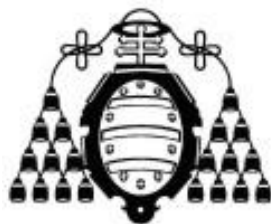


UNIVERSIDAD DE OVIEDO



MÁSTER EN INGENIERÍA WEB

TRABAJO FIN DE MÁSTER

“METAMODELOS DE PROCESOS DE COMERCIO ELECTRÓNICO”

DIRECTOR: B.Cristina Pelayo García-Bustelo

CODIRECTOR: Giovanni Mauricio Tarazona Bermúdez

AUTOR: Ana Belén Rodríguez Arias

Agradecimientos

Agradecer en primer lugar mi directora Cristina Pelayo García y a mi codirector Giovanni Mauricio Tarazona, por su dedicación, ayuda y esfuerzo dedicados a lo largo de todo el proceso.

Por otro lado, agradecer a mis compañeros y amigos que siempre están ahí para darme ánimos y aconsejarme en momentos en los que no tenía claro por dónde seguir.

Por último, agradecer a mi familia por apoyarme siempre y en especial a mis padres por estar a mi lado siempre y darme ánimos para seguir adelante en los momentos que resultaron más difíciles.

Muchas gracias a todos.

Resumen

El proyecto está enmarcado dentro de las líneas de investigación del grupo MDE-RG de la Universidad de Oviedo en cooperación con el grupo de investigación en Comercio Electrónico Colombiano de la Universidad Distrital Francisco José de Caldas en Colombia. El objetivo del proyecto es la validación e implementación de un metamodelo de comercio electrónico, propuesto en una Tesis Doctoral adscrita al departamento de informática de la universidad de Oviedo.

El comercio electrónico es la compra y venta de productos o servicios a través de medios electrónicos como Internet y otras redes informáticas, su implementación ofrece ventajas a vendedores y compradores. La aplicación y uso de comercio electrónico facilita a vendedores acceder a segmentos de mercados estrechos mientras que los compradores pueden beneficiarse al acceder a los mercados mundiales con mayor disponibilidad de productos y variedad de ofertas a costos reducidos, esta situación mejora la calidad del producto y la creación de nuevas formas de negocio.

Las soluciones de comercio electrónico disponibles actualmente en el mercado se caracterizan, en general, por el alto coste, la diversidad de los datos y formatos, esto genera dificultades técnicas que restringen tanto la interoperabilidad como la reutilización de la información. Por otra parte está el interés de los proveedores de plataformas de comercio electrónico por mantener cuotas de mercado, que impiden a sus usuarios la migración de sus tiendas virtuales a otros proveedores. Esta situación influye para que las empresas no incursionen de manera decidida a utilizar internet como un canal de distribución formal.

La solución propuesta, DSL TUNOS COMMERCE, es una aplicación orientada a la integración de plataformas de comercio electrónico. Las funcionalidades previstas facilitan la tarea de integradores tecnológicos, consultores/asesores o usuarios no expertos que requieran mejorar su experiencia en la importación y exportación de metadatos de las plataformas Prestashop y Magento, principales plataformas existentes en el mercado.

Su instalación y operación permitirá reducir sustancialmente esfuerzos de programación, tales como tareas de configuración independiente, consultas redundantes, migraciones complejas, optimizando los beneficios al momento de gestionar contenidos de comercio electrónico. Su arquitectura, desarrollo y herramientas cumplen los estándares y están elaboradas con tecnologías de última generación

Palabras Clave

Comercio Electrónico, ontologías, interoperabilidad, ingeniería dirigida por modelos (MDE), metamodelos, Lenguaje de Dominio Especifico DSL.

Abstract

The project is framed within the research lines MDE-RG group at the University of Oviedo in cooperation with the research group GICOECOL at the Francisco José de Caldas District University in Colombia. The project's objective is the validation and implementation of an e-commerce metamodel proposed in a doctoral thesis attached to Computer Science Department at the University of Oviedo.

E-commerce is the buying and selling of products or services by electronic means such as the Internet and other computer networks, its implementation offers advantages to sellers and buyers. The application and use of e-commerce facilitates access to sellers narrow market segments while buyers can benefit from access to world markets more product availability and diversity of offers reduced costs, this situation improves product quality and the creation of new forms of business.

Ecommerce solutions available on the market are generally characterized by the high cost, various data and formats, this creates technical difficulties that restrict both interoperability and reuse. On the other hand is the interest of providers of e-commerce platforms to maintain market shares, hindering its users to migrate their virtual stores to other providers. This situation influences to companies decisively not use internet as a formal distribution channel.

The proposed solution, DSL Tunos COMMERCE, is an application oriented to the integration of e-commerce platform. The functionalities provided easy the task of integrating technology, consultants / advisors or non-expert users who need to improve their experience in the import and export of metadata Prestashop and Magento platforms, main platforms on the market.

Its installation and operation will allow substantially reduce programming efforts such as independent setup tasks, redundant queries, complex migration, optimizing the benefits when managing e-commerce content. Its architecture, development and tools fulfill standards and are made with the latest technology.

Índice General

CAPÍTULO 1. MEMORIA DEL PROYECTO	18
1.1 RESUMEN DE LA MOTIVACIÓN, OBJETIVOS Y ALCANCE DEL PROYECTO	18
1.2 ALCANCE	18
1.3 RESUMEN DE TODOS LOS ASPECTOS	18
1.3.1 Memoria del proyecto	19
1.3.2 Introducción	19
1.3.3 Aspectos teóricos.....	19
1.3.4 Solución propuesta	19
1.3.5 Planificación del proyecto y resumen de presupuestos.....	19
1.3.6 Análisis del sistema	20
1.3.7 Diseño del sistema	20
1.3.8 Implementación del sistema	20
1.3.9 Desarrollo de las pruebas	20
1.3.10 Manuales del Sistema	21
1.3.11 Conclusiones y ampliaciones	21
CAPÍTULO 2. INTRODUCCIÓN	22
2.1 JUSTIFICACIÓN DEL PROYECTO	22
2.2 OBJETIVOS DEL PROYECTO.....	23
2.3 ESTUDIO DE LA SITUACIÓN ACTUAL.....	24
2.3.1 Evaluación de Plataformas de Comercio electrónico	24
2.3.2 Evaluación de Módulos de exportación de Datos	26
2.3.3 Creación de una ontología de comercio electrónico	28
2.3.4 Metamodelo de comercio electrónico	40
CAPÍTULO 3. ASPECTOS TEÓRICOS	43
3.1 COMERCIO ELECTRÓNICO	43
3.2 INTEROPERABILIDAD.....	45
3.3 ONTOLOGÍAS.....	47
3.4 MAPAS DE CONOCIMIENTO	48
3.5 MDE	49
3.6 LENGUAJES DE DOMINIO ESPECÍFICO (DSL).....	50
3.6.1 Tipos de lenguajes de dominio específico (DSL)	51
3.6.2 Partes de un lenguaje de dominio específico (DSL).....	52
3.7 INGENIERÍA DIRIGIDA POR MODELOS (MDE) CON ECLIPSE	53
3.7.1 Meta-metamodelo ecore de eclipse.....	53
3.7.2 El Metamodelo	54
3.7.3 Construcción del editor para el modelo o lenguaje de dominio específico (DSL)	55
3.7.4 El Modelo	55
3.7.5 Proceso de generación de código.....	55
CAPÍTULO 4. SOLUCIÓN PROPUESTA.....	56
4.1 DEFINICIÓN DE LA SOLUCIÓN PROPUESTA	56
CAPÍTULO 5. PLANIFICACIÓN DEL PROYECTO Y RESUMEN DE PRESUPUESTOS.....	61
5.1 PLANIFICACIÓN	61
5.2 RESUMEN DEL PRESUPUESTO	67

CAPÍTULO 6. ANÁLISIS	68
6.1 DEFINICIÓN DEL SISTEMA	68
6.1.1 <i>Determinación del Alcance del Sistema</i>	68
6.2 REQUISITOS DEL SISTEMA	69
6.2.1 <i>Obtención de los Requisitos del Sistema</i>	69
6.2.2 <i>Identificación de Actores del Sistema</i>	71
6.2.3 <i>Especificación de Casos de Uso</i>	71
6.3 IDENTIFICACIÓN DE LOS SUBSISTEMAS EN LA FASE DE ANÁLISIS	86
6.3.1 <i>Descripción de los Subsistemas</i>	86
6.3.2 <i>Descripción de los Interfaces entre Subsistemas</i>	88
6.4 DIAGRAMA DE CLASES PRELIMINAR DEL ANÁLISIS	89
6.4.1 <i>Diagramas de Paquetes</i>	89
6.4.2 <i>Diagramas de clases</i>	92
6.4.3 <i>Descripción de las Clases</i>	98
6.5 ANÁLISIS DE CASOS DE USO Y ESCENARIOS	109
6.5.1 <i>Caso de uso: Registro de usuarios</i>	109
6.5.2 <i>Caso de uso: Configurar tienda</i>	109
6.5.3 <i>Caso de uso: Importar elementos</i>	110
6.5.4 <i>Caso de uso: Importar XML</i>	111
6.5.5 <i>Caso de uso: exportar elementos</i>	112
6.5.6 <i>Caso de Uso: Generar CSV</i>	113
6.5.7 <i>Caso de uso: generar XML</i>	113
6.6 ANÁLISIS DE INTERFACES DE USUARIO	115
6.6.1 <i>Pantallas Plataforma Web</i>	115
6.6.2 <i>Pantallas Plataforma GMF</i>	133
6.6.3 <i>Descripción del Comportamiento de la Interfaz</i>	134
6.6.4 <i>Diagrama de Navegabilidad</i>	134
6.7 ESPECIFICACIÓN DEL PLAN DE PRUEBAS	137
6.7.1 <i>Pruebas unitarias</i>	137
6.7.2 <i>Pruebas de usabilidad</i>	137
6.7.3 <i>Pruebas de integración y del sistema</i>	137
CAPÍTULO 7. DISEÑO DEL SISTEMA	149
7.1 ARQUITECTURA DEL SISTEMA	149
7.1.1 <i>Arquitectura de la aplicación web</i>	150
7.1.2 <i>Diagramas de Paquetes</i>	151
7.1.3 <i>Diagrama de Despliegue</i>	154
7.2 DISEÑO DE CLASES	156
7.2.1 <i>Diagramas de Clases</i>	156
7.3 DIAGRAMAS DE INTERACCIÓN	168
7.3.1 <i>Caso de uso: Registrar usuario</i>	168
7.3.2 <i>Caso de uso: Configurar tienda</i>	170
7.3.3 <i>Caso de uso: Importar elementos</i>	172
7.3.4 <i>Caso de uso: Exportar elementos</i>	174
7.3.5 <i>Caso de uso: Generar CSV</i>	176
7.3.6 <i>Caso de uso: Generar XML</i>	177
7.4 DISEÑO DE LA BASE DE DATOS	178
7.4.1 <i>Descripción del SGBD Usado</i>	178
7.4.2 <i>Integración del SGBD en Nuestro Sistema</i>	178
7.4.3 <i>Diagrama E-R</i>	179
7.5 DISEÑO DE LA INTERFAZ	180

7.5.1	Pantallas plataforma Web.....	180
7.5.2	Pantallas Plataforma GMF.....	199
7.6	ESPECIFICACIÓN TÉCNICA DEL PLAN DE PRUEBAS	200
7.6.1	Pruebas Unitarias	200
7.6.2	Pruebas de Integración y del Sistema.....	200
7.6.3	Pruebas de Usabilidad	211
7.6.4	Pruebas de rendimiento.....	215
CAPÍTULO 8.	IMPLEMENTACIÓN DEL SISTEMA.....	216
8.1	LENGUAJES DE PROGRAMACIÓN.....	216
8.1.1	Java.....	216
8.1.2	XML.....	216
8.1.3	XHTML.....	216
8.1.4	CSS	217
8.1.5	Javascript	217
8.1.6	SQL.....	217
8.2	TECNOLOGÍAS UTILIZADAS	218
8.2.1	DSL	218
8.2.2	Plataforma Web.....	218
8.3	HERRAMIENTAS Y PROGRAMAS USADOS PARA EL DESARROLLO	227
8.3.1	Spring tool suite.....	227
8.3.2	Eclipse	227
8.3.3	Tomcat.....	227
8.3.4	MySQL.....	227
8.3.5	Protegé.....	228
8.4	CREACIÓN DEL SISTEMA	229
8.4.1	Problemas Encontrados.....	229
8.4.2	. Descripción Detallada de las Clases.....	229
CAPÍTULO 9.	DESARROLLO DE LAS PRUEBAS.....	230
9.1	PRUEBAS UNITARIAS	230
9.2	PRUEBAS DE USABILIDAD	230
9.2.1	Análisis de los resultados, cambios efectuados y conclusiones	236
9.3	PRUEBAS DE INTEGRACIÓN Y DEL SISTEMA	238
9.4	PRUEBAS DE RENDIMIENTO	251
CAPÍTULO 10.	MANUALES DEL SISTEMA	252
10.1	MANUAL DE INSTALACIÓN Y EJECUCIÓN	252
10.1.1	GMF.....	252
10.1.2	Plataforma Web.....	253
10.2	MANUAL DE USUARIO	260
10.2.1	Manual de usuario de la aplicación web.....	260
10.2.2	Manual de usuario GMF.....	304
10.3	MANUAL DEL PROGRAMADOR.....	309
10.3.1	GMF.....	309
10.3.2	Aplicación web	309
CAPÍTULO 11.	CONCLUSIONES Y AMPLIACIONES	314
11.1	CONCLUSIONES	314
11.2	AMPLIACIONES	314
CAPÍTULO 12.	PRESUPUESTO	316

12.1	MEDICIONES	316
12.2	CUADRO DE PRECIOS	316
12.3	TOTALES	316
CAPÍTULO 13.	REFERENCIAS BIBLIOGRÁFICAS	319
CAPÍTULO 14.	APÉNDICES	322
14.1	GLOSARIO Y DICCIONARIO DE DATOS	322
14.2	CONTENIDO ENTREGADO EN EL CD-ROM	324
14.3	ÍNDICE ALFABÉTICO.....	325
14.4	ANEXO I: PLATAFORMAS ANALIZADAS.....	326
14.4.1	MAGENTO.....	326
14.4.2	Arquitectura de Magento.....	329
14.4.3	Funcionalidades de Magento	333
14.4.4	Sistemas de plantillas.....	334
14.5	PRESTASHOP	336
14.5.1	Instalación	337
14.5.2	Sistema de directorios.....	339
14.5.3	Arquitectura del sistema	341
14.5.4	Factores Claves:	341
14.5.5	Listado de Funcionalidades Completas de Prestashop	342
14.6	OSCOMMERCE.....	348
14.6.1	Instalación	349
14.6.2	Factores Clave:.....	351
14.6.3	Listado de Funcionalidades Completas de Prestashop	351
14.7	MAPAS DE CONOCIMIENTO	354
14.7.1	Prestashop.....	357
14.7.2	OsCommerce.....	358
14.8	ANEXO II :MODELOS E-R DE LAS PLATAFORMAS CON LAS QUE SE INTEROPERA.....	359
14.8.1	Magento.....	359
14.8.2	Prestashop.....	360
14.9	ANEXO III: DESCRIPCIÓN DETALLADA DE LAS CLASES	361

Índice de Figuras

Ilustración 1 Software más popular de comercio electrónico	25
Ilustración 2 Cuota de mercado de comercio electrónico	25
Ilustración 3 Metodología creación de Ontologías Protégé	28
Ilustración 4 -Alcance de la ontología	28
Ilustración 5 Definición de conceptos para la ontología	30
Ilustración 6 Clases ontología	30
Ilustración 7 Email Tienda	31
Ilustración 8 Miembros de la clase Email Tienda	31
Ilustración 9 Diagrama I	32
Ilustración 10 Diagrama II	32
Ilustración 11 Diagrama III	32
Ilustración 12 Diagrama IV	33
Ilustración 13 Diagrama V	33
Ilustración 14 Diagrama VI	34
Ilustración 15 Diagrama VII	34
Ilustración 16 Diagrama VIII	35
Ilustración 17 Diagrama IX	35
Ilustración 18 Diagrama X	35
Ilustración 19 Diagrama XI	36
Ilustración 20 Diagrama XII	36
Ilustración 21 Diagrama XIII	37
Ilustración 22 Diagrama XIV	37
Ilustración 23 Diagrama XV	37
Ilustración 24 Diagrama XVI	38
Ilustración 25 Diagrama XVII	38
Ilustración 26 Diagrama XVIII	39
Ilustración 27 Diagrama XIX	39
Ilustración 28 Diagrama XX	40
Ilustración 29 DSL tunos commerce	41
Ilustración 30 Metamodelo Inicial	42
Ilustración 31 Actividades de comercio electrónico	44
Ilustración 32: Niveles de la Interoperabilidad	45
Ilustración 33 Modelo, metamodelo t meta-metamodelo	50
Ilustración 34 Modelos generados en MDA.	53
Ilustración 35 Modelo Ecore con sus relaciones, atributos y operaciones	54
Ilustración 36 GMF Overview (Foundation, 2010b)	55
Ilustración 37 mapa de conocimiento tunos commerce	57
Ilustración 38 metamodelo inicial	58
Ilustración 39 Metamodelo final para la gestión del catálogo de productos	59
Ilustración 40 Arquitectura global del sistema	60
Ilustración 41 Diagrama de Gantt	66
Ilustración 42 Diagrama general de casos de uso del usuario web	72
Ilustración 43 Diagrama general casos de uso usuario GMF	73
Ilustración 44 Subcasos de uso "Crear categoría"	75
Ilustración 45 Subcasos de uso "Crear Producto"	77
Ilustración 46 Diagrama de subcasos de uso "Crear Proveedor"	78
Ilustración 47 Caso de uso importar elementos	80

Ilustración 48 Caso de uso exportar elementos	80
Ilustración 49 Caso de uso "Generar CSV"	81
Ilustración 50 Caso de uso "Editar datos de usuario"	82
Ilustración 51 Caso de uso "Crear Categoría UML"	82
Ilustración 52 Caso de uso "Crear subcategoría"	83
Ilustración 53 Caso de uso "Crear Producto UML"	84
Ilustración 54 Caso de uso "Crear Proveedor UML"	84
Ilustración 55 Caso de uso "Relacionar un producto con una o varias categorías"	85
Ilustración 56 Caso de uso "Relacionar un producto con un proveedor"	85
Ilustración 57 Caso de uso "Generar XML"	85
Ilustración 58 Pasos para la creación del GMF	87
Ilustración 59 Diagrama de paquetes plataforma web	90
Ilustración 60 Meta-modelo	92
Ilustración 61 Diagrama de clases paquete entities	93
Ilustración 62 Diagrama de clases paquete DAO	94
Ilustración 63 Diagrama de clases paquete service	95
Ilustración 64 Diagrama de clases paquete actions	96
Ilustración 65 Diagrama de clases paquete intercommunication	97
Ilustración 66 Diagrama de robustez para el caso de uso "Registrar usuario"	109
Ilustración 67 Diagrama de robustez para el caso de uso "Configurar tienda"	109
Ilustración 68 Diagrama de robustez para el caso de uso "Importar elementos"	110
Ilustración 69 de robustez para el caso de uso "Importar XML"	111
Ilustración 70 Diagrama de robustez para el caso de uso "Exportar elementos"	112
Ilustración 71 Diagrama de robustez para el caso de uso "Generar CSV"	113
Ilustración 72 Diagrama de robustez para el caso de uso "Generar XML"	113
Ilustración 73 Pantalla inicio de sesión	115
Ilustración 74 Pantalla de registro	116
Ilustración 75 Pantalla de intro	117
Ilustración 76 Pantalla menú categorías	118
Ilustración 77 Crear categoría	119
Ilustración 78 Seleccionar categoría padre	119
Ilustración 79 modificar Categoría	120
Ilustración 80 Eliminar Categoría	120
Ilustración 81 Ver detalle de categoría	121
Ilustración 82 Exportar categoría	121
Ilustración 83 Pantalla Crear Producto	122
Ilustración 84 Pantalla seleccionar una o varias categorías	123
Ilustración 85 Pantalla seleccionar proveedor	123
Ilustración 86 Pantalla Modificar producto	124
Ilustración 87 Pantalla eliminar producto	125
Ilustración 88 Pantalla de ver detalle de producto	126
Ilustración 89 Pantalla Exportar producto	126
Ilustración 90 Pantalla crear proveedor	127
Ilustración 91 Pantalla modificar proveedor	127
Ilustración 92 Pantalla eliminar proveedor	128
Ilustración 93 Pantalla ver detalle proveedor	128
Ilustración 94 Pantalla de tiendas	129
Ilustración 95 Pantalla de crear tienda	129
Ilustración 96 Pantalla modificar tienda	130
Ilustración 97 Pantalla eliminar tienda	130
Ilustración 98 Pantalla ver detalle tienda	131

Ilustración 99 Pantalla importar elementos.....	131
Ilustración 100 Pantalla parseador XML	132
Ilustración 101 Pantalla datos personales	132
Ilustración 102 Pantalla modificar datos personales	133
Ilustración 103 Pantalla GMF	133
Ilustración 104 Diagrama de navegabilidad aplicación Web	135
Ilustración 105 Diagrama de navegabilidad GMF.....	136
Ilustración 106 Arquitectura del sistema	149
Ilustración 107 Arquitectura de la plataforma web	150
Ilustración 108 Diagrama de paquetes.....	151
Ilustración 109 Diagrama de despliegue	154
Ilustración 110 Diagrama de clases del paquete model del GMF en la fase de diseño	156
Ilustración 111 Diagrama de clases del paquete model.impl	157
Ilustración 112 Diagrama de clases del paquete entities en la fase de diseño	159
Ilustración 113 Diagrama de clases del paquete DAO en la fase de diseño	160
Ilustración 114 Diagrama de clases del paquete Service en la fase de diseño	161
Ilustración 115 Diagrama de clases del paquete Service.impl en la fase de diseño	162
Ilustración 116 Diagrama de clases del paquete actions en la fase de diseño	163
Ilustración 117 Diagrama de clases del paquete forms en la fase de diseño	164
Ilustración 118 Diagrama de clases del paquete intercommunications en la fase de diseño	165
Ilustración 119 Diagrama de clases del paquete validators en la fase de diseño	166
Ilustración 120 Diagrama de clases del paquete utils en la fase de diseño	167
Ilustración 121 Diagramas de interacción: registro de usuarios	168
Ilustración 122 Diagramas de interacción: configurar tienda	170
Ilustración 123 Diagramas de interacción: importar proveedores.....	172
Ilustración 124 Diagramas de interacción: exportar proveedores	174
Ilustración 125 Diagramas de interacción: generar CSV.....	176
Ilustración 126 Diagramas de interacción: generar XML.....	177
Ilustración 127 SGBD.....	178
Ilustración 128 Diagrama de entidad relación	179
Ilustración 129 Pantalla inicio de sesión	180
Ilustración 130 Pantalla de registro	181
Ilustración 131 Pantalla de intro	181
Ilustración 132 Pantalla de Categorías	182
Ilustración 133 Pantalla crear categoría.....	183
Ilustración 134 Pantalla seleccionar categoría padre.....	183
Ilustración 135 Modificar categoría	184
Ilustración 136 Eliminar Categoría	184
Ilustración 137 Ver Detalle Categoría	185
Ilustración 138 Exportar Categoría.....	185
Ilustración 139 Pantalla de productos.....	186
Ilustración 140 Pantalla crear producto	187
Ilustración 141 Pantalla Seleccionar Categorías.....	188
Ilustración 142 Pantalla Seleccionar Proveedor	188
Ilustración 143 Pantalla modificar Producto	189
Ilustración 144 Pantalla eliminar Producto	190
Ilustración 145 Pantalla de ver detalle de producto	191
Ilustración 146 Pantalla exportar producto	191
Ilustración 147 Pantalla de proveedores.....	192
Ilustración 148 Pantalla de crear proveedor	193
Ilustración 149 Pantalla modificar proveedor	193

Ilustración 150 Pantalla eliminar Proveedor.....	193
Ilustración 151 Pantalla de Ver detalle	194
Ilustración 152 Pantalla exportar proveedor	194
Ilustración 153 Pantalla de tiendas.....	195
Ilustración 154 Pantalla crear tienda	195
Ilustración 155 Pantalla modificar tienda	196
Ilustración 156 Pantalla eliminar tienda	196
Ilustración 157 Pantalla de ver detalle tienda	197
Ilustración 158 Importar elementos tienda	197
Ilustración 159 Pantalla parseador XML	198
Ilustración 160 Pantalla de datos personales	198
Ilustración 161 Pantalla modificar usuario	199
Ilustración 162 Pantalla GMF	199
Ilustración 163 GMF DashBoard	218
Ilustración 164 Esquema de capas Spring	220
Ilustración 165 Descripción de los flujos.....	221
Ilustración 166 Estructura de flujos	222
Ilustración 167 Foundation	224
Ilustración 168 Elementos de Spring Security	225
Ilustración 169 Hibernate	226
Ilustración 170 importación de proyectos	252
Ilustración 171 Ejecutar como un proyecto eclipse.....	253
Ilustración 172 descarga de apache tomcat.....	254
Ilustración 173 ejecución del tomcat	254
Ilustración 174 ejecución Tomcat	255
Ilustración 175 Apache tomcat funcionando	255
Ilustración 176 Instalación de Mysql paso I	256
Ilustración 177 Mysql instalación Paso II	256
Ilustración 178 Instalación de Mysql Paso III	257
Ilustración 179 Instalación de Mysql Paso IV	257
Ilustración 180 Instalación de Mysql Paso IV	258
Ilustración 181 Finalización de la instalación de Mysql	258
Ilustración 182 Tunosdb.....	259
Ilustración 183 Inicio de sesión.....	260
Ilustración 184 Pantalla de registro de usuarios	261
Ilustración 185 Usuario registrado correctamente	261
Ilustración 186 Errores en el formulario de registro de un nuevo usuario.....	262
Ilustración 187 Cerrar sesión	262
Ilustración 188 Pantalla de desconectar sesión correctamente.....	262
Ilustración 189 Pantalla principal (intro)	263
Ilustración 190 Menú horizontal.....	263
Ilustración 191 Volver a la pantalla principal.....	264
Ilustración 192 Sección categorías	264
Ilustración 193 Botón volver	265
Ilustración 194 Buscar categoría por nombre	266
Ilustración 195 Crear Categoría.....	266
Ilustración 196 Seleccionar categoría padre	267
Ilustración 197 Confirmación de crear una nueva categoría.....	267
Ilustración 198 Categoría creada correctamente	268
Ilustración 199 Enlace ver detalle de una categoría.....	268
Ilustración 200 Ver detalle de una categoría	269

Ilustración 201 Modificar categoría	269
Ilustración 202 Modificar Categoría	270
Ilustración 203 Confirmación de modificar categoría	270
Ilustración 204 Categoría modificada correctamente	271
Ilustración 205 Opción eliminar Categoría	271
Ilustración 206 Eliminar Categoría	271
Ilustración 207 Confirmación al eliminar categoría	272
Ilustración 208 Error al eliminar categoría	272
Ilustración 209 Exportar categoría	272
Ilustración 210 Exportar categoría	273
Ilustración 211 confirmación exportar categoría	273
Ilustración 212 Sección productos	274
Ilustración 213 Buscar producto por nombre	275
Ilustración 214 Seleccionar un proveedor como parámetro de búsqueda	276
Ilustración 215 Crear nuevo producto	277
Ilustración 216 Seleccionar proveedor	278
Ilustración 217 Seleccionar categorías	278
Ilustración 218 Confirmación de crear un nuevo producto	279
Ilustración 219 Producto creado correctamente	279
Ilustración 220 Generar archivo CSV	280
Ilustración 221 Enlace ver detalle de un producto	280
Ilustración 222 Ver detalle de un producto	281
Ilustración 223 Modificar producto	281
Ilustración 224 Modificar producto	282
Ilustración 225 Confirmación de modificar producto	283
Ilustración 226 Producto modificado correctamente	284
Ilustración 227 Opción eliminar Producto	284
Ilustración 228 Eliminar producto	284
Ilustración 229 Confirmación al eliminar producto	285
Ilustración 230 Exportar producto	285
Ilustración 231 Exportar producto	286
Ilustración 232 confirmación exportar producto	286
Ilustración 233 Sección Proveedores	287
Ilustración 234 Buscar proveedor por nombre	288
Ilustración 235 Crear nuevo proveedor	288
Ilustración 236 Confirmación de crear un nuevo proveedor	289
Ilustración 237 Proveedor creado correctamente	289
Ilustración 238 Ver detalla de proveedor	289
Ilustración 239 Ver detalle de un proveedor	290
Ilustración 240 Modificar proveedor	290
Ilustración 241 Modificar proveedor	291
Ilustración 242 Confirmación de modificar proveedor	291
Ilustración 243 Mensaje de proveedor modificad correctamente	291
Ilustración 244 Opción eliminar Proveedor	292
Ilustración 245 Eliminar proveedor	292
Ilustración 246 Confirmación al eliminar producto	292
Ilustración 247 Exportar proveedor	293
Ilustración 248 Exportar proveedor	293
Ilustración 249 confirmación exportar proveedor	293
Ilustración 250 Sección tiendas	294
Ilustración 251 Crear nueva tienda	295

Ilustración 252 Confirmación de crear una nueva tienda	296
Ilustración 253 Tienda creada correctamente	296
Ilustración 254 Ver detalla de tienda	296
Ilustración 255 Ver detalle de una tienda	297
Ilustración 256 Modificar tienda	297
Ilustración 257 Modificar tienda	298
Ilustración 258 Confirmación de modificar tienda	299
Ilustración 259 Mensaje de tienda modificada correctamente	299
Ilustración 260 Opción eliminar Tienda	299
Ilustración 261 Eliminar tienda	300
Ilustración 262 Confirmación al eliminar tienda	300
Ilustración 263 Importar elementos	300
Ilustración 264 Importar elementos II	301
Ilustración 265 confirmación importar elementos.....	301
Ilustración 266 Parseador XML	302
Ilustración ¡Error! Marcador no definido. Parseador XML II	302
Ilustración ¡Error! Marcador no definido. Parseador importando correctamente XML	302
Ilustración ¡Error! Marcador no definido. Error en el XML	303
Ilustración ¡Error! Marcador no definido. Datos personales del usuario	303
ilustración ¡Error! Marcador no definido. Modificar datos personales.....	303
Ilustración 267 Usuario modificado correctamente.....	304
Ilustración 268 Partes que forman el GMF.	305
Ilustración 269 Introducir valores a las figuras	306
Ilustración 270 Relacionar dos elementos entre si	308
Ilustración 271 Botón ejecutar.....	308
Ilustración 272 Etapas para la construcción del GMF	309
Ilustración 273 Configuraciones.....	310
Ilustración 274 Flujos	312
Ilustración 275 Instalación Magento –Paso 1	327
Ilustración 276 Instalación Magento –Paso 2.....	328
Ilustración 277 Instalación Magento –Paso 3	328
Ilustración 278 Instalación Magento –Paso 4	329
Ilustración 279 Composición de módulos Magento	331
Ilustración 280 Bloques estructurales de Magento.....	332
Ilustración 281 Bloques de contenido Magento	333
Ilustración 282 Sistema de plantillas Magento	335
Ilustración 283 Instalación Prestashop -Paso I.....	338
Ilustración 284 Instalación Prestashop -Paso 2.....	338
Ilustración 285 Instalación Prestashop -Paso 3.....	338
Ilustración XX Instalación Prestashop -Paso 4.....	339
Ilustración 286 Instalación Prestashop -Paso 5.....	339
Ilustración 287 Arquitectura de Prestashop	341
Ilustración 288 Instalación osCommerce – Paso 1	349
Ilustración 289 Instalación osCommerce – Paso 2	350
Ilustración 290 Instalación osCommerce – Paso 3	350
Ilustración 291 Instalación osCommerce – Paso 4	350
Ilustración 292-Catálogo Magento.....	355
Ilustración 293-Producto Magento	356
Ilustración 294Catálogo Prestashop	357
Ilustración 295 Catálogo OsCommerce	358
Ilustración 296Producto osCommerce	358

Ilustración 297	Modelo ER Magento.....	359
Ilustración 298	Modelo ER Prestashop	360

Capítulo 1. Memoria del Proyecto

1.1 Resumen de la Motivación, Objetivos y Alcance del Proyecto

La globalización de la economía ha llevado a los mercados a buscar nuevas herramientas para hacer negocios, introduciendo en esta nueva dinámica a las tecnologías de la información que promueven una transición de la economía tradicional a una electrónica, en dicho contexto, la incorporación de las tecnologías de la información y las comunicaciones implica un rediseño organizacional que requiere el cambio de las técnicas administrativas tradicionales por aquellas que integran conocimiento y sistemas de información

Las nuevas aplicaciones y herramientas que fomenten aumento de la conectividad y las relaciones en red también están cambiando las expectativas de los individuos como consumidores comerciales y como ciudadanos. La influencia del cambio tecnológico se extiende más allá de las aplicaciones específicas que mejoran la eficiencia del gobierno, la transparencia, o la capacidad de respuesta.

El auge del comercio electrónico según el informe de (Telefónica, 2013) Casi un 70% de la población española es internauta, y, de ellos, el 72,6% acceden a Internet a diario. Este porcentaje llega al 85% si bajamos al tramo de edad de 16 a 24 años. Internet es ya una realidad consolidada y omnipresente en nuestra sociedad. Se encuentra tan asentada en nuestro país, que no hay iniciativa en economía, política o educación que se pueda diseñar sin tenerla en cuenta. Otro dato que cada año sorprende es la fortaleza del crecimiento del comercio electrónico, con un aumento del 19,3%, que contrasta con el desarrollo del comercio tradicional “offline.

Esta situación justifica la construcción de herramientas tecnológicas que faciliten a empresas y ciudadanos la implementación, integración e interoperabilidad de soluciones de comercio electrónico. La propuesta de implementación de un Lenguaje de Dominio Especifico (DSL TUNOS COMMERCE) sustentado en un metamodelo es el objetivo central de este proyecto.

1.2 Alcance

Se requiere el desarrollo de un Lenguaje de Dominio Especifico (DSL) como La Aplicación que despliegue un contenido para procesos interoperabilidad y reutilización de la información entre plataformas de e-commerce. Esto se logrará a partir de los requisitos provistos sustentado en los principios propuestos por la Ingeniería Dirigida por Modelos (MDE), específicamente la propuesta de la Arquitectura Dirigida por Modelos (MDA), de esta forma se posibilitará minimizar el tiempo y esfuerzo necesarios para la interoperabilidad entre soluciones de comercio electrónico

1.3 Resumen de Todos los Aspectos

1.3.1 Memoria del proyecto

El comercio electrónico sus aplicaciones y propuesta de soluciones a sus actuales rectos, constituyen un tema de actualidad, en función del auge que los negocios en internet tiene en el desarrollo económico de España, Europa y el mundo en general, como mecanismo dinamizador de la economía y la creación de valor para el empresariado

1.3.2 Introducción

La introducción permitirá el primer acercamiento a todos los ámbitos del proyecto centrándose en su justificación, sus objetivos de manera más concreta que el apartado de memoria y un estudio de la situación actual en el que se identifiquen y describan sistemas similares al que se va a desarrollar.

En el primer apartado de la introducción se ofrece una justificación del proyecto DSL Tunos commerce en general y del módulo que implementará el presente trabajo además de sus características principales y los objetivos a alcanzar de manera más específica.

En el estudio de la situación actual se presentan las metodología de análisis de las plataformas de comercio electrónico intervenidas su justificación y se analizan las similitudes y diferencias, así como el desarrollo de una ontología que permito la especificación del dominio de la aplicación

1.3.3 Aspectos teóricos

Esta sección está destinada a describir de manera precisa aquellos conceptos teóricos, herramientas y tecnologías existentes tales como conceptualización del comercio electrónico, conceptos de ingeniería dirigida por modelos, lenguaje de dominio específico o DSL, Ontologías, interoperabilidad....

1.3.4 Solución propuesta

En esta sección se hace una breve descripción de cómo se ha llevado a cabo el desarrollo del metamodelo. En ella se hace un recorrido de los pasos seguidos desde el inicio del proyecto hasta la definición de la solución.

1.3.5 Planificación del proyecto y resumen de presupuestos

Esta sección se divide en dos subsecciones principales. La primera trata de ofrecer una planificación preliminar en la que se describa el plazo de ejecución del proyecto de forma que puedan fijarse las expectativas de quienes van a recibir el producto resultado del mismo.

Se ha incluido un cronograma realizado con GanttProject tipo diagrama de Gantt que incluye una planificación inicial de los plazos en los que se llevarán a cabo las diferentes tareas del proyecto. La segunda sección aporta un presupuesto aproximado del precio de los recursos a utilizar para la realización del proyecto.

1.3.6 Análisis del sistema

En la fase de análisis de un proyecto se efectúa una primera descripción precisa y clara del mismo.

En este proyecto, la sección de análisis incluye una definición más precisa del sistema, los requisitos del sistema definitivos y que deben comprobarse al finalizar el proyecto, identificación y descripción de los casos de uso, diagrama de clases con las clases más importantes que se piensan utilizar y su cometido, análisis y escenarios de los casos de uso identificados, prototipos de las pantallas de la aplicación y especificación del plan de pruebas.

1.3.7 Diseño del sistema

La fase de diseño profundiza y detalla los procesos que fueron presentados en el análisis del sistema. Durante esta fase se desarrollan, revisan y documentan las estructuras de datos a utilizar en el proyecto final, la estructura de la aplicación y los detalles a nivel de código de la misma.

En la sección de diseño del presente proyecto se incluye un diagrama de paquetes de la aplicación a nivel general, concretando cada paquete a través de su diagrama de clases. Asimismo se incluyen los diagramas de interacción y actividades para los principales casos de uso, el diseño de la interfaz a través las pantallas ya elaboradas de la aplicación, un listado detallado de todos los elementos involucrados en el diseño del DSL y la especificación del plan de pruebas que se llevará a cabo en secciones posteriores.

1.3.8 Implementación del sistema

La implementación del sistema abarca toda aquella información relativa a la generación de código desarrollado. Por tanto se incluye un breve resumen de los lenguajes de programación utilizados en el proyecto, las herramientas y programas usados para el desarrollo del mismo, los problemas encontrados durante la implementación y cómo se resolvieron y una guía detallada de las clases más importantes de la aplicación.

1.3.9 Desarrollo de las pruebas

Toda solución informática ha de probarse para asegurarse que el producto software final funciona correctamente y no presenta errores se realizaron una serie de casos de prueba de diferente ámbito que aseguren este aspecto.

En esta sección se incluyen los resultados de las pruebas unitarias, pruebas de integración y pruebas de usabilidad a usuarios, así como los cambios que se han efectuado para resolver los errores detectados en esta fase.

1.3.10 Manuales del Sistema

Esta sección contiene los manuales que permiten trabajar con la herramienta a un usuario no experto o sirven de ayuda a eventuales desarrolladores que quieran modificar o mejorarla en un futuro. Este proyecto en concreto ofrece los siguientes manuales:

- Manual de instalación y ejecución.
- Manual del usuario.
- Manual de programador.

1.3.11 Conclusiones y ampliaciones

En esta sección se exponen brevemente los resultados del desarrollo de proyecto, el cumplimiento de los objetivos, así como las recomendaciones y aportaciones que brindo en la curva de experiencia el desarrollo del proyecto.

La construcción de la aplicación prevista como solución a la problemática planteada sugiere ampliaciones no previstas en el alcance inicial del proyecto. Las mismas son presentadas en este apartado.

Capítulo 2. Introducción

2.1 Justificación del Proyecto

El escenario competitivo en el que se desenvuelven las organizaciones empresariales las obliga a generar valor como condición de sostenibilidad, la rápida evolución de las Tecnologías de la Información y las Comunicaciones (TIC's), hacen necesario incorporar al marco estratégico organizacional y tecnológico la modernización de los procesos empresariales. El auge que el comercio electrónico como tecnología ha tenido en la industria mundial y su impacto en la estructura organizacional, ha llevado al desarrollo de múltiples soluciones informáticas de manera paralela al comercio tradicional. (Tarazona-Bermudez, Pelayo G-Bustelo, & Sanjuan, 2013)

Las empresas se ven obligadas a implementar cambios en su organización para aprovechar al máximo las oportunidades que les brinda el comercio electrónico, de lo contrario, se verán forzados a la quiebra. Por lo tanto, la aplicación efectiva de la estrategia de transformación e-business resulta ser un factor crítico para obtener una ventaja competitiva sostenible (Qingfeng, Wenbo, & Lihua, 2008).

Para realizar el análisis de la información, es necesario contar con herramientas que nos ayuden a realizar la toma de decisiones respecto a un conocimiento previo concerniente al comercio electrónico (Tarazona, Gómez, & Montenegro, 2012). La definición de un DSL un prototipo de Lenguaje de dominio específico interoperable de comercio electrónico facilitara el desarrollo de software para el flujo de información de las organizaciones, dado el volumen de las transacciones y las necesidades de integración, se propicia el desarrollo de herramientas que faciliten la realización eficaz de negocios por parte de las empresas de forma que puedan representar una ventaja competitiva en sus costos, que permita aumentar la velocidad de las operaciones generando frutos en aspectos financieros y comerciales (Montenegro-marín & Cueva-lovelle, 2012a)

2.2 Objetivos del Proyecto

Implementación de un lenguaje de Dominio Especifico DSL a partir de un metamodelo de de procesos de comercio electrónico para obtener la reutilización y la interoperabilidad

Objetivos Específicos

- A partir de la propuesta de un metamodelo de procesos de comercio electrónico, especificar el modelo funcional de una aplicación que permita interoperabilidad entre plataformas de comercio electrónico.
- Definir los requisitos funcionales proponiendo un modelo conceptual y una arquitectura del prototipo con sustento en el análisis de plataformas de comercio electrónico y la ontología.
- Construir con GMF (Graphical Modeling Framework) de Eclipse un lenguaje de dominio específico (DSL) que permita generar metadatos en un lenguaje interoperable y reutilizable XML para interconexión con plataformas de comercio electrónico.
- Proponer e implementar una base de datos que soporte los procesos de comercio electrónico definidos para la interoperabilidad y la reutilización, en el contexto del prototipo.
- Desarrollar una aplicación WEB que provea una interfaz amigable para la interconexión entre el DSL propuesto y las plataforma de comercio electrónico seleccionadas
- Implementar el prototipo, desarrollar las pruebas y documentar la memoria del proyecto.

2.3 Estudio de la Situación Actual

Las soluciones de comercio electrónico disponibles actualmente adolecen en general de alto coste de implementación y reducida interoperabilidad, esto obedece a las dificultades técnicas que implica la complejidad de información, datos y formatos con la que se estructuran las plataformas actuales. Por otra parte está el interés de los proveedores de gestores de contenidos de comercio electrónico por mantener cuotas de mercado que imposibilitan a sus usuarios la migración de sus tiendas virtuales a otros proveedores. Esta situación dificulta que las empresas incursionen de manera decidida a utilizar internet como un canal de distribución formal. Son enormes y validadas las posibilidades que significan nuevos mercados en la red, aun en tiempos de crisis. Las empresas se ven obligadas a implementar cambios en su organización para aprovechar al máximo las oportunidades que les brinda el comercio electrónico, de lo contrario, se verán forzados a la quiebra. Por lo tanto, la aplicación efectiva de la estrategia de transformación de negocios electrónicos resulta ser un factor crítico para obtener una ventaja competitiva sostenible (Qingfeng et al., 2008).

Para efectos del estudio de la situación actual se resumirá en un primer momento la presentación de las plataformas complementando su análisis descrito en el Anexo II de este documento.

En lo pertinente a las herramientas disponibles para migración de datos desde y hacia algunas plataformas las trataremos en un siguiente apartado.

2.3.1 Evaluación de Plataformas de Comercio electrónico

A fin de orientar la construcción de una posible solución se revisaron las principales plataformas y gestores de contenido de comercio electrónico disponibles en el mercado (Anexos I y II). El propósito era especificar los requisitos tecnológicos, los ambientes de desarrollo y estándares de implementación de estas plataformas. Se revisaron estadísticas disponibles en estudios recientes que orientaran la selección de la misma. Se consultaron en particular dos encuestas (Tarazona & Rodríguez, 2013).

El primer estudio evaluaba los mejores sitios web de comercio electrónico, con más de un millón de visitas según lo determinado por el volumen de tráfico de datos. Este fue realizado Quantcast Corp. y la firma de investigación de marketing en Internet info Analytica Consulting Pvt. Ltd., publicado el pasado septiembre de 2012. De acuerdo a la encuesta (Robertshaw, 2012), Magento encabezó el estudio, con una participación del 23% en el mercado, como la plataforma líder de software de comercio electrónico. Zen Cart también comparte un gran crecimiento hasta el 12,9%. Después de estas dos plataformas principales están VirtueMart, osCommerce y PrestaShop con porcentajes más bajos. Sus resultados se muestran en la ilustración I.

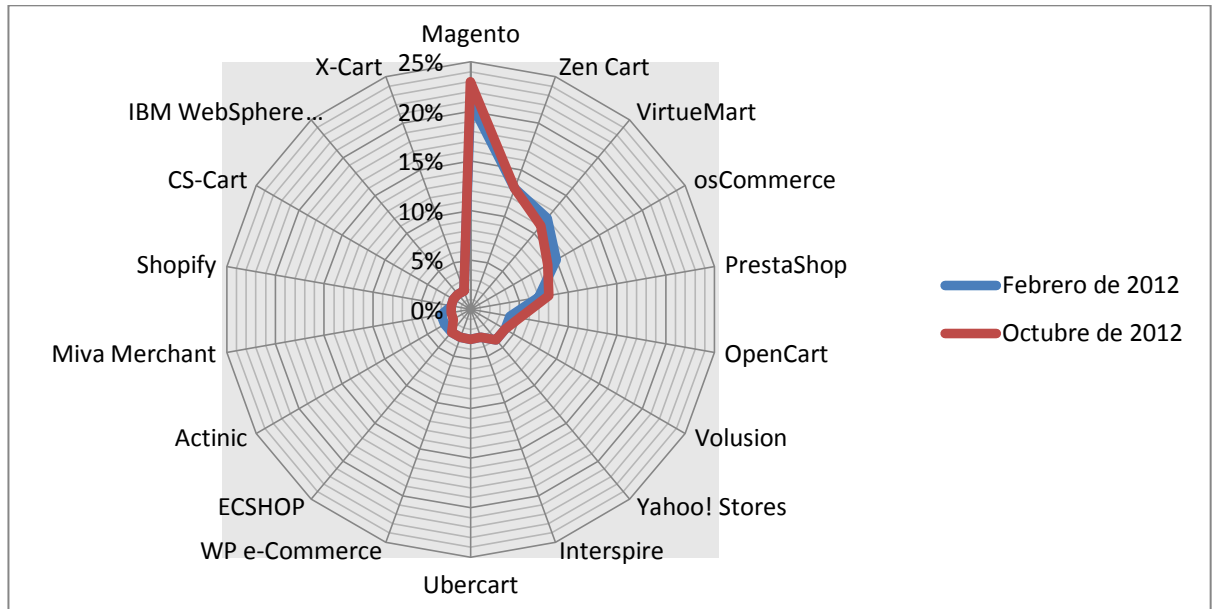


Ilustración 1 Software más popular de comercio electrónico

El segundo estudio evalúa la cuota de mercado según datos de wappalyzer.com en diciembre de 2012. Wappalyzer es una herramienta disponible en la red que descubre las tecnologías utilizadas en los sitios web. Detecta los sistemas de gestión de contenidos, tiendas web, servidores web, los marcos de JavaScript, herramientas analíticas entre otros. Los resultados se muestran en la Ilustración 2.

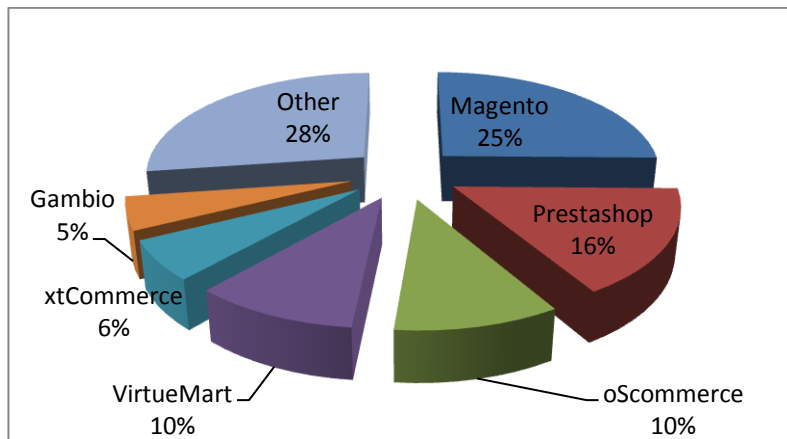


Ilustración 2 Cuota de mercado de comercio electrónico

Con fundamento en los estudios previos se seleccionaron las plataformas Magento, Prestashop y oScommerce por ser las más representativas del mercado. Se realizó un análisis sustentado en variables de evaluación con características asociadas, como se describe en la siguiente tabla

	Instalación Requisitos	Administración	Arquitectura	Soporte & Comunidad
Magento	Complicada instalación. Requiere configuraciones especiales, cache de PHP, optimizaciones de ejecución de PHP...	Administración compleja aunque bastante potente.	-Modelo de datos EAV lo que permite gran flexibilidad. -Código complejo y difícil de comprender.	Amplia comunidad, con miles de módulos, foros de soporte
Prestashop	Requisitos de instalación muy básicos y consume pocos recursos	Administración sencilla pero falta documentación, y con los constantes cambios, es difícil encontrar algunas opciones	-La programación de PHP es bastante aceptable, utiliza programación orientada a objetos y está muy bien estructurada. -La principal problemática es la continua evolución por lo que impide que se consolide una versión estable.	-Su comunidad está en crecimiento pero de momento no llega al nivel de sus dos competidores más cercanos (osCommerce y Magento)
OSCommerce	Sencillo de instalar y con pocos requisitos de instalación	Administración rápida ya que tiene muchas menos opciones que Prestashop y Magento. Además es más rápido ya que no incorpora Javascript	Difícil de programar ya que el código está mal organizado y no se utilizan plantillas.	Es el primero de ellos en aparecer por lo que tiene una enorme comunidad de seguidores.

2.3.2 Evaluación de Módulos de exportación de Datos

Las plataformas de comercio electrónico seleccionadas, se han aproximado en la construcción de plugins que se aproximan a la migración de datos. Todos ellos operan en una única solución y para un caso específico de acciones, su formato de conversión es CSV. Son complejas en su instalación y demandan conocimiento experto para el usuario así como mayores tiempos de ejecución.

Estas soluciones se comercializan, no son estables y están orientadas a hacer migraciones específicas de un conjunto de datos de las entidades del modelo relacional de cada solución específica.

Se identificaron los siguientes módulos diseñados por las empresas que respaldan las plataformas o sus comunidades de desarrollo:

Plataforma	Modulo	Descripción	Observaciones
Prestashop	Modulo Advanced CSV Export for 1.5 (Produts&Orders /Cron/CSV)	Advanced CSV Export es una herramienta para Productos & pedidos de exportación en formato CSV, dentro de la Plataforma Prestashop, compatible con PrestaShop v1.5.0.0 - v1.5.4.1	✓ Uniplataforma ✓ Tiene coste ✓ Formato CSV
Prestashop	Modulo Export Orders Pro - with customers & products	El módulo de exportación Exporta órdenes a CSV, con descarga directa y actualizar su estado a la vez.	✓ Uniplataforma ✓ Tiene coste ✓ Formato CSV
Prestashop	Modulo Export orders / customers / catalog	Muestra un resumen de todos los pedidos realizados a través de su tienda, los clientes registrados y su catálogo. Compatible con PrestaShop v1.5.0.0 - v1.5.4.1.	✓ Uniplataforma ✓ Tiene coste ✓ Formato CSV
Prestashop	Modulo Atoo-Sync GesCom - Ciel Gestion Commerciale	- Enviar artículos Ciel Gestion Commerciale en Prestashop - Enviar comandos Prestashop en Ciel Gestion Commerciale	✓ Uniplataforma ✓ Tiene coste ✓ Formato CSV
Prestashop	Módulo XML feeds	"XML Feeds" las exportaciones de módulos "PrestaShop" de categorías de productos y en XML.	✓ Uniplataforma ✓ Tiene coste ✓ Formato CSV
Magento	Módulo Lengow_Export,	Si la versión de Magento es inferior a la versión 1.5, debe utilizar la versión 1.0 de Magento Connect, se requiere activar el módulo en la sección System=>Configuration=>Advanced Estas operaciones se realizan al conectarse con una cuenta administrador Magento.	✓ Uniplataforma ✓ No especifica formato

2.3.3 Creación de una ontología de comercio electrónico

En la documentación publicada en el documento “ontology development 101” (Noy & McGuinness, 2000) el cual propone una metodología de 7 pasos como posible camino para crear una ontología como se muestra en la Ilustración 3

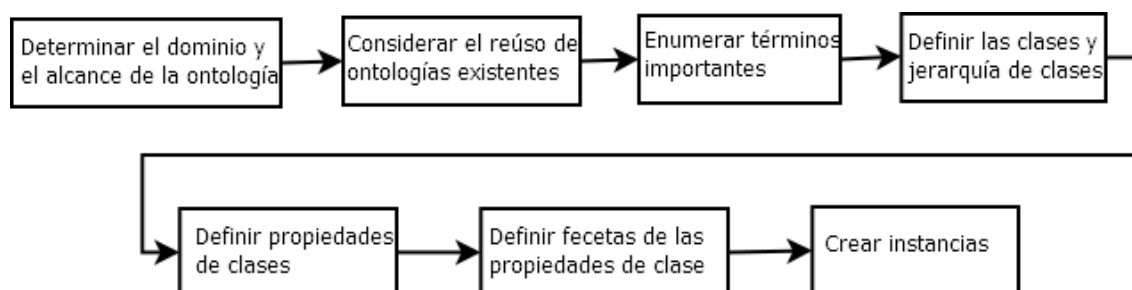


Ilustración 3 Metodología creación de Ontologías Protégé

Para el desarrollo de la ontología de comercio electrónico se siguen los pasos propuestos en esta metodología utilizando la información que ya se ha conseguido durante el desarrollo del trabajo previo y adaptando cada etapa con miras al producto final que se quiere conseguir.

Paso 1: Determinar el dominio y alcance de la ontología:

El dominio de la ontología son las plataformas de comercio electrónico, acotando las características que resultan básicas para cualquier plataforma de comercio electrónico y agregando los conceptos necesarios para interoperar módulos de clientes de distintas herramientas. Con base en lo anterior se tendrá como objetivo: Explicitar las características que debe tener una herramienta de comercio electrónico que pueda interoperar con módulos de productos/clientes desarrollados en otras plataformas de comercio electrónico:

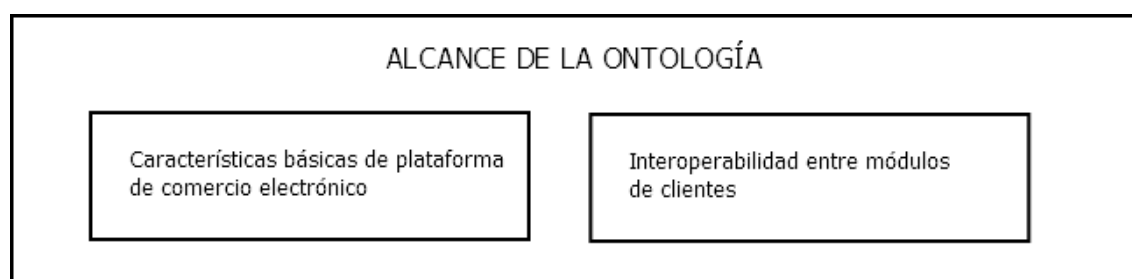


Ilustración 4 -Alcance de la ontología

Paso 2: Considerar el reúso de ontologías existentes

Para considerar el reúso de ontologías existentes se buscaron ontologías que estuviesen dentro del mismo dominio y se revisó su generalidad con miras a decidir si realmente aplican para ser consideradas como parte de la ontología que se desarrolla en este trabajo, dentro de estas se encontraron:

- **GoodRelations:** Es una ontología enfocada en el tratamiento de productos, su propósito es generar una especificación para dar un lineamiento a diferentes plataformas de cómo debe organizarse el catálogo, tiene como ventaja la fácil interoperabilidad entre los productos de las plataformas que se hayan creado o adaptado bajo este esquema, sin embargo su enfoque es de alta modularidad hacia los productos y su tratamiento y no es de alguna forma conveniente agregar una visión tan modular a nuestra ontología la cual pretende ser muy general en todos los módulos.
- **TradeOWL:** Esta ontología está enfocada en el manejo comercial y especialmente en el despacho de pedidos y hace énfasis en productos eléctricos, de energía y ambientales.
- **Rosettanet:** Desarrolla estándares universales para la cadena de suministro global, con miras a impulsar el comercio colaborativo entre distinta organizaciones, es decir, que su foco está centrado en el comercio B2B

Dentro de la revisión realizada no se encuentran ontologías que puedan reusarse para el dominio específico que se está realizando por lo cual no resulta apropiado para el desarrollo de la ontología el reuso de ontologías ya existentes.

Paso 3: Enumerar términos importantes para la ontología

De acuerdo con la revisión bibliográfica y la investigación de las plataformas existentes se proponen los siguientes términos para la creación de la ontología

Términos		
Cliente	Informe	Pago
Empresa	Venta	Producto
Boletín de noticias	Pago	Inventario
Ubicación	Catálogo	Tasa
Envío	Categoría	Tienda
Factura	Subcategoría	Nombre
Datos personales	Datos de ubicación	Nombre de producto
Nivel de reabastecimiento	Teléfono	Fax
E-mail	Autenticación	Imagen de producto

Paso 4: Definir las clases y jerarquías de clases

Durante la revisión de las plataformas se realizó un proceso top-down en el cual se revisaron los conceptos inmersos en las funcionalidades de cada una de estas plataformas, el producto de este análisis especificado en la segunda fase de la investigación es un mapa mental para cada una de las plataformas que fueron estudiadas Ver anexo (mapa de oscommerce, magento y prestashop), posterior a ello con base en la revisión bibliográfica realizada se definió dentro de todos los conceptos vistos en las plataformas aquellos que resultan básicos para una plataforma de comercio electrónico adicionando aquellos relacionados con las funcionalidades de interoperabilidad entre módulos de clientes, este proceso anteriormente mencionado se resume en el diagrama:

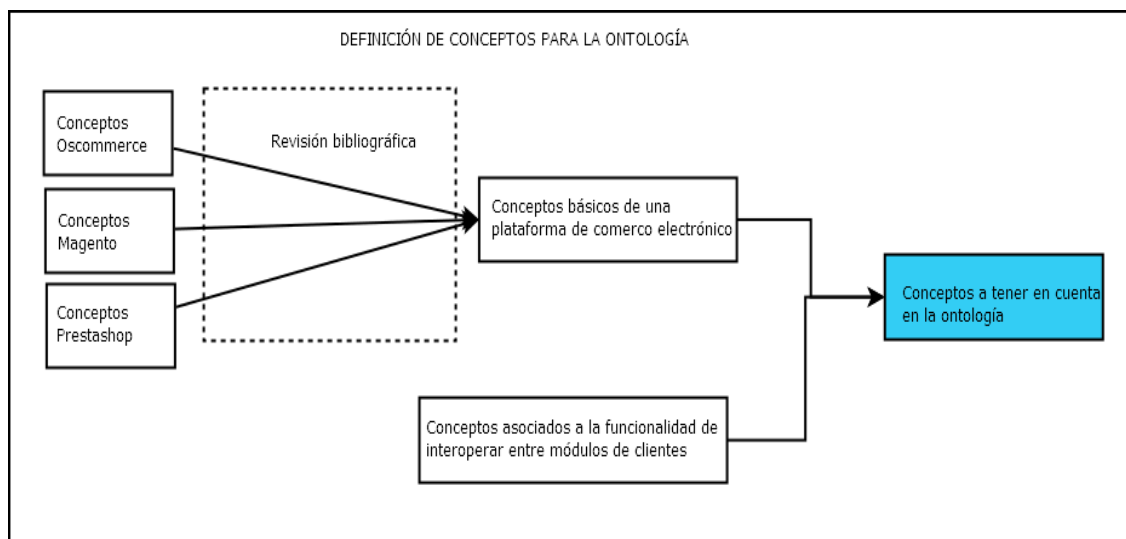


Ilustración 5 Definición de conceptos para la ontología

El producto de este trabajo se encuentra plasmado en el anexo (Ver mapa TUNOS) este anexo resume los conceptos anteriormente mencionados en un solo mapa mental y fue con base en este que se generaron las clases y la jerarquía que se muestra a continuación:



Ilustración 6 Clases ontología

Paso 5: Definir las propiedades de clases – slots

De acuerdo al mapa mental que resultó de los trabajos previos se establecen los nodos hojas como miembros de clases, es importante no tomar esta definición como miembros únicos, es

decir que los nodos hoja no son una representación de un objeto único en la vida real, en cambio agrupan un conjunto de individuos que comparten las mismas características, esta definición se toma sin perjuicio de caer en errores teniendo en cuenta que el nivel de profundidad que se establece para cada ontología depende del propósito de la misma, para ejemplificar la aclaración anterior se toma el nodo hoja “Emisor” que será representado como miembro de la clase Email Tienda de acuerdo a las siguientes figuras:



Ilustración 7 Email Tienda

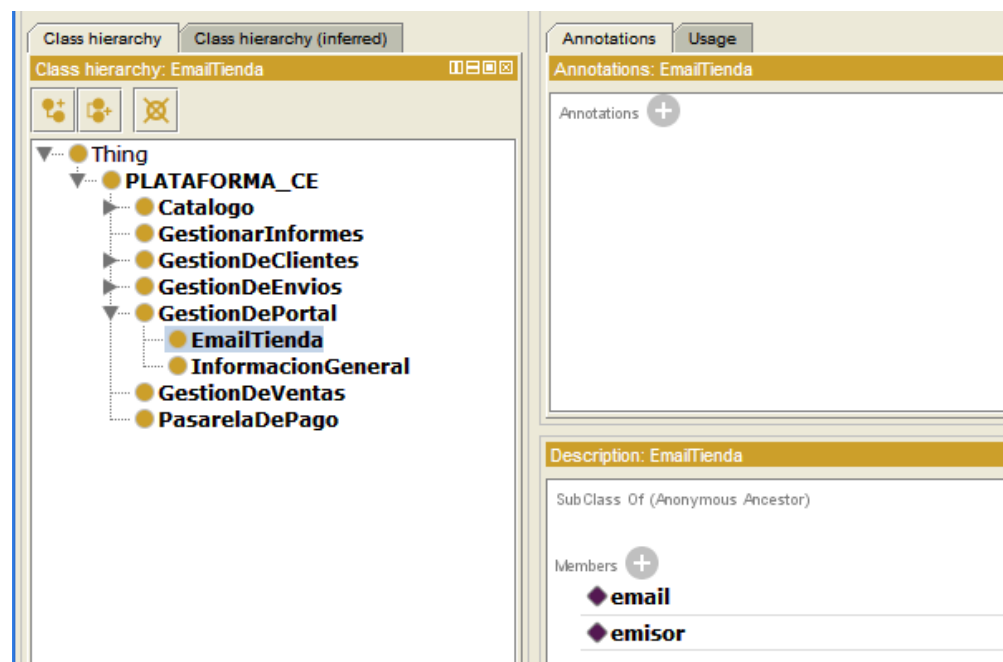


Ilustración 8 Miembros de la clase Email Tienda

Debe interpretarse entonces que emisor agrupa a todo aquel que pueda resultar ser un emisor en la vida real y no representa un emisor en un caso particular.

Para completar este paso se definieron las propiedades existentes entre cada una de la clases definidas y los miembros, para el primer nivel se estableció la propiedad “se compone de” desde la plataforma de comercio electrónico hacia cada una de su subclases del nivel inmediatamente siguiente:

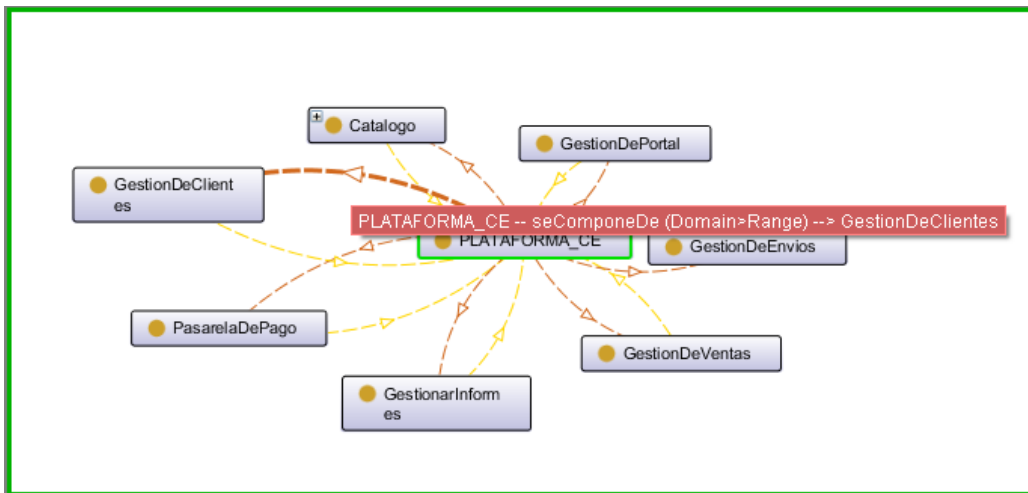


Ilustración 9 Diagrama I

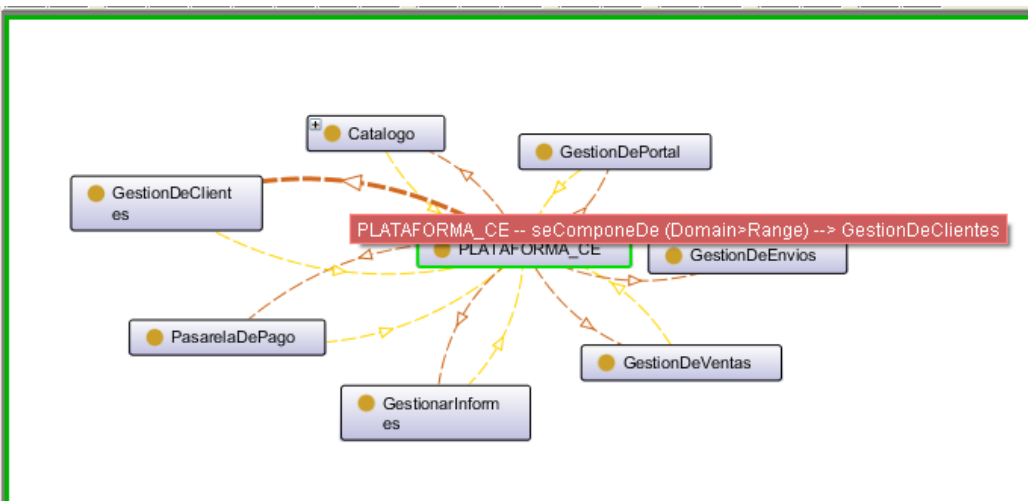


Ilustración 10 Diagrama II

Del mismo modo se establece como propiedad inversa que cada uno de los módulos “es parte de” la plataforma de comercio electrónico como se muestra en los siguientes ejemplos:

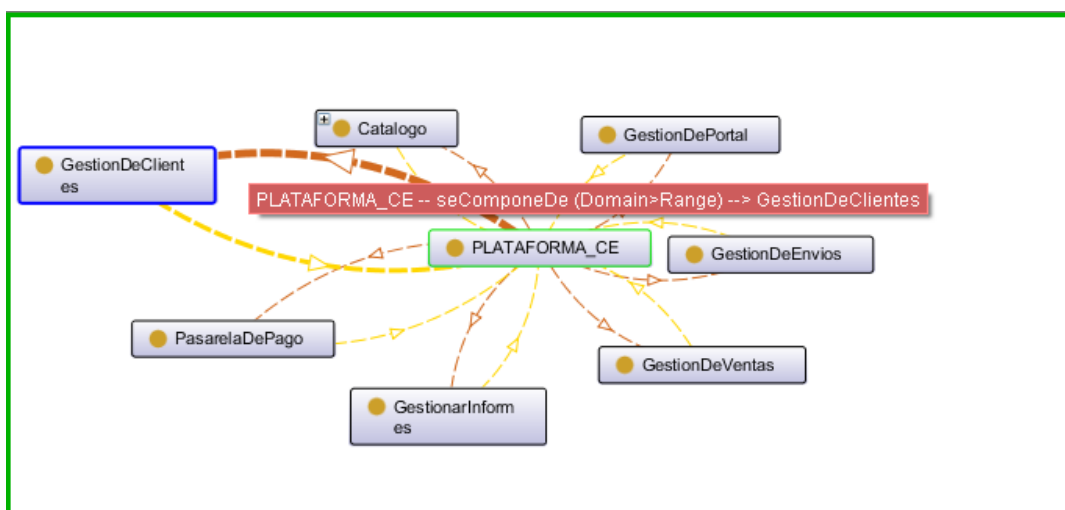


Ilustración 11 Diagrama III

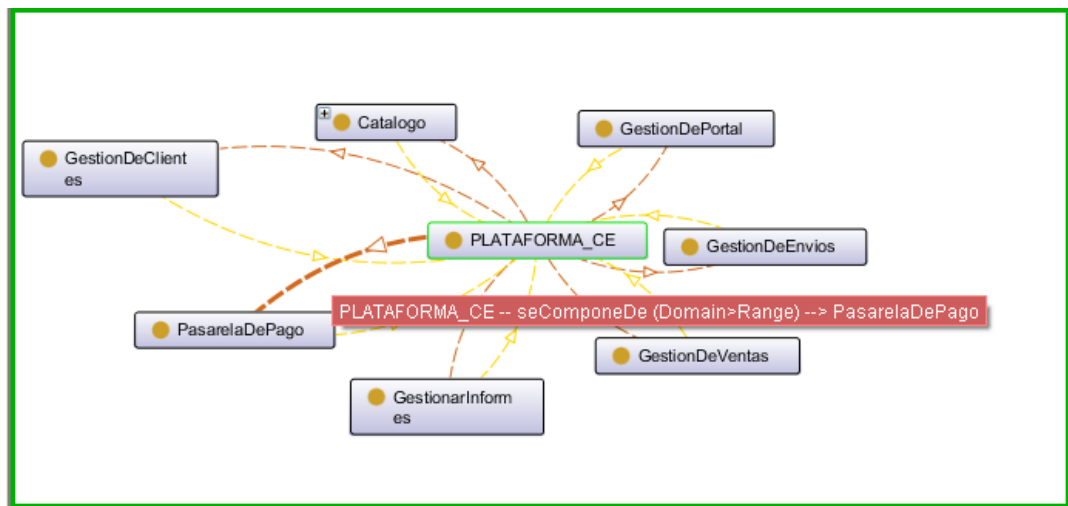


Ilustración 12 Diagrama IV

Para los niveles siguientes se definen las relaciones así:

Gestión de clientes:

Se utilizan las propiedades “administra” y “es administrado por” entre el segundo y tercer nivel como se indica:

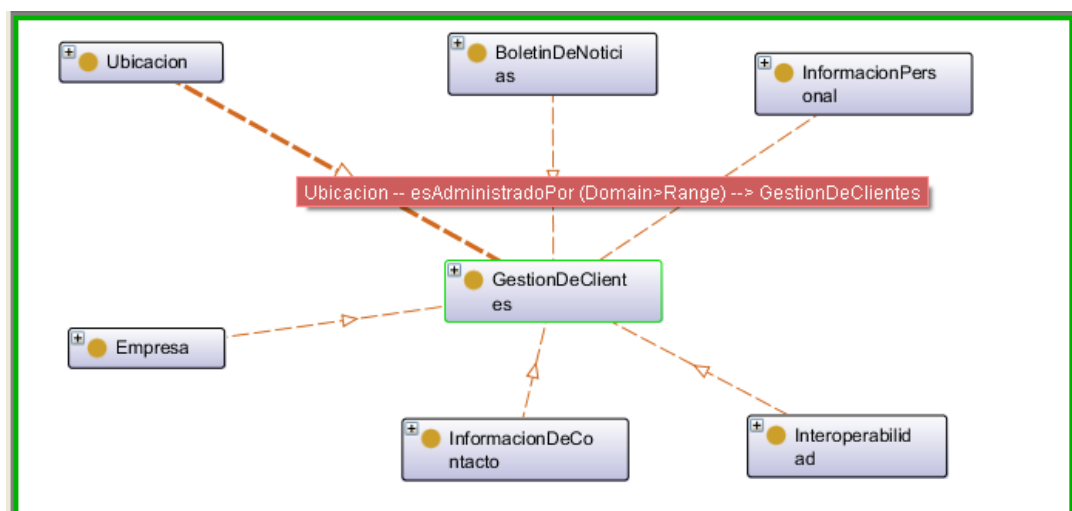


Ilustración 13 Diagrama V

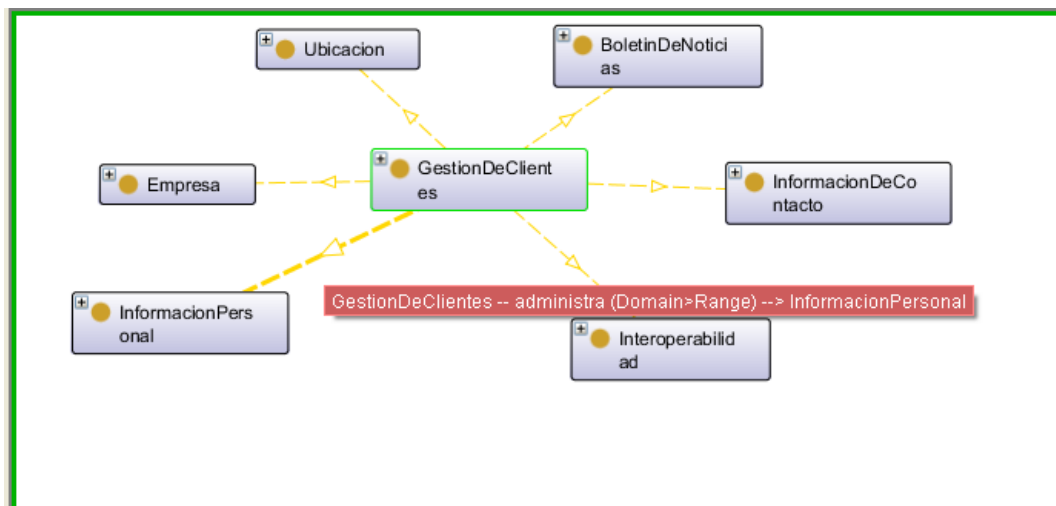


Ilustración 14 Diagrama VI

Finalmente para los miembros se deja la propiedad por defecto “has individual” quedando gestión de cliente con la siguiente configuración

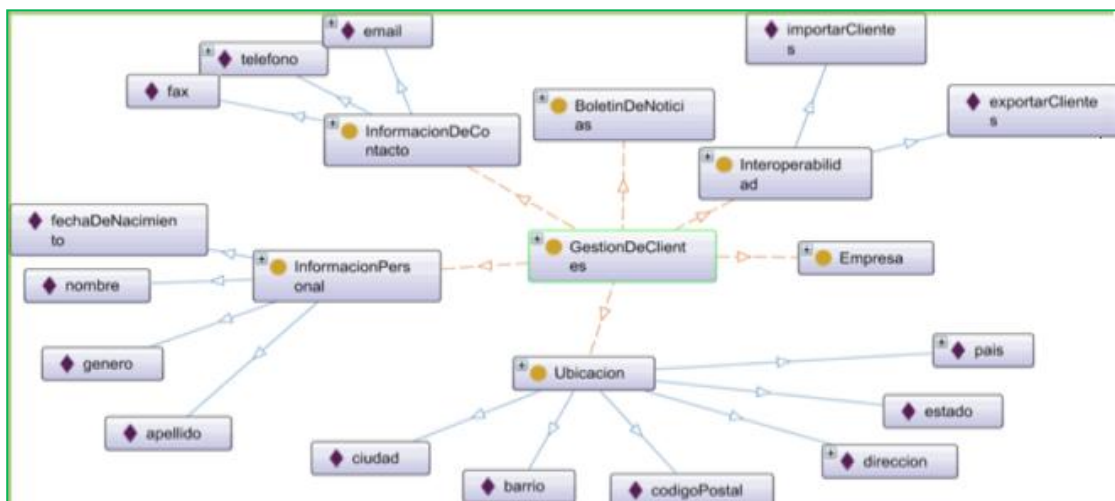


Ilustración 15 Diagrama VII

Gestión de envíos:

Se utilizan las propiedades “administra” y “es administrado por” entre el segundo y tercer nivel como se indica:

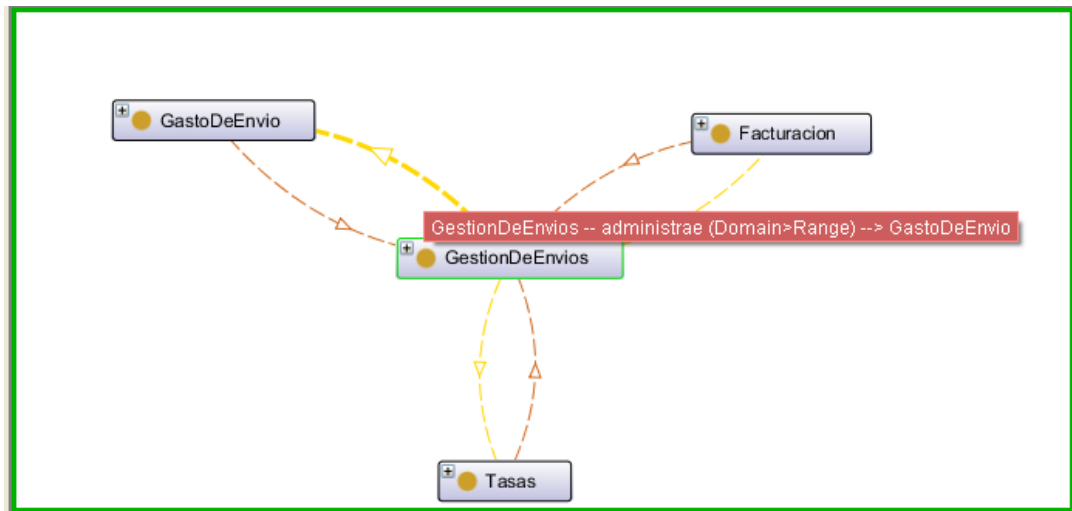


Ilustración 16 Diagrama VIII

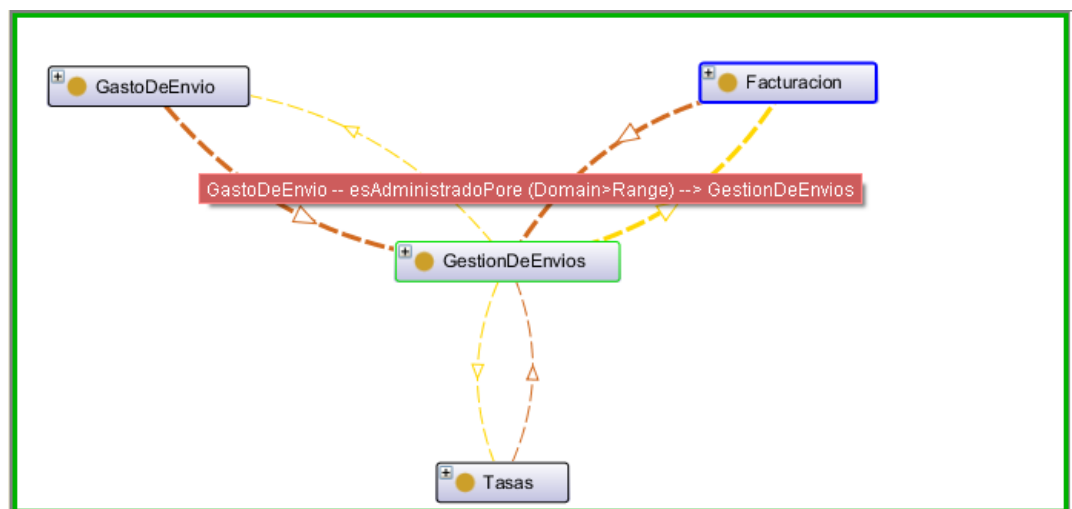


Ilustración 17 Diagrama IX

Agregando los miembros de clase, se pueden ver los niveles inferiores de gestión de envíos así

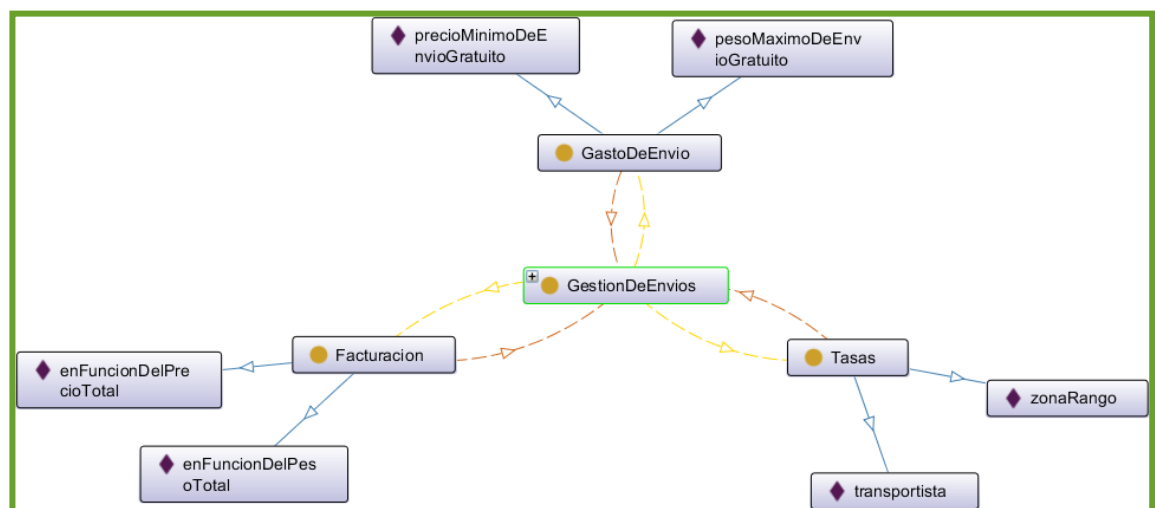


Ilustración 18 Diagrama X

Gestión de portal:

Se utilizan las propiedades “administra” y “es administrado por” entre el segundo y tercer nivel como se indica:

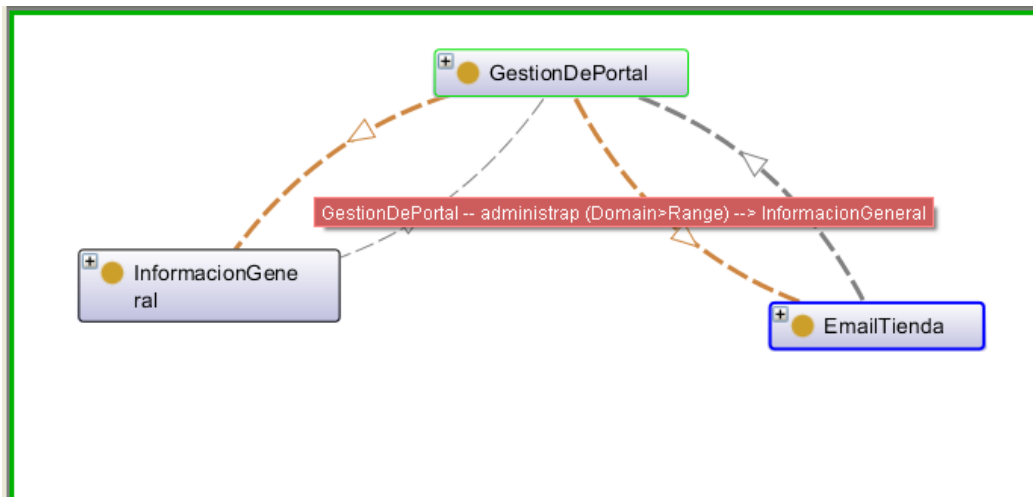


Ilustración 19 Diagrama XI

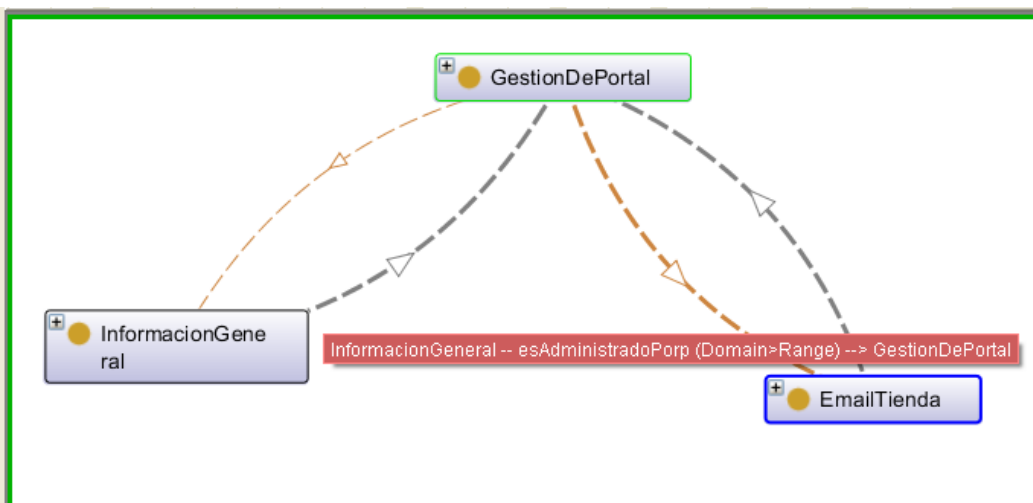


Ilustración 20 Diagrama XII

Agregando los miembros de clase, se pueden ver los niveles inferiores de gestión de envíos así

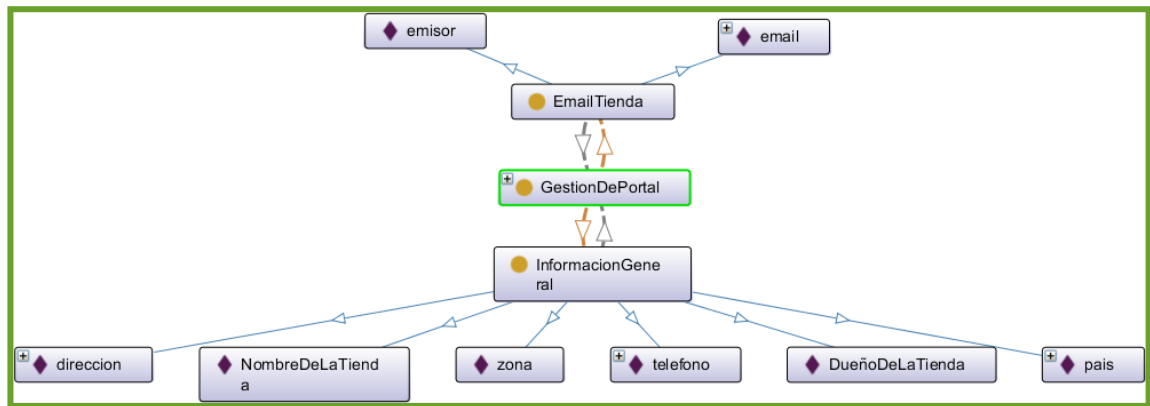


Ilustración 21 Diagrama XIII

Catálogo:

Se utilizan las propiedades “administra” y “es administrado por” entre el segundo y tercer nivel como se indica:

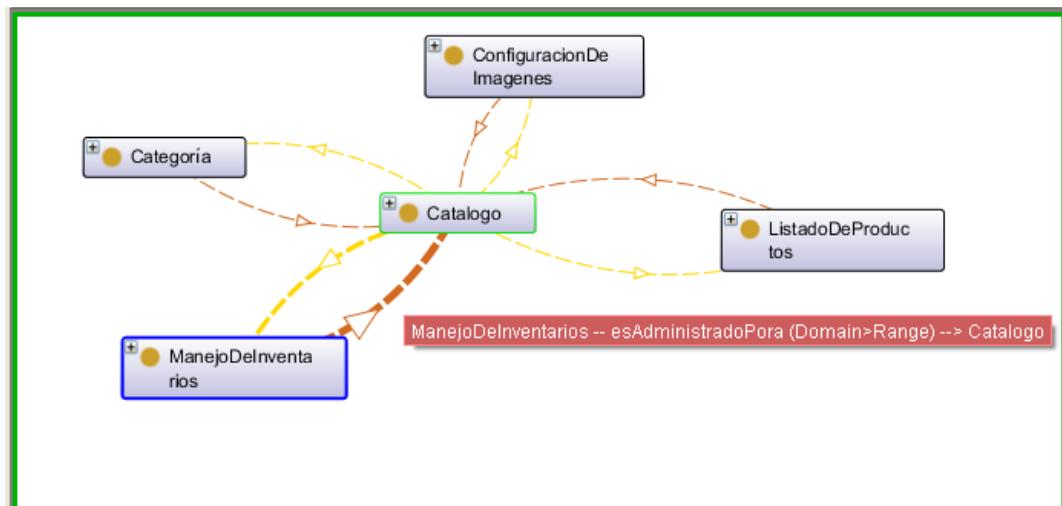


Ilustración 22 Diagrama XIV

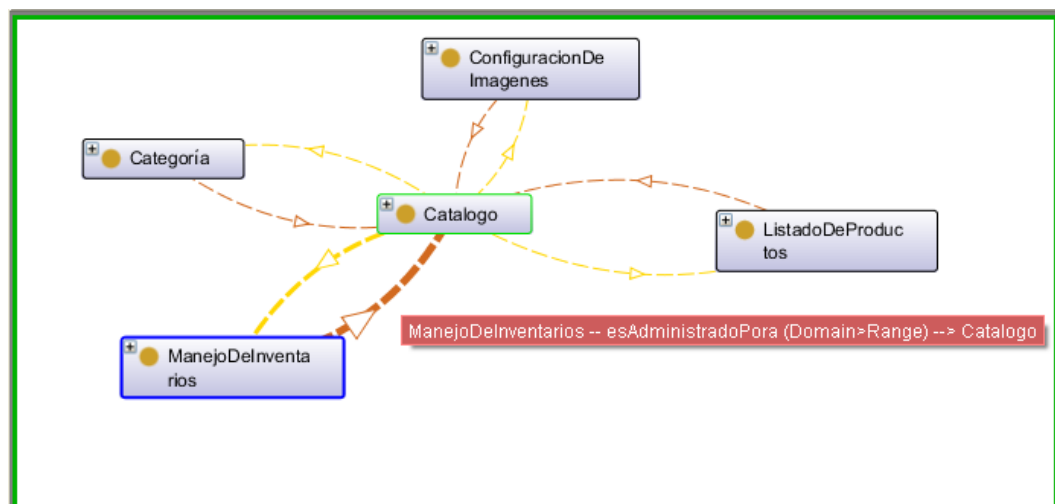


Ilustración 23 Diagrama XV

Para el concepto categoría se manejan más niveles por las características propias de producto, sub-categoría y categoría, para ello se estableció la relación compone-se compone de acuerdo a las siguientes ilustraciones:

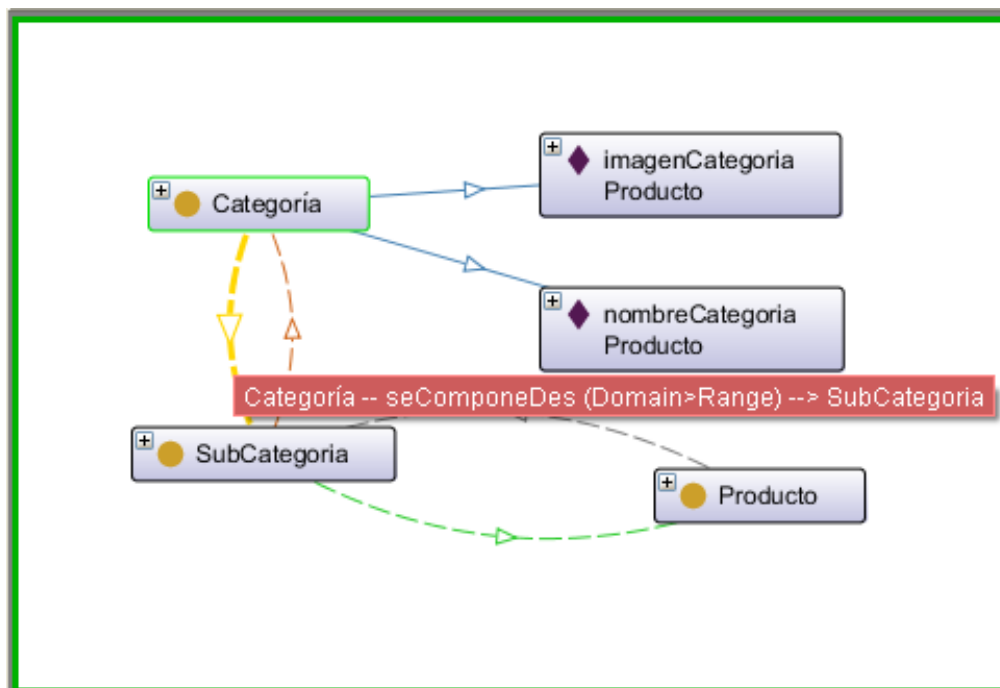


Ilustración 24 Diagrama XVI

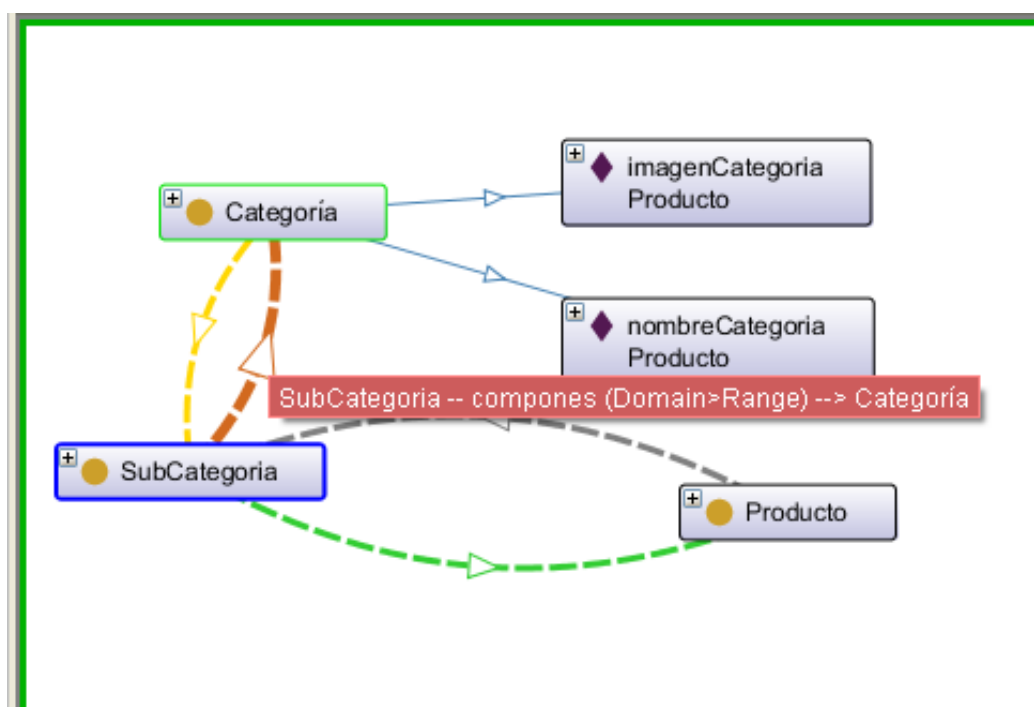


Ilustración 25 Diagrama XVII

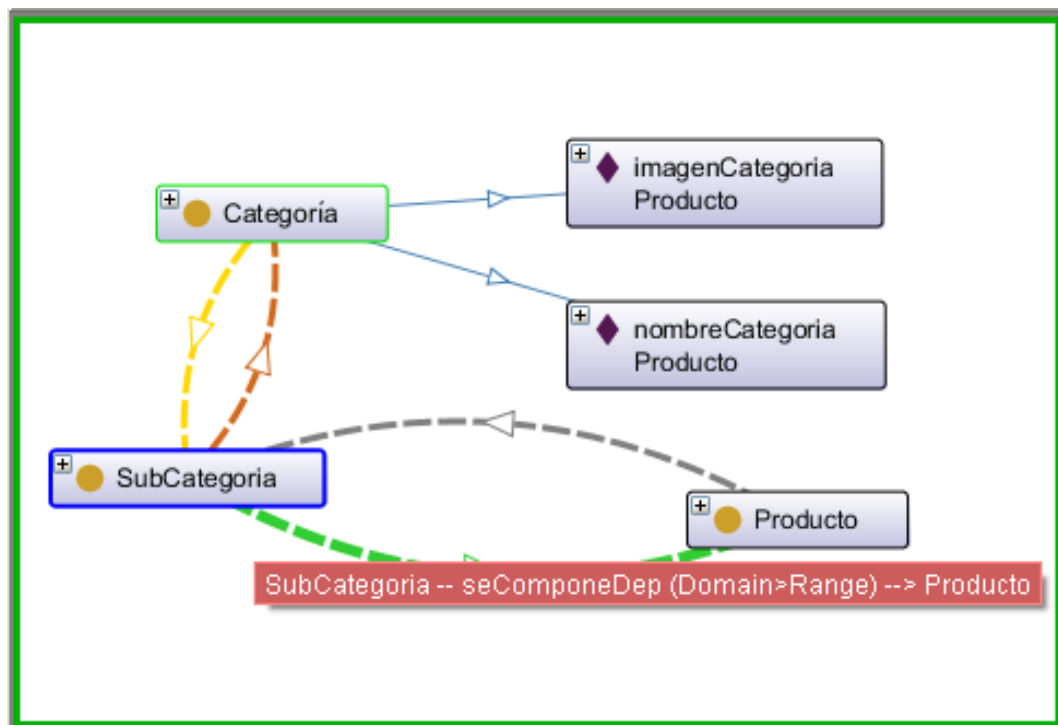


Ilustración 26 Diagrama XVIII

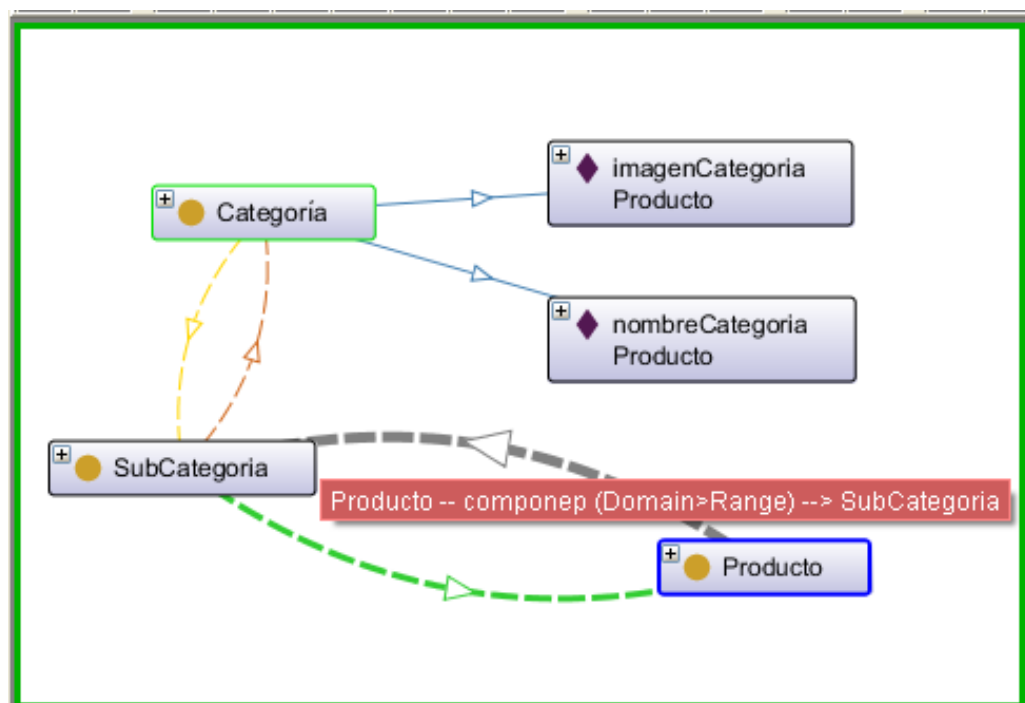


Ilustración 27 Diagrama XIX

Para los individuos de imágenes, nombre y número de ordenamiento se establecen los mismos individuos pues la representación a nivel técnico es análoga

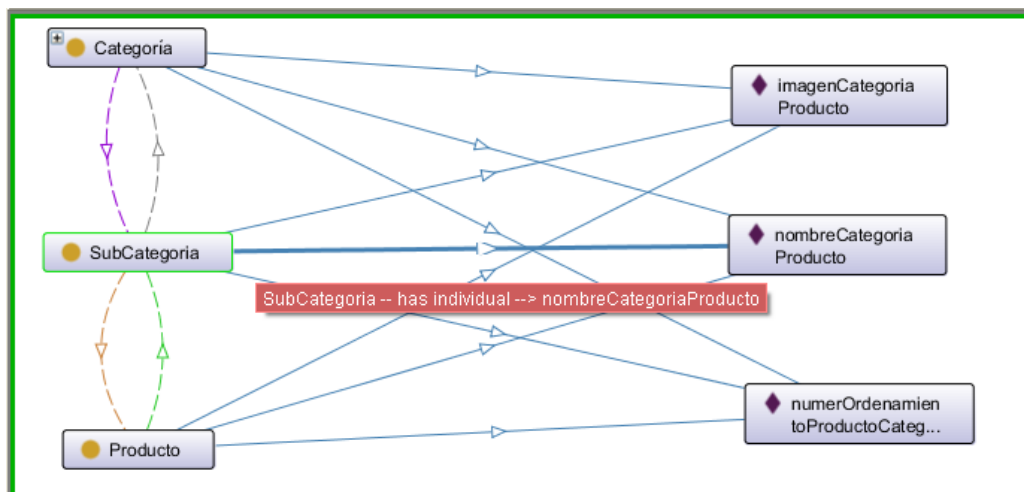


Ilustración 28 Diagrama XX

Paso 6: Definir las facetas de las propiedades de clase

Las facetas de las propiedades de clases son los lineamientos que se establecen para cada una de las propiedades que se crearon y asociaron en el paso anterior, particularmente la herramienta protégé y el documento que provee como base para la construcción de una ontología (Noy & McGuinness, 2000) sugieren cuatro tipos de facetas: Cardinalidad, tipo de valor, dominio y rango. Estas consideraciones se tendrán en cuenta al momento de desarrollar el producto de software con miras a no perder generalidad en la ontología realizada.

Paso 7: Crear instancias

El proceso de creación de instancia para este tipo de ontología será representado al final del trabajo con la creación de la herramienta misma.

2.3.4 Metamodelo de comercio electrónico

La ontología propuesta es el resultado de la caracterización de los procedimientos y la especificación semántica de la gestión de las plataformas previamente estudiadas. Se desarrolló con la herramienta Protégé, una herramienta interfaz gráfica para el procesamiento de datos cuya salida es la ontología en idioma OWL. En la Ilustración 29 se visualiza la jerarquía de clases de la ontología y se describe el modelo conceptual de la herramienta provista (DSL – TUNOS COMMERCE). (Tarazona & Rodríguez, 2013)

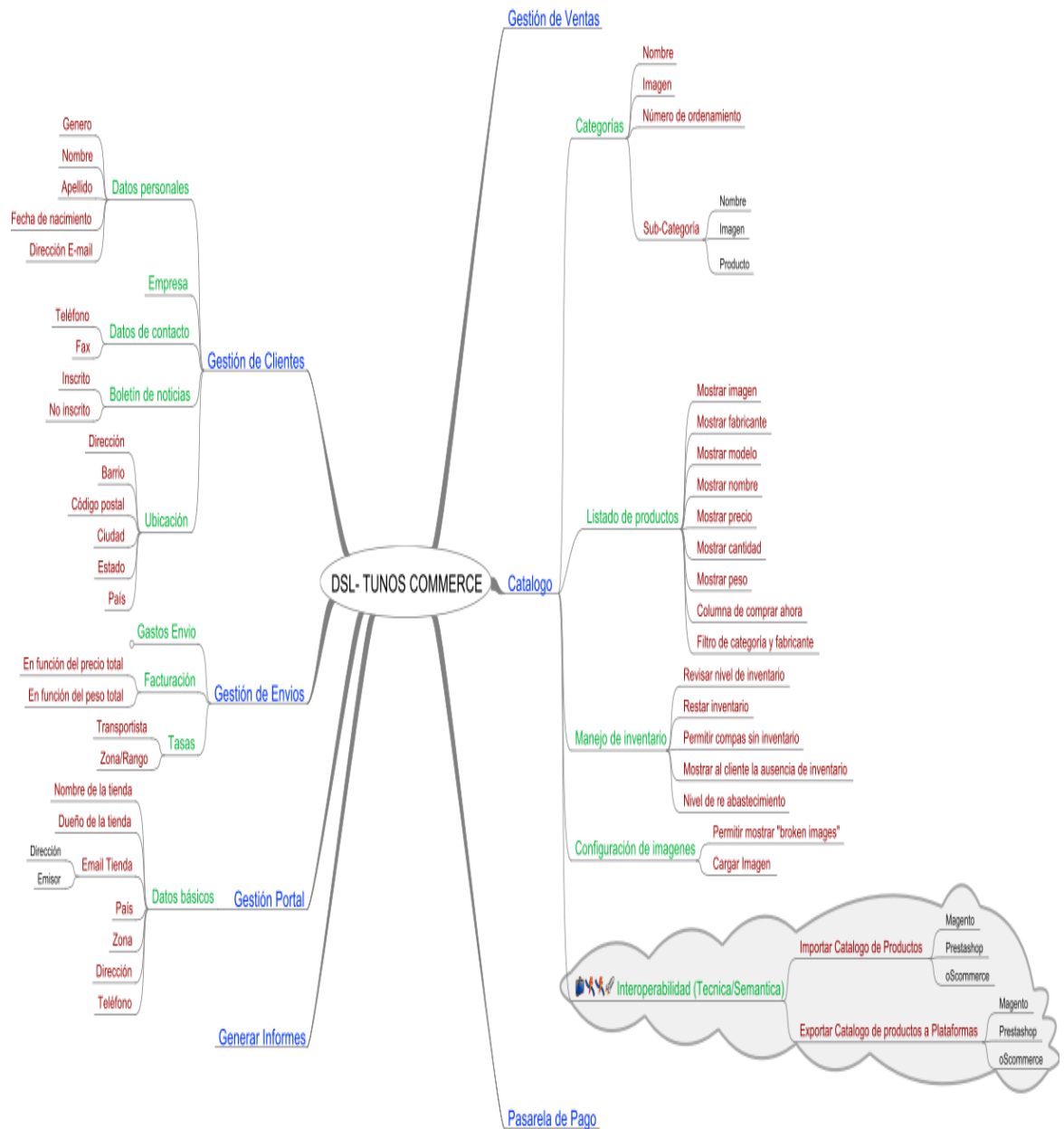


Ilustración 29 DSL tunos commerce

La herramienta generada (DSL), posibilita la interoperabilidad con las plataformas seleccionadas a partir de un generador que accede a los repositorios de información como catálogo de productos o información de clientes entre otros, permitiéndole migrar (exportar/importar) información preseleccionada facilitando su proceso de gestión de la implementación de su sitio de comercio electrónico.

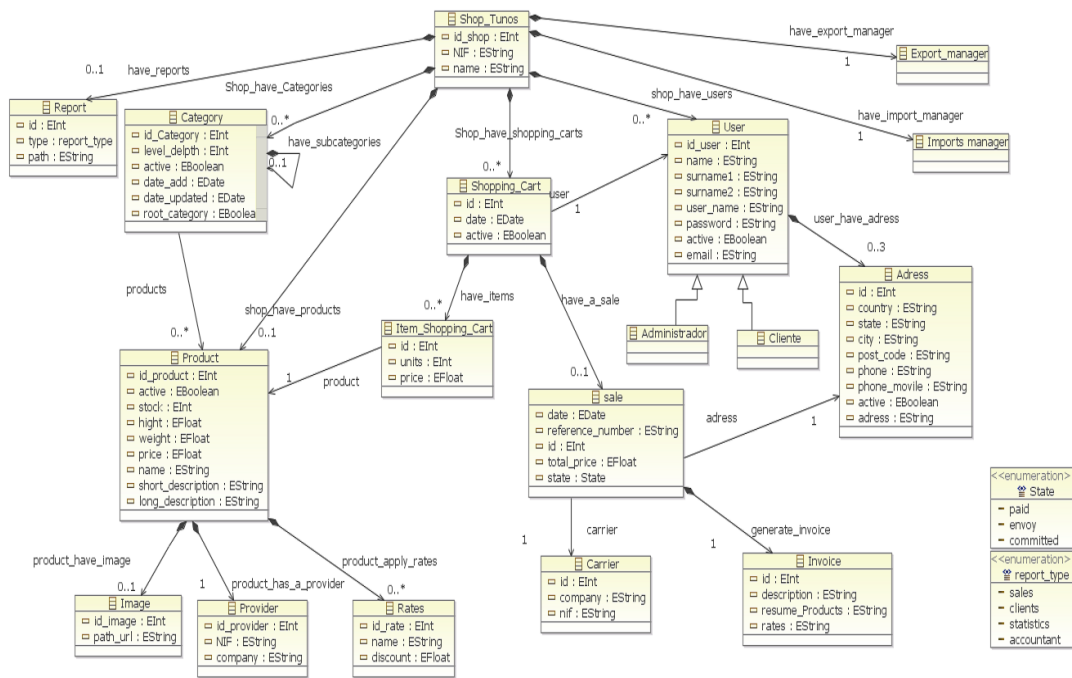


Ilustración 30 Metamodelo Inicial

Capítulo 3. Aspectos Teóricos

3.1 Comercio electrónico

El comercio electrónico (e-commerce), se define como la compra de productos de proveedores y su venta a clientes que utilizan las TIC's. Hay varios modelos de comercio electrónico, empresa a empresa (B2B) empresa a consumidor (B2C) entre empresas y gobierno electrónico (B2G). En un sentido amplio, negocios electrónicos (e-business) abarca todo tipo de colaboraciones con socios comerciales, utilizando las TIC's, y contempla los efectos legales. Una interacción de negocios, es el intercambio electrónico de documentos comerciales o un mensaje que contiene un documento de negocios vital en un proceso empresarial, este tipo de actividades desempeñan un papel clave en la colaboración. Desde finales de los años 1960, las empresas han utilizado los sistemas de información para el intercambio electrónico de datos con sus socios comerciales.(Tarazona & Rodriguez, 2013)

La integración electrónica ha dado lugar a cambios dramáticos en la definición de una empresa, con el surgimiento de empresas virtuales cuyas capacidades para ofrecer sus productos al mercado se definen en gran medida por su capacidad para organizar y mantener una red de relaciones de negocios, en lugar de por su capacidad para fabricar un producto o prestar un servicio. Para entender una empresa individual, es necesario estudiar las redes de negocios en que está inmersa (Zwass, 1996)

El comercio electrónico, tiene un conductor de las decisiones estratégicas en el dominio business to business (B2B), esto resulta un impacto favorable en internet. Existe un consenso sobre la influencia en Internet como plataforma de desarrollo de canales alternativos y/o complementarios de distribución. La proliferación del comercio electrónico puede estar relacionado con múltiples procesos de negocio.(Faroughian, Kalafatis, Ledden, Samouel, & Tsogas, 2012)

Frente a la evidencia de que la Web resultó ser un canal efectivo para el comercio, se han sugerido diversos puntos de vista sobre la forma en que las empresas deben desarrollar una estrategia de Comercio Electrónico.

Sung-Chi y otros (Chu, Leung, Hui, & Cheung, 2007), proponen cuatro áreas para desglosar la tecnología básica que proporcionan las funciones que permiten llevar a cabo las actividades de comercio electrónico, en la siguiente figura se representan las cuatro áreas.

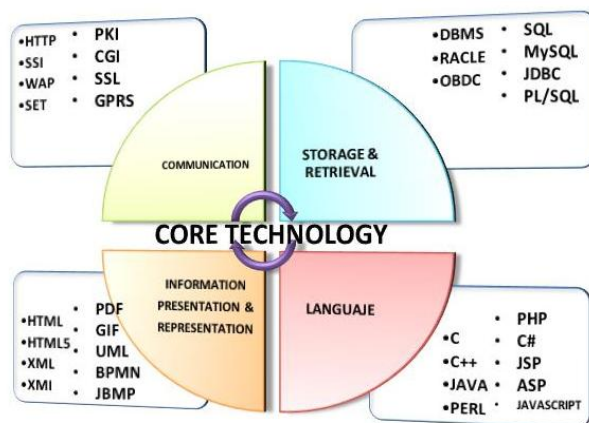


Ilustración 31 Actividades de comercio electrónico

- **Comunicación:** Para cualquiera de las partes que participan en las actividades de comercio electrónico debe establecerse un canal virtual.
- **Presentación de la información y la representación:** Este especifica la presentación de la información (su formato) y cómo deben organizarse para el intercambio (texto simple, imágenes gráficas, sonido y video, o una combinación).
- **Lenguaje:** Los pasos precisos y lógicos para la manipulación de los datos y los recursos informáticos es el componente crítico que forma los cimientos de la capacidad de inteligencia de un Sitio Web.
- **Almacenamiento y recuperación:** Esta es la estructura de datos para la grabación y recuperación de datos y la información a través de vía local y control remoto.

El análisis de modelos de comercio electrónico es un proceso complejo que implica seleccionar e implementar el modelo más adecuado que facilite la transformación empresarial. Después de que una empresa ha seleccionado un adecuado modelo de comercio electrónico, debe garantizar un desarrollo constante que le permita sobrevivir en la intensa competencia global. El modelo apropiado es un factor clave a considerar y posee un enorme potencial de transformación empresarial pero no es suficiente si no se orienta su mejoramiento (Zou & Deng, 2011).

3.2 Interoperabilidad

La IEEE define la interoperabilidad como la capacidad de dos o más sistemas o componentes para intercambiar información y para usar la información que se ha intercambiado (Institute of Electrical and Electronics Engineers, 1990). Dentro de su ámbito de aplicación, especifica un conjunto de elementos comunes tales como el vocabulario, conceptos, principios, políticas, directrices, recomendaciones, normas, especificaciones y prácticas (Wegner, 1996). La interoperabilidad debe ser considerado en tres niveles (Vernadat, 2010):

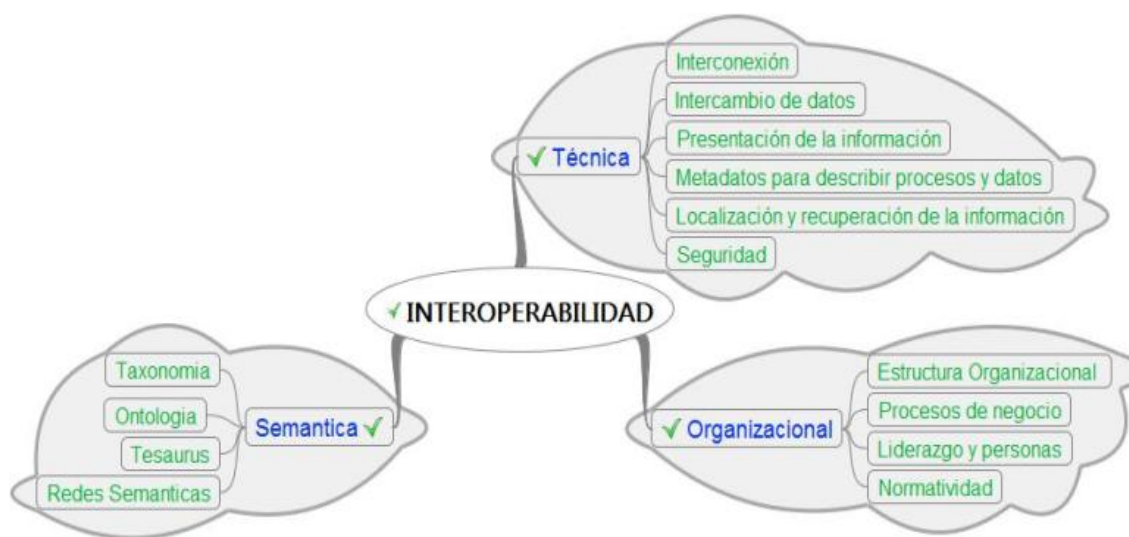


Ilustración 32: Niveles de la Interoperabilidad

Nivel organizacional: procesos coordinados en los que diferentes organizaciones a lograr una meta previamente acordada y mutuamente beneficiosa. En la práctica, implica la integración de los procesos empresariales y el intercambio de datos relacionados (SLAs, UDDI, BPEL, ebXML)

Nivel semántico: significado preciso de la información intercambiada que se conserva y entendido por todas las partes, permite a las organizaciones para procesar información de fuentes externas de manera significativa (ontologías, CPV, vocabularios, UBL, OTA, WSDL, WSMO)

Nivel técnico: la planificación de las cuestiones técnicas involucradas en la vinculación de los sistemas y servicios informáticos, Incluye aspectos como las especificaciones de interfaz, servicios de interconexión, servicios de integración de datos, la presentación y el intercambio de datos, etc. (SOA, XML, JSON).

Existen multiplicidad de plataformas que proveen soluciones de comercio electrónico, de igual forma las empresas desarrollan sus propios portales conforme a sus particulares requerimientos. En el momento de seleccionar una solución e-commerce algunas característica esenciales son la integración con las herramientas de reporting, los sistemas de recomendación, el e-CRM, el ERP, los repositorios multimedia, las herramientas de Marketing, arboles de categorías, carrito de compra, backoffice, multilinguaje, multimonedas, pasarelas de pago, seguridad, avisos por email, entre otros. Cada solución desarrolla estos procesos

atendiendo un mínimo de estándares, que limitan la reutilización y la interoperabilidad situación que da sustento a la propuesta desarrollada.

La automatización del proceso de venta que se logra vías transacciones en línea, reduce la carga global de personal de apoyo. El sitio web es el portal a través del cual la mayoría de las transacciones electrónicas se llevan a cabo, el grado de automatización del proceso de venta se verá influido por el énfasis puesto en la integración del negocio con el modelo de comercio electrónico implementado. Por lo tanto, el rendimiento de la empresa puede estar influenciado por las funcionalidades y orientaciones del e-business. (Soto & Meroño, 2009)

Cuando los datos se procesan y se comunican de forma automática, el tiempo de procesamiento se reduce. Por lo tanto, las interacciones de negocio automatizados, haciendo uso de las TIC's, puede ser más rápidas y menos propensas a errores que las interacciones de negocios manuales que utilizan correo electrónico, llamadas telefónicas o faxes. Desgraciadamente, no son fáciles de automatizar los procesos de negocio y sus interacciones. Los sistemas de información no son interoperables debido a las diferencias entre las compañías. Habría menos problemas en el comercio electrónico si todas las empresas utilizan los mismos sistemas de información, los mismos significados de los términos y los mismos modos de operación (Nurmilaakso & Kotinurmi, 2004)

Si los sistemas de información no son interoperables, la intervención humana es necesaria para preparar los datos que han de procesarse. Las diferencias en infraestructura tecnológica entre empresas son inevitables, los costos de implementación obligan a su utilización a pesar de las dificultades de interoperabilidad. Afortunadamente, las normas pueden ofrecer una manera de reducir estos costos. Las normas ponen en orden a la complejidad y la incertidumbre mediante la reducción de la variedad. La normalización de documentos de negocios y procesos de negocio promueve interoperabilidad mediante la armonización de los significados de los términos y las modalidades de las operaciones. Las Empresas pueden hacer más eficientes sus sistemas de información, siempre y cuando use el mismo tipo de documentos de negocios, procesos de negocio y optimicen sus interfaces de mensajería. (Nurmilaakso, 2008)

Las investigaciones futuras deben orientar a fondo nuevos estándares de comercio electrónico para determinar el aporte de valor a un conjunto de empresas y comprobar si las normas propuestas están a la altura de sus expectativas y sostenibilidad en su contexto. La recopilación de datos para estudios de este tipo, son una tarea formidable que contribuiría al estado del arte. (Kauremaa, Nurmilaakso, & Kari, 2010)

3.3 Ontologías

Las ontologías proveen de una comprensión compartida y consensuada del conocimiento de un dominio que puede ser comunicada entre personas y sistemas heterogéneos. Fueron desarrolladas en el área de la inteligencia artificial para facilitar el intercambio y reutilización de conocimiento (Márquez-Solís, 2007)

Los lenguajes para la descripción de ontologías son adaptaciones de XML, entre estos lenguajes están:

- SHOE (Simple HTML Ontology Extensions): lenguaje de representación del conocimiento basado en HTML.
- OWL (online Writing lab): habilita el procesado semántico de la información mediante máquinas.
- XOL (ontology Exchange Language): es un lenguaje de intercambio de ontologías.
- IFF (Information Flow Framework): es el lenguaje de marcas asociado a la información Flow Framework.
- RDF (Resource Description Language): es una recomendación del W3C basado en XML, que proporciona la tecnología para escribir metadatos que describen recursos en la web
- WSMML (Web Service Modeling Language): es un lenguaje para el modelado de servicios web semánticos basado en WSMO (web service Modeling Ontology).

Las ontologías han sido utilizadas en distintos ámbitos (comercio electrónico, ingeniería, turismo...) para aumentar la interoperabilidad entre las fuentes. Igualmente son utilizadas para mejorar la comunicación entre los tomadores de decisiones y los usuarios. También son útiles para la especificación de la semántica de los conceptos utilizados por diversas fuentes heterogéneas de una manera bien definida y sin ambigüedades (Selma et al., 2012). Las características de los lenguajes de representación orientados a la Web, tales como ontologías OWL (Ontology Web Lenguaje) colaboran en las conceptualizaciones genéricas que se pueden utilizar en varios dominios. De esta manera se facilita la creación de sistemas basados en portales Web con capacidad de automatizar procesos de negocio (García, Ruiz, López, & González, 2011).

3.4 Mapas de conocimiento

Este instrumento propuesto por Novak (Novak & Gowin, 2002), se usan como un medio para la descripción y comunicación de conceptos dentro de la teoría de asimilación, una teoría del aprendizaje que ha tenido una enorme influencia en la educación (Ausubel, Novak, & Hanesian, 1986). La teoría está basada en un modelo constructivista de los procesos cognitivos humanos. El mapa mental es la principal herramienta metodológica de la teoría de asimilación para determinar lo que el estudiante ya sabe. De acuerdo con Novak y Gowin, los mapas mentales han ayudado a personas de todas las edades a examinar los más variados campos de conocimiento en ambientes educativos (Ausubel et al., 1986)

El mapa mental es una representación gráfica de un conjunto de conceptos y sus relaciones sobre un dominio específico de conocimiento, construida de tal forma que las interrelaciones entre los conceptos son evidentes. En este esquema, los conceptos se representan como nodos rotulados y las relaciones entre conceptos como arcos rotulados conectándolos. De esta forma, los mapas conceptuales representan las relaciones significativas entre conceptos en forma de proposiciones o frases simplificadas: dos o más conceptos ligados por palabras para formar una unidad semántica. Los mapas conceptuales proveen una interfaz elegante y fácil de comprender para navegar en un sistema (Cañas, Ford, & Hayes, 1997) y de este modo facilitar su comprensión

Los mapas mentales son muy utilizados para el modelamiento en ingeniería del software trabajos como (Byrd, T. A. , Cossick, K. L., and Zmud, 1992), (Montenegro, 2011) (Montenegro-marín & Cueva-lovelle, 2012b) lo han empleado de manera satisfactoria, además es un excelente insumo a la hora de construir ontologías Para visualizar los módulos que componen a cada plataforma se optó por realizarlo mediante la representación de mapas mentales o de conocimiento, ya que es una manera formal de realizar el análisis de requerimientos en la ingeniería de software. La herramienta para la construcción de los mapas debería cumplir un requisito esencial; Debe permitir hacer mapas mentales o de conocimiento que se puedan visualizar gráficamente para su posterior análisis, la herramienta debe ser de libre distribución y multiplataforma.

3.5 MDE

Ingeniería orientada a modelos (MDE) es un paradigma de desarrollo de software que se centra en la creación y explotación de modelos de dominio (es decir, representaciones abstractas de los conocimientos y actividades que rigen un dominio de aplicación particular), más que en conceptos informáticos (o algoritmos). Este paradigma se concibió cuando el Object Management Group (OMG) desarrolló la arquitectura de diseño orientado a modelos Model-driven Architecture (MDA).

MDA propone basar el desarrollo de software en modelos especificados utilizando UML, para que, a partir de esos modelos, se realicen transformaciones que generen código u otro modelo, con características de una tecnología particular (o con menor nivel de abstracción)(Anacleto, 2006).

La idea principal del desarrollo utilizando MDA es la separación de la especificación del sistema de la implementación y los aspectos puntuales del mismo de acuerdo a la plataforma en la cual se desee realizar el desarrollo. Para este fin, el marco de trabajo MDA especifica tres modelos (Antolín, 2009):

Modelo Independiente de Computación (CIM – Computation Independent Model). Este modelo está centrado en el dominio del sistema así como en los requerimientos, detalles de la estructura y procesamiento del sistema. Los CIMs se asocian a los modelos de dominio y es recomendable que sean especificados a partir de un vocabulario común para todos los elementos (computacionales o humanos) implicados en el desarrollo del sistema.

Modelo Independiente de Plataforma (PIM - Platform Independent Model). Este modelo es el encargado de mostrar la especificación del sistema teniendo en cuenta no solo las especificaciones de funcionamiento propias del sistema -especificadas en el CIM- sino también las especificaciones para la implementación en un medio informático. El Modelo PIM representa los aspectos que no cambiarán de una plataforma a otra, de acuerdo a una tecnología o método de implantación escogido para la representación informática.

Modelo Específico de Plataforma (PSM - Platform Specific Model). Este modelo combina el modelo independiente de plataforma (PIM) con los detalles y características propias del uso de una plataforma de desarrollo. En el modelo PSM se puede observar cómo el sistema usa las herramientas de la plataforma para el cumplimiento de los objetivos trazados en la etapa de especificación inicial.

La arquitectura dirigida por modelos surge como la respuesta de la ingeniería de software a la industrialización del desarrollo de software. MDA es la propuesta de la OMG, que centra sus esfuerzos en reconocer que la interoperabilidad es fundamental y el desarrollo de modelos permite la generación de nuevos modelos que luego al ser vinculados proveerán una solución a todo un sistema. Una buena interpretación de lo que es un modelo, un metamodelo y un meta-metamodelo se encuentra en la siguiente ilustración. Un metamodelo está constituido por herramientas que permite la creación de un modelo, que es una descripción de uno o varios elementos del dominio o mundo real y finalmente el meta-metamodelo describe a esos metamodelos planteados, generando un grado de abstracción supremamente alto en el cuál coinciden todos los modelos.

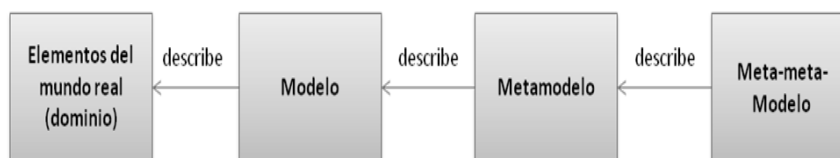


Ilustración 33 Modelo, metamodelo y meta-metamodelo

Existen cuatro espacios de modelamiento, los niveles base M0 que son los elementos del mundo real, los niveles M1 que son los programas informáticos, los niveles M2 que sería la especificación de UML, ODM, Java, C#, XML u otras y que para este caso será el metamodelo a construir, finalmente están los niveles M3 que son los de mayor abstracción.

Una propuesta de desarrollo para abordar la solución a la problemática propuesta, parte de la generación de una ontología y el posterior modelamiento de los procesos de e-commerce estandarizados, recurriendo MDA (Henderson-Sellers, 2011) (Mora et al., 2011), para lograr la transformación de los modelos propuestos de modo tal que permitan ir de lo más general (requerimientos) a lo más particular (despliegue de la solución) a través de las conversiones de dichos modelos (González et al., 2011).

3.6 Lenguajes de dominio específico (DSL)

Muchos de los problemas que aparecen durante el desarrollo del software surgen una y otra vez de forma repetitiva, además en muchas ocasiones estos problemas pertenecen a un dominio concreto. Para dar solución a estos problemas se puede utilizar un GPL (General Purpose Language) como es Java y C# o se puede recurrir a la utilización de un DSL (Domain Specific Language). Un DSL no es más que un lenguaje definido específicamente para resolver los problemas de un dominio concreto.

A partir del contexto de DSL se puede hablar de DSM (Modelado de Dominio Específico) que tiene su origen en la existencia de muchos desarrollos de software similares para un mismo dominio, una parte común y una parte variable. La parte común podría desarrollarse utilizando las técnicas de desarrollo tradicionales y la parte variable podría desarrollarse utilizando un DSL. Un ejemplo sería la realización de sistemas para la realización de encuestas, que pueden compartir un mismo motor y una misma base de datos pero que tiene que adaptarse a las diferentes encuestas. Como la única parte variable es la encuesta se podría definir un DSL para definir las.

Los DSL intercambian generalidad por expresividad en un dominio limitado. Ofrecen notaciones y construcciones orientadas a un dominio particular, presentando ganancias sustanciales en términos de expresividad y facilidad de uso en comparación con los GPL para el dominio en cuestión. Estas ganancias se corresponden en mejoras de la productividad y reducción de los costes de mantenimiento. Algunos lenguajes de dominio específico se listan a continuación:

- ✓ EBNF: Especificación de sintaxis.
- ✓ Excel: Hojas de cálculo.
- ✓ HTML: Páginas Web de hipertexto.
- ✓ Make: Construcción de software.
- ✓ Matlab: Matemáticas Computacionales.
- ✓ SQL: Consultas a bases de datos.

- ✓ VHDL: Diseño de hardware.

3.6.1 Tipos de lenguajes de dominio específico (DSL)

Se pueden distinguir tres clasificaciones para los DSL:

Desde un punto de vista de la construcción del lenguaje en:

1. Internos: Utilizan un determinado lenguaje anfitrión para darle la apariencia de otro lenguaje concreto. Un ejemplo claro son lo que actualmente se conoce como Fluent Interfaces.
2. Externos: Tiene su propia sintaxis y es necesario un parser para poder procesarlos. Un ejemplo claro de DSL externo es SQL.

Desde el punto de vista del formato del lenguaje:

1. Textuales: La mayoría de los lenguajes informáticos son textuales y están formados por un conjunto ordenado de sentencias. Un ejemplo muy conocido de DSL textual es SQL utilizado para realizar consultas a una base de datos. Una forma de crear DSLs textuales es mediante la creación de una determinada gramática (por ejemplo utilizando EBNF) y posteriormente crear o utilizar un parser para dicha gramática, para en etapas posteriores poder interpretar el DSL o generar código.
2. Gráficos: En los últimos años están ganando gran aceptación los lenguajes gráficos, podrían citarse como ejemplo UML. La creación de un lenguaje grafico es similar a la de un lenguaje textual, la única diferencia es que en lugar de usar texto para representar los conceptos, se utilizan conectores y figuras simples.

Desde el punto de vista del dominio del problema:

1. Horizontales: Los DSL horizontales son aquellos en los que el cliente que utilizará el lenguaje no pertenece a ningún dominio específico. Un ejemplo son los editores visuales de entornos de desarrollo que permiten generar interfaces de usuario automáticamente (por ejemplo Windows Forms de visual Studio).
2. Vertical: A diferencia de los DSL horizontales, el cliente que utilizará el lenguaje pertenece al mismo dominio que el lenguaje en sí. Como en el ejemplo anterior para un lenguaje de definición de encuestas, los usuarios finales serían los expertos en estadística encargados de definir dichas encuestas.

3.6.2 Partes de un lenguaje de dominio específico (DSL)

1. Los lenguajes de dominio específico son el elemento principal de cualquier solución de dominio específico. El lenguaje se define como un metamodelo, una notación específica y generalmente una herramienta que lo soporta para facilitar la usabilidad. La idea básica es no utilizar conceptos de lenguajes de programación de propósito general. En su lugar, se utilizan conceptos y reglas de dominio.
2. La sintaxis abstracta de un lenguaje especifica su estructura, es decir, las construcciones, propiedades y conectores que pueda tener dicho lenguaje. La sintaxis concreta es necesaria para especificar la notación específica con la que los usuarios del lenguaje podrán utilizarlos. Idealmente cada concepto del dominio y del lenguaje se mapearan a una representación en la notación específica. Es importante reseñar que una misma sintaxis abstracta podría tener diferentes sintaxis concretas.

3.7 Ingeniería dirigida por modelos (MDE) con eclipse

3.7.1 Meta-metamodelo ecore de eclipse

El meta-metamodelo que se empleara será Ecore, este se encuentra en el paquete org.eclipse.emf.ecore y es la especificación más alta que existe en la pirámide de los modelos (M3) de la siguiente figura, sobre ella se construirá el metamodelo del proyecto, la especificación de Ecore se puede consultar en (Eclipse Foundation, 2013)

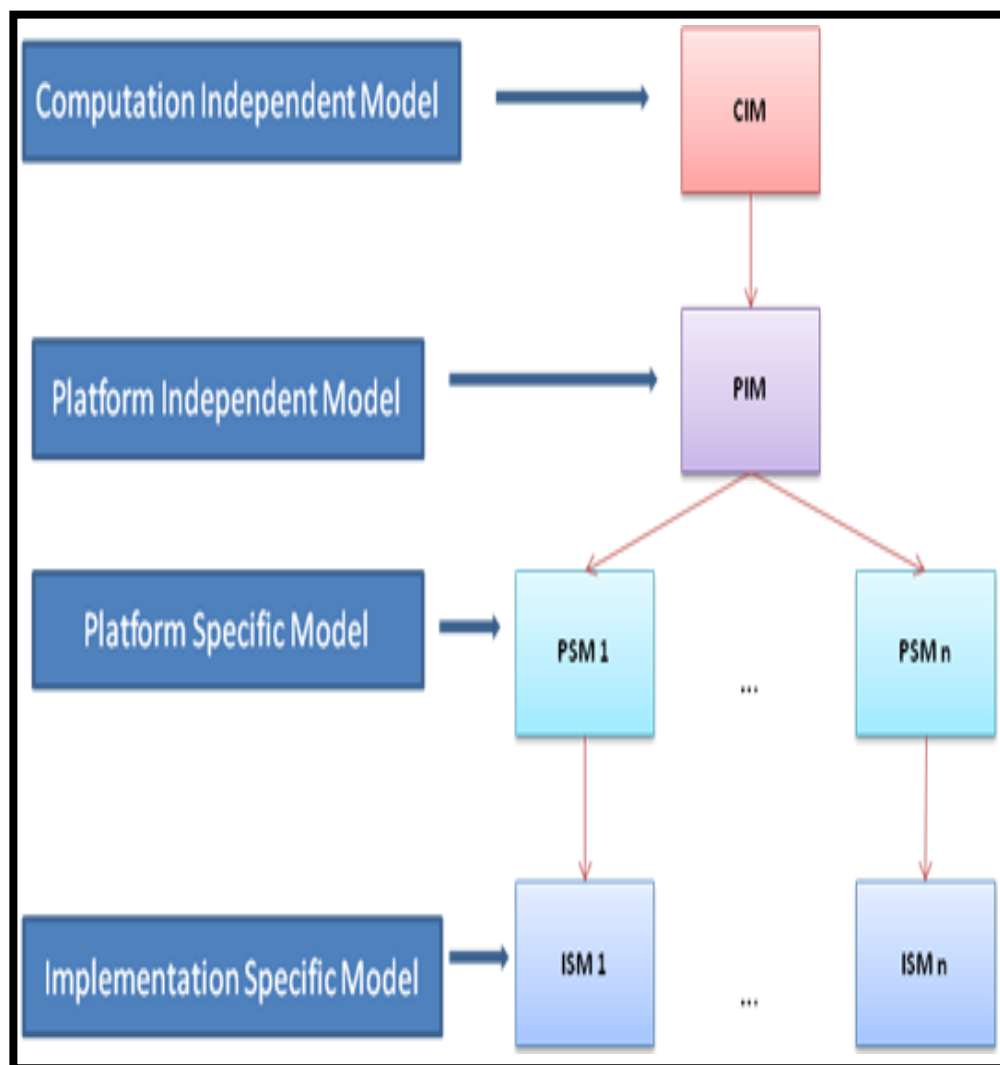


Ilustración 34 Modelos generados en MDA.

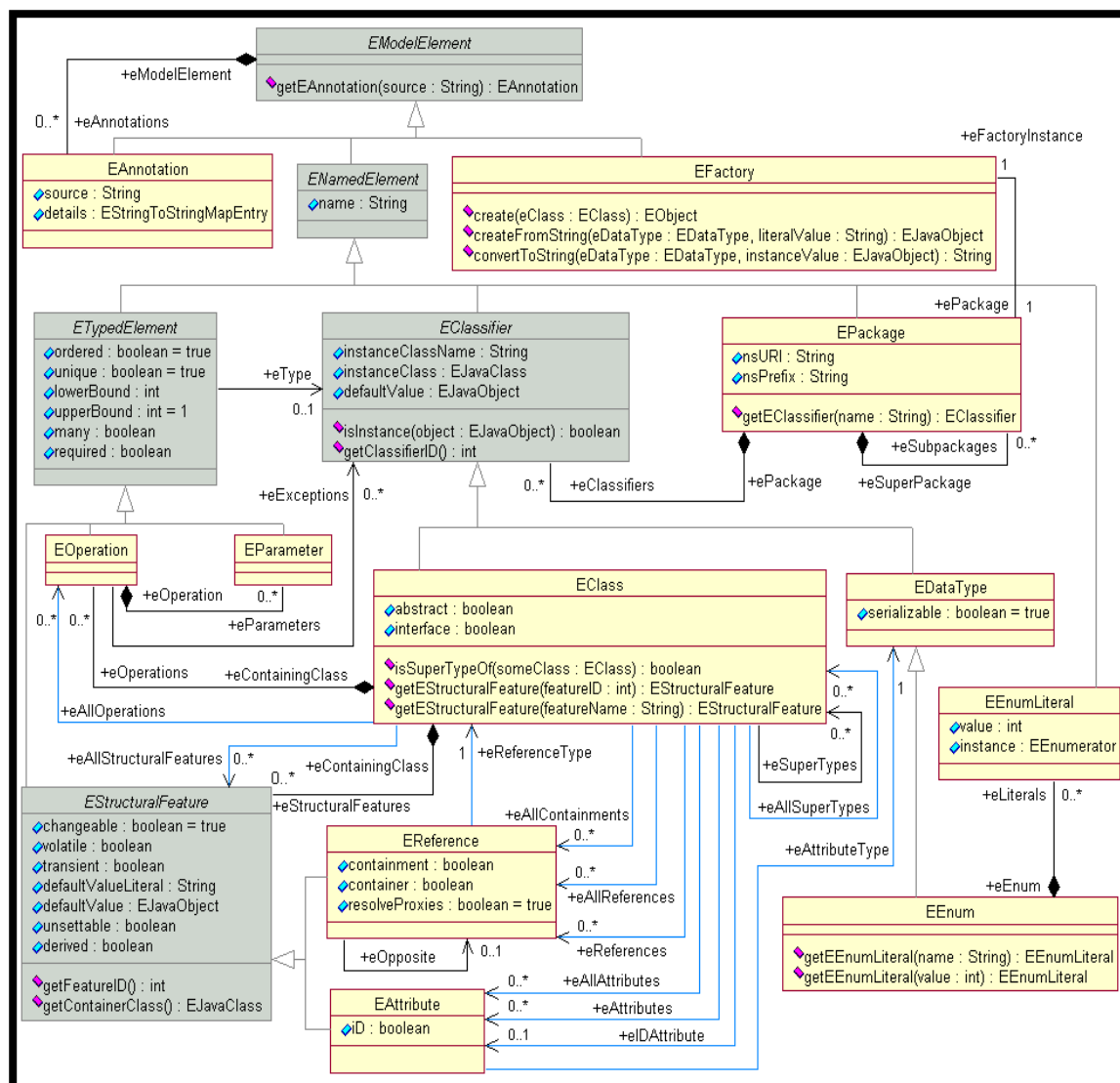


Ilustración 35 Modelo Ecore con sus relaciones, atributos y operaciones

3.7.2 El Metamodelo

Para la utilización de Ecore en Eclipse es necesario tener instalado el plugin de EMF (Eclipse Modeling Framework), este plugin provee básicamente dos herramientas para construir un modelo basado en Ecore, una el Ecore Model que es un editor manual que funciona en un estilo de árbol de navegación para la creación del modelo basado en Ecore, la otra es el Ecore Diagram siendo un editor grafico similar a las herramientas graficas para la creación de diagramas de clases UML. Cualquiera de las dos formas que se utilice para crear el diagrama basado en Ecore.

3.7.3 Construcción del editor para el modelo o lenguaje de dominio específico (DSL)

Como se está trabajando bajo Eclipse, para esta etapa se debe emplear GMF Tooling (Graphical Modeling Framework Tooling) que hace parte del proyecto GMP (Graphical Modeling Project)(Eclipse Foundation, 2012) . El proceso para la construcción del DSL Grafico se visualiza en la siguiente ilustración

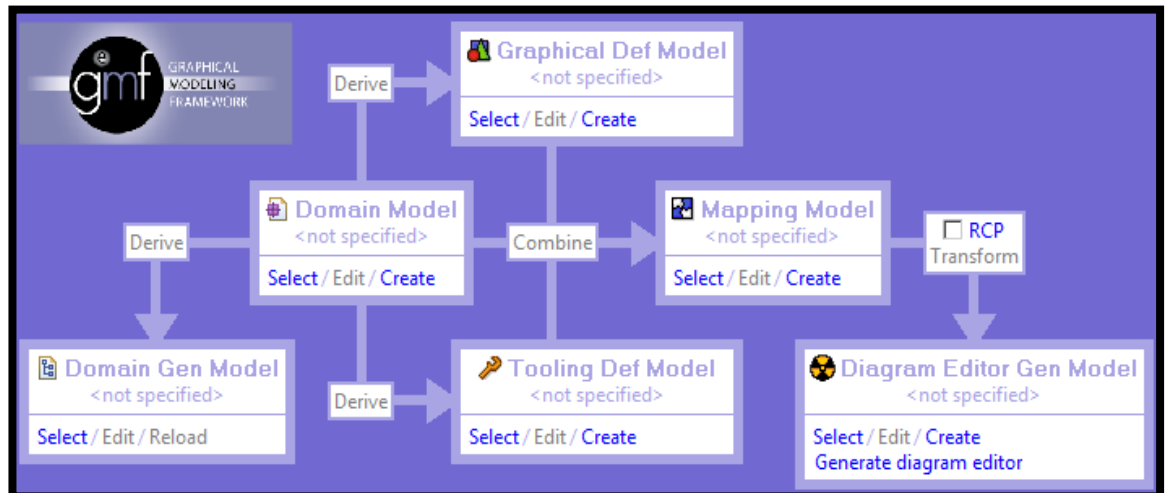


Ilustración 36 GMF Overview (Foundation, 2010b)

3.7.4 El Modelo

El diagrama que se obtenga como resultado de emplear el editor de DSL, tendrá asociado un fichero XMI que basará su sintaxis en el metamodelo creado.

3.7.5 Proceso de generación de código

En este último paso, existen varias tecnologías que se integran a Eclipse, como son Aceleo, Jet, Xpand, MOFScript, todas emplean el mismo principio, la creación de reglas de transformación basándose en un metamodelo.

Finalmente aplicando las tecnologías de transformación sobre el modelo creado se obtiene el código en el formato definido para su despliegue.

Capítulo 4. Solución Propuesta

4.1 Definición de la solución propuesta

A fin de lograr la validación de un metamodelo específico para funciones de interoperabilidad y reutilización de comercio electrónico se elaboró un estudio de los principales gestores de contenido de comercio electrónico disponibles en el mercado. En análisis permitió seleccionar las plataformas con mayor importancia por su funcionalidad, cuotas de mercado y por tratarse de ser aplicaciones en código abierto, ellas fueron Prestashop, Magento y OsCommerce, pueden verse los estudios realizados en los Anexos I y II.

Con ayuda de mapas de conocimiento individuales para cada una de las plataformas se logró una representación y abstracción del conjunto de conceptos que describen a cada una de ellas.

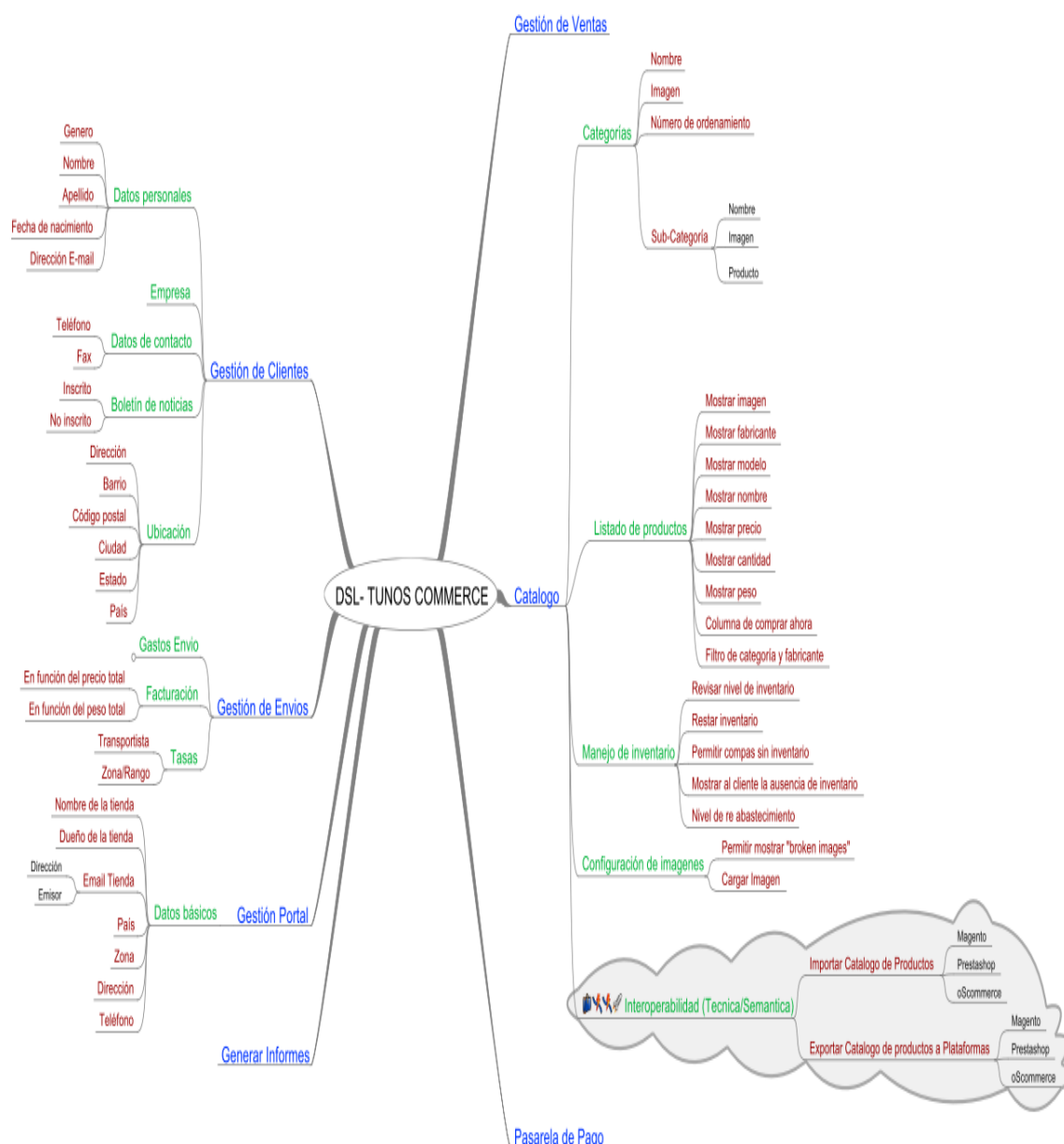


Ilustración 37 mapa de conocimiento tunos commerce

Con estos resultados se desarrolló una ontología que es el resultado de la caracterización de los procedimientos y la especificación semántica de la gestión de las plataformas previamente estudiadas. Se desarrolló con la herramienta Protégé, una herramienta interfaz gráfica para el procesamiento de datos cuyo resultado es la ontología en lenguaje OWL. De esta forma se logró la abstracción del dominio del problema que se pretendía resolver y se definió una semántica de procedimientos esenciales para la interoperabilidad entre plataformas.

Con el ánimo de reducir la complejidad que implica la migración de datos, la diferencia entre las plataformas, la disparidad semántica, entre otros se propuso un metamodelo inicial.

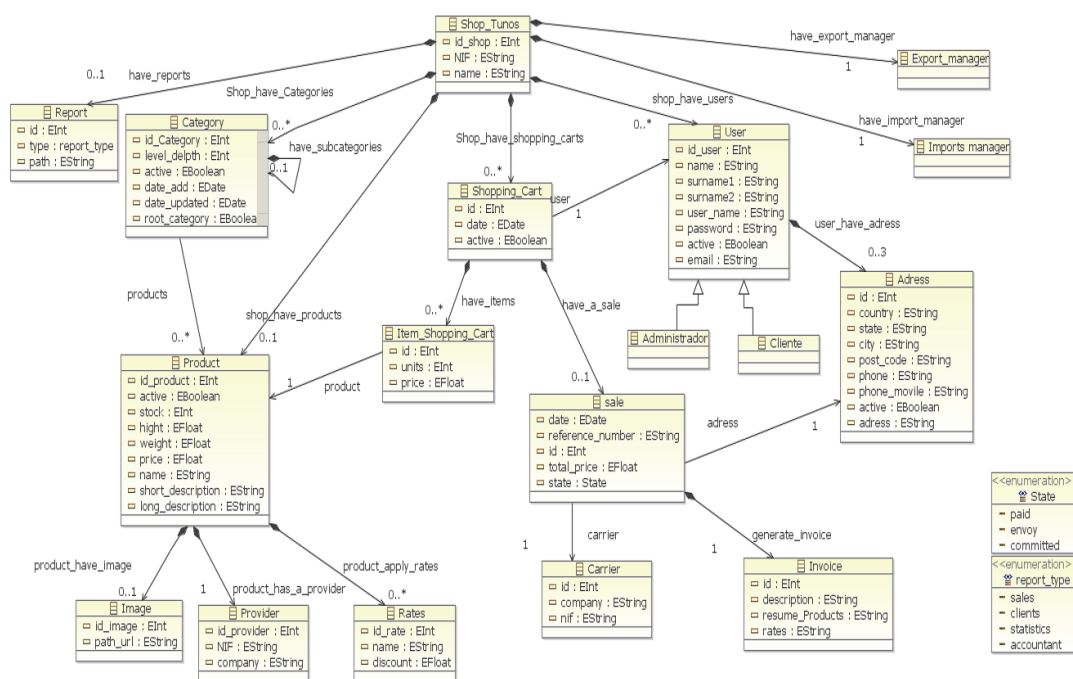


Ilustración 38 metamodelo inicial

Este metamodelo fue evaluado e implementado únicamente con aquellas meta-clases críticas para la interoperabilidad de catálogo de productos por ser este un proceso clave en la gestión de portales de comercio electrónico.

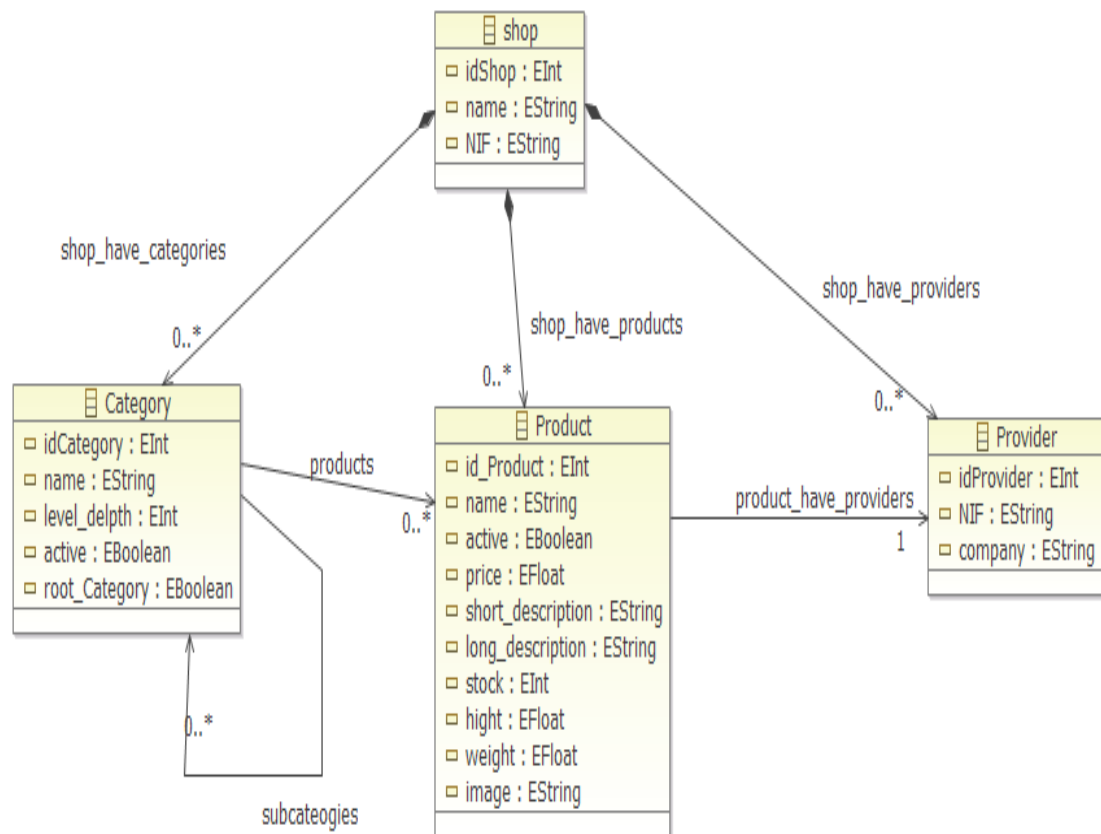


Ilustración 39 Metamodelo final para la gestión del catálogo de productos

A partir de esto se elabora una arquitectura global del sistema donde se puede ver cada uno de los subsistemas y como estos interpelarán entre sí y con el resto de las plataformas.

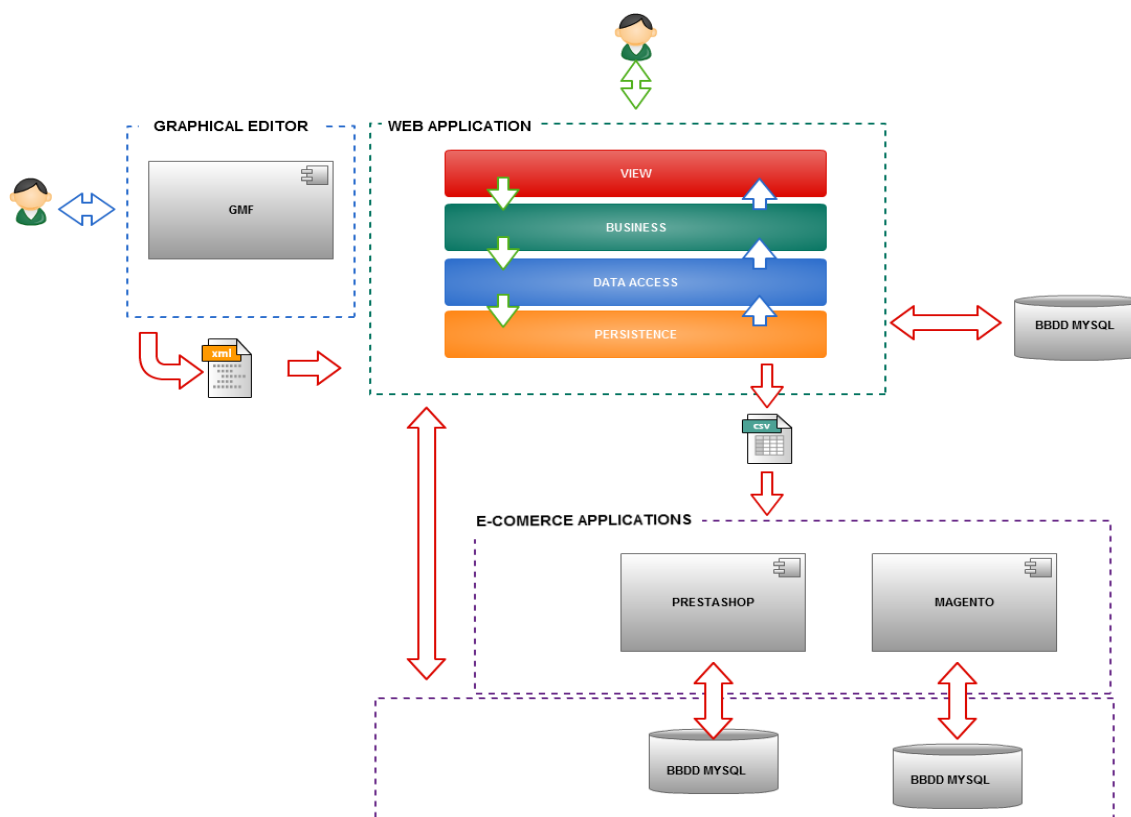


Ilustración 40 Arquitectura global del sistema

Como se puede observar a partir de metamodelo propuesto se generará un GMF que permita crear a través de diagramas UML catálogos de productos definidos por el usuario.

Por otro lado se define un entorno web en el que cada usuario podrá registrarse, crear y mantener su propio catálogo de productos. Se creará interoperabilidad con las plataformas de comercio electrónico Prestashop y Magento a través de las propias bases de datos, desde las cuales se podrá obtener e insertar elementos (productos, categorías y proveedores).

La plataforma web se comunicará con el GMF a través de un XML generado por este que recoge el catálogo de productos creados por el usuario.

Capítulo 5. Planificación del Proyecto y Resumen de Presupuestos

5.1 Planificación

El proyecto tiene como fecha de inicio 04/06/12 y fin 05/07/2013. El número total de tareas es un total de 89. Con 34 de ellas consideradas como críticas, divididas en 5 fases:

- Recopilación de la información (63 días)
- Estudio de plataformas de comercio electrónico (80,25 días)
- Alineamiento y definición de semántica de Procesos (75 días)
- Desarrollo y elaboración de la propuesta (108,75 días)
- Difusión de resultados (90,75 días)

A continuación se muestra de forma tabulada la lista de las tareas planificadas para el desarrollo del sistema, así como la duración, fechas de comienzo y fin.

Id	Nombre	Duración	Comienzo	Fin	Trabajo (h)
1	Fase 1: Recopilación de información	63 días	04/06/2012	29/08/2012	418 horas
2	Definir la lista de fuentes que se van a tomar en cuenta para la definición de procesos	16 horas	04/06/2012	05/06/2012	16 horas
3	Definición de fuentes secundarias de información	40 horas	06/06/2012	12/06/2012	40 horas
4	Definición de fuentes primarias	56 horas	06/06/2012	14/06/2012	56 horas
5	Definir Condiciones y variables claves para la implantación de una solución	8 horas	06/06/2012	06/06/2012	8 horas
6	Reunión de Control y Revisión de Avances	2 horas	07/06/2012	07/06/2012	4 horas
7	Establecer metodología de implementación	96 horas	07/06/2012	25/06/2012	96 horas
8	Definición de variables críticas	50 horas	25/06/2012	03/07/2012	50 horas
9	Caracterización de los procesos y procedimientos de Comercio Electrónico	46 horas	03/07/2012	11/07/2012	46 horas
10	Revisión de procesos empresariales de comercio electrónico	48 horas	11/07/2012	19/07/2012	48 horas

11	Reunión de Control y Revisión de Avances	2 horas	19/07/2012	19/07/2012	6 horas
12	Definición de procedimientos de una solución de Comercio Electrónico	48 horas	22/08/2012	29/08/2012	48 horas
13	Fase 2: Estudio de plataformas de Comercio Electrónico	80,25 días	01/08/2012	21/11/2012	644 horas
14	Definición de Gestores de Contenido de Comercio Electrónico (plataformas) sujetas de análisis	10 horas	30/08/2012	31/08/2012	10 horas
15	Instalación de Magento	2 horas	01/08/2012	01/08/2012	2 horas
16	Instalación Prestashop	2 horas	01/08/2012	01/08/2012	2 horas
17	Instalación oScommerce	2 horas	02/08/2012	02/08/2012	2 horas
18	Reunión de Control y Revisión de Avances	2 horas	02/08/2012	02/08/2012	2 horas
19	Desarrollo de mapas de conocimiento	16 horas	20/08/2012	21/08/2012	32 horas
20	Instalación y estudio de la Herramienta FreeMind	2 horas	31/08/2012	31/08/2012	2 horas
21	Mapa de Conocimiento Magento	32 horas	31/08/2012	06/09/2012	64 horas
22	Mapa de Conocimiento Prestashop	32 horas	06/09/2012	12/09/2012	64 horas
23	Mapa de Conocimiento oScommerce	32 horas	12/09/2012	18/09/2012	64 horas
24	Análisis de Plataformas	96 horas	27/08/2012	20/09/2012	160 horas
25	Reunión de Control y Revisión de Avances	4 horas	05/10/2012	08/10/2012	6 horas
26	Condiciones de implementación y requisitos tecnológicos	8 horas	10/10/2012	10/10/2012	8 horas
27	Requisitos de Hardware requerido	4 horas	11/10/2012	11/10/2012	4 horas
28	Requisitos de software requerido	8 horas	12/10/2012	12/10/2012	8 horas
29	Requisitos de Humaware requerido	8 horas	12/10/2012	12/10/2012	8 horas
30	Análisis funciones de Catálogo de Productos	50 horas	15/10/2012	23/10/2012	96 horas
31	Definición de los procedimientos a implementar	16 horas	23/10/2012	25/10/2012	32 horas
32	Reunión de Control y Revisión de Avances	2 horas	25/10/2012	25/10/2012	6 horas
33	Análisis de procesos de gestión de catálogo de productos	24 horas	25/10/2012	30/10/2012	24 horas
34	Documentación de Resultados de Evaluación de Plataformas de Comercio Electrónico	46 horas	05/11/2012	21/11/2012	48 horas
35	Fase 3: Alineamiento y definición de semántica de Procesos	75 días	02/10/2012	14/01/2013	560 horas
36	Establecer características comunes para la definición del metamodelo	48 horas	02/10/2012	09/10/2012	48 horas
37	Diagrama de flujos de procesos de una solución integrada de Comercio Electrónico	24 horas	10/10/2012	12/10/2012	24 horas

38	Generación de Taxonomía de procesos	48 horas	05/11/2012	12/11/2012	96 horas
39	Reunión de Control y Revisión de Avances	2 horas	13/11/2012	13/11/2012	6 horas
40	Definición de condiciones de Interoperabilidad organizacional, semántica y técnica	40 horas	14/11/2012	20/11/2012	40 horas
41	Definir Ontología de Proceso E-commerce	24 horas	21/11/2012	23/11/2012	24 horas
42	Instalación de la Herramienta Protégé (OWL)	2 horas	26/11/2012	26/11/2012	2 horas
43	Determinación del Dominio y Alcance de la Ontología	8 horas	27/11/2012	27/11/2012	16 horas
44	Consideración de Reutilización de Ontologías Existentes	16 horas	28/11/2012	29/11/2012	16 horas
45	Enumeración de términos importantes para la Ontología	24 horas	30/11/2012	04/12/2012	48 horas
46	Reunión de Control y Revisión de Avances	2 horas	05/12/2012	05/12/2012	6 horas
47	Definir Clases y Jerarquía de Clases	24 horas	06/12/2012	10/12/2012	24 horas
48	Definir Propiedades de las Clases: Slots	24 horas	11/12/2012	13/12/2012	24 horas
49	Cardinalidad y tipo de Valor de los Slots	8 horas	14/12/2012	14/12/2012	8 horas
50	Implementación de la Ontología de Proceso E-commerce Propuesta	32 horas	17/12/2012	20/12/2012	32 horas
51	Documentación de los Resultados	48 horas	24/12/2012	31/12/2012	96 horas
52	Reunión de Control y Revisión de Avances	2 horas	21/12/2012	21/12/2012	6 horas
53	Especificación de la Arquitectura Funcional	18 horas	01/01/2013	03/01/2013	36 horas
54	Alineación de Arquitectura y Ontología Propuesta y Definición de Mapa Conceptual del Metamodelo	8 horas	14/01/2013	14/01/2013	8 horas
55	Fase 4: Desarrollo y elaboración de la propuesta	108,75 días	14/01/2013	13/06/2013	778 horas
56	(MDE) Definición de Entidades del Metamodelo	48 horas	14/01/2013	21/01/2013	48 horas
57	Definición de Atributos y Métodos	48 horas	22/01/2013	29/01/2013	48 horas
58	Reunión de Control y Revisión de Avances	2 horas	30/01/2013	30/01/2013	6 horas
59	Construcción MDE	24 horas	30/01/2013	04/02/2013	24 horas
60	(DSL) Domain Generation Model	24 horas	04/02/2013	07/02/2013	24 horas
61	Graphical definition Model	8 horas	07/02/2013	08/02/2013	8 horas
62	Tooling Definition Model	4 horas	08/02/2013	08/02/2013	4 horas
63	Mapping Model	16 horas	08/02/2013	12/02/2013	16 horas
64	Diagram Editor Generation Model	24 horas	12/02/2013	15/02/2013	24 horas

65	Generación de XML	24 horas	15/02/2013	20/02/2013	24 horas
66	Reunión de Control y Revisión de Avances	2 horas	20/02/2013	20/02/2013	6 horas
67	Análisis de la Base de Datos a implementar	20 horas	21/02/2013	25/02/2013	20 horas
68	Análisis de las BD de las plataformas	40 horas	25/02/2013	04/03/2013	80 horas
69	Identificación de Entidades y atributos comunes	4 horas	04/03/2013	04/03/2013	4 horas
70	Definición de modelo E-R para la solución	8 horas	05/03/2013	05/03/2013	8 horas
71	Análisis y Diseño de la Aplicación y servicios WEB	40 horas	06/03/2013	12/03/2013	40 horas
72	Estudios de Herramientas tecnológicas a utilizar	8 horas	13/03/2013	13/03/2013	16 horas
73	Instalación de Herramientas y puesta en marcha del entorno	32 horas	14/03/2013	19/03/2013	32 horas
74	Reunión de Control y Revisión de Avances	2 horas	20/03/2013	20/03/2013	6 horas
75	Diseño de interfaces	16 horas	20/03/2013	22/03/2013	16 horas
76	Conexión con la BD	2 horas	22/03/2013	22/03/2013	2 horas
77	Implementación de la web	160 horas	25/03/2013	19/04/2013	160 horas
78	Estudio de Alternativas de interoperabilidad de prototipo con plataformas	32 horas	22/04/2013	13/06/2013	32 horas
79	Interconexión entre Prototipo y Plataforma de comercio electrónico seleccionadas	60 horas	15/04/2013	24/04/2013	60 horas
80	Pruebas Unitarias	10 horas	25/04/2013	26/04/2013	10 horas
81	Pruebas de Integración y del Sistema	10 horas	26/04/2013	29/04/2013	10 horas
82	Reunión de Control y Revisión de Avances	2 horas	29/04/2013	29/04/2013	2 horas
83	Pruebas de Usabilidad	8 horas	29/04/2013	30/04/2013	16 horas
84	Pruebas de Accesibilidad	8 horas	30/04/2013	01/05/2013	16 horas
85	Pruebas de Rendimiento	16 horas	01/05/2013	06/06/2013	16 horas
86	Fase 5: Difusión de Resultados	90,75 días	01/03/2013	05/07/2013	112 horas
87	Generación de la Memoria	200 horas	01/03/2013	21/05/2013	100 horas
88	Tiempo de margen para posibles reajustes	15 días	14/06/2013	05/07/2013	0 horas
89	Presentación de Resultados	4 horas	05/07/2013	05/07/2013	12 horas

A parte de los hitos de cada fase existen otros cinco hitos a lo largo del proyecto:

- Definir Condiciones y variables claves para la implantación de una solución (T. 5)
- Análisis de Plataformas (T.24)
- Interconexión entre Prototipo y Plataforma de comercio electrónico seleccionadas (T.79)
- Generación de la Memoria(T.87)
- Presentación de Resultados (T.89)

En cuanto a la ruta crítica existen dos tareas que están en paralelo “Alineación de Arquitectura y Ontología Propuesta y Definición de Mapa Conceptual del Metamodelo” (T.54) y “(MDE) Definición de Entidades del Metamodelo” (T.56). Esto es posible ya que cada una de las tareas está asignada a dos recursos diferentes que pueden llevarlas perfectamente en paralelo. Las tareas con mayor tiempo estimado se corresponden a tareas de implementación y documentación:

- Análisis de plataformas(T.24)
- Implementación de la web (T.77)
- Generación de la memoria(T.87)

Se ha incluido una tarea “Tiempo de margen para posibles reajustes”, con un tiempo estimado de 15 días aproximadamente para tener colchón de margen para posibles reajustes que se vayan produciendo a lo largo del proyecto además de disponer de una media de holgura de unas 55,7 horas (6.96 días).

En la figura mostrada a continuación se muestra el diagrama de Gantt en el que pueden observarse las etapas seguidas que ya se han descrito en detalle anteriormente.

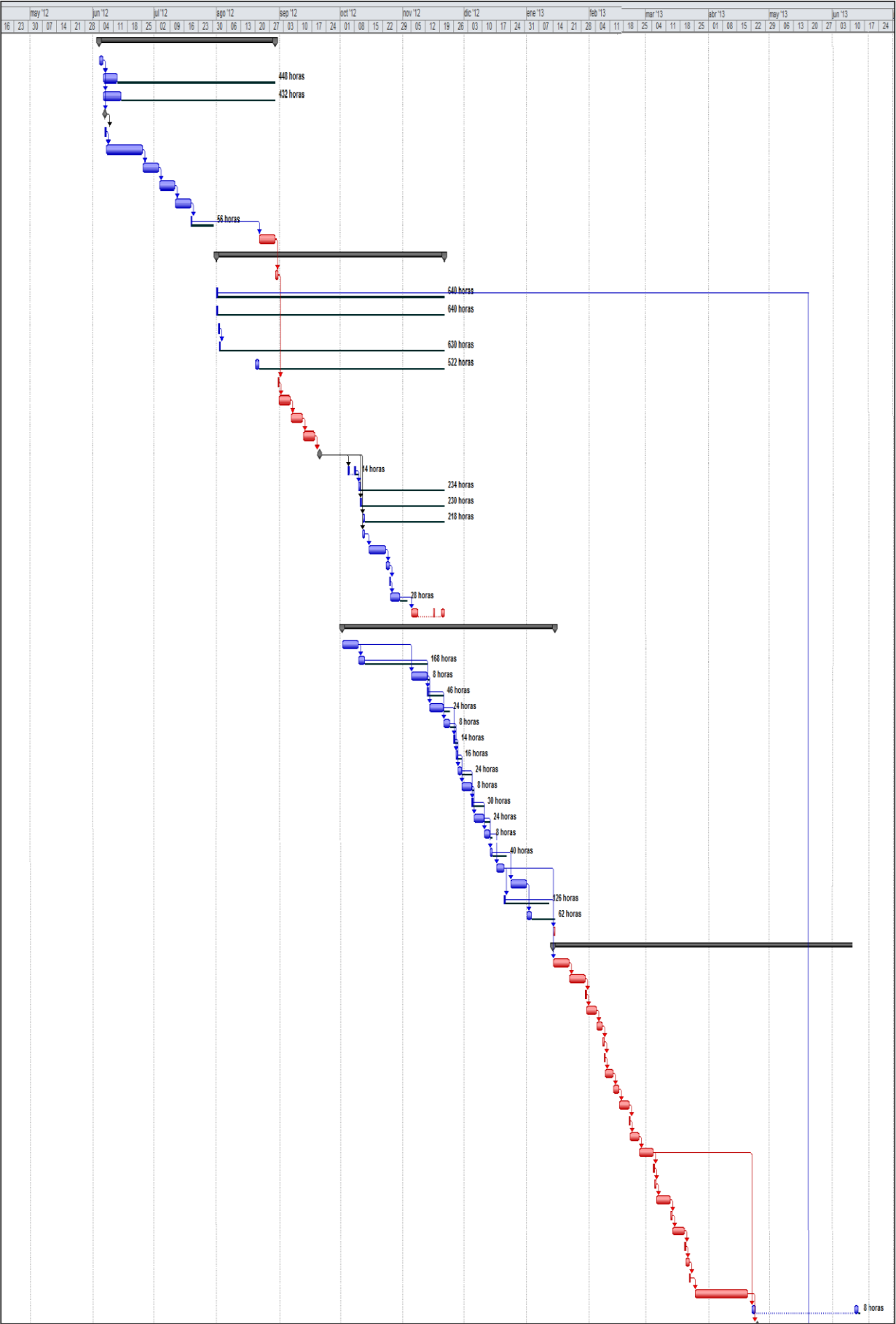


Ilustración 41 Diagrama de Gantt

5.2 Resumen del Presupuesto

El presupuesto del proyecto se valorará de la siguiente manera:

Ítem	Concepto	Cantidad	Total
01	Coste de personal de desarrollo	3	64478€.
02	Licencias Software	1	697€
03	Hardware	5	1.360€
04	Oficina	1	2.600€
05	Materiales (Papelería, libros...)	1	1.200€
Subtotal			70.335€
Beneficio (25%)			17.583,75€
IVA (21%)			14.770,35
TOTAL			102.689,1€

El coste de personal de desarrollo engloba todos los gastos relacionados con las tres fases de desarrollo software: análisis, diseño e implementación, incluyendo en la fase de implementación las pruebas necesarias.

Se incluye dentro del presupuesto del proyecto todos los costes de amortización de licencias. Dentro del Ítem Hardware se incluyen todos los gastos relacionados con el hardware necesario para la realización del proyecto (tres PCs, un servidor y una impresora).

Los demás Ítems Oficina y materiales de oficina se engloba por un lado el alquiler del local y por otro todos los gastos relacionados con el material de oficina (papel, tinta, CDs, libros...), electricidad, agua, Internet y teléfono.

Teniendo en cuenta todos los conceptos anteriores el valor del subtotal estimado asciende a 70.335€ a este valor se le aplica un 25% de beneficio y el 21% de IVA, obteniendo un total de 102.689,1€. Para mas información ver Capítulo 11 (Presupuesto)

Capítulo 6. Análisis

Este apartado contendrá toda la especificación de requisitos y toda la documentación del análisis de la aplicación, a partir de la cual se elaborará posteriormente el diseño.

6.1 Definición del Sistema

6.1.1 Determinación del Alcance del Sistema

Se requiere el desarrollo de un Lenguaje de Dominio Específico (DSL) como aplicación Web que despliegue funcionalidades para procesos interoperabilidad y reutilización de la información entre plataformas de e-commerce, específicamente Prestashop y Magento. Los criterios de selección de estas plataformas fueron explicados previamente, así como el análisis de las mismas.

En la aplicación se podrán crear, modificar importar y exportar en formatos interoperables y reusables como XML, CSV, la información de catálogo de productos por ser este un proceso crítico en la gestión de contenidos de comercio electrónico de igual manera integrará un GMF que permitirá a usuarios no expertos la gestión de contenidos de catálogo de productos y su exportación a las plataformas previstas.

La aplicación deberá estar construida con herramientas tecnológicas innovadoras, cumpliendo estándares y requisitos y garantizado efectividad, eficiencia y usabilidad para usuarios profesionales y no expertos de gestores de contenido de comercio electrónico.

Esto se logrará a partir de los requisitos provistos sustentado en los principios propuestos por la Ingeniería Dirigida por Modelos (MDE), específicamente la propuesta de la Arquitectura Dirigida por Modelos (MDA), de esta forma se posibilitará minimizar el tiempo y esfuerzo necesarios para la interoperabilidad y la reutilización.

6.2 Requisitos del Sistema

6.2.1 Obtención de los Requisitos del Sistema

6.2.1.1 Requisitos funcionales

Código	Nombre Requisito	Descripción del Requisito
R1.1	Gestión de Usuarios	La aplicación debe permitir añadir un usuario al sistema, loguearse y tener su propia información en la aplicación.
R1.2	Gestión de categorías	En la aplicación, el usuario registrado podrá gestionar la creación de una nueva categoría, subcategorías, modificar las existentes, visualizar detalles
R1.3	Gestión de productos	En la aplicación, el usuario registrado podrá gestionar la creación de un nuevo producto, modificar los existentes, visualizar detalles, listar productos, generar código QR (<i>quick response code</i> , «código de respuesta rápida»),
R1.4	Gestión de proveedores	En la aplicación, el usuario registrado podrá gestionar la creación de un nuevo proveedor, modificar los existentes, visualizar detalles, listar proveedores
R1.5	Configuración de tiendas	El usuario debe poder configurar sus tiendas de tipo Prestahop o Magento para interoperar con ellas.
R1.6	Importar elementos	Una vez configurada la tienda, el usuario registrado podrá importar los elementos de esta (categorías, productos y proveedores). No sólo se importarán de la tienda las propias entidades sino que también se importarán las relaciones entre ellas
R1.7	Exportar elementos	Una vez creado un producto, una categoría o un proveedor y si el usuario tiene alguna tienda configurada Prestashop/Magento este podrá exportarse.
R1.8	Generar CSV	La aplicación debe poder generar archivos CSV con el catálogo de productos y compatible con las plataformas Prestashop y Magento
R1.9	Modelado del catálogo de datos	La aplicación debe poder modelar mediante un DSL un catálogo de datos de comercio electrónico y generar a partir de este un XML que recoja toda la información.

R1.10	Importar XML	El sistema permite importar elementos (categorías, productos o proveedores) a través de XML. Este XML tiene que tener un formato exactamente igual al que genera nuestro GMF.

6.2.1.2 Requisitos no funcionales

6.2.1.2.1 Requisitos tecnológicos

Código	Nombre Requisito	Descripción del Requisito
R2.1	Sistema operativo de la aplicación	Es necesario el uso del sistema operativo Windows para la gestión de contenidos de comercio electrónico
R2.2	Procedimientos de interoperabilidad y reusabilidad de la información	Para aprovechar las funcionalidades de la aplicación deberá estar configurada una tienda Prestashop y/o Magento
R2.3	Sistema de gestión de bases de datos	Se empleará un sistema de gestión de bases de datos de tipo Mysql
R2.4	Conexión a internet	El equipo del usuario y el servidor donde se aloje la aplicación web deben contar con una conexión a internet funcional.
R2.5	Navegador Web	El usuario podrá utilizar la aplicación web desde al menos dos navegadores Mozilla Firefox (v.2.0) y Google Chrome (v.2.4)
R2.6	Contenedor de Servlets	Para el despliegue de la aplicación web se empleará un apache Tomcat V 6.0

6.2.1.2.2 Requisitos del sistema

Código	Nombre Requisito	Descripción del Requisito
R3.1	Procesador	Debe utilizarse una CPU con procesador Pentium IV a 3 GHz o superior.
R3.2	Memoria RAM	Se requiere al menos 1GB de RAM en equipos de 32 bits o 2GB en uno de 64 bits.

6.2.1.2.3 Requisitos de usabilidad

Código	Nombre Requisito	Descripción del Requisito
R4.1	Interfaz intuitiva	La interfaz de gestión de contenidos de comercio electrónico e interoperabilidad de plataformas debe ser intuitiva y fácil de utilizar para que el usuario pueda desarrollar sus actividades.
R4.2	Conocimiento sobre la implementación de plataformas de comercio electrónico	El usuario deberá tener unos conocimientos mínimos de comercio electrónico para poder interoperar con las plataformas.

6.2.1.2.4 Tiempos de respuesta

Código	Nombre Requisito	Descripción del Requisito
R5.1	Tiempos de respuesta	La interfaz de la aplicación deberá funcionar de manera fluida y sin demasiadas ralentizaciones que puedan despistar a la navegación del usuario.

6.2.2 Identificación de Actores del Sistema

- Usuario Web
- Usuario GMF

6.2.2.1 Usuario Web

Es el actor que utilizará la herramienta web con la que podrá crear productos, categorías, proveedores e interactuar con otras plataformas de comercio electrónico. El usuario web no tiene por qué tener conocimientos muy altos sobre comercio electrónico, únicamente ha de ser capaz de poder manejar a nivel básico las plataformas “Prestashop” y/o “Magento.”

6.2.2.2 Usuario GMF

Es el actor que utilizará la herramienta GMF para generar los XML que contengan el catálogo de productos, categorías, proveedores y las relaciones entre estas. El actor GMF no se le presuponen conocimientos sobre el desarrollo de GMF's ni su funcionamiento.

6.2.3 Especificación de Casos de Uso

A lo largo de esta sección se estudiarán los casos de uso de la aplicación que involucren a los actores anteriormente citados

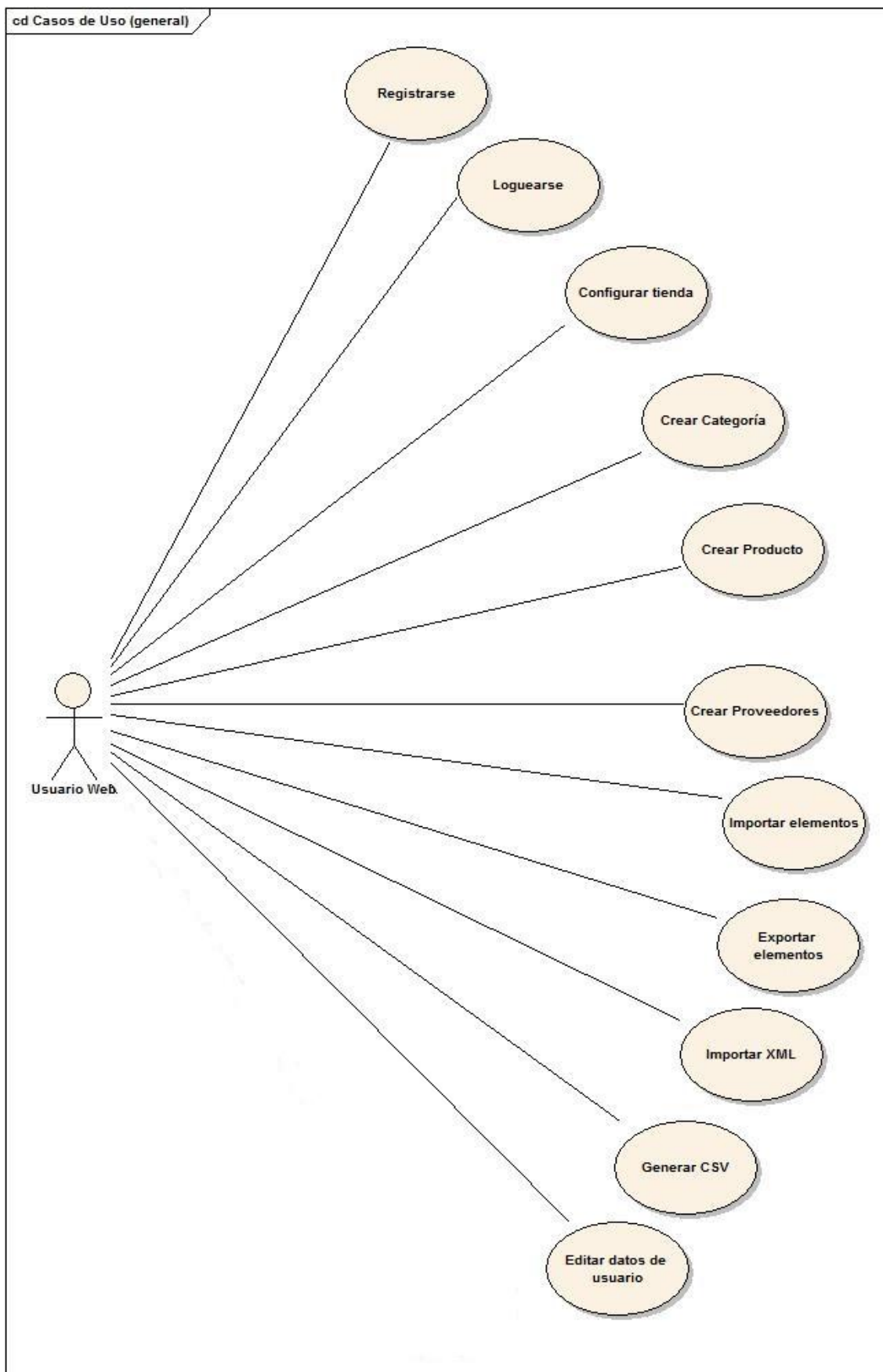


Ilustración 42 Diagrama general de casos de uso del usuario web

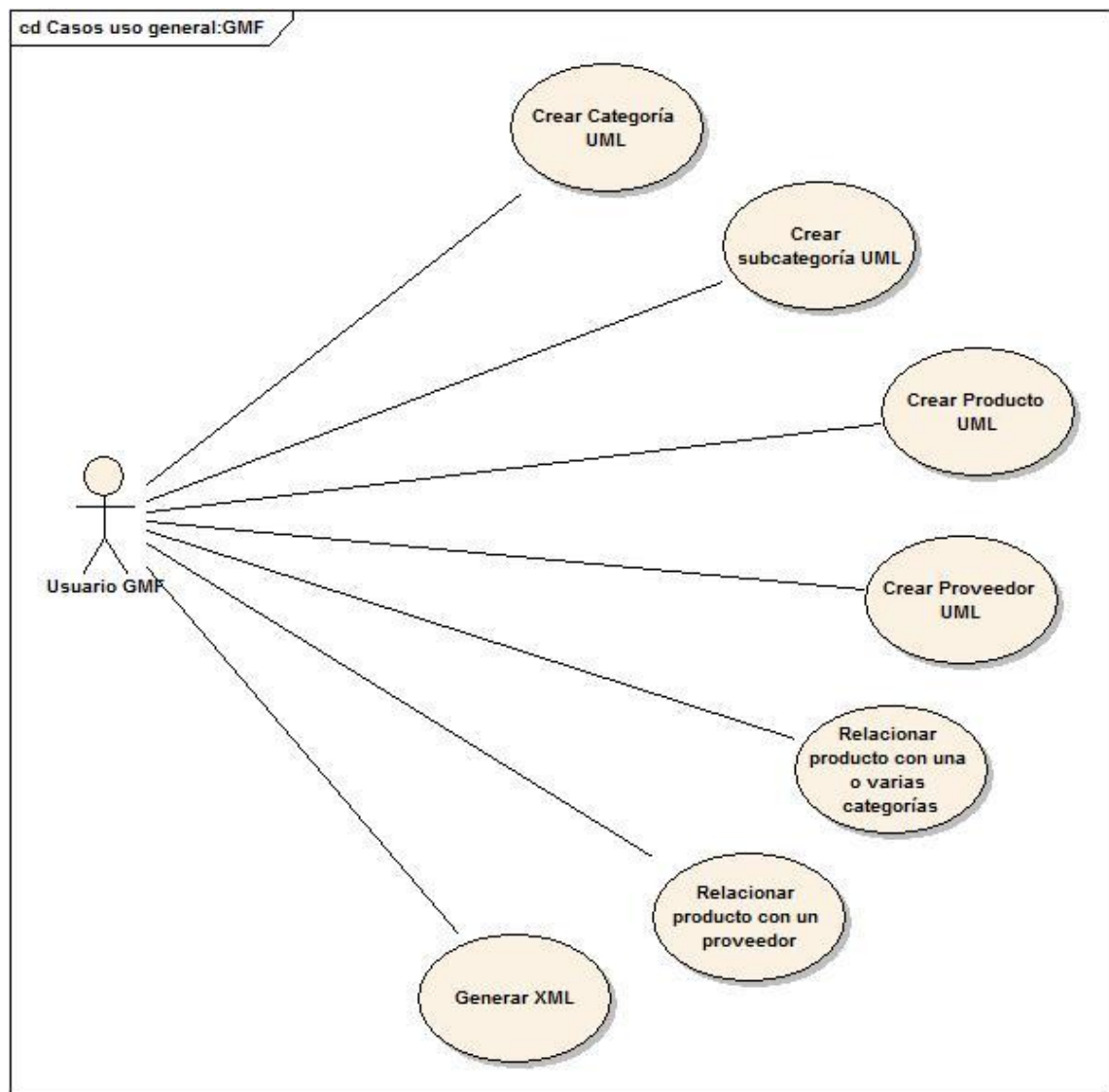


Ilustración 43 Diagrama general casos de uso usuario GMF

6.2.3.1 Registrarse

Nombre del Caso de Uso	
Registrarse	
Descripción	
Para poder acceder al sistema el usuario tendrá la posibilidad de registrarse.	

6.2.3.2 Loguearse

Nombre del Caso de Uso	
Loguearse	
Descripción	
Una vez que el usuario está registrado en el sistema éste podrá loguearse para poder acceder a la plataforma y realizar aquí las actividades que crea convenientes. Se asegura que el usuario podrá acceder al sistema con total seguridad.	

6.2.3.3 Configurar tienda

Nombre del Caso de Uso
Configurar tienda
Descripción
Una vez logueado en el sistema el usuario podrá crear y configurar su tienda desde la plataforma web para poder interoperar con ella. En este caso se pueden configurar dos tipos de plataformas diferentes: “Prestashop” y “Magento”. Durante la configuración de la tienda se pedirán datos relacionados con el acceso a la base de datos de la plataforma a configurar. Para que funcione todo correctamente será necesario que la base de datos de la plataforma que se pretende configurar sea accesible desde el exterior.

6.2.3.4 Crear categoría

Nombre del Caso de Uso
Crear categoría
Descripción
El usuario tendrá la posibilidad de crear categorías dentro de la plataforma web. Las categorías creadas por un usuario sólo podrán ser visibles por el usuario en la propia plataforma. Una vez creada una categoría el usuario podrá desarrollar varias operaciones sobre esta. Los diversos procedimientos posible en esta funcionalidad los subcasos de uso que se presentan a continuación.

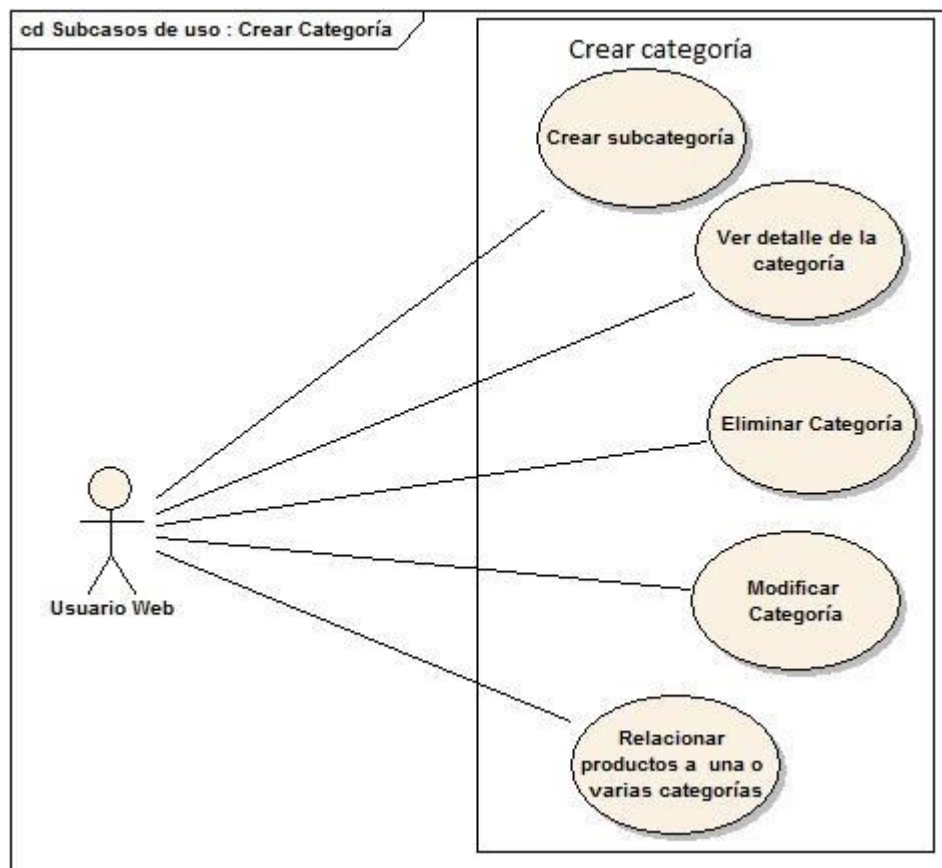


Ilustración 44 Subcasos de uso "Crear categoría"

6.2.3.4.1 Subcaso de uso: crear subcategoría

Nombre del Caso de Uso	
Crear subcategoría	
Descripción	
El usuario tendrá la posibilidad de crear subcategorías dentro de una misma categoría ya creada, y crearle a éstas a su vez más subcategorías de manera que pueda formarse una estructura jerárquica.	
<pre> graph LR Ropa[Ropa] --> RopaMujer[Ropa de mujer] Ropa --> RopaHombre[Ropa de hombre] RopaMujer --> Faldas[Faldas] RopaMujer --> Camisas[Camisas] </pre>	

6.2.3.4.2 Subcaso de uso: ver detalle de la categoría

Nombre del Caso de Uso	
Ver detalle de la categoría	
Descripción	
Una vez creada una categoría el usuario podrá consultar en todo momento toda la información relacionada con ella. De manera que si se producen cambios en ésta el usuario podrá consultarlos.	

6.2.3.4.3 Subcaso de uso: eliminar categoría

Nombre del Caso de Uso
Eliminar categoría
Descripción
Una vez creada una categoría el usuario podrá eliminarla. Si una categoría tiene a su vez subcategorías se optará por únicamente desvincularla de estas sin hacer un barrido de eliminación hacia sus categorías hijo.

6.2.3.4.4 Subcaso de uso: modificar categoría

Nombre del Caso de Uso
Modificar categoría
Descripción
Una vez creada una categoría el usuario podrá modificar su información de conformidad a sus requerimientos.

6.2.3.4.5 Subcaso de uso: relacionar productos a una o más categorías

Nombre del Caso de Uso
Relacionar productos a una o más categorías
Descripción
Una vez creada una o más categorías el usuario podrá agrupar sus productos por categoría. De esta manera se consigue que los productos puedan quedar clasificados por categoría.

6.2.3.5 Crear producto

Nombre del Caso de Uso
Crear producto
Descripción
El usuario tendrá la posibilidad de crear productos dentro de la plataforma web. Los productos creados por un usuario sólo podrán ser visibles por éste. Una vez creada un producto el usuario podrá hacer varias operaciones sobre él. Por ello se ha considerado conveniente dividir este caso de uso en los subcasos que se incluyen a continuación.

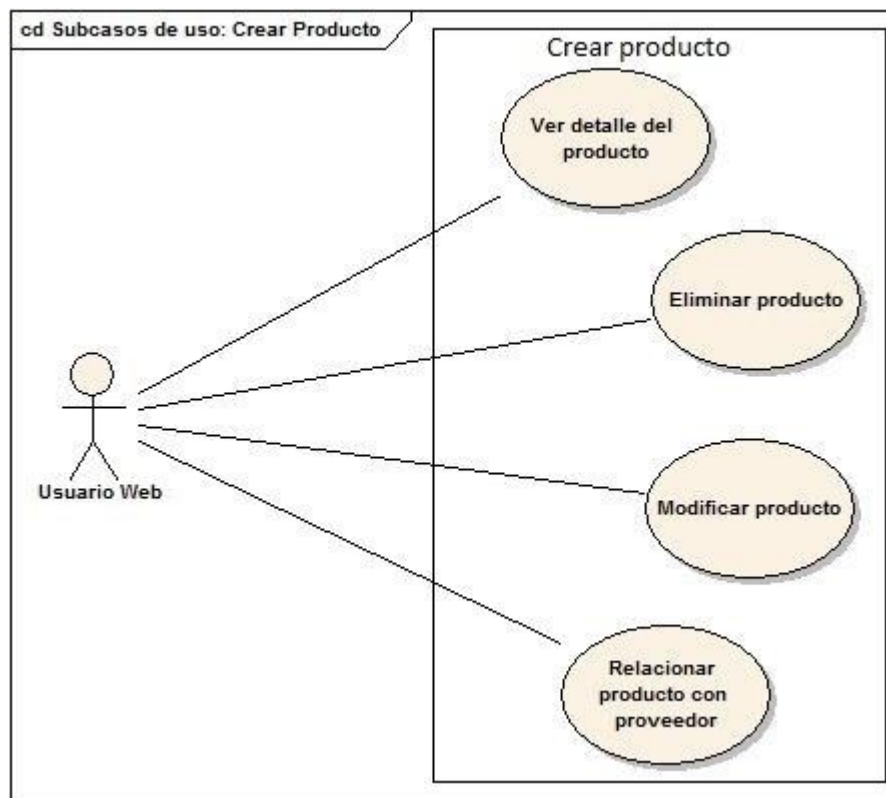


Ilustración 45 Subcasos de uso "Crear Producto"

6.2.3.5.1 Subcaso de uso: ver detalle del producto

Nombre del Caso de Uso
Ver detalle del producto
Descripción
Una vez creado un producto el usuario podrá consultar en todo momento toda la información relacionada con él. De manera que si se producen cambios en éste el usuario podrá consultarlos.
Otra de las cosas que se considera oportuna a la hora de mostrar el detalla de un producto es generar un código QR con información sobre éste de manera que el usuario si lo desea puede manejar ésta información a través de un dispositivo móvil.

6.2.3.5.2 Subcaso de uso: eliminar producto

Nombre del Caso de Uso
Eliminar producto
Descripción
Una vez creada un producto el usuario podrá eliminarlo. Si un producto está vinculado a una o más categorías o a un proveedor únicamente se desvincula de estos de manera que sólo se borra la información del producto quedando los demás intactos en el sistema.

6.2.3.5.3 Subcaso de uso: modificar producto

Nombre del Caso de Uso
Modificar producto
Descripción
Una vez creado un producto el usuario podrá modificar su información de acuerdo a los requerimientos de su negocio.

6.2.3.5.4 Subcaso de uso: relacionar productos con proveedor

Nombre del Caso de Uso
Relacionar producto con proveedor
Descripción
Cuando se crea el producto este se puede relacionar con un único proveedor. Esta funcionalidad facilita la gestión de inventario y puede ser útil para integrar plataformas de ERP

6.2.3.6 Crear Proveedor

Nombre del Caso de Uso
Crear proveedor
Descripción
El usuario tendrá la posibilidad de crear proveedores dentro de la plataforma web. Los proveedores creados por un usuario sólo podrán ser visibles por éste. Una vez creado un proveedor el usuario podrá desarrollar procedimientos importante en las tareas de control de inventarios y re-aprovisionamiento de mercaderías. Por ello se ha considerado conveniente dividir este caso de uso en los subcasos que se incluyen a continuación.

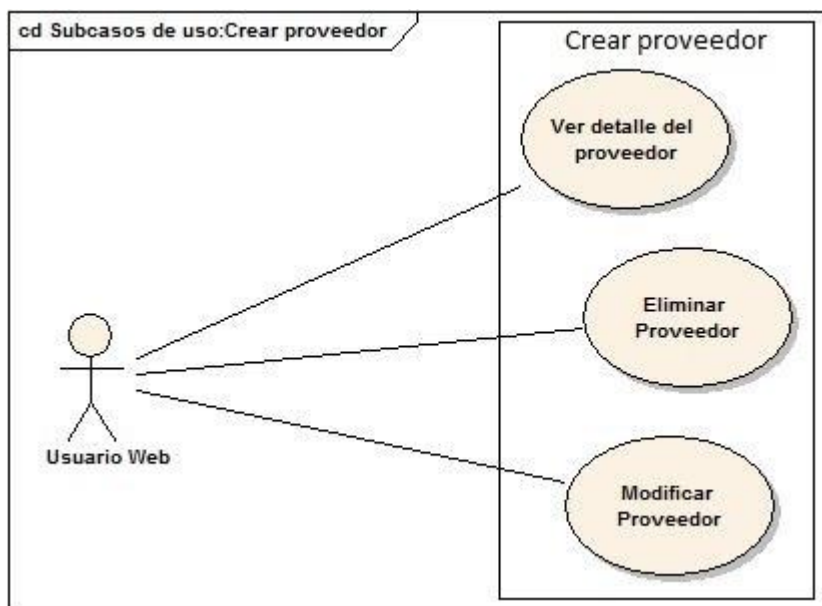


Ilustración 46 Diagrama de subcasos de uso "Crear Proveedor"

6.2.3.6.1 Subcaso de uso: ver detalle del proveedor

Nombre del Caso de Uso
Ver detalle del proveedor
Descripción
Una vez creado un proveedor el usuario podrá consultar la información detallada previamente almacenada del mismo. La gestión de proveedores es una tarea importante en el control de inventarios. La información detallada del mismo facilita esta funcionalidad. La interoperabilidad de plataformas en procedimientos propios de proveedores es una ventaja funcional.

6.2.3.6.2 Subcaso de uso: eliminar proveedor

Nombre del Caso de Uso
Eliminar proveedor
Descripción
Una vez creado un proveedor el usuario podrá eliminarlo. Sí este proveedor está relacionado con uno o varios producto se desvinculará la relación dejando estos productos momentáneamente sin proveedor.

6.2.3.6.3 Subcaso de uso: modificar proveedor

Nombre del Caso de Uso
Modificar proveedor
Descripción
Una vez creado un proveedor el usuario podrá modificar su información a su antojo.

6.2.3.7 Caso de uso: Importar elementos

Nombre del Caso de Uso
Importar elementos
Descripción
Esta es una de las funcionalidades más poderosas de la solución propuesta. Una vez están correctamente configuradas las plataformas con las que se va a interoperar (Prestashop v.1.5.4 o Magento v.1.6) se pueden importar conjuntos de datos de estas. De manera que puedan quedar almacenados en nuestro sistema. Los elementos que se pueden importar son: • Jerarquía de categorías • Productos • Proveedores • Relaciones de los productos y sus categorías • Relaciones de los productos y sus proveedores En el caso particular de los gestores de contenido de comercio electrónico Magento o Prestashop tiene un modelo relacional complejo, cada uno de los sistemas con los que se puede inter-operar en el procedimiento de importación maneja gran cantidad de información de cada uno de estos elementos (ver diagramas de entidad relación del Anexo II)

La funcionalidad prevista opta por substraer únicamente la información más relevante y que las plataformas con las que se inter-opera tienen en común y de esa manera lograr la reutilización y la interoperabilidad entre plataformas.

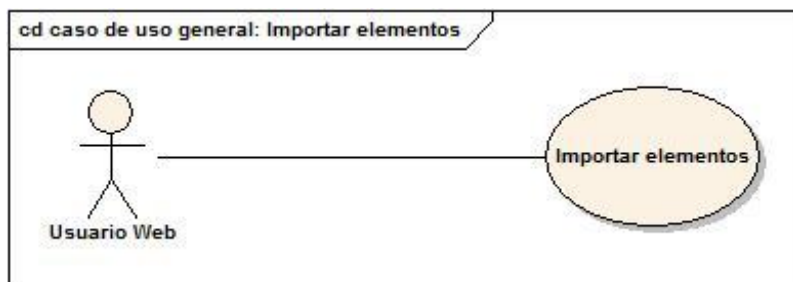


Ilustración 47 Caso de uso importar elementos

6.2.3.8 Exportar elementos

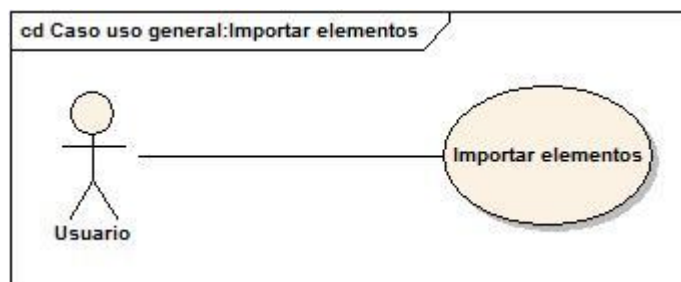
Nombre del Caso de Uso
Exportar elementos
Descripción
A partir de la información almacenada en el sistema propuesto (repositorio), se prevé la funcionalidad de exportar metadatos a las plataformas ya previstas (Prestashop v.1.5.4 o Magento v.1.6) . Una vez están correctamente configuradas las plataformas con las que se va a inter operar, se pueden exportar individualmente elementos de nuestra plataforma hacia cualquiera de las anteriormente mencionadas. Los elementos que se pueden exportar son: • Categorías • Productos • Proveedores • Relaciones de los productos y sus categorías • Relaciones de los productos y sus proveedores



Ilustración 48 Caso de uso exportar elementos

6.2.3.9 Caso de uso: Importar XML

Nombre del Caso de Uso
Importar XML generado por el DSL
Descripción
El usuario tendrá la posibilidad de importar el XML generado por nuestro DSL (o un XML que siga la misma estructura) y crear los elementos contenidos en este. Esta funcionalidad permite flexibilidad, escalabilidad y reutilización de información con el propósito integrar plataforma de comercio electrónico. Los metadatos generados en XML, cumplen los estándares necesarios para la migración descrita.



6.2.3.10 Generar CSV

Nombre del Caso de Uso
Generar CSV
Descripción
Como funcionalidad adicional El usuario tendrá la posibilidad de generar un CSV del catálogo de productos contenidos en la aplicación. El CSV generado será compatible con la plataforma Magento y con la plataforma Prestashop. De esta forma la herramienta propuesta puede integrarse con los plugins diseñados de manera independiente por cada una de las plataformas para gestionar información del catálogo de productos. La ventaja de este procedimiento es que ahorra tiempo de esfuerzo y programación de un experto en cada una de las plataformas, dado que la funcionalidad es transparente para el usuario, solo se requiere respetar la estructura del archivo plano generado

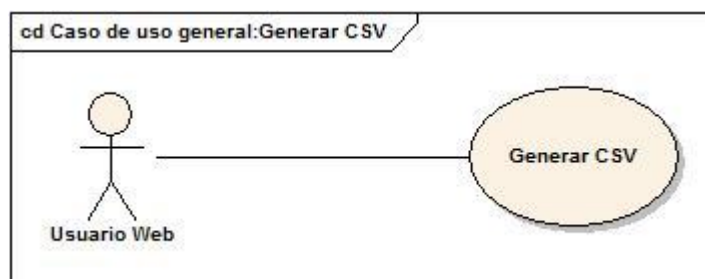


Ilustración 49 Caso de uso "Generar CSV"

6.2.3.11 Editar datos de usuario

Nombre del Caso de Uso
Editar datos de usuario
Descripción
Una vez ingresado en el sistema el usuario podrá modificar si cree conveniente sus datos personales: nombre, apellidos, e-mail, contraseña de acceso....

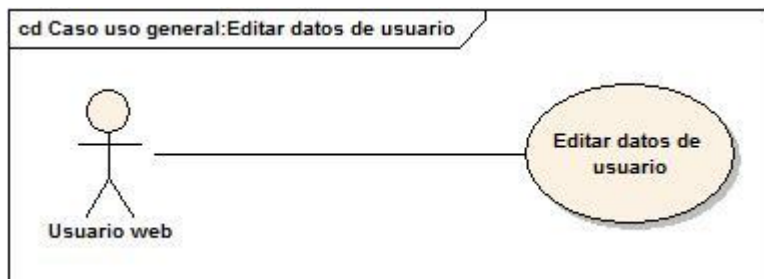


Ilustración 50 Caso de uso “Editar datos de usuario”

6.2.3.12 Crear Categoría UML

Nombre del Caso de Uso
Crear Categoría UML
Descripción
El usuario GMF podrá crear una entidad gráfica UML de tipo categoría con todos sus atributos

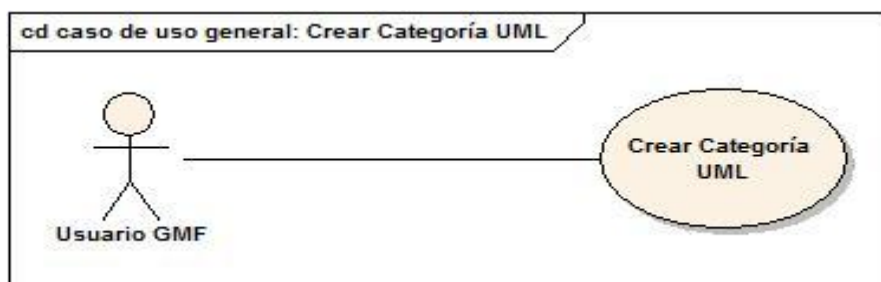


Ilustración 51 Caso de uso “Crear Categoría UML”

6.2.3.13 Crear Subcategoría UML

Nombre del Caso de Uso
Crear Subcategoría UML
Descripción
El usuario GMF podrá crear una sub-categoría de una categoría base, quedando así una estructura jerárquica.

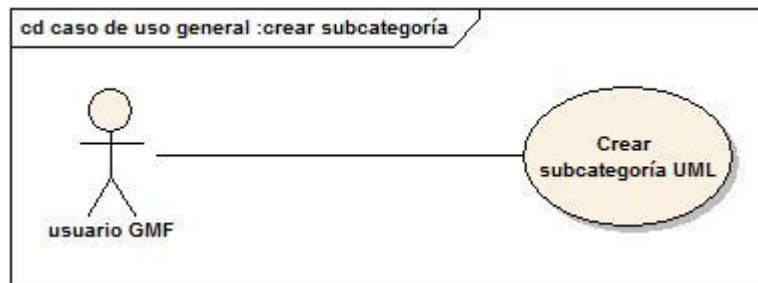


Ilustración 52 Caso de uso "Crear subcategoría"

6.2.3.14 Crear Producto UML

Nombre del Caso de Uso
Crear Producto UML
Descripción
El usuario GMF podrá crear una entidad gráfica UML de tipo producto con todos sus atributos

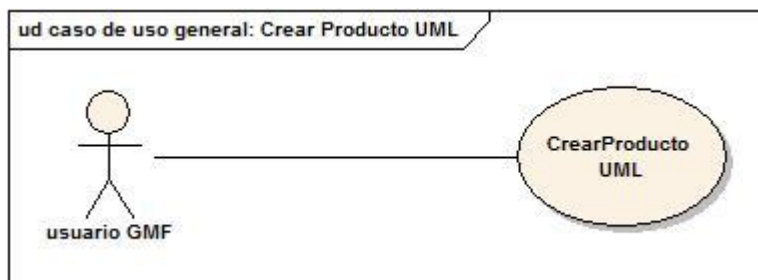


Ilustración 53 Caso de uso “Crear Producto UML”

6.2.3.15 Crear Proveedor UML

Nombre del Caso de Uso
Crear Proveedor UML
Descripción
El usuario GMF podrá crear una entidad gráfica UML de tipo proveedor con todos sus atributos

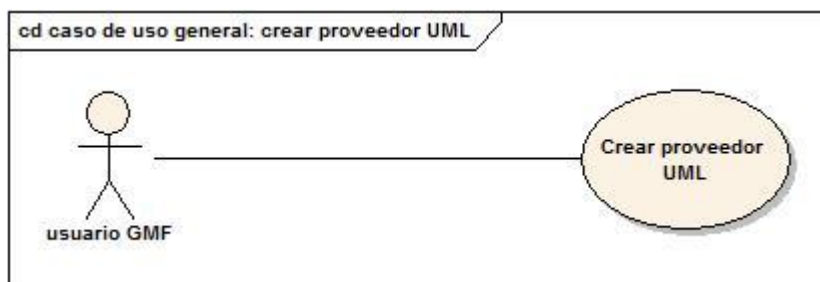


Ilustración 54 Caso de uso “Crear Proveedor UML”

6.2.3.16 Relacionar un producto con una o varias categorías

Nombre del Caso de Uso
Relacionar un producto con una o varias categorías UML
Descripción
El usuario GMF podrá relacionar en el GMF un producto creado con una o varias categorías mediante enlaces UML.

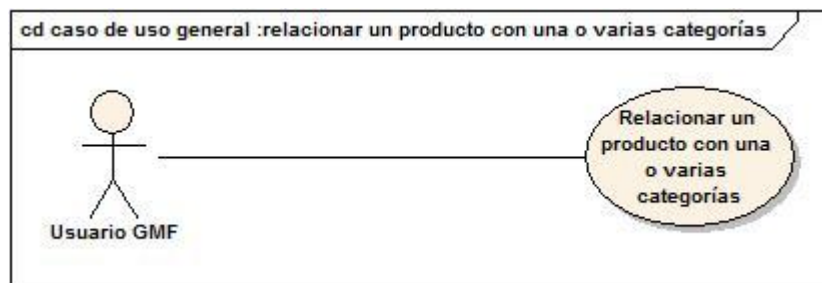


Ilustración 55 Caso de uso “Relacionar un producto con una o varias categorías”

6.2.3.17 Relacionar un producto con un proveedor

Nombre del Caso de Uso
Relacionar un producto con un proveedor
Descripción
El usuario GMF podrá relacionar en el GMF un producto creado con un único proveedor

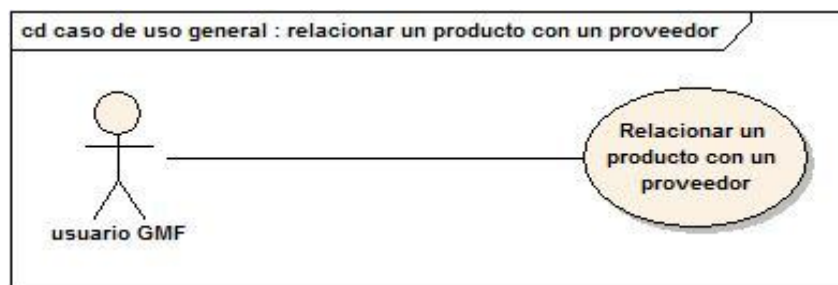


Ilustración 56 Caso de uso “Relacionar un producto con un proveedor”

6.2.3.18 Generar XML

Nombre del Caso de Uso
Generar XML
Descripción
Una vez que el usuario GMF crea el diagrama de clases UML éste podrá generar un XML con toda la información de las clases y relaciones entre ellas.

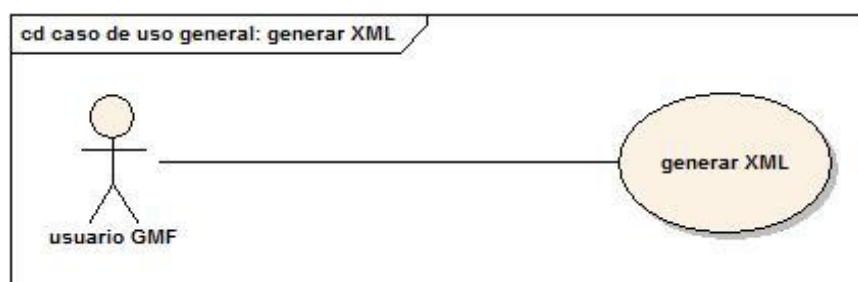


Ilustración 57 Caso de uso “Generar XML”

6.3 Identificación de los Subsistemas en la Fase de Análisis

El objetivo de esta sección es analizar el sistema para poder descomponerlo en sistemas más pequeños (subsistemas) que faciliten su posterior análisis.

6.3.1 Descripción de los Subsistemas

Los subsistemas que abarcan la total funcionalidad del proyecto son:

- DSL (Domain-Specific Lenguaje)
- Plataforma web

6.3.1.1 DSL

Este subsistema consiste en la construcción de un DSL (Domain-Specific Lenguaje) con GMF para la creación de elementos del catálogo de datos para plataformas de e-commerce (productos, categorías, proveedores...).

El GMF se desarrollará a partir de un metamodelo sobre la plataforma Eclipse y se apoya en los frameworks EMF (Eclipse Modeling Framework) y GEF (Graphical Editing Framework). Los pasos a seguir para la construcción del DSL son los siguientes:

- Definir el modelo de dominio (sintaxis abstracta).
- Definir el modelo gráfico (sintaxis concreta).
- Definir el modelo de herramientas.
- Definir el modelo de correspondencias.
- Incluir las restricciones en el modelo de correspondencias.
- Escribir transformaciones modelo-código. En este caso el código generado será XML.

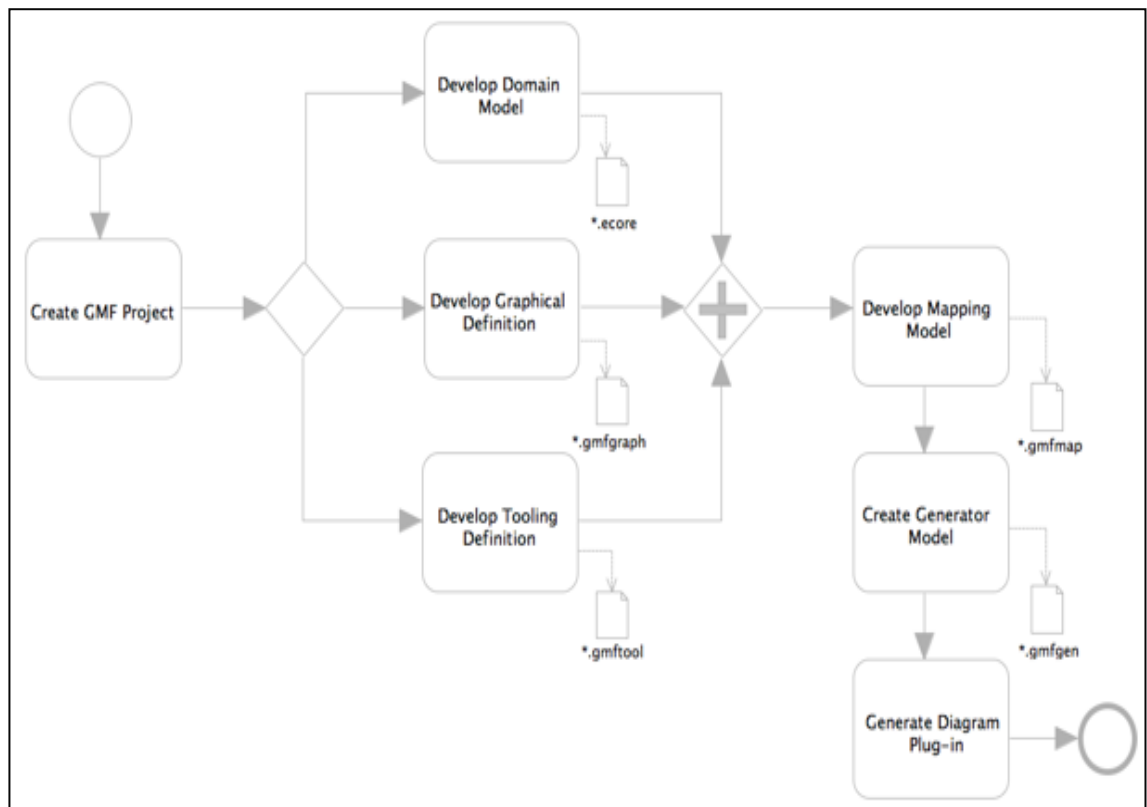


Ilustración 58 Pasos para la creación del GMF

6.3.1.2 Plataforma Web

El subsistema plataforma web consta de una aplicación web para la gestión del catálogo de datos e-commerce (categorías, productos, proveedores) y de usuarios. Este subsistema además será el encargado de intercomunicarse con Prestashop y Magento.

6.3.2 Descripción de los Interfaces entre Subsistemas

Los subsistemas anteriormente citados se comunican de forma local por lo que no será necesaria la definición de interfaces específicos. El XML generado por el DSL será importado directamente por el usuario a la plataforma web.

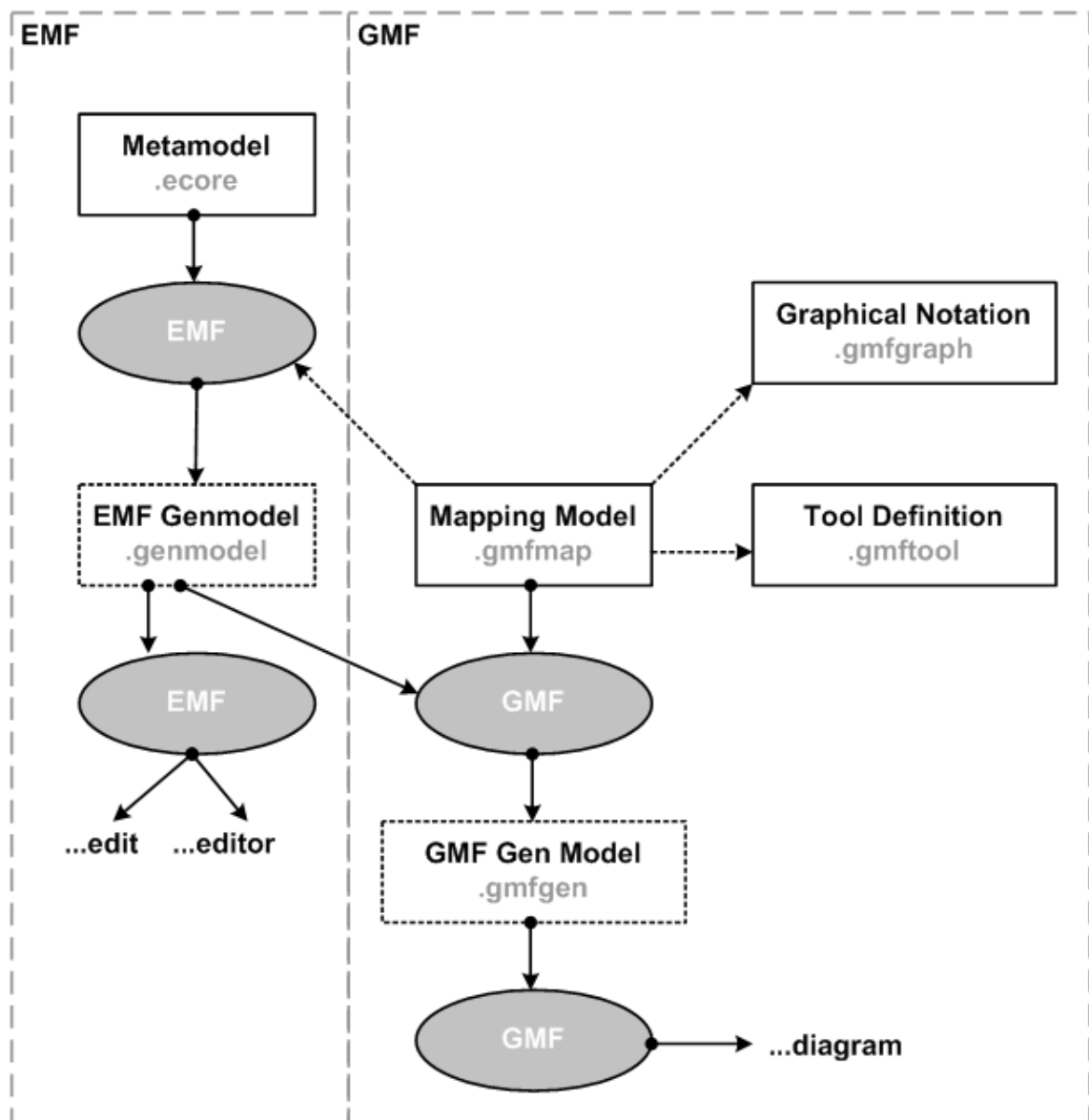
6.4 Diagrama de Clases Preliminar del Análisis

Dada la evidente diferenciación entre ambos subsistemas y su amplitud se optará a analizar cada uno por separado, a continuación se hará una breve descripción de los paquetes y las clases principales de cada uno de ellos.

6.4.1 Diagramas de Paquetes

6.4.1.1 DSL

En este caso no se define un diagrama de paquetes al uso sino que a continuación se muestra una imagen de las partes que formarán nuestro GMF.



6.4.1.2 Plataforma Web

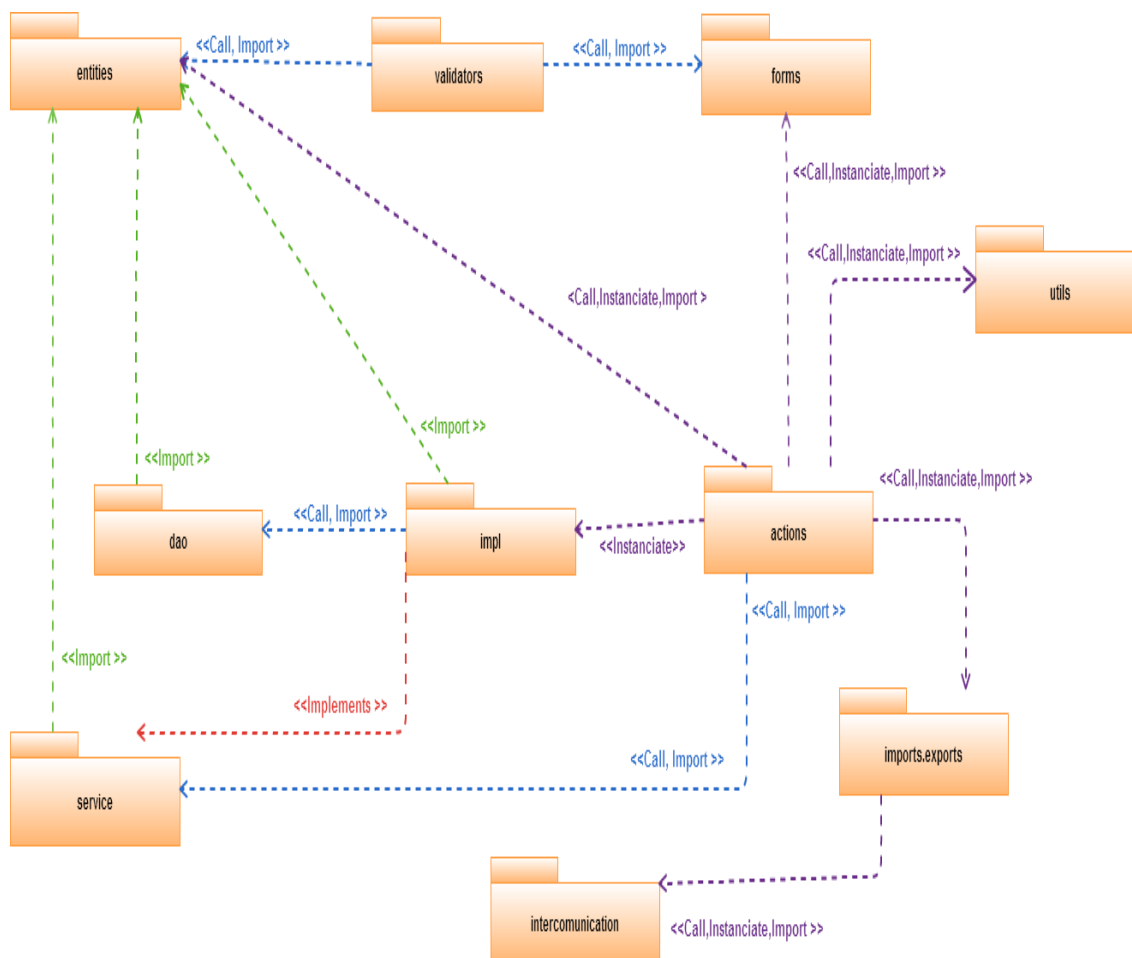


Ilustración 59 Diagrama de paquetes plataforma web

La estructura de desglose de paquetes está basada en el patrón de diseño MVC. El uso de este patrón mediante Spring Framework permite facilitar la identificación de funcionalidades de cada una de las clases que los componen.

Para ello se han definido las siguientes capas reconocidas en la estructura como paquetes Java que quedan definidas detalladamente a continuación:

Paquete	Descripción
com.tunos.commerce.entities	Contiene las clases java que representan el modelado orientado a objetos de las diferentes entidades del sistema. Integran también las anotaciones Hibernate que facilitarán el ORM de la persistencia de dichos objetos en la base de datos y sus relaciones.
com.tunos.commerce.dao	Contiene las interfaces de las clases que facilitarán el acceso a la persistencia. También las clases que implementan dichos métodos
com.tunos.commerce.service	Contiene las interfaces de los servicios que ofrecerá cada una de los manejadores de cada entidad. Serán consumidos por las acciones.
com.tunos.commerce.service.impl	Contiene la implementación de los métodos definidos por cada una de las interfaces el paquete service de cada manejador.
com.tunos.commerce.actions	Contiene las clases que definirán mediante sus métodos las diferentes acciones a realizar en el flujo de control con la interfaz de usuario. Los métodos de estas clases serán invocados desde la capa de flujo de SWF mediante estados de acción. Heredan los métodos de la clase de <code>org.springframework.webflow.action.MultiAction</code> .
com.tunos.commerce.forms	Contiene las clases que representan los objetos de presentación que recopilan la información (formularios, tablas ...) suministrada o consumida por el usuario.
com.tunos.commerce.validators	Contiene los métodos de validación de los formularios definidos en el paquete forms. Cada clase implementa los métodos definidos por la interfaz <code>org.springframework.validation.Validator</code> que facilitan las acciones de validación en el servidor.
com.tunos.commerce.intercommunication	Tiene los drivers de conexión establecer conexiones externas
com.tunos.commerce.intercommunication.imports.exports	Contiene las clases y métodos necesarios para la intercomunicación de la plataforma con Magento y Prestashop
com.tunos.commerce.utils	Contiene herramientas necesarias para la gestión de entidades que se reutilizarán en los diferentes paquetes.

6.4.2 Diagramas de clases

6.4.2.1 DSL

El meta-modelo propuesto para el DSL es el siguiente:

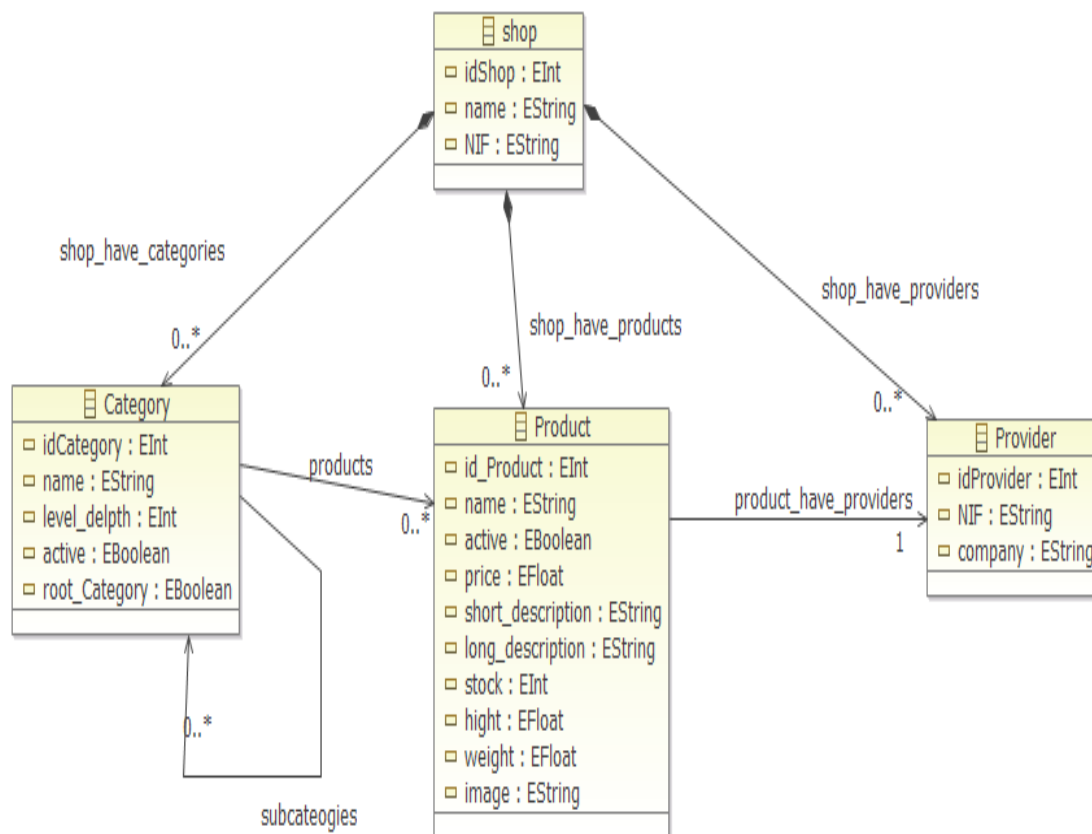


Ilustración 60 Meta-modelo

6.4.2.2 Plataforma Web

6.4.2.2.1 Diagrama de clases paquete entities

A continuación se muestra el contenido del paquete entities

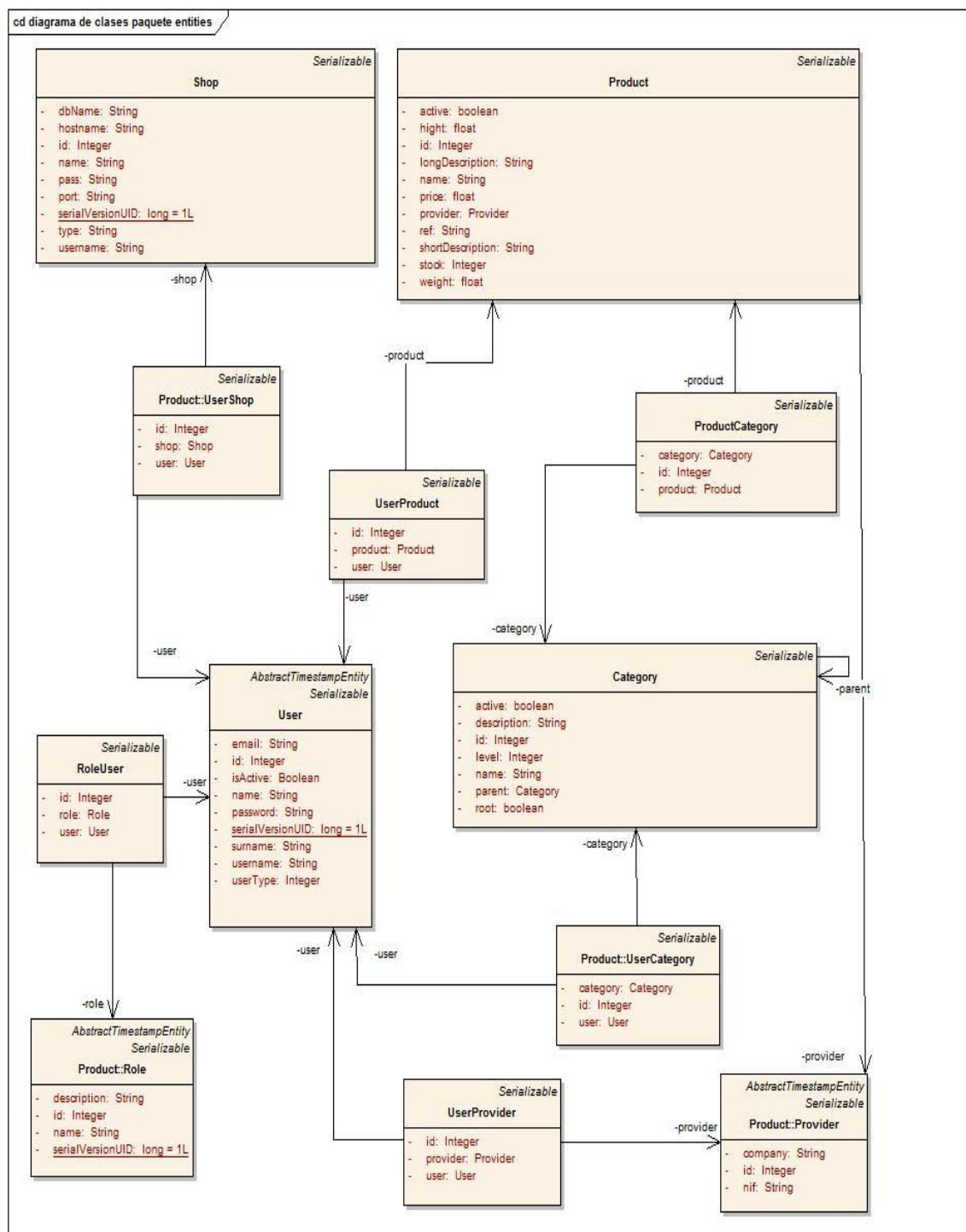


Ilustración 61 Diagrama de clases paquete entities

6.4.2.2.2 Diagrama de clases paquete DAO

A continuación se muestra el contenido de las clases del paquete DAO

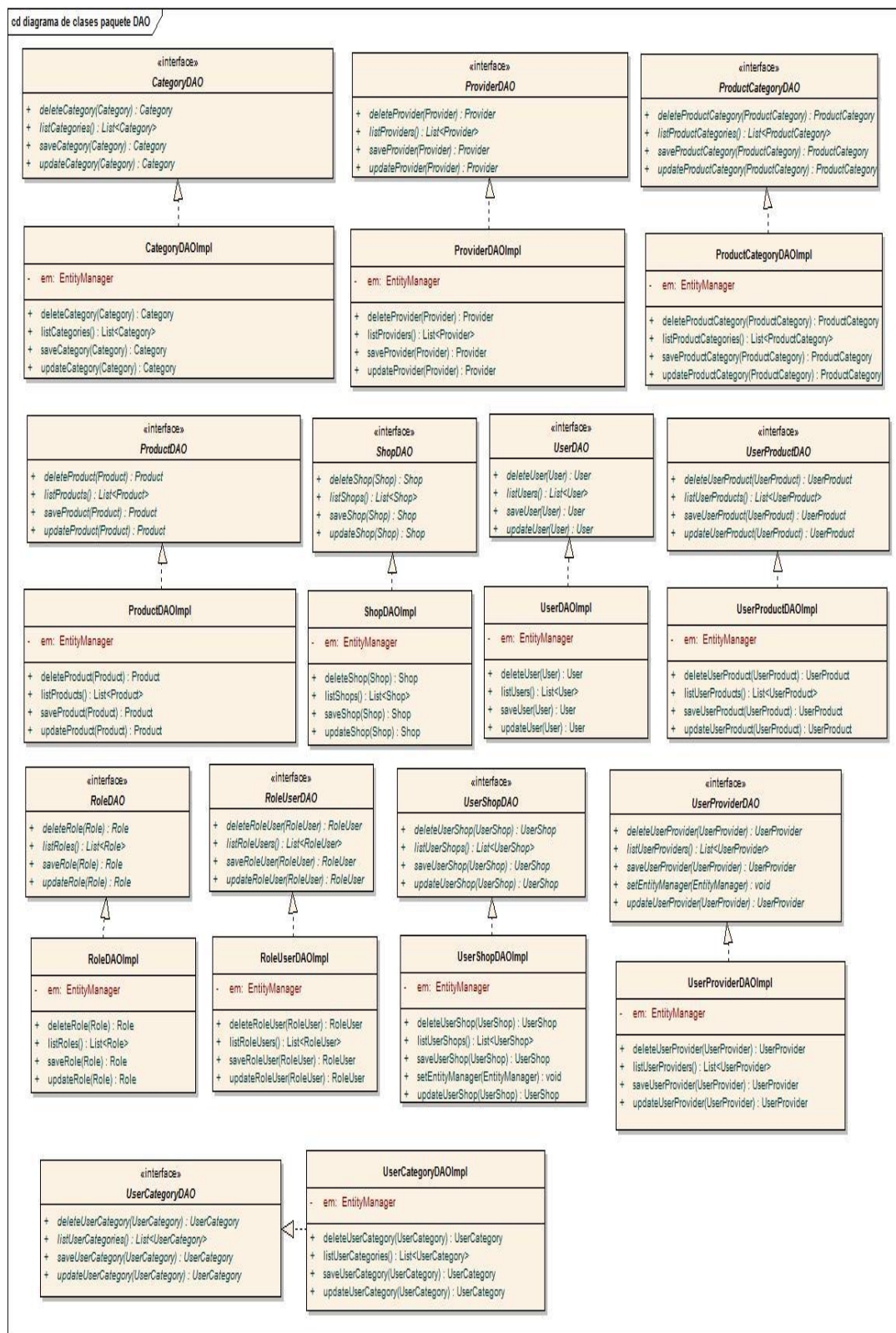


Ilustración 62 Diagrama de clases paquete DAO

6.4.2.2.3 Diagrama de clases paquete service

A continuación se muestra el contenido del paquete service



Ilustración 63 Diagrama de clases paquete service

6.4.2.2.4 Diagramas de clases paquete actions

A continuación se muestra el contenido del paquete actions.



Ilustración 64 Diagrama de clases paquete actions

6.4.2.2.5 Diagrama de clases paquete intercommunication

A continuación se muestra el contenido del paquete intercommunication

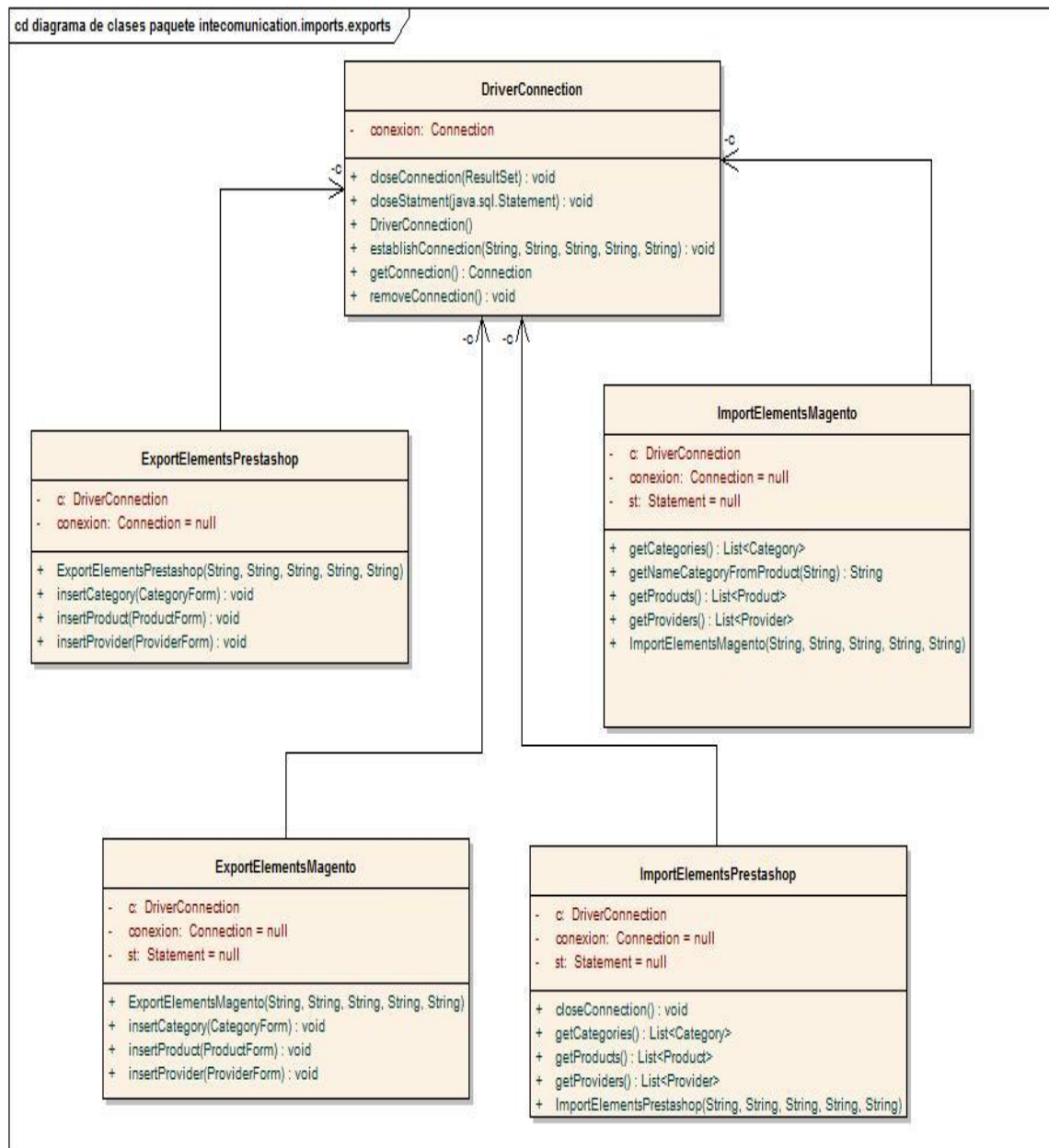


Ilustración 65 Diagrama de clases paquete intercommunication

6.4.3 Descripción de las Clases

6.4.3.1 Subsistema DSL

A continuación se describen las metaclases del metamodelo planteado para el DSL:

Nombre de la Clase
Shop
Descripción
Es una representación de la tienda
Responsabilidades
Es la Metaclass base y en ella se engloban los atributos necesarios y las relaciones con el resto de metaclasses.
Atributos Propuestos
idShop : entero que almacena el id de la tienda name : cadena que almacena el nombre de la tienda NIF : cadena que representa el NIF de la tienda

Nombre de la Clase
Category
Descripción
Es una representación de las categorías de la tienda
Responsabilidades
Clasificadora de los diferentes productos del catálogo de la tienda
Atributos Propuestos
idCategory : entero que almacena el id de la categoría name : cadena que almacena el nombre de la categoría active : booleano que indica si la categoría esta activa o desactiva root_category : booleano que sirve para indicar si la categoría es una categoría base o es una subcategoría

Nombre de la Clase
Product
Descripción
Es una representación de los productos de la tienda
Responsabilidades
Almacenar información de los productos de la tienda
Atributos Propuestos
Id_product : entero que almacena el id del producto Name : cadena que almacena el nombre del producto Active : booleano que indica si el producto está activo o desactivo Price : flotante que almacena el precio del producto Short_description : cadena que almacena una descripción corta acerca del producto Long_description : cadena que almacena una descripción más extensa del producto Stock : entero que indica el número de existencias del producto en la tienda Hight : flotante que indica lo que mide en altura el producto Weight : flotante que indica lo que mide en anchura el producto Image : cadena que almacena el path de la imagen del producto

Nombre de la Clase	
Provider	
Descripción	
Es una representación de los proveedores /fabricantes de la tienda	
Responsabilidades	
Almacenar información de los proveedores de la tienda	
Atributos Propuestos	
Id_provider: entero que almacena el id del proveedor NIF: cadena que almacena el NIF del proveedor Company: cadena que indica la compañía a la que pertenece	

6.4.3.2 Subsistema Plataforma Web

6.4.3.2.1 Paquete entities

Nombre de la Clase	
Shop	
Descripción	
Es una representación de la tienda	
Responsabilidades	
Clase encargada de representar y mapear entidades de tipo tienda	
Atributos Propuestos	
Id: identificador único de la tienda en la base de datos dbName: cadena que sirve para almacenar el nombre de la base de datos a la que es necesario conectarse para poder importar o exportar elementos hostname: cadena que almacena el nombre del host en el que se encuentra la base de datos de la tienda a configurar name: cadena que almacena el nombre de la tienda pass: cadena que representa el password de acceso a la base de datos de la tienda type: cadena que representa el tipo de la tienda. Ésta podrá ser de tipo Magento o Prestashop username: nombre de usuario para acceder a la base de datos de la tienda	

Nombre de la Clase	
Category	
Descripción	
Es una representación de las categorías de la tienda	
Responsabilidades	
Clase encargada de representar y mapear entidades de tipo categoría	
Atributos Propuestos	
id: identificador único de la categoría en la base de datos name: cadena que almacena el nombre de la categoría active: booleano que indica si la categoría esta activa o desactiva level: entero que representa el nivel de la categoría parent: almacena la relación con la categoría padre en caso de tenerlo root: booleano que indica si la categoría es una categoría base description: cadena que almacena la descripción de la categoría	

Nombre de la Clase
Product
Descripción
Es una representación de los productos de la tienda
Responsabilidades
Clase encargada de representar y mapear entidades de tipo producto
Atributos Propuestos
id: identificador único del producto en la base de datos name: cadena que almacena el nombre del producto active: booleano que indica si la producto está activo o desactivo weight: flotante que almacena el ancho del producto hight: Flotante que almacena la altura del producto logDescription: cadena que almacena una descripción larga del producto Price: flotante que almacena el precio del producto Provider: almacena la relación con un proveedor en caso de tenerlo Ref: cadena que almacena el número de referencia del producto shortDescription: cadena que almacena una breve descripción del producto stock: entero que almacena el número de unidades que hay en stock de ese producto

Nombre de la Clase
ProductCategory
Descripción
Es una representación de la relación entre productos y categorías
Responsabilidades
Clase encargada de representar y mapear entidades de tipo productCategory
Atributos Propuestos
Id: identificador único product: almacena la relación de un producto category: almacena la relación de una categoría

Nombre de la Clase
Provider
Descripción
Es una representación de un proveedor de la tienda
Responsabilidades
Clase encargada de representar y mapear entidades de tipo Provider
Atributos Propuestos
Id: identificador único Nif: cadena que representa el NIF del proveedor Company: cadena que representa la compañía a la que pertenece el proveedor

Nombre de la Clase
Role
Descripción
Es una representación de los tipos de roles que existentes en el sistema
Responsabilidades
Clase que representa el rol que puede ocupar un usuario dentro del sistema.
Atributos Propuestos
Id: identificador único Name: cadena que representa el nombre del rol Description: cadena para almacenar una pequeña descripción del significado del rol

Nombre de la Clase
RoleUser
Descripción
Es una representación de la relación entre roles y usuarios
Responsabilidades
Clase encargada de representar y mapear entidades de tipo RoleUser
Atributos Propuestos
Id: identificador único Role: almacena la relación con un rol User: almacena la relación con un usuario

Nombre de la Clase
User
Descripción
Clase que representa al usuario dentro del sistema. Se guardará toda la información necesaria sobre su identidad
Responsabilidades
Clase encargada de representar y mapear entidades de tipo User
Atributos Propuestos
Id: identificador único Name: almacena el nombre del usuario Surname: almacena los apellidos del usuario Email: cadena que almacena la dirección email del usuario Password: cadena para almacenar el password del usuario userType: entero que representa el tipo de un usuario isActive: boolean que indica si el usuario está activo o desactivo

Nombre de la Clase
UserCategory
Descripción
Es una representación de la relación entre usuarios y categorías
Responsabilidades
Clase encargada de representar y mapear entidades de tipo UserCategory
Atributos Propuestos
Id: identificador único user: almacena la relación de un usuario category: almacena la relación de una categoría

Nombre de la Clase
UserProduct
Descripción
Es una representación de la relación entre usuarios y productos
Responsabilidades
Clase encargada de representar y mapear entidades de tipo UserProduct
Atributos Propuestos
Id: identificador único user: almacena la relación de un usuario product: almacena la relación de una producto

Nombre de la Clase
UserProvider
Descripción
Es una representación de la relación entre usuarios y proveedores
Responsabilidades
Clase encargada de representar y mapear entidades de tipo UserProvider
Atributos Propuestos
Id: identificador único user: almacena la relación de un usuario provider: almacena la relación de un proveedor

Nombre de la Clase
UserShop
Descripción
Es una representación de la relación entre usuarios y tienda
Responsabilidades
Clase encargada de representar y mapear entidades de tipo UserShop
Atributos Propuestos
Id: identificador único user: almacena la relación de un usuario shop: almacena la relación de una tienda

6.4.3.2.2 Paquete DAO

En este paquete se almacenan los interfaces de persistencia de cada una de las clases anteriores. Inicialmente todas las interfaces constan de los métodos básicos de listado, almacenamiento, borrado y actualización de las entidades. A continuación se describe la clase ShopDAO que servirá de guía para el resto de las interfaces

Nombre de la Clase
ShopDAO
Descripción
Interfaz de los métodos de persistencia de la entidad <i>Shop</i>
Responsabilidades
Clase de definir los métodos de persistencia de la clase <i>Shop</i>
Métodos Propuestos
saveShop : método de hacer un persist de una entidad <i>Shop</i> que se le pasa como parámetro deleteShop : método encargado de eliminar de la base de datos la entidad <i>Shop</i> que se le pasa como parámetro updateShop : método encargado de actualizar en la base de datos los atributos correspondientes de la entidad <i>Shop</i> que se pasa como parámetro listShops : método encargado de listar todas las tiendas existentes en la base de datos

6.4.3.2.3 Paquete Service

En este paquete se definen los interfaces que darán servicio a cada uno de los DAO definidos. Inicialmente al igual que en paquete DAO se proponen únicamente los servicios básicos de listado, almacenamiento, borrado y actualización de las entidades. A continuación se describe la clase *ShopService* que servirá como guía para el resto de las entidades del paquete:

Nombre de la Clase
ShopService
Descripción
Interfaz de los servicios que ofrecerá el manejador de la entidad <i>Shop</i>
Responsabilidades
Clase de definir los servicios ofrecidos sobre la clase <i>Shop</i>
Métodos Propuestos
saveShop : método para almacenar entidades <i>shop</i> . deleteShop : método encargado de eliminar entidades <i>shop</i> . updateShop : método encargado de actualizar entidades <i>shop</i> . listShops : método encargado de listar todas las tiendas existentes en la base de datos

6.4.3.2.4 Paquete Actions

Nombre de la Clase
ShopAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad <i>Shop</i>
Responsabilidades
En esta clase se implementan el conjunto de acciones realizadas sobre la entidad Shop
Métodos Propuestos
deleteShop: recoge de la capa de presentación los datos y elimina la entidad shop invocando a los métodos del paquete cervices importShopElements: recoge de la capa de presentación la selección de elementos que el usuario desea importar y solicita los servicios del paquete intercomunicación listShopByUser: lista todas las tiendas creadas por el usuario y se las pasa a la capa de presentación por medio de las forms reviewShop: obtiene los detalles de la tienda solicitada y devuelve los datos a la capa de presentación saveShop: obtiene los datos proporcionados desde la capa de presentación e invoca los servicios necesarios para guardarla en la base de datos updateShop: obtiene los datos proporcionados desde la capa de presentación e invoca los servicios necesarios para actualizar los cambios en la base de datos

Nombre de la Clase
CategoryAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad <i>Category</i>
Responsabilidades
En esta clase se implementan el conjunto de acciones realizadas sobre la entidad Category
Métodos Propuestos
deleteCategory : recoge de la capa de presentación los datos y elimina la entidad Category junto con sus relaciones invocando a los métodos del paquete services exportCategory: se encarga de obtener los datos de presentación e invocar a los métodos necesarios del paquete intercommunication para que se realice la exportación. listAllCategoriesByUser: obtiene la lista de categorías de un usuario y lo pasa a la capa de presentación reviewCategory: obtiene los detalles de una categoría saveCategory: recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para almacenar los datos updateCategory: recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para actualizar los datos

Nombre de la Clase
ProductAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad <i>Product</i>
Responsabilidades
En esta clase se implementan el conjunto de acciones realizadas sobre la entidad Product
Métodos Propuestos
deleteProduct :recoge de la capa de presentación los datos y elimina la entidad Product junto con sus relaciones invocando a los métodos del paquete services exportProduct : se encarga de obtener los datos de presentación e invocar a los métodos necesarios del paquete intercommunication para que se realice la exportación. listAllCategoriesProduct : obtiene la lista de categorías a las que pertenece un producto listAllProductsByUser : lista el conjunto de productos pertenecientes a un usuario reviewProduct : obtiene los detalles de un producto saveProduct : recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para almacenar los datos updateProduct : recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para actualizar los datos

Nombre de la Clase
ProviderAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad <i>Provider</i>
Responsabilidades
En esta clase se implementan el conjunto de acciones realizadas sobre la entidad Provider
Métodos Propuestos
deleteProvider :recoge de la capa de presentación los datos y elimina la entidad Provider junto con sus relaciones invocando a los métodos del paquete services exportProvider : se encarga de obtener los datos de presentación e invocar a los métodos necesarios del paquete intercommunication para que se realice la exportación. getProduct : obtiene un determinado producto desde la capa de negocio y la pasa a presentación listAllProvidersByUser : lista el conjunto de productos pertenecientes a un usuario saveProvider : recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para almacenar los datos updateProvider : recoge de la capa de presentación los datos e invoca los métodos necesarios de la capa service para actualizar los datos

Nombre de la Clase
ParserAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control para parsear el XML y guardar los datos
Responsabilidades
En esta clase se implementan el conjunto de acciones para parsear y guardar los datos del XML
Métodos Propuestos
uploadFile :recoge el XML, parsea y llama a los servicios necesarios para guardar los datos

Nombre de la Clase
UserAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad <i>User</i>
Responsabilidades
En esta clase se implementan el conjunto de acciones realizadas sobre la entidad User
Métodos Propuestos
createUser: recoge los datos de la interfaz y llama a los servicios necesarios para registrar un usuario. La clave del usuario se almacena aplicando el algoritmo SHA-1 (Secure Hash Algorithm)
updateUser: recoge los datos de la interfaz y llama a los servicios necesarios para actualizar la información del usuario.

Nombre de la Clase
Generate CSV
Descripción
Clase que implementa los métodos necesarios para generar las acciones necesarias para crear un archivo CSV con los productos del usuario
Responsabilidades
En esta clase se implementan el conjunto de acciones para crear archivos CSV
Métodos Propuestos
getProducts: obtiene los productos de un usuario
generateCSV: con el conjunto de productos genera un archivo CSV compatible con las plataformas Magento y Prestashop

Nombre de la Clase
QRCodeAction
Descripción
Clase que implementa los métodos necesarios para generar las acciones necesarias para crear un QR con la información de los productos
Responsabilidades
En esta clase se implementan el conjunto de acciones para crea QR's
Métodos Propuestos
generateQRCode: con la información de un producto genera un código QR.

6.4.3.2.5 Paquete intercommunication

Nombre de la Clase
DriverConnection
Descripción
Clase encargada de establecer las conexiones externas para poder realizar la intercomunicación
Responsabilidades
Establecer y cerrar conexiones Mysql
Métodos Propuestos
closeConnection: cierra una conexión abierta closeStatment: cierra un statment abierto establishConnection: establece una conexión mysql con una plataforma externa getConnection: devuelve la conexión activa removeConnection: elimina la conexión

Nombre de la Clase
ExportElementsPrestashop
Descripción
Clase encargada hacer las exportaciones a prestashop
Responsabilidades
Hacer exportaciones hacia prestashop
Atributos Propuestos
DriverConnection: almacena una instancia a la clase DriverConection encargada de gestionar las conexiones mysql
Métodos Propuestos
insertCategory: exporta una categoría a prestashop. Además de exportar la propia categoría exporta el nivel jerárquico en el que se encuentra insertProduct: exporta un producto a prestashop. Además de exportar la información del propio producto exporta las relaciones con sus categorías y proveedores insertProvider: exporta la información de un proveedor almacenado en la plataforma a la plataforma prestashop

Nombre de la Clase
ExportElementsMagento
Descripción
Clase encargada hacer las exportaciones a Magento
Responsabilidades
Hacer exportaciones hacia Magento
Atributos Propuestos
DriverConnection: almacena una instancia a la clase DriverConection encargada de gestionar las conexiones mysql
Métodos Propuestos
insertCategory: exporta una categoría a Magento. Además de exportar la propia categoría exporta el nivel jerárquico en el que se encuentra insertProduct: exporta un producto a Magento. Además de exportar la información del propio producto exporta las relaciones con sus categorías y proveedores insertProvider: exporta la información de un proveedor almacenado en la plataforma a la plataforma Magento

Nombre de la Clase
ImportElementsPrestashop
Descripción
Clase encargada hacer las importaciones desde prestashop
Responsabilidades
Hacer importaciones desde prestashop para almacenarlas en la base de datos
Atributos Propuestos
DriverConnection: almacena una instancia a la clase DriverConection encargada de gestionar las conexiones mysql
Métodos Propuestos
getCategories: obtiene la información de las categorías de prestashop para almacenarlas en nuestro sistema. La importación se hace de forma jerárquica
getProducts: obtiene la información de los productos de prestashopp para almacenarlos en nuestro sistema. Al hacer la importación también se traen las relaciones entre productos , categorías y proveedores
getProviders: obtiene la información de los proveedores de prestashop para almacenarlos en nuestro sistema

Nombre de la Clase
ImportElementsMagento
Descripción
Clase encargada hacer las importaciones desde Magento
Responsabilidades
Hacer importaciones desde Magento para almacenarlas en la base de datos
Atributos Propuestos
DriverConnection: almacena una instancia a la clase DriverConection encargada de gestionar las conexiones mysql
Métodos Propuestos
getCategories: obtiene la información de las categorías de Magento para almacenarlas en nuestro sistema. La importación se hace de forma jerárquica
getProducts: obtiene la información de los productos de Magento para almacenarlos en nuestro sistema. Al hacer la importación también se traen las relaciones entre productos , categorías y proveedores
getProviders: obtiene la información de los proveedores de Magento para almacenarlos en nuestro sistema

6.5 Análisis de Casos de Uso y Escenarios

6.5.1 Caso de uso: Registro de usuarios

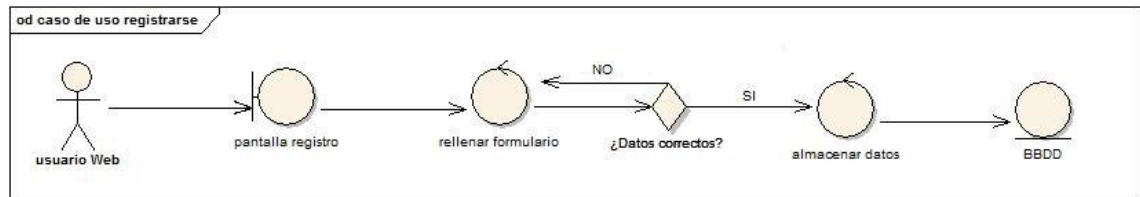


Ilustración 66 Diagrama de robustez para el caso de uso "Registrar usuario"

Registrar usuario	
Precondiciones	El usuario no debe estar registrado ya en el sistema
Poscondiciones	- Se guardarán los datos del usuario en la base de datos. - La clave se almacenará codificada en SHA-1 - Se anunciará al usuario que la operación se ha realizado correctamente
Actores	Usuario Web
Descripción	El usuario rellenará el formulario de registro de la web. Si todos los datos son correctos se guardarán los datos en la base de datos y se informará al usuario que todo ha ido correctamente. Para que el formulario esté correctamente cumplimentado se deben cumplir las siguientes condiciones: - Deben rellenarse los campos obligatorios - El nombre de usuario no debe estar repetido en la base de datos - El email debe estar correctamente formado - La contraseña y la confirmación de la contraseña deben coincidir
Excepciones	Existe algún problema de acceso a la base de datos y estos no pueden guardarse correctamente.

6.5.2 Caso de uso: Configurar tienda

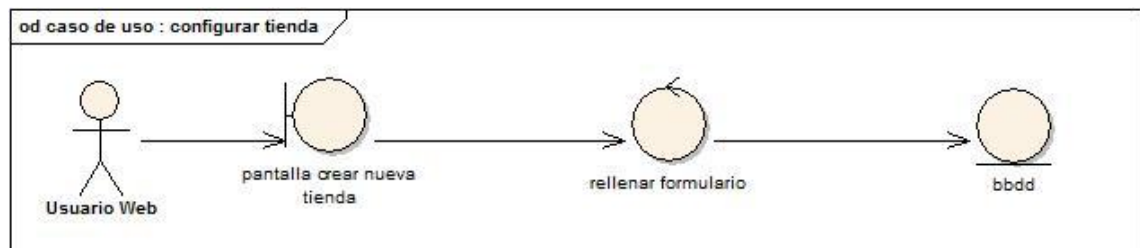


Ilustración 67 Diagrama de robustez para el caso de uso "Configurar tienda"

Configurar tienda	
Precondiciones	• El usuario debe estar registrado en el sistema • El usuario debe estar logueado • El usuario debe poseer una tienda virtual Prestahop o Magento • La base de datos de la plataforma con la que se quiere intercambiar información debe estar accesible desde el exterior de la máquina donde está desplegada.
Poscondiciones	La configuración de la tienda se almacenará en la base de datos. Las contraseñas de la base de datos de la plataforma a configurar se encriptarán mediante el cifrado XOR.
Actores	Usuario Web
Descripción	El usuario rellenará el formulario de registro de la web. Configuración de la tienda. Para ello necesitará saber los siguientes datos: • El nombre de la base de dato de la plataforma con la que se va a interoperar • El host donde está alojada • El número de puerto • El nombre de usuario y contraseña para poder acceder
Excepciones	Que los datos de configuración introducidos por el usuario no sean correctos por lo que no se podrá hacer importaciones ni exportaciones de elementos en el futuro.

6.5.3 Caso de uso: Importar elementos

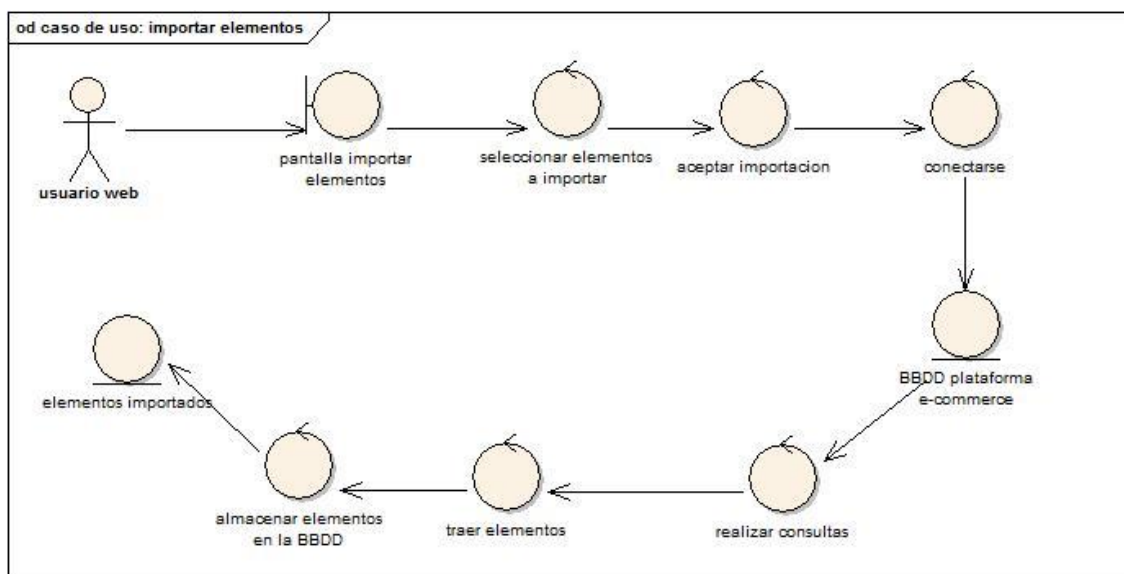


Ilustración 68 Diagrama de robustez para el caso de uso "Importar elementos"

Importar elementos	
Precondiciones	- El usuario debe estar registrado en el sistema - El usuario debe estar logueado - El usuario debe poseer una tienda virtual Prestahop o Magento - La base de datos de la plataforma con la que se quiere intercambiar información debe estar accesible desde el exterior de la máquina donde está desplegada.
Poscondiciones	Se obtendrán los datos provenientes de la base de datos de la plataforma y se almacenarán en la base de datos de nuestra plataforma.
Actores	Usuario Web
Descripción	El usuario seleccionará que tipos de elementos desea importar (Categorías, productos o proveedores) A continuación se importarán los elementos y las relaciones existentes entre ellos.
Excepciones	- Que la tienda no esté correctamente configurada y no se pueda acceder a la base de datos de la plataforma. - Que el usuario haya manipulado la base de datos de la plataforma y no coincidan los nombres de las tablas o sus campos.

6.5.4 Caso de uso: Importar XML

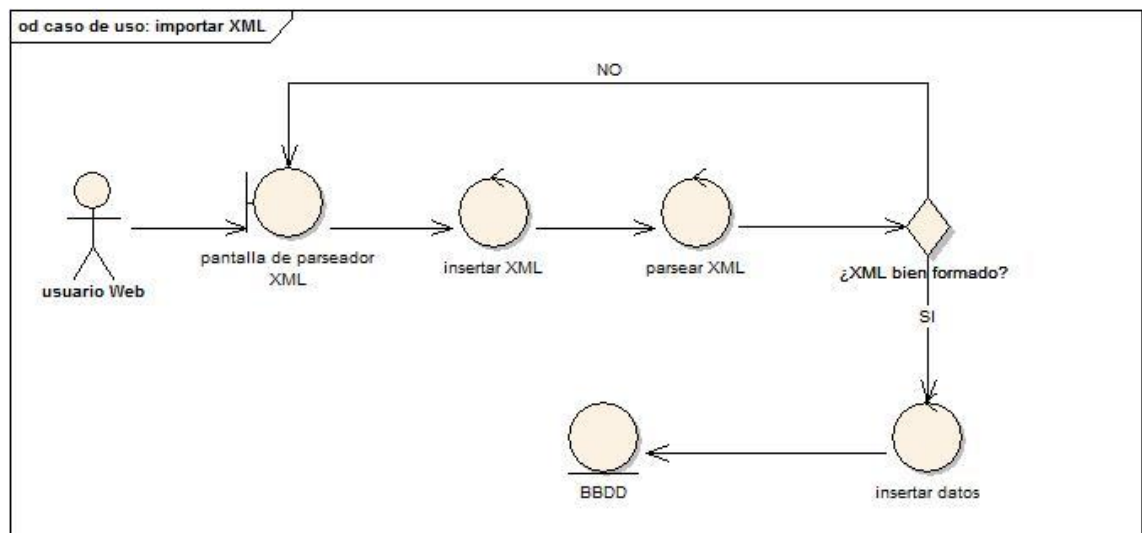


Ilustración 69 de robustez para el caso de uso "Importar XML"

Importar XML	
Precondiciones	- El usuario debe estar registrado en el sistema - El usuario debe estar logueado - Se debe haber generado un XML con el GMF previamente o en su defecto conocer como es la estructura del XML a parsear.
Poscondiciones	La información del XML estará almacenada en el sistema.
Actores	Usuario Web
Descripción	Se parseará el XML, en caso de estar bien formado la información que contiene será insertada en la base de datos de la plataforma. En caso de

	ser incorrecta se avisará al usuario.
Excepciones	Que el XML no esté correctamente formado con lo cual no podrá hacerse el insertado de la información en la base de datos

6.5.5 Caso de uso: exportar elementos

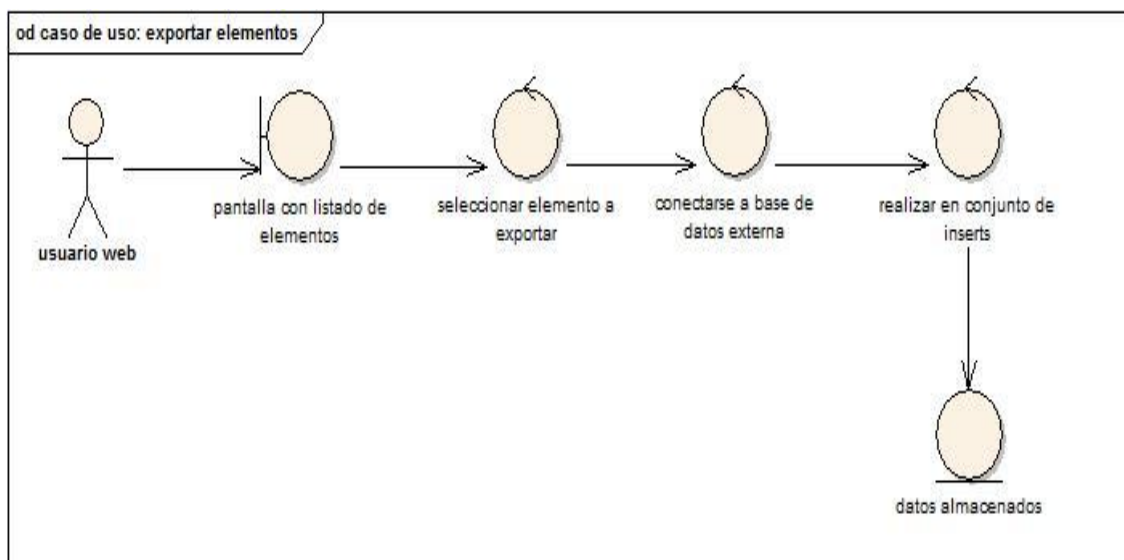


Ilustración 70 Diagrama de robustez para el caso de uso “Exportar elementos”

Exportar elementos	
Precondiciones	- El usuario debe estar registrado en el sistema - El usuario debe estar logueado - El usuario debe poseer una tienda virtual Prestahop o Magento - El usuario debe haber creado algún elemento para exportar. - La base de datos de la plataforma con la que se quiere intercambiar información debe estar accesible desde el exterior de la máquina donde está desplegada.
Poscondiciones	Se abrirá una conexión con la base de datos externa y se insertará el elemento seleccionado en las tablas correspondientes. Finalmente el elemento estará exportado y visible desde el administrador de la plataforma con la que se realizó la exportación
Actores	Usuario Web
Descripción	El usuario seleccionará el elemento que se desea exportar y creará este mismo elemento en la plataforma con la cual nos estamos comunicando.
Excepciones	- Que la tienda no esté correctamente configurada y no se pueda acceder a la base de datos de la plataforma. - Que el usuario haya manipulado la base de datos de la plataforma y no coincidan los nombres de las tablas o sus campos. - Que los datos insertados no sean los correctos y el elemento no sea visible desde el panel de administración de la plataforma con la que se realizó la exportación

6.5.6 Caso de Uso: Generar CSV

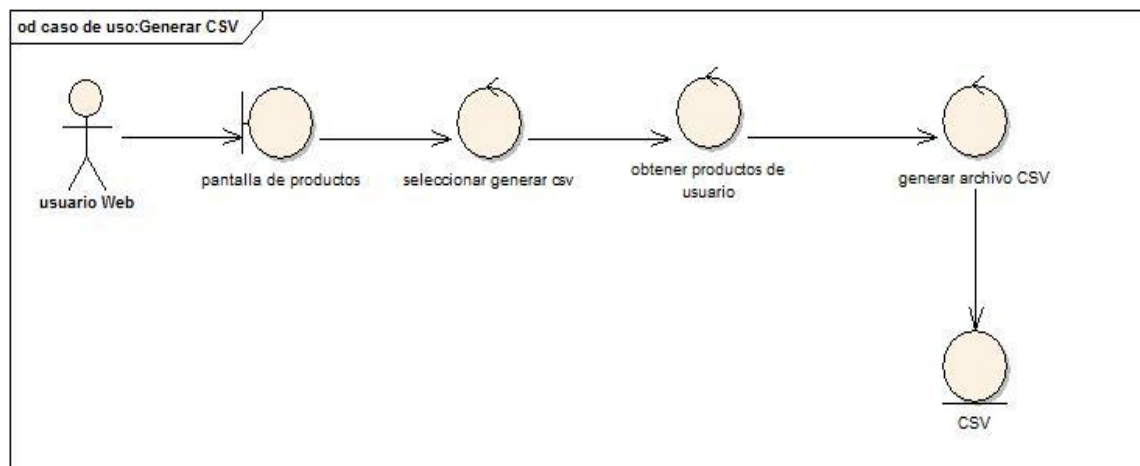


Ilustración 71 Diagrama de robustez para el caso de uso "Generar CSV"

Generar CSV	
Precondiciones	- El usuario debe estar registrado en el sistema - El usuario debe estar logueado
Poscondiciones	Se genera un archivo CVS con el catálogo de productos del usuario. Este fichero CVS será compatible con la plataforma Magento y Prestashop
Actores	Usuario Web
Descripción	Una vez que el usuario selecciona "Generar CVS" se obtiene la lista de productos y se crea un archivo CSV compatible con las plataformas Prestashop y Magento.
Excepciones	Que el CVS no pueda generarse correctamente.

6.5.7 Caso de uso: generar XML

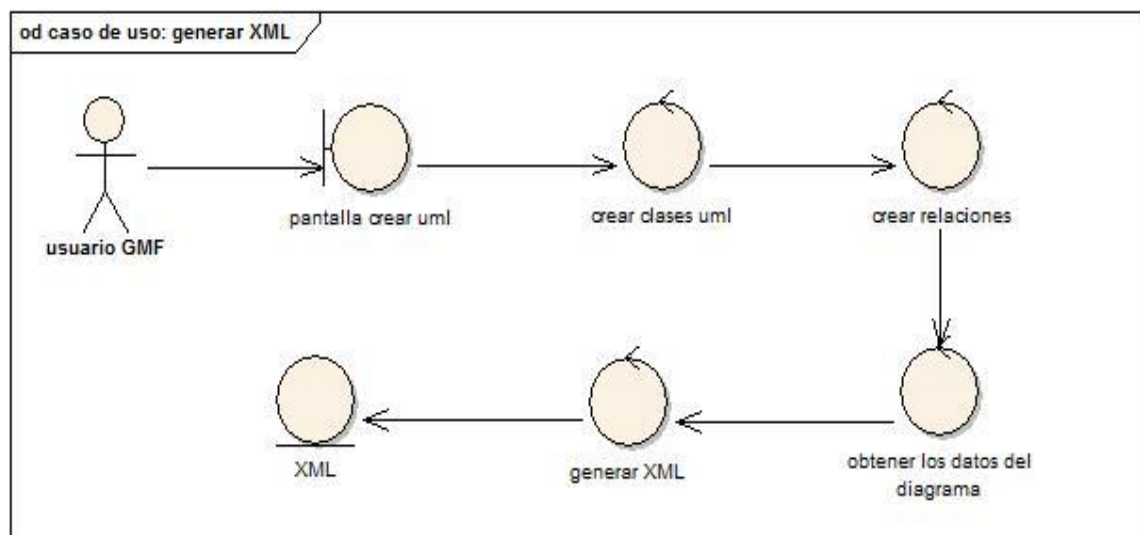


Ilustración 72 Diagrama de robustez para el caso de uso "Generar XML"

Generar XML	
Precondiciones	El usuario debe haber creado el UML con las entidades en el GMF
Poscondiciones	Se genera un archivo XML recogiendo toda la información del diagrama
Actores	Usuario GMF
Descripción	Una vez que el usuario desarrolló el UML en el GMF se genera un XML que recoge toda esta información
Excepciones	Que el usuario no genere un diagrama UML correcto y falle al intentar generar el XML

6.6 Análisis de Interfaces de Usuario

El sistema completo poseerá dos interfaces de interacción con usuario:

- Plataforma Web
- GMF

A continuación se definirán las pantallas de ambos subsistemas por separado.

6.6.1 Pantallas Plataforma Web

6.6.1.1 Pantalla de inicio de sesión

Pantalla inicial en la cual el usuario en caso de estar registrado podrá iniciar sesión o si no es así acceder a la pantalla de registro.

Diagrama de la interfaz de usuario para la pantalla de inicio de sesión. La interfaz se muestra dentro de un navegador web con la barra de direcciones que indica 'A Web Page' y la URL 'http://unoscomerce.com'. El contenido principal de la página es un formulario centrado con el título 'Bienvenido'. El formulario tiene un encabezado 'Iniciar Sesión' y contiene los siguientes elementos:

- Un campo de texto etiquetado 'Usuario'.
- Un campo de texto etiquetado 'Contraseña'.
- Un botón 'Registrarse'.
- Un botón 'Entrar'.
- Un checkbox etiquetado 'Recordar mis datos'.

Ilustración 73 Pantalla inicio de sesión

6.6.1.2 Pantalla de registro

Pantalla desde la cual el usuario podrá registrarse en el sistema. Para ello deberá rellenar un formulario y darle al botón "registrar". Si todos los datos son correctos una vez el usuario haya sido registrado ya podrá iniciar sesión.

The illustration shows a web browser window titled "A Web Page" with the address bar displaying "http://tunoscommerce.com". The main content area contains a registration form titled "Registrar nuevo usuario". The form includes the following fields and buttons:

- Nombre de usuario
- Nombre
- Apellidos
- E-mail
- Contraseña
- Repita la Contraseña
- Registrar
- Cancelar

Ilustración 74 Pantalla de registro

6.6.1.3 Pantalla de Intro

Es la pantalla que se carga inicialmente tras haberse logueado. En esta pantalla se expone a nivel informativo datos sobre la aplicación y las plataformas con las que es posible interoperar. Desde aquí a través de un menú horizontal se podrá acceder al resto de la aplicación.



Ilustración 75 Pantalla de intro

6.6.1.4 Pantallas de Categorías

Pantalla principal donde se muestran todas las categorías del usuario. Desde aquí se podrán hacer todas las operaciones posibles en relación a Categorías (crear, modificar, buscar, eliminar...)

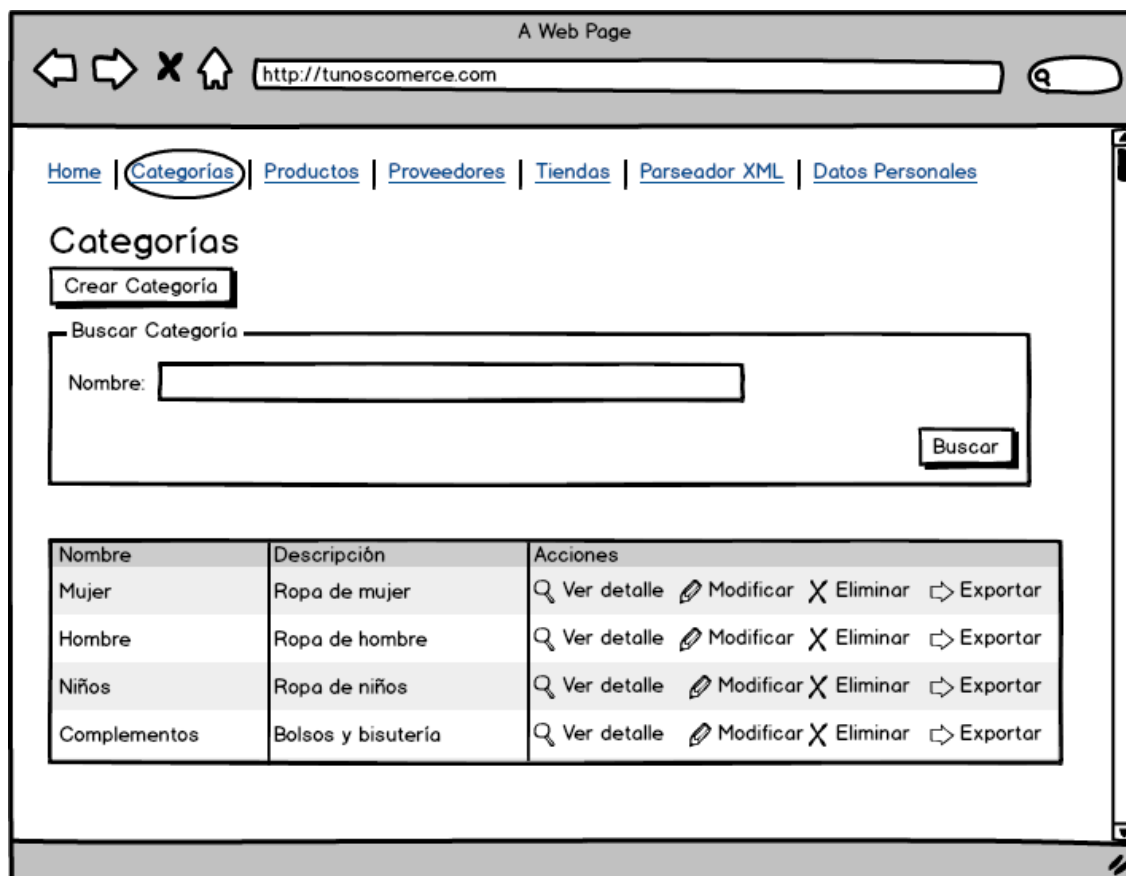


Ilustración 76 Pantalla menú categorías

6.6.1.4.1 Crear Categoría

Formulario para crear categorías. Aquí se especificarán los datos necesarios para crear la categoría así como la jerarquización de esta. Si la categoría a crear es una categoría base esta se deja sin categoría padre, por el contrario si es una subcategoría se especificará la categoría padre.

A Web Page

http://tunoscommerce.com

Home | **Categorías** | Productos | Proveedores | Tiendas | Parseador XML | Datos Personales

Crear Categoría

Nombre:

Descripción:

Categoría Padre: Sin Padre

☐ Activa

Ilustración 77 Crear categoría

En la siguiente ventana se puede seleccionar el padre de la categoría.

A Web Page

http://tunoscommerce.com

Home | Categorías | **Productos** | Proveedores | Tiendas | Parseador XML | Datos Personales

Seleccionar Categoría Padre

Lista de Categorías disponibles

Nombre	Descripción	Acciones
Mujer	Ropa de mujer	Seleccionar
Hombre	Ropa de hombre	Seleccionar
Niños	Ropa de niños	Seleccionar
Complementos	Bolsos y bisutería	Seleccionar

Ilustración 78 Seleccionar categoría padre

6.6.1.4.2 Modificar categoría

Formulario para modificar la categoría

A Web Page

http://tunoscommerce.com

Home | **Categorías** | Productos | Proveedores | Tiendas | Parseador XML | Datos Personales

Modificar Categoría

Nombre: Categoría 1

Descripción: Descripción de la categoría

Categoría Padre: Categoría0 **Modificar Padre**

☐ Activa

Modificar **Cancelar**

Ilustración 79 modificar Categoría

6.6.1.4.3 Eliminar categoría

Formulario para eliminar la categoría.

A Web Page

http://tunoscommerce.com

Home | **Categorías** | Productos | Proveedores | Tiendas | Parseador XML | Datos Personales

Confirma que desea eliminar la Categoría 1

Nombre: Categoría 1

Descripción: Descripción de la categoría

Categoría Padre: Categoría0

Estado: Activa

Eliminar **Cancelar**

Ilustración 80 Eliminar Categoría

6.6.1.4.4 Ver detalle de categoría

Opción ver el detalle de una categoría.

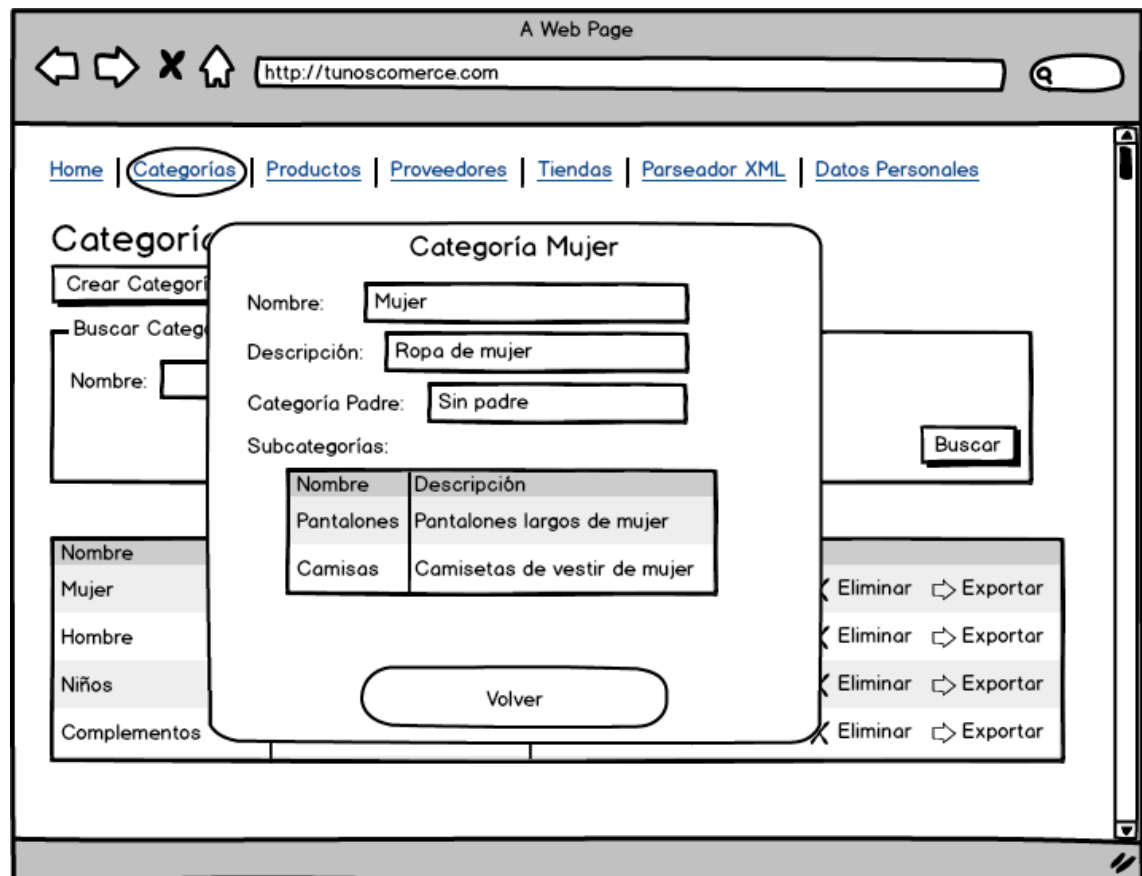


Ilustración 81 Ver detalle de categoría

6.6.1.4.5 Exportar categoría

Formulario de exportación de una categoría

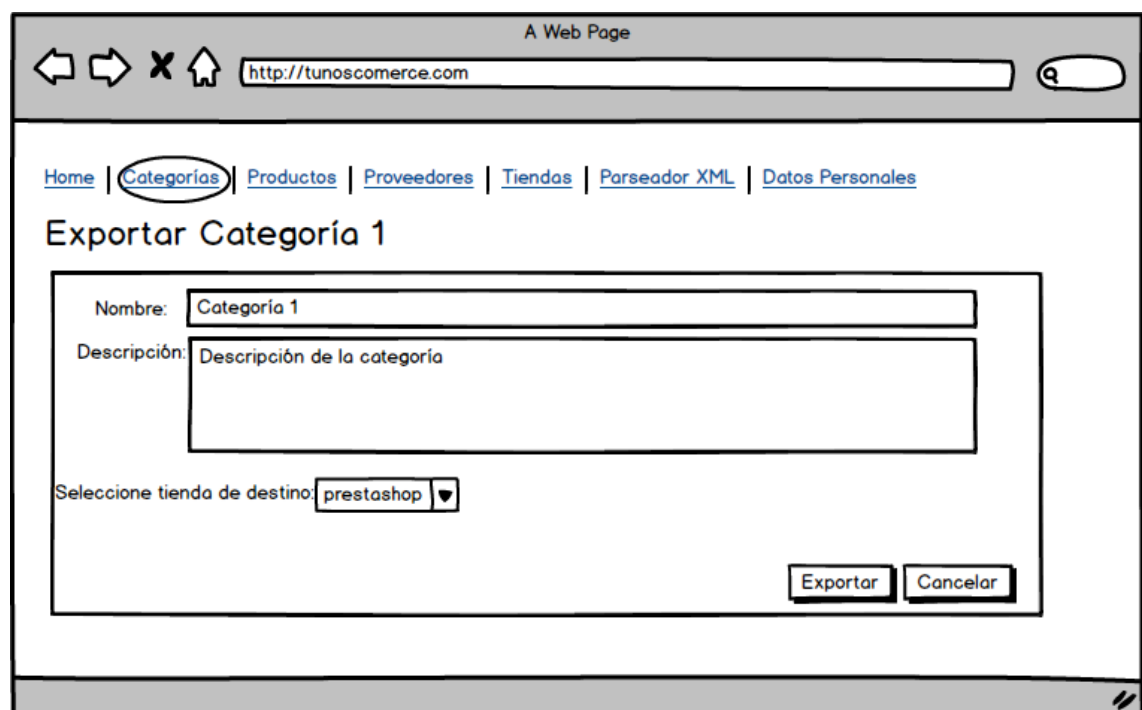


Ilustración 82 Exportar categoría

6.6.1.5 Pantallas de Productos

Pantalla principal donde se muestran todos los productos del usuario. Desde aquí se podrán hacer todas las operaciones posibles en relación a los productos (crear, modificar, buscar, eliminar...)

6.6.1.5.1 Pantalla crear producto

En esta pantalla se introducirán todos los datos relacionados con el producto. Además se podrá seleccionar un proveedor y una o varias categorías con las que el producto se podrá relacionar.

The screenshot shows a web browser window titled 'A Web Page' with the address bar displaying 'http://tunoscommerce.com'. The navigation menu includes links for Home, Categorías, Productos (highlighted), Proveedores, Tiendas, Parseador XML, and Datos Personales. The main heading is 'Crear producto'. The form contains the following fields and controls:

- Nombre: Input field with 'Producto 1'.
- Descripción Corta: Input field with 'Descripción corta del producto...'.
- Descripción Larga: Textarea with 'Descripción larga del producto'.
- Precio: Input field with '12 €'.
- Alto: Input field with '120 cm'.
- Ancho: Input field with '90 cm'.
- Proveedor: Label 'Sin proveedor' followed by a 'Seleccionar' button.
- Categorías: Label 'Sin categoría' followed by a 'Seleccionar' button.
- ☐ Activo
- Buttons: 'Crear' and 'Cancelar' at the bottom right.

Ilustración 83 Pantalla Crear Producto

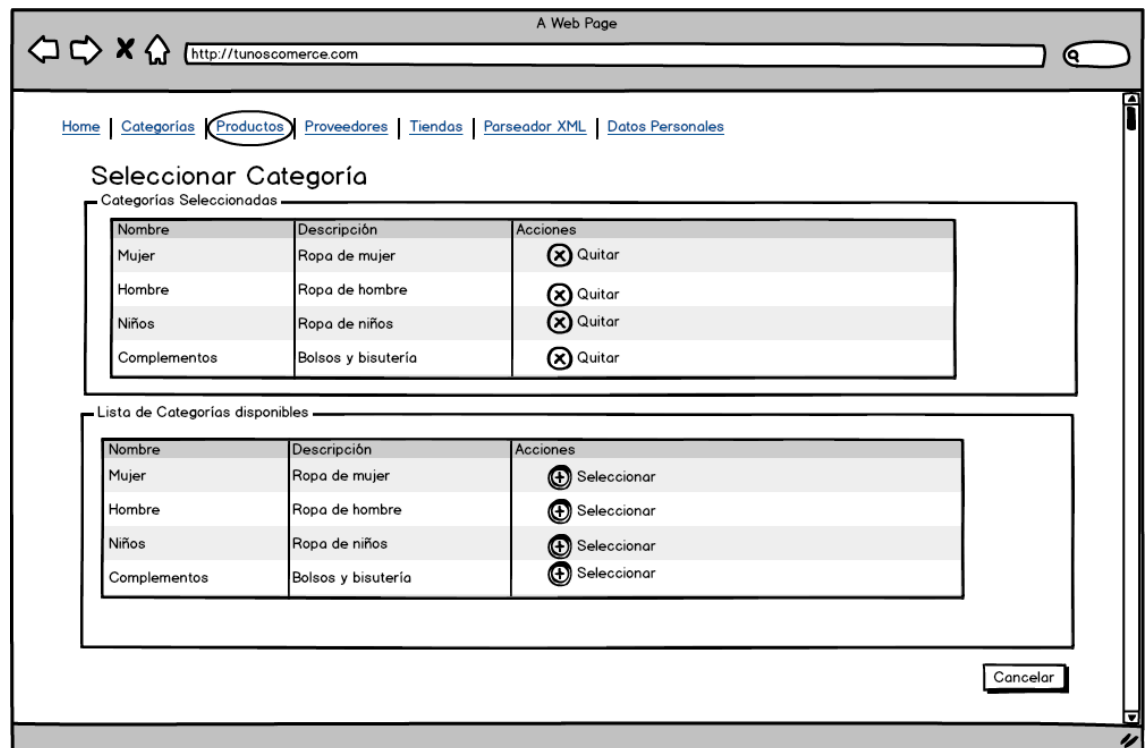


Ilustración 84 Pantalla seleccionar una o varias categorías

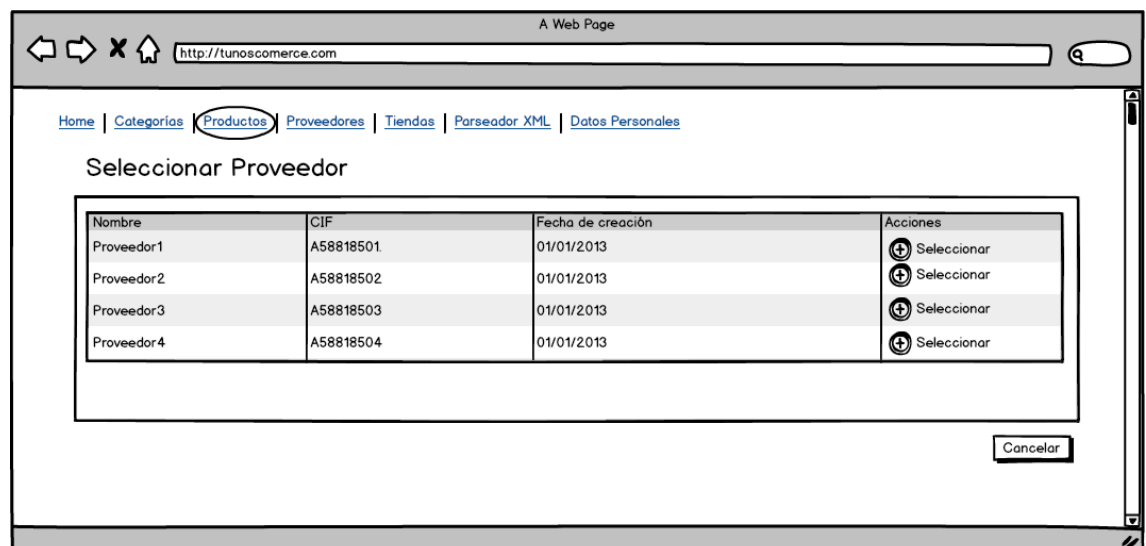


Ilustración 85 Pantalla seleccionar proveedor.

6.6.1.5.2 Pantalla modificar producto

A Web Page

http://tunoscommerce.com

[Home](#) | [Categorías](#) | [Productos](#) | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Modificar producto

Nombre:

Descripción Corta:

Descripción Larga:

Precio:

Alto:

Ancho:

Proveedor: Sin proveedor

Categorías:

Nombre	Descripción
Categoría1	Descripción categoría1

☒ Activo

Ilustración 86 Pantalla Modificar producto

6.6.1.5.3 Pantalla eliminar producto

A Web Page

http://tunoscommerce.com

[Home](#) | [Categorías](#) | **[Productos](#)** | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Confirma que desea eliminar producto

Nombre:

Descripción Corta:

Descripción Larga:

Precio:

Alto:

Ancho:

Proveedor:

Categorías:

Nombre	Descripción
Categoría1	Descripcion categoria1
Categoría2	Descripción categoria2

Ilustración 87 Pantalla eliminar producto

6.6.1.5.4 Pantalla de ver detalle de producto

A Web Page
http://tunoscommerce.com

Home | Categorías | **Productos** | Proveedores | Tiendas | Parseador XML | Datos Personales

Productos

Crear Producto Generar CSV

Buscar Producto

Nombre:

Proveedor:

Nombre	Descripción
Producto 1	descripción
Producto2	descripción
Producto3	descripción
Producto4	descripción

Producto 1

Nombre:

Precio:

Ref-123444445

Descripción Corta:

Descripción Larga:

Alto:

Ancho:

Proveedor:

Categorías:

Nombre	Descripción
Categoría1	Descripción categoría1
Categoría2	Descripción categoría2

Volver

Modificar X Eliminar Exportar

Modificar X Eliminar Exportar

Modificar X Eliminar Exportar

Modificar X Eliminar Exportar

Buscar

Ilustración 88 Pantalla de ver detalle de producto

6.6.1.5.5 Pantalla exportar producto

A Web Page
http://tunoscommerce.com

Home | Categorías | **Productos** | Proveedores | Tiendas | Parseador XML | Datos Personales

Exportar Producto 1

Nombre:

Descripción Corta:

Descripción Larga:

Precio:

Alto:

Ancho:

Proveedor: Sin proveedor

Categorías:

Nombre	Descripción
Categoría1	Descripción categoría1
Categoría2	Descripción categoría2

Estado: Activo

Seleccione tienda de destino:

Exportar Cancelar

Ilustración 89 Pantalla Exportar producto

6.6.1.6 Pantallas de Proveedores

Pantalla principal donde se muestran todos los proveedores creados por el usuario. Desde aquí se podrán hacer todas las operaciones posibles en relación a los proveedores (crear, modificar, buscar, eliminar...)

6.6.1.6.1 Pantalla crear proveedor

A Web Page
http://tunoscommerce.com

[Home](#) | [Categorías](#) | [Productos](#) | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Crear Proveedor

Compañía:

CIF:

Ilustración 90 Pantalla crear proveedor

6.6.1.6.2 Pantalla modificar proveedor

A Web Page
http://tunoscommerce.com

[Home](#) | [Categorías](#) | [Productos](#) | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Modificar Proveedor

Compañía:

CIF:

Ilustración 91 Pantalla modificar proveedor

6.6.1.6.3 Pantalla eliminar proveedor

Ilustración 92 Pantalla eliminar proveedor

6.6.1.6.4 Pantalla ver detalle proveedor

Nombre	CIF	Fecha de creación	Acciones
Proveedor1	A58818501		Modificar X Eliminar ⇨ Exportar
Proveedor2	A58818502		Modificar X Eliminar ⇨ Exportar
Proveedor3	A58818503	01/01/2013	Ver detalle ✎ Modificar X Eliminar ⇨ Exportar
Proveedor4	A58818504	01/01/2013	Ver detalle ✎ Modificar X Eliminar ⇨ Exportar

Ilustración 93 Pantalla ver detalle proveedor

6.6.1.7 Pantallas de tiendas

Pantalla principal donde se muestran todas las tiendas configuradas por el usuario. Desde aquí se podrán hacer todas las operaciones posibles en relación a los proveedores (crear, modificar, buscar, eliminar, importar elementos...)

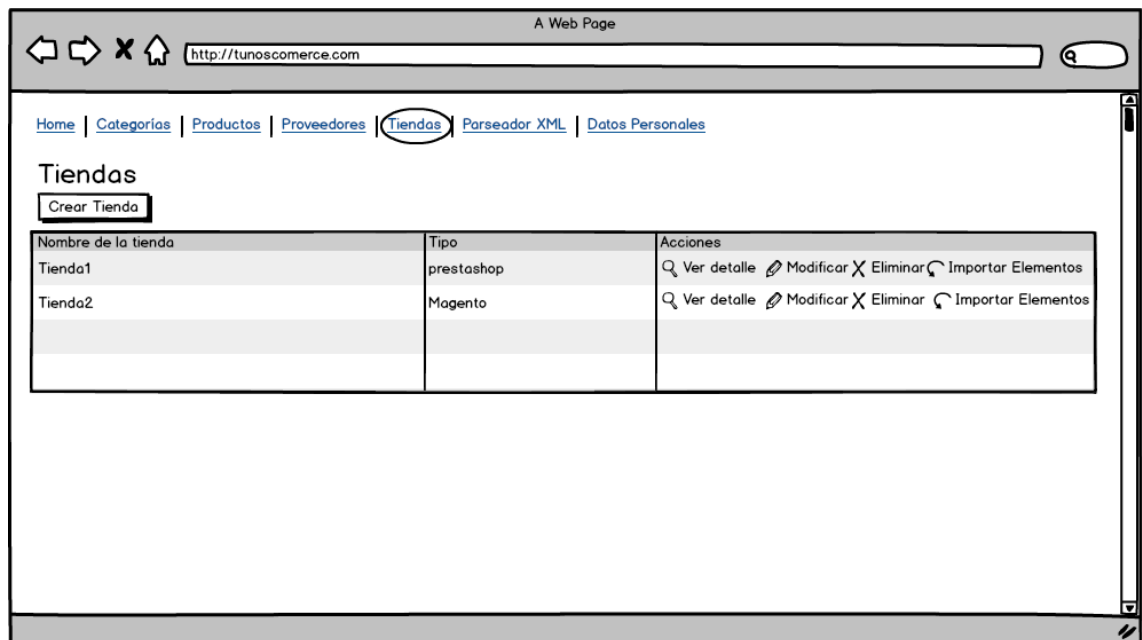


Ilustración 94 Pantalla de tiendas

6.6.1.7.1 Pantalla de crear tienda

Home | Categorías | Productos | Proveedores | **Tiendas** | Parseador XML | Datos Personales

Crear tienda

Nombre:

Tipo:

Nombre de la base de datos:

Host de la base de datos:

Puerto de la base de datos:

Usuario de la base de datos:

Contraseña de la base de datos:

Repita Contraseña de la base de datos:

Ilustración 95 Pantalla de crear tienda

6.6.1.7.2 Pantalla de modificar tienda

A Web Page

http://tunoscommerce.com

[Home](#) | [Categorías](#) | [Productos](#) | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Crear tienda

Nombre:

Tipo:

Nombre de la base de datos:

Host de la base de datos:

Puerto de la base de datos:

Usuario de la base de datos:

Contraseña de la base de datos:

Repita Contraseña de la base de datos:

Ilustración 96 Pantalla modificar tienda

6.6.1.7.3 Pantalla eliminar tienda

A Web Page

http://tunoscommerce.com

[Home](#) | [Categorías](#) | [Productos](#) | [Proveedores](#) | [Tiendas](#) | [Parseador XML](#) | [Datos Personales](#)

Confirma que desea eliminar Tienda 1

Nombre:

Tipo:

Nombre de la base de datos:

Host de la base de datos:

Puerto de la base de datos:

Usuario de la base de datos:

Ilustración 97 Pantalla eliminar tienda

6.6.1.7.4 Pantalla de ver detalle tienda

Ilustración 98 Pantalla ver detalle tienda

6.6.1.7.5 Pantalla importar elementos

Ilustración 99 Pantalla importar elementos

6.6.1.8 Pantalla de Parseador XML

Pantalla principal donde se puede pegar el XML generado por el GMF y a partir de este crear en la plataforma todos los elementos que contiene el XML y sus relaciones.



Ilustración 100 Pantalla parseador XML

6.6.1.9 Pantalla de Datos personales

Esta pantalla contiene la información del usuario logueado. Esta información puede ser modificada.

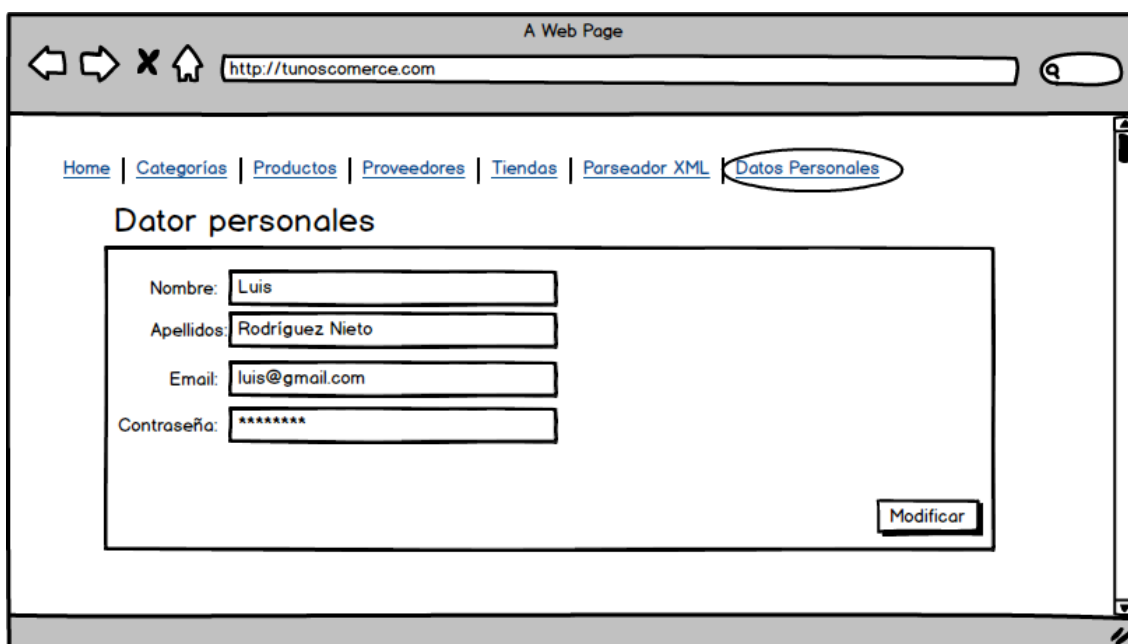


Ilustración 101 Pantalla datos personales

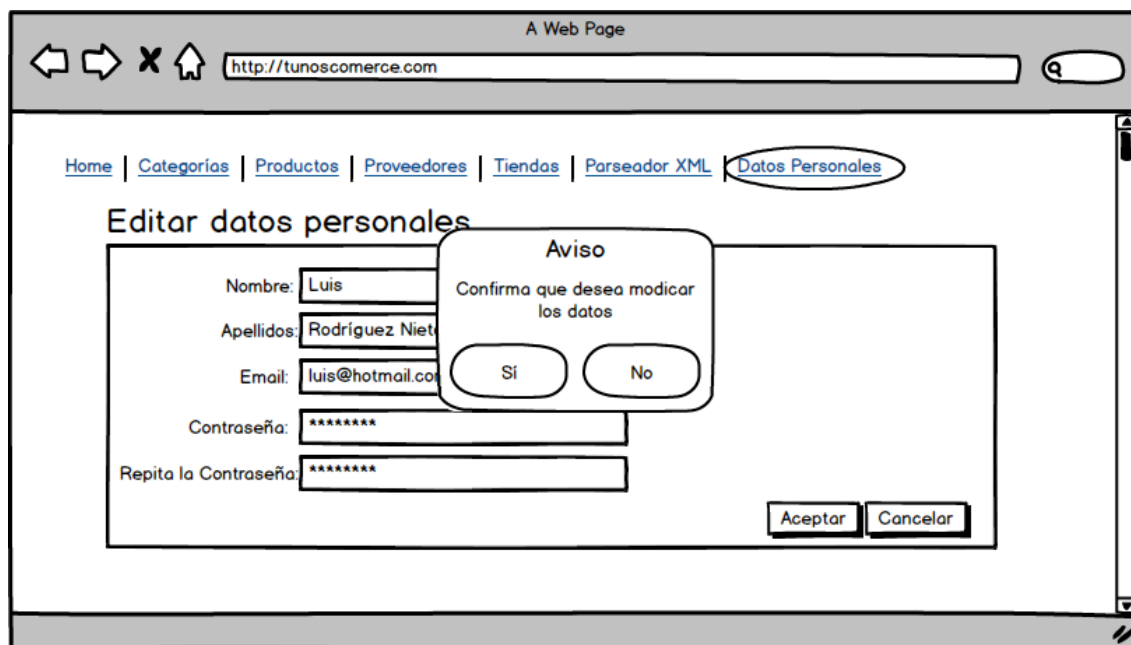


Ilustración 102 Pantalla modificar datos personales

6.6.2 Pantallas Plataforma GMF

El interfaz del GMF está compuesto únicamente de una sola pantalla, en ella se podrán definir las funcionalidades específicas requeridas por el usuario de forma visual a través de un diagrama UML a partir del cual se generará el XML que servirá de entrada a la aplicación web.

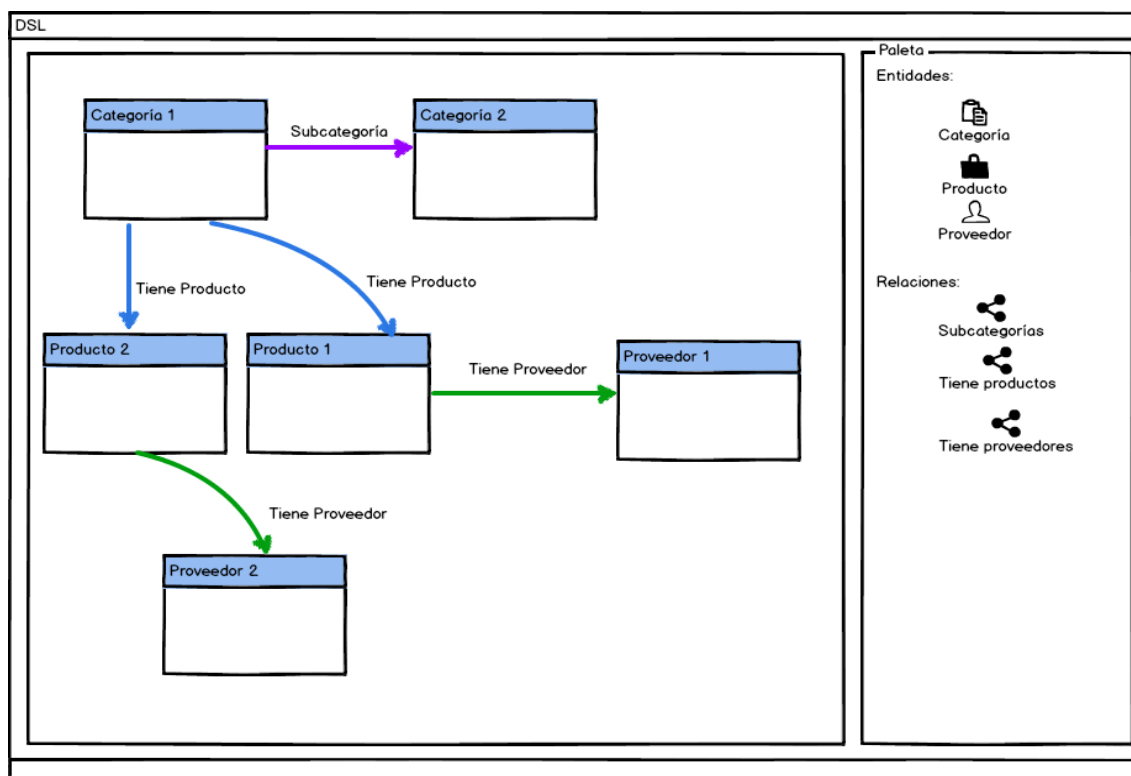


Ilustración 103 Pantalla GMF

6.6.3 Descripción del Comportamiento de la Interfaz

6.6.3.1 Aplicación Web

Todas las interfaces de usuario son de un aspecto muy similar, y tanto el apartado de tiendas, como de categorías, como de productos, como la de proveedores siguen la misma estructura.

Todas las tablas están paginadas permitiendo mostrar la información de una manera más organizada.

Uno de los requisitos de usabilidad, es la robustez de la interfaz. Tanto los mensajes de error como los informativos se mostrarán en diálogos explicativos y en ningún caso el hecho de producirse un error ha de alterar el funcionamiento normal de la interfaz o provocar el cierre de alguna ventana.

Gracias al uso del framework “Foundation las interfaces de usuario se redimensionarán a un tamaño adecuado dependiendo del tipo de la pantalla del dispositivo manejado.

6.6.3.2 GMF

La interfaz del GMF pretende ser lo más amigable posible y fácil de usar, con colores e iconos identificativos para cada tipo de entidad a representar.

6.6.4 Diagrama de Navegabilidad

6.6.4.1 Aplicación Web

Como se comentó en el anterior punto. Los apartados de tiendas, categorías, productos y proveedores tienen prácticamente todos la misma estructura por lo que en el siguiente diagrama se representa únicamente el árbol de navegabilidad de proveedores, los demás serían análogos a este.

6.6.4.2 GMF

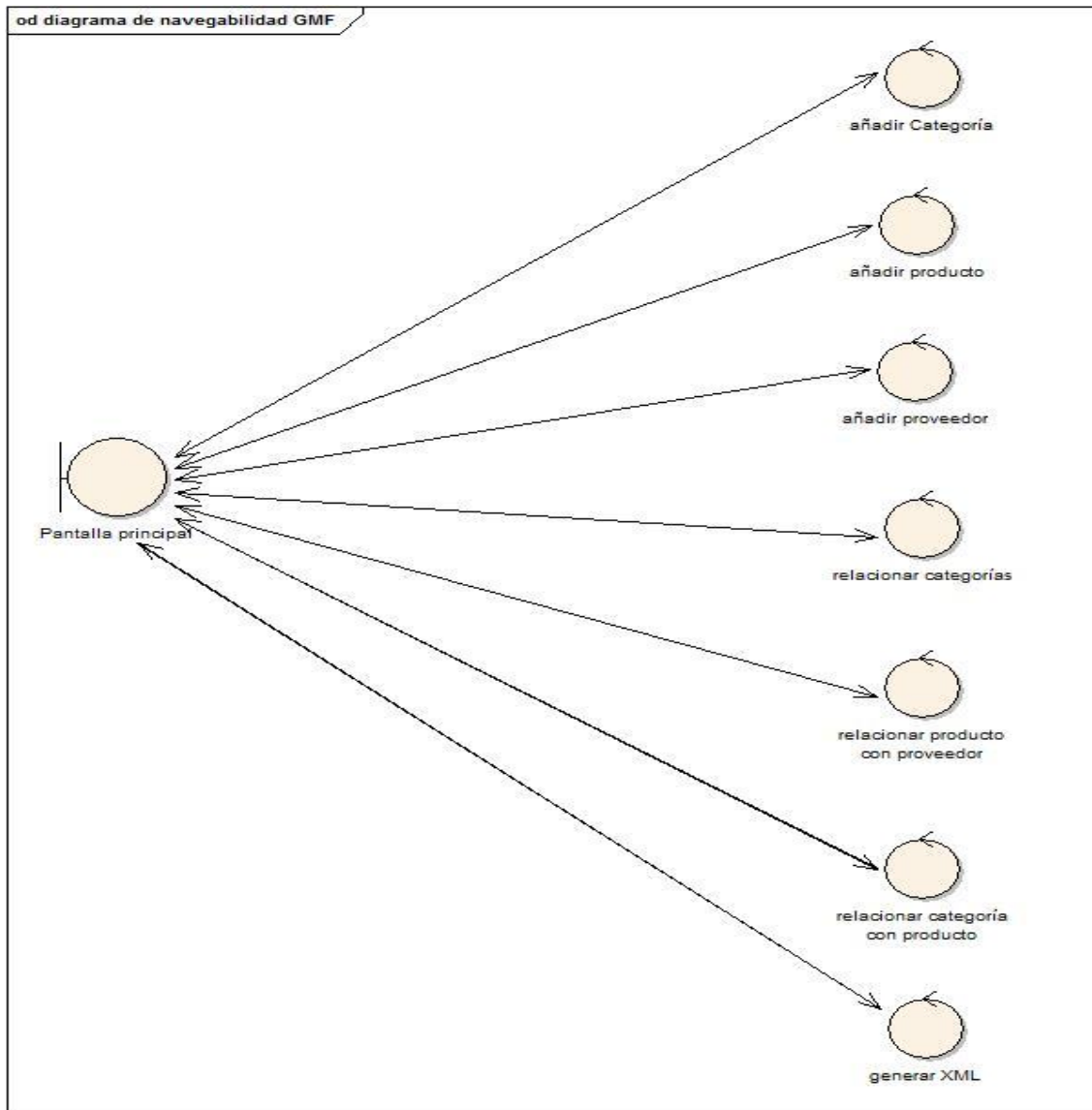


Ilustración 105 Diagrama de navegabilidad GMF

6.7 Especificación del Plan de Pruebas

El plan de pruebas se centrará en testear los puntos más críticos de la aplicación. Por ello, se realizarán pruebas unitarias, de integración, del sistema y de usabilidad.

6.7.1 Pruebas unitarias

Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado.

En el presente proyecto cada nueva funcionalidad será probada. En caso de que se produzcan fallos, se hará una revisión y corrección de los errores detectados, hasta que los resultados de las pruebas tengan un resultado satisfactorio.

6.7.2 Pruebas de usabilidad

Las pruebas de usabilidad son pruebas efectuadas por usuarios con el objeto de determinar si los contenidos y funcionalidades de una herramienta son entendibles y manejables. Para este tipo de análisis se necesita un grupo de personas a las que se les hace una serie de preguntas sobre la herramienta probada. Los resultados obtenidos durante este proceso permiten mejorar algunos de los aspectos y funcionalidades de la aplicación.

En este caso se harán una serie de pruebas a diferentes usuarios expertos e inexpertos en aplicaciones de comercio electrónico.

6.7.3 Pruebas de integración y del sistema

El objetivo de estas pruebas es verificar el correcto ensamblaje entre los distintos componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas. Se probará cada uno de los módulos de la aplicación web y el GFM.

6.7.3.1 Caso de uso: registrarse

Caso de Uso 1: registrarse	
Prueba	Resultado Esperado
Registrar un usuario no existente	El sistema posee un usuario más
Prueba	Resultado Esperado
Intentar registrar un usuario que ya existe	El sistema indica al usuario que ese nombre ya está registrado y no lo guarda
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.2 Caso de uso: loguearse

Caso de Uso 2: loguearse	
Prueba	Resultado Esperado
Introducir usuario y contraseña correctos	El sistema permite entrar al usuario
Prueba	Resultado Esperado
Introducir usuario valido y contraseña incorrecta	El sistema indica al usuario que los datos son incorrectos y no lo deja entrar en el sistema
Prueba	Resultado Esperado
Introducir usuario inválido y contraseña válida	El sistema indica al usuario que los datos son incorrectos y no lo deja entrar en el sistema

6.7.3.3 Caso de uso: configurar tienda

Caso de Uso 3: configurar tienda	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de tienda y guarda la opción	El sistema almacena y muestra una nueva tienda para ese usuario en la web
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la tienda e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.4 Caso de uso: crear categoría

Caso de Uso 4 : crear categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y guarda la opción	El sistema almacena y muestra una nueva categoría para ese usuario en la web
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.4.1 Subcaso de uso: crear una subcategoría

Caso de uso 4: crear categoría. Subcaso de Uso : crear una subcategoría

Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y selecciona una categoría ya creada en el sistema como categoría padre	El sistema almacena correctamente la subcategoría y la muestra correctamente al usuario
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la categoría
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.4.2 Subcaso de uso: ver detalle de una categoría

<i>Caso de uso 4: crear categoría.Subcaso de Uso : ver detalle de una categoría</i>	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de una categoría	Los datos de la categoría se muestran correctamente al usuario

6.7.3.4.3 Subcaso de uso: eliminar una categoría

<i>Caso de uso 4: crear categoría.Subcaso de uso : eliminar una categoría</i>	
Prueba	Resultado Esperado
El usuario intenta eliminar una categoría sin hijos	El sistema elimina correctamente la categoría y le indica al usuario que todo ha ido correctamente
Prueba	Resultado Esperado
El usuario intenta eliminar una categoría que contiene subcategorías	El sistema le indica al usuario que esa categoría no puede eliminarse ya que contiene subcategorías y no elimina la categoría del sistema
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.4.4 Subcaso de uso: modificar una categoría

Caso de uso 4: crear categoría. Subcaso de uso : modificar una categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de categoría y guarda la opción	El sistema modifica la categoría e indica al usuario que los cambios se han realizado con éxito
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.4.5 Subcaso de uso: relacionar productos a una o más categorías

Caso de uso 4: crear categoría. Subcaso de uso : relacionar productos a una o más categorías	
Prueba	Resultado Esperado
El usuario asocia un producto con una o más categorías	El sistema asocia el producto con las categorías seleccionadas
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.5 Caso de uso: crear un producto

Caso de Uso 5 : crear producto	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de productos y guarda la opción	El sistema almacena y muestra un nuevo producto para ese usuario en la web
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la productos e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.5.1 Subcaso de uso: ver detalle de un producto

<i>Caso de Uso 5 : crear producto Subcaso de uso: ver detalle de un producto</i>	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un producto	Los datos del producto se muestran correctamente al usuario

6.7.3.5.2 Subcaso de uso: eliminar un producto

<i>Caso de Uso 5 : crear producto Subcaso de uso : eliminar un producto</i>	
Prueba	Resultado Esperado
El usuario elimina un producto	El sistema elimina correctamente el producto del sistema y se lo comunica al usuario
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.5.3 Subcaso de uso: modificar un producto

<i>Caso de Uso 5 : crear producto Subcaso de uso : modificar un producto</i>	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de producto y guarda la opción	El sistema modifica el producto e indica al usuario que los cambios se han realizado con éxito
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del producto e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.5.4 Subcaso de uso: relacionar producto con proveedor

<i>Caso de Uso 5 : crear producto Subcaso de uso : relacionar producto con proveedor</i>	
Prueba	Resultado Esperado
El usuario asocia un producto con un proveedor	El sistema asocia el producto con el proveedor seleccionado
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.6 Caso de uso: crear un proveedor

Caso de Uso 6 : crear proveedor	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de proveedor y guarda la opción	El sistema almacena y muestra un nuevo proveedor para ese usuario en la web
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la proveedor intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.6.1 Subcaso de uso: ver detalle de un proveedor

Caso de Uso 6 : crear proveedor Subcaso de Uso : ver detalle de un producto	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un proveedor	Los datos del proveedor se muestran correctamente al usuario

6.7.3.6.2 Subcaso de uso: eliminar un proveedor

Caso de Uso 6 : crear proveedor Subcaso de uso : eliminar un producto	
Prueba	Resultado Esperado
El usuario elimina un proveedor	El sistema elimina correctamente el proveedor del sistema y lo desvincula con todos los productos que esté asociado , además de indicar al usuario que todo ha ido correctamente
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.6.3 Subcaso de uso: modificar un proveedor

<i>Caso de Uso 6 : crear proveedor Subcaso de uso : modificar un proveedor</i>	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de proveedor y guarda la opción	El sistema modifica el proveedor e indica al usuario que los cambios se han realizado con éxito
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del proveedor e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.7 Caso de uso: importar elementos

<i>Caso de Uso 7 : importar elementos</i>	
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada correctamente	El sistema importa los elementos y notifica al usuario que se han importado con éxito
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada incorrectamente	El sistema no puede conectarse a la tienda por lo que avisa al usuario que la operación no pudo realizarse
Prueba	Resultado Esperado
El usuario intenta importar elementos de una tienda cuya base de datos está corrupta.	El sistema intenta importar pero se produce un error y avisa al usuario que la operación no pudo realizarse.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.8 Caso de uso: exportar elementos

Caso de Uso 8 : exportar elementos	
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda configurada correctamente	El sistema exporta los elementos y notifica al usuario que se han importado con éxito
Prueba	Resultado Esperado
El usuario exportar elementos a una tienda configurada incorrectamente	El sistema no puede conectarse a la tienda por lo que avisa al usuario que la operación no pudo realizarse
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda cuya base de datos está corrupta.	El sistema intenta exportar los elementos pero se produce un error y avisa al usuario que la operación no pudo realizarse.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.9 Caso de uso: importar XML

Caso de Uso 9 : importar XML	
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato correcto	El sistema importa los elementos contenidos en el XML y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato incorrecto	El sistema importar el XML pero al ser el formato incorrecto se notifica al usuario el problema y el sistema permanece sin cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.10 Caso de uso: Generar CSV

Caso de Uso 10 : importar XML	
Prueba	Resultado Esperado
El usuario genera un CSV	El sistema genera un CSV

6.7.3.11 Caso de uso: Editar datos de usuario

Caso de uso 11: editar datos de usuario	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de sus datos personales y guarda la opción	El sistema modifica los datos del usuario e indica al éste que los cambios se han realizado con éxito
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación de sus datos personales e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

6.7.3.12 Caso de uso: Crear categoría UML

Caso de uso 12: crear categoría UML	
Prueba	Resultado Esperado
El usuario arrastra un componente categoría de la paleta de elementos a la zona de dibujo	El sistema pinta una categoría con sus atributos y características.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción categoría	El sistema pinta una categoría con sus atributos y características.

6.7.3.13 Caso de uso: Crear subcategoría UML

Caso de uso 13: crear subcategoría UML	
Prueba	Resultado Esperado
El usuario enlaza dos categorías e indica cual es la categoría base	El sistema pinta un nexo de unión entre las dos categorías creadas por el usuario

6.7.3.14 Caso de uso: Crear producto UML

Caso de uso 14: crear producto UML	
Prueba	Resultado Esperado
El usuario arrastra un componente producto de la paleta de elementos a la zona de dibujo	El sistema pinta un producto con sus atributos y características.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción producto	El sistema pinta un producto con sus atributos y características.

6.7.3.15 Caso de uso: Crear proveedor UML

Caso de uso 15: crear proveedor UML	
Prueba	Resultado Esperado
El usuario arrastra un componente proveedor de la paleta de elementos a la zona de dibujo	El sistema pinta un proveedor con sus atributos y características.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción proveedor	El sistema pinta un proveedor con sus atributos y características.

6.7.3.16 Caso de uso: Relacionar un producto con una o varias categorías

Caso de uso 16: relacionar un producto con una o varias categorías	
Prueba	Resultado Esperado
El usuario enlaza dos o más categorías con un producto	El sistema pinta un nexo de unión entre las categorías y el producto

6.7.3.17 Caso de uso: Relacionar un producto un proveedor

Caso de uso 17: Relacionar un producto con un proveedor	
Prueba	Resultado Esperado
El usuario enlaza un producto con un proveedor	El sistema pinta un nexo de unión entre el producto y el proveedor
Prueba	Resultado Esperado
El usuario intenta enlazar un producto con más de un proveedor	El sistema pinta un nexo de unión entre el producto y el primer proveedor con el que se intenta unir pero no permite unirlo con más de uno

6.7.3.18 Caso de uso: Generar un XML

Caso de uso 18: Generar un XML	
Prueba	Resultado Esperado
El usuario genera un XML después de haber creado elementos XML	El sistema genera un XML con toda la información de los elementos UML creados y sus relaciones
Prueba	Resultado Esperado
El usuario intenta generar un XML sin haber creado previamente elementos UML	El sistema crea un XML pero sin información de ningún elemento

Capítulo 7. Diseño del Sistema

7.1 Arquitectura del Sistema

A continuación se muestra un diagrama de alto nivel de la arquitectura de la aplicación. Como se puede observar la aplicación esta compuesta por dos partes claramente diferenciadas, el GMF y la aplicación web. Estos interactúan a través de un archivo XML.

La aplicación web se comunicará a con las diferentes plataformas de comercio electrónico a través de las bases de datos de estas.

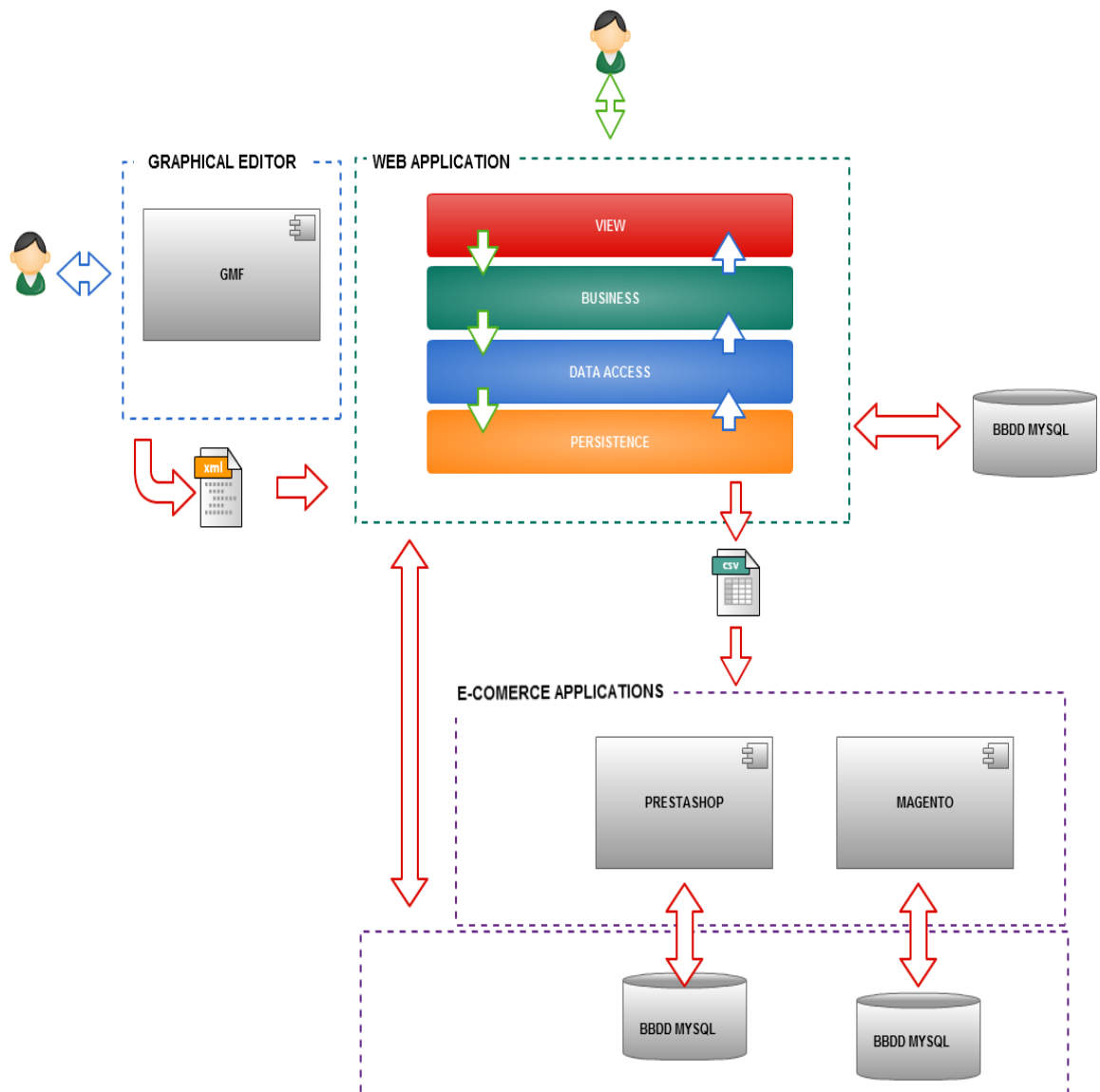


Ilustración 106 Arquitectura del sistema

7.1.1 Arquitectura de la aplicación web

En esta estructura se distinguen todos los bloques que forman la plataforma web desglosados para una mejor comprensión del sistema y que de esta forma se pueda guiar el desarrollo del sistema.

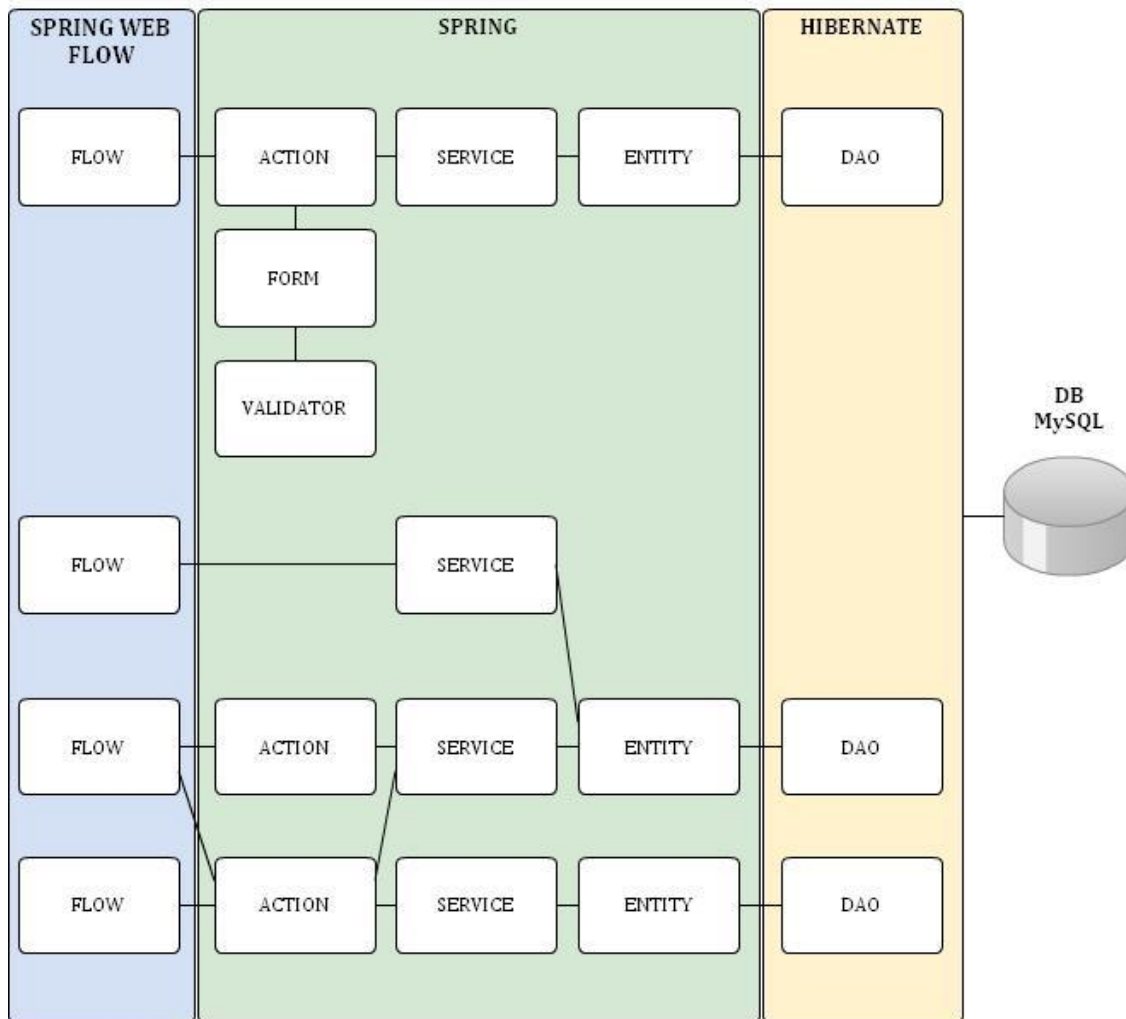


Ilustración 107 Arquitectura de la plataforma web

7.1.2 Diagramas de Paquetes

A continuación se muestra el diagrama de paquetes de la aplicación web, en los siguientes subapartados se describirá con más detalle cada uno de ellos.

7.1.2.1 Plataforma web

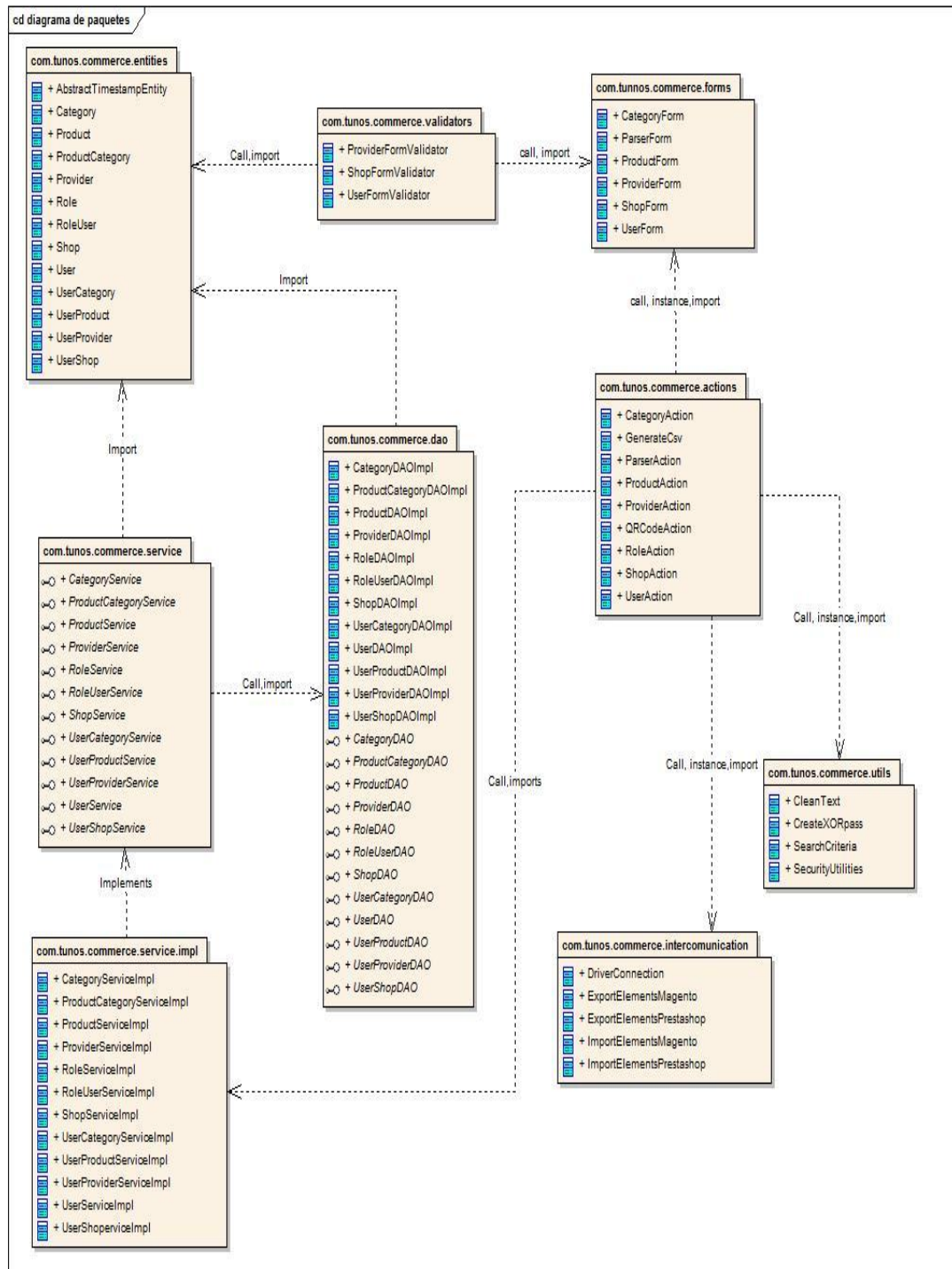


Ilustración 108 Diagrama de paquetes

7.1.2.1.1 Paquete entities

Este paquete contiene todas las clases Java que representan el modelo Orientado a objetos de las diferentes entidades manejadas en el sistema.

En cada una de estas entidades se hace el mapeo de las notaciones Hibernate que facilitan la ORM de la persistencia en la base de datos de todas las entidades y relaciones modeladas. El uso de estas anotaciones evita que se necesiten crear ficheros de configuración de tipo (miEntidad.hbm.xml)

7.1.2.1.2 Paquete DAO

En este paquete se encuentran todos los interfaces DAO y sus implementaciones encargados de abstraer y encapsular los accesos a la base de datos de cada una de las entidades definidas en el anterior paquete *entities*.

7.1.2.1.3 Paquete service

En este paquete se define el conjunto de interfaces de los servicios que ofrecerá cada uno de los manejadores de cada entidad.

El conjunto de servicios ofrecidos desde este paquete será consumido por las acciones contenidas en el paquete *actions*

7.1.2.1.4 Paquete service.impl

Este paquete contiene la implementación de los métodos definidos en las interfaces del paquete *service*.

Estos métodos consumen directamente las funciones de acceso a la base de datos ofrecidas dentro del paquete DAO.

7.1.2.1.5 Paquete actions

Este paquete será el encargado de almacenar las clases que definirán mediante sus métodos las acciones a realizar en el flujo de control de la interfaz de usuario.

El conjunto de métodos de estas clases se invocarán desde la capa de flujo SWF (Spring Web Flow) mediante estados de acción. Todas las clases heredan de la super clase "org.springframework.webflow.action.FormAction"

7.1.2.1.6 Paquete Forms

Este paquete contiene las clases que representan los objetos de presentación. Estas clases recopilan toda la información que es necesaria consumir desde presentación (formularios, tablas...) .Todas las clases deben implementar "java.io.Serializable".

7.1.2.1.7 Paquete Validators

Este contiene las clases que tienen como función validar los formularios definidos en el paquete forms.

Cada una de las clases implementa los métodos definidos por la interfaz “org.springframework.validation.Validator” cuya funcionalidad es facilitar en gran medida las acciones de validación en el servidor.

7.1.2.1.8 Paquete intercommunication

Este paquete está compuesto por las clases necesarias para poder establecer una conexión externa y hacer las importaciones y exportaciones de los diferentes elementos (categorías, productos, proveedores) hacia las plataformas e-commerce Magento y Prestashop.

7.1.2.1.9 Paquete utils

Este paquete contiene el conjunto de herramientas necesarias para la gestión de entidades que se reutilizan en los diferentes paquetes.

Además en este paquete se encontrarán métodos útiles para la codificación y decodificación

7.1.3 Diagrama de Despliegue

A continuación se muestra el diagrama de despliegue de la aplicación, para a continuación describir cada uno de sus componentes.

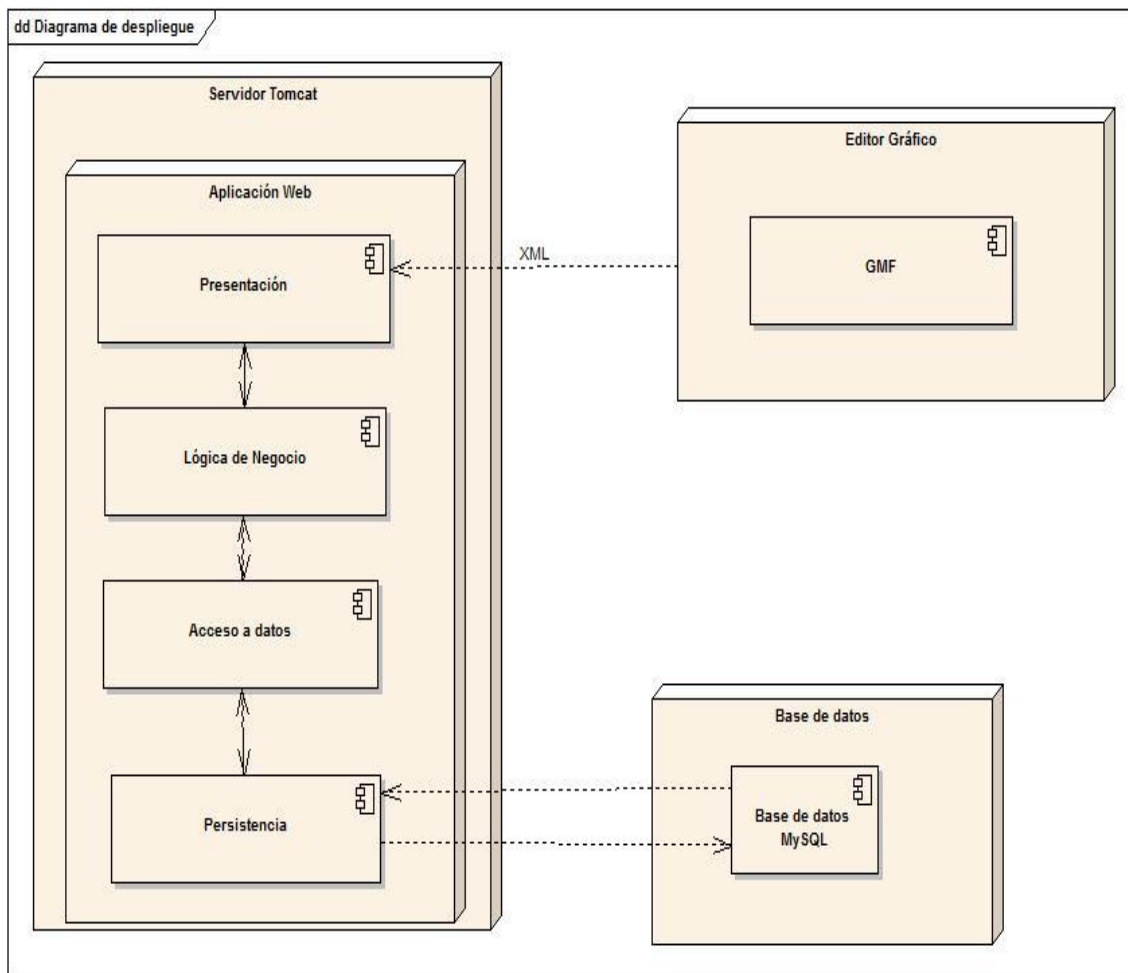


Ilustración 109 Diagrama de despliegue

7.1.3.1 Aplicación Web

La plataforma web contiene los diferentes módulos que permiten realizar la mayoría de las tareas de la aplicación. Estos módulos o capas son:

- Presentación
- Lógica de negocio
- Acceso a datos
- Persistencia

La aplicación web se encuentra contenida en un servidor Apache Tomcat.

7.1.3.2 Servidor Tomcat

La aplicación web se encuentra desplegada sobre un Apache Tomcat que funciona como contenedor de servlets.

7.1.3.3 Base de datos

Contiene la base de datos MySQL que usa la aplicación web para guardar los distintos datos que maneja. Se comunica con la aplicación a través del módulo de persistencia de la aplicación web.

7.1.3.4 GMF

Editor gráfico GMF, aquí se pueden diseñar los diagramas UML con los diferentes elementos del catálogo de comercio electrónico (categorías, producto o proveedores).

El GMF genera un XML con la información del diagrama UML y este se puede ser procesado por la aplicación web.

7.2 Diseño de Clases

7.2.1 Diagramas de Clases

7.2.1.1 Aplicación GMF

En el siguiente apartado se describen los atributos y métodos del paquete model del GMF. Únicamente se muestran las clases de este paquete debido a que es el más importante. Los demás se obvian debido al gran volumen de clases que se autogeneran al crear el GMF.

7.2.1.1.1 Diagrama de clases del paquete model

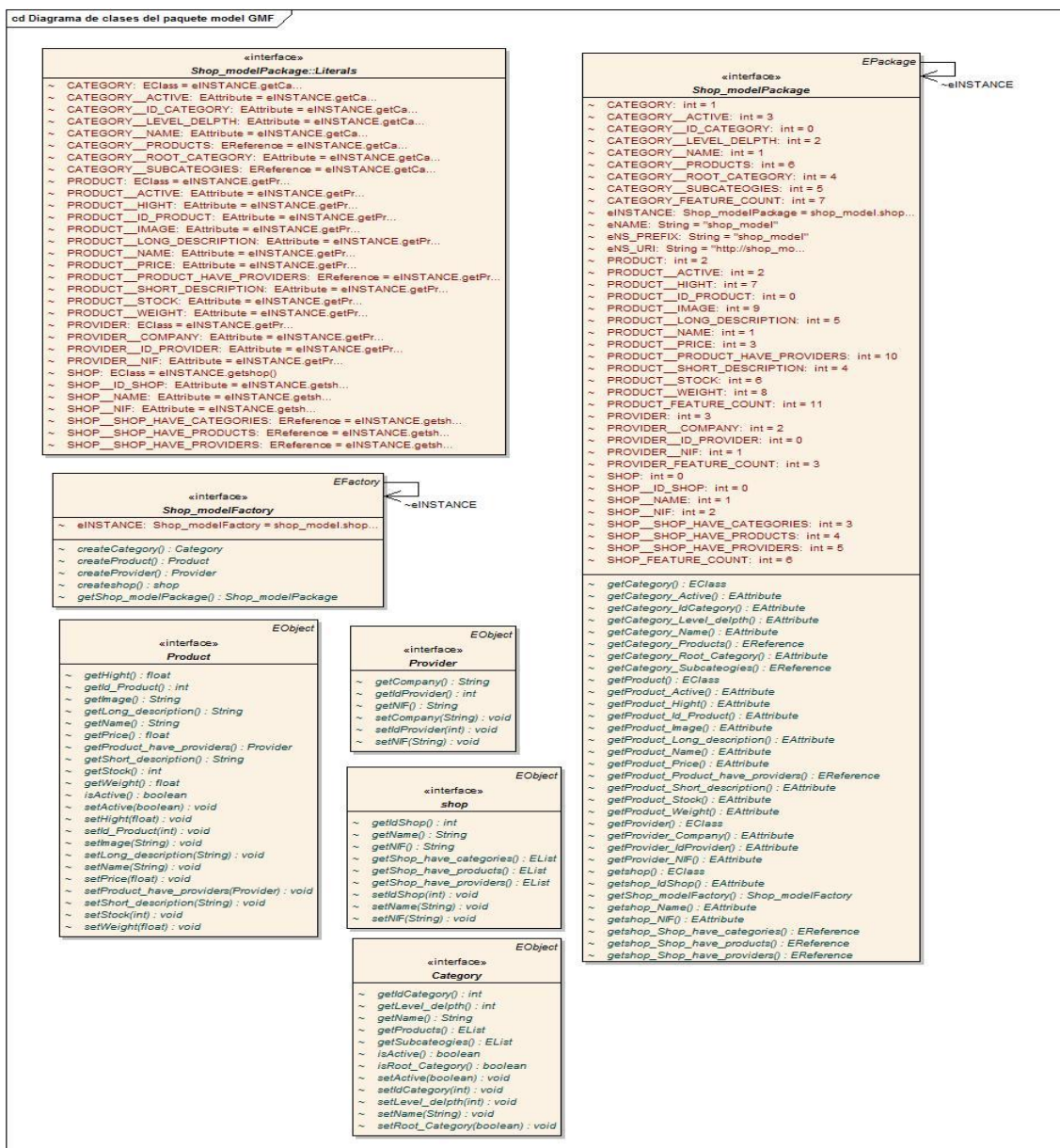


Ilustración 110 Diagrama de clases del paquete model del GMF en la fase de diseño

7.2.1.1.2 Diagrama de clases del paquete model.impl



Ilustración 111 Diagrama de clases del paquete model.impl

7.2.1.2 Aplicación Web

En el siguiente apartado, se describen los atributos y métodos de cada una de las clases que forman parte de la aplicación web. Cada diagrama se corresponde con un paquete de los ya descritos anteriormente.

7.2.1.2.1 Diagrama de clases paquete entities

A continuación se muestra el diagrama de clases que componen el paquete entities. Como ya se mencionó anteriormente contiene las clases Java que representan el modelado orientado a objetos de las diferentes entidades del sistema.

Todas las clases incluyen las anotaciones Hibernate que facilitan la persistencia de dichos objetos en la base de datos y sus relaciones.

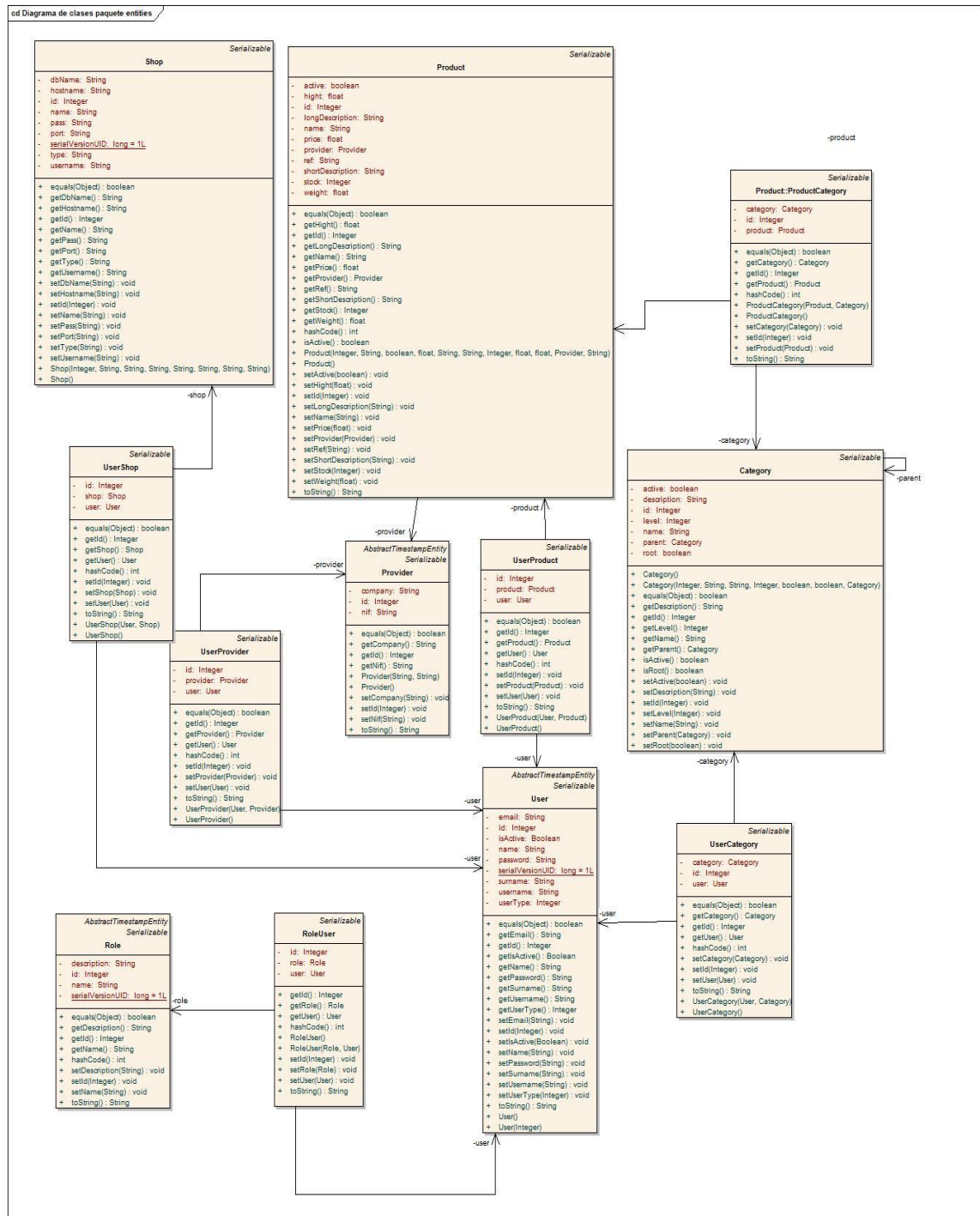


Ilustración 112 Diagrama de clases del paquete entities en la fase de diseño

7.2.1.2.2 Diagrama del clases del paquete DAO

Como se ha descrito anteriormente este paquete está estructurado siguiendo el patrón de diseño DAO. Este permite acceder a los datos de la aplicación solamente a través de los métodos definidos en DAO.

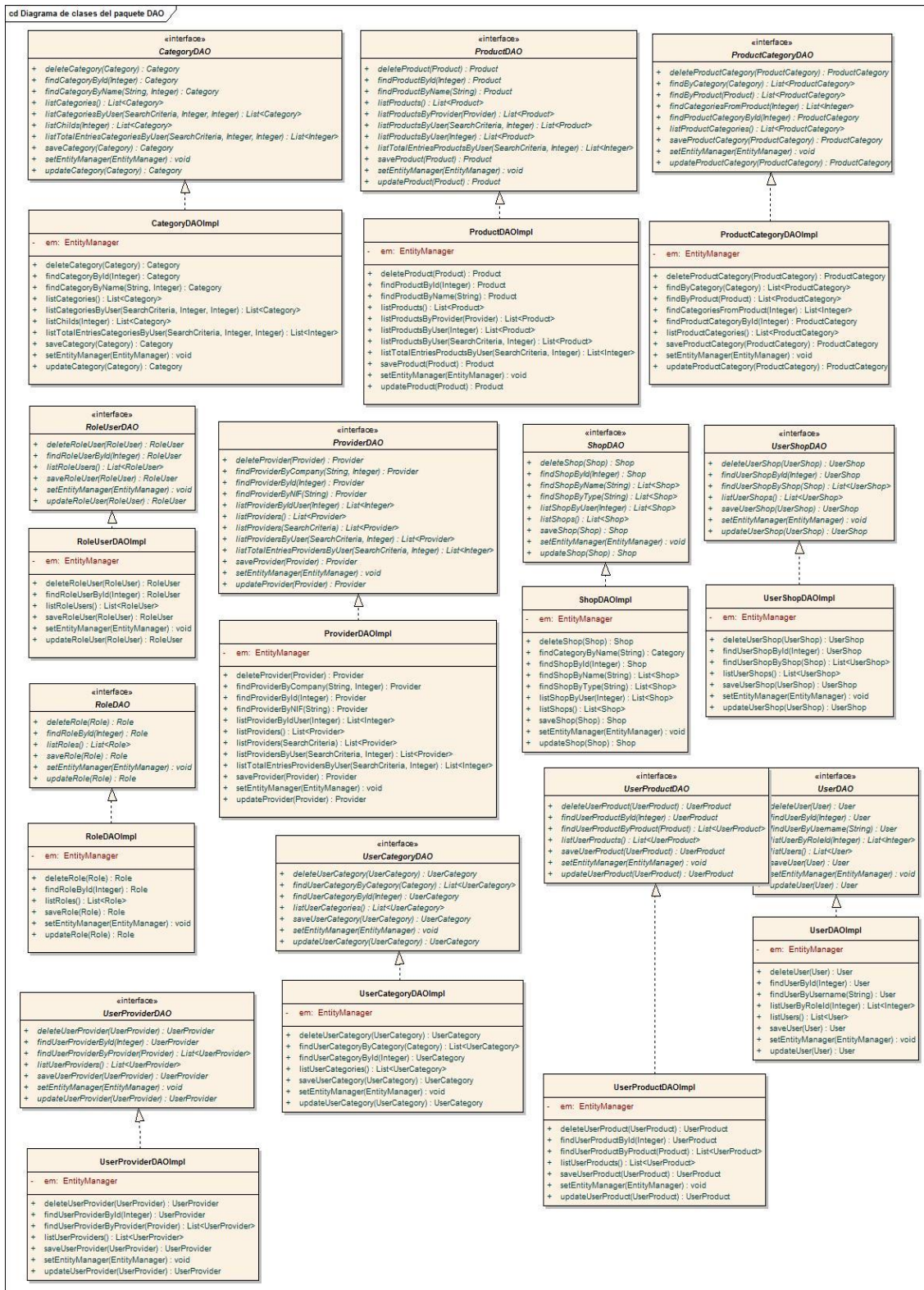


Ilustración 113 Diagrama de clases del paquete DAO en la fase de diseño

7.2.1.2.3 Diagrama de clases paquete service

Este paquete contiene las interfaces de los servicios que ofrecerá cada uno de los manejadores de cada una de las entidades.

Cada uno de los métodos definidos en estas interfaces será consumido directamente por las clases del paquete actions.

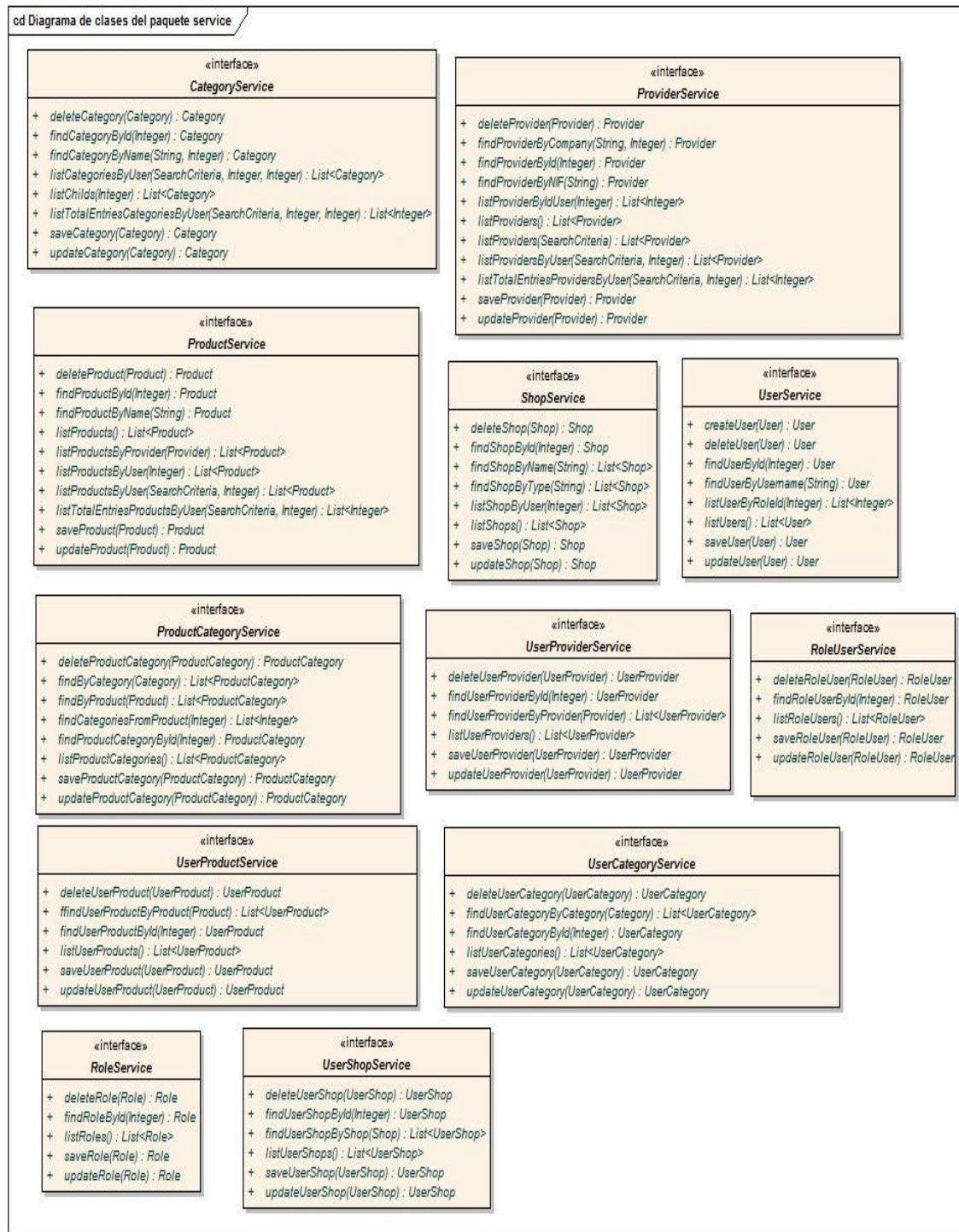


Ilustración 114 Diagrama de clases del paquete Service en la fase de diseño

7.2.1.2.4 Diagrama de clases paquete service.impl

Las clases de este paquete implementan los métodos definidos en el anterior paquete (service).

Las clases de este paquete se comunican con las clases del paquete DAO para el intercambio de datos.

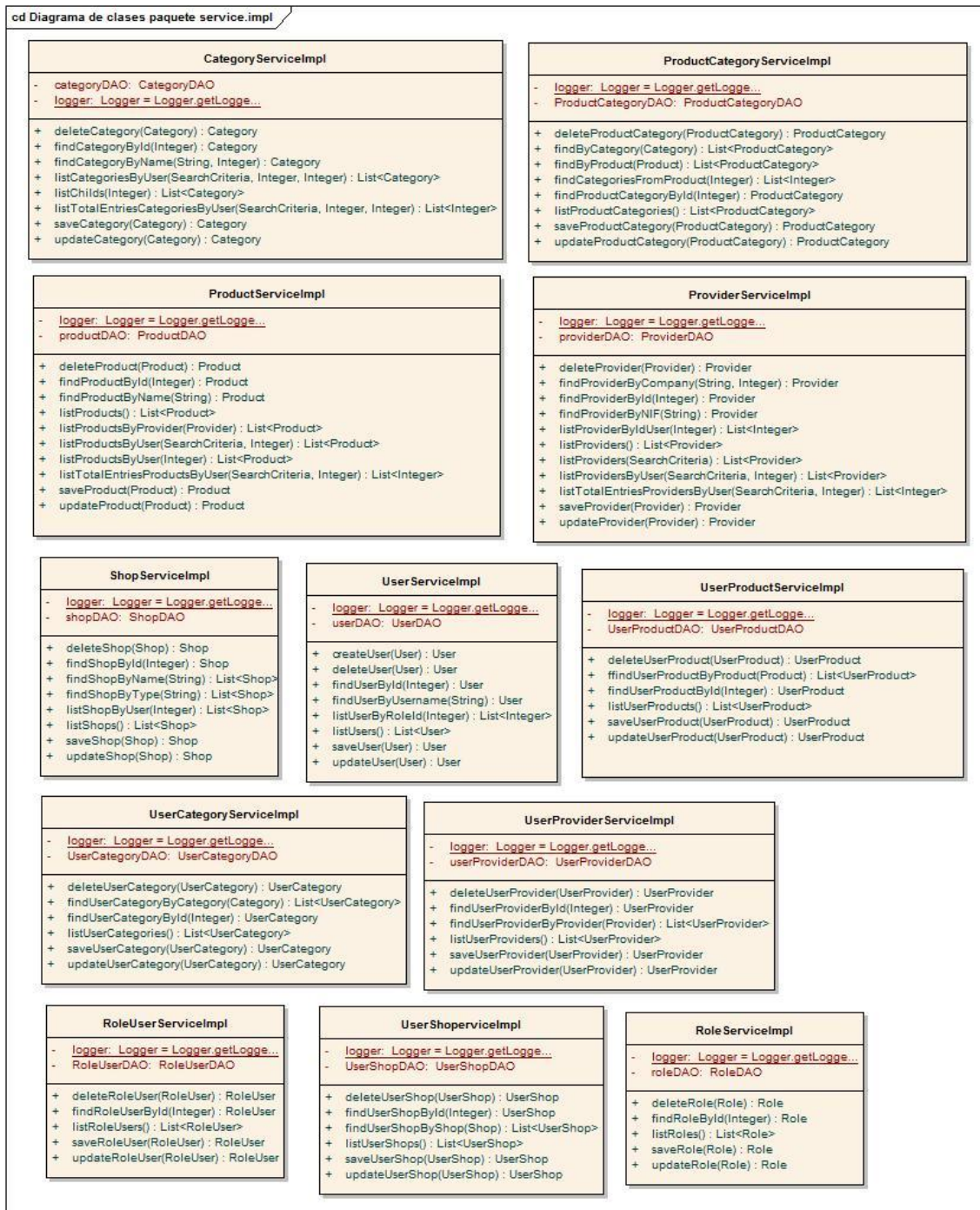


Ilustración 115 Diagrama de clases del paquete Service.impl en la fase de diseño

7.2.1.2.5 Diagrama de clases paquete actions

Este paquete contiene las clases que definirán mediante sus métodos las diferentes acciones que se podrán realizar desde la interfaz de usuario.

El conjunto de métodos de cada una de las clases se invocarán desde la capa de flujo SWF mediante estados acción.

Cada una de las clases hereda de la clase `org.springframework.webflow.action.FormAction`.



Ilustración 116 Diagrama de clases del paquete actions en la fase de diseño

7.2.1.2.6 Diagrama de clases del paquete forms

Este paquete contiene todas las clases que representan los objetos de presentación y almacenarán toda la información que el usuario suministra o consume directamente.



Ilustración 117 Diagrama de clases del paquete forms en la fase de diseño

7.2.1.2.7 Diagrama de clases del paquete intercommunication

En este paquete se encuentran las clases dedicadas a intercomunicar la plataforma web con las otras dos plataformas de comercio electrónico Magento y Prestashop.

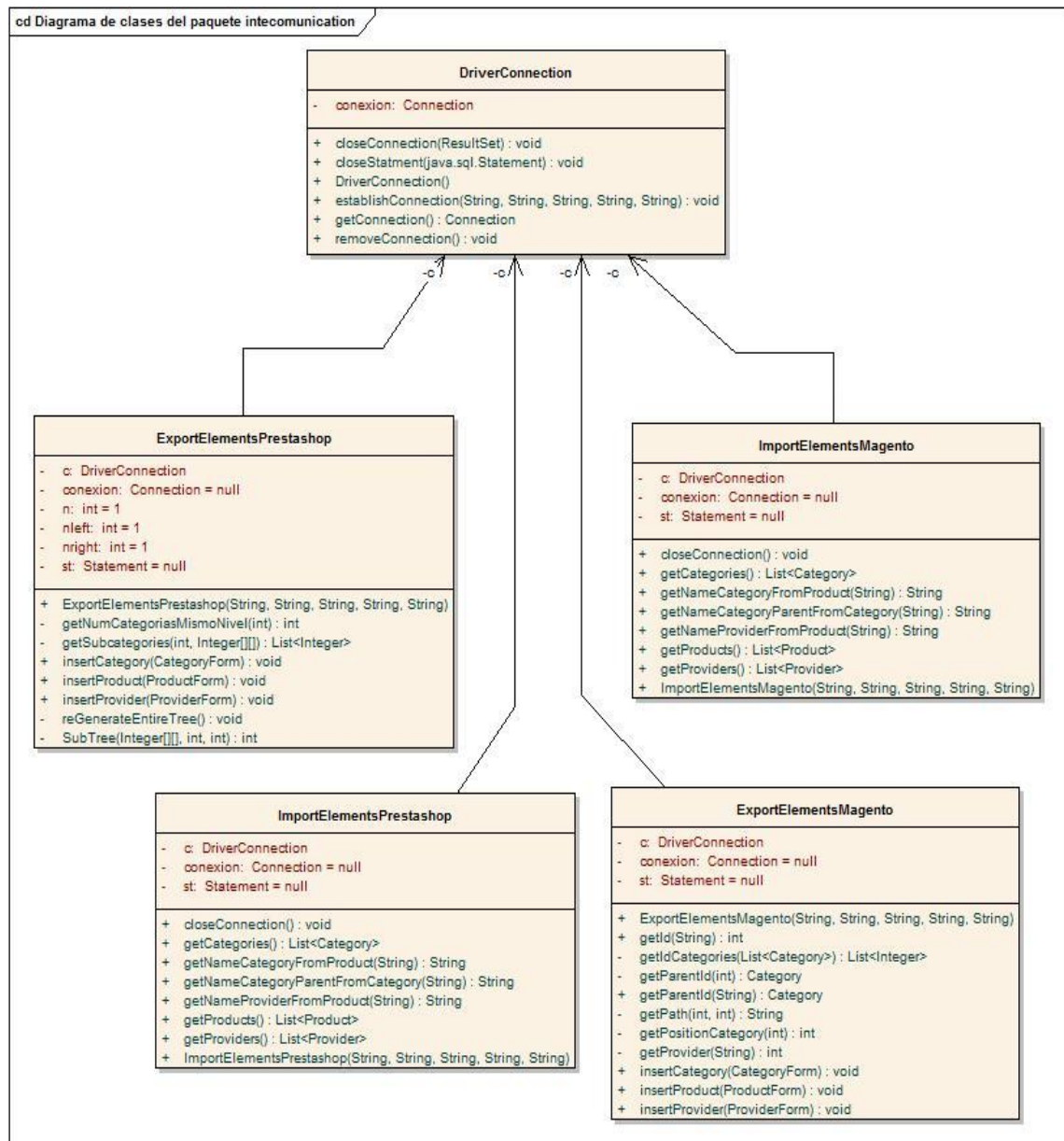


Ilustración 118 Diagrama de clases del paquete intercommunications en la fase de diseño

7.2.1.2.8 Diagrama de clases del paquete validators

Este paquete contiene las clases cuya funcionalidad es validar algunos de los campos de los formularios rellenos por el usuario.

Cada uno de los validadores implementa los métodos definidos por la interfaz `org.springframework.validation.Validator`.



Ilustración 119 Diagrama de clases del paquete validators en la fase de diseño

7.2.1.2.9 Diagrama de clases del paquete utils

Este paquete contiene todas las herramientas necesarias para la gestión de entidades que se reutilizarán en los diferentes paquetes.

En este caso aquí se encuentran las clases empleadas para la paginación de las diferentes tablas de la interfaz de usuario y varias clases dedicadas a la codificación de contraseñas del usuario.

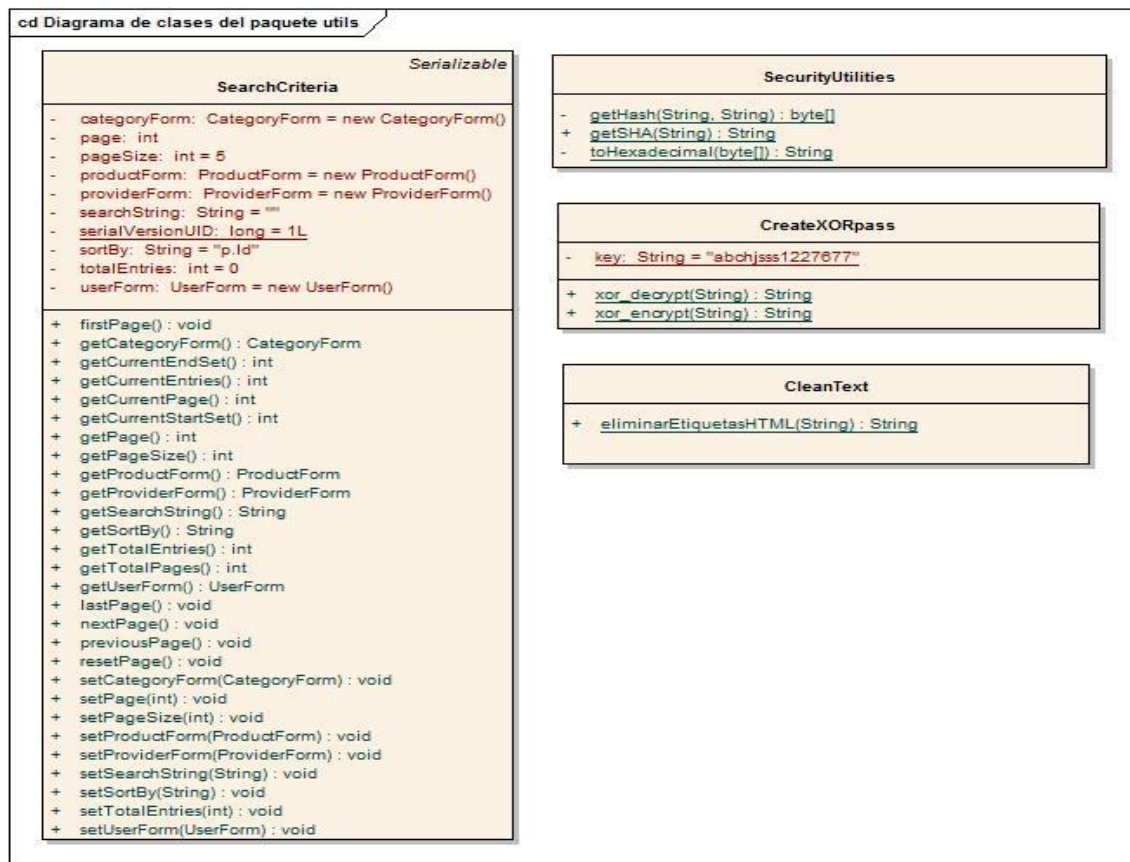


Ilustración 120 Diagrama de clases del paquete utils en la fase de diseño

7.3 Diagramas de interacción

7.3.1 Caso de uso: Registrar usuario

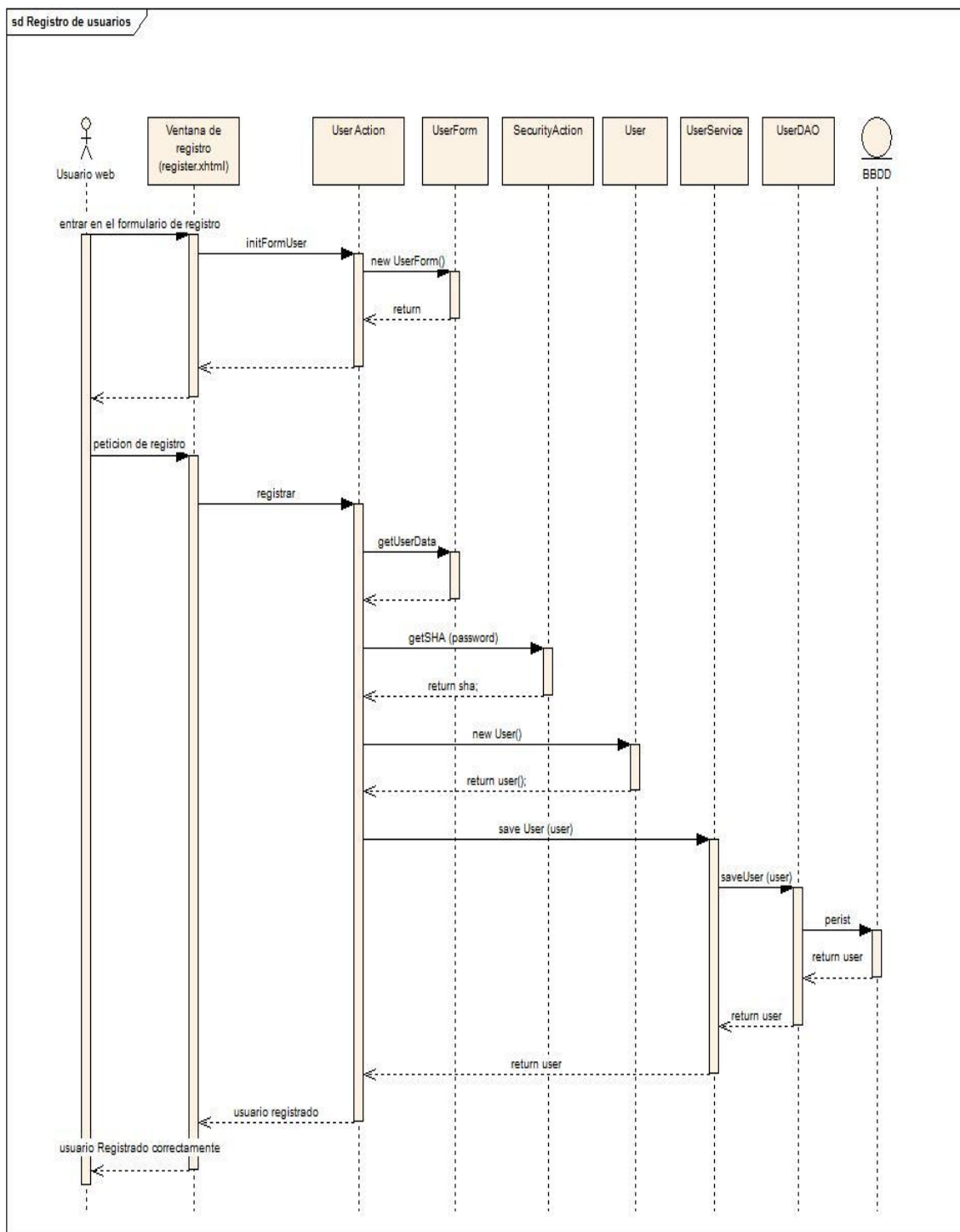


Ilustración 121 Diagramas de interacción: registro de usuarios

1. En primer lugar el usuario entra a la ventana de registro y se inicializa en el flow la entidad UserForm.
2. A continuación el usuario cumplimenta el formulario de registro y le da a la opción registrar.
3. Desde UserAction se recogen del flow encapsulados en la UserForm. A continuación se invoca a la clase SecurityAction contenida en el paquete utils para que codifique la contraseña de usuario en "SHA-1".
4. Una vez codificada la contraseña del usuario se crea un objeto User y se invoca el servicio UserAction para que los datos del usuario sean guardados.
5. La clase UserService invoca el método correspondiente de la capa DAO para que este haga el persist.
6. Desde la capa DAO se hace el persist del usuario directamente a la base de datos y se retorna el usuario guardado como confirmación que la operación se ha realizado con éxito.

7.3.2 Caso de uso: Configurar tienda

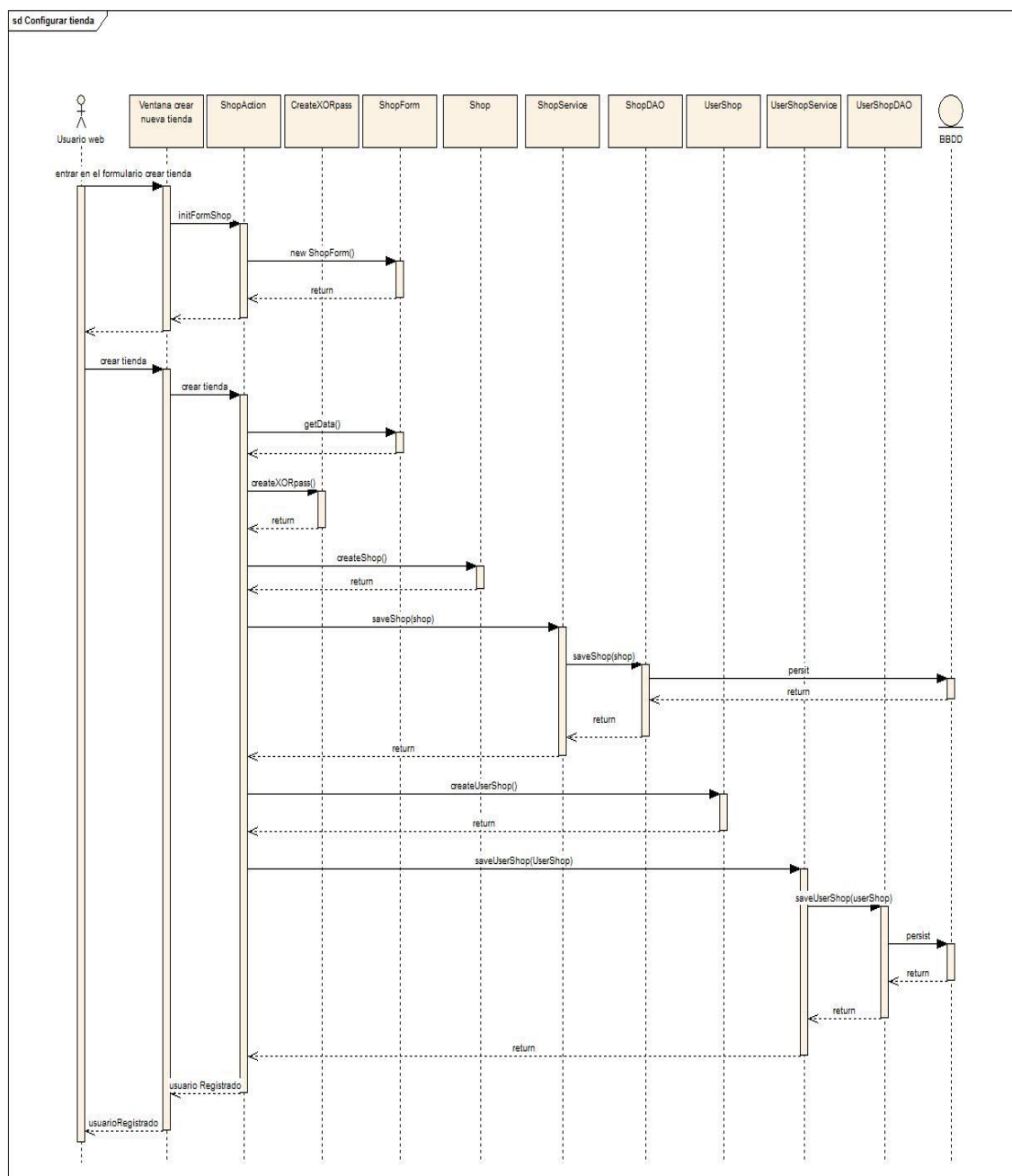


Ilustración 122 Diagramas de interacción: configurar tienda

1. En primer lugar el usuario entra a la ventana de crear nueva tienda, esto provoca que se inicialice en el flow una entidad de tipo ShopForm.
2. A continuación el usuario cumplimenta el formulario de creación de tienda y le da a la opción guardar.
3. Desde ShopAction se recogen los datos del flow encapsulados en ShopAction. Este para codificar las contraseñas de acceso a la base de datos externa invoca a CreateXORPass. En este caso se codifica con un XOR puesto que este algoritmo tiene vuelta atrás cosa que en la codificación "SHA-1" no es posible.
4. A continuación desde ShopAction se crea un objeto Shop y se invoca a la clase ShopService para que se almacenen los datos en la base de datos.

5. ShopService a su vez invoca a los métodos correspondientes de ShopDAO para que este haga la persistencia.
6. Además de guardar la configuración de la tienda se debe guardar una relación entre la tienda y el usuario que la crea. Para ello ShopAction crea un objeto de tipo UserShop e invoca a UserShopService para que se haga la persistencia.
7. UserShopService invoca a la clase UserShopDAO para que este guarde los datos. Una vez hecho esto se avisa al usuario que el registro ha sido satisfactorio.

7.3.3 Caso de uso: Importar elementos

Puesto que la importación de elementos a efectos de invocación de métodos es muy similar para todos los elementos (categorías, productos o proveedores) a continuación se hace el diagrama de interacción para el caso de proveedores. Los demás serían análogos a este.

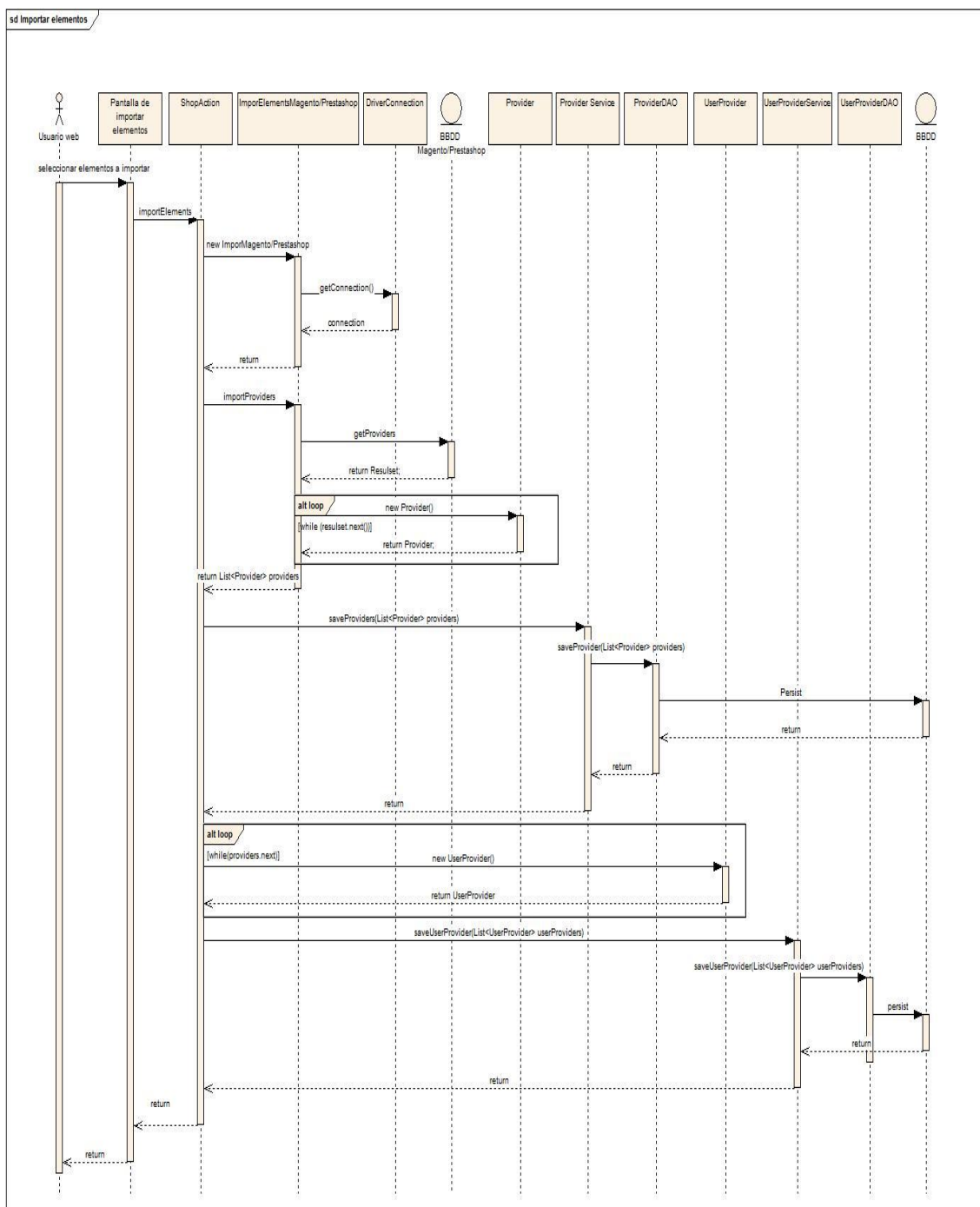


Ilustración 123 Diagramas de interacción: importar proveedores

1. En primer lugar el usuario en la ventana de importar elementos selecciona que elementos de la tienda desea importar
2. A continuación ShopAction obtiene los datos de presentación que le indican los tipos de elementos a importar y sobre que tienda se realiza la operación. Conforme a estos datos ShopAction crea una instancia a ImportElementsMagento si el tipo de tienda a importar elementos es Magento o ImportElementsPrestashop si con la tienda que vamos a iterar es de tipo Prestashop.
3. La clase ImportElements crea una instancia DriverConnection que se encarga de abrir una conexión en la base de datos con la que se va a operar.
4. Posteriormente la clase ShopAction invoca por cada uno de los elementos a importar los métodos necesarios de ImportElements. En este ejemplo se invoca el método importProviders.
5. La clase ImportElements hace las diferentes consultas que es necesario realizar para poder obtener, en este caso, los proveedores de la tienda. En todos los casos es necesario crear varias consultas para obtener los datos de un elemento ya que cada dato que se intenta recoger esta en diferentes tablas de la base de datos desde la que se intentan importar los elementos.
6. Una vez que se obtienen los resultados se va creando la lista de objetos Providers obtenidos. Estos datos se retornan a ShopAction.
7. ShopAction invoca los servicios de ProviderService para que se guarden los datos en la base de datos de la aplicación, este a su vez invoca al DAO correspondiente que hace los persist.
8. Además de guardar cada uno de los proveedores es necesario relacionarlos con el usuario que ha realizado la importación. Para ello se crean instancias UserProvider y se realiza una llamada a UserProviderService para que guarde los datos. Esta clase a su vez invoca al DAO correspondiente que guarda los datos.
9. Si todas estas operaciones se realizan correctamente se informará debidamente al usuario.

7.3.4 Caso de uso: Exportar elementos

Puesto que la exportación de elementos a efectos de invocación de métodos es muy similar para todos los elementos (categorías, productos o proveedores) a continuación se hace el diagrama de interacción para el caso de proveedores. Los demás serían análogos a este.

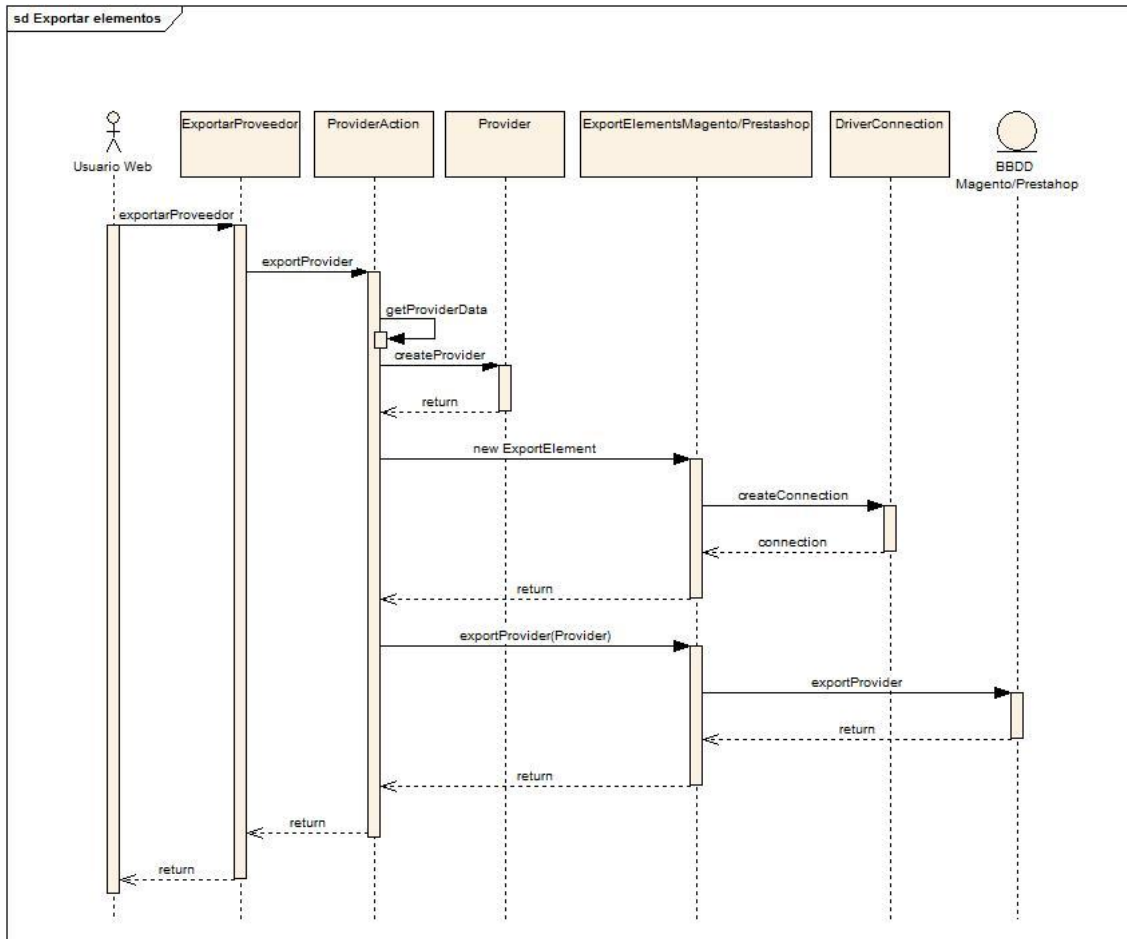


Ilustración 124 Diagramas de interacción: exportar proveedores

1. En primer lugar el usuario sobre un proveedor indica que desea exportarlo y hacia que tienda lo desea exportar.
2. A continuación ProviderAction obtiene los datos de presentación que le indican el proveedor que debe exportar y hacia que tienda se realiza la operación. Conforme a estos datos ProviderAction crea una instancia a ExportElementsMagento si el tipo de tienda a exportar el proveedor es Magento o ExportElementsPrestashop si con la tienda que vamos a iterar es de tipo Prestashop.
3. La clase ExportElements crea una instancia DriverConnection que se encarga de abrir una conexión en la base de datos con la que se va a realizar la exportación
4. Posteriormente ProviderAction invoca al método exportar de la clase ExportElements pasándole como parámetro el proveedor a exportar.
5. La clase ExportElements hace los diferentes INSERTS en la base de datos correspondiente y las consultas que son necesarias hacer para exportar el proveedor. Habrá que hacer varias consultas y varios INSERTS diferentes ya que tanto la base de

datos de Magento como de Prestashop son bastante complejas y los datos están dispersos en varias tablas.

6. Si todas estas operaciones se realizan correctamente se informará debidamente al usuario.

7.3.5 Caso de uso: Generar CSV

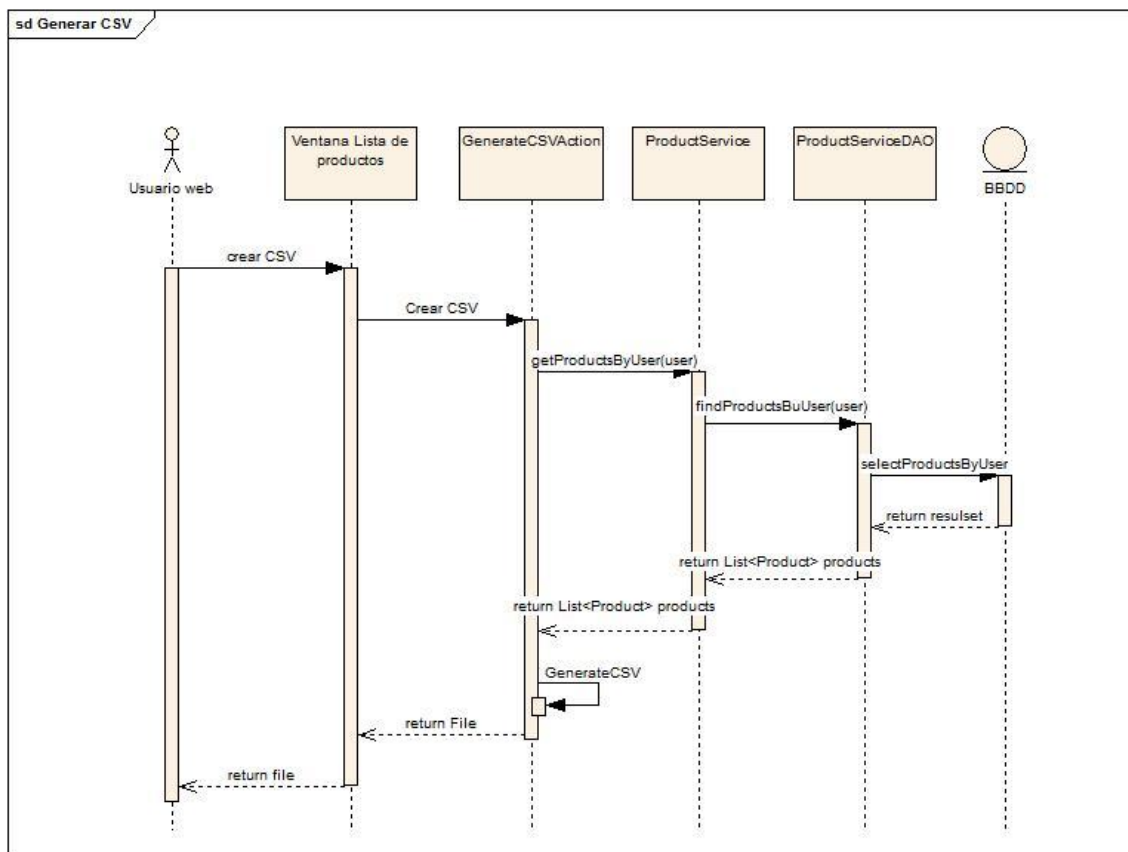


Ilustración 125 Diagramas de interacción: generar CSV

1. El usuario indicará en primer lugar que desea exportar su catálogo de productos a formato CSV.
2. GenerateCSVAction recibe la llamada desde presentación e instancia a ProductService para obtener la lista de productos del usuario
3. ProductService invoca a ProductDAO para que este obtenga directamente los datos de la base de datos.
4. Una vez que GenerateCSVAction tiene la lista con los productos se crea un archivo CSV, compatible tanto con la plataforma Magento como con Prestashop, y se retorna el archivo al usuario.

7.3.6 Caso de uso: Generar XML

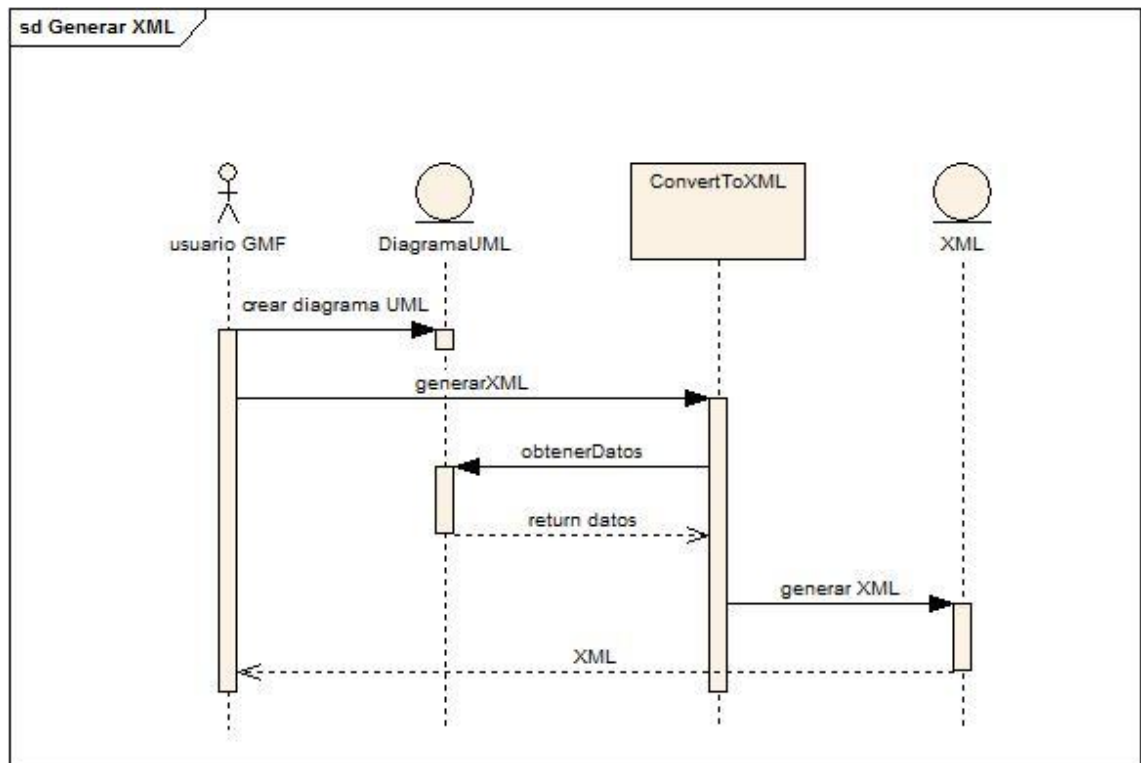


Ilustración 126 Diagramas de interacción: generar XML

En primer lugar el usuario crea mediante los diferentes tipos de elementos UML el usuario crea un diagrama. Cuando el diagrama ha sido creado el usuario indica que desea generar un archivo XML. ConvertXML obtiene los datos del diagrama UML y los convierte a un fichero XML. Este fichero es retornado al usuario

7.4 Diseño de la Base de Datos

7.4.1 Descripción del SGBD Usado

Como framework para la persistencia se utiliza Hibernate. Las clases que se mapean con hibernate se encuentran recogidas en el paquete entities descrito anteriormente.

Para la creación de datos iniciales para el funcionamiento de la web se crea un script SQL encargado de inicializar algunos campos de la base de datos necesarios para el funcionamiento de la web.

El servidor de base de datos elegido para el subsistema software es MySQL.

7.4.2 Integración del SGBD en Nuestro Sistema

El sistema de gestión de bases de datos en nuestro sistema se encuentra integrado de la siguiente manera.

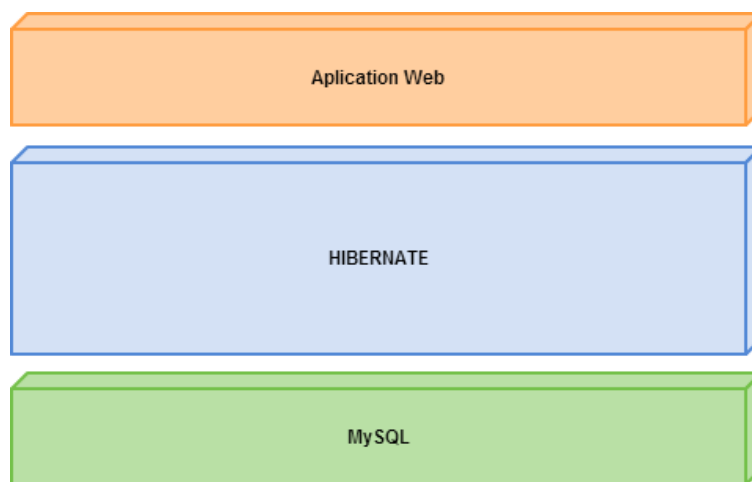


Ilustración 127 SGBD

7.4.3 Diagrama E-R

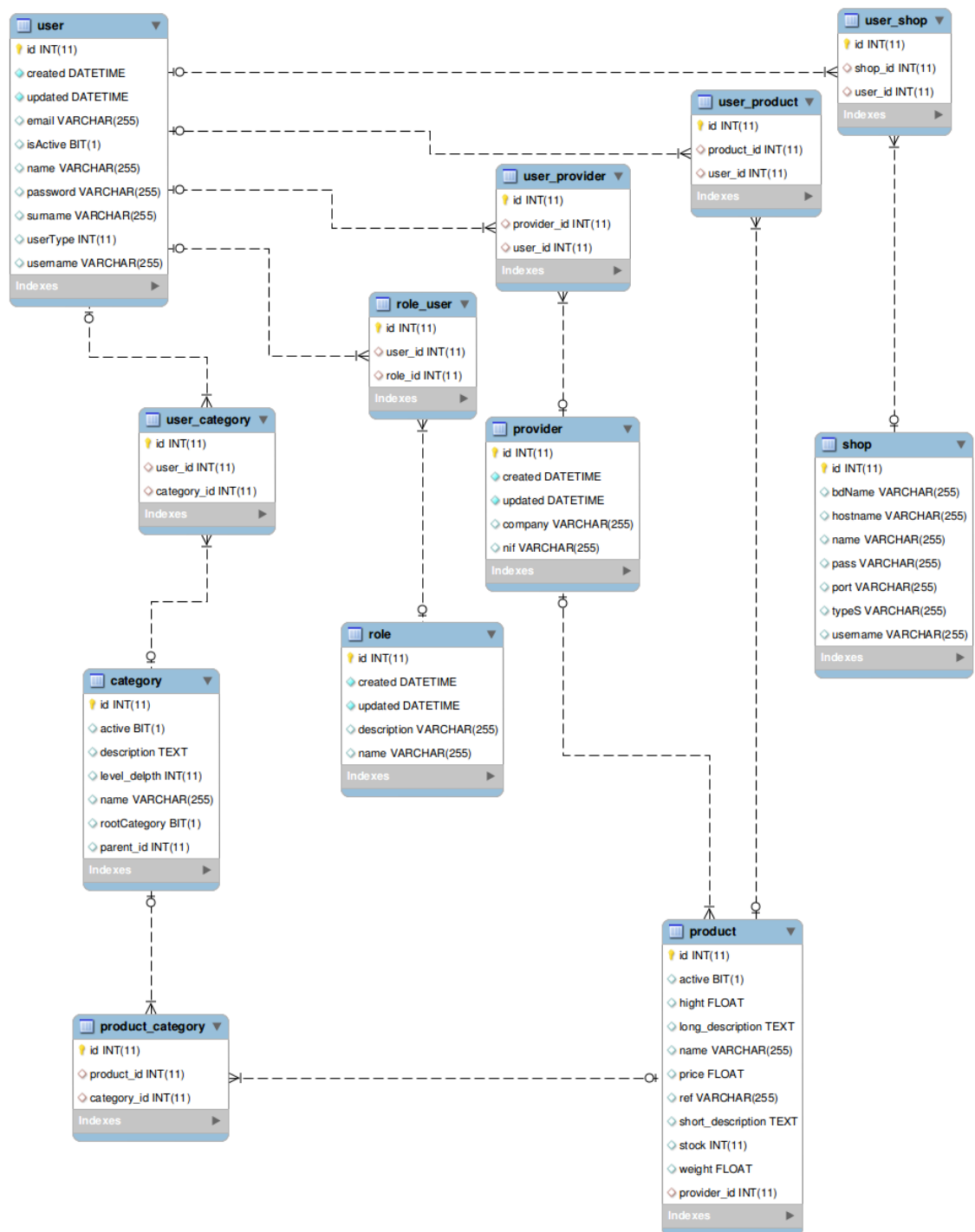


Ilustración 128 Diagrama de entidad relación

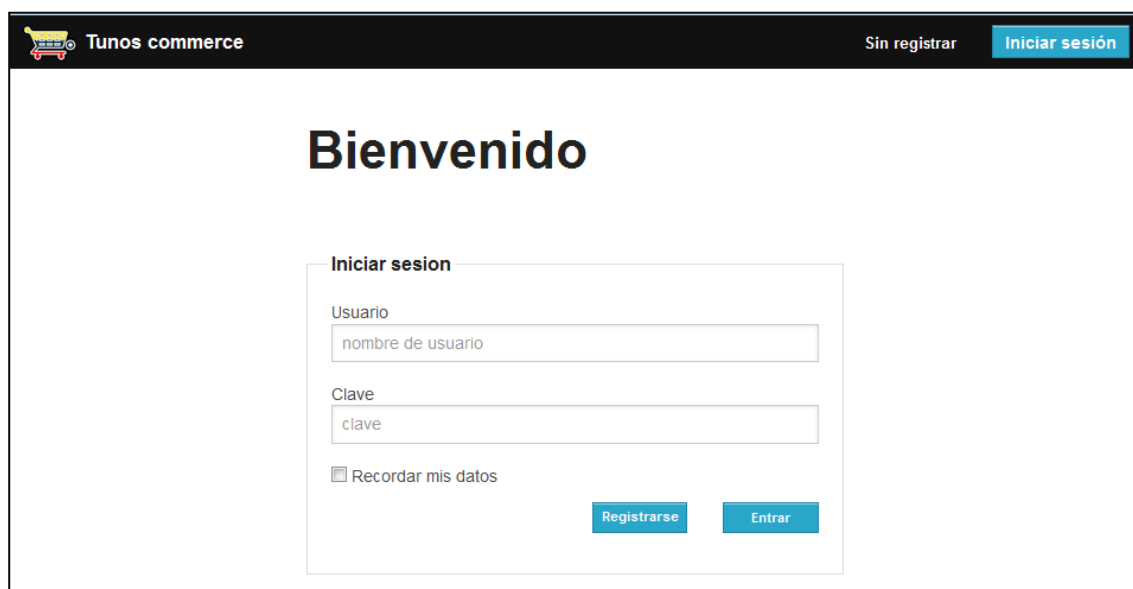
7.5 Diseño de la Interfaz

En este apartado se describe en profundidad el desarrollo de la interfaz planteados durante el análisis. A continuación se mostrará el diseño de las interfaces definitivas de los diferentes subsistemas.

7.5.1 Pantallas plataforma Web

7.5.1.1 Pantalla de inicio de sesión

La aplicación web permite acceso a través de usuario y contraseña, en caso de no estar registrado el usuario podrá acceder desde esta pantalla a la pantalla de registro.



The screenshot shows a web interface for 'Tunos commerce'. At the top, there is a navigation bar with the site logo, the name 'Tunos commerce', and links for 'Sin registrar' and 'Iniciar sesión'. The main content area features a large 'Bienvenido' heading. Below it is a login form titled 'Iniciar sesion'. The form contains two input fields: 'Usuario' with the placeholder text 'nombre de usuario' and 'Clave' with the placeholder text 'clave'. There is a checkbox labeled 'Recordar mis datos'. At the bottom of the form are two buttons: 'Registrarse' and 'Entrar'.

Ilustración 129 Pantalla inicio de sesión

7.5.1.2 Pantalla de registro

En esta pantalla se observa un pequeño formulario que en caso de no estar registrado el usuario sí desea entrar en la aplicación deberá cumplimentar. Si algunos de los datos introducidos no son correctos se informará indicando al usuario cual es el problema

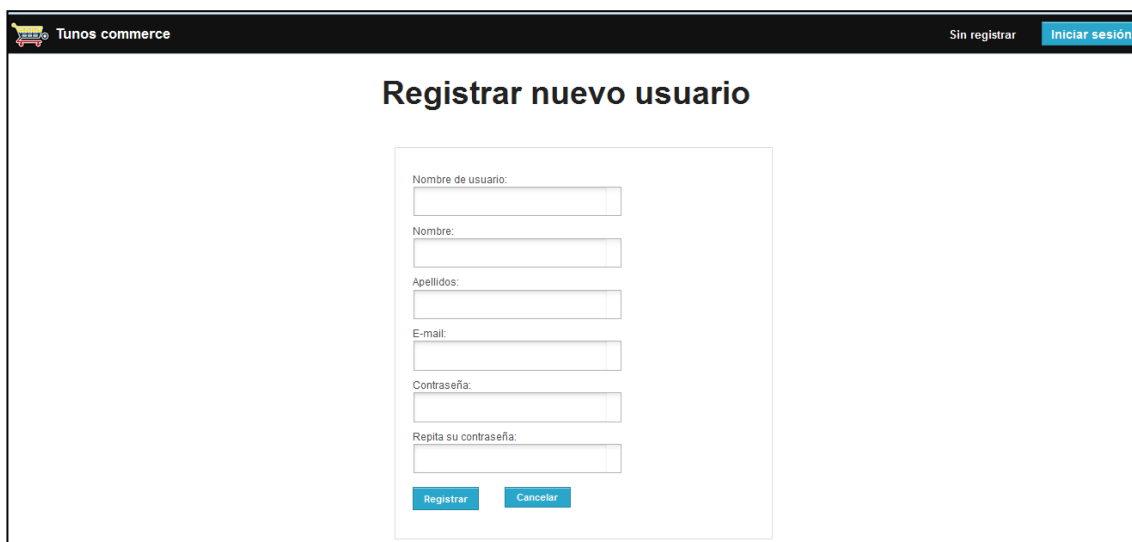


Ilustración 130 Pantalla de registro

7.5.1.3 Pantalla de intro

Tras haberse logueado el usuario esta será la pantalla inicial que mostrará la aplicación web. En esta pantalla como puede observarse se hace una pequeña descripción de la funcionalidad que tiene la aplicación y las ventajas que aporta.

Para acceder al resto de las opciones se utilizará un menú horizontal en el que se podrán seleccionar las diferentes opciones.



Ilustración 131 Pantalla de intro

7.5.1.4 Pantallas de categorías

Pantalla principal que se mostrará para la gestión de categorías. Desde esta pantalla se podrá acceder a todas las funcionalidades que el usuario desee realizar sobre las categorías (Buscar, ver detalle, crear nueva categoría, ver subcategorías ...).

En la tabla de categorías existe la posibilidad de navegar para ver las subcategorías que una categoría puede contener. La tabla estará paginada para tener lo más organizada posible la información de las categorías.

Mostrando el intervalo 1-5 de 7 resultados

NOMBRE	DESCRIPCIÓN	
Mujer	Ropa de mujer	Ver detalle Modificar Eliminar Exportar
Hombre	Ropa de hombre	Ver detalle Modificar Eliminar Exportar
Complementos	Complementos	Ver detalle Modificar Eliminar Exportar
Ropa Deportiva	Ropa deportiva	Ver detalle Modificar Eliminar Exportar

Volver

Página 1 de 1

Ilustración 132 Pantalla de Categorías

7.5.1.4.1 Crear Categoría

La pantalla crear categoría es un pequeño formulario en el que se deberán introducir todos los datos de la categoría que se desea crear.

Ilustración 133 Pantalla crear categoría

Para seleccionar la categoría padre se crea una nueva pantalla en la que se muestran todas las categorías disponibles.

NOMBRE	DESCRIPCIÓN	
Mujer	Ropa de mujer	Seleccionar
Hombre	Ropa de hombre	Seleccionar
Complementos	Complementos	Seleccionar
Ropa Deportiva	Ropa deportiva	Seleccionar

Ilustración 134 Pantalla seleccionar categoría padre

7.5.1.4.2 Modificar categoría

En la pantalla modificar categoría se pueden editar todos los datos relativos a una categoría.

The screenshot shows the 'Modificar Categoría Mujer' form in the Tunos commerce admin interface. The header includes the logo, 'Tunos commerce', the user 'administrador', and a 'Cerrar sesión' button. A breadcrumb trail shows 'Categorías' as the active section, with links to 'Productos', 'Proveedores', 'Tiendas', 'Parseador XML', and 'Datos personales'. The form title is 'Modificar Categoría Mujer'. It contains the following fields and controls:

- Nombre:** A text input field containing 'Mujer'.
- Descripción:** A text area containing 'Ropa de mujer'.
- Categoría Padre:** A dropdown menu showing 'Sin Padre'.
- Modificar padre:** A blue button.
- Activa:** A checkbox that is checked, with the label 'Activa'.
- Modificar:** A blue button.
- Cancelar:** A blue button.

Ilustración 135 Modificar categoría

7.5.1.4.3 Eliminar Categoría

Una vez seleccionada la opción eliminar categoría esta opción se verifica que realmente quiere eliminarse para posteriormente si se acepta realizar la operación.

The screenshot shows the 'Confirma que desea eliminar la Categoría Mujer' form in the Tunos commerce admin interface. The header is identical to the previous screenshot. The breadcrumb trail is the same. The form title is 'Confirma que desea eliminar la Categoría Mujer'. It contains the following fields and controls:

- Nombre:** 'Mujer'
- Descripción:** 'Ropa de mujer'
- Categoría Padre:** 'Sin Padre'
- Activa:** 'Activado'
- Eliminar:** A blue button.
- Cancelar:** A blue button.

Ilustración 136 Eliminar Categoría

7.5.1.4.4 Ver detalle de Categoría

El sistema permitirá consultar los detalles de las categorías creadas. En esta ventana se podrá ver toda la información relativa a una determinada categoría.

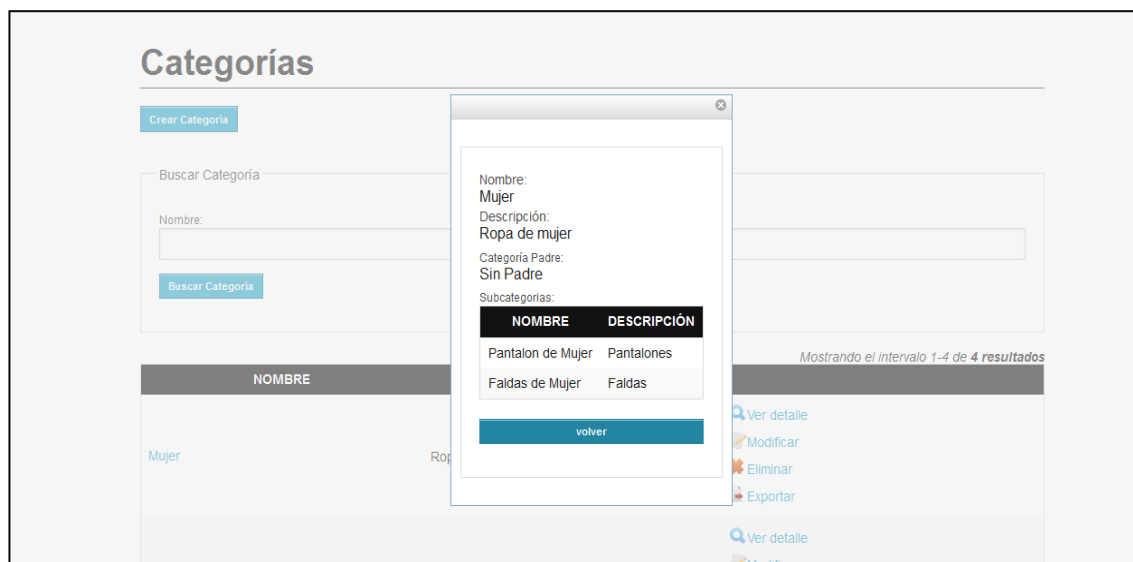


Ilustración 137 Ver Detalle Categoría

7.5.1.4.5 Exportar Categoría

Desde este pequeño formulario permitirá exportar una categoría a la tienda seleccionada.

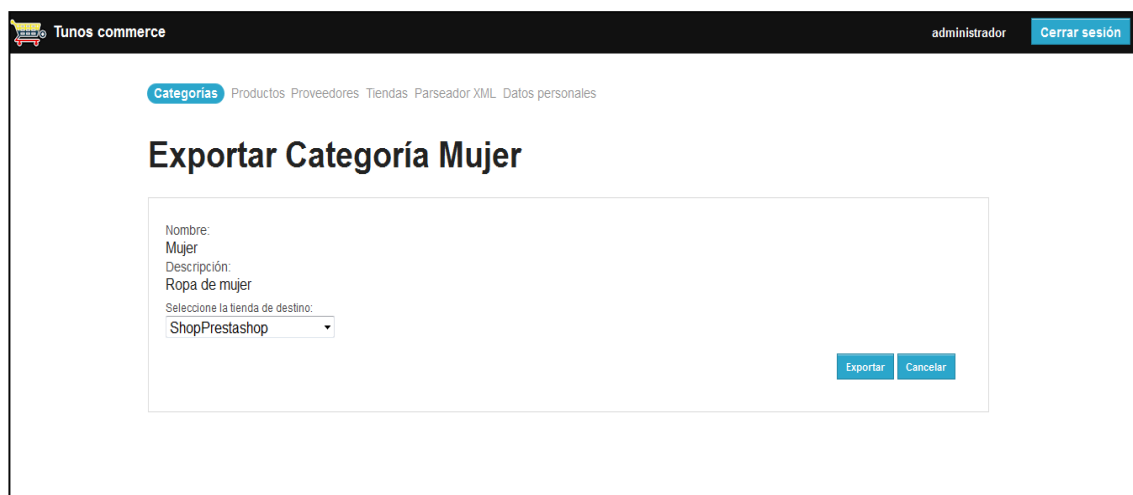


Ilustración 138 Exportar Categoría

7.5.1.5 Pantallas de productos

Pantalla principal que se mostrará para la gestión de productos. Desde esta pantalla se podrá acceder a todas las funcionalidades que el usuario desee realizar sobre productos (crear producto, crear CVS, modificar producto)

En la tabla de productos se mostrará la información de cada producto existente y presentará las opciones disponibles a realizar el (ver detalle, modificar, eliminar o exportar.).

 Tunos commerce
 administrador [Cerrar sesión](#)

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Productos

[Crear un nuevo producto](#)
[Generar CSV de Productos](#)

Buscar Producto

Nombre:
 Proveedor: Sin Proveedor
 [Seleccionar](#) [Quitar](#)

[Buscar Producto](#)

Mostrando el intervalo 1-5 de 16 resultados

NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR
iPod Nano	Nuevo diseño. Nuevas aplicaciones. Ahora disponible en 8 y 16 Go. iPod nano, más rock que nunca.	124.582	2	Apple Computer, Inc.
iPod shuffle	iPod shuffle, el walkman más portátil del mundo, ahora en azul, verde, rosa y rojo.	66.0535	2	Apple Computer, Inc.
MacBook Air	MacBook Air es ultra fino, ultra portátil y ultra diferente de todo el resto. Pero no se pierden kilos y centímetros en tan solo una noche. Todo esto es el resultado de un nuevo invento de normas. De un sinfín de novedades sin cable. Y de una revolución en el diseño. Con MacBook Air, la informática móvil adquiere una nueva dimensión.	1504.18	2	Apple Computer, Inc.
MacBook	MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.	1170.57	2	Apple Computer, Inc.
iPod touch	Interfaz multi-touch revolucionaria Pantalla panorámica color de 3,5 pulgadas Wi-Fi (802.11b/g) 8 mm de espesor Safari, YouTube, iTunes Wi-Fi Music Store, Correo, Mapas, Bolsa, El tiempo, Notas	241.639	2	

Página 1 de 4 [Siguiente](#) [Último](#)

Ilustración 139 Pantalla de productos

7.5.1.5.1 Pantalla crear producto

La pantalla crear producto es un pequeño formulario en el que se deberán introducir todos los datos del producto que se desea crear.

Tunos commerce administrador Cerrar sesión

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Crear producto

Nombre:

Descripción Corta:

Descripción Larga:

Número de Referencia:

Precio:

Número de Unidades:

Alto:

Ancho:

Proveedor: Sin Proveedor [Seleccionar](#)

Categorías: Producto no asignado a ninguna categoría [Añadir Categorías](#)

☐ Activo

[Crear](#) [Cancelar](#)

Ilustración 140 Pantalla crear producto

Desde el formulario habrá una opción que permitirá seleccionar una o varias categorías a las que pertenecerá el producto a crear.

Tunos commerce administrador Cerrar sesión

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Categorizar el producto Ordenador Portátil

Categorías Seleccionadas

NOMBRE	DESCRIPCIÓN	
Mujer	Mujer	Quitar
Hombre	Hombre	Quitar

Lista de Categorías disponibles

Mostrando el intervalo 1-5 de 6 resultados

NOMBRE	DESCRIPCIÓN	
Mujer	Ropa de mujer	Seleccionar
Hombre	Ropa de hombre	Seleccionar
Complementos	Complementos	Seleccionar
Zapatos de vestir	Zapatos de Vestir	Seleccionar
Zapatillas deportivas	Zapatillas deportivas	Seleccionar

Volver Página 1 de 2 Siguiente > Último >

Aceptar Cancelar

Ilustración 141 Pantalla Seleccionar Categorías

Desde esté formulario también se podrá relacionar el producto a crear con un determinado proveedor.

Tunos commerce administrador Cerrar sesión

Categorías Productos **Proveedores** Tiendas Parseador XML Datos personales

Seleccionar Proveedor

Mostrando el intervalo 1-5 de 8 resultados

COMPANÍA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
ZARA	12222	24.05.2013	24.05.2013	Seleccionar
H&M	11111	24.05.2013	24.05.2013	Seleccionar
Pinkies	0000	24.05.2013	24.05.2013	Seleccionar
Mango	8765	24.05.2013	24.05.2013	Seleccionar
pull&bear	8765	24.05.2013	24.05.2013	Seleccionar

Página 1 de 2 Siguiente > Último >

Cancelar

Ilustración 142 Pantalla Seleccionar Proveedor

7.5.1.5.2 Pantalla modificar producto

Desde este formulario se podrán modificar los diferentes campos de un determinado producto. Al igual que en el formulario de crear desde aquí también se podrán modificar o añadir las categorías a las que pertenece un producto así como el proveedor.

Tunos commerce administrador [Cerrar sesión](#)

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Modificar producto MacBook

Nombre:
MacBook

Descripción Corta:
MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.

Descripción Larga:
Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin

Número de Referencia:
123345555-MF

Precio:
1170.57

Número de Unidades:
2

Alto:
0.0

Ancho:
0.75

Proveedor:
Apple Computer, Inc.
[Seleccionar](#)

Categorías:

NOMBRE	DESCRIPCIÓN
Portátiles	El último procesador Intel, un disco duro más grande, con profusión de memoria y otras novedades. Todo en sólo 2.59 cm libres de cualquier obstrucción. Las nuevas portátiles Mac cumplir rendimiento, potencia y conectividad de una computadora de escritorio. Sin la parte del escritorio.

[Añadir Categorías](#)

☐ Activo

[Modificar](#) [Cancelar](#)

Ilustración 143 Pantalla modificar Producto

7.5.1.5.3 Pantalla eliminar Producto

Una vez seleccionada la opción eliminar producto esta opción se verifica que realmente quiere eliminarse para posteriormente si se acepta realizar la operación.

Confirma que desea eliminar el producto MacBook

Nombre: MacBook
 Descripción Corta: MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.
 Descripción Larga: Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin dificultad tus aplicaciones preferidas.
 Número de Referencia:
 Precio: 1170.57
 Número de Unidades: 2
 Alto: 0.0
 Ancho: 0.75
 Proveedor: Apple Computer, Inc.
 Activo: Desactivado
 Categorías:

NOMBRE	DESCRIPCIÓN
Portátiles	El último procesador intel, un disco duro más grande, con profusión de memoria y otras novedades. Todo en sólo 2.59 cm libres de cualquier obstrucción. Las nuevas portátiles Mac: cumplir rendimiento, potencia y conectividad de una computadora de escritorio. Sin la parte del escritorio.

Eliminar Cancelar

Ilustración 144 Pantalla eliminar Producto

7.5.1.5.4 Pantalla de ver detalle de producto

El sistema permitirá consultar los detalles de los productos creadas. En esta ventana se podrá ver toda la información relativa a un determinado producto. Para facilitar la gestión de los productos y su información podrá ver un código QR que se generará a partir del número de referencia de cada uno de los productos.

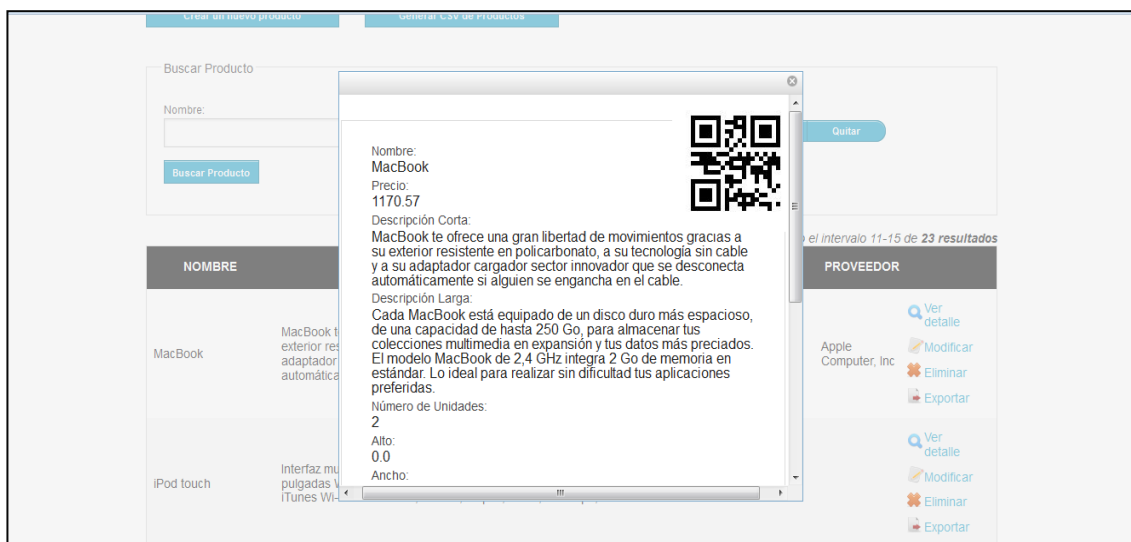


Ilustración 145 Pantalla de ver detalle de producto

7.5.1.5.5 Pantalla exportar producto

El sistema permite exportar un producto creado a las tiendas configuradas en la web. Este es el formulario que permitirá realizar esta operación. En él se mostrará la información del producto a exportar y se podrá seleccionar la tienda de destino.

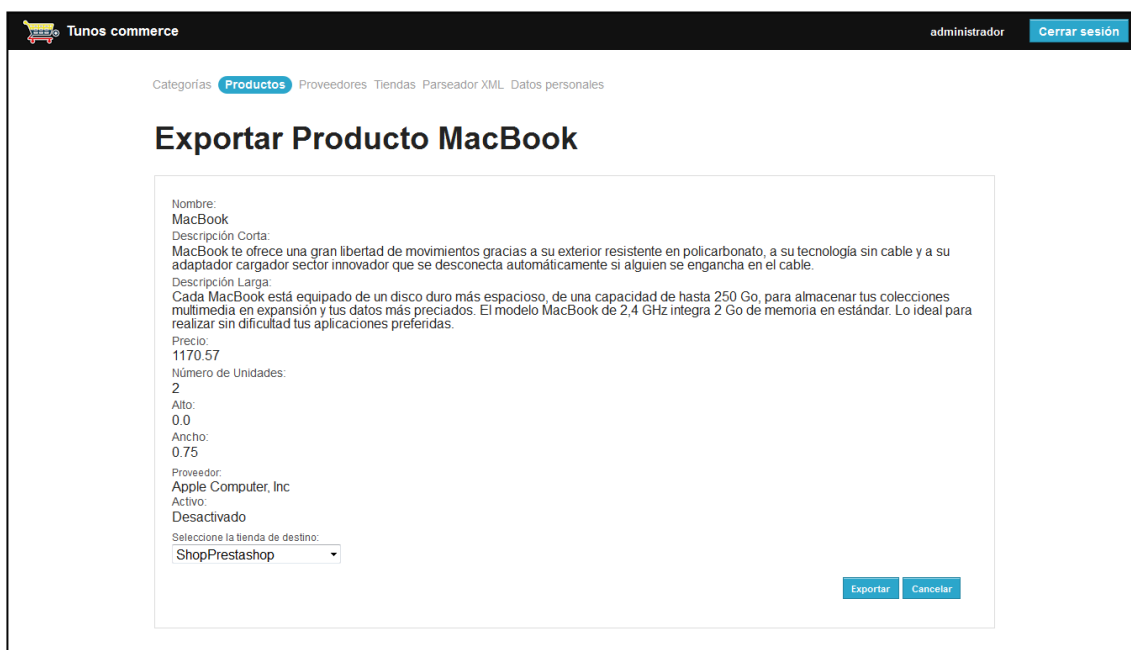


Ilustración 146 Pantalla exportar producto

7.5.1.6 Pantallas de proveedores

Pantalla principal que se mostrará para la gestión de proveedores. Desde esta pantalla se podrá acceder a todas las funcionalidades que el usuario desee realizar sobre los proveedores (crear proveedor, modificar, eliminar...)

En la tabla de proveedores se mostrará la información de cada proveedor existente y presentará las opciones disponibles a realizar el (ver detalle, modificar, eliminar o exportar.).

The screenshot shows the 'Proveedores' (Suppliers) management page. At the top, there's a navigation bar with 'Tunos commerce' and user roles 'administrador' and 'Cerrar sesión'. Below this is a breadcrumb trail: 'Categorías Productos Proveedores Tiendas Parseador XML Datos personales'. The main title is 'Proveedores'. A button 'Crear nuevo proveedor' is visible. Below it is a search form with fields for 'Compañía:' and 'CIF:', and a 'Buscar Proveedor' button. The search results are displayed in a table with columns: 'COMPANÍA', 'CIF', 'FECHA DE CREACIÓN', and 'FECHA DE MODIFICACIÓN'. The table shows four suppliers: LEVIS, PEPE JEANS, Apple Computer, Inc, and Shure Incorporated. Each row has a set of action links: 'Ver detalle', 'Modificar', 'Eliminar', and 'Exportar'. At the bottom, there's a pagination bar showing 'Página 2 de 3'.

COMPANÍA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
LEVIS	8721	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar
PEPE JEANS	8722	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar
Apple Computer, Inc		06.07.2013	06.07.2013	Ver detalle Modificar Eliminar Exportar
Shure Incorporated		06.07.2013	06.07.2013	Ver detalle Modificar Eliminar Exportar

Mostrando el intervalo 6-10 de 11 resultados

<|Primero <Anterior Página 2 de 3 Siguiente> Último>

Ilustración 147 Pantalla de proveedores

7.5.1.6.1 Pantalla de crear proveedor

La pantalla crear proveedor es un pequeño formulario en el que se deberán introducir los datos necesarios.

Ilustración 148 Pantalla de crear proveedor

7.5.1.6.2 Pantalla de modificar proveedor

En la pantalla modificar proveedor se pueden editar todos los datos relativos a un proveedor.

Ilustración 149 Pantalla modificar proveedor

7.5.1.6.3 Pantalla eliminar proveedor

Una vez seleccionada la opción eliminar proveedor esta opción se verifica que realmente quiere eliminarse para posteriormente si se acepta realizar la operación.

Ilustración 150 Pantalla eliminar Proveedor

7.5.1.6.4 Pantalla de ver detalle

El sistema permitirá consultar los detalles de los proveedores ya creados.

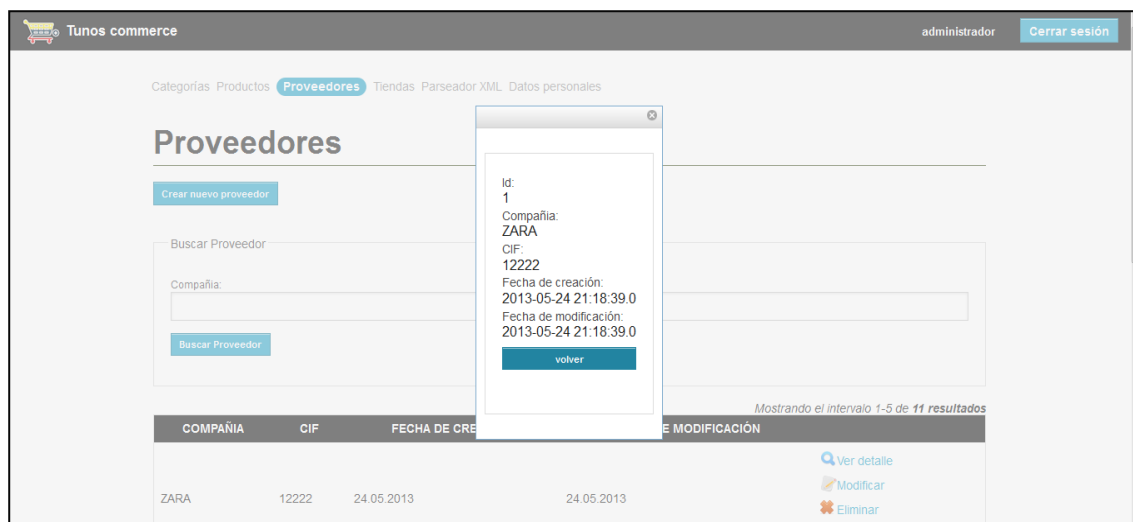


Ilustración 151 Pantalla de Ver detalle

7.5.1.6.5 Pantalla de exportar proveedor

Desde este pequeño formulario permitirá exportar un proveedor a la tienda seleccionada.



Ilustración 152 Pantalla exportar proveedor

7.5.1.7 Pantallas de tiendas

Pantalla principal que se mostrará para la gestión de tiendas. Desde esta pantalla se podrá acceder a todas las funcionalidades que el usuario desee realizar sobre tiendas.

En la tabla de tiendas se mostrará la información de cada tienda existente y presentará las opciones disponibles a realizar el (ver detalle, modificar, eliminar, importar elementos...).

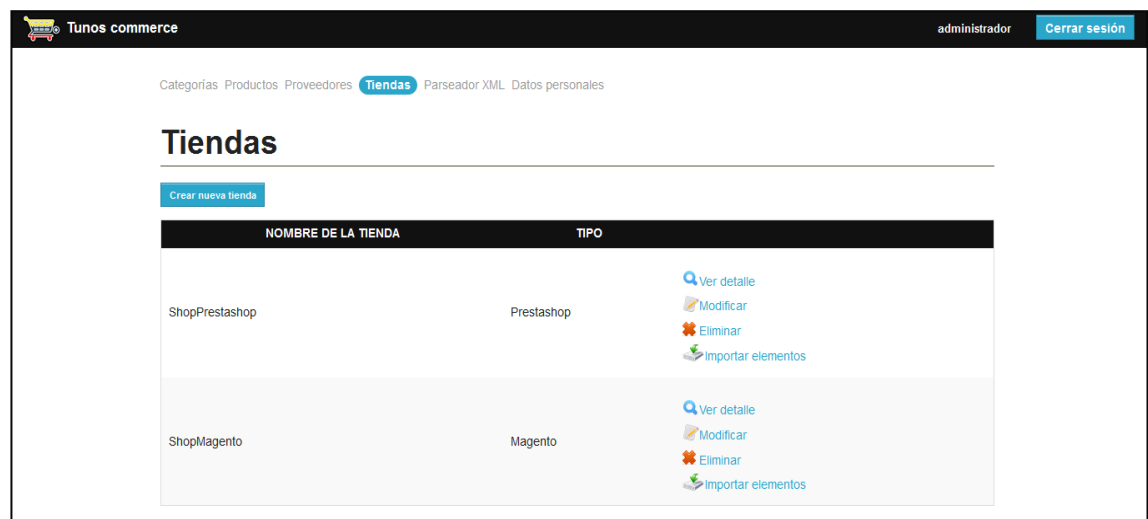


Ilustración 153 Pantalla de tiendas

7.5.1.7.1 Pantalla crear tienda

La pantalla crear tienda es un pequeño formulario en el que se deberán introducir todos los datos de configuración de la tienda.

The screenshot shows the 'Crear nueva tienda' form in the Tunos commerce system. The header is the same as the previous screenshot. The main title is 'Crear nueva tienda'. The form contains the following fields: 'Nombre de la tienda' (text input), 'Tipo' (dropdown menu with 'Prestashop' selected), 'Nombre de la base de datos' (text input), 'Host de la BBDD' (text input), 'Número de puerto de la BBDD' (text input), 'Usuario de la BBDD' (text input), 'Password de la BBDD' (text input), and 'Repita el Password de la BBDD' (text input). At the bottom right are 'Crear' and 'Cancelar' buttons.

Ilustración 154 Pantalla crear tienda

7.5.1.7.2 Pantalla modificar tienda

En la pantalla modificar tienda se pueden editar todos los datos relativos a una tienda ya creada.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores **Tiendas** Parseador XML Datos personales

Modificar tienda ShopPrestashop

Nombre de la tienda:
ShopPrestashop

Tipo:
Prestashop

Host de la BBDD:
156.35.98.46

Número de puerto de la BBDD:
3306

Usuario de la BBDD:
root

Password de la BBDD:
••••

Repita el Password de la BBDD:
••••

Modificar Cancelar

Ilustración 155 Pantalla modificar tienda

7.5.1.7.3 Pantalla eliminar

Una vez seleccionada la opción eliminar tienda esta opción se verifica que realmente quiere eliminarse para posteriormente si se acepta realizar la operación.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores **Tiendas** Parseador XML Datos personales

Confirma que desea eliminar la tienda ShopPrestashop

Nombre de la tienda:
ShopPrestashop

Tipo:
Prestashop

Host de la BBDD:
156.35.98.46

Número de puerto de la BBDD:
3306

Usuario de la BBDD:
root

Eliminar Volver

Ilustración 156 Pantalla eliminar tienda

7.5.1.7.4 Pantalla de ver detalle

El sistema permitirá consultar los detalles de las tiendas creadas. En esta ventana se podrá ver toda la información relativa a una determinada tienda.

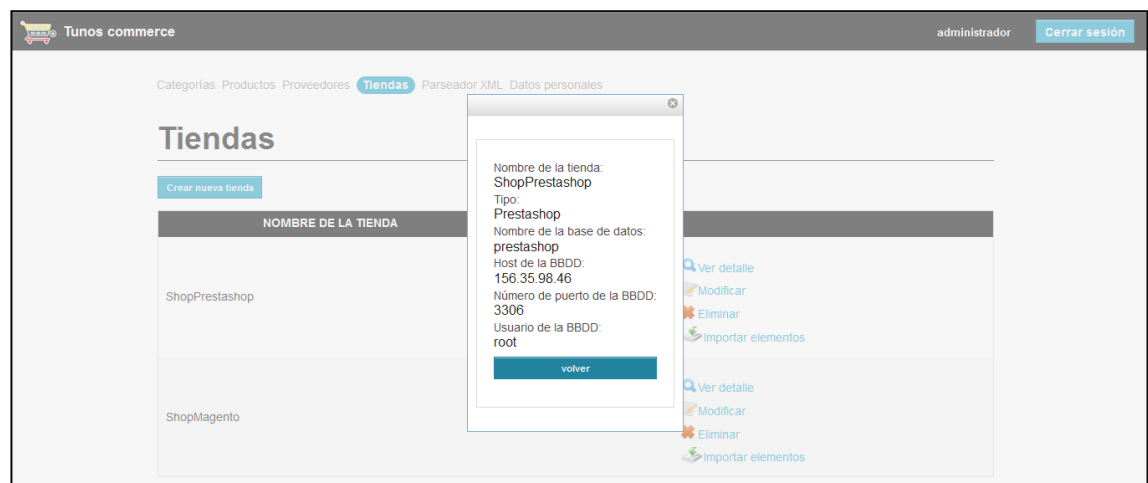


Ilustración 157 Pantalla de ver detalle tienda

7.5.1.7.5 Importar elementos

El sistema permitirá importar elementos desde las tiendas creadas. Aquí se seleccionan los elementos a importar (Categorías, productos o proveedores) .



Ilustración 158 Importar elementos tienda

7.5.1.8 Pantalla parseador XML

Desde esta pantalla se podrán importar elementos a (categorías , productos o proveedores) a partir de un XML.

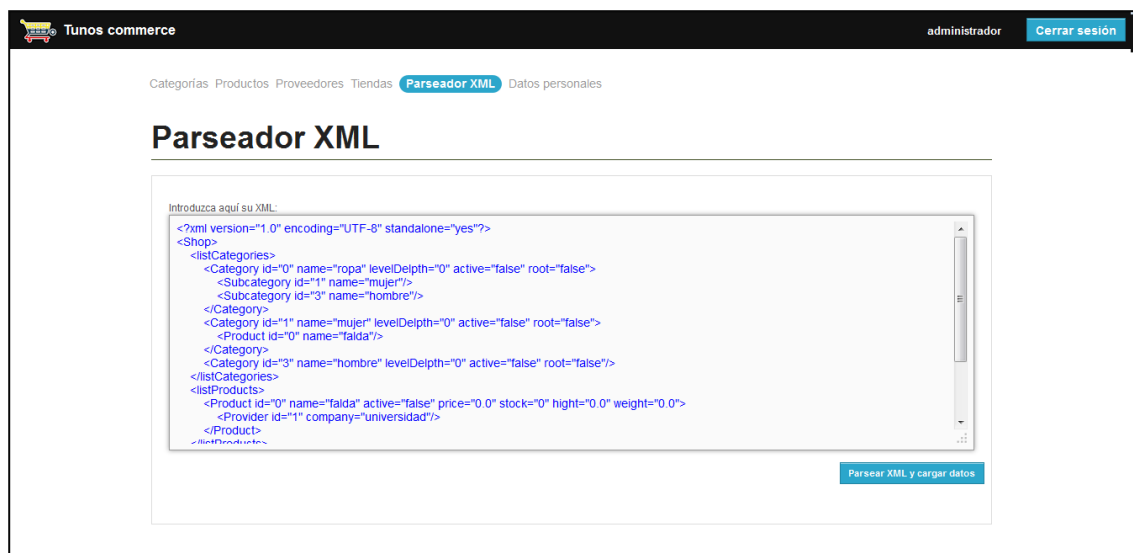


Ilustración 159 Pantalla parseador XML

7.5.1.9 Pantalla de datos personales

En esta pantalla podrán verse los datos personales del usuario logueado.



Ilustración 160 Pantalla de datos personales

7.5.1.9.1 Pantalla modificar usuario

El usuario podrá modificar sus datos personales mediante este formulario en el que simplemente se han de modificar y guardar.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores Tiendas Parseador XML Datos personales

Modificar datos de usuario administrador

Nombre:

Apellidos:

Contraseña:

Contraseña:

Repita su contraseña:

Aceptar Cancelar

Ilustración 161 Pantalla modificar usuario

7.5.2 Pantallas Plataforma GMF

El interfaz del GMF está compuesto únicamente de una sola pantalla en la que se podrán crear el UML que posteriormente generará el XML.

En la parte derecha de la pantalla se encuentran iconos significativos de cada una de las entidades o relaciones que se pueden crear. Para crear elementos en la parte UML simplemente hará falta arrastrar el icono y editar sobre la pantalla.

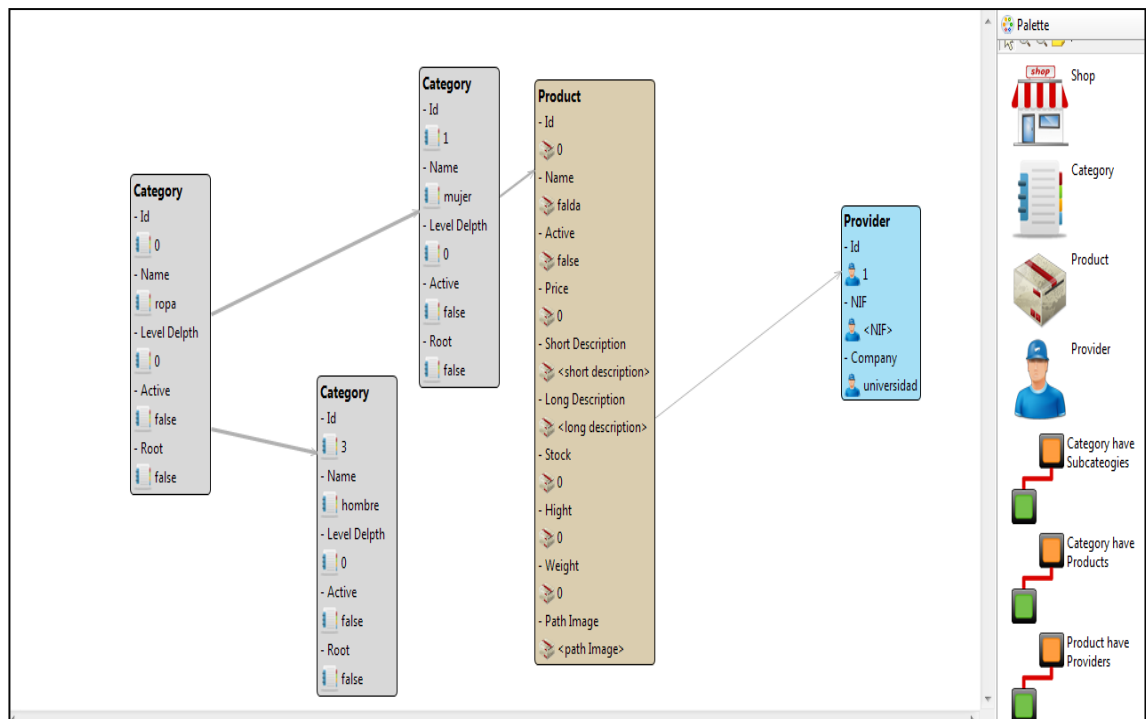


Ilustración 162 Pantalla GMF

7.6 Especificación Técnica del Plan de Pruebas

7.6.1 Pruebas Unitarias

Las pruebas unitarias se llevarán a cabo durante toda la fase de desarrollo para asegurar que no se implementa nuevo código sobre código erróneo. Para ello se probarán tanto con datos correctos como incorrectos para comprobar que la aplicación está blindada para obtener una aplicación funcional aunque el usuario intente meter datos erróneos.

7.6.2 Pruebas de Integración y del Sistema

Las siguientes pruebas se realizarán cuando se haya finalizado la implementación del sistema completo, corrigiendo los fallos a medida que se encuentren y repitiendo la comprobación cuando se crea que se haya resuelto la deficiencia

7.6.2.1 Caso de uso: registrarse

<u>Caso de Uso 1: registrarse</u>	
Prueba	Resultado Esperado
Registrar un usuario no existente	El sistema valida que los datos del usuario son correctos: • Todos los campos requeridos están cubiertos • El nombre de usuario no está repetido • El email esta correctamente formado • Y las contraseña y la repetición de contraseña coinciden Una vez hecho y comprobado que todo es correcto se guarda el usuario en la base de datos. Para ello se avisa al usuario que la operación se ha realizado con éxito. Para comprobar que el usuario está correctamente registrado se hace la prueba de logueo en el sistema.
Prueba	Resultado Esperado
Intentar registrar un usuario que ya existe	El sistema antes de crear el nuevo usuario comprueba que todos los campos del formulario son válidos y que el usuario no existe. En este caso como el usuario ya está registrado el validador lanzará un error indicando que el nombre de usuario ya existe en el sistema.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios y se vuelve a la ventana de login.

7.6.2.2 Caso de uso: loguearse

Caso de Uso 2: loguearse	
Prueba	Resultado Esperado
Introducir usuario y contraseña correctos	El sistema comprueba que el usuario y contraseña son los adecuados y le da acceso al sistema.
Prueba	Resultado Esperado
Introducir usuario valido y contraseña incorrecta	El sistema busca al usuario y comprueba que la contraseña no es la correcta por lo que no le da acceso al sistema avisando al usuario que las credenciales de acceso no son las correctas.
Prueba	Resultado Esperado
Introducir usuario inválido y contraseña válida	El sistema busca al usuario y comprueba que no existe en el sistema por lo que no le da acceso al sistema avisando al usuario que las credenciales de acceso no son las correctas.

7.6.2.3 Caso de uso: configurar tienda

Caso de Uso 3: configurar tienda	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de tienda y guarda la opción	El sistema comprueba que todos los campos del formulario de la tienda que son obligatorios están cubiertos, una vez comprobado esto determina si la contraseña y su repetición coinciden. Puesto que es así se guarda el registro de una nueva tienda.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la tienda e intenta guardar los cambios	El sistema comprueba que todos los campos del formulario de la tienda que son obligatorios están cubiertos, una vez comprobado esto determina si la contraseña y su repetición coinciden. En este caso ambas contraseñas no coinciden por lo que se avisará al usuario para que revise estos errores.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema cancela la operación de configuración de tienda y vuelve a la página de listado de tiendas.

7.6.2.4 Caso de uso: crear categoría

Caso de Uso 4 : crear categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado almacena la nueva categoría en el sistema, avisando al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no guarda la categoría y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no almacena la categoría y vuelve a la pantalla de listados de categorías.

7.6.2.4.1 Subcaso de uso: crear una subcategoría

Caso de uso 4: Crear Categoría. Subcaso de Uso : crear una subcategoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y selecciona una categoría ya creada en el sistema como categoría padre	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado almacena la nueva categoría en el sistema y la relaciona con la categoría que el usuario ha seleccionado como padre. Una vez guardados los cambios avisa al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no guarda la categoría y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no almacena la categoría y vuelve a la pantalla de listados de categorías.

7.6.2.4.2 Subcaso de uso: ver detalle de una categoría

Caso de uso 4: Crear Categoría. Subcaso de Uso : ver detalle de una categoría	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de una categoría	Se obtiene el conjunto de datos de la categoría de la base de datos y se muestran al usuario.

7.6.2.4.3 Subcaso de uso: eliminar una categoría

Caso de uso 4: Crear Categoría. Subcaso de uso : eliminar una categoría	
Prueba	Resultado Esperado
El usuario intenta eliminar una categoría sin hijos	El sistema comprueba que la categoría a eliminar no tiene ninguna subcategoría asignada, por lo que elimina la correctamente la categoría y le indica al usuario que la operación se ha realizado correctamente.
Prueba	Resultado Esperado
El usuario intenta eliminar una categoría que contiene subcategorías	El sistema comprueba que la categoría a eliminar tiene subcategorías asignadas, por lo que rechaza la petición de borrado de la categoría e indica al usuario que la operación no puede realizarse ya que la categoría que pretende eliminar tiene subcategorías asignadas.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no elimina la categoría y vuelve a la pantalla de listados de categorías.

7.6.2.4.4 Subcaso de uso: modificar una categoría

Caso de uso 4: Crear Categoría. Subcaso de uso : modificar una categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de categoría y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado modifica la categoría seleccionada. Cuando se terminan de guardar los cambios avisa al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación de la categoría e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no modifica la categoría y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no modifica la categoría y vuelve a la pantalla de listados de categorías.

7.6.2.4.5 Subcaso de uso: relacionar productos a una o más categorías

Caso de uso 4: Crear Categoría. Subcaso de uso: relacionar productos a una o más categorías	
Prueba	Resultado Esperado
El usuario asocia un producto con una o más categorías	El sistema crea las asociaciones entre el producto y las categorías que el usuario ha elegido y guarda los cambios en la base de datos.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

7.6.2.5 Caso de uso: crear un producto

Caso de Uso 5 : crear producto	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de productos y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado almacena el nuevo producto en el sistema, avisando al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la productos e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no guarda el producto y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no almacena el nuevo producto y vuelve a la pantalla de listados de productos

7.6.2.5.1 Subcaso de uso: ver detalle de un producto

Caso de uso 5: crear producto. Subcaso de Uso : ver detalle de un producto	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un producto	Se obtiene el conjunto de datos del producto y se genera el código QR y se muestran los datos al usuario.

7.6.2.5.2 Subcaso de uso: eliminar un producto

<i>Caso de uso 5: crear producto. Subcaso de uso : eliminar un producto</i>	
Prueba	Resultado Esperado
El usuario elimina un producto	El sistema elimina el producto de la base de datos y lo desvincula de todas las entidades con las que estuviera relacionado. Una vez realizada la operación envía un aviso al usuario indicando que todo ha ido correctamente
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios y vuelve a la pantalla de listado de productos

7.6.2.5.3 Subcaso de uso: modificar un producto

<i>Caso de uso 5: crear producto. Subcaso de uso : modificar un producto</i>	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de producto y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado modifica el producto seleccionado. Cuando se terminan de guardar los cambios avisa al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del producto e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no modifica el producto y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no modifica el producto y vuelve a la pantalla de listados de productos

7.6.2.5.4 Subcaso de uso: relacionar producto con proveedor

<i>Caso de uso 5: crear producto. Subcaso de uso : relacionar producto con proveedor</i>	
Prueba	Resultado Esperado
El usuario asocia un producto con un proveedor	El sistema asocia relaciona el producto con el proveedor indicados y guarda los cambios en la base de datos.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

7.6.2.6 Caso de uso: crear un proveedor

Caso de uso 6: crear proveedor	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de proveedor y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado almacena el nuevo proveedor en el sistema, avisando al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la proveedor intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no guarda el proveedor y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no almacena el nuevo proveedor y vuelve a la pantalla de listados de proveedores

7.6.2.6.1 Subcaso de uso: ver detalle de un proveedor

Caso de uso 6: crear proveedor Subcaso de Uso : ver detalle de un proveedor	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un proveedor	Se obtiene el conjunto de datos del proveedor de la base de datos y se muestran al usuario.

7.6.2.6.2 Subcaso de uso: eliminar un proveedor

Caso de uso 6: crear proveedor Subcaso de uso : eliminar un proveedor	
Prueba	Resultado Esperado
El usuario elimina un proveedor	El sistema elimina el proveedor de la base de datos y lo desvincula de todas las entidades con las que estuviera relacionado. Una vez realizada la operación envía un aviso al usuario indicando que todo ha ido correctamente
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios y vuelve a la pantalla de listado de proveedores

7.6.2.6.3 Subcaso de uso: modificar un proveedor

Caso de uso 6: crear proveedor Subcaso de uso : modificar un proveedor	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de proveedor y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, una vez verificado que el formulario está correctamente cumplimentado modifica el proveedor. Cuando se terminan de guardar los cambios avisa al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del proveedor e intenta guardar los cambios	El sistema comprueba que hay campos obligatorios en el formulario que no están rellenos por lo que no modifica el proveedor y le indica al usuario que partes de éste no están correctamente cumplimentados.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema no modifica el proveedor y vuelve a la pantalla de listados de productos

7.6.2.7 Caso de uso: importar elementos

Caso de Uso 7 : importar elementos	
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada correctamente	El sistema importa los elementos seleccionados por el usuario y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada incorrectamente	El sistema intenta conectarse a la tienda pero al ser la configuración incorrecta la conexión no puede realizarse y por lo tanto no se pueden importar los elementos. Se informa al usuario que la operación no pudo realizarse con éxito.
Prueba	Resultado Esperado
El usuario intenta importar elementos de una tienda cuya base de datos está corrupta.	El sistema intenta importar los elementos pero al estar la base de datos corrupta no puede realizar la conexión. Se informa al usuario que la operación no pudo realizarse con éxito
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

7.6.2.8 Caso de uso: exportar elementos

Caso de Uso 8 : exportar elementos	
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda configurada correctamente	El sistema exporta los elementos y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
Prueba	Resultado Esperado
El usuario exportar elementos a una tienda configurada incorrectamente	El sistema intenta conectarse a la tienda pero al ser la configuración incorrecta la conexión no puede realizarse y por lo tanto no se pueden exportar los elementos. Se informa al usuario que la operación no pudo realizarse con éxito.
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda cuya base de datos está corrupta.	El sistema intenta exportar los elementos pero al estar la base de datos corrupta no puede realizar la conexión. Se informa al usuario que la operación no pudo realizarse con éxito
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

7.6.2.9 Caso de uso: importar XML

Caso de Uso 9 : importar XML	
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato correcto	El sistema importa parsea el XML y almacena cada uno de los elementos y sus relaciones en la base de datos. Finalmente notifica al usuario que la operación se ha realizado con éxito
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato incorrecto	El sistema intenta parsear el XML importado pero el formato de este al no ser el correcto se produce un error, con esto se notifica al usuario el problema y el sistema permanece sin cambios
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.

7.6.2.10 Caso de uso: Generar CSV

Caso de Uso 10: importar XML	
Prueba	Resultado Esperado
El usuario genera un CSV	El sistema recoge la lista del productos del usuario y genera un archivo de tipo CSV que será compatible con las plataformas Magento y Prestashop

7.6.2.11 Caso de uso: Editar datos de usuario

Caso de uso 11: editar datos de usuario	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de sus datos personales y guarda la opción	El sistema comprueba que todos los campos obligatorios del formulario están completos, y que la contraseña y repetición de contraseña coinciden, una vez verificado que el formulario está correctamente cumplimentado actualiza los datos del usuario. Cuando se terminan de guardar los cambios avisa al usuario que la operación se ha realizado con éxito.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación de sus datos personales e intenta guardar los cambios	El sistema valida el formulario y comprueba que existen errores, por lo que le indica al usuario cuales son los problemas y no actualiza los cambios.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios y se vuelve a la sección de datos personales.

7.6.2.12 Caso de uso: Crear categoría UML

Caso de uso 12: crear categoría UML	
Prueba	Resultado Esperado
El usuario arrastra un componente categoría de la paleta de elementos a la zona de dibujo	El sistema pinta una categoría con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción categoría	El sistema pinta una categoría con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.

7.6.2.13 Caso de uso: Crear subcategoría UML

Caso de uso 13: crear subcategoría UML	
Prueba	Resultado Esperado
El usuario enlaza dos categorías e indica cual es la categoría base	El sistema comprueba el tipo de entidades que se quieren enlazar y pinta un nexo de unión entre las dos categorías creadas por el usuario

7.6.2.14 Caso de uso: Crear producto UML

Caso de uso 14: crear producto UML	
Prueba	Resultado Esperado
El usuario arrastra un componente producto de la paleta de elementos a la zona de dibujo	El sistema pinta un producto con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción producto	El sistema pinta un producto con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.

7.6.2.15 Caso de uso: Crear proveedor UML

Caso de uso 15: crear proveedor UML	
Prueba	Resultado Esperado
El usuario arrastra un componente proveedor de la paleta de elementos a la zona de dibujo	El sistema pinta un proveedor con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción proveedor	El sistema pinta un proveedor con sus atributos y características. Cada uno de los atributos es editable y el usuario puede introducir datos.

7.6.2.16 Caso de uso: Relacionar un producto con una o varias categorías

Caso de uso 16: crear subcategoría UML	
Prueba	Resultado Esperado
El usuario enlaza dos o más categorías con un producto	El sistema comprueba que tipos de entidades se están enlazando y pinta un nexo de unión entre las categorías y el producto

7.6.2.17 Caso de uso: Relacionar un producto un proveedor

Caso de uso 17: Relacionar un producto con un proveedor	
Prueba	Resultado Esperado
El usuario enlaza un producto con un proveedor	El sistema pinta un nexo de unión entre el producto y el proveedor
Prueba	Resultado Esperado
El usuario intenta enlazar un producto con más de un proveedor	El sistema pinta un nexo de unión entre el producto y el primer proveedor con el que se intenta unir pero no permite unirlo con más de uno

7.6.2.18 Caso de uso: Generar un XML

Caso de uso 18: Generar un XML	
Prueba	Resultado Esperado
El usuario genera un XML después de haber creado elementos XML	El sistema va recogiendo y procesando todos los datos que contiene el diagrama UML, con toda esta información genera un XML en un directorio del proyecto previamente establecido.
Prueba	Resultado Esperado
El usuario intenta generar un XML sin haber creado previamente elementos UML	El sistema no obtiene información del diagrama UML por lo que crea un XML pero sin información de ningún elemento

7.6.3 Pruebas de Usabilidad

Se efectuarán pruebas de usabilidad por diversos usuarios con el fin de determinar si los contenidos y funcionalidades de la herramienta son realmente entendibles y manejables. Intentando mejorar con ello algunos de los aspectos y funcionalidades del prototipo.

Por ello se han elegido distintos tipos de usuario que se adaptan a los eventuales perfiles de las personas que pueden usar la aplicación:

- **Usuarios de tipo 1:** Usuario con amplios conocimientos en el manejo de plataformas de comercio electrónico
- **Usuarios de tipo 2:** Usuario inexperto en el uso y conocimiento de plataformas de comercio electrónico

Las pruebas se llevarán a cabo online y con un guión creado expresamente para este tipo de fin. Al finalizar las pruebas los usuarios rellenarán un cuestionario para determinar las impresiones obtenidas con el uso de la herramienta y determinar que partes de la aplicación se pueden mejorar.

¿El sitio web muestra de forma precisa y completa los contenidos o servicios que ofrece?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿La estructura general del sitio web está orientada al usuario?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Es coherente el diseño general del sitio web?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Emplea un lenguaje claro y conciso?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

La estructura de organización y navegación, ¿Es la más adecuada?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Los enlaces son fácilmente reconocibles (visitados, activos...)?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Es predecible la respuesta del sistema antes de hacer clic sobre el enlace?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Se ha evitado la redundancia de enlaces?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Los procedimientos son fácilmente accesibles?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Muestra los resultados de la búsqueda de forma comprensible para el usuario?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Se visualiza correctamente con diferentes resoluciones de pantalla?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Tiene el usuario el control sobre la interfaz?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

Cuando se produce un error, ¿se informa de forma clara al usuario de lo ocurrido y de cómo solucionar el problema

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Es usuario habitual de plataformas de comercio electrónico?

1. Todos los días
2. Varias veces a la semana
3. Ocasionalmente
4. Nunca, o casi nunca

¿Qué tipo de actividades realiza con plataformas de comercio electrónico?

1. Es parte de mi trabajo o profesión
2. Lo uso para transacciones comerciales
3. Nunca he hecho uso de una plataforma de comercio electrónico

¿Con que frecuencia realiza transacciones de comercio electrónico?

1. Habitualmente
2. Rara vez
3. Nunca

¿Ha usado alguna vez gestores de contenido de comercio Electrónico como prestashop y/o magento?

1. Sí
2. No

¿Conoce la existencia de algún producto software que posibilite la interoperabilidad y reutilización entre plataformas?

1. Sí
2. No

¿Ha gestionado alguna vez una tienda de comercio electrónico?

1. Sí
2. No

¿Qué grado de dificultad le atribuiría usted a la importación y exportación del catálogo de productos (categorías, productos y proveedores) de una plataforma de comercio electrónico a otra?

1. Fácil
2. Medio
3. Difícil
4. Muy difícil

Qué factor considera mas importante cuando realiza transacciones de comercio electrónico?

1. Fácil de usar
2. Intuitivo
3. Rápido
4. Funcional

¿Esta herramienta ofrece una asistencia útil para la gestión de sitios de comercio electrónico?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Esta aplicación ofrece una forma eficaz de desarrollar la tarea indicada?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿La forma de interoperar entre aplicaciones (plataformas) mediante el uso de la aplicación es comprensible?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo

4. Totalmente de acuerdo

¿El uso de la aplicación disminuye la complejidad de interoperar entre plataformas?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

¿Este editor puede considerarse útil y usable?

1. Totalmente en desacuerdo
2. En desacuerdo
3. De acuerdo
4. Totalmente de acuerdo

7.6.4 Pruebas de rendimiento

Las pruebas de rendimiento en este caso se limitaran a probar que las operaciones de la web no tienen tiempos de respuesta demasiado elevados como para que de la impresión al usuario que la aplicación a sufrido un error o cuelgue.

Capítulo 8. Implementación del Sistema

8.1 Lenguajes de Programación

8.1.1 Java

Java es un lenguaje de programación de propósito general basado en clases y orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. En la actualidad es uno de los lenguajes de programación más populares, en concreto en aplicaciones cliente-servidor para web, con unos 10 millones de usuarios registrados.

Se ha usado éste lenguaje para implementar la aplicación al completo, a excepción de la base de datos.

8.1.2 XML

XML se corresponde con las siglas en inglés para “lenguaje de marcas extensible”. Es por tanto un lenguaje de marcado desarrollado por el W3C (World Wide Web Consortium) que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas.

Es una tecnología sencilla que se ve complementada con otras que la enriquecen y la dotan de muchas posibilidades. Al tratarse de un estándar en el intercambio de información sus usos van desde bases de datos, editores de texto hasta servicios web u hojas de cálculo. En el presente proyecto se utilizará para intercomunicar el DSL y la aplicación web.

8.1.3 XHTML

XHTML (en inglés eXtensible HyperText Markup Language) se utiliza para generar documentos y contenidos de hipertexto generalmente publicados en la WEB. Es además una reformulación del lenguaje HTML que se puede jactar de ser ahora compatible con XML.

Un lenguaje de marcado nos permite dejar indicaciones (marcas) en un documento que sirven para diferenciar distintos tipos de contenidos, estructuras o secciones. Por ejemplo hay marcas (elementos, etiquetas) que pueden "marcar" qué es un título, y su lugar en una jerarquía de importancia, o advertir que una palabra debe estar enfatizada, o bien que una oración es además un enlace hacia otro recurso.

En este proyecto se usa XHTML para la generación de las vistas de la web.

8.1.4 CSS

El CSS (hojas de estilo en cascada) es un lenguaje que define la apariencia de un documento escrito en un lenguaje de marcado (por ejemplo, HTML). Así, a los elementos de la página web creados con HTML se les dará la apariencia que se desee utilizando CSS: colores, espacios entre elementos, tipos de letra... separando de esta forma la estructura de la presentación.

En este caso se utiliza CSS para maquetar la Web.

8.1.5 Javascript

Javascript es un lenguaje de programación utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web. Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Se usa Javascript para lograr algunos efectos en la web.

8.1.6 SQL

La sigla que se conoce como SQL corresponde a la expresión inglesa Structured Query Language (entendida en español como Lenguaje de Consulta Estructurado), la cual identifica a un tipo de lenguaje vinculado con la gestión de bases de datos de carácter relacional que permite la especificación de distintas clases de operaciones entre éstas. Gracias a la utilización del álgebra y de cálculos relacionales, el SQL brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla.

En este caso se usa SQL para la gestión de consultas de la base de datos de la aplicación así como para obtener e insertar los datos en las dos plataformas de comercio electrónico (Magento y Prestashop)

8.2 Tecnologías utilizadas

8.2.1 DSL

Para la construcción del DSL se han empleado básicamente dos tecnologías Eclipse Modeling Framework (EMF) y Graphical Modeling Framework (GMF), EMF se empleó para la definición del metamodelo y GMF es una solución que mejora la relación EMF/GEF (Eclipse Graphical Editing Framework) con la utilización de metamodelos que permiten la definición de la sintaxis abstracta y concreta, también define facilidades para el mapping entre las dos y una forma sencilla de realizar una interfaz personalizada para él. La construcción del DSL se realizó bajo la plataforma eclipse y este debe tener instalados los plugins de EMF, GEF y GMF. Los pasos que hay que seguir se ven reflejados en el siguiente dashboard.

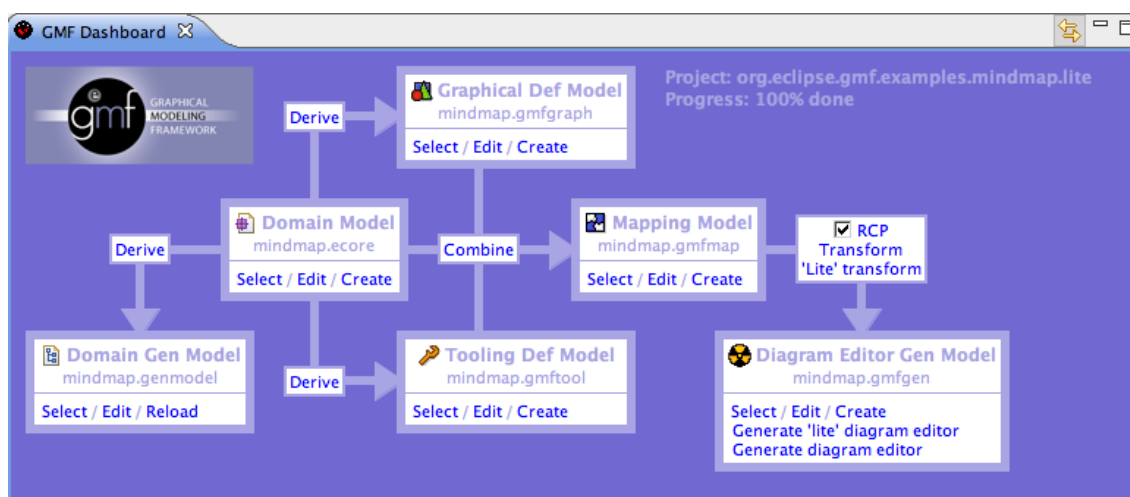


Ilustración 163 GMF DashBoard

8.2.2 Plataforma Web

Para el desarrollo de la aplicación web se han elegido tecnologías ampliamente utilizadas en el desarrollo de aplicaciones Java. El desarrollo se basa principalmente en Spring Framework cuya estructura principal se muestra en la siguiente figura. A continuación se describe cada uno de los módulos:

- **Core:** provee la funcionalidad base del framework Spring. El principal componente de este módulo es el BeanFactory, una implementación del patrón Factory. Su función es separar la configuración y dependencias del resto de la aplicación.
- **Spring AOP:** módulo que integra la programación orientada a aspectos dentro de Spring. De esta manera el programador puede habilitar AOP a cualquier objeto manejado desde Spring.
- **Spring DAO:** esta capa de abstracción permite manejar las excepciones y mensajes de error desde diferentes proveedores de bases de datos. Su función es abstraer y encapsular el acceso a los datos y además maneja la conexión con la fuente de datos para obtener y guardar los datos.

- **Spring ORM:** este módulo ofrece herramientas de mapeo O/R, en este caso se ha decidido utilizar la más común, que es Hibernate.
- **Spring MVC:** El modelo vista-controlador (MVC) permite construir la aplicación web de manera sencilla. Es configurable y permite añadir varias tecnologías de vista como JSP , Tiles...

Sobre la arquitectura base de Spring framework se añade otro bloque que amplía en gran medida la funcionalidad base del framework. Permitiendo crear módulos autónomos y reutilizables llamados flujos, dicha autonomía está dada por la capacidad de guiar al usuario a través de una secuencia organizada de páginas y manejar el seguimiento de esas actividades por sí solo.

Sobre este módulo se sitúa el módulo de control de seguridad que sigue los principios de Spring Security y se centra en proveer mecanismos de autenticación y autorización para aplicaciones basadas en Spring. Esto se hace gracias a una serie de filtros que proporciona Spring Security para realizar las diferentes tareas de seguridad que se necesiten en la aplicación.

Finalmente para facilitar el desarrollo de interfaces de usuario la arquitectura incluye Spring Facesm , básicamente consiste en la integración de Java Server Faces (JSF) dentro del framework Spring .

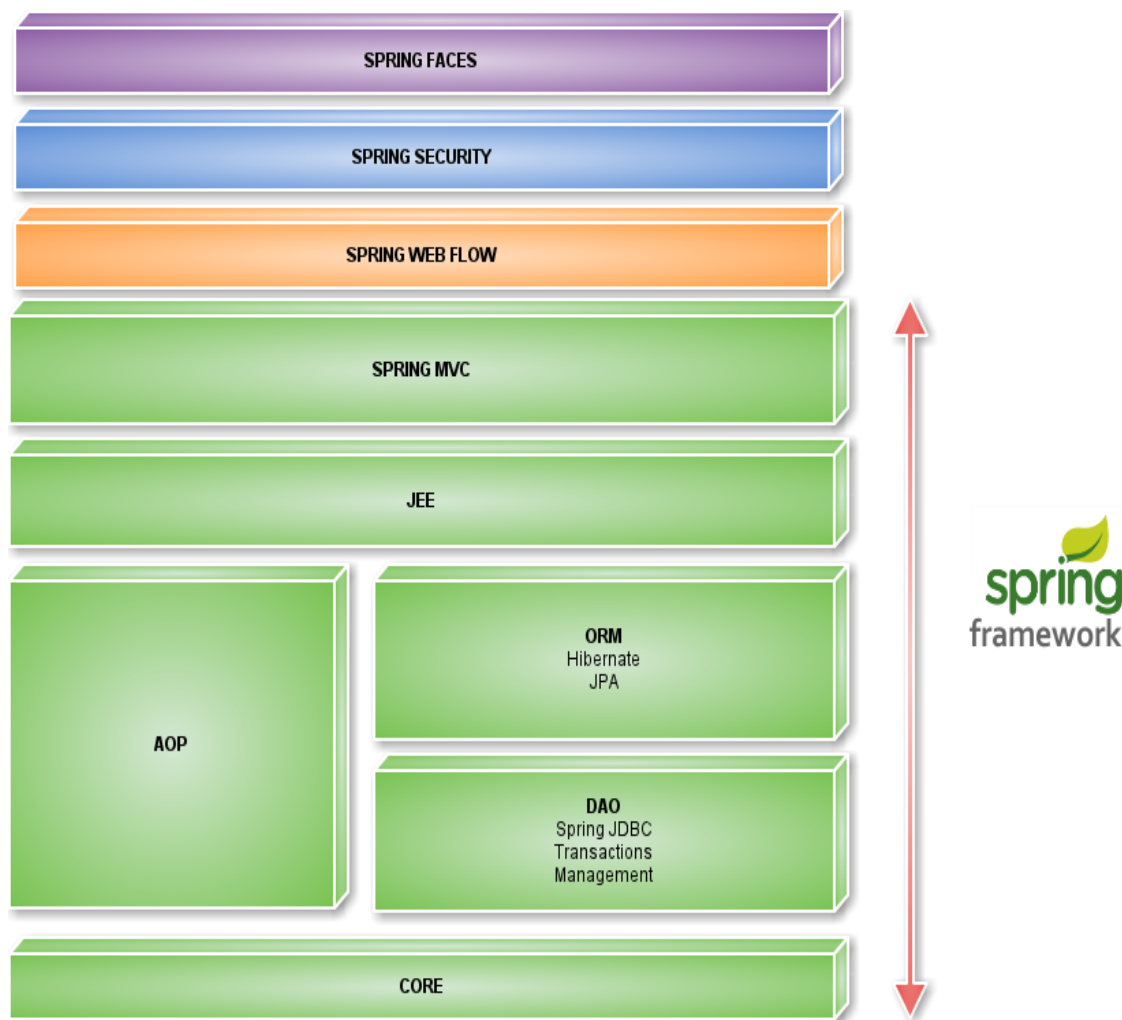


Ilustración 164 Esquema de capas Spring

8.2.2.1 Tecnologías utilizadas para las vistas

Para el desarrollo del front-end en la aplicación web se ha recurrido a la utilización de dos tecnologías Java. Por un lado la detección de eventos de la interfaz del usuario es gestionada en la capa de control mediante Spring Web Flow (SWF). Por otro lado Java Server Faces (JSF) es el framework sobre el que se desarrollarán las interfaces de usuario.

SWF nos permite controlar las llamadas a las diferentes acciones que tengamos implementadas en la parte de back-end. Para ello es necesario definir el diagrama de cada uno de los flujos de acción y especificar cada una de las vistas que se muestren al usuario.

Para cada una de las historias de usuario abordadas se han definido flujos de control básico.

Cada uno de estos diagramas define el comportamiento del flujo para realizar las acciones CRUD de cada una de las entidades. Estos flujos contienen los siguientes estados, acciones y vistas:

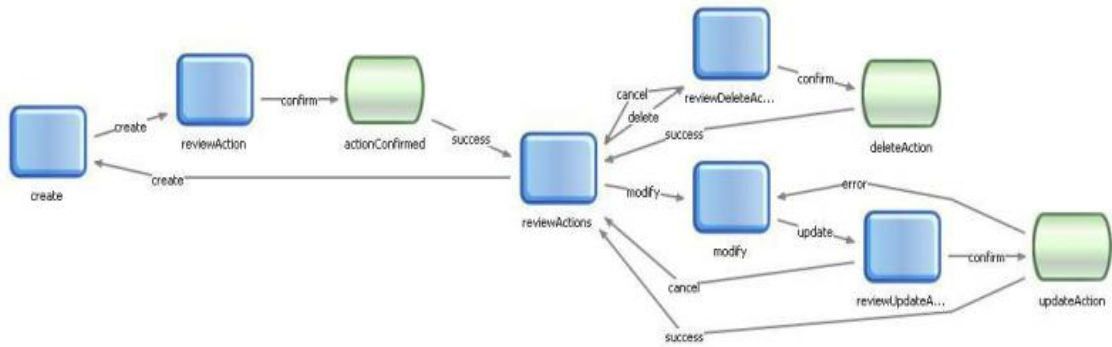


Ilustración 165 Descripción de los flujos

La codificación de estos flujos se realiza en los archivos xml soportados por SWF.

```
<?xml version="1.0" encoding="UTF-8"?>
<flow xmlns="http://www.springframework.org/schema/webflow"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/webflow
http://www.springframework.org/schema/webflow/spring-webflow-2.0.xsd"
      start-state="reviewActions">

  <view-state id="create" model="actionForm">
    <on-render>
      <evaluate expression="actionAction.setupActionForm" />
    </on-render>
    <transition on="create" to="reviewAction" />
  </view-state>

  <view-state id="reviewAction">
    <transition on="confirm" to="actionConfirmed" />
  </view-state>

  <action-state id="actionConfirmed">
    <evaluate expression="actionAction.saveAction" />
    <transition on="success" to="reviewActions" />
    <transition on="error" to="register" />
  </action-state>

  <view-state id="reviewActions">
    <on-render>
      <evaluate expression="actionAction.listAllActions" />
    </on-render>
    <transition on="delete" to="reviewDeleteAction">
      <set name="flowScope.actionSelected"
value="actionList.selectedRow" />
    </transition>
    <transition on="modify" to="modify">
      <set name="flowScope.actionSelected"
value="actionList.selectedRow" />
    </transition>

    <transition on="create" to="create" />
  </view-state>

  <view-state id="reviewDeleteAction">
    <transition on="confirm" to="deleteAction" />
    <transition on="cancel" to="reviewActions" />
  </view-state>

  <action-state id="deleteAction">
    <evaluate expression="actionAction.deleteAction" />
    <transition on="success" to="reviewActions" />
    <transition on="error" to="register" />
  </action-state>

  <view-state id="modify" model="actionForm">
    <on-render>
      <evaluate expression="actionAction.modifyActionForm" />
    </on-render>
    <transition on="update" to="reviewUpdateAction" />
  </view-state>
</flow>
```

```

</view-state>

<view-state id="reviewUpdateAction">
  <transition on="confirm" to="updateAction" />
  <transition on="cancel" to="reviewActions" />
</view-state>

<action-state id="updateAction">
  <evaluate expression="actionAction.updateAction" />
  <transition on="success" to="reviewActions" />
  <transition on="error" to="modify" />
</action-state>

</flow>

```

Se define un archivo de definición de flujo para cada entidad de la misma forma que se ha especificado en el archivo anterior adaptándolo a las diferentes páginas de vista y sus acciones.

La estructura de definición de flujos de la aplicación queda de la siguiente forma:

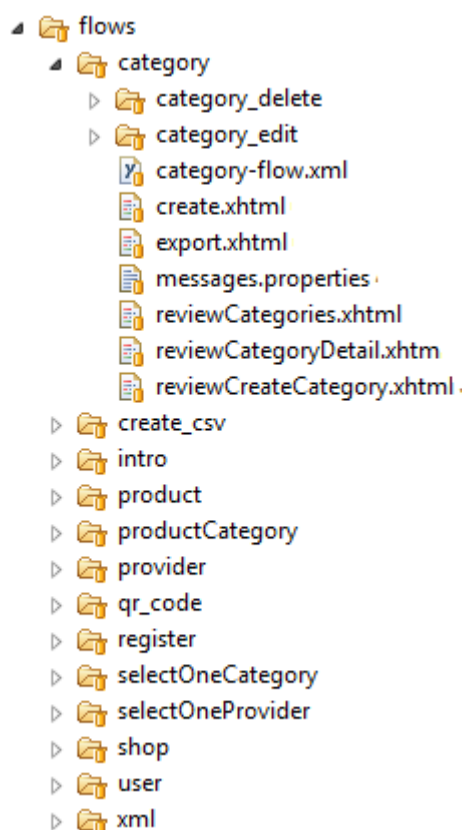


Ilustración 166 Estructura de flujos

En cuanto al framework JSF, este simplifica las interfaces de usuario de JEE. Para ello contiene un conjunto de APIs que permiten representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad. Además consta de un conjunto por defecto de componentes para la interfaz de usuario y etiquetas personalizadas que permiten expresar una interfaz JSF dentro de una página JSP. Los objetivos[3] que se quieren conseguir al utilizar JSF son:

Definir un conjunto simple de clases base de Java para componentes de la interfaz de usuario, estado de los componentes y eventos de entrada. Estas clases tratarán los aspectos del ciclo de vida de la interfaz de usuario, controlando el estado de un componente durante el ciclo de vida de su página.

Proporcionar un conjunto de componentes para la interfaz de usuario, incluyendo los elementos estándares de HTML para representar un formulario. Estos componentes se obtendrán de un conjunto básico de clases base que se pueden utilizar para definir componentes nuevos.

Proporcionar un modelo de JavaBeans para enviar eventos desde los controles de la interfaz de usuario del lado del cliente a la aplicación del servidor.

Definir APIs para la validación de entrada, incluyendo soporte para la validación en el lado del cliente.

Especificar un modelo para la internacionalización y localización de la interfaz de usuario. Automatizar la generación de salidas apropiadas para el objetivo del cliente, teniendo en cuenta todos los datos de configuración disponibles del cliente (por ejemplo versión del navegador)

8.2.2.2 Tecnologías usadas para el maquetado de las interfaces

A parte del CSS básico, para el maquetado de las interfaces web se hace uso del framework Foundation. Este es un framework de desarrollo que permite la creación de sitios web preparados para visualizarse en cualquier tamaño de pantalla de cualquier dispositivo ya que dependiendo del tamaño de la pantalla utilizado redimensiona los elementos de la interfaz para que puedan verse adecuadamente.

Este framework de CSS y Javascript cuenta con cuadrículas en la que ajustar los diseños web, botones, formularios, plantillas, etc, para que las interfaces sean compatibles con los tamaños de pantallas de cualquier dispositivo.



Ilustración 167 Foundation

8.2.2.3 Tecnologías para el control de acceso

Hasta hace poco era muy común desarrollar ciertos elementos de seguridad manualmente corriendo el riesgo de cometer algún error o dejar algún agujero de seguridad. Hoy en día es preferible utilizar una de las múltiples herramientas o frameworks que existen y que nos permiten añadir seguridad a diferentes puntos de una aplicación teniendo el respaldo de toda una comunidad de desarrolladores y usuarios que evolucionan y testean estos frameworks a diario.

Para la realización de la plataforma web se utiliza Spring Security. Es parte del framework de Spring proveniente de Acegi y que ofrece una gran potencia para securizar aplicaciones basadas en Spring ofreciendo mecanismos de autenticación y control de acceso. Es sencillo de aprender, configurar y desplegar. Su mejor característica es su baja intrusividad en el código, gracias a la utilización de AOP (Aspect Oriented Programming), por lo que permite su implantación repercutiendo mínimamente en el código fuente de la aplicación.

Esta librería es una librería madura que está ampliamente extendida para el desarrollo de este tipo de aplicaciones. Por tanto, está ampliamente probada y tiene buen soporte tanto informativo como correctivo. Por esta razón se cree que Spring Security es capaz de aportar la robustez necesaria al control de acceso de nuestra aplicación.

Ofrece una lista de proveedores para realizar la autenticación, permitiendo implementar de una manera sencilla la autenticación de un usuario, utilizando estos proveedores como base de la implementación. Algunos de los proveedores que vienen por defecto en Spring aparecen en el gráfico posterior. En caso de que no sirva ningún proveedor de los que te ofrece Spring existe la posibilidad de generarse uno propio implementado su interface o utilizando sus clases abstractas.

Spring security utiliza una serie de filtros servlet que recogen las peticiones para comprobar la autenticación y aplicar la seguridad necesaria, siendo capaz de bajar a aplicar seguridad a nivel

de métodos aplicando AOP. En el diagrama siguiente se muestran los elementos principales de Spring Security:

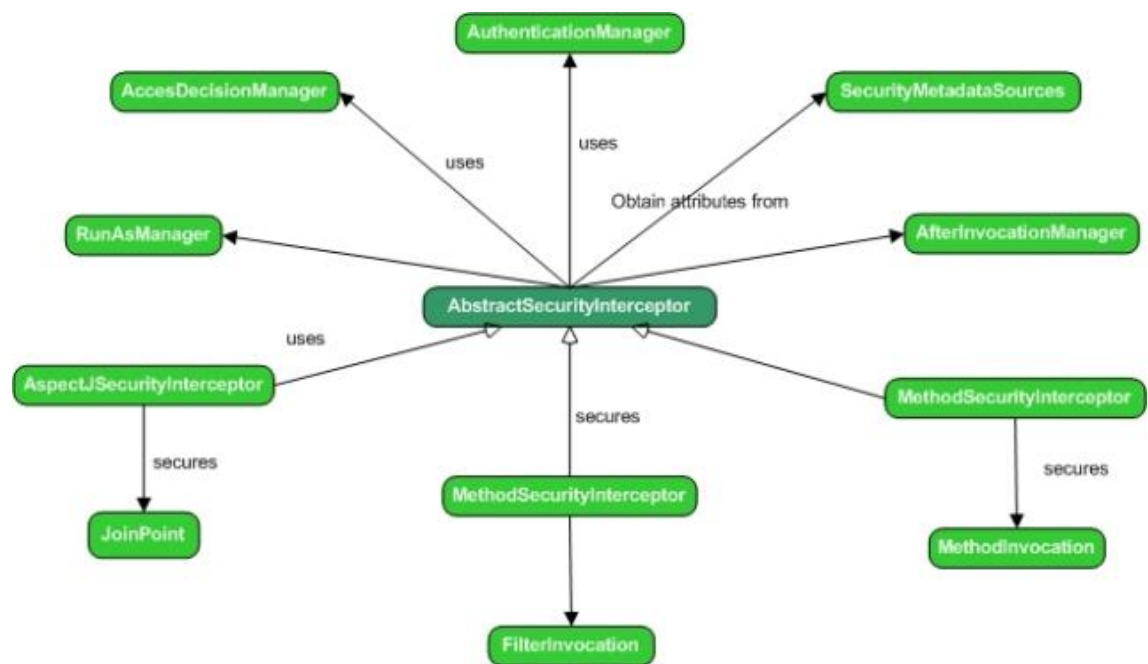


Ilustración 168 Elementos de Spring Security

8.2.2.4 Persistencia

8.2.2.4.1 Hibernate

Como framework para la persistencia se utiliza Hibernate, con este framework se consigue tratar la base de datos como si fuera un almacén orientado a objetos ya que éste actúa como mediador conectando ambos entornos.

Dispone de un lenguaje de consultas derivado del SQL muy potente y amigable llamado HQL. También permite, si es necesario, realizar las consultas con SQL nativo o con otras API's, para realizar búsquedas avanzadas como por ejemplo Criteria.

Su cometido se basa en mapear objetos a tablas en bases de datos relacionales y poderlos recuperar después. El funcionamiento de la tecnología se sustenta en los ficheros .hbm que definen cómo debe ser mapeada la clase en la base de datos. La integración con Spring es perfecta, ya que Spring se desarrolló para poder integrar nuevas tecnologías fácilmente.

A continuación se muestra una imagen del papel que desempeñaría el framework dentro de la aplicación.

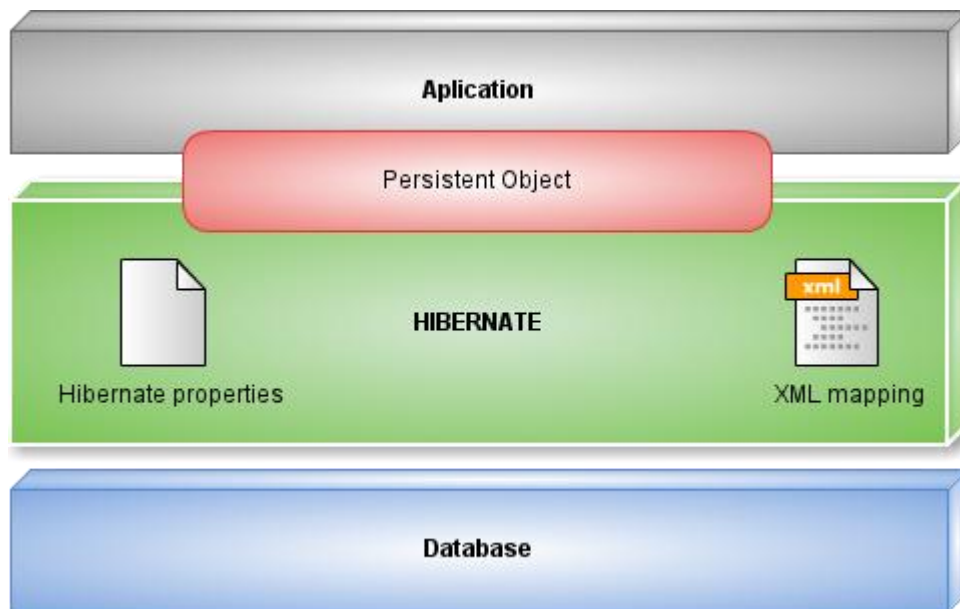


Ilustración 169 Hibernate

8.3 Herramientas y Programas Usados para el Desarrollo

8.3.1 Spring tool suite

SpringSource Tool Suite es un IDE Java basado en Eclipse, personalizado y extendido con software especializado para el desarrollo de aplicaciones Spring, cuyo desarrollo corre a cargo de SpringSource y que es distribuido bajo la licencia SpringSource Tool Suite End User License Agreement.

STS como también se le conoce a SpringSource Tool Suite se instala los servidores SpringSource Cloud Foundry, SpringSource dm Server y SpringSource tc Server, además brinda soporte para agregar fácilmente otros application servers como JBoss, JOnAS, IBM WebSphere y Tomcat.

8.3.2 Eclipse

Eclipse es uno de los más famosos entornos de desarrollo software. Ofrece soporte en múltiples idiomas y está escrito de forma mayoritaria en Java. Se usa para desarrollar aplicaciones en Java aunque con la instalación de diversos *plug-ins* también se puede programar en una amplia variedad de lenguajes: C, C++, PHP, Python, Ruby, etc.

Para la creación del GMF se ha descargado directamente un eclipse preparado para el desarrollo de este tipo de aplicaciones “Eclipse Modeling Tools”

Además de los plugins para el crear el GMF también se utiliza Maven para la gestión de dependencias y el plugin de SVN para gestionar el repositorio.

8.3.3 Tomcat

Para desplegar la aplicación web se hace uso de un servidor Apache Tomcat. Este funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages de Sun Microsystems

8.3.4 MySQL

El servidor de base de datos elegido para el subsistema software es MySQL .

- Sistema de gestión de bases de datos relacional
- Multiusuario
- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.

- Sistema de licencia dual Los usuarios pueden elegir entre usar el software MySQL como un producto Open Source bajo los términos de la licencia GNU General Public License o pueden adquirir una licencia comercial estándar de MySQL AB.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc).
- Gran portabilidad entre sistemas.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.

8.3.5 Protegé

Protegé (<http://protege.stanford.edu/overview/>) es una plataforma libre y de código abierto que provee una comunidad de usuarios cada vez mayor con un conjunto de herramientas para la construcción de modelos de dominio y las aplicaciones basadas en el conocimiento de las ontologías. En su esencia, Protegé implementa un amplio conjunto de estructuras y acciones que apoyen la creación, visualización y manipulación de ontologías en diversos formatos de representación de conocimiento de modelado. Protegé se puede personalizar para ofrecer soporte de dominio de usar para la creación de modelos de conocimiento y la introducción de datos. Además, Protegé se puede ampliar por medio de una arquitectura plug-in y una interfaz de programación de aplicaciones basado en Java (API) para la construcción de herramientas y aplicaciones basadas en el conocimiento.

En los últimos años, las ontologías se han adoptado en muchos negocios y la comunidad científica como una forma de acción, la reutilización y el dominio de conocimiento del proceso. Las ontologías son ahora fundamentales para muchas aplicaciones, tales como portales de conocimiento científico, la gestión de la información e integración de sistemas, comercio electrónico y servicios web semánticos. La plataforma Protegé soporta dos formas principales de ontologías de modelos:

El Protegé-Frames editor permite a los usuarios crear y poblar ontologías que están basado en cuadros , de acuerdo con la Open Knowledge Base protocolo de conectividad (OKBC) . En este modelo, una ontología consiste en un conjunto de clases organizadas en una jerarquía de subsunción para representar conceptos más destacados de un dominio, un conjunto de espacios asociados a las clases para describir sus propiedades y relaciones, y un conjunto de instancias de dichas clases - ejemplares individuales de los conceptos que tienen valores específicos para sus propiedades.

El Protegé-OWL editor permite a los usuarios construir ontologías para la Web Semántica , en particular en el de la W3C Web Ontology Language (OWL) . "Una ontología OWL puede incluir descripciones de clases, propiedades y sus instancias. Dada esa ontología, la semántica formal de OWL especifica cómo derivar sus consecuencias lógicas, es decir, los hechos no literalmente presentes en la ontología, sino que entraña la semántica. Estas vinculaciones pueden basarse en un solo documento o varios documentos distribuidos que se han combinado con los mecanismos de OWL definida

8.4 Creación del Sistema

En esta sección se incluyen todos aquellos aspectos con los que nos hemos encontrado a lo largo de la implementación.

8.4.1 Problemas Encontrados

8.4.1.1 Poco nivel de conocimiento en el desarrollo de herramientas de modelado específicas

Aunque la alumna contaba algunos conocimientos adquiridos en el máster sobre el tema, ya que se impartió una asignatura dedicada a ello. El nivel de complicación en la creación de estas herramientas es bastante alto y el nivel de experiencia adquirido en la asignatura era relativamente pequeño. Por lo que se tuvieron bastantes problemas a la hora de crear la herramienta GMF.

8.4.1.2 Desconocimiento acerca del manejo y las funcionalidades precisas de plataformas de comercio electrónico

La alumna nunca había tenido contacto con este tipo de plataformas, ni con los procedimientos de comercio electrónico por lo que fue necesario investigar a fondo cada una de ellas, instalarlas y examinar cada una de las funcionalidades de las que disponen para poder realizar la intercomunicación entre el prototipo y estas.

8.4.1.3 Intercomunicación entre plataformas

Las bases de datos de las plataformas con las que se interconecta la aplicación tienen un nivel de complejidad bastante elevado, por lo que fue necesario estudiar cada una de ellas a fondo para poder realizar la conexión entre la aplicación web y cada una de ellas. En el Anexo II se pueden ver los modelos de entidad relación de las plataformas Magento y Prestashop con los que está interconectada la aplicación.

8.4.2 Descripción Detallada de las Clases

Muchas de las clases del DSL son autogeneradas por la propia herramienta de desarrollo, por lo que únicamente se considera oportuno describir con detalle cada una de las clases de la aplicación web. Esta descripción completa se hace en el Anexo III Descripción de las Clases.

Capítulo 9. Desarrollo de las Pruebas

9.1 Pruebas Unitarias

Cada prueba unitaria se realizó al finalizar la implementación de una nueva función o componente. Para ello se comprueba que los resultados individuales de cada componente o función son los esperados. En caso de que se produzcan fallos (el resultado esperado no coincidía con el resultado obtenido), se hace una revisión y corrección de los errores detectados, hasta todas las pruebas tengan un resultado satisfactorio.

9.2 Pruebas de Usabilidad

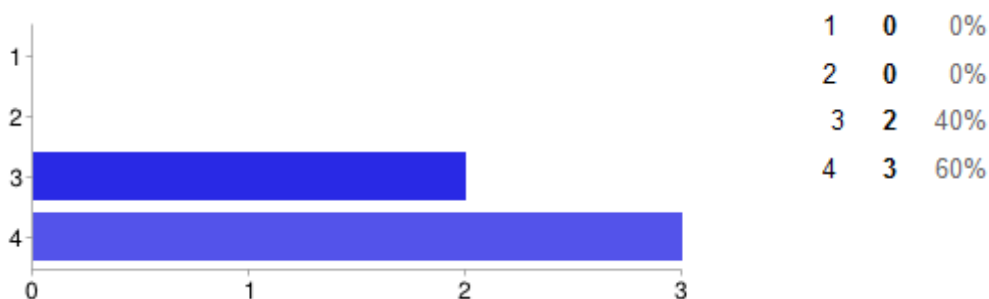
A continuación se adjuntan los resultados de las pruebas de usabilidad realizadas a usuarios varios usuarios. Se recuerda que los tipos de usuarios definidos son los siguientes:

- **Usuarios de tipo 1:** Usuario con amplios conocimientos en el manejo de plataformas de comercio electrónico
- **Usuarios de tipo 2:** Usuario inexperto en el uso y conocimiento de plataformas de comercio electrónico

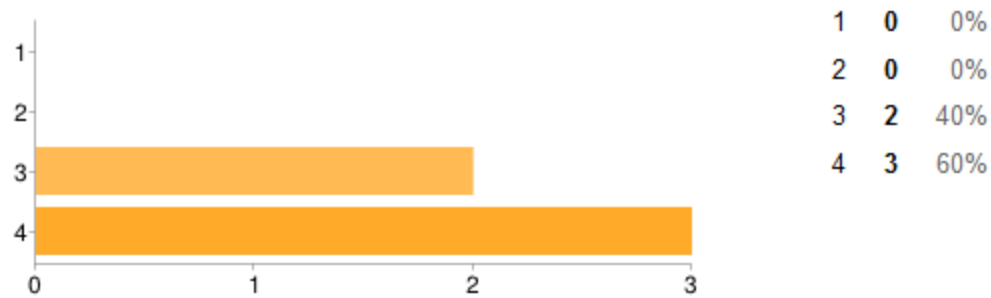
Las pruebas se han efectuado sobre un total de cinco personas, siendo dos de ellas usuarios de tipo 1 y el resto de tipo 2.

En primer lugar los usuarios siguieron un guión con diferentes actividades a realizar dentro de la aplicación, al concluir este cubrieron un cuestionario de preguntas tanto de carácter general como específicas de la aplicación. Los resultados obtenidos a cada una de las preguntas del formulario fueron las siguientes.

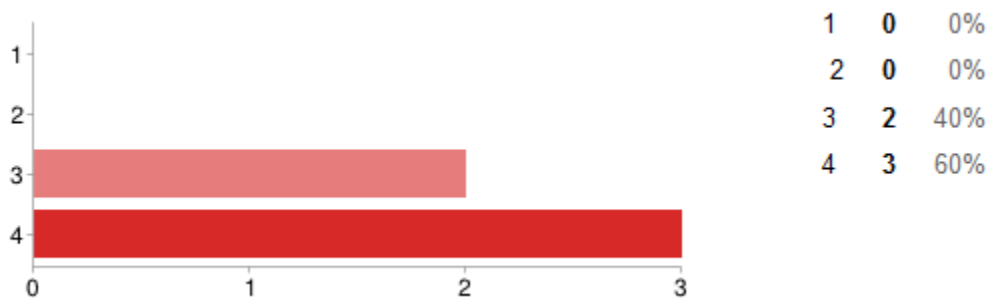
¿El sitio web muestra de forma precisa y completa los contenidos o servicios que ofrece?



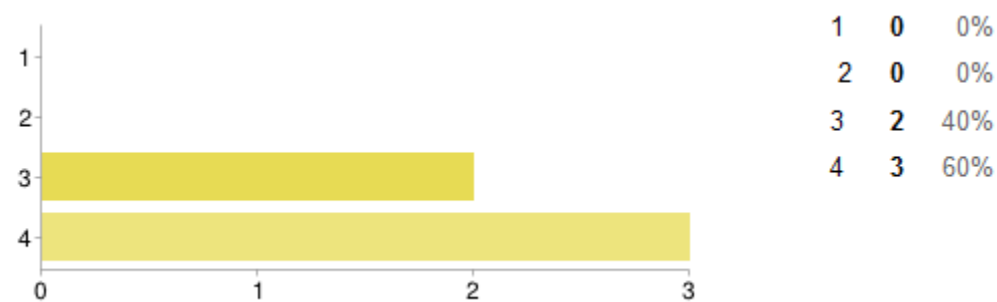
¿La estructura general del sitio web está orientada al usuario?



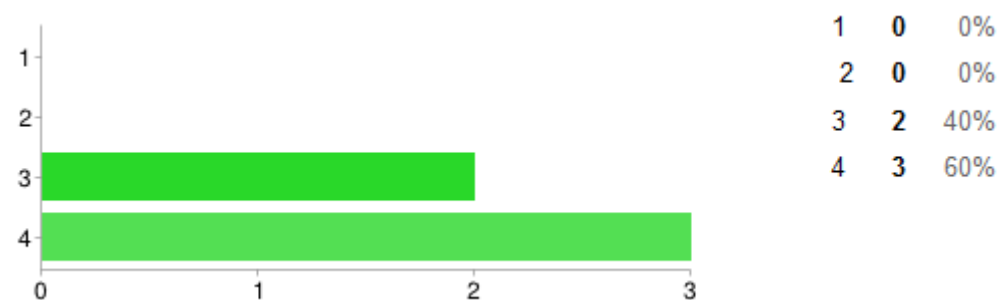
¿Es coherente el diseño general del sitio web?



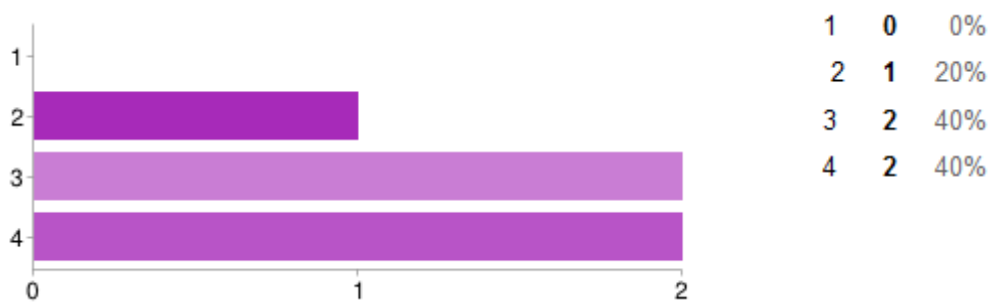
¿Emplea un lenguaje claro y conciso?



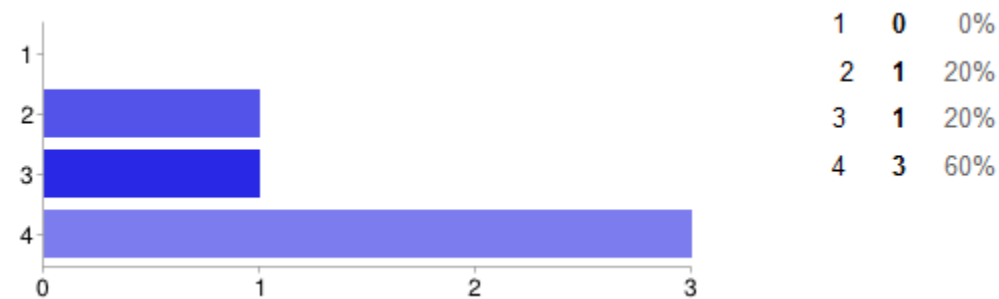
La estructura de organización y navegación, ¿Es la más adecuada?



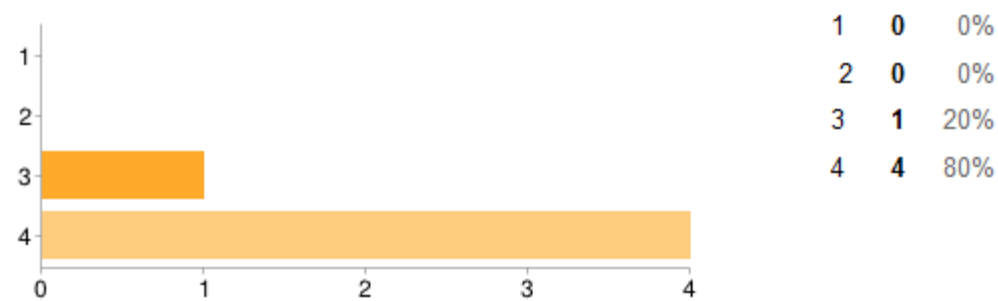
¿Los enlaces son fácilmente reconocibles (visitados, activos...)?



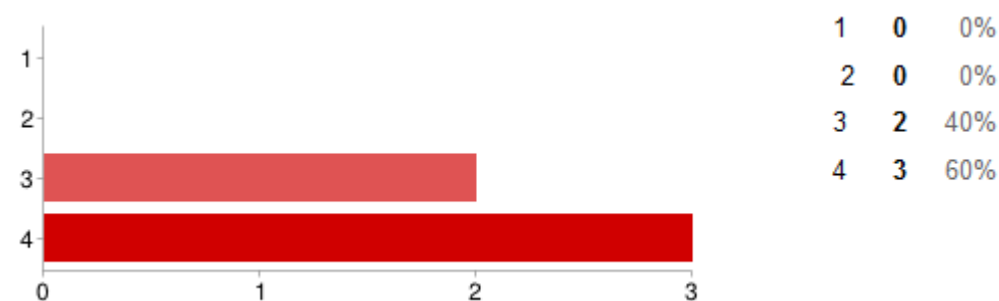
¿Es predecible la respuesta del sistema antes de hacer clic sobre el enlace?



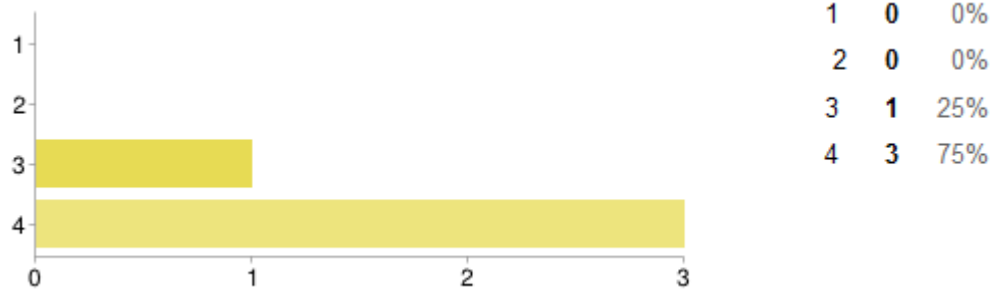
¿Se ha evitado la redundancia de enlaces?



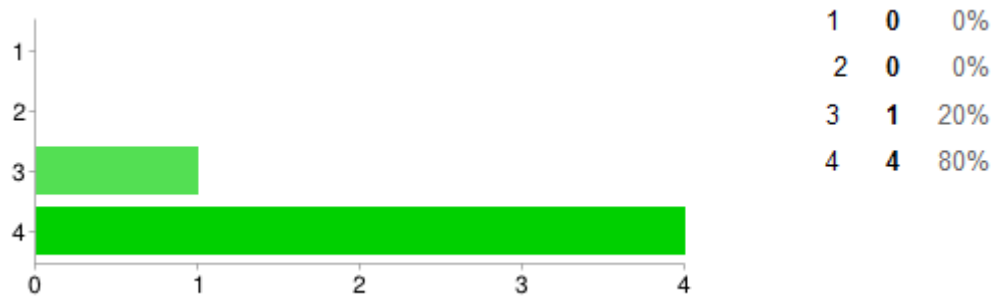
¿Los procedimientos son fácilmente accesibles?



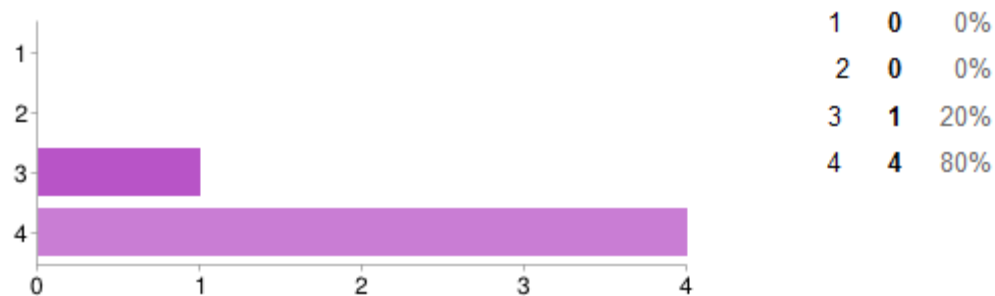
¿Muestra los resultados de la búsqueda de forma comprensible para el usuario?



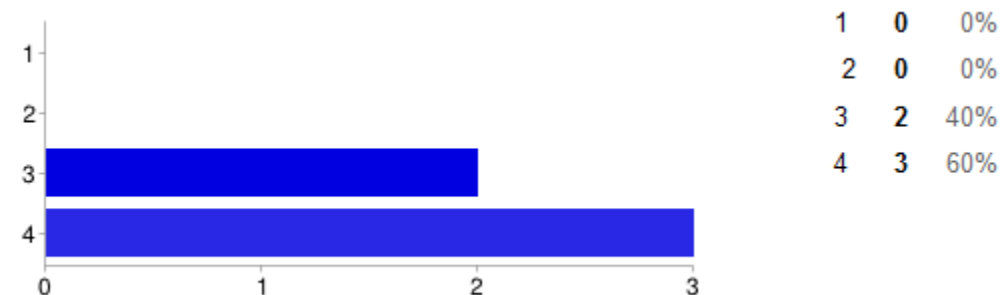
¿Se visualiza correctamente con diferentes resoluciones de pantalla?



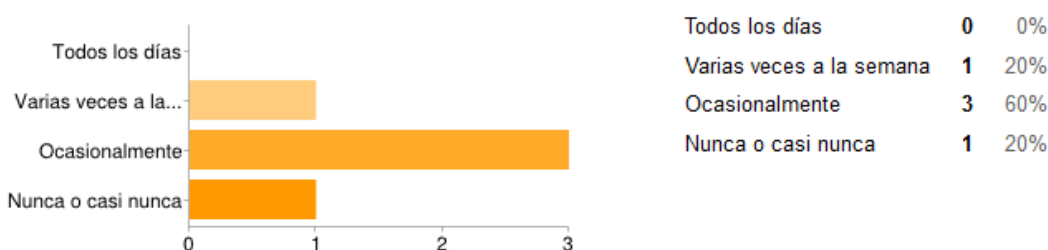
¿Tiene el usuario el control sobre la interfaz?



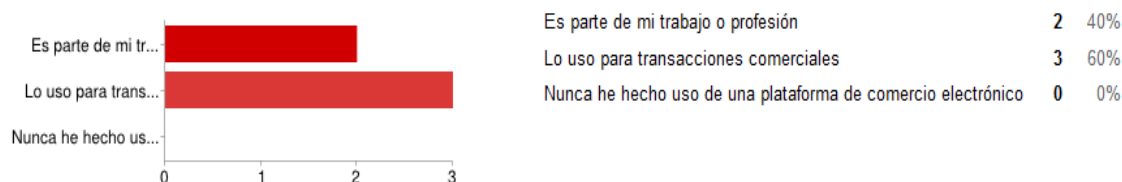
Cuando se produce un error, ¿se informa de forma clara al usuario de lo ocurrido y de cómo solucionar el problema



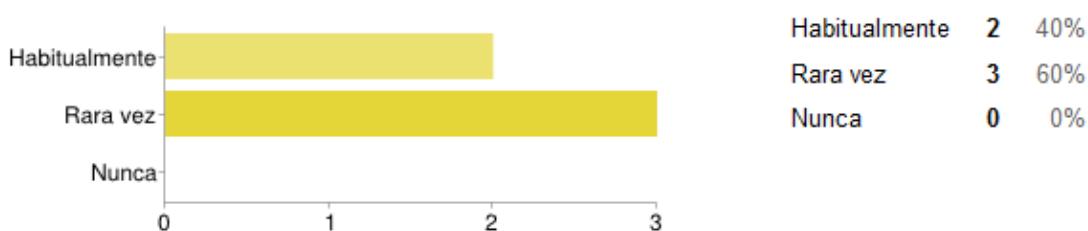
¿Es usuario habitual de plataformas de comercio electrónico?



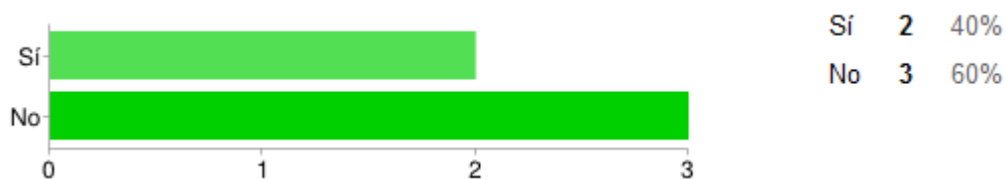
¿Qué tipo de actividades realiza con plataformas de comercio electrónico?



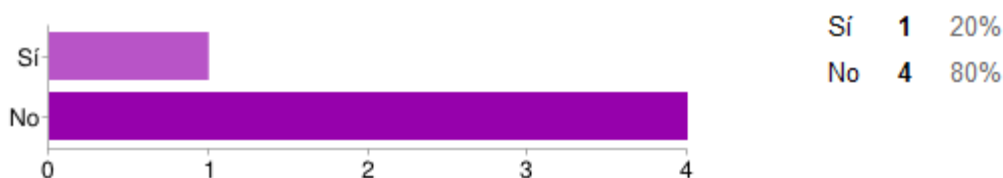
¿Con qué frecuencia realiza transacciones de comercio electrónico?



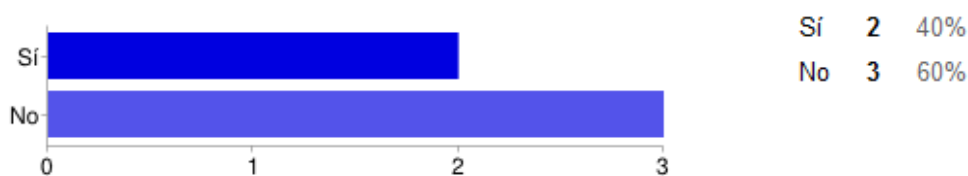
¿Ha usado alguna vez gestores de contenido de comercio Electrónico como prestashop y/o magento?



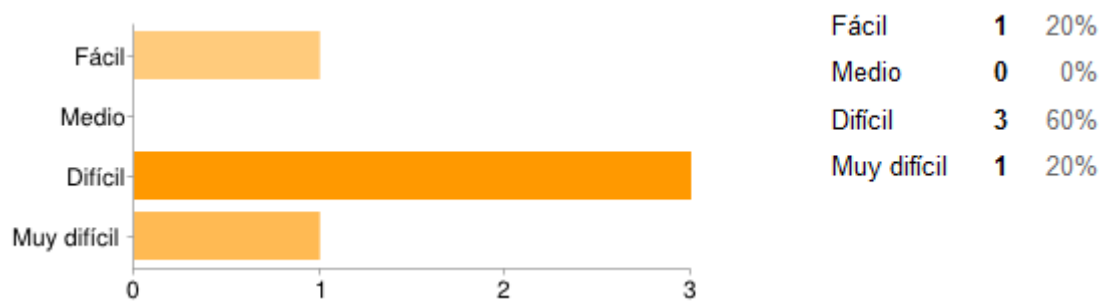
¿Conoce la existencia de algún producto software que posibilite la interoperabilidad y reutilización entre plataformas?



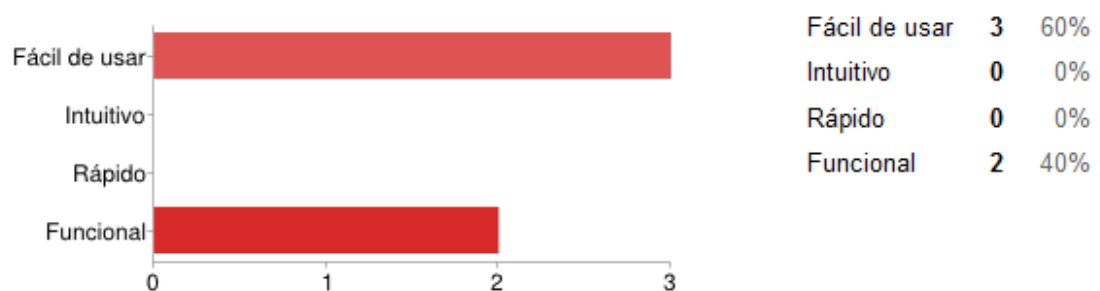
¿Ha gestionado alguna vez una tienda de comercio electrónico?



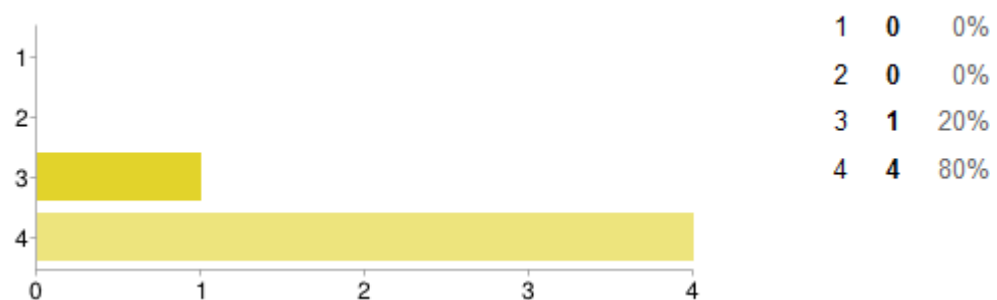
¿Qué grado de dificultad le atribuiría usted a la importación y exportación del catálogo de productos (categorías, productos y proveedores) de una plataforma de comercio electrónico a otra?



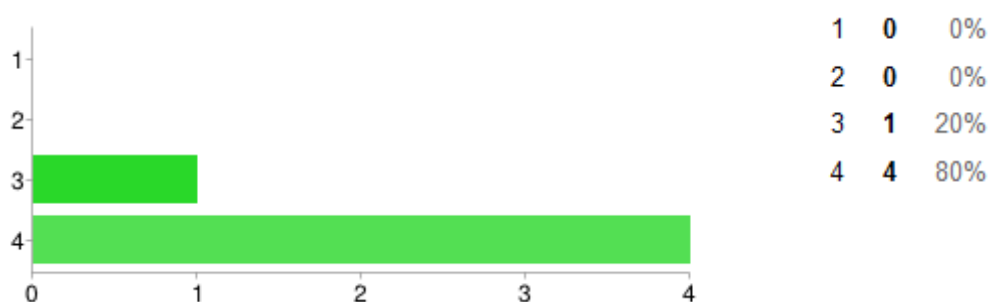
¿Qué factor considera mas importante cuando realiza transacciones de comercio electrónico?



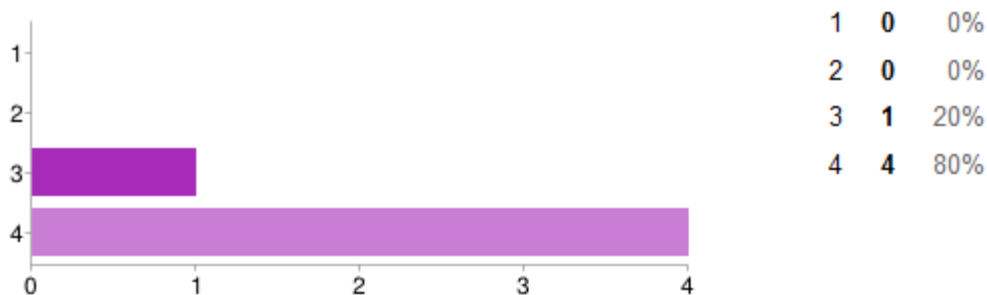
¿Esta herramienta ofrece una asistencia útil para la gestión de sitios de comercio electrónico?



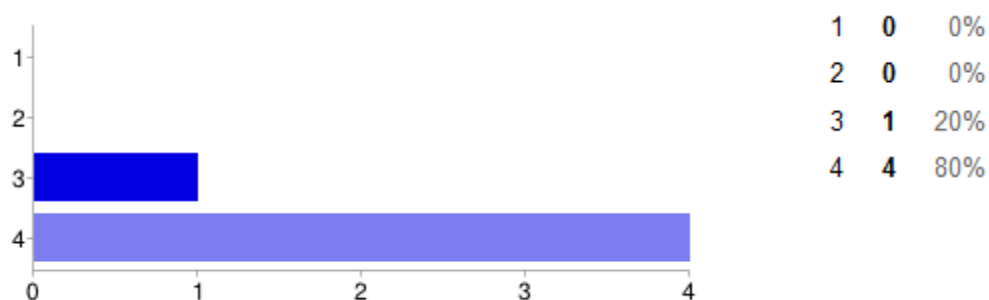
¿Esta aplicación ofrece una forma eficaz de desarrollar la tarea indicada?



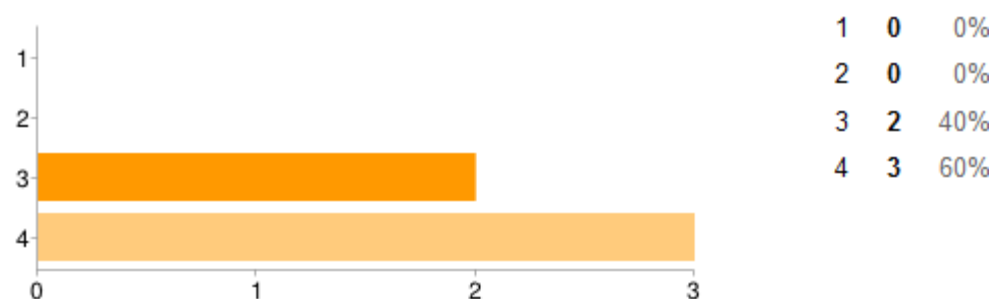
¿La forma de interoperar entre aplicaciones (plataformas) mediante el uso de la aplicación es comprensible?



¿El uso de la aplicación disminuye la complejidad de interoperar entre plataformas?



¿Este editor puede considerarse útil y usable?



9.2.1 Análisis de los resultados, cambios efectuados y conclusiones

Análisis de resultados pruebas:

- Al consultar a los usuarios acerca de si la aplicación web muestra de forma precisa y completa los contenidos o servicios que ofrece, los mismos afirman, que este propósito se cumple, un 60% de los mismos se manifiesta muy de acuerdo y un 40% de acuerdo.
- El diseño general del sitio web es valorado como coherente por los usuarios de manera favorable un 60% de los mismos se manifiesta muy acuerdo y un 40% de acuerdo, al ser consultados a este respecto.

- Frente a las preguntas sobre lo claro y conciso del lenguaje, y como valoran la estructura de organización y navegación, un 60% de los usuarios se manifiesta muy acuerdo y un 40% de acuerdo, lo que implica un alto porcentaje de aceptación al respecto.
- Se consultó sobre si los enlaces son fácilmente reconocibles, un 20% de los encuestados está parcialmente de acuerdo, un 40% de acuerdo y un 40% muy de acuerdo.
- Frente al cuestionamiento de si resulta predecible la respuesta del sistema antes de hacer clic sobre el enlace un 20% de los encuestados está parcialmente de acuerdo, un 40% de acuerdo y un 40% muy de acuerdo.
- Un 80% de los encuestados valoran muy positivamente la ausencia de redundancia de enlaces, y un 20% se muestran de acuerdo
- Se preguntó si la aplicación Muestra los resultados de la búsqueda de forma comprensible para el usuario, un 75% estuvieron muy de acuerdo y un 25% de acuerdo con esta afirmación.
- Efectuada la prueba se consulto a los usuarios frente a la visualización correcta con diferentes resoluciones de pantalla, y si tiene el usuario el control sobre la interfaz 20% manifiestan estar de acuerdo y un 80% muy de acuerdo, al respecto.
- Ante la pregunta: ¿se informa de forma clara al usuario de lo ocurrido y de cómo solucionar el problema? un 60% de los usuarios se manifiesta muy acuerdo y un 40% de acuerdo.

Caracterización de los usuarios:

- Del total de participantes son usuarios habituales de plataformas de comercio electrónico varias veces por semana un 20%, ocasionalmente 60% y casi nunca un 20%.
- El tipo de actividades realiza con plataformas de comercio electrónico, hacen parte de su trabajo o profesión un 40%, el restante, hacen uso de plataformas para transacciones comerciales 60%
- El 60% ha hecho uso de gestores de comercio electrónico como magento y prestashop, un 40% han gestionado específicamente Prestashop y magento.
- El 80% de los consultados desconocen herramientas para la interoperabilidad entre plataformas, 20% conoce de la existencia de al menos una.
- El 80% considera que el grado de dificultad de la importación y exportación del catálogo de productos (categorías, productos y proveedores) de una plataforma de comercio electrónico a otra es complejo, el 20% restante le atribuye menor complejidad.
- Los factores que se consideran más importante cuando realiza transacciones de comercio electrónico es su funcionalidad en un 40% y su facilidad de uso en un 60%.

Cumplimiento del objetivo previsto

- Ante pregunta de si ¿Esta herramienta ofrece una asistencia útil para la gestión de sitios de comercio electrónico? El 80% de los usuarios están muy de acuerdo y un 20% de acuerdo. La respuesta frente a las posibilidades de interoperar entre aplicaciones

(plataformas) mediante el uso de la aplicación es comprensible, tienen el mismo comportamiento.

- Un 80 % considera que con el uso de la aplicación se disminuye la complejidad de interoperabilidad y reutilización entre plataformas.
- El 60% está muy de acuerdo ante la utilidad y usabilidad del editor un 40% está de acuerdo. Esto indica que el 100% de los usuarios aprueba en alto grado el cumplimiento de los objetivos previstos.

9.3 Pruebas de integración y del sistema

En estas pruebas ejecutamos las pruebas funcionales ya diseñadas anteriormente y anotamos el resultado que obtenemos, comparándolo con el que especificamos anteriormente.

9.3.1.1 Caso de uso: registrarse

Caso de Uso 1: Registrarse	
Prueba	Resultado Esperado
Añadir un usuario no existente	El sistema posee un usuario más
	Resultado Obtenido
	Inicialmente el usuario se guardaba correctamente en la base de datos pero no era posible loguearse con él, ya que faltaba relacionar el usuario con un tipo de rol para que la aplicación pudiera loguearlo.
Prueba	Resultado Esperado
Intentar registrar un usuario que ya existe	El sistema indica al usuario que ese nombre ya está registrado y no lo guarda
	Resultado Obtenido
	El sistema avisa correctamente al usuario que ese nombre de usuario existe y debe cambiarlo si quiere registrarse.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado Obtenido
	El sistema permanece sin cambios

9.3.1.2 Caso de uso: loguearse

Caso de Uso 2: Loguearse	
Prueba	Resultado Esperado
Introducir usuario y contraseña correctos	El sistema permite entrar al usuario
	Resultado Obtenido
	El sistema permite loguearse al usuario
Prueba	Resultado Esperado
Introducir usuario valido y contraseña incorrecta	El sistema indica al usuario que los datos son incorrectos y no lo deja entrar en el sistema
	Resultado Obtenido
	El sistema indica correctamente al usuario que sus credenciales no son correctas
Prueba	Resultado Esperado
Introducir usuario inválido y contraseña válida	El sistema indica al usuario que los datos son incorrectos y no lo deja entrar en el sistema
	Resultado Obtenido
	El sistema indica correctamente al usuario que sus credenciales no son correctas

9.3.1.3 Caso de uso: configurar tienda

Caso de Uso 3: configurar tienda	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de tienda y guarda la opción	El sistema almacena y muestra una nueva tienda para ese usuario en la web
	Resultado obtenido
	La tienda se guarda correctamente y se genera el mensaje <i>"Tienda creada correctamente"</i>
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la tienda e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
	Resultado obtenido
	El sistema no guarda a la tienda y surge el mensaje <i>"la contrasela y la repetición de la contraseña deben coincidir"</i>
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se vuelve a la ventana de listado de tiendas y la tienda no se ha guardado.

9.3.1.4 Caso de uso: Crear categoría

Caso de Uso 4: crear categoría

Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y guarda la opción	El sistema almacena y muestra una nueva categoría para ese usuario en la web
	Resultado obtenido
	La tienda se guarda correctamente la categoría y surge el mensaje de “categoría guardada correctamente”
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
	Resultado obtenido
	El sistema comprueba que varios de los campos obligatorios no están cubiertos y evita la creación de la categoría
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se vuelve a la ventana de categorías y la categoría no se ha guardado

9.3.1.4.1 Subcaso de uso: Crear subcategoría

<i>Caso de Uso 4: crear categoría.Subcaso de uso:crear subcategoría</i>	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de categoría y selecciona una categoría ya creada en el sistema como categoría padre	El sistema almacena correctamente la subcategoría y la muestra correctamente al usuario
	Resultado obtenido
	En el momento de realizar el test se comprueba que a pesar que las subcategorías no se guardan correctamente. Por ello fue necesario corregir el error y volver a hacer la prueba asta que esta fue correcta., guardándose la categoría y avisando al usuario con el mensaje “categoría creada correctamente”.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
	Resultado obtenido
	El sistema comprueba que varios de los campos obligatorios no están cubiertos y evita la creación de la categoría
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se vuelve a la ventana de categorías y la subcategoría no se ha guardado

9.3.1.4.2 Subcaso de uso: ver detalle de una categoría

Caso de Uso 4: crear categoría Subcaso de uso:ver detalle de una categoría	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de una categoría	Los datos de la categoría se muestran correctamente al usuario
	Resultado obtenido
	Inicialmente se detecta que alguno de los campos de la categoría no se muestran correctamente por lo que es necesario corregir el error y volver a ejecutar la prueba hasta que todo tiene el resultado esperado

9.3.1.4.3 Subcaso de uso: eliminar categoría

Caso de Uso 4: crear categoría .Subcaso de uso:eliminar categoría	
Prueba	Resultado Esperado
El usuario intenta eliminar una categoría sin hijos	El sistema elimina correctamente la categoría y le indica al usuario que todo ha ido correctamente
	Resultado obtenido
	El sistema borra correctamente la categoría del sistema y avisa al usuario con el mensaje “categoría eliminada correctamente”

9.3.1.4.4 Subcaso de uso: modificar categoría

Caso de Uso 4: crear categoría .Subcaso de Uso :modificar categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de categoría y guarda la opción	El sistema modifica la categoría e indica al usuario que los cambios se han realizado con éxito
	Resultado obtenido
	El sistema modifica correctamente la categoría y avisa al usuario con el mensaje “categoría modificada correctamente”
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación de la categoría e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
	Resultado obtenido
	El sistema avisa al usuario de que faltan campos obligatorios por cumplimentar y no guarda los cambios

9.3.1.4.5 Subcaso de uso: relacionar productos a una o más categorías

Caso de Uso 4: crear categoría Subcaso de uso: relacionar productos a una o más categorías	
Prueba	Resultado Esperado
El usuario asocia un producto con una o más categorías	El sistema asocia el producto con las categorías seleccionadas
	Resultado obtenido

	El sistema asocia el producto y la categoría y puede verse correctamente la nueva categoría asociada al producto en ver detalle
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se cancela la operación y no se producen cambios

9.3.1.5 Caso de uso: Crear producto

Caso de Uso 5: crear producto	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de productos y guarda la opción	El sistema almacena y muestra un nuevo producto para ese usuario en la web
	Resultado obtenido
	La tienda se guarda correctamente el producto y surge el mensaje de “producto creado correctamente”
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la productos e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
	Resultado obtenido
	El sistema comprueba que varios de los campos obligatorios no están cubiertos y evita la creación del producto
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se vuelve a la ventana de productos y el producto no se ha guardado

9.3.1.5.1 Subcaso de uso: ver detalle de producto

Caso de Uso 5: crear producto. Subcaso de uso: ver detalle de un producto	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un producto	Los datos del producto se muestran correctamente al usuario
	Resultado obtenido
	Inicialmente todos los campos del producto se muestran correctamente menos el código QR, ha sido necesario corregir el fallo y volver a hacer las pruebas asta que el resultado ha sido el óptimo.

9.3.1.5.2 Subcaso de uso: eliminar producto

Caso de Uso 5: crear producto. Subcaso de uso: eliminar producto	
Prueba	Resultado Esperado
El usuario elimina un producto	El sistema elimina correctamente el producto del sistema y se lo comunica al usuario
	Resultado obtenido
	El sistema borra correctamente el producto del sistema y avisa al usuario con el mensaje “producto eliminado correctamente”

9.3.1.5.3 Subcaso de uso: modificar producto

Caso de Uso 8: modificar categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de producto y guarda la opción	El sistema modifica el producto e indica al usuario que los cambios se han realizado con éxito
	Resultado obtenido
	El sistema modifica correctamente el producto y avisa al usuario con el mensaje “producto modificado correctamente”
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del producto e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
	Resultado obtenido
	El sistema avisa al usuario de que faltan campos obligatorios por cumplimentar y no guarda los cambios

9.3.1.5.4 Subcaso de uso: relacionar producto con proveedor

Caso de Uso 5: crear producto.Subcaso de uso: relacionar productos con proveedor	
Prueba	Resultado Esperado
El usuario asocia un producto con un proveedor	El sistema asocia el producto con el proveedor seleccionado
	Resultado obtenido
	El sistema asocia el producto con el proveedor, para comprobarlo se mira el detalle del producto y se comprueba que todo está correcto
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se cancela la operación y no se producen cambios

9.3.1.6 Caso de uso: Crear proveedor

Caso de Uso 6: crear proveedor	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de creación de proveedores y guarda la opción	El sistema almacena y muestra un nuevo producto para ese usuario en la web
	Resultado obtenido
	Inicialmente se detecta que el sistema no valida correctamente los números de CIF por lo que es necesario realizar una serie de cambios para que esto funcione finalmente correctamente.
Prueba	Resultado Esperado
El usuario rellena mal el formulario de creación de la proveedor e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda la configuración
	Resultado obtenido
	El sistema comprueba que varios de los campos obligatorios no están cubiertos y evita la creación del proveedor
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	Se vuelve a la ventana de proveedores y el proveedor no se ha guardado

9.3.1.6.1 Subcaso de uso: ver detalle de proveedor

Caso de Uso 6:crear proveedor.Subcaso de uso: ver detalle de un proveedor	
Prueba	Resultado Esperado
El usuario selecciona ver detalle de un proveedor	Los datos del proveedor se muestran correctamente al usuario
	Resultado obtenido
	Todos los detalles de proveedor se muestran correctamente.

9.3.1.6.2 Subcaso de uso: eliminar producto

Caso de Uso 6:crear proveedor.Subcaso de uso:eliminar proveedor	
Prueba	Resultado Esperado
El usuario elimina un proveedor	El sistema elimina correctamente el proveedor del sistema y se lo comunica al usuario
	Resultado obtenido
	El sistema borra correctamente el proveedor del sistema y avisa al usuario con el mensaje “proveedor eliminado correctamente”

9.3.1.6.3 Subcaso de uso: modificar proveedor

Caso de Uso 8:modificar categoría	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de proveedor y guarda la opción	El sistema modifica el proveedor e indica al usuario que los cambios se han realizado con éxito
	Resultado obtenido
	El sistema modifica correctamente el proveedor y avisa al usuario con el mensaje “proveedor modificado correctamente”
Prueba	Resultado Esperado
El usuario rellena mal el formulario de modificación del proveedor e intenta guardar los cambios	El sistema le indica al usuario en que parte del formulario existe un error y no guarda los cambios
	Resultado obtenido
	El sistema avisa al usuario de que faltan campos obligatorios por cumplimentar y no guarda los cambios

9.3.1.7 Caso de uso: importar elementos

Caso de Uso 7 : importar elementos	
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada correctamente	El sistema importa los elementos seleccionados por el usuario y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
	Resultado obtenido
	Inicialmente importar correctamente los elementos pero existe un problema entre las relaciones entre categorías , por lo que es necesario reajustar los métodos un par de veces para que la operación se realice correctamente.
Prueba	Resultado Esperado
El usuario intenta importar elementos con una tienda configurada incorrectamente	El sistema intenta conectarse a la tienda pero al ser la configuración incorrecta la conexión no puede realizarse y por lo tanto no se pueden importar los elementos. Se informa al usuario que la operación no pudo realizarse con éxito.
	Resultado obtenido
	El sistema no puede importar los elementos e indica al usuario que ha habido un error.
Prueba	Resultado Esperado
El usuario intenta importar elementos de una tienda cuya base de datos está corrupta.	El sistema intenta importar los elementos pero al estar la base de datos corrupta no puede realizar la conexión. Se informa al usuario que la operación no pudo realizarse con éxito
	Resultado obtenido
	El sistema no puede importar los elementos e indica al usuario que ha habido un error.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	El sistema no importa ningún elemento y vuelve al listado de tiendas.

9.3.1.8 Caso de uso: exportar elementos

Caso de Uso 8:exportar elementos	
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda configurada correctamente	El sistema exporta los elementos y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
	Resultado obtenido
	Inicialme se comprueba que existen campos de los elementos que no exporta correctamente. Habrá que realizar varios cambios asta que finalmente el resultado es el esperado
Prueba	Resultado Esperado
El usuario exportar elementos a una tienda configurada incorrectamente	El sistema intenta conectarse a la tienda pero al ser la configuración incorrecta la conexión no puede realizarse y por lo tanto no se pueden exportar los elementos. Se informa

	al usuario que la operación no pudo realizarse con éxito.
	Resultado obtenido
	El sistema no puede exportar los elementos e indica al usuario que ha habido un error.
Prueba	Resultado Esperado
El usuario intenta exportar elementos a una tienda cuya base de datos está corrupta.	El sistema intenta exportar los elementos pero al estar la base de datos corrupta no puede realizar la conexión. Se informa al usuario que la operación no pudo realizarse con éxito
	Resultado obtenido
	El sistema no puede exportar los elementos e indica al usuario que ha habido un error.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	El sistema no exporta ningún elemento

9.3.1.9 Caso de uso: importar XML

Caso de Uso 9:importar XML	
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato correcto	El sistema importa los elementos contenidos en el XML y las relaciones entre ellos notificándole al usuario que todo ha ido correctamente
	Resultado obtenido
	El sistema indica al usuario que los elementos se importaron correctamente. Se verifica que cada uno de ellos está creado correctamente en el sistema
Prueba	Resultado Esperado
El usuario intenta importar un XML con formato incorrecto	El sistema importar el XML pero al ser el formato incorrecto se notifica al usuario el problema y el sistema permanece sin cambios
	Resultado obtenido
	El sistema indica que el formato del XML no es el adecuado y no importa los elementos.
Prueba	Resultado Esperado
Cancelar la Operación	El sistema permanece sin cambios.
	Resultado obtenido
	El sistema no importa ningún elemento

9.3.1.10 Caso de uso:Generar CSV

Caso de Uso 10:Generar CSV

Prueba	Resultado Esperado
El usuario genera un CSV	El sistema recoge la lista del productos del usuario y genera un archivo de tipo CSV que será compatible con las plataformas Magento y Prestashop
	Resultado obtenido
	El sistema genera un archivo CSV correctamente. Para verificar que es compatible con las plataformas Magento y Prestashop se importa este archivo en cada una de ellas y se comprueba que funciona correctamente.

9.3.1.11 Caso de uso: Editar datos de usuario

Caso de Uso 11:Generar CSV	
Prueba	Resultado Esperado
El usuario rellena correctamente el formulario de modificación de sus datos personales y guarda la opción	El sistema modifica los datos del usuario e indica al éste que los cambios se han realizado con éxito
	Resultado obtenido
	Inicialmente no validaba correctamente el formulario y no hacía una comprobación que la contraseña y la repetición de contraseña fueran iguales. Una vez detectado el problema se soluciona y se vuelve a probar asta que el funcionamiento es el adecuado

9.3.1.12 Caso de uso: Crear Categoría UML

Caso de Uso 12:Crear categoría UML	
Prueba	Resultado Esperado
El usuario arrastra un componente categoría de la paleta de elementos a la zona de dibujo	El sistema pinta una categoría con sus atributos y características.
	Resultado obtenido
	El sistema pinta correctamente la categoría con sus atributos.
Prueba	Resultado Esperado
El usuario dentro de la zona de dibujo selecciona la opción categoría	El sistema pinta una categoría con sus atributos y características.
	Resultado obtenido
	El sistema pinta correctamente la categoría con sus atributos.

9.3.1.13 Caso de uso: crear subcategoría UML

Caso de Uso 13:Crear subcategoría UML	
Prueba	Resultado Esperado
El usuario enlaza dos categorías e indica cual es la	El sistema pinta un nexo de unión entre las dos categorías creadas por el usuario

categoría base	
	Resultado obtenido
	Se tuvo que retocar el estilo de los enlaces debido a que no los pintaba correctamente.Finalmente funcionó correctamente

9.3.1.14 Caso de uso: crear productoUML

Caso de Uso 14:Crear producto UML	
Prueba	Resultado Esperado
El usuario arrastra un componente producto de la paleta de elementos a la zona de dibujo	El sistema pinta un producto con sus atributos y características.
	Resultado obtenido
	Se pinta correctamente el producto y sus atributos.

9.3.1.15 Caso de uso: crear proveedor UML

Caso de Uso 15:Crear proveedor UML	
Prueba	Resultado Esperado
El usuario arrastra un componente proveedor de la paleta de elementos a la zona de dibujo	El sistema pinta un proveedor con sus atributos y características.
	Resultado obtenido
	Se pinta correctamente el proveedor y sus atributos.

9.3.1.16 Caso de uso: relacionar un producto con un proveedor

Caso de Uso 16:relacionar un producto con un proveedor	
Prueba	Resultado Esperado
El usuario enlaza un producto con un proveedor	El sistema pinta un nexo de unión entre el producto y el proveedor
	Resultado obtenido
	Se pinta correctamente el enlace entre el producto y el proveedor
Prueba	Resultado Esperado
El usuario intenta enlazar un producto con más de un	El sistema pinta un nexo de unión entre el producto y el primer proveedor con el que se intenta unir pero no permite

proveedor	unirlo con más de uno
	Resultado obtenido
	Únicamente permite enlazar un producto con un único proveedor e impide que se relacione con más

9.3.1.17 Caso de uso: generar XML

Caso de Uso 17: generar XML	
Prueba	Resultado Esperado
El usuario genera un XML después de haber creado elementos XML	El sistema genera un XML con toda la información de los elementos UML creados y sus relaciones
	Resultado obtenido
	Se genera correctamente un XML con toda la información recogida de los XML. Los campos de cada una de las entidades que encuentra a null no los pone en el XML
Prueba	Resultado Esperado
El usuario intenta generar un XML sin haber creado previamente elementos UML	El sistema crea un XML pero sin información de ningún elemento
	Resultado obtenido
	Se crea un XML sin información.

9.4 Pruebas de rendimiento

Las pruebas de rendimiento de la aplicación se han limitado a comprobar que el flujo entre ventanas y operaciones en la web es suficientemente aceptable para que los usuarios no tengan la sensación de errores o cuelgue.

Únicamente se han detectado unos tiempos de respuesta algo más elevados cuando se realizan operaciones a gran escala entre las plataformas, pero aun así se consideran tiempos de respuesta asumiblemente aceptables.

Capítulo 10. Manuales del Sistema

10.1 Manual de Instalación y ejecución

10.1.1 GMF

En este caso para generar un instalador propio para el GMF era un trabajo bastante complicado por lo que se opta por ejecutar desde el propio eclipse ya que nos proporciona un entorno gráfico apropiado para ello.

Para instalar y ejecutar el GMF se requiere tener instalado un eclipse con todos los plugins. Lo más sencillo es bajarse el entorno con todo ya instalado (<http://www.eclipse.org/downloads/packages/eclipse-modeling-tools/junosr1>).



Se importa el conjunto de proyectos mostrado a continuación y proporcionado en el cd de instalación.

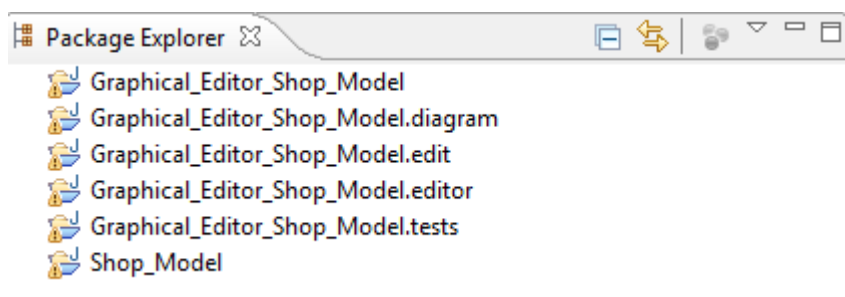


Ilustración 170 importación de proyectos

El proyecto “Graphical_Edditor_Shop_Model.diagram” se ejecutará como una aplicación eclipse.

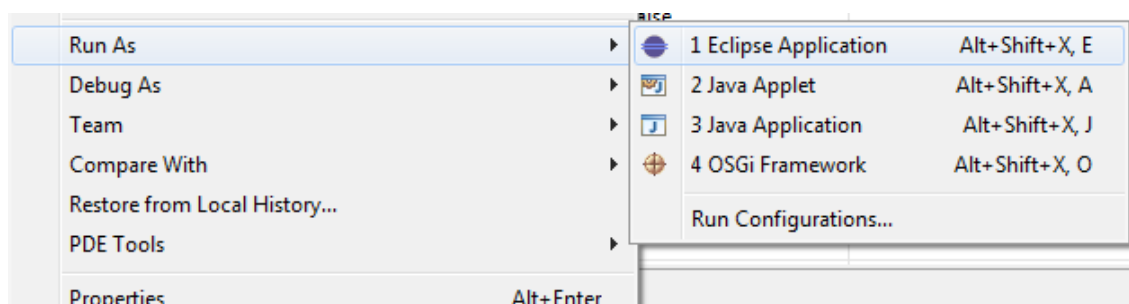


Ilustración 171 Ejecutar como un proyecto eclipse

En la nueva ventana de eclipse que nos abre importaremos el proyecto “MDEWebModel”. En este proyecto se encuentra un archivo llamado “Diagram.shop_model_diagram”. Se hace doble clic sobre él. Si todo ha ido correctamente nos abrirá el editor gráfico.

10.1.2 Plataforma Web

Los requisitos necesarios para la instalación de la aplicación web son los siguientes:

- Apache Tomcat
- MySQL

10.1.2.1 Instalación de Apache Tomcat

1. Descargar Tomcat de su sitio oficial (<http://tomcat.apache.org/>) y seleccionar la descarga que prefiera. En nuestro caso se usará la versión 6.x. Se seleccionará el archivo a descargar. En este caso se descarga la versión .zip.

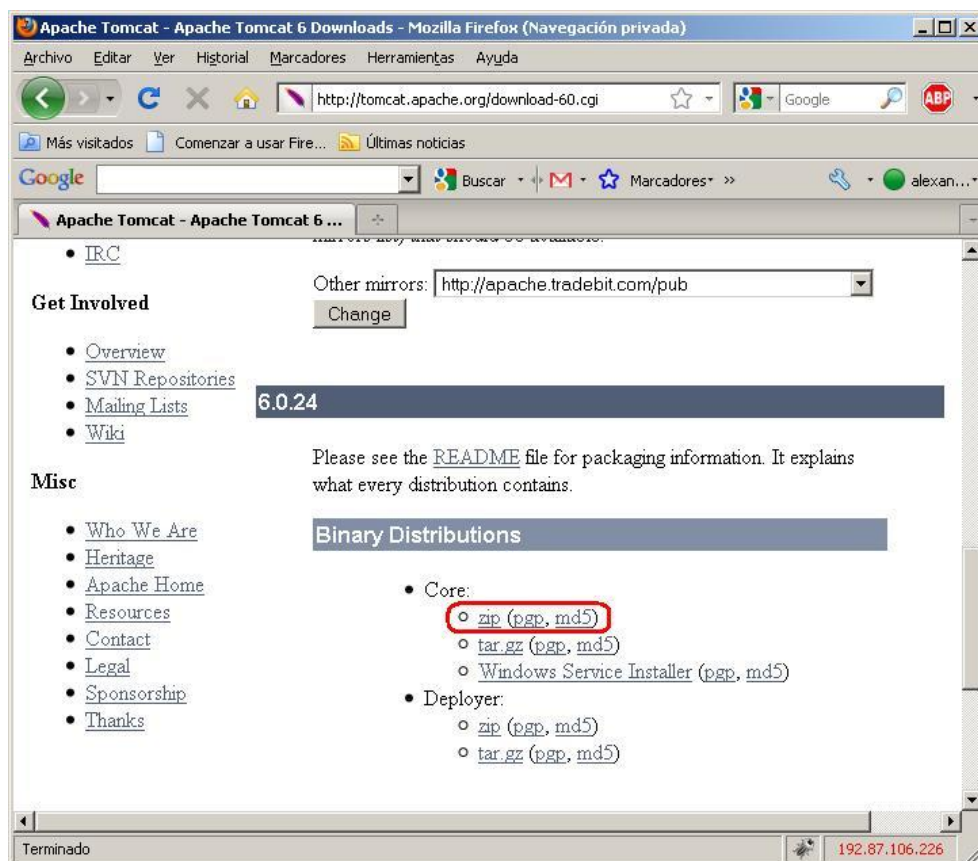


Ilustración 172 descarga de apache tomcat

- Se descomprime el archivo en el disco duro. Para que pueda funcionar correctamente se debe tener Java instalado y la variable JAVA_HOME configurada apuntando al directorio de instalación.
- Se arranca el servidor ejecutando el archivo bin\startup.bat.

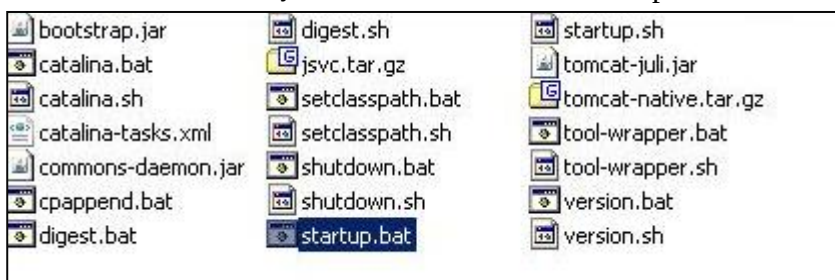
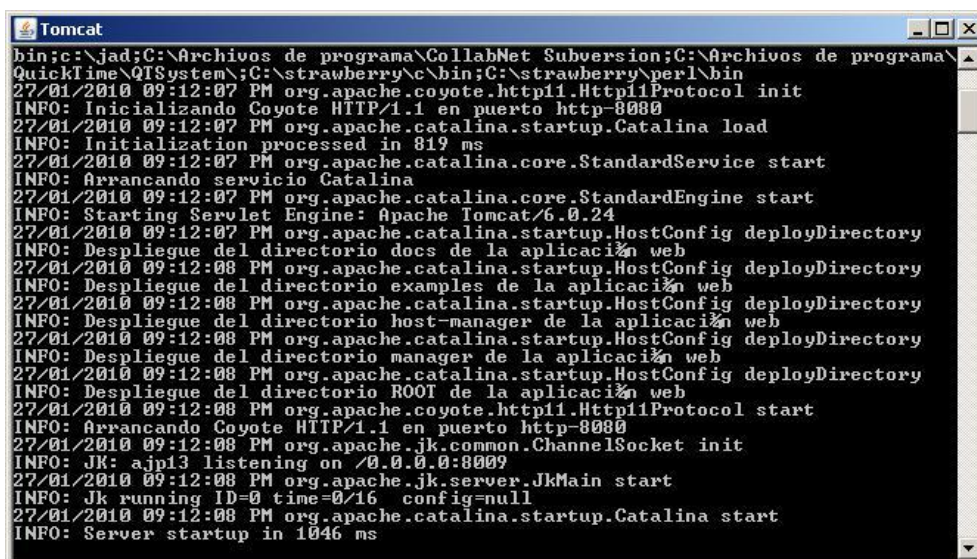


Ilustración 173 ejecución del tomcat

- Si todo funciona correctamente debe aparecer una consola de comandos como la que se ve a continuación:



```

bin;c:\jad;C:\Archivos de programa\CollabNet Subversion;C:\Archivos de programa\
QuickTime\QTSystem;C:\strawberry\c\bin;C:\strawberry\perl\bin
27/01/2010 09:12:07 PM org.apache.coyote.http11.Http11Protocol init
INFO: Inicializando Coyote HTTP/1.1 en puerto http-8080
27/01/2010 09:12:07 PM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 819 ms
27/01/2010 09:12:07 PM org.apache.catalina.core.StandardService start
INFO: Arrancando servicio Catalina
27/01/2010 09:12:07 PM org.apache.catalina.core.StandardEngine start
INFO: Starting Servlet Engine: Apache Tomcat/6.0.24
27/01/2010 09:12:07 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Despliegue del directorio docs de la aplicación web
27/01/2010 09:12:08 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Despliegue del directorio examples de la aplicación web
27/01/2010 09:12:08 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Despliegue del directorio host-manager de la aplicación web
27/01/2010 09:12:08 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Despliegue del directorio manager de la aplicación web
27/01/2010 09:12:08 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Despliegue del directorio ROOT de la aplicación web
27/01/2010 09:12:08 PM org.apache.coyote.http11.Http11Protocol start
INFO: Arrancando Coyote HTTP/1.1 en puerto http-8080
27/01/2010 09:12:08 PM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
27/01/2010 09:12:08 PM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/16 config=null
27/01/2010 09:12:08 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 1046 ms

```

Ilustración 174 ejecución Tomcat

- Para comprobar que todo está correctamente instalado abra el navegador y visite la dirección <http://localhost:8080>. Deberá aparecer una página como esta.

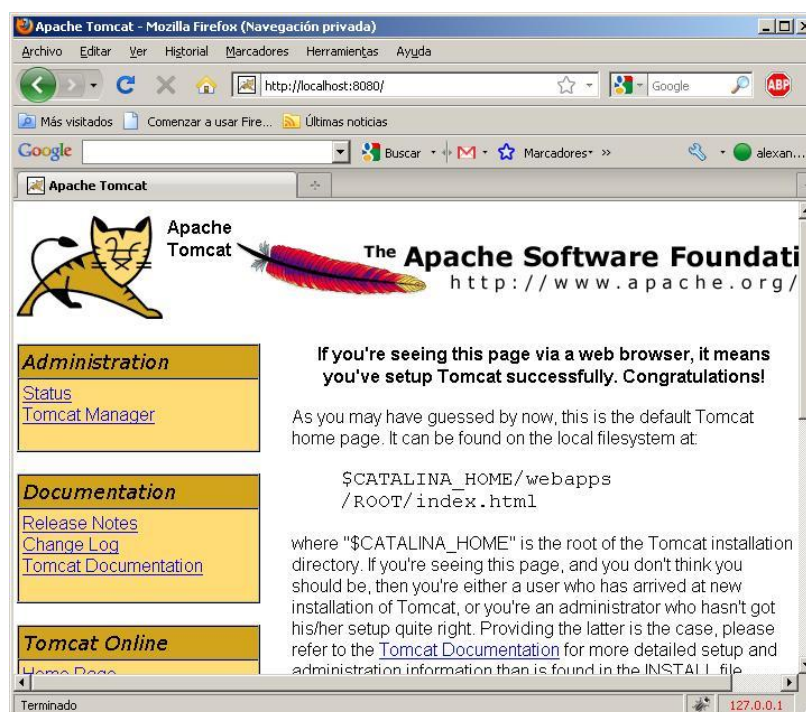


Ilustración 175 Apache tomcat funcionando

10.1.2.2 Instalación de Mysql

- En primer lugar debemos descargarnos MySQL. Para ello nos dirigimos a la sección de descargas de la página oficial (<http://dev.mysql.com/downloads/>) y se elegirá MySQL Community Server, que es la versión gratuita del producto. Selecciona Windows como plataforma y elegir el instalador MSI que mejor se adapte a su sistema operativo (32 o 64 bits).

2. Se arranca el ejecutable y aparecerá un instalador como el siguiente.

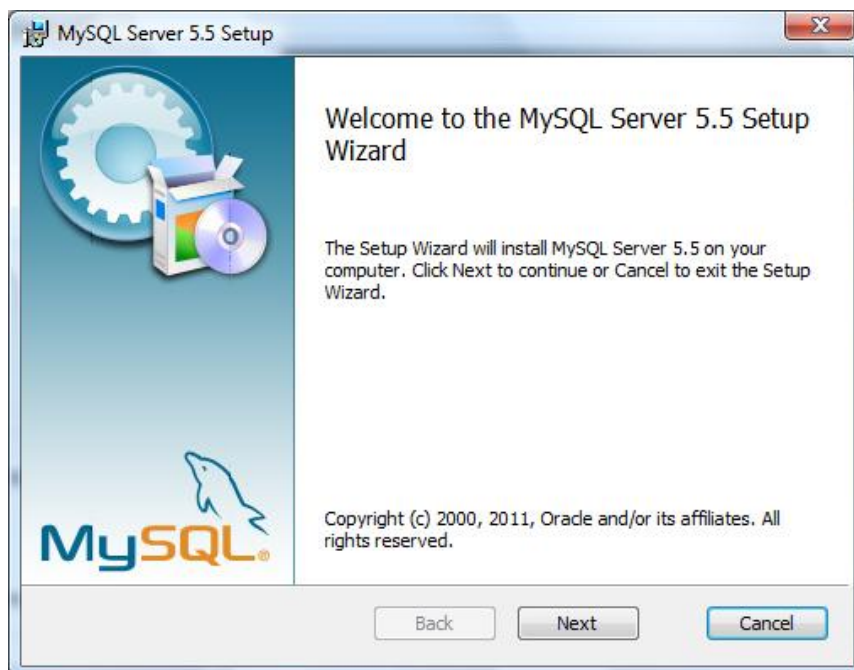


Ilustración 176 Instalación de Mysql paso I

3. Se aceptan los términos y se sigue una instalación típica. Cuando se llega a la siguiente ventana se presionará el botón Install.

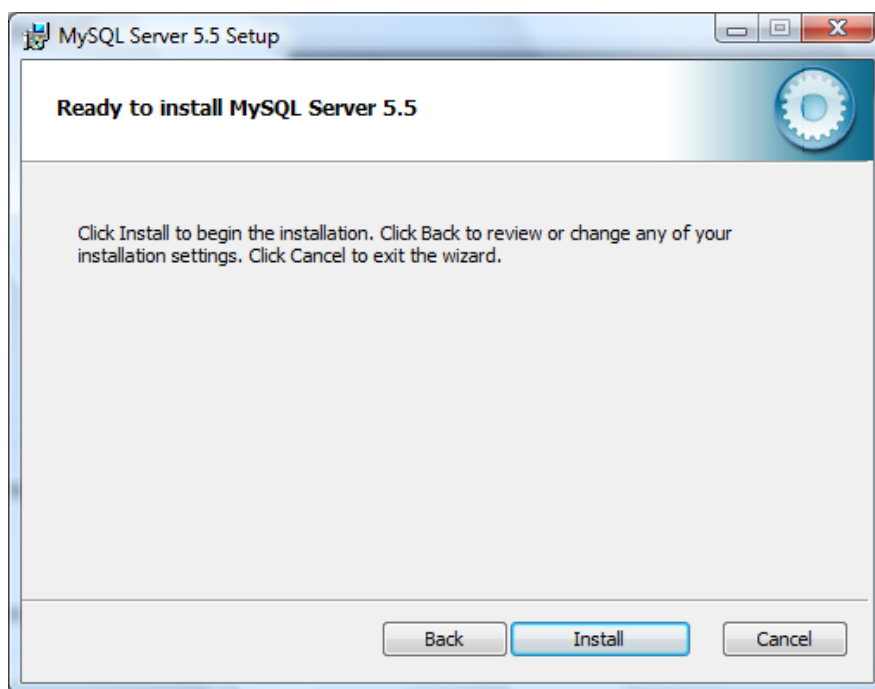


Ilustración 177 Mysql instalación Paso II

4. Una vez instalado *MySQL*, la siguiente fase es la configuración del servidor en sí mismo. Optamos por *Detailed Configuration*, de modo que se optimice la configuración del servidor *MySQL*.

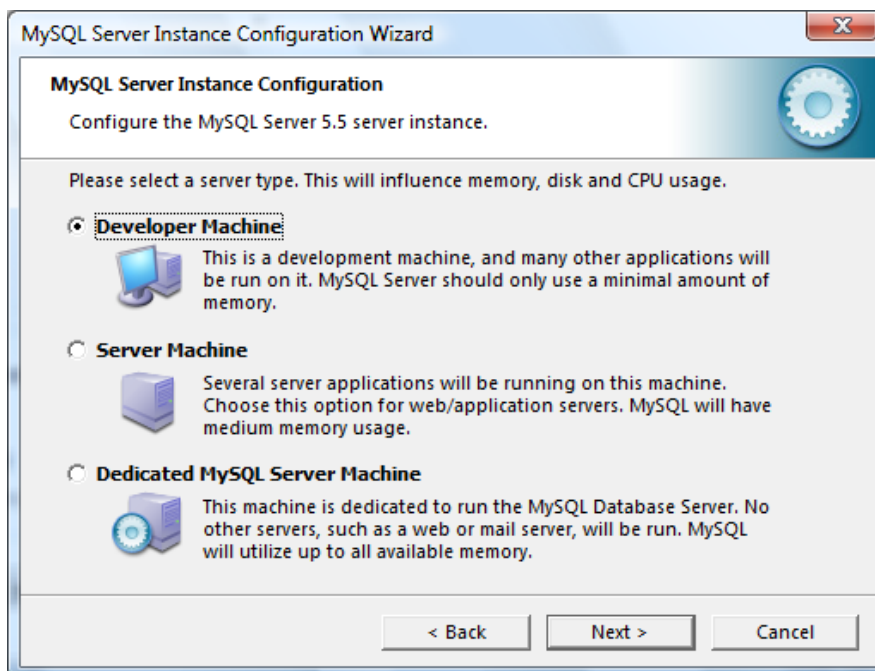


Ilustración 178 Instalación de Mysql Paso III

5. Se continua la instalación y se selecciona, *Multifunctional Database*.

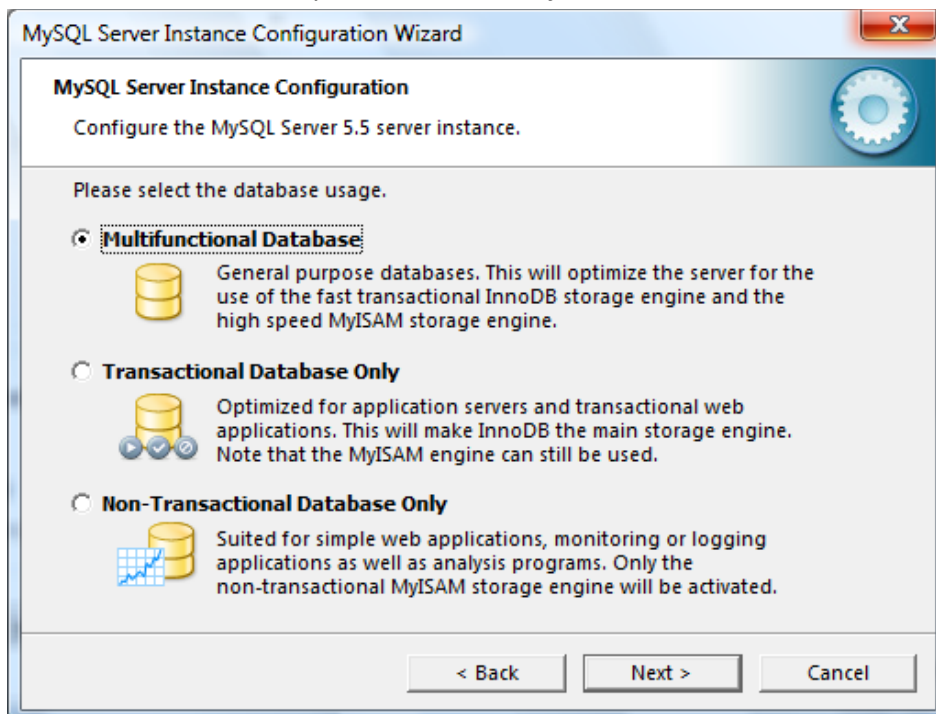


Ilustración 179 Instalación de Mysql Paso IV

6. Se prodigue la instalación dejando todas las opciones que trae por defecto, e introducimos una contraseña para el usuario root (en este caso para que se conecte correctamente la web la contraseña que le pondremos será root).

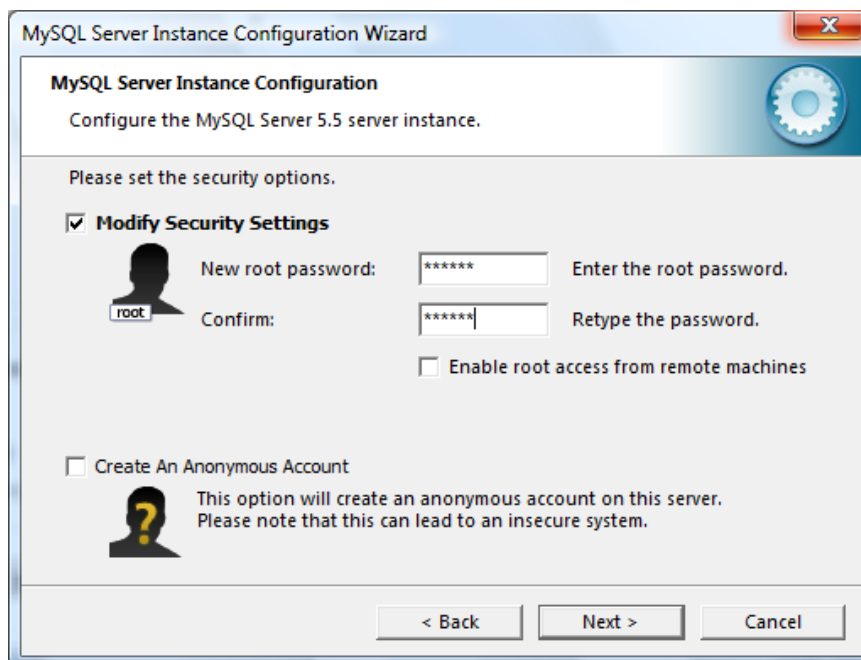


Ilustración 180 Instalación de Mysql Paso IV

7. Para finalizar se genera el fichero de configuración y se arrancar el servicio.

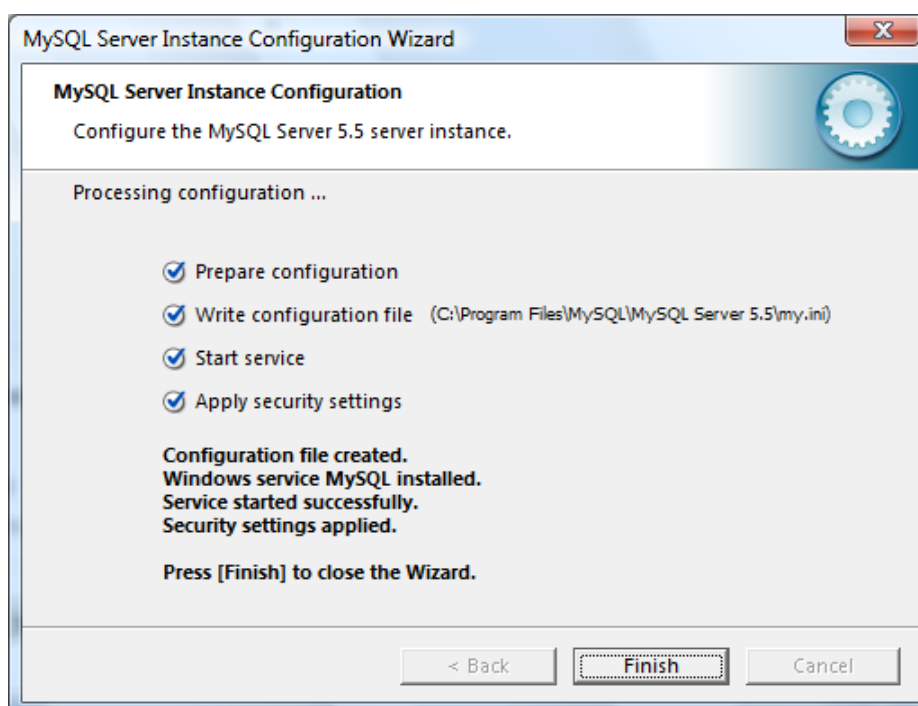


Ilustración 181 Finalización de la instalación de Mysql

10.1.2.3 Instalación y ejecución de la web

Una vez que se tienen instalados tanto el Tomcat como Mysql se procede a instalar la web. Para ello simplemente es necesario crear una base de datos con el nombre de “tunosdb”.

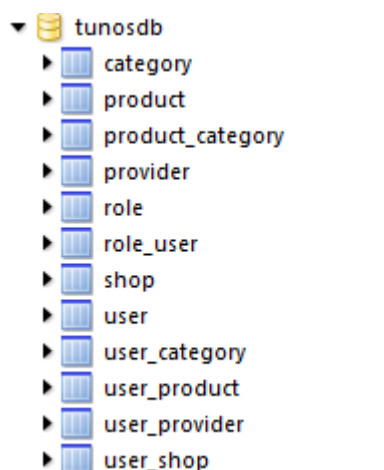


Ilustración 182 Tunosdb

Tanto las tablas como el contenido mínimo de datos para que la aplicación funcione se crearán automáticamente cuando se arranque la web.

En segundo lugar se incorporara el fichero commerce.war (proporcionado en el CD de instalación del proyecto) al directorio del tomcat “tomcat\webapps”.

Si todo está correctamente configurado se podrá acceder a la web en el navegador mediante la URL <http://localhost:8080/commerce/spring/login>.

10.2 Manual de Usuario

A continuación se detalla cómo debe utilizarse tanto la aplicación web como el GMF.

10.2.1 Manual de usuario de la aplicación web

10.2.1.1 Acceso a la plataforma (Login)

La aplicación web *Tunos Commerce* permite el acceso a la plataforma a través de un usuario y una contraseña. Si el usuario se encuentra dado de alta en la aplicación podrá acceder utilizando el formulario de Iniciar sesión.

En este formulario deberá introducir los siguientes campos:

- Usuario: nombre de usuario que se utilizó cuando se registró en la plataforma.
- Clave: contraseña que se utilizó cuando se registró en la plataforma

Si el usuario y la clave son correctos la aplicación le dará acceso al sistema.

Si el usuario desea recordar sus datos de inicio de sesión desde el navegador donde está accediendo debe marcar el checkbox *"recordar mis datos"*.

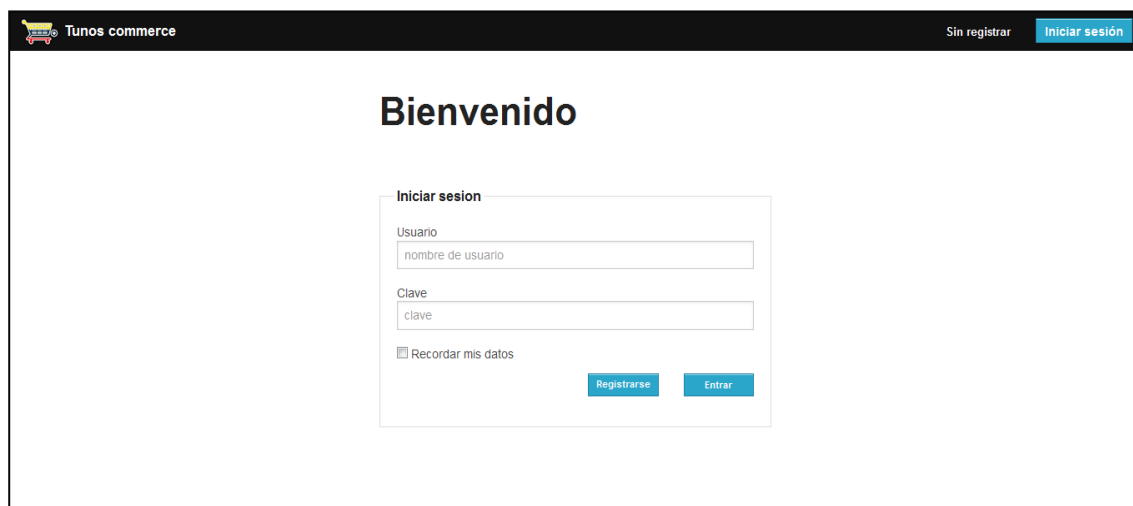


Ilustración 183 Inicio de sesión

10.2.1.2 Registro de usuarios

Sí el usuario no está registrado en el sistema es necesario que se registre para poder iniciar sesión. Para ello a partir de la venta de login se seleccionará la opción registrarse. A continuación aparecerá el siguiente formulario.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de creación de un nuevo usuario y se retornará a la página de inicio de sesión.

Si por el contrario se desea proseguir, en este formulario de creación de nuevo usuario se pueden introducir los siguientes detalles:

- Nombre de usuario: esté será el nombre que se le asignará para iniciar sesión
- Nombre: nombre de usuario
- Apellidos: apellidos del usuario
- E-mail: Correo electrónico del usuario. Ha de ser un correo electrónico con formato válido del tipo [ususario@dominio.foo](#)
- Contraseña: Contraseña que utilizará el usuario junto su nombre de usuario para acceder al sistema. No se mostrarán los caracteres al escribir la contraseña, únicamente una representación gráfica mediante un punto por cada carácter introducido.
- Repetir contraseña: Con el fin de validar la contraseña ha de volver a escribirse la contraseña utilizada en el campo Contraseña. No se mostrarán los caracteres al escribir la contraseña, únicamente una representación gráfica mediante un punto por cada carácter introducido. Tanto la contraseña como al repetirla deben coincidir para que el registro sea válido.

The screenshot shows the 'Registrar nuevo usuario' form. At the top, there is a header bar with the 'Tunos commerce' logo on the left and 'Sin registrar' and 'Iniciar sesión' links on the right. The main title of the form is 'Registrar nuevo usuario'. The form itself is a white box containing several input fields: 'Nombre de usuario:', 'Nombre:', 'Apellidos:', 'E-mail:', 'Contraseña:', and 'Repita su contraseña:'. Each field has a corresponding text input box. At the bottom of the form, there are two buttons: 'Registrar' and 'Cancelar'.

Ilustración 184 Pantalla de registro de usuarios

Cuando todos los datos han sido cubiertos se seleccionará la opción registrar. En caso de ir todo correctamente le aparecerá un mensaje similar al siguiente:

The screenshot shows the 'Registro de usuarios' confirmation screen. It has the same header bar as the previous screen. The main title is 'Registro de usuarios'. Below the title, there is a green message box that says 'Usuario creado correctamente.' and a blue 'Aceptar' button.

Ilustración 185 Usuario registrado correctamente

En caso de haber algún tipo de fallo en la cumplimentación del formulario se le indicará debidamente en la parte superior del formulario.

Ilustración 186 Errores en el formulario de registro de un nuevo usuario

10.2.1.3 Salir de la plataforma (Logout)

Para cerrar la sesión en la plataforma el usuario debe seleccionar el botón de cerrar sesión que se encuentra en la parte superior derecha

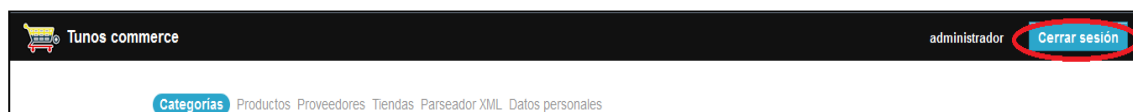


Ilustración 187 Cerrar sesión

Si todo ha ido correctamente le aparecerá el siguiente mensaje:

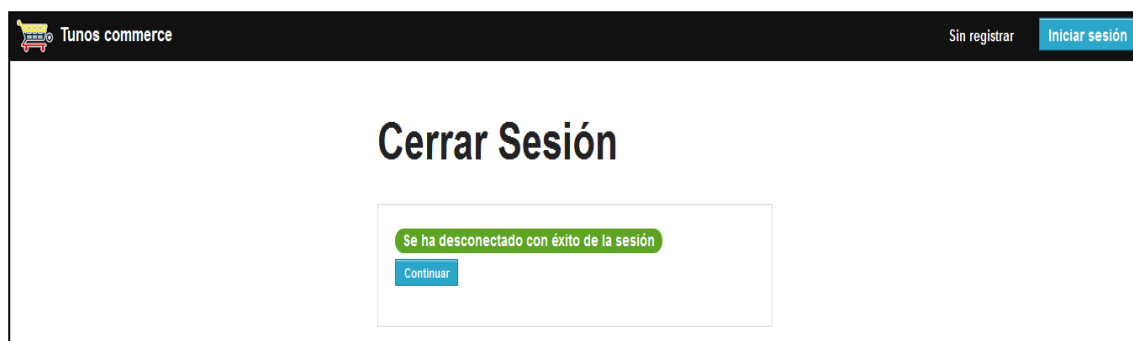


Ilustración 188 Pantalla de desconectar sesión correctamente

10.2.1.4 Pantalla principal (intro)

Una vez el usuario ha iniciado sesión la pantalla que aparece en primer lugar es la que se muestra a continuación.



Ilustración 189 Pantalla principal (intro)

Como se puede ver en la figura esta página muestra información acerca de la plataforma y las opciones y ventajas que esta ofrece al usuario. Desde esta pantalla se podrá acceder al resto de las opciones seleccionando los diferentes ítems que contiene el menú superior.

Las opciones disponibles en el menú son las siguientes:

- **Categoría:** En esta opción se podrá acceder al apartado categoría, en él se podrán realizar todas las opciones disponibles con relación a este elemento.
- **Productos:** En esta opción se podrá acceder al apartado producto, aquí se podrán ver y realizar todas las operaciones relativas a los productos.
- **Proveedores:** En esta opción se podrá acceder al apartado proveedor, aquí se podrán ver y realizar todas las operaciones relativas a los productos.
- **Tiendas:** En esta opción se podrá acceder al apartado tiendas, aquí se podrán configurar las diferentes tiendas que disponga el usuario de tipo Magento o Prestashop.
- **Parseador XML:** Esta opción está dedicada a la importación de elementos (categorías, productos o proveedores,) a partir de un XML.
- **Datos personales:** Esta opción está dedicada a todo lo relativo a los datos del usuario que ha iniciado sesión.

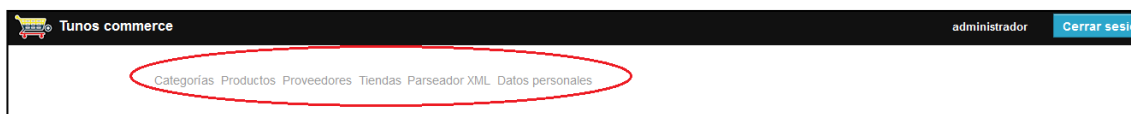


Ilustración 190 Menú horizontal

Sí desde alguna otra ventana el usuario desea volver a esta ventana principal basta con seleccionar el icono o nombre superior izquierdo de la pantalla.

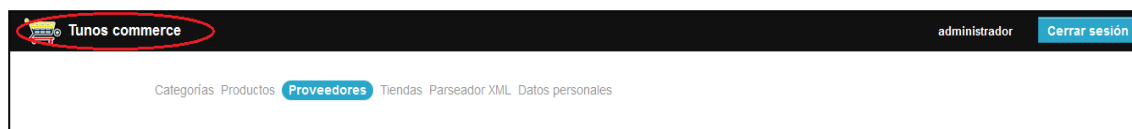


Ilustración 191 Volver a la pantalla principal

10.2.1.5 Gestión de categorías

El sistema permite ver un listado de las categorías creadas por el usuario. Se podrá acceder a la sección de Categorías mediante la opción de menú categorías.

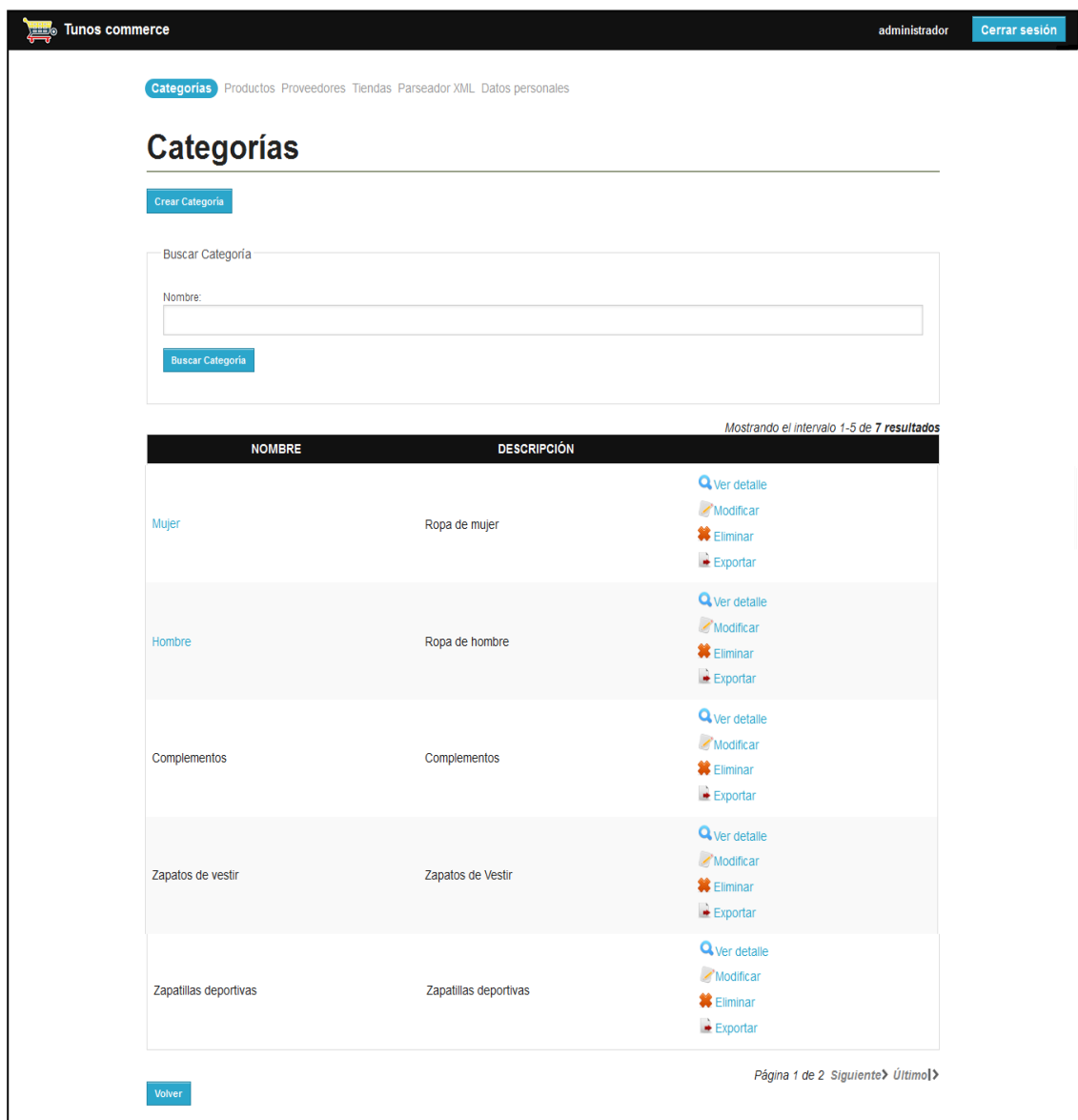


Ilustración 192 Sección categorías

Algunos de los datos que se podrán observar sobre cada registro de categoría listada serán:

- Nombre: Nombre de la categoría
- Descripción: Descripción de la categoría.
- Acciones: Enlaces a las acciones que se pueden realizar para cada categoría (Ver detalle, modificar, eliminar o exportar)

Los resultados del listado se muestran paginados. Para navegar sobre las sucesivas páginas basta con seleccionar la acción Siguiente o Anterior según se deseé.

También se podrá navegar por las categorías. Para ver las categorías hijo que estas tienen, basta con clicar sobre el enlace del nombre de la categoría y si esta tiene hijos (el nombre de las categorías que tienen hijos están resaltados con color azul y son clicables mientras que las categorías que no tienen sub-categorías aparecen en negro y no son clicables) se accederá a estos. Para volver al nivel inicial se clicará al botón *volver* que aparece debajo de la tabla.



Ilustración 193 Botón volver

10.2.1.5.1 Buscar categoría

En los listados de categorías existe la posibilidad de encontrar una categoría o conjunto de categorías en concreto utilizando la Búsqueda de Categorías.

En este caso las categorías se pueden buscar por nombre. Para buscar una categoría por nombre se introducirá el nombre de esta en el formulario de buscar y a continuación se seleccionará el botón buscar.

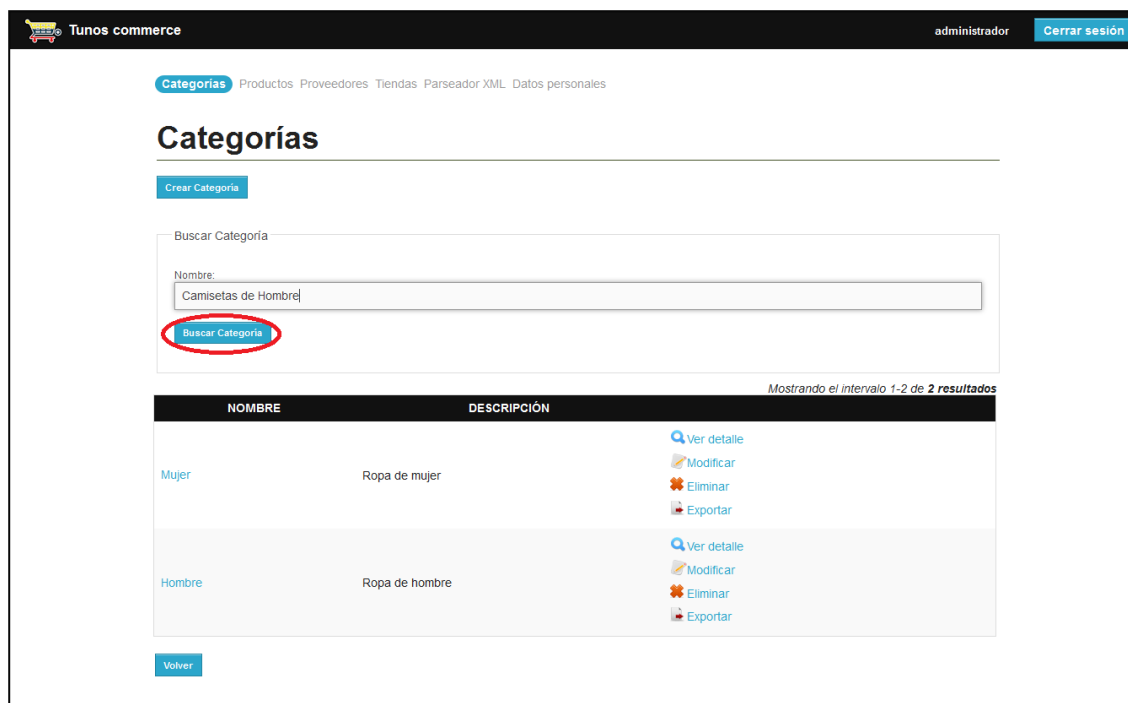


Ilustración 194 Buscar categoría por nombre

10.2.1.5.2 Crear categoría

Seleccionando la sección de Categorías, se podrá observar una opción Nueva categoría. Al seleccionar esta opción se muestra un formulario para la creación de una nueva categoría

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de creación de la nueva categoría y se le retornará a la página de Listado y Búsqueda de Categorías.

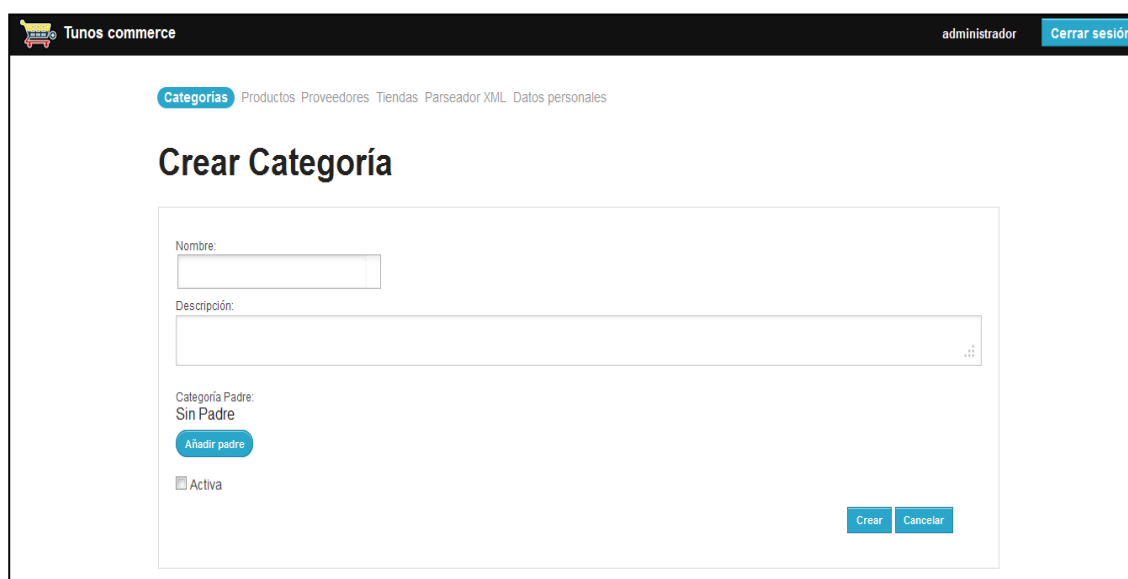


Ilustración 195 Crear Categoría

Si por el contrario desea proseguir, en este formulario de creación de nueva categoría se pueden introducir los siguientes detalles:

- Nombre: Nombre de la categoría.
- Descripción: Descripción detallada de la categoría
- Padre: categoría padre de la categoría que se está creando

Para seleccionar la categoría padre basta con acceder al listado de categorías a través del enlace “Añadir Padre” y seleccionar la categoría deseada.

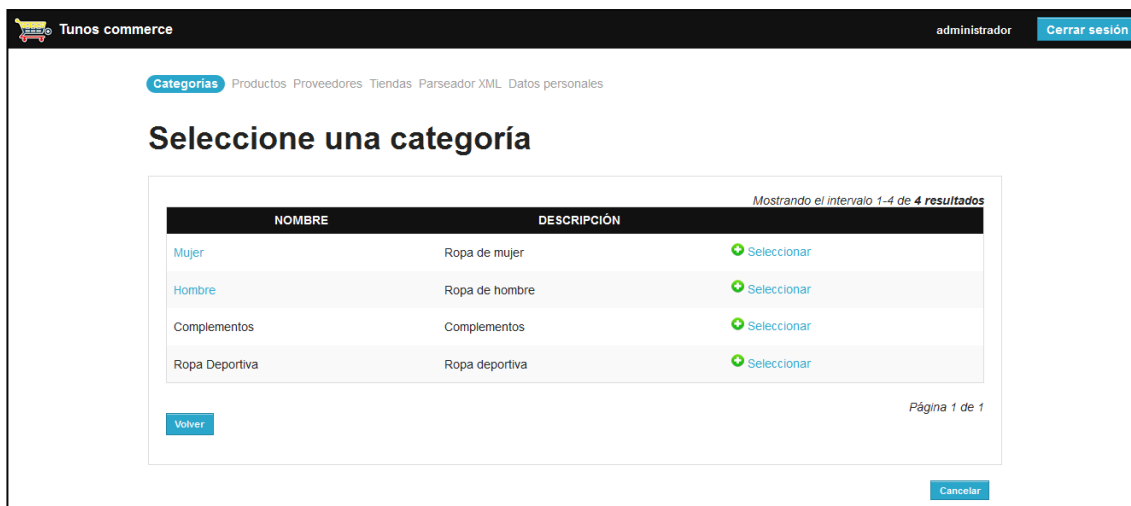


Ilustración 196 Seleccionar categoría padre

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente. Una vez rellenados los campos necesarios para la creación de un nueva categoría se debe seleccionar la acción Crear.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Crear.



Ilustración 197 Confirmación de crear una nueva categoría

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Crear*. Por el contrario si se desea cancelar el proceso se seleccionará *Cancelar* y si se desea volver a la pantalla anterior de creación para cambiar algún campo se seleccionará *Volver*.

Si se ha decidido confirmar el proceso, la nueva categoría será creada en el sistema y podrá visualizarse en el listado de categorías.

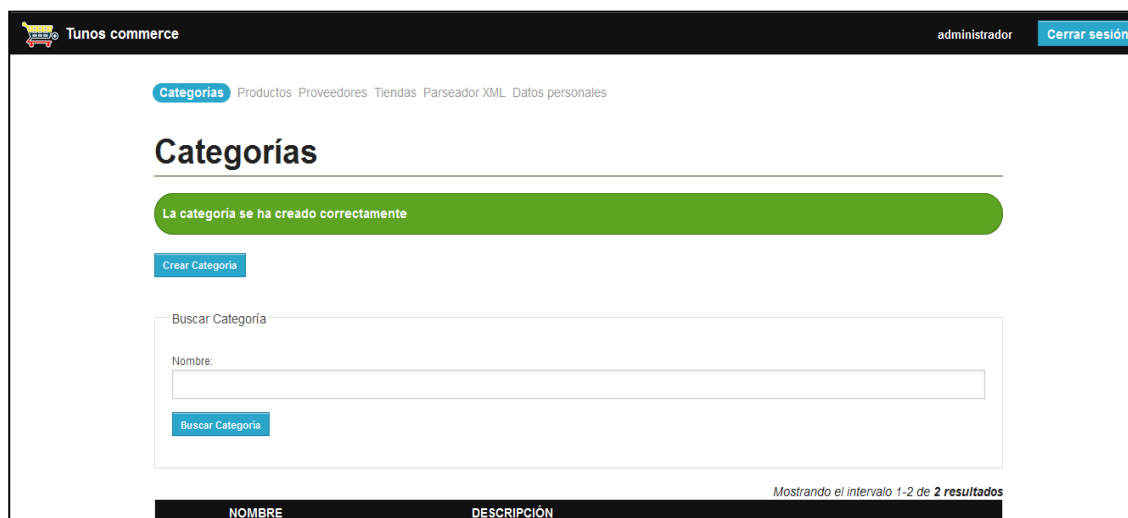


Ilustración 198 Categoría creada correctamente

10.2.1.5.3 Ver detalle de una categoría

El sistema permite consultar los detalles de las categorías creadas. Se puede acceder a los detalles de la categoría pulsando sobre el enlace de ver detalle de la categoría que se desea visualizar.

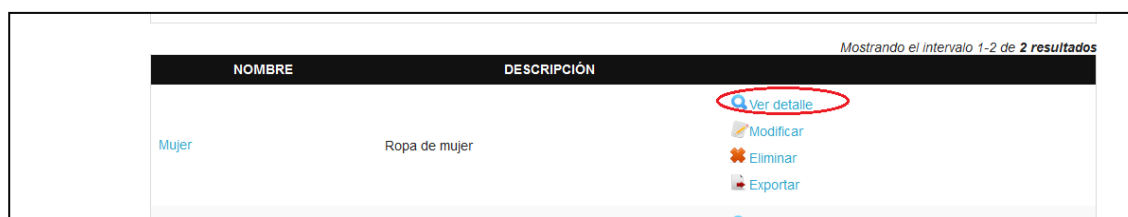


Ilustración 199 Enlace ver detalle de una categoría

Se mostrará una pantalla con las propiedades de la categoría. Los detalles de la categoría que se muestran son los siguientes:

- Nombre: Nombre de la categoría.
- Descripción: Descripción de la categoría.
- Subcategorías: Lista de subcategorías de esa categoría

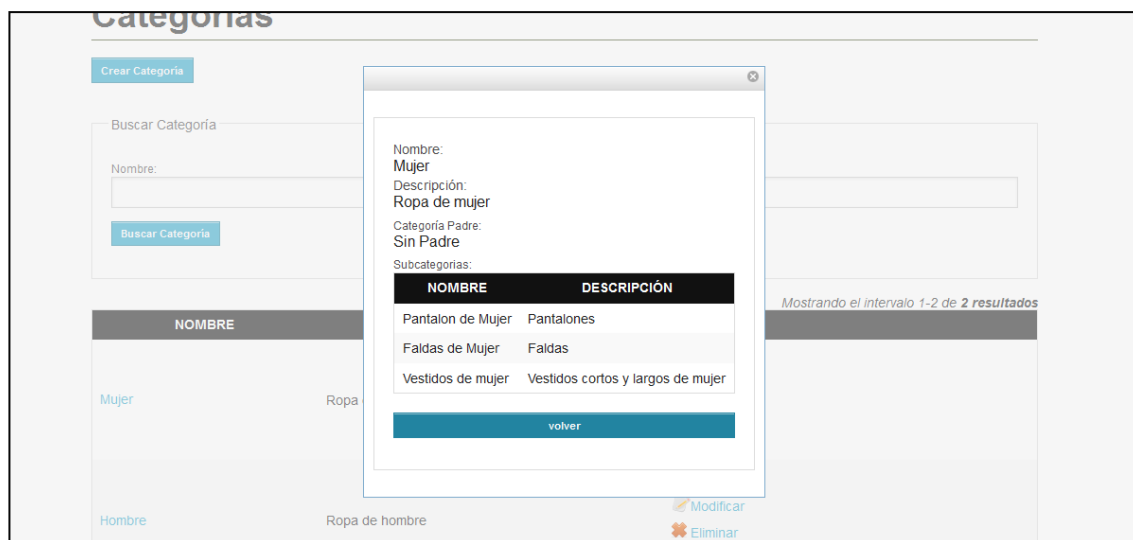


Ilustración 200 Ver detalle de una categoría

10.2.1.5.4 Modificar Categoría

Una vez creada una categoría esta puede modificarse. Para ello se pulsará el enlace *modificar*.



Ilustración 201 Modificar categoría

Al entrar en esa opción aparecerá un formulario cubierto con los datos actuales de dicha categoría. Estos campos son editables y pueden modificarse.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de modificación de la categoría y se le retornará a la sección de Categorías.

Si por el contrario desea proseguir, en este formulario de modificación de categoría se pueden introducir los siguientes detalles:

- Nombre: Nombre de la categoría.
- Descripción: Descripción de la categoría.
- Categoría padre: Categoría padre de la categoría a modificar.

The screenshot shows the 'Modificar Categoría Mujer' form in the Tunos commerce system. The form is titled 'Modificar Categoría Mujer' and is located under the 'Categorías' menu. The form contains the following fields and controls:

- Nombre:** A text input field containing the value 'Mujer'.
- Descripción:** A text area containing the value 'Ropa de mujer'.
- Categoría Padre:** A dropdown menu showing 'Sin Padre'.
- Modificar padre:** A button to update the parent category.
- Activa:** A checkbox that is checked, indicating the category is active.
- Modificar:** A button to save the changes.
- Cancelar:** A button to cancel the changes.

Ilustración 202 Modificar Categoría

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente.

Una vez rellenados los campos necesarios para la modificación de la categoría se seleccionará la acción Modificar.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Modificar.

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Modificar*. Por el contrario si se desea cancelar el proceso se seleccionará Cancelar y si se desea volver a la ventana de edición se seleccionará *volver*.

The screenshot shows the confirmation screen titled 'Confirma que desea modificar la Categoría Mujer'. The screen displays a summary of the changes to be made:

- Nombre:** Mujer
- Descripción:** Ropa de mujer.
- Categoría Padre:** Sin Padre
- Activa:** Activado

At the bottom of the summary, there are three buttons: **Modificar**, **Volver**, and **Cancelar**.

Ilustración 203 Confirmación de modificar categoría

Si se ha decidido confirmar el proceso, la categoría será modificado en el sistema y podrá visualizarse en el listado de categorías.

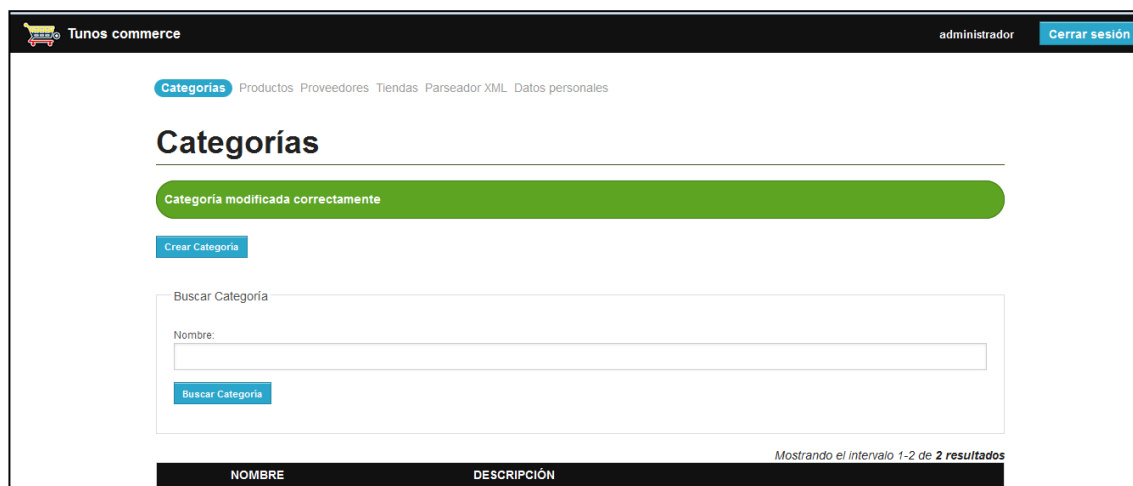


Ilustración 204 Categoría modificada correctamente

10.2.1.5.5 Eliminar Categoría

Una vez creada una categoría esta puede eliminarse. Para ello se pulsará el enlace *eliminar de la categoría deseada*.

Para eliminar una categoría esta no puede tener sub-categorías asociadas.



Ilustración 205 Opción eliminar Categoría

Al entrar en esa opción aparecerá una ventana con la información de la categoría a eliminar.



Ilustración 206 Eliminar Categoría

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de eliminación de la categoría y se le retornará a la sección de Categorías.

Si por el contrario desea proseguir, en este formulario de eliminación de categoría se seleccionará eliminar.

Si se ha decidido confirmar el proceso, la categoría será eliminada del sistema y desaparecerá de su lista de categorías.

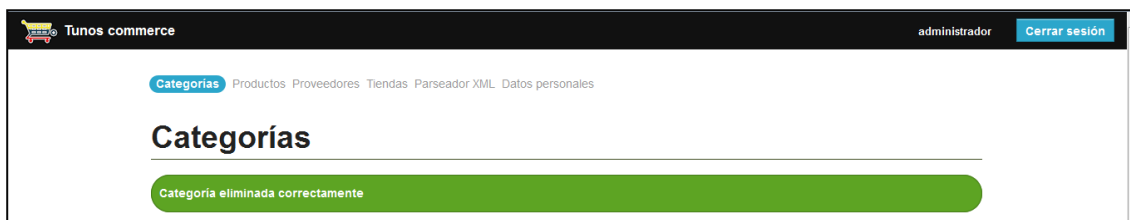


Ilustración 207 Confirmación al eliminar categoría

Por el contrario si hay algún problema se informará al usuario del error. Este es un ejemplo de un error producido si se intenta eliminar una categoría que tiene subcategorías asociadas.

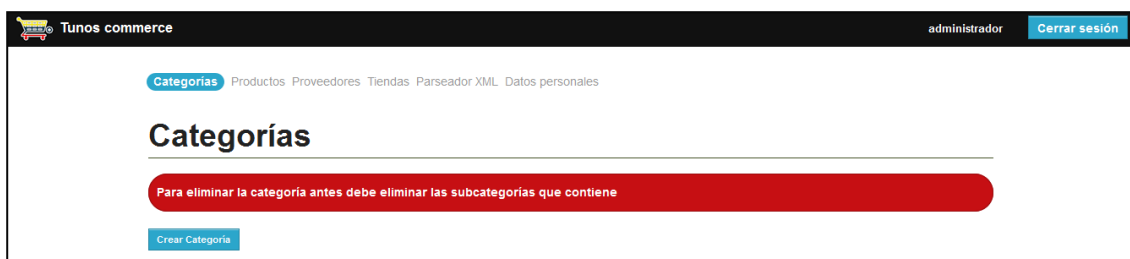


Ilustración 208 Error al eliminar categoría

10.2.1.5.6 Exportar categoría

Una vez creada una categoría y si el usuario tiene alguna tienda configurada esta puede exportarse. Para ello se pulsará el enlace *exportar de la categoría deseada*.

NOMBRE		DESCRIPCIÓN	Mostrando el intervalo 1-2 de 2 resultados	
Mujer		Ropa de mujer.	Ver detalle	
			Modificar	
			Eliminar	
			Exportar	

Ilustración 209 Exportar categoría

Al entrar en esa opción aparecerá una ventana con la información de la categoría a exportar y un combobox donde podrá seleccionar la tienda de destino.

The screenshot shows the 'Exportar Categoría Mujer' form in the Tunos commerce system. The header includes the logo and 'Tunos commerce', the user role 'administrador', and a 'Cerrar sesión' button. The breadcrumb trail is 'Categorías > Productos > Proveedores > Tiendas > Parseador XML > Datos personales'. The form title is 'Exportar Categoría Mujer'. The form fields are: 'Nombre:' with the value 'Mujer', 'Descripción:', 'Ropa de mujer:', and 'Seleccione la tienda de destino:' with a dropdown menu showing 'ShopPrestashop', 'ShopPrestashop', and 'ShopMagento'. There are 'Exportar' and 'Cancelar' buttons at the bottom right of the form.

Ilustración 210 Exportar categoría

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de exportación de la categoría y se le retornará a la sección de Categorías.

Si por el contrario desea proseguir, en este formulario de exportar categoría se seleccionará exportar.

Si se ha decidido confirmar el proceso, la categoría será exportada a la tienda a la que haya seleccionado como destino. Si todo ocurre con normalidad le deberá salir un mensaje de confirmación.

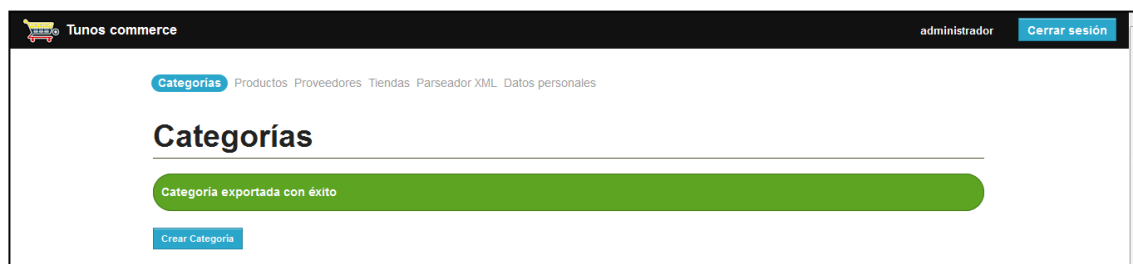


Ilustración 211 confirmación exportar categoría

10.2.1.6 Gestión de productos

El sistema permite ver un listado de los productos creados por el usuario. Se podrá acceder a la sección de Productos mediante la opción de menú productos.

The screenshot displays the 'Productos' section of the 'Tunos commerce' system. At the top, there's a navigation bar with 'Categorías', 'Productos' (selected), 'Proveedores', 'Tiendas', 'Parseador XML', and 'Datos personales'. Below this, the 'Productos' title is followed by two buttons: 'Crear un nuevo producto' and 'Generar CSV de Productos'. A search section titled 'Buscar Producto' includes a 'Nombre:' input field, a 'Proveedor:' dropdown set to 'Sin Proveedor', and a 'Buscar Producto' button. The main area shows a table of products with the following data:

NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR
iPod Nano	Nuevo diseño. Nuevas aplicaciones. Ahora disponible en 8 y 16 Go. iPod nano, más rock que nunca.	124.582	2	Apple Computer, Inc
iPod shuffle	iPod shuffle, el walkman más portátil del mundo, ahora en azul, verde, rosa y rojo.	66.0535	2	Apple Computer, Inc
MacBook Air	MacBook Air es ultra fino, ultra portátil y ultra diferente de todo el resto. Pero no se pierden kilos y centímetros en tan solo una noche. Todo esto es el resultado de un nuevo invento de normas. De un sinfín de novedades sin cable. Y de una revolución en el diseño. Con MacBook Air, la informática móvil adquiere una nueva dimensión.	1504.18	2	Apple Computer, Inc
MacBook	MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.	1170.57	2	Apple Computer, Inc
iPod touch	Interfaz multi-touch revolucionaria Pantalla panorámica color de 3,5 pulgadas Wi-Fi (802.11b/g) 8 mm de espesor Safari, YouTube, iTunes Wi-Fi Music Store, Correo, Mapas, Bolsa, El tiempo, Notas	241.639	2	

Each product row includes action links: 'Ver detalle', 'Modificar', 'Eliminar', and 'Exportar'. The footer indicates 'Mostrando el intervalo 1-5 de 16 resultados' and 'Página 1 de 4 Siguiente > Último >>'.

Ilustración 212 Sección productos

Algunos de los datos que se podrán observar sobre cada registro de producto listado serán:

- Nombre: Nombre del producto
- Descripción corta: Breve descripción del producto.
- Descripción larga: descripción más extensa del producto.

- Acciones: Enlaces a las acciones que se pueden realizar para cada producto (Ver detalle, modificar, eliminar o exportar)

Los resultados del listado se muestran paginados. Para navegar sobre las sucesivas páginas basta con seleccionar la acción Siguiente o Anterior según se deseé.

10.2.1.6.1 Buscar producto

En los listados de productos existe la posibilidad de encontrar un producto o conjunto de productos en concreto utilizando la Búsqueda de Productos.

En este caso los productos se pueden buscar por nombre , por proveedor o por ambos. Para buscar un producto por nombre se introducirá el nombre de este en el formulario de buscar y a continuación se seleccionará el botón buscar.



Ilustración 213 Buscar producto por nombre

En el caso añadir un proveedor a los parámetros a buscar deberá añadirse el proveedor deseada seleccionando en el campo Proveedor la opción Seleccionar

Una vez seleccionada esta opción se presentará un listado navegable donde podremos seleccionar el proveedor deseado. Si por el contrario se desea quitar el proveedor de los criterios de búsqueda se ha de seleccionar la opción Quitar.

[Categorías](#) [Productos](#) [Proveedores](#) [Tiendas](#) [Parseador XML](#) [Datos personales](#)

Seleccionar Proveedor

Mostrando el intervalo 1-5 de 8 resultados

COMPAÑIA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
ZARA	12222	24.05.2013	24.05.2013	+ Seleccionar
H&M	11111	24.05.2013	24.05.2013	+ Seleccionar
Pinkies	0000	24.05.2013	24.05.2013	+ Seleccionar
Mango	8765	24.05.2013	24.05.2013	+ Seleccionar
pull&bear	8765	24.05.2013	24.05.2013	+ Seleccionar

Página 1 de 2 [Siguiente](#) [Último](#)

Cancelar

Ilustración 214 Seleccionar un proveedor como parámetro de búsqueda

Una vez configurados los parámetros de búsqueda ha de seleccionarse la opción Buscar. La consulta devolverá un listado de Productos que corresponden con los criterios indicados.

10.2.1.6.2 Crear producto

Seleccionando la sección de Productos, se podrá observar una opción Nuevo producto. Al seleccionar esta opción se muestra un formulario para la creación de un nuevo producto.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de creación de un nuevo producto y se le retornará a la página de Listado y Búsqueda de Productos.

Tunos commerce administrador Cerrar sesión

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Crear producto

Nombre:

Descripción Corta:

Descripción Larga:

Número de Referencia:

Precio:

Número de Unidades:

Alto:

Ancho:

Proveedor:
Sin Proveedor
Seleccionar

Categorías:
Producto no asignado a ninguna categoría
Añadir Categorías

☐ Activo

Crear Cancelar

Ilustración 215 Crear nuevo producto

Si por el contrario desea proseguir, en este formulario de creación de un nuevo producto se pueden introducir los siguientes detalles:

- Nombre: Nombre del producto.
- Descripción corta: una breve descripción del producto
- Descripción larga: descripción más extensa del producto
- Número de referencia: número identificativo del producto
- Precio: coste en euros del producto
- Número de unidades: cantidad de stock del producto
- Alto: Medida para indicar el alto del producto
- Ancho: Medida para indicar el ancho del producto
- Proveedor: proveedor o fabricante del producto
- Categorías: clasificación del producto en una o varias categorías
- Activo: indica si el usuario está activo o desactivo en la tienda.

Para seleccionar un proveedor basta con acceder al listado de proveedores a través del enlace “Seleccionar” colocado en la opción proveedor y seleccionar el proveedor deseado.

Seleccionar Proveedor

COMPANIA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
ZARA	12222	24.05.2013	24.05.2013	Seleccionar
H&M	11111	24.05.2013	24.05.2013	Seleccionar
Pinkies	0000	24.05.2013	24.05.2013	Seleccionar
Mango	8765	24.05.2013	24.05.2013	Seleccionar
pull&bear	8765	24.05.2013	24.05.2013	Seleccionar

Página 1 de 2 [Siguiente](#) [Último](#)

[Cancelar](#)

Ilustración 216 Seleccionar proveedor

Para seleccionar las categorías a las que se desea que pertenezca un producto simplemente se accederá al listado de categorías a través del enlace “Añadir Categorías” y se seleccionarán las categorías deseadas.

Categorizar el producto Ordenador Portátil

Categorías Seleccionadas

NOMBRE	DESCRIPCIÓN	
Mujer	Mujer	Quitar
Hombre	Hombre	Quitar

Lista de Categorías disponibles

NOMBRE	DESCRIPCIÓN	
Mujer	Ropa de mujer	Seleccionar
Hombre	Ropa de hombre	Seleccionar
Complementos	Complementos	Seleccionar
Zapatos de vestir	Zapatos de Vestir	Seleccionar
Zapatillas deportivas	Zapatillas deportivas	Seleccionar

Página 1 de 2 [Siguiente](#) [Último](#)

[Volver](#)

[Aceptar](#) [Cancelar](#)

Ilustración 217 Seleccionar categorías

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente. Una vez rellenados los campos necesarios para la creación de un nuevo producto se debe seleccionar la acción Crear.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Crear.

The screenshot shows the 'Confirma crear producto Falda' screen. At the top, there's a navigation bar with 'Tunos commerce' and 'administrador' with a 'Cerrar sesión' button. Below the navigation bar, there's a breadcrumb trail: 'Categorías > Productos > Proveedores > Tiendas > Parseador XML > Datos personales'. The main heading is 'Confirma crear producto Falda'. Below this, there's a summary of the product details:

- Nombre: Falda
- Descripción Corta: Falda de mujer
- Descripción Larga: Palda de mujer corta floreada
- Número de Referencia: 558884456ERRR
- Precio: 40.0
- Número de Unidades: 50
- Alto: 10.0
- Ancho: 20.0
- Proveedor: ELLE
- Activo: Activado
- Categorías:

Below the summary, there's a table with two columns: 'NOMBRE' and 'DESCRIPCIÓN'. The table has one row with 'Mujer' in the 'NOMBRE' column and 'Ropa de mujer.' in the 'DESCRIPCIÓN' column. At the bottom right of the table, there are three buttons: 'Crear', 'Volver', and 'Cancelar'.

Ilustración 218 Confirmación de crear un nuevo producto

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Crear*. Por el contrario si se desea cancelar el proceso se seleccionará *Cancelar* y si se desea volver a la pantalla anterior de creación para cambiar algún campo se seleccionará *Volver*.

Si se ha decidido confirmar el proceso, el nuevo producto será creado en el sistema y podrá visualizarse en el listado de productos.

The screenshot shows the 'Productos' screen. At the top, there's a navigation bar with 'Tunos commerce' and 'administrador' with a 'Cerrar sesión' button. Below the navigation bar, there's a breadcrumb trail: 'Categorías > Productos > Proveedores > Tiendas > Parseador XML > Datos personales'. The main heading is 'Productos'. Below this, there's a green banner that says 'Producto creado correctamente'. Below the banner, there are two buttons: 'Crear un nuevo producto' and 'Generar CSV de Productos'. Below these buttons, there's a search section titled 'Buscar Producto'. It has a 'Nombre:' label and a text input field. To the right of the input field, there's a 'Proveedor:' label and a dropdown menu showing 'Sin Proveedor'. Below the dropdown menu, there are two buttons: 'Seleccionar' and 'Quitar'. At the bottom left of the search section, there is a 'Buscar Producto' button.

Ilustración 219 Producto creado correctamente

10.2.1.6.3 Crear CSV

Seleccionando la sección de Productos, se podrá observar una opción Generar CSV de productos. Al seleccionar esta opción se genera un archivo CSV con toda la lista de productos que será compatible con las plataformas Magento y Prestashop.

El archivo podrá ser guardado para su posterior uso.

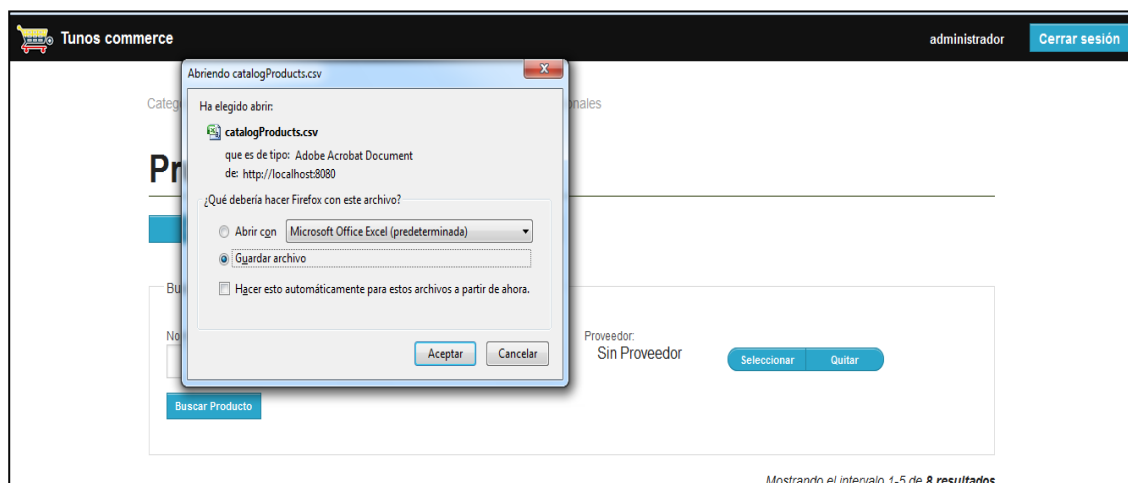


Ilustración 220 Generar archivo CSV

10.2.1.6.4 Ver detalle de un producto

El sistema permite consultar los detalles de los productos creados. Se puede acceder a los detalles de un producto pulsando sobre el enlace de ver detalle del producto que se desea visualizar.

Mostrando el intervalo 1-5 de 8 resultados

NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR	
producto1	descripcion corta	12.0	4	ZARA	Ver detalle Modificar Eliminar Exportar

Ilustración 221 Enlace ver detalle de un producto

Se mostrará una pantalla con las propiedades del producto. Los detalles del producto que se muestran son los siguientes:

- Nombre: Nombre del producto.
- Descripción corta: una breve descripción del producto
- Descripción larga: descripción más extensa del producto
- Número de referencia: número identificativo del producto
- Precio: coste en euros del producto
- Número de unidades: cantidad de stock del producto
- Alto: Medida para indicar el alto del producto
- Ancho: Medida para indicar el ancho del producto

- Proveedor: proveedor o fabricante del producto
- Categorías: clasificación del producto en una o varias categorías
- Activo: indica si el usuario está activo o desactivo en la tienda.
- Imagen QR con información del producto



Ilustración 222 Ver detalle de un producto

10.2.1.6.5 Modificar Producto

Una vez creado un producto este puede modificarse. Para ello se pulsará el enlace *modificar*.



Ilustración 223 Modificar producto

Al entrar en esa opción aparecerá un formulario cubierto con los datos actuales de dicho producto. Estos campos son editables y pueden modificarse.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de modificación de la categoría y se le retornará a la sección de Productos.

Si por el contrario desea proseguir, en este formulario de modificación de producto se pueden introducir los siguientes detalles:

- Nombre: Nombre del producto.
- Descripción corta: una breve descripción del producto
- Descripción larga: descripción más extensa del producto
- Número de referencia: número identificativo del producto
- Precio: coste en euros del producto
- Número de unidades: cantidad de stock del producto
- Alto: Medida para indicar el alto del producto
- Ancho: Medida para indicar el ancho del producto
- Proveedor: proveedor o fabricante del producto
- Categorías: clasificación del producto en una o varias categorías
- Activo: indica si el usuario está activo o desactivo en la tienda.

Tunos commerce administrador Cerrar sesión

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Modificar producto MacBook

Nombre:
MacBook

Descripción Corta:
MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.

Descripción Larga:
Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin

Número de Referencia:
123345555-MF

Precio:
1170.57

Número de Unidades:
2

Alto:
0.0

Ancho:
0.75

Proveedor:
Apple Computer, Inc.
Seleccionar

Categorías:

NOMBRE	DESCRIPCIÓN
Portátiles	El último procesador intel, un disco duro más grande, con profusión de memoria y otras novedades. Todo en sólo 2,59 cm libres de cualquier obstrucción. Las nuevas portátiles Mac cumplir rendimiento, potencia y conectividad de una computadora de escritorio. Sin la parte del escritorio.

Añadir Categorías

☒ Activo

Modificar Cancelar

Ilustración 224 Modificar producto

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente.

Una vez rellenados los campos necesarios para la modificación del producto se seleccionará la acción Modificar.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar *Modificar*.

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Modificar*. Por el contrario si se desea cancelar el proceso se seleccionará *Cancelar* y si se desea volver a la ventana de edición se seleccionará *volver*.

Tunos commerce administrador [Cerrar sesión](#)

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Confirma modificar producto MacBook

Nombre:
MacBook

Descripción Corta:
MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.

Descripción Larga:
Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin dificultad tus aplicaciones preferidas.

Número de Referencia:
Precio:
1170.57

Número de Unidades:
2

Alto:
0.0

Ancho:
0.75

Proveedor:
Apple Computer, Inc

Activo:
Desactivado

Categorías:

NOMBRE	DESCRIPCIÓN
Portátiles	El último procesador Intel, un disco duro más grande, con profusión de memoria y otras novedades. Todo en sólo 2,59 cm libres de cualquier obstrucción. Las nuevas portátiles Mac cumplir rendimiento, potencia y conectividad de una computadora de escritorio. Sin la parte del escritorio.

[Modificar](#) [Volver](#) [Cancelar](#)

Ilustración 225 Confirmación de modificar producto

Si se ha decidido confirmar el proceso, la categoría será modificado en el sistema y podrá visualizarse en el listado de productos.

Tunos commerce administrador [Cerrar sesión](#)

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Productos

Producto modificado correctamente

[Crear un nuevo producto](#) [Generar CSV de Productos](#)

Buscar Producto

Nombre:

Proveedor: Sin Proveedor [Seleccionar](#) [Quitar](#)

[Buscar Producto](#)

Mostrando el intervalo 11-15 de 24 resultados

NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR

Ilustración 226 Producto modificado correctamente

10.2.1.6.6 Eliminar Producto

Una vez creado un producto este puede eliminarse. Para ello se pulsará el enlace *eliminar del producto deseado*.

Mostrando el intervalo 11-15 de 24 resultados				
NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR
MacBook Air	MacBook Air es ultra fino, ultra portátil y ultra diferente de todo el resto. Pero no se pierden kilos y centímetros en tan solo una noche. Todo esto es el resultado de un nuevo invento de normas. De un sinfín de novedades sin cable. Y de una revolución en el diseño. Con MacBook Air, la informática móvil adquiere una nueva dimensión.	1504.18	2	Apple Computer, Inc

Ver detalle

Modificar

Eliminar

Exportar

Ilustración 227 Opción eliminar Producto

Al entrar en esa opción aparecerá una ventana con la información del producto a eliminar.

Tunos commerce

administrador

Cerrar sesión

Categorías
Productos
Proveedores
Tiendas
Parseador XML
Datos personales

Confirma que desea eliminar el producto MacBook

Nombre:
MacBook

Descripción Corta:
MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.

Descripción Larga:
Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin dificultad tus aplicaciones preferidas.

Número de Referencia:

Precio:
1170.57

Número de Unidades:
2

Alto:
0.0

Ancho:
0.75

Proveedor:
Apple Computer, Inc

Activo:
Desactivado

Categorías:

NOMBRE	DESCRIPCIÓN
Portátiles	El último procesador Intel, un disco duro más grande, con profusión de memoria y otras novedades. Todo en sólo 2,59 cm libres de cualquier obstrucción. Las nuevas portátiles Mac cumplir rendimiento, potencia y conectividad de una computadora de escritorio. Sin la parte del escritorio.

Eliminar
Cancelar

Ilustración 228 Eliminar producto

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de eliminación del producto y se le retornará a la sección de Productos.

Si por el contrario desea proseguir, en este formulario de eliminación de producto se seleccionará eliminar.

Si se ha decidido confirmar el proceso, el producto será eliminado del sistema y desaparecerá de su lista de productos.

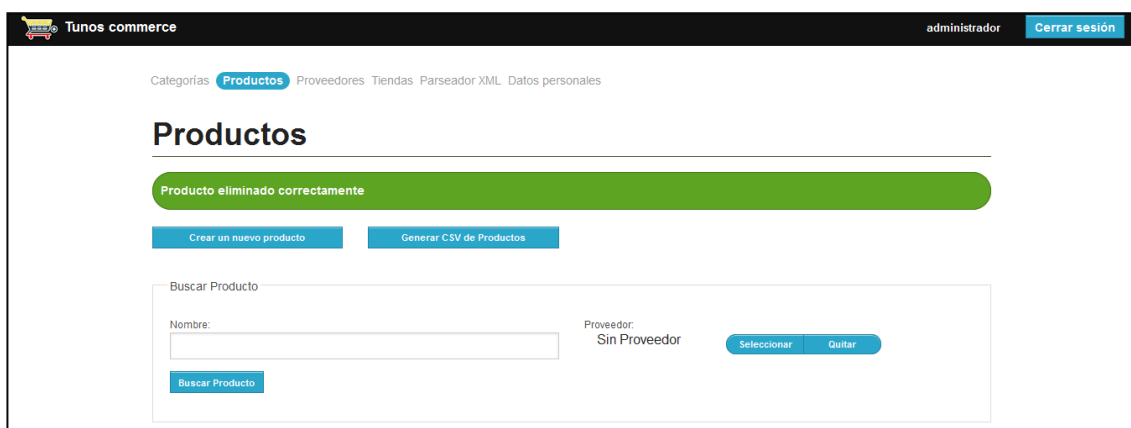


Ilustración 229 Confirmación al eliminar producto

10.2.1.6.7 Exportar producto

Una vez creado un producto y si el usuario tiene alguna tienda configurada este puede exportarse. Para ello se pulsará el enlace *exportar del producto deseado*.

Mostrando el intervalo 11-15 de 24 resultados

NOMBRE	DESCRIPCIÓN CORTA	PRECIO	NÚMERO DE UNIDADES	PROVEEDOR	
MacBook Air	MacBook Air es ultra fino, ultra portátil y ultra diferente de todo el resto. Pero no se pierden kilos y centímetros en tan solo una noche. Todo esto es el resultado de un nuevo invento de normas. De un sinfín de novedades sin cable. Y de una revolución en el diseño. Con MacBook Air, la informática móvil adquiere una nueva dimensión.	1504.18	2	Apple Computer, Inc	Ver detalle Modificar Eliminar Exportar

Ilustración 230 Exportar producto

Al entrar en esa opción aparecerá una ventana con la información del producto a exportar y un combobox donde podrá seleccionar la tienda de destino.

The screenshot shows the 'Exportar Producto MacBook' form in the Tunos commerce admin interface. The form contains the following fields and values:

- Nombre:** MacBook
- Descripción Corta:** MacBook te ofrece una gran libertad de movimientos gracias a su exterior resistente en policarbonato, a su tecnología sin cable y a su adaptador cargador sector innovador que se desconecta automáticamente si alguien se engancha en el cable.
- Descripción Larga:** Cada MacBook está equipado de un disco duro más espacioso, de una capacidad de hasta 250 Go, para almacenar tus colecciones multimedia en expansión y tus datos más preciados. El modelo MacBook de 2,4 GHz integra 2 Go de memoria en estándar. Lo ideal para realizar sin dificultad tus aplicaciones preferidas.
- Precio:** 1170.57
- Número de Unidades:** 2
- Alto:** 0.0
- Ancho:** 0.75
- Proveedor:** Apple Computer, Inc
- Activo:** Desactivado
- Seleccione la tienda de destino:** ShopPrestashop (selected)

At the bottom right of the form are two buttons: 'Exportar' and 'Cancelar'.

Ilustración 231 Exportar producto

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de exportación del producto y se le retornará a la sección de Productos.

Si por el contrario desea proseguir, en este formulario de exportar producto se seleccionará exportar.

Si se ha decidido confirmar el proceso, el producto será exportado a la tienda a la que haya seleccionado como destino. Si todo ocurre con normalidad le deberá salir un mensaje de confirmación.

The screenshot shows the 'Productos' section in the Tunos commerce admin interface. A green banner at the top indicates 'Producto exportado con éxito'. Below the banner are two buttons: 'Crear un nuevo producto' and 'Generar CSV de Productos'. Underneath is a search form with the label 'Buscar Producto' and a text input field. To the right of the input field, it says 'Proveedor: Sin Proveedor' with 'Seleccionar' and 'Quitar' buttons. At the bottom left of the search form is a 'Buscar Producto' button.

Ilustración 232 confirmación exportar producto

10.2.1.7 Gestión de proveedores

El sistema permite ver un listado de los proveedores creados por el usuario. Se podrá acceder a la sección de Proveedores mediante la opción de menú proveedores.

Proveedores

Crear nuevo proveedor

Buscar Proveedor

Compañía: CIF:

Buscar Proveedor

Mostrando el intervalo 6-10 de 11 resultados

COMPAÑÍA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
LEVIS	8721	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar
PEPE JEANS	8722	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar
Apple Computer, Inc		06.07.2013	06.07.2013	Ver detalle Modificar Eliminar Exportar
Shure Incorporated		06.07.2013	06.07.2013	Ver detalle Modificar Eliminar Exportar

<|Primero <Anterior Página 2 de 3 Siguiente> Último|>

Ilustración 233 Sección Proveedores

Algunos de los datos que se podrán observar sobre cada registro de proveedor listado serán:

- Nombre: Nombre del proveedor
- CIF: número de CIF del proveedor
- Fecha de creación: Fecha en la que se creó el proveedor
- Fecha de modificación: Fecha en la que se modificó por última vez el proveedor
- Acciones: Enlaces a las acciones que se pueden realizar para cada proveedor (Ver detalle, modificar, eliminar o exportar)

Los resultados del listado se muestran paginados. Para navegar sobre las sucesivas páginas basta con seleccionar la acción Siguiente o Anterior según se deseé.

10.2.1.7.1 Buscar proveedor

En los listados de proveedores existe la posibilidad de encontrar una producto o conjunto de en concreto utilizando la Búsqueda de Proveedores.

En este caso los proveedores se pueden buscar por nombre , por CIF o por ambos. Para buscar un producto por nombre o por CIF se introducirán los valores en el formulario de buscar y a continuación se seleccionará el botón buscar.



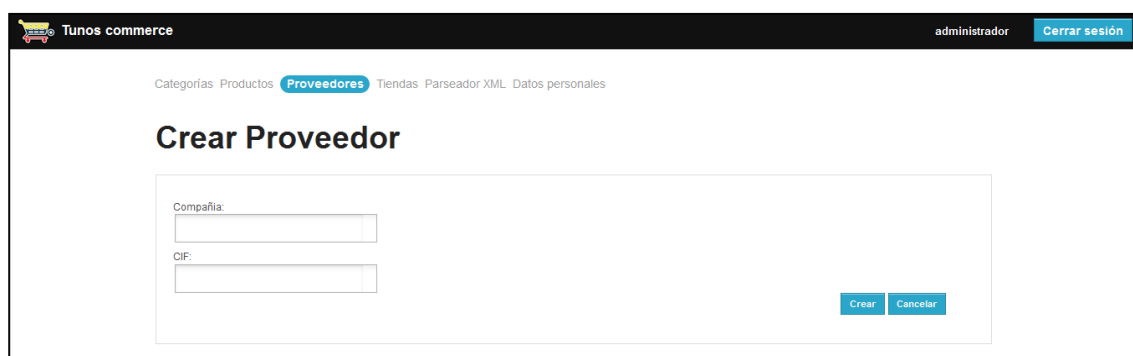
The screenshot shows the 'Proveedores' (Suppliers) section of the 'Tunos commerce' system. The page has a dark header with the logo and navigation links. The main content area has a title 'Proveedores' and a button 'Crear nuevo proveedor'. Below this is a search form titled 'Buscar Proveedor' with two input fields: 'Compañía' (containing 'ZARA') and 'CIF'. A red circle highlights the 'Buscar Proveedor' button.

Ilustración 234 Buscar proveedor por nombre

10.2.1.7.2 Crear proveedor

Seleccionando la sección de Proveedores, se podrá observar una opción Nuevo proveedor. Al seleccionar esta opción se muestra un formulario para la creación de un nuevo proveedor.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de creación de un nuevo proveedor y se le retornará a la página de Listado y Búsqueda de Proveedores.



The screenshot shows the 'Crear Proveedor' (Create Supplier) form in the 'Tunos commerce' system. The form has a title 'Crear Proveedor' and two input fields: 'Compañía' and 'CIF'. At the bottom right of the form are two buttons: 'Crear' and 'Cancelar'.

Ilustración 235 Crear nuevo proveedor

Si por el contrario desea proseguir, en este formulario de creación de un nuevo proveedor se pueden introducir los siguientes detalles:

- Compañía: Nombre de la compañía del proveedor.
- CIF: número de CIF del proveedor

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente. Una vez rellenados los campos necesarios para la creación de un nuevo proveedor se debe seleccionar la acción Crear.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Crear.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores Tiendas Parseador XML Datos personales

Confirma crear proveedor Packard Bell

Compañía:
Packard Bell
CIF:
A58818501

Crear Volver Cancelar

Ilustración 236 Confirmación de crear un nuevo proveedor

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Crear*. Por el contrario si se desea cancelar el proceso se seleccionará *Cancelar* y si se desea volver a la pantalla anterior de creación para cambiar algún campo se seleccionará *Volver*.

Si se ha decidido confirmar el proceso, el nuevo producto será creado en el sistema y podrá visualizarse en el listado de proveedores.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores Tiendas Parseador XML Datos personales

Proveedores

El proveedor se ha creado correctamente

Crear nuevo proveedor

Ilustración 237 Proveedor creado correctamente

10.2.1.7.3 Ver detalle de un proveedor

El sistema permite consultar los detalles de los proveedores creados. Se puede acceder a los detalles de un proveedor pulsando sobre el enlace de ver detalle del proveedor que se desea visualizar.

Packard Bell	A58818501	06.07.2013	06.07.2013	Ver detalle Modificar Eliminar Exportar

Ilustración 238 Ver detalla de proveedor

Se mostrará una pantalla con las propiedades del producto. Los detalles del producto que se muestran son los siguientes:

- Compañía: Nombre de la compañía del proveedor.
- CIF: número de CIF del proveedor

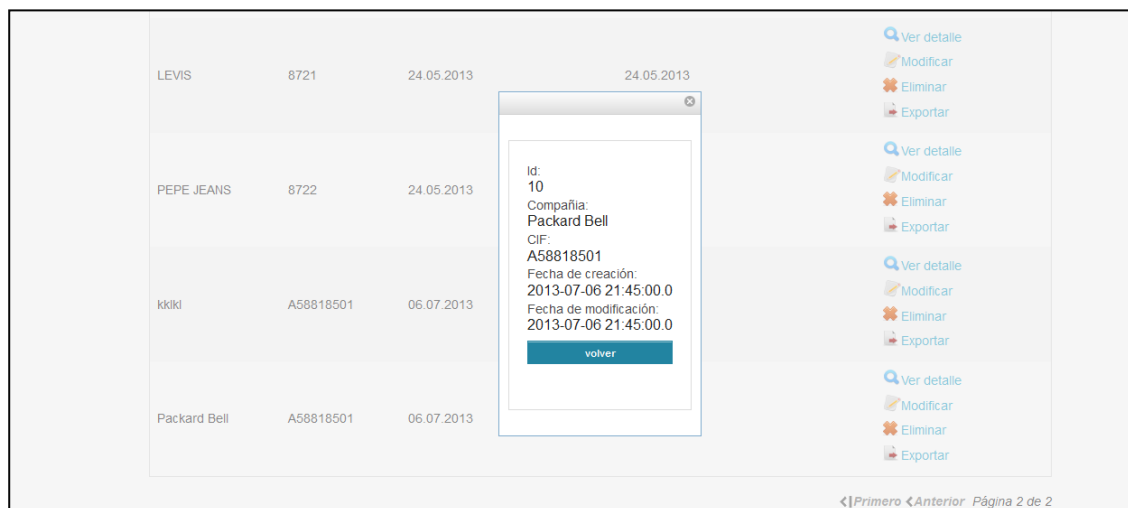


Ilustración 239 Ver detalle de un proveedor

10.2.1.7.4 Modificar Proveedor

Una vez creado un producto este puede modificarse. Para ello se pulsará el enlace *modificar*.



Ilustración 240 Modificar proveedor

Al entrar en esa opción aparecerá un formulario cubierto con los datos actuales de dicho proveedor. Estos campos son editables y pueden modificarse.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de modificación de la categoría y se le retornará a la sección de Proveedores.

Si por el contrario desea proseguir, en este formulario de modificación de proveedor se pueden introducir los siguientes detalles:

- Compañía: Nombre de la compañía del proveedor.
- CIF: número de CIF del proveedor

The screenshot shows the 'Modificar Proveedor ZARA' page. At the top, there's a navigation bar with 'Tunos commerce' logo, 'administrador' role, and a 'Cerrar sesión' button. Below the navigation bar, there's a breadcrumb trail: 'Categorías Productos Proveedores Tiendas Parseador XML Datos personales'. The main heading is 'Modificar Proveedor ZARA'. The form contains two input fields: 'Compañía:' with the value 'ZARA' and 'CIF:' with the value '12222'. At the bottom right of the form, there are two buttons: 'Modificar' and 'Cancelar'.

Ilustración 241 Modificar proveedor

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente.

Una vez rellenados los campos necesarios para la modificación del proveedor se seleccionará la acción Modificar.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Modificar.

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Modificar*. Por el contrario si se desea cancelar el proceso se seleccionará Cancelar y si se desea volver a la ventana de edición se seleccionará *volver*.

The screenshot shows the 'Confirma que desea modificar el proveedor' page. It has the same navigation bar and breadcrumb trail as the previous page. The main heading is 'Confirma que desea modificar el proveedor'. The form displays a summary of the data to be modified: 'Compañía: ZARA' and 'CIF: 12222'. At the bottom right, there are three buttons: 'Modificar', 'Volver', and 'Cancelar'.

Ilustración 242 Confirmación de modificar proveedor

Si se ha decidido confirmar el proceso, la categoría será modificada en el sistema y podrá visualizarse en el listado de productos.

The screenshot shows the 'Proveedores' page. It has the same navigation bar and breadcrumb trail. The main heading is 'Proveedores'. Below the heading, there is a green success message: 'Proveedor modificado correctamente'.

Ilustración 243 Mensaje de proveedor modificado correctamente

10.2.1.7.5 Eliminar Proveedor

Una vez creado un proveedor este puede eliminarse. Para ello se pulsará el enlace *eliminar del proveedor deseado*.

Mostrando el intervalo 1-5 de 10 resultados

COMPañIA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
ZARA	12222	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar

Ilustración 244 Opción eliminar Proveedor

Al entrar en esa opción aparecerá una ventana con la información del proveedor a eliminar.

Tunos commerce administrador Cerrar sesión

Categorías Productos **Proveedores** Tiendas Parseador XML Datos personales

Confirma que desea eliminar el proveedor

Compañía:
ZARA
CIF:
12222

Eliminar Cancelar

Ilustración 245 Eliminar proveedor

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de eliminación del proveedor y se le retornará a la sección de Proveedores.

Si por el contrario desea proseguir, en este formulario de eliminación del proveedor se seleccionará eliminar.

Si se ha decidido confirmar el proceso, el proveedor será eliminado del sistema y desaparecerá de su lista de proveedores

Tunos commerce administrador Cerrar sesión

Categorías **Productos** Proveedores Tiendas Parseador XML Datos personales

Productos

Producto eliminado correctamente

Crear un nuevo producto Generar CSV de Productos

Buscar Producto

Nombre:

Proveedor: Sin Proveedor

Buscar Producto

Seleccionar Quitar

Ilustración 246 Confirmación al eliminar producto

10.2.1.7.6 Exportar proveedor

Una vez creado un proveedor y si el usuario tiene alguna tienda configurada este puede exportarse. Para ello se pulsará el enlace *exportar del proveedor deseado*.

Mostrando el intervalo 1-5 de 10 resultados

COMPañIA	CIF	FECHA DE CREACIÓN	FECHA DE MODIFICACIÓN	
ZARA	12222	24.05.2013	24.05.2013	Ver detalle Modificar Eliminar Exportar

Ilustración 247 Exportar proveedor

Al entrar en esa opción aparecerá una ventana con la información del proveedor a exportar y un combobox donde podrá seleccionar la tienda de destino.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores Tiendas Parseador XML Datos personales

Exportar Proveedor ZARA

Compañía:
ZARA
CIF:
12222

Seleccione la tienda de destino:

ShopPrestashop
ShopPrestashop
ShopMagento

Exportar Cancelar

Ilustración 248 Exportar proveedor

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de exportación del proveedor y se le retornará a la sección de Proveedores.

Si por el contrario desea proseguir, en este formulario de exportar proveedor se seleccionará exportar.

Si se ha decidido confirmar el proceso, el proveedor será exportado a la tienda a la que haya seleccionado como destino. Si todo ocurre con normalidad le deberá salir un mensaje de confirmación.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores Tiendas Parseador XML Datos personales

Proveedores

Proveedor exportado con éxito

Ilustración 249 confirmación exportar proveedor

10.2.1.8 Gestión de tiendas

El sistema permite ver un listado de las tiendas creadas por el usuario. Se podrá acceder a la sección de tiendas mediante la opción de menú Tiendas.

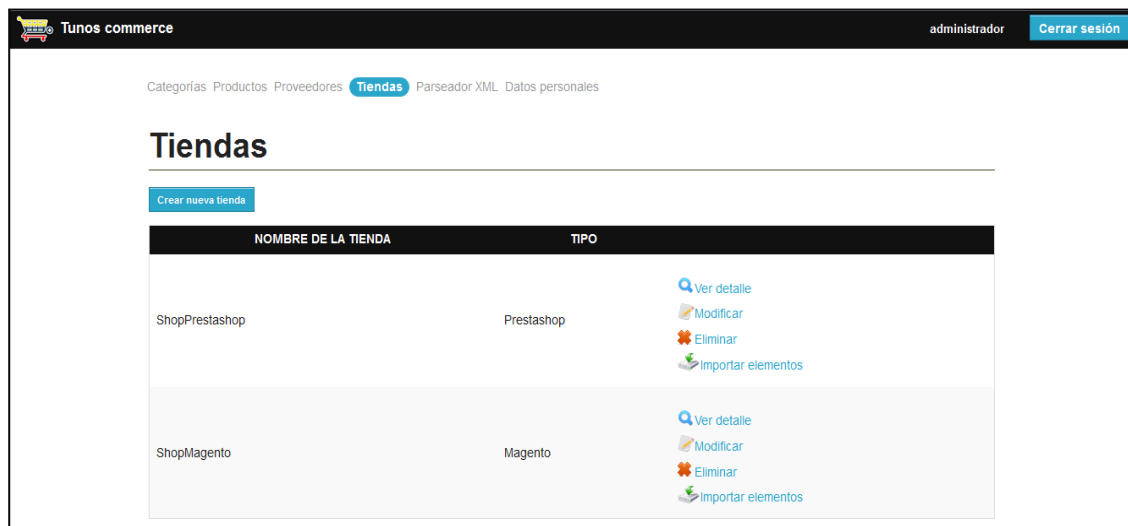


Ilustración 250 Sección tiendas

Algunos de los datos que se podrán observar sobre cada registro de tienda listado serán:

- Nombre: Nombre de la tienda
- Tipo: Indica de qué tipo de tienda se trata, en este caso sólo hay dos tipos de tienda posibles, Prestashop y Magento.

10.2.1.8.1 Crear tienda

Seleccionando la sección de Tiendas, se podrá observar una opción Nuevo tienda. Al seleccionar esta opción se muestra un formulario para la creación de una nueva tienda.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de creación de una nueva tienda y se le retornará a la página de Listado de tiendas.

Crear nueva tienda

Nombre de la tienda:

Tipo:
Prestashop

Nombre de la base de datos:

Host de la BBDD:

Número de puerto de la BBDD:

Usuario de la BBDD:

Password de la BBDD:

Repita el Password de la BBDD:

Crear Cancelar

Ilustración 251 Crear nueva tienda

Si por el contrario desea proseguir, en este formulario de creación de se pueden introducir los siguientes detalles:

- Nombre de la tienda: Nombre que se quiere asignar a la tienda.
- Tipo de tienda: sí es prestashop o Magento
- Nombre de la base de datos: nombre de la base de datos de la tienda que se quiere configurar
- Host de la base de datos donde está alojada la tienda
- Número de puerto en el que está la base de datos que se está configurando
- Usuario de acceso de la base de datos
- Password de la base de datos
- Confirmación del password de la base de datos

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente. Una vez rellenados los campos necesarios para la creación de una nueva tienda se debe seleccionar la acción Crear.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Crear.

Tunos commerce administrador [Cerrar sesión](#)

[Categorías](#) [Productos](#) [Proveedores](#) **Tiendas** [Parseador XML](#) [Datos personales](#)

Confirma que desea crear la tienda Tienda prestashop

Nombre de la tienda:
Tienda prestashop

Tipo:
Prestashop

Nombre de la base de datos:
prestashopdb

Host de la BBDD:
156.65.89.56

Número de puerto de la BBDD:
3306

Usuario de la BBDD:
root

[Crear](#) [Volver](#) [Cancelar](#)

Ilustración 252 Confirmación de crear una nueva tienda

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Crear*. Por el contrario si se desea cancelar el proceso se seleccionará *Cancelar* y si se desea volver a la pantalla anterior de creación para cambiar algún campo se seleccionará *Volver*.

Si se ha decidido confirmar el proceso, la nueva tienda será creada en el sistema y podrá visualizarse en el listado de tiendas.

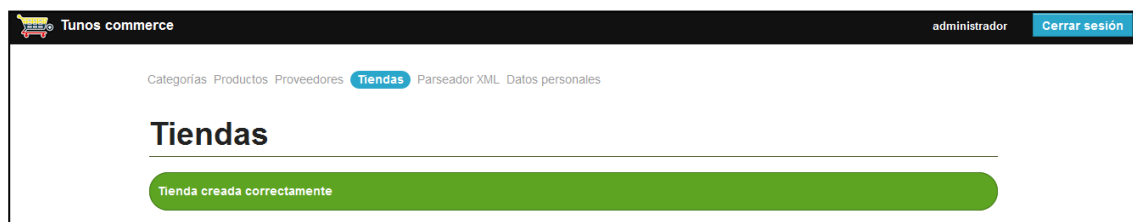


Ilustración 253 Tienda creada correctamente

10.2.1.8.2 Ver detalle de una tienda

El sistema permite consultar los detalles de las tiendas ya creadas. Se puede acceder a los detalles de una tienda pulsando sobre el enlace de ver detalle de la tienda que se desea visualizar.

NOMBRE DE LA TIENDA	TIPO	
ShopPrestashop	Prestashop	Ver detalle Modificar Eliminar Importar elementos

Ilustración 254 Ver detalla de tienda

Se mostrará una pantalla con las propiedades de la tienda. Los detalles de la tienda que se muestran son los siguientes:

- Nombre de la tienda: Nombre que se quiere asignar a la tienda.
- Tipo de tienda: sí es prestashop o Magento
- Nombre de la base de datos: nombre de la base de datos de la tienda que se quiere configurar
- Host de la base de datos donde está alojada la tienda
- Número de puerto en el que está la base de datos que se está configurando
- Usuario de acceso de la base de datos

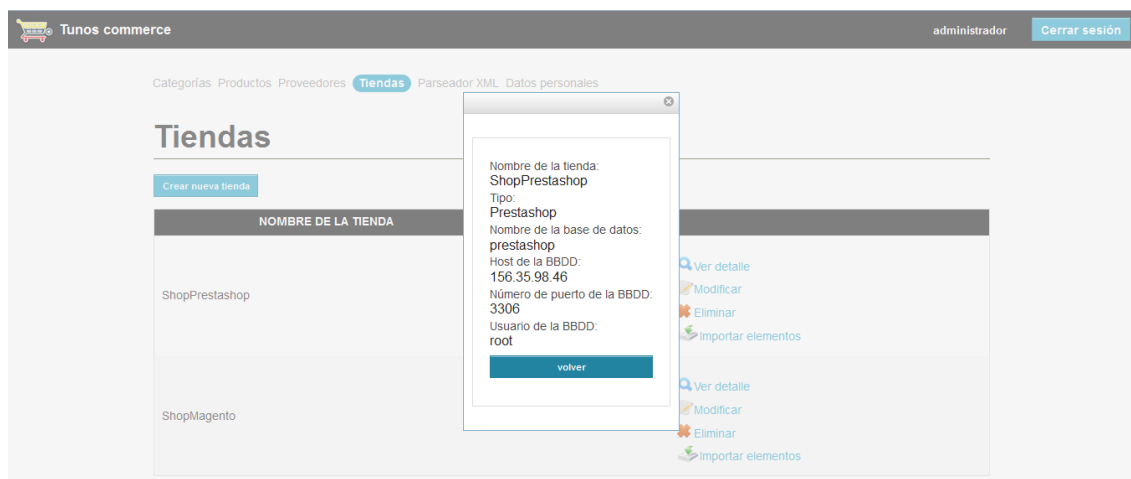


Ilustración 255 Ver detalle de una tienda

10.2.1.8.3 Modificar tienda

Una vez creada una tienda esta puede modificarse. Para ello se pulsará el enlace *modificar*.

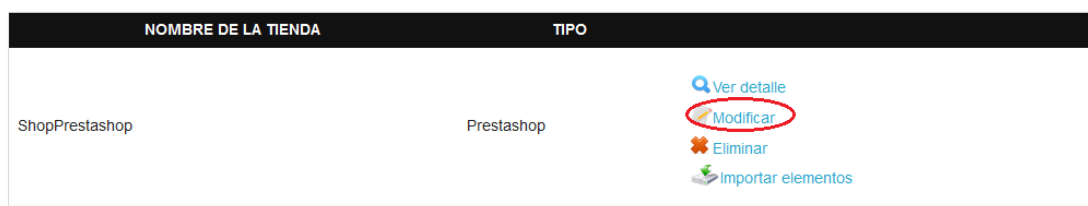


Ilustración 256 Modificar tienda

Al entrar en esa opción aparecerá un formulario cubierto con los datos actuales de dicha tienda. Estos campos son editables y pueden modificarse.

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de modificación de la tienda y se le retornará a la sección de Proveedores.

Si por el contrario desea proseguir, en este formulario de modificación de la tienda se pueden introducir los siguientes detalles:

- Nombre de la tienda: Nombre que se quiere asignar a la tienda.

- Tipo de tienda: si es prestashop o Magento
- Nombre de la base de datos: nombre de la base de datos de la tienda que se quiere configurar
- Host de la base de datos donde está alojada la tienda
- Número de puerto en el que está la base de datos que se está configurando
- Usuario de acceso de la base de datos
- Password de la base de datos
- Confirmación del password de la base de datos

The screenshot shows the 'Modificar tienda ShopPrestashop' form within the Tunos commerce administration interface. The interface has a dark header with the logo 'Tunos commerce' on the left, the user role 'administrador' in the center, and a 'Cerrar sesión' button on the right. Below the header, there is a navigation bar with links: 'Categorías', 'Productos', 'Proveedores', 'Tiendas' (highlighted in blue), 'Parseador XML', and 'Datos personales'. The main content area is titled 'Modificar tienda ShopPrestashop'. It contains a form with the following fields: 'Nombre de la tienda:' with the value 'ShopPrestashop'; 'Tipo:' with the value 'Prestashop'; 'Host de la BBDD:' with the value '156.35.98.46'; 'Número de puerto de la BBDD:' with the value '3306'; 'Usuario de la BBDD:' with the value 'root'; 'Password de la BBDD:' with masked characters '••••'; and 'Repita el Password de la BBDD:' with masked characters '••••'. At the bottom right of the form, there are two buttons: 'Modificar' and 'Cancelar'.

Ilustración 257 Modificar tienda

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicándole el campo correspondiente.

Una vez rellenados los campos necesarios para la modificación de la tienda se seleccionará la acción Modificar.

Si ha habido algún problema al validar los campos introducidos en el formulario el sistema generará un aviso y podrán ser modificados. Una vez modificados se vuelve a seleccionar Modificar.

En una nueva pantalla se mostrará un resumen de los campos que serán guardados. Si se está conforme se puede confirmar la opción seleccionando *Modificar*. Por el contrario si se desea cancelar el proceso se seleccionará Cancelar y si se desea volver a la ventana de edición se seleccionará *volver*.

The screenshot shows a confirmation dialog box titled "Confirma que desea modificar la tienda ShopPrestashop". The dialog contains the following information:

- Nombre de la tienda: ShopPrestashop
- Tipo: Prestashop
- Host de la BBDD: 156.35.98.46
- Número de puerto de la BBDD: 3306
- Usuario de la BBDD: root

At the bottom right of the dialog are three buttons: "Modificar", "Volver", and "Cancelar". The top navigation bar includes "Tunos commerce", "administrador", and "Cerrar sesión". The breadcrumb trail is "Categorías > Productos > Proveedores > Tiendas > Parseador XML > Datos personales".

Ilustración 258 Confirmación de modificar tienda

Si se ha decidido confirmar el proceso, la tienda será modificada en el sistema y podrá visualizarse en el listado de tiendas.

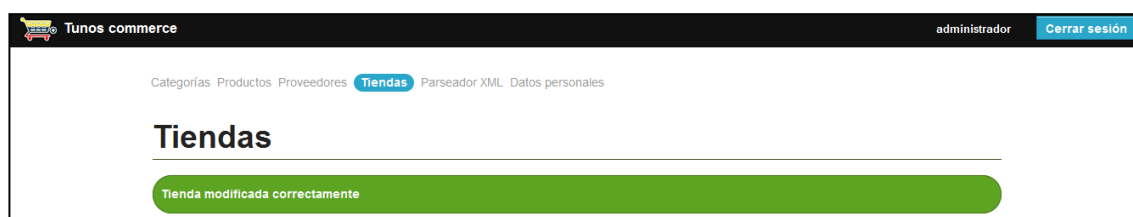


Ilustración 259 Mensaje de tienda modificada correctamente

10.2.1.8.4 Eliminar Tienda

Una vez creada una tienda esta puede eliminarse. Para ello se pulsará el enlace *eliminar de la tienda deseada*.

NOMBRE DE LA TIENDA	TIPO	
ShopPrestashop	Prestashop	Ver detalle Modificar Eliminar Importar elementos

Ilustración 260 Opción eliminar Tienda

Al entrar en esa opción aparecerá una ventana con la información de la tienda a eliminar.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores **Tiendas** Parseador XML Datos personales

Confirma que desea eliminar la tienda Tienda prestashop

Nombre de la tienda:
Tienda prestashop

Tipo:
Prestashop

Host de la BBDD:
156.65.89.56

Número de puerto de la BBDD:
3306

Usuario de la BBDD:
root

Eliminar Volver

Ilustración 261 Eliminar tienda

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de eliminación de la tienda y se le retornará a la sección de Tiendas.

Si por el contrario desea proseguir, en este formulario de eliminación de la tienda se seleccionará eliminar.

Si se ha decidido confirmar el proceso, la tienda será eliminada del sistema y desaparecerá de su lista de tiendas.

Tunos commerce administrador Cerrar sesión

Categorías Productos Proveedores **Tiendas** Parseador XML Datos personales

Tiendas

Tienda eliminada correctamente

Ilustración 262 Confirmación al eliminar tienda

10.2.1.8.5 Importar elementos

Una vez configurada la tienda pueden importarse los elementos de esta (categorías, productos y proveedores). No sólo se importarán de la tienda las propias entidades sino que también se importarán las relaciones entre ellas.

NOMBRE DE LA TIENDA	TIPO	
ShopPrestashop	Prestashop	Ver detalle Modificar Eliminar Importar elementos

Ilustración 263 Importar elementos

Al entrar en esa opción aparecerá una ventana en la que se podrá elegir que elementos de la tienda se desean importar.

Ilustración 264 Importar elementos II

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de importación y se le retornará a la sección de Tiendas.

Si por el contrario desea proseguir, en este formulario de importar elementos se seleccionarán los elementos que se desea importar y se seleccionará el botón importar. Si todo transcurre con normalidad deberá salir un mensaje de confirmación.

Ilustración 265 confirmación importar elementos

10.2.1.9 Parseador XML

El sistema permite importar elementos (categorías, productos o proveedores) a través de XML. Este XML tiene que tener un formato exactamente igual al que genera nuestro GMF.

Se podrá acceder a la sección del parseador mediante la opción de menú Parseador XML.

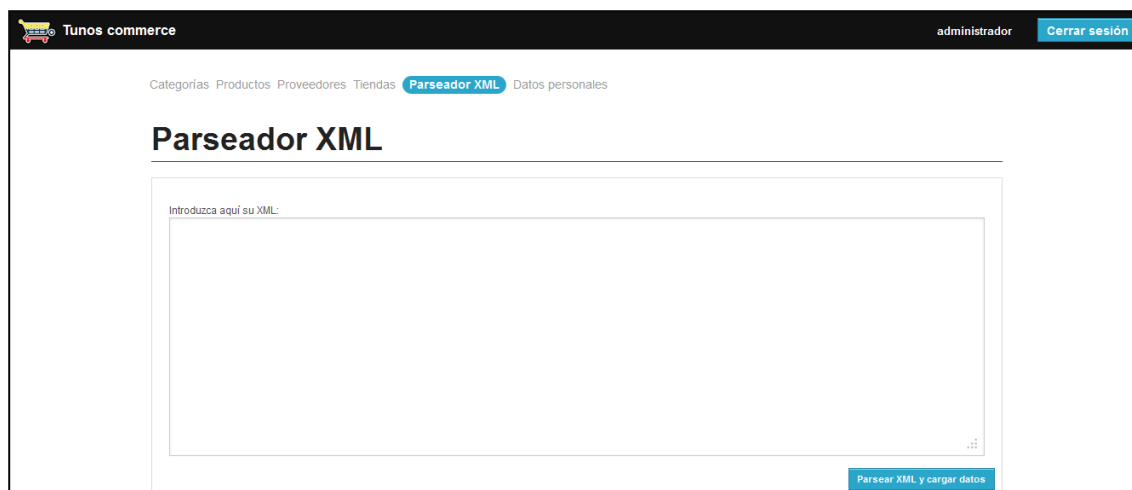


Ilustración 266 Parseador XML

Para importar los elementos de un XML simplemente basta con incorporar el XML en el campo correspondiente del formulario y seleccionar la opción *Parsear XML y cargar datos*

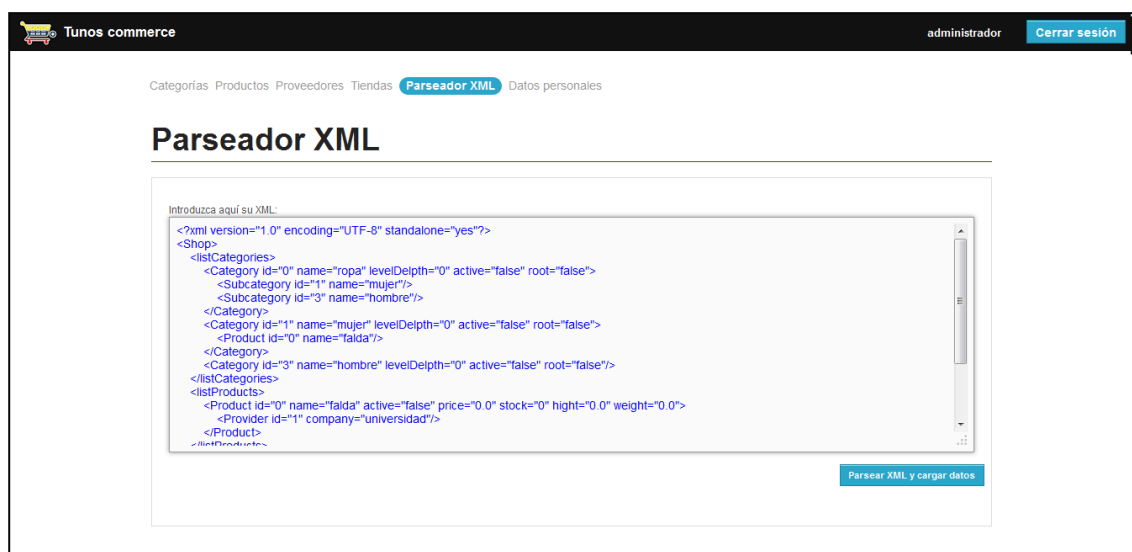


Ilustración ¡Error! Marcador no definido. Parseador XML II

Si todo ha ido con normalidad aparecerá un mensaje indicando que el proceso ha sido correcto.

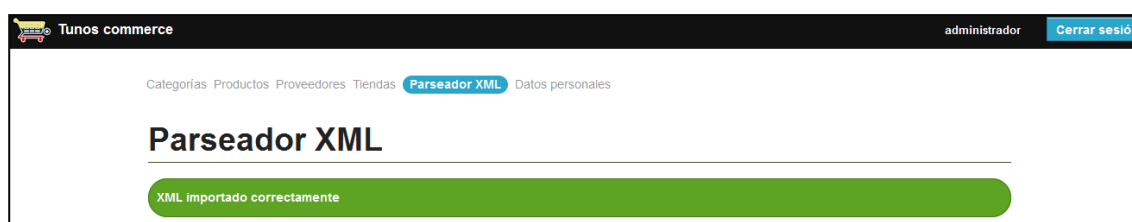


Ilustración ¡Error! Marcador no definido. Parseador importando correctamente XML

Por el contrario si el XML no tiene un formato adecuado o se produce algún error surgirá el siguiente mensaje informando al usuario.

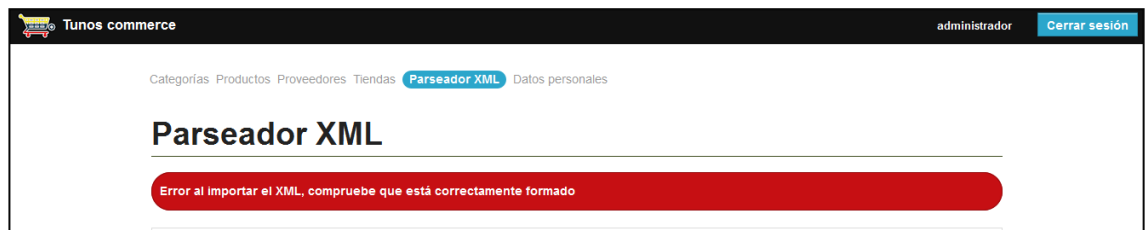


Ilustración ¡Error! Marcador no definido. Error en el XML

10.2.1.10 Datos personales

El sistema permite ver y editar los propios datos personales del usuario logueado. Se podrá acceder a la sección de datos personales mediante la opción de menú Datos personales.



Ilustración ¡Error! Marcador no definido. Datos personales del usuario

10.2.1.10.1 Modificar datos personales

Para acceder a esta opción se seleccionará la opción modificar del apartado datos personales.

Al entrar en esa opción aparecerá un formulario cubierto con los datos actuales de dicho usuario. Estos campos son editables y pueden modificarse.

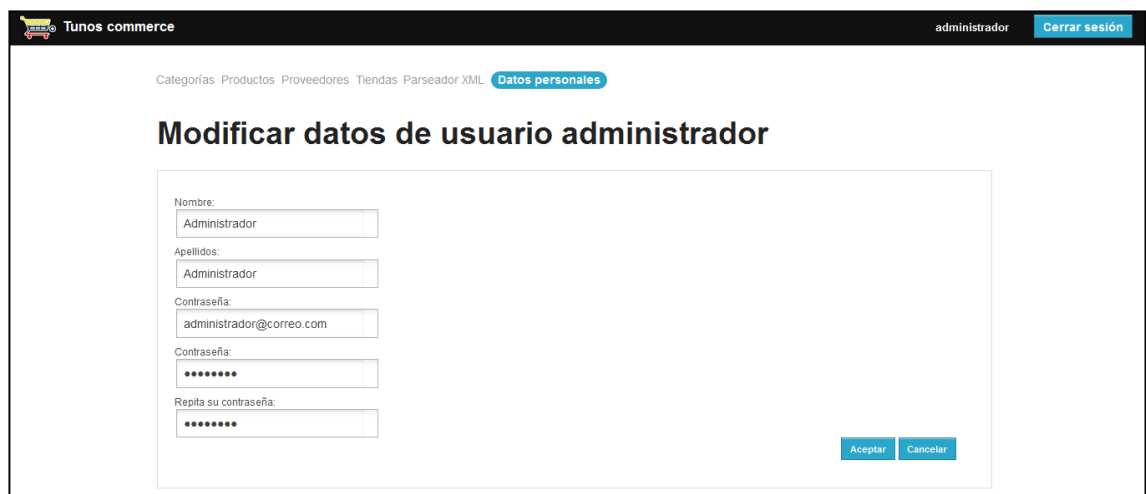


ilustración ¡Error! Marcador no definido. Modificar datos personales

Si se desea cancelar esta acción en cualquier momento se puede seleccionar la opción Cancelar. Esta acción cancelará el proceso de modificación del usuario y se retornará a la página de datos personales.

Si por el contrario se desea proseguir, en este formulario de modificación de usuario se pueden introducir los siguientes detalles:

- Nombre: Nombre del usuario
- Apellidos: Apellidos del usuario
- Email: Correo electrónico del usuario donde se enviarán los avisos del sistema. Ha de ser un correo electrónico con formato válido del tipo usuario@domino.foo
- Contraseña: Contraseña que utilizará el usuario junto su nombre de usuario para acceder al sistema. No se mostrarán los caracteres al escribir la contraseña, únicamente una representación gráfica mediante un punto por cada carácter introducido.
- Repetir contraseña: Con el fin de validar la contraseña ha de volver a escribirse la contraseña utilizada en el campo Contraseña. No se mostrarán los caracteres al escribir la contraseña, únicamente una representación gráfica mediante un punto por cada carácter introducido.

Si algún campo no estuviera correctamente cubierto el sistema mostrará una alerta indicando el campo correspondiente. Una vez rellenados los campos necesarios para la modificación del usuario se seleccionará la acción Modificar.

Si todo ha ido correctamente se mostrará un mensaje indicando que la operación se ha realizado con éxito.

The screenshot shows the 'Datos personales de administrador' page. At the top, there's a navigation bar with 'Tunos commerce' and 'administrador' with a 'Cerrar ses' button. Below the navigation bar, there's a breadcrumb trail: 'Categorías Productos Proveedores Tiendas Parseador XML Datos personales'. The main heading is 'Datos personales de administrador'. A green bar indicates 'Datos de usuario modificados correctamente'. Below this, a box contains the user details: 'Nombre: Administrador', 'Apellidos: Administrador', 'E-mail: administrador@correo.com', and 'Contraseña: *****'. A 'Modificar' button is located at the bottom right of the details box.

Ilustración 267 Usuario modificado correctamente

10.2.2 Manual de usuario GMF

El GMF está formado de una única pantalla, en ella se distinguen principalmente dos elementos:

1. **Ventana de dibujo:** zona sobre la cual se ubicarán las diferentes figuras UML.

2. **Paleta de elementos:** paleta de cada uno de los elementos que tienen representación gráfica en la ventana de dibujo.

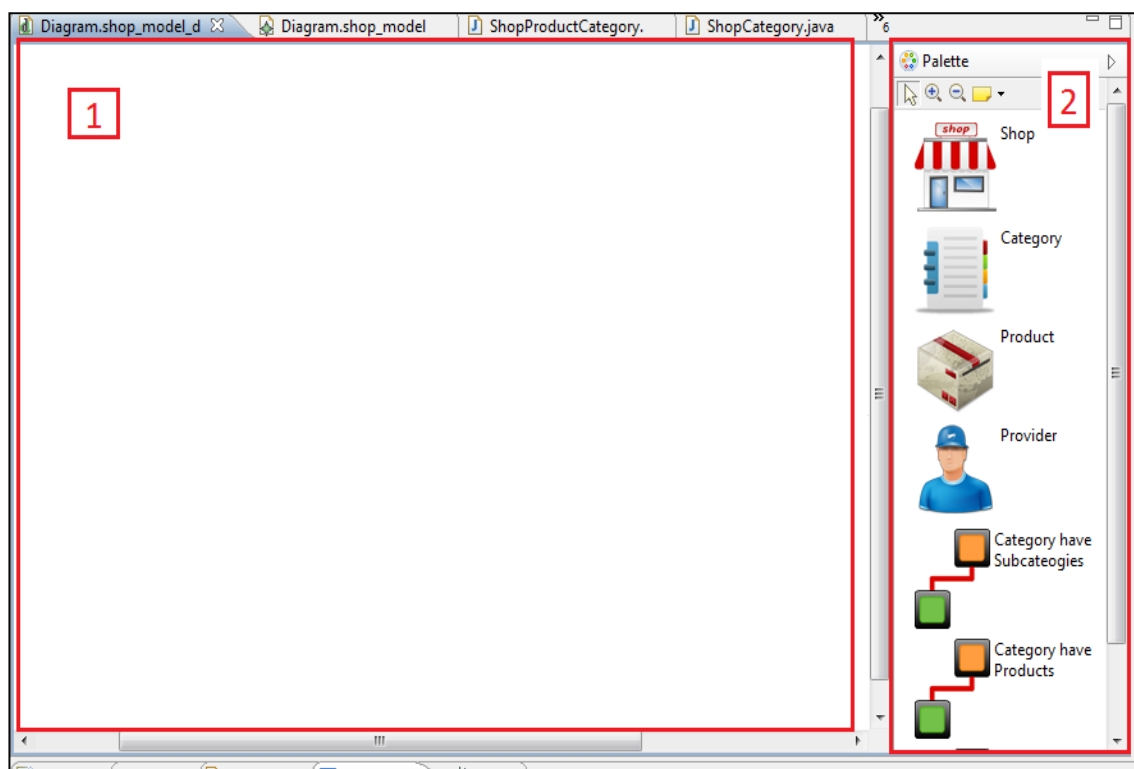
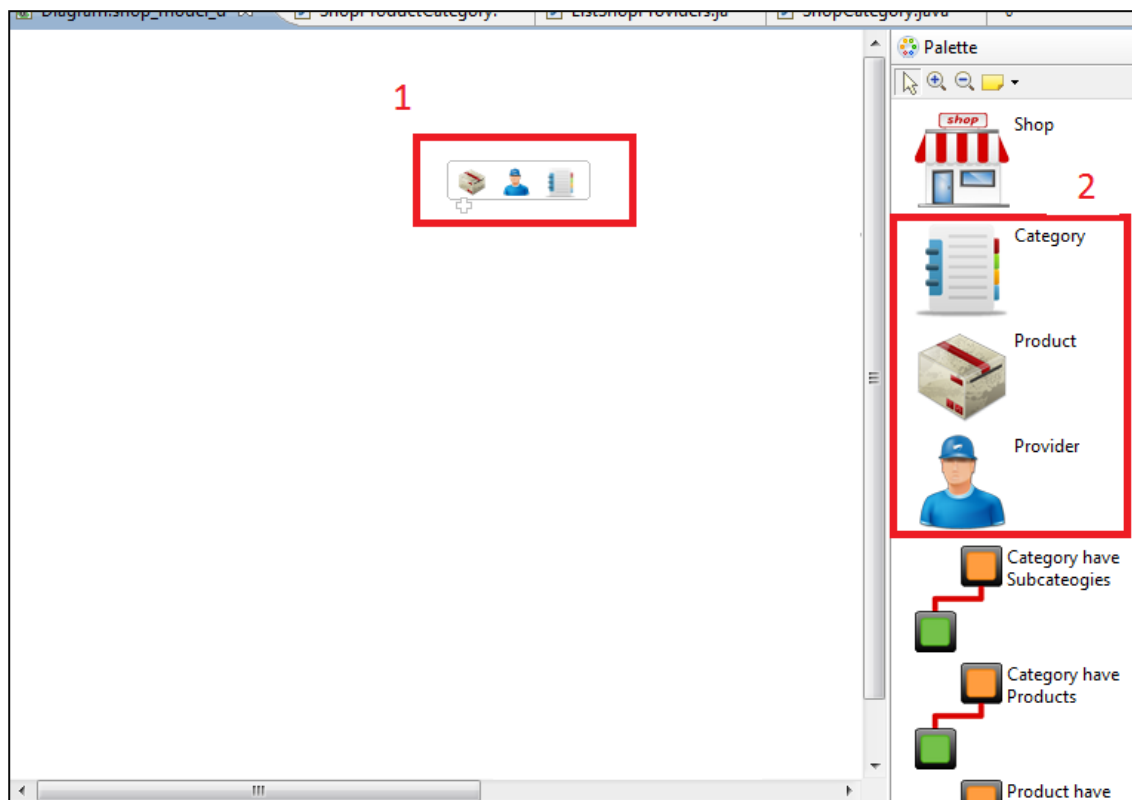


Ilustración 268 Partes que forman el GMF.

10.2.2.1 Creación de elementos

Para crear elementos en los diagramas existen dos opciones:

1. Presionar con el ratón en un elemento de la paleta (categoría, producto o proveedor) colocado a la derecha del editor. A continuación se pulsará con el ratón en la posición de la ventana de dibujo sobre la que se quiere ubicar la figura.
2. Alternativamente se puede dejar el cursor del ratón sobre la ventana durante unos dos segundos y aparecerá un menú flotante con un icono para cada elemento que se puede crear en la posición en la que se encuentra el cursor.



Cada una de las figuras tiene etiquetas editables, aparecerá un campo de texto para que introduzca su valor mediante el teclado.

Category	
- Id	0
- Name	<name>
- Level Delpth	0
- Active	false
- Root	false

Ilustración 269 Introducir valores a las figuras

Cada una de los elementos de las figuras tiene los siguientes campos:

Categoría		
Propiedad	Descripción	Tipo
idCategory	Identificador de la categoría	Valor numérico

name	Nombre de la categoría	Cadena de texto
level_delpth	Nivel de profundidad de la categoría	Valor entero
active	Indica si la categoría está activa o no	Valor booleano • true: activada • false: desactivada
Root_category	Indica si la categoría es una categoría base o no	Valor booleano • true: categoría base • false: no es una categoría base

Producto		
Propiedad	Descripción	Tipo
Id_Product	Número identificador del producto	Valor numérico
Name	Nombre del producto	Cadena de texto
Active	Nivel de profundidad de la categoría	Valor booleano • true: activada • false: desactivada
Short_description	Descripción breve del producto	Cadena de texto
Long_description	Descripción más extensa de un producto	Cadena de texto
Stock	Indica el número de unidades en stock del producto	Valor numérico
Hight	Valor para indicar las medidas del producto, en este caso la altura	Valor numérico con decimales
weight	Valor para indicar las medidas del producto, en este caso el ancho	Valor numérico con decimales
Image	Ruta en la que se encuentra la ruta de la imagen	Cadena de texto

Proveedor		
Propiedad	Descripción	Tipo
IdProvider	Número identificador del proveedor	Valor numérico
NIF	NIF del proveedor	Cadena de texto
Company	Compañía a la que pertenece el prveedor	Cadena de texto

10.2.2.2 Creación de relaciones

Para crear una relación entre dos elementos se seleccionará en la paleta el botón de la relación que desee crear. En este caso existen tres tipos de relaciones:

- Relacionar categorías entre sí
- Relacionar categorías con uno o varios productos
- Relacionar un producto con un único proveedor

Una vez seleccionada la herramienta se presionará sobre el primer elemento a unir mediante la relación, y, sin soltar, se arrastrará el cursor sobre el segundo elemento a unir, y soltar el botón del ratón.

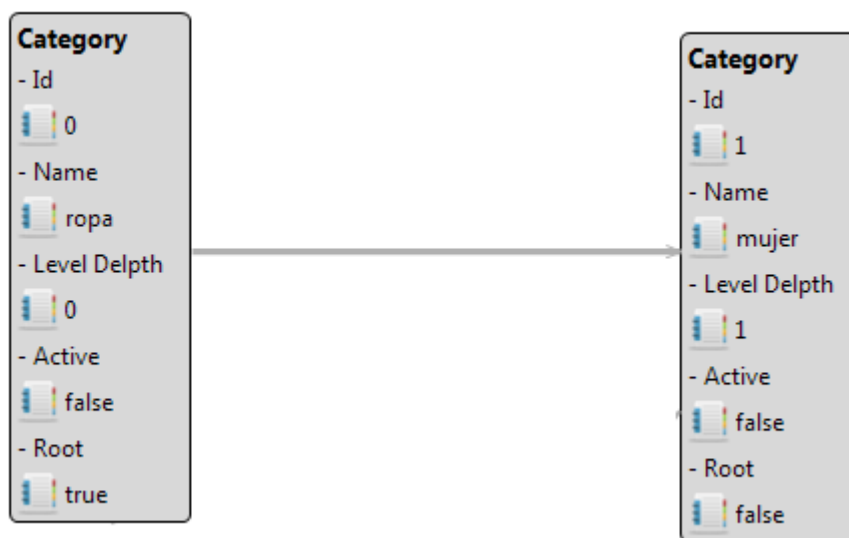


Ilustración 270 Relacionar dos elementos entre si

10.2.2.3 Generación de XML

Una vez que el diagrama está terminado se podrá generar un XML que recogerá toda la información recogida de las clases y relaciones creadas.

Para ello se presionará sobre el botón ejecutar de la aplicación:



Ilustración 271 Botón ejecutar

Una vez hecho esto se creará un xml en la carpeta del proyecto “src-gen”. El XML tendrá un aspecto similar a este:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Shop>
  <listCategories>
    <Category id="0" name="ropa" levelDelpth="0" active="false" root="true">
```



```
<Subcategory id="1" name="mujer"/>
  <Subcategory id="3" name="hombre"/>
</Category>
<Category id="1" name="mujer" levelDelpth="1" active="false" root="false">
  <Product id="0" name="falda"/>
</Category>
  <Category id="3" name="hombre" levelDelpth="0" active="false" root="false"/>
</listCategories>
<listProducts>
  <Product id="0" name="falda" active="false" price="0.0" shortDescription="falda
corta" longDescription="Falda de mujer corta" stock="10" hight="10.2" weight="11.5">
    <Provider id="1" nif="78957622R" company="ZARA"/>
  </Product>
</listProducts>
<listProviders>
  <Provider id="1" nif="78957622R" company="ZARA"/>
</listProviders>
</Shop>
```

10.3 Manual del Programador

10.3.1 GMF

Sí se hace algún cambio en el metamodelo, añadiendo un nuevo campo, quitándolo o añadiendo una nueva entidad es necesario volver a pasar por las diferentes etapas de creación del GME.

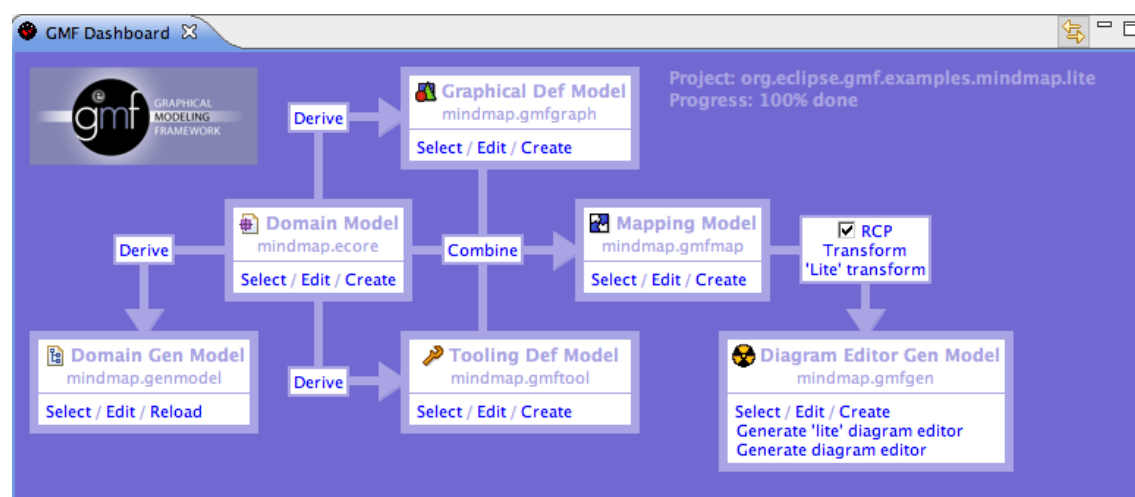


Ilustración 272 Etapas para la construcción del GMF

Todos estos pasos son estándar, existen tutoriales y guías en la web oficial que el programador podrá seguir a la hora de modificar el GMF (<http://wiki.eclipse.org/GMF>).

10.3.2 Aplicación web

A continuación se describirán algunos consejos y técnicas para continuar con el desarrollo de la aplicación web.

10.3.2.1 Configuraciones

El framework de Spring contiene un módulo de seguridad que facilita de forma no intrusiva la autenticación de usuarios, permite realizar la autenticación configurando sus clases desde el xml. Todas estas configuraciones se encuentran en la carpeta `conf` de la `WEB-INF`:

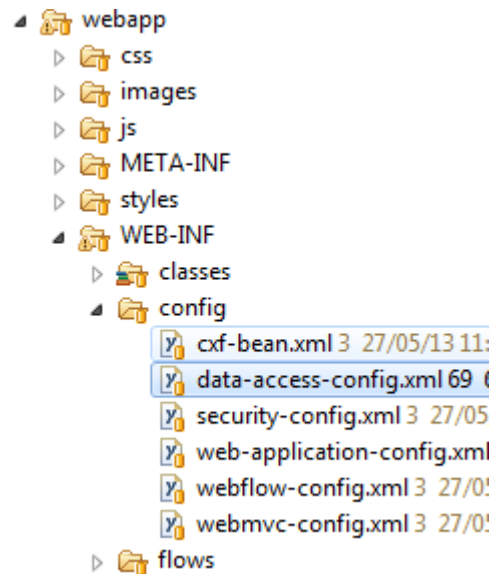


Ilustración 273 Configuraciones

Punto de entrada (archivo `security_config`):

Con el código de abajo indicamos la URL de la página de Login y Logout. Spring utiliza esta información para redirigir al usuario, es decir, cuando se intenta acceder a una página y no existe sesión creada, Spring redirige al usuario a la página de Login para autenticarlo, si la autenticación ha sido correcta lo vuelve a redirigir a la página que había intentado acceder previamente. Si el usuario cierra la sesión Spring destruye la sesión y lo redirige a la página de login.

```
<security:http auto-config="true">
  <security:form-login login-page="/spring/login" login-processing-
url="/spring/loginProcess" default-target-url="/spring/intro" authentication-failure-
url="/spring/login?login_error=1" />
  <security:logout logout-url="/spring/logout" logout-success-
url="/spring/logoutSuccess" />
</security:http>
```

Autenticación (`security_config`):

Definimos las queries que debe de realizar Spring para autenticar al usuario y cargar en sesión la lista de roles que posee para poder luego establecer los permisos en el resto de la aplicación.

```
<security:jdbc-user-service id="userDetailsService" data-source-ref="dataSource"
  users-by-username-query="select username, password, true from user where username=?"
  authorities-by-username-query="select u.username, r.name from user u
    inner join role_user
    on (u.id = role_user.USER_ID)
    inner join role r
    on (r.id = role_user.ROLE_ID)
```

```
where u.username=?" />
```

Acceso a datos (data_acces_config):

Se definen las configuraciones de acceso a datos en este fichero , en él se indican el protocolo de acceso, url, usuario y contraseña.

```
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
    <property name="url" value="jdbc:mysql://localhost:3306/tunosdb"/>
    <property name="username" value="root"/>
    <property name="password" value="root"/>
</bean>
```

10.3.2.2 Creación de flujos

En la arquitectura general del sistema Web existen frameworks que pertenecen a la parte de front-end, donde se interactúa con el usuario.

A través del navegador se puede acceder a la aplicación web que gestiona el sistema. El usuario accede a las vistas que son gestionadas por los flujos que se hayan definido previamente. Cuando el usuario entra en una vista que requiere identificación, este accederá a una vista donde podrá identificarse a través de su nombre de usuario y una contraseña, que se validará y permitirá al usuario acceder al sistema si ha realizado con éxito este paso.

El conjunto de flujos y vistas se creará en la carpeta flows, en este caso como se puede ver en la siguiente figura los flujos se encuentran agrupados principalmente por funcionalidad.

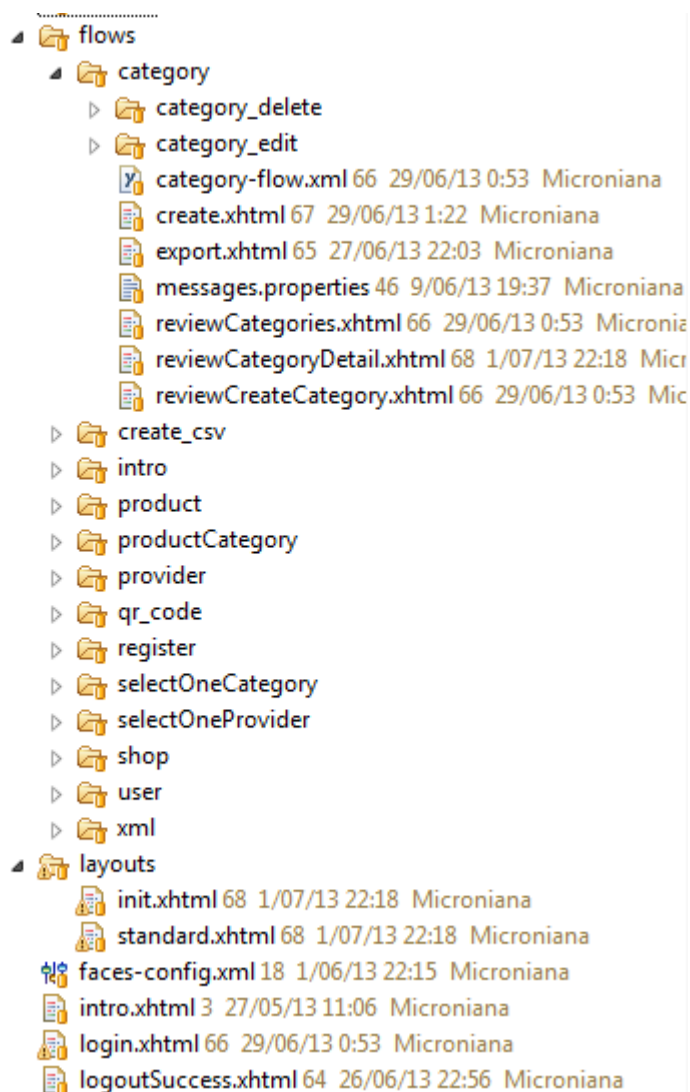


Ilustración 274 Flujos

10.3.2.2.1 Estructura de los flujos

Cada uno de los flujos cuenta con un estado inicial.

```
<?xml version="1.0" encoding="UTF-8"?>
<flow xmlns="http://www.springframework.org/schema/webflow"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/webflow
http://www.springframework.org/schema/webflow/spring-webflow-2.0.xsd"
      start-state="initFormCategory">
```

Dentro del flujo se pueden definir acciones o vistas. Este es el ejemplo de una acción que inicializa una Form de tipo category:

```
<action-state id="initFormCategory">
    <evaluate expression="categoryAction.createCategoryForm" />
    <transition on="success" to="reviewCategories" />
</action-state>
```

Las vistas están compuestas por transiciones que llevan a otras acciones u otras vistas, este es un ejemplo de la vista para editar categorías:

```
<view-state id="edit">
  <on-render>
    <set name="flowScope.ACTIVE_MENU" value="'CATEGORY'" type="string" />
  </on-render>
  <transition on="update" to="reviewUpdateCategory">
  </transition>
  <transition on="cancel" to="editCategoryEnd">
  </transition>
  <transition on="setParent" to="updateCategorySelect">
    <evaluate expression="categoryAction.saveParentOnFlow" />
  </transition>
</view-state>
```

Por último cada flujo cuenta con un estado final, este estado final representaría el fin del flujo de edición de categorías

```
<end-state id="editCategoryEnd" />
```

Capítulo 11. Conclusiones y Ampliaciones

y

11.1 Conclusiones

Con la herramienta descrita será posible la migración de información relevante desde el sitio web de comercio electrónico a otras plataformas existentes en el mercado. De esta forma se posibilitará minimizar el tiempo y esfuerzo necesarios para la creación de soluciones de comercio electrónico.

La consecución de interoperabilidad y reutilización en plataformas de comercio electrónico, supondrá un avance significativo hacia el desarrollo de un entorno digital común más seguro para las interacciones entre gobiernos, empresas y ciudadanos.

Los grandes actores en internet, redes sociales y principales servicios de búsqueda de información, cuentan ya con estrategias y tecnologías que permiten el intercambio de información en formatos estructurados estándar. Esta situación obliga a las empresas que hacen uso del comercio electrónico a mantenerse al día y hacer de esta evolución una ventaja competitiva.

El uso de tecnologías en la web como Spring Web Flow permitió desarrollar aplicaciones una aplicación flexible, altamente cohesiva y con un acoplamiento bastante bajo. El desarrollo con esta tecnología fue completamente satisfactorio y permitió crear las interfaces web con mayor facilidad y rehuso de componentes de navegación.

Otra de las ventajas de tecnologías utilizadas como el GMF es que son herramientas altamente potentes y que actualmente están tomando cada vez más importancia en el mercado.

11.2 Ampliaciones

Como otras ampliaciones futuras la idea es que la herramienta aumente el abanico de posibilidades de integración hacia más tipos de plataformas de comercio electrónico como puede ser OsCommerce y además se aumente el número de campos que cada entidad que se pueden manejar por ejemplo a productos poder añadir imágenes, color, descuento aplicables....

Por otro lado las investigaciones futuras deben orientar a fondo nuevos estándares de comercio electrónico para determinar el aporte de valor a un conjunto de empresas. Ha de tenerse en cuenta el marco jurídico aplicable que incentiva al uso de herramientas como la aquí propuesta, en particular, la Norma técnica Española de interoperabilidad de reutilización de recursos de información aprobada Mediante la Resolución de 19 de febrero de 2013. Esta norma establece condiciones comunes sobre selección, identificación, descripción, formato, condiciones de uso y puesta a disposición de los documentos y recursos de información elaborados o custodiados por el sector público, relativos a numerosos ámbitos de interés como la información social,

económica, jurídica, turística, sobre empresas, educación, etc., cumpliendo plenamente con lo establecido en la Ley 37/2007 de 16 de noviembre sobre reutilización de la información del sector público

Capítulo 12. Presupuesto

12.1 Mediciones

Como se puede ver en el diagrama de Gantt del apartado 4.1, el desarrollo del proyecto se prolonga durante 12 meses aproximadamente. El tiempo estimado para cada día es de unas 8 horas diarias. A continuación se muestra un resumen aproximado del total de horas destinadas a cada fase del proyecto.

Código Unidad	Concepto	Unidades	Cantidad total
1	Recopilación de la información	Horas	418
2	Estudio de plataformas de comercio electrónico	Horas	644
3	Alineamiento y definición de semántica de Procesos	Horas	560
4	Desarrollo y elaboración de la propuesta	Horas	778
5	Difusión de resultados	Horas	112

12.2 Cuadro de precios

Para calcular los costes por hora de desarrollo se crean tres recursos y se les asignan diferentes tasas dependiendo del rango que desempeñan. Los precios se han considerado en base a varias web que hacen estimaciones salariales actuales (<http://www.tufuncion.com/trabajo-programador>).

Código Unidad	Recurso	Tasa
1	Director de proyecto	50€/h
2	Analista	36€/h
3	Programador	17€/h

12.3 Totales

Teniendo en cuenta los valores estimados anteriormente y los demás costes se calcula el siguiente presupuesto.

El coste de cada una de las fases se obtiene mediante el cálculo de coste de cada una de las tareas realizadas en cada fase según el recurso asignado y el número de horas que se le estiman.

Item	SubItem	Concepto	Duración	Precio Unitario	Total
01		Gastos de personal de desarrollo			33.648€
	001	Recopilación de la información	418 horas		15.066€
		Director	4 horas	50€/hora	200 €
		Analista	412horas	36€/hora	14.832€
		Desarrollador	2 horas	17€/hora	34€
	002	Estudio de plataformas de comercio electrónico	644 horas		17.650€
		Director	4 horas	50€/hora	200€
		Analista	346 horas	36€/hora	12.452€
		Desarrollador	294 horas	17€/hora	4.988 €
	003	Alineamiento y definición de semántica de Procesos	560 horas		14.430€
		Director	6 horas	50€/hora	300€
		Analista	248 horas	36€/hora	8.928€
		Desarrollador	306horas	17€/hora	5.202 €
	004	Desarrollo y elaboración de la propuesta	778 horas		15.210€
		Director	6 horas	50€/hora	300€
		Analista	94 horas	36€/hora	3.384€
		Desarrollador	678horas	17€/hora	11.526 €
	005	Difusión de resultados	778 horas		2.122€
		Director	4 horas	50€/hora	200€
		Analista	4 horas	36€/hora	144€
		Desarrollador	104horas	17€/hora	1.768 €
02		Licencias software			697€
	001	Enterprise architect		Precio unitario	697€
03		Hardware			1360€
	001	Ordenador sobremesa de		Precio unitario / Amortización (800h*0.40)	320€
	002	Ordenador sobremesa de		Precio unitario / Amortización (800h*0.40)	320€
	003	Ordenador sobremesa de		Precio unitario / Amortización (800h*0.40)	320€
	004	Servidor		Precio unitario / Amortización (900*0.40)	360€
	005	Impresora		Precio unitario / Amortización (100*0.40)	40€
04		Oficina			2.600€
	001	Uso de laboratorio	12 meses	216,66€/mes	2.600€
05		Materiales			1200€
				Subtotal	70.335€
				Beneficio (25%)	17.583,75€
				IVA (21%)	14.770,35

TOTAL	102.689,1€

La tabla muestra los diferentes gastos del proyecto. Se han incluido la amortización del software y hardware necesarios para el desarrollo del proyecto, alquiler del laboratorio, así como los gastos de material de oficina durante el periodo de desarrollo. Sobre el total de los costes, se aplica un incremento del 25% de beneficio más un 21% de IVA obteniéndose así el total del presupuesto para el cliente.

Capítulo 13. Referencias Bibliográficas

- Anacleto, V. A. (2006). MDA: Reusabilidad Orientada al Negocio. *epiwiki*. Retrieved from
- Antolín, R. G. (2009). APLICACIÓN DE INGENIERÍA DIRIGIDA POR MODELOS (MDE) EN PROCESOS DE NEGOCIO (BPM).
- Ausubel, D., Novak, D., & Hanesian, H. (1986). *Educational Psychology: A Cognitive View*. (R. & W. Holt, Ed.) (2nd ed., pp. 29 -30). New York: Warbel & Peck.
- Byrd, T. A. , Cossick, K. L., and Zmud, R. W. (1992). A synthesis of research on requirements analysis and knowledge acquisition techniques. *MIS Quarterly*, 16(1), 117 -138.
- Cañas, A., Ford, K., & Hayes, P. (1997). Colaboración en la construcción de conocimiento mediante mapas conceptuales. *Institute for Human ...*. Retrieved from http://skat.ihmc.us/rid=1216436284856_756283406_19530/Colaboraci%C3%B3n.pdf
- Chu, S.-C., Leung, L. C., Hui, Y. V., & Cheung, W. (2007). Evolution of e-commerce Web sites: A conceptual framework and a longitudinal study. *Information & Management*, 44(2), 154-164. doi:10.1016/j.im.2006.11.003
- Eclipse Foundation. (2012). Eclipse documentation. 2012. Retrieved from <http://help.eclipse.org/indigo/index.jsp>
- Eclipse Foundation. (2013). Ecore Tools. Retrieved from http://wiki.eclipse.org/index.php/Ecore_Tools
- Faroughian, F. F., Kalafatis, S. P., Ledden, L., Samouel, P., & Tsogas, M. H. (2012). Value and risk in business-to-business e-banking. *Industrial Marketing Management*, 41(1), 68-81. Elsevier Inc. doi:10.1016/j.indmarman.2011.11.012
- Garcia, A., Ruiz, B., Lopez, J. L., & Gonzalez, I. (2011). Semantic model for knowledge representation in e-business. *Knowledge-Based Systems*, 24(2), 282-296. Elsevier B.V. doi:10.1016/j.knosys.2010.09.006
- González, R., Martínez, O. S., Cueva, J., García-Bustelo, B. C., Gayo, J. E. L., & Ordoñez, P. (2011). Recommendation System based on user interaction data applied to intelligent electronic books. *Computers in Human Behavior*, 27(4), 1445-1449.
- Henderson-Sellers, B. (2011). Bridging metamodels and ontologies in software engineering. *Journal of Systems and Software*, 84(2), 301-313. Elsevier Inc. doi:10.1016/j.jss.2010.10.025
- Institute of Electrical and Electronics Engineers. (1990). *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York.
- Kauremaa, J., Nurmilaakso, J.-M., & Kari, T. (2010). E-business enabled operational linkages: The role of RosettaNet in integrating the telecommunications supply chain. *International*

Journal of Production Economics, 127(2), 343-357. Elsevier.
doi:10.1016/j.ijpe.2009.08.024

Montenegro, C. (2011). *Modelado específico de dominio para la construcción de learning objects independientes de la plataforma*. Universidad de Oviedo.

Montenegro-marín, C. E., & Cueva-lovelle, J. M. (2012a). Domain specific language for the generation of learning management systems modules. *Journal of Web Engineering*, 11(1), 23-50.

Montenegro-marín, C. E., & Cueva-lovelle, J. M. (2012b). Domain specific language for the generation of learning management systems modules. *Journal of Web Engineering*, 11(1), 23-50.

Mora, B., García, F., Ruiz, F., Piattini, M., Boronat, A., Gómez, A., Carsí, J. Á., et al. (2011). Based on MDA. *IEEE Latin America Transactions*, 9(1), 130-137.

Márquez-Solís, S. (2007). *La Web Semántica* (primera ed., p. 30).

Novak, D., & Gowin, D. B. (2002). *Learning How to Learn* (p. 216). Cambridge University Press.

Noy, N. F., & McGuinness, D. L. (2000). *Ontology Development 101 : A Guide to Creating Your First Ontology*.

Nurmilaakso, J.-M. (2008). EDI, XML and e-business frameworks: A survey. *Computers in Industry*, 59(4), 370-379. doi:10.1016/j.compind.2007.09.004

Nurmilaakso, J.-M., & Kotinurmi, P. (2004). A review of XML-based supply-chain integration. *Production Planning & Control*, 15(6), 608-621. doi:10.1080/09537280412331283937

Qingfeng, Z., Wenbo, C., & Lihua, H. (2008). E-Business Transformation : An Analysis Framework Based on Critical Organizational Dimensions. *Tsinghua Science and Technology*, 13(3), 408-413.

Robertshaw, T. (2012). October 2012 Ecommerce Survey. Retrieved from <http://tomrobertshaw.net/2012/11/october-2012-ecommerce-survey/>

Selma, K., Ilyès, B., Ladjel, B., Eric, S., Stéphane, J., & Baron Michael. (2012). Ontology-based structured web data warehouses for sustainable interoperability: requirement modeling, design methodology and tool. *Computers in Industry*, 63(8), 799 - 812. Retrieved from <http://dx.doi.org/10.1016/j.compind.2012.08.001>,

Soto, P., & Meroño, A. (2009). Evaluating Internet technologies business effectiveness. *Telematics and Informatics*, 26(2), 211-221. doi:10.1016/j.tele.2008.01.004

Tarazona, G. M., Gómez, M., & Montenegro, C. E. (2012). Buenas prácticas para implementación del comercio electrónico en pymes. *Visión Electrónica: algo más que un estado sólido*, 2, 31-45. Retrieved from <http://revistas.udistrital.edu.co/ojs/index.php/visele/article/view/3884/5499>

Tarazona, G. M., & Rodríguez, L. (2013). Model-Driven Engineering for Electronic Commerce. In V. García, J. M. Cueva-lovelle, C. Pelayo-García, & O. Sanjuán (Eds.), *Progressions and*

Innovations in ModelDriven Software Engineering (Vol. i, pp. 196 - 208). Hershey USA: Engineering Science Reference (an imprint of IGI Global).

Tarazona-Bermudez, G. M., Pelayo G-Bustelo, C., & Sanjuan, O. (2013). Metamodelo de Procesos de Comercio Electronico (E-Commerce) para lograr la Reutilización y la Interoperabilidad. *CISTI'2013, la 8ª Conferencia Ibérica de Sistemas y Tecnologías de Información*, 2(1), 284 - 289.

Telefónica. (2013). *Fundación Telefónica* (p. 186). Barcelona: Ariel S.A. Retrieved from http://e-libros.fundacion.telefonica.com/sie12/aplicacion_sie/ParteA/pdf/SIE_2012.pdf

Vernadat, F. B. (2010). Technical, semantic and organizational issues of enterprise interoperability and networking. *Annual Reviews in Control*, 34(1), 139-144.

Wegner, P. (1996). Interoperability. *ACM Computing Surveys*, 28(1), 285-287.
doi:10.1145/234313.234424

Zou, Z., & Deng, Y. (2011). The Evaluation Model of E-business Based on Process. *Energy Procedia*, 13, 5601-5608. doi:10.1016/j.egypro.2011.12.208

Zwass, V. (1996). Electronic Commerce : Structures and Issues. *International Journal of Electronic Commerce*, 1(1), 3 -23.

Capítulo 14. Apéndices

14.1 Glosario y Diccionario de Datos

- **Comercio Electrónico:** también conocido como e-commerce (electrónico commerce en inglés), consiste en la compra y venta de productos o de servicios a través de medios electrónicos, tales como Internet y otras redes informáticas.
- **DAO (Data Access Object):** es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.
- **CSS:** hacen referencia a un lenguaje de hojas de estilos usado para describir la presentación semántica (el aspecto y formato) de un documento escrito en lenguaje de marcas.
- **CSV(comma-separated values):** son un tipo de documento en formato abierto sencillo para representar datos en forma de tabla, en las que las columnas se separan por comas y las filas por saltos de línea
- **DSL (Domain Specific Language):** lenguaje de dominio específico. Se trata de un lenguaje de programación creado con un objetivo concreto.
- **Eclipse:** entorno de desarrollo utilizado para implementar principalmente con código Java.
- **Foundation:** tecnología utilizada para la maquetación de páginas web
- **GMF(Graphical Modeling Framework):** Componente generativo infraestructura de tiempo de ejecución para el desarrollo de editores gráficos
- **Hibernate:** es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java
- **Interoperabilidad:** habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.
- **Java:** lenguaje de programación de propósito general orientado a objetos.
- **JPA (Java Persistence Api):** es la API de persistencia desarrollada para la plataforma Java EE.
- **JSF (Java Server Faces):** es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE.
- **MDE(Ingeniería dirigida a Modelos):** una propuesta para la construcción de software en la que se le atribuye a los modelos el papel principal de todo el proceso, frente a las propuestas tradicionales basadas en lenguajes de programación, plataformas de objetos y componentes software
- **MySQL:** sistema de gestión de bases de datos relacional, multihilo y multiusuario
- **MVC (model view controller):** es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.
- **Ontología:** es una rama de la metafísica que estudia lo que hay.
- **QR (Quick Response code) :** es un módulo útil para almacenar información en una matriz de puntos o un código de barras bidimensional
- **SWF(Spring Web Flow):** es el subproyecto de la Spring Framework que se centra en proporcionar la infraestructura para la creación y ejecución de aplicaciones web ricas.
- **Tomcat:** contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation.

- **URL (Uniform Resource Locator):** dirección única que identifica una página web en Internet
- **XML (Extensible Markup Language):** lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible.

14.2 Contenido Entregado en el CD-ROM

En esta sección se incluye una descripción de los contenidos que se encuentran en el CD presentado junto con la documentación.

Directorio	Contenido
<i>./ Directorio raíz del CD</i>	Contiene un fichero leeme.txt explicando toda esta estructura.
<i>./Metamodelos_de_procesos_de_comercio_electronico</i>	Contiene toda la estructura de directorios del proyecto.
<i>./Metamodelos_de_procesos_de_comercio_electronico /BBDD</i>	Contiene un script de creación de la base de datos, con datos de ejemplo.
<i>./Metamodelos_de_procesos_de_comercio_electronico /herramientas</i>	Contiene las herramientas básicas para el desarrollo del proyecto
<i>./Metamodelos_de_procesos_de_comercio_electronico /documentación</i>	Contiene en su raíz el presente fichero .pdf que representa la documentación del proyecto.
<i>./Metamodelos_de_procesos_de_comercio_electronico /documentación/diagramas</i>	Contiene muchos de los diagramas utilizados en el análisis y el diseño de la documentación.
<i>./Metamodelos_de_procesos_de_comercio_electronico /documentación/javadoc</i>	Contiene los ficheros de documentación de los subsistemas DSL y aplicación web, generados automáticamente mediante Javadoc
<i>./Metamodelos_de_procesos_de_comercio_electronico /documentación/mockups</i>	Contiene las diferentes interfaces de usuario de la aplicación realizados durante la fase de análisis y diseño.
<i>./Metamodelos_de_procesos_de_comercio_electronico /documentación/planificación</i>	Contiene un archivo creado mediante Microsoft Project que refleja la planificación seguida en el proyecto
<i>./Metamodelos_de_procesos_de_comercio_electronico /fuentes</i>	Contiene dos directorios /DSL y /Web, cada uno de ellos contiene los archivos fuentes de los respectivos subsistemas.
<i>./Metamodelos_de_procesos_de_comercio_electronico /mavenDependences</i>	Contiene el repositorio de dependencias necesarias para el desarrollo de la Web.
<i>./Metamodelos_de_procesos_de_comercio_electronico /ontología</i>	Contiene un archivo .OWL construido con Protege que contiene la ontología desarrollada
<i>./Metamodelos_de_procesos_de_comercio_electronico /WAR</i>	Contiene un archivo .war necesario para desplegar la aplicación web.

14.3 Índice Alfabético

C

CSS, 217, 223, 322, 332

CSV, 26, 27, 68, 69, 81, 106, 107, 114, 176, 280, 322, 341, 342, 366

D

DAO, 94, 95, 103, 104, 152, 160, 162, 169, 173, 218, 322

DSL, 5, 18, 19, 20, 23, 50, 51, 55, 63, 68, 69, 80, 81, 87, 89, 90, 93, 99, 216, 218, 229, 322

E

Eclipse, 23, 52, 54, 55, 87, 227, 322

G

GMF, 23, 55, 68, 70, 71, 73, 82, 84, 85, 87, 88, 90, 112, 115, 132, 134, 135, 137, 155, 156, 199, 218, 227, 229, 252, 260, 301, 304, 305, 309, 314, 322

H

Hibernate, 92, 152, 159, 178, 219, 225, 226, 322

I

interoperabilidad, 4, 5, 18, 19, 23, 24, 29, 45, 46, 47, 49, 64, 68, 70, 71, 79, 314

Interoperabilidad, 45, 63, 322

J

Java, 50, 91, 152, 159, 216, 218, 219, 220, 223, 227, 228, 254, 322

M

Magento, 4, 24, 25, 26, 27, 62, 68, 69, 70, 71, 74, 79, 80, 81, 88, 92, 100, 107, 108, 109, 110, 111, 113, 114, 153, 165, 173, 174, 175, 176, 217, 229, 263, 280, 294, 295, 297, 298, 326, 327, 328, 329, 330,

331, 332, 333, 334, 336, 355, 356, 359, 407, 409, 446, 447, 451, 452, 465, 484

MDE, 4, 5, 18, 49, 52, 63, 65, 68, 322

MVC, 91, 219, 322, 340, 341

MySQL, 178, 227, 228, 253, 255, 256, 322, 326, 327, 333, 337, 351

O

ontología, 19, 23, 28, 29, 30, 31, 40, 50

Ontología, 63, 65, 322

Oscommerce, 341

P

Prestashop, 4, 25, 26, 27, 62, 68, 69, 70, 71, 74, 79, 80, 81, 88, 92, 100, 107, 114, 153, 165, 173, 174, 175, 176, 217, 229, 263, 280, 294, 336, 338, 339, 340, 341, 342, 351, 357, 360, 407, 409, 449, 450, 453, 454, 455, 465, 484

Q

QR, 69, 77, 107, 190, 281, 322, 379

S

SWF, 92, 152, 163, 220, 221, 322

T

Tomcat, 70, 154, 155, 227, 253, 255, 259, 322

U

URL, 259, 310, 323, 337, 339, 344, 345

X

XML, 23, 27, 45, 47, 50, 64, 68, 69, 70, 71, 80, 81, 85, 87, 89, 106, 112, 114, 115, 132, 133, 155, 177, 197, 198, 199, 216, 263, 301, 302, 303, 308, 323, 332, 334, 335, 340

14.4 Anexo I: Plataformas Analizadas

14.4.1 MAGENTO

Magento es un sistema potente de tiendas en línea, es open source, con la posibilidad de tener acceso a la tienda en dispositivos móviles phone, Android. Se trata de un sistema escalable con las versiones que son pagas. Entre las muchas funcionalidades podemos destacar.

14.4.1.1 Medios de pago

Tiene la posibilidad de implementar PayPal, Google Checkout, USAepay, entre otros. Además puede crear nuevos métodos de pago y contar con una buena lista de módulos ya creados para los pagos.

14.4.1.2 Estadísticas e informes

El sistema de estadísticas es una gran diferencia comparado con las otras tiendas Magento tiene un sistema completo de ellas contando con la posibilidad de integrar fácilmente Google Analytics.

14.4.1.3 SEO

Magento cuenta con un sistema de URLs amigables

14.4.1.4 Requerimientos del Servidor

Prerrequisitos técnicos	
Sistema Operativo	-Linux, Windows, u otro sistema operativo compatible con UNIX
Servidor Web	-Servidor Web Apache (1.x o 2.x) -PHP 5.2.0 o superior, con las(os) siguientes extensiones/Agregados PDO/MySQL MySQLi Mcrypt Mhash Simplexml DOM Curl
Base de datos	MySQL 4.1.20 o superior
Servidor de correo	-Un Agente de Transferencia de Email (MTA) compatible con Sendmail(Magento se conectará directamente a un servidor SMTP si usted no dispone de un MTA)
Hosting	Unix hosting
Navegador web	Mozilla firefox, Google Chrome...

14.4.1.5 Procedimiento de instalación

Se describe el procedimiento sobre como instalar Magento en un servidor.

Pasos:

1. Se comprueba que las características del servidor son las que Magento requiere.
2. Para descargar la última versión de Magento es necesario registrarse y a continuación descargar [Magento 1.0.19870.1.zip](http://www.magentocommerce.com/download) de la web <http://www.magentocommerce.com/download>
3. Crea una base de datos MySQL para Magento .
4. Se descomprime el .zip en el directorio htdocs de nuestro servidor Apache
5. A continuación se usa el navegador habitual para acceder a la carpeta magento/install. Si se subido directamente los archivos al directorio raíz, la dirección será <http://www.ejemplo.com/install>. Por el contrario, la dirección será <http://www.ejemplo.com/magento/install>.
6. Se iniciara el asistente instalación de magento. En la primera pantalla tendremos que aceptar los términos y condiciones, cuando lo tengamos pulsamos en Continue.



Ilustración 275 Instalación Magento –Paso 1

7. La siguiente pantalla nos permite escoger el idioma que deseemos, la zona horaria y la moneda por defecto que maneja nuestra tienda online, seleccionamos la que nos corresponda y a continuación pulsamos en Continue.

The screenshot shows the 'Localization' step of the Magento installation. On the left, a sidebar lists the installation steps: Download, License Agreement, Localization (highlighted), Configuration, Create Admin Account, and You're All Set. Below this list, a link to the 'Installation Guide' is provided. The main area is titled 'Localization' and contains a 'Locale settings' section. This section has three fields: 'Locale' (set to 'Español (España) / español (España)'), 'Time Zone' (set to 'Romance Standard Time (Europe/Paris)'), and 'Default Currency' (set to 'euro'). A 'Continue' button is located at the bottom right of the form. A note at the bottom right indicates that fields marked with an asterisk are required.

Ilustración 276 Instalación Magento –Paso 2

8. Configuramos la base de datos. Aquí se pone el nombre de la base de datos que creamos anteriormente y el usuario y la contraseña para poder acceder.

The screenshot shows the 'Configuration' step of the Magento installation. The sidebar on the left highlights 'Configuration' as the current step. The main area is titled 'Configuration' and contains two sections: 'Database Connection' and 'Web access options'. The 'Database Connection' section includes fields for 'Database Type' (MySQL), 'Host' (localhost), 'Database Name' (magento), 'User Name' (root), and 'User Password'. It also includes a 'Tables Prefix' field and a note about specifying server port or UNIX socket. The 'Web access options' section includes fields for 'Base URL' (http://localhost/magento/) and 'Admin Path' (admin), and a checkbox for 'Enable Charts' which is checked.

Ilustración 277 Instalación Magento –Paso 3

9. A continuación se crea el usuario para acceder a magento. En este caso se crea un usuario Admin.

Ilustración 278 Instalación Magento –Paso 4

10. Se sigue con el resto de pasos y se finaliza la instalación.

Solución de algunos errores producidos durante la instalación

Durante la instalación nos surgió el error **PHP Extensión "curl" must be loaded** para solucionarlo tuvimos que acceder al archivo **php.ini** abrirlo y buscar:

`;extension=php_curl.dll`

Y dejarla de la siguiente manera:

`extension=php_curl.dll`

Otro error que nos surge durante la instalación es **PHP Extensión "mcrypt" must be loaded** lo solucionamos cambiando en el archivo **php.ini** la línea

`;extension=php_mcrypt.dll`

Por: `extension=php_mcrypt.dll`

14.4.2 Arquitectura de Magento

Magento está construido sobre el Zend Framework, el mismo proporciona una librería de código orientado a objetos fue construido con tres principios fundamentales:

- **Flexibilidad:** El código de Magento permite una personalización adecuada a las necesidades de negocio.
- **Actualizable:** Separando el código del núcleo, del de la comunidad y las personalizaciones; Magento puede ser personalizado sin perder la habilidad de actualizarse.
- **Velocidad y Seguridad:** Los estándares de codificación usados por los desarrolladores, siguen las mejores prácticas para maximizar la eficiencia del software y proporcionan una segura vitrina en línea.

14.4.2.1 Núcleo

El núcleo del código es una colección de módulos desarrollados o certificados por el equipo principal de desarrollo de Magento. La edición de los archivos del núcleo es compleja.

14.4.2.2 Local

Las extensiones locales son personalizaciones de Magento que residen únicamente en una copia local del usuario. Estas extensiones serán puestas en una carpeta local de tal forma que no interfieran con las actualizaciones del núcleo del código, y en un orden para diferenciarlos de las contribuciones de la comunidad. Existen diferentes tipos de extensiones, todas residen en el mismo directorio.

Las extensiones locales funcionan tal como lo hace el código del núcleo, sólo el directorio es diferente.

La ubicación de la carpeta local es *app/code/local*

14.4.2.3 Comunidad

Las contribuciones de la comunidad se descargan y se ubican en la carpeta “community”. Al igual que las extensiones locales se mantiene separado del código del núcleo de Magento para que los propietarios de las tiendas puedan disfrutar de funcionalidades adicionales sin comprometer la capacidad de actualizarse a futuras versiones de Magento.

La ubicación de la carpeta “community” es *app/code/community*.

14.4.2.4 Extensiones

Uno o más paquetes de archivos unidos para extender la funcionalidad de Magento. Términos estrictos y condiciones prohíben que las extensiones modifiquen el núcleo del código, asegurando que cualquier funcionalidad extendida no le prohíba la actualización cuando una nueva versión de Magento sea lanzada.

Las extensiones pueden ser instaladas desde el panel de Administración o descargadas desde “Magento Connect”.

Hay tres tipos de extensiones, y residen en una de las dos ubicaciones descritas anteriormente.

14.4.2.5 Módulos

Un Módulo es una extensión que extiende las características y funcionalidades de Magento. Algunos ejemplos concretos de módulos podrían ser las integraciones adicionales con diferentes pasarelas de pago, o herramientas para elementos promocionales.

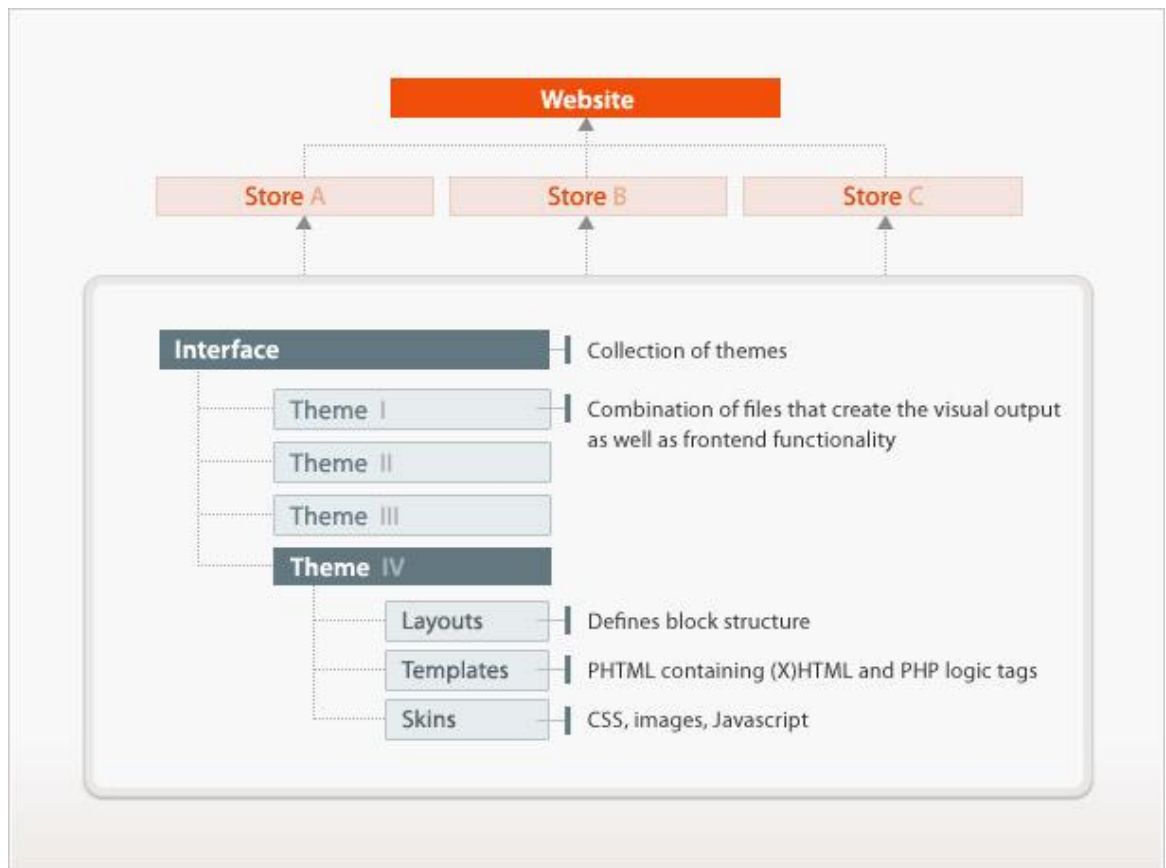


Ilustración 279 Composición de módulos Magento

14.4.2.6 Interfaz

Una interfaz es una colección de temas que determinan la salida visual y funcionalidades del “sitio” (front-end) de su tienda. Una interfaz puede ser asignada a nivel de sitio web, a nivel de tienda o a nivel de Sitio Web y la tienda, a través del panel de administración.

14.4.2.7 Temas

Un tema es una combinación de un diseño(layout), plantilla (template) y/o archivos de máscara (skins) que crean la experiencia visual de su tienda. Magento está construido con la capacidad de cargar múltiples temas a la vez y por lo tanto distingue los temas en dos tipos.

- **Temas por defecto** - Cada interfaz viene con un tema llamado ‘default’ que es el tema principal de un interfaz. Cuando asigna una interfaz a una tienda, la aplicación automáticamente le echa un vistazo a este tema y lo carga en el “sitio” (front-end). A fin de personalizar el diseño de una tienda, se puede, modificar este tema, o crear un tema “Non-Default” adicional y cargarlo junto al que está por defecto. El tema por defecto debe contener todos los diseños, plantillas y máscaras para correr una tienda libre de errores y por esa razón es el menor tema en la jerarquía de temas.
- **Temas “Non-Default”** - Un tema “non-default” puede contener tantos archivos como crea que necesite. Este tipo de tema está pensado para crear cambios de diseños temporales a una tienda, sin tener que crear un nuevo tema por defecto completo.

Creando algunas imágenes y actualizando algo del CSS, puede fácilmente cambiar la tienda de algo normal a un nuevo diseño.

14.4.2.7.1 Los componentes de un tema:

- **Diseños(Layouts)** - Los diseños son básicamente archivos XML que definen la estructura en bloque para diferentes páginas. También controlan los metadatos y la codificación de la página. Los archivos de diseño están separados por cada módulo. Cada módulo trae consigo su propio archivo de diseño.
- **Plantillas(Templates)** - Las plantillas son archivos PHTML que contienen marcas HTML y cualquier etiqueta PHP necesaria para crear la lógica para la presentación visual de la información y características.
- **Máscaras (Skins)** - Las máscaras son bloques específicos Javascript, CSS e imágenes que complementan su HTML.

14.4.2.8 Bloques

Los bloques son la forma en que Magento distingue el arreglo de funcionalidades en el sistema y crea una forma modular para manejar este arreglo desde el punto visual y funcional. Hay dos tipos de bloques y trabajan juntos para crear la salida visual.

14.4.2.8.1 Bloques Estructurales

Son bloques creados con el único propósito de asignar una estructura visual a una página de la tienda. Ejemplo podrían ser la cabecera, columna izquierda, columna derecha, columna principal y pie.

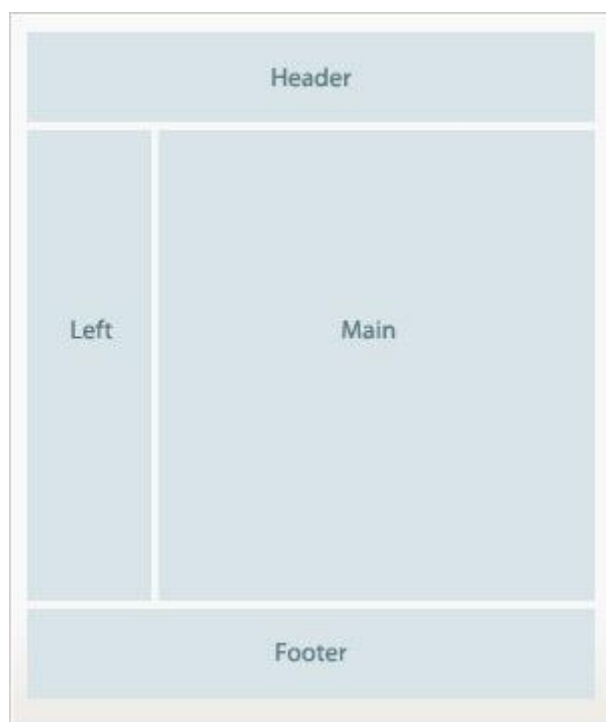


Ilustración 280 Bloques estructurales de Magento

14.4.2.8.2 Bloques de Contenido

Estos son bloques que producen el contenido dentro de cada bloque estructural. Son representaciones de cada característica funcional en una página y emplea archivos del template para generar el HTML a ser insertado dentro de su bloque estructural padre. Ejemplo son la lista de categorías, mini carros, etiquetas y listas de productos.



Ilustración 281 Bloques de contenido Magento

14.4.3 Funcionalidades de Magento

Magento como plataforma de eCommerce es una muy eficiente solución de código abierto (GRATIS y SEGURA). Ofrece una extensiva gama de herramientas para crear y administrar su tienda online. Magento funciona en un ambiente Apache/MySQL/PHP y requiere un proveedor hospedaje web apropiado y confiable. El soporte profesional puede ser dado por la empresa quien creo el software o bien alguna de las cientos de empresas que trabajan y conocen Magento por una mensualidad. La gran gama de opciones y capacidades del programa es uno de los principales beneficios de Magento.

A continuación les presento una lista de las principales opciones de Magento que vienen incluidas y que les pueden ayudar crear una online de manera profesional.

- **Estadísticas de uso e informes:** El programa está integrado con Google Analytics (page view, etc) y ofrece diferentes tipos de reportes.
- **Vistas de productos:** imágenes múltiples para productos, opciones de comentarios, listas de deseos o “wishlist” y mucho más.

- **Vistas del catálogo:** Fácil navegación, filtros avanzados de productos, comparación de productos.
- **Administración de catálogo:** Adm de inventarios, importe y exporte productos, diferentes tipos de impuestos por lugar si es necesario, atributos adicionales de productos.
- **Cuentas de clientes.** estatus de órdenes e historia, email wishlist, RSS de productos, listas de correos “newsletter”, direcciones de envíos.
- **Servicio al cliente:** opciones para las cuentas de los clientes, perfil de cliente, formulario de contacto, sistema de administración y rastreo de ordenes con historial, emails personalizables de órdenes.
- **Administración de órdenes:** Admis pueden generar ordenes desde el sistema, crear diferentes facturas para envíos y créditos, pueden crear ordenes sus empleados.
- **Pagos:** Diferentes medios de pagos, tarjetas de crédito, PayPal, Authorize.net, Google Checkout, cheques, money orders, ordenes de compra y soportes para módulos externos de empresas de procesamientos de pagos.
- **Envíos:** Envíos a diferentes direcciones, costo fijo por compra, soporte a UPS, UPS XML, FedEx, USPS y DHL.
- **Checkouts:** Checkout con una sola página, soporte para SSL, checkout sin necesidad de tener o crear una cuenta.
- **Optimizar para Motores de Búsqueda (SEO):** Es la aplicación mas amigable para SEO de e-commerce, cuenta con soporte para Google Sitemaps.
- **Soporte internacional:** múltiples lenguajes y monedas, opcion para paises permitidos para registro de clientes, compras, envíos, etc. Tenga control de quien puede o no comprar en su tienda.
- **Marketing y herramientas de promoción:** cupones, descuentos y diferentes parámetros para ofertas.
- **Administración del sitio:** Controle diferentes website y tiendas, múltiples lenguajes, tablas de impuestos para EEUU y otros países, personalización del diseño por medio de templates.
- **Llaves de encryption:** almacenamiento seguro de datos sensibles en el código de la base de datos.
- **Comercio móvil:** Optimización para iPhone. El programa viene por defecto con una plantilla desplegar correctamente su sitio en iPhones.
- **Extensiones:** Amplia librería de extensiones pagadas para aumentar las funcionalidades de Magento.

14.4.4 Sistemas de plantillas

El sistema de plantillas en Magento es complejo y causa confusión en algunos usuarios por la opción de usar PHP regularmente para el lenguaje de plantillas. Sin embargo, profundizando en él nos encontramos con un sistema flexible y avanzado en PHP.

Una página completa es reproducida como un conjunto jerarquizado de archivos de plantilla (técnicamente, bloques anidados en conjunto). No hay “widgets” explícitos en el sistema, es decir, no se encuentra una determinada Clase de “form” ni Clase de “button” o de objeto. Las Clases están subordinadas a bloques que se sitúan en el límite entre los widgets y las plantillas.

El conjunto jerarquizado de las plantillas y bloques es controlado por un archivo XML, o más

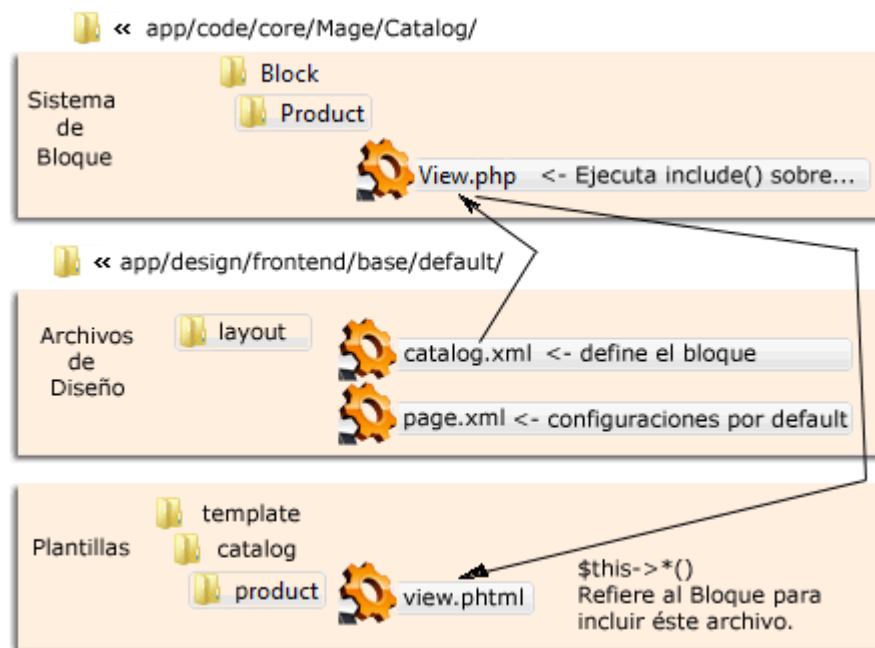


Ilustración 282 Sistema de plantillas Magento

específicamente, de un conjunto de archivos XML. Archivos de Diseño (Layout).

Los archivos de diseño controlan la estructura de cualquier reproducción final de la página. Se encuentran en la carpeta “layout” bajo el esquema del tema seleccionado. Hay una serie de archivos XML cuyos nombres se refieren a un módulo individual, pero todas las letras están en minúsculas,

mientras que los nombres de los módulos utilizan el método tradicional conocido como “camel-case”. El archivo XML más importante es `page.xml`.

```
app/design/frontend/base/default/layout/ para ver. 1.4.x o mayor
app/design/frontend/default/default/layout/ para ver. 1.3.x o menor
...
page.xml
catalogsearch.xml
catalog.xml
checkout.xml
cms.xml
contacts.xml
...
```

El archivo `page.xml` especifica la estructura predeterminada de la página. Todas las modificaciones de cualquiera de los otros archivos XML son modificaciones de las configuraciones bajo la etiqueta `<default>` XML. La siguiente es una lista de etiquetas que son comunes a todos los archivos de diseño.

```
<layout>
<default>
<reference>
<block>
<action>
<update>
```

Estas etiquetas siguen un patrón que se relaciona con el módulo, el controlador y la acción de la solicitud web determinada. Si la etiqueta sólo tiene dos partes, separadas por un guión bajo,

como <cms_page>, entonces estas configuraciones se aplican a todas las peticiones en ese módulo y el controlador.

```
<cms_page>
<cms_index_defaultindex>
<cms_index_defaultnoroute>
<customer_account_index>
<tag_customer_view>
<catalog_product_view>
```

14.4.4.1 Archivos de Plantilla

Los archivos de plantilla, son simplemente archivos PHP + HTML sin formato que terminan en extensión .phtml. La sintaxis utilizada en estos archivos intenta utilizar las características del lenguaje de plantillas de la sintaxis de PHP. Se alterna la estructura de la sintaxis de lazo de PHP, que utiliza los dos puntos (:) y la de tipo endwhile, endfor y endif, en estos archivos.

La estructura de los directorios tiene la estructura de los módulos correspondientes, siguiendo las convenciones del framework Zend, aunque no es obligatorio. Se puede optar por romper las convenciones y mantener todos sus archivos de plantilla en un sólo directorio en la construcción de módulos personalizados, de este modo será más fácil de gestionar los archivos.

Todos estos archivos se encuentran en el subdirectorio o carpeta “page”. Los archivos .phtml en este directorio son el de mayor nivel de cambio que se puede aplicar a cualquier página. Otorgan estructura de 1, 2, ó 3 columnas a cualquier página, así como el abastecimiento páginas tipo “panel” y un diseño para imprimir amigable.

Aunque se puede agregar cualquier archivo de plantilla de nivel superior en el directorio “page” al tema seleccionado, los archivos base/default son la única opción disponible para ti a través de la interfaz de administración.

14.5 Prestashop

Prestashop es una solución de e-comercio para generar y administrar tiendas on-line de una manera sencilla. El proyecto fue creado en 2007 (“About Prestashop,” n.d.) en Francia por **Igor Schumberger** y **Bruno Lévêque**. Ambos pretendían hacer más accesible el comercio electrónico

Dispone de un back-office muy potente que le permite administrar en tiempo real la tienda (catálogo, historial de pedidos, gastos de envío, clientes, carritos de compra...). Además ofrece un panel de administrador completo, de utilización simple, que propone más de 200 aplicaciones en estándar, y permite una gran adaptación y personalización.

Es bastante fácil de utilizar y bastante eficiente en cuanto a rapidez. Actualmente PrestaShop SA cuenta con una plantilla de 60 colaboradores y una comunidad de más de 110.000 usuarios. En la versión 1.5 dan soporte a multitiendas poniéndose al nivel de funcionalidad de herramientas como Magento, una de las líderes del mercado.

Otra de las grandes ventajas de la herramienta es que cuenta con un gran número de desarrolladores

14.5.1 Instalación

La aplicación es bastante sencilla de instalar, basta con descargar la aplicación (<http://www.prestashop.com/es/descarga>) y seguir los pasos indicados en la web.

Los requisitos de instalación básica son los siguientes:

Prerrequisitos técnicos	
Servidor Web	-Apache Web Server 1.3 o posterior, un mínimo de 64 Mb de RAM. -PHP 5.2 o posterior
Base de datos	MySQL 5.0 o posterior.
Editor de texto	Notepad++,writebox..
Hosting	Unix hosting
Navegador web	Mozilla firefox, Google Chrome...
Cliente FTP	Filezilla,GoFTP...

En caso de no tener un cliente FTP donde alojar la web para probar la herramienta de manera sencilla se pueden utilizar un servidor XAMPP. Este incluye dentro de la misma herramienta:

- Servidor Apache
- Bases de datos de MySQL,
- Servidor de correo SMTP
- PHP con panel PHPMyAdmin.

Para la instalación con esta herramienta, en primer lugar se descomprime la aplicación en el directorio “..\htdocs”.

A continuación vamos a nuestro navegador web y accedemos a la URL “<http://localhost:8080/prestashop/install/>”.

Inmediatamente nos aparece un instalador con el siguiente aspecto:

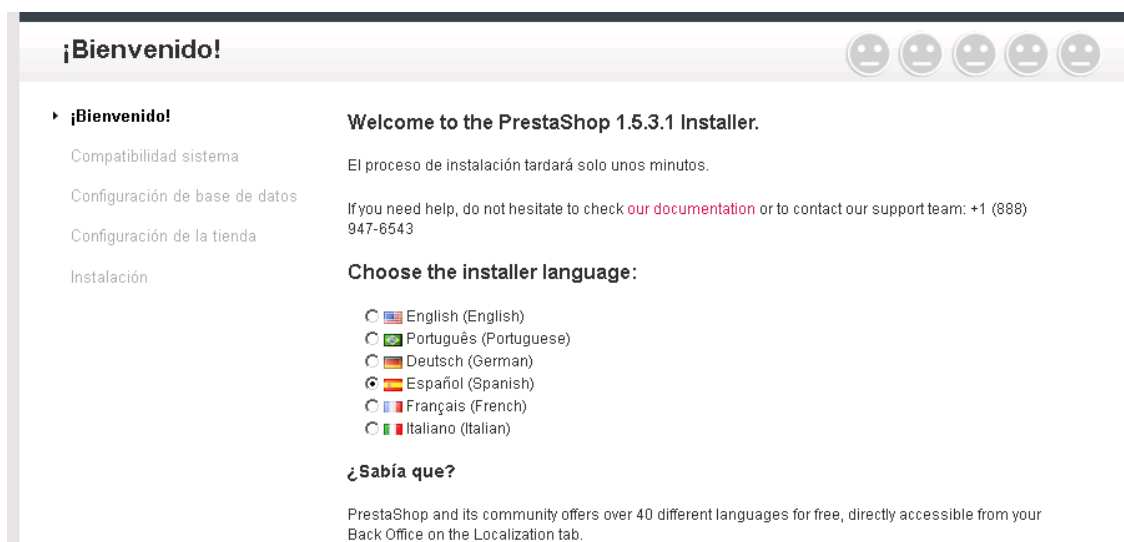
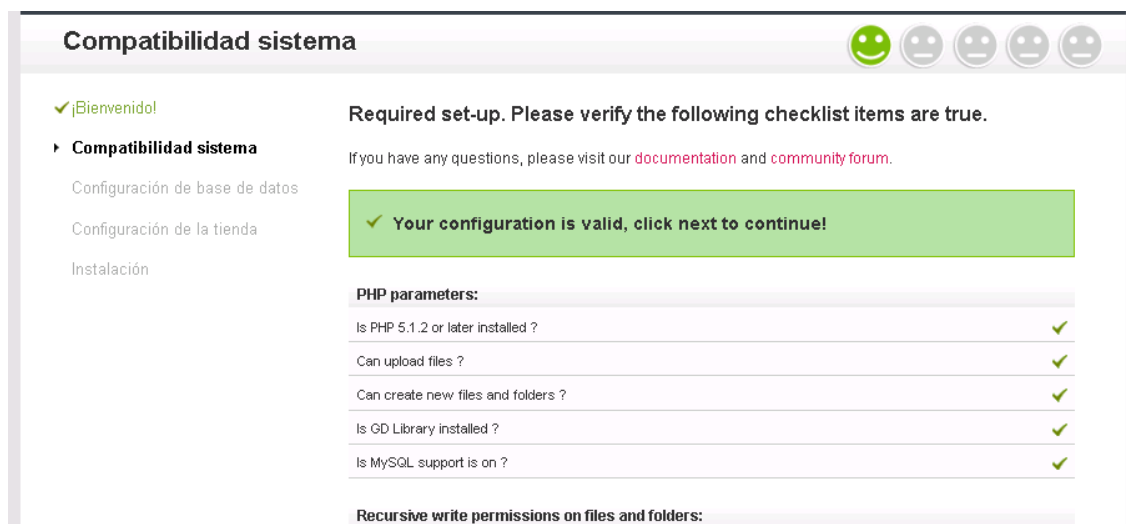
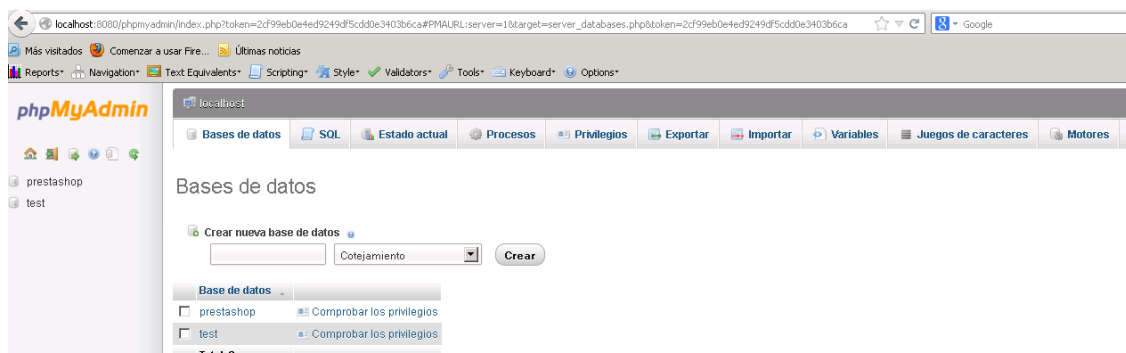


Ilustración 283 Instalación Prestashop -Paso 1

Seguimos la instalación guiada. En el segundo paso comprobamos que la compatibilidad de nuestro sistema es el adecuado para realizar la instalación

*Ilustración 284 Instalación Prestashop -Paso 2*

En el siguiente paso es necesario crear la base de datos. Para ello entramos en phpMyAdmin (“http://localhost:8080/phpmyadmin”) que nos facilitará la administración de la nueva base de datos. Accedemos a la opción Base de Datos y creamos una nueva con el nombre de “prestashop”

*Ilustración 285 Instalación Prestashop -Paso 3*

Una vez creada configuramos los parámetros en el instalador de prestashop.

Configuración de base de datos

- ✓ ¡Bienvenido!
- ✓ Compatibilidad sistema
- ▶ **Configuración de base de datos**
- Configuración de la tienda
- Instalación

Configure su base de datos llenando los siguientes campos:

You have to create a database, help available in [our online documentation](#).

Database server address:
If you want to use a different port, add :XX after your server address where XX is your port number.

Nombre de la base de datos:

Inicio de la base de datos:

Contraseña de la base de datos:

Tipo de base de datos:

Prefijo de las tablas:

Verify now!

✓ Database is connected

Ilustración XX Instalación Prestashop -Paso 4

El siguiente paso es dar nombre a la tienda y poner nuestros datos personales. Si todo está correctamente configurado empezará la instalación de la aplicación web.

Instalación

- ✓ ¡Bienvenido!
- ✓ Compatibilidad sistema
- ✓ Configuración de base de datos
- ✓ Configuración de la tienda
- ▶ **Instalación**

Create database tables ...

12%

✓ Create settings.inc file

Ilustración 286 Instalación Prestashop -Paso 5

Una vez instalado se borra el directorio /install y se renombra el directorio /admin, en este caso el nombre asignado es /admin1.

Finalmente la web quedará desplegada en la URL local <http://localhost:8080/prestashop/index.php>.

Para la administración de la página podremos entrar en : <http://localhost:8080/prestashop/admin1/index.php>

14.5.2 Sistema de directorios

Una vez que se descomprime la aplicación nos encontramos con el siguiente sistema de directorios:

Directorio	Descripción
Admin	Contiene todos los archivos relacionados con panel de administración de la tienda, cuando realizamos la instalación de la tienda online, se nos obliga a renombrar para poder acceder al panel. Prestashop nos recomienda que aparte de la protección que incluye Prestashop incluyamos la protección en el htaccess, bueno...., es una forma más de proteger nuestra información.
/cache	Contiene los archivos temporales que usa Prestashop para no sobrecargar el servidor.
/clases	Contiene los archivos que controlan todos los objetos y las clases de las mismas.
/config	Contiene los archivos que corresponden a la configuración de Prestashop, no es recomendable manipularlos si no tenemos conocimientos avanzados de la herramienta, algunos de ellos los podremos controlar desde el panel de administración de Prestashop, por ejemplo la caché.
/controllers	Contiene todos los archivos relacionados con el controlador de Prestashop MVC (Modelos, Vista, Controlador) , en definitiva la arquitectura de Prestashop, nunca se deben modificar los archivos de esta carpeta, debemos aprender a desarrollar nuestras necesidades fuera de ellas, mediante clases o mediante nuevos controladores. Si hemos modificado esta carpeta y actualizamos la herramienta nos podemos encontrar que todos los cambios se pierdan
/css	Contiene todos los archivos de estilos que no pertenecen a las plantillas que se usan en la parte pública de la tienda, generalmente son los archivos del backend.
/docs	Contiene los archivos referentes a la documentación de Prestashop, se recomienda borrarla una vez instalada la tienda.
/download	Es la carpeta donde se guardan los archivos pertenecientes a la productos virtuales o que se pueden descargar, por ejemplo, mp3, pdf, etc...
/img	Contiene todas las imágenes que trae por defecto Prestashop y de los productos, categorías u otras imágenes que se usan en la tienda, pero no contiene imágenes relativas a la plantilla que se esté usando en la tienda. En la subcarpeta /p, se almacenan las imágenes de los productos y en la subcarpeta /admin las imágenes del administrador.
/install	Contiene todos los archivos necesarios para realizar la instalación de Prestashop, se recomienda borrar todos los archivos como medida de seguridad.
/js	Contiene todos los archivos JavaScript que son necesarios para que Prestashop funcione correctamente, tanto en el panel de administración como en la parte pública de la tienda, por ejemplo la librería JQuery, pero no se incluyen los JavaScript de la plantilla que esté funcionando ya que estas se incluyen en el mismo directorio de la plantilla.
/localization	Contiene los archivos referentes a la localización, como pueden ser los idiomas, monedas, impuestos, estados, pesos, etc.
/log	Contiene los log de Prestashop.
/mails	Contiene todas las plantillas ordenadas por idiomas de los emails que se envían para notificar a los clientes, cuando se realiza un pedido, cuando se cambia de estado, cuando se registran en la tienda, etc...
/modules	Contiene todos los módulos de prestashop que se usan en la tienda, para instalar los módulos a través de un ftp , este es el lugar donde deben estar ubicados.
/override	Esta es la carpeta desconocida por la mayoría, y es la carpeta donde podremos insertar nuestras modificaciones del core de prestashop, podremos anular, modificar o crear nuevas clases o controladores a las que trae por defecto prestashop, esta función apareció con la versión de Prestashop 1.4.
/themes	Donde insertaremos todas las plantillas que queramos usar en nuestra tienda Prestashop.
/tolos	Otra de las carpetas importantes de Prestashop, aquí se incluyen todas las herramientas o librerías externas a Prestashop, por ejemplo Smarty, FPDF, Swift o PEAR XML.
/translations	Contiene subcarpetas ordenadas por idiomas donde podremos encontrar todas las traducciones de la tienda, no es recomendable editar estos textos directamente sobre estos archivos ya que cualquier error puede originar que no se vean, lo mejor editarlas desde el panel.
/upload	En esta carpeta se guardan todos los archivos que los clientes suben para poder personalizar los productos que así lo permitan.
/webservice	Otra de las nuevas funcionalidades de Prestashop, contiene los archivos necesarios para poder trabajar con la API de Prestashop, y hasta aquí puedo leer...

14.5.3 Arquitectura del sistema

PrestaShop es un software basado en una arquitectura tres capas, como nos muestra el siguiente esquema:

Arquitectura de 3 capas de PrestaShop

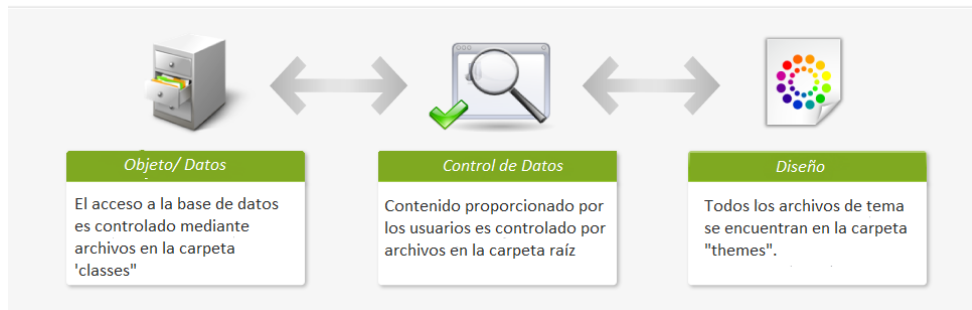


Ilustración 287 Arquitectura de Prestashop

Sigue un modelo MVC ("Modelo-Vista-Controlador") en el que las ventajas son varias de cara a posibles desarrolladores:

- La posibilidad de comprender fácilmente el código fuente del software,
- Modificaciones rápidas y simples para los desarrolladores,
- La posibilidad para los grafistas/integradores de trabajar en la carpeta temas sin necesidad de comprender ni leer el código PHP,
- La posibilidad para los desarrolladores de preparar datos y módulos complementarios para el integrador

Además Prestashop se basa en torno a un principio de módulos que facilita la personalización de la herramienta, su extensión y utilización.

Uno de los principales módulos es el módulo de migración de datos desde tiendas de Oscommerce a tiendas de prestashop, que puede venir bien para gente que desee migrar de herramienta.

Otra forma alternativa de migrar los datos de una manera sencilla es importando todo el catálogo de productos desde archivos CSV/Excel .

14.5.4 Factores Claves:

- Permite integrar varios idiomas.
- Gestión de multitud módulos de pago. Es muy sencillo incorporar nuevos módulos ampliar la características de la tienda.
- Grupos de cliente integrado.
- Fácil instalación con la mayoría de opciones.
- Permite atributos, personalizable y sencilla de usar.
- Permite definir **productos físicos o virtuales** (descargas).

- Muy fácil de usar.
- Bajo consumo de CPU.
- Permite introducir códigos de barras
- Panel de administración muy intuitivo y sencillo.
- Es muy rápido.
- Prestastore: Tienda de módulos ya desarrollados.
- Prestashop 1.5 dispone de multitienda.
- La comunidad en española sigue creciendo muy rápido.
- Prestashop 1.5, versión para móviles tanto para Iphone como Android.
- Posibilidad de adquirir APP nativa para Iphone, Android o tablets.
- Arquitectura Modelo /Vista/Controlador tanto en front office como backoffice.
- Prestashop 1.5 ha mejorado considerablemente los envíos y almacenes.

14.5.5 Listado de Funcionalidades Completas de Prestashop

Administración de Catálogo	
Funcionalidad	Descripción
Elección de condiciones del producto	Permite elegir entre varias opciones disponibles para la administración de los productos
Cargar archivos CSV de productos	Permite cargar la lista de productos a partir de un archivo en formato .CSV
Crear un modo de catálogo	Permite crear uno o varios catálogos con la lista de productos deseada y varias opciones de configuración.
Cargas masivas	Además de importar productos permite cargar otro tipo de entidades como categorías, combinaciones, clientes, direcciones...
Seleccionar productos no disponibles	Permite hacer la compra con productos que no están en stock pero que pueden reabastecerse en su momento
Navegación por capas	Permite navegar por diferentes capas del catálogo gracias a las clasificaciones y categorías
Mostrar precios por unidad en los productos	Permite calcular unitariamente los precios de cada uno de los productos (a algunos se les pueden aplicar rebajas, ofertas...).
Gestión de códigos UPC	Permite gestionar códigos de barra de tipo UPC.
Archivos de inventario	Permite gestionar el inventario y generar informes
Notificación de reabastecimiento	Cuando un producto se termina permite configurar la notificación de reabastecimiento del producto
Cantidad mínima de compra	Permite configurar cantidades mínimas de compra en un carrito.
Categorías y subcategorías ilimitadas	Permite categorizar y subcategorizar según se crea conveniente la lista de productos de la tienda
Atributos ilimitados (tallas, colores, etc.)	Permite añadir los atributos necesarios a cada uno de los productos disponibles
Combinaciones de producto ilimitadas	Permite crear combinaciones de los productos (ejemplo en una tienda de zapatos podemos tener la necesidad de crear 44 tallas con 20 colores diferentes y con ello crear las combinaciones pertinentes)
Generador de combinaciones automatizado	Además de poder crear de forma automática la combinación de los productos, permite hacerlo de manera automática.
Características de producto ilimitadas	Permite incluir todo tipo de características a los

	productos.
Avisos de catálogo(páginas y productos inactivos)	Permite crear diferentes avisos en el catálogo.
Categorías segmentadas a grupos específicos	Permite segmentar las diferentes categorías de los productos
Paquetes de productos y complementos	Permite agrupar por paquetes los diferentes productos y complementos
Precios decrecientes en porcentaje	Permite listar los diferentes productos de más a menos precio
Precios decrecientes en importe fijo	Permite listar los diferentes productos en función al importe fijo
Referencias del producto (ISBN-EAN13)	Permite incluir referencias ISBN a los productos
Facturas y recibos de entrega PDF	Permite exportar las facturas a formato pdf
Cargar lista de productos fabricados	Permite cargar listas de proveedores y fabricantes
Gestión de fabricantes	Permite la administración de los diferentes fabricantes de productos
Gestión de los proveedores	Permite gestionar los diferentes proveedores de productos
Personalizar productos individual o colectivamente	Permite personalizar un producto acorde con las necesidades requeridas o agrupar las características
Venta cruzada	Permite la venta cruzada (recomendaciones de productos asociados a los de la compra)
Filtrar producto por atributos, términos o datos estadísticos	Permite hacer búsquedas de productos filtrando por atributo, precio, color...
Editar etiquetas de producto	Permite crear etiquetas para los diferentes productos del catálogo
Gestión de devoluciones	Dispone de un sistema de gestión de devoluciones
Reporte de productos agregados a la lista de regalo	Permite gestionar los reportes de productos agregados a la lista de regalo
Venta de productos descargables: MP3s, softwares, etc.	Permite la venta de productos descargables , como pueden ser música, libros , software...
Exportar pedidos y facturas	Permite exportar pedidos y facturas a formato pdf
Elección de condiciones del producto	Permite seleccionar diferentes condiciones asociadas a un producto.

Visualización del producto	
Funcionalidad	Descripción
Comparar productos	Permite comparar entre los diferentes atributos o categorías de los diferentes productos
Agregar a la lista de regalo	Permite agregar productos a listas regalo
Enviar lista de regalo por e-mail	Permite enviar la lista regalo por mail al destinatario deseado
Imágenes ilimitadas con redimensionamiento automático	Permite incorporar la cantidad de imágenes deseadas a un producto y las miniaturas que se muestran se generan automáticamente.
Múltiples imágenes por producto	Permite agregar diversas imágenes a un producto
Zoom-in a las imágenes de productos	Dispone de un sistema de zoom para las imágenes de los productos
Protección filigrana en las imágenes de productos	Permite incorporar filigranas o marcas de agua a las imágenes de los productos para evitar que estés se usen en otras webs
Búsqueda rápida de productos	Permite hacer diversidad de búsquedas de los productos
Comentarios de los clientes acerca de los productos	Permite que los diferentes clientes puedan añadir comentarios a los diferentes productos
Reseñas de los clientes	Permite incluir distintivos a los diferentes clientes del sistema
Búsqueda por alias	Permite la búsqueda por alias de los clientes
Búsqueda Ajax	Permite hacer búsquedas Ajax de los productos
Visualización de los precios: con o sin IVA	Permite visualizar los diferentes precios de los productos con o sin IVA

Elección de número de productos a mostrar por página	Permite configurar el número de productos que se muestran por página
Notificaciones por e-mail de productos 'out-of-stock'	Permite incluir alertas de notificación por e-mail cuando un producto ha sido reabastecido y ya vuelve a haber stock.
Mostrar cantidades disponibles	Permite mostrar que cantidad de cada producto está disponible o en stock
Navegación por Capas	Permite ver los productos agrupados por capas o categorías
Clasificación de productos por pertinencia, precio, etc.	Permite la visualización del producto conforme a varias clasificaciones: categoría, precio...
Imprimir las fichas de los productos	Permite al cliente imprimir la ficha del producto deseado
Adjuntar documentos a los productos (guías de usuario, etc.)	Permite adjuntar a la tienda documentos como guías de usuario...
Agregar producto a una lista de regalo	Permite agregar un producto a una lista regalo
Mostrar productos en la misma categoría	Permite visualizar los productos por categorías
Exportar productos a los buscadores más importantes	Permite exportar los productos a los buscadores más importantes para que los clientes puedan encontrarlos de manera sencilla y rápida.
Fijación de precios a varios niveles	Permite fijar precios por diferentes niveles (cantidad comprada, descuentos...)
Construir atributos de productos	Permite personalización de los atributos de los productos
Marcar productos	Permite marcar productos conforme a normas establecidas
Estadísticas de página de producto: Fecha de compra última	Permite crear estadísticas de un producto por fecha de compra
Mostrar el contenido de cesta	Permite mostrar todos los productos incorporados al carrito de compra
Mostrar categorías de producto	Permite mostrar las diferentes categorías a las que está asociado un producto

Administración del sitio

Funcionalidad	Descripción
Módulo para importar tiendas	Dispone de un módulo para importar tiendas completas (productos, categorías...)
Compatible con todos los navegadores web	Es compatible con todos los navegadores web (IE Explorer, Mozilla Firefox...)
Favicon personalizable	Permite personalizar el favicon que se desee agregar al sitio web
Integración con un CRM o ERP (Webservice)	Permite integrar la herramienta con un CRM (Customer relationship management) o un ERP (Enterprise Resource Planning)
Función de ayuda en el back office	Dispone de sistema de ayuda para la administración de la tienda
Personalizar accesos directos en el back office	Permite personalizar los accesos directos a las configuraciones que se usen más a menudo
Búsqueda universal en el back office	Permite hacer todo tipo de búsquedas en el back office
Editor de texto	Dispone de un editor de texto
Maneje tipos de letra extravagantes	Dispone de diferentes fuentes de letra
E-mails SMTP: envíe e-mails por vía de un formulario de contacto	Dispone de un servidor SMTP para el envío de e-mails

Optimización para los motores de búsqueda

Funcionalidad	Descripción
Amigable a los buscadores	Es amigable para los buscadores más importantes
Reescribe URL	Permite la redirección automática
URL simplificado para cada producto	Permite simplificar automáticamente la URL de cada

	uno de los productos
Mapa de sitio de Google	Permite enlazar directamente con Google
Mapa de sitio auto generado	Genera automáticamente el mapa del sitio web
Archivos robot.txt automáticos	Permite escribir automáticamente en el archivo robot.txt para que los crawlers de búsqueda encuentren los productos de una manera más optima
Archivos .htaccess automáticos	Permite hacer .htaccess automáticos
Enlaces permanentes	Permite crear enlaces fijos de los productos
Edite etiquetas de producto	Permite editar las etiquetas de búsqueda de un producto
Etiquetas de título, meta etiquetas, meta descripción, etc.	Permite editar título, meta etiquetas, meta descripción, etc.
En conformidad con W3C	Dispone de conformidad W3C
Flujo RSS	Dispone de RSS
Nube de palabras	Permite crear nubes de tags
Reduzca el tiempo de carga de los productos	Permite reducir el tiempo de carga de los productos (varias opciones de configuración)
URL canónico: previene el contenido duplicado	Dispone de URLs únicas para cada producto evitando el contenido duplicado

Finalización de compra	
Funcionalidad	Descripción
Finalización de compra de una Página	PrestaShop ofrece una finalización de compra en una sola página
Finalización de compra como invitado	Permite hacer compras como invitado
Renueve un pedido en un 'click'	Permite refrescar el pedido
Personalice el estado del pedido	Permite dar diferentes estados a los pedidos
Mensajes del pedido predefinidos	Dispone de un conjunto de mensajes predefinidos para los pedidos
Establezca la cantidad mínima del pedido	Permite establecer el umbral mínimo de pedidos de una compra
Ordena productos 'out-of-stock'	Permite hacer pedidos fuera de stock
Facturas y recibos de entrega PDF	Permite generar facturas en formato PDF
Ofertas especiales	Permite crear ofertas
Cesta Ajax	La cesta de la compra esta implementada con Javascript y Ajax
Capacidad para venta sugestiva	Tiene capacidad de venta sugestiva, esto es una forma sutil de vender. El cliente compra lo que sabe que tenemos, la venta sugestiva trata de completar esa compra con algo que pueda resultar atrayente.
Envoltura de regalo	Tiene la opción de envoltura de regalos
Envoltura de regalo facturable	Tiene la opción de envoltura facturable de regalos
Cesta de compras guardada, configurada por vía de tiempo de expiración	Permite guardar la cesta de la compra y configurar el tiempo de expiración
Coloque condiciones de venta al finalizar la compra	Permite configurar las condiciones al finalizar la venta
Llenado automático de formularios con ReversoForm	Dispone de autocompletado para algunos formularios

Envío	
Funcionalidad	Descripción
Transportistas ilimitados	Permite incluir una lista ilimitada de transportistas disponibles
Destinos ilimitados	Permite incluir listas de destinatarios
Descuentos de envío	Permite configurar descuentos para el envío
Transportistas por zonas o provincias	Permite crear configuraciones para zonas determinadas o provincias
Seguimiento del paquete	Dispone de un sistema de seguimiento para los paquetes

Notificaciones por e-mail del envío	Permite notificar el envío del pedido
Recoger en la tienda	Dispone de la opción de recoger el pedido en la tienda
Cargos por manejo	Permite añadir ciertos cargos por el manejo del pedido (si supera un tamaño, una dimensión, llevarlo al mismo domicilio...)
Posibilidad de regalar los gastos de entrega	En las compras regalo permite regalar también los gastos de envío
Facturación por precio o peso	Permite hacer la facturación por el peso del pedido
Domicilios de facturación y envío diferentes	Permite enviar un pedido a un domicilio diferente del domicilio de facturación
Estimados por cargos postales	Permite hacer estimaciones de gastos de envío por código postal
USPS, FedEx, UPS, CanadaPost, etc.	Dispone de una lista predeterminada de transportistas o agencias dedicadas a la entrega de productos
Comparación de precios de envío, con Envoimoincher	Permite comparar los precios de envío
Envío con Mondial Relay	Permite el seguimiento del pedido con Mondial Relay
Entrega personal con Dejala	Permite entrega personal con Dejala

Marketing	
Funcionalidad	Descripción
Agregar videos con Treepodia	Permite incorporar videos de promoción
Enviar e-mails con Mailjet	Permite el envío de emails publicitarios
Redirigir a los visitantes, con Critéo	Permite redirigir a los usuarios para que conozcan la tienda
Google Adsense	Dispone de posibilidad de configurar Google Adsense. Esto es un programa gratuito que posibilita que los editores online aumenten sus ingresos mostrando los anuncios relevantes en una amplia variedad de contenido online
Exportar productos a los buscadores, con Shopping Flux	Permite exportar publicidad a los buscadores
Exportar productos a Twenga	Permite exportar productos a Twenga (tienda online)
Exportar productos a eBay	Permite exportar productos a eBay(tienda online)
Seguimiento automatizado de e-mails	Dispone de un servicio automatizado de e-mails
Suscripción al boletín de noticias	Permite la suscripción a boletines de noticias
Programa de Lealtad	Dispone de un programa lealtad para premiar a los clientes leales a la tienda
Programa 'recomienda a un amigo'	Dispone de un sistema para que clientes puedan recomendar la tienda a amigos
Productos combinados	Permite combinar la venta de varios productos en una única oferta
Productos observados recientemente	Permite obtener la lista de productos más visitada
Cupones/Vales	Permite creación de cupones o vales de descuento
Integrar Google Adwords	Permite crear anuncios en Google
Programa de Afiliados	Permite crear programas para afiliados
Videos presentación de producto	Permite crear videos publicitarios para productos
Cupones de franqueo	Permite crear cupones de franqueo
Herramientas promocionales	Dispone de otras herramientas promocionales
E-mails SMTP	Permite el envío de E-mails mediante SMTP

Inicio de sesión del cliente	
Funcionalidad	Descripción
Inicio de sesión del cliente	Dispone de un sistema de logueado fácil de usar y seguro
Cuentas de clientes	Permite crear cuentas cliente propias
Posibilidad de enviar mensajes por vía de su cuenta	Desde la cuenta de cada uno de los clientes permite el envío de mensajes a otros usuarios

Gestión de devoluciones	Dispone de un sistema de gestión de devoluciones de los productos de los que el cliente no está satisfecho
Chat en vivo con Advize	Dispone de un chat en vivo desde el cual los usuarios pueden consultar dudas, problemas de instalación...

Traducciones	
Funcionalidad	Descripción
Numerosas traducciones disponibles, más de 40	Permite internacionalizar a la web a más de 40 idiomas diferentes
Ubicación geográfica	Permite seleccionar el idioma por ubicación geográfica
Posibilidad de elegir el idioma deseado	Permite seleccionar el idioma en el que se quiere presentar la web
Importar/exportar paquetes de traducción	Dispone de módulos o paquetes externos que permiten añadir nuevas traducciones
Herramientas de traducción en línea	Dispone de traductor en línea

Seguridad	
Funcionalidad	Descripción
Analizar las órdenes, con Fia-Net	Analiza las órdenes con FIA-NET, ésta asegura un clima de confianza entre compradores y vendedores en Internet. Su propósito es ayudar a los compradores a distinguir a los comerciantes más confiables en la web al mostrar su nivel de servicio.
Múltiples usuarios	El sistema es multiusuario
Establecer permisos de seguridad para usuarios	Permite configurar el nivel de seguridad para los usuarios de la web
Modo en mantenimiento	Dispone de un sistema de mantenimiento
En conformidad con PCI	Dispone de un sistema de seguridad PCI que permite mantener la seguridad en las cuentas de usuario
En conformidad con SSL	Opera bajo el protocolo SSL (<i>Secure Sockets Layer</i>) que encripta la información.
Back office segura	El sistema de administración de la tienda es seguro
Establecer la expiración de la contraseña	Permite establecer tiempos de expiración de contraseñas
Encriptación de contraseña y cookie	Encripta la contraseña y las cookies
Resistencia a los ataques de tipo XSS, CSRF, inyecciones SQL,	Dispone de sistemas de seguridad contra ataques de tipo XSS, CSRF, SQL injection...
Bloqueo a las inyecciones de encabezado de e-mail	Evita ataques de inyecciones de encabezado de email
Encriptación de cookies	Dispone de un sistema de encriptación de cookies
Bloqueo de las tentativas de recuperación de contraseñas repetidas	Dispone de un sistema contra ataques de tipo fuerza bruta
Seguimiento del fraude en línea con Secuvad	Permite la detección y el seguimiento de tentativas de fraude en línea

Localización/Impuestos	
Funcionalidad	Descripción
Tasas ilimitadas	Permite crear tasas a medida
Formato del domicilio de acuerdo al país	Permite crear formatos de direcciones a medida de acuerdo con el país donde esté el usuario
Divisas ilimitadas	Dispone de un conjunto de divisas por defecto y permite incorporar nuevas divisas conforme a las necesidades de la tienda
Sincronización de los tipos de cambio	Si el sistema opera en diferentes países permite sincronizar los tipos de cambio de divisas
Tasas por estado, provincia o país	Permite establecer tasas dependiendo del estado ,provincia o país
Husos horarios	Permite configurar el huso horario

Formateo de divisas	Permite el formateo de divisas (incluir los números decimales deseados, redondeo)
Configurar Eco Tax	Permite configurar la ecotasa, en función de los niveles establecidos en ese momento

Análisis e informes	
Funcionalidad	Descripción
Exportar listas de boletín	Permite exportar los boletines a informes pdf
Notificación de novedades PrestaShop en el back office	Dispone de un sistema de notificación de novedades para el back office
Seguimiento a la actividad del visitante	Permite hacer un análisis de la actividad de los visitantes y compradores de la web
Ver perfiles de clientes	Permite ver los perfiles de los clientes de la tienda
Informes de pedidos y ventas	Permite crear informes para pedidos y ventas
Estadísticas de afiliados	Dispone de un sistema de creación de estadísticas para afiliados
Estadísticas del boletín	Permite hacer estadísticas para el boletín
Visualizaciones de cestas Google	Permite visualizar las cestas de Google
Integrado con Google Analytics	Está integrado con, Google Analytics , servicio gratuito de <u>estadísticas de sitios web</u> .
Reportes de palabras claves	Permite crear informes con la lista de palabras clave o nube de tags
Gestión de “páginas no encontradas”	Permite analizar las páginas no encontradas por los clientes
Ubicación geográfica	Permite establecer la ubicación geográfica en los informes
Gestión CPC	Permite gestionar el CPC, esta es una de las herramientas básicas para controlar el presupuesto de la campaña.
Búsquedas en la tienda	Permite hacer búsquedas en la tienda
Estadísticas de transportistas	Dispone de un sistema de creación de estadísticas para transportistas
Ver mejores proveedores	Dispone de un sistema de creación de estadísticas para proveedores
Categorías, cupones, productos de mejor desempeño	Dispone de un sistema para analizar las ventas por categorías, cupones...
Estadística de interacción de clientes	Permite hacer análisis estadísticos sobre la interacción de los clientes

En la versión 1.5 además incluye funcionalidades multitienda, nuevas plantillas estándar de interfaz de usuario...

14.6 oScommerce

OsCommerce es una solución de tienda en línea desarrollada bajo licencia pública general GNU¹. OsCommerce se inició en el 2000 y desde entonces se ha convertido a través de distintas versiones en una herramienta robusta. Para ello se tiene un equipo que se encarga de las funcionalidades y una comunidad de propietarios de la tienda, desarrolladores y proveedores de servicios que se centran en las características adicionales.

¹ Para mayor información relacionada con la licencia ver <http://www.gnu.org/licenses/quick-guide-gplv3.html>

La tienda está compuesta de dos módulos principalmente, el catálogo (front) donde los clientes de cada tienda pueden navegar para conocer y comprar los distintos productos y una herramienta de administración (backend) donde el administrador puede establecer distintas opciones relacionadas con el funcionamiento y la imagen de la tienda.

Actualmente osCommerce maneja dos versiones principalmente: La primera es “osCommerce Online Merchant v3.0” recomendada para las personas que están interesadas en generar nuevos desarrollos y adelantarse a las características que vendrán en las nuevas versiones estables; la segunda es “osCommerce Online Merchant v2.3” La cual es la versión estable más reciente por lo tanto se recomienda para las personas que vayan a crear una tienda, sobre esta versión se harán descripciones funcionales y técnicas en el desarrollo de este documento.

14.6.1 Instalación

Para la instalación de osCommerce se debe realizar la descarga desde la página <http://www.oscommerce.com/Products>, esta carpeta descargada debe descomprimirse en la carpeta de desarrollos configurada en el servidor web que se tenga instalado y luego de eso se debe ingresar a <http://localhost/oscommerce-3.0.2/oscommerce/>², aparecerá la pantalla que se muestra a continuación:

Welcome To Your New Online Store! Language:

Thank you! for choosing osCommerce Online Merchant to power your new online store! We welcome you to join and participate in our growing community of store owners, developers, and service providers.

Please visit our [Support](#) page if problems are experienced with the installation or operation of your online store.

Server Recommendations	
PHP Version	5.3.8 ✓
PHP Settings	
register_globals	Off ✓
magic_quotes_gpc	Off ✓
file_uploads	On ✓
session.auto_start	Off ✓

Server Requirements

- PHP v5.3+ (with PDO MySQL extension)
- MySQL v4.1.13+ or v5.0.7+

Installation Type

Click on New Install to begin the installation procedure.

[▶ New Install](#)

Ilustración 288 Instalación osCommerce – Paso 1

Luego de dar clic en el botón new Install aparecerá la pantalla donde se configura la base de datos

² La palabra localhost puede variar de acuerdo a la dirección establecida para el servidor local

Step 1: Web/Database Server

The web server is responsible for serving the pages of the online store to visitors and customers.

The database server stores the content of the online store such as product information, customer information, and orders that have been made.

Please consult your server administrator if your database server parameters are not known.

Web Server

WWW Address

The web address to the online store.

Database Server

Database Server

The address of the database server in the form of a hostname or IP address.

Username

The username used to connect to the database server.

Password

The password that is used together with the username to connect to the database server.

Database Name

The name of the database to hold the data in.

Database Server Port

The database server port number to connect to. (If unsure please leave this value empty)

Database Type

The database server software that is used.

Database Table Prefix

The prefix to use for the database tables.

Ilustración 289 Instalación osCommerce – Paso 2

Finalmente se deben llenar los campos propios de la tienda

Step 2: Online Store Settings

Please enter the name of your online store and the contact information for the store owner.

The administrator username and password are used to log into the Administration Tool.

Online Store Settings

Store Name

The name of the online store that is presented to the public.

Store Owner Name

The name of the store owner that is presented to the public.

Store Owner E-Mail Address

The e-mail address of the store owner that is presented to the public.

Administrator Username

The administrator username to use for the administration tool.

Administrator Password

The password to use for the administrator account.

Time Zone

The time zone to use for dates.

☒ **Import Sample Data**
Inserting sample data into the database is recommended for first time installations.

Ilustración 290 Instalación osCommerce – Paso 3

Luego de esto la herramienta muestra una pantalla donde se puede dirigir tanto al catálogo como a la herramienta de administración.

Finished!

The installation and configuration was successful!

Ilustración 291 Instalación osCommerce – Paso 4

Los requisitos de instalación básica son los siguientes:

Prerrequisitos técnicos	
Servidor Web	Cualquier servidor que soporte -PHP v5.3 o superior (con extension PDO MySQL)
Base de datos	v4.1.13+ or v5.0.7+
Editor de texto	Notepad++,writebox.. o cualquier editor de texto

14.6.1.1.1 Sistema de directorios

Una vez que se descomprime la aplicación nos encontramos con el siguiente sistema de directorios:

Directorio	Descripción
Catalog	Es la carpeta general, contiene a todas las carpetas que se describen en el resto de la tabla u además dentro de ella, sin subcarpeta, están contenidos todos los códigos fuente relacionados con el catálogo, es decir el front de la tienda que resulta visible para los clientes
Admin	Contiene todos los archivos relacionados con panel de administración de la tienda, principalmente se compone de códigos fuente e imágenes
Download	Contiene los archivos que se presentan en la tienda como productos descargables
ext	Contiene los módulos que sirven de enlace a otras páginas o tecnologías como por ejemplo las herramientas de pago
Images	Contiene las distintas imágenes que se muestran en la tienda
Includes	Contiene las funciones que son comunes o reutilizables
Install	Contiene los archivos relacionados con la interfaz que se muestra cuando se instala la herramienta, es decir la que permite establecer el nombre de la tienda, la base de datos etc.

14.6.2 Factores Clave:

- Permite integrar varios idiomas.
- Se pueden incorporar diferentes medios de pago
- Fácil instalación.
- Permite atributos, personalizable y sencilla de usar.
- Permite definir productos descargables.
- Panel de administración muy intuitivo y sencillo.

14.6.3 Listado de Funcionalidades Completas de Prestashop

Configuración	
Funcionalidad	Descripción
Gestionar administradores	Permite crear y eliminar administradores, igualmente permite habilitar htppassword para la protección de las claves
Gestión de logo	Permite establecer el logo que se usará para la tienda
Gestión de atributos de la tienda	Se pueden establecer distintos atributos propios de la tienda: nombre de la tienda, nombre del dueño de la tienda, dirección de la tienda etc.
Establecer valores mínimos y máximos	Permite configurar el tamaño mínimo y máximo de caracteres que son aceptados para distintos datos en la tienda
Gestionar imágenes	Permite definir el tamaño (ancho y alto) de las imágenes pequeñas, de cabecera y de categoría, al igual que habilitar el cálculo de tamaño de imágenes y habilitar si se muestran imágenes de error

Configurar envío y embalaje	Permite navegar por diferentes capas del catálogo gracias a las clasificaciones y categorías
Administrar listados	En esta funcionalidad se establecen las características que se quieren mostrar de cada producto y la forma como se quieren listar
Administrar inventario	Se puede definir diferentes características que ayudan al manejo de las existencias dentro de la tienda
Gestión de la cache	Permite habilitar y deshabilitar el uso de cache
Opciones de E-mail	Se establecen los lineamientos técnicos para el envío de correos electrónicos a los clientes a través de la tienda
Configuraciones técnicas de descarga	Permite configurar la forma como se realizan las descargas desde la tienda
Opciones de compresión	Permite activar o desactivar GZIP ³
Opciones de sesión	Permite definir el directorio del servidor donde se almacenaran las sesiones y establecer algunas revisiones de seguridad

Catálogo	
Funcionalidad	Descripción
Administrar categorías	Se puede crear eliminar y modificar categorías, las categorías y subcategorías
Administrar características de productos	Esta funcionalidad permite crear diferentes características de un producto genérico y asociar un valor a cada característica, para que el cliente pueda escoger las características en el momento de la compra
Administrar proveedores	Se pueden crear, eliminar, o modificar características de los fabricantes de productos
Administración de comentarios	Se pueden publicar, no publicar y modificar los comentarios hechos en la tienda
Configuración de oferta	A través de esta opción se marcan nuevos precios para los productos y la fecha durante la cual será válido ese precio
Configuración de productos esperados	Se publican los productos que están próximos a ser ofrecidos en la tienda y su fecha de lanzamiento

Módulos	
Funcionalidad	Descripción
Módulo para importar tiendas	Dispone de un módulo para importar tiendas completas (productos, categorías...)
Compatible con todos los navegadores web	Es compatible con todos los navegadores web (IE Explorer, Mozilla Firefox...)
Favicon personalizable	Permite personalizar el favicon que se desee agregar al sitio web
Integración con un CRM o ERP (Webservice)	Permite integrar la herramienta con un CRM (Customer relationship management) o un ERP (Enterprise Resource Planning)
Función de ayuda en el back office	Dispone de sistema de ayuda para la administración de la tienda
Personalizar accesos directos en el back office	Permite personalizar los accesos directos a las configuraciones que se usen más a menudo
Búsqueda universal en el back office	Permite hacer todo tipo de búsquedas en el back office
Editor de texto	Dispone de un editor de texto
Maneje tipos de letra extravagantes	Dispone de diferentes fuentes de letra
E-mails SMTP: envíe e-mails por vía de un formulario de contacto	Dispone de un servidor SMTP para el envío de e-mails

³ GZIP es una herramienta de compresión y descompresión de archivos con licencia GNU, para más información <http://www.gzip.org/>

Clientes	
Funcionalidad	Descripción
Administración de clientes	Lista los clientes registrados en la tienda, permite editar los datos de los cliente o ser redirigido a las órdenes del mismo
Administración de ordenes	Se ven y editan los datos asociados a las ordenes creadas por los clientes

Lugares/Tasas	
Funcionalidad	Descripción
Establecer países	Se puede agregar, editar o eliminar países
Establecer Zonas	Permite establecer zonas asociadas a cada país definido
Establecer Zonas de tasas	Permite crear zonas a las cuales aplica una tasa específica y agregar una descripción
Establecer tipos de tasas	Permite crear, editar o eliminar tipos de tasas
Establecer tarifas de tasas	Permite definir la tarifa (porcentaje) que se cobra para cada tipo de tasa

Localización	
Funcionalidad	Descripción
Definir monedas	Definir distintas monedas y la conversión entre ellas con base en una moneda definida como referencia.
Establecer lenguajes	Permite definir lenguajes, asociarles una imagen y un código
Configuración de estatus de ordenes	Se definen las etapas por las cuales pasan las ordenes desde el momento de la compra hasta la entrega

Reportes	
Funcionalidad	Descripción
Ver reporte de productos vistos	Lista los productos del más visto al menos visto y muestra la cantidad de vistas que ha tenido cada uno
Ver reporte de productos vendidos	Lista los productos del más vendido al menos vendido y la cantidad de ventas que ha tenido cada uno
Ver reporte de compras de clientes	Lista los clientes según la cantidad de dinero que han gastado en compras en la tienda y muestra dicha cantidad para cada cliente

Herramientas	
Funcionalidad	Descripción
Ver grabación de acciones	Se pueden ver las acciones que han realizado los usuarios de la tienda
Definir copia de base de datos	Se puede escoger el directorio donde se quieren hacer copias de seguridad de la base de datos, también permite realizar las copias y restaurarlas
Administrar los “banners”	Se pueden cambiar las imágenes grandes principales que se ven en la tienda
Controlar la cache	Se establecen diferentes configuraciones asociadas a la cache
Definir lenguaje	Se puede establecer otro lenguaje dentro de los que están configurados
Enviar E-mails	Se pueden enviar E-mails masivos a los compradores registrados en la tienda
Administrar anuncios	Se pueden crear diferentes anuncios para que sean

	publicados a los usuarios de la tienda
Ver seguridad de directorios	Permite ver la seguridad que está establecida para los distintos directorios de la tienda
Ver información del servidor	Muestra las configuraciones del servidor
Verificar versión de osCommerce	Se puede ver la versión de osCommerce que se está utilizando
Ver quien está en línea	Muestra los usuarios que se encuentran navegando en la tienda

14.7 Mapas de Conocimiento

A continuación se presentarán los mapas de conocimiento desarrollados para cada una de las plataformas

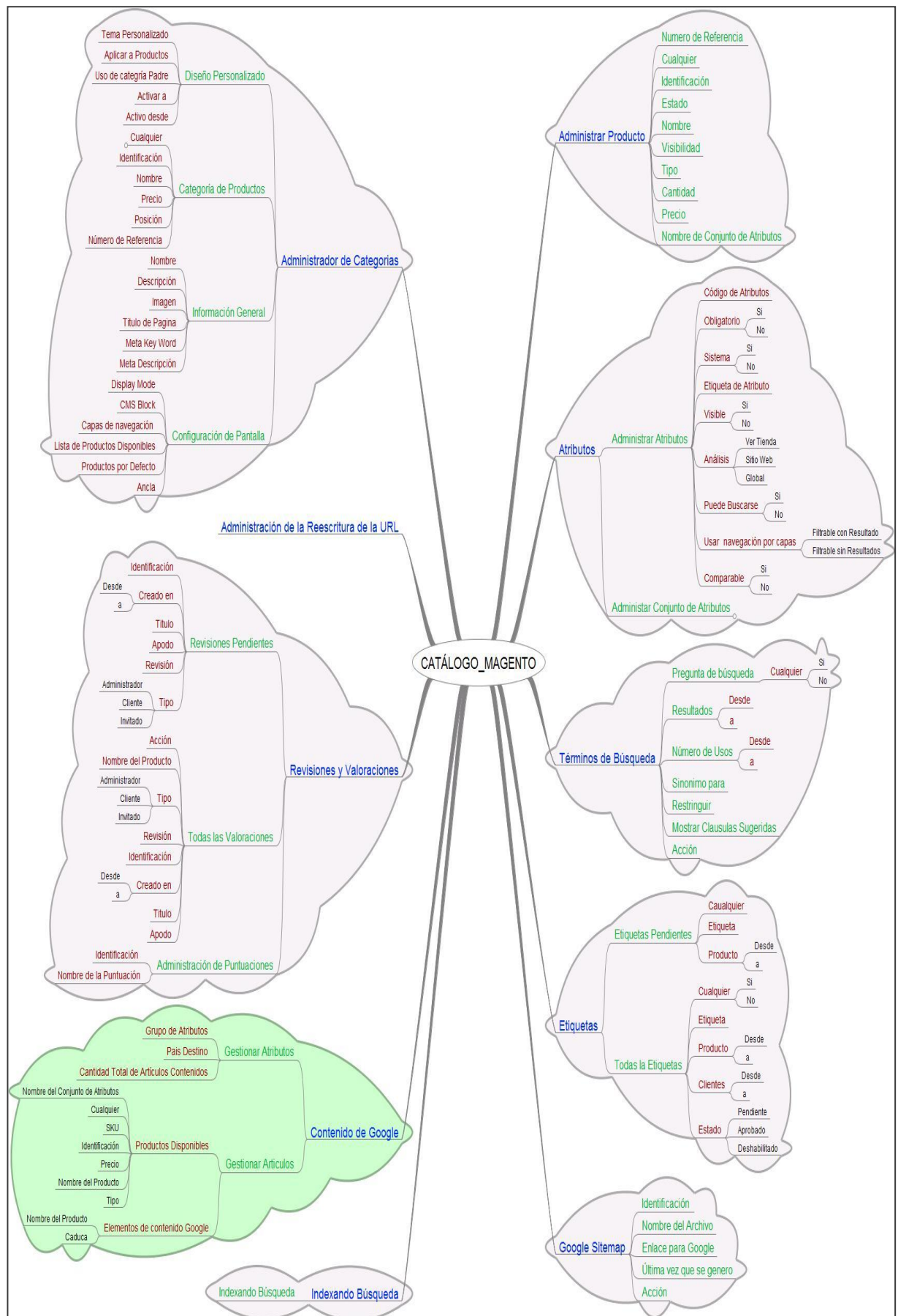


Ilustración 292-Catálogo Magento

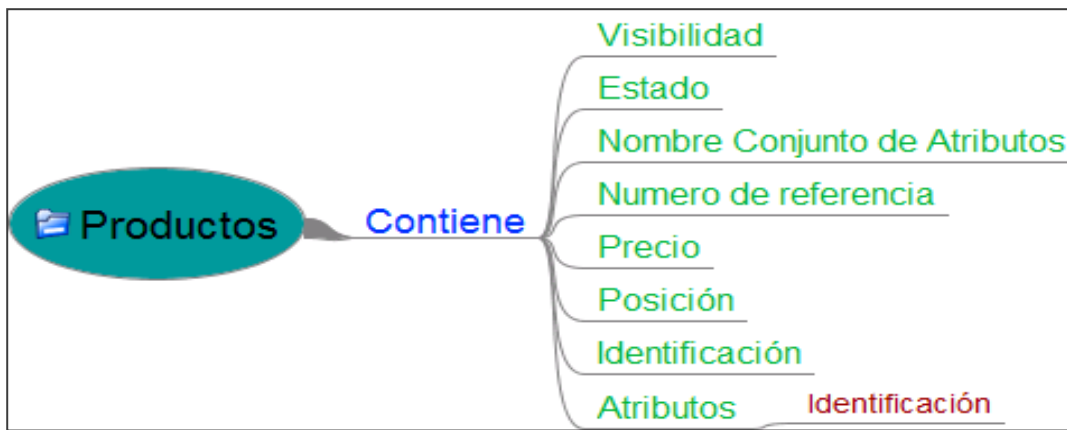


Ilustración 293-Producto Magento

14.7.1 Prestashop

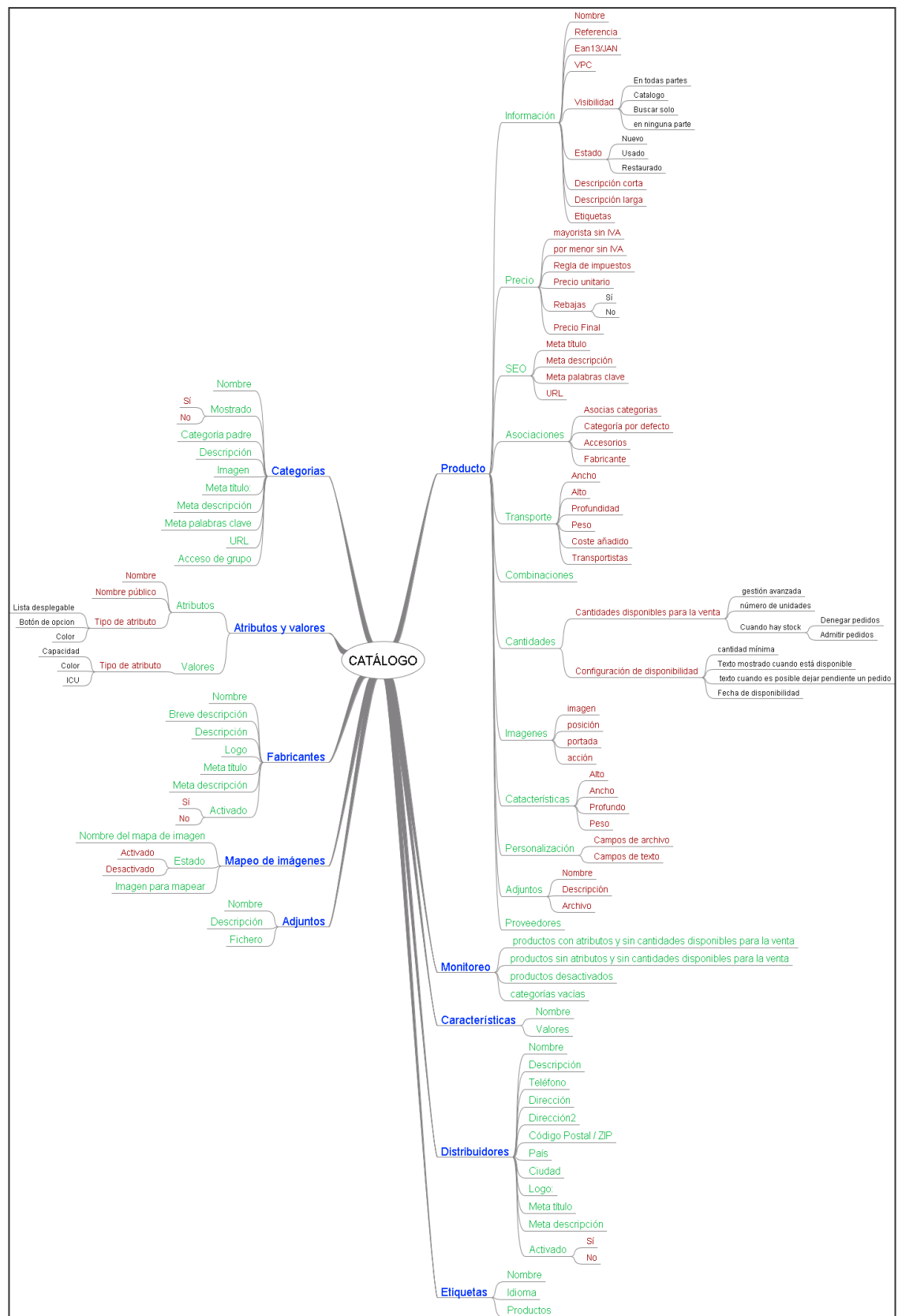


Ilustración 294 Catálogo Prestashop

14.7.2 OsCommerce

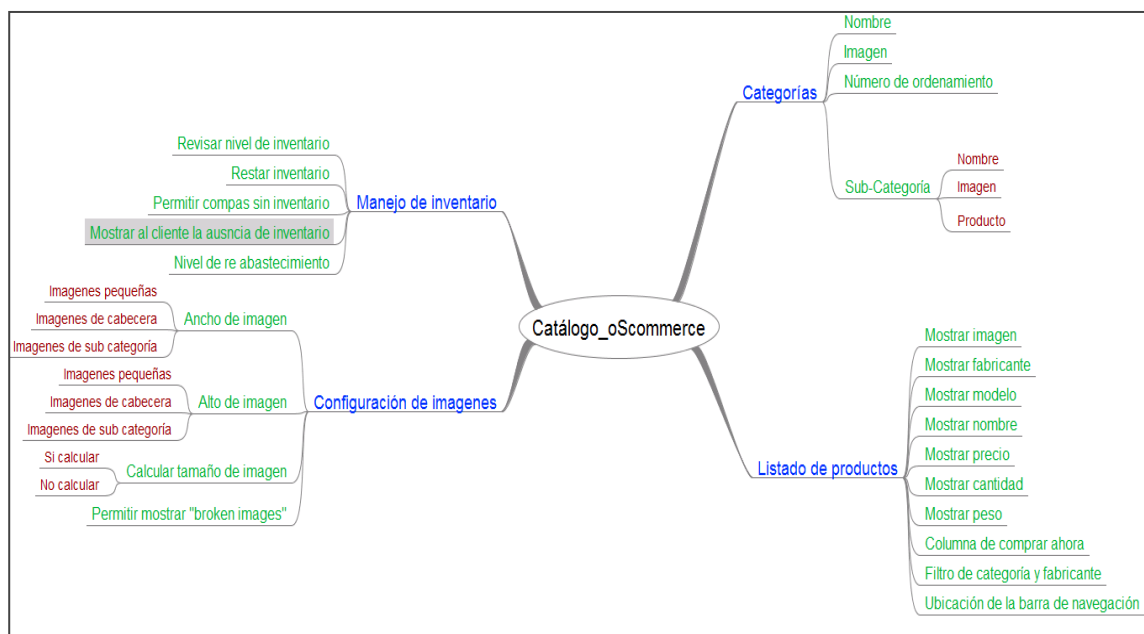


Ilustración 295 Catálogo OsCommerce

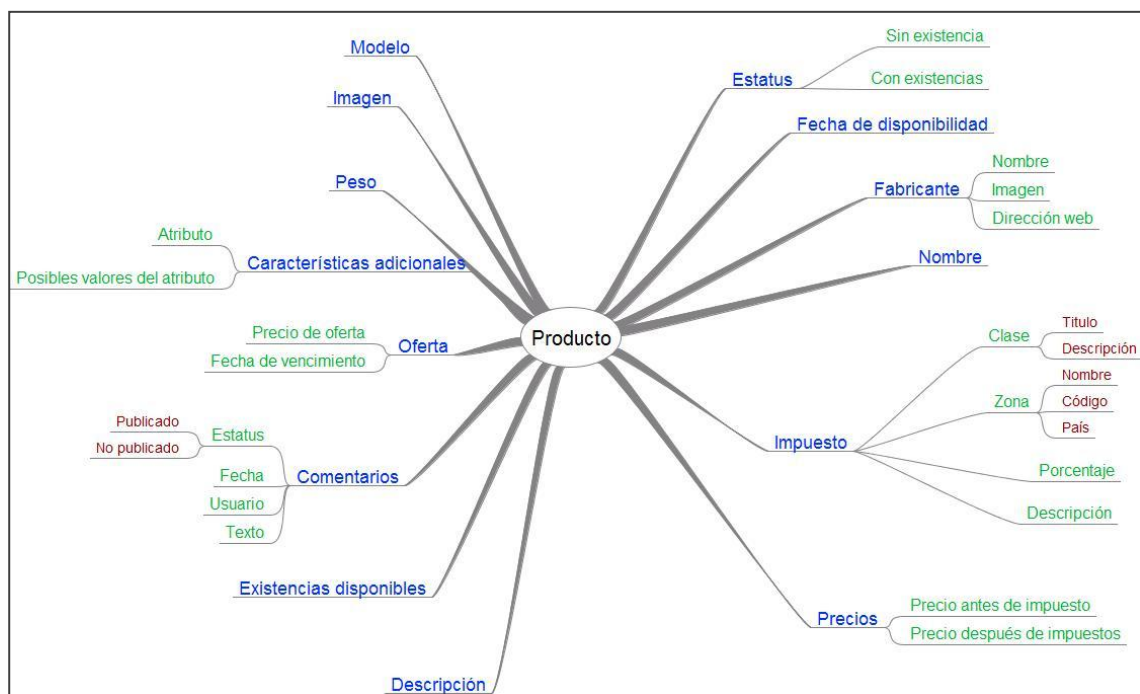


Ilustración 296 Producto osCommerce

14.8 Anexo II :Modelos E-R de las plataformas con las que se interopera

14.8.1 Magento

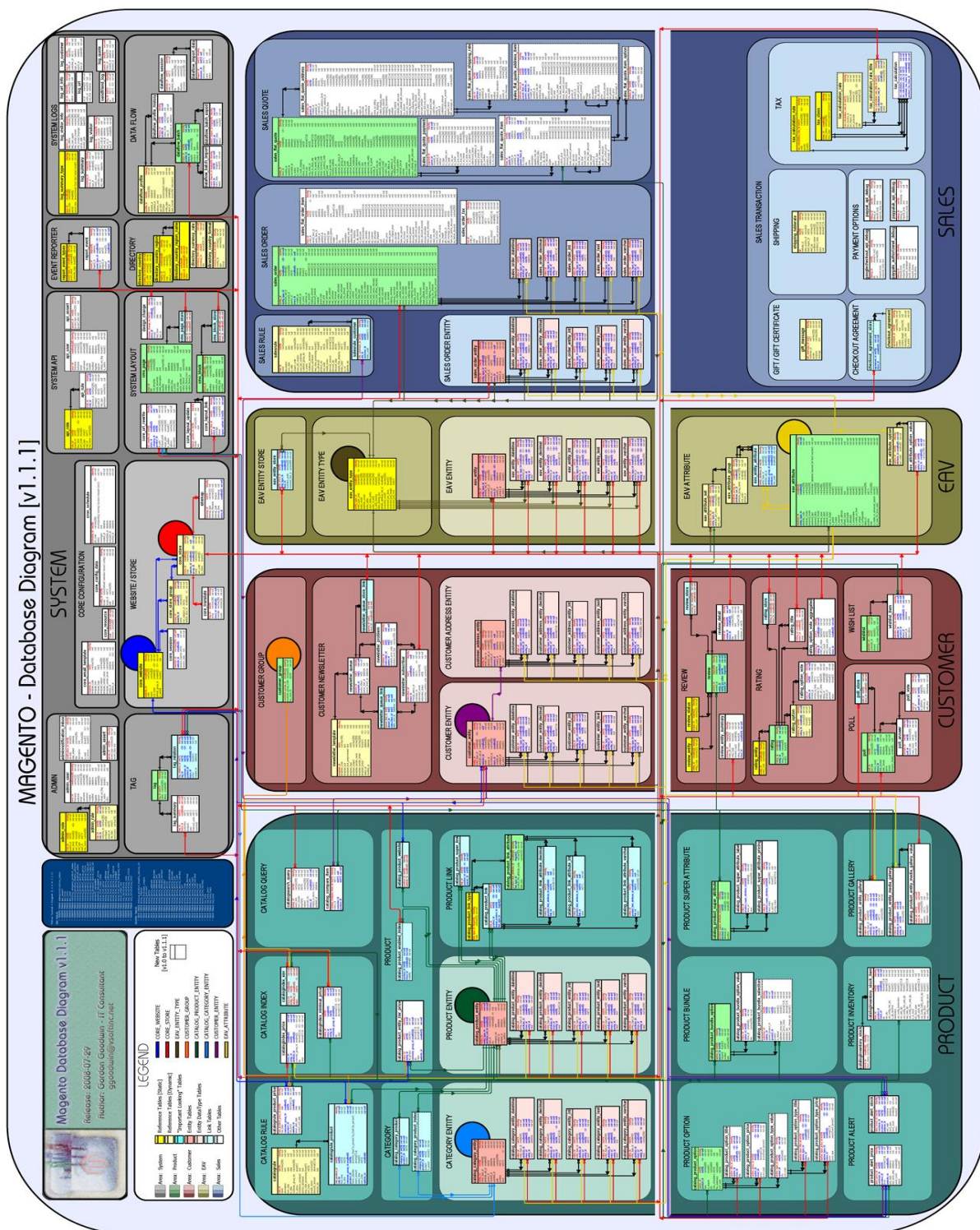


Ilustración 297 Modelo ER Magento



14.9 Anexo III: Descripción detallada de las clases

A continuación se hace una descripción detallada de todas las clases de la aplicación web.

14.9.1.1 *Class CategoryAction*

[com.tunos.commerce.actions](#)

```
java.lang.Object
```

```
└─ org.springframework.webflow.action.AbstractAction
```

```
└─ org.springframework.webflow.action.MultiAction
```

```
└─ org.springframework.webflow.action.FormAction
```

```
└─ com.tunos.commerce.actions.CategoryAction
```

All Implemented Interfaces:

```
org.springframework.webflow.execution.Action,  
org.springframework.beans.factory.InitializingBean
```

```
@org.springframework.stereotype.Service(value="categoryAction")  
public class CategoryAction
```

```
extends org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados principalmente con la entidad Category

Atributos

SC_LIST_CATEGORIES

```
private static String SC_LIST_CATEGORIES
```

categoryService

```
private CategoryService categoryService
```

userCategoryService

```
private UserCategoryService userCategoryService
```


productCategoryService

```
private ProductCategoryService productCategoryService
```

shopService

```
private ShopService shopService
```

userService

```
private UserService userService
```

Detalle de los métodos

selectCategoryFromTable

```
public org.springframework.webflow.execution.Event selectCategoryFromTable(org.springframework.webflow.execution.RequestContext context)
```

Permite seleccionar los detalles una categoría seleccionada por el usuario en una tabla
Returns: Event (success o error)

createCategoryForm

```
public org.springframework.webflow.execution.Event createCategoryForm(org.springframework.webflow.execution.RequestContext context)
```

Permite inicializar CategoryForm en el flujo
Returns: Event (success o error)

listAllCategoriesByUser

```
public org.springframework.webflow.execution.Event listAllCategoriesByUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Lista el conjunto de categorías pertenecientes a un usuario
Returns: Event (success o error)
Throws: Exception

reviewCategory

```
public org.springframework.webflow.execution.Event reviewCategory(org.springframework.webflow.execution.RequestContext context)
```

Obtiene los detalles de una categoría
Returns: Event (success o error)

getSearchCriteria

```
private SearchCriteria getSearchCriteria(org.springframework.webflow.execution
.RequestContext context, String criteriaName)
```

Gets the search criteria.

Parameters: context - the context

Returns: the search criteria

resetSearchCriteria

```
public org.springframework.webflow.execution.Event resetSearchCriteria(org.spr
ingframework.webflow.execution.RequestContext context)
```

Resetea el search Criteria

Returns:Event (success o error)

setSearchCriteria

```
private void setSearchCriteria(org.springframework.webflow.execution.RequestCo
ntext context, SearchCriteria searchCriteria, String criteriaName)
```

Set el search criteeria

listCategoriesPreviousPage

```
public org.springframework.webflow.execution.Event listCategoriesPreviousPage (
org.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las táblas, se encarga de listar la página anterior

Returns:Event (success o error)

listCategoriesNextPage

```
public org.springframework.webflow.execution.Event listCategoriesNextPage(org.
springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de categorías, se encarga de listar la siguiente página

Returns:Event (success o error)

listCategoriesFirstPage

```
public org.springframework.webflow.execution.Event listCategoriesFirstPage (org
.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de categorías, se encarga de listar la primera página

Returns:Event (success o error)

listCategoriesLastPage

```
public org.springframework.webflow.execution.Event listCategoriesLastPage(org.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de categorías, se encarga de listar la última página

Returns:Event (success o error)

resetSearchCriteriaPageCategories

```
public org.springframework.webflow.execution.Event resetSearchCriteriaPageCategories(org.springframework.webflow.execution.RequestContext context)
```

Resetea el número de página en el que se encuentra el search criteria de la tabla categorías

Returns:Event(Success o error)

saveParentOnFlow

```
public org.springframework.webflow.execution.Event saveParentOnFlow(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de guardar la categoría padre de la categoría seleccionada en el flujo de datos

Returns:Event (success o error)

getParentOnFlow

```
public org.springframework.webflow.execution.Event getParentOnFlow(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de obtener el padre de una categoría del flujo de datos

Returns:Event (success o error)

saveCategory

```
public org.springframework.webflow.execution.Event saveCategory(org.springframework.webflow.execution.RequestContext context)
```

throws Exception

Método encargado de obtener los datos de una categoría del flujo de datos y de llamara CategoryService para que la guarde en la base de datos así como de llamar a UserService para que relacione la categoría con el usuario

Returns:Event (success o error)

Throws:Exception

getCategory

```
public org.springframework.webflow.execution.Event getCategory(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de coger del flujo de datos una categoryForm refrescar sus datos y volver a guardarla en el flujo de datos

Returns:Event (success o error)

updateCategory

```
public org.springframework.webflow.execution.Event updateCategory(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe del flujo de datos una categoryForm con los datos de una categoría y se encarga de llamar al servicio CategoryService para que haga un update con los cambios realizados por el usuario

Returns:Event (success o error)

Throws:Exception

deleteCategory

```
public org.springframework.webflow.execution.Event deleteCategory(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe del flujo de datos una categoryForm que debe eliminar Para ello en primer lugar comprueba si la categoría tiene hijos, en caso afirmativo devuelve un error indicando al usuario que la categoría no puede eliminarse. Si la categoría no tiene hijos se desvinculará de los productos que estén categorizados con esta categoría y se invocan los métodos necesarios de categoryService para que la elimine de la base de datos y la desvincule del usuario.

Returns: Event (success o error)

Throws: Exception

reloadCategoryFormWithDisponibleShops

```
public org.springframework.webflow.execution.Event reloadCategoryFormWithDisponibleShops(org.springframework.webflow.execution.RequestContext context) throws Exception
```

obtiene una categoryForm y a ella le inserta los datos con los tipos de tienda disponibles

Returns: Event (success o error)

Throws: Exception

exportCategory

```
public org.springframework.webflow.execution.Event exportCategory(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe una form con los datos de la categoría, comprueba a que tipo de tienda pertenece e invoca los métodos necesarios del paquete intercommunication para que se conecten a la base de datos externa y exporten los datos

Returns:Event (success o error)

Throws:Exception

14.9.1.2 Class GenerateCsv

[com.tunos.commerce.actions](#)

```
java.lang.Object
```

```
└─ org.springframework.webflow.action.AbstractAction
```

```
└─ org.springframework.webflow.action.MultiAction
```

```
└─ org.springframework.webflow.action.FormAction
```

```
└─ com.tunos.commerce.actions.GenerateCsv
```

All Implemented Interfaces:

```
org.springframework.webflow.execution.Action,  
org.springframework.beans.factory.InitializingBean
```

```
@org.springframework.stereotype.Service(value="generateCVSAction")  
public class GenerateCsv extends org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar archivos CSV con el catálogo de productos de un usuario

Atributos

productService

```
private ProductService productService
```

userService

```
private UserService userService
```

Detalle de los métodos

generateCSV

```
public void generateCSV(org.springframework.webflow.execution.RequestContext context)
```

Método destinado a generar un archivo csv que contenga el catálogo de productos del usuario y enviarlo como response al navegador web

getProductsCSV

```
private String getProductsCSV (User user)
```

Método encargado de obtener la lista completa de productos de un usuario y crear un string con las columnas necesarias del csv para que pueda ser compatible con prestashop y magento

Returns:String

14.9.1.3 Class ParserAction

[com.tunos.commerce.actions](#)

```
java.lang.Object
```

```
└─ org.springframework.webflow.action.AbstractAction
```

```
└─ org.springframework.webflow.action.MultiAction
```

```
└─ org.springframework.webflow.action.FormAction
```

```
└─ com.tunos.commerce.actions.ParserAction
```

All Implemented Interfaces:

```
org.springframework.webflow.execution.Action,  
org.springframework.beans.factory.InitializingBean
```

```
@org.springframework.stereotype.Service(value="parserAction")  
public class ParserAction  
extends org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para parsear el xml

Atributos

userService

```
private UserService userService
```

providerService

```
private ProviderService providerService
```

userProviderService

```
private UserProviderService userProviderService
```

userProductService

```
private UserProductService userProductService
```

productService

```
private ProductService productService
```

productCategoryService

```
private ProductCategoryService productCategoryService
```

categoryService

```
private CategoryService categoryService
```

userCategoryService

```
private UserCategoryService userCategoryService
```

Detalle de los métodos

createForm

```
public org.springframework.webflow.execution.Event createForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de inicializar una form de tipo parserForm y meterla en el flujo de datos

Returns:Event (success o error)

uploadFile

```
public org.springframework.webflow.execution.Event uploadFile(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de obtener el xml de la presentación, parsearlo y llamar a los métodos necesarios para que almacenen la información (categorías, productos y proveedores) en la base de datos

Returns:Event (success o error)

14.9.1.4 Class *ProductAction*

[com.tunos.commerce.actions](#)

```
java.lang.Object
```

```
└─ org.springframework.webflow.action.AbstractAction
```

```
    └─ org.springframework.webflow.action.MultiAction
```

```
        └─ org.springframework.webflow.action.FormAction
```

```
            └─ com.tunos.commerce.actions.ProductAction
```

All Implemented Interfaces:

```
org.springframework.webflow.execution.Action,  
org.springframework.beans.factory.InitializingBean
```

```
@org.springframework.stereotype.Service(value="productAction")  
public class ProductAction extends  
org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad Product

Atributos

SC_LIST_PRODUCTS

```
private static String SC_LIST_PRODUCTS
```

productService

```
private ProductService productService
```

providerService

```
private ProviderService providerService
```

userProductService

```
private UserProductService userProductService
```

userService

```
private UserService userService
```

categoryService

```
private CategoryService categoryService
```

productCategoryService

```
private ProductCategoryService productCategoryService
```

shopService

```
private ShopService shopService
```

Detalle de los métodos

createProductForm

```
public org.springframework.webflow.execution.Event createProductForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de inicializar una form de tipo productForm y meterla en el flujo de datos

Returns:Event (success o error)

setupProductForm

```
public org.springframework.webflow.execution.Event setupProductForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de actualizar una form de tipo productForm y meterla en el flujo de datos

Returns:Event (success o error)

selectProviderFromTable

```
public org.springframework.webflow.execution.Event selectProviderFromTable(org.springframework.webflow.execution.RequestContext context)
```

Permite seleccionar los detalles un proveedor seleccionada por el usuario en una tabla

Returns:Event (success o error)

resetSearchProductForm

```
public org.springframework.webflow.execution.Event resetSearchProductForm(org.springframework.webflow.execution.RequestContext context)
```

Permite actualizar el form de búsqueda de productos

Returns:Event (success o error)

createSearchProductForm

```
public org.springframework.webflow.execution.Event createSearchProductForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de crear un productForm para las búsquedas de productos

Returns: Event (success o error)

saveProviderOnFlow

```
public org.springframework.webflow.execution.Event saveProviderOnFlow(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de almacenar el proveedor de un producto en el flujo de datos

Returns: Event (success o error)

getProviderOnFlow

```
public org.springframework.webflow.execution.Event getProviderOnFlow(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de obtener del flujo de datos el proveedor de un producto

Returns: Event (success o error)

saveProduct

```
public org.springframework.webflow.execution.Event saveProduct(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener del flujo los datos de un producto para almacenarlos en la base de datos. Para guardar los datos se invocan los métodos necesarios de productService y para crear los vínculos entre el usuario y el producto se invocan los métodos necesarios de userProductService

Returns: Event (success o error)

Throws: Exception

listAllProductsByUser

```
public org.springframework.webflow.execution.Event listAllProductsByUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener el conjunto de productos pertenecientes al usuario que está logueado

Returns: Event (success o error)

Throws: Exception

getSearchCriteria

```
private SearchCriteria getSearchCriteria(org.springframework.webflow.execution
.RequestContext context, String criteriaName)
```

Gets the search criteria.

Parameters: context - the context

Returns: the search criteria

resetSearchCriteria

```
private SearchCriteria resetSearchCriteria(org.springframework.webflow.executi
on.RequestContext context,String criteriaName)
```

Resetea el search Criteria

Returns:Event (success o error)

setSearchCriteria

```
private void setSearchCriteria(org.springframework.webflow.execution.RequestCo
ntext context, SearchCriteria searchCriteria,String criteriaName)
```

Set el search criteria

listProductsPreviousPage

```
public org.springframework.webflow.execution.Event listProductsPreviousPage(or
g.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las táblas, se encarga de listar la página anterior

Returns:Event (success o error)

listProductsNextPage

```
public org.springframework.webflow.execution.Event listProductsNextPage(org.sp
ringframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de productos, se encarga de listar la siguiente página

Returns:Event (success o error)

listProductsFirstPage

```
public org.springframework.webflow.execution.Event listProductsFirstPage(org.s
pringframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de productos, se encarga de listar la primera página

Returns:Event (success o error)

listProductsLastPage

```
public org.springframework.webflow.execution.Event listProductsLastPage(org.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de productos, se encarga de listar la última página

Returns:Event (success o error)

resetSearchCriteriaPageProducts

```
public org.springframework.webflow.execution.Event resetSearchCriteriaPageProducts(org.springframework.webflow.execution.RequestContext context)
```

Resetea el número de página en el que se encuentra el search criteria de la tabla productos

Returns:Event(Success o error)

reviewProduct

```
public org.springframework.webflow.execution.Event reviewProduct(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de obtener del flujo de datos un producto, incorporar datos a la form que le faltan y volver a guardarlo en el flujo de datos

Returns:Event(Success o error)

getProduct

```
public org.springframework.webflow.execution.Event getProduct(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de obtener del flujo de datos un producto

Returns:Event(Success o error)

selectCategoriesFromTable

```
public org.springframework.webflow.execution.Event selectCategoriesFromTable(org.springframework.webflow.execution.RequestContext context)
```

Permite seleccionar un conjunto de categorías de una tabla para poder relacionarlas con un producto

Returns:Event(Success o error)

deselectCategoriesFromTable

```
public org.springframework.webflow.execution.Event deselectCategoriesFromTable(org.springframework.webflow.execution.RequestContext context)
```

Permite desvincular un conjunto de categorías de una tabla de un determinado producto

Returns:Event(Success o error)

listAllCategoriesProduct

```
public org.springframework.webflow.execution.Event listAllCategoriesProduct (org.springframework.webflow.execution.RequestContext context)
```

Permite listar el conjunto de categorías pertenecientes a un producto

Returns:Event(Success o error)

saveSelectCategoriesOnFlow

```
public org.springframework.webflow.execution.Event saveSelectCategoriesOnFlow (org.springframework.webflow.execution.RequestContext context)
```

Permite obtener un conjunto de categorías del flujo de datos, actualizar las forms y volver a guardar en el flujo de datos

Returns:Event(Success o error)

updateProduct

```
public org.springframework.webflow.execution.Event updateProduct (org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener los datos de un producto del flujo de datos y de llamar a ProductService para que actualice los datos en la base de datos

Returns:Event (success o error)

Throws:Exception

deleteProduct

```
public org.springframework.webflow.execution.Event deleteProduct (org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe del flujo de datos una productForm que debe eliminar a continuación invoca los métodos necesarios de productService y userProductService para que eliminen el producto de la base de datos

Returns:Event (success o error)

Throws:Exception

reloadProductFormWithDisponibleShops

```
public org.springframework.webflow.execution.Event reloadProductFormWithDisponibleShops (org.springframework.webflow.execution.RequestContext context) throws Exception
```

Obtiene una productForm y a ella le inserta los datos con los tipos de tienda disponibles

Returns: Event (success o error)

Throws:Exception

exportProduct

```
public org.springframework.webflow.execution.Event exportProduct(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe una form con los datos del producto, comprueba a que tipo de tienda pertenece e invoca los métodos necesarios del paquete intercommunication para que se conecten a la base de datos externa y exporten los datos

Returns: Event (success o error)

Throws: Exception

14.9.1.5 Class ProviderAction

[com.tunos.commerce.actions](#)

```
java.lang.Object
```

```
└ org.springframework.webflow.action.AbstractAction
```

```
└ org.springframework.webflow.action.MultiAction
```

```
└ org.springframework.webflow.action.FormAction
```

```
└ com.tunos.commerce.actions.ProviderAction
```

All Implemented Interfaces:

org.springframework.webflow.execution.Action,

org.springframework.beans.factory.InitializingBean

```
@org.springframework.stereotype.Service(value="providerAction")
public class ProviderAction extends
org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad Provider

Atributos

SC_LIST_PROVIDERS

```
private static String SC_LIST_PROVIDERS
```

providerService

```
private ProviderService providerService
```

productService

```
private ProductService productService
```

userProviderService

```
private UserProviderService userProviderService
```

userService

```
private UserService userService
```

shopService

```
private ShopService shopService
```

Detalle de los métodos

reviewProvider

```
public org.springframework.webflow.execution.Event reviewProvider(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de obtener del flujo de datos un proveedor, incorporar datos a la form que le faltan y volver a guardarlo en el flujo de datos

Returns: Event (Success o error)

createProviderForm

```
public org.springframework.webflow.execution.Event createProviderForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de inicializar una form de tipo providerForm y meterla en el flujo de datos

Returns: Event (success o error)

saveProvider

```
public org.springframework.webflow.execution.Event saveProvider(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener del flujo los datos de un proveedor para almacenarlos en la base de datos. Para guardar los datos se invocan los métodos necesarios de providerService y para crear los vínculos entre el usuario y el producto se invocan los métodos necesarios de userProviderService

Returns: Event (success o error)

Throws: Exception

listAllProvidersByUser

```
public org.springframework.webflow.execution.Event listAllProvidersByUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener el conjunto de proveedores pertenecientes al usuario que está logueado

Returns:Event (success o error)

Throws:Exception

getSearchCriteria

```
private SearchCriteria getSearchCriteria(org.springframework.webflow.execution.  
.RequestContext context, String criteriaName)
```

Gets the search criteria.

Parameters: context - the context

Returns: the search criteria

setSearchCriteria

```
private void setSearchCriteria(org.springframework.webflow.execution.RequestCo  
ntext context, SearchCriteria searchCriteria, String criteriaName)
```

Set el search criteria

listProvidersPreviousPage

```
public org.springframework.webflow.execution.Event listProvidersPreviousPage(o  
rg.springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las táblas, se encarga de listar la página anterior

Returns:Event (success o error)

listProvidersNextPage

```
public org.springframework.webflow.execution.Event listProvidersNextPage(org.s  
pringframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de proveedores, se encarga de listar la siguiente página

Returns:Event (success o error)

listProvidersFirstPage

```
public org.springframework.webflow.execution.Event listProvidersFirstPage(org.  
springframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de proveedores, se encarga de listar la primera página

Returns:Event (success o error)

listProvidersLastPage

```
public org.springframework.webflow.execution.Event listProvidersLastPage(org.s  
pringframework.webflow.execution.RequestContext context)
```

Método dedicado a la paginación de las tablas de productos, se encarga de listar la última página

Returns:Event (success o error)

resetSearchCriteriaPageProviders

```
public org.springframework.webflow.execution.Event resetSearchCriteriaPageProviders(org.springframework.webflow.execution.RequestContext context)
```

Resetea el número de página en el que se encuentra el search criteria de la tabla proveedores

Returns:Event(Success o error)

updateProvider

```
public org.springframework.webflow.execution.Event updateProvider(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener los datos de un producto del flujo de datos y de llamar a ProviderService para que actualice los datos en la base de datos

Returns: Event (success o error)

Throws: Exception

deleteProvider

```
public org.springframework.webflow.execution.Event deleteProvider(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe del flujo de datos una providerForm que debe eliminar a continuación invoca los métodos necesarios de providerService y userProviderService para que eliminen el proveedor de la base de datos

Returns:Event (success o error)

Throws:Exception

reloadProviderFormWithDisponibleShops

```
public org.springframework.webflow.execution.Event reloadProviderFormWithDisponibleShops(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Actualiza la form del proveedor con las tiendas disponibles

Throws:Exception

exportProvider

```
public org.springframework.webflow.execution.Event exportProvider(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Obtiene una providerForm y a ella le inserta los datos con los tipos de tienda disponibles
Returns:Event (success o error)
Throws:Exception

14.9.1.6 Class QRCodeAction

[com.tunos.commerce.actions](#)

java.lang.Object

```

└─ org.springframework.webflow.action.AbstractAction
    └─ org.springframework.webflow.action.MultiAction
        └─ org.springframework.webflow.action.FormAction
            └─ com.tunos.commerce.actions.QRCodeAction
  
```

All Implemented Interfaces:

org.springframework.webflow.execution.Action,
 org.springframework.beans.factory.InitializingBean

```

@org.springframework.stereotype.Service(value="qRCodeAction")
public class QRCodeAction extends
org.springframework.webflow.action.FormAction
  
```

Clase encargada de generar un código QR a partir de la información de un producto

Detalle de los métodos

generateQRCode

```
public void generateQRCode(org.springframework.webflow.context.ExternalContext
context)throws IOException
```

Genera un código QR con la información del producto y lo coloca como response

Parameters: context - the context

Returns:the event

Throws:IOException

14.9.1.7 Class ShopAction

[com.tunos.commerce.actions](#)

java.lang.Object

```

└─ org.springframework.webflow.action.AbstractAction
    └─ org.springframework.webflow.action.MultiAction
        └─ org.springframework.webflow.action.FormAction
            └─ com.tunos.commerce.actions.ShopAction
  
```

All Implemented Interfaces:

org.springframework.webflow.execution.Action,
org.springframework.beans.factory.InitializingBean

```
@org.springframework.stereotype.Service(value="shopAction")  
public class ShopAction extends org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad Shop

Atributos

userShopService

```
private UserShopService userShopService
```

shopService

```
private ShopService shopService
```

userService

```
private UserService userService
```

productService

```
private ProductService productService
```

categoryService

```
private CategoryService categoryService
```

providerService

```
private ProviderService providerService
```

userProviderService

```
private UserProviderService userProviderService
```

userProductService

```
private UserProductService userProductService
```


userCategoryService

```
private UserCategoryService userCategoryService
```

productCategoryService

```
private ProductCategoryService productCategoryService
```

Detalle de los métodos

setupShopForm

```
public org.springframework.webflow.execution.Event setupShopForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de actualizar una form de tipo shopForm y meterla en el flujo de datos

Returns:Event (success o error)

saveShop

```
public org.springframework.webflow.execution.Event saveShop(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener del flujo los datos de una tienda para almacenarlos en la base de datos. Para guardar los datos se invocan los métodos necesarios de shopService y para crear los vínculos entre el usuario y el producto se invocan los métodos necesarios de userShopService. Para almacenar las contraseñas de la base de datos externa se utiliza CreateXORpass para codificarlas.

Returns: Event (success o error)

Throws:Exception

reviewShop

```
public org.springframework.webflow.execution.Event reviewShop(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de obtener del flujo de datos una tienda, incorporar datos a la form que le faltan y volver a guardarlo en el flujo de datos

Returns:Event(Success o error)

updateShop

```
public org.springframework.webflow.execution.Event updateShop(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener los datos de una tienda del flujo de datos y de llamara ShopService para que actualice los datos en la base de datos

Returns: Event (success o error)

Throws:Exception

listShopsByUser

```
public org.springframework.webflow.execution.Event listShopsByUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener el conjunto de tiendas pertenecientes al usuario que está logueado y guardarlas en el flujo de datos

Returns: Event (success o error)

Throws:Exception

deleteShop

```
public org.springframework.webflow.execution.Event deleteShop(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método que recibe del flujo de datos una shopForm que debe eliminar a continuación invoca los métodos necesarios de shopService y userShopService para que eliminen la tienda de la base de datos

Returns: Event (success o error)

Throws: Exception

importShopElements

```
public org.springframework.webflow.execution.Event importShopElements(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener de presentación los tipos de datos que se deben importar. Se crea una instancia a las clases de intercommunication y se obtienen los diferentes elementos. Se crean las relaciones entre ellos.

Returns:Event (success o error)

Throws:Exception

14.9.1.8 *Class UserAction*

com.tunos.commerce.actions

java.lang.Object

└─ org.springframework.webflow.action.AbstractAction

└─ org.springframework.webflow.action.MultiAction

└─ org.springframework.webflow.action.FormAction

└─ **com.tunos.commerce.actions.UserAction**

All Implemented Interfaces:

org.springframework.webflow.execution.Action,
org.springframework.beans.factory.InitializingBean

```
@org.springframework.stereotype.Service(value="userAction")
public class UserAction extends org.springframework.webflow.action.FormAction
```

Clase que implementa los métodos necesarios para generar las acciones de control consumiendo los servicios relacionados con la entidad User

Atributos

userService

```
private UserService userService
```

roleUserService

```
private RoleUserService roleUserService
```

roleService

```
private RoleService roleService
```

Detalle de los métodos

reviewUserForm

```
public org.springframework.webflow.execution.Event reviewUserForm(org.springframework.webflow.execution.RequestContext context)
```

Método que se encarga de obtener del flujo de datos un usuario, incorporar datos a la form que le faltan y volver a guardarlo en el flujo de datos

Returns: Event(Success o error)

updateUser

```
public org.springframework.webflow.execution.Event updateUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener los datos de un usuario del flujo de datos y de llamar a UserService para que actualice los datos en la base de datos

Returns: Event (success o error)

Throws: Exception

createUser

```
public org.springframework.webflow.execution.Event createUser(org.springframework.webflow.execution.RequestContext context) throws Exception
```

Método encargado de obtener del flujo los datos de un usuario para almacenarlos en la base de datos. Para guardar los datos se invocan los métodos necesarios de userService .

Returns: Event (success o error)

Throws: Exception

setupUserForm

```
public org.springframework.webflow.execution.Event setupUserForm(org.springframework.webflow.execution.RequestContext context)
```

Método encargado de actualizar una form de tipo shopForm y meterla en el flujo de datos

Returns:Event (success o error)

14.9.1.9 Interface CategoryDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[CategoryDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface CategoryDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad Category

Detalle de los métodos

listCategories

```
List<Category> listCategories()
```

Lista todas las categorías existentes en la base de datos

saveCategory

```
Category saveCategory(Category category)
```

Método para hacer persist de una categoría

Returns: Category

updateCategory

```
Category updateCategory(Category category)
```

Permite hacer update a una categoría

Returns:Category

deleteCategory

[Category](#) deleteCategory([Category](#) category)

Permite eliminar de la base de datos una categoría

Returns:category

findCategoryById

[Category](#) findCategoryById(Integer id)

Busca una categoría por identificador

Returns:Category

findCategoryByName

[Category](#) findCategoryByName(String name,Integer user)

Busca una categoría por nombre

Returns:category

setEntityManager

void setEntityManager(EntityManager em)

Hace un set del entity manager

listTotalEntriesCategoriesByUser

List<Integer> listTotalEntriesCategoriesByUser([SearchCriteria](#) searchCriteria,
Integer userId,
Integer parentId)

Lista todas las categorías de un usuario y devuelve una lista con sus identificadores

Returns:List

listCategoriesByUser

List<[Category](#)> listCategoriesByUser([SearchCriteria](#) searchCriteria,
Integer userId,
Integer parentId)

Lista todas las categorías de un usuario y devuelve una lista de categorías

Returns: List <Category> categories

listChilds

List<[Category](#)> listChilds(Integer parentId)

Lista las subcategorías de una categoría

Returns: List <Category> childs

14.9.1.10 Class CategoryDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ com.tunos.commerce.dao.CategoryDAOImpl

All Implemented Interfaces:

[CategoryDAO](#)

@org.springframework.stereotype.Repository

public class **CategoryDAOImpl** extends Object implements [CategoryDAO](#)

Clase que implementa los métodos de persistencia declarados en la interfaz CategoryDAO utilizando JPA

Atributos

em

private EntityManager em

Constructor Detail

CategoryDAOImpl

public **CategoryDAOImpl**()

Detalle de los métodos

listCategories

public List<[Category](#)> **listCategories**()

Lista todas las categorías existentes en la base de datos

Specified by:

listCategories in interface [CategoryDAO](#)

Returns: List<[Category](#)> categories

saveCategory

public [Category](#) **saveCategory**([Category](#) category)

Método para hacer persist de una categoría

Specified by: **saveCategory** in interface [CategoryDAO](#)

Returns: Category

updateCategory

```
public Category updateCategory(Category category)
```

Permite hacer update a una categoría

Specified by: `updateCategory` in interface [CategoryDAO](#)

Returns: Category

deleteCategory

```
public Category deleteCategory(Category category)
```

Permite eliminar de la base de datos una categoría

Specified by: `deleteCategory` in interface [CategoryDAO](#)

Returns: category

findCategoryById

```
public Category findCategoryById(Integer id)
```

Busca una categoría por identificador

Specified by: `findCategoryById` in interface [CategoryDAO](#)

Returns: category

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Hace un set del entity manager

Specified by: `setEntityManager` in interface [CategoryDAO](#)

listTotalEntriesCategoriesByUser

```
public List<Integer> listTotalEntriesCategoriesByUser(SearchCriteria searchCriteria, Integer userId, Integer parentId)
```

Lista todas las categorías de un usuario y devuelve una lista con sus identificadores

Specified by: `listTotalEntriesCategoriesByUser` in interface [CategoryDAO](#)

Returns: List

listCategoriesByUser

```
public List<Category> listCategoriesByUser(SearchCriteria searchCriteria, Integer userId, Integer parentId)
```

Lista todas las categorías de un usuario y devuelve una lista de categorías

Specified by: [listCategoriesByUser](#) in interface [CategoryDAO](#)

Returns: List<[Category](#)> categories

listChilds

```
public List<Category> listChilds(Integer parentId)
```

Lista las subcategorías de una categoría

Specified by: [listChilds](#) in interface [CategoryDAO](#)

Returns: List <[Category](#)> categories

findCategoryByName

```
public Category findCategoryByName(String name, Integer user)
```

Busca una categoría por nombre

Specified by: [findCategoryByName](#) in interface [CategoryDAO](#)

Returns: category

14.9.1.11 Interface ProductCategoryDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[ProductCategoryDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface ProductCategoryDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad ProductCategory

Detalle de los métodos

listProductCategories

```
List<ProductCategory> listProductCategories()
```

Lista el todos los elementos de tipo ProductCategory que hay en la base de datos

Returns: List<[ProductCategory](#)> productCategories

saveProductCategory

```
ProductCategory saveProductCategory(ProductCategory productCategory)
```

Guarda en la base de datos la relación ProductCategory

Returns: ProducCategory

updateProductCategory

[ProductCategory](#) **updateProductCategory** ([ProductCategory](#) productCategory)

Actualiza la relación productCategory en la base de datos

Returns: ProductCategory

deleteProductCategory

[ProductCategory](#) **deleteProductCategory** ([ProductCategory](#) productCategory)

Elimina de la base de datos la entidad ProductCategory

Returns: ProductCategory

findProductCategoryById

[ProductCategory](#) **findProductCategoryById** (Integer id)

Busca un productCategory por identificador

Returns: ProductCategory

findCategoriesFromProduct

List<Integer> **findCategoriesFromProduct** (Integer id)

Busca una relación de tipo productCategory por id de producto y retorna una lista de enteros con el id de los elementos encontrados

Returns: List <Integer> ids

setEntityManager

void **setEntityManager** (EntityManager em)

set el entity manager

findByCategory

List<[ProductCategory](#)> **findByCategory** ([Category](#) c)

Busca las relaciones de tipo ProductCategory por categoría

Returns: List<ProductCategory> productCategories

findByProduct

List<[ProductCategory](#)> **findByProduct** ([Product](#) p)

Busca las relaciones de tipo ProductCategory por producto

Returns: List<[ProductCategory](#)> productCategories

14.9.1.12 Class *ProductCategoryDAOImpl*

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.ProductCategoryDAOImpl`

All Implemented Interfaces:

[ProductCategoryDAO](#)

```
@org.springframework.stereotype.Repository
public class ProductCategoryDAOImpl extends Object implements
ProductCategoryDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz `ProductCategoryDAO` utilizando JPA.

Atributos

em

```
private EntityManager em
```

The em.

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: `setEntityManager` in interface [ProductCategoryDAO](#)

Parameters: `em` - the new entity manager

listProductCategories

```
public List<ProductCategory> listProductCategories()
```

Lista el todos los elementos de tipo `ProductCategory` que hay en la base de datos

Specified by: `listProductCategories` in interface [ProductCategoryDAO](#)

Returns: List

saveProductCategory

```
public ProductCategory saveProductCategory(ProductCategory productCategory)
```

Guarda en la base de datos la relación `ProductCategory`

Specified by: `saveProductCategory` in interface [ProductCategoryDAO](#)

Returns: ProducCategory

updateProductCategory

```
public ProductCategory updateProductCategory(ProductCategory productCategory)
```

Actualiza la relación productCategory en la base de datos

Specified by: `updateProductCategory` in interface [ProductCategoryDAO](#)

Returns: ProductCategory

deleteProductCategory

```
public ProductCategory deleteProductCategory(ProductCategory productCategory)
```

Elimina de la base de datos la entidad ProducCategory

Specified by: `deleteProductCategory` in interface [ProductCategoryDAO](#)

Returns: ProductCategory

findProductCategoryById

```
public ProductCategory findProductCategoryById(Integer id)
```

Busca un productCategory por identificador

Specified by: `findProductCategoryById` in interface [ProductCategoryDAO](#)

Returns: ProductCategory

findCategoriesFromProduct

```
public List<Integer> findCategoriesFromProduct(Integer id)
```

Busca una relación de tipo productCategory por id de producto y retorna una lista de enteros con el id de los elementos encontrados

Specified by: `findCategoriesFromProduct` in interface [ProductCategoryDAO](#)

Returns: List

findByCategory

```
public List<ProductCategory> findByCategory(Category c)
```

Busca las relaciones de tipo ProductCategory por categoría

Specified by: `findByCategory` in interface [ProductCategoryDAO](#)

Returns: List

findByProduct

```
public List<ProductCategory> findByProduct(Product p)
```

Busca las relaciones de tipo ProductCategory por producto

Specified by: [findByProduct](#) in interface [ProductCategoryDAO](#)

Returns: List

14.9.1.13 Interface ProductDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[ProductDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface ProductDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad Category

Detalle de los métodos

listProducts

```
List<Product> listProducts()
```

Lista el conjunto de productos existentes sin restricciones

Returns: List

saveProduct

```
Product saveProduct(Product product)
```

Guarda un producto en la base de datos

Returns: product

updateProduct

```
Product updateProduct(Product product)
```

Actualiza un producto en la base de datos

Returns: product

deleteProduct

```
Product deleteProduct(Product product)
```

Elimina un producto en la base de datos

Returns: product

findProductById

```
Product findProductById(Integer id)
```

Busca un producto por identificador

Returns:Product

setEntityManager

```
void setEntityManager(EntityManager em)
```

set entity manager

findProductByName

```
Product findProductByName(String name)
```

Busca un producto por nombre

Returns:product

listProductsByProvider

```
List<Product> listProductsByProvider(Provider provider)
```

Lista un conjunto de productos por proveedor

Returns: List

listProductsByUser

```
List<Product> listProductsByUser(SearchCriteria criteria,Integer idUser)
```

Lista los productos por usuario

Returns:List

listTotalEntriesProductsByUser

```
List<Integer> listTotalEntriesProductsByUser(SearchCriteria criteria,Integer idUser)
```

Lista todos los productos de un determinado usuario y devuelve una lista de enteros

Returns:List

listProductsByUser

```
List<Product> listProductsByUser(Integer idUser)
```

Lista los productos de un usuario

14.9.1.14 Class ProductDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.ProductDAOImpl`

All Implemented Interfaces:

[ProductDAO](#)

```
@org.springframework.stereotype.Repository
public class ProductDAOImpl extends Object implements ProductDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz ProductDAO utilizando JPA.

Atributos

em

```
private EntityManager em
```

The e.m

Detalle de los métodos

listProducts

```
public List<Product> listProducts()
```

Lista el conjunto de productos existentes sin restricciones

Specified by: `listProducts` in interface [ProductDAO](#)

Returns: List

saveProduct

```
public Product saveProduct(Product product)
```

Guarda un producto en la base de datos

Specified by: `saveProduct` in interface [ProductDAO](#)

Returns: product

updateProduct

```
public Product updateProduct(Product product)
```

Actualiza un producto en la base de datos

Specified by: `updateProduct` in interface [ProductDAO](#)

Returns: product

deleteProduct

```
public Product deleteProduct(Product product)
```

Elimina un producto en la base de datos

Specified by: [deleteProduct](#) in interface [ProductDAO](#)

Returns: product

findProductById

```
public Product findProductById(Integer id)
```

Busca un producto por identificador

Specified by: [findProductById](#) in interface [ProductDAO](#)

Returns: Product

listProductsByProvider

```
public List<Product> listProductsByProvider(Provider provider)
```

Lista un conjunto de productos por proveedor

Specified by: [listProductsByProvider](#) in interface [ProductDAO](#)

Returns: List

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Set entity manager

Specified by: [setEntityManager](#) in interface [ProductDAO](#)

findProductByName

```
public Product findProductByName(String name)
```

Busca un producto por nombre

Specified by: [findProductByName](#) in interface [ProductDAO](#)

Returns: product

listProductsByUser

```
public List<Product> listProductsByUser(Integer idUser)
```

Lista los productos de un usuario

Specified by: [listProductsByUser](#) in interface [ProductDAO](#)

listProductsByUser

```
public List<Product> listProductsByUser(SearchCriteria criteria,Integer idUser)
```

Lista los productos por usuario

Specified by: `listProductsByUser` in interface `ProductDAO`

Returns: `List`

listTotalEntriesProductsByUser

```
public List<Integer> listTotalEntriesProductsByUser(SearchCriteria criteria,  
                                                    Integer idUser)
```

Lista todos los productos de un determinado usuario y devuelve una lista de enteros

Specified by: `listTotalEntriesProductsByUser` in interface `ProductDAO`

Returns: `List`

14.9.1.15 Interface ProviderDAO

com.tunos.commerce.dao

All Known Implementing Classes:

[ProviderDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface ProviderDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad Provider

Detalle de los métodos

listProviders

```
List<Provider> listProviders()
```

Lista todos los proveedores de la base e datos

Returns: `List`

saveProvider

```
Provider saveProvider(Provider provider)
```

Guarda un proveedor en la base de datos

Returns: `provider`

updateProvider

```
Provider updateProvider(Provider provider)
```

Actualiza un proveedor en la base de datos

Returns: provider

deleteProvider

[Provider](#) deleteProvider([Provider](#) provider)

Elimina un proveedor de la base de datos

Returns: provider

findProviderById

[Provider](#) findProviderById(Integer id)

Busca un proveedor por id

Returns: provider

findProviderByNIF

[Provider](#) findProviderByNIF(String nif)

Busca un proveedor por nif

Returns: provider

findProviderByCompany

[Provider](#) findProviderByCompany(String company, Integer user)

Busca un proveedor por compañía

Returns: Provider

setEntityManager

void setEntityManager(EntityManager em)

set entity Manager

listProviderByIdUser

List<Integer> listProviderByIdUser(Integer idUser)

Lista los proveedores por usuario y devuelve unicamente una lista de enteros con los id's

Returns: List

listProviders

List<[Provider](#)> listProviders([SearchCriteria](#) criteria)

Lista los proveedores

Returns:List

listProvidersByUser

```
List<Provider> listProvidersByUser(SearchCriteria criteria, Integer idUser)
```

Lista los proveedores por usuario teniendo en cuenta el searchCriteria

Returns:List

listTotalEntriesProvidersByUser

```
List<Integer> listTotalEntriesProvidersByUser(SearchCriteria criteria, Integer idUser)
```

Lista el conjunto de proveedores de un usuario y devuelve una lista de enteros

Returns:List

14.9.1.16 Class *ProviderDAOImpl*

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.ProviderDAOImpl`

All Implemented Interfaces:

[ProviderDAO](#)

```
@org.springframework.stereotype.Repository
```

```
public class ProviderDAOImpl extends Object implements ProviderDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz ProviderDAO utilizando JPA.

Atributos

em

```
private EntityManager em
```

The em.

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: `setEntityManager` in interface [ProviderDAO](#)

Parameters: `em` - the new entity manager

listProviders

```
public List<Provider> listProviders()
```

Lista todos los proveedores de la base e datos

Specified by: `listProviders` in interface [ProviderDAO](#)

Returns: List

saveProvider

```
public Provider saveProvider(Provider provider)
```

Guarda un proveedor en la base de datos

Specified by: `saveProvider` in interface [ProviderDAO](#)

Returns: provider

updateProvider

```
public Provider updateProvider(Provider provider)
```

Actualiza un proveedor en la base de datos

Specified by: `updateProvider` in interface [ProviderDAO](#)

Returns: provider

deleteProvider

```
public Provider deleteProvider(Provider provider)
```

Elimina un proveedor de la base de datos

Specified by: `deleteProvider` in interface [ProviderDAO](#)

Returns: provider

findProviderById

```
public Provider findProviderById(Integer id)
```

Busca un proveedor por id

Specified by: `findProviderById` in interface [ProviderDAO](#)

Returns: provider

findProviderByNIF

```
public Provider findProviderByNIF(String nif)
```

Busca un proveedor por nif

Specified by: [findProviderByNIF](#) in interface [ProviderDAO](#)

Returns: provider

findProviderByCompany

```
public Provider findProviderByCompany(String company, Integer user)
```

Busca un proveedor por compañía

Specified by: [findProviderByCompany](#) in interface [ProviderDAO](#)

Returns: Provider

listProviderByIdUser

```
public List<Integer> listProviderByIdUser(Integer idUser)
```

Lista los proveedores por usuario y devuelve unicamente una lista de enteros con los id's

Specified by: [listProviderByIdUser](#) in interface [ProviderDAO](#)

Returns: List

listProviders

```
public List<Provider> listProviders(SearchCriteria criteria)
```

Lista los proveedores

Specified by: [listProviders](#) in interface [ProviderDAO](#)

Returns: List

listProvidersByUser

```
public List<Provider> listProvidersByUser(SearchCriteria criteria, Integer idUser)
```

Lista los proveedores por usuario teniendo en cuenta el searchCriteria

Specified by: [listProvidersByUser](#) in interface [ProviderDAO](#)

Returns: List

listTotalEntriesProvidersByUser

```
public List<Integer> listTotalEntriesProvidersByUser(SearchCriteria criteria,  
Integer idUser)
```

Lista el conjunto de proveedores de un usuario y devuelve una lista de enteros

Specified by: [listTotalEntriesProvidersByUser](#) in interface [ProviderDAO](#)

Returns: List

14.9.1.17 Interface RoleDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[RoleDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface RoleDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad Role

Detalle de los métodos

listRoles

```
List<Role> listRoles()
```

Método encargado de listar todos los roles

Returns: List

saveRole

```
Role saveRole(Role role)
```

Se encarga de guardar un rol

Returns: Role

updateRole

```
Role updateRole(Role role)
```

Se encarga de actualizar un Rol

Returns: Role

deleteRole

```
Role deleteRole(Role role)
```

Se encarga de eliminar un rol

Returns: Role

findRoleById

```
Role findRoleById(Integer id)
```

Se encarga de buscar un rol por Id

setEntityManager

```
void setEntityManager(EntityManager em)
```

set EntityManager

14.9.1.18 Class RoleDAOImpl

[com.tunos.commerce.dao](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.dao.RoleDAOImpl
```

All Implemented Interfaces:

[RoleDAO](#)

```
@org.springframework.stereotype.Repository  
public class RoleDAOImpl extends Object implements RoleDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz RoleDAO utilizando JPA.

Atributos

em

```
private EntityManager em
```

The em.

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

set entityManager

Specified by: [setEntityManager](#) in interface [RoleDAO](#)

listRoles

```
@org.springframework.transaction.annotation.Transactional  
public List<Role> listRoles()
```

Método encargado de listar todos los roles

Specified by: [listRoles](#) in interface [RoleDAO](#)

Returns: List

saveRole

```
@org.springframework.transaction.annotation.Transactional
public Role saveRole(Role role)
```

Se encarga de guardar un rol

Specified by: [saveRole](#) in interface [RoleDAO](#)

Returns:Role

updateRole

```
@org.springframework.transaction.annotation.Transactional
public Role updateRole(Role role)
```

Se encarga de actualizar un Rol

Specified by: [updateRole](#) in interface [RoleDAO](#)

Returns:Role

deleteRole

```
@org.springframework.transaction.annotation.Transactional
public Role deleteRole(Role role)
```

Se encarga de eliminar un rol

Specified by: [deleteRole](#) in interface [RoleDAO](#)

Returns:Role

findRoleById

```
public Role findRoleById(Integer id)
```

Se encarga de buscar un rol porId

Specified by: [findRoleById](#) in interface [RoleDAO](#)

14.9.1.19 Interface RoleUserDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[RoleUserDAOImpl](#)

```
@org.springframework.stereotype.Component
public interface RoleUserDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad RoleUser

Detalle de los métodos

listRoleUsers

```
List<RoleUser> listRoleUsers()
```

Lista todos los RoleUsers

Returns:List

saveRoleUser

```
RoleUser saveRoleUser(RoleUser roleUser)
```

Guarda un RoleUser en la base de datos

Returns:RoleUser

updateRoleUser

```
RoleUser updateRoleUser(RoleUser roleUser)
```

Actualiza un RoleUser de la base de datos

Returns: RoleUser

deleteRoleUser

```
RoleUser deleteRoleUser(RoleUser roleUser)
```

Elimina un RoleUser de la base de datos

Returns:RoleUser

findRoleUserById

```
RoleUser findRoleUserById(Integer id)
```

Busca un RoleUser por id

Returns:RoleUser

setEntityManager

```
void setEntityManager(EntityManager em)
```

Sets the entity manager.

Parameters:em - the new entity manager

14.9.1.20 Class RoleUserDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.RoleUserDAOImpl`

All Implemented Interfaces:

[RoleUserDAO](#)

```
@org.springframework.stereotype.Repository
public class RoleUserDAOImpl extends Object implements RoleUserDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz RoleUserDAO utilizando JPA

Atributos

em

```
private EntityManager em
```

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: `setEntityManager` in interface [RoleUserDAO](#)

Parameters: `em` - the new entity manager

listRoleUsers

```
public List<RoleUser> listRoleUsers()
```

Lista todos los RoleUsers

Specified by: `listRoleUsers` in interface [RoleUserDAO](#)

Returns: List

saveRoleUser

```
public RoleUser saveRoleUser(RoleUser roleUser)
```

Guarda un RoleUser en la base de datos

Specified by: `saveRoleUser` in interface [RoleUserDAO](#)

Returns: RoleUser

updateRoleUser

```
public RoleUser updateRoleUser(RoleUser roleUser)
```

Actualiza un RoleUser de la base de datos

Specified by: [updateRoleUser](#) in interface [RoleUserDAO](#)

Returns: RoleUser

deleteRoleUser

```
public RoleUser deleteRoleUser(RoleUser roleUser)
```

Elimina un RoleUser de la base de datos

Specified by: [deleteRoleUser](#) in interface [RoleUserDAO](#)

Returns: RoleUser

findRoleUserById

```
public RoleUser findRoleUserById(Integer id)
```

Busca un RoleUser por id

Specified by: [findRoleUserById](#) in interface [RoleUserDAO](#)

Returns: RoleUser

14.9.1.21 Interface ShopDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[ShopDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface ShopDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad Shop

Detalle de los métodos

listShops

```
List<Shop> listShops()
```

Lista el conjunto de entidades de tipo Shop que hay en la base de datos

Returns: List

saveShop

```
Shop saveShop(Shop shop)
```

Guarda una Shop en la base de datos

Returns:Shop

updateShop

[Shop](#) updateShop ([Shop](#) shop)

Actualiza una Shop de la base de datos

Returns:Shop

deleteShop

[Shop](#) deleteShop ([Shop](#) shop)

Elimina una Shop de la base de datos

Returns:shop

findShopById

[Shop](#) findShopById (Integer id)

Busca una shop por id

Returns:Shop

findShopByType

List<[Shop](#)> findShopByType (String type)

Busca una shop por tipo (el tipo puede ser Prestashop o Magento)

Returns: Shops

findShopByName

List<[Shop](#)> findShopByName (String name)

Busca una shop por nombre

Returns:shops

setEntityManager

void setEntityManager (EntityManager em)

Sets the entity manager.

Parameters:em - the new entity manager

listShopByUser

```
List<Shop> listShopByUser(Integer userId)
```

Lista el conjunto de tiendas de un usuario

Returns:List

14.9.1.22 Class ShopDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.ShopDAOImpl`

All Implemented Interfaces:

[ShopDAO](#)

```
@org.springframework.stereotype.Repository  
public class ShopDAOImpl extends Object implements ShopDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz ShopDAO utilizando JPA.

Atributos

em

```
private EntityManager em
```

Detalle de los métodos

findCategoryByName

```
public Category findCategoryByName(String name)
```

listShops

```
public List<Shop> listShops()
```

Lista el conjunto de entidades de tipo Shop que hay en la base de datos

Specified by: `listShops` in interface [ShopDAO](#)

Returns:List

saveShop

```
public Shop saveShop(Shop shop)
```

Guarda una Shop en la base de datos

Specified by: `saveShop` in interface [ShopDAO](#)

Returns: Shop

updateShop

```
public Shop updateShop(Shop shop)
```

Actualiza una Shop de la base de datos

Specified by: `updateShop` in interface [ShopDAO](#)

Returns: Shop

deleteShop

```
public Shop deleteShop(Shop shop)
```

Elimina una Shop de la base de datos

Specified by: `deleteShop` in interface [ShopDAO](#)

Returns: shop

findShopById

```
public Shop findShopById(Integer id)
```

Busca una shop por id

Specified by: `findShopById` in interface [ShopDAO](#)

Returns: Shop

listShopByUser

```
public List<Shop> listShopByUser(Integer userId)
```

Lista el conjunto de tiendas de un usuario

Specified by: `listShopByUser` in interface [ShopDAO](#)

Returns: List

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: `setEntityManager` in interface [ShopDAO](#)

Parameters: `em` - the new entity manager

findShopByType

```
public List<Shop> findShopByType(String type)
```

Busca una shop por tipo (el tipo puede ser Prestashop o Magento)

Specified by: `findShopByType` in interface [ShopDAO](#)

Returns:Shop

findShopByName

```
public List<Shop> findShopByName (String name)
```

Busca una shop por nombre

Specified by: [findShopByName](#) in interface [ShopDAO](#)

Returns:shop

14.9.1.23 Interface UserCategoryDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[UserCategoryDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface UserCategoryDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad UserCategory

Detalle de los métodos

listUserCategories

```
List<UserCategory> listUserCategories ()
```

Lista todas las UserCategories que hay en la base de datos

Returns:List

saveUserCategory

```
UserCategory saveUserCategory (UserCategory userCategory)
```

Guarda una UserCategory en la base de datos

Returns:UserCategory

updateUserCategory

```
UserCategory updateUserCategory (UserCategory userCategory)
```

Actualiza una UserCategory en la base de datos

Returns:UserCategory

deleteUserCategory

```
UserCategory deleteUserCategory (UserCategory userCategory)
```

Elimina una UserCategory en la base de datos

Returns:UserCategory

findUserCategoryById

[UserCategory](#) findUserCategoryById(Integer id)

Busca una UserCategory por id

Returns:UserCategory

findUserCategoryByCategory

List<[UserCategory](#)> findUserCategoryByCategory([Category](#) category)

Lista las UserCategory por categoría

Returns:List

setEntityManager

void setEntityManager(EntityManager em)

Sets the entity manager.

Parameters:em - the new entity manager

14.9.1.24 Class UserCategoryDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└ com.tunos.commerce.dao.UserCategoryDAOImpl

All Implemented Interfaces:

[UserCategoryDAO](#)

@org.springframework.stereotype.Repository

public class **UserCategoryDAOImpl** extends Object implements [UserCategoryDAO](#)

Clase que implementa los métodos de persistencia declarados en la interfaz UserCategoryDAO utilizando JPA.

Atributos

em

private EntityManager **em**

The em.

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: [setEntityManager](#) in interface [UserCategoryDAO](#)

Parameters: em - the new entity manager

listUserCategories

```
public List<UserCategory> listUserCategories()
```

Lista todas las UserCategories qye hay en la base de datos

Specified by: [listUserCategories](#) in interface [UserCategoryDAO](#)

Returns: List

saveUserCategory

```
public UserCategory saveUserCategory(UserCategory userCategory)
```

Guarda una UserCategory en la base de datos

Specified by: [saveUserCategory](#) in interface [UserCategoryDAO](#)

Returns: UserCategory

updateUserCategory

```
public UserCategory updateUserCategory(UserCategory userCategory)
```

Actualiza una UserCategory en la base de datos

Specified by: [updateUserCategory](#) in interface [UserCategoryDAO](#)

Returns: UserCategory

deleteUserCategory

```
public UserCategory deleteUserCategory(UserCategory userCategory)
```

Elimina una UserCategory en la base de datos

Specified by: [deleteUserCategory](#) in interface [UserCategoryDAO](#)

Returns: UserCategory

findUserCategoryById

```
public UserCategory findUserCategoryById(Integer id)
```

Busca una UserCategory por id

Specified by: [findUserCategoryById](#) in interface [UserCategoryDAO](#)

Returns: UserCategory

findUserCategoryByCategory

```
public List<UserCategory> findUserCategoryByCategory (Category category)
```

Lista las UserCategory por categoría

Specified by: `findUserCategoryByCategory` in interface [UserCategoryDAO](#)

14.9.1.25 Interface UserDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[UserDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface UserDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad User

Detalle de los métodos

listUsers

```
List<User> listUsers ()
```

Lista los usuarios.

Returns: the list

saveUser

```
User saveUser (User user)
```

Guarda un usuario.

Parameters: `user` - the user

Returns: the user

updateUser

```
User updateUser (User user)
```

Actualiza un usuario.

Parameters: `user` - the user

Returns: the user

deleteUser

```
User deleteUser (User user)
```

Elimina un usuario.

Parameters: `user` - the user

Returns: the user

findUserById

[User](#) **findUserById**(Integer id)

Busca un usuario por Id

Parameters: `id` - the id

Returns: the user

findUserByUsername

[User](#) **findUserByUsername**(String username)

Busca un usuario por username

Parameters: `username` - the username

Returns: the user

setEntityManager

void **setEntityManager**(EntityManager em)

Sets the entity manager.

Parameters: `em` - the new entity manager

listUserByRoleId

List<Integer> **listUserByRoleId**(Integer role_id)

Lista los usuarios por role_id

14.9.1.26 Class UserDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.UserDAOImpl`

All Implemented Interfaces:

[UserDAO](#)

```
@org.springframework.stereotype.Repository  
public class UserDAOImpl extends Object implements UserDAO
```

Clase que implementa los métodos de persistencia declarados en la interfaz UserDAO utilizando JPA

Atributos

em

```
private EntityManager em
```

The em.

Detalle de los métodos

setEntityManager

```
public void setEntityManager(EntityManager em)
```

Sets the entity manager.

Specified by: [setEntityManager](#) in interface [UserDAO](#)

Parameters: `em` - the new entity manager

listUsers

```
public List<User> listUsers()
```

Lista los usuarios.

Specified by: [listUsers](#) in interface [UserDAO](#)

Returns: the list

saveUser

```
public User saveUser(User user)
```

Guarda un usuario.

Specified by: [saveUser](#) in interface [UserDAO](#)

Parameters: `user` - the user

Returns: the user

updateUser

```
public User updateUser(User user)
```

Actualiza un usuario.

Specified by: [updateUser](#) in interface [UserDAO](#)

Parameters: `user` - the user

Returns: the user

deleteUser

```
public User deleteUser(User user)
```

Elimina un usuario.

Specified by: `deleteUser` in interface [UserDAO](#)

Parameters: `user` - the user

Returns: the user

findUserById

```
public User findUserById(Integer id)
```

Busca un usuario por Id

Specified by: `findUserById` in interface [UserDAO](#)

Parameters: `id` - the id

Returns: the user

findUserByUsername

```
public User findUserByUsername(String username)
```

Busca un usuario por username

Specified by: `findUserByUsername` in interface [UserDAO](#)

Parameters: `username` - the username

Returns: the user

listUserByRoleId

```
public List<Integer> listUserByRoleId(Integer role_id)
```

Lista los usuarios por role_id

Specified by: `listUserByRoleId` in interface [UserDAO](#)

14.9.1.27 Interface UserProductDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[UserProductDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface UserProductDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad UserProduct

Detalle de los métodos

listUserProducts

```
List<UserProduct> listUserProducts ()
```

Lista todos los user products

Returns:List

saveUserProduct

```
UserProduct saveUserProduct (UserProduct UserProduct)
```

Guarda un userProduct

Returns:UserProduct

updateUserProduct

```
UserProduct updateUserProduct (UserProduct UserProduct)
```

Actualiza un UserProduct

Returns:UserProduct

deleteUserProduct

```
UserProduct deleteUserProduct (UserProduct UserProduct)
```

Elimina un userProduct

Returns:UserProduct

findUserProductById

```
UserProduct findUserProductById (Integer id)
```

Busca un UserProduct por id

Returns:UserProduct

setEntityManager

```
void setEntityManager (EntityManager em)
```

Sets the entity manager.

Parameters:em - the new entity manager

findUserProductByProduct

```
List<UserProduct> findUserProductByProduct (Product product)
```

Busca un UserProduct por product

Returns:UserProduct

14.9.1.28 Class UserProductDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.UserProductDAOImpl`

All Implemented Interfaces:

[UserProductDAO](#)

@org.springframework.stereotype.Repository
public class **UserProductDAOImpl** extends Object implements [UserProductDAO](#)

Clase que implementa los métodos de persistencia declarados en la interfaz UserProductDAO utilizando JPA

Atributos

em

private EntityManager **em**

The em.

Detalle de los métodos

setEntityManager

public void **setEntityManager**(EntityManager em)

Sets the entity manager.

Specified by: [setEntityManager](#) in interface [UserProductDAO](#)

Parameters:em - the new entity manager

listUserProducts

public List<[UserProduct](#)> **listUserProducts**()

Lista todos los user products

Specified by: [listUserProducts](#) in interface [UserProductDAO](#)

Returns:List

saveUserProduct

public [UserProduct](#) **saveUserProduct**([UserProduct](#) userProduct)

Guarda un userProduct

Specified by: [saveUserProduct](#) in interface [UserProductDAO](#)

Returns:UserProduct

updateUserProduct

```
public UserProduct updateUserProduct(UserProduct userProduct)
```

Actualiza un UserProduct

Specified by: `updateUserProduct` in interface [UserProductDAO](#)

Returns:UserProduct

deleteUserProduct

```
public UserProduct deleteUserProduct(UserProduct UserProduct)
```

Elimina un userProduct

Specified by: `deleteUserProduct` in interface [UserProductDAO](#)

Returns:UserProduct

findUserProductById

```
public UserProduct findUserProductById(Integer id)
```

Busca un UserProduct por id

Specified by: `findUserProductById` in interface [UserProductDAO](#)

Returns:UserProduct

findUserProductByProduct

```
public List<UserProduct> findUserProductByProduct(Product product)
```

Busca un UserProduct por product

Specified by: `findUserProductByProduct` in interface [UserProductDAO](#)

Returns:UserProduct

14.9.1.29 Interface *UserProviderDAO*

com.tunos.commerce.dao

All Known Implementing Classes:

[UserProviderDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface UserProviderDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad UserProvider

Detalle de los métodos

listUserProviders

List<[UserProvider](#)> **listUserProviders**()

Lista todos los UserProviders de la base de datos

Returns:List

saveUserProvider

[UserProvider](#) **saveUserProvider**([UserProvider](#) userProvider)

Guarda un UserProvider

Returns:UserProvider

updateUserProvider

[UserProvider](#) **updateUserProvider**([UserProvider](#) userProvider)

Actualiza un UserProvider

Returns:userProvider

deleteUserProvider

[UserProvider](#) **deleteUserProvider**([UserProvider](#) userProvider)

Elimina un UserProvider

Returns:userProvider

findUserProviderById

[UserProvider](#) **findUserProviderById**(Integer id)

Busca un UserProvider por id

Returns:UserProvider

setEntityManager

void **setEntityManager**(EntityManager em)

Sets the entity manager.

Parameters:em - the new entity manager

findUserProviderByProvider

List<[UserProvider](#)> **findUserProviderByProvider**([Provider](#) p)

Busca un UserProvider por proveedor

Returns:UserProvider

14.9.1.30 Class UserProviderDAOImpl

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.UserProviderDAOImpl`

All Implemented Interfaces:

[UserProviderDAO](#)

@org.springframework.stereotype.Repository
public class **UserProviderDAOImpl** extends Object implements [UserProviderDAO](#)

Clase que implementa los métodos de persistencia declarados en la interfaz UserProviderDAO utilizando JPA

Atributos

em

private EntityManager **em**

The em.

Detalle de los métodos

setEntityManager

public void **setEntityManager**(EntityManager em)

Sets the entity manager.

Specified by: `setEntityManager` in interface [UserProviderDAO](#)

Parameters:em - the new entity manager

listUserProviders

public List<[UserProvider](#)> **listUserProviders**()

Lista todos los UserProviders de la base de datos

Specified by: `listUserProviders` in interface [UserProviderDAO](#)

Returns:List

saveUserProvider

public [UserProvider](#) **saveUserProvider**([UserProvider](#) userProvider)

Guarda un UserProvider

Specified by: [saveUserProvider](#) in interface [UserProviderDAO](#)

Returns: UserProvider

updateUserProvider

```
public UserProvider updateUserProvider(UserProvider userProvider)
```

Actualiza un UserProvider

Specified by: [updateUserProvider](#) in interface [UserProviderDAO](#)

Returns: userProvider

deleteUserProvider

```
public UserProvider deleteUserProvider(UserProvider userProvider)
```

Elimina un UserProvider

Specified by: [deleteUserProvider](#) in interface [UserProviderDAO](#)

Returns: userProvider

findUserProviderById

```
public UserProvider findUserProviderById(Integer id)
```

Busca un UserProvider por id

Specified by: [findUserProviderById](#) in interface [UserProviderDAO](#)

Returns: UserProvider

findUserProviderByProvider

```
public List<UserProvider> findUserProviderByProvider(Provider p)
```

Busca un UserProvider por proveedor

Specified by: [findUserProviderByProvider](#) in interface [UserProviderDAO](#)

Returns: UserProvider

14.9.1.31 Interface UserShopDAO

[com.tunos.commerce.dao](#)

All Known Implementing Classes:

[UserShopDAOImpl](#)

```
@org.springframework.stereotype.Component  
public interface UserShopDAO
```

Interfaz de los métodos de persistencia y recuperación de datos de la entidad UserSShop

Detalle de los métodos

listUserShops

```
List<UserShop> listUserShops ()
```

Lista el conjunto de UserShop

Returns:List

saveUserShop

```
UserShop saveUserShop (UserShop userShop)
```

Guarda un UserShop

Returns:UserShop

updateUserShop

```
UserShop updateUserShop (UserShop userShop)
```

Actualiza un UserShop

Returns:UserShop

deleteUserShop

```
UserShop deleteUserShop (UserShop userShop)
```

Elimina un UserShop

Returns:UserShop

findUserShopById

```
UserShop findUserShopById (Integer id)
```

Busca un userShop por id

Returns:UserShop

findUserShopByShop

```
List<UserShop> findUserShopByShop (Shop shop)
```

Buscar UserShop por shop

Returns:List

setEntityManager

```
void setEntityManager (EntityManager em)
```

Sets the entity manager.

Parameters:em - the new entity manager

14.9.1.32 Class *UserShopDAOImpl*

[com.tunos.commerce.dao](#)

java.lang.Object

└─ `com.tunos.commerce.dao.UserShopDAOImpl`

All Implemented Interfaces:

[UserShopDAO](#)

@org.springframework.stereotype.Repository

public class **UserShopDAOImpl** extends Object implements [UserShopDAO](#)

Clase que implementa los métodos de persistencia declarados en la interfaz UseShopDAO utilizando JPA

Atributos

em

private EntityManager **em**

The em.

Detalle de los métodos

setEntityManager

public void **setEntityManager**(EntityManager em)

Sets the entity manager.

Specified by: [setEntityManager](#) in interface [UserShopDAO](#)

Parameters:em - the new entity manager

listUserShops

public List<[UserShop](#)> **listUserShops**()

Lista el conjunto de UserShop

Specified by: [listUserShops](#) in interface [UserShopDAO](#)

Returns: List

saveUserShop

public [UserShop](#) **saveUserShop**([UserShop](#) userShop)

Guarda un UserShop

Specified by: `saveUserShop` in interface [UserShopDAO](#)

Returns: UserShop

updateUserShop

```
public UserShop updateUserShop(UserShop userShop)
```

Actualizar un UserShop

Specified by: `updateUserShop` in interface [UserShopDAO](#)

Returns: UserShop

deleteUserShop

```
public UserShop deleteUserShop(UserShop userShop)
```

Elimina un UserShop

Specified by: `deleteUserShop` in interface [UserShopDAO](#)

Returns: UserShop

findUserShopById

```
public UserShop findUserShopById(Integer id)
```

Busca un userShop por id

Specified by: `findUserShopById` in interface [UserShopDAO](#)

Returns: UserShop

findUserShopByShop

```
public List<UserShop> findUserShopByShop(Shop shop)
```

Buscar UserShop por shop

Specified by: `findUserShopByShop` in interface [UserShopDAO](#)

Returns: List

14.9.1.33 Class AbstractTimestampEntity

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└ com.tunos.commerce.entities.AbstractTimestampEntity
```

Direct Known Subclasses:

[Provider](#), [Role](#), [User](#)

```
public class AbstractTimestampEntity extends Object
```

The Class AbstractTimestampEntity

Atributos

created

```
private Date created
```

The created.

updated

```
private Date updated
```

The updated.

Detalle de los métodos

onCreate

```
protected void onCreate()
```

On create.

onUpdate

```
protected void onUpdate()
```

On update.

getCreated

```
public Date getCreated()
```

Gets the created.

Returns:the created

getUpdated

```
public Date getUpdated()
```

Gets the updated.

Returns:the updated

14.9.1.34 Class Category

[com.tunos.commerce.entities](#)

java.lang.Object

└─ `com.tunos.commerce.entities.Category`

All Implemented Interfaces:

Serializable

```
public class Category extends Object implements Serializable
```

Clase que representa una Categoría dentro del sistema

Atributos

id

```
private Integer id
```

name

```
private String name
```

description

```
private String description
```

level

```
private Integer level
```

active

```
private boolean active
```

root

```
private boolean root
```

parent

```
private Category parent
```

14.9.1.35 Class Product

[com.tunos.commerce.entities](#)

java.lang.Object

└─ com.tunos.commerce.entities.Product

All Implemented Interfaces:

Serializable

```
public class Product extends Object implements Serializable
```

Clase que representa una Producto dentro del sistema

Atributos

id

```
private Integer id
```

name

```
private String name
```

active

```
private boolean active
```

price

```
private float price
```

ref

```
private String ref
```

shortDescription

```
private String shortDescription
```

longDescription

```
private String longDescription
```

stock

```
private Integer stock
```

hight

```
private float hight
```

weight

```
private float weight
```

provider

```
private Provider provider
```

14.9.1.36 Class ProductCategory

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.ProductCategory
```

All Implemented Interfaces:

Serializable

```
public class ProductCategory extends Object implements Serializable
```

Clase que representa una Product_category dentro del sistema

Atributos

id

```
private Integer id
```

product

```
private Product product
```

category

private [Category](#) category

14.9.1.37 Class Provider

[com.tunos.commerce.entities](#)

java.lang.Object

└ [com.tunos.commerce.entities.AbstractTimestampEntity](#)

└ com.tunos.commerce.entities.Provider

All Implemented Interfaces:

Serializable

public class **Provider** extends [AbstractTimestampEntity](#) implements Serializable

Clase que representa un proveedor dentro del sistema

Atributos

id

private Integer id

nif

private String nif

company

private String company

14.9.1.38 Class Role

[com.tunos.commerce.entities](#)

java.lang.Object

└ [com.tunos.commerce.entities.AbstractTimestampEntity](#)

└ com.tunos.commerce.entities.Role

All Implemented Interfaces:

Serializable

```
public class Role extends AbstractTimestampEntity implements Serializable
```

Clase que representa un rol dentro del sistema

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

description

```
private String description
```

14.9.1.39 Class RoleUser

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.RoleUser
```

All Implemented Interfaces:

Serializable

```
public class RoleUser extends Object implements Serializable
```

Clase que representa un rol_user dentro del sistema

Atributos

id

```
private Integer id
```

role

```
private Role role
```

user

```
private User user
```

14.9.1.40 Class Shop

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.Shop
```

All Implemented Interfaces:

Serializable

```
public class Shop extends Object implements Serializable
```

Clase que representa una tienda dentro del sistema

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

type

```
private String type
```

hostname

```
private String hostname
```

port

```
private String port
```

username

```
private String username
```

pass

```
private String pass
```

dbName

```
private String dbName
```

14.9.1.41 Class User

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.AbstractTimestampEntity
```

```
    └─ com.tunos.commerce.entities.User
```

All Implemented Interfaces:

Serializable

```
public class User extends AbstractTimestampEntity implements Serializable
```

Clase que representa al usuario dentro del sistema. Se guardará toda la información necesaria sobre su identidad.

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

surname

```
private String surname
```

email

```
private String email
```

username

```
private String username
```

password

```
private String password
```

userType

```
private Integer userType
```

isActive

```
private Boolean isActive
```

14.9.1.42 Class UserCategory

com.tunos.commerce.entities

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.UserCategory
```

All Implemented Interfaces:

Serializable

```
public class UserCategory extends Object implements Serializable
```

Clase que representa una entidad de tipo user_category dentro del sistema

Atributos

id

```
private Integer id
```

user

```
private User user
```

category

```
private Category category
```

14.9.1.43 Class UserProduct

[com.tunos.commerce.entities](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.entities.UserProduct
```

All Implemented Interfaces:

Serializable

```
public class UserProduct extends Object implements Serializable
```

Clase que representa una entidad de tipo user_product dentro del sistema

Atributos

id

```
private Integer id
```

user

```
private User user
```

product

```
private Product product
```

14.9.1.44 *Class UserProvider*

[com.tunos.commerce.entities](#)

java.lang.Object

└─ `com.tunos.commerce.entities.UserProvider`

All Implemented Interfaces:

Serializable

```
public class UserProvider extends Object implements Serializable
```

Clase que representa una entidad de tipo user_provider dentro del sistema

Atributos

id

```
private Integer id
```

user

```
private User user
```

provider

```
private Provider provider
```

14.9.1.45 *Class UserShop*

[com.tunos.commerce.entities](#)

java.lang.Object

└─ `com.tunos.commerce.entities.UserShop`

All Implemented Interfaces:

Serializable

```
public class UserShop extends Object implements Serializable
```

Clase que representa una entidad de tipo user_shop dentro del sistema

Atributos

id

```
private Integer id
```

user

```
private User user
```

shop

```
private Shop shop
```

14.9.1.46 Class *CategoryForm*

com.tunos.commerce.forms

```
java.lang.Object
```

```
└─ com.tunos.commerce.forms.CategoryForm
```

All Implemented Interfaces:

Serializable

```
public class CategoryForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad Category en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

description

```
private String description
```

level

```
private Integer level
```

active

```
private boolean active
```

root

```
private boolean root
```

parent

```
private Category parent
```

childs

```
private List<Category> childs
```

listShop

```
private List<SelectItem> listShop
```

exportShop

```
private String exportShop
```

14.9.1.47 Class ParserForm

[com.tunos.commerce.forms](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.forms.ParserForm
```

All Implemented Interfaces:

Serializable

```
public class ParserForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad Parser en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

description

```
private String description
```

14.9.1.48 Class ProductForm

com.tunos.commerce.forms

```
java.lang.Object
```

```
└─ com.tunos.commerce.forms.ProductForm
```

All Implemented Interfaces:

Serializable

```
public class ProductForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad Product en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

active

```
private boolean active
```

price

```
private float price
```

shortDescription

```
private String shortDescription
```

longDescription

```
private String longDescription
```

stock

```
private Integer stock
```

hight

```
private float hight
```

weight

```
private float weight
```

provider

```
private Provider provider
```

categories

```
private List<Category> categories
```

ref

```
private String ref
```

listShop

```
private List<SelectItem> listShop
```

14.9.1.49 Class ProviderForm

com.tunos.commerce.forms

```
java.lang.Object
```

```
└─ com.tunos.commerce.forms.ProviderForm
```

All Implemented Interfaces:

Serializable

```
public class ProviderForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad Provider en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

The Constant serialVersionUID.

id

```
private Integer id
```

company

```
private String company
```

nif

```
private String nif
```

created

```
private Date created
```

updated

```
private Date updated
```

listShop

```
private List<SelectItem> listShop
```

exportShop

```
private String exportShop
```

14.9.1.50 Class ShopForm

com.tunos.commerce.forms

```
java.lang.Object
```

```
└─ com.tunos.commerce.forms.ShopForm
```

All Implemented Interfaces:

Serializable

```
public class ShopForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad Shop en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private Integer id
```

name

```
private String name
```

type

```
private String type
```

hostname

```
private String hostname
```

port

```
private String port
```

username

```
private String username
```

pass

```
private String pass
```

repass

```
private String repass
```

dbName

```
private String dbName
```

listTypes

```
private List<SelectItem> listTypes
```

activeCategory

```
private boolean activeCategory
```

activeProduct

```
private boolean activeProduct
```

activeProvider

```
private boolean activeProvider
```

14.9.1.51 Class *UserForm*

[com.tunos.commerce.forms](#)

java.lang.Object

└─ `com.tunos.commerce.forms.UserForm`

All Implemented Interfaces:

Serializable

```
public class UserForm extends Object implements Serializable
```

Clase que representa la información de los formularios utilizados para la gestión de la entidad User en la capa de presentación

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

id

```
private int id
```

name

```
private String name
```

surname

```
private String surname
```

email

```
private String email
```

username

```
private String username
```

password

```
private String password
```

repassword

```
private String repassword
```

14.9.1.52 Class DriverConnection

[com.tunos.commerce.intercomunicacion](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.intercomunicacion.DriverConnection
```

```
public class DriverConnection extends Object
```

Clase encargada de establecer las conexiones externas

Atributos

conexion

```
private Connection conexion
```

Constructor Detail

DriverConnection

```
public DriverConnection()
```

constructor

Detalle de los métodos

establishConnection

```
public void establishConnection(String hostname, String dbName,String port,String user,String password)throws ClassNotFoundException,SQLException
```

Establece la conexion mysql a una base de datos externa

Throws:

ClassNotFoundException
SQLException

getConnection

```
public Connection getConnection()
```

Obtiene la conexion

Returns:connection

closeConnection

```
public void closeConnection(ResultSet rs) throws SQLException
```

Cierra la conexión

Throws:SQLException

closeStatment

```
public void closeStatment(Statement stmt) throws SQLException
```

Cerrar el statment

Throws:SQLException

removeConnection

```
public void removeConnection() throws SQLException
```

Eliminar la conexión

Throws:SQLException

14.9.1.53 Class ExportElementsMagento

com.tunos.commerce.intercommunication.imports.exports

```
java.lang.Object
```

```
└ com.tunos.commerce.intercommunication.imports.exports.ExportElementsMagento
```

```
public class ExportElementsMagento extends Object
```

clase que almacena las funcionalidades necesarias para exportar elementos (categorías, productos o proveedores) a Magento

Atributos

conexion

```
private Connection conexion
```

st

```
private Statement st
```

c

```
private DriverConnection c
```

Constructor Detail**ExportElementsMagento**

```
public ExportElementsMagento(String hostname,String dbName,String port,String user,String password)throws SQLException,ClassNotFoundException
```

Constructor. Instancia un DriverConection y abre una conexion con Magento

Throws:

SQLException

ClassNotFoundException

Detalle de los métodos**insertProvider**

```
public void insertProvider(ProviderForm p)throws SQLException
```

Inserta un proveedor en la base de datos de magento

Throws:SQLException

insertProduct

```
public void insertProduct(ProductForm p) throws SQLException
```

Inserta un producto en la base de datos de magento

Throws:SQLException

getIdCategories

```
private List<Integer> getIdCategories(List<Category> categories)throws SQLException
```

Coge el id de las categorías existentes en Magento

Returns:List categorias

Throws:SQLException

insertCategory

```
public void insertCategory(CategoryForm category) throws SQLException
```

Inserta una categoría en la base de datos de Magento

Throws:SQLException

getPositionCategory

```
private int getPositionCategory(int parent_id) throws SQLException
```

Recoge la posición en la que se encuentra una categoría

Returns: posicion

Throws:SQLException

getPath

```
private String getPath(int id_parent,int entity_id) throws SQLException
```

Get path

Returns: get path

Throws:SQLException

getParentId

```
private Category getParentId(int id) throws SQLException
```

Coger el id de una categoría

Returns: parentCategory

Throws:SQLException

getParentId

```
public Category getParentId(String nameCategory)throws SQLException
```

Obtiene el padre de una categoría

Returns: parentCategory

Throws:SQLException

getId

```
private int getId(String nameCategory) throws SQLException
```

Obtiene el id de una categoría

Returns:entero

Throws:SQLException

getProvider

```
private int getProvider(String provider) throws SQLException
```

Obtiene el id de un proveedor

Returns : id

Throws: SQLException

14.9.1.54 Class *ExportElementsPrestashop*

com.tunos.commerce.intercommunication.imports.exports

java.lang.Object

L

com.tunos.commerce.intercommunication.imports.exports.ExportElementsPrestashop

```
public class ExportElementsPrestashop extends Object
```

clase que recoge todas las funcionalidades necesarias para exportar elementos (productos,categorías o proveedores) a Prestashop

Atributos

conexion

```
private Connection conexion
```

st

```
private Statement st
```

c

```
private DriverConnection c
```

nleft

```
private int nleft
```

nright

```
private int nright
```

n

```
private int n
```

Constructor Detail

ExportElementsPrestashop

```
public ExportElementsPrestashop(String hostname,String dbName,String port,String user,String password)throws SQLException,ClassNotFoundException
```

Constructor. Instancia un DriverConection y abre una conexion con Prestashop

Throws:SQLException,ClassNotFoundException

Detalle de los métodos

insertProvider

```
public void insertProvider(ProviderForm p)throws SQLException
```

Inserta un proveedor en la base de datos de Prestashop

Throws:SQLException

insertProduct

```
public void insertProduct(ProductForm p) throws SQLException
```

Inserta un producto en la base de datos de Prestashop

Throws:SQLException

insertCategory

```
public void insertCategory(CategoryForm category) throws SQLException
```

Inserta una categoría en la base de datos de Prestashop

Throws:SQLException

reGenerateEntireTree

```
private void reGenerateEntireTree()
```

Método encargado de regenerar el árbol de categorías de prestahop una vez se inserta una nueva

SubTree

```
private int SubTree(Integer[][] categories,int id_category, int catCalculate)
```

Método encargado de ir recalculando árboles de subcategorías en prestashop

Returns:n

getSubcategories

```
private List<Integer> getSubcategories(int category, Integer[][] categories)
```

Obtener la lista de subcategorías de una categoría

Returns: lista de enteros con los ids de las subcategorías

getNumCategoriasMismoNivel

```
private int getNumCategoriasMismoNivel(int level)
```

Obtiene el número de categorías existentes en el mismo nivel que el que se pasa como parámetro

Returns: número de categorías del mismo nivel

14.9.1.55 Class ImportElementsMagento

com.tunos.commerce.intercommunication.imports.exports

```
java.lang.Object
```

```
└ com.tunos.commerce.intercommunication.imports.exports.ImportElementsMagento
```

```
public class ImportElementsMagento extends Object
```

Obtiene todas las funcionalidades necesarias para importar elementos(productos, categorías o proveedores) de Magento

Atributos

conexion

```
private Connection conexion
```

st

```
private Statement st
```

c

```
private DriverConnection c
```

Constructor Detail

ImportElementsMagento

```
public ImportElementsMagento(String hostname, String dbName, String port, String user, String password) throws SQLException, ClassNotFoundException
```

Constructor. Instancia un DriverConection y abre una conexión con Magento

Throws:SQLException,ClassNotFoundException

Detalle de los métodos

closeConnection

```
public void closeConnection() throws SQLException
```

cierra la conexión actual

Throws:SQLException

getProviders

```
public List<Provider> getProviders() throws SQLException
```

Obtiene una lista de proveedores de la base de datos de Magento

Returns:List proveedores

Throws:SQLException

getCategories

```
public List<Category> getCategories() throws SQLException
```

Obtiene una lista de categorías de la base de datos de Magento

Returns:List categorías

Throws:SQLException

getProducts

```
public List<Product> getProducts() throws SQLException
```

Obtiene una lista de productos de la base de datos de Magento

Returns: List productos

Throws:SQLException

getNameCategoryParentFromCategory

```
public String getNameCategoryParentFromCategory(String nameCategory) throws SQLException
```

Obtiene el nombre de la categoría del padre de una categoría

Returns: nombre de la categoría padre

Throws:SQLException

getNameProviderFromProduct

```
public String getNameProviderFromProduct(String nameProduct) throws SQLException
```


Obtiene el nombre del proveedor de un producto

Returns: nombre del proveedor

Throws:SQLException

getNameCategoryFromProduct

```
public String getNameCategoryFromProduct(String nameProduct)throws SQLException
```

Obtiene el nombre de la categoría de un producto

Returns:nombre de la categoría

Throws:SQLException

14.9.1.56 Class ImportElementsPrestashop

[com.tunos.commerce.intercommunication.imports.exports](#)

```
java.lang.Object
```

```
L
```

```
com.tunos.commerce.intercommunication.imports.exports.ImportElementsPrestashop
```

```
public class ImportElementsPrestashop extends Object
```

Obtiene todas las funcionalidades necesarias para importar elementos (productos,categorías o proveedores) de la base de datos de Prestashop

Atributos

conexion

```
private Connection conexion
```

st

```
private Statement st
```

c

```
private DriverConnection c
```

Constructor Detail

ImportElementsPrestashop

```
public ImportElementsPrestashop(String hostname,String dbName,String port,String user,String password)throws SQLException,ClassNotFoundException
```

Constructor. Instancia un DriverConection y abre una conexion con Prestashop

Throws:SQLException, ClassNotFoundException

Detalle de los métodos

closeConnection

```
public void closeConnection() throws SQLException
```

Cierra la conexion actual

Throws:SQLException

getProducts

```
public List<Product> getProducts() throws SQLException
```

Obtiene la lista de productos de la base de datos de Prestashop

Returns:List products

Throws:SQLException

getCategories

```
public List<Category> getCategories() throws SQLException
```

Obtiene la lista de categorías de la base de datos de Prestashop

Returns:List categorias

Throws:SQLException

getNameCategoryParentFromCategory

```
public String getNameCategoryParentFromCategory(String nameCategory)
                                              throws SQLException
```

Obtiene el nombre de la categoría padre de la categoría que se pasa como parámetro

Returns:nombre de la categoría padre

Throws:SQLException

getNameProviderFromProduct

```
public String getNameProviderFromProduct(String nameProduct) throws SQLException
```

Obtiene el nombre del proveedor de un producto

Returns:nombre del proveedor

Throws:SQLException

getNameCategoryFromProduct

```
public String getNameCategoryFromProduct(String nameProduct) throws SQLException
```

Obtiene el nombre de la categoría de un producto

Returns: nombre de la categoría

Throws: SQLException

getProviders

```
public List<Provider> getProviders() throws SQLException
```

Obtiene la lista de proveedores de la base de datos de Prestashop

Returns: List proveedores

Throws: SQLException

14.9.1.57 Interface CategoryService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[CategoryServiceImpl](#)

```
public interface CategoryService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad Category

Detalle de los métodos

saveCategory

```
Category saveCategory(Category category)
```

Guarda una categoría

Returns: category

updateCategory

```
Category updateCategory(Category category)
```

Actualiza una categoría

Returns: category

deleteCategory

```
Category deleteCategory(Category category)
```

elimina una cateoría

Returns: category

findCategoryById

[Category](#) findCategoryById(Integer id)

Busca una categoría por id

Returns:category

listTotalEntriesCategoriesByUser

```
List<Integer> listTotalEntriesCategoriesByUser(SearchCriteria searchCriteria,  
                                              Integer userId,  
                                              Integer parentId)
```

Lista todas las categorías de un usuario y retorna sus ids

Returns:List ids

listCategoriesByUser

```
List<Category> listCategoriesByUser(SearchCriteria searchCriteria,  
                                   Integer userId,  
                                   Integer parentId)
```

Lista las categorías de un usuario teniendo en cuenta las restricciones del searchCriteria....

Returns:category

listChilds

```
List<Category> listChilds(Integer parentId)
```

Lista los hijos de una categoría

Returns:List childs

findCategoryByName

[Category](#) findCategoryByName(String name, Integer user)

Busca una categoría por nombre

Returns:category

14.9.1.58 Interface ProductCategoryService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[ProductCategoryServiceImpl](#)

```
public interface ProductCategoryService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad ProductCategory

Detalle de los métodos

listProductCategories

```
List<ProductCategory> listProductCategories()
```

Lista el conjunto de entidades productCategory existentes en el sistema

Returns:List productCategory

saveProductCategory

```
ProductCategory saveProductCategory(ProductCategory productCategory)
```

Guarda una entidad de tipo ProductCategory en la base de datos

Returns:ProductCategory

updateProductCategory

```
ProductCategory updateProductCategory(ProductCategory productCategory)
```

Actualiza una entidad de tipo ProductCategory de la base de datos

Returns:ProductCategory

deleteProductCategory

```
ProductCategory deleteProductCategory(ProductCategory productCategory)
```

Elimina de la base de datos un ProductCategory

Returns:ProductCategory

findProductCategoryById

```
ProductCategory findProductCategoryById(Integer id)
```

Busca un ProductCategory por id

Returns:ProductCategory

findByProduct

```
List<ProductCategory> findByProduct(Product p)
```

Busca un productCategory por id

Returns:ProductCategory

findCategoriesFromProduct

```
List<Integer> findCategoriesFromProduct(Integer id)
```

Lista los productos por categoría y retorna sus id's

Returns:List ids

findByCategory

```
List<ProductCategory> findByCategory(Category c)
```

Lista los ProductCategory por categoría

Returns:List productCategories

14.9.1.59 Interface ProductService

com.tunos.commerce.service

All Known Implementing Classes:

[ProductServiceImpl](#)

```
public interface ProductService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad ProductCategory

Detalle de los métodos

listProducts

```
List<Product> listProducts()
```

Lista todos los productos guardados en la base de datos

Returns:List productos

saveProduct

```
Product saveProduct(Product product)
```

Guarda un producto en la base de datos

Returns:Product

listProductsByProvider

```
List<Product> listProductsByProvider(Provider provider)
```

Lista los productos por proveedor

Returns:List productos

updateProduct

[Product](#) updateProduct([Product](#) product)

Actualiza el producto

Returns:Product

deleteProduct

[Product](#) deleteProduct([Product](#) product)

Elimina el product

Returns:Product

findProductById

[Product](#) findProductById(Integer id)

Busca un producto por id

Returns:product

findProductByName

[Product](#) findProductByName(String name)

Busca un producto por nombre

Returns:Product

listProductsByUser

List<[Product](#)> listProductsByUser(Integer idUser)

Lista los productos de un usuario

Returns:List productos

listProductsByUser

List<[Product](#)> listProductsByUser([SearchCriteria](#) criteria, Integer idUser)

Lista los productos de un usuario y teniendo en cuenta las restricciones del
searchCriteria

Returns:List productos

listTotalEntriesProductsByUser

List<Integer> listTotalEntriesProductsByUser([SearchCriteria](#) criteria,Integer idUser)

Lista el conjunto de productos de un usuario

Returns:List productos

14.9.1.60 Interface ProviderService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[ProviderServiceImpl](#)

```
public interface ProviderService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad Provider

Detalle de los métodos

listProviders

```
List<Provider> listProviders()
```

Lista todos los proveedores existentes en la base de datos

Returns:listProviders

saveProvider

```
Provider saveProvider(Provider provider)
```

Guarda un proveedor en la base de datos

Returns:provider

updateProvider

```
Provider updateProvider(Provider provider)
```

Actualiza un proveedor en la base de datos

Returns:provider

deleteProvider

```
Provider deleteProvider(Provider provider)
```

Elimina un proveedor de la base de datos

Returns:provider

findProviderById

```
Provider findProviderById(Integer id)
```


Busca un proveedor por id

Returns:provider

findProviderByNIF

[Provider](#) findProviderByNIF(String nif)

Busca un proveedor por NIF

Returns:provider

findProviderByCompany

[Provider](#) findProviderByCompany(String company,Integer user)

Busca un proveedor por compañía

Returns:Provider

listProviderByIdUser

List<Integer> listProviderByIdUser(Integer idUser)

Lista los proveedores por id de usuario

Returns:List ids_proveedores: identificador de los proveedores

listProviders

List<[Provider](#)> listProviders([SearchCriteria](#) criteria)

Lista los proveedores teniendo en cuenta el searchCriteria

Returns: list providers

listProvidersByUser

List<[Provider](#)> listProvidersByUser([SearchCriteria](#) criteria, Integer idUser)

Lista los proveedores por usuario y teniendo el cuenta el searchCriteria

Returns:List proveedores

listTotalEntriesProvidersByUser

List<Integer> listTotalEntriesProvidersByUser([SearchCriteria](#) criteria,
Integer idUser)

Lista todos los proveedores teniendo en cuenta las restricciones del searchCriteria.

Retorna una lista de enteros con los id's de los proveedores

Returns:List ids

14.9.1.61 *Interface RoleService*

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[RoleServiceImpl](#)

```
public interface RoleService
```

Detalle de los métodos

listRoles

```
List<Role> listRoles()
```

Lista el conjunto de roles contenidos en la base de datos

Returns:List roles

saveRole

```
Role saveRole(Role role)
```

Guarda un rol en la base de datos

Returns:rol

updateRole

```
Role updateRole(Role role)
```

Actualiza un rol de la base de datos

Returns:role

deleteRole

```
Role deleteRole(Role role)
```

Elimina el rol de la base de datos

Returns:role

findRoleById

```
Role findRoleById(Integer id)
```

Busca un rol por id

Returns:role

14.9.1.62 Interface RoleUserService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[RoleUserServiceImpl](#)

```
public interface RoleUserService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad RoleUser

Detalle de los métodos

listRoleUsers

```
List<RoleUser> listRoleUsers ()
```

Lista los RoleUsers que hay en la base de datos

Returns:List roleUsers

saveRoleUser

```
RoleUser saveRoleUser (RoleUser roleUser)
```

Guarda un RoleUser en la base de datos

Returns:RoleUser

updateRoleUser

```
RoleUser updateRoleUser (RoleUser roleUser)
```

Actualiza un RoleUser de la base de datos

Returns:roleUser

deleteRoleUser

```
RoleUser deleteRoleUser (RoleUser roleUser)
```

Elimina un roleUser de la base de datos

Returns:roleUser

findRoleUserById

```
RoleUser findRoleUserById(Integer id)
```

Busca un roleUser por id

Returns:roleUser

14.9.1.63 Interface ShopService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[ShopServiceImpl](#)

```
public interface ShopService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad Shop

Detalle de los métodos

listShops

```
List<Shop> listShops ()
```

Lista las tiendas que hay en la base de datos

Returns:List shops

saveShop

```
Shop saveShop (Shop shop)
```

Guarda una tienda en la base de datos

Returns:shop

updateShop

```
Shop updateShop (Shop shop)
```

Actualiza una tienda de la base de datos

Returns:shop

deleteShop

```
Shop deleteShop (Shop shop)
```

Elimina una tienda de la base de datos

Returns:shop

findShopById

```
Shop findShopById(Integer id)
```

Busca una tienda por id

Returns:shop

findShopByType

List<[Shop](#)> **findShopByType**(String type)

Busca una tienda por tipo(los tipos podrían ser Prestashop o Magento)

Returns:List shops

findShopByName

List<[Shop](#)> **findShopByName**(String name)

Lista las tiendas por nombre

Returns:List shops

listShopByUser

List<[Shop](#)> **listShopByUser**(Integer userId)

Lista las tiendas por usuario

Returns:List shops

14.9.1.64 Interface *UserCategoryService*

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[UserCategoryServiceImpl](#)

```
public interface UserCategoryService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad UseCategory

Detalle de los métodos

listUserCategories

List<[UserCategory](#)> **listUserCategories**()

Lista el conjunto de UserCategory que hay en la base de datos

Returns:List userCategories

saveUserCategory

[UserCategory](#) **saveUserCategory**([UserCategory](#) userCategory)

Guarda una UserCategory en la base de datos

Returns:UserCategory

updateUserCategory

[UserCategory](#) updateUserCategory ([UserCategory](#) userCategory)

Actualiza una UserCategory de la base de datos

Returns:userCategory

deleteUserCategory

[UserCategory](#) deleteUserCategory ([UserCategory](#) userCategory)

Elimina una userCategory de la base de datos

Returns:UserCategory

findUserCategoryById

[UserCategory](#) findUserCategoryById (Integer id)

Busca una userCategory por id

Returns:UserCategory

findUserCategoryByCategory

List<[UserCategory](#)> findUserCategoryByCategory ([Category](#) category)

Lista el conjunto de UserCategories filtradas por Category

Returns:List userCategories

14.9.1.65 Interface UserProductService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[UserProductServiceImpl](#)

```
public interface UserProductService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad UseProduct

Detalle de los métodos

listUserProducts

List<[UserProduct](#)> listUserProducts ()

Lista el conjunto de UserProviders que hay en la base de datos

Returns:List userProducts

saveUserProduct

[UserProduct](#) saveUserProduct([UserProduct](#) userProduct)

Guarda un userProduct en la base de datos

Returns:UserProduct

updateUserProduct

[UserProduct](#) updateUserProduct([UserProduct](#) userProduct)

Actualiza un userProduct de la base de datos

Returns:userProduct

deleteUserProduct

[UserProduct](#) deleteUserProduct([UserProduct](#) userProduct)

Elimina un userProduct de la base de datos

Returns:userProduct

findUserProductById

[UserProduct](#) findUserProductById(Integer id)

Busca un userProduct filtrado por id

Returns:userProduct

ffindUserProductByProduct

List<[UserProduct](#)> ffindUserProductByProduct([Product](#) product)

Lista el conjunto de UserProducts de la base de datos filtrado por producto

Returns:List userProducts

14.9.1.66 Interface UserProviderService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[UserProviderServiceImpl](#)

```
public interface UserProviderService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad UseProvider

Detalle de los métodos

listUserProviders

```
List<UserProvider> listUserProviders ()
```

Lista el conjunto de UserProviders almacenados en la base de datos

Returns:List userProviders

saveUserProvider

```
UserProvider saveUserProvider(UserProvider userProvider)
```

Guarda un userProvider en la base de datos

Returns:UserProvider

updateUserProvider

```
UserProvider updateUserProvider(UserProvider userProvider)
```

Actualiza un userProvider de la base de datos

Returns:userProvider

deleteUserProvider

```
UserProvider deleteUserProvider(UserProvider userProvider)
```

Elimina un userProvider de la base de datos

Returns:UserProvider

findUserProviderById

```
UserProvider findUserProviderById(Integer id)
```

Busca un userProvider filtrado por id

Returns:UserProvider

findUserProviderByProvider

```
List<UserProvider> findUserProviderByProvider(Provider p)
```

Busca los userProviders que hay en la base de datos filtrados por proveedor

Returns:List userProviders

14.9.1.67 Interface UserService

com.tunos.commerce.service

All Known Implementing Classes:

[UserServiceImpl](#)

```
public interface UserService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad User

Detalle de los métodos

listUsers

```
List<User> listUsers()
```

Lista los usuarios

Returns:List users

saveUser

```
User saveUser (User user)
```

Guarda un usuario

Returns:user

updateUser

```
User updateUser (User user)
```

Update user.

Parameters:user - the user

Returns:the user

deleteUser

```
User deleteUser (User user)
```

Delete user.

Parameters:user - the user

Returns:the user

findUserById

```
User findUserById(Integer id)
```

Busca un usuario por id

Returns:the user

findUserByUsername

```
User findUserByUsername(String username)
```

Busca un usuario por username

Returns:user

createUser

[User](#) createUser([User](#) user)

Crea un usuario

Returns:usuario

listUserByRoleId

List<Integer> listUserByRoleId(Integer role_id)

Lista los usuarios por id del rol, retorna la lista de enteros de esos usuarios

Returns:List ids

14.9.1.68 Interface UserShopService

[com.tunos.commerce.service](#)

All Known Implementing Classes:

[UserShopServiceImpl](#)

```
public interface UserShopService
```

Interfaz de los servicios que ofrecerá el manejador de la entidad UserShop

Detalle de los métodos

listUserShops

List<[UserShop](#)> listUserShops()

Lista los userShop existentes en la base de datos

Returns:List userShops

saveUserShop

[UserShop](#) saveUserShop([UserShop](#) userShop)

guarda un userShop

Returns:userShop

updateUserShop

[UserShop](#) updateUserShop([UserShop](#) userShop)

Actualiza un userShop
Returns:userShop

deleteUserShop

[UserShop](#) deleteUserShop([UserShop](#) userShop)

Elimina un userShop
Returns:userShop

findUserShopById

[UserShop](#) findUserShopById(Integer id)

Busca un userShop por id
Returns:userShop

findUserShopByShop

List<[UserShop](#)> findUserShopByShop([Shop](#) shop)

Lista los userShop filtrados por Shop
Returns:List userShops

14.9.1.69 Class CategoryServiceImpl

[com.tunos.commerce.service.impl](#)

java.lang.Object

└ com.tunos.commerce.service.impl.CategoryServiceImpl

All Implemented Interfaces:

[CategoryService](#)

```
@org.springframework.stereotype.Service(value="categoryService")
public class CategoryServiceImpl extends Object implements CategoryService
```

Clase que implementa los servicios declarados en la interfaz CategoryDAO.

Atributos

categoryDAO

private [CategoryDAO](#) categoryDAO

The user dao.

Detalle de los métodos

saveCategory

```
@org.springframework.transaction.annotation.Transactional  
public Category saveCategory(Category category)
```

guarda una categoría

Specified by: [saveCategory](#) in interface [CategoryService](#)

updateCategory

```
@org.springframework.transaction.annotation.Transactional  
public Category updateCategory(Category category)
```

Actualiza una categoría

Specified by: [updateCategory](#) in interface [CategoryService](#)

deleteCategory

```
@org.springframework.transaction.annotation.Transactional  
public Category deleteCategory(Category category)
```

Elimina una cateoría

Specified by: [deleteCategory](#) in interface [CategoryService](#)

Returns:category

findCategoryById

```
public Category findCategoryById(Integer id)
```

Busca una categoría por id

Specified by: [findCategoryById](#) in interface [CategoryService](#)

Returns:category

listTotalEntriesCategoriesByUser

```
public List<Integer> listTotalEntriesCategoriesByUser(SearchCriteria searchCriteria, Integer userId, Integer parentId)
```

Lista todas las categorías de un usuario y retorna sus ids

Specified by: [listTotalEntriesCategoriesByUser](#) in interface [CategoryService](#)

Returns:List ids

listCategoriesByUser

```
public List<Category> listCategoriesByUser(SearchCriteria searchCriteria, Integer userId, Integer parentId)
```

Lista las categorías de un usuario teniendo en cuenta las restricciones del searchCriteria....

Specified by: [listCategoriesByUser](#) in interface [CategoryService](#)

listChilds

```
public List<Category> listChilds(Integer parentId)
```

Lista los hijos de una categoría

Specified by: [listChilds](#) in interface [CategoryService](#)

Returns: List childs

findCategoryByName

```
public Category findCategoryByName(String name,Integer user)
```

Busca una categoría por nombre

Specified by: [findCategoryByName](#) in interface [CategoryService](#)

Returns: category

14.9.1.70 Class ProductCategoryServiceImpl

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.ProductCategoryServiceImpl
```

All Implemented Interfaces:

[ProductCategoryService](#)

```
@org.springframework.stereotype.Service(value="productCategoryService")
public class ProductCategoryServiceImpl extends Object
implements ProductCategoryService
```

Clase que implementa los servicios declarados en la interfaz ProductCategoryDAO.

Atributos

ProductCategoryDAO

```
private ProductCategoryDAO ProductCategoryDAO
```

Detalle de los métodos

listProductCategories

```
public List<ProductCategory> listProductCategories()
```

Lista el conjunto de entidades productCategory existentes en el sistema

Specified by: [listProductCategories](#) in interface [ProductCategoryService](#)

Returns:List productCategory

saveProductCategory

```
@org.springframework.transaction.annotation.Transactional  
public ProductCategory saveProductCategory(ProductCategory productCategory)
```

Guarda una entidad de tipo ProductCategory en la base de datos

Specified by: [saveProductCategory](#) in interface [ProductCategoryService](#)

updateProductCategory

```
@org.springframework.transaction.annotation.Transactional  
public ProductCategory updateProductCategory(ProductCategory productCategory)
```

Actualiza una entidad de tipo ProductCategory de la base de datos

Specified by: [updateProductCategory](#) in interface [ProductCategoryService](#)

Returns:ProductCategory

deleteProductCategory

```
@org.springframework.transaction.annotation.Transactional  
public ProductCategory deleteProductCategory(ProductCategory productCategory)
```

Elimina de la base de datos un ProductCategory

Specified by: [deleteProductCategory](#) in interface [ProductCategoryService](#)

Returns:ProductCategory

findProductCategoryById

```
@org.springframework.transaction.annotation.Transactional  
public ProductCategory findProductCategoryById(Integer id)
```

Busca un ProductCategory por id

Specified by: [findProductCategoryById](#) in interface [ProductCategoryService](#)

Returns:ProductCategory

findCategoriesFromProduct

```
public List<Integer> findCategoriesFromProduct(Integer id)
```

Busca un productCategory por id

Specified by: [findCategoriesFromProduct](#) in interface [ProductCategoryService](#)

Returns:ProductCategory

findByCategory

```
public List<ProductCategory> findByCategory(Category c)
```

Lista los productos por categoría y retorna sus id's

Specified by: `findByCategory` in interface [ProductCategoryService](#)

Returns: List ids

findByProduct

```
public List<ProductCategory> findByProduct(Product p)
```

Lista los ProductCategory por categoría

Specified by: `findByProduct` in interface [ProductCategoryService](#)

Returns: List productCategories

14.9.1.71 Class ProductServiceImpl

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.ProductServiceImpl
```

All Implemented Interfaces:

[ProductService](#)

```
@org.springframework.stereotype.Service(value="productService")
public class ProductServiceImpl extends Object implements ProductService
```

Clase que implementa los servicios declarados en la interfaz ProductDAO.

Atributos

productDAO

```
private ProductDAO productDAO
```

Detalle de los métodos

listProducts

```
public List<Product> listProducts()
```

Lista todos los productos guardados en la base de datos

Specified by: `listProducts` in interface [ProductService](#)

Returns: List productos

saveProduct

```
@org.springframework.transaction.annotation.Transactional
public Product saveProduct(Product product)
```

Guarda un producto en la base de datos

Specified by: `saveProduct` in interface [ProductService](#)

Returns: Product

updateProduct

```
@org.springframework.transaction.annotation.Transactional  
public Product updateProduct(Product product)
```

Actualiza el producto

Specified by: `updateProduct` in interface [ProductService](#)

Returns: Product

deleteProduct

```
@org.springframework.transaction.annotation.Transactional  
public Product deleteProduct(Product product)
```

Elimina el product

Specified by: `deleteProduct` in interface [ProductService](#)

Returns: Product

findProductById

```
@org.springframework.transaction.annotation.Transactional  
public Product findProductById(Integer id)
```

Busca un producto por id

Specified by: `findProductById` in interface [ProductService](#)

Returns: product

findProductByName

```
public Product findProductByName(String name)
```

Busca un producto por nombre

Specified by: `findProductByName` in interface [ProductService](#)

Returns: Product

listProductsByUser

```
public List<Product> listProductsByUser(SearchCriteria criteria, Integer idUser  
)
```

Lista los productos de un usuario y teniendo en cuenta las restricciones del
searchCriteria

Specified by: `listProductsByUser` in interface [ProductService](#)

Returns: List productos

listTotalEntriesProductsByUser

```
public List<Integer> listTotalEntriesProductsByUser(SearchCriteria criteria,
                                                    Integer idUser)
```

Lista el conjunto de productos de un usuario

Specified by: [listTotalEntriesProductsByUser](#) in interface [ProductService](#)

Returns: List productos

listProductsByProvider

```
public List<Product> listProductsByProvider(Provider provider)
```

Lista los productos por proveedor

Specified by: [listProductsByProvider](#) in interface [ProductService](#)

Returns: List productos

listProductsByUser

```
public List<Product> listProductsByUser(Integer idUser)
```

Lista los productos de un usuario

Specified by: [listProductsByUser](#) in interface [ProductService](#)

Returns: List productos

14.9.1.72 Class *ProviderServiceImpl*

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.ProviderServiceImpl
```

All Implemented Interfaces:

[ProviderService](#)

```
@org.springframework.stereotype.Service(value="providerService")
public class ProviderServiceImpl extends Object
implements ProviderService
```

Clase que implementa los servicios declarados en la interfaz ProviderDAO

Atributos

providerDAO

```
private ProviderDAO providerDAO
```

Detalle de los métodos

listProviders

```
public List<Provider> listProviders()
```

Lista todos los proveedores existentes en la base de datos

Specified by: `listProviders` in interface [ProviderService](#)

Returns: `listProviders`

saveProvider

```
@org.springframework.transaction.annotation.Transactional  
public Provider saveProvider(Provider provider)
```

Guarda un proveedor en la base de datos

Specified by: `saveProvider` in interface [ProviderService](#)

Returns: `provider`

updateProvider

```
@org.springframework.transaction.annotation.Transactional  
public Provider updateProvider(Provider provider)
```

Actualiza un proveedor en la base de datos

Specified by: `updateProvider` in interface [ProviderService](#)

deleteProvider

```
@org.springframework.transaction.annotation.Transactional  
public Provider deleteProvider(Provider provider)
```

Elimina un proveedor de la base de datos

Specified by: `deleteProvider` in interface [ProviderService](#)

Returns: `provider`

findProviderById

```
@org.springframework.transaction.annotation.Transactional  
public Provider findProviderById(Integer id)
```

Busca un proveedor por id

Specified by: `findProviderById` in interface [ProviderService](#)

Returns: `provider`

findProviderByNIF

```
@org.springframework.transaction.annotation.Transactional  
public Provider findProviderByNIF(String nif)
```

Busca un proveedor por NIF

Specified by:[findProviderByNIF](#) in interface [ProviderService](#)

Returns:provider

findProviderByCompany

```
@org.springframework.transaction.annotation.Transactional  
public Provider findProviderByCompany(String company, Integer user)
```

Bbusca un proveedor por compañía

Specified by:[findProviderByCompany](#) in interface [ProviderService](#)

Returns:Provider

listProviderByIdUser

```
public List<Integer> listProviderByIdUser(Integer idUser)
```

Lista los proveedores por id de usuario

Specified by:[listProviderByIdUser](#) in interface [ProviderService](#)

Returns:List ids_proveedores: identificador de los proveedores

listProviders

```
public List<Provider> listProviders(SearchCriteria criteria)
```

Lista los proveedores teniendo en cuenta el searchCriteria

Specified by:[listProviders](#) in interface [ProviderService](#)

listProvidersByUser

```
public List<Provider> listProvidersByUser(SearchCriteria criteria, Integer idUser)
```

Lista los proveedores por usuario y teniendo en cuenta el searchCriteria

Specified by:[listProvidersByUser](#) in interface [ProviderService](#)

Returns:List proveedores

listTotalEntriesProvidersByUser

```
public List<Integer> listTotalEntriesProvidersByUser(SearchCriteria criteria,  
Integer idUser)
```

Lista todos los proveedores teniendo en cuenta las restricciones del searchCriteria.

Retorna una lista de enteros con los id's de los proveedores

Specified by:[listTotalEntriesProvidersByUser](#) in interface [ProviderService](#)

Returns:List ids

14.9.1.73 Class RoleServiceImpl

[com.tunos.commerce.service.impl](#)

java.lang.Object

└─ `com.tunos.commerce.service.impl.RoleServiceImpl`

All Implemented Interfaces:

[RoleService](#)

```
@org.springframework.stereotype.Service(value="roleService")
public class RoleServiceImpl extends Object implements RoleService
```

Clase que implementa los servicios declarados en la interfaz RoleDAO.

Atributos

roleDAO

private [RoleDAO](#) roleDAO

Detalle de los métodos

listRoles

```
public List<Role> listRoles()
```

Lista el conjunto de roles contenidos en la base de datos

Specified by: `listRoles` in interface [RoleService](#)

Returns: List roles

saveRole

```
public Role saveRole(Role role)
```

Guarda un rol en la base de datos

Specified by: `saveRole` in interface [RoleService](#)

Returns: rol

updateRole

```
public Role updateRole(Role role)
```

Actualiza un rol de la base de datos

Specified by: `updateRole` in interface [RoleService](#)

Returns: role

deleteRole

```
public Role deleteRole(Role role)
```

Elimina el rol de la base de datos

Specified by: `deleteRole` in interface [RoleService](#)

Returns: role

findRoleById

```
public Role findRoleById(Integer id)
```

Busca un rol por id

Specified by: `findRoleById` in interface [RoleService](#)

Returns: role

14.9.1.74 Class RoleUserServiceImpl

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.RoleUserServiceImpl
```

All Implemented Interfaces:

[RoleUserService](#)

```
@org.springframework.stereotype.Service(value="roleUserService")
```

```
public class RoleUserServiceImpl extends Object implements RoleUserService
```

Clase que implementa los servicios declarados en la interfaz RoleUserDAO.

Atributos

RoleUserDAO

```
private RoleUserDAO RoleUserDAO
```

Detalle de los métodos

listRoleUsers

```
public List<RoleUser> listRoleUsers()
```

Lista los RoleUsers que hay en la base de datos

Specified by: `listRoleUsers` in interface [RoleUserService](#)

Returns: List roleUsers

saveRoleUser

```
@org.springframework.transaction.annotation.Transactional  
public RoleUser saveRoleUser(RoleUser roleUser)
```

Guarda un RoleUser en la base de datos

Specified by: [saveRoleUser](#) in interface [RoleUserService](#)

Returns: RoleUser

updateRoleUser

```
@org.springframework.transaction.annotation.Transactional  
public RoleUser updateRoleUser(RoleUser roleUser)
```

Actualiza un RoleUser de la base de datos

Specified by: [updateRoleUser](#) in interface [RoleUserService](#)

Returns: roleUser

deleteRoleUser

```
@org.springframework.transaction.annotation.Transactional  
public RoleUser deleteRoleUser(RoleUser roleUser)
```

Actualiza un RoleUser de la base de datos

Specified by: [deleteRoleUser](#) in interface [RoleUserService](#)

Returns: roleUser

findRoleUserById

```
@org.springframework.transaction.annotation.Transactional  
public RoleUser findRoleUserById(Integer id)
```

Elimina un roleUser de la base de datos

Specified by: [findRoleUserById](#) in interface [RoleUserService](#)

Returns: roleUser

14.9.1.75 Class ShopServiceImpl

[com.tunos.commerce.service.impl](#)

java.lang.Object

└─ `com.tunos.commerce.service.impl.ShopServiceImpl`

All Implemented Interfaces:

[ShopService](#)

```
@org.springframework.stereotype.Service(value="shopService")  
public class ShopServiceImpl extends Object implements ShopService
```

Clase que implementa los servicios declarados en la interfaz ShopDAO.

Atributos

shopDAO

```
private ShopDAO shopDAO
```

The user dao.

Detalle de los métodos

listShops

```
public List<Shop> listShops ()
```

Lista las tiendas que hay en la base de datos

Specified by: `listShops` in interface [ShopService](#)

Returns: List shops

saveShop

```
@org.springframework.transaction.annotation.Transactional  
public Shop saveShop(Shop shop)
```

Guarda una tienda en la base de datos

Specified by: `saveShop` in interface [ShopService](#)

Returns: shop

updateShop

```
@org.springframework.transaction.annotation.Transactional  
public Shop updateShop(Shop shop)
```

Actualiza una tienda de la base de datos

Specified by: `updateShop` in interface [ShopService](#)

Returns: shop

deleteShop

```
@org.springframework.transaction.annotation.Transactional  
public Shop deleteShop(Shop shop)
```

Elimina una tienda de la base de datos

Specified by: `deleteShop` in interface [ShopService](#)

Returns: shop

findShopById

```
@org.springframework.transaction.annotation.Transactional  
public Shop findShopById(Integer id)
```

Busca una tienda por id

Specified by: [findShopById](#) in interface [ShopService](#)

Returns:shop

findShopByType

```
public List<Shop> findShopByType (String type)
```

Busca una tienda por tipo(los tipos podrían ser Prestashop o Magento)

Specified by: [findShopByType](#) in interface [ShopService](#)

Returns:List shops

findShopByName

```
public List<Shop> findShopByName (String name)
```

Lista las tiendas por nombre

Specified by: [findShopByName](#) in interface [ShopService](#)

Returns:List shops

listShopByUser

```
public List<Shop> listShopByUser (Integer userId)
```

Lista las tiendas por usuario

Specified by: [listShopByUser](#) in interface [ShopService](#)

Returns:List shops

14.9.1.76 Class *UserCategoryServiceImpl*

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.UserCategoryServiceImpl
```

All Implemented Interfaces:

[UserCategoryService](#)

```
@org.springframework.stereotype.Service(value="userCategoryService")
```

```
public class UserCategoryServiceImpl
```

```
extends Object implements UserCategoryService
```

Clase que implementa los servicios declarados en la interfaz UserCategoryDAO.

Atributos

UserCategoryDAO

```
private UserCategoryDAO userCategoryDAO
```

The user dao.

Detalle de los métodos

listUserCategories

```
public List<UserCategory> listUserCategories ()
```

Lista el conjunto de UserCategory que hay en la base de datos

Specified by: [listUserCategories](#) in interface [UserCategoryService](#)

Returns: List userCategories

saveUserCategory

```
@org.springframework.transaction.annotation.Transactional  
public UserCategory saveUserCategory (UserCategory userCategory)
```

Guarda una UserCategory en la base de datos

Specified by: [saveUserCategory](#) in interface [UserCategoryService](#)

Returns: UserCategory

updateUserCategory

```
@org.springframework.transaction.annotation.Transactional  
public UserCategory updateUserCategory (UserCategory userCategory)
```

Actualiza una UserCategory de la base de datos

Specified by: [updateUserCategory](#) in interface [UserCategoryService](#)

Returns: userCategory

deleteUserCategory

```
@org.springframework.transaction.annotation.Transactional  
public UserCategory deleteUserCategory (UserCategory userCategory)
```

Elimina una userCategory de la base de datos

Specified by: [deleteUserCategory](#) in interface [UserCategoryService](#)

Returns: UserCategory

findUserCategoryById

```
@org.springframework.transaction.annotation.Transactional  
public UserCategory findUserCategoryById (Integer id)
```

Busca una userCategory por id

Specified by: [findUserCategoryById](#) in interface [UserCategoryService](#)

findUserCategoryByCategory

```
public List<UserCategory> findUserCategoryByCategory (Category category)
```

Lista el conjunto de UserCategories filtradas por Category

Specified by: [findUserCategoryByCategory](#) in interface [UserCategoryService](#)

Returns: List userCategories

14.9.1.77 Class UserProductServiceImpl

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.UserProductServiceImpl
```

All Implemented Interfaces:

[UserProductService](#)

```
@org.springframework.stereotype.Service(value="userProductService")  
public class UserProductServiceImpl extends Object  
implements UserProductService
```

Clase que implementa los servicios declarados en la interfaz UserProductDAO.

Atributos

UserProductDAO

```
private UserProductDAO UserProductDAO
```

Detalle de los métodos

listUserProducts

```
public List<UserProduct> listUserProducts ()
```

Lista el conjunto de UserCategory que hay en la base de datos

Specified by: [listUserProducts](#) in interface [UserProductService](#)

Returns: List userCategories

saveUserProduct

```
@org.springframework.transaction.annotation.Transactional  
public UserProduct saveUserProduct (UserProduct userProduct)
```

Guarda una UserCategory en la base de datos

Specified by:

[saveUserProduct](#) in interface [UserProductService](#)

Returns: UserCategory

updateUserProduct

```
@org.springframework.transaction.annotation.Transactional
public UserProduct updateUserProduct(UserProduct userProduct)
```

Actualiza una UserCategory de la base de datos

Specified by: `updateUserProduct` in interface [UserProductService](#)

Returns: userCategory

deleteUserProduct

```
@org.springframework.transaction.annotation.Transactional
public UserProduct deleteUserProduct(UserProduct userProduct)
```

Elimina una userCategory de la base de datos

Specified by: `deleteUserProduct` in interface [UserProductService](#)

Returns: UserCategory

findUserProductById

```
@org.springframework.transaction.annotation.Transactional
public UserProduct findUserProductById(Integer id)
```

Busca una userCategory por id

Specified by: `findUserProductById` in interface [UserProductService](#)

Returns: userProduct

ffindUserProductByProduct

```
public List<UserProduct> ffindUserProductByProduct(Product product)
```

Lista el conjunto de UserCategories filtradas por Category

Specified by: `ffindUserProductByProduct` in interface [UserProductService](#)

Returns: List userCategories

14.9.1.78 Class *UserProviderServiceImpl*

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.UserProviderServiceImpl
```

All Implemented Interfaces:

[UserProviderService](#)

```
@org.springframework.stereotype.Service(value="userProviderService")
public class UserProviderServiceImpl extends Object
implements UserProviderService
```

Clase que implementa los servicios declarados en la interfaz UserProviderDAO

Atributos

userProviderDAO

```
private UserProviderDAO userProviderDAO
```

Detalle de los métodos

listUserProviders

```
public List<UserProvider> listUserProviders ()
```

Lista el conjunto de UserProviders almacenados en la base de datos

Specified by: [listUserProviders](#) in interface [UserProviderService](#)

Returns: List userProviders

saveUserProvider

```
@org.springframework.transaction.annotation.Transactional  
public UserProvider saveUserProvider (UserProvider userProvider)
```

Guarda un userProvider en la base de datos

Specified by: [saveUserProvider](#) in interface [UserProviderService](#)

Returns: UserProvider

updateUserProvider

```
@org.springframework.transaction.annotation.Transactional  
public UserProvider updateUserProvider (UserProvider userProvider)
```

Actualiza un userProvider de la base de datos

Specified by: [updateUserProvider](#) in interface [UserProviderService](#)

Returns: userProvider

deleteUserProvider

```
@org.springframework.transaction.annotation.Transactional  
public UserProvider deleteUserProvider (UserProvider userProvider)
```

Elimina un userProvider de la base de datos

Specified by: [deleteUserProvider](#) in interface [UserProviderService](#)

Returns: UserProvider

findUserProviderById

```
public UserProvider findUserProviderById (Integer id)
```

Busca un userProvider filtrado por id

Specified by: [findUserProviderById](#) in interface [UserProviderService](#)

Returns: UserProvider

findUserProviderByProvider

```
public List<UserProvider> findUserProviderByProvider (Provider p)
```

Busca los userProviders que hay en la base de datos filtrados por proveedor

Specified by: [findUserProviderByProvider](#) in interface [UserProviderService](#)

Returns: List userProviders

14.9.1.79 Class UserServiceImpl

[com.tunos.commerce.service.impl](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.service.impl.UserServiceImpl
```

All Implemented Interfaces:

[UserService](#)

```
@org.springframework.stereotype.Service(value="userService")
public class UserServiceImpl extends Object implements UserService
```

Clase que implementa los servicios declarados en la interfaz UserDao

Atributos

userDAO

```
private UserDAO userDAO
```

The user dao.

Detalle de los métodos

listUsers

```
@org.springframework.transaction.annotation.Transactional
public List<User> listUsers ()
```

Lista los usuarios

Specified by: [listUsers](#) in interface [UserService](#)

Returns: List users

saveUser

```
@org.springframework.transaction.annotation.Transactional
public User saveUser (User user)
```

Guarda un usuario

Specified by: [saveUser](#) in interface [UserService](#)

Returns: user

updateUser

```
@org.springframework.transaction.annotation.Transactional  
public User updateUser(User user)
```

Update user.

Specified by: [updateUser](#) in interface [UserService](#)

Parameters: user - the user

Returns: the user

deleteUser

```
@org.springframework.transaction.annotation.Transactional  
public User deleteUser(User user)
```

Delete user.

Specified by: [deleteUser](#) in interface [UserService](#)

Parameters: user - the user

Returns: the user

findUserById

```
@org.springframework.transaction.annotation.Transactional  
public User findUserById(Integer id)
```

Busca un usuario por id

Specified by: [findUserById](#) in interface [UserService](#)

findUserByUsername

```
@org.springframework.transaction.annotation.Transactional  
public User findUserByUsername(String username)
```

Busca un usuario por username

Specified by: [findUserByUsername](#) in interface [UserService](#)

Returns: user

createUser

```
public User createUser(User user)
```

Crea un usuario

Specified by: [createUser](#) in interface [UserService](#)

Returns: usuario

listUserByRoleId

```
public List<Integer> listUserByRoleId(Integer role_id)
```

Lista los usuarios por id del rol, retorna la lista de enteros de esos usuarios

Specified by: `listUserByRoleId` in interface [UserService](#)

Returns: List ids

14.9.1.80 Class UserShopServiceImpl

[com.tunos.commerce.service.impl](#)

`java.lang.Object`

└ `com.tunos.commerce.service.impl.UserShopServiceImpl`

All Implemented Interfaces:

[UserShopService](#)

```
@org.springframework.stereotype.Service(value="userShopService")
public class UserShopServiceImpl extends Object implements UserShopService
```

Clase que implementa los servicios declarados en la interfaz UserShopDAO.

Atributos

UserShopDAO

```
private UserShopDAO UserShopDAO
```

Detalle de los métodos

listUserShops

```
public List<UserShop> listUserShops()
```

Lista los userShop existentes en la base de datos

Specified by: `listUserShops` in interface [UserShopService](#)

Returns: List userShops

saveUserShop

```
@org.springframework.transaction.annotation.Transactional
public UserShop saveUserShop(UserShop userShop)
```

Guarda un userShop

Specified by: `saveUserShop` in interface [UserShopService](#)

Returns: userShop

updateUserShop

```
@org.springframework.transaction.annotation.Transactional
public UserShop updateUserShop(UserShop userShop)
```

Actualiza un userShop

Specified by: [updateUserShop](#) in interface [UserShopService](#)

Returns: userShop

deleteUserShop

```
@org.springframework.transaction.annotation.Transactional
public UserShop deleteUserShop(UserShop userShop)
```

Elimina un userShop

Specified by: [deleteUserShop](#) in interface [UserShopService](#)

Returns: userShop

findUserShopById

```
@org.springframework.transaction.annotation.Transactional
public UserShop findUserShopById(Integer id)
```

Busca un userShop por id

Specified by: [findUserShopById](#) in interface [UserShopService](#)

Returns: userShop

findUserShopByShop

```
public List<UserShop> findUserShopByShop(Shop shop)
```

Lista los userShop filtrados por Shop

Specified by: [findUserShopByShop](#) in interface [UserShopService](#)

Returns: List userShops

14.9.1.81 *Class CleanText*

[com.tunos.commerce.utils](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.utils.CleanText
```

```
public class CleanText extends Object
```

Clase empleada para eliminar tags HTML de un texto

Detalle de los métodos

eliminarEtiquetasHTML

```
public static String eliminarEtiquetasHTML(String cadena)
```


14.9.1.82 Class CreateXORpass

[com.tunos.commerce.utils](#)

java.lang.Object

└─ com.tunos.commerce.utils.CreateXORpass

```
public class CreateXORpass
```

```
extends Object
```

Clase empleada para la codificación de contraseñas por XOR

Atributos

key

```
private static String key
```

Detalle de los métodos

xor_encrypt

```
public static String xor_encrypt(String message)
```

Encripta la contraseña con un XOR

Returns: contraseña codificada

xor_decrypt

```
public static String xor_decrypt(String message)
```

Desencripta la contraseña con un XOR

Returns: contraseña en plano

14.9.1.83 Class SearchCriteria

[com.tunos.commerce.utils](#)

java.lang.Object

└─ com.tunos.commerce.utils.SearchCriteria

All Implemented Interfaces:

Serializable

```
public class SearchCriteria extends Object implements Serializable
```

La clase search criteria empleada para la paginación de las tablas

Atributos

serialVersionUID

```
private static final long serialVersionUID
```

searchString

```
private String searchString
```

pageSize

```
private int pageSize
```

totalEntries

```
private int totalEntries
```

page

```
private int page
```

sortBy

```
private String sortBy
```

providerForm

```
private ProviderForm providerForm
```

userForm

```
private UserForm userForm
```

productForm

```
private ProductForm productForm
```

categoryForm

```
private CategoryForm categoryForm
```

Detalle de los métodos

getPageSize

```
public int getPageSize()
```

Obtiene el tamaño máximo de página de la lista de resultados.

Returns:pageSize

setPageSize

```
public void setPageSize(int pageSize)
```

Insertamos el tamaño máximo de página de la lista de resultados.

getPage

```
public int getPage()
```

Obtenemos la pagina actual

Returns:página actual

setPage

```
public void setPage(int page)
```

Actualizamos la página actual

nextPage

```
public void nextPage()
```

Página siguiente

previousPage

```
public void previousPage()
```

página anterior

lastPage

```
public void lastPage()
```

última página

firstPage

```
public void firstPage()  
  
    primera página.
```

resetPage

```
public void resetPage()  
  
    Reseteamos la página
```

14.9.1.84 Class SecurityUtilities

com.tunos.commerce.utils

```
java.lang.Object  
  
└─ com.tunos.commerce.utils.SecurityUtilities
```

```
public class SecurityUtilities extends Object
```

Clase empleada para hashear las contraseñas de los usuario

Detalle de los métodos

getSHA

```
public static String getSHA(String input) throws NoSuchAlgorithmException, UnsupportedEncodingException
```

Generamos el SHA-1 a partir de la contraseña de un usuario

Throws:

NoSuchAlgorithmException
UnsupportedEncodingException

getHash

```
private static byte[] getHash(String input, String algorithm)  
throws NoSuchAlgorithmException, UnsupportedEncodingException
```

Se obtiene la hash

Throws:NoSuchAlgorithmException
UnsupportedEncodingException

toHexadecimal

```
private static String toHexadecimal(byte[] datos)
```

14.9.1.85 Class ProviderFormValidator

com.tunos.commerce.validators

```
java.lang.Object
```

```
└─ com.tunos.commerce.validators.ProviderFormValidator
```

All Implemented Interfaces:

```
org.springframework.validation.Validator
```

```
@org.springframework.stereotype.Component
public class ProviderFormValidator extends Object
implements org.springframework.validation.Validator
```

Clase encargada de validar los formularios de creación y edición de proveedores

Atributos**providerService**

```
private ProviderService providerService
```

The user service.

Detalle de los métodos**validateRegister**

```
public void validateRegister(ProviderForm providerForm,
                             org.springframework.validation.Errors errors)
```

Valida el registro

Parameters:errors - the errors

validateCreate

```
public void validateCreate(ProviderForm providerForm,
                           org.springframework.binding.validation.ValidationContext context)
```

Valida la creación de Proveedores

Parameters:context - the context

supports

```
public boolean supports(Class clazz)
```

Specified by: supports in interface org.springframework.validation.Validator

validate

```
public void validate(Object o,  
    org.springframework.validation.Errors errors)
```

Specified by: validate in interface org.springframework.validation.Validator

isCifValido

```
private boolean isCifValido(String cif)
```

Comprueba que el cif de un proveedor es un cif válido

14.9.1.86 Class ShopFormValidator

com.tunos.commerce.validators

```
java.lang.Object
```

```
└─ com.tunos.commerce.validators.ShopFormValidator
```

All Implemented Interfaces:

org.springframework.validation.Validator

```
@org.springframework.stereotype.Component
```

```
public class ShopFormValidator extends Object implements  
org.springframework.validation.Validator
```

Clase dedicada a la validación de los formularios de creación y de modificación de tienda

Atributos

shopService

```
private ShopService shopService
```

Detalle de los métodos

validateRegister

```
public void validateRegister(ShopForm shopForm,  
    org.springframework.binding.validation.Validation  
Context context)
```

Validate register.

validateCreate

```
public void validateCreate(ShopForm shopForm,  
                           org.springframework.binding.validation.ValidationCo  
ntext context)
```

Validación al crear una tienda

Parameters: context - the context

validateEdit

```
public void validateEdit(ShopForm shopForm,  
                           org.springframework.binding.validation.ValidationCont  
ext context)
```

Validar al editar una tienda

Parameters: context - the context

supports

```
public boolean supports(Class clazz)
```

Specified by: supports in interface org.springframework.validation.Validator

validate

```
public void validate(Object o,  
                     org.springframework.validation.Errors errors)
```

Specified by: validate in interface org.springframework.validation.Validator

14.9.1.87 Class *UserFormValidator*

[com.tunos.commerce.validators](#)

```
java.lang.Object
```

```
└─ com.tunos.commerce.validators.UserFormValidator
```

All Implemented Interfaces:

org.springframework.validation.Validator

```
@org.springframework.stereotype.Component  
public class UserFormValidator extends Object  
implements org.springframework.validation.Validator
```

Clase dedicada a la validación de formularios de Usuario

Atributos

userService

```
private UserService userService
```

Detalle de los métodos

validateRegister

```
public void validateRegister(UserForm userForm,  
                             org.springframework.binding.validation.Validation  
Context context)
```

Validate register.

Parameters:userForm - the user form

validateCreate

```
public void validateCreate(UserForm userForm,  
                             org.springframework.binding.validation.ValidationCo  
ntext context)
```

Valida la creación de usuarios

Parameters:userForm - the user form

context - the context

validateEdit

```
public void validateEdit(UserForm userForm,  
                             org.springframework.binding.validation.ValidationCont  
ext context)
```

Valida la modificación de usuarios

Parameters:userForm - the user form,context - the context

supports

```
public boolean supports(Class clazz)  
    Specified by:supports in interface org.springframework.validation.Validator
```

validate

```
public void validate(Object o,org.springframework.validation.Errors errors)
```


Specified by: `validate` in interface `org.springframework.validation.Validator`