7-16 Opciones de usuario

Si elegimos la opción Modificar Datos será mostrada la siguiente pantalla:

The screenshot shows a web application interface for modifying user data. On the left, there is a navigation menu with categories: 'Elige un Catalogo' (Sonido, Juegos, Programación, Impresión, Almacenamiento), 'Su carrito esta vacío' (Total: €0.00), 'Acceda a sus Descargas' (Descargas), and 'Si aun no esta registrado, hagalo aquí' (Registrarse, Usted ya esta registrado). The main content area is titled 'Información de Usuario' and shows the user is logged in as 'beita'. Below this, there are three links: '» Cerrar sesión «', '» Modificar Datos «', and '» Darse de Baja «'. The 'Modificar Datos' link is active, leading to a form with the following fields: 'Nombre' (Beatriz), 'Apellidos' (Rico Perez), 'Password' (masked with dots), and 'Confirme Password' (masked with dots). A 'Modificar' button is located below the password fields. At the top right, there is a search bar with a dropdown menu set to ' Toda la Tienda ' and a 'Buscar' button.

Figura 7-17 Modificar datos personales

En ella podemos hacer las modificaciones necesarias y pulsando el botón se actualizarán los datos en la base de datos.

Si seleccionamos Darse de Baja la pantalla que aparecerá será la siguiente:

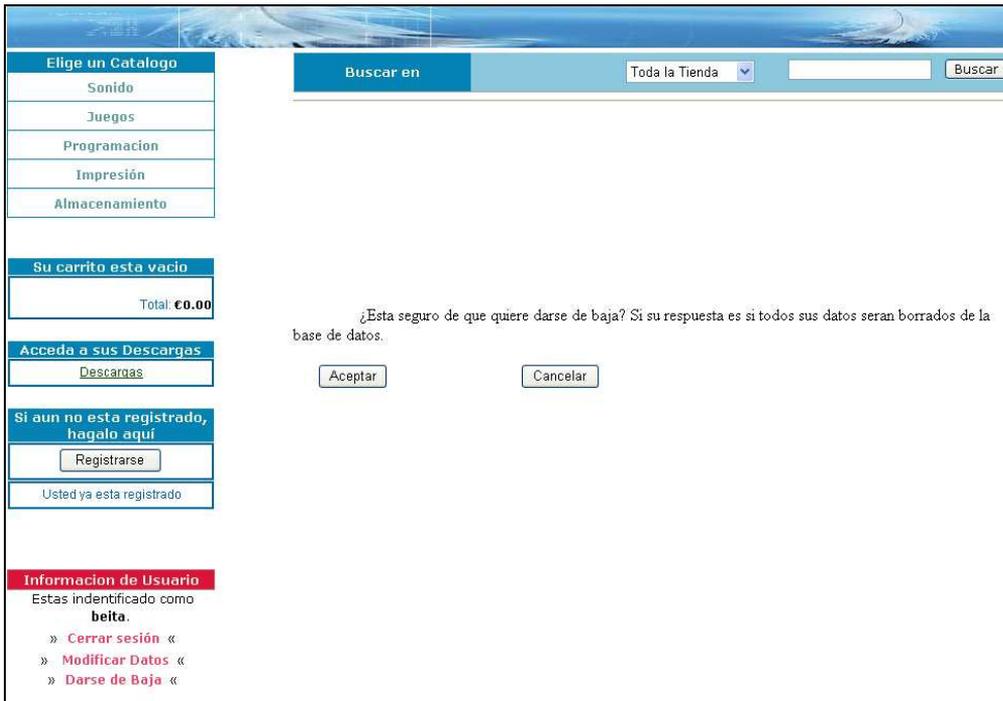


Figura 7-18 Baja de usuario

Como podemos observar, en esta pantalla se nos ofrecen dos opciones: confirmar la opción de darnos de baja o cancelar.

Para poder ver los productos ofertados en la tienda pasamos a seleccionar alguno de los catálogos proporcionados en la parte izquierda de la página. Al pulsar en cualquiera de ellos nos mostrará una serie de productos. Si estamos interesados en alguno en particular, lo seleccionamos haciendo clic en su nombre y aparecerá una nueva pantalla con los detalles de dicho producto:

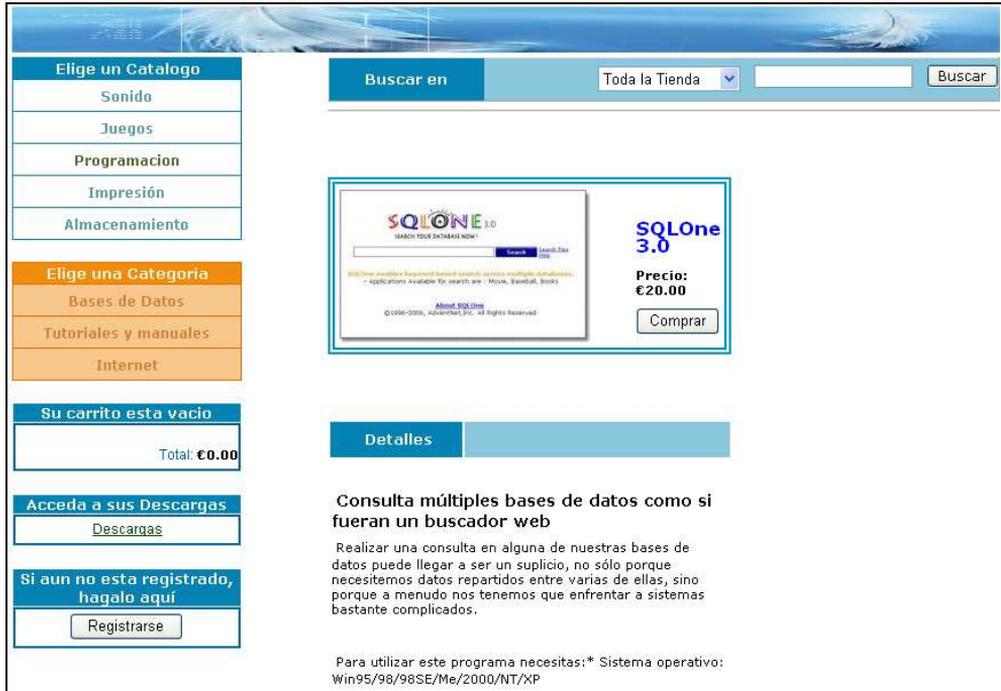


Figura 7-19 Ver producto

Si el producto resulta de nuestro agrado y deseamos comprarlo pulsaremos el botón *Comprar* que nos llevará a la siguiente pantalla en la que podremos decidir si continuar nuestra compra o pagar el producto:

The screenshot shows a shopping cart interface. On the left, there is a navigation menu with categories like 'Sonido', 'Juegos', 'Programacion', 'Impresión', and 'Almacenamiento'. Below this, a summary box shows 'Su carrito de la compra contiene' with a total of €20.00. Further down are links for 'Descargas' and a 'Registrarse' button. At the bottom left, a red box indicates 'Información de Usuario' with a link to 'Iniciar sesión'. On the right, a search bar is at the top. Below it, a message states 'Su Carrito de la Compra contiene:'. A table lists the product 'SQLOne 3.0' with a price of €20.00 and an 'Eliminar' button. Below the table, the total is shown as 'Total: €20.00' with a 'Continuar Comprando' button. The 'Métodos de pago' section is titled 'Pago por Pay-Pal' and includes an 'Aceptar' button.

Producto	Precio	
SQLOne 3.0	€20.00	<input type="button" value="Eliminar"/>
Total: €20.00		<input type="button" value="Continuar Comprando"/>

Métodos de pago

Pago por Pay-Pal

Figura 7-20 Carrito de la compra con una única forma de pago, Paypal

En este caso vemos que tan sólo se puede pagar a través de PayPal, pero más adelante según añadamos paquetes veremos que tendremos la opción de realizar el pago con otros métodos. Si deseamos continuar con el proceso de compra pulsaremos el botón *Aceptar*:

The screenshot shows a web checkout interface. On the left, there is a navigation menu with categories: 'Elige un Catalogo' (with sub-items: Sonido, Juegos, Programacion, Impresión, Almacenamiento), 'Su carrito de la compra contiene' (with a link to 'ver detalles' and item 'SQLOne 3.0, €20.00', total '€20.00'), 'Acceda a sus Descargas' (with a 'Descargas' link), and 'Si aun no esta registrado, hagalo aquí' (with a 'Registrarse' button). At the bottom left, there is a red box for 'Informacion de Usuario' stating 'No estas identificado' and a link to 'Iniciar sesión'. On the right, there is a search bar with 'Buscar en' and a dropdown menu set to ' Toda la Tienda', followed by a search input field and a 'Buscar' button. Below the search bar, there is an 'Email *' input field, a 'Procesar pedido:' label, and a 'Pagar' button. To the right of the 'Pagar' button are two payment options: 'Check out with PayPal' (with the tagline 'The safer, easier way to pay') and 'Express Checkout'. A note below the payment options states: '* este email será utilizado para procesar el pedido y enviarle sus productos'.

Figura 7-21 Pago con PayPal

Al introducir nuestra dirección de correo electrónico y pulsar en el botón o en el icono el sistema nos redirigirá a la pagina web de PayPal (<http://www.paypal.es>) donde siguiendo las instrucciones podrá llevar a cabo el pago.

7.2.2 Paquete Credit Card

Si a la solución desarrollada en el apartado anterior le añadiéramos el paquete de Credit Card, aparte de añadir algunas funcionalidades, el aspecto de varias interfaces cambiaría. Por ejemplo en la pagina de registro de usuarios aparte de la solicitud de los datos personales también se solicitarían los datos de la tarjeta de crédito:

Datos de la Tarjeta de Crédito

Tipo Tarjeta de Pago

Nombre titular **Apellidos titular**

Número tarjeta **Fecha caducidad**

Número CVV

Figura 7-22 Insertar datos de tarjeta

En este caso tenemos la posibilidad de introducir nuestros datos referentes a la tarjeta de crédito en el momento del registro aunque no es obligatorio hacerlo.

Otra de las interfaces modificadas al añadir el paquete Credit Card es la utilizada para mostrar las diferentes formas de pago ya que ahora tenemos la opción de pagar usando tarjeta de crédito:

The screenshot shows a web application interface for a shopping cart. On the left, there is a navigation menu with categories like 'Elige un Catalogo', 'Sonido', 'Juegos', 'Programacion', 'Impresión', and 'Almacenamiento'. Below this, there are sections for 'Su carrito de la compra contiene' (showing 'SQLOne 3.0, €20.00'), 'Acceda a sus Descargas', and 'Si aun no esta registrado, hagalo aquí' (with a 'Registrarse' button). At the bottom left, there is a user information section for 'beita' with links for 'Cerrar sesión', 'Modificar Datos', and 'Darse de Baja'.

On the right side, there is a search bar with 'Buscar en' and 'Buscar' buttons. Below it, the cart summary shows 'Su Carrito de la Compra contiene:' followed by a table:

Producto	Precio	
SQLOne 3.0	€20.00	<input type="button" value="Eliminar"/>
Total: €20.00		<input type="button" value="Continuar Comprando"/>

Below the table, there are sections for 'Métodos de pago': 'Pago por Pay-Pal' with an 'Aceptar' button, and 'Pago con Tarjeta' with an 'Aceptar' button.

Figura 7-23 Carrito de la compra con dos formas de pago, Paypal y Tarjeta de crédito

Por lo tanto si desea realizar su compra a través de la tarjeta de crédito debe pulsar el botón *Aceptar*. A partir de aquí el sistema puede llevarnos por dos caminos diferentes dependiendo de las acciones realizadas hasta el momento. Uno de ellos es que haya realizado el registro en otras visitas a la pagina o simplemente no ha introducido los datos de la tarjeta de crédito a la hora de registrarse. En ese caso le aparecerá la siguiente pantalla en la que deberá introducir los datos pertinentes:

The screenshot shows a web interface for credit card payment. On the left, there is a navigation menu with categories like 'Elige un Catalogo' (Sonido, Juegos, Programacion, Impresión, Almacenamiento), 'Su carrito de la compra contiene' (SQLOne 3.0, €20.00, Total: €20.00), 'Acceda a sus Descargas', and 'Si aun no esta registrado, hagalo aqui' (Registrarse, Usted ya esta registrado). Below this is 'Informacion de Usuario' (Estas indentificado como beita, with links for Cerrar sesión, Modificar Datos, and Darse de Baja). At the top right, there is a search bar with 'Buscar en' and 'Buscar' buttons. The main section is titled 'Datos de la Tarjeta de Crédito' and contains the following fields: 'Tipo Tarjeta de Pago' (VISA), 'Nombre titular' (Beatriz), 'Apellidos titular' (Castaño Gallego), 'Número tarjeta' (4539520249984005), 'Fecha caducidad' (06/2016), and 'Número CVV' (123). At the bottom of the form are buttons for 'Enviar Datos Tarjeta', 'Pagar', and 'Cancelar'.

Figura 7-24 Pago con tarjeta de crédito

En la imagen podemos observar que los campos están rellenos. Esto ocurre si el usuario ha visitado mas veces nuestra pagina y su tarjeta ha quedado almacenada. Podemos actualizar los datos o realizar la compra con la tarjeta mostrada. Si cancelamos, nos permitirá optar de nuevo por la forma de pago con la que queremos realizar la compra (Figura 7-23). Si aceptamos, nos mostrara los datos de los productos que deseamos comprar así como de los datos introducidos:

Elige un Catalogo

- Sonido
- Juegos
- Programacion
- Impresión
- Almacenamiento

Buscar en

Confirmacion del pago

Producto	Precio
SQLOne 3.0	€20.00
TOTAL	€20.00

Datos Tarjeta

Tipo Tarjeta de Credito VISA

Nombre Titular Beatriz **Apellidos Titular** Castaño Gallego

Número tarjeta 4918479643806002 **Fecha Caducidad** 4/2010

Número CVV 111

Su carrito de la compra contiene

[\(ver detalles\)](#)
SQLOne 3.0, €20.00
Total: **€20.00**

Acceda a sus Descargas

[Descargas](#)

Si aun no esta registrado, hagalo aqui

Usted ya esta registrado.

Informacion de Usuario

Estas indentificado como **beita.**

- » [Cerrar sesión](#) «
- » [Modificar Datos](#) «
- » [Darse de Baja](#) «

Figura 7-25 Confirmación del pago con tarjeta de crédito

Si cancela el pago, al igual que en la pantalla anterior, el sistema le redirigirá de nuevo a la pagina para seleccionar el método de pago con el que quiere realizar la compra (Figura 7-23). Si acepta, le mostrara una nueva interfaz en el que se le informa de que el pedido se ha realizado con éxito y se le asignara un número de pedido.

Elige un Catalogo

- Sonido
- Juegos
- Programacion
- Impresión
- Almacenamiento

Su carrito esta vacío

Total: €0.00

Acceda a sus Descargas

[Descargas](#)

Si aun no esta registrado, hagalo aqui

Usted ya esta registrado

Información de Usuario

Estas indentificado como **beita**.

- » [Cerrar sesión](#) «
- » [Modificar Datos](#) «
- » [Darse de Baja](#) «

Buscar en

PEDIDO CONFIRMADO

Su Pedido ha sido procesado correctamente, el N° de pedido es: 13082008044425

Este numero sera requerido para procesar cualquier reclamacion, asi que le recomendamos que lo guarde en lugar seguro

Le informaremos mediante correo electronico (beita@hotmail.com) del estado de sus productos.

Su pedido consta de

Producto	Precio
SQLOne 3.0	€20.00
Total: €20.00	

Gracias por depositar su confianza en nosotros, atentamente Uva Commerce

Ahora puede acceder a su descarga directa

Codigo de Descarga *

* este código sólo será válido para una descarga.

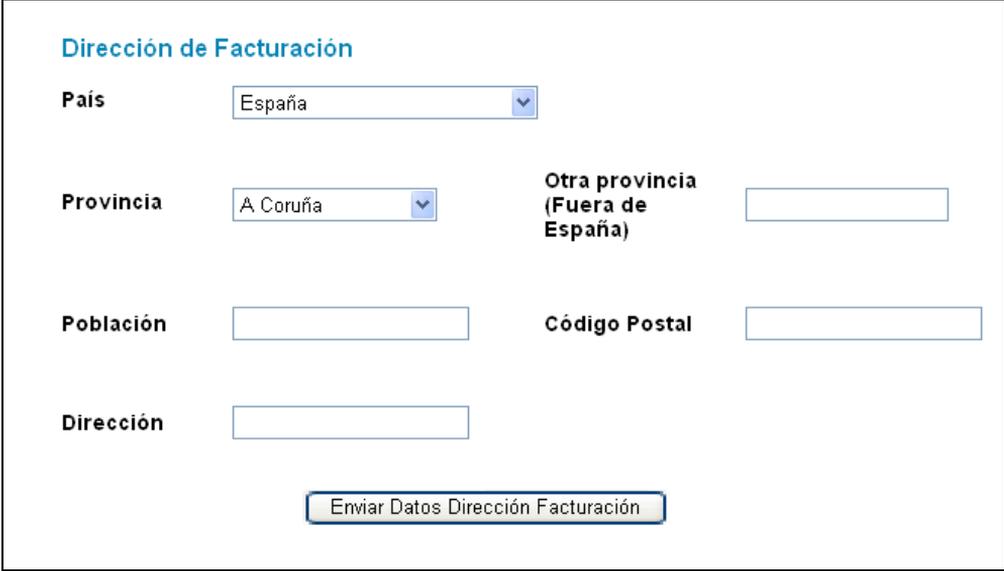
Figura 7-26 Confirmación del pedido realizado

Como podemos ver en la imagen, tenemos la posibilidad de descargarnos el producto que hemos adquirido. Para ello tiene que cumplirse que se trate de un producto electrónico.

El otro camino tendrá lugar si esta es la primera vez que visita nuestra pagina y en el momento del registro optó por introducir los datos de la tarjeta. En ese caso le aparecerá una pagina mostrándole sus datos para confirmar su compra directamente (Figura 7-25). El proceso continuara de igual forma que con la otra opción.

7.2.3 Paquete Billing Address

Añadiendo este paquete, lo que se pretende es poder almacenar la dirección fiscal donde el cliente desea que se le envíen las facturas. Por lo tanto, las interfaces mostradas en las que se solicitaban la inserción de datos cambiaran. Al igual que con los datos de la tarjeta, los datos de la dirección de facturación se solicitaran a la hora del registro pero en este caso tampoco es obligatorio rellenarlos. También tendrá que rellenarlos, pero esta vez de forma obligatoria, cuando se seleccione el tipo de pago y aparezca el formulario.



Dirección de Facturación

País

Provincia **Otra provincia (Fuera de España)**

Población **Código Postal**

Dirección

Figura 7-27 Insertar dirección de facturación

7.2.4 Paquete Shipping Address

Con este paquete, damos la posibilidad al cliente de poder vender productos físicos ya que gracias a él se va a poder almacenar la dirección de envío donde el usuario desea que le lleguen los pedidos realizados. De igual forma que con el paquete de Billing Address, los datos de envío se solicitan a la hora del registro y también es opcional rellenarlos. Si y solo si alguno de los productos que ha adquirido el cliente es físico, se piden los datos de la dirección de envío en cualquiera de los métodos de pago.

Dirección de Envío

País

Provincia **Otra provincia (Fuera de España)**

Población **Código Postal**

Dirección

Figura 7-28 Insertar dirección de envío

7.2.5 Paquete Quick Checkout

El paquete Quick Checkout como ya hemos explicado anteriormente, nos proporciona una nueva forma de pago para aquellos usuarios que ya hayan realizado más compras en nuestra página. Por lo tanto si se añade este paquete la pagina que nos muestra las diferentes formas de pago será la siguiente:

The screenshot shows a web interface for a shopping cart. At the top, there is a search bar with the text "Buscar en" and a dropdown menu set to "Toda la Tienda". Below the search bar, the text "Su Carrito de la Compra contiene:" is displayed. A table lists the items in the cart:

Producto	Precio	
SQLOne 3.0	€20.00	<input type="button" value="Eliminar"/>

Below the table, the total is shown as "Total: €20.00" and a button "Continuar Comprando" is present. Underneath, the "Métodos de pago" section is visible, with sub-sections for "Pago por Pay-Pal" (with an "Aceptar" button) and "Pago con Tarjeta" (with an "Aceptar" button). A link "Si desea pagar utilizando el método rápido" points to a "QuickCheckout" button. On the left side, there are navigation menus: "Elige un Catalogo" (with options: Sonido, Juegos, Programacion, Impresión, Almacenamiento), "Su carrito de la compra contiene" (with a "ver detalles" link and a summary of the cart item), "Acceda a sus Descargas" (with a "Descargas" button), "Si aun no esta registrado, hagalo aqui" (with a "Registrarse" button and a "Usted ya esta registrado" link), and "Informacion de Usuario" (with the text "Estas indentificado como beita." and links for "Cerrar sesión", "Modificar Datos", and "Darse de Baja").

Figura 7-29 Carrito de la compra con opción de pago por método rápido

El botón QuickCheckout solamente estará activo si el cliente tiene almacenados los datos de su tarjeta y de la dirección de facturación. Si seleccionamos esta forma de pago el sistema nos llevara directamente a la pantalla de confirmación de datos (Figura 7.25) y el proceso de pago continuara normalmente como se ha explicado en el apartado del paquete CreditCard.

Con este paquete también tendremos la posibilidad de elegir un nuevo perfil Quick Checkout seleccionando como forma de pago el pago con tarjeta.

Información de Usuario

Estas indentificado como **beta**.

» [Cerrar sesión](#) «
 » [Modificar Datos](#) «
 » [Darse de Baja](#) «

Dirección de Facturación

Pais

Provincia **Otra provincia (Fuera de España)**

Población **Código Postal**

Dirección

Perfil QuickCheckout

Pulse este boton si desea establecer estos datos como nuevo perfil Quick Checkout:

Figura 7-30 Modificación de datos del perfil Quick Checkout

Si aceptamos se establecerán esos datos como nuevo perfil, sino permanecerá el que teníamos hasta ahora.

8 Configuración del producto final

8.1 Introducción

A lo largo de toda esta memoria se ha explicado detenidamente como dar soporte a la variabilidad en la interfaz grafica en un entorno web. Pero para que dos productos finales sean distintos no solo se han de diferenciar gráficamente sino también a nivel de configuración. Para esta labor, aparte del proyecto web, se ha implementado un proyecto de ventana. Esta aplicación solamente esta diseñada para los posibles desarrolladores de la línea de producto, no para el administrador de la tienda.

Para la realización de dicha aplicación hemos considerado necesaria la utilización de una base de datos en la que se almacenen los atributos de los paquetes desarrollados hasta el momento. Por ello, creímos conveniente adjuntar a la memoria este apéndice que no es otra cosa que una pequeña memoria de la aplicación de ventana en la que explicaremos todo el proceso de desarrollo.

8.2 Análisis

8.2.1 Documento de requisitos

8.2.1.1 Requisitos funcionales

FRQ-0016	Seleccionar paquetes
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir al Desarrollador seleccionar los paquetes necesarios para la creación del producto final.</i>
Importancia	Vital

FRQ-0017	Configuración de la base de datos
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir al Desarrollador especificar la base de datos que va a utilizar o crear una nueva.</i>
Importancia	vital

DESARROLLO DE LA SOLUCIÓN

FRQ-0018	Configurar la tienda
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>permitir al Desarrollador establecer algunos parámetros de la tienda como pueden ser el nombre o la moneda que se va a utilizar.</i>
Importancia	vital

8.2.1.2 Requisitos no funcionales

NFR-0010	Sistema flexible
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>ser altamente flexible ante nuevos paquetes que deseen ser añadidos a la configuración.</i>
Importancia	vital

NFR-0011	Paquetes incompatibles
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>no permitir insertar en el sistema dos paquetes que no sean compatibles entre sí.</i>
Importancia	vital

NFR-0012	Paquetes requeridos
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>insertar automáticamente los paquetes necesarios para la existencia de otros.</i>
Importancia	vital

NFR-0013	Integración de paquetes de forma automática
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>reconocer la existencia de todos los paquetes que hayan sido añadidos en la base de datos.</i>
Importancia	importante

NFR-0014	Tiempo de respuesta
Versión	1.0 (13/08/2008)
Dependencias	Ninguno
Descripción	El sistema deberá <i>reaccionar a cualquier evento en tiempo inferior a 1 minuto</i>
Importancia	importante

8.2.2 Modelo de casos de uso

8.2.2.1 Actores

ACT-0004	Desarrollador
Versión	1.0 (18/08/2008)
Descripción	Este actor representa <i>al usuario encargado de la configuración del producto final</i>
Comentarios	Ninguno

8.2.2.2 Diagrama de casos de uso

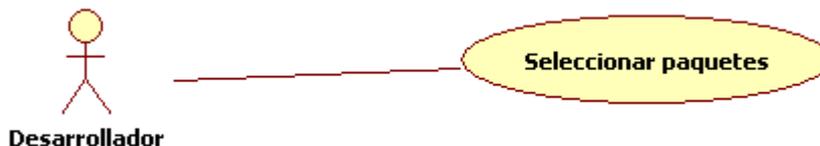


Figura 8-1 Diagrama de casos de uso

DESARROLLO DE LA SOLUCIÓN

8.2.2.3 Especificación casos de uso

UC-0015	Seleccionar paquetes	
Versión	1.0 (13/08/2008)	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando <i>el Desarrollador desee configurar los paquetes que van a ser añadidos al sistema</i>	
Precondición	En la base de datos deben aparecer el nombre, el nombre del padre, y el nombre que tiene dentro de la configuración de los paquetes desarrollados.	
Secuencia normal	Paso	Acción
	1	El sistema <i>mostrará una lista de todos los paquetes especificados en la base de datos.</i>
	2	El actor Desarrollador (ACT-0003) <i>seleccionara los paquetes que desea añadir al producto final.</i>
	3	El sistema <i>modificará los archivos pertinentes</i>
Postcondición	Los archivos de configuración quedaran modificados para que solo se tengan en cuenta los paquetes añadidos.	
Excepciones	Paso	Acción
	2	<i>Si el paquete seleccionado requiere la presencia de otros para correcto funcionamiento, el sistema selecciona automáticamente los paquetes requeridos., a continuación este caso de uso continúa</i>
	2	<i>Si el paquete es incompatible con otro ya seleccionado, el sistema no permitirá que ese paquete sea seleccionado, a continuación este caso de uso continúa</i>

8.2.2.4 Diagrama de secuencia

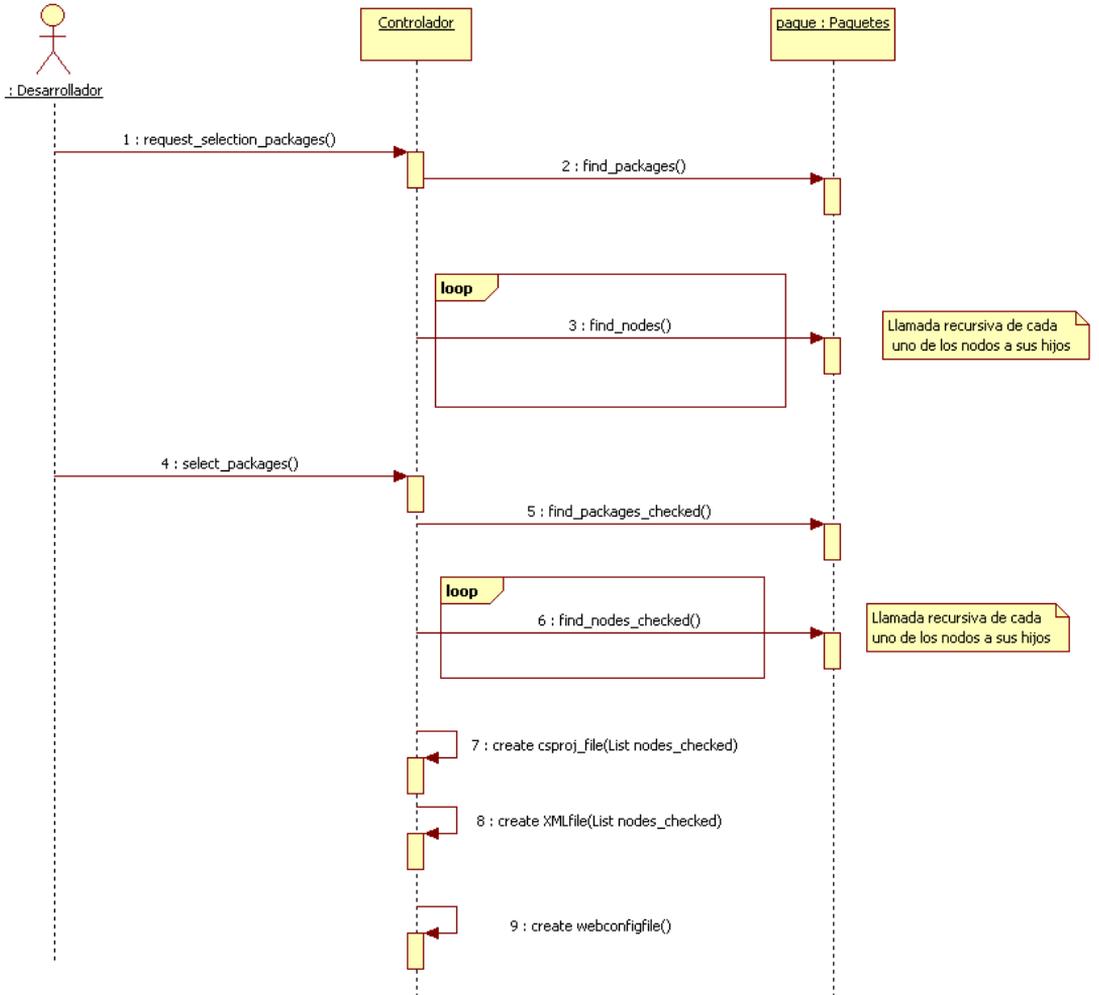


Figura 8-2 Diagrama de secuencia CU-0015 Seleccionar paquetes

8.3 Diseño

8.3.1 Diagrama de clases

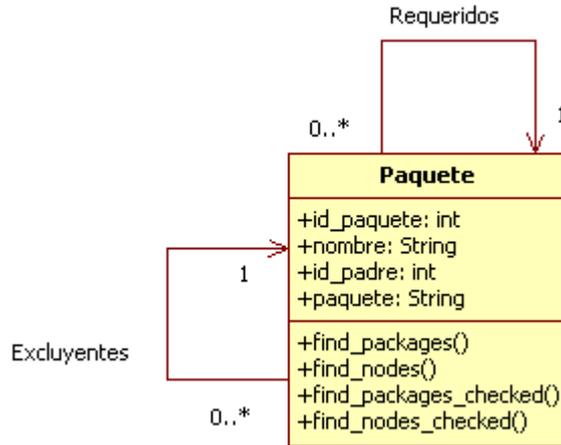


Figura 8-3 Diagrama de clases de la Configuración

Especificación

Class Paquete	
Es una clase que representa los atributos de los paquetes que se van implementando y añadiendo al sistema.	
Atributos	Operaciones
+id_paquete: int Representa el numero que identifica al paquete.	+find_packages(): Busca en la base de datos los paquetes que forman los nodos raíz del árbol.
+nombre: string Representa el nombre que se desea que aparezca en la interfaz de configuración.	+find_nodos() Método recursivo que busca los nodos hijos que cuelgan de cada nodo.
+id_padre: int Representa el número identificativo del padre del nodo en el árbol.	+find_packages_checked(): Método que localiza los nodos raíz seleccionados.
+paquete:string Representa el nombre dado al paquete en la implementación.	+find_nodos_checked(): Método recursivo que localiza los nodos hijos de cada nodo seleccionados.

8.3.2 Diagramas de secuencia

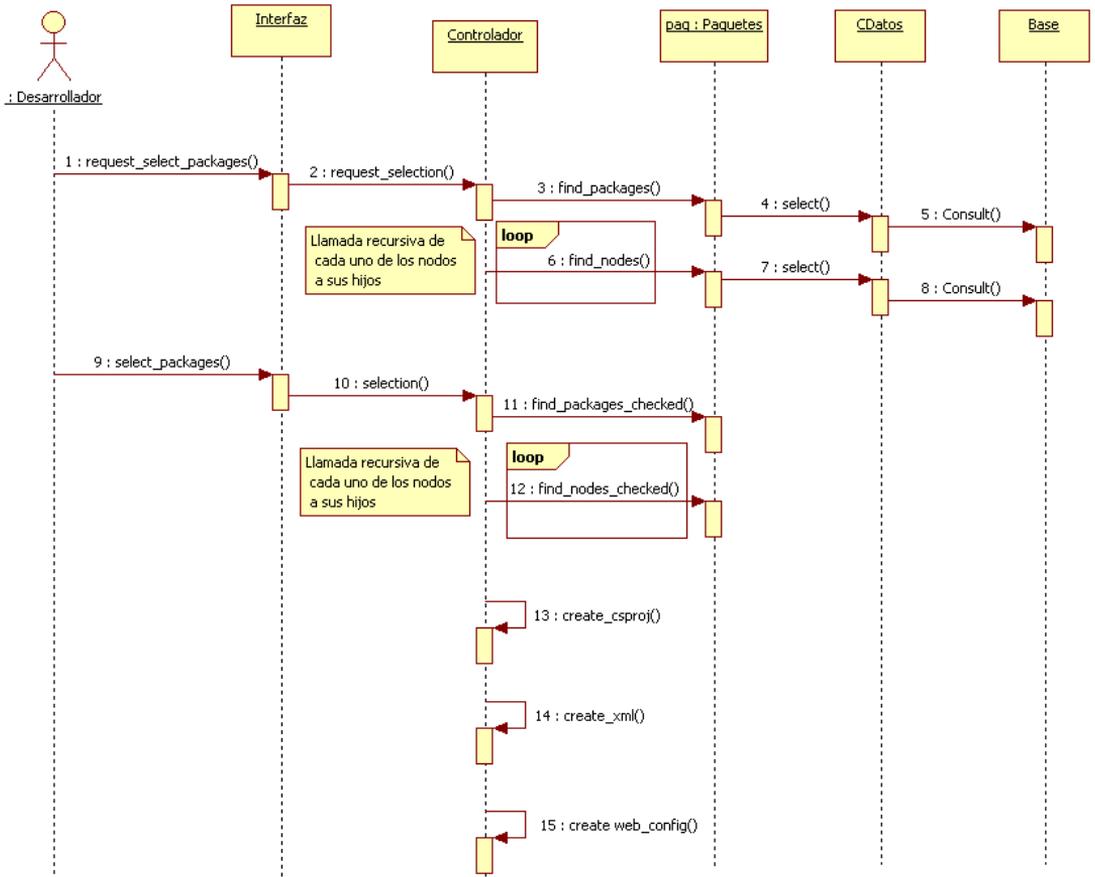


Figura 8-4 Diagrama de secuencia CU-0015 Seleccionar paquetes

8.3.3 Diseño de la base de datos

Modelo relacional

- **Paquete** (id_paquete, padre, nombre, paquete)
- **Excluidos** (id_paquete, id_excluye)
- **Requeridos** (id_paquete, id_requiere)

Descripción de tablas

Paquete: Almacena los datos de los paquetes necesarios para la configuración del sistema.

Nombre Atributo	Tipo	Descripción
id_paquete	int	Identificador para cada paquete, es único.
padre	int	Identificador del padre del paquete.
nombre	varchar(50)	Nombre que aparecerá en la interfaz.
paquete	varchar(50)	Nombre dado al paquete en la implementación.

Excluidos: Relaciona cada paquete con los paquetes que son incompatibles con él.

Nombre Atributo	Tipo	Descripción
id_paquete	int	Identificador del paquete.
Id_excluye	int	Identificador del paquete incompatible con él.

Requeridos: Relaciona cada paquete con los paquetes de los que es dependiente.

Nombre Atributo	Tipo	Descripción
id_paquete	int	Identificador del paquete.
Id_requiere	int	Identificador del paquete requerido para su existencia .

8.3 Implementación de la solución

8.3.1 Configuración en Visual Studio. Archivo .csproj.

Hasta ahora hemos explicado cómo conseguir dar soporte a la variabilidad a nivel de interfaz gráfica. Pero lo que se pretende conseguir es una línea de producto real con productos finales completamente distintos entre sí y no sólo a nivel de interfaz. Es decir, que los productos finales sean diferentes tanto en su cara interna como en su cara externa.

Por lo tanto, también la configuración debe variar según sean elegidos unos paquetes u otros. En Visual Studio, el archivo que maneja la configuración del proyecto, es aquel cuyo nombre es el del proyecto pero con extensión .csproj. En este archivo es en el que se describe qué clases, controles o formularios son los que se van a compilar. Por esta razón, este archivo debe ser modificado si

queremos que reconozca los paquetes, con todos los elementos que se deben incluir o no a la hora de la compilación final.

El archivo Nombre_proyecto.csproj es editable con cualquier editor de textos.

Un fragmento de código de este archivo perteneciente a la inclusión del paquete Credit sería:

```
<Compile Include="Credit\OrderCreditCard.aspx.cs">
  <SubType>ASPXCodeBehind</SubType>
  <DependentUpon>OrderCreditCard.aspx</DependentUpon>
</Compile>

<Compile Include="Credit\OrderCreditCard.aspx.designer.cs">
  <DependentUpon>OrderCreditCard.aspx</DependentUpon>
</Compile>

<Compile Include="Credit\RegCreditCard.ascx.cs">
  <DependentUpon>RegCreditCard.ascx</DependentUpon>
  <SubType>ASPXCodeBehind</SubType>
</Compile>

<Compile Include="Credit\RegCreditCard.ascx.designer.cs">
  <DependentUpon>RegCreditCard.ascx</DependentUpon>
</Compile>

<Compile Include="Credit\ShoppingCartCreditCard.ascx.cs">
  <DependentUpon>ShoppingCartCreditCard.ascx</DependentUpon>
  <SubType>ASPXCodeBehind</SubType>
</Compile>

<Compile Include="Credit\ShoppingCartCreditCard.ascx.designer.cs">
  <DependentUpon>ShoppingCartCreditCard.ascx</DependentUpon>
</Compile>
```

Como podemos ver, estas sentencias nos muestran todos los formularios del paquete Credit junto con sus controles de usuario y clases correspondientes. Por lo tanto, modificando este fichero podremos incluir en la compilación los paquetes seleccionados.

8.3.2 Proyecto de configuración

Con el fin de manejar las tareas de configuración, también se ha llevado a cabo la realización de una aplicación Windows orientada a que el desarrollador de la línea de productos escoja los parámetros adecuados en los siguientes aspectos:

1. Seleccionar los paquetes que formaran parte del producto final.
2. Configuración de la Base de Datos.
3. Configuración de la Tienda Online.

Este proyecto de configuración fue desarrollado por los compañeros que realizaron el proyecto predecesor al nuestro, pero en su aplicación tan solo manejaban la posibilidad de escoger sus cuatro

DESARROLLO DE LA SOLUCIÓN

paquetes de modo que cuando se añadiera alguno más este proyecto tendría que modificarse. Lo que hemos hecho ahora ha sido automatizar el primer aspecto de los tres que hablábamos anteriormente. Es decir, el nuevo proyecto de configuración permite añadir nuevos paquetes introduciendo sus atributos en una base de datos y sin necesidad de cambiar la aplicación Windows.

El mecanismo propuesto para automatizar la selección de paquetes se basa en la utilización de una base de datos de la que extraeremos el nombre de los paquetes para mostrárselo al desarrollador en el formulario, y en la que señalaremos las dependencias entre los paquetes. Las dependencias pueden ser de dos tipos: requerir o excluir. Si un paquete requiere la presencia de otro significa que éste no puede ser seleccionado si el otro no lo está; y si un paquete excluye a otro significa que no se pueden escoger los dos paquetes a la vez. Hasta el momento no existen en el proyecto paquetes excluyentes pero si que tenemos paquetes que no se pueden añadir si otros no están. Por ejemplo, el paquete de pago rápido no puede incluirse si no se incluye también el de pago con tarjeta y el de dirección de facturación, y éstos a su vez no pueden añadirse sin el paquete de registrarse.

TABLA REQUIERE	
PAQUETE	REQUIERE A
Paquete Tarjeta de Crédito	Paquete Registrarse
Paquete Dirección de Facturación	Paquete Registrarse
Paquete Dirección de Envío	Paquete Registrarse
Paquete Pago Método Rápido	Paquete Tarjeta de Crédito
Paquete Pago Método Rápido	Paquete Dirección de Facturación
Paquete Producto Físico	Paquete Dirección de Envío
Paquete Dirección Envío	Paquete Producto Físico
Paquete Descarga Directa	Paquete Producto Electrónico

Una vez elegidos los paquetes, esta aplicación lo que hará será modificar el archivo .csproj explicado anteriormente de forma automática a nuestro gusto, con lo que a la hora de compilar el proyecto obtendremos el producto final requerido, además de configurar correctamente el archivo “paquetes.xml” necesario.

En la segunda sección se maneja todo lo relacionado con la base de datos necesaria para el sistema. Se da la posibilidad de elegir el Servidor de la Base de Datos de entre los implementados en la máquina, y una vez hecha la elección, se muestran las distintas bases de datos instaladas en él para seleccionarla como predeterminada. También se da la opción de crear una nueva base con todas las tablas y campos requeridos según se decidió en la fase de diseño, obviamente vacíos.

En la última sección, se configura todo lo relativo al comercio electrónico en sí para una mayor personalización, como puede ser el nombre de la tienda que será utilizado en diferentes partes de ella, el número de productos que aparecerán en cada página, el email de la cuenta receptora de los pagos a través de Paypal, el tipo de moneda utilizado con sus respectivos formatos, y finalmente crear el rol de administrador con su contraseña que será el encargado de llevar a cabo la gestión interna del sistema de ventas, típicamente el actor administrador de nuestros casos de uso.

Estas últimas dos secciones, también estarán disponibles en una aplicación similar orientada hacia el usuario final, para que pueda seleccionar correctamente la base de datos, además de poder hacer cambios en los campos de personalización de su tienda.

8.4 Pruebas

8.4.1 Pruebas de caja blanca

Al igual que con la aplicación web, las pruebas de caja blanca se han ido llevando a cabo de forma paralela al desarrollo del código. Estas pruebas nos han permitido asegurar el correcto funcionamiento de los métodos que se iban implementado.

8.4.2 Pruebas de caja negra

PRUEBA	RESULTADO ESPERADO	RESULTADO OBTENIDO
Seleccionar los paquetes deseados, elegir o crear la base de datos y establecer parámetros de la web, como el nombre o la moneda a utilizar.	El sistema muestra en pantalla el formulario y lleva a cabo la creación de los diferentes archivos necesarios para configurar el producto final.	CORRECTO
Seleccionar simultáneamente dos paquetes incompatibles.	El sistema no permite esta acción	CORRECTO
Seleccionar solamente uno de dos paquetes dependientes.	El sistema no permite esta acción	CORRECTO
Crear la configuración sin haber insertado el nombre de la pagina web.	El sistema no permite esta acción	CORRECTO
Crear la configuración sin haber seleccionado la moneda con la que se van a realizar los pagos.	El sistema no permite esta acción	CORRECTO

8.5 Manual

En este manual, dirigido principalmente al desarrollador del sistema, indicaremos unas pautas para la utilización de la aplicación. El objetivo principal es la creación de un archivo autoinstalable que contenga el producto final de la línea de producto y que será el que ejecute el usuario final o administrador para la instalación de su página de comercio electrónico personalizada.

Para la creación del archivo el desarrollador necesitará la herramienta de desarrollo Visual Studio. Ya dentro de ella deberá seguir los siguientes pasos:

1. Abrir el archivo "LPS_ecommerce.sln" que contiene información de la solución de todo el proyecto.

DESARROLLO DE LA SOLUCIÓN

2. Ejecutar el proyecto de Configuración. Aparecerá una ventana como la mostrada en la siguiente imagen. En ella seleccionaremos los paquetes que queremos incluir en el producto final y los parámetros referentes a la tienda: base de datos, nombre, moneda, etc... Estos parámetros podrán ser modificados mas adelante haciendo uso de la aplicación de usuario a la que hicimos referencia en la memoria y cuyo funcionamiento se detalló en el capitulo de manuales.

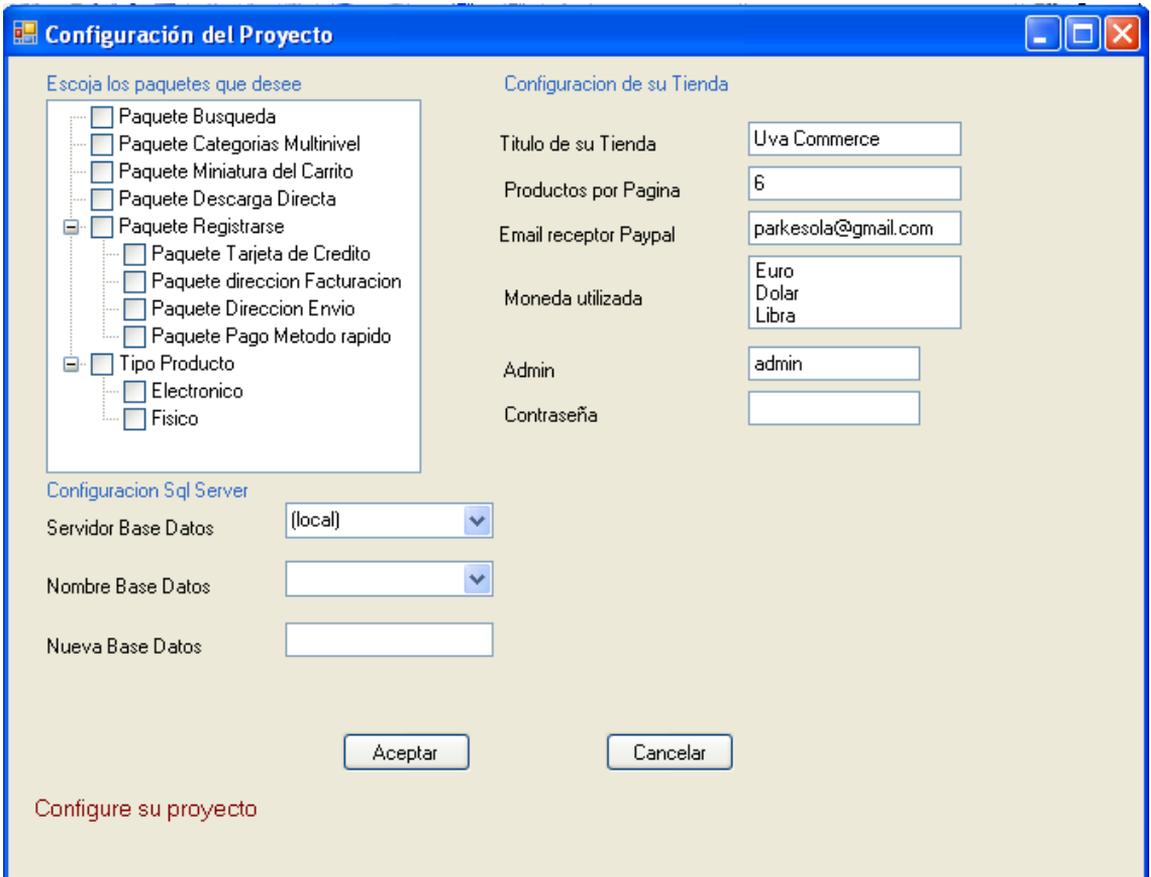


Figura 8-5 Configuración

3. Cuando hayamos rellenados los campos, pulsamos el botón "Aceptar". El sistema mostrará un mensaje informativo indicando que la configuración se ha realizado correctamente. Ya de nuevo en el Visual Studio veremos una ventana pop-up como la mostrada a continuación. Esta ventana aparece como reacción a la modificación del archivo .csproj que como ya se ha explicado a lo largo de la memoria, es el encargado de recoger los archivos pertenecientes a los paquetes elegidos y obviar aquellos que no se van a adjuntar al producto final.

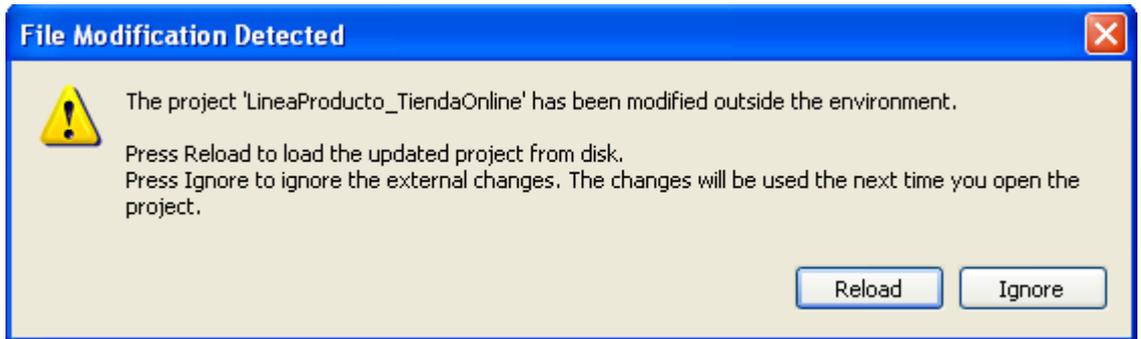


Figura 8-6 Modificar .csproj

4. Hacemos clic en el botón “Reload” o “Volver a cargar”, según este trabajando con la versión en inglés o la versión en español. En este momento ya podemos crear el producto final de nuestra línea. Para ello nos dirigimos de nuevo a la ventana del Explorador de Soluciones en el Visual Studio desde donde generamos el ejecutable tal como muestra la figura:

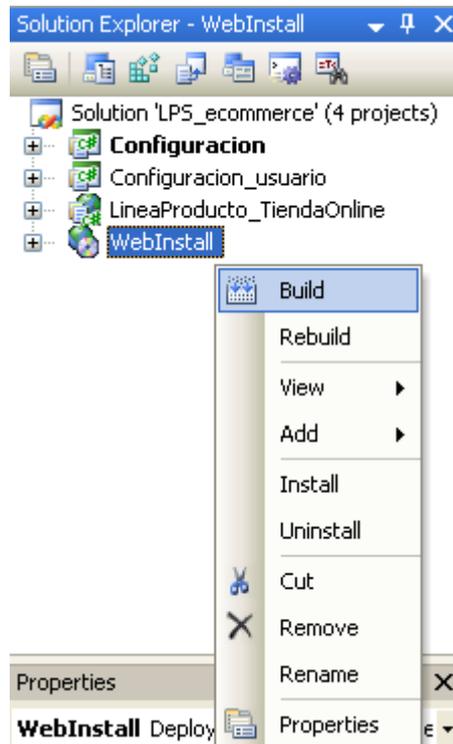


Figura 8-7 Crear proyecto

5. El archivo quedará almacenado en la carpeta Debug del proyecto de instalación. Siga las instrucciones del Manual de instalación en el capítulo Manuales para instalarlo en otros servidores.

8.6 Conclusiones del capítulo

Partiendo del objetivo establecido al iniciar esta aplicación, que no es otro que facilitar la tarea de la modificación de los archivos de configuración a futuros desarrolladores, podemos decir que se ha conseguido satisfactoriamente.

Nuestra principal meta era automatizar el proceso de forma que el código de la aplicación se tuviera que rectificar lo menos posible cada vez que un nuevo desarrollador implementara nuevos paquetes. Esto se ha logrado gracias a la utilización de la base de datos como ya hemos explicado en anteriores apartados.

En un futuro se podría realizar una nueva aplicación en la que el sistema detectara directamente los paquetes implementados hasta el momento sin necesidad de tener que hacer referencia a ellos en la base de datos.

CONCLUSIONES

9 Conclusiones

9.1 Resumen y valoraciones del trabajo

Tras finalizar la implementación y viendo los resultados conseguidos tras la realización de las pruebas, podemos afirmar que los objetivos propuestos en las primeras etapas del proyecto se han cumplido totalmente.

Nuestro trabajo comenzó a desarrollarse a partir de un proyecto anterior, por lo que en primer lugar, tuvimos que entender los mecanismos y el modo de trabajo utilizados por nuestros compañeros. Uno de nuestros objetivos principales era la implementación de nuestras tareas sin que el proyecto anterior perdiera ninguna funcionalidad e intentando en todo momento modificar lo menos posible sus clases e interfaces. Este objetivo se ha cumplido con éxito gracias a la distribución en paquetes y a que todos nuestros paquetes son independientes de los suyos.

Una vez más ha quedado probada la variabilidad de una línea de productos ya que aparte de los cuatro paquetes opcionales que ofrecían nuestros compañeros, nosotras hemos implementado cinco más que se pueden añadir a la solución inicial o alternar en diferentes combinaciones.

A continuación se enumerarán los objetivos conseguidos:

- ✓ Desarrollar una línea de producto que nos permita la selección de un producto final personalizado para cada cliente.
- ✓ Ampliación de la aplicación para manejar la variabilidad de una línea de productos añadiendo a la misma las siguientes funcionalidades:
 - Registro de usuarios.
 - Identificación de los mismos en el sistema.
 - Pago de compras a través de tarjeta de crédito.
 - Pago usando un método rápido que almacena un perfil del usuario.
 - Emisión de facturas asociadas a una dirección fiscal.
 - Compra de productos físicos.
- ✓ Implementación de una aplicación para la creación de los ficheros de configuración .csproj más automatizada que la realizada para el proyecto anterior. Esto ayudará a que en proyectos futuros no sea necesario modificar el código de la configuración para poder añadir los nuevos paquetes, sino que bastará con que se registren los atributos necesarios en la base de datos asociada.
- ✓ Correcta instalación del certificado SSL en el servidor IIS. Debido al elevado tráfico de datos que conlleva el registro de clientes y el pago por tarjeta se creyó necesario el tratamiento de los datos de forma segura. Esto aporta confianza al usuario impidiendo acciones malintencionadas.

CONCLUSIONES

Finalmente, podemos decir que este proyecto nos ha enseñado a saber enfrentarnos a una aplicación de esta envergadura y a conocer nuestras capacidades y conocimientos. Hemos tenido que realizar las labores de análisis y desarrollo, así como la implementación y las pruebas y desde un principio hemos sabido organizar el tiempo y dividir el trabajo para que todo saliera correctamente.

También nos hemos familiarizado con el mundo del comercio electrónico que es una actividad en auge en la época en la que vivimos, ya que cada vez son más las personas que compran por medio de Internet y las empresas que desean realizar sus ventas a través de la red.

Por último, resaltar la utilidad de las herramientas utilizadas, porque tanto el .NET como el manejo de bases de datos con SQL Server son muy demandados en el mundo laboral. Hemos partido desde cero en el uso de estas herramientas ya que las desconocíamos por completo, y hemos superado con creces el proceso de aprendizaje.

9.2 Problemas y dificultades

El primer problema que se nos planteó fue el partir de un proyecto realizado por otros compañeros y que para nosotras era totalmente desconocido. Por lo tanto, tuvimos que estudiar y comprender completamente el funcionamiento de su aplicación y los mecanismos utilizados por ellos para después aplicar métodos similares y que el resultado final fuera homogéneo.

Aun así, algunas líneas de trabajo seguidas por ellos no eran válidas para nosotras. Un ejemplo de esto lo encontramos en los requisitos no funcionales: una de sus necesidades era que todos los paquetes fueran independientes entre sí. En nuestro caso esto no ocurre, ya que existen paquetes que requieren de otros para su correcto funcionamiento. Esto nos llevó a enfrentarnos al problema de manejar todos los nuevos elementos tanto a nivel de interfaz como en el ámbito de configuración, teniendo presente que existían dependencias entre algunos de ellos.

También hemos encontrado costoso el aprendizaje de ASP.NET y del lenguaje C#, a pesar de que en la asignatura de Ingeniería de Software II trabajamos en un ámbito similar utilizando Java.

En lo que se refiere al uso de paginas seguras, la búsqueda de información ha sido una ardua tarea, ya que nunca habíamos tratado con servidores y mucho menos con la instalación de certificados en estos. También hay que tener en cuenta que la mayoría de estos certificados no son gratuitos y tuvimos que encontrar una pagina que nos proporcionara uno de prueba durante 90 días.

Por último, la planificación del trabajo ha sido una dura tarea, ya que no podíamos trabajar a la vez sobre la aplicación porque era difícil sincronizar las modificaciones hechas por separado. Debido a ello, repartíamos las tareas de investigación, implementación y desarrollo de la memoria de forma que no fuera complicado poner los cambios en común.

9.3 Trabajos futuros

Como hemos dicho anteriormente, este PFC es una ampliación de otro proyecto previo. Por esta razón, al igual que la parte realizada por nosotras, existen otras muchas extensiones a desarrollar. Todo el desarrollo de la línea de productos está basado en el modelo de características de la tesis de

maestría de Lau y en él vienen especificados los trabajos futuros a realizar para completar la funcionalidad final.

Algunas de las funciones que se pueden añadir son:

- Completar el registro detallando características del usuario como pueden ser datos demográficos (edad, ingresos, nivel de educación...) o hobbies e intereses.
- Implementar nuevas formas de pago.
- Desarrollar distintas opciones de envío de pedidos.
- Realizar una nueva configuración para detectar los paquetes implementados hasta el momento sin necesidad de detallarlo en la base de datos.

Sabemos por lo tanto que es posible que este proyecto vaya a dar pie otros nuevos. Por esta razón, a la hora de escribir el código hemos procurado utilizar nombres significativos para las variables y métodos, así como comentarios que ayuden a dar a una visión más clara de nuestro trabajo a futuros desarrolladores.

BIBLIOGRAFÍA

Referencias Bibliográficas

P.C. Clements and L. Northrop. *Software Product Lines: Practices and Patterns*. SEI Series in Software Engineering. Addison–Wesley, August 2001.

K. Czarnecki, M. Antkiewicz. Mapping features to models: A template approach based on superimposed variants. In *Proceedings of International Conference Generative Programming and Component Engineering (GPCE'05)*, vol. 3676 of LNCS, Springer, 2005.

Freeman, P. “*A Perspective on Reusability*”. IEEE Tutorial: Software Reusability (ed. P. Freeman), IEEE Computer Society Press, pp. 2-8. 1987.

José Antonio González Seco, 2001, *El Lenguaje de Programación C#*

Karlsson, Even-André. “*Software Reuse. A Holistic Approach*”. John Wiley & Sons Ltd. 1995.

Sean Quan Lau. *Domain Analysis of E-commerce Systems Using Feature-Based Model Templates*. MSc Thesis, Electrical and Computer Engineering, University of Waterloo, 2006.

McIlroy, M. D. “*Mass-produced Software Components*”. In J.M. Buxton, P. Naur, and B. Randell, editors, *Software Engineering Concepts and Techniques*; 1968 NATO Conference on Software Engineering, pp. 88-98. Van Nostrand Reinhold, 1976.

M. Parihar, 2002, *La Biblia de ASP.NET*, Ediciones Anaya Multimedia S.A.

Svahnberg, M., van Gurp, J., and Bosch, J., 2001, “*On the Notion of Variability in Software Product Lines*”, In *Proc. of the Working IEEE/IFIP Conference on Software Architecture*. IEEE Computer Society Press, 2001.

J. Templeman, D.Vitter, 2002, *La Biblia de Visual Studio .NET*, Ediciones Anaya Multimedia S.A.

Fuentes Web

<http://www.adrformacion.com/cursos/aspnet35av/leccion1/tutorial2.html>, página web donde se puede encontrar un curso de ASP .NET avanzado (Última visita 11 Julio, 2008)

<http://www.desarrolloweb.com> , página donde se pueden encontrar diversos manuales sobre la programación web, así como un foro donde los usuarios se ayudan entre ellos. (Última visita 2 Agosto, 2008)

<http://www.dotnetclubs.com/forums/t/128.aspx>, foro de estudiantes donde se plantean y resuelven problemas relacionados con la informática y la programación (Última visita 16 Agosto, 2008)

<http://www.elguille.info> , página web de consultas donde se pueden encontrar multitud de recursos para desarrolladores de software. (Última visita 12 Agosto, 2008)

<http://www.forosdelweb.com/>, foro cuyo tema principal es el diseño y desarrollo de páginas web (Última vista 19 Agosto, 2008)

<http://www.instantssl.com/ssl-certificate-products/free-ssl-certificate.html>, página web que permite descargar gratuitamente certificados de seguridad para un periodo de prueba de 90 días (Última visita 17 Agosto, 2008)

<http://www.locualo.net/programacion/activar-ssl-iis-certificado-digital-prueba/00000079.aspx> , página web donde se pueden encontrar las instrucciones para activar un certificado SSL (Última visita 17 Agosto, 2008)

<http://www.microsoft.com/downloads/details.aspx?displaylang=es&FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5>, página oficial de descarga del Framework .NET 2.0 de Microsoft (Última visita 14 Junio, 2008)

<http://www.microsoft.com/spanish/msdn/vstudio/express/SQL/default.msp>, página de descarga oficial de SQL Server 2005 de Microsoft (Última visita 15 Junio, 2008)

<http://www.softwareproductlines.com> , sitio web dedicado a las líneas de producto software, donde el autor Charles W. Krueger, explica de forma concisa todo lo relevante a las mismas (Última visita 20 Julio, 2008)

<http://www.tutorial-enlace.net/>, página web con multitud de tutoriales sobre Visual Studio (Última visita 8 Julio, 2008)

APÉNDICES

Apéndice A .NET

En este capítulo explicaremos en qué consiste la plataforma .NET y, con ello, la herramienta de desarrollo Visual Studio .NET. Además, trataremos el estudio de la programación del lenguaje C# mostrando sus principales características. Indicaremos cuales son las nuevas ideas que aporta al entorno de los desarrolladores y explicaremos por qué surge realmente la necesidad de crear un nuevo lenguaje de programación.

A.1 Plataforma Microsoft .NET

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones.

Su propuesta es ofrecer una manera rápida, económica, segura y robusta de desarrollar aplicaciones permitiendo una integración más rápida y ágil entre empresas, y un acceso más simple y universal a toda la información desde cualquier tipo de dispositivo.

Con estos objetivos, Internet aparece como la base de un sistema operativo distribuido sobre el cual se ejecutarán aplicaciones que estarán preparadas para relacionarse entre sí de manera transparente. La programación del futuro se hará sobre un gran sistema operativo que residirá en Internet de forma que la información y las aplicaciones, servicios en este caso, ya no estarán en nuestro PC, sino en la Red.

Microsoft proporciona una plataforma que incluye los siguientes componentes básicos:

- Infraestructura de servidores, incluyendo Windows y .NET Enterprise Servers.
- Software de dispositivos .NET para hacer posible una nueva generación de dispositivos inteligentes (ordenadores, teléfonos, PDAs, consolas de juegos, etc) que puedan funcionar en .NET
- Herramientas de programación para crear servicios Web XML, con soporte multilinguaje: .NET Framework y Visual Studio.

A.1.1 .NET Framework

Infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que facilitan el desarrollo de aplicaciones. Mediante esta herramienta, el proceso de encontrar un servicio web e integrarlo en una aplicación resulta transparente para usuarios y desarrolladores.

Según se puede ver en la figura siguiente, el Framework de .NET es un entorno de ejecución y un componente de desarrollo multilinguaje.

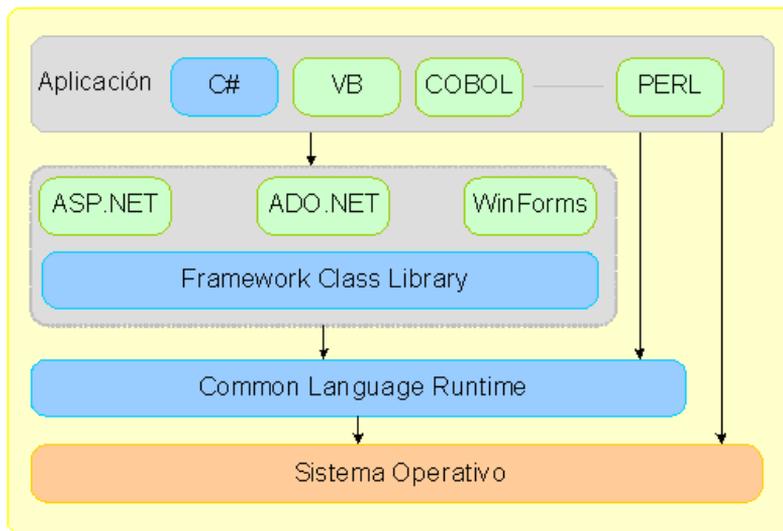


Figura A-1 El Framework de .NET

A.1.2 Lenguajes de compilación

.NET Framework soporta múltiples lenguajes de programación, pudiendo desarrollar cualquier aplicación con cualquiera de los más de 30 lenguajes adaptados a .NET, tales como C#, C++, Visual Basic e incluso Cobol.

A.1.3 Biblioteca de clases (Framework Class Library)

La biblioteca de clases de .NET Framework es una biblioteca de clases, interfaces y tipos de valor que se incluye en Microsoft .NET Framework SDK. Esta biblioteca brinda acceso a la funcionalidad del sistema y es la base sobre la que se crean las aplicaciones, los componentes y los controles de .NET Framework.

Se pueden utilizar las clases tal y como están, o bien derivarlas en las clases que se vayan a utilizar en la aplicación.

La Biblioteca de Clases Base se clasifica en tres grupos base:

- ASP.Net y Servicios Web XML para construir aplicaciones y servicios Web
- Windows Forms para desarrollar interfaces de usuario
- ADO.NET para conectar las aplicaciones a bases de datos

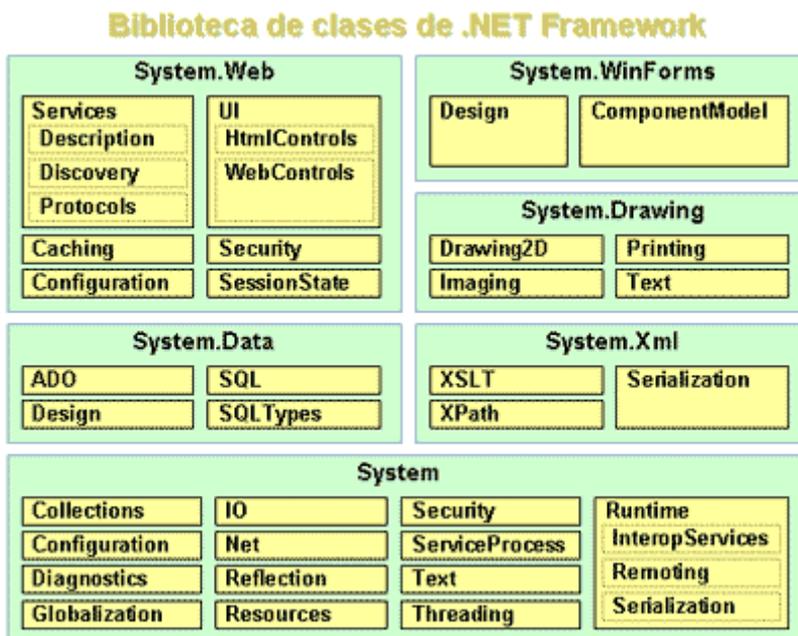


Figura A-2 Biblioteca de clases de .NET

A.1.4 Entorno de Ejecución Común de los Lenguajes CLR (Common Language Runtime)

El *Common Language Runtime (CLR)* constituye el núcleo de .NET Framework.. Se trata de una máquina virtual que administra la ejecución del código y engloba una serie de características comunes a todos los lenguajes de programación. Algunas de estas características son las siguientes:

- Ejecución multiplataforma, cualquier plataforma para la que exista una versión del CLR podrá ejecutar cualquier aplicación .NET
- Integración de lenguajes, ya que por ejemplo es posible escribir una clase en C# que herede de otra escrita en Visual Basic.NET que, a su vez, herede de otra escrita en C++ con extensiones gestionadas.
- Gestión de memoria automático, incluyendo un recolector de basura que evita que el programador tenga que tener en cuenta cuándo ha de destruir los objetos que dejen de serle útiles.
- Seguridad de tipos, de modo que en todas las conversiones que se realicen los tipos sean compatibles.
- Aislamiento de procesos, asegurando que desde código perteneciente a un determinado proceso no se pueda acceder a código o datos pertenecientes a otro.

APÉNDICES

- Todos los errores que se puedan producir durante la ejecución de una aplicación se propagan mediante excepciones.
- Es capaz de trabajar con aplicaciones divididas en múltiples hilos de ejecución que pueden ir evolucionando por separado en paralelo o intercalándose.
- Ofrece la infraestructura necesaria para crear objetos remotos y acceder a ellos de manera completamente transparente a su localización real.
- Proporciona mecanismos de seguridad avanzada para restringir la ejecución de ciertos códigos o los permisos asignados a los mismos según su procedencia o el usuario que los ejecute.

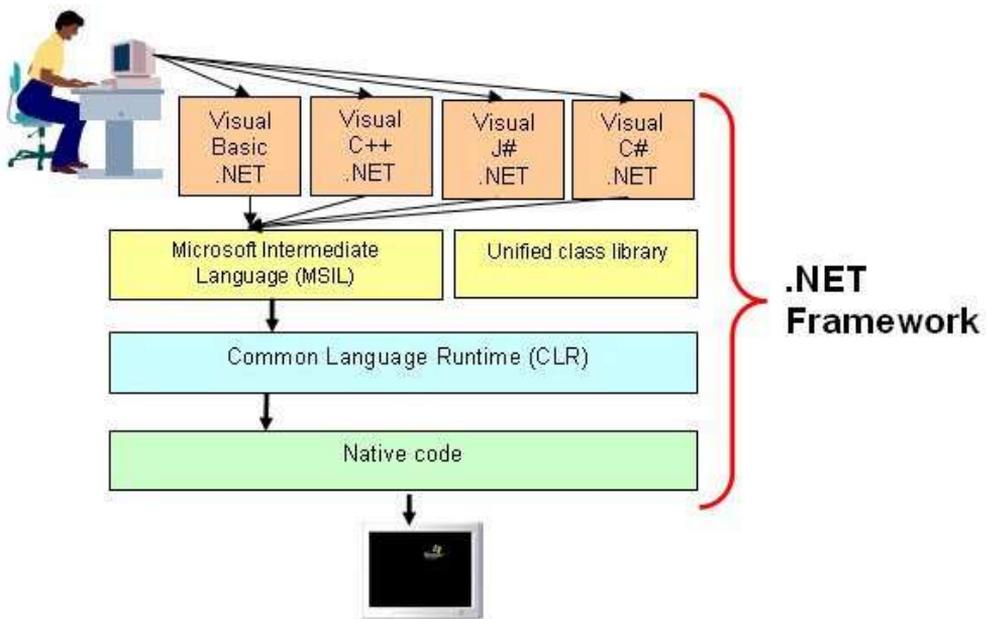


Figura A-3 Funcionamiento de .NET Framework

Su sistema de funcionamiento es el siguiente: el código fuente se compila para crear código intermedio. Posteriormente, es convertido a código nativo por un compilador Just In Time (JIT). Este código nativo es el código específico de la CPU del ordenador sobre el que se está ejecutando el JIT, que se encuentra situado en el CLR del Framework. Después de esta conversión, el código ya puede ser ejecutado.

A.1.5 Visual Studio

Es un conjunto complejo de herramientas de desarrollo para construir aplicaciones Web, servicios Web, aplicaciones Windows o de escritorio y aplicaciones para dispositivos móviles. Se pueden crear soluciones utilizando varios lenguajes y en las que la parte de diseño se implementa separadamente con respecto a la programación.

A.1.6 ASP.NET y Visual C#

A.1.6.1 ASP.NET

ASP.NET es un [framework para aplicaciones web](#) desarrollado y comercializado por [Microsoft](#). Con la llegada de ASP.NET se ha facilitado enormemente el desarrollo de aplicaciones y la productividad de los programadores, ya que permite dotar de funciones adicionales a un aplicación Web y escribir una menor cantidad de código. La mayor ventaja es que permite trabajar con cualquier lenguaje de programación .NET.

Además, las aplicaciones Web permiten utilizar lenguajes de programación compilados, lo que hace que la ejecución sea mucho más rápida. El proceso es sencillo: un cliente solicita la ejecución de una aplicación que reside en un determinado servidor. Si dicha página no se ha compilado nunca, el sistema .NET Framework se encarga de compilar la aplicación, ejecutarla y devolver la respuesta de la ejecución al cliente. Esta respuesta se envía al navegador Web en formato HTML, soportado por todos los navegadores.

A la vez, ofrece un alto rendimiento, fiabilidad y seguridad por lo que los usuarios tienen más confianza a la hora de utilizar las aplicaciones ASP.NET.

A.1.6.2 Visual C#

Es el nuevo lenguaje diseñado por Microsoft para crear una amplia gama de aplicaciones que se ejecutan en .NET Framework.. C# es simple, eficaz, con seguridad de tipos y orientado a objetos. Con él se pretende dar una alternativa a Java, que es su más directo competidor en el mundo de Internet.

C# combina elementos de múltiples lenguajes como C++, Java, Delphi o Visual Basic, pero es el único diseñado específicamente para ser utilizado en la plataforma .NET. Debido a esto, programar en C# resulta mucho más sencillo e intuitivo que con el resto de lenguajes.

C# es el lenguaje nativo para acceder a todos los servicios que proveerá .NET en el futuro, y sigue evolucionando continuamente para proporcionar mayor eficiencia y funcionalidad.

A continuación enumeramos las principales características que definen al lenguaje de programación C#. Algunas de estas características no son propias del lenguaje, sino de la plataforma .NET, aunque se listan aquí ya que tienen una implicación directa en el lenguaje:

- Sencillez de uso : C# elimina muchos elementos añadidos por otros lenguajes y que facilitan su uso y comprensión. Es por ello que se dice que C# es autocontenido.
- Modernidad : Al ser C# un lenguaje de última generación, incorpora elementos que se ha demostrado a lo largo del tiempo que son muy útiles para el programador y que otros lenguajes habría que simularlos.
- Orientado a objetos : C# soporta todas las características del paradigma de la programación orientada a objetos, como la encapsulación, la herencia y el polimorfismo.
- Orientado a componentes : La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular.
- Recolección de basura : Todo lenguaje incluido en la plataforma .NET tiene a su disposición el recolector de basura del CLR. Esto implica que no es necesario incluir instrucciones de destrucción de objetos en el lenguaje.
- Seguridad de tipos : C# incluye mecanismos de control de acceso a tipos de datos, lo que garantiza que no se produzcan errores difíciles de detectar. Para ello, el lenguaje provee de una serie de normas de sintaxis, no se pueden usar variables no inicializadas previamente, y en el acceso a tablas se hace una comprobación de rangos.
- Instrucciones seguras : Para evitar errores comunes como se producían programando en otros lenguajes, en C# se han impuesto una serie de restricciones en el uso de instrucciones de control más comunes.
- Unificación de tipos : En C# todos los tipos derivan de una superclase común llamada System.Object, por lo que automáticamente heredarán todos los miembros definidos en esta clase. Es decir, son objetos.
- Extensión de los operadores básicos : C# permite redefinir el significado de la mayoría de los operadores (incluidos el de la conversión) cuando se apliquen a diferentes tipos de objetos.
- Extensión de modificadores : C# ofrece, a través de los atributos, la posibilidad de añadir a los metadatos del módulo resultante de la compilación de cualquier fuente información adicional a la generada por el compilador que luego podrá ser consultada en tiempo de ejecución a través de la biblioteca de reflexión de .NET.
- Eficiente : En C#, todo el código incluye numerosas restricciones para garantizar su seguridad, no permitiendo el uso de punteros. Sin embargo, y a diferencia de Java, existen modificadores para saltarse esta restricción, pudiendo manipular objetos a través de punteros.
- Compatible : Para facilitar la migración de programadores de C++ o Java a C#, no sólo se mantiene una sintaxis muy similar a la de los dos anteriores lenguajes, sino que el CLR también ofrece la posibilidad de acceder a código nativo escrito como funciones sueltas no orientadas a objetos.

Apéndice B Herramientas utilizadas

Microsoft Visual Studio 2005

Microsoft Visual Studio es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones Web ASP.NET. Visual C#, Visual C++, Visual J#, y Visual Basic utilizan el mismo entorno de desarrollo integrado (IDE) pudiendo así compartir herramientas y crear soluciones en distintos lenguajes.

Utilizando esta herramienta se pueden desarrollar sitios, aplicaciones y servicios web en cualquier entorno que soporte la plataforma .NET, siendo posible crear aplicaciones que se intercomunicuen entre estaciones de trabajo, páginas web y dispositivos móviles.

SQL Server 2005

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (SGBD) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Así de tener unas ventajas que a continuación se pueden describir.

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, Sybase ASE, PostgreSQL o MySQL.

SQL Server 2005 es más que un sistema gestor de Bases de Datos ya que incluye múltiples componentes y servicios que la convierten en una plataforma de aplicaciones corporativas.

Con la aparición de SQL Server 2005 el mundo de las Bases de datos está cambiando. Los desarrolladores ahora pueden ubicar su código apropiadamente en relación a su funcionalidad, acceder a datos nativos como XML, y construir sistemas complejos que sean manejados por el servidor de Bases de Datos. Estos puntos hacen que el desarrollo de Bases de Datos esté encaminado hacia una integración.

REM

REM (Requisite Management) es una herramienta experimental gratuita de Gestión de Requisitos diseñada para soportar la fase de Ingeniería de Requisitos de un proyecto de desarrollo software de acuerdo con la metodología definida en la Tesis Doctoral "Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información", presentada por Amador Durán en septiembre de 2000 en la Universidad de Sevilla.

En nuestro caso hemos utilizado esta herramienta para la descripción de usuarios, actores, requisitos y casos de uso.

StarUml

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

StarUML es una herramienta para el modelamiento de software basado en los estándares UML (Unified Modeling Language) y MDA (Model Driven Architecture). Ofrece soporte completo al diseño UML mediante el uso de diagramas de casos de uso, diagramas de secuencia, diagramas de clases y diagramas de estado. Además, se puede exportar toda la documentación generada a cualquier programa de Microsoft Office (Word, Excel, PowerPoint...)

Esta herramienta la hemos utilizado para diseñar todos los diagramas necesarios para el análisis y el diseño, casos de uso, clases, secuencia y estado.

Microsoft Word

Microsoft Word es un procesador de texto creado por Microsoft, y actualmente integrado en la *suite* ofimática Microsoft Office.

Hemos utilizado esta herramienta para la creación del documento de la memoria.

PDF Creator 3.0

PDF Creator instala una impresora virtual en nuestro sistema y convierte la salida de cualquier programa a un archivo PDF. Resulta extremadamente útil, porque se puede editar un documento con cualquier programa conocido por el usuario.

Microsoft PowerPoint 2000

Programa diseñado para hacer presentaciones prácticas con texto esquematizado, fácil de entender, animaciones de texto e imágenes, imágenes prediseñadas o importadas desde imágenes de la computadora.

Apéndice C. Contenido del CD-Rom

El CD-ROM adjunto consta de cuatro directorios principales:

- Directorio Código de la Aplicación: contiene todo el código de la solución.
- Directorio Documentación: en él se encuentra todo el documento de esta memoria en formato PDF. Así como el archivo UML de análisis y diseño.
- Directorio Productos Finales: en él hay dos ejecutables: version1.exe y version2.exe, que corresponden a dos productos diferentes, del total de productos de la línea.
- Directorio Software Entorno Desarrollo y Explotación: aquí se han incluido los programas necesarios para el correcto funcionamiento de la aplicación, como son el SQL Server y el Framework .NET 2.0