UNIVERSIDAD DE OVIEDO

ESCUELA DE INGENIERÍA INFORMÁTICA

PROYECTO FIN DE MÁSTER

"INTEGRACIÓN DE UNA SOLUCIÓN DE MARKETING AUTOMATION EN PLATAFORMA E-COMMERCE"

DIRECTOR: Daniel Fernández Lanvin

AUTOR: Estefanía Menéndez Freije

Vº Bº del Director del Proyecto

Agradecimientos

A Luis Álvarez Puertas, por darme la oportunidad de formar parte de RICOH.

A Fernando Prieto Ruiz, por asignarme el desarrollo del presente proyecto.

Resumen

El año 2015, ha sido el año de la explosión del comercio electrónico. A día de hoy es indispensable para el éxito y la rentabilidad de cualquier empresa, debido al uso generalizado de Internet y de los dispositivos móviles.

Sectores como la alimentación que se situaban muy por detrás de otros como la moda y la electrónica, están adquiriendo mayor protagonismo. Debido a ello, Supermercados DIA (Distribuidora Internacional de Alimentación), decide actualizar sus estrategias de Marketing, para que sean acordes a la evolución tecnológica. Se decanta por la utilización de una solución de Marketing Automation, que simplifique la creación de sus estrategias de marketing y que incluya elementos útiles para sus clientes, que faciliten sus compras y favorezcan el aumento de ingresos.

En conclusión, se ha desarrollado la integración de la plataforma de comercio electrónico Hybris con BrainSINS, una solución de Marketing Automation. Uno de los resultados más característicos de dicha integración, será la inclusión de los productos recomendados proporcionados por BrainSINS en diferentes plantillas de la página web. De esta forma en función de la página en que se encuentre el usuario, así como de diferentes parámetros asociados al mismo, se le mostrará aquellos productos que tienen una alta probabilidad de resultarle interesantes y por consiguiente, de ser añadidos a su carrito, aumento el precio final de la compra.

Palabras Clave

Marketing, comercio electrónico, Hybris, productos recomendados, ventas.

Abstract

In the year 2015, there was a revolution related to ecommerce. Nowadays, it is essential for the success and profitability of any company, due to widespread use of the Internet and mobile devices.

Sectors like the food industry was behind fashion and electronics back's. However, it is becoming more important. Because of that, DIA Supermarkets decide to update their marketing strategies, to be in line with technological advances. They decide to use a marketing automation solution that simplifies the creation of marketing strategies. In addition, it includes useful elements for its customers, easier to do shopping and to promote supermarket revenue growth.

In conclusion, I have developed the integration of Hybris eCommerce platform with the marketing automation solution called BrainSINS. One of the most remarkable results of this integration, it is the inclusion of BrainSINS' recommended products in different website templates. By this way, according to the visited page and some customer's parameters, the page will include products that have a big probability to be added to customer's cart. This action will increase the total cost of the purchase.

Keywords

Marketing, e-commerce, Hybris, recommended products, sales.

Índice General

CAPÍTULO 1. INTRODUCCIÓN 21

1.1 MOTIVACIÓN Y JUSTIFICACIÓN DEL PROYECTO 21

CAPÍTULO 2. INGENIERÍA DE LA DOCUMENTACIÓN 23

2.1 INTRODUCCIÓN 23
2.2 HISTORIA DEL COMERCIO ELECTRÓNICO 23
2.2.1 Evolución 23
2.2.2 Situación Actual 33
2.2.3 Conclusiones 36
2.3 SOLUCIONES DE COMERCIO ELECTRÓNICO 37
2.3.1 Introducción 37
2.3.2 Soluciones Open Source 37
2.3.3 Soluciones de Pago 42
2.4 MARKETING AUTOMATION 46
2.4.1 Definición 46
2.4.2 Propósito 46
2.4.3 Rentabilidad 46
2.4.4 Beneficios 48
2.5 COMERCIO ELECTRÓNICO 50
2.5.1 Tendencias 50
2.5.2 Miedos y preocupaciones 55
2.5.3 Deseos y tendencias medio plazo 57
2.6 SOLUCIONES MARKETING AUTOMATION 59
2.6.1 SAP Hybris Marketing 59
2.6.2 Webtrekk 62
2.6.3 BrainSINS 66

CAPÍTULO 3. PLANIFICACIÓN DEL PROYECTO Y RESUMEN DE PRESUPUESTOS 71

3.1 PLANIFICACIÓN 71
3.2 RESUMEN DEL PRESUPUESTO 73

CAPÍTULO 4. ANÁLISIS 75

4.1 DEFINICIÓN DEL SISTEMA 75
4.1.1 Determinación del Alcance del Sistema 75
4.2 REQUISITOS DEL SISTEMA 76
4.2.1 Obtención de los Requisitos del Sistema 76
4.2.2 Identificación de Actores del Sistema 83
4.2.3 Especificación de Casos de Uso 84
4.3 IDENTIFICACIÓN DE LOS SUBSISTEMAS EN LA FASE DE ANÁLISIS 89
4.3.1 Identificación de los Subsistemas en la Fase de Análisis 90
4.3.2 Descripción de los Interfaces entre Subsistemas 93
4.4 DIAGRAMA DE CLASES PRELIMINAR DEL ANÁLISIS 93
4.4.1 Diagrama de Clases Sistema Brainsins 94
4.4.2 Diagrama de Clases Sistema Integración Web 100
4.5 ANÁLISIS DE CASOS DE USO Y ESCENARIOS 103

4.5.1 Caso de Uso 1 ... *103*
4.5.2 Caso de Uso 2. Acceso a la página home ... *106*
4.5.3 Caso de Uso 3. Acceso Página Detalle de Producto ... *107*
4.5.4 Caso de Uso 4. Acceso Página Categorías ... *109*
4.5.5 Caso de Uso 5. Acceso a la página del carrito ... *110*
4.5.6 Caso de Uso 6. Página de Confirmación de Pedido ... *112*
4.5.7 Caso de Uso 7. Añadir Producto al Carrito ... *113*
4.5.8 Caso de Uso 8. Añadir unidades al carrito ... *114*
4.5.9 Caso de Uso 9. Eliminar Unidades del Carrito ... *116*
4.5.10 Caso de Uso 10. Inclusión de Recomendador ... *117*
4.6 Análisis de Interfaces de Usuario ... 119
4.6.1 Descripción de la Interfaz ... *119*
4.6.2 Descripción del Comportamiento de la Interfaz ... *119*
4.6.3 Diagrama de Navegabilidad ... *119*
4.7 Especificación del Plan de Pruebas ... 120
4.7.1 Pruebas Unitarias ... *120*
4.7.2 Pruebas de Integración y Sistema ... *120*
4.7.3 Pruebas de Usabilidad ... *124*

CAPÍTULO 5. DISEÑO DEL SISTEMA ... 125

5.1 Arquitectura del Sistema ... 125
5.1.1 Diagramas de extensiones ... *125*
5.1.2 Diagramas de Componentes ... *127*
5.1.3 Diagramas de Despliegue ... *131*
5.2 Diseño de Clases ... 134
5.2.1 Diagrama de Clases Exportación ... *134*
5.2.2 Diagrama Clases Negocio Extensión Diabrainsins ... *135*
5.2.3 Diagrama de Clases Negocio Extensión diaacceleratorfacades ... *136*
5.2.4 Diagrama Clases Persistencia Extensión diaacceleratorcore ... *137*
5.3 Diagrama de Clases Integración Sitio Web ... 138
5.3.1 Diagrama de clases Presentación ... *138*
5.3.2 Diagrama de Clases Presentación Paquete brainsins.service ... *140*
5.3.3 Diagrama de Clases Negocio Extensión diaacceleratorfacades ... *140*
5.4 Diagramas de Interacción y Estados ... 142
5.4.1 Caso de Uso 1: Ejecutar Cronjob ... *142*
5.4.2 Caso de Uso 2: Acceso Página Home ... *144*
5.4.3 Caso de Uso 3: Acceso Página Detalle de Producto ... *145*
5.4.4 Caso de Uso 4: Acceso Página Categorías ... *146*
5.4.5 Caso de Uso 5: Acceso Página Carrito ... *147*
5.4.6 Caso de Uso 6: Acceso Página Confirmación Pedido ... *148*
5.4.7 Caso de Uso 7: Añadir Producto al Carrito ... *149*
5.4.8 Caso de Uso 8: Añadir Unidades al Carrito ... *150*
5.4.9 Caso de Uso 9: Eliminar Unidades del Carrito ... *150*
5.4.10 Caso de Uso 10: Inclusión Recomendador en Página ... *150*
5.5 Diseño de la Interfaz ... 152
5.5.1 General ... *152*
5.5.2 Funcionalidad añadir al carrito ... *152*

CAPÍTULO 6. IMPLEMENTACIÓN DEL SISTEMA ... 153

6.1 Estándares y Normas Seguidos ... 153

6.2 LENGUAJES DE PROGRAMACIÓN 153
6.2.1 Java 153
6.2.2 Spring Framework 154
6.2.3 JSP 154
6.2.4 CSS 155
6.2.5 JavaScript 155
6.2.6 jQuery 155
6.3 HERRAMIENTAS Y PROGRAMAS USADOS PARA EL DESARROLLO 156
6.3.1 Eclipse 156
6.3.2 Hybris 156
6.3.3 HAC 156
6.3.4 HMC 157
6.3.5 CMSCockpit 157
6.3.6 BrainSINS Platform 157
6.3.7 Oracle SQL Developer 157
6.4 CREACIÓN DEL SISTEMA 158
6.4.1 Problemas Encontrados 158
6.4.2 Descripción Detallada de las Clases 159

CAPÍTULO 7. DESARROLLO DE LAS PRUEBAS 175

7.1 PRUEBAS DE INTEGRACIÓN Y DEL SISTEMA 175
7.2 PRUEBAS DE USABILIDAD Y ACCESIBILIDAD 188
7.2.1 Pruebas de Usabilidad 188
7.2.2 Pruebas de Accesibilidad 193
7.3 PRUEBAS DE RENDIMIENTO 197

CAPÍTULO 8. MANUALES DEL SISTEMA 199

8.1 MANUAL DE INSTALACIÓN 199
8.1.1 Preparación del entorno 199
8.2 MANUAL DE EJECUCIÓN 199
8.3 MANUAL DE GESTOR DIA 200
8.3.1 BrainSINS 200
8.3.2 Recomendadores BrainSINS 201
8.3.3 Integración vía CMS 202
8.3.4 Caso práctico 202
8.4 MANUAL DEL CLIENTE DIA 208
8.4.1 Introducción 208
8.4.2 Apariencia 208
8.4.3 Funcionalidad 209
8.4.4 Añadir producto 209
8.4.5 Añadir una unidad 209
8.4.6 Eliminar una unidad 209
8.4.7 Unidades incluidas en el carrito 210
8.5 MANUAL DEL PROGRAMADOR 210
8.5.1 BrainSINS 210
8.5.2 Descripción de la Integración 211
8.5.3 Importacion del Catálogo XML 214
8.5.4 Vista de los Recomendadores 215
8.5.5 Inclusión del Recomendador vía CMS 217
8.5.6 Integración Front 217

CAPÍTULO 9. CONCLUSIONES Y AMPLIACIONES 219

9.1 CONCLUSIONES 219
9.2 AMPLIACIONES 220

CAPÍTULO 10. REFERENCIAS BIBLIOGRÁFICAS 221

10.1 REFERENCIAS EN INTERNET 221

CAPÍTULO 11. APÉNDICES 224

11.1 GLOSARIO Y DICCIONARIO DE DATOS 224
11.2 CONTENIDO ENTREGADO 226
11.3 CÓDIGO FUENTE 227
11.3.1 Extensión diabrainsins 227
11.3.2 Extensión diaacceleratorstorefront 253
11.3.3 Extensión diaacceleratatorfacades 315
11.3.4 Extensión diaacceleratorcore 325

Índice de Figuras

Ilustración 1: Gráfica evoluitva de Internautas compradores 24
Ilustración 2: Lugar habitual de compras por Internet....24
Ilustración 3: Las 10 redes sociales más usadas del mundo 25
Ilustración 4: Usuarios activos en redes sociales 25
Ilustración 5: Volumen comercio electrónico B2C 26
Ilustración 6: Evolución Internautas compradores 26
Ilustración 7: Canales frecuentes de búsqueda de información....27
Ilustración 8: Internautas que han comprado productos online vs offline 27
Ilustración 9: Internautas que han comprado productos online vs offline....28
Ilustración 10: Internautas compradores 28
Ilustración 11: Dispositivos de acceso a Internet en Porcentaje 29
Ilustración 12: Disponibilidad de Correo Electrónico en Porcentaje 29
Ilustración 13: Volumen del Comercio Electrónico en Millones de Euros....30
Ilustración 14: Acceso a Internet en Hogares 30
Ilustración 15: Dispositivos de Acceso a Internet en Porcentaje....31
Ilustración 16: Disponibilidad de Correo Electrónico en Porcentaje 31
Ilustración 17: Volumen de Compras Online 32
Ilustración 18: Previsión de Facturación Black Friday 33
Ilustración 19: Porcentaje de ventas Black Friday....34
Ilustración 20: Promedio de facturación Black Friday....34
Ilustración 21: Porcentaje de empresas con inversión en el soporte 35
Ilustración 22: Porcentaje de empresas que aumentan inversión en el soporte....36
Ilustración 23: Adopción por Empresas B2B 47
Ilustración 24: Opiniones sobre Precios....48
Ilustración 25: Características Hybris....60
Ilustración 26: Viaje de Cliente....64
Ilustración 27: Ejemplo de viaje de cliente en Pedido....64
Ilustración 28: Especificación del viaje de cliente 65
Ilustración 29: Opiniones de clientes....66
Ilustración 30: Esquema de funcionamiento 67
Ilustración 31: Diagrama de Gantt – Parte I....71
Ilustración 32: Diagrama de Gantt – Parte II....72
Ilustración 33: Diagrama de Gantt – Parte IV 72
Ilustración 34: Resumen del Presupuesto....74
Ilustración 35: Actores del Sistema 84
Ilustración 36: Casos de Uso Generales 85
Ilustración 37: Casos de Uso Acceso a las páginas del sitio web 86
Ilustración 38: Casos de Uso Funcionalidad Añadir al carrito....88
Ilustración 39: Diagrama de paquetes de los subsistemas....90
Ilustración 40: Diagrama de Clases Integración Web 92
Ilustración 41: Diagrama de Clases Extensión diabrainsins....94
Ilustración 42: Diagrama Clases diaacceleratorfacades 97
Ilustración 43: Diagrama Clases Extensión diaacceleratorcore 98
Ilustración 44: Diagrama de Clases Extensión diaacceleratorstorefront 100
Ilustración 45: Diagrama Caso de Uso Ejecutar Job – Parte I 103

Ilustración 46: Diagrama Caso de Uso Ejecutar Job – Parte II 103
Ilustración 47: Diagrama Caso de Uso Ejecutar Job – Parte III 104
Ilustración 48: Caso de Uso Acceso página home 106
Ilustración 49: Caso de Uso Acceso página detalle de producto 107
Ilustración 50: Caso de Uso Acceso página categorías 109
Ilustración 51: Caso de Uso Acceso página del carrito 110
Ilustración 52: Caso de Uso Acceso página confirmación pedido 112
Ilustración 53: Caso de Uso Añadir producto al carrito 113
Ilustración 54: Caso de Uso Inclusión de recomendador 117
Ilustración 55: Diagrama extensiones 125
Ilustración 56: Diagrama Componentes Exportación 128
Ilustración 57: Diagrama Componentes Sitio Web 130
Ilustración 58: Diagrama de Despliegue 132
Ilustración 59: Diagrama Clases Exportación 134
Ilustración 60: Diagrama de Clases Diabrainsins 136
Ilustración 61: Diagrama Clases Diaacceleratorfacades 137
Ilustración 62: Diagrama Clases Extensión diaacceleratorcore 138
Ilustración 63: Diagrama de Clases Presentación 139
Ilustración 64: Diagrama de Clases Presentación 140
Ilustración 65: Diagrama de Clases Negocio 141
Ilustración 66: Diagrama Secuencia Caso de Uso 1 – Parte I 143
Ilustración 67: Diagrama Secuencia Caso de Uso 1 – Parte II 144
Ilustración 68: Diagrama Secuencia Caso de Uso 2 145
Ilustración 69: Diagrama Secuencia Caso de Uso 3 146
Ilustración 70: Diagrama Secuencia Caso de Uso 4 147
Ilustración 71: Diagrama Secuencia Página Carrito 148
Ilustración 72: Diagrama Secuencia Caso de Uso 6 149
Ilustración 73: Diagrama de Secuencia Caso de Uso 7 150
Ilustración 74: Diagrama Secuencia Caso de Uso 10 151
Ilustración 75: Interfaz Carruseles BrainSINS 152
Ilustración 76: Integración CMS – PASO 1 203
Ilustración 77: Integración CMS – Paso 2 203
Ilustración 78: Integración CMS – Paso 4 204
Ilustración 79: Integración CMS – Paso 5 204
Ilustración 80: Integración CMS – Paso 6 205
Ilustración 81: Integración CMS – Paso 7 205
Ilustración 82: Integración CMS – Paso 8 206
Ilustración 83: Integración CMS – Paso 9 207
Ilustración 84: Integración CMS – Paso 11 207
Ilustración 85: Integración CMS - Resultado 208
Ilustración 86: BrainSINS Apariencia 209
Ilustración 87: Importación XML – Paso 1 a 214
Ilustración 88: Integración XML – Paso 1 b 215
Ilustración 89: Integración XML – Paso 2 215
Ilustración 90: BrainSINS listado 216
Ilustración 91: BrainSINS Estilo 217

Capítulo 1. Introducción

1.1 Motivación y Justificación del Proyecto

El año 2015, ha sido el año de la explosión del comercio electrónico. A día de hoy es indispensable para el éxito y la rentabilidad de cualquier empresa, debido al uso generalizado de Internet y de los dispositivos móviles.

Sectores como la alimentación que se situaban muy por detrás de otros como la moda y la electrónica, están adquiriendo mayor protagonismo. Debido a ello, Supermercados DIA (Distribuidora Internacional de Alimentación), decide actualizar sus estrategias de Marketing, para que sean acordes a la evolución tecnológica. Se decanta por la utilización de una solución de Marketing Automation, que simplifique la creación de sus estrategias de marketing y que incluya elementos útiles para sus clientes , que faciliten sus compras y favorezcan el aumento de ingresos.

En conclusión, se ha desarrollado la integración de la plataforma de comercio electrónico Hybris con BrainSINS, una solución de Marketing Automation. Uno de los resultados más característicos de dicha integración, será la inclusión de los productos recomendados proporcionados por BrainSINS en diferentes plantillas de la página web. De esta forma en función de la página en que se encuentre el usuario, así como de diferentes parámetros asociados al mismo, se le mostrará aquellos productos que tienen una alta probabilidad de resultarle interesantes y por consiguiente, de ser añadidos a su carrito, aumentando el precio final de la compra.

Capítulo 2. Ingeniería de la Documentación

2.1 Introducción

A lo largo de este capítulo, se van a mostrar los diferentes aspectos estudiados en la elaboración del proyecto. Se va a comenzar por un estudio del comercio electrónico, comenzando por el inicio de su historia, para continuar con sus progresos. También se mostrarán las diferentes soluciones que existen en el mercado, tanto opensource como de pago.

Por otro lado, se explicará la definición del Marketing Automation, con todo lo que ello implica.

2.2 Historia del Comercio Electrónico

En los últimos años, la sociedad ha registrado un cambio en sus vidas cotidianas y en su forma de interaccionar. Este cambio, se ha producido con la inclusión de las nuevas tecnologías de la información y la comunicación (TIC) en el día a día de las personas.

El sector empresarial, está viviendo una nueva revolución en el modo de venta y captación de nuevos clientes. Todos los puntos de venta, con independencia del tipo de producto que ofrezcan, necesitan estar presentes en Internet. De otra forma, no podrán sobrevivir en esta nueva era, donde el comercio electrónico juega un papel fundamental.

2.2.1 Evolución

En los estudios llevados a cabo por el Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información (ONTSI), se puede observar el gran crecimiento que ha sufrido España en el B2C.

El Estudio sobre comercio electrónico B2C 2010 llevado a cabo en Octubre de 2010, muestra cómo el aumento del número de internautas, ha sido el motor principal para el aumento de compras por Internet.

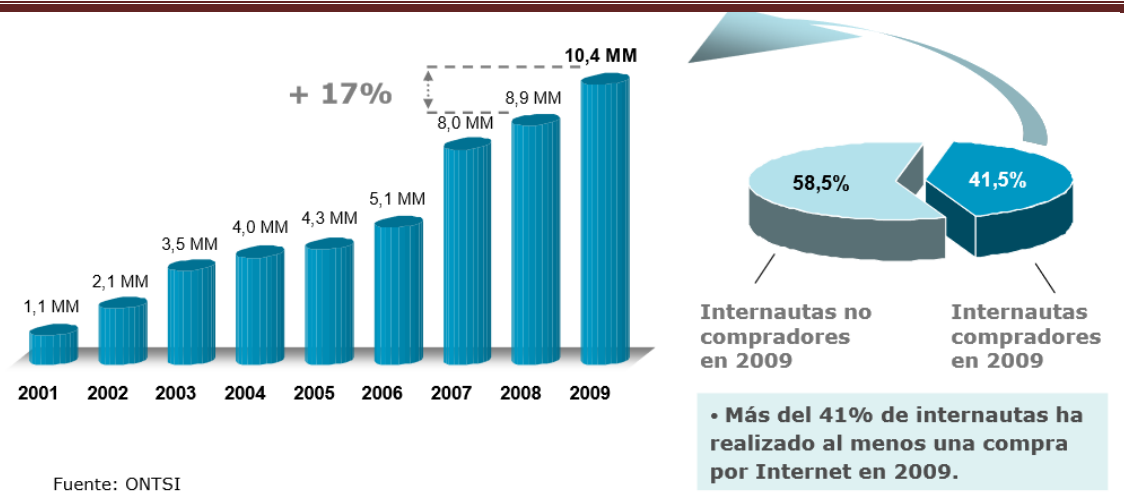


Ilustración 1: Gráfica evoluitva de Internautas compradores

Además, nueve de cada diez compradores on-line efectúa la compra desde el hogar.

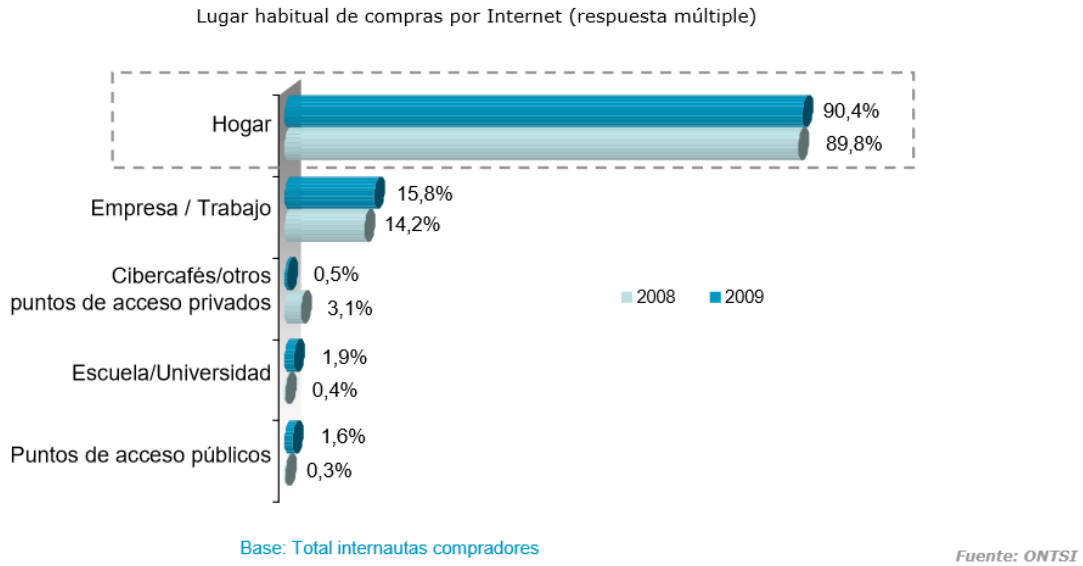


Ilustración 2: Lugar habitual de compras por Internet

[ONTSI10]

Según la ONTSI, el año 2010 ha marcado un punto de inflexión para el sector TIC mundial. Dicho mercado movió en el mundo 2,75 billones de euros, debido a la expansión de Internet móvil y de las redes sociales. Se produce el inicio de la "Administración electrónica", hacia el "gobierno abierto". Se produce la apuesta por el "Cloud Computing" y la extensión del uso de terminales inteligentes y tarjetas electrónicas. Combinan las funciones de la telefonía, el ordenador, la conexión a Internet y el acceso a mundos multimedia.

Las 10 redes sociales más usadas en el mundo.

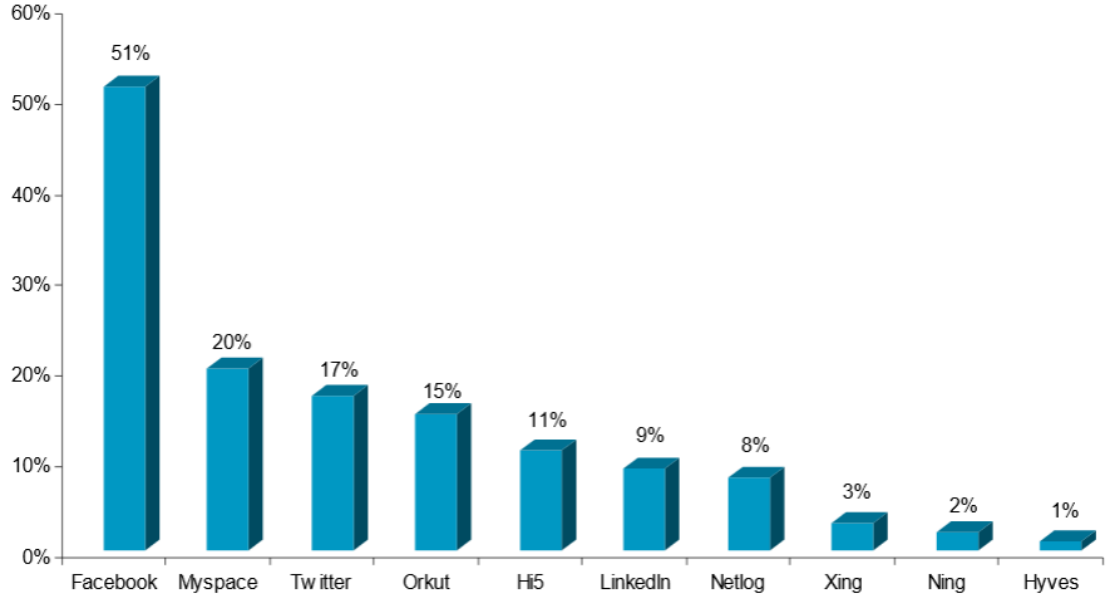


Ilustración 3: Las 10 redes sociales más usadas del mundo

Usuarios activos de Internet que gestionan algún perfil en una red social.

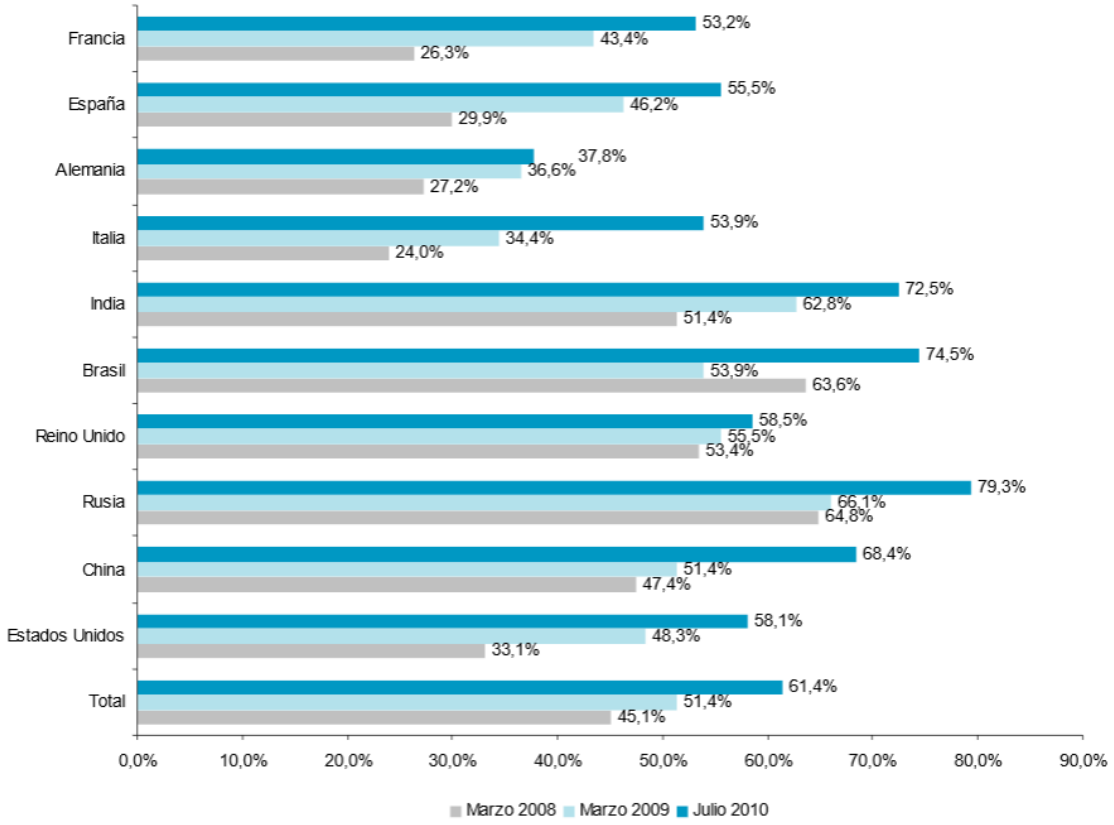
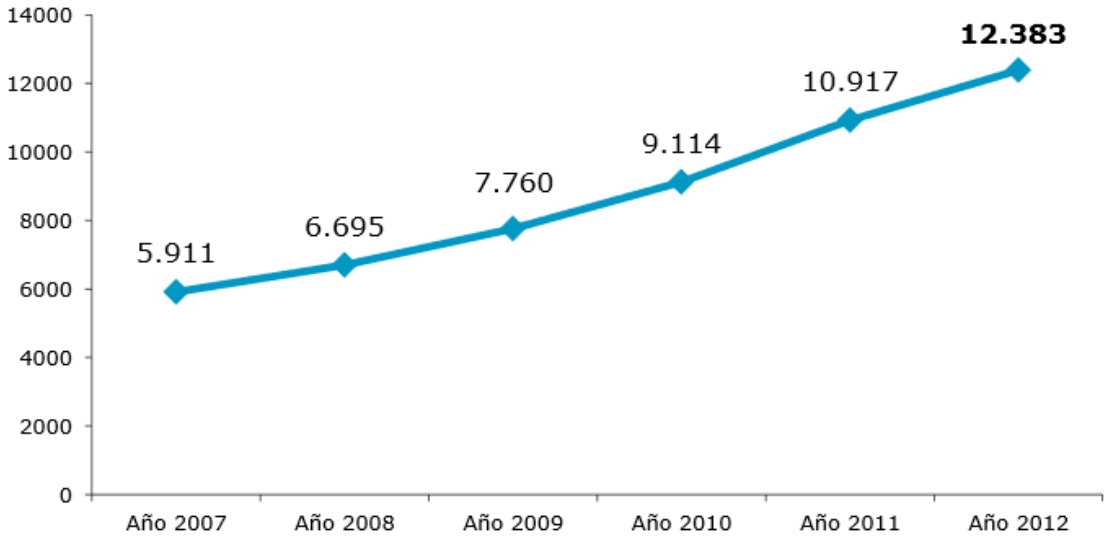


Ilustración 4: Usuarios activos en redes sociales

[ONTSI11]

Según un estudio de la situación del B2C en España llevado a cabo en 2012.

Volumen del comercio electrónico B2C en millones de euros.



Fuente: Panel Hogares, ONTSI

Ilustración 5: Volumen comercio electrónico B2C

Evolución en el número de internautas e internautas compradores.

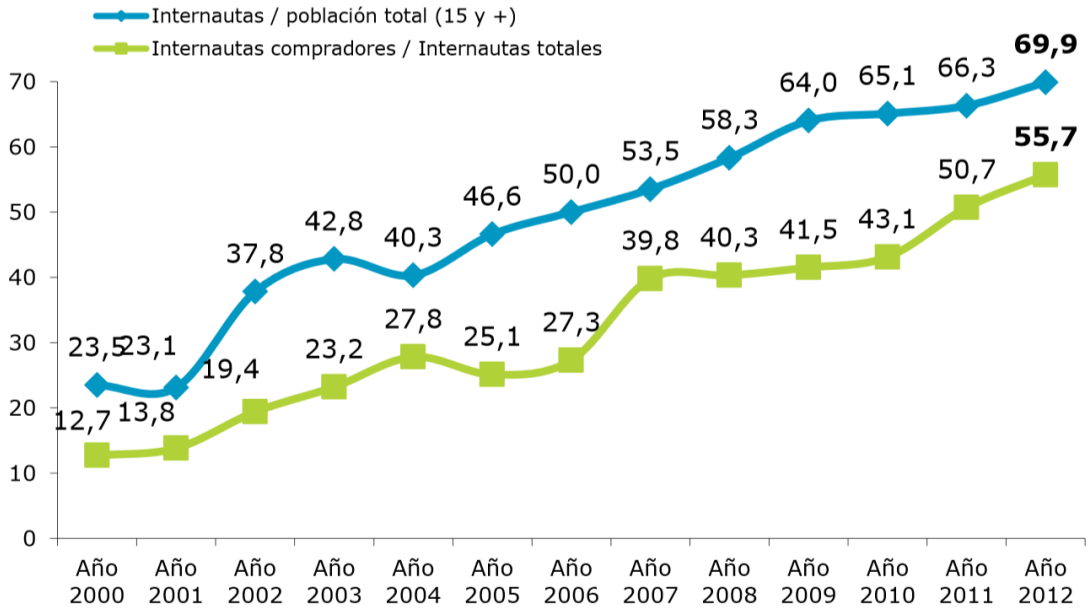
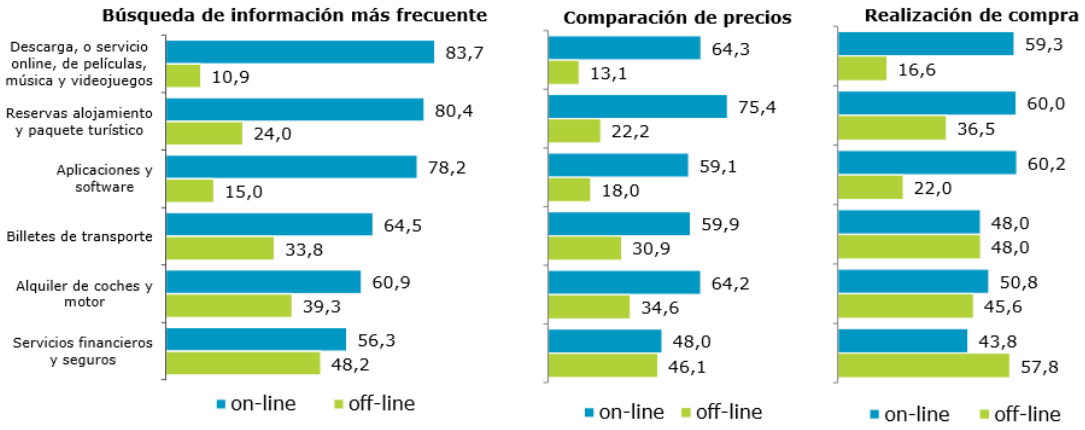


Ilustración 6: Evolución Internautas compradores

Se puede decir que el incremento observado en el volumen de comercio electrónico B2C de 2012 se debe al incremento de internautas compradores y en menor medida al incremento de internautas general.

Tabla resumen comercio electrónico 2012.

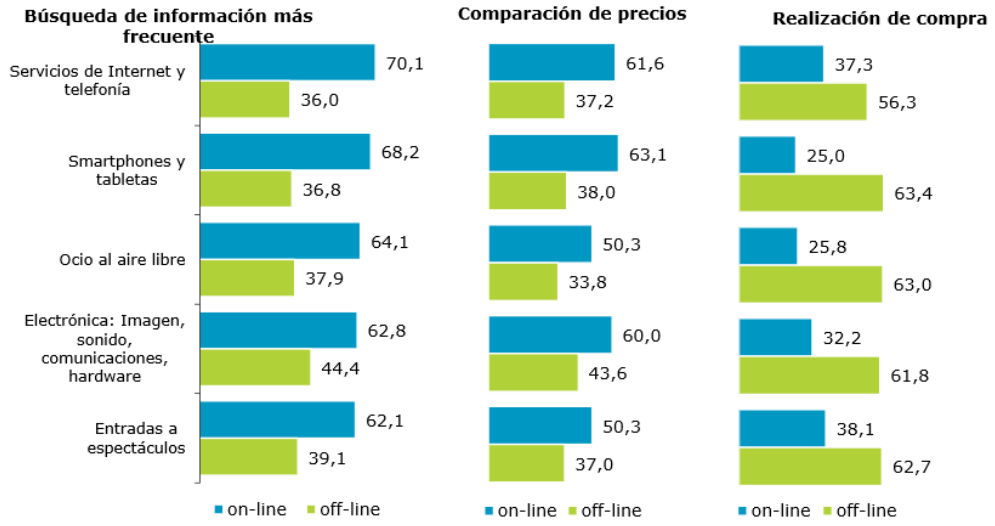
Canal más frecuente de búsqueda de información, comparación de precio y compra (%)



Base: Internautas que han comprado el producto (online u offline) en 2012
Fuente: Panel Hogares, ONTSI

Ilustración 7: Canales frecuentes de búsqueda de información

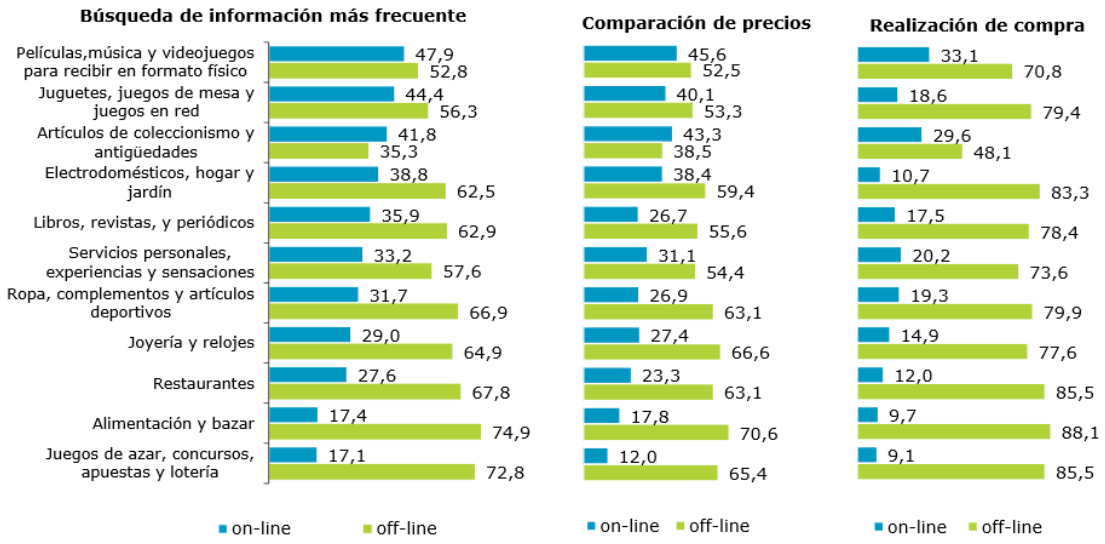
Se observa un comportamiento homogéneo en todos los productos y servicios dentro de la categoría. La búsqueda y comparación de precios se realiza principalmente en el canal online mientras que se compra principalmente offline.



Base: Internautas que han comprado el producto (online u offline) en 20112
Fuente: Panel Hogares, ONTSI

Ilustración 8: Internautas que han comprado productos online vs offline

En todos los casos, la búsqueda de información, la comparación de precios y la compra es principalmente y con diferencia offline.



Base: Internautas que han comprado el producto (online u offline) en 2012
Fuente: Panel Hogares, ONTSI

Ilustración 9: Internautas que han comprado productos online vs offline

Aparecen las compras desde movilidad.

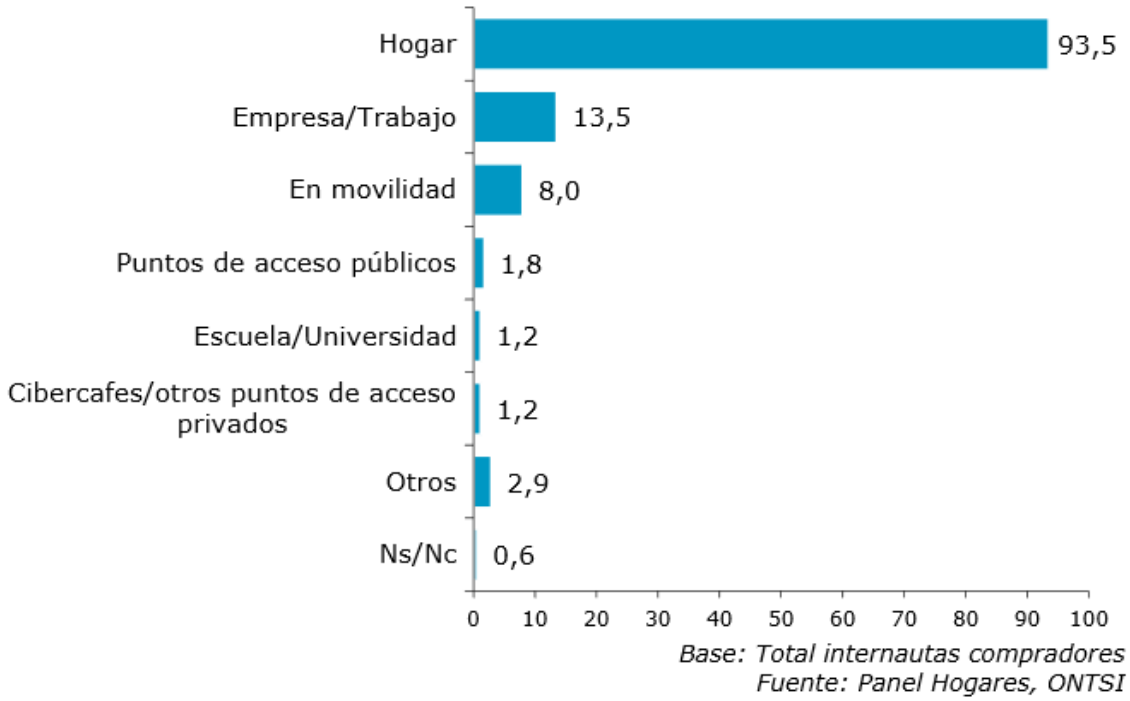


Ilustración 10: Internautas compradores

[ONTSI12]

Según el informe anual de la sociedad en red 2013 (edición 2014)

Dispositivo de acceso a internet en porcentaje:

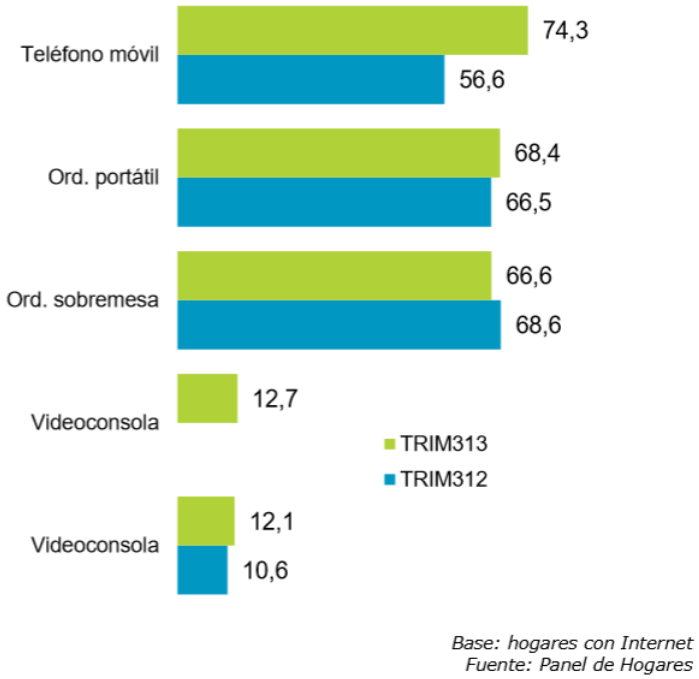


Ilustración 11: Dispositivos de acceso a Internet en Porcentaje

La conexión a través del teléfono móvil continúa en su tendencia ascendente con un notable incremento de 17,7 puntos porcentuales, es utilizada en el 74,3% de los hogares. En el tercer semestre el ordenador portátil supera al ordenador de sobremesa.

Disponibilidad de correo electrónico en porcentaje:

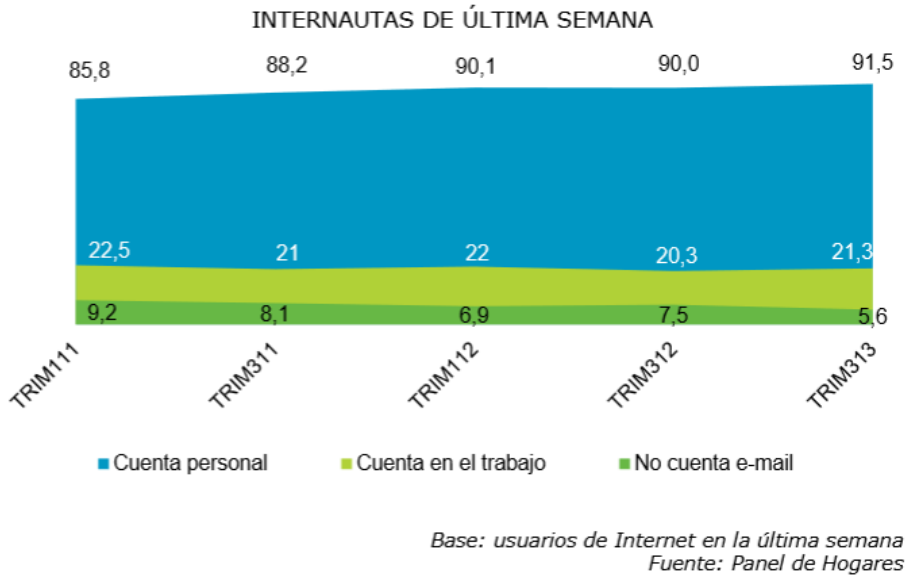


Ilustración 12: Disponibilidad de Correo Electrónico en Porcentaje

El 91,5%, de aquellos que hicieron uso de la Red en la última semana, disponen de cuenta de correo personal y el 21,3% tiene una cuenta en el trabajo. Solo un 5,6% no dispone de cuenta de correo electrónico.

Volumen del comercio electrónico en España (Millones de €)

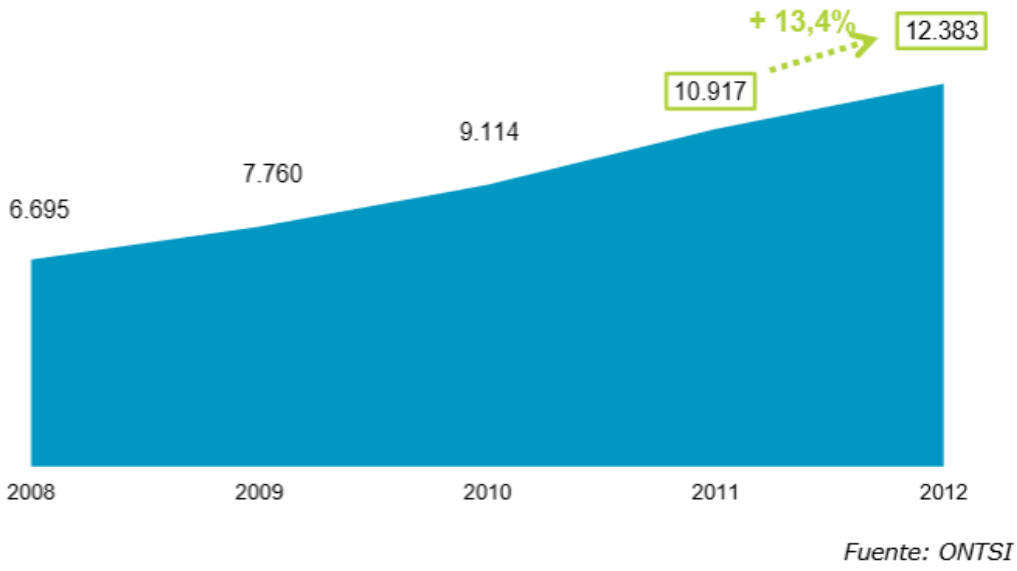


Ilustración 13: Volumen del Comercio Electrónico en Millones de Euros

Incremento del 13.4% respecto a 2010. Este nivel de crecimiento electrónico es relevante ya que se produce en un contexto económico muy adverso, como es el de la crisis económica. Esto pone de manifiesto, que esta nueva forma de comercio, posee una gran capacidad de generar rendimientos, incluso en situaciones adversas. Además, se trata de un canal en unos casos adicional y en otros único, e incluso, en el caso de algunos negocios, un canal sustitutorio del tradicional.

[ONTSI13]

Según el informe 2014.

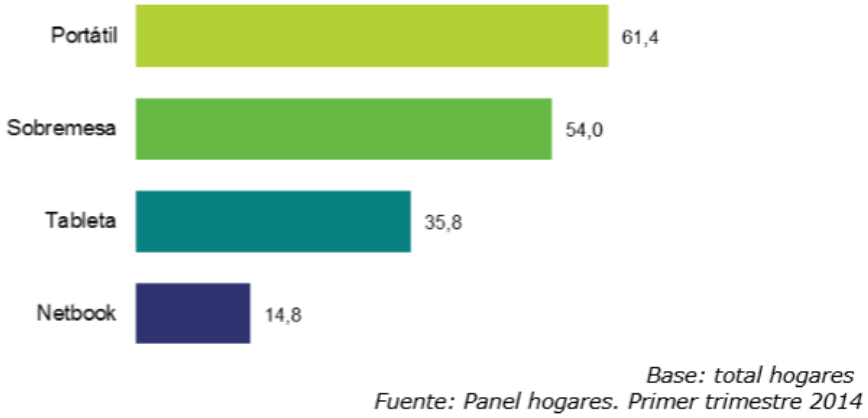


Ilustración 14: Acceso a Internet en Hogares

Dispositivos de acceso a Internet en porcentaje.

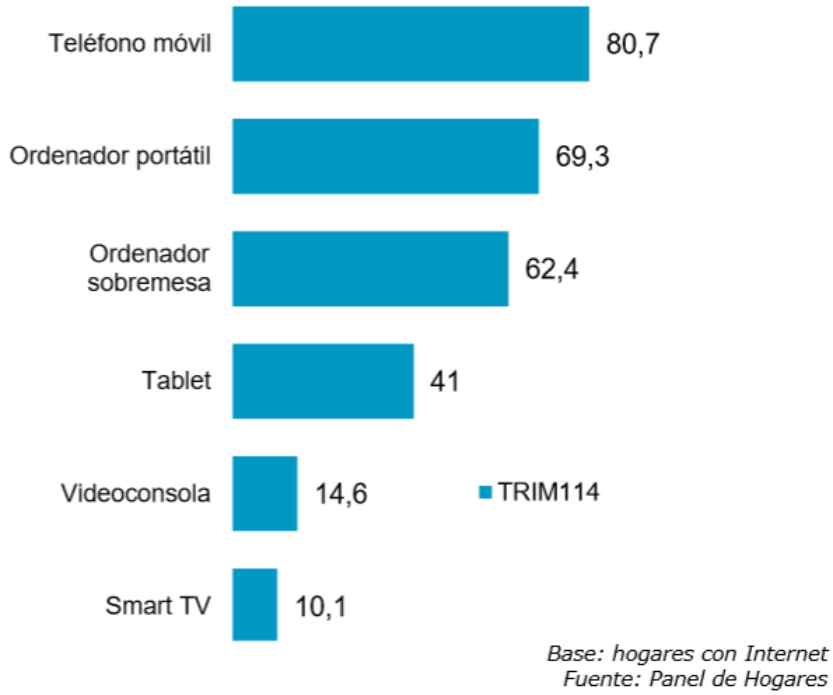


Ilustración 15: Dispositivos de Acceso a Internet en Porcentaje

Ocho de cada diez hogares utiliza el teléfono móvil para acceder a Internet.

Disponibilidad de correo electrónico en porcentaje.

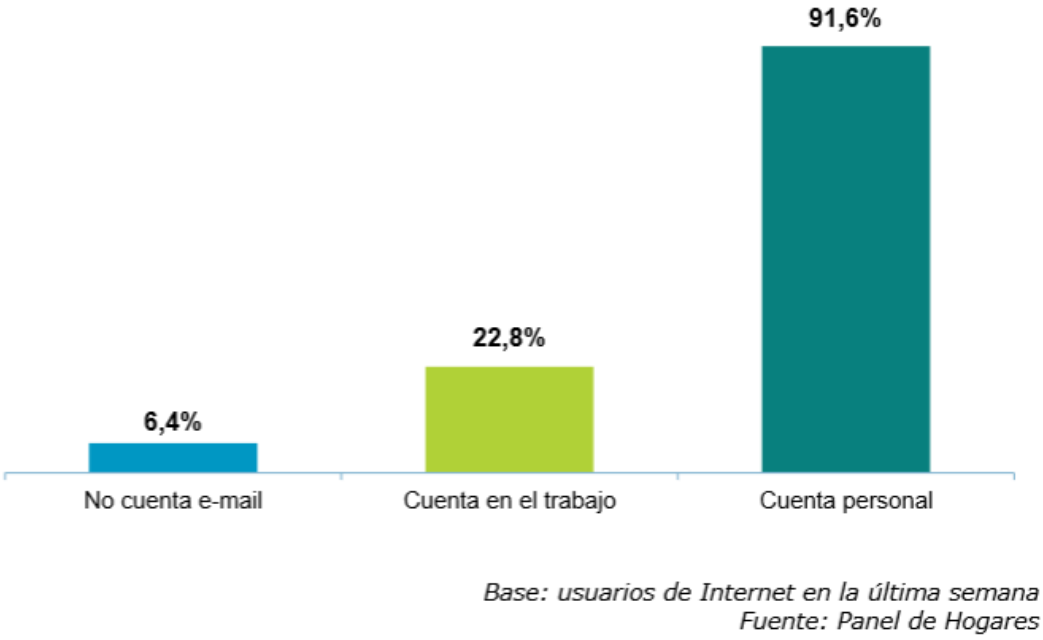


Ilustración 16: Disponibilidad de Correo Electrónico en Porcentaje

Volumen del comercio electrónico B2C en millones de euros.

Para comprender este crecimiento hay que tener en cuenta que en 2013 el incremento del gasto medio por comprador coincide con la evolución positiva de las dos variables principales que se utilizan para estimar el volumen del B2C, el porcentaje de internautas y la proporción de los que realizan compras online.

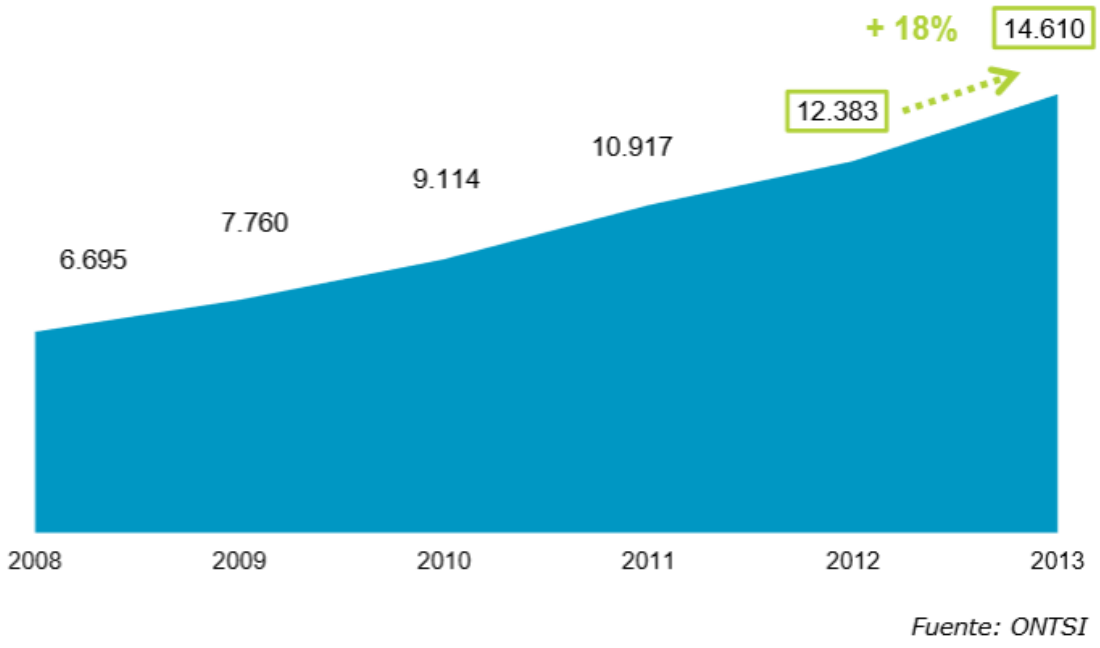


Ilustración 17: Volumen de Compras Online

Este aumento no se entendería sin el continuo crecimiento de la población internauta, la cual alcanzó la cifra de 28,4 millones, representando un 73,1% de la probación española mayor de 15 años y aumentó con respecto al año anterior un 4,4%. Crece de manera destacable el porcentaje de internautas que han efectuado compras en el último año.

El volumen de compra total, el gasto medio por individuo comprador, ha crecido un 3,9% en 2013, pasando de 816€ en 2012 a 848€, invirtiendo así la tendencia de baja de los dos años anteriores.

Los internautas compradores con mayor gasto medio son hombres, mayores de 65 años, residentes en poblaciones de 20 a 50 mil habitantes y clase media-alta.

Por categorías, los billetes de transporte y las reservas de alojamiento y paquetes turísticos siguen manteniendo una posición predominante. Ropa, complementos y artículos deportivos y entradas a espectáculos continúan creciendo.

[ONTSI14]

2.2.2 Situación Actual

Desde la Asociación Española de Tecnología Digital (adigital), podemos observar la situación actual del comercio electrónico en España.

En el informe "Informe eCommerce Black Friday 2015" de adigital, podemos extraer las siguientes conclusiones: **[ADIGITALBF15]**

En este informe se lleva a a cabo un estudio sobre "Black Friday" y "Ciber Monday", momento en que se inicia la campaña de Navidad.

Se trata de una campaña que cada vez tiene más peso en España. Un 93,10% de las empresas españolas lo conocen, siendo un 68,97% las empresas que van a llevar una oferta especial en dichas fechas. Cabe destacar que el 100% de las empresas que hicieron campaña en 2014, repiten en 2015. Además un 20% de las empresas realizarán campaña en 2015.

"La previsión de facturación en Black Friday ha sido de un total de 1172 millones de euros como se puede observar en el siguiente diagrama:" **[ADIGITALBF15]**

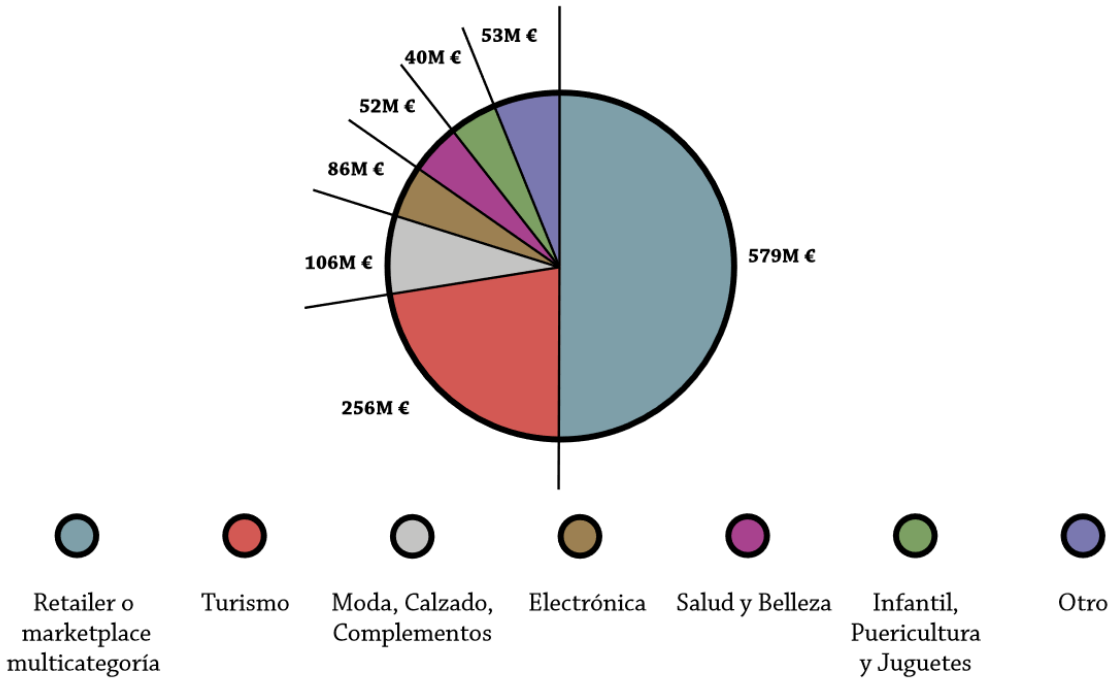


Ilustración 18: Previsión de Facturación Black Friday

"En promedio, el incremento de las ventas en 2015 frente a 2014 en la semana Black Friday se sitúa en 10,6%." **[ADIGITALBF15]**

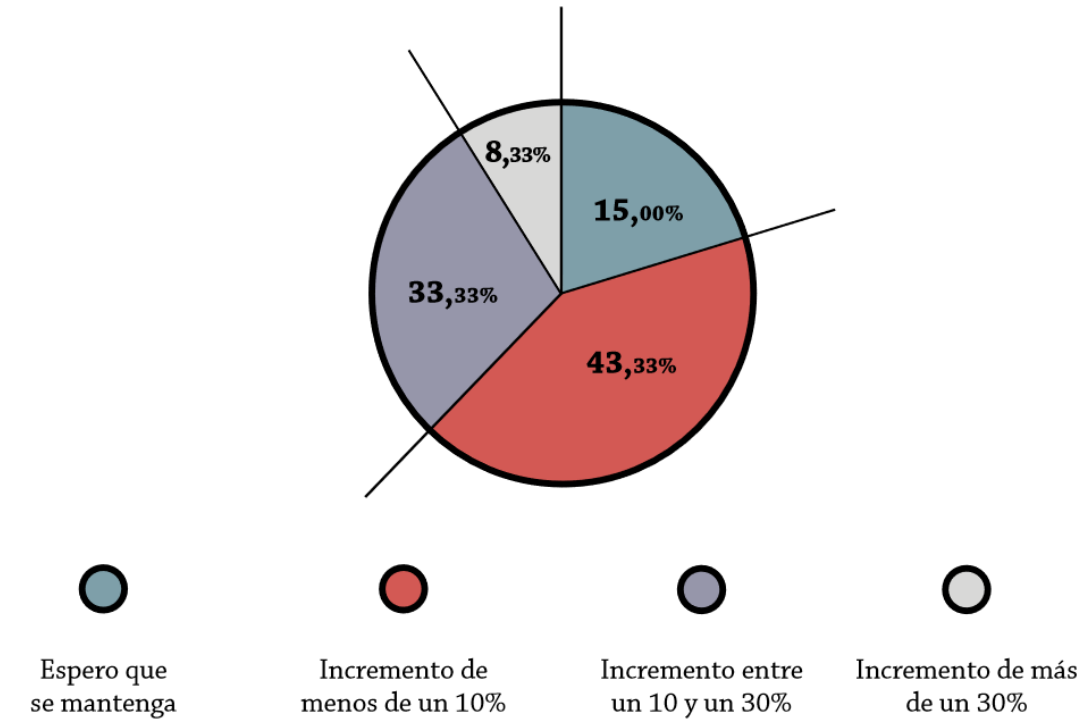


Ilustración 19: Porcentaje de ventas Black Friday

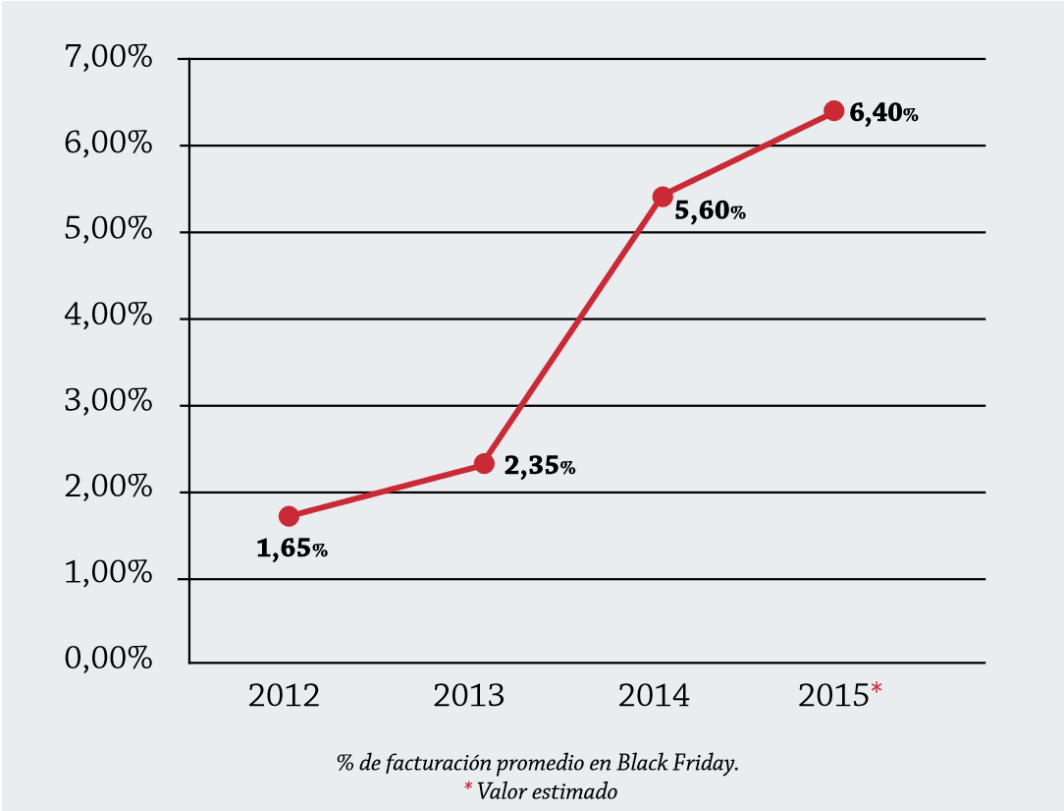


Ilustración 20: Promedio de facturación Black Friday

La evolución anual del porcentaje de facturación en Black Friday frente a la facturación anual.

El 66,67% llevan a cabo una inversión en publicidad para la campaña. Siendo los soportes más utilizados los que se muestran a continuación. **[ADIGITALBF15]**

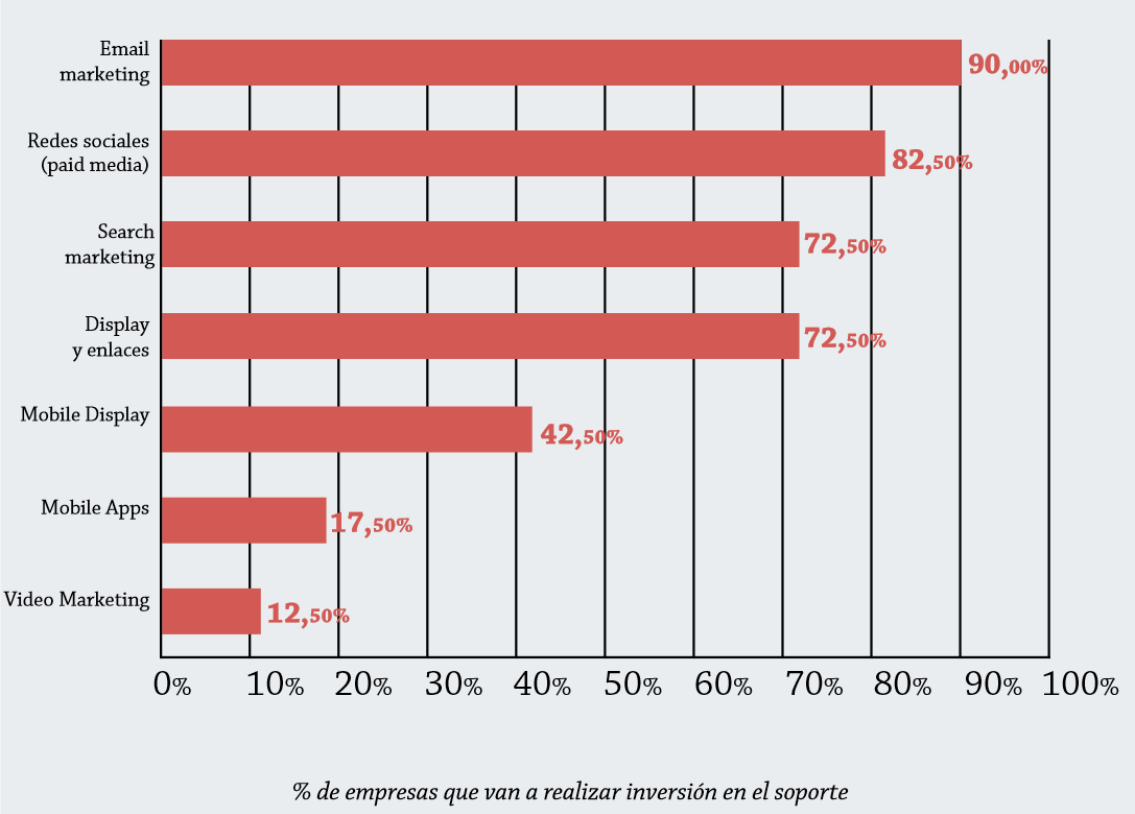


Ilustración 21: Porcentaje de empresas con inversión en el soporte

Los soportes donde más crece la inversión en las campañas son:

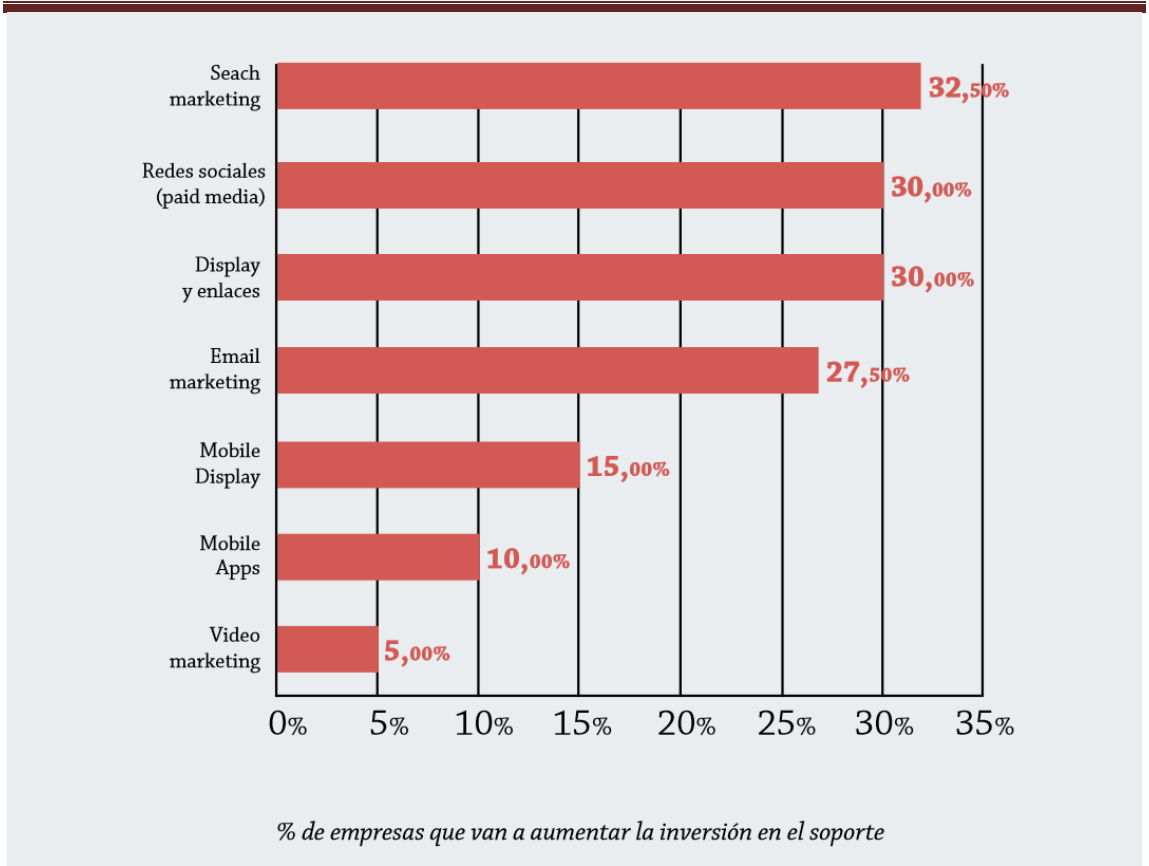


Ilustración 22: Porcentaje de empresas que aumentan inversión en el soporte

2.2.3 Conclusiones

Como se ha podido observar a lo largo de este apartado, el comercio electrónico ha ido desarrollándose con la inclusión de Internet y el uso de dispositivos móviles en la vida cotidiana.

El año 2010, representa un punto de inflexión para el Comercio Electrónico a nivel mundial. Dicho mercado llegó a mover en el mundo 2,75 millones de euros. Es a partir de este momento, cuando se convierte en una pieza clave para todas las empresas, permitiendo incrementar su facturación incluso en una situación difícil, debido a la Crisis Económica. Todas las empresas buscan hacerse un hueco en Internet para incrementar sus ventas. Llegando a aparecer incluso nuevos modelos de negocio, orientados exclusivamente a las ventas por Internet.

El Comercio Electrónico genera nuevos tipos de campañas publicitarias, que además, se realizan en nuevos tipos de soportes. Email marketing, Redes sociales... Debido a ello, las empresas tienen que adaptarse a nuevas formas publicitarias, apareciendo la necesidad de utilizar algún tipo de software que les pueda ayudar en estos nuevos casos.

Aparece además el concepto de omnicanalidad, que se basa en la integración de todos los canales existentes en el mercado. Buscando generar una sensación única a través de la combinación de cualquier vía de comunicación, lo que mejora la relación con los clientes.

2.3 Soluciones de Comercio Electrónico

2.3.1 Introducción

El comercio electrónico, genera en las distintas empresas, la necesidad de disponer de una herramienta que permita mostrar el catálogo de los productos que ofrece, permitir su venta a través de la red, llevar a cabo tareas de gestión de productos, clientes, tiendas... Gestionar los propios contenidos mostrados en su web...

Para llevar a cabo todas las tareas mencionadas anteriormente de un modo centralizado, aparecen soluciones de Comercio Electrónico. Éstas variarán en función de la empresa donde quiera utilizarse, el volumen de ventas, la complejidad del negocio... Aparecen tanto soluciones Open-Source, ideales para PYMES, como soluciones de pago, para grandes empresas.

2.3.2 Soluciones Open Source

2.3.2.1 PrestaShop

"PrestaShop, se trata de una de las soluciones más populares en el mundo. Es número uno en España con más de 40.000 tiendas, cuyo ranking es liderado por Madrid." **[Pablo15]**

PrestaShop se trata de una compañía comercial, que busca situarse en el nivel más alto con respecto a sus competidores. Posee múltiples funcionalidades:

Plantillas: Más de 2000 plantillas profesionales que causan una gran impresión en los clientes. Sus diseños están adaptados para móviles. Permite además su personalización.

CMS eCommerce: más de 310 funcionalidades integradas. Divididas en diferentes apartados:

- Experiencia de compra completa e intuitiva.
- Diseño de la ficha de producto.
- Búsqueda de productos filtrada e instantánea. Permite a tu cliente encontrar el producto que busca fácilmente con filtros de búsqueda, como precio, color o marca, entre muchos otros más.
- Navegación intuitiva. Navegación accesible con opciones personalizables y un diseño profesional.
- Back office centralizado.
- Funcionalidades de instalación rápida. Cientos de funcionalidades activables con un sólo clic.
- Interfaz intuitiva. Administración, actualización y edición de productos, pedidos, clientes, contenido y mucho más con un back-office potente y fácil de utilizar.
- Análisis de datos completos y útiles. Permite obtener una visión instantánea del negocio gracias a estadísticas y análisis únicos.

- Vender en Internet: Permite vender online de manera confidencial con potentes herramientas de pago y envío.
- Acepta pagos instantáneamente. Posee más de cincuenta soluciones y pasarelas de pago con mayor renombre mundial. Acepta pagos online de las principales tarjetas de crédito y débito, cheques electrónicos, tarjetas regalo...
- Cuenta con los principales transportistas y empresas de logística. Ofrece a los clientes de todo el mundo un envío seguro, instalando todas las empresas y zonas necesarias. Controla los detalles logísticos, como las tarifas del peso, así como los descuentos, directamente desde tu back-office.
- Configuración sencilla de los impuestos. Cuenta con un sistema de seguimiento avanzado para reconocer la procedencia de los clientes y calcular automáticamente los impuestos aplicables.
- Proceso de pago personalizado. Personalización del canal de pago para ofrecer una experiencia de compra sencilla. Ofrece a los clientes la comodidad de un proceso de compra exprés, de invitado y de cuenta; incluso acepta pedidos por teléfono.
- Herramienta de atención al cliente integrada. Herramientas para la gestión de devoluciones, creación de cuentas de clientes y modificación de pedidos. Sistema de gestión de clientes centralizado.
- M-Commerce: Permite la venta de todos los productos con un front-office y un back-office adaptados a todos los dispositivos.
- Permite mantenerse cercano a los clientes sea cual sea el lugar en el que se encuentran. La plantilla por defecto viene con un carrito de compra integrado adaptado para los dispositivos móviles, lo que permitirá la visualización correcta en todos los dispositivos, lo que hará que los clientes puedan navegar y comprar en cualquier momento y desde cualquier lugar.
- Proceso de pago adaptado al m-commerce. Aumenta las ventas facilitando el pago desde móviles y tablets de forma sencilla y segura.
- Gestión de la tienda desde cualquier lugar y sin restricciones de horarios. Permite una administración sencilla de toda la tienda desde un dispositivo móvil gracias a un back-office totalmente adaptado.
- Permite impulsar las ventas mediante dispositivos móviles con Shopgate. Se trata de una aplicación nativa que ayuda a sacar el máximo partido a la tienda.
- Marketing online: Incluye todo lo necesario para dirigir el tráfico, aumentar las ventas y fidelizar a los clientes.
- Situarse en los primeros resultados de los buscadores. Las funcionalidades avanzadas en el SEO, como personalización de meta títulos, meta descripciones y URL, permitirá que la tienda aparezca en los primeros resultados de búsqueda, captando la atención de los clientes.
- Utilización de los datos sobre el negocio para mejorar las ventas. Las herramientas avanzadas de análisis colocadas directamente en el back-office hacen que sea aún más sencillo entender lo que ocurre en tu negocio de comercio electrónico. Permitiendo tomar decisiones más inteligentes basándose en los datos precisos en tiempo real.
- Funcionalidades de marketing online totalmente equipadas.
- Permite la proposición de ofertas irresistibles al añadir productos a la cesta. Permite animar a los clientes a comprar más utilizando reglas dinámicas de fijación de precios,

como “compra 2 y llévate 1 gratis”. Fijación de precios según los grupos personalizados de los clientes, umbrales de descuentos de pedidos y mucho más.

- Ofrecer cupones y vales descuento. Crear códigos de descuento personalizados que permitan a los clientes ahorrar una cantidad de dinero o un porcentaje, o beneficiarse de gastos de envío gratis. Premiando la los clientes por hitos como sus primeros pedidos, cumpleaños y referidos.
- Animar a los clientes a que dejen su opinión. Los comentarios de otros compradores dan confianza al indeciso y mejoran el SEO con contenido único.
- Fidelización de clientes. Conseguir que los clientes vuelvan a la tienda, gracias a las campañas de fidelización y recompensas por recomendar a sus amigos, o permitiéndoles expresarse fácilmente en las redes sociales. Con esto, se consigue que los clientes queden satisfechos y hablen bien de la tienda.
- Aprovechamiento del correo electrónico como herramienta de marketing. Establece un canal de comunicación fluido con los clientes haciendo que vuelvan a la tienda, invitándoles a suscribirse al boletín informativo y a otros envíos automáticos. Generación de campañas de correo electrónico eficaces para ampliar rápidamente la base de clientes.
- Internacional: Abrir la tienda a consumidores de todo el mundo.
- Traducir a 65 idiomas. Es una tarea muy sencilla, simplemente es necesario seleccionar un idioma y traducirlo directamente en el back-office. Transformación de la tienda en un negocio global con una web totalmente localizada que permite vender desde y hacia cualquier lugar.
- Localización de divisas, impuestos y unidades de medida. Muestra las divisas, los impuestos y las unidades específicas de cada país a compradores de todo el mundo. Permite conocer la procedencia de los clientes gracias a las capacidades avanzadas de geolocalización, que perfeccionan la experiencia de compra internacional.
- Logística flexible basada en la localización. Permite la definición de reglas para los pedidos y los envíos internacionales; fija tarifas de envío, transportistas preferidos y almacenes específicos por zonas personalizadas.

2.3.2.2 Magento

Se trata de la plataforma de comercio electrónico líder a nivel mundial. Posee una versión open-source, Comunity Edition, y otra versión Enterprise, de pago.

Detrás tiene un buen soporte formado por desarrolladores y partnes que continuamente se encargan de generar contribuciones de código, creando nuevas extensiones y llevando a cabo participaciones en los foros y grupos de usuarios de Magento.

Buenas características: Incluye las características necesarias para crear, promocionar y gestionar una tienda atractiva y responsiva. Dedicada a un comercio global, con soporte para casi todos los idiomas, monedas y tasas locales.

Plataforma fácilmente extensible. Es fácil de configurar para conocer las necesidades del negocio. Permitiendo la selección de las extensiones necesarias de entre múltiples opciones,

en el Marketplace de Magento. Fácil de encontrar desarrolladores entrenados y dispuestos a llevar a cabo actualizaciones e integraciones.

Comunidad vibrante. Magento dispone de una comunidad de desarrolladores y comerciales que comparten nuevas ideas, se ayudan en los distintos problemas y colaboran para mejorar la plataforma.

Entrenamientos y recursos extensos. Documentación precisa disponible para ayudar a la iniciación. Multitud de cursos de entrenamiento ayudarán a aprender las bases o mejorar las competencias como desarrollador Magento.

Perfecto para negocios pequeños. Permite la creación de un sitio web con pasos sencillos y rápidos.

Dispone de multitud de extensiones. A continuación, se incluye una descripción de las más destacadas en función de la categoría a la que pertenecen:

Módulos destinados a la experiencia de usuario:

- Extensión de blog para Magento. Integración en multitienda, con capacidad para mostrarse solo en aquellas tiendas en las que se marquen en la consola. Posibilidad de duplicar posts entre diferentes blogs. Capacidad de edición de comentarios de los usuarios.
- Facebook connect and like: Dar la capacidad de conectarse y comprar haciendo login a través de la cuenta de Facebook, evitando los clásicos problemas de creación de una cuenta como ingresar un correo y contraseña, validarla... y acordarse de la contraseña. También incluye la capacidad de añadir el botón “Me gusta” en la página de producto. Gratuita.

Módulos destinados al marketing:

- Ultimate Advertising suite: Permite la ubicación de banners promocionales e informativos en diversas partes del proceso de compra de Magento. Permite la utilización de texto e imágenes en los banners. Además, permite monitorizar el número de impresiones y clics de los banners. Precio de 79$.
- Checkout Promo: Permite mostrar banners promocionales en el carrito de la compra y a lo largo del proceso de pedido. Se activan en función de las reglas promocionales previamente definidas. No se trata de promociones de cupones, sino promociones de venta cruzada y venta complementaria, al estilo “compra este producto y recibe un 20% de descuento” . Permite segmentar a los clientes, gracias a las ilimitadas reglas que pueden incluirse. Los banners se manejan mediante bloques, con lo que se pueden ubicar en el lugar que se desee.
- Loyalty Program: Programa de puntos con los que recompensar a los clientes con descuentos usando las reglas de negocio del carrito. Se pueden crear promociones en función del histórico de compras, crear packs de productos o packs promocionales, estimular a los visitantes a que se registren mediante regalo de puntos.... Tiene un coste de 129$.

- Multiple Coupons: Permite a los clientes canjear múltiples cupones de descuento para un pedido. Se trata de un módulo sencillo, tanto para el cliente final como para el gestor de Magento, permitiéndole lidiar con reglas como limitar el uso de los cupones en ciertos pedidos y también mostrar los cupones que ya han sido usados.
- ShareMe: Objetivo de promover los productos de una tienda online, así como las ofertas e incluso la propia tienda a través de las redes sociales, pudiendo beneficiarse así del efecto del marketing boca a boca. Permite ofrecer descuentos a cambio de Likes. El descuento es generado como un cupón único que luego se canjea en el carrito de la compra.

2.3.2.3 WooCommerce

Se trata de un plugin dedicado al comercio electrónico. Fue creado para ser integrado con Wordpress. Posee cientos de extensiones tantos gratuitas como de pago. Está siendo utilizada por el 30% de las tiendas, lo que le hace estar presente en mas tiendas que cualquier otra plataforma.

Destaca por ser una herramienta muy sencilla e intuitiva. Dispone de una buena documentación y soporte, donde destaca una sección de vídeo tutoriales.

En el apartado de las vistas, dispone de pocos temas gratuitos, pero la mayoría de ellos, están adaptados a los dispositivos móviles.

Posee un gran número de plugins. Destaca "WPML Multilingual CMS", se encarga de traducir todo el contenido y productos de la tienda a los idiomas que se establezcan.

Principales funcionalidades:

- Gestión de diferentes formas de pago: Contra reembolso, Paypal, transferencia bancaria, cheque, pasarela de pago con tarjeta.
- Configuración de los gastos de envío, que pueden ser gratis, de precio único o según peso, medida del paquete o país de envío.
- Configuración de los impuestos, puede estar incluido en el precio o calculado a parte, puedes imponer impuestos diferentes por producto.
- Permite crear cupones descuento, con una cantidad fija o un porcentaje, para productos concretos o categorías de productos.
- Permite añadir diferentes atributos y variaciones a las características del producto (color, tallas...). Variantes.
- Se pueden vender productos físicos o digitales.
- Ofrece informes de inventaria que informan del stock disponible de los productos en venta.
- Ofrece otros informes del estado de los pedidos, ventas y clientes.
- Incorpora los campos de SEO necesarios para el posicionamiento en buscadores.
- Poca inversión de tiempo y dinero.

Módulos destinados al marketing:

- Predictive Marketing for WooCommerce: Permite mostrar recomendaciones relevantes y personalizadas en cualquiera de las plantillas de la tienda. Permite además el envío de emails donde incluir cupones descuento, enviar recomendaciones... Todas las recomendaciones se basan en el comportamiento de los clientes. Gratuita.

Módulos de experiencia de usuario:

- Instant Search+: Se trata del módulo más rápido y avanzado, basado en la búsqueda en al nube. Permite a los clientes encontrar el producto que deseen de una forma rápida y eficaz. De pago.

2.3.3 Soluciones de Pago

2.3.3.1 Hybris

Se trata de una solución de comercio omnicanal facilitando la interacción con los clientes estén donde estén. Disponen de aplicaciones de comercio B2B y B2C que incluyen oferta de ominicanalidad, gestión de datos maestros (también llamada gestión de contenido de productos, o gestión de información de productos), gestión de pedidos, remarketing de SeeWhy y búsqueda y ofertas potentes.

2.3.3.1.1 Características

Se sitúa entre los líderes en software de comercio electrónico según las firmas de analistas independientes Forrester y Gartner.

Se basa en una tecnología y una arquitectura robustas, ágiles y abiertas, diseñadas para integrarse con sus sistemas existentes y ampliarse para satisfacer el aumento de la demanda.

Creación de un proyecto de comercio de forma rápida con hybris Commerce Accelerator. Las soluciones omnicanal están preconfiguradas para B2C y B2B, así como para regiones y sectores concretos.

Interacción con los clientes de forma contextual y coherente en todos los puntos de contacto gracias a la oferta omnicanal.

Ofrece experiencia multisitio, multirregional y multidivisa con las opciones de internacionalización incluidas en nuestro software de comercio.

[Hybris15]

2.3.3.1.2 Extensiones

Hybris dispone de multitud de extensiones que cubren todas las necesidades de la empresa que necesite con el software de comercio electrónico.

Commerce Accelerator:

Se trata de un framework Web listo para ser utilizado, a partir de él se da comienzo a la implementación sencilla de construir y mantener una solución omnicanal.

- Páginas de contenido estándar con integración WCMS para escritorio y móvil.
- Capacidad de búsqueda completa e integración con Apache Solr.
- Detalle de producto con indicadores de disponibilidad.
- Localizador de tiendas integrado con Google Maps.
- Análisis de clientes e integración con redes sociales.
- Carrito de compra persistente con múltiples opciones de checkout.
- Gestión de cuentas de usuario e historial de pedidos.

BaseCommerce extension:

- Framework básico para la compartición de items y tipos entre las distintas extensiones.
- Desacoplamiento modular.
- Modelo de datos
- Servicio de stock

Búsqueda y navegación:

- Mecanismo de búsquedas facetadas
- Integración con Solr.
- Indexación

Omni Commerce Connect:

- Call-Center
- iOs, Android...
- Spring MVC
- RESTful Commerce API

Addons:

- Permite extender el acelerador propio de Hybris sin necesidad de modificarlo.
- Extensión regular que provee de componentes adicionales frontend.

Web Content Management System (WCMS)

- Sistema software que provee de una página web, de escritura, colaboración y administración diseñada para permitir a los usuarios con poco conocimiento en programación, crear y administrar el contenido de una página con relativa facilidad.
- Administración de todo tipo de páginas.
- Contenido personalizable.
- Edición y previsualización en tiempo real.

Módulo BTG de personalización avanzada:

- Segmentación de clientes en diferentes grupos

- Utilización de la segmentación para la personalización del contenido.
- BTG (Behavioral Targeting Groups) el comportamiento de un cliente o las características del mismo en una tienda online se pueden utilizar para clasificarlo en diferentes grupos o segmentos. Estos segmentos se pueden utilizar para acciones de marketing.
- Reglas, acciones de salida,estadísticas.

Módulo Pricing:

- Factoría de precios.
- Cálculos de precios.
- Cálculo de impuestos.
- Cálculo de descuentos.

Cupones y promociones:

- Tipos de descuentos.
- Restricciones (precio de pedido, gasto del cliente, nuevo cliente, precio del pedido actual, por productos, por usuario, franja temporal).

Formas de pago:

- Creación de órdenes de suscripción y eliminación de las mismas.
- Órdenes de autorización (chequeo de tarjeta de crédito válida , autorización del precio de la compra).

Administración de pedidos:

- Flujo de pedidos
- Comprobación de pagos
- Detección de fraude
- Estrategias de división de pedidos.
- Historial de pedidos.
- Administración de almacenes.

2.3.3.2 Demandware

Provee de una plataforma Saas (Software as a Service) para modelos de negocio B2C.

Puntos fuertes:

- Reputación: Fuerte reputación entre minoristas y grandes marcas que buscan una plataforma de comercio electrónico que sea rápida, que provea de una interfaz de usuario rica y que minimice la versión en recursos financieros y técnicos.
- Estabilidad: En los estudios se pueden apreciar altos niveles de satisfacción con la escalabilidad de la plataforma, soporte para multisites y microsites, capacidad para crear y administrar sitios web y aplicaciones nativas, capacidad para localizar la

experiencia de usuario y front, aplicación de mejoras y actualizaciones con un impacto mínimo en la disponibilidad del sistema.

- Crecimiento: Fuerte crecimiento basado en su reputación con minoristas y marcas, escalabilidad y disponibilidad de su plataforma, fuertes capacidades de experiencia de usuario y la capacidad de soporte a desarrollos globales.

Puntos débiles:

- Rentabilidad: Para aquellas empresas que necesiten tanto capacidades B2C como B2B o que poseen complejos requerimientos en B2B, deben comparar las capacidades de Demandware con otros vendedores que tengan una amplia funcionalidad B2B.
- Ingresos de modelo compartidos: En algunos casos las licencias pueden tener un coste muy alto.

2.3.3.3 Infosys

Se trata de la plataforma líder en comercio electrónico. Provee de más de un model B2B, B2C, combina un conjunto de funcionalidades para guiar las ventas por medio de un amplio muestreo, catálogos y canales frontales para el conocimiento de los diferentes requisitos.

Contexto de negocio:

- Contiene componentes que engloban diseño, facturación, marketing y ventas por medio de componentes con alta tecnología.
- Líder en el sector, con una comunidad de usuarios global, está presenta en múltiples países.
- Aumento de ventas y reducción de costes.
- Estímulo de negocio:
- Plataforma de comercio electrónico robusta con soporte para actividades comerciales y de ventas que mejora las oportunidades de cross-selling.
- Permite investigaciones rigurosas, colecciones y ordenación de productos.

Contexto de cliente:

- Cambio en las estrategias de marketing.
- En los últimos tiempos el modelo de negocio ha cambiado. Aparecen un gran número de pequeñas y medianas empresas, vendedores independientes y distribuidores.
- Infosys crea un canal administrativo extensible en distintos canales y socios, permitiendo compartir información, flujos de pedidos, todo ello compartido.

Solución

Este proyecto permite crear una infraestructura web con apoyo de 'Worldwide Samples' (WWS) application. La aplicación existente tuvo que ser migrada del servidor de Microsoft Server 3.0 con SQL Server 2000 en la parte del back a un servidor IBM WebSphere Commerce 6.0 con Oracle 10g en la parte del back. Siendo la primera de las aplicaciones que la compañía tuvo la visión de construir en una plataforma de e-commerce mejorada.

- Ventas guiadas por medio de ejemplos extensos, que permite a los diseñadores encontrar soluciones que se adapten a sus necesidades.
- Storefronts que pueden satisfacer las distintas necesidades en términos de líneas de producto, canales, precios, monedas, idiomas, geografías y sectores.
- Permite a las fuerzas de ventas, distribuidores, revendedores y casas de diseño:
    - Acceso preciso a la información del producto.
    - Introducción de nuevos productos de forma rápida.
    - Obtención en tiempo real de precios y disponibilidad.
    - Realizar pedidos.
    - Conocer el estado del pedido y su localización.

[Infosys15]

2.4 Marketing Automation

2.4.1 Definición

"Se trata de una metodología que basa su funcionamiento en la utilización de software para automatizar todos los procesos derivados de una estrategia de marketing digital, como la segmentación, la generación de workflows, gestión de campañas, lead nuturing... ". **[Mañez_15]**

2.4.2 Propósito

Tiene como propósito facilitar a las empresas la priorización y ejecución de sus acciones de marketing de una forma más dinámica, ágil y eficiente, propiciando con ello el alcance de sus objetivos en menor tiempo.

Además, se trata de una herramienta que busca tanto adquirir nuevos clientes, como retenerlos. Tan importante es conseguir nuevos clientes como favorecer que éstos continúen consumiendo los servicios de la empresa.

Por consiguiente, busca incrementar las ventas de la empresa.

[Mañez_15][Diaz_15]

2.4.3 Rentabilidad

Durante los últimos años, diferentes estudios avalan la utilidad del Marketing Automation en el incremento de las ventas. De hecho, en ellos se pone de manifiesto cómo cada vez son más las empresas que lo utilizan, convirtiéndolo en un ingrediente indispensable para el éxito.

Los estudios más importantes llevados a cabo han sido los que se enumeran a continuación:

- Estudio realizado por Abeerden Group sobre el "Estado del Marketing Automation en 2014".
- Estudio llevado a cabo por Act-On and Gleanster Reserach, donde se pone de manifiesto que el 69% de las empresas B2B hacen uso del mismo como herramienta para la adquisición de nuevos clientes y el 50% lo hace para la retención de los mismos.
- El análisis de Forrester, indica que las empresas que han implementado esta solución con el objetivo de mejorar sus procesos, han incrementado sus ventas en un 10%.

Por otro lado, en el resumen del estudio el estudio sobre las estadísticas de uso de esta solución, llevado a cabo en diciembre de 2015, Jordie van Rijn pone de manifiesto en su página "emailmonday", de 2011 a 2014 once veces más de las empresas B2B utilizan marketing automation. Esto puede observarse en la gráfica "Incremento de uso Marketing Automation B2B". Además, la mayoría de los usuarios piensan que el gasto merece la pena, como se muestra en la gráfica "El coste del Marketing Automation merece la pena". **[Riaj_15]**

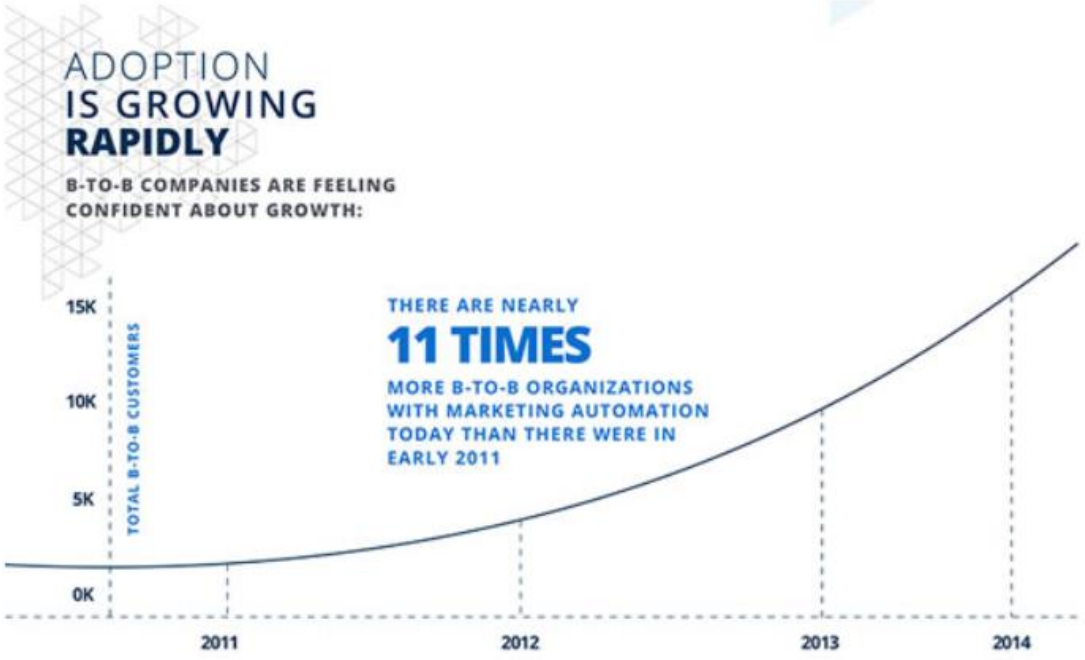


Ilustración 23: Adopción por Empresas B2B

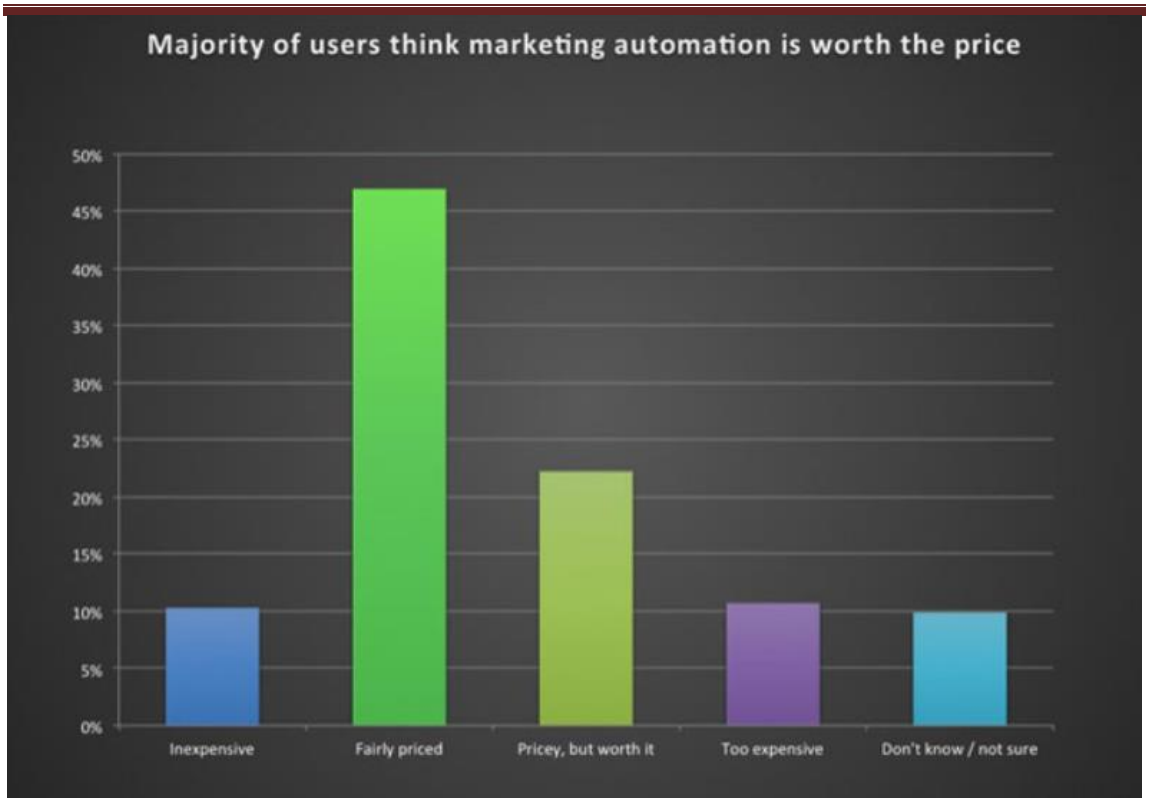


Ilustración 24: Opiniones sobre Precios

2.4.4 Beneficios

Se trata de una herramienta muy potente que genera excelentes resultados, tanto en pequeñas como en grandes empresas, en términos de ejecución de campañas, administración eficiente del tiempo, contacto con los clientes de una forma más personalizada... Acciones que manualmente serían muy difíciles de llevar a cabo con éxito.

Segmentación del público objetivo:

- Segmenta a los clientes en base a una serie de reglas, facilitando la creación de contenido interesante, en función del segmento de clientes.

Administración eficiente del tiempo:

- Se pueden crear diferentes campañas y programas de mensajes para una hora y fecha en el futuro. Por ejemplo, crear y programar todas las publicaciones de un mes completo en Facebook.

Mayor efectividad en el cultivo de leads:

- Ayudar a conectar con los clientes desde el principio de su interacción con los activos de tu compañía.
- Si un visitante se inscribe en tu newsletter, puedes conectarte con ellos de forma inmediata a través de un mensaje personalizado de bienvenida y confirmación de su suscripción, que llegará a la bandeja de entrada de su correo de forma automática.

Monitorización de tus acciones:

- Realizar seguimiento y monitorización de cada acción que estés llevando a cabo para cada segmentación de clientes, lo que a su vez permitirá obtener gráficos detallados, estadísticas y datos que posteriormente podrás analizar con el fin de optimizar las campañas para el futuro.

Optimización de recursos:

- Gestión eficaz del presupuesto. Reducción de necesidad de personal para recolectar información, crear y distribuir mensajes entre los contactos.
- Conversión en tiempo real:
- Capacidad de hablar el mismo lenguaje que el cliente, a través de hábitos de navegación: duración de la visita, tasa de rebote, porcentaje de clics...
- Creación de imagen de marca en múltiples canales:
- Canales como email, Facebook, Twitter...
- Potenciar la marca, hacer que sea más fácil aumentar la visibilidad del negocio y atraer a nuevos clientes.

[Mañez_15]

2.4.4.1 Áreas Optimizables

Sistematización de Campañas:

- Crear, automatizar y trackear todas las campañas y flujos de trabajo, integrando todos los canales de comunicación en una misma Plataforma de Marketing Automation. De esta forma, se podrá generar un mayor ROI concentrando el esfuerzo en la creatividad de la estrategia.
- Se pueden gestionar las Campañas Omnicanal y ejecutarlas desde diversos canales.
- Dentro de lo que se llama Lead Management, se puede convertir visitas en prospectos, luego en leads y finalmente en clientes reales mediante estrategias de Lead Generation y Nurturing.
- Monitorear el comportamiento y el desempeño de los clientes Ver cómo interactúan las visitas en la web, dónde hace clic un suscriptor en una Campaña de Email Marketing, en qué contenido realiza like o retweet en las redes, qué recursos se descarga y cuáles son las Call to Actions más inteligentes de los blogs.

Email Marketing:

- Involucrar a tus suscriptores en piezas segmentadas. Gracias a estas plataformas puedes construir fácilmente campañas automatizadas y crear relaciones personalizadas. Por ejemplo, puede enviarle un email con contenido diferente a quienes ya abrieron otra campaña enviada con anterioridad, o con un contenido específico a quienes rellenaron un formulario y revelaron un dato importante de su corporación...

- Permite enviar campañas mensuales, promociones semanales con descuentos, con vencimientos de facturas y pagos, darle la bienvenida a tus nuevos suscriptores.
- Entrega fácilmente los contenidos evitando su duplicidad. Mediante la tecnología SmartList, se pueden agregar o eliminar los contactos de forma automatizada siguiendo el comportamiento de los clientes, sus datos demográficos o sus acciones transaccionales a través del tiempo. Además, permite testear los contenidos, analizar su efectividad con reportes detallados e integrar Email Marketing con otros canales.

Social Media Marketing:

- El 70% de los consumidores confía en las recomendaciones de sus amigos. Procura contar con una plataforma que permita integrar todos los medios sociales de la empresa al sitio web, landing pages y campañas de email.
- Además permite construir un Social CRM en donde se integren.

Marketing Management:

- Automatización de un sistema para gestionar los recursos, tanto humanos como económicos. Según datos de Gartner, la automatización de esta planificación aporta ganancias sustanciales. De un 5 a un 20% de reducción en la inversión de Campañas con bajo rendimiento, hasta un 10% de ahorro en los presupuestos anuales y de un 60 a un 90% de reducción en programas no alineados a una estrategia global.

Reporte de Métricas y Estadísticas:

- Obtención de métricas bajo demanda, sin necesidad de innumerables cruces de datos.
- Permite analizar la efectividad de los programas y acciones para evaluar cuál tiene mejor retorno de la inversión (ROI), crear reporte AdHoc y tableros de control, dinámicos e intuitivos.
- Mediante un model CRM se podrán definir en detalle las fases del Ciclo de Compra por el que va atravesando el Lead y monitorear cómo se mueve a través del Funnel de Ventas, las fases de la venta.

2.5 Comercio Electrónico

2.5.1 Tendencias

Las tendencias más destacadas de los profesionales del eCommerce:

1. Usuario en el centro. La principal preocupación de los profesionales en 2015 es poner a sus usuarios en el dentro de todos los procesos
2. Mobiel First. El mobile commerce ha pasado a ser una realidad, y no basta con estar preparados para móviles, si no que hay que ser “mobile first”.
3. Omni-Channel. Los retailers tradicionales abren canal online y los pure players tiendas físicas, creando un futuro totalmente “Omni-Channel”.

4. Personalización. Generar experiencias de usuario únicas y ser capaces de ofrecer y de adaptar la oferta a cada demanda individual. La personalización importa.
5. Mayor competitividad. Y todo esto sin olvidarnos que el mercado es cada vez más complejo, más competitivo y más profesionalizado.

2.5.1.1 Usuario en el centro

El "rey" es el consumidor, y pasa a estar en el centro de todo. Empezamos a ser "consumer centric".

Como bien dice Carlos Andonegui (CEO de VinoPremier), "Las empresas sin los clientes no son nada, ni en online ni en offline, así que ¡pongamos al cliente en el centro de todo!. De acuerdo con Mireyea Masclans (Operations E-commerce Manager de Toys'R Us), "Esta mayor experiencia de los consumidores ha elevado sus expectativas y su exigencia ya es tan alta como baja en su fidelidad a una web tras una mala experiencia de compra".

Como aclara Daniel Vázquez (COO de eCommbits): "Cuanto mejor se conozca al cliente también se aumentará la distancia entre tu eCommerce y tu competencia".

Situar al cliente en el centro no es algo nuevo para otro tipo de negocios, de hecho, es algo más que se importa desde la venta tradicional a la venta online. Para Demis Torres (Sales Director de Rakuten Spain), "la clave será replicar en el mundo online el trato personal, la seguridad y la fiabilidad presente en la experiencia de compra offline de hoy en día".

Para conseguir poner al usuario en el centro, necesitamos adaptarnos a los hábitos de consumo de nuestros clientes, de ahí que las siguientes dos tendencias para 2015 sea el Mobile Commerce (porque nuestros clientes utilizan ya el móvil como un canal principal), y el Omni-Channel (porque los clientes quieren comprar cuándo, cómo y dónde ellos prefieran). Según Alex Vallbona (Managin Director Spain de Birchbox), debemos trabajar en una "experiencia e compra única, que debe ir más allá del contacto online y debe concebirse como un proceso único, fluido y holístico", conectando todos los canales y con especial énfasis en el canal mobile.

Todo esto nos lleva a que "empezaremos a oír cada vez mas conceptos como viaje del cliente, buyer persona, inbound marketing, lead nurturing y qualification, remarketing, automatización de marketing, analítica web y de cliente".

2.5.1.2 Mobile First

Ya no vale estar solamente 'adaptado' a mobile. Éste pasa a ser el primer canal.

Según Aquilino Peña, fundador de Kibo Ventures, uno de los fondos de capital riesgo más importantes de nuestro país, "el próximo año va a ser el año del M-Commerce", el despegue definitivo de una tendencia que hemos visto nacer y crecer a unos ritmos vertiginosos en los últimos 3 años. Tal y como apunta Mireya Masclans (Opertions E-Commerce Manager de Toys'R Us), "la penetración de los smartphones en España es abrumadora y las visitas a través de móviles representan ya el 30% de las visitas totales".

Aunque dentro de M-Commerce incluimos tanto smartphones como tablets, para Marta Esteve (CEO de Soysuper.com), "el tablet es importante, pero el móvil siempre está y es personal", por lo que la mayoría de los esfuerzos han de ir en esta línea.

Cuando hablamos de M-Commerce, también debemos fijarnos en cómo debemos adaptar algunos elementos básicos de nuestra estrategia de venta online, como es el caso de la retención de los clientes. Para Inés Ures (CMO de Groupalia), "el email marketing, hasta ahora el rey del marketing de retención, pasará a competir con el push móvil, donde la capacidad de segmentar y personalizar es tan potente o más que en el desktop".

Uno de los grandes retos del mobile commerce para2015 será conseguir mejorar la conversión, ya que tal y como apunta Patrick Dost (Sales Director de BrainSINS), "el smartphone convierte hasta 10 veces menos que un PC o portátil", por lo que en la actualidad gran parte del tráfico móvil no genera beneficios a las empresas de eCommerce.

Como los expertos incorporan capacidades de geolocalización, algunos expertos también vislumbran cómo pueden afectar estas tecnologías a los negocios desde distintos puntos de vista. Para Estefanía Lacarte (Head of Communications de Groupon), "tras la aparición de las plataformas de geolocalización, en Groupon ya se ha implementado esta tecnología", lo que permite a los usuarios ver los establecimientos que se encuentran más cerca de ellos. Para Javier Galán (Director General de 20milproductos.com), "los usuarios a través de sus smartphones, van a incrementar sus búsquedas locales con intención de compra en retail local", por lo que los negocios locales también deben ser capaces de adaptarse a la nueva realidad para ser visibles gracias a este tipo de tecnologías.

2.5.1.3 Omni-Channel

Nos dirijimos a un mundo donde las barreras del comercio son nulas.

"El ecommerce en 2015 dejará de ser ecommerce para ser omnichannel commerce", una frase directa de Nacho Somalo, uno de los mayores expertos de Comercio Electrónico de nuestro país, que deja más clara otra de las grandes tendencias para 2015. Según explica Francisco Carrero (CEO de BrainSINS), esta tendencia viene dada por "un entorno en el que los retailers se están convirtiendo en los principales actores del eCommerce y los pure-players están comenzando a hacerse experimentos con tiendas físicas".

Para dejar claro el concepto, Javier Echaleku (Director General de Kuombo), nos comenta que la omni-canalidad "no solo es estar presenta en varios canales de venta, si no que todos ellos estén conectados en una misma estrategia global".

Pero decirlo es más fácil que hacerlo, para Jorge Cano (E-Commerce Manager de Uno de 50), en relación a la omnicanalidad, "2015 va a ser un año de aprendizaje para todos (marcas, proveedores y clientes), y empezaremos a ver algunos logros en los más grandes del sector, que parece que ya empiezan a hacer sus primeros pinos, aunque en mi opinión queda mucho camino por recorrer". Pero los peligros de no adaptarse rápidamente son muchos, tal y como apunta Inés Ures (CMO de Groupalia): "sólo los players ingeniosos y que puedan competir en todos los canales y soportes serán ganadores".

Ser Omni-Channel implica ser capaces de alinear múltiples estrategias dentro de nuestra compañía, desde tener presencia en canales que generan tráfico y ventas adicionales, como apunta Noelia Ruiz (Project Manager de BrainSINS), al comentar que "los grandes del ecommerce tiene sus canales de venta pero también han de pensar en estar presentes en los marketplaces que distribuyen sus productos", a decisiones mucho más complejas y con mayores implicaciones en la empresa, como apunta Mariano Nava (Sales Manager de ASPA Consulting): "la mayoría de los retailers consideran esencial la capacidad de contar con un sistema amplio de visibilidad del inventario independientemente del canal desde el cuál se realice la búsqueda.

A día de hoy ya estamos viendo bastantes acercamientos al Omni-Channel, aunque todavía son pequeños pasos que a medio plazo si darán lugar a una estrategia Omni-Channel Global. Por ejemplo, Carlos Garijo (Account Manager de Electrolux para Mediamarkt), apunta que muchos retailers ya están empezando a "facilitar a los clientes de un establecimiento un acceso a través de tabletas a su Ecommerce, pero sobre todo, a un comprador online con otros productos". Según apunta Llorenc Palomas (Sales & Marketing Director de Trelogi), "Munich Sports, por ejemplo, ya ha incorporado la opción de compra online y recoger el producto en sus tiendas físicas".

Y a futuro veremos mucho más, como explica Iván Márquez (Business Development en Geomobile) al explicarnos que "a través de una señal Wi-Fi o del Bluetooth Low Energy de los smartphones de los consumidores que hayan expresado su voluntad de recibir notificaciones, tenemos la oportunidad de realizar estrategias eficaces de Proximity Marketing", lo que nos permite conectar entornos digital con entornos físicos abriendo un campo infinito de oportunidades.

2.5.1.4 Personalización

Si el usuario está en el centro, debemos ofrecerle experiencias relevantes.

Para Francisco Carrero (CEO de BrainSINS), "2015 es el año de personalización multicanal", ya que además de adaptarnos a un entorno donde debemos mantener una coherencia entre los múltiples canales de venta de los retailers, debemos ser capaces de hacer que "el cliente sienta que es único y que hablamos exclusivamente con él", tal y como apunta Mireya Masclans (Operation E-Commerce Manager de Toys'R Us).

En definitiva, tal y como explica Jordi Ordoñez (consultor E-Commerce), "a la gente nunca le ha gustado que le sugieras cosas que le importan un pepino. Cuando más les conozcas, más les venderás. Y si no, ya les venderá otro". De esta forma la personalización se convierte en uno de los conductores que ayudará a aumentar las ventas a los retailers. En palabras de Iván Gil (CEO de Posizionate), "mediante la personalización lograremos una experiencia de usuario atractiva que ayudará a generar más ventas".

Para Gosia Pajkowska de Vente Privee, debemos "conocer al cliente para responder a sus necesidades, acompañarle durante el proceso de la compra", y "conseguir que el producto esté ajustado a los gustos y necesidades de los clientes", algo que nos matiza Estela Gil (Marketing & SEO en Posizionate).

Para muchos otros esta tendencia también queda muy clara, Marta Esteve (CEO de Soysuper.com), "veremos también más ofertas personalizadas y precios más cambiantes, ofreciendo así buenas oportunidades". Esto tiene que estar ligado a la experiencia de usuario, tal y como matiza Luz De-León (Fundadora de Diga33 y experta en UX): "cada uno de nuestros compradores sabe que es una oportunidad de venta y se valora como tal, esperan ser tratados de una forma especial".

Para Yolanda Lozano (Country Manager Spain de Webtrekk), la personalización también plantea importantes retos, como el de "diferenciar usuarios de personas es totalmente clave para el 2015, será el reto para muchos de los players de intranet". Pero superado estos retos, los beneficios son muy importantes. Alex Vallbona (Managing Director Spain de Brichbox) explica con detalle que "las empresas buscarán ser relevantes para cada uno de sus clientes, creando experiencias a medida a través de contenidos, productos y comunicaciones personalizadas para lograr mayor recurrencia en las compras".

Los más visionarios como Philippe Lardy (CTO de Ydral y CEO de OpenExpo), van un paso más allá y comentan que en breve veremos técnicas como el "pretargeting dirigiéndose a potenciales clientes, proponiéndoles productos o servicios adaptados antes de que tengan conciencia de lo que necesitan".

2.5.1.5 Competitividad

Todos a vender online, y competimos con todo el mundo.

"El Ecommerce en 2015 ya no es un juego de niños", deja claro José Carlos Cortizo (CMO de BrainSINS). Estamos en un mercado que claramente está creciendo, de hecho según Jorge González Marcos (Senior Manager Loyalty de Philips), "sobrepasaremos los 20 millones de compradores de eCommerce España en 2015", y según Jorge Sorial (Director de Desarrollo de Negocio en SagePay), en 2015 veremos "la incorporación masiva de la PYME española al comercio electrónico".

En palabras de Roberto Palencia (General Manager de Foro de Economía Digital), "la mejora de la capacidad económica general hará que aumente el volumen de negocio y sobre todo aumenten el número de operaciones lo que permitirá profesionalizar más los procesos y aumentar las diferencias entre el negocio impulsado por profesionales bien formados y los intentos voluntariosos pero lamentablemente con poco recorrido".

Y para ser competitivos debemos de empezar a "afinar el tiro", ya que según apunta Txampa (CTO de Santafixie.com) no es sostenible en el tiempo para la mayoría de ecommerce tener un CAC más alto que la rentabilidad que le sacan a cada cliente". Con un argumento relativamente similar. Oscar Reales (Consultor E-Commerce), profundiza todavía más sobre el tema "el número de usuarios que compra dos o más veces en un mismo sitio es insuficiente. Y sin esta fidelización, todos los costes desequilibran mucho la balanza de resultados".

Para José Rincón (ATM's Department Director en Bankia), "las empresas del sector van a tener que diferenciarse creando valor más allá de la propia transacción". Marta Panera, Internacional PR Director de Showroomprive.com también considera que las empresas tienen

que adaptarse a la nueva realidad, ya que "los que no, seguirán en la disputa de abrir o no los domingos, mientras sus ciudadanos compran desde su salón en un pure player".

2.5.2 Miedos y preocupaciones

Áreas que necesitan desarrollarse para poder alcanzar las tendencias eCommerce 2015.

1. Pagos. Los profesionales del eCommerce están esperando a comprender el impacto de los nuevos métodos de pago (incluidos móviles).
2. Fidelización. Para hacer crecer los negocios y para reducir costes, la fidelización es un aspecto crítico que puede dejar a más de uno fuera de juego.
3. Big Data. Tener las capacidades para analizar datos resulta crítico para optimizar los negocios, y más con grandes volúmenes de datos.
4. Logística. One-day delivery, envíos en menos de 12 horas, puntos de conveniencia... La logística juega un papel clave en la experiencia.
5. Contenidos. Fotografías de mayor tamaño e interactivas, vídeos, opiniones... Hay que dar al usuario la información que necesita para la compra.

2.5.2.1 Pagos

El impacto de los pagos móviles y nuevas formas de pago eCommerce.

"2015 será el fin del plástico", una predicción de Ignacio Riesco (CEO de interativ4), y que define muy bien todo lo que está pasando en el mercado de los pagos móviles. Para Elena Álvarez (Business Development Leader en Popular Payments), los "mobile payments redondean la propuesta disrupta del mobile, con un mercado que ya alcanza los 325.000 millones de dólares".

El tema de los pagos va más allá de los pagos móviles. Para Jordi Rosell (Consultor E-Commerce), 2015 será el año en que "los comercios electrónicos se van a cansar de las pasarelas virtuales y van a priorizar métodos de pago mas fáciles de usar y con mejor fiabilidad". Además, debemos tener en cuenta otros movimientos colaterales. Tal y como apunta Leticia Collado (Marketing Manager Spain de Lyoness), "el 2015 será el año decisivo para la moneda virtual: ¿ganarán en uso y aceptación, sobrevivirán residualmente o se irán desvaneciendo poco a poco sucumbiendo a las presiones legislativas y preocupaciones por su seguridad?".

2.5.2.2 Fidelización

Después de poner el foco durante años en captación y conversión, llega la fidelización.

Los eCommerce se centrarán en generar "contenidos, productos y comunicaciones personalizadas para lograr mayor recurrencia en las compras", tal y como apunta Alex Vallbona (Managin Director Spain de Birchbox).

La necesidad de recurrencia de compra es clara, ya que tal y como nos explica Txampa (CTO de Santafixie.com), “no es sostenible en el tiempo para la mayoría de ecommerce tener un CAC más alto que la rentabilidad que le sacan a cada cliente”, por lo que aumentar la recurrencia es el camino a la rentabilidad.

Oscar Reales (Consultor E-Commerce), va un paso más allá explicando que “los eCommerces basados en la recurrencia tienen una idea clara: 'vender (muchas) más veces al mismo usuario'. La recurrencia, en su más pura vertiente, se basa en las suscripciones a un servicio o producto que es consumido periódicamente, si bien con un poco de imaginación, se pueden portar al modelo recurrente muchos negocios actuales, o al menos hacerlo de forma mixta”.

2.5.2.3 Big Data

Conocer a los usuarios requiere analizar grandes cantidades de datos.

Para Elena Gómez del Pozuelo (Presidenta de aDigital), “las empresas de ecommerce debemos estar obsesionadas por el análisis de datos de comercio y centrarnos mucho más en la conversión que en atraer más visitas a la web”. Ahora bien, cada vez “tenemos más información que nunca y hay que utilizarla para segmentar y personalizar al máximo la oferta”, tal y como apunta Mireya Masclans (Operations E-Commerce Manager de Toys'R Us). Pero esto supone un reto tecnológico que no está al alcance de la mayoría.

Oscar Gutierrez (Digital Manager en Lekue), apunta que “cada vez será mayor el uso de inteligencia y big data para elaborar acciones de marketing cada vez más personalizadas y medir sus resultados con mejoras continuas”.

De esta forma el impacto de Big Data se produce sobre toda la estrategia de la empresa, tal y como explica con gran claridad Xavi Comella (IT & eCommerce Manager en Textura): “la estrategia deberá basarse en cómo desarrollar y aprovechar al máximo lo que el BIG DATA nos ofrece hoy en día para poder anticiparnos a las necesidades de nuestros clientes, ofreciéndoles lo que necesitan en cada momento determinado, sin que estos se sientan 'perseguidos'”.

2.5.2.4 Logística

Los envíos son parte fundamental de la experiencia de compra.

Juan Sandes (CEO de Celeritas) lo tiene muy claro, “debemos hacer confluir las expectativas del comprador entre los e-merchant y los logísticos mediante servicios de valor añadido como concertación de la entrega, algo que nos llevará a mejorar la experiencia de compra de usuario y generar más ventas para todos”. En esta línea, Leticia Collado (Marketing Manager Spain de Lyoness), apunta que “en 2015 asistiremos al incremento y desarrollo de las operaciones 1-day delivery, same day-delivery, incuso 1-hour delivery”, algo que ya es habitual en otros países.

Victor Juarez (CEO de Mi Tienda de Arte), plantea ver clara la necesidad en este aspecto “esperemos que en 2015 los players de la logística abandonen la guerra de precios para lanzarse a la claridad, compromiso y rapidez que tiendas y clientes venimos demandando, adaptándose así a las necesidades y demandas de un canal en continuo cambio.

2.5.2.5 Contenidos

Para ser capaces de conectar con nuestros usuarios necesitamos todo tipo de contenidos.

Los contenidos se han convertido en un papel clave en casi cualquier estrategia de marketing, y eso en un entorno donde los "reyes" de los contenidos, los medios tradicionales, están viviendo la pero de sus épocas.

Para Gerad Rius (Canal Online de Abacus), "el crecimiento relevante del ecommerce en 2015 vendrá de la capacidad que tengan las marcas de ofrecen contenidos relevantes, originales y de calidad, en diferentes formatos, vinculados a la información de cada uno de los productos que publiciten en su tienda".

Para Montse Labiaga (Directora de Fotografía eCommerce), dentro de todos los contenidos, los visuales y multimedia serán los dominantes. "En 2015 veremos que los eCommerce apuestan cada vez más por fotografías más grandes poniendo sus productos en valor". Además, también viviremos la explosión de los contenidos visuales interactivos, como las fotografías 360º que permiten recrear a los usuarios la sensación de tener los productos en sus manos".

2.5.3 Deseos y tendencias medio plazo

1. Nichos. Todavía quedan oportunidades de negocio en nichos muy especializados, así como de tendencia global.
2. Marketplaces verticales. Los marketplaces generalistas han demostrado su eficacia, pero han de pasar el testigo a marketplaces más especializados.
3. Valor añadido. Con un mercado mucho más duro, competir solo en precio es imposible. Hay que ser capaces de aportar valor añadido.
4. Ecommerce alimentación. Quizás en 2015 el eCommerce de alimentación no acabe de despegar, pero se seguirán dando importantes pasos.
5. Ecommerce emocional. Para conectar con nuestros usuarios hemos de ser capaces de generar emociones, el branded content es una gran herramienta.

2.5.3.1 Nichos

Para David Carro (Digital Capabilities Manager en Vodafone), "la mayoría de los nichos verticales donde el eCommerce representa oportunidades de negocio están siendo cubiertos (retaling, category killers, servicios, etc.)", pero como bien apunta Javier Alonso (e-Commerce Senior Advisor en Ricoh), "el foco en sectores y nichos concretos será una tendencia natural para los que no puedan pelear con gigantes".

Jordi Ordoñez (Consultor E-Commerce), también apunta que los nichos son un campo muy interesante, ya que según él, "los nichos van a funcionar muy bien. Tiendas especializadas, focalizadas en un producto y un target específico. La menor competencia y la barrera de entrada de la especialización junto al trato diferencial que le puedes dar a un cliente que busca algo en lo que eres experto son victoria segura".

En los últimos años hemos visto despegar nichos bastante concretos como Promofarma.com dentro del nicho de las parafarmacias, o dentro del vertical de deportes algunas tiendas de nicho con gran crecimiento como Santafixie.com, Padelmania.com, etc. Todavía quedan nichos por cubrir que están empezando a ser descubiertos, como latiendadelapicultor.com, pero debemos ser rápidos en encontrar estos nichos e internacionalizarlos rápidamente para lograr cuota de mercado.

2.5.3.2 Marketplaces verticales

Marketplaces que explotan nichos o conjuntos de nichos.

Para Mac Vicente (CEO de Rakuten Spain), "gracias a la aparición y crecimiento de los marketplaces, los pequeños y medianos comercios tendrán oportunidades reales para digitalizarse". Ahora bien, los actuales players en el mercado de los marketplaces son generalistas tratando de cubrir todos o casi todos los verticales del mercado (Amazon Rakuten, eBay, etc).

David Masó (Co-fundador de Promofarma.com), deja clara su opinión al respecto: "los marketplaces verticales van a ganar protagonismo y contribuir en el crecimiento a nuevas categorías", ya que "la experiencia de usuario está enfocada a cubrir necesidades muy concretas a los usuarios en un vertical de negocio".

Y en los últimos años ya hemos visto empezar a surgir marketplaces verticales, como PromoFarma.com, Uvinum.com, Modalia.com, etc... Seguro que durante 2015 y los años venideros seguiremos viendo crecer esa tendencia.

2.5.3.3 Valor añadido

Juan Sandes (CEO de Celeritas) apunta que 2015 es "el año de la consolidación de servicios de valor añadido", haciendo especial énfasis en la logística, pero esta tendencia también aplica a otras áreas del negocio de venta online. Por ejemplo, Iván Navas (CEO de Doofinder), explica que "entiendo que el factor clave es la generación de valor añadido en la tienda, a través, entre otras cosas, de la atención al cliente y la proactividad comercial".

Alex Vallbona (Managin Director Spain de Birchbox) también refuerza este punto al comentar que hay que "convertir a esos clientes en fieles fans, que valoren otras propuestas más personalizadas y sofisticadas, mas allá del precio del producto o servicio".

Para José Rincón (ATM's Department Director de Bankia), esto también resulta crucial, como explica al decir que "las empresas del sector van a tener que diferenciarse creando valor más allá de la propia transacción".

Según Juan Luis Rico (CEO de la LetsBonus), "el consumidor será más exigente y será fundamental seguir innovando en la propuesta de valor".

2.5.3.4 Ecommerce de alimentación

Para Diego Sebastián de Erice (Director eCommerce de DIA), 2015 es el año en el que "los eCommerce de alimentación se empiezan a quitar el chándal", es decir, "aunque todavía son pocos los consumidores que habitualmente realizan su compra online, cada vez más personas se animan a probar la compra online" dentro de este vertical.

Para Juan Miguel de Sande (eCommerce Manager de Quesos de la Huz), "grandes marketpalces están invirtiendo en potenciar el sector gourmet, y algunos fabricantes de alimentación y bebidas están vendiendo directamente online. Todo apunto a que está muy cerca la explosión del ecommerce español en este sector".

Según Demis Torres (Sales Director de Rakuten Spain), "sectores como la alimentación y el hogar se beneficiarán de la humanización del ecommerce español, como resultado de la importancia para comprar el mejor vino, elegir la mejor cesta de fruta de la temporada o el mueble que se adapte a su comedor del cliente".

2.5.3.5 Ecommerce emocional

Estela Gil (Marketing & SEO en Posizionate) apunta que "el marketing emocional seguirá siendo el tipo de publicidad o promoción más popular, acercar la marca o producto al usuario, apelando a sus emociones y gustos".

Siguiendo esta línea, Carmen Santamaría (E-commerce Manager de Loewe), deja clara su opinión y apuesta pro el branded content con su comentario: "todo lo anterior, viene precedido por un enamoramiento, por inspirar al consumidor contando nuestra historia, nuestras historias. Será el año del Branded Content, de la creación de contenidos de interés, de hacer que el cliente se sienta identificado con nuestras historias y sea partícipe de las mismas, por que no estará comprando un producto por Internet, estará navegando, apasionándose y adquiriendo un sueño.".

Para Tomeu Ozonas (CEO de Proretoque), "la foto cada vez más huirá de la presentación neutra del producto y poco a poco irá acompañándose de formas de fotografías que transmitan emoción".

[Brainsins_slide_15]

2.6 Soluciones Marketing Automation

2.6.1 SAP Hybris Marketing

2.6.1.1 Descripción

La solución de comercio electrónico Hybris, también incorpora un módulo dedicado al marketing, que se nutre de los datos del propio site como son productos, clientes y pedidos.

Se encarga de ejecutar triggers para la recolección de datos que se obtienen del navegador como son los clics, el registro de la actividad del usuario en el site, su comportamiento...

Gracias a esta información, por cada cliente, sabremos las tareas que está llevando a cabo en tiempo real en el site, lo que debería hacer, y lo que ha realizado con anterioridad. Hybris permite en tiempo real, presentarle al usuario el contenido relevante para él, que se le mostrará por medio de la personalización del site, de apps, de anuncios y de email.

Hybris provee además de una plataforma abierta, donde entran en juego importantes partners como Facebook, Google Analytics, IBM, skrux, Marketo OpenText, sprinklr, turn y bloomreach.

[Hybris marketing submit transparencias]

2.6.1.2 Finalidad

El Marketing Automation de Hybris busca conocer a sus clientes, saber lo que desean comprar y poderles ofrecer, al igual que ocurren en las tiendas físicas, los productos que mejor se adaptan a sus necesidades.

El Marketing Contextual combina tres tipos de información de cliente.

- Past interactions: Historial de transacciones, incluyendo pedidos, compras, pagos, etc.
- Comportamientos anticipados: Uso de la analítica predictiva de los marcadores de tendencias.
- Mostrar motivaciones e incentivos. En el contexto actual.

Con estos conocimientos del cliente, podemos individualizar las experiencias del cliente. Ofrecer promociones... cuyos resultados serán capturados en el perfil del usuario para mejorar la próxima ocasión y mostrar los productos más relevantes.

2.6.1.3 Productos de Hybris Marketing

El hybris marketing consiste en nueve productos. Permiten, tanto conocer lo que ha hecho el usuario con anterioridad, como conocer sus acciones en tiempo real.

Ilustración 25: Características Hybris

A continuación, se indica la descripción de los productos, agrupados en base a sus funciones.

2.6.1.3.1 Conocimiento del usuario

Hybris Marketing Data Management. Información de todas las interacciones que ha llevado a cabo el usuario en las diferentes fuentes, con el objetivo de trackear el contexto, los intereses y así llevar a cabo predicciones en tiempo real. Posee analíticas y herramientas potentes que permiten analizar y descubrir patrones de comportamientos de todos los usuarios , así como visualizar el viaje del usuario en el site.

Hybris Marketing Segmentation. Crear grupos de usuarios y segmentos. Descubrimiento de tendencias ocultas e identificación de micro-segmentos. Incluye una rica extensión de operaciones específicas en las interfaces de usuario, que permite la creación de árboles de segmentación a gran velocidad.

Hybris Marketing Recommendation. Se encarga de mostrar recomendaciones inteligentes en tiempo real. Generar algoritmos predictivos en tiempo real para desarrollar contextos relevantes, recomendaciones personalizadas basadas en el comportamiento del usuario. Se beneficia de los modelos de aprendizaje propios, para optimizar las recomendaciones y así ofrecer las mejores e incrementar el tamaño de los pedidos.

2.6.1.3.2 Crear experiencias de usuario contextuales relevantes

Hybris Marketing Convert. Aprovechar la navegación del usuario para conocer sus comportamientos y su intención de compra para así reorientarlos y hacerles sencilla la compra. Mostrar anuncios gráficos orientados en base a los contenidos vistos recientemente, o enviar emails recordatorios para un cliente que ha abandonado su carrito de compra, haciendo que regresen y completen la compra con un solo clic.

Hybris Marketing Acquisition. Planificar y ejecutar campañas personalizadas, por medio de emails, SMS/texto y otros medios sociales. Realizar campañas teniendo como base las redes sociales, como Facebook. Recibiendo análisis detallados de la efectividad de las campañas.

Hybris Marketing Loyalty. Su función es la de crear programas de fidelidad multicanal que transforma puntos y recompensas en la verdadera lealtad del cliente. Mejora las tasas de retención del cliente aumentando su valor. Fomenta la promoción de la marca y hace referencia a ella por medio de los medios sociales.

2.6.1.3.3 Mercado ágil y rápido

Hybris Marketing Planning. Alinear a los equipos de marketing con planes transparentes. Administrar el presupuesto completo; obtener conocimiento de las ganancias y de los gastos en tiempo real. Facilitar la administración de las campañas, todas las campañas de marketing podrán ser visualizadas de un solo vistazo en el calendario. Llevar un registro de las métricas de rendimiento.

Hybris Marketing Orchestration. Dirigir el engagment con los clientes a través de múltiples canales. Plan de engagment con el cliente, basado en canales, ofreciendo contenido relevante

del cliente tanto entrante como saliente. Optimizar las estrategias de contacto a través de canales y contenidos de compromiso con el cliente.

Hybris Marketing Insights. Ganancia de la comprensión en tiempo real. Las actuaciones de marketing se gestionan por medio de un tablero que muestra las KPI del marketing a través de su organización. Identificación rápida de las oportunidades y profundización para descubrir información clave sobre segmentos, productos, canales, presupuestos, campañas y activos.

2.6.1.4 Aportaciones

El marketing incluido te ayudará en :

- Desarrollar una vista de cliente enriquecida para cada cliente individual y administrar clientes y prospectos conocidos y desconocidos.
- Aumentar en engagement de los clientes en tiempo real con las intenciones y motivaciones del usuario.
- Aumentar la dirección de las conversiones del cliente con contenido contextual relevante, haciendo hincapié en las ofertas.
- Reaccionar rápidamente ante las oportunidades de venta, siendo más efectivo. Habilitar las acciones de marketing en tiempo real incrementan la transparencia y la colaboración interna.

[Solution Brief_hybris_Marketing_en]

2.6.2 Webtrekk

2.6.2.1 Descripción

Se trata de una suite de inteligencia digital. Se encarga de recoger, gestionar y utilizar datos para crear una aproximación user-centric al negocio y a la estrategia de marketing. **[Webtrekk]**

2.6.2.2 Funciones

Esta solución se encarga de llevar a cabo las siguientes funciones:

2.6.2.2.1 Análisis del visitante.

- Tiene en cuenta la primera página que ve el visitante y los elementos trackeados de la impresión que se lleva.
- Analiza el origen del cliente a la página. Si proviene de un buscador, de otra página web , o ha entrado directamente en el site.
- Analiza si el tráfico ha venido de una campaña de marketing.
- Mide las páginas o secciones que han sido el punto de acceso de los clientes.
- Conocer la página que ha hecho abandonar a los clientes el site.

2.6.2.2.2 Análisis de las visitas

- Páginas que han sido más vistas como página principal
- Conocer qué links de la página han sido utilizados.
- Qué áreas de la página han sido más utilizadas.
- Niveles de engagment de los usuarios con el site.

2.6.2.2.3 Métricas

- Los analizadores pueden incluir métricas y fórmulas
- Rangos de conversión. Porcentaje de visitantes que han llevado a cabo por ejemplo un pedido.
- Rango de páginas vistas por un cliente por sesión iniciada.
- Porcentaje de páginas que se han convertido en las últimas vistas por los clientes.
- Media de la duración de los visitantes.
- Visitantes que han vuelto a la página.
- Rango de nuevos visitantes.

2.6.2.3 Análisis

2.6.2.3.1 Análisis del Marketing

Conocer si el acceso a la web fue por medio de un motor de búsqueda, un medio social u otro tipo de referencia. O por el contrario, el usuario accedió de forma directa.

Conocer si el acceso a la pagina fue producido a través de una campaña de marketing o no. Se encarga de generar un árbol que muestra una vista resumida de las búsquedas que hicieron que el usuario llegase a la página.

En las búsquedas del propio sitio conocer las marcas más buscadas por los usuarios, así como las palabras y oraciones.

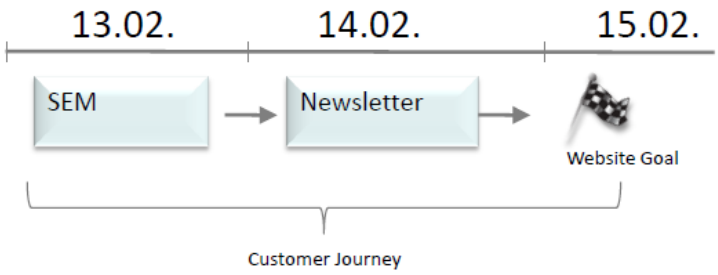
2.6.2.3.2 Análisis de Campañas

Webtrekk Q3 ofrece un análisis de las características de las campañas, lo que te permite evaluar tus actividades de marketing online.

Permite conocer los clics efectuados en las campañas, el tiempo de inactividad de los visitantes, medidas de los visitantes.

Duración media de las visitas, media de las páginas vistas.

Conocer el viaje del cliente en nuestra página. Eventos que han hecho que el visitante consiguiera su meta. Un viaje de cliente por definición siempre incluye la realización de una meta al final del viaje de cliente.



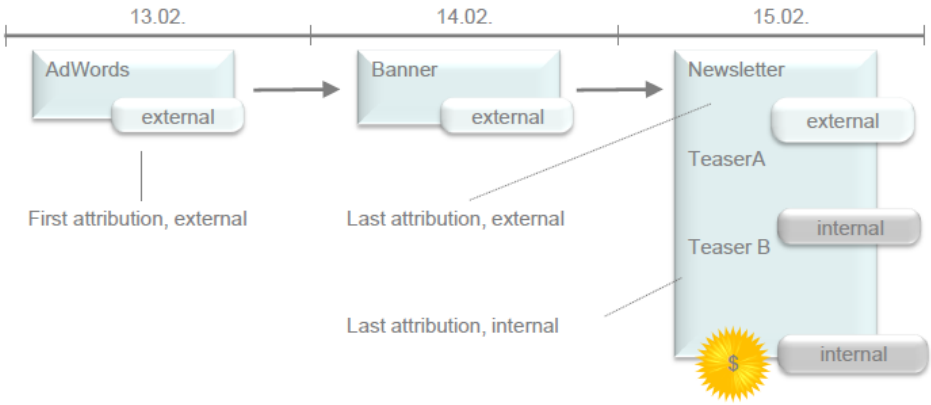
The Customer Journey enables a post-click evaluation of campaigns based on the website goals, as well as attribution rules such as 'last ad media wins'.

Ilustración 26: Viaje de Cliente

El viaje de cliente comienza con el primer contacto con la campaña y termina cuando se consiguen realizar una o más metas del site.

Example:

Customer Journey to the website target 'Order'.



Note: The Customer Journey represents bundled information for Webtrekk. The relationship between all items of information in the customer journey therefore relates to the search scope 'Action'.

Ilustración 27: Ejemplo de viaje de cliente en Pedido

El viaje de cliente se guarda cuando se ha llevado a cabo todo el proceso.

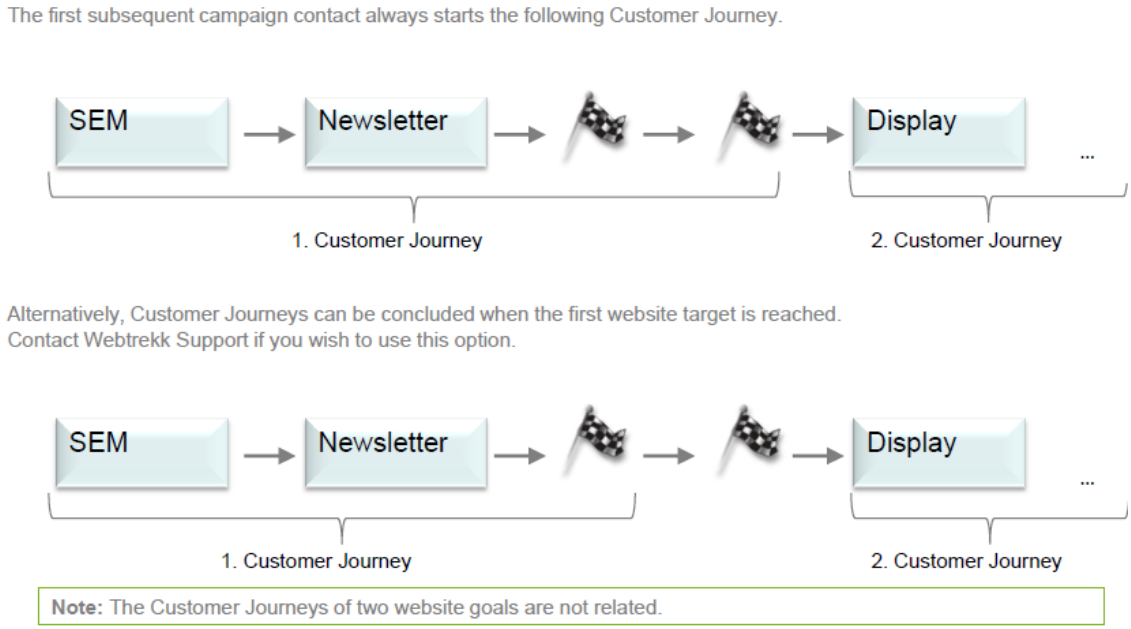


Ilustración 28: Especificación del viaje de cliente

2.6.2.3.3 Análisis de Rutas de página, , Procesos y Profundización en el compromiso

Rutas de Página

Las rutas de página indican el comportamiento del usuario en la navegación. Se observan todas las páginas a las que ha accedido.

En el análisis se conocen, la página precedida, la segunda página precedida, la siguiente página, la segunda página precedida. El orden gráfico de las páginas.

Processes

En los procesos se puede analizar una serie definida de páginas. Por ejemplo, el proceso de checkout, alta en suscripción, proceso de registro.

Profundización en el compromiso

Lleva un seguimiento del nivel de los objetivos que se alcanzan durante una visita y habilita todos los objetos y métricas para ser analizados de acuerdo a los niveles de los objetivos.

Existe una fórmula que automáticamente crea un ratio de conversión que muestra la relación del mejor grupo de todas las visitas.

2.6.2.3.4 Superposición y Mapas de Calor

Superposición

Con la superposición se puede mostrar el uso de los enlaces y formularios.

Mapas de calor

El mapa de calor muestra cada clic en la web sin importar si hace clic en un vínculo o no. Cada clic es visible. La frecuencia de uso se muestra con colores, siendo el rojo el más frecuente y al color azul indicador de menor frecuencia.

2.6.3 BrainSINS

2.6.3.1 Descripción

Se trata de una solución de personalización todo-en-uno para minoristas.

BrainSINS se trata de una solución en la nube que utiliza Big Data y aprendizaje adaptativo para ofrecer comercio digital personalizado y gamificación.

2.6.3.1.1 Detalle

Se puede integrar y empezar a utilizarse en 5 minutos.

Ayuda a incrementar las ventas en una media del 20%.

Permite a los SME competir a nivel empresarial.

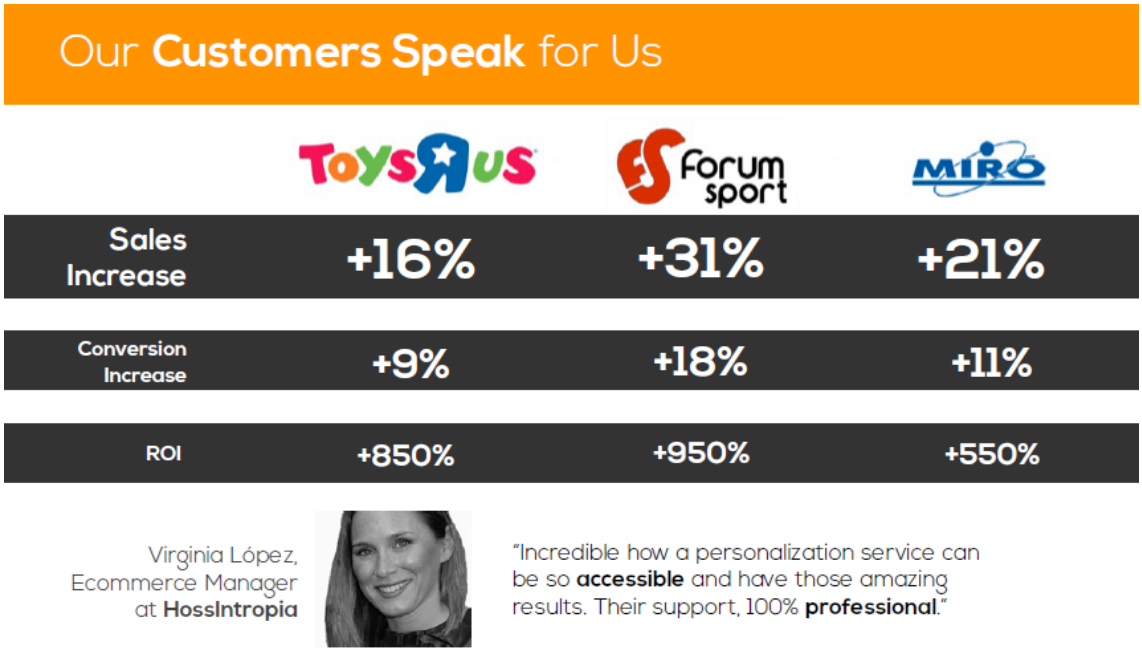


Ilustración 29: Opiniones de clientes

2.6.3.2 Funciones

Ofrece a los clientes lo que necesitan en el momento adecuado.

- En la tienda, ofrece recomendadores personalizados, así como landing pages dinámicas.

- En las compras, Cross-selling, metas de comportamiento, recuperación de carritos abandonados.
- En el engage (envolver). Newsletters personalizadas y gamificación.
- Para hacer estas operaciones, se nutre del perfil del usuario y de su comportamiento,de la información contextual, y del catálogo de productos.

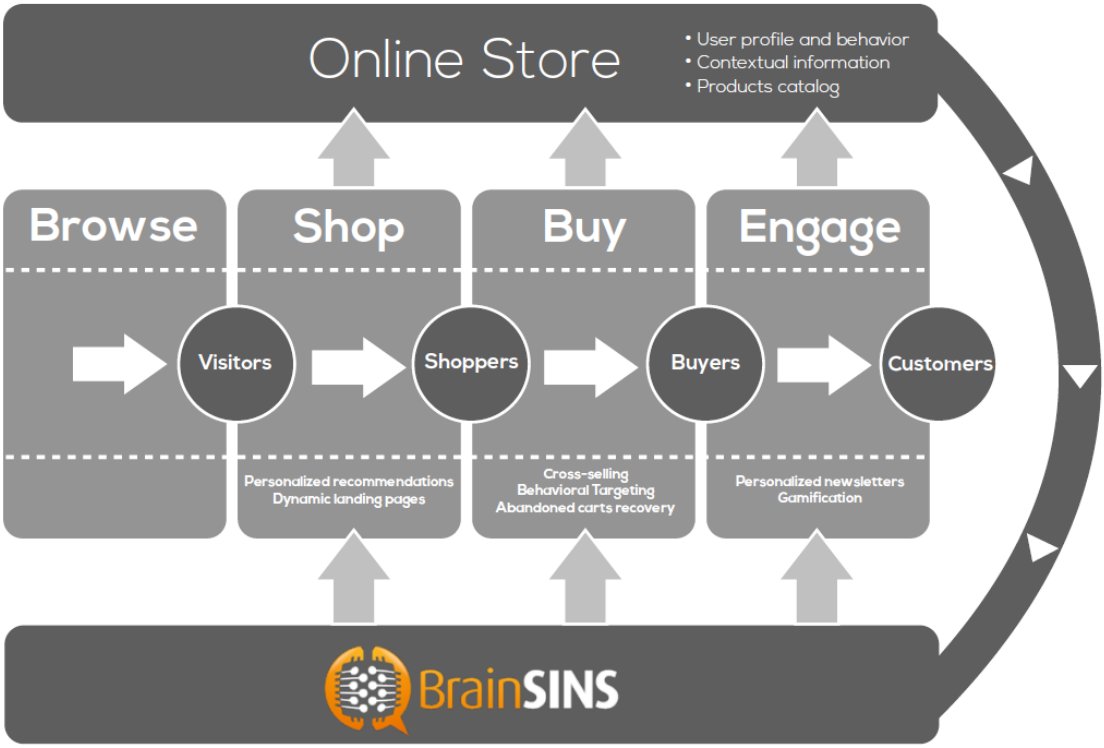


Ilustración 30: Esquema de funcionamiento

2.6.3.3 Soluciones

2.6.3.3.1 Productos Recomendados

- La única solución de personalización que permite a los retailers definir sus propias estrategias de recomendación.
- Provee un storefront para cada uno de tus clientes.
- Las recomendaciones son extensibles por medio del ordenador, móvil y otros canales.

2.6.3.3.2 Email Retargeting

- Comunicaciones personales uno-a-uno.
- Recuperación de carritos abandonados y pedidos.
- Automatización de emails basados en el comportamiento del usuario.

2.6.3.3.3 Behavioral Targeting

- Permite al equipo de IT a centrarse en lo que es importante.
- Adapta tu tienda online por comportamiento del usuario en tiempo real.

- Crea landing pages dinámicas basadas en los intereses del usuario.

2.6.3.3.4 Gamification

- Engage a tus clientes creando formidables experiencias de usuario.
- Premia aquellos comportamientos de usuario que más encajen con sus necesidades.
- Conecta los premios virtuales con cupones y descuentos.

2.6.3.4 Las características del core y sus beneficios

- Integración realmente rápida. Añade un simple JavaScript a tu página y nosotros haremos toda la magia.
- CRM Invisible. Almacenamos todas las acciones del usuario permitiéndote integrarlo en tu CRM.
- Analíticas de negocio. Te damos la información que necesitas para administrar tu negocio online.
- Editor en tiempo real. Modifica cada aspecto de tu suite personalizada en tiempo real.
- Reglas de negocio. Establece tus metas de negocio en nuestro editor de reglas de negocio para adaptar nuestra suite a tu negocio.
- Multi-canal. Establece tus metas de negocio en nuestro editor de reglas de negocio para hacer nuestra suite adaptable a tus necesidades.

2.6.3.5 Recomendadores

2.6.3.5.1 Página de inicio

- **Basado en tu historial de navegación**: Recomienda productos basados en el último producto visto por el visitante.
- **Los más vendidos en los últimos x días**: Muestra los más vendidos durante un periodo predefinido de tiempo.

2.6.3.5.2 Página de producto.

- **Clientes que compraron este producto también compraron**: Usando un producto como base se buscan productos comprados por clientes que también compraron el producto base.
- **Se suelen comprar juntos frecuentemente**: Muestra productos que aparecieron junto con el producto base en otras compras (muy útil en moda).

2.6.3.5.3 Página de categoría.

- **Más vendidos por categoría**: Muestra un ranking del top de ventas de la categoría en cuestión
- **Productos vistos recientemente**: Muestra los productos visitados recientemente por ese mismo cliente, es decir un historial de los productos que ha visitado.

2.6.3.5.4 Página de carrito

- **Los clientes que compraron este producto también compraron**: Usando un producto como base, se buscan productos comprados por clientes que también compraron el producto base.
- **Se suelen comprar juntos frecuentemente**: Muestra productos que aparecieron junto con el producto base en otras compras (muy útil en moda).

2.6.3.5.5 Página de checkout

- **Productos vistos recientemente**: Muestra los productos visitados recientemente por ese mismo cliente, es decir, un historial de los productos que ha visitado.
- **Se suelen comprar juntos recientemente**: Muestra productos que aparecieron junto con el producto base en otras compras (muy útil en moda).

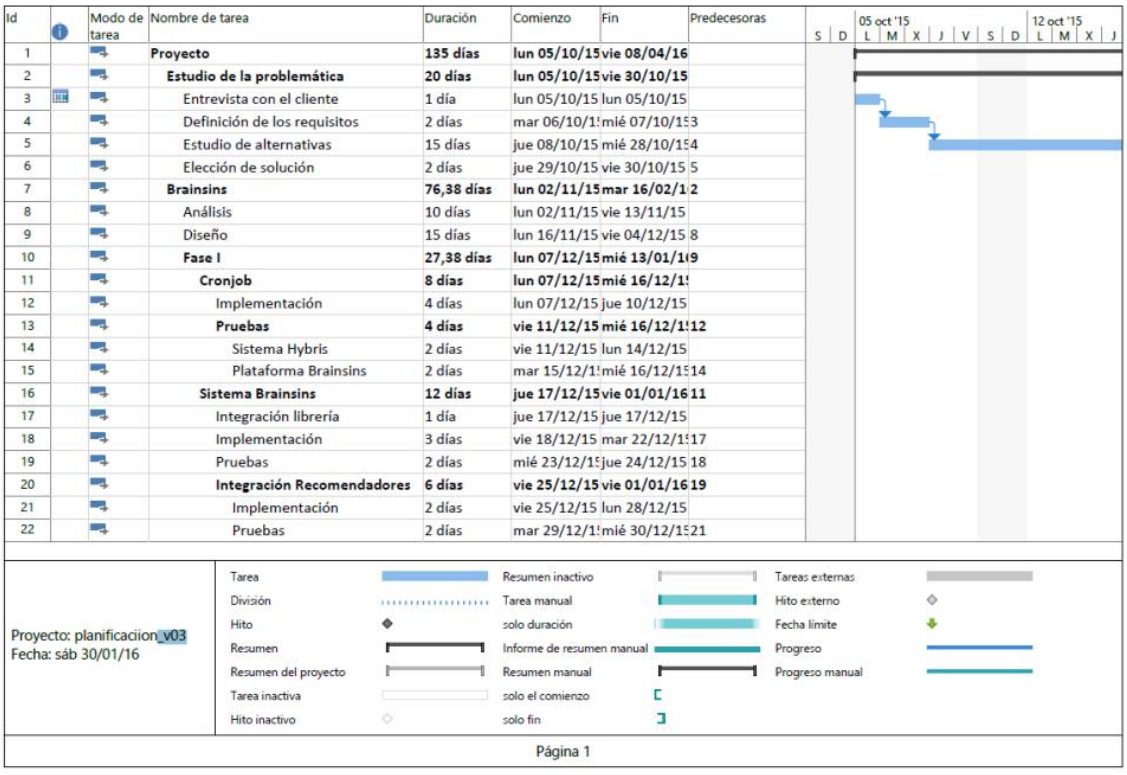
Capítulo 3. Planificación del Proyecto y Resumen de Presupuestos

3.1 Planificación

En esta sección, se debe describir el plazo de ejecución del proyecto de forma que puedan fijarse las expectativas de quienes van a recibir el producto resultado del mismo.

Debe contener un cronograma explicitando las entregas parciales, hitos intermedios y duración del proyecto a partir de la fecha de iniciación del mismo. Esto debe realizarse mediante alguna metodología de gestión de proyectos, que sirva de guía sobre la descomposición en tareas, la asignación de recursos, etc.

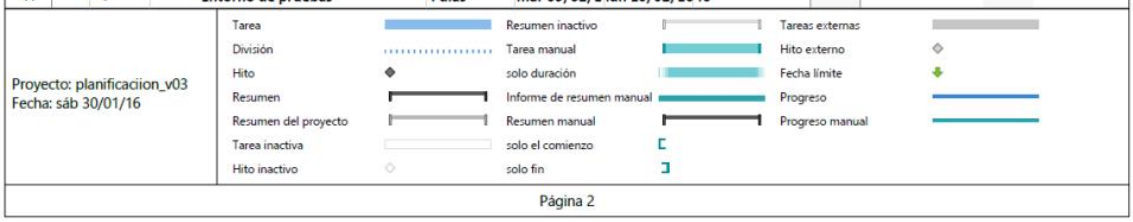
Es buena idea incluir diagramas que describan la planificación del avance del proyecto, por ejemplo es posible utilizar un diagrama Gantt:



Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1		**Proyecto**	**135 días**	**lun 05/10/15**	**vie 08/04/16**	
2		**Estudio de la problemática**	**20 días**	**lun 05/10/15**	**vie 30/10/15**	
3		Entrevista con el cliente	1 día	lun 05/10/15	lun 05/10/15	
4		Definición de los requisitos	2 días	mar 06/10/15	mié 07/10/15	3
5		Estudio de alternativas	15 días	jue 08/10/15	mié 28/10/15	4
6		Elección de solución	2 días	jue 29/10/15	vie 30/10/15	5
7		**Brainsins**	**76,38 días**	**lun 02/11/15**	**mar 16/02/16**	2
8		Análisis	10 días	lun 02/11/15	vie 13/11/15	
9		Diseño	15 días	lun 16/11/15	vie 04/12/15	8
10		**Fase I**	**27,38 días**	**lun 07/12/15**	**mié 13/01/16**	9
11		**Cronjob**	**8 días**	**lun 07/12/15**	**mié 16/12/15**	
12		Implementación	4 días	lun 07/12/15	jue 10/12/15	
13		**Pruebas**	**4 días**	**vie 11/12/15**	**mié 16/12/15**	12
14		Sistema Hybris	2 días	vie 11/12/15	lun 14/12/15	
15		Plataforma Brainsins	2 días	mar 15/12/15	mié 16/12/15	14
16		**Sistema Brainsins**	**12 días**	**jue 17/12/15**	**vie 01/01/16**	11
17		Integración librería	1 día	jue 17/12/15	jue 17/12/15	
18		Implementación	3 días	vie 18/12/15	mar 22/12/15	17
19		Pruebas	2 días	mié 23/12/15	jue 24/12/15	18
20		**Integración Recomendadores**	**6 días**	**vie 25/12/15**	**vie 01/01/16**	19
21		Implementación	2 días	vie 25/12/15	lun 28/12/15	
22		Pruebas	2 días	mar 29/12/15	mié 30/12/15	21

Ilustración 31: Diagrama de Gantt – Parte I

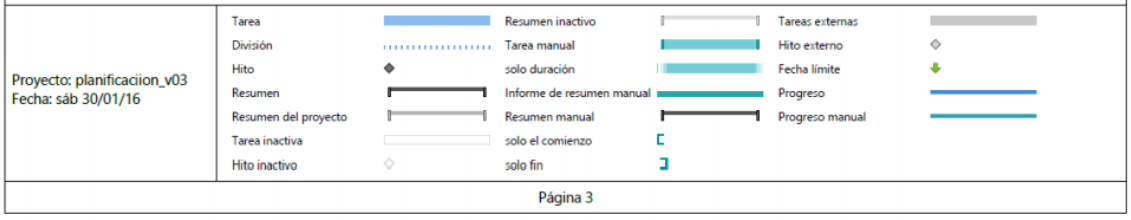
Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
23		Maquetación	1 día	jue 31/12/15	jue 31/12/15	21;22
24		Pruebas	1 día	vie 01/01/16	vie 01/01/16	23
25		Pruebas conjuntas	3 días	lun 04/01/16	mié 06/01/16	24
26		**Entorno de pruebas**	**4,38 días**	**jue 07/01/16**	**mié 13/01/16**	**25**
27		Preparación plataforma Brainsins entorno de pruebas	3 horas	jue 07/01/16	jue 07/01/16	
28		Despliegue	1 día	jue 07/01/16	vie 08/01/16	27
29		Pruebas	3 días	vie 08/01/16	mié 13/01/16	28
30		**Fase II**	**24 días**	**mié 13/01/16**	**mar 16/02/16**	**10**
31		**Cronjob**	**4 días**	**mié 13/01/16**	**mar 19/01/16**	
32		Implementación fase II	2 días	mié 13/01/16	vie 15/01/16	
33		**Pruebas**	**4 días**	**mié 13/01/16**	**mar 19/01/16**	
34		Sistema Hybris	2 días	mié 13/01/16	vie 15/01/16	
35		Plataforma Brainsins	2 días	vie 15/01/16	mar 19/01/16	34
36		**Sistema Brainsins**	**15 días**	**mar 19/01/16**	**mar 09/02/16**	**33**
37		Integración Compra Hybris	10 días	mar 19/01/16	mar 02/02/16	
38		Pruebas	2 días	mar 02/02/16	jue 04/02/16	37
39		Maquetación	1 día	jue 04/02/16	vie 05/02/16	38
40		Pruebas	2 días	vie 05/02/16	mar 09/02/16	39
41		**Pruebas**	**4 días**	**mar 09/02/16**	**lun 15/02/16**	**40**
42		Pruebas conjuntas	2 días	mar 09/02/16	jue 11/02/16	
43		Pruebas regresión	2 días	jue 11/02/16	lun 15/02/16	42
44		**Entorno de pruebas**	**4 días**	**mar 09/02/16**	**lun 15/02/16**	**40**



Página 2

Ilustración 32: Diagrama de Gantt – Parte II

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
45		Actualización plataforma Brainsins entorno de pruebas	4 horas	mar 09/02/16	mar 09/02/16	
46		Despliegue	1 día	mar 09/02/16	mié 10/02/16	
47		Pruebas	3 días	mié 10/02/16	lun 15/02/16	46
48		Manuales	1 día	lun 15/02/16	mar 16/02/16	44
49		**Entorno de producción**	**0,63 días**	**lun 15/02/16**	**lun 15/02/16**	**44**
50		Preparación del entorno	4 horas	lun 15/02/16	lun 15/02/16	
51		Despliegue	1 hora	lun 15/02/16	lun 15/02/16	50
52		Documentación	135 días	lun 05/10/15	vie 08/04/16	



Página 3

Ilustración 33: Diagrama de Gantt – Parte IV

3.2 Resumen del Presupuesto

A continuación, se muestra una tabla con el presupuesto del Trabajo Fin de Máster. Éste se ha calculado en base a las tareas incluidas en la planificación. En él se indican las subtareas, que tendrán un precio en función de su duración. Además, se incluyen los precios de las licencias y del hardware utilizado.

El proyecto tiene una duración aproximada de cuatro meses. Será llevado a cabo de lunes a viernes con una jornada laboral completa. La documentación será llevada a cabo de forma paralela durante los fines de semana. Debido a ello, se necesitará de dos meses más para su finalización y revisión. Por consiguiente, el proyecto tendrá una duración total de seis meses.

Item	SubItem	Concepto	Cantidad	Unidad	Precio Unitario	TOTAL
1		**Estudio de la problemática**				**1.525,00 €**
	1	Entrevista con el cliente	1	días	100,00 €	100,00 €
	2	Definición de requisitos	2	días	75,00 €	150,00 €
	3	Estudio de alternativas	15	días	75,00 €	1.125,00 €
	4	Elección de solución	2	días	75,00 €	150,00 €
2		**Brainsins**				**1.750,00 €**
	1	Análisis	10	días	70,00 €	700,00 €
	2	Diseño	15	días	70,00 €	1.050,00 €
2.1		**Fase I**				**2.030,00 €**
	1	Cronjob Implementación	4	días	70,00 €	280,00 €
	2	Cronjob Pruebas	4	días	70,00 €	280,00 €
	3	Sistema Brainsins	5	días	70,00 €	350,00 €
	4	Sistema Brainsins Pruebas	2	días	70,00 €	140,00 €
	5	Integración Recomendadores	3	días	70,00 €	210,00 €
	6	Integración Recomendadores Pruebas	3	días	70,00 €	210,00 €
	7	Pruebas conjuntas	3	días	70,00 €	210,00 €
	8	Entorno de pruebas - Preparación - Despliegue	2	días	70,00 €	140,00 €
	9	Entorno de pruebas - Pruebas	3	días	70,00 €	210,00 €
2.2		**Fase II**				**1.890,00 €**
	1	Cronjob Implementación	2	días	70,00 €	140,00 €
	2	Cronjob Pruebas	4	días	70,00 €	280,00 €
	3	Sistema Brainsins Integración compra Hybris	11	días	70,00 €	770,00 €
	4	Sistema Brainsins Pruebas	4	días	70,00 €	280,00 €
	5	Pruebas conjuntas	2	días	70,00 €	140,00 €
	6	Entorno de pruebas - Preparación - Despliegue	1	días	70,00 €	70,00 €
	7	Entorno de pruebas - Pruebas	3	días	70,00 €	210,00 €
3		**Pase a producción**				**259,10 €**
	1	Preparación del entorno	4	horas	54,90 €	219,60 €
	2	Despliegue	1	horas	39,50 €	39,50 €
6		**Licencias Software**				**1.660,17 €**
	1	Open Office	4	meses	0,00 €	0,00 €
	2	Microsoft Office	7	meses	15,75 €	110,25 €
	3	Microsoft Project 2016	7	meses	88,92 €	622,42 €
	4	Microsoft Visio 2010	1	meses	27,50 €	27,50 €
	5	Licencia Hybris para Equipo	1	unidad	100,00 €	100,00 €
	6	Licencia Hybris cluster 2 nodos	1	unidad	800,00 €	800,00 €
	7	Eclipse	4	meses	0,00 €	0,00 €
7		**Hardware**				**194,44 €**
	1	Equipo portátil	1	unidad	194,44 €	194,44 €
					Subtotal	9.308,71 €
					IVA (21%)	1.954,83 €
					TOTAL	**11.263,54 €**

Ilustración 34: Resumen del Presupuesto

Capítulo 4. 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.

4.1 Definición del Sistema

4.1.1 Determinación del Alcance del Sistema

El objetivo principal del presente proyecto, consiste en desarrollar la integración de BrainSINS, una solución de marketing automation, con la plataforma de comercio electrónico Hybris.

La empresa de alimentación DIA, desea mejorar, entre otras, una de las soluciones de marketing más utilizadas en la actualidad, los recomendadores de productos. Hasta el momento, sólo tenía la posibilidad de utilizar los carruseles de productos que la propia plataforma de Hybris le ofrecía. Sin embargo, estos carruseles tienen grandes desventajas. Los productos recomendados en cada uno de los carruseles son totalmente fijos, es decir, para cada uno de los carruseles, se definen un conjunto fijo de productos a mostrar, cuyos criterios de selección no tienen una base sólida. Además, todos los clientes de la web van a recibir la misma recomendación. Los productos recomendados no tienen en cuenta la tienda en la que se encuentra el usuario, con lo que se recomendarán productos que no se comercializan en dicha tienda. De este modo, no se estaban consiguiendo los objetivos que se buscaban. La probabilidad de que los productos fuesen incluidos en el carrito del usuario era tremendamente bajo, pues cada cliente tiene distintos gustos, edades... aspectos que no se tienen en cuenta.

Por todo ello, DIA necesita de una solución inteligente. Necesita que los productos recomendados realmente tengan una alta probabilidad de ser añadidos en el carrito de cada cliente, para así incrementar las ventas y el precio final del pedido. Los puntos débiles de los recomendadores de Hybris, se convierten en los puntos fuertes de BrainSINS. Pero existe un problema, los productos recomendados no ofrecen la funcionalidad de ser añadidos directamente al carrito del usuario. Ésta es condición indispensable para DIA, que además necesita que la solución sea visualmente idéntica al carrusel de productos de Hybris, y que estos carruseles puedan sera añadidos a las diferentes páginas del sitio web por medio de la herramienta CMSCOCKPIT proporcionada por Hybris.

- Los recomendadores de BrainSINS tienen que ser visualmente idénticos a los recomendadores de Hybris.
- Cada producto incluido en el carrusel de BrainSINS dispondrá de la funcionalidad de ser añadidos directamente al carrito del usuario desde el propio recomendador, al igual que sucede en el recomendador de Hybris.
- Mostrar productos recomendados en base a unos criterios sólidos.

- Mostrar los productos recomendados en función de la tienda en la que se encuentra el usuario.
- Los productos recomendados deberán variar en función del tipo de página.
- Los productos recomendados varían en función del cliente que ha iniciado sesión.
- Mostrar diferentes tipos de recomendadores.

Para que BrainSINS pudiera proveer de la funcionalidad indicada anteriormente, es necesario que el sitio web le envíe cierta información.

- Diariamente, proveer del catálogo de productos de DIA, que se encuentran en la versión online, y para los cuales existen stock, en función de la tienda. El formato del fichero deberá ser XML y deberá seguir la especificación propia de BrainSINS.
- Integrar la librería BrainSINS vía javascript, necesaria para enviar la información de la página en la que se encuentra el cliente, así como la información asociada a dicho tipo de página y al propio usuario.
- Enviar tracking de todos los clics que ha llevado a cabo el usuario en los recomendadores.
- Enviar tracking de todos los productos que se han añadido o eliminado del carrito del cliente, tanto si han sido llevados a cabo o no desde el propio recomendador.
- Generar el HMTL y CSS dedicados al recomendador y productos recomendados de Brainsins para que sea visualmente idéntico al carrrusel de productos recomendados de Hybris.
- Incluir en cada uno de los productos mostrados la funcionalidad de añadir el producto directamente al carrito del usuario.

4.2 Requisitos del Sistema

4.2.1 Obtención de los Requisitos del Sistema

4.2.1.1 Especificación textual

Este proyecto, servirá para integrar la solución de marketing automation de BrainSINS en la herramienta de comercio electrónico Hybris. El proceso de integración debe cumplir tanto con los requisitos de DIA como con los de BrainSINS.

4.2.1.1.1 Generación de un fichero XML con el conjunto de productos ofertados por DIA

BrainSINS necesita conocer el catálogo de productos de DIA, para poder llevar a cabo su recomendación. Además, necesita conocer los productos que se ofrecen en cada tienda, junto a otros datos de cada uno de los productos. Por cada producto se deberá indicar:

- IdProduct: Identificador del producto.
- Name: Nombre del producto, descripción.
- Url: URL de la página de detalle del producto.

- ImageUrl: URL de la imagen del producto.
- Categories: Especificación textual de las categorías a las que pertenece el producto.
- Price: Precio del producto en una de las tiendas.
- Prices: Precio del producto en cada una de la tiendas en las que se oferta.
- Misc: Este campo incluirá información adicional, no obligatoria para BrainSINS. En nuestro caso, lo utilizaremos para incluir la media de las puntuaciones obtenidas en cada producto, la unidad de venta y el tipo de venta, datos necesarios para el botón "Añadir al carrito".

Esta información, estará contenida en un fichero XML, que seguirá el formato de BrainSINS.

4.2.1.1.2 Actualización automatizada del catálogo de productos

El catálogo de los productos de DIA se actualiza diariamente. De este modo, es indispensable, que BrainSINS reciba este fichero todos los días, una vez se haya llevado a cabo la actualización. De este modo, sólo se mostrarán los productos que realmente se venden en la tienda, con su información actualizada.

Será necesario el desarrollo de un Job encargado de crear el fichero XML con el conjunto de los productos vendidos por DIA, que se se encuentren en la versión online del catálogo de productos, cuyo estado sea aprobado y que además posean stock. En el fichero se indicará el precio de cada producto en función de la tienda, de tal forma que BrainSINS pueda mostrar el precio correcto, en función de la tienda en la que se encuentre el usuario.

El Job deberá de tener configurado un Trigger para que se ejecute diariamente a una hora determinada.

La inclusión del Job en el sistema será llevado a cabo vía IMPEX en la herramienta HAC de la plataforma de Hybris.

4.2.1.1.3 Publicación del catálogo de Productos

Es necesario que el fichero se encuentre en una dirección pública para que BrainSINS pueda acceder a él. El Job almacenará el fichero en una ruta compartida por ambos nodos de la aplicación.

Por medio de la creación de un enlace simbólico en el sistema Linux, el fichero será publicado.

4.2.1.1.4 Definición HTML del recomendador en BrainSINS

Es preciso, que el HTML del recomendador de BrainSINS sea especificado en su plataforma web.

4.2.1.1.5 Definición HTML de los productos recomendados en BrainSINS

La definicón HTML de los productos recomendados será especificado en su plataforma web.

4.2.1.1.6 Mostrar productos recomendados en función de la tienda

DIA dispone de diferentes tiendas desde la que ofrece venta online. Necesita que los productos recomendados estén disponibles y se oferten en la tienda en la que se encuentra el usuario. Para ello, el fichero de productos, contendrá el precio del producto por tienda, para aquellas tiendas en las que el producto se oferte y además se encuentre disponible.

4.2.1.1.7 Incorporación de la funcionalidad 'Añadir al carrito' '+' '-'

DIA necesita que los productos recomendados por BrainSINS puedan ser añadidos al carrito del usuario, por medio de la funcionalidad 'Añadir al carrito' que incorporan los carruseles de Hybris. Cabe destacar que esta funcionalidad no es ofrecida por BrainSINS.

Para la incorporación de esta funcionalidad será necesario añadir los estilos adecuados una vez se ha recibido la respuesta de BrainSINS con los productos recomendados.

4.2.1.1.8 Recomendador BrainSINS visualmente idéntico al de Hybris

Se trata de un requisito definido por DIA, en el que solicita que el recomendador de BrainSINS sea visualmente idéntico al de Hybris, es decir, para el usuario se tratará del mismo recomendador. Esto es muy importante, ya que ambos carruseles serán utilizados.

Cada producto recomendado deberá incluir la siguiente información:

- Imagen del producto.
- Descripción del producto.
- Valoración del producto, que será mostrado por medio de cinco estrellas.
- Precio del producto en la tienda en la que se encuentra el usuario.

Tanto a través de la imagen del producto, como de su descripción, se podrá acceder a la página del detalle del producto.

Además, los productos recomendados, podrán ser incluidos en el carrito por medio de la funcionalidad “Añadir al carrito”.

4.2.1.1.9 Mostrar más de un tipo de recomendador BrainSINS por página

En las páginas del sitio web, se permitirá incluir más de un tipo de recomendador de BrainSINS.

4.2.1.1.10 Inclusión de los recomendadores vía herramienta Hybris

Los recomendadores de BrainSINS deberán de ser incluidos en las páginas del site, por medio de la herramienta cmscockpit de Hybris. Se trata de una herramienta desde la que se permite configurar las diferentes páginas del site. Añadir elementos, eliminarlos...

Se define que los recomendadores se incluyan dentro de un elemento párrafo de Hybris, en el que se incluirá un sencillo div que contendrá el identificador del propio div, así como el identificador numérico del propio recomendador de BrainSINS.

4.2.1.1.11 Envío de información a BrainSINS

El sistema BrainSINS deberá de recibir diferente información en función de la página web en la que se encuentra el usuario. De esta forma recabará la información adecuada para mostrar los diferentes tipos de recomendadores que incorpora.

El sistema diferencia cinco tipos de páginas:

- **Página principal del sitio web**
  - Tipo de página: **'home'.**
  - Email del cliente del sitio web.
  - El identificador de la tienda en la que se encuentra el usuario.
- **Página donde se muestra el detalle del producto**
  - Tipo de página**: 'product'**.
  - Email del cliente del sitio web.
  - El identificador de la tienda en la que se encuentra el cliente.
  - El identificador del producto en el catálogo.
- **Página de categorías**
  - Tipo de página: **'category'**.
  - Email de cliente del siitio web.
  - Identificador de la tienda en la que se encuentra el cliente.
  - El identificador de la categoría en la que se encuentra el usuario.
- **Página del carrito / checkout**
  - En el caso de DIA, la página del carrito se trata del primer paso del checkout.
  - Tipo de página: **'checkout'**.
  - Email del cliente del sitio web.
  - Identificador de la tienda en la que se encuentra el usuario.
  - Información del carrito del usuario, pares de valores con el identificador del producto y las unidades del mismo incluidas en el carrito.
- **Página del confirmación de compra**
  - Tipo de página **'thankYou'**.
  - Email del cliente que ha realizado la compra.
  - Identificador de la tienda donde se ha realizado la compra.
  - Precio total de la compra.

4.2.1.1.12 Envío de información a BrainSINS para la inclusión del recomendador

En aquellas páginas en la que se deseen incluir recomendadores, será necesario enviar los identificadores de los elementos HTML donde incluirlos, junto al identificador numérico del propio recomendador.

BrainSINS dispone de diferentes recomendadores en función del tipo de página en la que se encuentre el usuario. Éstos se indican a continuación, en base al tipo de página:

Página HOME

- **Basado en tu historial de navegación.**

    - Identificador: 1
    - Coloca como primer producto el último visitado por el cliente. El resto son productos similares/alternativos al último visitado. En este caso se tienen en cuenta las categorías de los productos. Cuantas más categorías tengan en común, más probabilidades tiene de salir unos u otros.
- **Los más vendidos.**
    - Identificador: 2
    - Productos más vendidos. Para los más vendidos se basan en el cierre de compras que se registran gracias al tracking.
- **Recomendaciones de nuevos productos.**
    - Identificador: 3
    - Recomendador que muestra nuevos productos en base a la fecha de subida que tienen en el sistema. Se pueden utilizar otros factores, como un campo extra en el catálogo que indique realmente la fecha de entrada al stockage.

Página PRODUCT

- **Los clientes que compraron este producto también compraron.**
    - Identificador: 4
    - Usando un producto como base, se buscan productos comprados por clientes que también compraron el producto base.
- **Se suelen comprar juntos frecuentemente.**
    - Identificador: 5
    - Muestra productos que aparecieron juntos con el producto base en otras compras.
- **Productos en categorías similares.**
    - Identificador: 6
    - Usando como base alguno de los últimos productos vistos por el usuario, se buscan productos similares por categorías y se muestran. Cuantas más categorías tengan en común con el producto base, más probabilidades tienen de salir unos u otros.

Página CART

- **Los clientes que compraron productos que aparecen en tu cesta de la compra también compraron.**
    - Identificador: 7
    - Cogiendo como base la cesta de la compra actual, muestra productos que se compraron junto con los de la cesta.
- **Productos que se compran junto con el último añadido al carrito.**
    - Identificador: 8
    - Productos que se añadieron al carrito junto con el último añadido al carrito.

Tipo CHECKOUT

- **Se suelen comprar juntos**

  - Identificador: 9
  - Cogiendo como base la cesta de la compra actual, muestra productos que se compraron junto con los de la cesta.
- **Productos vistos recientemente.**
  - Identificador: 10
  - Muestra los productos visitados recientemente por ese mismo cliente, es decir, un historial de los productos que ha visitado.

4.2.1.1.13 Tracking de los productos incluidos en el carrito

La herramienta BrainSINS también necesita tracking de los clics llevados a cabo en los recomendadores por los clientes.

4.2.1.1.14 Tracking del botón '+'

BrainSINS necesita de tracking de los productos que el cliente añade a su carrito tanto si se produce por medio del recomendador de Brainsins como si se realiza desde cualquier otro elemento de la vista.

4.2.1.1.15 Tracking del botón '-'

Brainsins también necesita tracking del botón '-' de forma análogo al tracking del botón '+'.

4.2.1.2 Requisitos Funcionales

A continuación se presenta la tabla que recoge los requisitos funcionales que debe satisfacer la integración.

Código	Nombre Requisito	Descripción del Requisito
R1.1	Incluir Job	Desde la herramienta HAC del sistema Hybris, el administrador debe incluir Job dedicado a la generación del catálogo de productos en sistema Hybris.
R1.2	Eliminar Job	Desde la herramienta HMC el administrador podrá eliminar el Job del sistema Hybris.
R2.1	Configurar Trigger para el Job	Desde la herramienta HMC, el administrador podrá configurar un trigger para el Job, donde indicarála fecha y hora de su ejecución.
R2.1	Incluir Recomendador	Desde la herramienta CMS el administrador podrá incluir el código HMTL necesario para la inclusión del

		recomendador.
R2.2	Eliminar Recomendador	Desde la herramienta CMS, el administrador podrá eliminar el elemento HTML donde incluir el recomendador.
R2.3	Actualizar Recomendador	Desde la herramienta CMS, el administrador podrá actualizar el recomendador que desea mostrar.
R3.1	Incluir Catálogo de Productos	Por medio de la herramienta Brainsins, el administrador, incluirá el catálogo XML generado por el Cronjob.
R3.2	Actualizar Catálogo de Productos	Por medio de la herramienta Brainsins, el administrador podrá actualizar la dirección donde se encuentra publicado el catálogo de productos.
R4.1	Definir el HMTL del recomendador	En la herramienta de Brainsins, el administrador podrá definir el HTML que la página del sito web recibirá como respuesta.
R4.2	Definir el código CSS del recomendador	El administrador de Brainsins podrá definir el CSS del recomendador
R5.1	Incluir código de la tienda.	El administrador de la herramienta Brainsins podrá incluir los códigos de las tiendas donde DIA ofrece venta online.
R5.2	Eliminar código de la tienda.	El administrador de la herramienta Brainsins podrá incluir los códigos de las tiendas donde DIA ofrece venta online.
R6.1	Envío de información a Brainsins	En función del tipo de página en la que se encuentre el usuario, se enviará a Brainsins la información adecuada.
R7.1	Tracking de las acciones añadir y eliminar del carrito	Brainsins recibirá los productos añadidos o eliminados del carrito por los clientes del sitio web.

R8.1	Añadir producto al carrito desde recomendador BrainSINS	Los productos recomendados en el carrusel de BrainSINS podrán ser añadidos al carrito, al igual que el resto del productos del sitio web.
R8.2	Actualizar unidades del producto en el carrito desde el recomendador BrainSINS	Se podrá actualizar las unidades del producto incluidos en el carrito con los botones '+' '-', al igual que el con el resto de productos del sitio web.

4.2.1.3 Requisitos No Funcionales

4.2.1.3.1 Requisitos de Clientes

A continuación se recogen los requisitos no funcionales de los clientes:

- Se presupone que los clientes del sitio web tienen conocimientos de usuario estándar.
- Se presupone que el personal de DIA tiene conocimientos sobre la herramienta CMS de Hybris.

4.2.1.3.2 Requisitos Tecnológicos

En la siguiente lista se recoge el conjunto de requisitos tecnológicos.

- La integración estará desarrollada utilizando estándares del W3C (HTML, CSS, etc).
- La integración se llevará a cabo con la solución Hybris que utiliza Java y el framework Spring MVC.
- La inclusión de la liberaría de BrainSINS se llevará a cabo por medio JavaScript y Jquery.
- Se presupone que los administradores de la herramienta Hybris tienen conocimiento de un usuario administrador.
- Se presupone que el usuario encargado de llevar a cabo la integración está familiarizado con entornos Linux.

4.2.1.3.3 Requisitos de Tiempo de Respuesta

Los recomendadores deberán cargarse en la página de forma inapreciable por el usuario, como si formase parte de la propia página.

4.2.2 Identificación de Actores del Sistema

En esta integración, tanto el sitio web de DIA como el sistema BrainSINS, deberán de trabajar de forma conjunta.

Por un lado, el sistema Hybris publicará en el sitio web de DIA su catálogo de productos en el formato especificado por BrainSINS. BrainSINS recogerá este fichero y lo añadirá a su sistema, para poder recomendar los productos.

Por otra lado, el sitio web enviará a BrainSINS la información asociada a las interacciones de sus clientes con él. Además, cuando en la página se requieran mostrar los recomendadores, el sitio web envirará la información asociada a los mismos y recibirá la respuesta de BrainsSINS con los recomendadores.

Las anteriores interacciones provocan la aparición de tres actores:

- **Usuario del sitio web.** Representa a los clientes del sitio Web que interactúan con la página, y cuyas acciones generan envío de información a BrainSINS. Además, podrán tanto incluir los productos recomendados en su carrito, como modificar las unidades, o bien eliminar el producto de su carrito.
- **Sistema Hybris.** Representa al propio sitio web de DIA, que será el encargado de enviar la información necesaria a BrainSINS, a partir de las acciones que llevan a cabo los clientes de DIA. También se encarga de publicar su catálogo de productos, que el sistema BrainSINS consultará.
- **Sistema BrainSINS.** Se encarga de procesar la información que recibe desde el sitio web , así como de generar los productos recomendados.

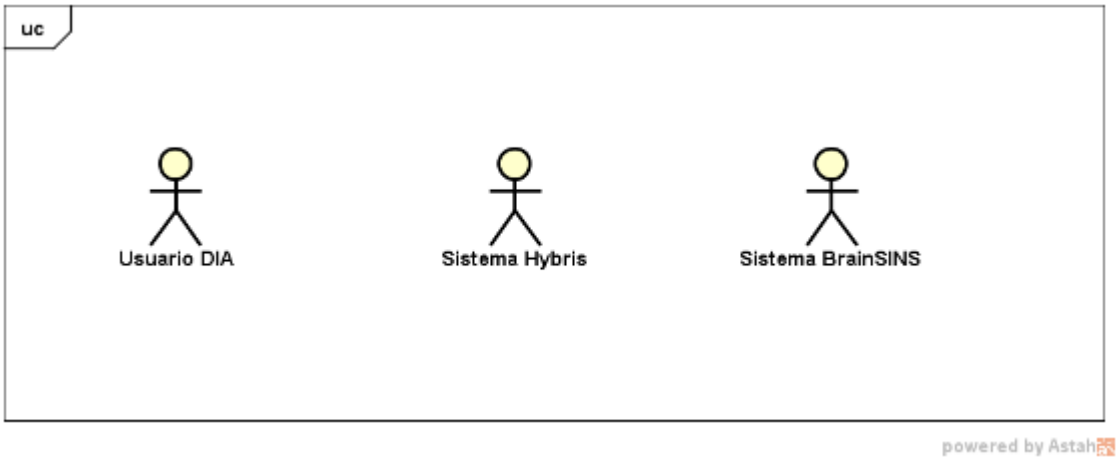


Ilustración 35: Actores del Sistema

4.2.3 Especificación de Casos de Uso

A continuación, se presentan los casos de uso de la integración , comenzando con un diagrama de casos de uso generalista, hasta llegar a los casos de uso concretos.

4.2.3.1 Casos de Uso Generales

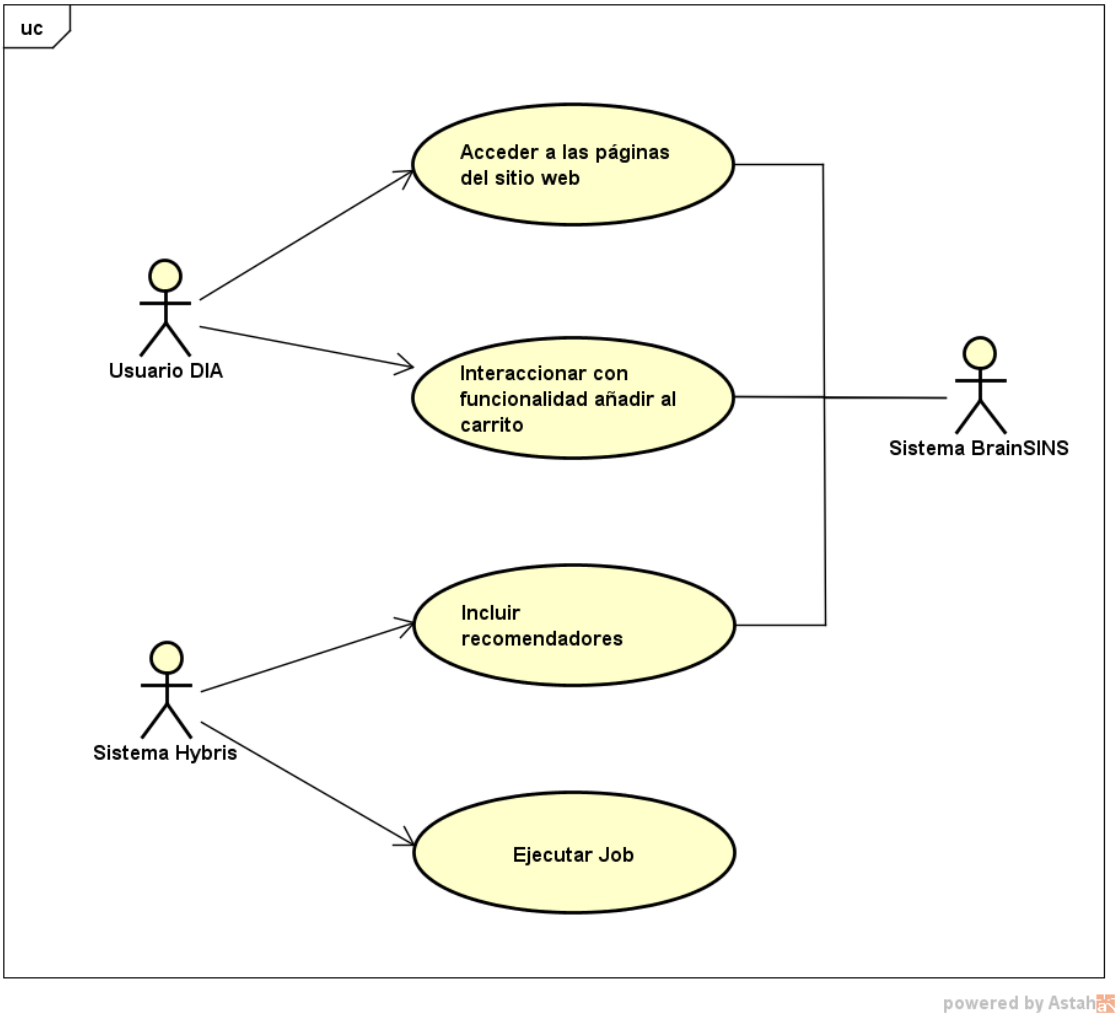


Ilustración 36: Casos de Uso Generales

Caso de Uso: Acceder a las páginas del sitio web

El cliente accede al sitio web. El sistema le muestra la página del sitio. El sistema envía a BrainSINS la información asociada a dicha página.

Caso de Uso: Interaccionar con las funcionalidades añadir al carrito

El usuario del sitio web lleva a cabo una acción en la funcionalidad 'Añadir al carrito'. El sistema Hybris enviará la información asociada a dicha acción al sistema BrainSINS.

Caso de Uso: Incluir recomendadores

Una vez el usuario accede a una página donde se debe incluir un recomendador, el sistema Hybris enviará al sistema BrainSINS la información referente al recomendador. Una vez el sistema BrainSINS recibe la información, enviará al sistema Hybris el recomendador solicitado. Una vez el sistema Hybris lo recibe, deberá recorrer cada uno de los productos incluidos en el recomendador y añadirles la funcionalidad 'Añadir al carrito'.

Caso de Uso: Ejecutar Job

En la fecha y horas determinadas por el Trigger del Job, el sistema Hybris llevará a cabo la ejecución de Job. El Job accederá a la base de datos y ejecutará la consulta que consistirá en tomar el conjunto de productos del catálogo online que dispongan de stock, que esten aprobados y que es estén a la venta al menos en una de las tiendas de DIA. La base de datos responderá con el listado de productos resultado de la consulta. El Job a partir de dicha respuesta, generará un XML con la información de los productos necesaria en el formato de BrainSINS. EL fichero se almacenará en la ruta indicada. Por último, el cronjob finalizará su ejecución quedando en el estado “Finished” y siendo su resultado de ejecución “Success”.

4.2.3.2 Casos de Uso Acceso a las páginas del sitio web

Dada la complejidad del caso de uso 'Acceder a las páginas del sitio web', a continuación se lleva a cabo un desglose del mismo.

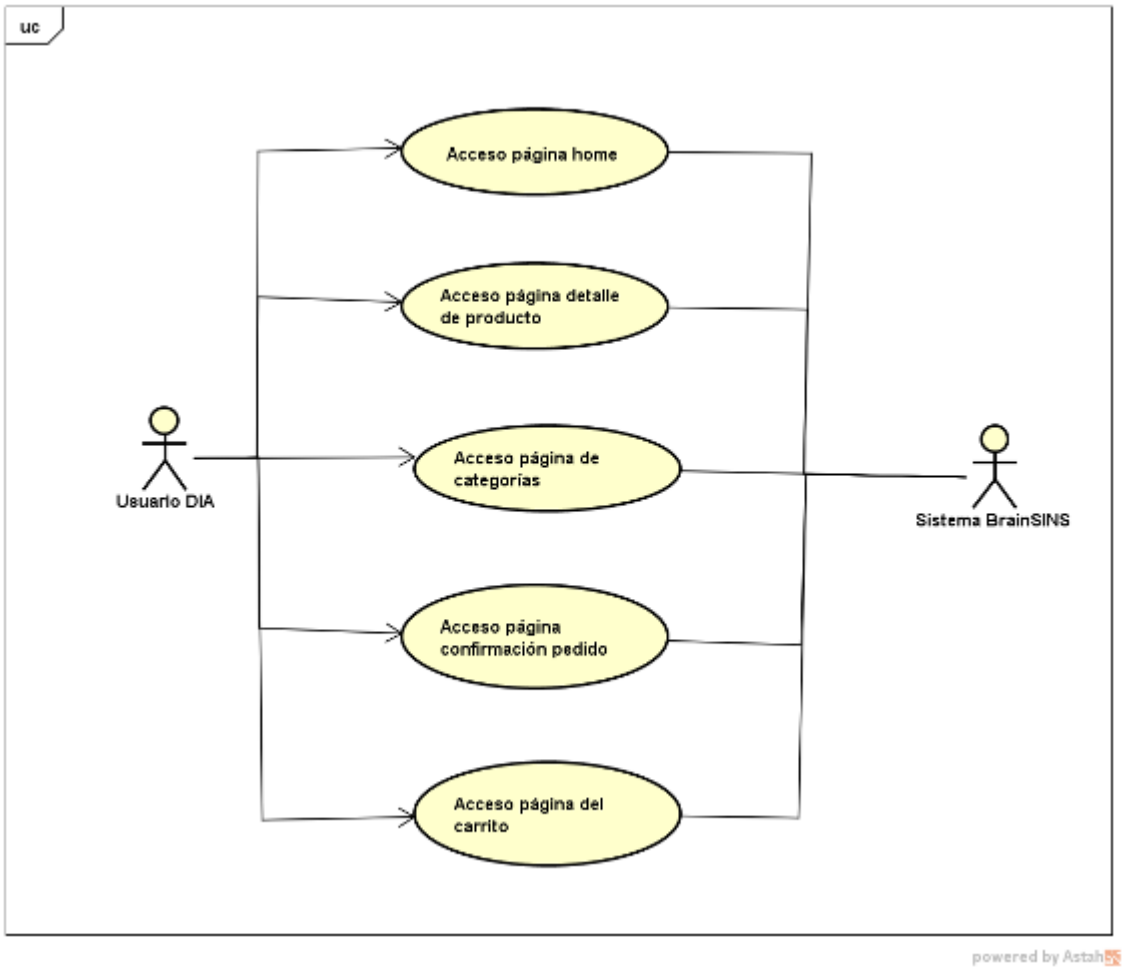


Ilustración 37: Casos de Uso Acceso a las páginas del sitio web

Caso de Uso: Acceso a la página home

El usuario solicita la página home. El sistema Hybris le muestra la página. El sistema Hybris por medio de la librería javaScript de BrainSINS envía los siguientes datos:

- Tipo de página 'home'
- Email del cliente que ha accedido a la página, en el caso de que haya iniciado sesión.
- Código identificador de la tienda en la que se encuentra el usuario.
- BrainSINS recibe la información.

Caso de uso: Acceso a la página de detalle de producto.

El usuario solicita la página del detalle de un producto. El sistema muestra al usuario dicha página. El sistema envía a BrainSINS la información asociada a dicha página:

- Tipo de página 'product'
- Email del usuario con el que ha llevado a cabo el inicio de sesión, en el caso de que haya iniciado sesión.
- Código identificador de la tienda en la que se encuentra el usuario.
- Identificador del producto.
- Categoría a la que pertenece el producto.

Caso de uso: Acceso a página de categorías.

El usuario solicita la página asociada a una categoría. El sistema Hybris le muestra un listado de productos que pertenecen a la categoría. El sistema, además, enviará a BrainSINS la información asociada a dicha página:

- Tipo de página en que se encuentra 'category'.
- Email del usuario con el que ha llevado a cabo el inicio de sesión en el caso de que haya iniciado sesión.
- Código identificador de la tienda en la que se encuentra el usuario.

Caso de uso: Acceso a la página de carrito.

El cliente solicita la página de su carrito, que también se corresponde con el primer paso del checkout. El sistema le muestra la página, en la que se incluyen todos los productos que forman parte de su carrito. El sistema, también envía a BrainSINS la información asociada a dicha página:

- Tipo de página 'cart'.
- Identificador del usuario con el que ha iniciado sesión, en el caso de haberlo hecho.
- Código identificador de la tienda en la que se encuentra el usuario.
- Listado de los productos que forman parte de su carrito.

Caso de uso: Acceso a la página de confirmación de pedido

El cliente finaliza todos los pasos del checkout. El sistema muestra la página de confirmación del pedido. A continuación, el sistema envía a BrainSINS la información asociada la página:

- Tipo de página en que se encuentra 'thankYou'
- Identificador del usuario en el sitio web.
- El coste total del pedido.

4.2.3.3 Casos de Uso Funcionalidad Añadir al carrito

Dada la complejidad del caso de uso 'Interaccionar con funcionalidad añadir al carrito', a continuación se muestra un desglose del mismo.

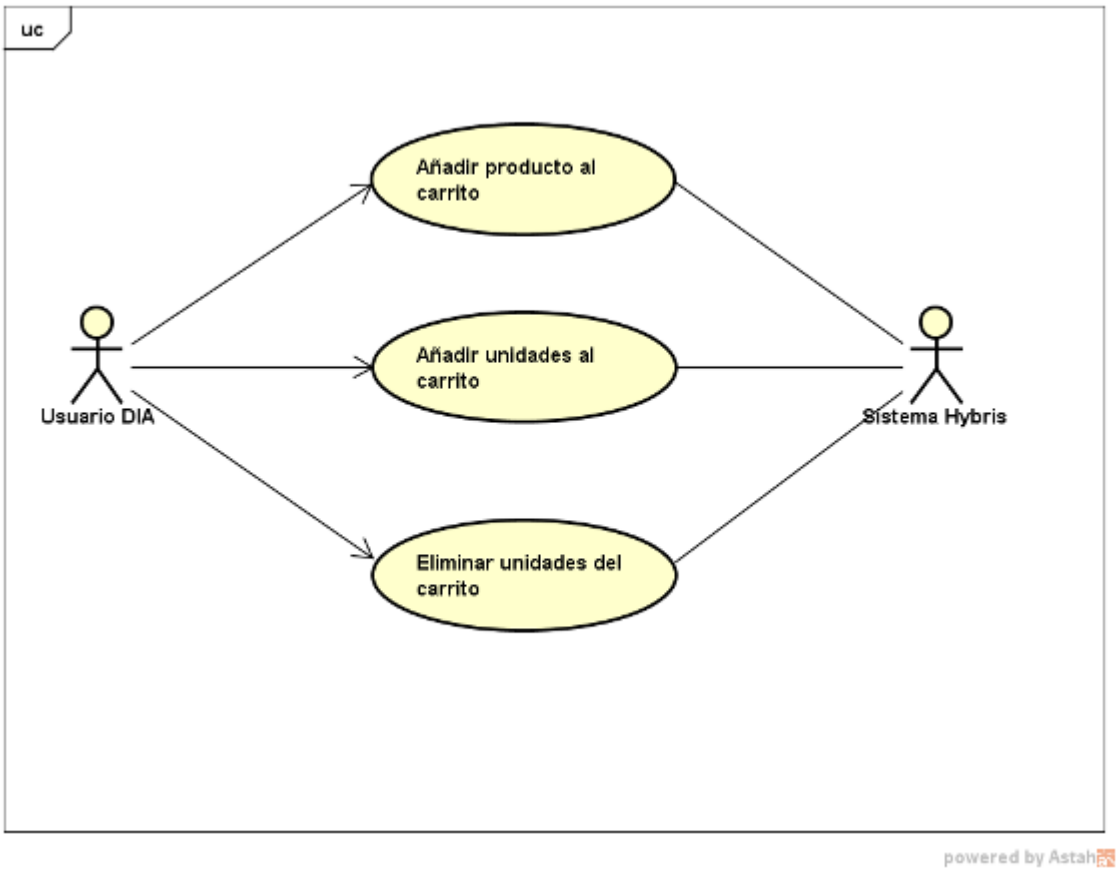


Ilustración 38: Casos de Uso Funcionalidad Añadir al carrito

Caso de uso: Añadir producto al carrito.

- El usuario selecciona el botón “Añadir” de uno de los productos . El sistema añade una unidad del producto al carrito del usuario. El sistema lleva a cabo el recálculo del total del carrito. El sistema actualiza el minicarrito del usuario, añadiendo el nuevo producto. Del formulario del producto añadido al carrito, se oculta el botón “Añadir” y se muestran los botones “+” y “-”, así como la cantidad incluida en el carrito, en este caso , una unidad. El sistema envía a BrainSINS la siguiente información:
  La página en la que se encuentra el usuario.
- El identificador del usuario en el site.
- El identificador del producto.
- La cantidad añadida al carrito, en este caso 1.
- El identificador del recomendador que incluye dicho botón, en el caso de que el producto forme parte de un recomendador.

Caso de uso. Añadir unidades al carrito.

El usuario selecciona el botón "+" de uno de los productos que previamente había incluido en su carrito. El sistema comprueba que exista stock. En caso afirmativo, el sistema añade una unidad del producto al carrito del usuario. El sistema lleva a cabo el recálculo del total del carrito. El sistema actualiza el minicarrito del usuario añadiendo el nuevo producto. El sistema actualiza las unidades del formulario del producto donde se incluye la funcionalidad 'Añadir al carrito'. El sistema envía a BrainSINS la siguiente información:

- La página en la que se encuentra el usuario.
- El identificador del usuario en el site.
- El identificador del producto.
- La cantidad añadida al carrito.
- El identificador del recomendador que incluye dicho botón, en el caso de que el producto forme parte de un recomendador.

Caso de uso: Eliminar unidades del carrito.

El usuario selecciona el botón "-" de uno de los productos recomendados por el recomendador de BrainSINS y que previamente había añadido a su carrito. El sistema elimina una unidad del producto en el carrito. En el caso de que este valor sea cero, el sistema eliminará el producto del carrito del usuario. El sistema llevará a cabo el recálculo del precio total del carrito. El sistema actualizará el componente minicarrito del usuario eliminando una unidad del producto o bien eliminando el producto del carrito del usuario en caso de que la unidad sea cero. El sistema actualizará las unidades del formulario del producto dedicado a interaccionar con el carrito del usuario. El sistema envía a BrainSINS la siguiente información:

- La página en la que se encuentra el usuario.
- El identificador del usuario en el site.
- El identificador del producto.
- La cantidad eliminada del carrito.
- El identificador del recomendador que incluye dicho botón, en el caso de que el producto forme parte de un recomendador.

4.3 Identificación de los Subsistemas en la Fase de Análisis

Partiendo de la información anterior, se puede observar que la integración, se compone de dos sistemas diferenciados.

Por un lado, se encuentra el sistema encargado de generar el fichero XML con el conjunto de productos que se encuentran en el catálogo de DIA. Este fichero será consumido por el sistema BrainSINS para poder llevar a cabo las recomendaciones de los productos.

Por otro lado, se encuentra el sistema encargado de la comunicación del sitio Web con el sistema BrainSINS. Tanto para el envío de la información asociada a las acciones del usuario en las páginas del sitio web, como para la recepción de los recomendadores de BrainSINS a incluir.

4.3.1 Identificación de los Subsistemas en la Fase de Análisis

En este apartado, se procederá a analizar los subsistemas de cada uno de los sistemas que forman parte de la integración. Cada subsistema se descompondrá en sistemas más pequeños que faciliten su posterior análisis.

Cabe destacar la organización en extensiones que lleva a cabo Hybris según las funcionalidades. Organización que será seguida en las subdivisiones.

4.3.1.1 Sistema diabrainsins

Se encarga de la creación de un fichero XML donde se incluyen el conjunto de productos que pertenecen al catálogo de la versión 'online' de DIA, que se encuentran disponibles para su venta al menos en una de las tiendas, que están aprobados y que poseen stock. Éste fichero seguirá la especificación de BrainSINS. Los productos de este fichero serán utilizados por BrainSINS en sus recomendadores. Sin ellos, BrainSINS no podrá llevar a cabo ninguna recomendación.

4.3.1.1.1 Descripción de los subsitemas

El sistema Diabrainsins, que forma parte de una nueva extensión que toma su nombre, se encargará de crear el fichero XML y de publicarlo en una carpeta compartida por ambos nodos.

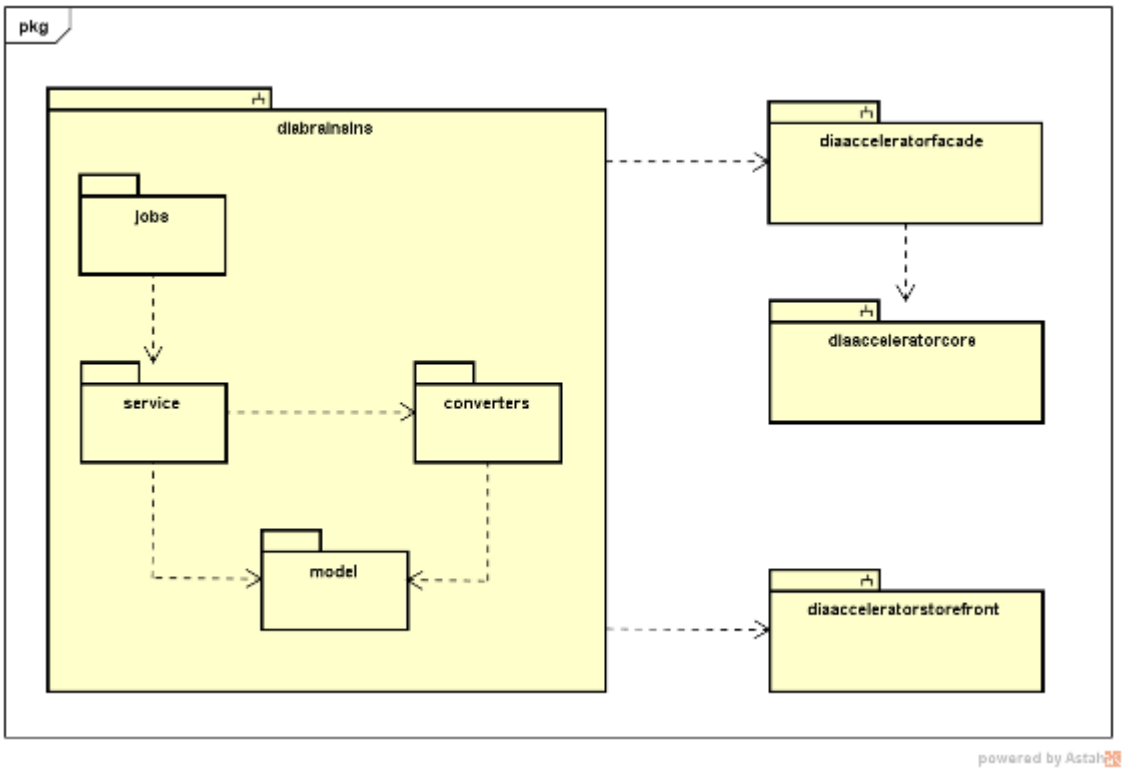


Ilustración 39: Diagrama de paquetes de los subsistemas

4.3.1.1.2 Diabrainsins

Se trata de la extensión dedicada a la generación del fichero XML donde se exportan el conjunto de productos del catálogo DIA.

En el paquete jobs se encuentra la clase BrainsinsCronjob. Esta clase se comunicará con la extensión diaacceleratorfacades por medio de la clase 'ProductFacade' para obtener el listado de productos 'DiaProductModel' a exportar.

El paquete 'service' contendrá a la clase 'BrainsinsService' encargada de convertir el listado de productos 'DiaProductModel' en los productos que siguen la especificación BrainSINS. Este se llevará a cabo utilizando las clases convertidoras incluidas en el paquete 'converters'.

Por último, una vez el fichero resultante sea publicado, el paquete 'diaacceleratorstorefront' será el encargado de responder a las peticiones en las que se solicite el fichero XML fruto de la exportación.

4.3.1.1.3 Diaacceleratorfacades

Esta extensión, está formada por las fachadas que se encargan de acceder a la capa de acceso a la datos del sistema.

En ella se incluye el servicio 'ExtendedProductService', encargado de llevar a cabo operaciones CRUD sobre la base de datos de la tabla 'Products', utilizando la extensión 'diaacceleratorcore'. En este servicio se incluirá una funcionalidad dedicada a la obtención de los productos a exportar para Brainsins.

4.3.1.1.4 Diaacceleratorcore

Se trata de la extensión que dispone de los servicios que se encargan de llevar a cabo operaciones CRUD sobre la base de datos. En está extensión será necesario añadir una nueva funcionalidad, dedicada a la obtención de los productos a exportar para Brainsins.

4.3.1.1.5 Diaacceleratorstorefront

Esta extensión contiene la funcionalidad de la capa de presentación del sitio web. Será la encargada de responder a las peticiones en las que se solicite el fichero XML fruto de la exportación.

4.3.1.2 Integración Web

Se trata del conjunto de cambios generados en el site de DIA para la inclusión de los recomendadores en diferentes páginas. También al envío de la información oportuna a BrainSINS para que los productos recomendados sean los mejores.

La integración Web necesita que el catálogo de productos incluido en la plataforma de BrainSINS se actualice de forma diaria. De otro modo, los productos incluidos en sus recomendadores estarán desactualizados, lo que producirá que el precio mostrado sea

incorrecto, que se estén recomendando productos que no están disponibles en el sitio web, o bien que suceda el caso contrario.

4.3.1.2.1 Descripción de los Subsistemas

Se trata de integrar el sitio web DIA con el sistema BrainSINS. Esta integración, se llevará a cabo con la inclusión de la liberaría de BrainSINS, a través de la que se enviarán los datos necesarios a BrainSINS. Así como se recibirán los recomendadores.

En el modelo de los controladores, será necesario añadir la información de BrainSINS, que más tarde será recuperada por el javaScript dedicado al funcionamiento de BrainSINS, encargado de trabajar con la librería de BrainSINS.

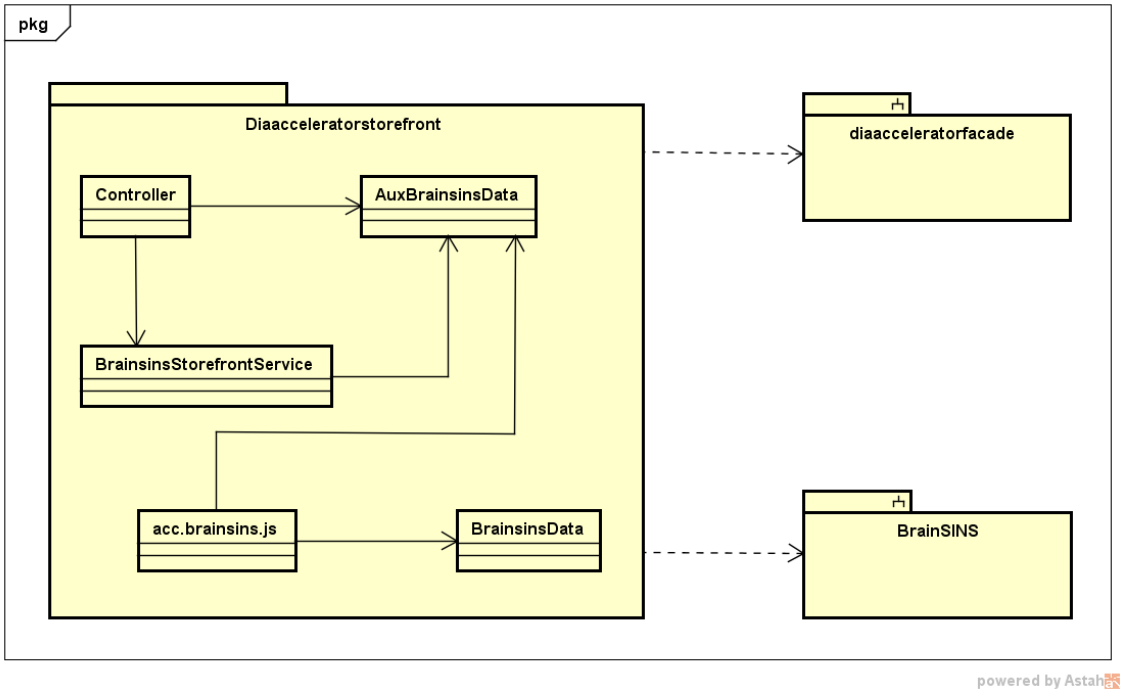


Ilustración 40: Diagrama de Clases Integración Web

4.3.1.2.2 Diaacceleratorstorefront

Esta extensión contiene la funcionalidad de la capa de presentación del sitio web. En ella se actualizará el modelo a través del servicio 'BrainsinsStorefrontService', con los datos a enviar a BrainSINS, encapsulados en el objeto 'AuxBrainsinsData'. En esta extensión también se llevará a cabo el envío de los datos a BrainSINS, a través del javascript 'acc.brainsins.js' que hará uso de librería proporcionada por BrainSINS. Se encargará de crear el objeto 'BrainSINSData' que recibirá el subsistema BrainSINS.

En algunos casos la información a incluir en el objeto model, deberá obtenerse a partir de la extensión 'diaacceleratorfacades', para seguir el patrón de diseño en capas. Dicha extensión será actualizada con los métodos necesarios para satisfacer toda la funcionalidad.

4.3.1.2.3 Diaacceleratorfacades

Esta extensión, está formada por las fachadas que se encargan de acceder a la capa de acceso a los datos del sistema.

4.3.1.2.4 BrainSINS

Representa al propio sistema BrainSINS, que recibirá la información adecuada en función de la página en la que se encuentre el usuario. Se encargará de enviar al 'diaacceleratorstorefront' los recomendadores a incluir en la página del sitio web.

4.3.1.3 Conclusión

Mediante la unión de estos sistemas y subsistemas, se compondrá el sistema final que permitirá dar cobertura a toda la funcionalidad planteada. Otro aspecto muy importante a tener en cuenta será cómo se llevará a cabo la comunicación entre los distintos subsistemas, aspecto que se trata en el siguiente apartado.

4.3.2 Descripción de los Interfaces entre Subsistemas

Una vez identificados los subsistemas, se pasará a describir cómo será la comunicación entre los mismos. En primer lugar, hay que destacar el hecho de que los distintos susbsistema se comunicarán por la red. BrainSINS recogerá el fichero resultado del cronjob por medio de una URL pública del site. El sitio web se comunicará con BrainSINS por medio de una API que nos proporcionan, a través de javaScript, que utiliza llamadas Ajax.

4.4 Diagrama de Clases Preliminar del Análisis

Gracias a los casos de uso y susbsistemas vistos anteriormente, en este apartado, se van a identificar las clases resultantes. Esta identificación se va a llevar a cabo de un modo muy esquemático, pues solo pretende ser una base para el posterior diseño.

Es necesario tener en cuenta, que parte del modelado utilizado es proporcionado directamente por la plataforma Hybris, como por ejemplo el modelado de productos, carrito, usuarios.... Estos serán explicados con detenimiento.

4.4.1 Diagrama de Clases Sistema Brainsins

4.4.1.1 Extensión diabrainsins

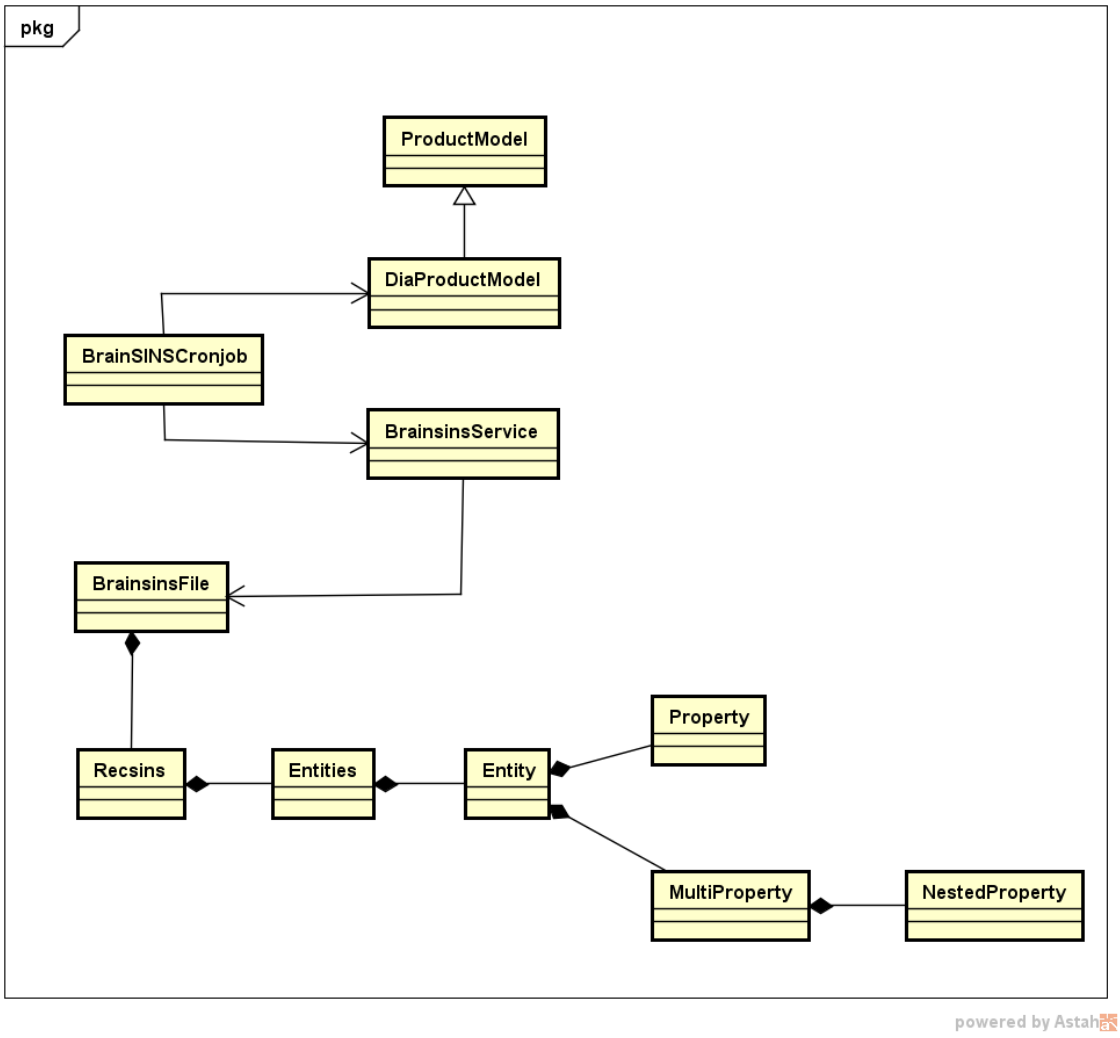


Ilustración 41: Diagrama de Clases Extensión diabrainsins

4.4.1.1.1 Descripción de las Clases

Las clases deberán estar organizadas en base a los subsistemas identificados anteriormente. Se relleanrá una tabla como la que se indica a continuación, por cada clase:

ProductModel
Descripción
Se trata de la clase base proporcionada por Hybris, encargadade representar los productos que forman parte del catálogo de productos.
Responsabilidades
En ella se incluyen los diferentes datos de los productos.
Atributos Propuestos
approvalStatus: se trata del estado del producto en el catálogo. Los artículos que se posean el

estado 'approved', serán los que se muestren. **averageRating**: media de las puntuaciones que el producto ha recibido por los usuarios del site **catalogVersion**: la versión del catálogo donde está incluido (staged/online) **code**: el identificador del producto **description**: la descripción del producto europe1Prices: Coleccion de PriceRowModel, el conjunto de precios que posee el producto, en función de la tienda en la que se oferta. **thumbnail**: la imagen en miniatura del producto, se trata de la imagen que va a mostrar el recomendador.
Métodos Propuestos

DiaProductModel
Descripción
Se trata de la clase que representa cada uno de los productos del catálogo del sitio web DIA.
Responsabilidades
En ella se incluyen datos de los productos específicos de este catálogo.
Atributos Propuestos
active: se trata de un booleano que indica si el producto se encuentra o no activo en el catálogo de productos. **warehousesForProduct**: se trata de un listado de los códigos de las tiendas donde se vende el producto. **webVisilibility**: se trata de un booleano que indica si está visible o no en la web.
Métodos Propuestos

BrainsinsFile
Descripción
Representa al fichero XML generado por el Cronjob.
Responsabilidades
Almacena el conjunto de los productos del catálogo online que se encuentran aprobados y que están disponibles en el catálogo online.
Atributos Propuestos
xml: una cadena con el conjunto de los productos formateada según el criterio de BrainSINS.
Métodos Propuestos

BrainsinsCronjob
Descripción
Contiene la lógica para la exportación del conjunto de productos del catálogo.
Responsabilidades
Ejecutar la lógica de exportación.
Atributos Propuestos
brainsinsService: Servicio encargado de convertir el listado de DiaProductModel en un string

siguiendo las especificaciones de BrainSINS. **configurationService**: Servicio encargado de acceder al fichero de configuración del site. **productService**: servicio encargado de buscar el conjunto de productos a exportar.
Métodos Propuestos
perform: método encargado de llevar a cabo la lógica.

Recsins
Descripción
Representa el nodo raíz del fichero xml donde se almacena la exportación de los productos. Pertenece al modelo del fichero.
Responsabilidades
Representar el nodo raíz del fichero.
Atributos Propuestos
entities: nodo hijo del nodo raíz recsins.
Métodos Propuestos

Entity
Descripción
Representa a cada uno de los nodos hijos del nodo entities. Representa cada uno de los productos exportados.
Responsabilidades
Contiene el conjunto de atributos que Brainsins necesita para generar los recomendadores de productos.
Atributos Propuestos
name: el nombre de la entidad, en nuestros casos, 'product'. **property**: el listado de las propiedades simples del producto. **multiproperty**: el listado de las propiedades compuestas del producto. Propiedades que engloban un conjunto de property.
Métodos Propuestos

Property
Descripción
Representa a cada uno de los atributos simples de los productos exportados.
Responsabilidades
Almacenar el nombre del atributo junto a su valor.
Atributos Propuestos
name: nombre del atributo del producto exportado. **value**: valor del atributo.
Métodos Propuestos

Multiproperty

Descripción
Representa aquellos atributos del producto que poseen más de un valor.
Responsabilidades
Almacenar el conjunto de valores del atributo.
Atributos Propuestos
name: nombre del atributo. **property**: el conjunto de las propiedades.
Métodos Propuestos

BrainsinsService
Descripción
Servicio encargado de integrar la aplicación con Brainsins
Responsabilidades
Convertir el listado de productos del catálogo en una cadena de caracteres formateada según las especificaciones de Brainsins y lista para ser guardada en el fichero XML.
Atributos Propuestos
Métodos Propuestos
convertProductosToXML: convertir el listado de productos en un XML, según las especificaciones de BrainSINS.

4.4.1.2 Extensión diaacceleratorfacades

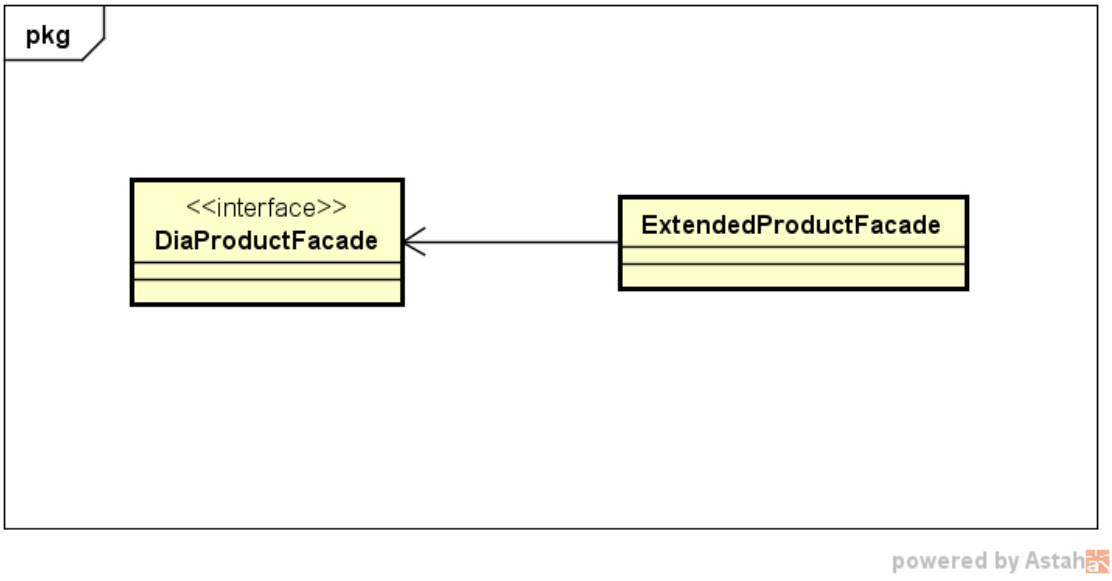


Ilustración 42: Diagrama Clases diaacceleratorfacades

DiaProductFacade
Descripción

Se trata de la interfaz de la fachada del servicio encargado, a través del que se llevan a cabo operaciones CRUD sobre la base de datos de los productos.
Responsabilidades
Interfaz de la fachada de productos
Atributos Propuestos
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

ExtendedProductFacade
Descripción
Implementacion de la interfaz DiaProductFacade
Responsabilidades
Llevar a cabo las operaciones CRUD sobre la base de datos de productos
Atributos Propuestos
extendedProductService: servicio que lleva a cabo operaciones CRUD sobre la base de datos de productos.
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

4.4.1.3 Extensión diaacceleratorcore

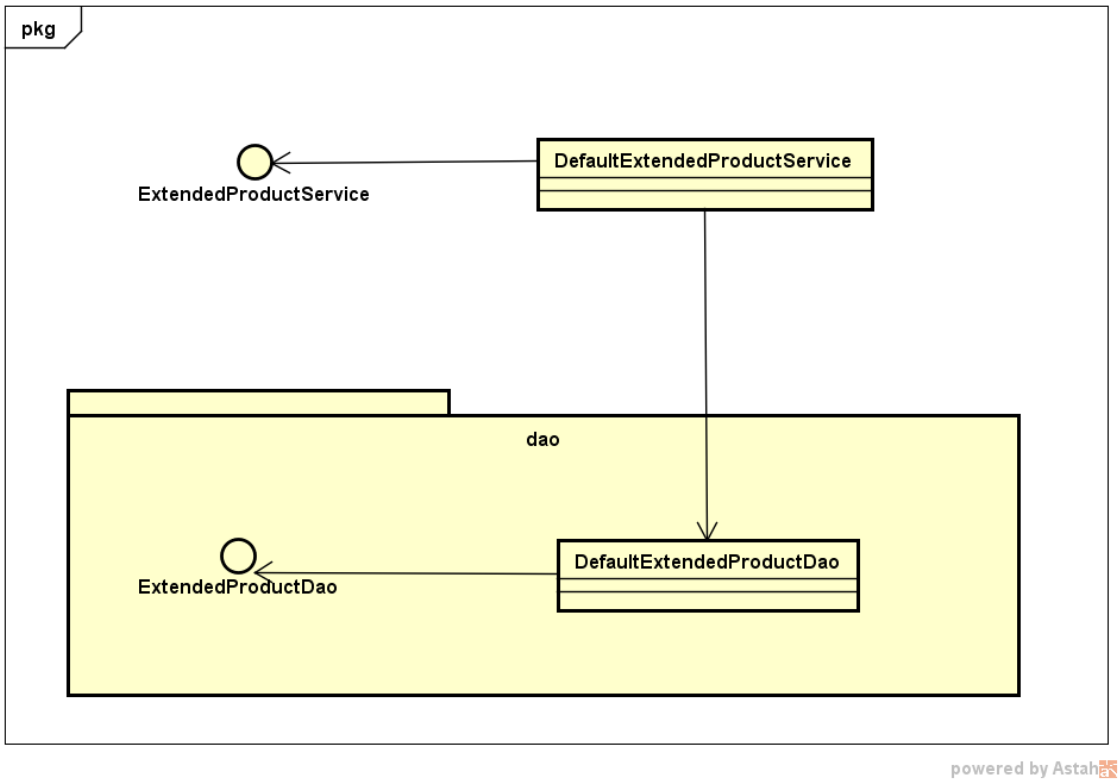


Ilustración 43: Diagrama Clases Extensión diaacceleratorcore

ExtendedProductService
Descripción
Interfaz del servicio CRUD de producto
Responsabilidades
Representa a la interfaz
Atributos Propuestos
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

DefaultExtendeProductService
Descripción
Llevar a cabo operaciones CRUD sobre la base de datos de productos
Responsabilidades
Representa a la interfaz
Atributos Propuestos
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

ExtendedProductDao
Descripción
Interfaz de clase desde donde se llevan a cabo las operaciones CRUD sobre la base de datos de productos.
Responsabilidades
Representa a la interfaz
Atributos Propuestos
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

DefaultExtendedProductDao
Descripción
Implementación de la interfaz 'ExtendedProductDao'.
Responsabilidades
Llevar a cabo las operaciones CRUD sobre la base de datos de productos.
Atributos Propuestos
Métodos Propuestos
findProductsToBrainsins: se encarga de buscar los productos del catálogo del site, a exportar para BrainSINS.

4.4.2 Diagrama de Clases Sistema Integración Web

4.4.2.1 Extensión diaacceleratorstorefront

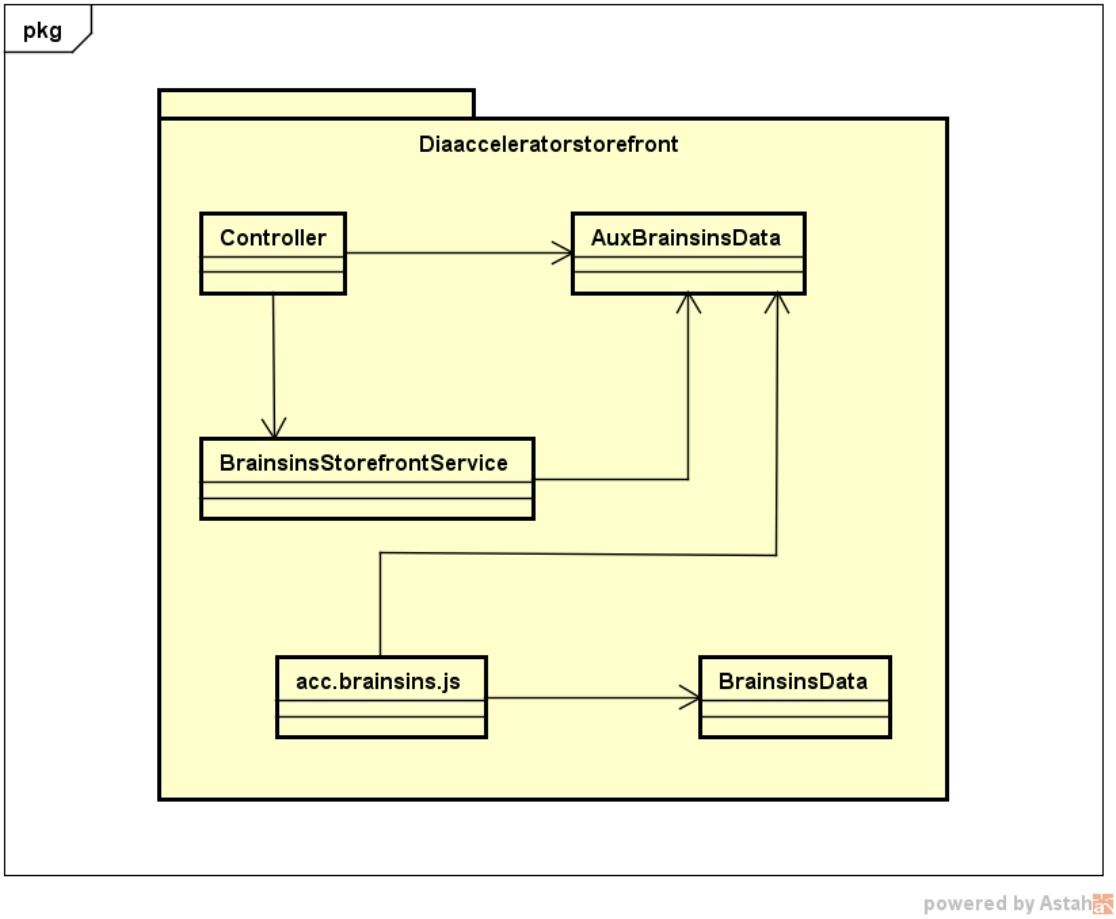


Ilustración 44: Diagrama de Clases Extensión diaacceleratorstorefront

Controller
Descripción
Recibe la petición del cliente .
Responsabilidades
Una vez recibida la petición, genera como respuesta la página solicitada.
Atributos Propuestos
brainsinsStorefrontService: actualiza el modelo de la página a enviar como respuesta, con el objeto AuxBrainsinsData.
Métodos Propuestos

BrainsinsStorefrontService
Descripción
Servicio dedicado a la integración de Brainsins en la web.

Responsabilidades
Se encarga de actualizar el modelo de la página con los datos a enviar a Brainsins.
Atributos Propuestos
Métodos Propuestos
UpdateModel: método encargado de actualizar el modelo de la página con la información de Brainsins.

AuxBrainsinsData

Descripción
Representa la información necesaria de enviar a Brainsins
Responsabilidades
Agrupar la información de Brainsins.
Atributos Propuestos
token: String que contiene el token identificador de Brainsins para el site de dia para el entorno: local, stag, dev, producción. **language**: String donde se incluye el código identificador de la tienda donde se encuentra el usuario. **userEmail**: String donde se incluye el email del usuario, en caso de que éste haya iniciado sesión en el sitio web. **productId**: String compuesto del código del producto que el usuario se encuentra visualizando. **totalAmount**: Double con el precio total del pedido. **typePage**: tipo de página que se encuentra visualizando el usuario. **categories**: Conjunto de categorías. **cart**: Hashmap <String, Long> mapa compuesto por cada uno de los producto que ha incluido el usuario en su carrito.
Métodos Propuestos
UpdateModel: método encargado de actualizar el modelo de la página con la información de Brainsins.

BrainsinsData

Descripción
Contiene la información a enviar a Brainsins
Responsabilidades
Incluir la información necesaria de enviar a Brainsins en función de la página y de los recomendadores a incluir
Atributos Propuestos
pageType: tipo de página que se encuentra visualizando el usuario. language: String donde se incluye el código identificador de la tienda donde se encuentra el usuario. **userEmail**: String donde se incluye el email del usuario, en caso de que éste haya iniciado sesión en el sitio web. **categories**: Conjunto de categorías. **cart**: mapa compuesto por cada uno de los producto que ha incluido el usuario en su carrito. **productId**: String compuesto del código del producto que el usuario se encuentra visualizando.

totalAmount: Double con el precio total del pedido. **recommenders**: listado de recomendadores a incluir en la página
Métodos Propuestos

acc.brainsins.js

Descripción
Clase javascript encargada de llevar a cabo la integración web de Brainsins
Responsabilidades
Se encarga de llevar a cabo la integración web de Brainsins, utilizando la librería proporcionada por Brainsins.
Atributos Propuestos
Métodos Propuestos

4.5 Análisis de Casos de Uso y Escenarios

4.5.1 Caso de Uso 1

Este caso de uso comprende tres acciones bien diferenciadas. Con el objetivo de facilitar su comprensión, se ha divido en tres diagramas, organizados en el orden de ejecución.

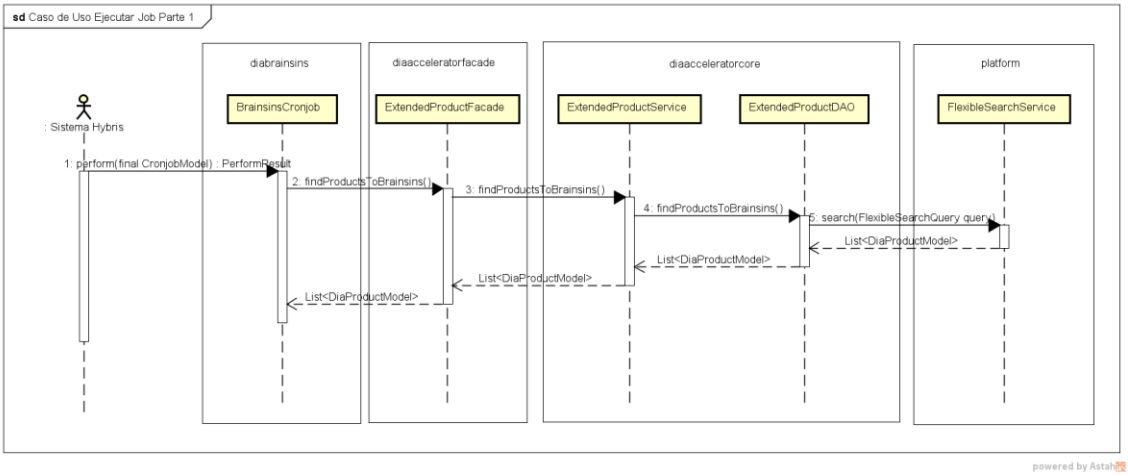


Ilustración 45: Diagrama Caso de Uso Ejecutar Job – Parte I

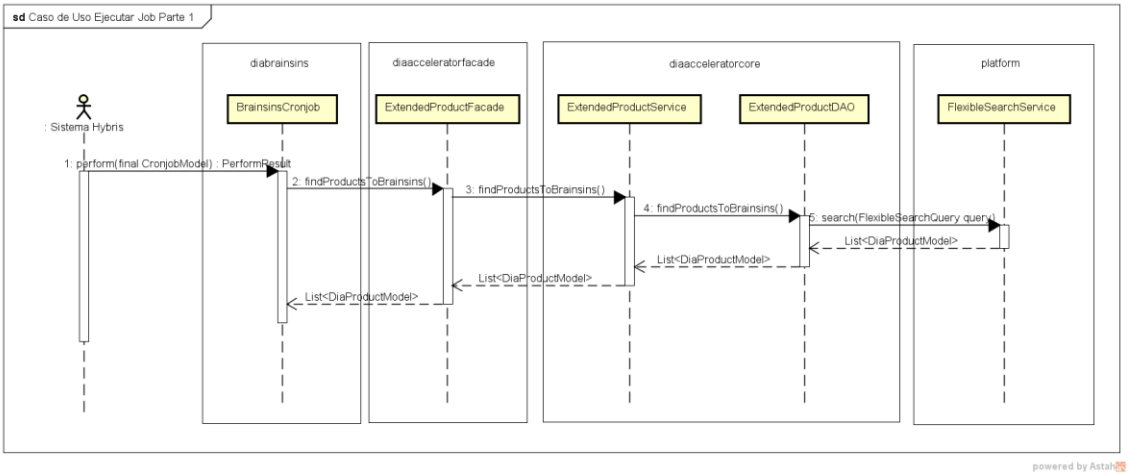


Ilustración 46: Diagrama Caso de Uso Ejecutar Job – Parte II

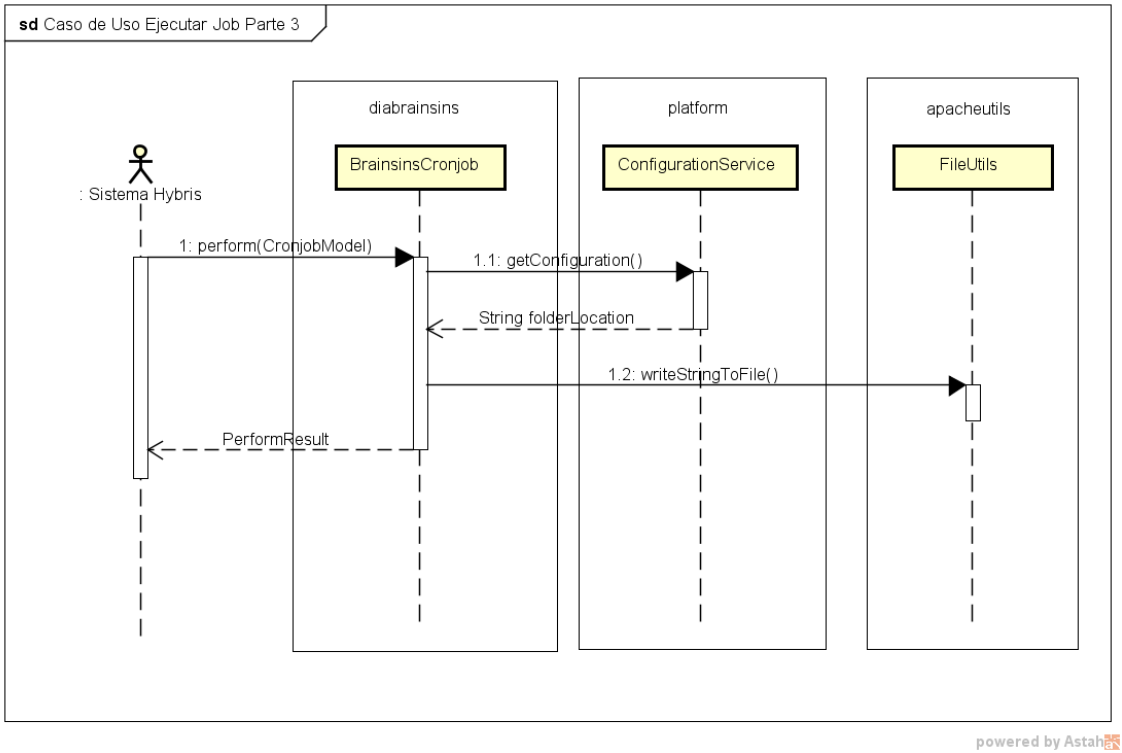


Ilustración 47: Diagrama Caso de Uso Ejecutar Job – Parte III

Ejecutar Job	
Alcance	Sistema Hybris
Nivel	User-goal
Actor principal	Sistema Hybris
Stakeholder e Intereses	• Sistema Hybris: necesita ejecutar el cronjob encargado de exportar el catálogo de productos a XML en una hora y fecha determinada.
Precondiciones	• El cronjob ha sido importado correctamente al sistema. • El trigger del cronjob ha sido configurado correctamente, para que se ejecute en una hora y fecha determinadas. • El cronjob se encuentra habilitado para su ejecución.
Postcondiciones	• El cronjob genera el fichero XML con los productos del catálogo online del sitio web. Productos candidatos a ser recomendados. • El fichero sigue la especificación de BrainSINS.
Escenario principal	1. En la fecha y hora indicadas en el trigger del cronjob, el sistema Hybris inicia la ejecución del cronjob. 2. El cronjob actualiza su estado a 'IN_PROGRESS'. 3. El sistema recoge la ubicación donde generará el fichero. 4. El sistema busca el listado de productos a exportar. 5. El sistema convierte el listado de productos a la especificación de BrainSINS.

	6. El sistema genera el fichero. 7. El cronjob actualiza su estado a 'FINISHED'.
Extensiones	

Excepciones	• 2a-7. La ruta de exportación del fichero no se encuentra especificada.
	  ○ El fallo se registra en el log del sistema.
	  ○ El cronjob actualiza su estado a 'ABORTED'.
	• 3a – 7. El sistema encuentra un fallo en la ejecución del cronjob.
	  ○ El fallo se registra en el log del sistema.
	  ○ El cronjob actualiza su estado a 'ABORTED'.
	• 3b - 7. El servidor se encuentra saturado.
	  ○ El fallo se registra en el log del sistema
	  ○ El cronjob actualiza su estado a 'ABORTED'.
	• 3c - 7. La base de datos no se encuentra disponible.
	  ○ El fallo se registra en el log del sistema.
	  ○ El cronjob actualiza su estado a 'ABORTED'.
	• 5a - 7 . Se produce un error durante la conversión.
	  ○ El fallo se registra en el log del sistema.
	  ○ El cronjob actualiza su estado a 'ABORTED'.
	• 6a – 7. Se produce un error en la generación del fichero.
	  ○ El fallo se registra en el log del sistema.
	  ○ El cronjob actualiza su estado a 'ABORTED'.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	Diariamente a las 7:30:00 AM
Misceláneo	El cronjob se ejecutará, una vez se haya actualizado la información de los productos.

4.5.2 Caso de Uso 2. Acceso a la página home

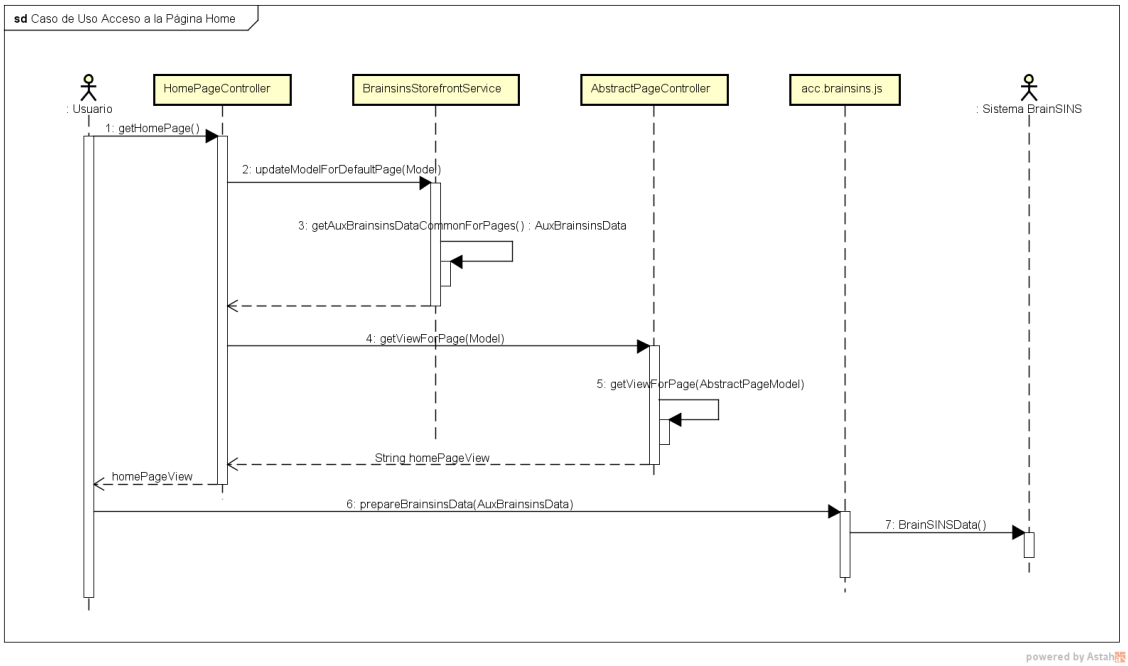


Ilustración 48: Caso de Uso Acceso página home

Acceso a la página home	
Alcance	Sistema BrainSINS
Nivel	User-goal
Actor principal	• Usuario, cliente del sitio web.
Stakeholder e Intereses	• Usuario:accede a la página home. • Envío de información a BrainSINS asociada a la página home.
Precondiciones	
Postcondiciones	• BrainSINS recibe la información asociada a la página home.
Escenario principal	1. El usuario solicita la página home del sitio web. 2. El controlador recibe la petición. 3. El sistema solicita los datos de BrainSINS asociados a la página home. 4. El sistema actualiza el modelo de la página a mostrar con los datos de BrainSINS. 5. El sistema muestra la página home. 6. El sistema envía a BrainSINS la información asociada a la página home: el token del sitio web, el email del usuario, en caso de que haya iniciado sesión, el identificador de la tienda donde se encuentra el usuario, el tipo de página 'home'.
Extensiones	

Excepciones	• 1a – 6. El servidor se encuentra saturado.
	  ○ Notificación del error al usuario.
	• 5a – 6. La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error al usuario.
	• 6a – 6. El sistema BrainSINS no se encuentra disponible.
	  ○ No recibirá los datos.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.3 Caso de Uso 3. Acceso Página Detalle de Producto

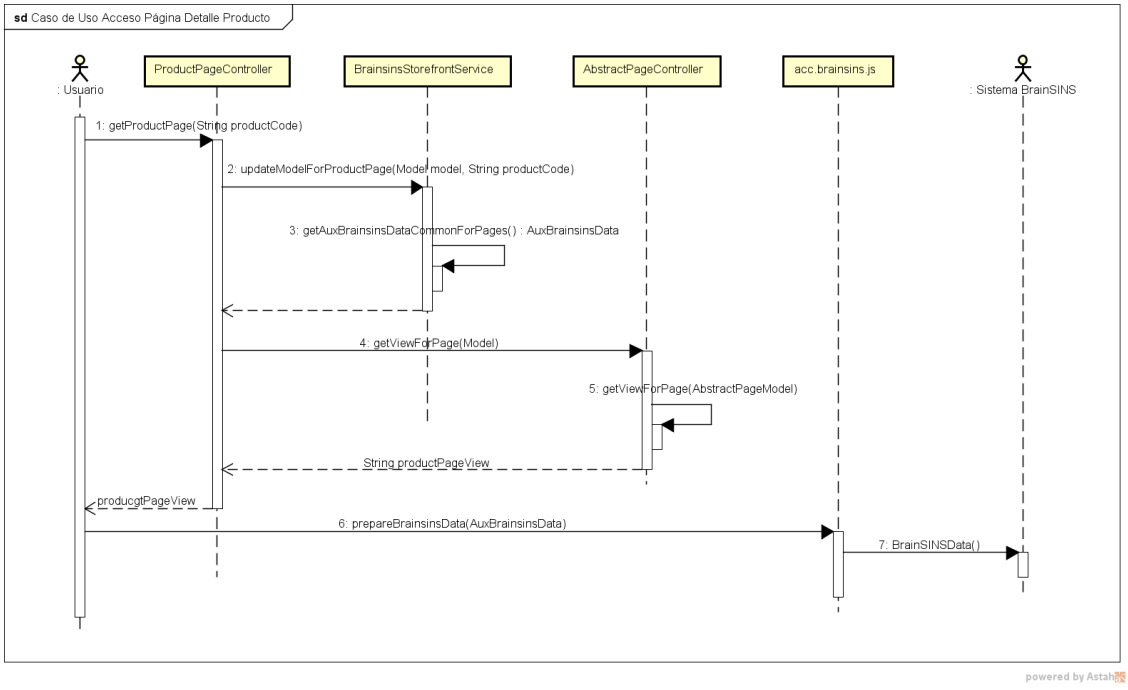


Ilustración 49: Caso de Uso Acceso página detalle de producto

Acceso a la página de detalle de producto	
Alcance	Sistema BrainSINS
Nivel	User-goal

Actor principal	• Usuario, cliente del sitio web.
Stakeholder e Intereses	• Usuario:accede a la página detalle de producto
Precondiciones	
Postcondiciones	• BrainSINS recibe la información asociada a la página de producto.
Escenario principal	1. El usuario solicita la página de un producto. 2. El controlador recibe la petición. 3. El sistema solicita los datos de BrainSINS asociados a la página de detalle de un producto. 4. El sistema actualiza el modelo de la página a mostrar con los datos de BrainSINS. 5. El sistema muestra la página del producto. 6. El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página, 'product', el email del usuario en caso de que haya iniciado sesión, el identificador del producto, el identificador de la tienda.
Extensiones	

Excepciones	• 1a – 6. El servidor se encuentra saturado.
	  ○ Notificación del error al usuario.
	• 5a – 6. La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error al usuario.
	• 6a – 6. El sistema BrainSINS no se encuentra disponible.
	  ○ No recibirá los datos.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.4 Caso de Uso 4. Acceso Página Categorías

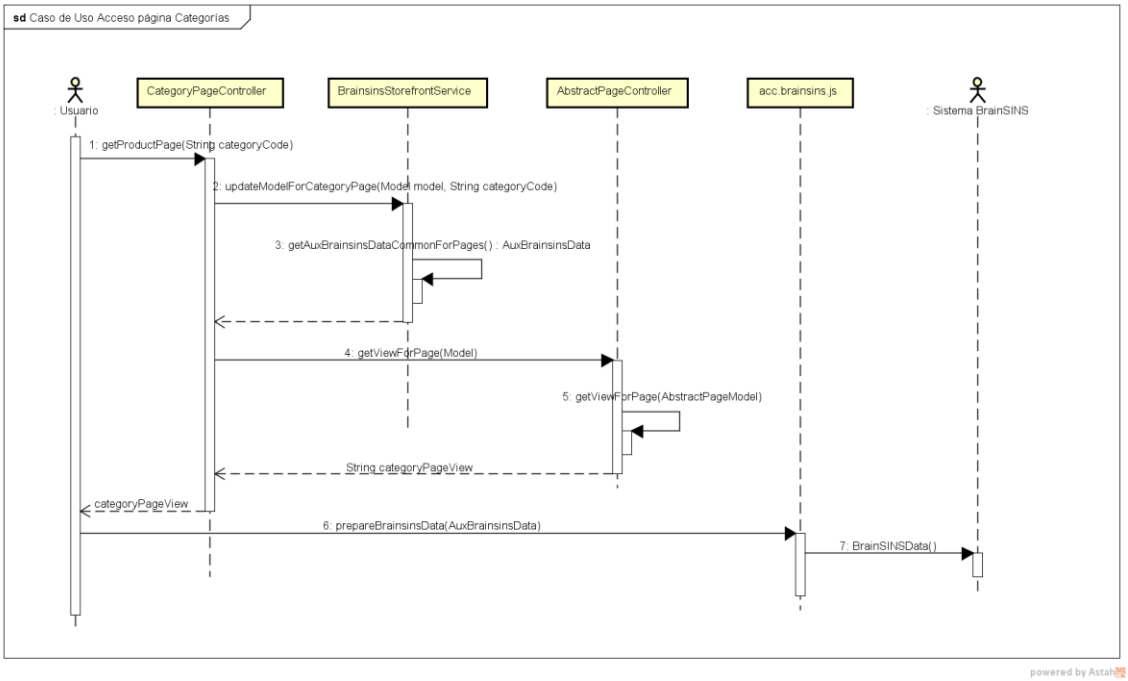


Ilustración 50: Caso de Uso Acceso página categorías

Acceso a la página de detalle de categorías	
Alcance	Sistema BrainSINS
Nivel	User-goal
Actor principal	• Usuario, cliente del sitio web.
Stakeholder e Intereses	• Usuario:accede a la página detalle de categorías
Precondiciones	
Postcondiciones	• BrainSINS recibe la información asociada a la página de categorías.
Escenario principal	1. El usuario selecciona en el menú principal la categoría de la que desea visualizar los productos ofertados. 2. El controlador recibe la petición. 3. El sistema solicita los datos de BrainSINS asociados a la página de detalle de categorías. 4. El sistema actualiza el modelo de la página a mostrar con los datos de BrainSINS. 5. El sistema muestra la página. 6. El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página, “category”, el email del usuario en caso de que haya iniciado sesión, la categoría del producto que ha buscado, el identificador de la tienda.

Extensiones	

Excepciones	• 1a – 6. El servidor se encuentra saturado.
	  ○ Notificación del error al usuario.
	• 5a – 6. La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error al usuario.
	• 6a – 6. El sistema BrainSINS no se encuentra disponible.
	  ○ No recibirá los datos.
	  ○ Notificación del error al usuario.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.5 Caso de Uso 5. Acceso a la página del carrito

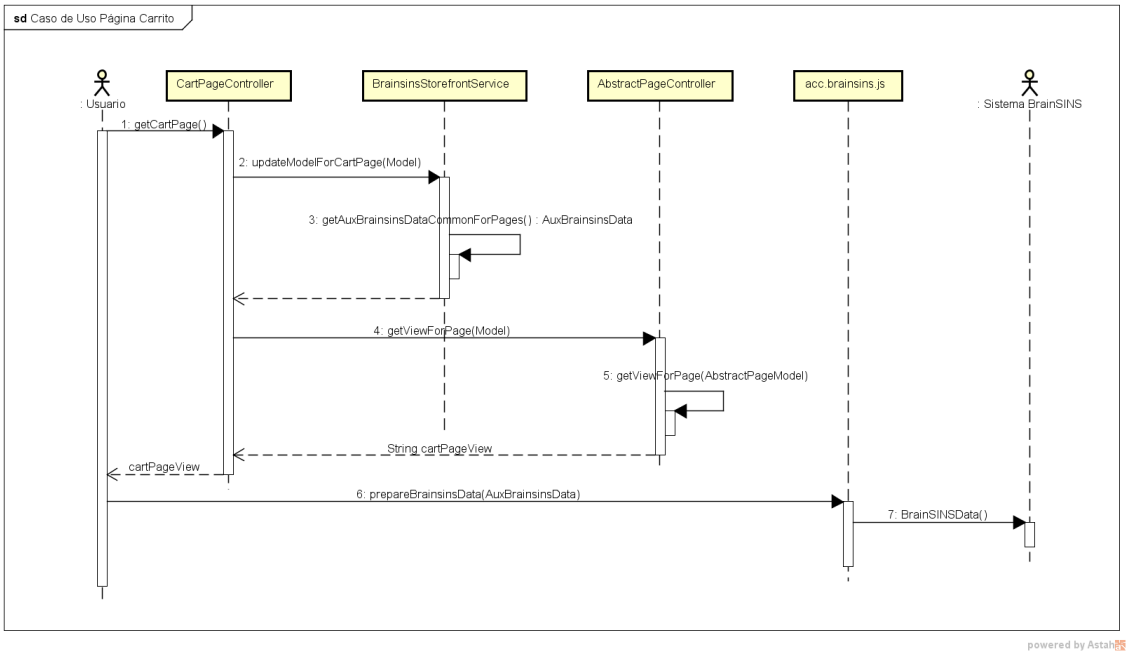


Ilustración 51: Caso de Uso Acceso página del carrito

Acceso a la página de detalle del carrito	
Alcance	Sistema BrainSINS
Nivel	User-goal

Actor principal	• Usuario, cliente del sitio web.
Stakeholder e Intereses	• Usuario:accede a la página detalle de categorías
Precondiciones	• El usuario ha iniciado sesión.
Postcondiciones	• BrainSINS recibe la información asociada a la página del carrito.
Escenario principal	1. El usuario solicita la página de su carrito. 2. El controlador recibe la petición. 3. El sistema solicita los datos de BrainSINS asociados a la página de carrito. 4. El sistema actualiza el modelo de la página a mostrar con los datos de BrainSINS. 5. El sistema muestra la página. 6. El sistema notifica a BrainSÍNS el acceso, incluyendo el tipo de página, "checkout", el email del usuario en caso de que haya iniciado sesión, el conjunto de productos incluidos en el carrito, así como sus precios, así como el identificador de la tienda.
Extensiones	

Excepciones	• 1a – 6. El servidor se encuentra saturado.
	  ○ Notificación del error al usuario.
	• 5a – 6. La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error al usuario.
	• 6a – 6. El sistema BrainSINS no se encuentra disponible.
	  ○ No recibirá los datos.
	  ○ Notificación del error al usuario.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.6 Caso de Uso 6. Página de Confirmación de Pedido

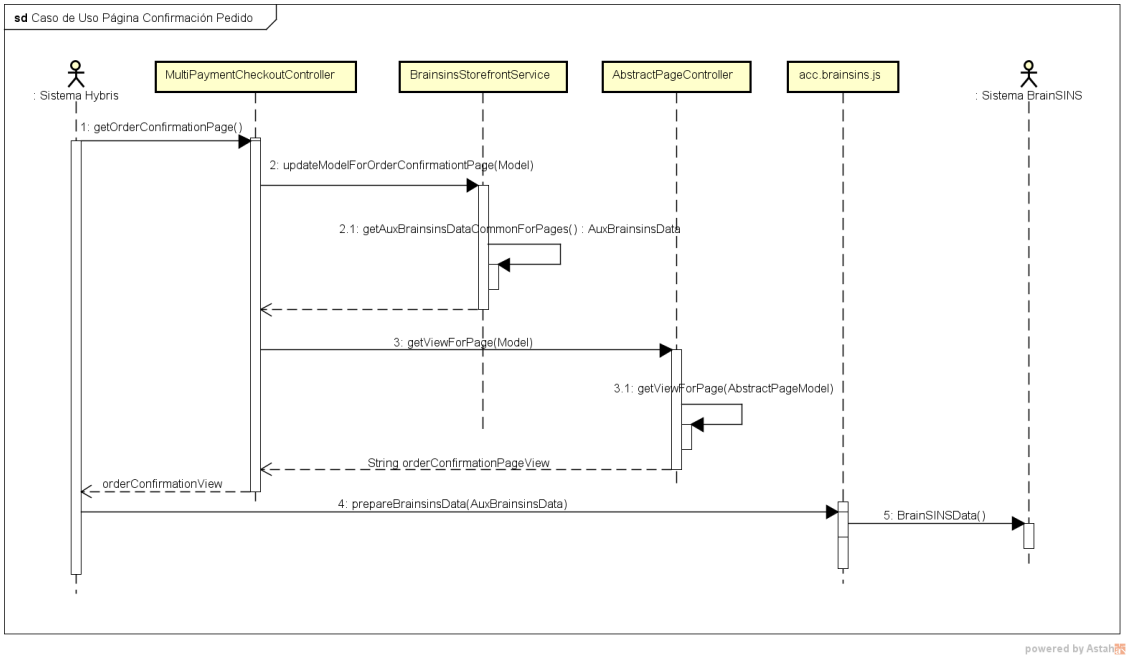


Ilustración 52: Caso de Uso Acceso página confirmación pedido

Acceso a la página confirmación de pedido	
Alcance	Sistema BrainSINS
Nivel	User-goal
Actor principal	• Sistema Hybris
Stakeholder e Intereses	• Sistema Hybris: mostrar al usuario la página de confirmación de su pedido
Precondiciones	• El usuario ha iniciado sesión en el sitio web. • El usuario ha completado todos los pasos del checkout de forma satisfactoria.
Postcondiciones	• BrainSINS recibe la información asociada a la página de confirmación de pedido.
Escenario principal	1. El sistema solicita la página de confirmación del pedido del usuario. 2. El controlador recibe la petición. 3. El sistema solicita los datos de BrainSINS asociados a la página de carrito. 4. El sistema actualiza el modelo de la página a mostrar con los datos de BrainSINS. 5. El sistema muestra la página. 6. El sistema notifica a BrainSÍNS el acceso, incluyendo el tipo de página,

	"thankYou", el email del usuario,el precio total de la compra, así como el identificador de la tienda.
Extensiones	

Excepciones	• 1a – 6. El servidor se encuentra saturado.
	  ○ Notificación del error al usuario.
	• 5a – 6. La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error al usuario.
	• 6a – 6. El sistema BrainSINS no se encuentra disponible.
	  ○ No recibirá los datos.
	  ○ Notificación del error al usuario.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.7 Caso de Uso 7. Añadir Producto al Carrito

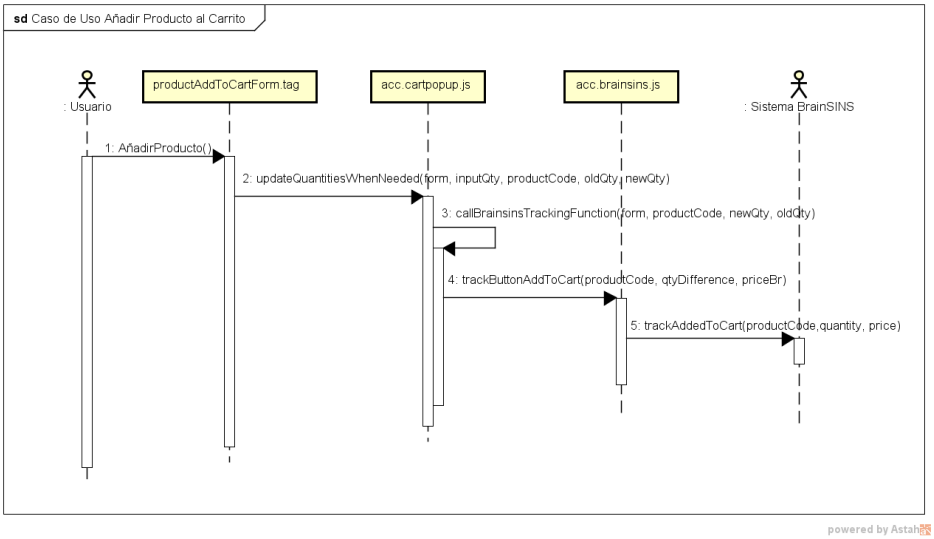


Ilustración 53: Caso de Uso Añadir producto al carrito

Añadir producto al carrito	
Alcance	Sitio Web
Nivel	User-goal

Actor principal	• El usuario selecciona el botón “Añadir” de uno de los productos.
Stakeholder e Intereses	• Añadir un nuevo producto a su carrito.

Precondiciones	• La página muestra productos con la opción de añadirlos al carrito, por medio de alguna de las siguientes opciones:
	  ○ Los productos a añadir se muestran en un recomendador de Brainsins. Cada producto incluye una simulación del botón añadir que se muestra en el resto de productos del sitio web.
	  ○ Recomendador de productos Hybris.
	  ○ Productos a un listado.

Postcondiciones	• El producto es añadido correctamente al carrito del usuario.
Escenario principal	1. El usuario hace clic sobre el botón añadir al carrito de un producto. 2. El sistema comprueba la disponibilidad del producto. 3. El sistema añade el producto en el carrito del usuario. 4. El sistema lleva a cabo el recálculo del total del carrito. 5. El sistema actualiza el minicarrito del usuario. 6. El sistema cambia el formulario de interacción con el carrito, ocultando el texto “Añadir” y mostrando los botones '+' y '-', junto a la cantidad añadida al carrito, en este caso, una unidad. 7. El sistema envía a Brainsins la información del producto.
Extensiones	

Excepciones	• 3a -7. El producto no se encuentra disponible en la tienda.
	  ○ Se notifica al usuario con un mensaje.
	  ○ Se vuelve al paso 1.
	• 2a – 7. La base de datos no se encuentra disponible.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.
	• 5a – 7. Se produce un error al actualizar el carrito del usuario.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.8 Caso de Uso 8. Añadir unidades al carrito

Añadir unidades al carrito

Alcance	Sitio Web
Nivel	User-goal
Actor principal	• El usuario del sitio Web
Stakeholder e Intereses	• El usuario desea añadir una unidad más del producto recomendado al carrito.

Precondiciones	• La página muestra productos con la opción de añadirlos al carrito, por medio de alguna de las siguientes opciones:
	  ○ Los productos a añadir se muestran en un recomendador de Brainsins. Cada producto incluye una simulación del botón añadir que se muestra en el resto de productos del sitio web.
	  ○ Recomendador de productos Hybris.
	  ○ Productos a un listado.
	• El usuario contiene en su carrito una unidad del producto del que desea añadir otra unidad más.

Postcondiciones	• Se añade una unidad más del producto al carrito del usuario.
Escenario principal	1. El usuario hace clic sobre el botón '+' de un producto. 2. El sistema comprueba la disponibilidad del producto. 3. El sistema añade el producto en el carrito del usuario. 4. El sistema lleva a cabo el recálculo del total del carrito. 5. El sistema actualiza el minicarrito del usuario. 6. El sistema actualiza el campo del formulario del producto donde se indica la cantidad con una unidad más. 7. El sistema envía a Brainsins la información del producto.
Extensiones	

Excepciones	• 3a -7. El producto no se encuentra disponible en la tienda.
	  ○ Se notifica al usuario con un mensaje.
	  ○ Se vuelve al paso 1.
	• 2a – 7. La base de datos no se encuentra disponible.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.
	• 5a – 7. Se produce un error al actualizar el carrito del usuario.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.9 Caso de Uso 9. Eliminar Unidades del Carrito

Eliminar unidades del carrito	
Alcance	Sitio Web
Nivel	User-goal
Actor principal	• El usuario del sitio Web
Stakeholder e Intereses	• El usuario desea eliminar una unidad del producto recomendado de su carrito.

Precondiciones	• La página muestra productos con la opción de añadirlos al carrito, por medio de alguna de las siguientes opciones:
	  ○ Los productos a añadir se muestran en un recomendador de Brainsins. Cada producto incluye una simulación del botón añadir que se muestra en el resto de productos del sitio web.
	  ○ Recomendador de productos Hybris.
	  ○ Productos a un listado.
	• El usuario contiene en su carrito el producto del que desea eliminar unidades.

Postcondiciones	• Se elimina una unidad del producto recomendado.
Escenario principal	1. El usuario hace clic sobre el botón '-' de un producto recomendado. 2. El sistema disminuye en una unidad el producto en el carrito del usuario, o bien lo elimina, si la resta llega a ser cero. 3. El sistema lleva a cabo el recálculo del total del carrito. 4. El sistema actualiza el minicarrito del usuario. 5. El sistema actualiza el campo del formulario del producto donde se indica la cantidad con una unidad menos, o bien, si la cantidad es cero, muestra de nuevo el botón “Añadir”. 6. El sistema envía a Brainsins la información del producto.
Extensiones	

Excepciones	• 2a – 6. La base de datos no se encuentra disponible.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.
	• 5a – 6. Se produce un error al actualizar el carrito del usuario.
	  ○ Se notifica el error al usuario con un mensaje.
	  ○ Se vuelve al paso 1.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.5.10 Caso de Uso 10. Inclusión de Recomendador

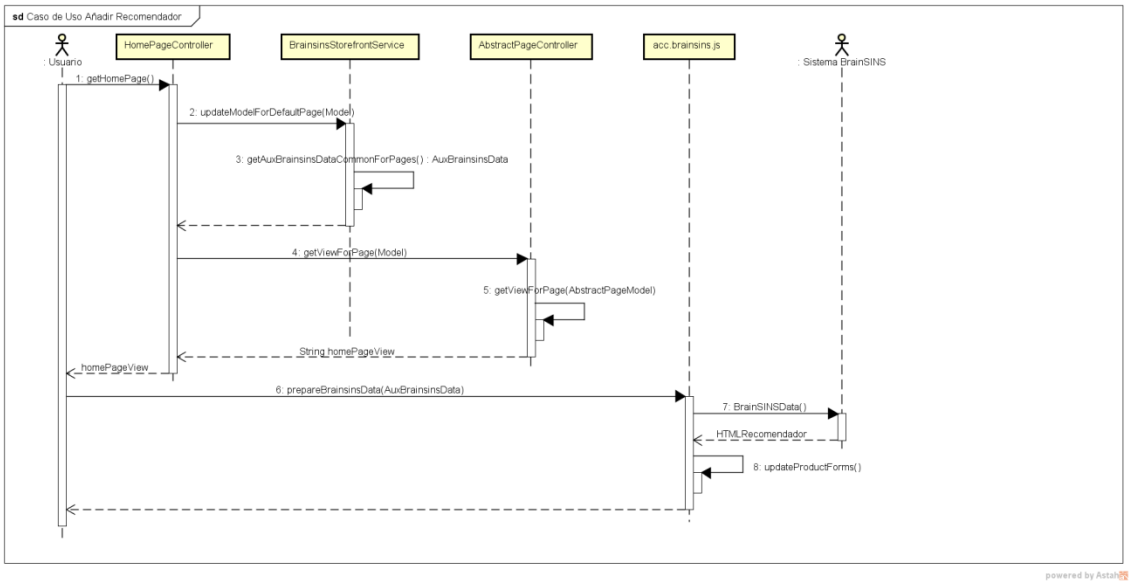


Ilustración 54: Caso de Uso Inclusión de recomendador

Inclusión del recomendador en la página	
Alcance	Sistema BrainSINS
Nivel	System-goal
Actor principal	• Sitio Web
Stakeholder e Intereses	• Mostrar en el vista un carrusel de productos recomendados de BrainSINS.
Precondiciones	• La vista incluye el código HTML donde incluir un recomendador de BrainSINS.
Postcondiciones	• La página muestra el recomendador de BrainSINS.
Escenario principal	1. El usuario solicita una página. 2. El controlador recibe la petición 3. El sistema solicita los datos de BrainSINS asociados a la página mostrada. 4. El sistema actualiza el modelo de la página a mostrar 5. El sistema muestra la página. 6. El sistema envía a BrainSINS los datos asociados de la página, además de enviar los identificadores HTML de los recomendadores donde será incluídos, así como los identificadores numéricos de los recomendadores. 7. El sistema BrainSINS envía el código HTML de los recomendadores. 8. El sistema incluye los recomendadores. 9. El sistema añade a cada uno de los productos recomendados el

	formulario de "Añadir al carrito", con la apariencia del resto de botones del site.
Extensiones	

Excepciones	• El sistema BrainSINS no se encuentra disponible.
	  ○ La página no muestra ningún recomendador.
	• La página no se encuentra disponible: fallo al intentar acceder a la página.
	  ○ Notificación del error.

Tecnología y variaciones de datos	N/a
Frecuencia de ocurrencia	
Misceláneo	

4.6 Análisis de Interfaces de Usuario

En este apartado, se va a llevar a cabo una descripción de la interfaz de usuario, que será incluída en diferentes páginas del sitio web.

4.6.1 Descripción de la Interfaz

La interfaz consiste en carruseles de productos recomendados por BrainSINS, que serán incluidos en diferentes páginas del site. Estos recomendadores tendrán el mismo aspecto que los recomendadores proporcionados por el sistema Hybris.

Cada recomendador se representa como un carrusel de productos. Por cada uno de los productos recomendados se muestra su imagen, su descripción, la valoración de los usuarios, el precio del producto en la tienda en la que se encuentra ubicado el usuario. Además, incluirá la funcionalidad de añadir productos al carrito, al igual que muestran en los recomendadores de Hybris.

La funcionalidad del formulario no es directamente proporcionada por el sistema BrainSINS. De este modo, una vez el sitio web ha recibido la respuesta de BrainSINS con los productos recomendados, será incluida la funcionalidad en cada uno de los productos.

4.6.2 Descripción del Comportamiento de la Interfaz

En este apartado se van a tratar varios aspectos referentes al comportamiento de la interfaz, las diferentes acciones que puede llevar a cabo el usuario con el recomendador y los productos recomendados.

El recomendador mostrará en una misma fila un conjunto de ocho productos recomendados. Se harán visibles aquellos que se consideren oportunos en función del espacio del que disponga dentro de la página. El usuario podrá visualizar aquellos productos que se encuentren ocultos, por medio de su interacción con dos botones ubicados en los extremos.

Desde cada producto recomendado, se podrá acceder a la página de detalle del producto por medio de su imagen y de su descripción.

El usuario podrá añadir los productos recomendados a su carrito a través del botón “Añadir”, o bien, por medio de los botones '+', '-', en el caso de que el producto haya sido incluido previamente en el carrito.

4.6.3 Diagrama de Navegabilidad

Desde los productos recomendados, se puede acceder a la página del detalle de producto haciendo clic sobre su imagen, o bien sobre su descripción.

4.7 Especificación del Plan de Pruebas

A continuación se expondrá el diseño del plan de pruebas de la aplicación. Dicho plan de pruebas estará separado en varios tipos de pruebas posibles:

- Unitarias
- De integración
- De sistema
- De usabilidad

4.7.1 Pruebas Unitarias

Clase: BrainSINSCronjob	
Prueba	**Resultado Esperado**
Ejecutar Job sin tener configurada la ubicación del xml de exportación	El sistema registra el inicio de la ejecución del cronjob. El sistema registra un error donde se indica la falta de la configuración del fichero de exportación del cronjob. El cronjob cambia su estado a 'ABORTED'. Se finaliza la ejecución del cronjob.
El sistema registra el inicio de la ejecución del cronjob.	El sistema registra el inicio de la ejecución del cronjob. El sistema recupera la ubicación del fichero de exportación. El sistema registra el error al intentar acceder a la base de datos. El cronjob cambia su estado a 'ABORTED'. El cronjob finaliza su ejecución.
El sistema registra un error donde se indica la falta de la configuración del fichero de exportación del cronjob.	El sistema registra el inicio de la ejecución del cronjob. El sistema recupera la ubicación del fichero de exportación. El sistema consulta los productos a exportar. El sistema convierte los productos al formato Brainsins. El sistema genera el fichero de exportación. El sistema cambia su estado a 'SUCCESS'. El cronjob finaliza su ejecución.

4.7.2 Pruebas de Integración y Sistema

Caso de Uso 1: Ejecutar Job	
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob, encontrándose el mismo correctamente configurado y estando disponible la base de datos	El sistema genera un fichero XML con la exportación del catálogo de productos, siguiendo la especificación de BrainSINS en la localización configurada.
El usuario / sistema ejecuta el cronjob sin tener configurado la ubicación del fichero de la exportación.	El sistema registra el problema encontrado en su log. El sistema cambia su estado a 'ABORTED'.
El usuario / sistema ejecuta el	El sistema registra el problema encontrado en su log. El

cronjob cuando la base de datos no se encuentra disponible.	sistema cambia su estado a 'ABORTED'.

Caso de Uso 2: Acceso Página Home	
Prueba	**Resultado Esperado**
Un cliente que ha iniciado sesión, solicita la página home, encontrándose ésta disponible.	Un cliente que ha iniciado sesión, solicita la página home, encontrándose ésta disponible.
Un cliente que ha iniciado sesión, solicita la página home, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
Un usuario que no ha iniciado sesión, solicita la página home, encontrándose ésta disponible.	El sistema mostrará la página home. Se notifica a BrainSINS el acceso a la página, enviando el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'home'.
Un usuario que no ha iniciado sesión, solicita la página home, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.

Caso de Uso 3: Acceso Página Detalle de Producto	
Prueba	**Resultado Esperado**
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta disponible.	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'product'.
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta disponible.	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'product'.
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.

Caso de Uso 4: Acceso Página de Categorías	
Prueba	**Resultado Esperado**

El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página disponible.	El sistema muestra los productos asociados a dicha categoría. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'category'.
El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página disponible.	El sistema muestra los productos asociados a dicha categoría. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'category'.
El cliente que no ha iniciado sesión, solicita la página de una categoría, estando la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible

Caso de Uso 5: Acceso Página del Carrito	
Prueba	**Resultado Esperado**
El cliente que ha iniciado sesión, solicita acceder a su carrito, encontrándose la página disponible.	El sistema muestra la página El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página 'checkout', el email del usuario, el conjunto de productos incluidos en el carrito, así como sus precios y el identificador de la tienda.
El cliente que ha iniciado sesión, solicita acceder a su carrito, encontrándose la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
El cliente que no ha iniciado sesión, solicita acceder a su carrito, encontrándose la página disponible.	El sistema muestra la página El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página 'checkout', el conjunto de productos incluidos en el carrito, así como sus precios y el identificador de la tienda.
El cliente que no ha iniciado sesión, solicita acceder a su carrito, encontrándose la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible

Caso de Uso 6: Acceso Página Confirmación de Pedido	
Prueba	**Resultado Esperado**
El cliente finaliza su compra satisfactoriamente, encontrándose la página de confirmación disponible.	El cliente finaliza su compra satisfactoriamente, encontrándose la página de confirmación disponible.
El cliente finaliza su compra satisfactoriamente, encontrándose la página de confirmación no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible

Caso de Uso 7: Añadir Producto al Carrito	
Prueba	**Resultado Esperado**
El cliente añade un producto del que existe stock en su carrito.	El sistema añade el producto en el carrito del usuario. Se recalcula el total del pedido. El sistema cambia el formulario de interacción con el carrito ocultando el botón “Añadir” y mostrando los botones '+' y '-' junto a las unidades añadidas al carrito. El sistema envía a BrainSINS el código de producto añadido, la unidad añadida y su precio.
El cliente solicita añadir un producto en su carrito del que no existe stock.	El sistema notifica al usuario que no existe stock. El sistema no altera el carrito del usuario. El producto continúa mostrando el botón 'Añadir'.

Caso de Uso 8: Añadir Unidades al Carrito	
Prueba	**Resultado Esperado**
El usuario selecciona el botón '+' de un producto del que existe stock	El sistema añade el producto en el carrito del usuario. Se recalcula el total del pedido. El sistema cambia el formulario de interacción con el carrito actualizando las unidades del producto incluidas en el carrito. El sistema envía a BrainSINS el código del producto añadido, la unidad añadida y su precio.
El usuario selecciona el botón '+' de un producto del que no existe stock	El sistema notifica al usuario que no existe stock. El sistema no altera el carrito del usuario . El sistema no altera el formulario.

Caso de Uso 9: Eliminar Unidades del Carrito	
Prueba	**Resultado Esperado**
El usuario selecciona el botón '-' de un producto.	El sistema decrementa en una unidad el producto en el carrito del usuario, en el caso de que sea cero, elimina el producto del carrito. Se recalcula el total del pedido. El sistema cambia el formulario de interacción con el carrito actualiza las unidades del producto incluidas en el carrito, o en el caso de que sea cero, oculta los botones '+' y '-' y muestra de nuevo el botón 'Añadir'. El sistema envía a BrainSINS el código del producto, la unidad decrementada y su precio.

Caso de Uso 10: Inclusión Recomendador	
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de un recomendador.	Se envía a BrainSINS la información asociada la página, junto al identificador del elemento HMTL donde incluir el recomenador, así como el identificador numérico del recomendador. El recomenador se incluye en el lugar especificado. A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.
La página mostrada incluye el código HTML para la	Se envía a BrainSINS la información asociada la página,junto al identificador del elemento HMTL donde incluir el

inclusión de un recomendador, encontrándose el sistema BrainSINS no disponible.	recomenador, así como el identificador numérico del recomendador. BrainSINS no recibe la información. El recomendador no se incluye.
La página mostrada incluye el código HTML para la inclusión de dos recomendadores, teniendo ambos el mismo identificador HTML	Se envía a BrainSINS la información asociada la página, junto a los identificadores de los elementos HTML donde incluir los recomenadores, así como su identificador numérico. Dado que ambos elementos HTML tienen el mismo identificador, sólo se incluirá en la página el primero de ellos. El recomenador se incluye en el lugar especificado. A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.
La página mostrada incluye el código HTML para la inclusión de dos recomendadores, teniendo ambos distinto identificador HTML, pero mismo identficador numérico.	Se envía a BrainSINS la información asociada la página, junto a los identificadores de los elementos HTML donde incluir los recomenadores, así como su identificador numérico. Dado que ambos elementos tienen el mismo identificador numérico, sólo se incluirá el primero de ellos. El recomenador se incluye en el lugar especificado. A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.

4.7.3 Pruebas de Usabilidad

En este apartado se especificarán los aspectos más importantes sobre la usabilidad:

El diseño de la interfaz y la interacción de la misma es coherente a través de las distintas pantallas. **[Hassan08].**

E look & feel se corresponde con los objetivos y servicios que ofrece**. [Hassan08].**

Se tratará de fomentar el reconocimiento frente a la memorización a la hora de utilizarlo. **[Nielsen94]**.

Se tratará de usar un diseño minimalista **[Nielsen94]**, evitando cualquier ruido que pueda dificultarle al usuario la utilización de la página **[Krug01]**.

La respuesta del sistema antes las interacciones del usuario deberá se predecible **[Hassan08]**.

Las fotografías se recortarán de manera adecuada y cuidando su resolución **[Hassan08]**.

Cuando se produce un error se informará al usuario de forma clara y no alarmista del problema y su solución **[Hassan08]**.

Capítulo 5. Diseño del Sistema

5.1 Arquitectura del Sistema

A lo largo de los siguientes apartados, se detallará la arquitectura del sistema a desarrollar, desde diferentes perspectivas, con el fin de que sirva de base para su posterior implementación.

5.1.1 Diagramas de extensiones

El sistema Hybris, lleva a a cabo la división de funcionalidades por medio de extensiones. A su vez estas extensiones tendrán una división en paquetes. A continuación se presenta un diagrama de alto nivel las extensiones que componen el sistema.

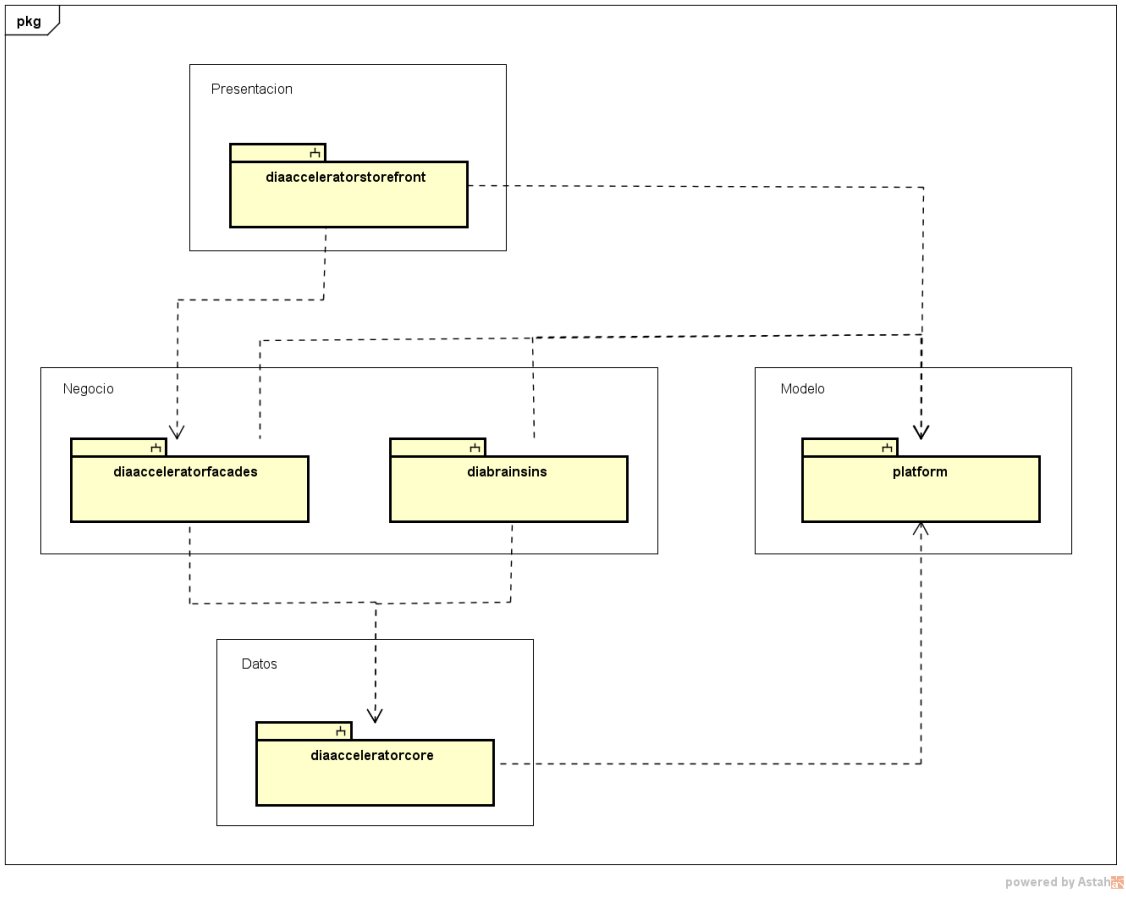


Ilustración 55: Diagrama extensiones

Como se puede observar en el diagrama, la arquitectura de la aplicación, aunque se estructure en extensiones, también sigue el patrón arquitectónico Layers [Buschmann96], quedando dividida en 3 capas. Cada extensión se corresponde con una capa.

- **Presentación**: Se trata de la capa encargada de interactuar con el usuario.
- **Negocio**: Se encarga de llevar a cabo toda la lógica del sistema.
- **Persistencia**: Esta capa se encarga de gestionar los datos y llevar a cabo las conexiones con las fuentes de datos.

Como es habitual en este tipo de arquitectura, cada capa sólo accederá a los servicios que le proporciona la capa inferior, de tal manera que son fácilmente intercambiables.

Por otra parte, el paquete Modelo, contendrá el modelo de dominio de la aplicación. En este caso se trata de un paquete que es transversal, ya que será utilizado por todas las capas de la aplicación. Cabe destacar, que la extensión 'diabrainsins' también contendrá un paquete modelo, propio de dicha extensión, que se compondrá de las distintas clases que componen el modelo de dominio del fichero de exportación, para llevar a cabo la exportación de los productos al formato de BrainSINS.

5.1.1.1 Extensión diaacceleratorstorefront

Esta extensión contiene la capa web. Se compone de la interfaz con la interactúan los clientes del sitio web. En esta capa se llevará a cabo el envío de información al sistema BrainSINS. Consistirá en :

- Páginas JSP y tags.
- Código de invocación a la capa de negocio.
- Código de utilidades para las tareas propias de la capa web.
- Archivos CSS.
- JavaScript, jQuery.

5.1.1.2 Extensión diaacceleratorfacades

En esta extensión se llevará a cabo el procesamiento de la aplicación siguiendo las reglas de negocio. Las acciones que el usuario solicite a través de la capa de presentación, serán procesadas en esta extensión. Para llevar a cabo dicha funcionalidad, también se utilizarán los servicios que provee la capa de persistencia, permitiendo la recuperación de información.

5.1.1.3 Extensión diabrainsins

Esta extensión contendrá la funcionalidad del job de exportación de productos. Para llevar a cabo la funcionalidad, también se utilizarán los servicios que provee la capa de persistencia, permitiendo la recuperación de información de los productos incluidos en el catálogo.

Además, se llevará a cabo la transformación de los productos en la especificación indicada por BrainSINS, a partir de los que se creará el fichero XML de exportación.

A su vez, esta extensión se dividirá en diferentes paquetes:

- **Converters**: Se compone las clases encargadas de convertir los productos a la especificación BrainSINS.

- **Jobs**: Se componen de la clase BrainsinsCronjob, donde se encuentra la lógica del job.
- **Model**: Se compone del modelo del fichero XML de exportación.
- **Service**: Contiene la lógica para la transformación de los productos a la especificación BrainSINS.

5.1.1.4 Extensión diaacceleratorcore

Esta extensión se encarga de gestionar la recuperación de la información de la base de datos. A través de ella se recuperará la información del conjunto de productos a exportar a BrainSINS, así como la información relativa al usuario que será necesaria enviar a BrainSINS.

Todos estos objetos siguen el patrón DAO (Data Access Object) [Sun02], de tal manera que cada DAO se encargará de gestionar un tipo de objetos.

5.1.1.5 Extensión platform

Se trata de una extensión transversal a toda la arquitectura. Será utiliza por las distintas clases de los paquetes que componen las extensiones de la aplicación. Este paquete contendrá el modelo del dominio.

5.1.2 Diagramas de Componentes

A continuación, se presentan los componentes principales, así como sus relaciones en un diagrama general del sistema. Se incorporará un diagrama de componentes tanto para la exportación, como para la integración web con BrainSINS.

5.1.2.1 Diagrama Componentes Exportación

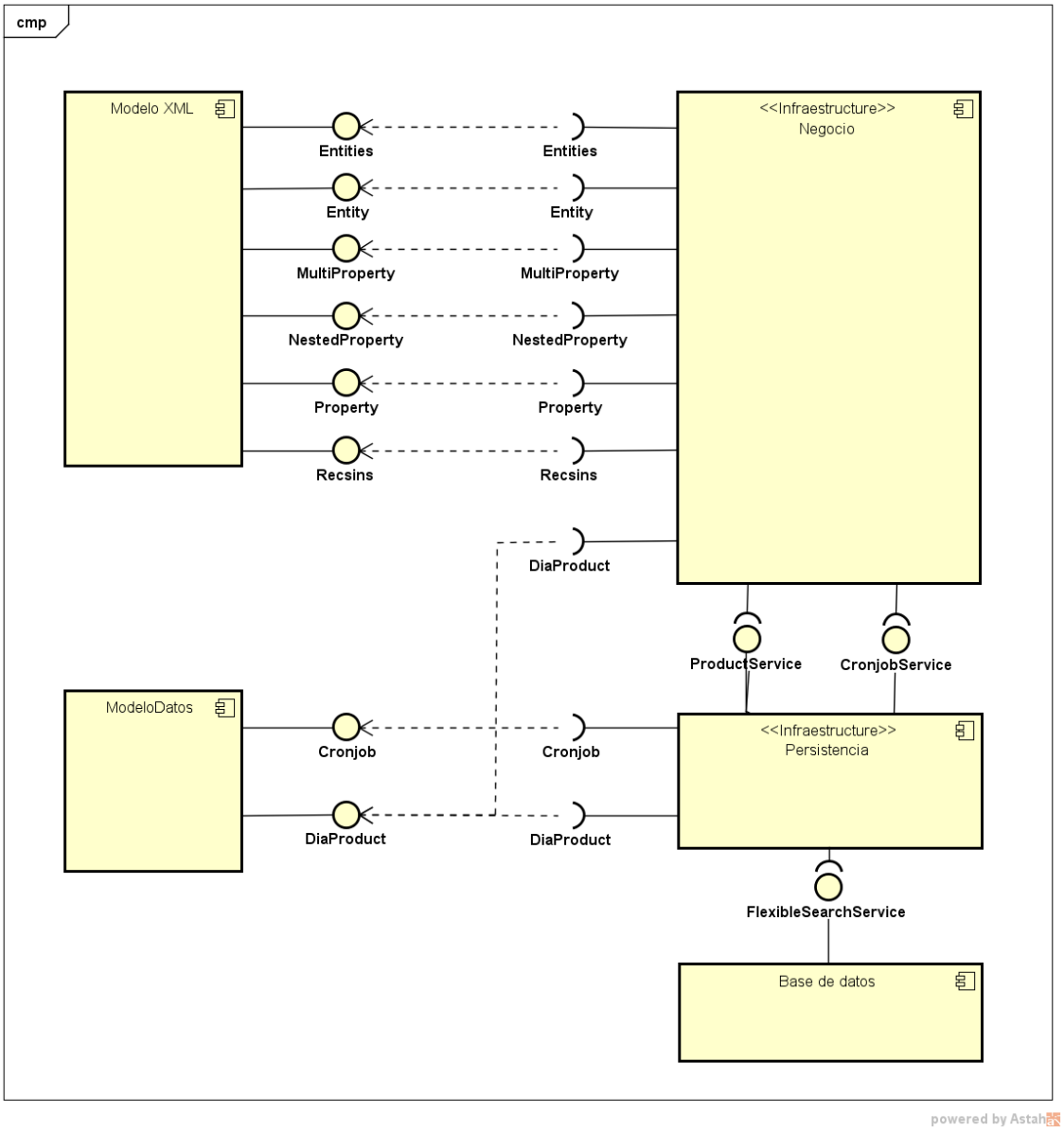


Ilustración 56: Diagrama Componentes Exportación

El componente Negocio, es un componente de infraestructura que modela la capa de lógica de negocio, donde se lleva a cabo la ejecución del Cronjob, que conlleva el procesamiento de datos.

Éste componente requiere de unas interfaces para el acceso a los datos, que son: CronjobService y ProductService. Éstas interfaces a su vez son proporcionadas por el componente Persistencia que se encarga de encapsular el acceso a los datos. Las interfaces implementarán el patrón Fachada [GoF95] del componente Persistencia, promoviendo un débil acoplamiento entre subsistemas, lo que facilita los cambios, haciendo que el sistema sea más mantenible [GoF95].

El componente Persistencia es un componente de infraestructura para el sistema. A través de la interfaz FlexibleSearchService, proporcionado directamente por el sistema Hybris, lleva a cabo el acceso a la base de datos del sistema.

El Modelo de datos proporciona principalmente la interfaz Cronjob y DiaProduct, que serán requeridas por todos los componentes de la infraestructura principal (las capas de la arquitectura) para llevar a cabo las operaciones propias de cada entidad.

Por su parte, el modelo XML proporciona a la capa de negocio los datos que serán requeridos para la conversión de los productos DiaProduct, a un XML que siga la especificación BrainSINS.

5.1.2.2 Diagrama Componentes Sitio Web

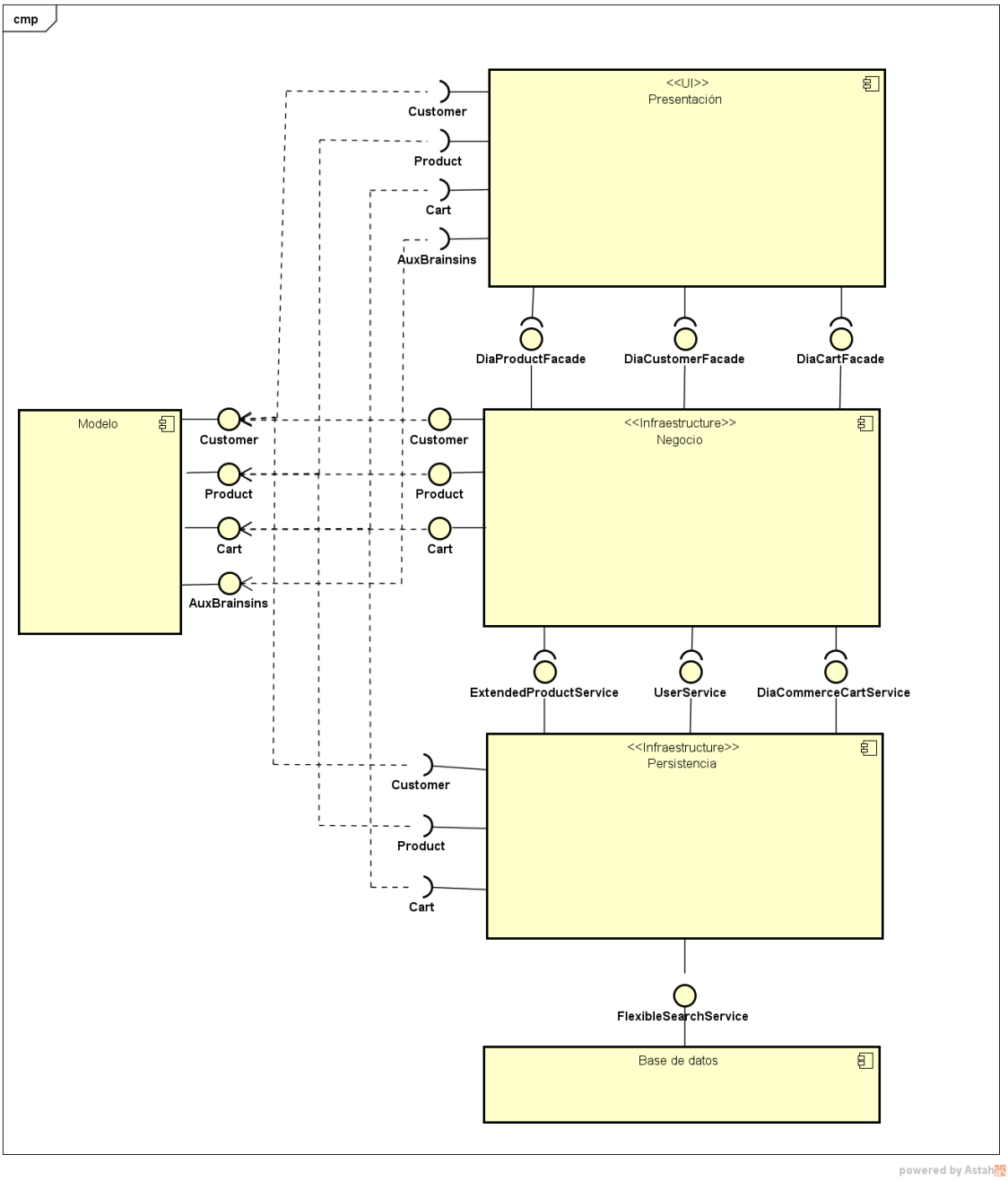


Ilustración 57: Diagrama Componentes Sitio Web

Como puede verse, la interacción con el usuario (estereotipada como User Interface o UI) se encuentra englobada bajo el componente Presentación. Este componente requiere varias interfaces para poder responder a las peticiones de los usuarios:

DiaProductFacade: proporciona acceso a los servicios del componente de Neogioc relacionados con los productos de Dia.

DiaCustomerFacade: proporciona acceso a los servicios del componente de Negocio relacionados con los clientes de Dia.

DiaCartFacade: proporciona acceso a los servicios del componente de Negocio relacionados con el carrito de los clientes.

El componente encargado de proveer estas interfaces es Negocio, como se ha indicado. Estas interfaces, implementan el patrón Fachada [GoF95] del componente Negocio. De este modo, se oculta al componente Presentación la estructura interna del componente Negocio, promoviendo un débil acoplamiento entre subsistemas, lo que facilita realizar cambios, haciendo que el sistema sea más mantenible [GoF95].

El componente Negocio es un componente de infraestructura que modela la capa de lógica de negocio, donde se lleva a cabo el procesamiento de los datos.

Este componente a su vez requiere de unas interfaces análogas a las anteriores para el acceso a los datos que son: ExtendedProductService, UserService, DiaCommerceCartService. Estas interfaces a su vez son proporcionadas por el componente Persistencia que se encarga de encapsular el acceso a los datos. Se trata de nuevo de una Fachada para el componente, con análogas ventajas al caso anterior.

El componente de Persistencia es un componente de infraestructura para el sistema. A través de la interfaz FlexibleSearchService, proporcionado directamente por el sistema Hybris, lleva a cabo el acceso a la base de datos del sistema.

El modelo proporciona principalmente las interfaces de Customer, Product y Cart, que serán requeridas por todos los componentes de la infraestructura.

Además, la interfaz AuxBrainsins encapsula la información necesaria de enviar a Brainsins, que será utilizada por la capa de Presentación y que será construído a partir de los resto de los servicios citados.

5.1.3 Diagramas de Despliegue

A continuación se presenta el diagrama general de despliegue de la aplicación.

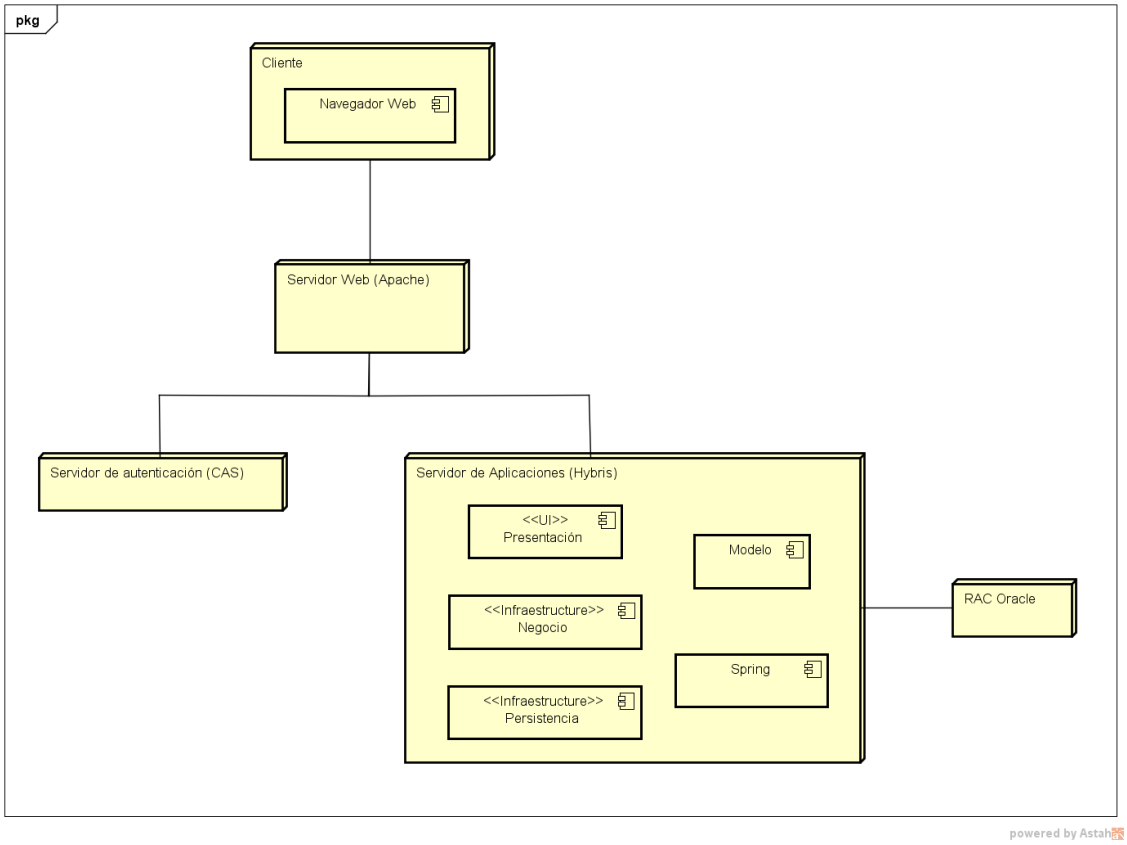


Ilustración 58: Diagrama de Despliegue

En esta diagrama se puede observar un nodo Cliente que contendrá un navegador web, mediante el cual se accederá a la aplicación, a través del nodo Servidor Web. A partir de este nodo, se accede al Nodo de autenticación (CAS) y al nodo del Servidor de Aplicaciones para poder atender a las peticiones de los usuarios. En este nodo se encontrarán todos los componentes de la aplicación, Presentación, Negocio, Persitencia y Modelo de la aplicación.

Por último, desde el nodo Servidor de Aplicaciones se accede al Rac de Oracle que se trata de un sistema con acceso a la base de datos del sistema.

5.1.3.1 Cliente

El nodo cliente representa a los dispositivos con los que los usuarios acceden al sitio web, a través de un navegador web.

5.1.3.2 Servidor Web (Apache)

Este nodo representa el servidor web al que el cliente realizará las peticiones y que devolverá las distintas páginas, comunicándose con el servidor web de aplicaciones.

Además, a través de él, también se llevarán a cabo las peticiones de autenticación, que serán ejecutadas en el servidor CAS.

5.1.3.3 Servidor de Autenticación (CAS)

Se trata del servidor encargado de autenticar a los clientes en el sitio web.

5.1.3.4 Servidor de Aplicaciones (Hybris)

Este nodo, contendrá los componentes de Presentación, Negocio y Persistencia, pertenecientes a la arquitectura de capas de la aplicación. También contendrá el Modelo.

5.1.3.5 RAC Oracle

Este nodo contendrá la Base de Datos de la aplicación, con los datos de clientes, productos, carritos... todas ellas entidades implicadas en la integración con BrainSINS.

5.2 Diseño de Clases

A continuación, se presenta el diseño de clases de la aplicación. En primer lugar se expone un diagrama general por funcionalidad. Es decir, por un lado se mostrará el diagrama de clases general, que tiene que ver con la exportación de los productos y por otro lado, se mostrará un diagrama de clases general que tiene que ver con la integración de BrainSINS en el sitio web.

Luego estos diagramas generales serán explicados en detalle.

5.2.1 Diagrama de Clases Exportación

El siguiente diagrama se corresponde con el diagrama general de la aplicación asociado a la exportación de los productos. Se divide en capas, que a vez se componen de distintas extensiones, pues como se ha comentado a lo largo del proyecto, la funcionalidad Hybris se divide en extensiones, que a su vez se corresponden con la funcionalidad de alguna de las capas de la aplicación.

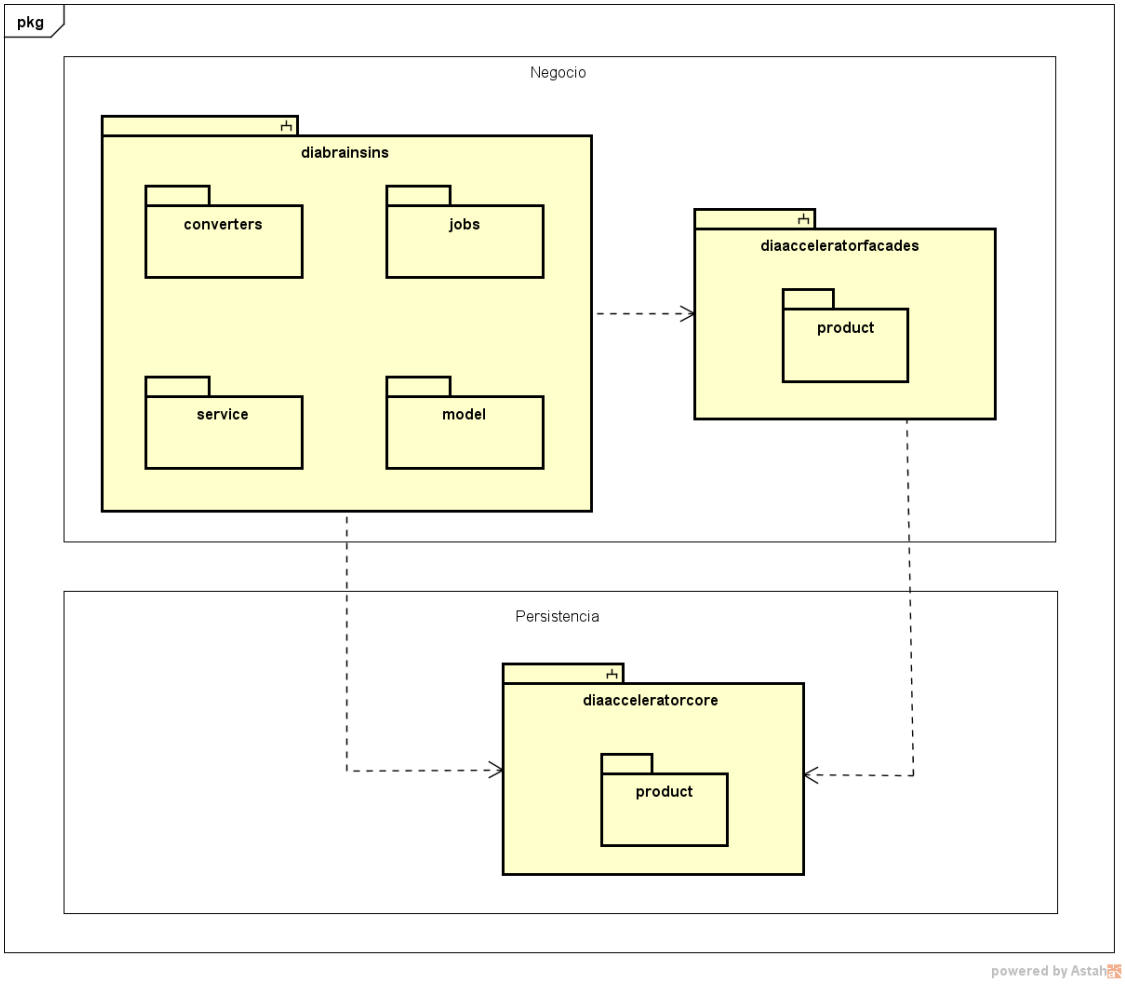


Ilustración 59: Diagrama Clases Exportación

5.2.2 Diagrama Clases Negocio Extensión Diabrainsins

Este diagrama se corresponde con la extensión Diabrainsins, que pertenece a la capa de negocio de la aplicación. Esta extensión se compone de cuatro paquetes.

- **Jobs**: contiene el cronjob encargado de llevar a cabo la exportación.
- **Service**: contiene al servicio encargado de llevar a cabo la transformación de productos a la especificación de BrainSINS.
- **Converters**: contiene al conjunto de conversores utilizados por el paquete Service.
- **Model**: contiene el modelo utilizado por el paquete Converters par al transformación de los productos.

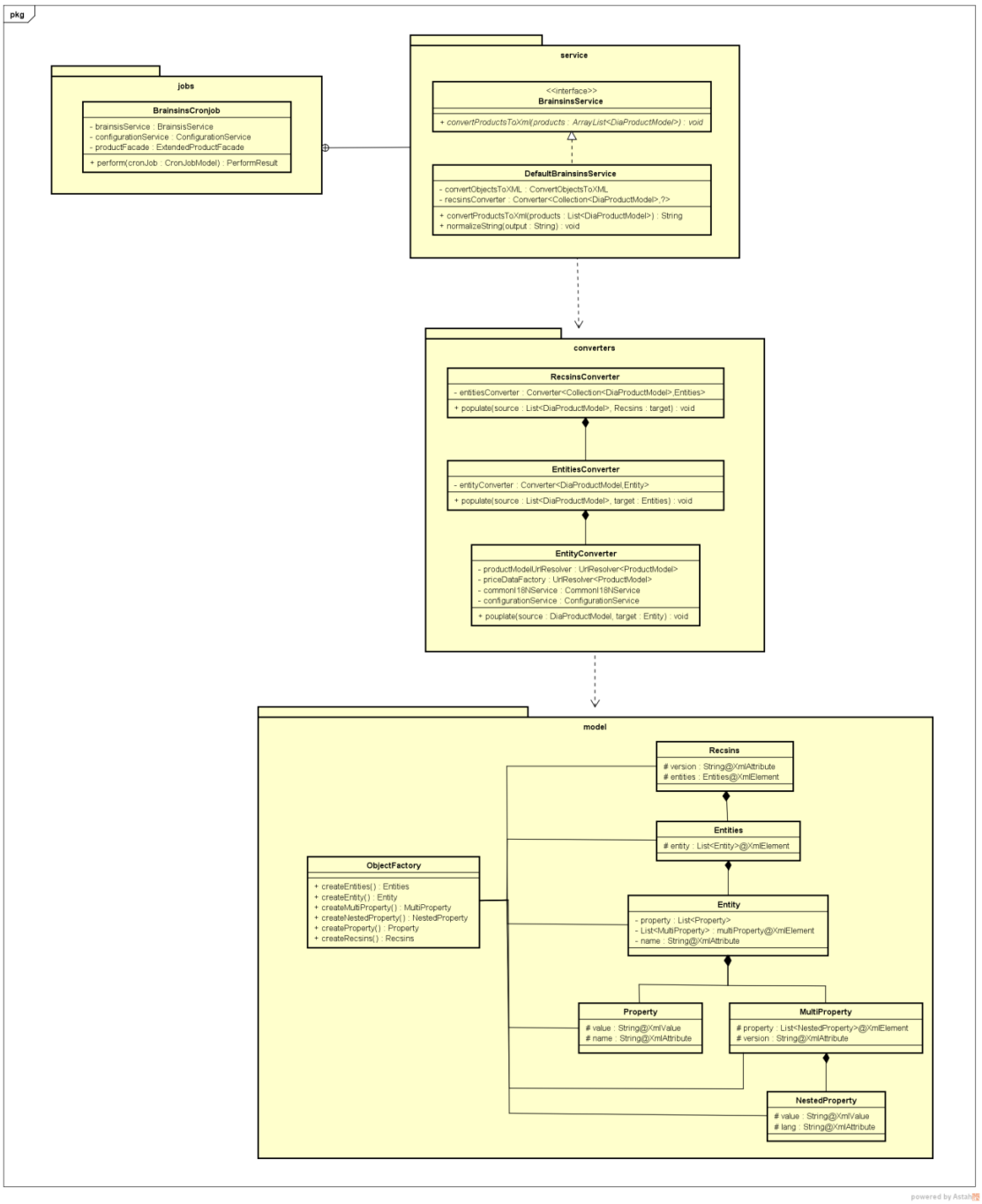


Ilustración 60: Diagrama de Clases Diabrainsins

5.2.3 Diagrama de Clases Negocio Extensión diaacceleratorfacades

Esta extensión, actúa como una capa intermedia entre las diferentes extensiones, evitando su acoplamiento. De este modo, se evita un acceso directo a los datos, capa de persistencia.

El Cronjob para llevar a cabo la exportación, antes, necesita conocer el conjunto de productos que se encuentran en el catálogo. Para ello utiliza la fachada ExtendedDiaProductFacade, que se encargará de acceder a la capa de persistencia.

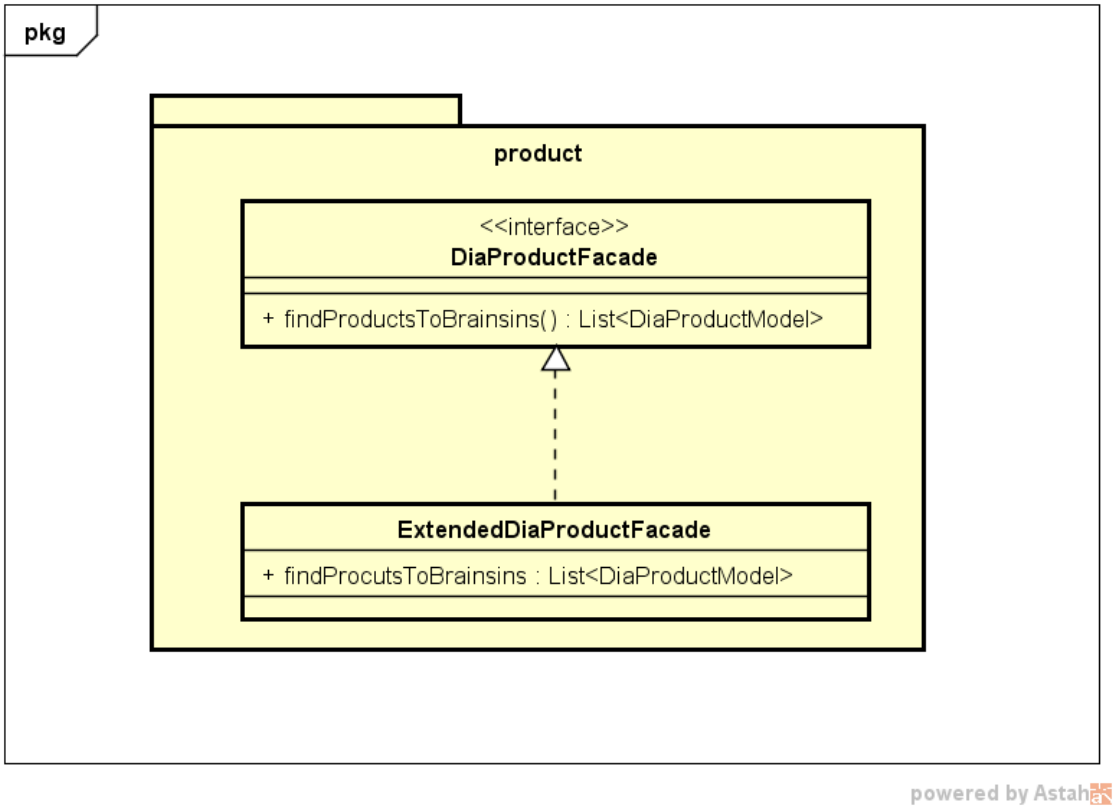


Ilustración 61: Diagrama Clases Diaacceleratorfacades

5.2.4 Diagrama Clases Persistencia Extensión diaacceleratorcore

Se trata de la extensión de persistencia, con acceso directo a la base de datos de la aplicación.

Posee un servicio, DefaultExtendedProductService, que se encarga de encapsular el acceso a la base de datos. Éste servicio será el que utilizará la capa de negocio mencionada anteriormente.

El acceso de directo a la base de datos, lo lleva a cabo por medio de la clase "ExtendedProductDAO". Se trata del DAO dedicado a los productos de DIA.

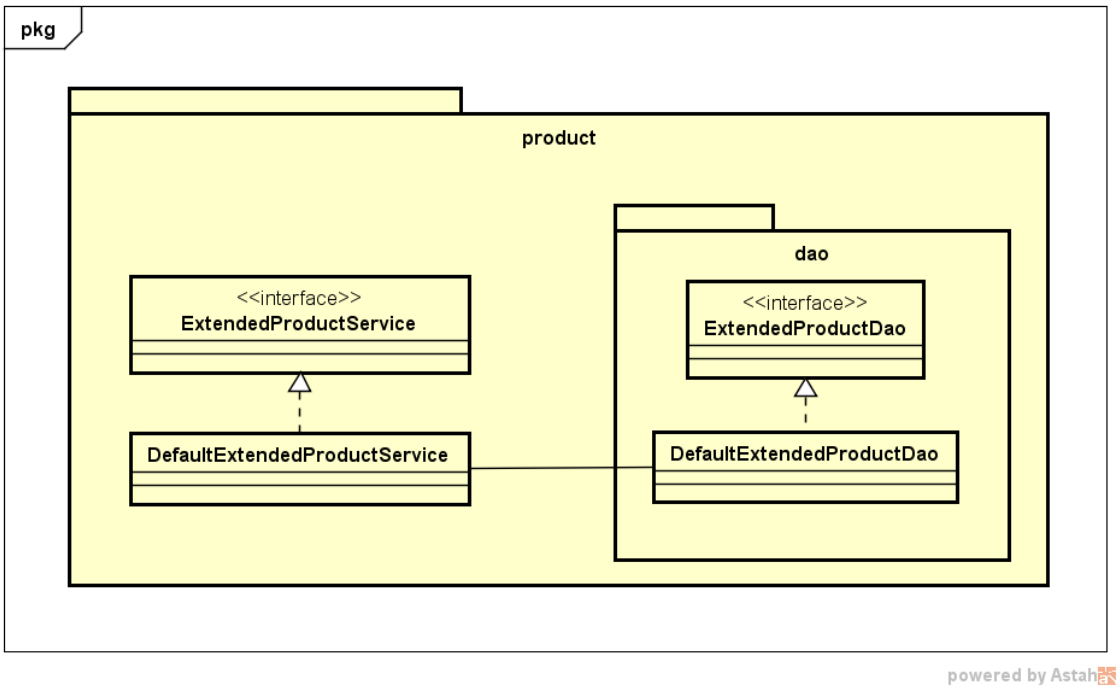


Ilustración 62: Diagrama Clases Extensión diaacceleratorcore

5.3 Diagrama de Clases Integración Sitio Web

El siguiente diagrama se corresponde con el diagrama general de la aplicación asociado a la integración del sitio Web con BrainSINS. Se divide en capas que a su vez se componen de distintas extensiones, ya que como se ha comentado a lo largo del proyecto, la funcionalidad Hybris se divide en extensiones, que a su vez se corresponden con la funcionalidad de alguna de las capas de la aplicación.

5.3.1 Diagrama de clases Presentación

La capa de presentación del sitio web, se corresponde con la extensión diaacceleratorstorefront.

En el subpaquete pages del paquete Controllers, se encuentran el conjunto de controladores de la aplicación. Se incluyen cinco, aquellos controladores donde es necesario integrar BrainSINS:

- HomePageController: controlador de la página home.
- ProductPageController: controlador de la página de detalle de producto.
- CategoryPageController: controlador de la página asociada a las diferentes categorías de los productos ofertados.
- CartPageController: se trata del controlador de la página del carrito.
- MultiPaymentCheckoutController: se trata del controlador de la página de confirmación del pedido. Ésta se muestra una vez el usuario a finalizado satisfactoriamente todos los pasos de su compra.

Todos estos controladores incluyen el servicio BrainSINSStorefrontService, que forma parte del paquete brainsins.service. Éste servicio se compone de distintos métodos asociados a cada uno de los controladores donde es necesaria incluir la integración. Cada método, se encarga de incluir en el modelo, la información a enviar a BrainSINS. Esta información se agrupa en el objeto AuxBrainsinsData , que forma parte del paquete brainsins.data.

Por último, el paquete donde se incluyen los diferentes javascript del sitio web, contiene el javascript "acc.brainsins.js". Este javascript contiene el objeto ACC.brainsins cuya función es la de enviar al sistema BrainSINS la información. Además, también se encarga de incluir en el formulario HTML cada uno de los productos recomendados las class e información de los productos incluidos en el carrito del usuario.

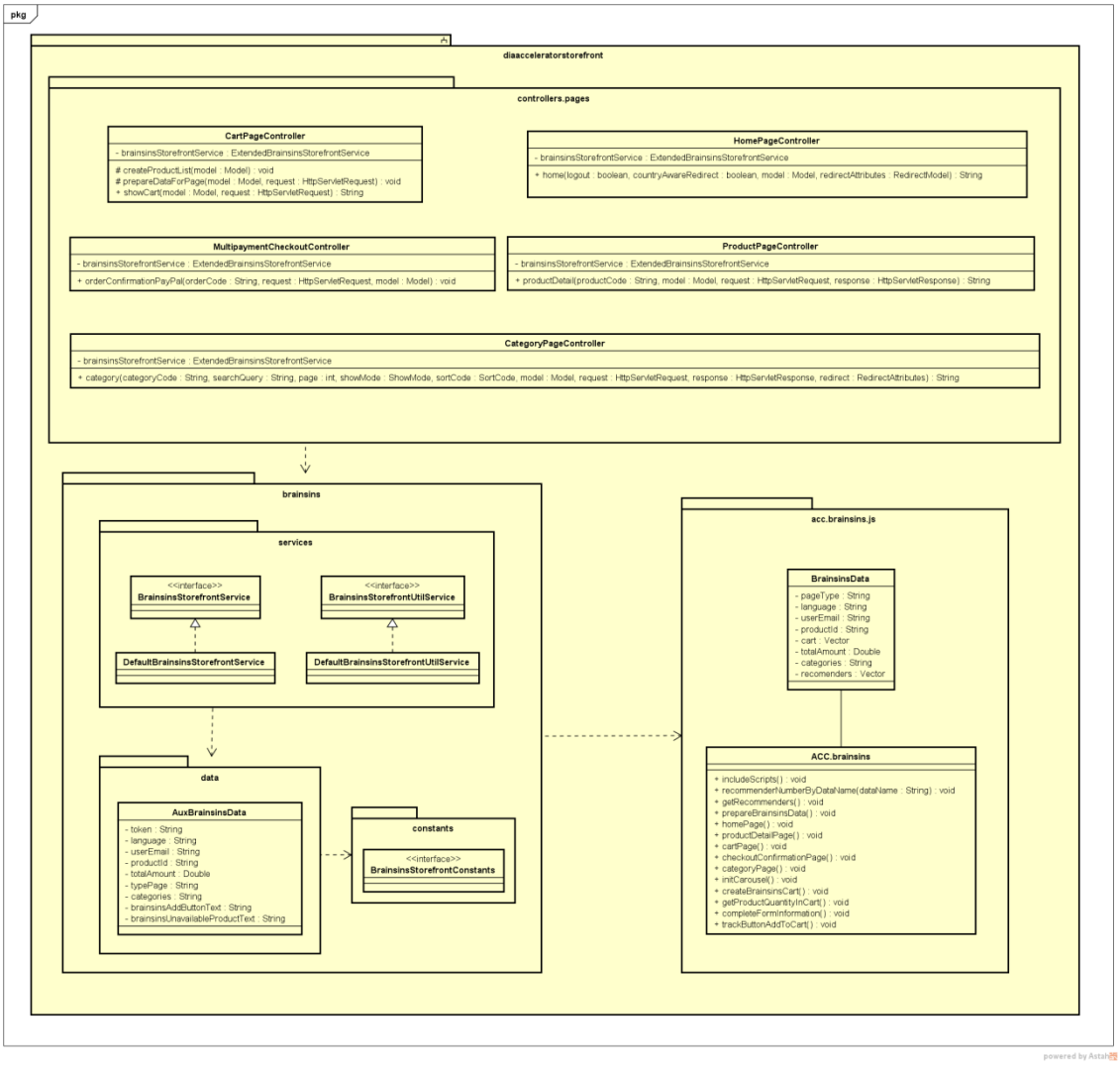


Ilustración 63: Diagrama de Clases Presentación

5.3.2 Diagrama de Clases Presentación Paquete brainsins.service

En este apartado, se incluyen las clases detalladas, que forman parte del paquete brainsins.service.

Este paquete se componen de las interfaces BrainsinsStorefrontService y BrainsinsStorefrontUtilsService y la implementación de las mismas.

ExtendedBrainsinsStorefrontService se trata de un servicio dedicado a la capa de presentación del sitio web. Se encarga de actualizar el modelo de con el objeto AuxBrainsinsData, que contendrá la información a enviar a BrainSINS. Por un lado, contiene el atributo SessionService, a partir del que accede a los datos de sesión del usuario. Por otro lado, posee el atributo DiaCustomerFacade para la obtención de datos del usuario.

ExtendedBrainsinsStorefrontService posee los métodos de utilidad de ExtendedBrainsinsStorefrontService. Posee el atributo ConfigurationService, encargado de recuperar la configuración del sitio web, es decir, consulta el fichero de configuración "local.properties". Además, contiene los atributos MessageSource e I18NService, que el propio sistema Hybris proporciona para la obtención de los distintos mensajes de la aplicación en función de su localización, idioma.

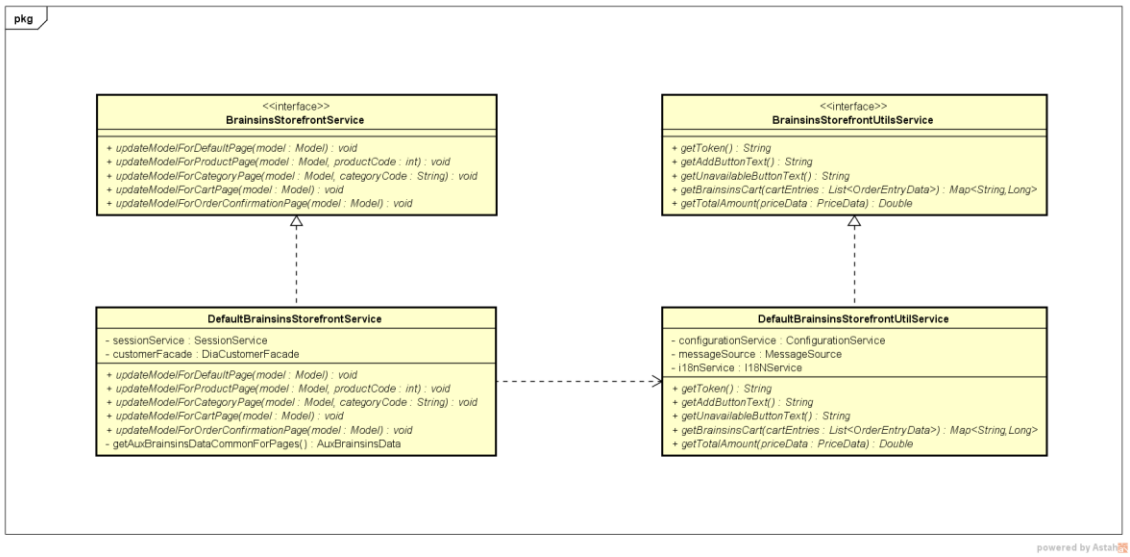


Ilustración 64: Diagrama de Clases Presentación

5.3.3 Diagrama de Clases Negocio Extensión diaacceleratorfacades

Esta extensión, actúa como una capa intermedia entre la capa de presentación y la de persistencia. De este modo, se evita su acoplamiento.

BrainSINS necesita conocer el email del usuario y la tienda en la que se encuentra situado. Para ello, el servicio BrainsinsStorefrontService, hará uso de la fachada DiaCustomerFacade, que será la clase encargada de proveerle dicha información.

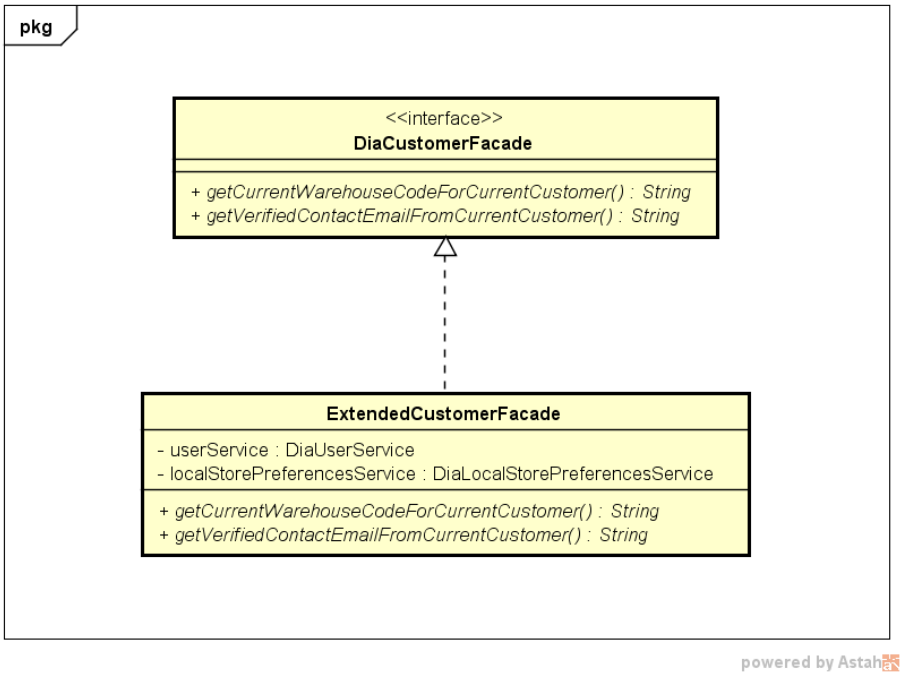


Ilustración 65: Diagrama de Clases Negocio

5.4 Diagramas de Interacción y Estados

A continuación se muestran los diagramas de interacción y estados más importantes.

5.4.1 Caso de Uso 1: Ejecutar Cronjob

Este caso de uso se corresponde con la exportacion del catálogo de productos de DIA a un fichero XML que sigue la especificación BrainSINS.

Este caso de uso comprende tres funcionalidades bien diferenciadas. Con el objetivo de facilitar su comprensión, se ha divido en tres diagramas, que se muestran en el orden de ejecución.

5.4.1.1 Diagramas de Secuencia

En el primer diagrama se muestra el proceso de obtención del listado de los productos a exportar a BrainSINS.

Como puede observarse, en el caso de que previamente no se haya configurado la localización del fichero de exportación, ni este proceso ni los sucesivos serán ejecutados. Esto provocará que el resultado del cronjob, PerformResult, sea 'ERROR' y su estado sea 'ABORTED'.

En el segundo diagrama, se muestra el proceso de conversión del listado de productos de tipo DiaProductModel a exportar. El resultado de esta conversión será un String que seguirá la especificación BrainSINS. Por último, a partir de dicho resultado, se creará el fichero final en formato XML que será almacenado en la ruta configurada. Una vez finalizado, se devolverá el resultado del cronjob, PerformResult, cuyos valor será 'SUCCESS' y su estado 'FINISHED'.

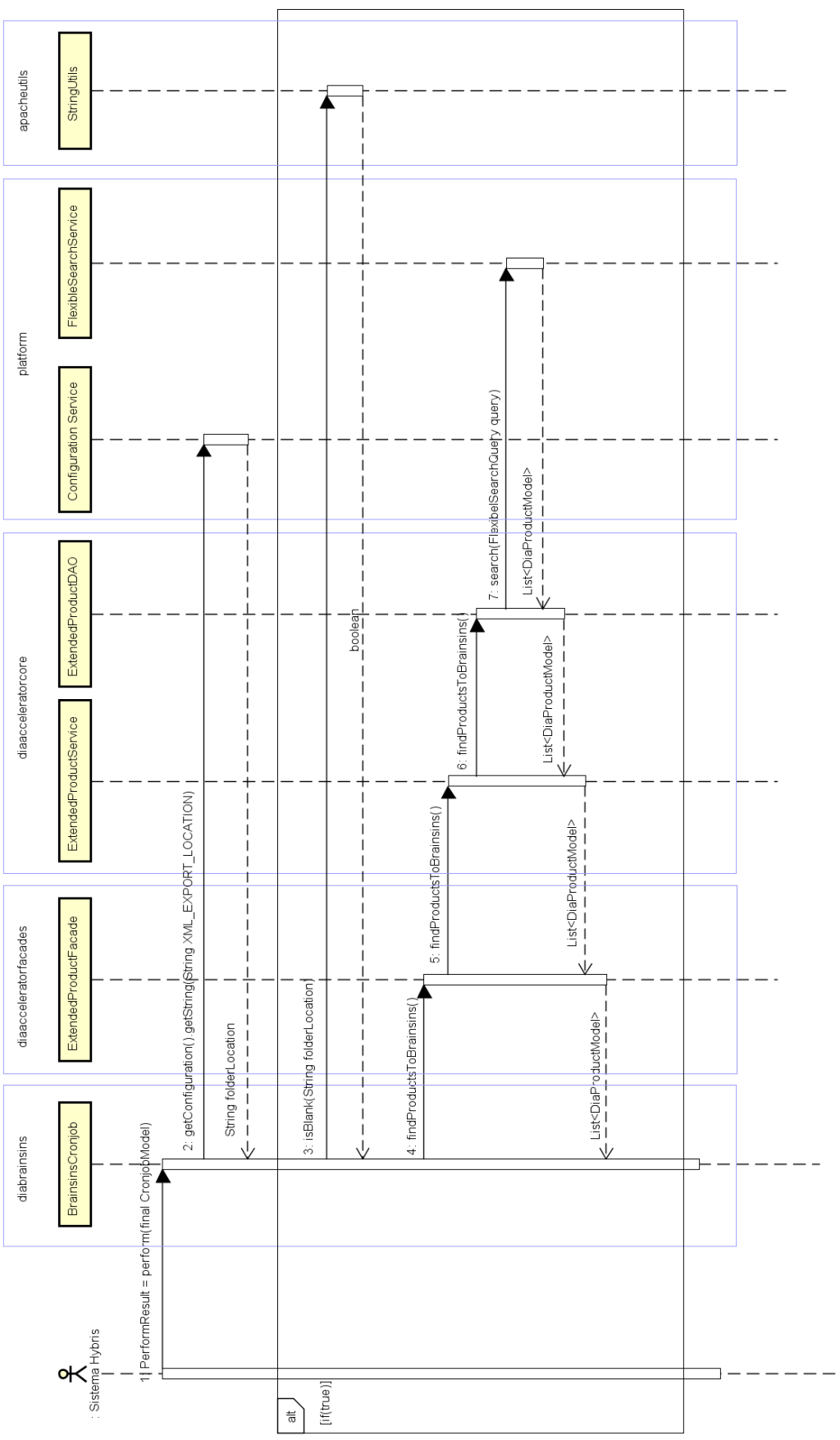


Ilustración 66: Diagrama Secuencia Caso de Uso 1 – Parte I

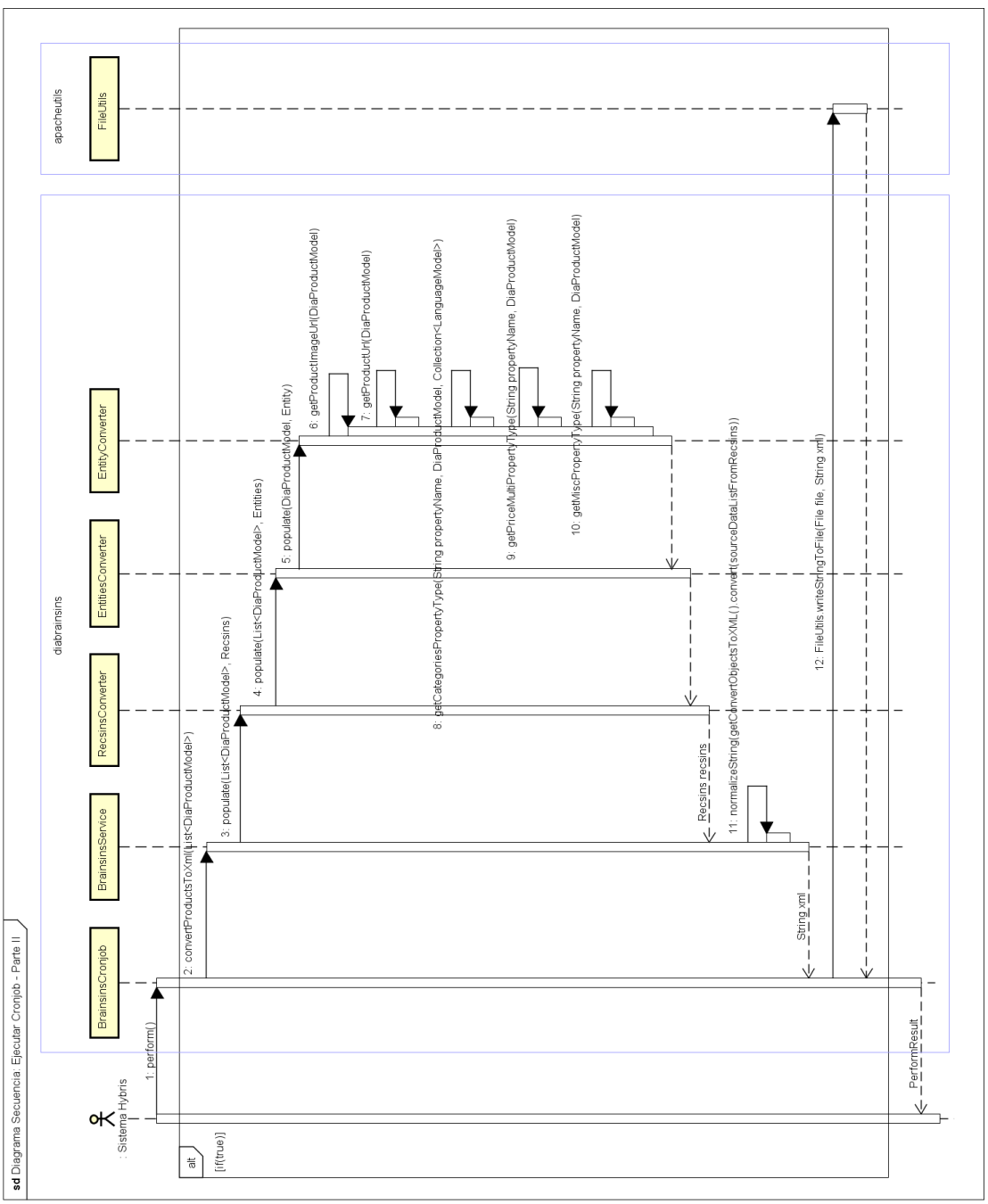


Ilustración 67: Diagrama Secuencia Caso de Uso 1 – Parte II

5.4.2 Caso de Uso 2: Acceso Página Home

Este caso de uso se corresponde con la solicitud de la página home que lleva a cabo un cliente. Este acción, desencadena la recuperación de la información asociada a esta página, que luego será enviada a BrainSINS.

5.4.2.1 Diagrama de Secuencia

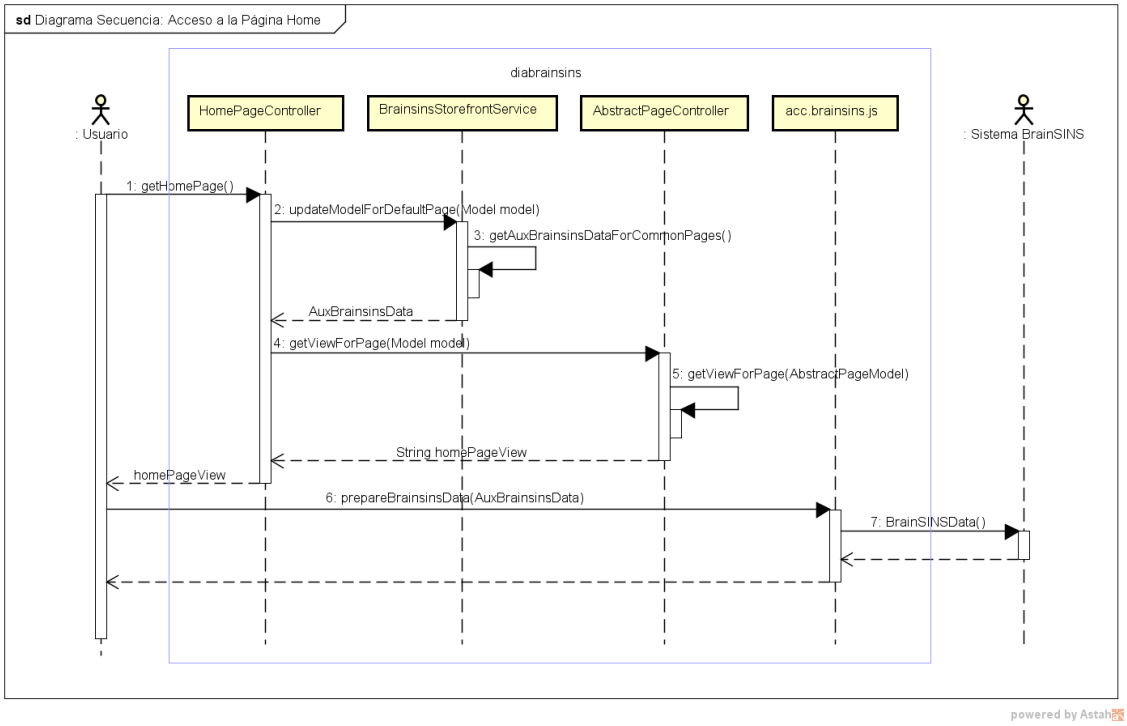


Ilustración 68: Diagrama Secuencia Caso de Uso 2

5.4.3 Caso de Uso 3: Acceso Página Detalle de Producto

Este caso de uso se corresponde con la solicitud de la página de detalle de un producto concreto, que lleva a cabo un cliente. Esta acción, desencadena la recuperación de la información asociada a esta página, que luego será enviada a BrainSINS.

5.4.3.1 Diagrama de Secuencia

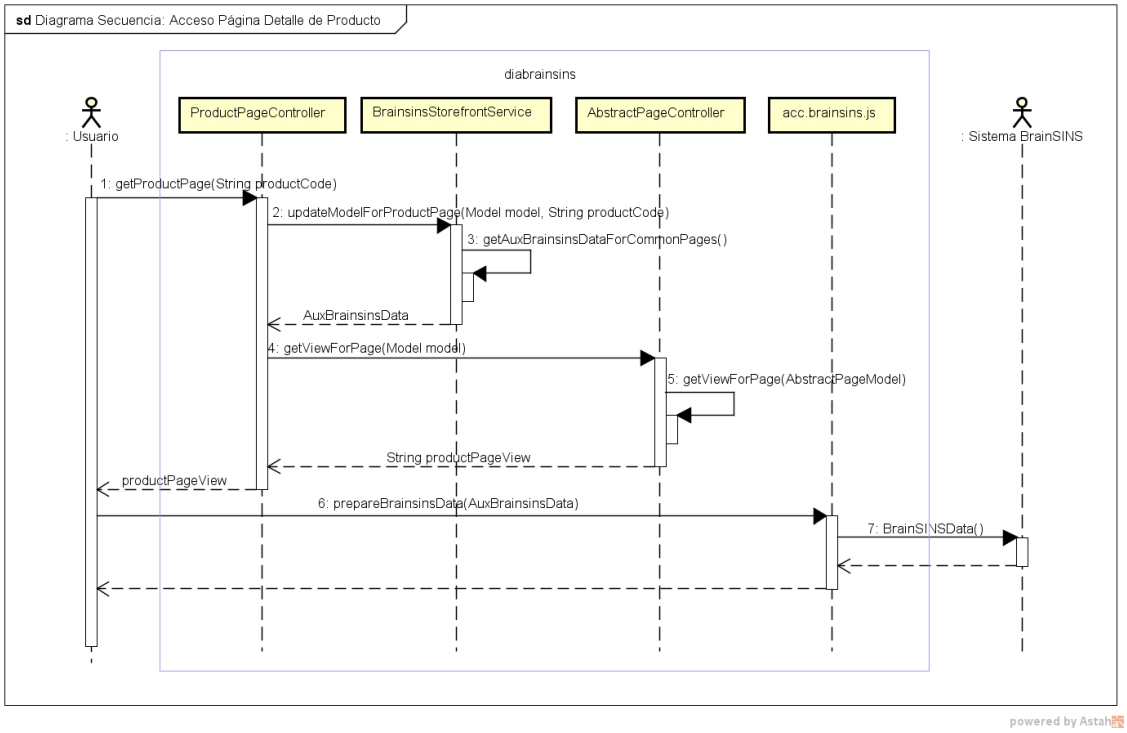


Ilustración 69: Diagrama Secuencia Caso de Uso 3

5.4.4 Caso de Uso 4: Acceso Página Categorías

El presente caso de uso, se corresponde con la solicitud de la página de categorías llevada a cabo por un cliente. Esta acción, desencadena la recuperación de la información asociada a esta página, que luego será enviada a BrainSINS.

5.4.4.1 Diagrama de Secuencia

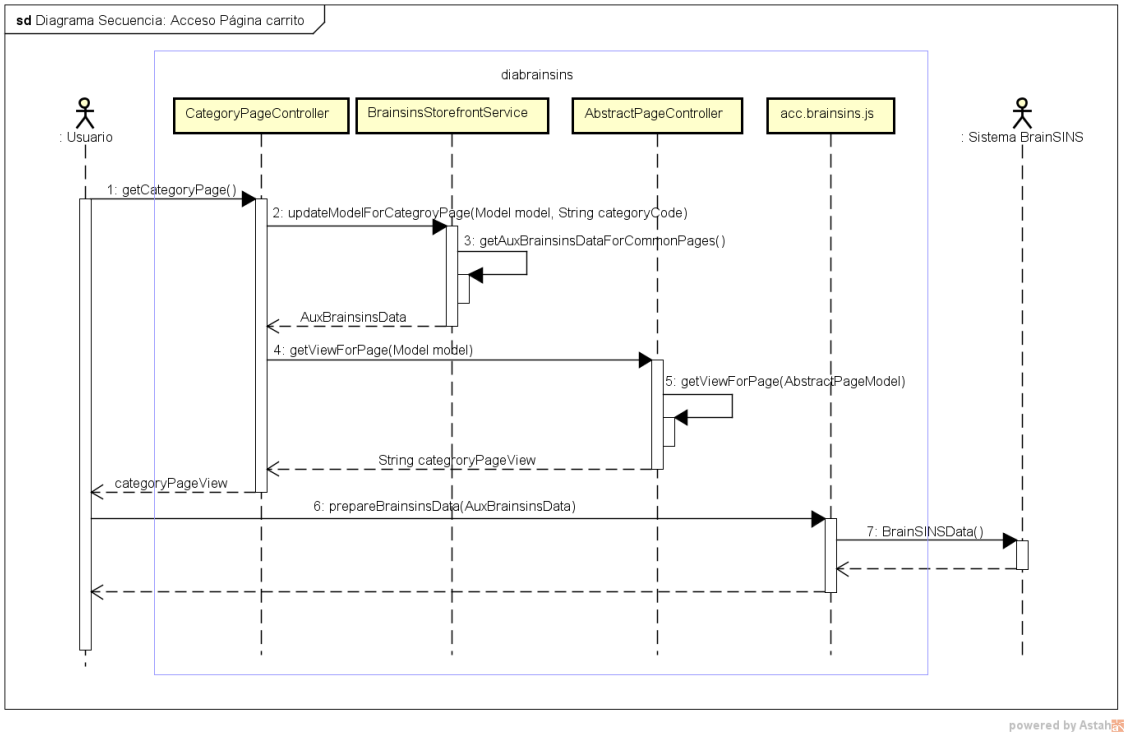


Ilustración 70: Diagrama Secuencia Caso de Uso 4

5.4.5 Caso de Uso 5: Acceso Página Carrito

El presente caso de uso, se corresponde con la solicitud de la página del carrito de un cliente. Esta petición desencadena la recuperación de la información asociada a esta página, que luego será enviada a BrainSINS.

5.4.5.1 Diagrama de Secuencia

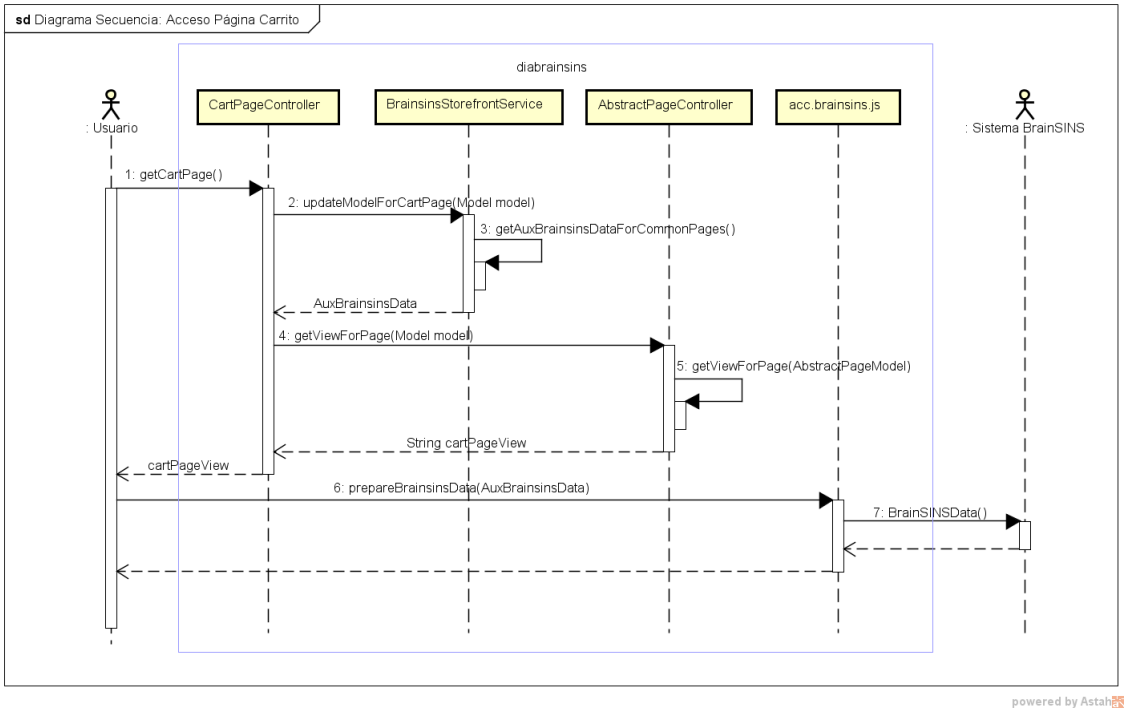


Ilustración 71: Diagrama Secuencia Página Carrito

5.4.6 Caso de Uso 6: Acceso Página Confirmación Pedido

Este caso de uso, cuando un cliente ha finalizado correctamente todos los pasos de la compra. El sistema le muestra la página de confirmación del pedido. Esta petición desencadena la recuperación de la información asociada a esta página, que luego será enviada a BrainSINS.

5.4.6.1 Diagrama de Secuencia

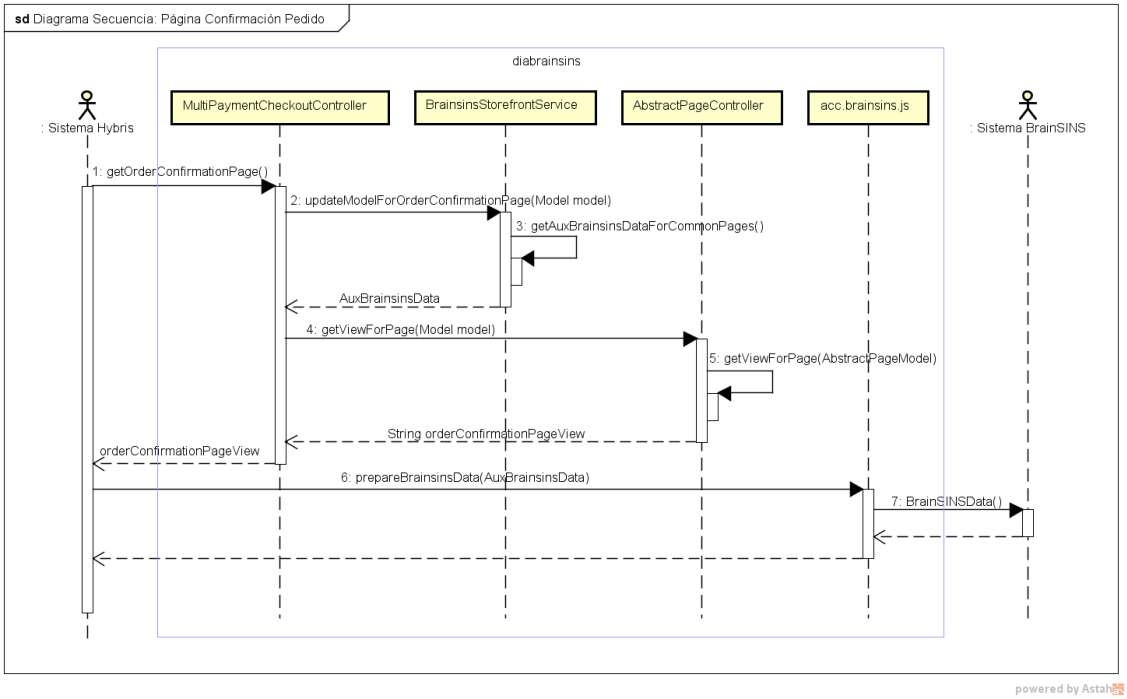


Ilustración 72: Diagrama Secuencia Caso de Uso 6

5.4.7 Caso de Uso 7: Añadir Producto al Carrito

Este caso de uso, se genera cuando un usuario selecciona el botón añadir de un producto.

5.4.7.1 Diagrama de Secuencia

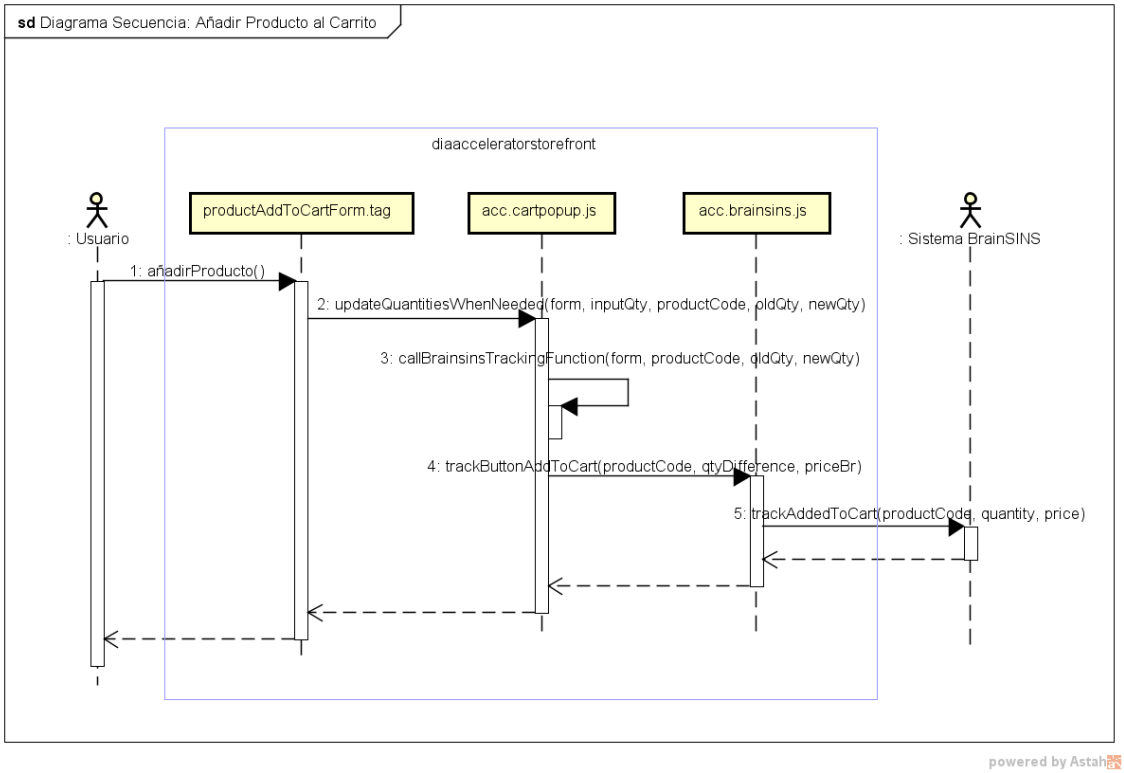


Ilustración 73: Diagrama de Secuencia Caso de Uso 7

5.4.8 Caso de Uso 8: Añadir Unidades al Carrito

La funcionalidad de este caso de uso es idéntico al anterior, pues en el método 'callBrainsinsTrackingFunction', se encarga de calcular la diferencia entre las unidades que había en el carrito, y las que ha añadido o eliminado el cliente. BrainSINS utiliza la misma función para llevar a cabo el tracking.

5.4.9 Caso de Uso 9: Eliminar Unidades del Carrito

La funcionalidad de este caso de uso es idéntico al anterior, pues en el método 'callBrainsinsTrackingFunction', se encarga de calcular la diferencia entre las unidades que había en el carrito, y las que ha añadido o eliminado el cliente. BrainSINS utiliza la misma función para llevar a cabo el tracking.

5.4.10 Caso de Uso 10: Inclusión Recomendador en Página

l actual caso de uso se produce cuando el HTML de la página posee el elemento donde incluir uno o varios recomendadores. Previamente Se ha tenido que enviar la información de los recomendadores a Brainsins, junto con el resto de la información referente al tipo de página.

5.4.10.1 Diagrama de Secuencia

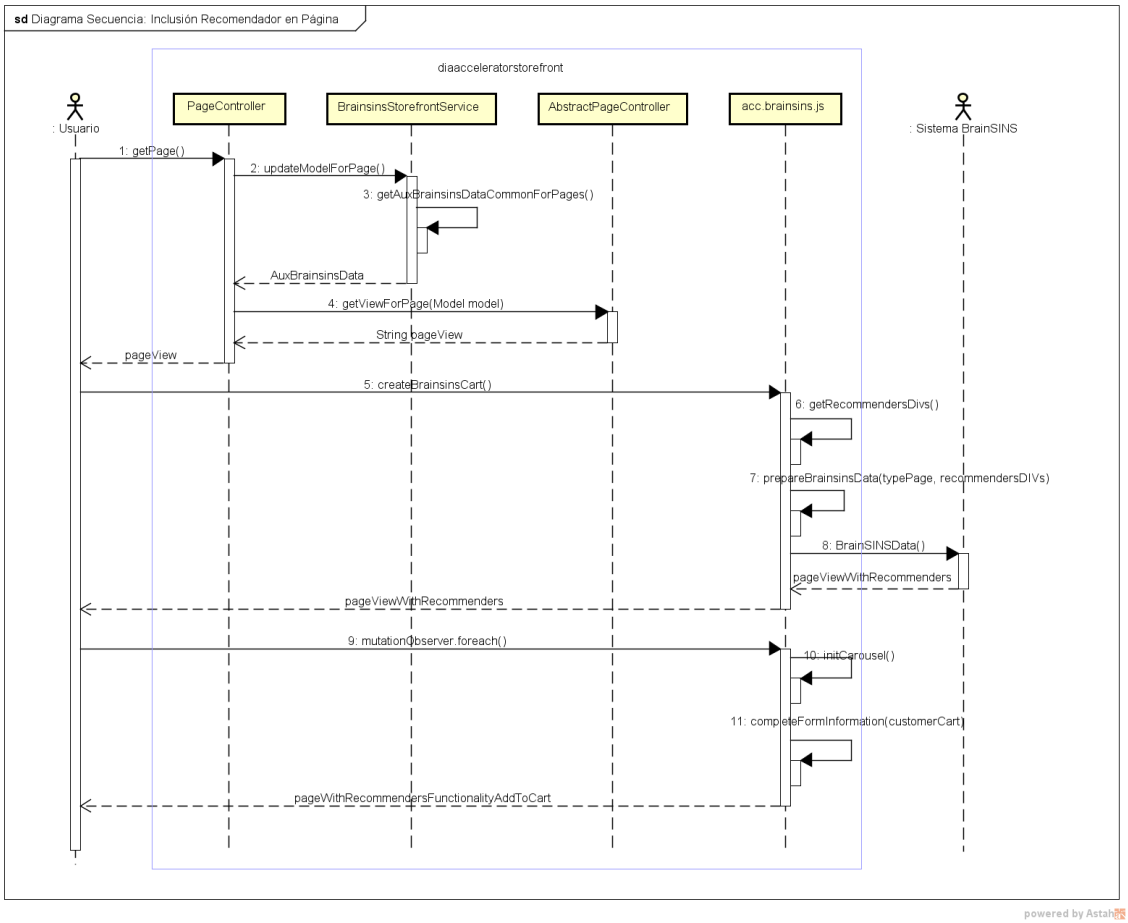


Ilustración 74: Diagrama Secuencia Caso de Uso 10

5.5 Diseño de la Interfaz

En esta sección se va a mostrar la interfaz definitiva de los carruseles.

5.5.1 General

La interfaz consta de un carrusel de productos recomendados. Como se ha comentado, es requisito imprescindible que la interfaz de los productos recomendados por BrainSINS sea idéntica a los carruseles propios del sistema Hybris. De este modo, los usuarios del sitio web lo apreciarán como un mismo componente.

La interfaz se compone de un carrusel de productos, cuya movilidad es proveída por los iconos que se incluyen en sus extremos.

Ilustración 75: Interfaz Carruseles BrainSINS

Como puede observarse en la imagen anterior, cada producto recomendado se compone de los siguientes elementos:

- Imagen del producto.
- Descripción
- Valoración proveída por los usuarios, en forma de estrellitas. En caso de que el producto no haya obtenido ninguna valoración, las estrellitas se mostrarán de color gris.
- Precio del producto en la tienda en la que se encuentra el usuario.
- Funcionalidad añadir al carrito.

Además, tanto desde la descripción del producto, como desde su imagen, el usuario podrá acceder a su página de detalle.

5.5.2 Funcionalidad añadir al carrito

Por defecto se muestra el botón “Añadir”, a no ser, que el carrito del cliente, este formado por alguno de los productos recomendados. En este caso, en lugar de mostrarse el botón añadir, se mostrarán los botones más y menos, y entre ambos, las unidades del producto incluidas en el carrito.

Capítulo 6. Implementación del Sistema

6.1 Estándares y Normas Seguidos

En el ciclo de vida de las aplicaciones, el mantenimiento es una de las fases que mayor tiempo requieren. Además, es habitual que sea llevado a cabo por personas distintas a la que crearon la aplicación. Por todo ello, es importante seguir una serie de estándares y normas para que su mantenimiento sea lo más sencillo posible, lo que repercutirá positivamente en el coste temporal.

Los principales criterios seguidos a la hora de implementar el presente proyecto son:

- Siguiendo el estándar Hybris, se ha creado una nueva extensión llamada 'diabrainsins' en la que se incluye toda la funcionalidad necesaria para la exportación del catálogo de productos de DIA.
- Para la parte de la presentación de la aplicación, en concreto, para la extensión 'diaaccelereatorstorefront' se ha desarrollado un servicio denominado 'BrainsinsStorefrontService', siguiendo el convenio de nombrado Hybris. Este servicio será inyectado en aquellos controladores donde sea necesario enviar la información a BrainSINS.
- Siguiendo la arquitectura en capas y el propio diseño de la aplicación DIA, se han creado nuevos métodos en las clases oportunas, siguiendo el patrón Facade y actualizando las diferentes interfaces.
- Se ha seguido el criterio habitual para el nombrado de clases en Java, capitalizando el nombre de la primera letra y también capitalizando la primera letra de cada palabra, que compone el nombre de la clase.
- También se ha seguido el criterio habitual para el nombrado de objetos en Java, escribiendo la primera letra de su nombre en minúscula y capitalizando el resto de palabras que componen su nombre.

6.2 Lenguajes de Programación

A continuación se describen los diferentes lenguajes de programación utilizados par ala implementación del proyecto.

6.2.1 Java

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de 1995.

La sintaxis básica del lenguaje deriva de la de C/C++, pero tiene un modelo de objetos más simple, eliminado los aspectos de bajo nivel que suelen inducir a errores, como la manipulación directa de punteros o memoria, facilitando su programación.

Se ejecuta en un entorno virtual. Los programas con este lenguaje presentan una serie de características, como por ejemplo, la portabilidad entre distintas plataformas sin necesidad de recompilar.

La versatilidad y eficiencia de la tecnología Java, la portabilidad de su plataforma y la seguridad que aporta, la han convertido en la tecnología ideal para su aplicación a redes.

Actualmente, la última versión es la 8.9.2, sin embargo, dada la versión del sistema de comercio electrónico Hybris utilizado, la versión 5.2, se hace obligatorio utilizar la versión 7 de Java.

6.2.2 Spring Framework

La aplicación es desarrollada con el Framework de Spring, que es el utilizado por la solución Hybris.

Spring Framework provee de un conjunto de conceptos, técnicas y una metodología de programación para el desarrollo de aplicaciones informáticas de alta calidad.

Spring fue escrito originalmente para la plataforma J2EE de Java, plataforma orientada al desarrollo de aplicaciones web. Ha ido evolucionando rápidamente hasta el día de hoy.

En la actualidad es referencia en el mundo de los frameworks para el desarrollo web. Impulsa una metodología de trabajo ágil, eficiente y de buena praxis, lo que lo hace de gran calidad y mantenibilidad.

Tiene una alta compatibilidad para la integración con otros frameworks y librerías de uso común para la creación de aplicaciones web, desde CompositeViews como Velocity o Tiles, APIS en capa de peristencia como Hibernato o JDO y otra miscelánea de recursos como JavaMail, Quartz...

6.2.3 JSP

Acrónimo de Java Server Pages, es una tecnología que permite genera contenido dinámico par web, de forma rápida y simplificada permitiendo un desarrollo rápido de aplicaciones independientes del servidor y plataforma. [Oracle]

Se trata de una tecnología desarrollada por la compañía Sun Microsystems, que permite la utilizacón de código Java mediante scripts.

6.2.4 CSS

Es el estándar publicado por el W3C, para la parte de la presentación de los contenidos. Es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en este documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formatos de sus documentos. [W3C08b].

El objetivo de CSS es servir para controlar la presentación de los contenidos escritos en HTML (en cualquiera de sus variantes). De este modo, se consigue la separación de contenido y presentación que ayuda a hacer que las páginas web sean más mantenibles [W3C11]. CSS está pensado para poder ser utilizado no sólo por los programadores sino también por diseñadores, por lo que resulta relativamente sencillo.

Por lo general, CSS es utilizado para maquetar medios continuos como son los navegadores, pero puede utilizarse también para la maquetación de medios paginados.

6.2.5 JavaScript

Javascript es un lenguaje de programación que se utiliza normalmente en el lado del cliente para la interacción de la interfaz. Se trata de un dialecto del estándar EMAScript.

La sintaxis del lenguaje Javascript está basada en la de C, con algunos aspectos como Java, aunque no tiene nada que ver con este otro lenguaje de programación.

Javascript forma parte del trío de lenguajes que permiten la separación de responsabilidades en el cliente que son:

- HTML (XHTML) para los contenidos.
- CSS para la presentación.
- Javascript para la integración.

6.2.6 jQuery

Jquery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interacturar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de neero de 2006 en el BarCamp NYC. Jquery es la biblioteca de JavaScript más utilizada. [Wikipedia].

Se trata de un software libre y de código abierto. Ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código. [Wikipedia].

6.3 Herramientas y Programas Usados para el Desarrollo

6.3.1 Eclipse

Se trata de un entorno de desarrollo integrado (IDE) de código abierto y multiplataforma para desarrollar lo que el proyecto llama “Aplicaciones de Cliente Enriquecido” (RCP), opuesto a las aplicaciones de “Cliente-liviano” basadas en navegadores.

Emplea módulos que implementan diferentes funcionalidades que pueden ser utilizadas en el entorno. De esta forma, es función del usuario la de elegir aquellas que necesita.

La definición proporcionada por el proyecto Eclipse acerca de su software es: “una especie de herramienta universal – un IDE abierto y extensible para todo y nada en particular”.

En este proyecto se utiliza la versión Eclipse Luna SR1 (4.4.1), ya que se trata de la versión más comunmente utilizada por los integrantes del equipo de desarrollo.

6.3.2 Hybris

Se trata de una solución de comercio omnicanal, basada en una sólida arquitectura modular abierta, que satisface todaslas necesidades de ventas y ejecución omnicanal.

Se trata de una arquitectura que responde al cliclo de vida del cliente, una plataforma combinada de comercio y gestión de pedidos.

Tiene facilidad de integración y amplia arquitectura orientada a servicios para trabajar con su infraestructura existente.

Utiliza la arquitectura más sencilla, limpia y novedosa. Resulta sencilla par alos desarrolladores de Spring. La base de su arquitectura es modular. La plataforma de comercio proporciona una estructura de desarrollo a partir de la integrar otras aplicaciones de hybris y diseñar aplicaciones y extensiones personalizadas.

Hybris ha sido desarrollada a partir de una tecnología basada en estándares, lo que rebaja los costes de la gestión de la plataforma y la simplifica. Incorpora estándares conocidos como Srping, SOLR, Groovy y Apache Commons. [Hybris Web].

6.3.3 HAC

La extensión HAC de Hybris, es la Web de administración de la aplicación por defecto de la suite de Comercio de Hybris. Provee la consola de Adminitración Hybris, que ofrece la funcionalidad de administración, monitorización y configuración la suite de Comercio de Hybris.

Además de otras funciones, permite chequear y monitorizar ajustes generales, así como inicializar o actualizar la Suite de Comercio de Hybris. [Wiki Hybris].

En la integración se hace uso de esta extensión, para incluir el Cronjob en la base de datos, y así poder ser utlizado.

6.3.4 HMC

La extensión HMC de Hybris es la interfaz gráfica de usuario de la suite de Comercio de Hybris. A través de ella, puedes llevar a cabo todas las tareas de administración. Se basa en la tecnología WebChips. [Wiki Hybris].

En la integración, se utiliza esta herramienta, para ejecución manual del cronjob de exportación del catalágo de productos. También para modificar sus características y tener un acceso directo a sus logs.

6.3.5 CMSCockpit

Se trata de una aplicación de backend utlizada para administrar el contenido de las páginas web. [Wiki Hybris].

A través de esta herramienta se pueden definir las distintas plantillas de las páginas, modificar su contenido...

En la integración, se hace uso de esta herramienta, para incluir el código HTML donde se insertarán los recomendadores de Hybris, en las páginas donde sean necesarias.

6.3.6 BrainSINS Platform

Se trata de la herramienta Web que provee el Sistema BrainSINS.

A través de ella se lleva a cabo la subida manual del fichero XML de exportación de productos. Así, como se definirá el HTML de los recomendadores y productos recomenados de cada uno de sus recomendadores. Además, con esta herramienta, es posible definir el código de las distintas tiendas, lo que en el caso de DIA, se correspondaré con el dato “language” que se enviará a BrainSINS en su integración.

6.3.7 Oracle SQL Developer

Oracle SQL Developer es un entorno de desarrollo integrado gratuito que simplifica el desarrollo y administración de Bases de Datos Oracle. Ofrece un completo entorno de desarrollo para trabajar con aplicaciones que utilicen PL/SQL, permite ejecutar consultas y scripts, una consola DBA para administrar la base de datos, una interfaz de informes, una solución de modelado de datos completa, y una plataforma de migración para mover una base de datos de terceros a Oracle. [Oracle16].

6.4 Creación del Sistema

Llevar a cabo la integración de este sistema, requiere de un conocimiento relativamente extenso de la solución de comercio electrónico Hybris. Además de un conocimiento amplio y detallado de la estructuración del proyecto concreto DIA, en el que se lleva a cabo la integración.

Como se ha podido observar, esta integración requiere conocer el funcionamiento de las herramientas HAC y HMC, así como de la herramienta WCMS, herramientas propias de la plataforma Hybris.

Por otro lado, ha sido necesario un estudio de la solución de Marketing Automation 'BrainSINS'. Conocer sus requisitos y funcionalidades, para llevar a cabo una correcta integración.

6.4.1 Problemas Encontrados

El desarrollo de esta integración, se inició en un momento en el que llevaba un tiempo relativamente de desarrollo con la tecnología Hybris. Éstos estiman un periodo de tres años de un aprendizaje. Además, aún era menor el tiempo en el que me encontraba como desarrolladora en el equipo de trabajo dedicado al cliente DIA. De este modo, previamente fue necesario un período de adaptación.

Cabe destacar que el desarrollo de esta integración ha sido un proyecto muy enriquecedor, pues requería de un conocimiento tanto de back-end como de front-end.

Por otro lado, RICOH se encontraba en un punto intermedio entre el cliente DIA y la empresa BrainSINS. Esto suponía conocer las necesidades de DIA y sus expectativas, así como conocer las funcionalidades aportadas por BrainSINS y su forma de integración

Uno de los mayores problemas encontrados, fue solventar uno de los requisitos de DIA, que consistía en incorporar la funcionalidad 'añadir al carrito' en todos los productos recomendados por BrainSINS. Funcionalidad que BrainSINS no ofrecía. Esto desembocó en conocer todo el funcionamiento incluido en los carruseles de Hybris, para poder trasladarlo a BrainSINS del modo más transparente posible para la aplicación. Esto se convirtió en todo un reto, pues la repuesta de BrainSINS es un HTML plano, donde es imposible hacerle conocer los productos que el usuario tenía en su carrito, para mostrar correctamente los botones.

Además, en las páginas del checkout ningún tipo de carrusel, mostraba correctamente las unidades del producto, en el caso de que alguno de ellos estuviese en el carrito del usuario. Pues el funcionamiento por defecto era la de basarse en el HTML del componente del minicarrito, que en ese páginas no existe. De este modo, es necesario desarrollar un nuevo tag HTML sencillo, donde solo se incluirá el identificador del productos y sus unidades.

Para dar una solución sencilla a la hora de incluir los carruseles en las diferentes páginas del sitio web, fue necesario llevar a cabo un estudio de la estructuración de las diferentes páginas, su organización, así como el conocimiento de los distintos tipos de slots y tipos de

componentes. El propio personal de DIA, de forma autónoma, será el encargado de incluir los distintos carruseles en sus páginas.

Otro de los inconvenientes que poseía BrainSINS era que no daba soporte al envío de diferentes precios en función de la tienda. Tras reunirnos con ellos, se optó por utilizar el campo 'language' para la incluir los diferentes precios en función del código de la tienda, dado que DIA no tiene internacionalizadas las descripciones de sus productos.

6.4.2 Descripción Detallada de las Clases

En esta sección se describirán las principales clases que componen el proyecto, separadas por extensiones.

6.4.2.1 Extensión diabrainsins

6.4.2.1.1 Clase BrainsinsCronjob

Se trata del cronjob encargado de llevar a cabo la exportación de los productos del catálogo de productos online de DIA, a un fichero XML que sigue la especificación BrainSINS.ç

Atributos estáticos

LOG: Logger	Log de la clase
XML_EXPORT_LOCATION: String	Contiene la property donde se indica la ubicación del fichero de exportacdión

Atributos privados

brainsinsService: BrainsinsService	Servicio encargado de crear el fichero XML de exportación.
configurationService: ConfigurationService	Servicio utilizado para recuperar la localización donde almacenar el XML de exportación. Se encarga de buscar en los ficheros “project.properties” y “local.properties”.
productFacade: ExtendedProductFacade	Fachada a través de la se consultan el conjunto de productos a exportar.

Métodos públicos

perform: PerformResult	Método encargado de llevar a cabo la ejecución del cronjob.

6.4.2.1.2 Clase BrainsinsService

Se trata de la interfaz del servicio de integración BrainSINS encargado de transformar el listado de productos a exportar en una cadena de texto, con el contenido XML de los productos a exportar, que seguirán la especificación BrainSINS.

Métodos

convertProductsToXml: String	Se encarga de convertir el listado de productos DiaProductModel a un String con el contenido XML que sigue la especificación BrainSINS.

6.4.2.1.3 Clase DefaultBrainsinsService

Esta clase implementa la interfaz BrainsinsService

Atributos privados

convertObjectsToXML onvertObjectsToXML	Clase proporcionada por Hybris. Se encargará de convertir el objeto Recsins a XML.
converter<Collection<DiaProductModel >, Recsins> recsinsConverter	Convierte una colección de objetos DiaProductModel a un objeto Recsins.

Métodos públicos

convertProductsToXml: String	convertProductsToXml: String

Métodos privados

normalizeString: String	Se encarga de resolver un problema con el XML que la jaxb no soporta directamente. Es utilizado por le método ConvertProductsToXml: String.

6.4.2.1.4 Clase RecsinsConverter

Se encarga de convertir un listado de objetos DiaProductModel en un objeto Recsins.

Atributos privados

converter<Collection<DiaProductModel>, Entities> entitiesConverter	converter<Collection<DiaProductModel>, Entities> entitiesConverter

Métodos públicos

populate:void	populate:void

6.4.2.1.5 Clase EntitiesConverter

Se encarga de convertir un listado de objetos DiaProductModel a un objeto Entities

Atributos privados

converter<DiaProductModel, Entity> entityConverter	converter<DiaProductModel, Entity> entityConverter

Métodos públicos

populate:void	populate:void

6.4.2.1.6 Clase EntityConverter

Se encarga de convertir un objeto DiaProductModel a un objeto Entity.

Atributos privados

Logger LOG	Log de la clase
String SECTION_MISC_START	Contenido del inicio de la sección 'misc'
String SECTION_MISC_END	Contenido del final de la sección 'misc'
String GROUP_PREFIX	Prefijo del grupo de precios que almacena el id de la tienda que está relacionado con un precio
UrlResolver<ProductModel> productModelUrlResolver	Factoría de precios
PriceDataFactory priceDataFactory	Factoría de precios
CommonI18NService commoni18NService	Servicio i18n
ConfigurationService configurationService	Se utiliza para consultar la configuración de la URL base del sitio web de DIA

Métodos públicos

populate: void	Se encarga de popular un objeto DiaProductModel a un objeto Entity

Métodos privados

getPropertyType: property	Genera un objeto Property a partir del parámetros name y value
getNestedPropertyType: NestedProperty	Genera un objeto NestedProperty a partir de los parámetros lang y value
getShopCodeFromUserGroup: String	Retorna un string con el código de la tienda a partir del parámetro usergroup.
getProductUrl: String	Retorna la url absoluta del producto que recibe como parámetro
getProductImageUrl: String	Retorna la url absoluta de la imagen del producto que recibe como parámetro

getSiteBaseUrl:String	Retorna la url base del siito web de DIA
getPriceMultiPropertyType; Multiproperty	Se encarga de crear un objeto Multiproperty con el atributo propertyName y el producto recibidos como parámetros
getPriceRow: PriceData	Retorna el precio de un producto a partir de los parámetros product y priceRow
getPropertyPrice: Property	Crea un objeto Property a partir de los parámetros propertyName y multiprice
getCategoriesPropertyType: Property	Crea un objeto property con las categorías del producto en cada uno de los idiomas disponibles
getMiscPropertyType: Property	Crear un objeto property con la información extra a partir de los parámetros propertyName y product. Por ejemplo, en ellas se incluyen las propiedades necesarias para la funcionalidad 'añadir al carrito', o mostrar las valoraciones de los usuarios
getAverageProductReviewsApproved: String	Se encarga de calcular la media de las valoraciones obtenidas por un producto
getProductReviewsApproved: List<Double>	Retorna e listado de las valoraciones aprobadas para un producto

6.4.2.1.7 Clase ObjectFactory

Se trata de la factoría para la creación de los objetos del XML. Se trata de una clase generada por JAXB a partir del fichero XSD.

Métodos públicos

createRecsins: Recsins	Crea un objeto Recsins
createEntities: Entities	Crea un objeto Entities
createNestedProperty: NestedProperty	Crea un objeto NestedProperty
createMultiProperty: MultiProperty	Crea un objeto MultiProperty
createEntity: Entity	Crea un objeto Entity

6.4.2.1.8 Clase Recsins

Se trata de una clase generada por JAXB a partir del fichero XSD. Esta clase contendrá a su vez un objeto Entities.

Atributos protegidos

Entities entities	XmlElement
String version	XmlAttribute

Métodos públicos

GetEntities: Entities	Retorna el valor de la propiedad Entities
SetEntities: void	Modifica el valor de la propiedad Entities
GetVersion: String	Retorna el valor de la propiedad versión
SetVersion: void	Modifica el valor de la pripiedad version

6.4.2.1.9 Clase Entities

Se trata de una clase generada por JAXB a partir del fichero XSD. Esta clase contendrá a su vez una lista de objetos Entity.

Atributos protegidos

List<Entity> entity	@XmlElement

Métodos públicos

GetEntity: List<Entity>	Retorna el valor de la propiedad entity

6.4.2.1.10 Clase Entity

Se trata de una clase generada por JAXB a partir del fichero XSD. Esta clase se corresponde con un objeto DiaProductModel según el formato BrainSINS.

Atributos protegidos

List<Property> property	@XmlElement
List<MultiProperty> multiProperty	@XmlElement
String name	@XmlAttribute

Métodos públicos

getProperty: List<Property>	Retorna el listado de las propiedades del producto
getMultiProperty: List<Multiproperty>	Retorna el listado de las propiedades múltiples del producto
getName: String	Retorna el valor de la propiedad name.
setName: void	Modifica el valor de la propiedad name

6.4.2.1.11 Clase Property

Se trata de una clase generada por JAXB a partir del fichero XSD. Cada propiedad está formado por un nombre y un valor.

Atributos protegidos

String value	@XmlValue
String name	@XmlAttribute

Métodos públicos

getValue: String	Devuelve el valor de la propiedad value
setValue: void	Modifica el valor de la propiedad value
getName: String	Devuelve el valor de la propiedad name
setName: void	Modifica el valor de la propiedad name

6.4.2.1.12 Clase MultiProperty

Se trata de una clase generada por JAXB a partir del fichero XSD. Cada multiproperty está formado por un listado de nestedproperties y un nombre.

Atributos protegidos

List<NestedProperty> property	@XmlElement
String name	@XmlAttribute

Métodos públicos

getProperty: List<NestedProperty>	Retorna el valor de la propiedad property
getName: String	Retorna el valor de la propiedad name
setName: void	Modifica el valor de la propiedad name

6.4.2.1.13 Clase NestedProperty

Se trata de una clase generada por JAXB a partir del fichero XSD. Cada nestedproperty está formado por atributo value y lang. En nuestro el atributo lang, se corresponde con el código de la tienda donde se oferta el producto.

Atributos protegidos

String value	@XmlValue
String lang	@XmlAttribute

Métodos públicos

getValue: String	Retorna el valor de la propiedad value
setValue: void	Modifica el valor de la propiedad value
getLang: String	Retorna el valor de la propiedad lang
setValue: void	Modifica el valor de la propiedad lang

6.4.2.2 Extensión diaacceleratorstorefront

6.4.2.2.1 Clase HomePageController

Se trata del controlador dedicado a la página home. En él ha sido necesario incluir el servicio BrainsinsStorefrontService.

Atributos privados

ExtendedBrainsinsStorefrontService brainsinsStorefrontService	Servicio dedicada a la integración de brainsins en el front

Métodos públicos

home: String	Muestra la página home

6.4.2.2.2 Clase ProductPageController

Se trata del controlador dedicado a la página de detalle de producto. En él ha sido necesario incluir el servicio BrainsinsStorefrontService.

Atributos privados

ExtendedBrainsinsStorefrontService brainsinsStorefrontService	Servicio dedicada a la integración de brainsins en el front

Métodos públicos

productDetail: String	Muestra la página del detalle de un producto

6.4.2.2.3 Clase CategoryPageController

Se trata del controlador dedicado a la página de categorías. En él ha sido necesario incluir el servicio BrainsinsStorefrontService.

Atributos privados

ExtendedBrainsinsStorefrontService brainsinsStorefrontService	Servicio dedicada a la integración de brainsins en el front

Métodos públicos

Category: String	Muestra la página de categorías

6.4.2.2.4 Clase CartPageController

Se trata del controlador dedicado a la página donde se muestra el contenido del carrito del cliente. En él ha sido necesario incluir el servicio BrainsinsStorefrontService.

Atributos privados

ExtendedBrainsinsStorefrontService brainsinsStorefrontService	Servicio dedicada a la integración de brainsins en el front

Métodos públicos

showCart: String	Muestra la página del carrito

Métodos protegidos

CreateProductList: void	Se encarga de generar el listado de productos que

	posee el usuario en su carrito. Es en este punto donde se llama al servicio de Brainsins.
PrepareDataForPage: void	Se encarga de llevar a cabo la preparación de la vista de la página, guardando en el modelo la información necesaria.

6.4.2.2.5 Clase MultiPaymentCheckoutController

Se trata del controlador dedicado a la página de confirmación del pedido del cliente. En él ha sido necesario incluir el servicio BrainsinsStorefrontService.

Atributos privados

ExtendedBrainsinsStorefrontService brainsinsStorefrontService	Servicio dedicada a la integración de brainsins en el front

Métodos públicos

orderConfirmationPayPal: String	Muestra la página de confirmacion del pedido

6.4.2.2.6 Clase BrainsinsStorefrontService

Se trata de la interfaz del servicio para la integración de BrainSINS en el sitio web de DIA

Métodos públicos

updateModelForDefaultPage: void	Se encarga de actualizar el model, que recibe como parámetro, con la información de BrainSINS en la página por defecto. Normalmente será la página home, o bien otra que no encaje con ninguno de los tipos especificados por brainsins, lo que permitirá incluir todos los recomendadores en cualquier página del sitio web.
updateModelForProductPage: void	Se encarga de actualizar el model, que recibe como parámetro, con la información de BrainSINS en la página de detalle de producto
updateModelForCategroyPage: void	Se encarga de actualizar el model, que reciben como parámetro, con la información de BrainSINS en la página de categorías.
updateModelForCartPage: void	Se encarga de incluir en el model que recibe como parámetro la información de BrainSINS asociada a la página del carrito del usuario.
updateModelForOrderConfirmationPage	Se encarga de incluir en el model que reciben como parámetro la informaición de BrainSINS asociada a la página de confirmación del pedido.

6.4.2.2.7 Clase ExtendedBrainsinsStorefrontService

Se trata de la implementación de la interfaz BrainsinsStorefrontService, dedicado a la integración de BrainSINS en el sitio web de DIA

Atributos privados

SessionService sessionService	Servicio proporcionado por Hybris para recuperar la sesión del usuario
ExtendedBrainsinsStorefrontUtilService brainsinsStorefrontUtilService	Servicio de utilidad para el servicio de brainsins
DiaCustomerFacade customerFacade	Fachada del cliente de dia

Métodos públicos

updateModelForDefaultPage: void	Se encarga de actualizar el model, que recibe como parámetro, con la información de BrainSINS en la página por defecto. Normalmente será la página home, o bien otra que no encaje con ninguno de los tipos especificados por brainsins, lo que permitirá incluir todos los recomenadores en cualquier página del sitio web.
updateModelForProductPage: void	Se encarga de actualizar el model, que recibe como parámetro, con la información de BrainSINS en la página de detalle de producto
updateModelForCategroyPage: void	Se encarga de actualizar el model, que reciben como parámetro, con la información de BrainSINS en la página de categorías.
updateModelForCartPage: void	Se encarga de incluir en el model que recibe como parámetro la información de BrainSINS asociada a la página del carrito del usuario.
updateModelForOrderConfirmationPage	Se encarga de incluir en el model que reciben como parámetro la informaición de BrainSINS asociada a la página de confirmación del pedido.

Métodos privados

getAuxBrainsinsDataCommonForPages: AuxBrainsinsData	Genera la información por defecto de BrainSINS que es necesario incluir en todas las páginas

6.4.2.2.8 Clase BrainsinsStorefrontUtilService

Se trata de la interfaz de la clase de utilidad para el servicio de Brainsins.

Métodos públicos

getToken: String	Devuelve el token que Brainsins ha asignado a DIA y que está asociado al entorno donde se encuentra desplegada la aplicación (local, dev, stag, pro)
getAddButtonText: String	Devuelve el texto internacionalizado del botón 'Añadir'.
getUnavailableButtonText: String	Devuelve el texto internacionalizado del botón 'No disponible'
getCategories: String	Devuelve el nombre de la categoría cuyo código es

	enviado como parámetro
getBrainsinsCart: HashMap<String, Long>	Devuelve un mapa con el listado de los códigos de los productos incluidos en el carrito del usuario, junto a las unidades añadidas
getTotalAmount: Double	Devuelve el precio total del pedido del usuario a partir del parámetro priceData que recibe como parámetro

6.4.2.2.9 Clase ExtendedBrainsinsStorefrontUtilService

Se trata de la implementación de la interfaz del servicio de utilidad Brainsins.

Atributos privados

Logger LOG	Log de la clase
ConfigurationService configurationService	Servicio de configuración Hybris
MessageSource messageSource;	Servicio Hybris de mensajes
I18NService i18NService	Servicio de internacionalización

Métodos públicos

getToken: String	Devuelve el token que Brainsins ha asignado a DIA y que está asociado al entorno donde se encuentra desplegada la aplicación (local, dev, stag, pro)
getAddButtonText: String	Devuelve el texto internacionalizado del botón 'Añadir'.
getUnavailableButtonText: String	Devuelve el texto internacionalizado del botón 'No disponible'
getCategories: String	Devuelve el nombre de la categoría cuyo código es enviado como parámetro
getBrainsinsCart: HashMap<String, Long>	Devuelve un mapa con el listado de los códigos de los productos incluidos en el carrito del usuario, junto a las unidades añadidas
getTotalAmount: Double	Devuelve el precio total del pedido del usuario a partir del parámetro priceData que recibe como parámetro

6.4.2.2.10 Clase AuxBrainsinsData

Se trata de la clase que pertenece al paquete del modelo de brainsins de la extensión diaacceleratorstorefront. Encapsula el conjunto de los datos que es necesario enviar a Brainsins en función de la página en la que se encuentra el usuario. Se añade al model de las distintas páginas del site donde es necesario llevar a cabo la integración con BrainSINS a través del servicio BrainsinsStorefrontService.

Atributos públicos

String token	Token que BrainSINS ha asignado a DIA
String language	Código de la tienda en la que se encuentra el

	usuario
String userEmail	Email del usuario que ha iniciado sesión el el sitio web
String productId	Código identificador de un producto
String totalAmount	Coste total del pedido de un cliente
String typePage	Tipo de página
String categories	Nombre de las categorias internacionalizadas
HashMap<String, Long> cart	Carrito del ususario
String brainsinsAddButtonText	Texto internacionalizado del boton 'Añadir'
String brainsinsUnavailableProductText	Texto internacionalizado del botón 'No disponible'

6.4.2.2.11 Clase BrainsinsStorefrontConstants

Se trata de la interfaz donde se incluyen las constantes para la clase AuxBrainsinsData.

Atributos de la interfaz ModelKey

String auxBrainsinsDataParameter	Identificador del parámetro AuxBrainsinsData en el model de la página

Atributos de la interfacz SessionAttributes

String totalPrice	Identificador del parámetro donde se almacena el precio total del pedido

Atributos de la interfaz Token

String property	Identificador de la property donde se incluye el valor del token de BrainSINS para DIA

Atributos de la interfaz CartIntegration

String unavailableMessage	Identificador de la property donde se define el texto del botón Añadir
String addToCartMessage	Identificador de la property donde se define el texto del botón No disponible

Atributos de la interfaz TypePage

String defaultPage	Identificador de la página home según la especificación de BrainSINS
String productPage	Identificador de la página de detalle de producto según la especificación de BrainSINS
String cartPage	Identificador de la página del carrito según la especificación de BrainSINS
String orderConfirmationPage	Identificador de la página de confirmación del pedido según la especificación de BrainSINS

String categoryPage	Identificador de la página de categrías según la especificación de BrainSINS

6.4.2.2.12 Clase acc.brainsins

Esta clase forma parte del javascript necesario para la integración de BrainSINS en el sitio web de DIA.

Atributos

BrainsinsData	Contiene la información a enviar a Brainsins
BraininsAddToCartFormElements	Parámetros necesarios de incluir en el formulario encargado de añadir los productos recomenados de Brainsins al carrito. Se trata de simular el formulario de añadir al carrito incluido en el site. De esta forma se ejecutará el javascript , acc.cartpopup.js, encargado de llevar a cabo dicha función de forma "transparente" desde el producto recomendado de Brainsins.

Métodos

includeScripts	Incluye el token y la librería de BrainSINS
recommenderNumberByDataName	Recupera el identificador del recomendador a partir del atributo data-name del div Convenio de nombrado : name="recommender-idRecomendadorHerramientaBrainsins"
getRecommenders	Retorna la lista de identificadores de Brainsins a mostrar La lista será vacía en el caso de que la página no incluya elementos dedicados a incluir ningún recomendador de brainsins Para que se incluya un recomendador en la página, ésta debe poseer un div cuyo id debe seguir el convenio de nombrado: id='brainsins-recommender-xx' donde xx es cualquier valor que se desee Utiliza el parámetro recommenderDIVS que contiene el conjunto de elementos div donde se ubicarán los recomendadores
prepareBrainsinsData	Se encarga de llamar a la función adecuada, según el parametro typePage que recibe la función
homePage	Envia a Brainsins los parámetros asociados al tipo de página 'homePage' Incluye el recomendador en el div del HTML
productDetailPage	Envia a Brainsins los parámetros asociados al tipo de página 'product' Incluye el recomendador en el div del HTML
cartPage	Envia a Brainsins los parámetros asociados al tipo de pagina 'checkout' Incluye el recomendador en el div del HTML

checkoutConfirmationPage	Sólo se encarga de enviar a Brainsins los parámetros asociados al tipo de pagina 'thankYou', ya que en está página no se desea mostrar nigún recomendador
categoryPage	Envía a Brainsins los parámetros asociados al tipo de página 'category' Incluye los recomendadores en los DIVS HTML que recibe como parámetro
initCarousel	Inicializa el plugin jquery para el carrusel de los recomendadores
createBrainsinsCart	Crea un carrito del usuario dedicado a braisnins, neceario para implementar adecuadamente en cada uno de los productos recomendados la funcionliada de añadir al carrito.
getProductQuantityInCart	Devuelve las unidades de un producto a partir del id del producto que se le pasa como parámetro
completeFormInformation	En el caso de que el producto no contenga precio o el valor "salesUnit" o bien el valor "soldByType" no se podrá añadir dicho producto al carrito ya que no existe stock en la tienda, o bien no se tiene la información necesaria para incluir el botón añadir al carrito, es decir, el producto no contiene todos los datos necearios en el catálogo.
preparePrice	Modifica el valor del precio a "No disponible" cuando el producto no contega la información necesaria para añadirla al carrito
prepareStars	Se encarga añadir el estilo necesario para que se muestren las estrellitas en un producto en función de las puntuaciones de los usuarios
prepareBrainsinsAddButtonText	Añade al botón de añadir el texto internacionalizado
prepareOutOfStock	Se emcarga de ocultar el botón que se le pasa como parámetro en el caso de que sea necesario, es decir, el producto recibido como respuesta de brainsins no tiene precio. En el caso de que "salesUnit" venga vacío y / o "soldByType" venga vacío, el producto estará mal creado en el catálogo de productos y no se tendrán los valores necesarios para pintar bien los botones de añadir al carrito. De modo que el comportamiento será el mismo que en el caso de que no tenga precio
prepareQuantityByProduct	Se encarga de acutalizar el input con la cantidad del producto añadido al carrito. En el caso de que "salesUnit" venga vacío y / o "soldByType" venga vacío, el producto estará mal creado en el catálogo de productos y no se tendrán los valores necesarios para pintar bien los botones de añadir al carrito. De modo que el comportamiento será el mismo que en el caso de que no tenga precio.

prepareSpanGr	Se encarga de incluir el texto "gr" en caso de que el atributo sodByType sea igual a "WEIGHT"
trackButtonAddToCart	Track user's action on button add to cart
trackClicRecommenderButton	Track user's action on brainsins button add to cart

6.4.2.2.13 Clase acc.cartpopup

El javascript dedicado al funcionamiento de añadir/actualizar el carrito es acc.cartpoup.js. Ha sido necesario actualizarlo para integrar BrainSINS con la funcionalidad de añadir al carrito.

Métodos

updateAllQuantities	Se encarga de actualizar los formularios de los productos mostrados en las páginas. Como por ejemplo los carruseles de productos. Se ejecuta con el acceso a una página. Si la página incluye el minicarrito, utiliza dicho HTML, o bien el componente lightCart.
updateAllQuantitiesFromMiniCart	Se encarga de actualizar los formularios de los productos mostrados en la página, basándose en el HTML del **minicarrito**.
updateAllQuantitiesFromLightCart	Se encarga de actualizar los formularios de los productos mostrados en la página, basándose en el HTML del tag **lightCartEntries**
updateQuantityEventHandler	se ejecuta cuando el usuario hace clic sobre uno de los botones de añadir al carrito. Evento que recoge la función acc.cartpoup.bindCartEvents

6.4.2.3 Extensión diaacceleratorfacades

6.4.2.3.1 Clase DiaProductFacade

Interfaz de la fachada de los productos de Dia

Métodos públicos

findProductsToBrainsins: List<DiaProductModel>	Se encarga de buscar los productos del catálogo online del sitio web de DIA que están aprobados y para los que existe stock. Éstos serán exportados a BrainSINS.

6.4.2.3.2 Clase ExtendedDiaProductFacade

Implementación de la interfaz DiaProductFacade

Atributos privados

ExtendedProductService	Servicio que lleva a cabo operaciones CRUD sobre la

extendedProductService	base de datos de productos

Métodos públicos

findProductsToBrainsins: List<DiaProductModel>	Se encarga de buscar los productos del catálogo online del sitio web de DIA que están aprobados y para los que existe stock. Éstos serán exportados a BrainSINS.

6.4.2.3.3 Clase DiaCustomerFacade

Contrato que amplia el comportamiento de la fachada de clientes por defecto. Ha sido necesario incluir algunos métodos para recuperar información concreta del usuario que se encuentra en el sitio web. De este modo se evita el alto coste temporal de popular toda la información del usuario.

Métodos públicos

getCurrentWarehouseCodeForCurrentCustomer: String	Código de la tienda en la que se encuentra situado el usuario en el sitio web. BrainSINS necesita conocerlo, para mostrar el precio de los productos asociados a la tienda
getVerifiedContactEmailFromCurrentCustomer: String	Retorna el email veficado del usuario que ha iniciado sesión en el sitio web

6.4.2.3.4 Clase ExtendedCustomerFacade

Implemetación de la fachada DiaCustomerFacade

Métodos públicos

getCurrentWarehouseCodeForCurrentCustomer: String	Código de la tienda en la que se encuentra situado el usuario en el sitio web. BrainSINS necesita conocerlo, para mostrar el precio de los productos asociados a la tienda
getVerifiedContactEmailFromCurrentCustomer: String	Retorna el email veficado del usuario que ha iniciado sesión en el sitio web

6.4.2.4 Extensión diaacceleratorcore

6.4.2.4.1 Clase ExtendedProductService

Interfaz del servicio CRUD de producto

Métodos públicos

findProductsToBrainsins	Lleva a cabo la búsqueda de los productos a exportar a BrainSINS.

6.4.2.4.2 Clase DefaultExtendedProductService

Implementación de la interfaz ExtendedProductService.

Atributos privados

ExtendedProductDao extendedProductDao	Operaciones CRUD sobre la tabla de productos

Métodos públicos

findProductsToBrainsins	Lleva a cabo la búsqueda de los productos a exportar a BrainSINS.

6.4.2.4.3 Clase ExtendedProductDao

Interfaz de la clase con acceso directo a la base de datos. Lleva a cabo las operaciones CRUD sobre la base de datos de productos.

Métodos públicos

findProductsToBrainsins	Lleva a cabo la búsqueda de los productos a exportar a BrainSINS.

6.4.2.4.4 Clase DefaultExtendedProductDao

Implementación de la interfaz ExtendedProductDao.

Atributos privados

String APPROVED_PRODUCTS	Consulta de los productos a exportar a BrainSINS

Métodos públicos

findProductsToBrainsins	Lleva a cabo la búsqueda de los productos a exportar a BrainSINS.

Capítulo 7. Desarrollo de las Pruebas

7.1 Pruebas de Integración y del Sistema

Caso de Uso 1: Ejecutar Job	
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob. Este se encuentra correctamente configurado y la base de datos se encuentra disponible	El sistema genera un fichero XML con la exportación del catálogo de productos, siguiendo la especificación de BrainSINS en la localización configurada. El estado del cronjob pasa a ser 'FINISHED'.
	Resultado Obtenido
	El sistema efectivamente genera el fichero de exportación correcto, en la localización configurada.
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob sin tener configurado la ubicación del fichero de la exportación.	El sistema registra el problema encontrado en su log. El sistema cambia el estado del job a 'ABORTED'.
	Resultado Obtenido
	Efectivamente, el sistema registra en el log que no la localización del fichero no ha sido configurada. El estado del croniob pasa a ser 'ABORTED'.
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob cuando la base de datos no se encuentra disponible.	El sistema registra el problema encontrado en su log. El sistema cambia sel estado del cronjob a 'ABORTED'.
	Resultado Obtenido
	Efectivamente, el sistema registra en el log el problema encontrado con la base de datos. El estado del cronjob pasa a ser 'ABORTED'.
Prueba	**Resultado Esperado**

El usuario / sistema ejecuta el cronjob cuando el catálogo online dispone sólo de diez productos aprobados, todos los productos se encuentran a la venta en alguna tienda y además todos los productos han recibido puntuaciones.	El sitema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML generado contiene los diez productos aprobados, donde por cada uno de ellos se indica: 'idProduct': Identificador del producto. 'name': descripción del producto. 'image-Url': url absoluta de la imagen del producto. 'url': url absoluta a la página de detalle del producto. 'categories': categorías a las que pertenece el producto. 'price': precio del producto en una de las tiendas. 'lang': código de la tienda donde se ofrece el producto, estando este disponible para su venta, en el momento de la exportación. A su vez se indica el precio del producto en dicha tienda. 'misc': campo opcional donde se indica: 'unit': la unidad del producto: salesUnit, pieces, gram. 'averageRating': media de las puntuaciones asignadas por los usuarios al producto, encontrándose éstas aprobadas para su visualización. 'salesUnit': unidad de venta: QUANTITY, QUANTITY_WEIGHT, WEIGHT.
	Resultado Obtenido
	Efectivamente, el sistema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML generado contiene los 10 productos aprobados, siendo todos sus campos correctos.
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob cuando el catálogo online dispone sólo de diez productos aprobados, de los que tres no se encuentran a la venta en ninguna tienda.	El sistema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML generado contiene un total de siete productos, es decir, aquellos que se encuentran a la venta en alguna tienda y que están aprobados. Para cada uno de ellos se indican las propiedades indicadas en el caso de uso anterior.
	Resultado Obtenido
	El sistema efectivamente ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML contiene los siete productos que están aprobados y que se encuentran a la venta en alguna tienda. Todas las propiedades de los productos se encuentran correctamente indicadas.
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob cuando el	El sistema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la

catálogo online dispone sólo de diez productos aprobados, de los que tres no se encuentran a la venta en ninguna tienda y además sólo tres han recibido puntuaciones que han sido aprobadas, otro ha recibido puntuación, pero ésta no ha sido aprobada.	localización indicada. El XML generado contiene un total de siete productos, es decir, aquellos que se encuentran a la venta en alguna tienda y que están aprobados. Para cada uno de ellos se indican las propiedades indicadas en el caso de uso anterior. Los productos cuyas puntuaciones han sido aprobadas, tendrán rellenado el campo 'averageRating' con la media de éstas. Áquel producto cuya puntuación no ha sido aprobada, tendrá el campo 'averageRating' con el valor '0.0' al igual que aquellos que no han recibido ninguna puntuación.
	Resultado Obtenido
	El sistema efectivamente ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML contiene los siete productos que están aprobados y que se encuentran a la venta en alguna tienda. Todas las propiedades de los productos se encuentran correctamente indicadas.
Prueba	**Resultado Esperado**
El usuario / sistema ejecuta el cronjob cuando el catálogo online dispone sólo de diez productos aprobados, todos los productos se encuentran a la venta en alguna tienda y además todos los productos han recibido puntuaciones. Los productos tienen diferentes tipos de unidad de peso y unidad de venta.	El sistema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML contiene los diez productos con todas sus propiedades correctamente indicas, siendo las propiedades de peso y unidad de venta correctas en cada uno de los casos
	Resultado Obtenido
	Efectivamente, el sistema ejecuta correctamente el Cronjob, quedando su estado en 'FINISHED'. El XML generado se guarda en la localización indicada. El XML contiene los diez productos que están aprobados y que se encuentran a la venta en alguna tienda. Todas las propiedades de los productos se encuentran correctamente indicadas, incluyendo a las propiedades 'unit' y 'soldByType'.

Caso de Uso 2: Acceso Página Home

Prueba	**Resultado Esperado**
Un cliente que ha iniciado sesión, solicita la página home, encontrándose ésta disponible.	El sistema mostrará la página home. Se notifica a BrainSINS el acceso a la página, enviando el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'home'.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página home y envía a BrainSINS la información adecuada.
Prueba	**Resultado Esperado**
Un cliente que ha iniciado sesión, solicita la página home, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no está disponible.
Prueba	**Resultado Esperado**
Un usuario que no ha iniciado sesión, solicita la página home, encontrándose ésta disponible.	El sistema mostrará la página home. Se notifica a BrainSINS el acceso a la página, enviando el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'home'.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página home y envía a BrainSINS la información adecuada.
Prueba	**Resultado Esperado**
Un usuario que no ha iniciado sesión, solicita la página home, encontrándose ésta no	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.

disponible.	
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no está disponible.

Caso de Uso 3: Acceso Página Detalle de Producto	
Prueba	**Resultado Esperado**
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta disponible.	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'product'.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página de detalle del producto que ha solicitado el usuario. Se envía a BrainSINS la información adecuada.
Prueba	**Resultado Esperado**
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no se encuentra disponible.
Prueba	**Resultado Esperado**
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta disponible.	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'product'.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página de detalle del producto que ha solicitado el usuario. Se envía a BrainSINS la información adecuada.

Prueba	**Resultado Esperado**
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no se encuentra disponible.

Caso de Uso 3: Acceso Página Detalle de Producto	
Prueba	**Resultado Esperado**
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta disponible.	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'product'.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página de detalle del producto solicitada. El sistema envía a BrainSINS los datos correctos.
Prueba	**Resultado Esperado**
El usuario que ha iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no se encuentra disponible.
Prueba	**Resultado Esperado**
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta	Se muestra la página del detalle del producto. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'product'.

disponible.	
	Resultado Obtenido
	Efectivamente, el sistema muestra la página de detalle del producto solicitada. El sistema envía a BrainSINS los datos correctos.
Prueba	**Resultado Esperado**
El usuario que no iniciado sesión, solicita la página de un producto concreta, encontrándose ésta no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible.
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no se encuentra disponible.

Caso de Uso 4: Acceso Página de Categorías	
Prueba	**Resultado Esperado**
El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página disponible.	El sistema muestra los productos asociados a dicha categoría. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, el email del usuario, así como el tipo de página 'category'.
	Resultado Obtenido
	El sistema, efectivamente, muestra la página con los productos correctos. El sistema envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
	Resultado Obtenido
	El sistema, efectivamente redirige al usuario a una página donde se le indica que la página solicitada no se encuentra disponible.
Prueba	**Resultado Esperado**

El cliente que ha iniciado sesión, solicita la página de una categoría, estando la página disponible.	El sistema muestra los productos asociados a dicha categoría. El sistema envía a BrainSINS el código del token asociado a DIA, el código de la tienda en la que se encuentra el usuario, así como el tipo de página 'category'.
	Resultado Obtenido
	El sistema, efectivamente, muestra la página con los productos correctos. El sistema envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El cliente que no ha iniciado sesión, solicita la página de una categoría, estando la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
	Resultado Obtenido
	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible

Caso de Uso 5: Acceso Página del Carrito	
Prueba	**Resultado Esperado**
El cliente que ha iniciado sesión, solicita acceder a su carrito, encontrándose la página disponible.	El sistema muestra la página El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página 'checkout', el email del usuario, el conjunto de productos incluidos en el carrito, así como sus precios y el identificador de la tienda.
	Resultado Obtenido
	El sistema, muestra efectivamente la página del carrito del usuario. El sistema envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El cliente que ha iniciado sesión, solicita acceder a su carrito, encontrándose la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
	Resultado Obtenido
	El sistema efectivamente redirige al usuario a una página donde se indica que la página solicitada no se encuentra

	disponible
Prueba	**Resultado Esperado**
El cliente que no ha iniciado sesión, solicita acceder a su carrito, encontrándose la página disponible.	El sistema muestra la página El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página 'checkout', el conjunto de productos incluidos en el carrito, así como sus precios y el identificador de la tienda.
	Resultado Obtenido
	El sistema, muestra efectivamente la página del carrito del usuario. El sistema envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El cliente que no ha iniciado sesión, solicita acceder a su carrito, encontrándose la página no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
	Resultado Obtenido
	El sistema efectivamente redirige al usuario a una página donde se indica que la página solicitada no se encuentra disponible

Caso de Uso 6: Acceso Página Confirmación de Pedido	
Prueba	**Resultado Esperado**
El cliente finaliza su compra satisfactoriamente, encontrándose la página de confirmación disponible.	El sistema muestra la página de confirmación del pedido. El sistema notifica a BrainSINS el acceso, incluyendo el tipo de página 'thankYou', el email del usuario, el total de la compra y el identificador de la tienda.
	Resultado Obtenido
	Efectivamente, el sistema muestra la página del confirmación del pedido. El sistema en envía a BrainSINS la información esperada.

Prueba	**Resultado Esperado**
El cliente finaliza su compra satisfactoriamente, encontrándose la página de confirmación no disponible.	El sistema mostrará una página indicando que la página solicitada no se encuentra disponible
	Resultado Obtenido
	El sistema efectivamente redirige al usuario a una página donde se indica que la página no se encuentra disponible

Caso de Uso 7: Añadir Producto al Carrito	
Prueba	**Resultado Esperado**
El cliente añade un producto del que existe stock en su carrito.	El sistema añade el producto en el carrito del usuario. Se recalcula el total del pedido. El sistema cambia el formulario de interacción con el carrito ocultando el botón “Añadir” y mostrando los botones '+' y '-' junto a las unidades añadidas al carrito. El sistema envía a BrainSINS el código de producto añadido, la unidad añadida y su precio
	Resultado Obtenido
	Efectivamente, el sistema añade el producto al carrito y su precio total es recalculado. El sistema actualiza el formulario del producto como se esperaba
Prueba	**Resultado Esperado**
El cliente solicita añadir un producto en su carrito del que no existe stock.	El sistema notifica al usuario que no existe stock. El sistema no altera el carrito del usuario. El producto continúa mostrando el botón 'Añadir'
	Resultado Obtenido
	Efectivamente, el sistema muestra un pop-up donde se le indica al usuario que no existe stock del producto. El formulario del producto continúa mostrando el botón 'Añadir'

Caso de Uso 8: Añadir Unidades al Carrito	
Prueba	**Resultado Esperado**
El usuario selecciona el botón '+' de un producto del que existe stock	El sistema añade el producto en el carrito del usuario. Se recalcula el total del carrito. El sistema cambia el formulario de interacción con el carrito actualizando las unidades del producto incluidas en el carrito. El sistema envía a BrainSINS el código del producto añadido, la unidad añadida y su precio.
	Resultado Obtenido
	Efectivamente, el sistema añade una unidad al carrito del usuario, se recalcula el total de carrito y se envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El usuario selecciona el botón '+' de un producto del que no existe stock	El sistema notifica al usuario que no existe stock. El sistema no altera el carrito del usuario . El sistema no altera el formulario.
	Resultado Obtenido
	Efectivamente, el sistema muestra un pop-up donde se le indica al usuario que no existe stock del producto. El formulario permanece en el estado esperado.
Prueba	**Resultado Esperado**
El usuario selecciona el botón '+' cuando la base de datos no se encuentra disponible.	El sistema muestra un pop-up donde se indica al usuario que se ha producido un error. El formulario no registra ningún cambio.
	Resultado Obtenido
	Efectivamente, el sistema muestra un pop-up con el mensaje esperado. El formulario del producto no registra ningún cambio.

Caso de Uso 9: Eliminar Unidades del Carrito	
Prueba	**Resultado Esperado**
El usuario selecciona el botón '-' de un producto, del que tenía más de una	El sistema decrementa en una unidad el producto en el carrito del usuario. Se recalcula el total del carrito. El sistema cambia el formulario de interacción con el carrito

unidad en su carrito.	decrementando una unidad. El sistema envía a BrainSINS el código del producto, la unidad decrementada y su precio.
	Resultado Obtenido
	Efectivamente, el sistema resta una unidad al producto en el carrito del usuario. Se recalcula el total de carrito y se envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El usuario selecciona el botón '-' de un producto, del que tenía una sola unidad en su carrito.	El sistema elimina el producto del carrito del usuario. El sistema recalcula el total del carrito. El sistema actualiza el formulario del producto decrementado, mostrando de nuevo el botón 'Añadir'. El sistema envía a BrainSINS el código del producto, la unidad decrementada y su precio.
	Resultado Obtenido
	Efectivamente, el sistema elimina el producto del carrito del usuario. Se recalcula el total de carrito y se envía a BrainSINS la información esperada.
Prueba	**Resultado Esperado**
El usuario selecciona el botón '-' cuando la base de datos no se encuentra disponible.	El sistema muestra un pop-up donde se indica al usuario que se ha producido un error. El formulario no registra ningún cambio.
	Resultado Obtenido
	Efectivamente, el sistema muestra un pop-up con el mensaje esperado. El formulario del producto no registra ningún cambio.

Caso de Uso 10: Inclusión Recomendador	
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de un recomendador.	Se envía a BrainSINS la información asociada la página, junto al identificador del elemento HMTL donde incluir el recomenador, así como el identificador numérico del recomendador. El recomenador se incluye en el lugar especificado.

	A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.
	Resultado Obtenido
	Efectivamente, se envía a BrainSINS la información esperada. El recomendador se sitúa en el lugar especificado y los productos muestran el formulario con la funcionalidad de añadir al carrito.
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de un recomendador, encontrándose el sistema BrainSINS no disponible.	Se envía a BrainSINS la información asociada la página,junto al identificador del elemento HMTL donde incluir el recomenador, así como el identificador numérico del recomendador. BrainSINS no recibe la información. El recomendador no se incluye.
	Resultado Obtenido
	Efectivamente, se envía a BrainSINS la información esperada, aunque ésta no es recibida. El recomendador no se incluye en la página.
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de dos recomendadores, teniendo ambos el mismo identificador HTML	Se envía a BrainSINS la información asociada la página, junto a los identificadores de los elementos HTML donde incluir los recomenadores, así como su identificador numérico. Dado que ambos elementos HTML tienen el mismo identificador, sólo se incluirá en la página el primero de ellos. El recomenador se incluye en el lugar especificado. A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.
	Resultado Obtenido
	El sistema, efectivamente envía a BrainSINS la información esperada. Se incluye el primer recomendador en la página, cuyos productos muestran la funcionalidad de añadir al carrito.
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de dos recomendadores, teniendo ambos distinto identificador HTML, pero mismo identficador	Se envía a BrainSINS la información asociada a la página, junto a los identificadores de los elementos HTML donde incluir los recomenadores, así como su identificador numérico. Dado que ambos elementos tienen el mismo identificador numérico, sólo se incluirá el primero de ellos. El recomendador se incluye en el lugar especificado.

numérico.	A cada uno de los productos recomendados se le incluye la funcionalidad de añadir al carrito.
	Resultado Obtenido
	El sistema, efectivamente envía a BrainSINS la información esperada. Se incluye el primer recomendador en la página, cuyos productos muestran la funcionalidad de añadir al carrito.
Prueba	**Resultado Esperado**
La página mostrada incluye el código HTML para la inclusión de un recomendador, con un identificador incorrecto.	El sistema envía a BrainSINS la información asociada a la página, junto al identificador del recomendador. Dado que el identificador es incorrecto, no se incluirá ningún recomendador en la página.
	Resultado Obtenido
	El sistema, efectivamente envía a BrainSINS la información esperada. No se incluye ningún recomendador en la página.

7.2 Pruebas de Usabilidad y Accesibilidad

7.2.1 Pruebas de Usabilidad

Una vez realizadas las pruebas de usabilidad planteadas en el diseño, se procederá en este apartado a incluir los resultados encontrados. Para ello se incluirán los cuestionarios realizadas por los usuarios y evaluadores de las pruebas. Teniendo en cuenta el objeto a probar, el carrusel BrainSINS.

Criterios	¿Cumplido?
Generales	
¿Cuáles son los objetivos del sitio web? ¿Son concretos y bien definidos? ¿Los contenidos y servicios que ofrece se corresponden con esos objetivos?	**N/A**
¿Tiene una URL correcta, clara y fácil de recordar? ¿Y las URL de sus páginas internas? ¿Son claras y permanentes?	**N/A**
¿Muestra de forma precisa y completa qué contenidos o servicios ofrece realmente el sitio web? El diseño de la página de inicio debe ser diferente al resto de páginas y cumplir la función de 'escaparate' del sitio.	**SI**
¿La estructura general del sitio web está orientada al usuario? Los sitios web deben estructurarse pensando en el usuario, sus objetivos y necesidades. La estructura interna de la empresa u organización, cómo funciona o se organiza no interesan al usuario.	**N/A**

¿El *look & feel* general se corresponde con los objetivos, características, contenidos y servicios del sitio web? Ciertas combinaciones de colores ofrecen imágenes más o menos formales, serias o profesionales.	**SI**
¿Es coherente el diseño general del sitio web? Se debe mantener una coherencia y uniformidad en las estructuras y colores de todas las páginas. Esto sirve para que el usuario no se desoriente en su navegación.	**SI**
¿Es reconocible el diseño general del sitio web? Cuánto más se parezca el sitio web al resto de sitios web, más fácil será de usar.	**SI**
¿El sitio web se actualiza periódicamente? ¿Indica cuándo se actualiza? Las fechas que se muestren en la página deben corresponderse con actualizaciones, noticias, eventos...no con la fecha del sistema del usuario.	**N/A**
Identidad e Información	
¿Se muestra claramente la identidad de la empresa-sitio a través de todas las páginas?	**N/A**
El Logotipo, **¿es significativo, identificable y suficientemente visible?**	**N/A**
El eslogan o *tagline*, **¿expresa realmente qué es la empresa y qué servicios ofrece?**	**N/A**
¿Se ofrece algún enlace con información sobre la empresa, sitio web, 'webmaster',...?	**N/A**
¿Se proporciona mecanismos para ponerse en contacto con la empresa? (email, teléfono, dirección postal, fax...)	**N/A**
¿Se proporciona información sobre la protección de datos de carácter personal de los clientes o los derechos de autor de los contenidos del sitio web?	**N/A**
En artículos, noticias, informes... **¿Se muestra claramente información sobre el autor, fuentes y fechas de creación y revisión del documento?**	**N/A**
Lenguaje y Redacción	
¿El sitio web habla el mismo lenguaje que sus usuarios? Se debe evitar usar un lenguaje corporativista. Así mismo, hay que prestarle especial atención al idioma, y ofrecer versiones del sitio en diferentes idiomas cuando sea necesario.	**NO**
¿Emplea un lenguaje claro y conciso?	**SI**
¿Es amigable, familiar y cercano? Es decir, lo contrario a utilizar un lenguaje constantemente imperativo, mensajes crípticos, o tratar con "desprecio" al usuario.	**N/A**
¿1 párrafo = 1 idea? Cada párrafo es un objeto informativo. Trasmita ideas, mensajes...Se deben evitar párrafos vacíos o varios mensajes en un mismo párrafo.	**N/A**
Rotulado	
Los rótulos, **¿son significativos?** Ejemplo: evitar rótulos del tipo "haga clic aquí".	**N/A**
¿Usa rótulos estándar? Siempre que exista un "estándar" comúnmente aceptado para el caso concreto, como "Mapa del Sitio" o "Acerca de...".	**N/A**

¿Usa un único sistema de organización, bien definido y claro? No se deben mezclar diferentes. Los sistemas de organización son: alfabético, geográfico, cronológico, temático, orientado a tareas, orientado al público y orientado a metáforas.	**SI**
¿Utiliza un sistema de rotulado controlado y preciso? Por ejemplo, si un enlace tiene el rótulo "Quiénes somos", no puede dirigir a una página cuyo encabezamiento sea "Acerca de"	**N/A**
El título de las páginas, **¿Es correcto? ¿Ha sido planificado?** Relacionado con la capacidad para poder buscar y encontrar el sitio *web*.	**N/A**
Estructura y Navegación	
La estructura de organización y navegación, **¿Es la más adecuada?** Hay varios tipos de estructuras: jerárquicas, hipertextual, facetada,...	**N/A**
En el caso de estructura jerárquica, **¿Mantiene un equilibrio entre Profundidad y Anchura?**	**SI**
En el caso de ser puramente hipertextual, **¿Están todos los clúster de nodos comunicados?** Aquí se mide la distancia entre nodos.	**N/A**
¿Los enlaces son fácilmente reconocibles como tales? ¿Su caracterización indica su estado (visitados, activos,...)? Los enlaces no sólo deben reconocerse como tales, sino que su caracterización debe indicar su estado, y ser reconocidos como una unidad	**N/A**
En menús de navegación, **¿Se ha controlado el número de elementos y de términos por elemento para no producir sobrecarga memorística?** No se deben superar los 7±2 elementos, ni los 2 o, como mucho, 3 términos por elemento.	**N/A**
¿Es predecible la respuesta del sistema antes de hacer clic sobre el enlace? Relacionado con el nivel de significación del rótulo del enlace, aunque también con: el uso de globos de texto, información contextual, la barra de estado del navegador,...	**SI**
¿Se ha controlado que no haya enlaces que no llevan a ningún sitio? Enlaces que no llevan a ningún sitio: Los enlaces rotos, y los que enlazan con la misma página que se está visualizando (por ejemplo enlaces a la "home" desde la misma página de inicio)	**SI**
¿Existen elementos de navegación que orienten al usuario acerca de dónde está y cómo deshacer su navegación? ...como *breadcrumbs*, enlaces a la página de inicio,...recuerde que el logo también es recomendable que enlace con la página de inicio.	**N/A**
Las imágenes enlace, **¿se reconocen como clicables? ¿Incluyen un atributo 'title' describiendo la página de destino?** En este sentido, también hay que cuidar que no haya imágenes que parezcan enlaces y en realidad no lo sean.	**NO**
¿Se ha evitado la redundancia de enlaces?	**N/A**
¿Se ha controlado que no haya páginas "huérfanas"? Páginas huérfanas: que aún siendo enlazadas desde otras páginas, éstas no enlacen con ninguna.	**SI**
Layout de la Página	
¿Se aprovechan las zonas de alta jerarquía informativa de la página para contenidos de mayor relevancia? (como por ejemplo la zona central)	**N/A**

¿Se ha evitado la sobrecarga informativa? Esto se consigue haciendo un uso correcto de colores, efectos tipográficos y agrupaciones para discriminar información. Los grupos diferentes de objetos informativos de una página deben ser 7±2.	**SI**
¿Es una interfaz limpia, sin ruido visual?	**SI**
¿Existen zonas en "blanco" entre los objetos informativos de la página para poder descansar la vista?	**SI**
¿Se hace un uso correcto del espacio visual de la página? Es decir, que no se desaproveche demasiado espacio con elementos de decoración, o grandes zonas en "blanco", y que no se adjudique demasiado espacio a elementos de menor importancia.	**SI**
¿Se utiliza correctamente la jerarquía visual para expresar las relaciones del tipo "parte de" entre los elementos de la página? (La jerarquía visual se utiliza para orientar al usuario)	**SI**
¿Se ha controlado la longitud de página? Se debe evitar en la medida de lo posible el *scrolling*. Si la página es muy extensa, se debe fraccionar.	**N/A**
Búsqueda (si es necesario, por la extensión del sitio, incorporar un buscador interno)	
¿Se encuentra fácilmente accesible? Es decir: directamente desde la home, y a ser posible desde todas las páginas del sitio, y colocado en la zona superior de la página.	**N/A**
¿Es fácilmente reconocible como tal?	**N/A**
¿Permite la búsqueda avanzada? (siempre y cuando, por las características del sitio web, fuera de utilidad que la ofreciera)	**N/A**
¿Muestra los resultados de la búsqueda de forma comprensible para el usuario?	**N/A**
¿La caja de texto es lo suficientemente ancha?	**N/A**
¿Asiste al usuario en caso de no poder ofrecer resultados para una consultada dada?	**N/A**
Elementos Multimedia	
¿Las fotografías están bien recortadas? ¿Son comprensibles? ¿Se ha cuidado su resolución?	**SI**
¿Las metáforas visuales son reconocibles y comprensibles por cualquier usuario? (prestar especial atención a usuarios de otros países y culturas)	**N/A**
¿El uso de imágenes o animaciones proporciona algún tipo de valor añadido?	**SI**
¿Se ha evitado el uso de animaciones cíclicas?	**SI**
Ayuda	
Si posee una sección de Ayuda, **¿Es verdaderamente necesaria?** Siempre que se pueda prescindir de ella simplificando los elementos de navegación e interacción, debe omitirse esta sección.	**N/A**
En enlace a la sección de Ayuda, **¿Está colocado en una zona visible y "estándar"?** La zona de la página más normal para incluir el enlace a la sección de Ayuda, es la superior derecha.	**N/A**
¿Se ofrece ayuda contextual en tareas complejas? (transferencias bancarias, formularios de registro...)	**N/A**
Si posee *FAQs*, **¿Es correcta tanto la elección como la redacción de las preguntas? ¿Y las respuestas?**	**N/A**

Accesibilidad (debería cubrirse con los test de Accesibilidad posteriores)	
¿El tamaño de fuente se ha definido de forma relativa, o por lo menos, la fuente es lo suficientemente grande como para no dificultar la legibilidad del texto?	**SI**
¿El tipo de fuente, efectos tipográficos, ancho de línea y alineación empleadas facilitan la lectura?	**SI**
¿Existe un alto contraste entre el color de fuente y el fondo?	**SI**
¿Incluyen las imágenes atributos 'alt' que describan su contenido?	**NO**
¿Es compatible el sitio web con los diferentes navegadores? ¿Se visualiza correctamente con diferentes resoluciones de pantalla? Se debe prestar atención a: *JScript*, *CSS*, tablas, fuentes...	**SI**
¿Puede el usuario disfrutar de todos los contenidos del sitio web sin necesidad de tener que descargar e instalar *plugins* adicionales?	**N/A**
¿Se ha controlado el peso de la página? Se deben optimizar las imágenes, controlar el tamaño del código *JScript*...	**N/A**
¿Se puede imprimir la página sin problemas? Leer en pantalla es molesto, por lo que muchos usuarios preferirán imprimir las páginas para leerlas. Se debe asegurar que se puede imprimir la página (no salen partes cortadas), y que el resultado es legible.	**N/A**
Control y Retroalimentación	
¿Tiene el usuario todo el control sobre el interfaz? Se debe evitar el uso de ventanas pop-up, ventanas que se abren a pantalla completa, banners intrusivos...	**SI**
¿Se informa constantemente al usuario acerca de lo que está pasando? Si el usuario tiene que esperar hasta que se termine una operación, se debe mostrar un mensaje indicándoselo y que debe esperar, con el tiempo de espera estimado o una barra de progreso.	**SI**
¿Se informa al usuario de lo que ha pasado? Por ejemplo, cuando un usuario valora un artículo o responde a una encuesta, se le debe informar de que su voto ha sido procesado correctamente.	**SI**
Cuando se produce un error, **¿se informa de forma clara y no alarmista al usuario de lo ocurrido y de cómo solucionar el problema?** Siempre es mejor intentar evitar que se produzcan errores a tener que informar al usuario del error.	**SI**
¿Posee el usuario libertad para actuar? NO restringir la libertad del usuario: Uso de animaciones que no pueden ser "saltadas", páginas en las que desaparecen los botones de navegación, no impida al usuario poder usar el botón derecho de su ratón...	**N/A**
¿Se ha controlado el tiempo de respuesta? Esto tiene que ver con el peso de cada página (accesibilidad) y tiene relación con el tiempo que tarda el servidor en finalizar una tarea y responder. El tiempo máximo que esperará un usuario son 10 segundos	**SI**

Tal y como se desprende de la tabla anterior, al tratarse la solución de un recomendador de productos, son pocas las pruebas de accesibilidad que pueden aplicarse. De los resultados de

las pruebas, se saca en conclusión que la mayoría de los aspectos son aplicados correctamente, a excepción de tres. El primero, es el relativo al lenguaje, se cumple en el caso del texto de los botones donde se incluye la funcionalidad de añadir los productos al carrito, sin embargo, la descripción de los productos sólo se muestra en español, pues el sitio web, sólo contiene la descripción de los productos en dicho idioma. Por otro lado, las imágenes no disponen del atributo "alt", así como los enlaces tampoco disponen del atributo "title", sin embargo, éstos pueden sacarse en conclusión, ya que se incluye la descripción de los productos.

7.2.2 Pruebas de Accesibilidad

A continuación, se detallan las pruebas de accesibilidad realizadas. La accesibilidad se ha tenido en cuenta a lo largo de todo el proceso de desarrollo.

7.2.2.1 Revisión Preliminar

Previamente a la evaluación de conformidad, se ha realizado una revisión preliminar que permite identificar los aspectos más graves y corregirlos.

7.2.2.2 Checklist del WCAG 1.0

A continuación se muestra una tabla con los resultados de la evaluación manual de accesibilidad.

Puntos de verificación Prioridad 1:

En general (Prioridad 1)	Sí	No	N/A
1.1 Proporcione un texto equivalente para todo elemento no textual (Por ejemplo, a través de "alt", "longdesc" o en el contenido del elemento). *Esto incluye:* imágenes, representaciones gráficas del texto, mapas de imagen, animaciones (Por ejemplo, *GIFs* animados), "applets" y objetos programados, "ascii art", marcos, scripts, imágenes usadas como viñetas en las listas, espaciadores, botones gráficos, sonidos (ejecutados con o sin interacción del usuario), archivos exclusivamente auditivos, banda sonora del vídeo y vídeos.		**X**	
2.1 Asegúrese de que toda la información transmitida a través de los colores también esté disponible sin color, por ejemplo mediante el contexto o por marcadores.			**X**
4.1 Identifique claramente los cambios en el idioma del texto del documento y en cualquier texto equivalente (por ejemplo, leyendas).		**X**	
6.1 Organice el documento de forma que pueda ser leído sin hoja de estilo. Por ejemplo, cuando un documento HTML es interpretado sin asociarlo a una hoja de estilo, tiene que ser posible leerlo.	**X**		
6.2 Asegúrese de que los equivalentes de un contenido dinámico son actualizados cuando cambia el contenido dinámico.			**X**
7.1 Hasta que las aplicaciones de usuario permitan controlarlo, evite provocar destellos en la pantalla.	**X**		
14.1 Utilice el lenguaje apropiado más claro y simple para el contenido	**X**		

de un sitio.			
Y si utiliza imágenes y mapas de imagen (Prioridad 1)	**Sí**	**No**	**N/A**
1.2 Proporcione vínculos redundantes en formato texto para cada zona activa de un mapa de imagen del servidor.			**X**
9.1 Proporcione mapas de imagen controlados por el cliente en lugar de por el servidor, excepto donde las zonas sensibles no puedan ser definidas con una forma geométrica.			**X**
Y si utiliza tablas (Prioridad 1)	**Sí**	**No**	**N/A**
5.1 En las tablas de datos, identifique los encabezamientos de fila y columna.			**X**
5.2 Para las tablas de datos que tienen dos o más niveles lógicos de encabezamientos de fila o columna, utilice marcadores para asociar las celdas de encabezamiento y las celdas de datos.			**X**
Y si utiliza marcos ("frames") (Prioridad 1)	**Sí**	**No**	**N/A**
12.1 Titule cada marco para facilitar su identificación y navegación.			**X**
Y si utiliza "applets" y "scripts" (Prioridad 1)	**Sí**	**No**	**N/A**
6.3 Asegure que las páginas sigan siendo utilizables cuando se desconecten o no se soporten los scripts, *applets* u otros objetos programados. Si esto no es posible, proporcione información equivalente en una página alternativa accesible.			**X**
Y si utiliza multimedia (Prioridad 1)	**Sí**	**No**	**N/A**
1.3 Hasta que las aplicaciones de usuario puedan leer en voz alta automáticamente el texto equivalente de la banda visual, proporcione una descripción auditiva de la información importante de la banda visual de una presentación multimedia.			**X**
1.4 Para toda presentación multimedia dependiente del tiempo (por ejemplo, una película o animación) sincronice alternativas equivalentes (por ejemplo, subtítulos o descripciones de la banda visual) con la presentación.			**X**
Y si todo lo demás falla (Prioridad 1)	**Sí**	**No**	**N/A**
11.4 Si, después de los mayores esfuerzos, no puede crear una página accesible, proporcione un vínculo a una página alternativa que use tecnologías *W3C*, sea accesible, tenga información (o funcionalidad) equivalente y sea actualizada tan a menudo como la página (original) inaccesible.			**X**

Puntos de verificación Prioridad 2:

En general (Prioridad 2)	**Sí**	**No**	**N/A**
2.2 Asegúrese de que las combinaciones de los colores de fondo y primer plano tengan el suficiente contraste para que sean percibidas por personas con deficiencias de percepción de color o en pantallas en blanco y negro [Prioridad 2 para las imágenes. Prioridad 3 para los textos].	**X**		
3.1 Cuando exista un marcador apropiado, use marcadores en vez de imágenes para transmitir la información.			**X**
3.2 Cree documentos que estén validados por las gramáticas formales publicadas.			**X**
3.3 Utilice hojas de estilo para controlar la maquetación y la presentación.	**X**		

3.4 Utilice unidades relativas en lugar de absolutas al especificar los valores en los atributos de los marcadores de lenguaje y en los valores de las propiedades de las hojas de estilo.	**X**		
3.5 Utilice elementos de encabezado para transmitir la estructura lógica y utilícelos de acuerdo con la especificación.			**X**
3.6 Marque correctamente las listas y los ítems de las listas.	**X**		
3.7 Marque las citas. No utilice el marcador de citas para efectos de formato tales como sangrías.			**X**
6.5 Asegúrese de que los contenidos dinámicos son accesibles o proporcione una página o presentación alternativa.			**X**
7.2 Hasta que las aplicaciones de usuario permitan controlarlo, evite el parpadeo del contenido (por ejemplo, cambio de presentación en periodos regulares, así como el encendido y apagado).	**X**		
7.4 Hasta que las aplicaciones de usuario proporcionen la posibilidad de detener las actualizaciones, no cree páginas que se actualicen automáticamente de forma periódica.			**X**
7.5 Hasta que las aplicaciones de usuario proporcionen la posibilidad de detener el redireccionamiento automático, no utilice marcadores para redirigir las páginas automáticamente. En su lugar, configure el servidor para que ejecute esta posibilidad.			**X**
10.1 Hasta que las aplicaciones de usuario permitan desconectar la apertura de nuevas ventanas, no provoque apariciones repentinas de nuevas ventanas y no cambie la ventana actual sin informar al usuario.			**X**
11.1 Utilice tecnologías *W3C* cuando estén disponibles y sean apropiadas para la tarea y use las últimas versiones que sean soportadas.	**X**		
11.2 Evite características desaconsejadas por las tecnologías *W3C*.	**X**		
12.3 Divida los bloques largos de información en grupos más manejables cuando sea natural y apropiado.	**X**		
13.1 Identifique claramente el objetivo de cada vínculo.	**X**		
13.2 Proporcione metadatos para añadir información semántica a las páginas y sitios.			**X**
13.3 Proporcione información sobre la maquetación general de un sitio (por ejemplo, mapa del sitio o tabla de contenidos).			**X**
13.4 Utilice los mecanismos de navegación de forma coherente.	**X**		
Y si utiliza tablas (Prioridad 2)	**Sí**	**No**	**N/A**
5.3 No utilice tablas para maquetar, a menos que la tabla tenga sentido cuando se alinee. Por otro lado, si la tabla no tiene sentido, proporcione una alternativa equivalente (la cual debe ser una versión alineada).	**X**		
5.4 Si se utiliza una tabla para maquetar, no utilice marcadores estructurales para realizar un efecto visual de formato.			**X**
Y si utiliza marcos ("frames") (Prioridad 2)	**Sí**	**No**	**N/A**
12.2 Describa el propósito de los marcos y cómo éstos se relacionan entre sí, si no resulta obvio solamente con el título del marco.			**X**
Y si utiliza formularios (Prioridad 2)	**Sí**	**No**	**N/A**
10.2 Hasta que las aplicaciones de usuario soporten explícitamente la asociación entre control de formulario y etiqueta, para todos los controles de formularios con etiquetas asociadas implícitamente, asegúrese de que la etiqueta está colocada adecuadamente.		**X**	
12.4 Asocie explícitamente las etiquetas con sus controles.		**X**	

Y si utiliza "applets" y "scripts" (Prioridad 2)	Sí	No	N/A
6.4 Para los *scripts* y *applets*, asegúrese de que los manejadores de eventos sean independientes del dispositivo de entrada.			X
7.3 Hasta que las aplicaciones de usuario permitan congelar el movimiento de los contenidos, evite los movimientos en las páginas.			X
8.1 Haga los elementos de programación, tales como *scripts* y *applets*, directamente accesibles o compatibles con las ayudas técnicas [Prioridad 1 si la funcionalidad es importante y no se presenta en otro lugar; de otra manera, Prioridad 2].			X
9.2 Asegúrese de que cualquier elemento que tiene su propia interfaz pueda manejarse de forma independiente del dispositivo.	X		
9.3 Para los "scripts", especifique manejadores de evento lógicos mejor que manejadores de eventos dependientes de dispositivos.			X

Puntos de verificación Prioridad 3:

En general (Prioridad 3)	Sí	No	N/A
4.2 Especifique la expansión de cada abreviatura o acrónimo cuando aparezcan por primera vez en el documento.			X
4.3 Identifique el idioma principal de un documento.		X	
9.4 Cree un orden lógico para navegar con el tabulador a través de vínculos, controles de formulario y objetos.	X		
9.5 Proporcione atajos de teclado para los vínculos más importantes (incluidos los de los mapas de imagen de cliente), los controles de formulario y los grupos de controles de formulario.			X
10.5 Hasta que las aplicaciones de usuario (incluidas las ayudas técnicas) interpreten claramente los vínculos contiguos, incluya caracteres imprimibles (rodeados de espacios), que no sirvan como vínculo, entre los vínculos contiguos.			X
11.3 Proporcione la información de modo que los usuarios puedan recibir los documentos según sus preferencias (por ejemplo, idioma, tipo de contenido, etc.).			X
13.5 Proporcione barras de navegación para destacar y dar acceso al mecanismo de navegación.			X
13.6 Agrupe los vínculos relacionados, identifique el grupo (para las aplicaciones de usuario) y, hasta que las aplicaciones de usuario lo hagan, proporcione una manera de evitar el grupo.			X
13.7 Si proporciona funciones de búsqueda, permita diferentes tipos de búsquedas para diversos niveles de habilidad y preferencias.			X
13.8 Localice la información destacada al principio de los encabezamientos, párrafos, listas, etc.			X
13.9 Proporcione información sobre las colecciones de documentos (por ejemplo, los documentos que comprendan múltiples páginas).			X
13.10 Proporcione un medio para saltar sobre un *ASCII art* de varias líneas.			X
14.2 Complemente el texto con presentaciones gráficas o auditivas cuando ello facilite la comprensión de la página.			X
14.3 Cree un estilo de presentación que sea coherente para todas las páginas.			X
Y si utiliza imágenes o mapas de imagen (Prioridad 3)	**Sí**	**No**	**N/A**

1.5 Hasta que las aplicaciones de usuario interpreten el texto equivalente para los vínculos de los mapas de imagen de cliente, proporcione vínculos de texto redundantes para cada zona activa del mapa de imagen de cliente.			**X**
Y si utiliza tablas (Prioridad 3)	**Sí**	**No**	**N/A**
5.5 Proporcione resúmenes de las tablas.			**X**
5.6 Proporcione abreviaturas para las etiquetas de encabezamiento.			**X**
10.3 Hasta que las aplicaciones de usuario (incluidas las ayudas técnicas) interpreten correctamente los textos contiguos, proporcione un texto lineal alternativo (en la página actual o en alguna otra) para *todas* las tablas que maquetan texto en paralelo, en columnas de palabras.			**X**
Y si utiliza formularios (Prioridad 3)	**Sí**	**No**	**N/A**
10.4 Hasta que las aplicaciones de usuario manejen correctamente los controles vacíos, incluya caracteres por defecto en los cuadros de edición y áreas de texto.	**X**		

7.3 Pruebas de Rendimiento

Durante el desarrollo del presente proyecto, tuvo lugar una auditoría Hybris, donde se llevaron a cabo diferentes pruebas de rendimiento. Uno de los inconvenientes de la aplicación en general, era el elevado número de consultas del carrito, lo que tenía como consecuencia un alto coste temporal, sobre todo, en los métodos populator. Debido a esto, en servicio de Brainsins dedicado a la integración en el front, se aprovecha de la información, que ha sido necesaria consultar en el método del controlador desde donde se invoca, evitando el acceso a la base de datos, de forma específica para este elemento.

Desde el punto de vista de los carruseles de Brainsins, fue necesario configurar en el plugin Owl Carousel, el lazy loading de las imágenes. De este modo, páginas como la 'home' que posee una media de 47 imágenes, tardará mucho menos tiempo en cargar, al llevar a cabo las peticiones de las imágenes que son visibles.

Capítulo 8. Manuales del Sistema

8.1 Manual de Instalación

Para el funcionamiento del sistema es necesario el siguiente software:

1. Sistema Hybris versión 5.2.

2. Base de datos de Oracle.

3. Servidor de aplicaciones Tomcat.

4. Programa FortiClient con el acceso configurado a la VPN de DIA.

Se parte de la base de que el usuario dispone de un sistema Hybris con su licencia.

8.1.1 Preparación del entorno

Se comienza con la inicialización del sistema Hybris. Para ello, una vez arrancado, se accede a la herramienta HAC a través de la URL "http://localhost:9001/hac". Una vez en él se accede a la pestaña 'Update' – Inicialization.

Tras la inicialización, que lleva un tiempo aproximado de dos horas, el sistema estará preparado para su uso.

A continuación, es necesario llevar a cabo una sincronización de los catálogos del site de DIA, así como una indexación solr.

Luego, será necesario configurar la VPN de DIA, para que podamos recibir las respuestas de BrainSINS.

8.2 Manual de Ejecución

Para comenzar será necesario ejecutar el cronjob en el sistema Hybris, encargado de llevar a cabo la exportación de los productos al fichero XML. Para poder llevar a cabo su ejecución, en el caso de que el cronjob no se encuentre incluido en e sistema, será necesario importarlo con su impex correspondiente por medio de la herramienta HAC que proporciona Hybris. Una vez incluido, se podrá ejecutar desde la herramienta HMC proporcionada por Hybris, desde System – Cronjobs. En ella buscamos el cronjob encargad de la exportación cuyo identificador es 'defaultBrainsinsCronjob'. Una vez en él, seleccionamos la opción 'Start'. Una vez finalizado, el fichero se encontrará en la ruta especificada 'NFS_DATA/transfer....'.

A continuación, será necesario publicar el fichero XML en un servidor público, para que BrainSINS pueda acceder a él. En el caso del entorno de producción, este fichero se genera en

una compartida por ambos nodos, la carpeta NFS_DATA. Desde el sistema operativo Linux donde se encuentra desplegada la aplicación, se crea un enlace simbólico desde la carpeta brainsins del front hasta la carpeta NFS_DATA.

La subida del fichero será llevada a cabo desde la herramienta proporconada por Hybris. En el caso de producción ésta subida se lleva a cabo de forma automática todos los días a las 7.15 de la mañana, una vez se ha recibido el fichero AS400, con todos los productos.

Una vez hecho esto, las páginas de DIA podrán mostrar los recomendadores en aquellas páginas donde así se ha especificado.

8.3 Manual de Gestor DIA

8.3.1 BrainSINS

8.3.1.1 Objetivo

El objetivo de este documento, es servir de manual a los gestores que trabajarán en el mantenimiento de las páginas del CMS, en las que se incluirán los recomendadores proporcionados por la herramienta BrainSINS.

Este documento, se encargará de mostrar la funcionalidad de dicha herramienta. Además, indicará los diferentes recomendadores que ofrece.

Así mismo, se mostrará de forma detallada y por medio de un ejemplo práctico, la inclusión de un recomendador en una de las páginas solicitadas por DIA%.

8.3.1.2 Descripción

BrainSINS, se trata de una herramienta indicada para mejorar la experiencia del usuario, lo que irá unido a un aumento de las ventas en aquellas páginas que lo incluyan. Para ello, se encarga de generar diferentes recomendadores de productos.

Un recomendador, consiste en la muestra de un conjunto de productos que tienen una alta probabilidad de ser comprados por el cliente, una vez le son mostrados. Actualmente, BrainSINS incluye diez tipos de recomendadores asociados al tipo de página en la que serán incluidos.

8.3.1.3 Integración

La integración de los recomendadores de BrainSINS, se llevará a cabo desde la herramienta **cmscockpit** del entorno DIA. Para ello, será necesario actualizar la plantilla CMS de la página donde incluir el recomendador.

8.3.2 Recomendadores BrainSINS

8.3.2.1 Listado

BrainSINS, incluye por defecto una serie de recomendadores, agrupados en función del tipo de página. El listado de recomendadores, acompañados de su identificador numérico, se muestra a continuación:

Recomendadores en Inicio:

1. Basado en tu historial de navegación
2. Los más vendidos
3. Recomendaciones de nuevos productos

Recomendadores en Producto:

4. Los clientes que compraron este producto también compraron
5. Se suelen comprar juntos frecuentemente
6. Productos en categorías similares

Recomendadores en Carrito:

7. Los clientes que compraron productos que aparecen en tu cesta de la compra también compraron
8. Productos que se compran junto con el último añadido al carrito

Recomendadores en Checkout:

9 .Se suelen comprar juntos

10. Productos vistos recientemente

Cualquier tipo de recomendador puede ser incluido en cualquier página. La agrupación anterior, es la aconsejada por BrainISNS.

8.3.3 Integración vía CMS

8.3.3.1 Convenio del Recomendador

8.3.3.1.1 Descripción

Para la integración de un recomendador vía CMS, es necesario añadir un componente de tipo **párrafo**, cuyo contenido estará formado por un **div** compuesto de un **identificador** y un atributo denominado data-name.

El **identificador**, deberá comenzar obligatoriamente por la cadena de texto que se indica a continuación: 'brainsins-recommender'

El atributo data-**name** sigue el patrón de nombrado indicado a continuación:

'recommender-x'

En el que x hace referencia al identificador numérico del recomendador dentro de la herramienta de BrainSINS. Los identificadores se indican en el apartado 2.1 Listado de este manual.

8.3.3.1.2 Ejemplo

Como resultado, el contenido del elemento párrafo a incluir en la página home, utilizando el recomendador de identificador 1, sería el siguiente:

En el caso de que se deseen incluir **varios recomendadores**, los elementos div del HTML deberán poseer distintos valores, pero siguiendo siempre el patrón de nombrado. Un ejemplo sería:

8.3.4 Caso práctico

8.3.4.1 Descripción

A continuación, se muestran los pasos necesarios para incluir el recomendador de BrainSINS en la primera pestaña de la página Home en el entorno de dev, por medio de la herramienta CMS.

8.3.4.1.1 PASO 1

Acceder a la herrameinta CMS del entorno http://10.32.106.210:9001/cmscockpit/index.zul

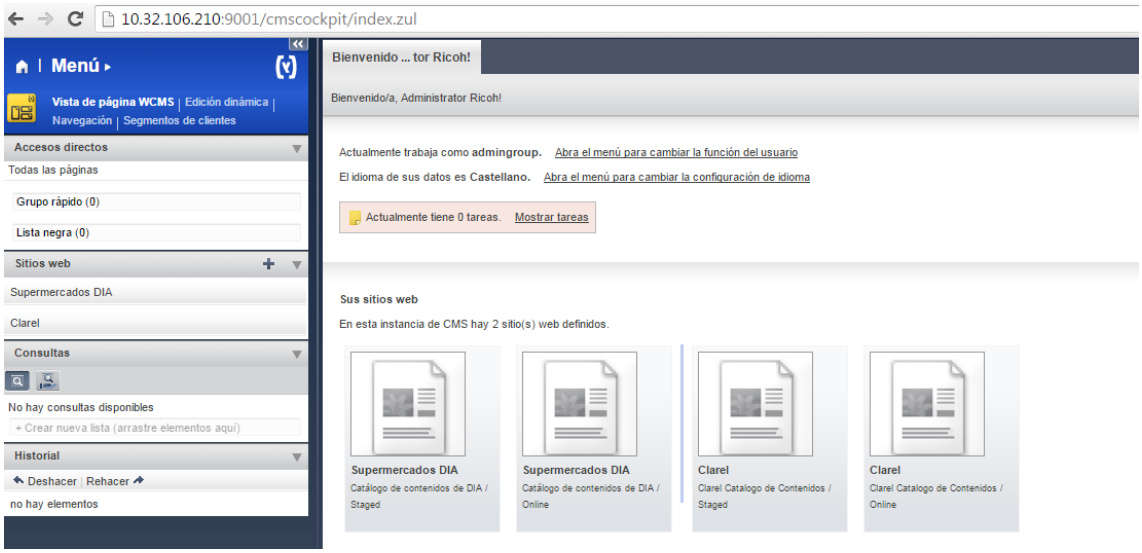


Ilustración 76: Integración CMS – PASO 1

8.3.4.1.2 PASO 2

Búsqueda de la plantilla CMS del catálogo Staged en la que se desea incluir el recomendador.

Para este ejemplo, se busca la página Home nueva.

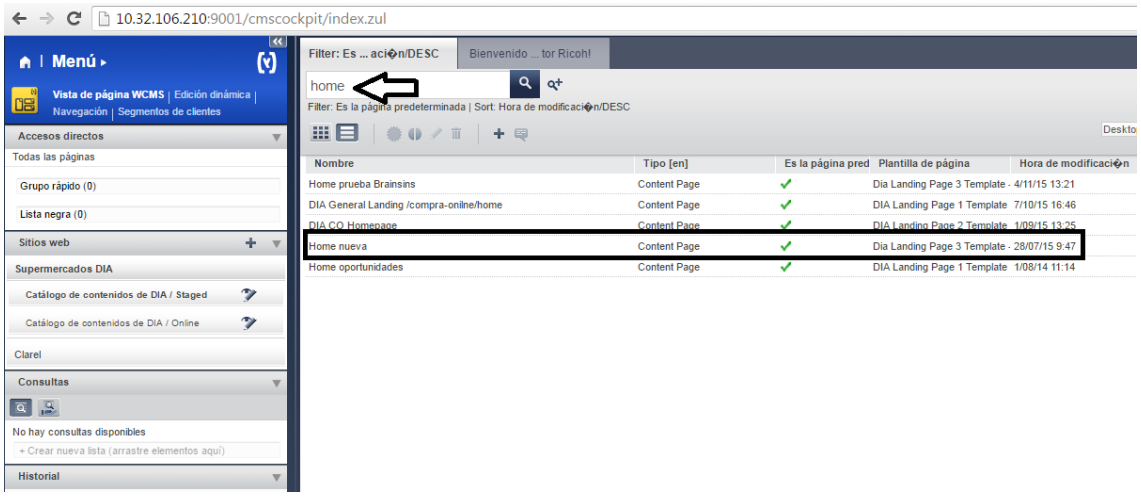


Ilustración 77: Integración CMS – Paso 2

8.3.4.1.3 PASO 3

Editar la plantilla de la página de la página Home nueva haciendo doble clic sobre su nombre.

8.3.4.1.4 PASO 4

Localizar el slot donde se desea insertar el recomendador y hacer clic sobre su botón +.

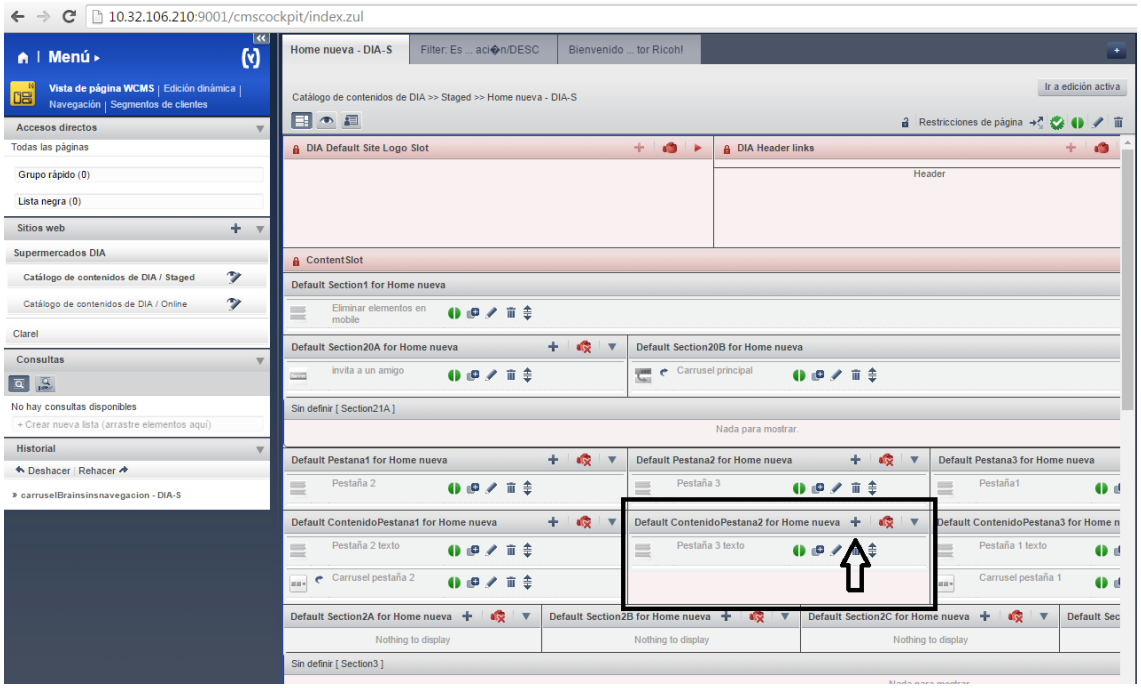


Ilustración 78: Integración CMS – Paso 4

8.3.4.1.5 PASO 5

En el pop-up, seleccionar el tipo de componente de tipo *Párrafo*.

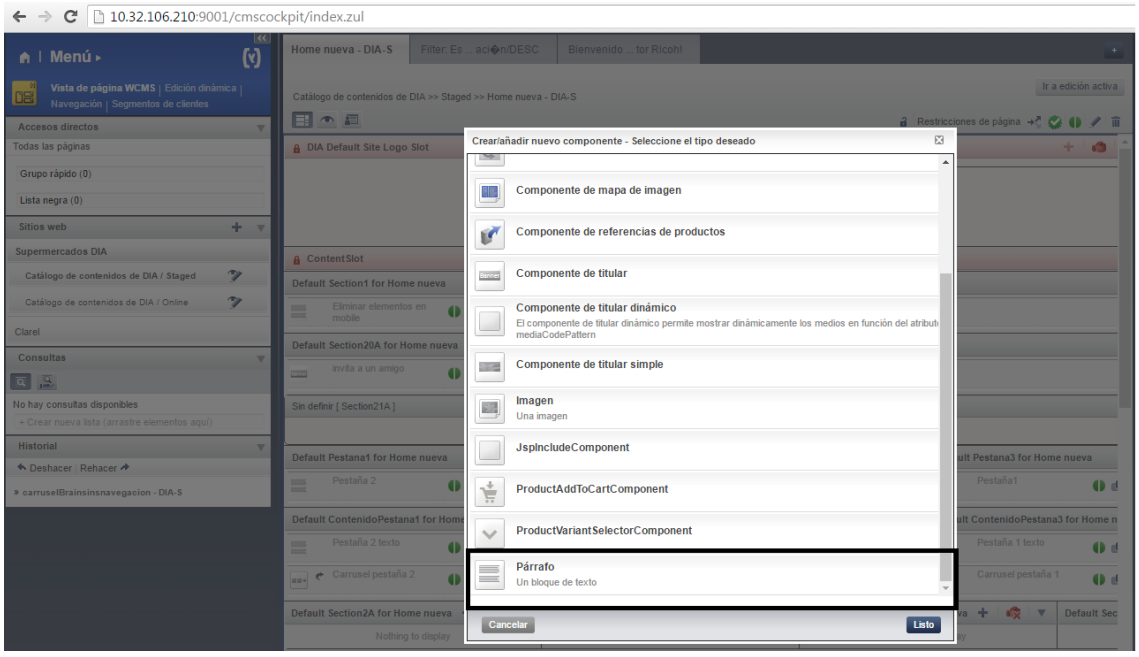


Ilustración 79: Integración CMS – Paso 5

8.3.4.1.6 PASO 6

En la siguiente ventana, seleccionar la opción Cree un nuevo elemento.

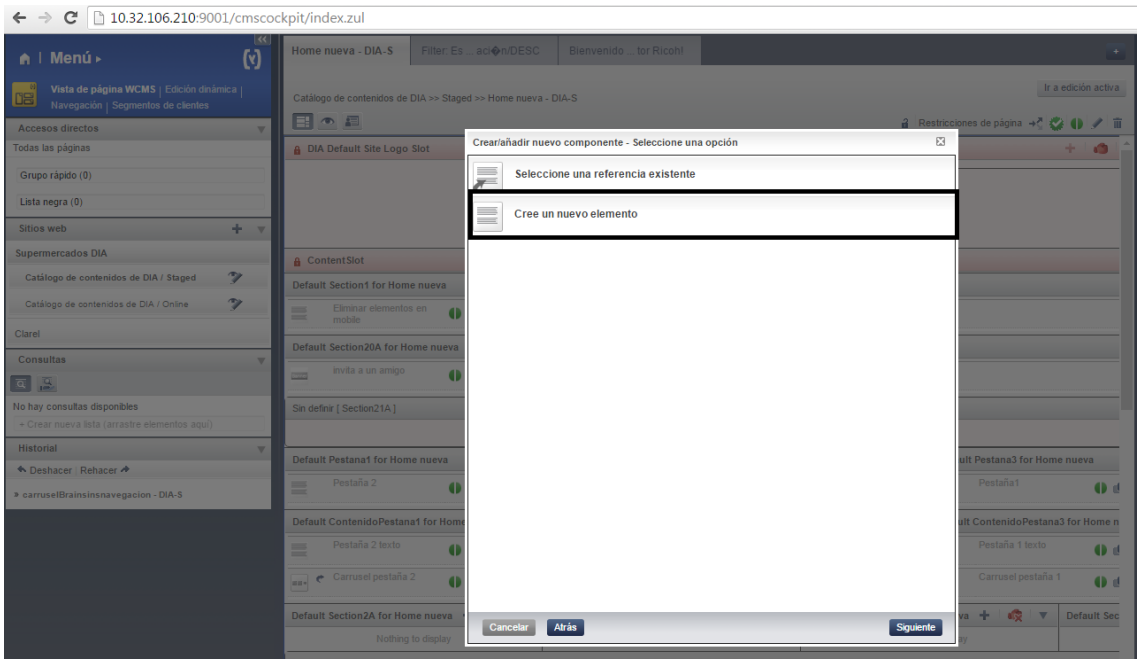


Ilustración 80: Integración CMS – Paso 6

8.3.4.1.7 PASO 7

A continuación, asignar un nombre al nuevo componente a añadir, por ejemplo, *BrainsinsRecommender1Home*.

Una vez incluido, hacer clic sobre el botón *Listo*.

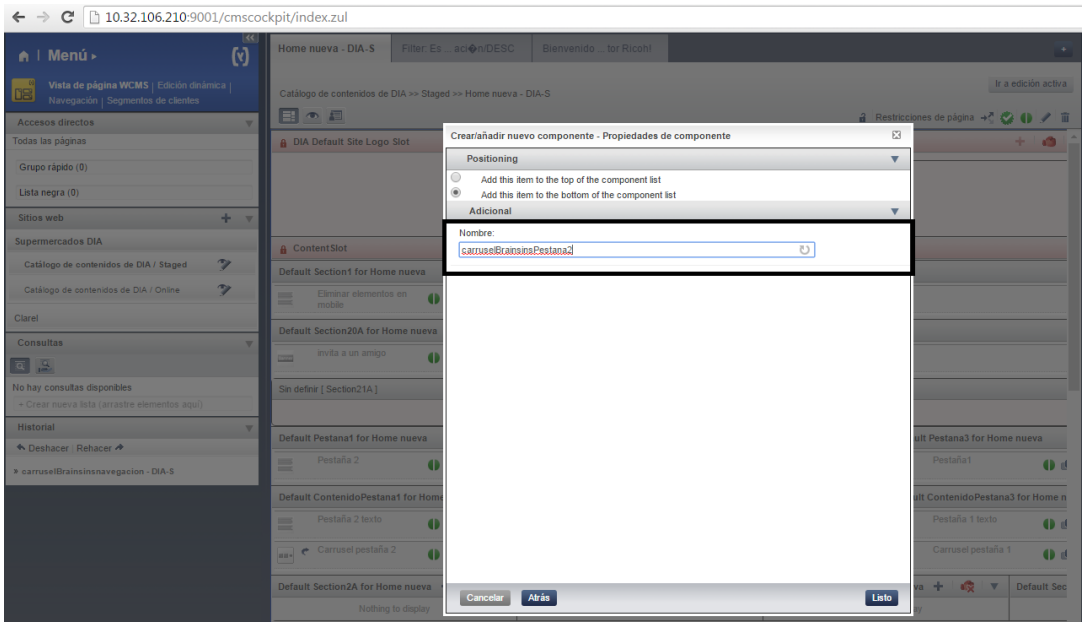


Ilustración 81: Integración CMS – Paso 7

8.3.4.1.8 PASO 8

Una vez incluido el componente, presionar el botón *Editar*.

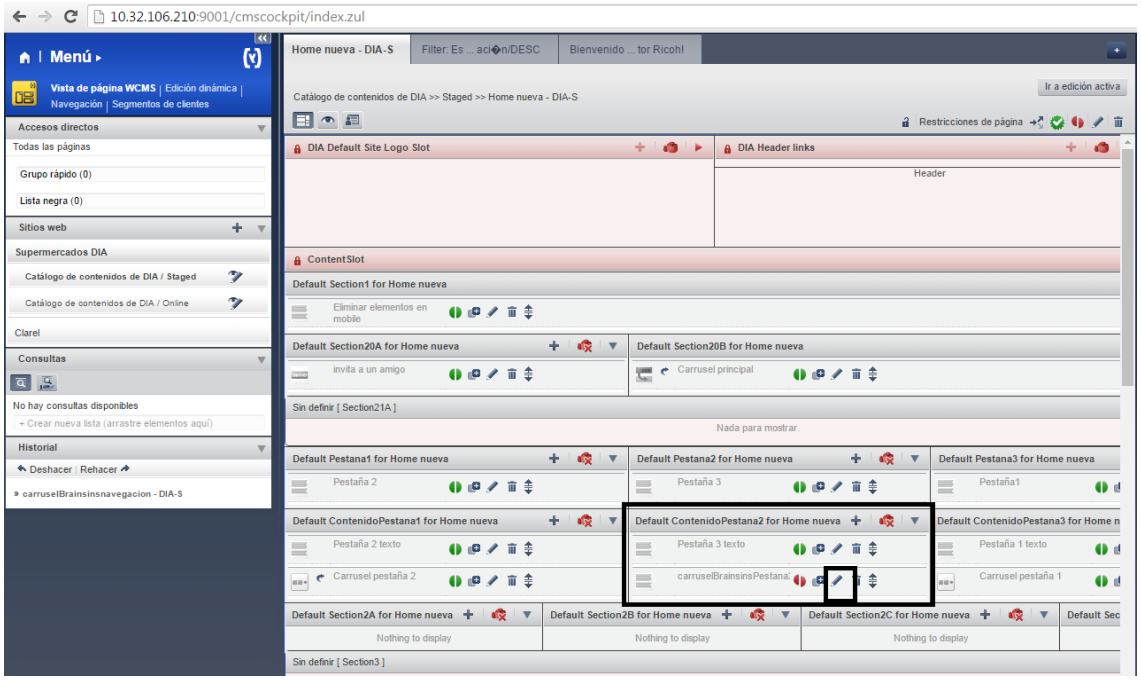


Ilustración 82: Integración CMS – Paso 8

8.3.4.1.9 PASO 9

En el apartado Contenido se incluye el siguiente componente HMTL:

```
<div id="brainsins- -recommender-homePage1" data-name="recommender-1"></div>
```

- **Id**: hace referencia al identificador del recomendador, identificador obligatorio a introducir en la página home.
- **data-name**: contiene el patrón recommender-x, siendo x es el identificador del recomendador en la herramienta de BrainSINS.

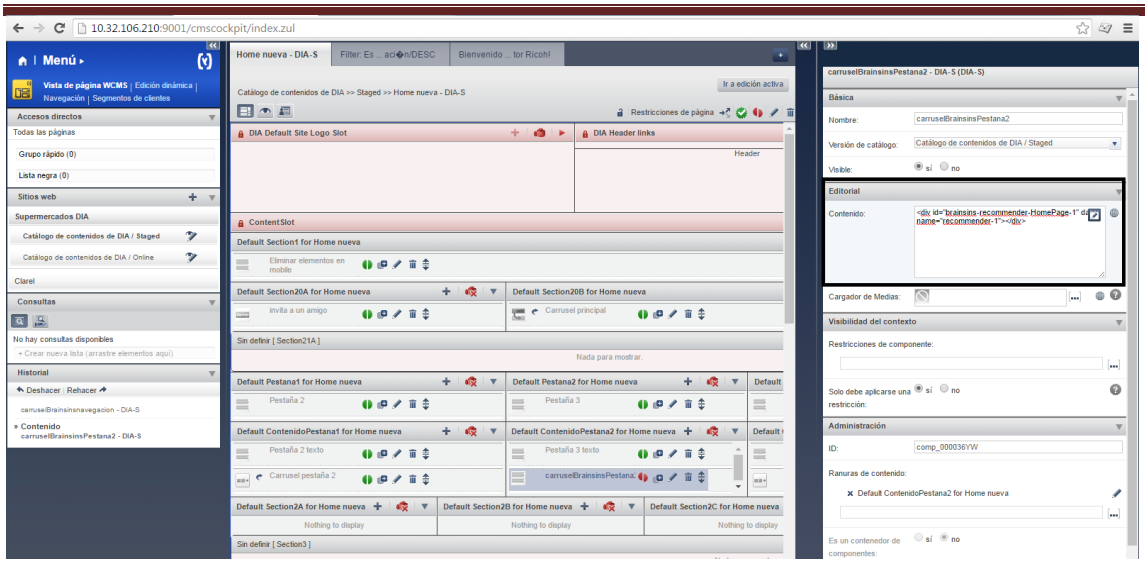


Ilustración 83: Integración CMS – Paso 9

8.3.4.1.10 PASO 10

Por último, sincronizar la plantilla haciendo clic sobre el círculo rojo del nuevo componente incluido.

8.3.4.1.11 PASO 11

Tras unos instantes, tanto el componente, como su template, habrán sido sincronizadas correctamente, de modo que el círculo pasará de color rojo a verde.

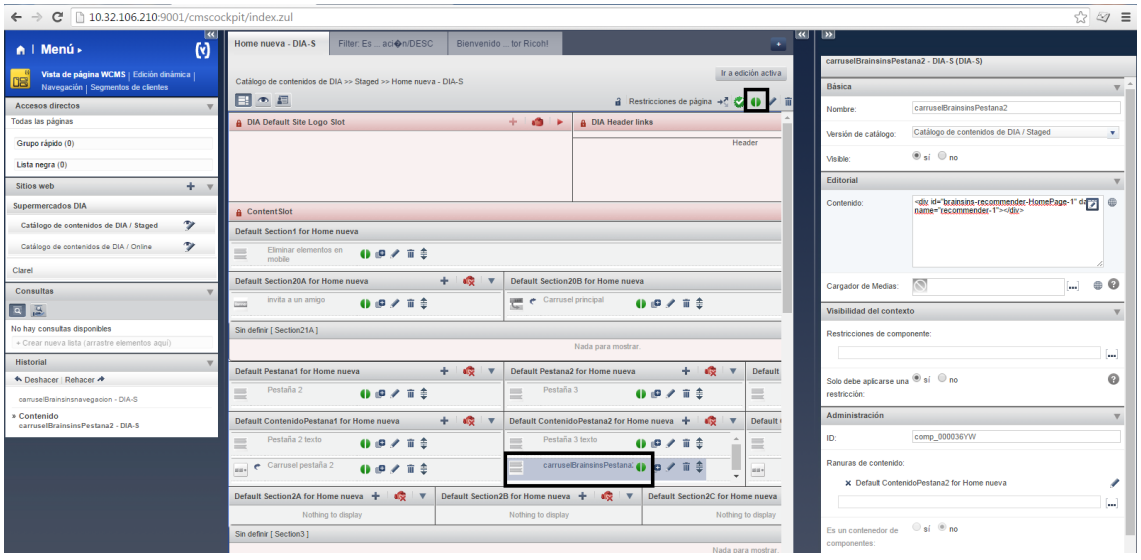


Ilustración 84: Integración CMS – Paso 11

8.3.4.1.12 RESULTADO

El recomendador de productos, aparecerá en la home del entorno de stag en la segunda pestaña, que tiene por título *Productos pensados para ti*.

Este recomendador, contendrá un conjunto de productos. Cada uno de ellos estará compuesto por su imagen, nombre, precio, puntuación media de los usuarios y la funiconalidad añadir al carrito. El precio mostrado será el de la tienda en la que se encuentra *ubicado* el usuario.

Además, haciendo clic sobre la imagen o el nombre de un producto, se accederá a la página del mismo.

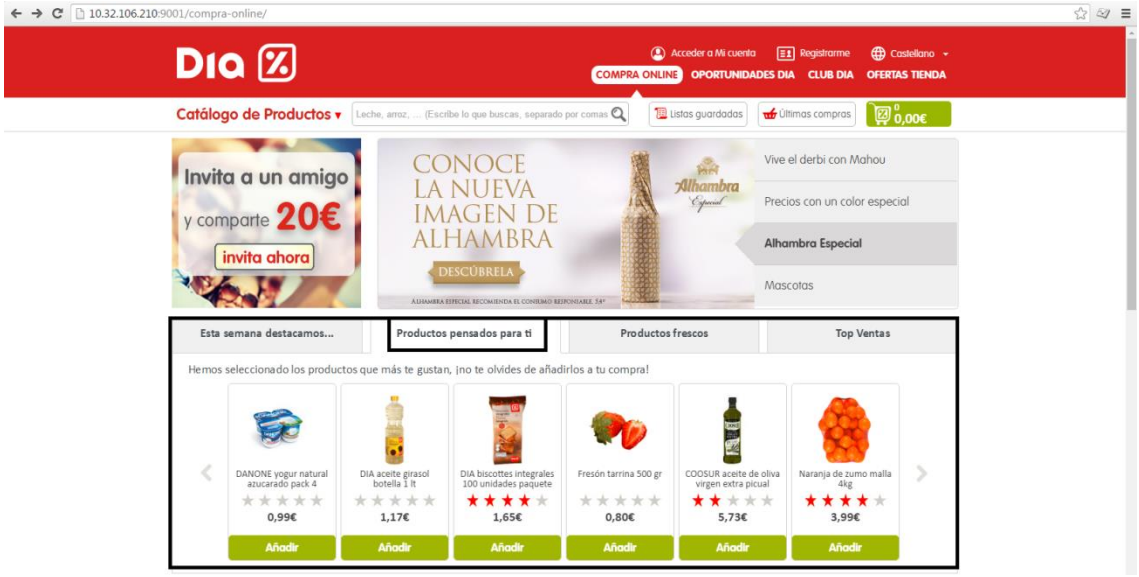


Ilustración 85: Integración CMS - Resultado

8.4 Manual del Cliente DIA

8.4.1 Introducción

Se trata del manual de utilización de los recomendadores, que serán utilizados por los clientes de DIA. Éstos, tienen el mismo aspecto y se comportan del mismo modo que los recomendadores proporcionados por Hybirs.

8.4.2 Apariencia

A continuación, se muestra la apariencia de un carrusel de productos recomenados por BrainSINS.

Ilustración 86: BrainSINS Apariencia

Por cada producto recomendado se muestra:

- Imagen del producto.
- Nombre del producto.
- Puntuación media de los clientes, mostrada por medio de estrellitas.
- Precio del producto.
- Botón “Añadir” al carrito.

El recomendador posee un botón en su extremo izquierdo y derecho, que permite la visualización de los productos que se encuentran ocultos en el carrusel.

8.4.3 Funcionalidad

8.4.3.1 Acceso a página de detalle de producto

Desde cada producto incluído en los recomendadores, se puede acceder a su página de detalle de producto, haciendo clic sobre su imagen, o bien, sobre su descripción.

8.4.4 Añadir producto

Por medio del botón ‘Añadir’, se permite añadir directamente un producto al carrito, al igual que sucede con el resto de productos en el sitio web.

8.4.5 Añadir una unidad

Una vez el producto se encuentra incluido en su carrito, haciendo clic sobre el botón ‘+’, se añadirá un unidad más del producto.

8.4.6 Eliminar una unidad

Una vez el producto se encuentra incluido en su carrito, haciendo clic sobre el botón ‘-‘, se eliminará una unidad del producto en su carrito.

8.4.7 Unidades incluidas en el carrito

Cuando uno de los productos mostrados el recomendador, se encuentra incluido en su carrito, en lugar de mostrar el botón 'Añadir' se mostrarán las unidades del producto incluidos en el carrito. Éstas unidades irán acompañadas de un botón en su extremo izquierdo y derecho, a través de los cuales se pueden añadir o bien elminar unidades del producto.

8.5 Manual del Programador

A continuación, se muestra el manual de mantenimiento del sistema BrainSINS.

8.5.1 BrainSINS

8.5.1.1 Objetivo

Dia% decide utilizar BrainSINS para mejorar la experiencia del usuario, lo que irá unido a un aumento de sus ventas. Para ello, se encarga de generar diferentes recomendadores de productos.

8.5.1.2 Descripción

Un recomendador, consiste en la muestra de un conjunto de productos que tienen una alta probabilidad de ser comprados por el cliente, una vez le son mostrados. Actualmente, BrainSINS incluye diez tipos de recomendadores asociados al tipo de página en la que serán incluidos.

8.5.1.3 Integración

La integración de esta herramienta con la funcionalidad de Dia% consta de los siguientes pasos:

1. Exportación del catálogo de productos en el formato BrainSINS Format XML
   a. Cronjob asociado a esta tarea.
2. Importación del catálogo de productos, en la herramienta de BrainSINS, por medio del fichero XML generado por el Cronjob.
3. Configuración de las plantillas de los recomendadores en la herramienta de BrainSINS.
4. Inclusión de los recomendadores en las vistas deseadas.
   a. Código javascript
   b. Actualización de las plantillas del CMS, donde incluir el recomendador.

8.5.2 Descripción de la Integración

8.5.2.1 Exportación del Catálogo

8.5.2.1.1 Introducción

Se crea una nueva extensión denominada **diabrainsins**. En esta extensión, se incluye todo el código del Cronjob que se encargará de crear la exportación del catálogo de productos.

8.5.2.1.2 Configuración

8.5.2.1.2.1 PASO 1

Es necesario incluir, en el fichero localextensions.xml del config, la nueva extensión:

```
<extension name="diabrainsins"/>
```

8.5.2.1.2.2 PASO 2

En el fichero project.properties de la extensión diabrainsins, es necesario incluir las siguientes propiedades:

```
## configuracion brainsins de SITE DIA
brainsins.save.product.xml.folder=/NFS_DATA/transfert/transfer/incoming/dia_ftp/ES/D/bra
insins/products.xml

brainsins.token.dia=BS-0452136291-1
```

La propiedad **brainsins.save.product.xml.folder** indica la localización y el nombre del fichero XML que tomará la exportación del catálogo en formato BrainSINS.

La propiedad brainsins.token.dia, hace referencia al token que nos proporciona BrainSINS para trabajar con su herramienta. Este token, es dependiente del entorno, ya que a él se asocia un catálago de productos que cambiará en función del entorno (local, dev, stag, pro).

8.5.2.1.2.3 PASO 3

Incluir en el fichero local.properties del config, la propiedad brainsins.token.dia, con el token correcto en función del entorno.

8.5.2.2 Estructura del XML

A continuación, se muestra la estructura del fichero de exportación del catálogo de productos.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><recsins>
    <entities>
        <entity name="product">
            <property name="idProduct">13964</property>
            <property name="name">DANONE yogur natural azucarado pack 4 unidades 125
g</property>
            <property
name="imageUrl">/medias/?context=bWFzdGVyfHJvb3R8MzEyOXxpbWFnZS9qcGVnfGg2Ny9oZjUvODg2Mzg
5NjQwMzk5OC5qcGd8YjdiY2YzMGRhYmEzNTU3ZDBjYjZjMGIxMGI4NzIwYTViNDlkYjNlNzFlMTc4MDM4ZGEwY2M
zYzY1MTI5ZDczYQ</property>
            <property name="url">http://stag.dia.es/compra-online/productos/lacteos-y-
huevos/yogures/yogur-natural/p/13964</property>
            <property name="categories">YOGUR NATURAL,PLAIN YOGURT,IOGURT
NATURAL</property>
            <property name="price">0.99</property>
```

```
            <property
name="misc"><![CDATA[{"unit":"pieces","averageRating":"0.00","soldByType":"QUANTITY"}]]>
</property>
            <multi_property name="multiprice">
                <property lang="10007">0.99</property>
                <property lang="10009">0.99</property>
                <property lang="468">0.99</property>
                <property lang="9851">0.99</property>
                <property lang="9727">0.99</property>
                <property lang="9832">0.99</property>
                <property lang="10139">0.99</property>
                <property lang="9740">0.99</property>
                <property lang="953">0.99</property>
                <property lang="901">0.99</property>
                <property lang="2252">0.99</property>
                <property lang="411">0.99</property>
                <property lang="10063">0.99</property>
                <property lang="10065">0.99</property>
                <property lang="10066">0.99</property>
                <property lang="952">0.99</property>
                <property lang="7528">0.99</property>
                <property lang="9489">0.99</property>
                <property lang="8256">0.99</property>
                <property lang="10134">0.99</property>
                <property lang="10131">0.99</property>
                <property lang="53113">0.99</property>
            </multi_property>
        </entity>
        …
        </entities>
</recsins>
```

8.5.2.2.1 Propiedades

El fichero está formado por las siguientes propiedades y multipropiedades:

- **idProduct**: id del producto en Hybris
- **name**: se trata de la descripción del producto que se muestra en la tienda, lo que equivaldría al nombre del producto.
- **imageUrl**: ruta absoluta de la imagen "thumbnail" del producto.
- **url**: ruta absoluta del enlace al propio producto.
- **categories**: la categoría del producto en los diferentes idiomas. La separación entre los idiomas se lleva a cabo por comas, no existiendo un orden establecido.
- **price**: precio del producto en una de las warehouse. No se ha especificado ningún criterio de selección. Éste campo solo es utilizado por Brainsins para llevar a cabo sus cálculos.
- **multiprice**: precio de los productos en las diferentes warehouse. Cada propiedad lang, incluye el identificador de la warehouse a la que se le asocia el precio del producto. Cada warehouse puede tener un precio distinto.
- **misc**: campo no obligatorio para BrainSINS. En nuestro caso, es utilizado para llevar a cabo la integración. Está formado por los siguientes campos:
    - **unit:** tipo de unidades del producto. (QUANTITTY, QUANTITY_WEIGHT, WEIGHT".
    - **averageRating:** media de las puntuaciones recibidas por los usuarios,utilizado para mostrar las estrellitas.
    - **soldByType:** forma de venta del producto. (salesUnit, pieces, gram).

8.5.2.2.2 Casuística

La propiedad **language/lang**, es utilizada para identificar el **warehouse** por medio de su id. Los productos pueden variar su precio en función del warehouse.

Es un caso especial de utilización de esta propiedad. Por lo general, se utiliza para identificar el idioma. En el caso de Dia% no es necesario, pues el nombre de los productos no está internacionalizado.

Para que Brainsins nos devuelva correctamente el precio de los productos recomendados, es necesario indicarle el código del warehouse en el que se encuentra el usuario. De esta forma, el precio de los productos recomendados será el asociado al código de warehouse en el que se encuentra en usuario.

8.5.2.3 Importación del Cronjob

En la HAC de Hybris, es necesario importar el Cronjob que se encargará de generar la exportación del catálogo. La secuencia de importación, está incluida en la extensión *diabrainsins/resources/impex/esentialdatajobs.impex*. Su contenido es el siguiente:

```
INSERT_UPDATE
ServicelayerJob;code[unique=true];springId;springIdCronJobFactory;

;defaultBrainsinsJob;defaultBrainsinsCronJob;;

INSERT_UPDATE
CronJob;code[unique=true];job(code);singleExecutable;sessionLanguage(i
socode)

;defaultBrainsinsCronJob;defaultBrainsinsJob;false;es


INSERT_UPDATE Trigger;cronjob(code)[unique=true];cronExpression

;defaultBrainsinsCronJob; 0 0 0 * * ?
```

8.5.2.4 Resultado del Cronjob

El Cronjob, genera como resultado el XML con los productos del catálogo, cuya ubicación y nombre serán los definidos en las propiedades del **project.properties** de la extensión **diabrainsins**.

Es obligatorio que el fichero XML sea público y accesible. Condición necesaria para poder llevar a cabo la carga del fichero en la herramienta de BrainSINS.

8.5.3 Importacion del Catálogo XML

8.5.3.1 Introducción

Es necesario importar el fichero XML, resultado del Cronjob, en la herramienta web de BrainSINS. De esta forma, tendrá la información necesaria para generar los recomendadores de productos.

Es condición imprescindible que el XML generado, se almacene en una carpeta pública de la web de DIA. Dicha ubicación le será indicada por medio del fichero de configuración **project.properties** de la extensión **diabrainsins**.

Para el caso del entorno de **producción**, se accederá a dicho XML por medio de la dirección:

http://www.dia.es/compra-online/brainsins/products.xml

En los entornos de prueba **DEV** y **STAG** , será necesario introducir dicho fichero xml a mano dentro del entorno de producción, ya que dev no es un entorno público (necesario utilizar VPN).

8.5.3.2 Acceso a la herramienta

La herramienta de BrainSINS, se encuentra accesible desde la página https://analytics.brainsins.com/. A ella se accederá con las credenciales asociadas al token utilizado en la exportación del catálogo. El token es el que aparece definido en el fichero **local.properties** del **config** bajo la propiedad **brainsins.token.dia**.

8.5.3.3 Importación del XML

Esta tarea, se lleva a cabo desde la pestaña **Optimización**, seleccionando en el menú izquierdo dentro del apartado **Productos**, la opción **Subir el catálogo de productos**.

8.5.3.3.1 PASO 1

Hacer clic sobre el botón **Opciones avanzadas**, que desplegará un formulario donde introducir la dirección del fichero XML. Además, en el combobox que aparece a la derecha, se permite indicar el intervalo de actualización de dicho fichero.

Ilustración 87: Importación XML – Paso 1 a

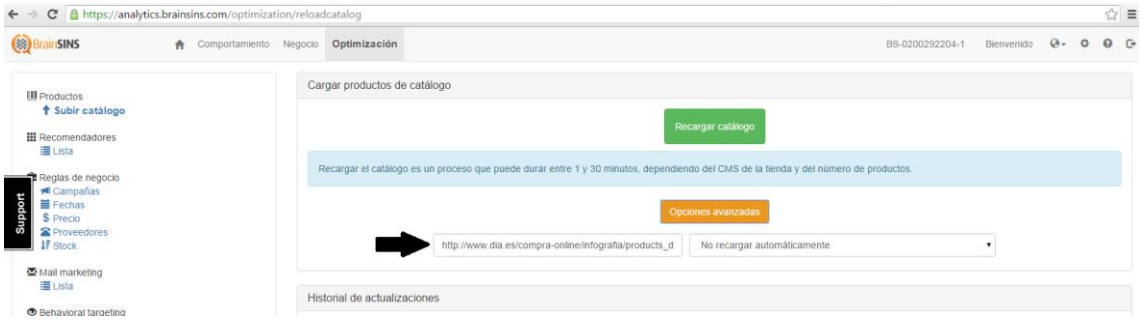


Ilustración 88: Integración XML – Paso 1 b

8.5.3.3.2 PASO 2

Pulsar el botón **Recargar catálogo**. Este proceso tardará en torno a 4 minutos.

Una vez importado, aparecerá la dirección de importación en el apartado **Historial de actualizaciones** con el resultado a **ok** en caso de haberse importado correctamente.

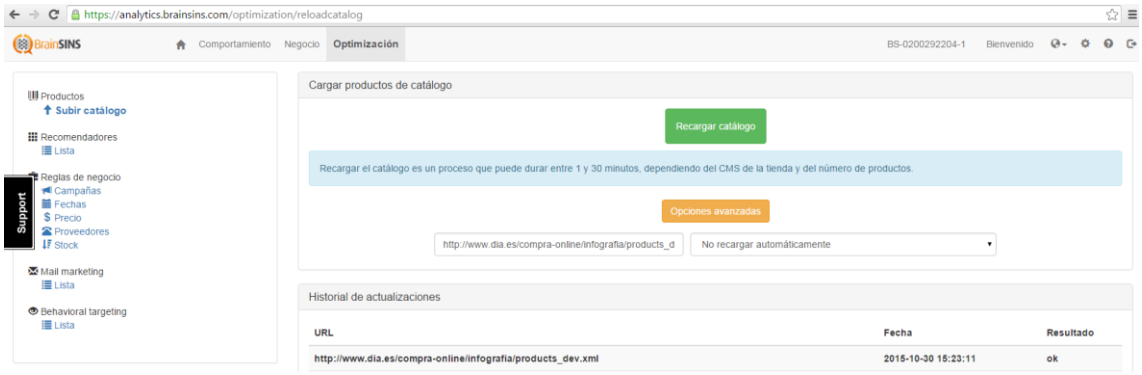


Ilustración 89: Integración XML – Paso 2

8.5.4 Vista de los Recomendadores

8.5.4.1 Recomendadores

BrainSINS, incluye por defecto una serie de recomendadores, agrupados en función del tipo de página. El listado de recomendadores, acompañados de su identificador numérico, se muestra a continuación:

Recomendadores en Inicio:

1. Basado en tu historial de navegación
2. Los más vendidos
3. Recomendaciones de nuevos productos

Recomendadores en Producto:

4. Los clientes que compraron este producto también compraron

5. Se suelen comprar juntos frecuentemente

6. Productos en categorías similares

Recomendadores en Carrito:

7. Los clientes que compraron productos que aparecen en tu cesta de la compra también compraron

8. Productos que se compran junto con el último añadido al carrito

Recomendadores en Checkout:

9 .Se suelen comprar juntos

10. Productos vistos recientemente

8.5.4.1.1 LISTADO

Se accede a ellos desde la pestaña **Optimización**, seleccionando en el menú izquierdo dentro del apartado **Recomendadores**, la opción **Lista**.

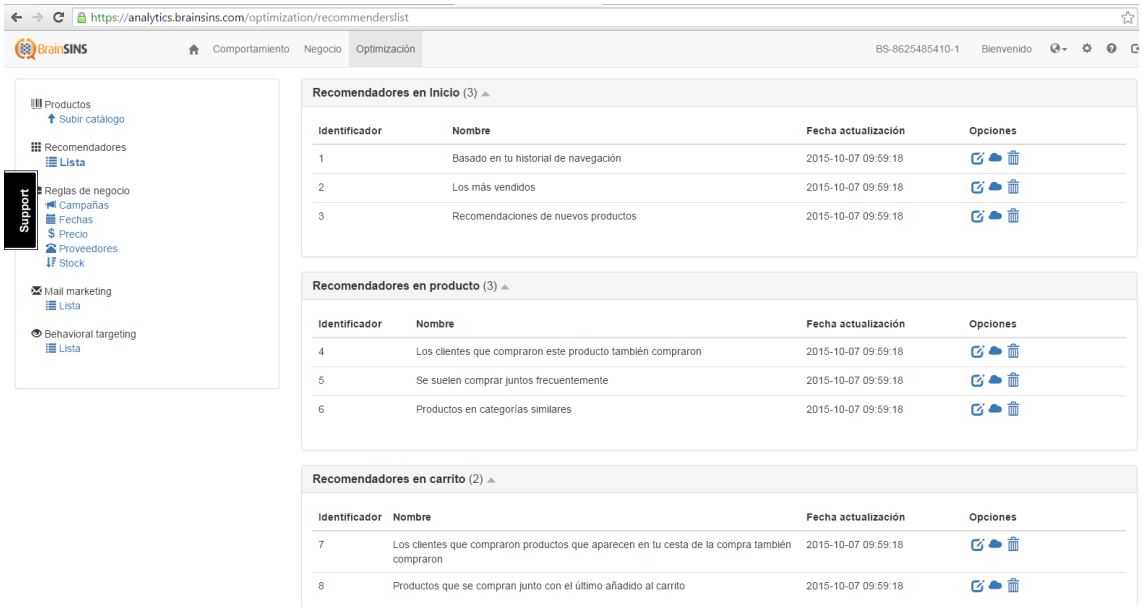


Ilustración 90: BrainSINS listado

8.5.4.1.2 ESTILO

Cada recomendador, tiene asociada una plantilla, estilo. Para acceder a ella, es necesario hacer clic sobre el icono izquierdo de las **Opciones**.

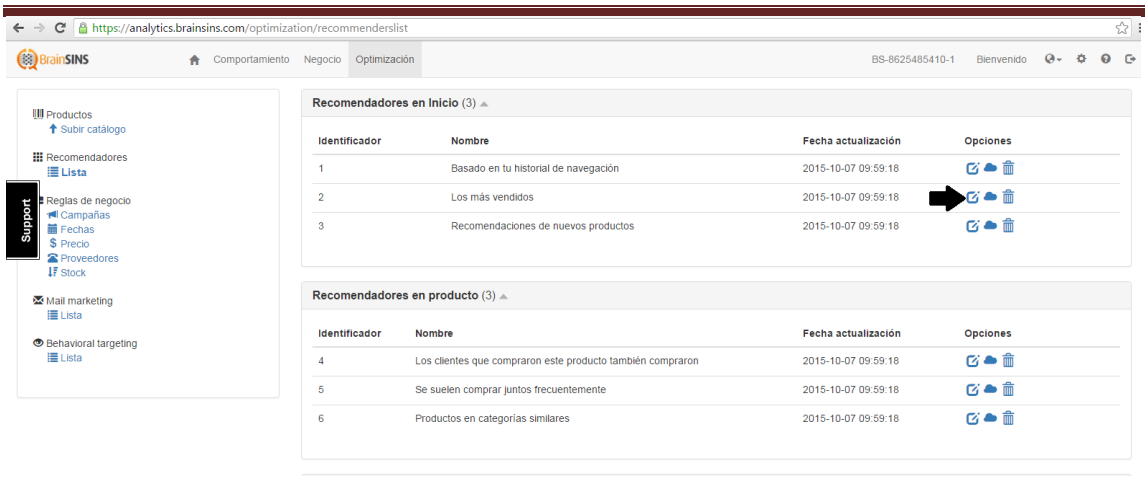


Ilustración 91: BrainSINS Estilo

8.5.4.1.2.1 Editor Simple

La opción anterior, nos dirige al editor simple de la plantilla. Desde BrainSINS se recomienda utilizar el Editor Avanzado.

8.5.4.1.2.2 Editor Avanzado

HTML Y CSS

El editor avanzado permite cambiar el propio HTML y CSS del recomendador.

PRECIO Y LENGUAJE

En la parte inferior de este editor, se permite configurar el formato del precio a mostrar. Además, como en el caso de DIA%, se utiliza la propiedad **language** para indicar el código del **warehouse**, es necesario incluir en cada plantilla, todos los **warehouse** dados de alta en el entorno con el formato del precio adecuado. Sin esto, brainsins no podrá mostrar ningún precio en los productos recomendados.

8.5.5 Inclusión del Recomendador vía CMS

Véase Manual de Gestor DIA.

8.5.6 Integración Front

Para la integración de BrainSINS, en el front, se ha creado el servicio "BrainsinsStoreFrontService", que se encarga de crear un objeto "AuxBrainsinsData" encargado de almacenar aquella información necesaria de enviar a BrainSINS.

Este servicio es incluido en los controladores que se encargan de mostrar las páginas, donde es necesario enviar a BrainSINS la información, o bien, donde se desea incluir recomenadores de BrainSINS.

Además, el javascript “acc.brainsins.js” se encarga de enviar a BrainSINS toda la información, así como de completar los estilos del formulario para mostrar correctamente la funcionalidad de añadir al carrito, mostrar las puntuaciones de los ususarios.

Capítulo 9. Conclusiones y Ampliaciones

9.1 Conclusiones

Como puede observarse en el capítulo 1, la finalidad de este proyecto es la de desarrollar la integración de la plataforma de comercio electrónico utilizada en el sitio web de DIA, Hybris, con la solución de Marketing Automation BrainSINS.

El desarrollo de esta integración se divide en dos fases. La primera, consiste en el desarrollo de un cronjob que se encargue de generar un fichero de exportación XML compuesto por los productos que DIA oferta en sus tiendas. Gracias a este fichero, BrainSINS podrá recomendar productos en base a ciertos parámetros. La segunda fase, consiste en integrar la comunicación con BrainSINS. A través de ella, se enviará a BrainSINS la información que necesite en base a la página en la que se encuentre el usuario, así como se recibirán los productos recomendados.

Se ha comenzado por el desarrollo de un cronjob en el sistema Hybris. Éste se encarga de consultar el catálogo de productos de la versión online de DIA, que están en el estado aprobado, que se encuentran a la venta al menos en una de las tiendas de DIA y que además posean stock. A partir del listado de los productos, el cronjob se encarga de generar un fichero de exportación XML que seguirá la especificación de BrainSINS. Ésta exportación será llevada a cabo todos los días, una vez el sistema Hybris ha recibido las actualizaciones de su catálogo. De este modo, BrainSINS ofrecerá los productos que realmente están disponibles para su venta.

Por último, se ha llevado a cabo la integración con el front. Para ello, entre otros, se ha desarrollado un servicio, encargado de incluir la información que BrainSINS necesita tanto para recabar información, y así ofrecer los mejores productos recomendados, como para mostrar dichos productos en las páginas del sitio web.

El desarrollo de ésta integración, ha sido un camino complejo, ya que como se indica en el apartado “Problemas encontrados”, han sido varios los problemas que han ido surgiendo.

Uno de los problemas más complicados de solucionar, ha sido el de incluir en cada uno de los productos recomendados, la funcionalidad *Añadir al carrito*, funcionalidad que BrainSINS no ofrecía. Fue necesario conocer todo el proceso que había detrás de esta funcionalidad, para así poder replicarlo para BrainSINS. Además, dado que la repuesta de BrainSINS es un HTML plano, fue necesario idear una estrategia, que se encargase de incluir la funcionalidad una vez se había recibido la repuesta de BrainSINS.

BrainSINS tampoco daba soporte a que un mismo producto pudiera tener diferentes precios. Sin embargo, esto era esencial para DIA, ya que un mismo producto podía ser ofertado en varias tiendas, con distinto precio. Para conseguir que BrainSINS recomendase los productos en función de la tienda en la que se encontraba situado el usuario, fue necesario, utilizar una multipropiedad compuesta por la etiqueta “lang” donde se indicaba el código de la tienda en

la que se ofertaba, etiqueta que iba acompañada por el precio del producto para dicha tienda. En todas las páginas donde se incluye BrainSINS es indispensable enviarle a BrainSINS el código de la tienda en la que se encuentra el usuario, de éste modo, sólo recomendará los productos que se ofertan en dicha tienda con el precio correcto.

Otra complejidad, fue la de conocer el funcionamiento de la herramienta CMS, así como de conocer su estructuración. Los recomendadores debían de ser incluidos en las diferentes páginas del sitio web, a través de ésta herramienta, de modo que el personal de DIA, pudiera incluirlos o eliminarlos de forma autónoma y sencilla.

En conclusión, se ha dado solución a todos los problemas encontrados. Además, la integración se ha automatizado correctamente. Por un lado, el cronjob es ejecutado de forma autónoma todos los días a la hora indicada en su trigger. Su resultado se almacena en una carpeta compartida por ambos nodos, a la que a su vez apunta un enlace simbólico creado en Linux desde la carpeta *brainsins*, carpeta pública en el front de DIA. Además, el proceso de actualización del fichero de exportación en el sistema BrainSINS, también se ha automatizado,. Todos los días a la hora acordada con el equipo de BrainSINS, descargan el fichero de exportación de la URL pública, la cual apunta a la carpeta compartida por los nodos.

9.2 Ampliaciones

En cuanto a las ampliaciones que se plantean al término de este proyecto, destacan:

- Mejora de la accesiblidad tanto de los productos recomendados por BrainSINS, como del resto de los productos mostrados en el sitio web.
- Integrar BrainSINS con el sitio web de CLAREL.

Capítulo 10. Referencias Bibliográficas

10.1 Referencias en Internet

[ONTSI10] Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información. "Estudio sobre Comercio Electrónico B2C".

http://www.ontsi.red.es/ontsi/es/estudios-informes/estudio-b2c-2010

[ONTSI11] Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información. "Informe anual 2010 (Edición 2011)".

http://www.ontsi.red.es/ontsi/es/estudios-informes/informe-anual-2010-edicion-2011

[ONTSI12] Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información. "Estudio B2C 2012 (Edición 2013)".

http://www.ontsi.red.es/ontsi/es/estudios-informes/estudio-b2c-2012-edici%C3%B3n-2013

[ONTSI13] Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información. "Informe anual 2013".

http://www.ontsi.red.es/ontsi/sites/default/files/informe_anual_la_sociedad_en_red_2013_ed._2014.pdf

[ONTSI14] Observatorio Nacional de las Telecomunicaciones y de la Sociedad de la Información. "Informe anual 2014".

http://www.ontsi.red.es/ontsi/es/estudios-informes/estudio-b2c-2014-edici%C3%B3n-2015

[ADIGITALBF15] Asociación Española de la Economía Digital."Informe eCommerce Black Friday 2015".

https://www.adigital.org/informes-estudios/informe-ecommerce-black-friday-2015/

[Pablo15] Ecommerce News. "PrestaShop alcanza las 40.000 tiendas online en España y se consolida como motor de comercio electrónico".

http://ecommerce-news.es/actualidad/prestashop-alcanza-las-40-000-tiendas-online-en-espana-y-se-consolida-como-el-motor-del-comercio-electronico-33462.html

[WooCommercePlugin] Woo Commerce Plugins.

http://www.woothemes.com/product-category/woocommerce-extensions/free-extensions/

[Hassan08] Hassan Montero, Y. "Guía de Evaluación Heurística de Sitios Web". http://www.nosolousabilidad.com/articulos/heuristica.htm

[Hybris15] Página oficial de Hybris.

http://www.hybris.com/es/commerce

[Infosys15] Informe sobre Infosys.

https://www.infosys.com/industries/high-technology/case-studies/Documents/semiconductor-technology-ecommerce.pdf

[Mañez_15] Adrián Máñez . "Qué es el Marketing Automation y cuáles son sus beneficios". http://blog.hubspot.es/marketing/que-es-el-marketing-automation-y-para-que-sirve

[Diaz_15] Melina Diaz. "Marketing Automation, qué es y cómo lograrlo". http://www.makingexperience.com/blog/marketing-automation-que-es

[Hubspot_15] Hubspot. "What is Marketing Automation".

http://www.hubspot.com/marketing-automation-information

[Rijn_15] Jordie Rijn – EmailMonday .

http://www.emailmonday.com/dma-national-client-email-report-2015

[Brainsins_slide_15] "Tendencias eCommerce2015 según 90 profesionales del eCommerce". http://es.slideshare.net/brainsins/tendencias-ecommerce2015-segn-90-profesionales-del-ecommerce

Soluciones de Comercio Electrónico

http://ecommerce-platforms.com/articles/7-best-free-wordpress-ecommerce-plug-ins

http://www.webbuildersguide.com/website-builder-articles/10-best-cms-for-ecommerce-websites/

http://www.businessnewsdaily.com/7707-best-ecommerce-software.html

http://www.sitepoint.com/10-best-self-hosted-ecommerce-solutions/

Estudio de las más populares

http://blog.aheadworks.com/2015/10/ecommerce-platforms-popularity-october-2015-top-five-solutions-take-three-quarters-of-the-market/

E-Commerce España

http://observatorioecommerce.com/

Extensiones de magento

http://www.sugerendo.com/blog/magento/las-mejores-extensiones-para-magento-ii-marketing/

http://ecommerce-news.es/actualidad/prestashop-alcanza-las-40-000-tiendas-online-en-espana-y-se-consolida-como-el-motor-del-comercio-electronico-33462.html

Comparación con Hybris

https://www.nbs-system.com/wp-content/uploads/Global%20rating%20table.html

http://es.slideshare.net/TeroJuola/oracle-atg-ibm-web-sphere-commerce-sap-hybris-magento-comparison

Tipos de comercio electrónico

https://es.shopify.com/blog/12621205-los-5-tipos-de-comercio-electronico

Capítulo 11. Apéndices

11.1 Glosario y Diccionario de Datos

- **Black Friday**: Día que inaugura la temporada de compras navideñas con significativas rebajas en muchas tiendas minoristas y grandes almacenes. Es un día después del Día de Acción de Gracias en Estados Unidos, y se celebra el día siguiente al cuarto jueves del mes de noviembre.2 Esta festividad comenzó en Estados Unidos, y poco a poco y con la ayuda de las nuevas tecnologías y la promoción de este día por parte de las distintas empresas se ha ido extendiendo por el resto de países del mundo. (Wikipedia).
- **B2B**: Se trata de un tipo de comercio electrónico. Es la abreviación de business to business (negocio a negocio), y es aquel en donde la transacción comercial únicamente se realiza entre empresas que operan en Internet, lo que quiere decir que no intervienen consumidores.
- **B2C**: Este es el tipo de comercio electrónico, también conocido como business to consumer (negocio a consumidor), es el más conocido y el que seguramente tú empleas. Es aquel que se lleva a cabo entre el negocio o, en este caso tienda virtual, y una persona interesada en comprar un producto o adquirir un servicio.
- **CiberMonday**: Término dedicado a describir el lunes siguiente del Día de Acción de Gracias en los Estados Unidos (cuarto jueves del mes de noviembre), y que se realiza tras el «viernes negro» (Black Friday, día siguiente de Acción de Gracias), creado por las empresas para persuadir a la gente a comprar por internet.
- **Cloud Computing**: El cloud computing consiste en la posibilidad de ofrecer servicios a través de Internet. La computación en nube es una tecnología nueva que busca tener todos nuestros archivos e información en Internet y sin depender de poseer la capacidad suficiente para almacenar información.
- **CMS**: Un CMS es un programa desarrollado para que cualquier usuario pueda administrar y gestionar contenidos de una web con facilidad y sin conocimientos de programación Web.
- **Comercio electrónico**: Consiste principalmente en intercambiar información comercial, ya sean productos o servicios, siempre en la red.
- **Cross-selling**: Llamamos cross selling o venta cruzada, a la estrategia de venta para vender productos accesorios o complementarios a los clientes.
- **Gobierno abierto**: Es una doctrina política caracterizada por la adopción de la filosofía del movimiento del software libre a los principios de la democracia. El Gobierno Abierto tiene como objetivo que los ciudadanos colaboren en la creación y el mejoramiento de servicios públicos y en el robustecimiento de la transparencia y la rendición de cuentas
- **Lead Nurturing**: Tiene como objetivo "madurar" a los leads. Lo hace a través de cadenas de Email automáticas en las que les ofrece, de forma paulatina, contenidos que les hacen avanzar en el ciclo de compra.

- **Leads**: Un lead es una persona que ha facilitado sus datos de contacto a través del formulario de una landing page y que pasa a formar parte de una base de datos.
- **Marketing Management**: Automatización de un sistema para gestionar los recursos, tanto humanos como económicos.
- **Marketing Automation**: Se trata de una metodología que basa su funcionamiento en la utilización de software para automatizar todos los procesos derivados de una estrategia de marketing digital, como la segmentación, la generación de workflows, gestión de campañas, lead nuturing...
- **Marketplaces horizontales**: En el marketing horizontal te diriges a un tipo de cliente concreto pero que puede pertenecer a sectores muy distintos. Se trata de productos o servicios que tienen un mercado amplio, como por ejemplo los artículos destinados a ejecutivos, productos de gran consumo o seguros. El único requisito es que no estén pensados para una industria específica.
- **Marketplaces verticales**: El marketing vertical es aquel en el que todos tus esfuerzos y acciones se centran en uno o pocos sectores de actividad. Esto puede suceder porque tu producto o servicio es muy especializado, o porque piensas que en dichos mercados obtendrás mayores ingresos o una cuota de ventas más amplia. Es el caso de las empresas que suministran tecnología para hospitales, software para escuelas o maquinaria para compañías energéticas, por ejemplo.
- **Omnicanalidad**: La omnicanalidad es la integración de todos los canales existentes en el mercado, de manera tal de generar caminos que se interrelacionen para que un cliente que inició una comunicación por una vía de interacción pueda continuarla por otra.
- **Prospecto**: Es aquel consumidor o empresa que tiene un interés en comprar su producto o servicio. Este consumidor o empresa puede ser ya un cliente de su empresa o no. Y el interés puede ser tanto para comprar un nuevo producto, comprar más del mismo producto que tiene, o simplemente comprar otros productos.
- **Retailers**: Se compone de las actividades relacionadas con la venta de productos directamente a los consumidores a través de canales tales como tiendas, centros comerciales, quioscos, máquinas expendedoras u otros lugares fijos, de acuerdo con el Diccionario Libre. En contraste, la comercialización directa al consumidor intenta completar una venta a través del teléfono, correo o página web de ventas.
- **Social Media Marketing**: tipo de marketing basado en la recomendación de tus amigos a través de redes sociales
- **Up-selling**: es una técnica de venta por la cual un vendedor induce al cliente a comprar productos más caros, actualizaciones u otros add-ons en un intento de hacer una venta más rentable.

11.2 Contenido Entregado

El contenido entregado se compone de:

- La presente documentación.
- El código fuente del proyecto. Donde por un lado se encuentran las propias extensiones de Hybris y dentro de la custom, las extensiones desarrolladas por el equipo DIA. Dentro de la extensiones incluidas en la carpeta custom, se incluye la extensión diabrainsins, donde se incluye la funcionalidad del cronjob, así como las distintas extensiones donde se añadido funcionalidad, como es el caso de la extensión diaacceleratorstorefront, donde entre otros, se ha incluido el servicio de integración de Brainsins.

11.3 Código Fuente

11.3.1 Extensión diabrainsins

11.3.1.1 BrainsinsCronjob.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.jobs;

import de.hybris.platform.cronjob.enums.CronJobResult;
import de.hybris.platform.cronjob.enums.CronJobStatus;
import de.hybris.platform.cronjob.model.CronJobModel;
import de.hybris.platform.servicelayer.config.ConfigurationService;
import de.hybris.platform.servicelayer.cronjob.AbstractJobPerformable;
import de.hybris.platform.servicelayer.cronjob.PerformResult;

import java.io.File;
import java.util.List;

import org.apache.commons.io.FileUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;

import com.b2b2000.dia.brainsins.service.BrainsinsService;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.facades.product.impl.ExtendedProductFacade;


/**
 * Cronjob encargado de llevar a cabo la exportacion de los productos del catalogo
online de DIA a un fichero XML que
 * sigue la especificación BrainSINS.
 *
 * @author <a href="mailto:estefania.menendez.ricoh@gmail.com>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class BrainsinsCronJob extends AbstractJobPerformable<CronJobModel>
{

	/**
	 * Log to show messages
	 */
	private static final Logger LOG = Logger.getLogger(BrainsinsCronJob.class);

	/**
	 * Location to save the xml with Brainsins products.
	 */
	private static final String XML_EXPORT_LOCATION =
"brainsins.save.product.xml.folder";

	/**
	 * Service to generate xml file with products.
	 */
	private BrainsinsService brainsisService;

	/**
	 * Service to gets the location to save the xml with catalog products
	 */
	private ConfigurationService configurationService;

	/**
	 * Product Facade
	 */
	private ExtendedProductFacade productFacade;
```

```
	@Override
	public PerformResult perform(final CronJobModel cronJob)
	{
		LOG.info(CronJobModel.CODE + CronJobModel.STATUS);

		try
		{
			final String folderFile =
this.configurationService.getConfiguration().getString(XML_EXPORT_LOCATION);

			if (StringUtils.isBlank(folderFile))
			{
				LOG.error(CronJobModel.CODE + "Property " +
XML_EXPORT_LOCATION + " not configured");
				return new PerformResult(CronJobResult.ERROR,
CronJobStatus.ABORTED);
			}

			final List<DiaProductModel> products =
getProductFacade().findProductsToBrainsins();

			final String xml =
getBrainsisService().convertProductsToXml(products);

			final File file = new File(folderFile);

			FileUtils.writeStringToFile(file, xml);

			LOG.info(CronJobModel.CODE + " result file: " + folderFile);

		}
		catch (final Throwable e)
		{
			LOG.error(CronJobModel.CODE + " produces an error : " +
e.getMessage(), e);
			return new PerformResult(CronJobResult.ERROR,
CronJobStatus.ABORTED);
		}

		LOG.info(CronJobModel.CODE + " has been SUCCESSFULLY executed ");

		return new PerformResult(CronJobResult.SUCCESS, CronJobStatus.FINISHED);
	}


	/**
	 * @return the brainsisService
	 */
	public BrainsinsService getBrainsisService()
	{
		return brainsisService;
	}



	/**
	 * @param brainsisService
	 *           the brainsisService to set
	 */
	public void setBrainsisService(final BrainsinsService brainsisService)
	{
		this.brainsisService = brainsisService;
	}



	/**
	 * @return the configurationService
	 */
	public ConfigurationService getConfigurationService()
	{
		return configurationService;
	}
```

```
	/**
	 * @param configurationService
	 *           the configurationService to set
	 */
	public void setConfigurationService(ConfigurationService configurationService)
	{
		this.configurationService = configurationService;
	}


	/**
	 * @return the productFacade
	 */
	public ExtendedProductFacade getProductFacade()
	{
		return productFacade;
	}


	/**
	 * @param productFacade
	 *           the productFacade to set
	 */
	public void setProductFacade(ExtendedProductFacade productFacade)
	{
		this.productFacade = productFacade;
	}
}
```

11.3.1.2 BraisinsService.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.service;

import java.util.List;

import com.b2b2000.dia.core.model.DiaProductModel;


/**
 * Service integration with aplication <em>BrainSINS</em> for DIA.
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es (EMF)</a>
 *
 */
public interface BrainsinsService
{

	/**
	 * Method to gets a String with content of a XML file from {@code
List<ProductModel>} recieved as a parameter.
	 *
	 * @param products
	 *           list of products for BrainSINS.
	 * @return the string of a xml file.
	 */
	String convertProductsToXml(final List<DiaProductModel> products);

}
```

11.3.1.3 DefaultBrainsinsService.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.service.impl;
```

```
import de.hybris.platform.servicelayer.dto.converter.Converter;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import com.b2b2000.dia.brainsins.model.Recsins;
import com.b2b2000.dia.brainsins.service.BrainsinsService;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.integrations.dataexport.generic.output.xml.ConvertObjectsToXML;


/**
 * Default implementation of {@link BraininsService}.
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Mennedez Freije
(EMF)</a>
 *
 */
public class DefaultBrainsinsService implements BrainsinsService
{

	/**
	 * Hybris class. Convert Recsins Object to XML
	 *
	 */
	@SuppressWarnings("deprecation")
	private ConvertObjectsToXML convertObjectsToXML;


	/**
	 * Converts a collection of DiaProductModel objects to a Recsins object
	 */
	private Converter<Collection<DiaProductModel>, Recsins> recsinsConverter;



	/**
	 * @return the convertObjectsToXML
	 */
	@SuppressWarnings("deprecation")
	public ConvertObjectsToXML getConvertObjectsToXML()
	{
		return convertObjectsToXML;
	}

	/**
	 * @param convertObjectsToXML
	 *           the convertObjectsToXML to set
	 */
	@SuppressWarnings("deprecation")
	public void setConvertObjectsToXML(final ConvertObjectsToXML convertObjectsToXML)
	{
		this.convertObjectsToXML = convertObjectsToXML;
	}

	/**
	 * @return the recsinsConverter
	 */
	public Converter<Collection<DiaProductModel>, Recsins> getRecsinsConverter()
	{
		return recsinsConverter;
	}

	/**
	 * @param recsinsConverter
	 *           the recsinsConverter to set
	 */
	public void setRecsinsConverter(final Converter<Collection<DiaProductModel>,
Recsins> recsinsConverter)
	{
		this.recsinsConverter = recsinsConverter;
	}
```

```
	/**
	 * This method converts a list of DiaProductModel to a String. This String
contains the XML that follows BrainSINS
	 * specification.
	 *
	 * @param products
	 *           The list of products to convert
	 */
	@SuppressWarnings("deprecation")
	@Override
	public String convertProductsToXml(final List<DiaProductModel> products)
	{
		final Recsins recsins = getRecsinsConverter().convert(products);
		final List<Object> sourceDataList = new ArrayList<Object>();

		sourceDataList.add(recsins);

		return normalizeString(getConvertObjectsToXML().convert(sourceDataList));
	}

	/**
	 * Method to normalize {@code output} received as parameter.
	 *
	 * @param output
	 *           text to normalize
	 * @return text normalized;
	 *
	 * @see <a
href="https://jaxb.java.net/faq/#marshalling_cdata">https://jaxb.java.net/faq/#marshalli
ng_cdata</a>
	 *
	 */
	private String normalizeString(String output)
	{
		// HACK: Tenemos un problema con el XML y la sección CDATA. JAXB no lo
soporta directamete.
		// Ver https://jaxb.java.net/faq/#marshalling_cdata y la solución que dan
se basa en meter CDATA a nivel de elemento,
		// pero BrainSINS solo lo espera a nivel de elemento (property), pero con
nombre (name = "misc")
		output = output.replaceAll("<property name=\"misc\">&lt;", "<property
name=\"misc\"><");
		output = output.replaceAll("&gt;</property>", "></property>");

		return output;
	}


}
```

11.3.1.4 RecsinsConverter.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.converters;

import de.hybris.platform.converters.impl.AbstractConverter;
import de.hybris.platform.servicelayer.dto.converter.Converter;

import java.util.Collection;
import java.util.List;

import com.b2b2000.dia.brainsins.model.Entities;
import com.b2b2000.dia.brainsins.model.Recsins;
import com.b2b2000.dia.core.model.DiaProductModel;


/**
 * Se encarga de convertir una colección de productos DiaProductModel a un objeto
Recsins
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
```

```
 *
 */
public class RecsinsConverter extends AbstractConverter<List<DiaProductModel>, Recsins>
{

	/**
	 * Converts a collection of DiaProductModel to Entities
	 */
	private Converter<Collection<DiaProductModel>, Entities> entitiesConverter;

	/**
	 * @return the entitiesConverter
	 */
	public Converter<Collection<DiaProductModel>, Entities> getEntitiesConverter()
	{
		return entitiesConverter;
	}


	/**
	 * @param entitiesConverter
	 *           the entitiesConverter to set
	 */
	public void setEntitiesConverter(final Converter<Collection<DiaProductModel>,
Entities> entitiesConverter)
	{
		this.entitiesConverter = entitiesConverter;
	}



	/**
	 * This method populate the conversion of a list of DiaProductModel to Recsins
object
	 *
	 * @param source
	 *           The list of DiaProductModel to convert
	 * @param target
	 *           Recsins object
	 *
	 */
	@Override
	public void populate(final List<DiaProductModel> source, final Recsins target)
	{

		if (source != null)
		{
			target.setEntities(this.getEntitiesConverter().convert(source));
		}
	}

}
```

11.3.1.5 EntitiesConverter.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.converters;

import de.hybris.platform.converters.Converters;
import de.hybris.platform.converters.impl.AbstractConverter;
import de.hybris.platform.servicelayer.dto.converter.Converter;

import java.util.List;

import com.b2b2000.dia.brainsins.model.Entities;
import com.b2b2000.dia.brainsins.model.Entity;
import com.b2b2000.dia.core.model.DiaProductModel;


/**
 * Se encarga de convertir un listado de objetos DiaProductModel a un objeto Entities
 *
```

```
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class EntitiesConverter extends AbstractConverter<List<DiaProductModel>,
Entities>
{

	private Converter<DiaProductModel, Entity> entityConverter;



	/**
	 * @return the entityConverter
	 */
	public Converter<DiaProductModel, Entity> getEntityConverter()
	{
		return entityConverter;
	}



	/**
	 * @param entityConverter
	 *            the entityConverter to set
	 */
	public void setEntityConverter(final Converter<DiaProductModel, Entity>
entityConverter)
	{
		this.entityConverter = entityConverter;
	}



	@Override
	public void populate(final List<DiaProductModel> source, final Entities target)
	{

		if (source != null)
		{
			target.getEntity().addAll(Converters.convertAll(source,
getEntityConverter()));
		}
	}

}
```

11.3.1.6 EntityConverter.java

```
/**
 *
 */
package com.b2b2000.dia.brainsins.converters;

import de.hybris.platform.catalog.model.classification.ClassificationClassModel;
import de.hybris.platform.category.model.CategoryModel;
import de.hybris.platform.commercefacades.product.PriceDataFactory;
import de.hybris.platform.commercefacades.product.data.PriceData;
import de.hybris.platform.commercefacades.product.data.PriceDataType;
import de.hybris.platform.commerceservices.url.UrlResolver;
import de.hybris.platform.converters.impl.AbstractConverter;
import de.hybris.platform.core.model.c2l.LanguageModel;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.customerreview.enums.CustomerReviewApprovalType;
import de.hybris.platform.customerreview.model.CustomerReviewModel;
import de.hybris.platform.europe1.model.PriceRowModel;
import de.hybris.platform.servicelayer.config.ConfigurationService;
import de.hybris.platform.servicelayer.i18n.CommonI18NService;

import java.math.BigDecimal;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Collection;
import java.util.HashMap;
```

```
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.Map.Entry;

import net.sourceforge.pmd.util.StringUtil;

import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;

import com.b2b2000.dia.brainsins.enums.BrainsinsPropertyNameEnum;
import com.b2b2000.dia.brainsins.model.Entity;
import com.b2b2000.dia.brainsins.model.MultiProperty;
import com.b2b2000.dia.brainsins.model.NestedProperty;
import com.b2b2000.dia.brainsins.model.Property;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.enums.SoldByType;


/**
 * Se encarga de convertir un objeto DiaProductModel a un objeto Entity
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class EntityConverter extends AbstractConverter<DiaProductModel, Entity>
{

	/**
	 * Log to show messages.
	 */
	private static final Logger LOG = Logger.getLogger(EntityConverter.class);

	/**
	 * Content of start section misc.
	 */
	private static final String SECTION_MISC_START = "<![CDATA[{";

	/**
	 * Content of end section misc.
	 */
	private static final String SECTION_MISC_END = "}]]>";

	/**
	 * (EMF) 20152909
	 *
	 * PriceRow's usergroup preffix that holds the shop id related with a price
	 *
	 */
	private static final String GROUP_PREFIX = "usergroup";


	/**
	 * Url resolver for products.
	 */
	private UrlResolver<ProductModel> productModelUrlResolver;

	/**
	 * Price data factory to create
	 */
	private PriceDataFactory priceDataFactory;

	/**
	 * Servicio i18n
	 */
	private CommonI18NService commonI18NService;

	/**
	 * (EMF) 20151002
	 *
	 * Service to gets site url
	 */
	private ConfigurationService configurationService;


	/**
```

```
	 * @return the productModelUrlResolver
	 */
	public UrlResolver<ProductModel> getProductModelUrlResolver()
	{
		return productModelUrlResolver;
	}

	/**
	 * @param productModelUrlResolver
	 *           the productModelUrlResolver to set
	 */
	public void setProductModelUrlResolver(final UrlResolver<ProductModel>
productModelUrlResolver)
	{
		this.productModelUrlResolver = productModelUrlResolver;
	}

	/**
	 * @return the priceDataFactory
	 */
	public PriceDataFactory getPriceDataFactory()
	{
		return priceDataFactory;
	}

	/**
	 * @param priceDataFactory
	 *           the priceDataFactory to set
	 */
	public void setPriceDataFactory(final PriceDataFactory priceDataFactory)
	{
		this.priceDataFactory = priceDataFactory;
	}

	/**
	 * @return the commonI18NService
	 */
	public CommonI18NService getCommonI18NService()
	{
		return commonI18NService;
	}

	/**
	 * @param commonI18NService
	 *           the commonI18NService to set
	 */
	public void setCommonI18NService(final CommonI18NService commonI18NService)
	{
		this.commonI18NService = commonI18NService;
	}



	/**
	 * @return the configurationService
	 */
	public ConfigurationService getConfigurationService()
	{
		return configurationService;
	}

	/**
	 * @param configurationService
	 *           the configurationService to set
	 */
	public void setConfigurationService(ConfigurationService configurationService)
	{
		this.configurationService = configurationService;
	}


	/**
	 *
	 */
	public EntityConverter()
	{
	}
```

```
	/**
	 * This populate the conversion of a DiaProductModel into an Entity object
	 */
	@Override
	public void populate(final DiaProductModel source, final Entity target)
	{

		target.setName(BrainsinsPropertyNameEnum.PRODUCT.toString());
		if (source != null)
		{
			// catalog languages
			final Collection<LanguageModel> languages =
source.getCatalogVersion() != null ? source.getCatalogVersion()
					.getLanguages() : null;

			// Product id

	target.getProperty().add(this.getPropertyType(BrainsinsPropertyNameEnum.IDPRODUCT
.toString(), source.getCode()));

			// Product description There is only a description for Spanish
			target.getProperty().add(

	this.getPropertyType(BrainsinsPropertyNameEnum.NAME.toString(),
source.getDescription(new Locale("es"))));

			// Product image URL
			if (source.getThumbnail() != null)
			{
				target.getProperty().add(

	getPropertyType(BrainsinsPropertyNameEnum.IMAGEURL.toString(),
this.getProductImageUrl(source)));
			}

			// Product url

	target.getProperty().add(getPropertyType(BrainsinsPropertyNameEnum.URL.toString()
, this.getProductUrl(source)));

			// Categories; (EMF) 20151002 It is now a Property not a
MultiProperty
			if (source.getSupercategories() != null &&
source.getSupercategories().size() > 0)
			{
				final Property value =
getCategoriesPropertyType(BrainsinsPropertyNameEnum.CATEGORIES.toString(), source,
languages);
				if (value.getName() != null && value.getValue() != null)
				{
					target.getProperty().add(value);
				}
			}

			// Prices
			if (source.getEurope1Prices() != null &&
source.getEurope1Prices().size() > 0)
			{
				/*
				 * (EMF) ; 20151008. It is necesary that the
MultiProperty's name begin with "multi". In other case, the xml
				 * won't be correct
				 */
				final MultiProperty multiPrice =
this.getPriceMultiPropertyType(BrainsinsPropertyNameEnum.MULTIPRICE.toString(),
						source);
				target.getMultiProperty().add(multiPrice);

	target.getProperty().add(this.getPropertyPrice(BrainsinsPropertyNameEnum.PRICE.to
String(), multiPrice));

			}

			/*
			 * (EMF) ; 20151028 ; Se utiliza la propiedad misc para almacenar
el atributo soldByType del producto, que
```

```
                        * posteriormente será utlizado al añadir el botón de compra
                        */

        target.getProperty().add(getMiscPropertyType(BrainsinsPropertyNameEnum.MISC.toStr
ing(), source));

                }
        }


        /**
         * Method to gets a {@code Property} with name and value received as parameters.
         *
         * @param name
         *            name of property.
         * @param value
         *            value of property.
         * @return a new property.
         */
        private Property getPropertyType(final String name, final String value)
        {
                final Property result = new Property();
                result.setName(name);
                result.setValue(value);
                return result;
        }


        /**
         * Method to gets a {@code NestedProperty} with the language received as first
parameter and value received as second
         * parameter.
         *
         * @param lang
         *            language for value with can be null.
         * @param value
         *            value for property.
         * @return property
         */
        private NestedProperty getNestedPropertyType(final String lang, final String
value)
        {

                final NestedProperty result = new NestedProperty();

                result.setLang(lang != null ? lang : lang);
                result.setValue(value);

                return result;
        }

        /**
         * EMF 20150929
         *
         * Method to gets a {@code String} with the shopCode from the String usergroup
received as parameter
         *
         * Method used in private MultiProperty getPriceMultiPropertyType(final String
propertyName, final DiaProductModel
         * product)
         *
         * @param usergroup
         *            the usergroup that belongs to a product price
         * @return shopCode from the usergroup
         */
        private String getShopCodeFromUserqroup(final String usergroup)
        {

                return (usergroup != null && StringUtils.isNotEmpty(usergroup) &&
StringUtil.substringsOf(usergroup, "_").length >= 1 ? StringUtil
                                .substringsOf(
                                usergroup, "_")[1] : null);
        }



        /**
```

```
         * EMF 20151002
         *
         * Method to gets {@code String} with the product absolute url
BrainsinsPropertyNameEnum.URL
         *
         * @return absolute url product
         */
        private String getProductUrl(final DiaProductModel product)
        {
                String result = "";

                if (product != null && getSiteBaseUrl() != null)
                {
                        result =
this.getSiteBaseUrl().concat(getProductModelUrlResolver().resolve(product));
                }



                return result;
        }


        /**
         * EMF 20151002
         *
         * Method to gets {@code String} with the absolute product image url
BrainsinsPropertyNameEnum.IMAGEURL
         *
         * @param product
         *           product to get url
         * @return absolute product image url
         */
        private String getProductImageUrl(final DiaProductModel product)
        {
                String result = "";


                if (getSiteBaseUrl() != null && product != null && product.getThumbnail()
!= null)
                {
                        // Delete from getSiteBaseUrl() "/compra-online"
                        final String baseUrl = this.getSiteBaseUrl();
                        final int position = baseUrl.lastIndexOf("/compra-online");

                        result = baseUrl.substring(0,
position).concat(product.getThumbnail().getURL());
                }

                return result;

        }

        /**
         * EMF 20151002
         *
         * Method to gets {@code String} with the base site url to compound the complete
url of a product and its image
         *
         * Mehtod used in populate method
         *
         * @return site base url
         */
        private String getSiteBaseUrl()
        {
                return
this.configurationService.getConfiguration().getString("website.dia.http");
        }

        /**
         * EMF 20150929
         *
         * Method to gets a {@code MultiProperty} with the propertyName received as first
parameter and all the prices
         * obtained from the product received as second parameter
         *
         * @param propertyName
```

```
         *           multiproperty name
         * @param product
         *           DiaProductModel from which get prices
         * @return a mew Multiproperty with the prices of the product in all the shops
         */
        private MultiProperty getPriceMultiPropertyType(final String propertyName, final
DiaProductModel product)
        {

                final MultiProperty result = new MultiProperty();
                result.setName(propertyName);

                if (product != null)
                {
                        for (final PriceRowModel priceRow : product.getEurope1Prices())
                        {
                                if (priceRow != null && priceRow.getPrice() != null)
                                {

                                        final String warehouse =
this.getShopCodeFromUserqroup(priceRow.getUg().getCode());
                                        // (EMF), 20160405, [DIAEC-2028]
                                        final PriceData priceData = getPriceRow(product,
priceRow);

                                        if (priceData == null)
                                        {
                                                LOG.info("Price data is null for product: "
+ product.getCode() + " in warehouse: " + warehouse);
                                        }
                                        else
                                        {

        result.getProperty().add(this.getNestedPropertyType(warehouse,
priceData.getValue().toString()));
                                        }
                                }
                        }
                }

                return result;
        }


        /**
         * (EMF), 20160405, [DIAEC-2028]
         *
         * This method gets the price row for a product. The price row that the webpage
shows for a product. This price
         * depends on soldbytype
         *
         * This method is based on
ExtendedCommercePriceService.getWebPriceForProduct(final ProductModel product)
         *
         * @param product
         * @param priceRow
         * @return
         */
        private PriceData getPriceRow(final ProductModel product, final PriceRowModel
priceRow)
        {

                PriceData priceData = null;

                if (product != null && product instanceof DiaProductModel && priceRow !=
null)
                {

                        final SoldByType soldByType = ((DiaProductModel)
product).getSoldByType();

                        // Price by weight specialization
                        if (SoldByType.QUANTITY_WEIGHT.equals(soldByType) &&
priceRow.getUnitFactor() != null)
                        {
                                final BigDecimal bulkPrice =
BigDecimal.valueOf(priceRow.getPrice().doubleValue());
```

```
                                final BigDecimal avgWeight =
BigDecimal.valueOf(((DiaProductModel) product).getAverageWeight()); // grames
                                final BigDecimal unitPrice =
bulkPrice.multiply(avgWeight).divide(new BigDecimal(priceRow.getUnitFactor()), 2);

                                priceData = getPriceDataFactory().create(PriceDataType.BUY,
BigDecimal.valueOf(unitPrice.doubleValue()),
                                                priceRow.getCurrency());
                        }
                        else
                        {
                                priceData = getPriceDataFactory().create(PriceDataType.BUY,
BigDecimal.valueOf(priceRow.getPrice().doubleValue()),
                                                priceRow.getCurrency());
                        }
                }

                return priceData;
        }


        /**
         * EMF 20151008
         *
         * Method to gets a {@code Property} with the propertyName received as first
parameter and the first price obtained
         * from Multiproperty
         *
         * @param propertyName
         *            propertyName
         *
         * @param multiprice
         *            MultiProperty from which gets the first price
         *
         * @return a new Property with the price
         */
        private Property getPropertyPrice(final String propertyName, final MultiProperty
multiprice)
        {

                final Property result = new Property();

                if (multiprice != null && multiprice.getProperty() != null &&
multiprice.getProperty().size() > 0)
                {

                        final NestedProperty singlePrice =
multiprice.getProperty().get(0);

                        if (singlePrice != null)
                        {
                                result.setName(propertyName);
                                result.setValue(singlePrice.getValue());
                        }

                }

                return result;
        }

        /**
         * Modified EMF 20151005 added web filter for category
         *
         * Method to gets {@code Property} with categories from {@code DiaProductModel}
received as first parameter for each
         * {@code LangugageModel} from {@code Collection<LanguageModel>} received as
second parameter.
         *
         * @param propertyName
         *            property name
         * @param product
         *            product to gets categories.
         * @param languages
         *            collection of languages.
         * @returna property with categories from product in each language
         */
```

```
	private Property getCategoriesPropertyType(final String propertyName, final
DiaProductModel product,
			final Collection<LanguageModel> languages)
	{

		final StringBuffer strCategories = new StringBuffer();

		for (final LanguageModel languageModel : languages)
		{

			for (final CategoryModel category : product.getSupercategories())
			{
				if (!(category instanceof ClassificationClassModel) &&
category.getCode() != null
						&&
category.getCode().toLowerCase().contains("web"))
				{
					final String name =
category.getName(Locale.forLanguageTag(languageModel.getIsocode()));

					if (name != null)
					{
						if (strCategories.length() > 0)
						{
							strCategories.append(",");
						}

	strCategories.append(category.getName(Locale.forLanguageTag(languageModel.getIsoc
ode())));
					}

				}
			}
		}

		return this.getPropertyType(propertyName, strCategories.toString());
	}


	/**
	 * Method to gets a {@code Property} with misc porperty from {@code
DiaProductModel} recieved as a first parameter.
	 *
	 * @param product
	 *           product to gets misc property.
	 *
	 * @return a property with misc properties from product.
	 */
	private Property getMiscPropertyType(final String propertyName, final
DiaProductModel product)
	{
		final Map<String, String> miscProperties = new HashMap<String, String>();

		/*
		 * (EMF) 20151105
		 *
		 * Nuevas propiedades necesarias para el formulario del botón Modificado
por (EMF) 20151127
		 */
		if (product != null)
		{
			/* QUANTITY, QUANTITY_WEIGHT, WEIGHT */
			miscProperties.put(DiaProductModel.SOLDBYTYPE,
(product.getSoldByType() != null) ? product.getSoldByType().toString()
					: StringUtils.EMPTY);

			/* Estrellitas */
			miscProperties.put(
					DiaProductModel.AVERAGERATING,
					(product.getProductReviews() != null) ?
this.getAverageProductReviewsApproved(this

	.getProductsReviewsApproved(product.getProductReviews())) : "0");

			/* salesUnit pieces, gram... */
			miscProperties.put(DiaProductModel.UNIT, (product.getUnit() !=
null && product.getUnit().getCode() != null) ? product
```

```
                                        .getUnit().getCode() : StringUtils.EMPTY);

                }

                Boolean bFirst = Boolean.TRUE;
                StringBuffer value = new StringBuffer(SECTION_MISC_START);

                for (final Entry<String, String> entry : miscProperties.entrySet())
                        {
                        if (bFirst.equals(Boolean.TRUE))
                        {
                                bFirst = Boolean.FALSE;
                        }
                        else
                        {
                                value.append(",");
                        }
                        value =
value.append("\"").append(entry.getKey()).append("\"").append(":").append("\"").append(e
ntry.getValue())
                                        .append("\"");
                        }

                value.append(SECTION_MISC_END);

                return getPropertyType(propertyName, value.toString());
        }


        /**
         * [DIAEC-1621] - Carrusel Brainsins: estrellitas con comentarios rechazados
         *
         * EMF - 20160212
         *
         * Calculate the average of the product list reviews that are approved.
         *
         * @param reviews
         *            list approved
         * @return average
         */
        private String getAverageProductReviewsApproved(final List<Double>
productReviews)
        {
                final DecimalFormat decimalFormat = new DecimalFormat("0.00");
                Double sum = 0.0;
                Integer count = 0;

                for (final Double review : productReviews)
                {
                        sum += review;
                        count++;

                }

                return (count > 0 ? decimalFormat.format(sum / count) :
decimalFormat.format(count));

        }

        /**
         * [DIAEC-1621] - Carrusel Brainsins: estrellitas con comentarios rechazados
         *
         * EMF - 20160212
         *
         * This method returns the product reviews approved from the Collection that
contains all the product reviews
         *
         * @param productReviews
         * @return
         */
        private List<Double> getProductsReviewsApproved(final
Collection<CustomerReviewModel> productReviews)
        {
                final List<Double> reviews = new ArrayList<Double>();

                for (final CustomerReviewModel review : productReviews)
                {
```

```
                        if (review.getApprovalStatus() != null &&
review.getApprovalStatus().equals(CustomerReviewApprovalType.APPROVED))
                        {
                                reviews.add(review.getRating());
                        }
                }

                return reviews;
        }


}
```

11.3.1.7 diaproducts.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

        <xsd:element name="recsins">
                <xsd:complexType>
                        <xsd:sequence>
                                <xsd:element name="entities" type="Entities" minOccurs="1"
maxOccurs="1"/>
                        </xsd:sequence>
                        <xsd:attribute name="version" type="xsd:string"/>
                </xsd:complexType>
        </xsd:element>

        <xsd:complexType name="Entities">
                <xsd:sequence>
                        <xsd:element name="entity" minOccurs="1" maxOccurs="unbounded"
type="Entity"/>
                </xsd:sequence>
        </xsd:complexType>

        <xsd:complexType name="Entity">
                <xsd:choice>
                        <xsd:element name="property" type="Property"
maxOccurs="unbounded"/>
                        <xsd:element name="multi_property" type="MultiProperty"
maxOccurs="unbounded"/>
                </xsd:choice>
                <xsd:attribute name="name" type="xsd:string"/>
        </xsd:complexType>

        <xsd:complexType name="Property">
                <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                                <xsd:attribute name="name" type="xsd:string"/>
                        </xsd:extension>
                </xsd:simpleContent>
        </xsd:complexType>

        <xsd:complexType name="MultiProperty">
                <xsd:sequence>
                        <xsd:element name="property" type="NestedProperty" minOccurs="1"
maxOccurs="unbounded"/>
                </xsd:sequence>
                <xsd:attribute name="name" type="xsd:string"/>
        </xsd:complexType>

        <xsd:complexType name="NestedProperty">
                <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                                <xsd:attribute name="lang" type="xsd:string"/>
                        </xsd:extension>
                </xsd:simpleContent>
        </xsd:complexType>

</xsd:schema>
```

11.3.1.8 ObjectFactory.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import javax.xml.bind.annotation.XmlRegistry;


/**
 * This object contains factory methods for each
 * Java content interface and Java element interface
 * generated in the com.b2b2000.dia.brainsins.model package.
 * <p>An ObjectFactory allows you to programatically
 * construct new instances of the Java representation
 * for XML content. The Java representation of XML
 * content can consist of schema derived interfaces
 * and classes representing the binding of schema
 * type definitions, element declarations and model
 * groups.  Factory methods for each of these are
 * provided in this class.
 *
 */
@XmlRegistry
public class ObjectFactory {


    /**
     * Create a new ObjectFactory that can be used to create new instances of schema
derived classes for package: com.b2b2000.dia.brainsins.model
     *
     */
    public ObjectFactory() {
    }

    /**
     * Create an instance of {@link Recsins }
     *
     */
    public Recsins createRecsins() {
        return new Recsins();
    }

    /**
     * Create an instance of {@link Entities }
     *
     */
    public Entities createEntities() {
        return new Entities();
    }

    /**
     * Create an instance of {@link NestedProperty }
     *
     */
    public NestedProperty createNestedProperty() {
        return new NestedProperty();
    }

    /**
     * Create an instance of {@link Property }
     *
     */
    public Property createProperty() {
        return new Property();
    }

    /**
```

```
     * Create an instance of {@link MultiProperty }
     *
     */
    public MultiProperty createMultiProperty() {
        return new MultiProperty();
    }

    /**
     * Create an instance of {@link Entity }
     *
     */
    public Entity createEntity() {
        return new Entity();
    }

}
```

11.3.1.9 Recsins.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;


/**
 * <p>Java class for anonymous complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
 * <pre>
 * &lt;complexType>
 *   &lt;complexContent>
 *     &lt;restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;sequence>
 *         &lt;element name="entities" type="{}Entities"/>
 *       &lt;/sequence>
 *       &lt;attribute name="version" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = {
    "entities"
})
@XmlRootElement(name = "recsins")
public class Recsins {

    @XmlElement(required = true)
    protected Entities entities;
    @XmlAttribute(name = "version")
    protected String version;

    /**
     * Gets the value of the entities property.
     *
```

```
     * @return
     *     possible object is
     *     {@link Entities }
     *
     */
    public Entities getEntities() {
        return entities;
    }

    /**
     * Sets the value of the entities property.
     *
     * @param value
     *     allowed object is
     *     {@link Entities }
     *
     */
    public void setEntities(Entities value) {
        this.entities = value;
    }

    /**
     * Gets the value of the version property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getVersion() {
        return version;
    }

    /**
     * Sets the value of the version property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setVersion(String value) {
        this.version = value;
    }

}
```

11.3.1.10 Entities.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import java.util.ArrayList;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;


/**
 * <p>Java class for Entities complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
```

```
 * <pre>
 * &lt;complexType name="Entities">
 *   &lt;complexContent>
 *     &lt;restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;sequence>
 *         &lt;element name="entity" type="{}Entity" maxOccurs="unbounded"/>
 *       &lt;/sequence>
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "Entities", propOrder = {
    "entity"
})
public class Entities {

    @XmlElement(required = true)
    protected List<Entity> entity;

    /**
     * Gets the value of the entity property.
     *
     * <p>
     * This accessor method returns a reference to the live list,
     * not a snapshot. Therefore any modification you make to the
     * returned list will be present inside the JAXB object.
     * This is why there is not a <CODE>set</CODE> method for the entity property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     * <pre>
     *    getEntity().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list
     * {@link Entity }
     *
     *
     */
    public List<Entity> getEntity() {
        if (entity == null) {
            entity = new ArrayList<Entity>();
        }
        return this.entity;
    }

}
```

11.3.1.11 Entity.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import java.util.ArrayList;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;
```

```
/**
 * <p>Java class for Entity complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
 * <pre>
 * &lt;complexType name="Entity">
 *   &lt;complexContent>
 *     &lt;restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;choice>
 *         &lt;element name="property" type="{}Property" maxOccurs="unbounded"/>
 *         &lt;element name="multi_property" type="{}MultiProperty"
maxOccurs="unbounded"/>
 *       &lt;/choice>
 *       &lt;attribute name="name" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "Entity", propOrder = {
    "property",
    "multiProperty"
})
public class Entity {

    protected List<Property> property;
    @XmlElement(name = "multi_property")
    protected List<MultiProperty> multiProperty;
    @XmlAttribute(name = "name")
    protected String name;

    /**
     * Gets the value of the property property.
     *
     * <p>
     * This accessor method returns a reference to the live list,
     * not a snapshot. Therefore any modification you make to the
     * returned list will be present inside the JAXB object.
     * This is why there is not a <CODE>set</CODE> method for the property property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     * <pre>
     *    getProperty().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list
     * {@link Property }
     *
     *
     */
    public List<Property> getProperty() {
        if (property == null) {
            property = new ArrayList<Property>();
        }
        return this.property;
    }

    /**
     * Gets the value of the multiProperty property.
     *
     * <p>
     * This accessor method returns a reference to the live list,
     * not a snapshot. Therefore any modification you make to the
     * returned list will be present inside the JAXB object.
     * This is why there is not a <CODE>set</CODE> method for the multiProperty
property.
     *
```

```
     * <p>
     * For example, to add a new item, do as follows:
     * <pre>
     *    getMultiProperty().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list
     * {@link MultiProperty }
     *
     *
     */
    public List<MultiProperty> getMultiProperty() {
        if (multiProperty == null) {
            multiProperty = new ArrayList<MultiProperty>();
        }
        return this.multiProperty;
    }

    /**
     * Gets the value of the name property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getName() {
        return name;
    }

    /**
     * Sets the value of the name property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setName(String value) {
        this.name = value;
    }

}
```

11.3.1.12 Property.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlType;
import javax.xml.bind.annotation.XmlValue;


/**
 * <p>Java class for Property complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
 * <pre>
 * &lt;complexType name="Property">
```

```
 *   &lt;simpleContent>
 *     &lt;extension base="&lt;http://www.w3.org/2001/XMLSchema>string">
 *       &lt;attribute name="name" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/extension>
 *   &lt;/simpleContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "Property", propOrder = {
    "value"
})
public class Property {

    @XmlValue
    protected String value;
    @XmlAttribute(name = "name")
    protected String name;

    /**
     * Gets the value of the value property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getValue() {
        return value;
    }

    /**
     * Sets the value of the value property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setValue(String value) {
        this.value = value;
    }

    /**
     * Gets the value of the name property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getName() {
        return name;
    }

    /**
     * Sets the value of the name property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setName(String value) {
        this.name = value;
    }

}
```

11.3.1.13 MultiProperty.java

```
//
```

```
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import java.util.ArrayList;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlType;


/**
 * <p>Java class for MultiProperty complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
 * <pre>
 * &lt;complexType name="MultiProperty">
 *   &lt;complexContent>
 *     &lt;restriction base="{http://www.w3.org/2001/XMLSchema}anyType">
 *       &lt;sequence>
 *         &lt;element name="property" type="{}NestedProperty" maxOccurs="unbounded"/>
 *       &lt;/sequence>
 *       &lt;attribute name="name" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/restriction>
 *   &lt;/complexContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "MultiProperty", propOrder = {
    "property"
})
public class MultiProperty {

    @XmlElement(required = true)
    protected List<NestedProperty> property;
    @XmlAttribute(name = "name")
    protected String name;

    /**
     * Gets the value of the property property.
     *
     * <p>
     * This accessor method returns a reference to the live list,
     * not a snapshot. Therefore any modification you make to the
     * returned list will be present inside the JAXB object.
     * This is why there is not a <CODE>set</CODE> method for the property property.
     *
     * <p>
     * For example, to add a new item, do as follows:
     * <pre>
     *    getProperty().add(newItem);
     * </pre>
     *
     *
     * <p>
     * Objects of the following type(s) are allowed in the list
     * {@link NestedProperty }
     *
     *
     */
    public List<NestedProperty> getProperty() {
        if (property == null) {
            property = new ArrayList<NestedProperty>();
        }
```

```
        return this.property;
    }

    /**
     * Gets the value of the name property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getName() {
        return name;
    }

    /**
     * Sets the value of the name property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setName(String value) {
        this.name = value;
    }

}
```

11.3.1.14 NestedProperty.java

```
//
// This file was generated by the JavaTM Architecture for XML Binding(JAXB) Reference
Implementation, v2.2.4-2
// See <a href="http://java.sun.com/xml/jaxb">http://java.sun.com/xml/jaxb</a>
// Any modifications to this file will be lost upon recompilation of the source schema.
// Generated on: 2015.10.13 at 10:03:56 AM CEST
//


package com.b2b2000.dia.brainsins.model;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlType;
import javax.xml.bind.annotation.XmlValue;


/**
 * <p>Java class for NestedProperty complex type.
 *
 * <p>The following schema fragment specifies the expected content contained within this
class.
 *
 * <pre>
 * &lt;complexType name="NestedProperty">
 *   &lt;simpleContent>
 *     &lt;extension base="&lt;http://www.w3.org/2001/XMLSchema>string">
 *       &lt;attribute name="lang" type="{http://www.w3.org/2001/XMLSchema}string" />
 *     &lt;/extension>
 *   &lt;/simpleContent>
 * &lt;/complexType>
 * </pre>
 *
 *
 */
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "NestedProperty", propOrder = {
    "value"
})
public class NestedProperty {

    @XmlValue
```

```
    protected String value;
    @XmlAttribute(name = "lang")
    protected String lang;

    /**
     * Gets the value of the value property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getValue() {
        return value;
    }

    /**
     * Sets the value of the value property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setValue(String value) {
        this.value = value;
    }

    /**
     * Gets the value of the lang property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getLang() {
        return lang;
    }

    /**
     * Sets the value of the lang property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setLang(String value) {
        this.lang = value;
    }

}
```

11.3.2 Extensión diaacceleratorstorefront

11.3.2.1 HomePageController.java

```
/*
 * [y] hybris Platform
 *
 * Copyright (c) 2000-2013 hybris AG
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of hybris
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with hybris.
 *
 *
 */
```

```
package com.b2b2000.dia.storefront.controllers.pages;

import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.cms2.model.pages.AbstractPageModel;

import javax.annotation.Resource;

import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.b2b2000.dia.facades.store.LocalStorePreferencesFacade;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.controllers.util.GlobalMessages;


/**
 * Controller for home page
 */
@Controller
@Scope("tenant")
@RequestMapping("/")
public class HomePageController extends DiaAbstractPageController
{

	@Resource(name = "localStorePreferencesFacade")
	private LocalStorePreferencesFacade localStorePreferencesFacade;

	/**
	 * EMF, Service to integrate BrainSINS
	 */
	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	@RequestMapping(method = RequestMethod.GET)
	public String home(
			@RequestParam(value = "logout", defaultValue = "false") final
boolean logout,
			@RequestParam(required = false, value = "countryAwareRedirect",
defaultValue = "false") final boolean countryAwareRedirect,
			final Model model, final RedirectAttributes redirectModel) throws
CMSItemNotFoundException
	{
		if (logout)
		{
			//[DSA 20160408][DIAEC-2081]Logout de Argentina, redireccion
			if (countryAwareRedirect)
			{
				final String urlLogout =
this.configurationService.getConfiguration().getString("website.clubdia.logout.ar");
				return REDIRECT_PREFIX + urlLogout;
			}
			else
			{
				GlobalMessages.addFlashMessage(redirectModel,
GlobalMessages.INFO_MESSAGES_HOLDER,
						"account.confirmation.signout.title");
				return REDIRECT_PREFIX + ROOT;
			}
		}

		//20150401 (DSA), incidencia DIAEC-108, necesario para cargar
correctamente el point of service
		localStorePreferencesFacade.getPointOfServiceModel();

		storeCmsPageInModel(model, getContentPageForLabelOrId(null));
		setUpMetaDataForContentPage(model, getContentPageForLabelOrId(null));
		updatePageTitle(model, getContentPageForLabelOrId(null));

		brainsinsStorefrontService.updateModelForDefaultPage(model);

		return getViewForPage(model);
```

```
        }

        protected void updatePageTitle(final Model model, final AbstractPageModel
cmsPage)
        {
                storeContentPageTitleInModel(model,
getPageTitleResolver().resolveHomePageTitle(cmsPage.getTitle()));
        }
}
```

11.3.2.2 ProductPageController.java

```
/*
 * [y] hybris Platform
 *
 * Copyright (c) 2000-2013 hybris AG
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of hybris
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with hybris.
 *
 *
 */
package com.b2b2000.dia.storefront.controllers.pages;

import de.hybris.platform.acceleratorservices.controllers.page.PageType;
import de.hybris.platform.catalog.enums.ProductReferenceTypeEnum;
import de.hybris.platform.category.CategoryService;
import de.hybris.platform.category.model.CategoryModel;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.cms2.model.pages.AbstractPageModel;
import de.hybris.platform.cms2.servicelayer.services.CMSPageService;
import de.hybris.platform.commercefacades.product.ProductOption;
import de.hybris.platform.commercefacades.product.data.BaseOptionData;
import de.hybris.platform.commercefacades.product.data.CategoryData;
import de.hybris.platform.commercefacades.product.data.ImageData;
import de.hybris.platform.commercefacades.product.data.ImageDataType;
import de.hybris.platform.commercefacades.product.data.ProductData;
import de.hybris.platform.commercefacades.product.data.ProductReferenceData;
import de.hybris.platform.commercefacades.product.data.ReviewData;
import de.hybris.platform.commercefacades.user.data.CustomerData;
import de.hybris.platform.commerceservices.strategies.CustomerNameStrategy;
import de.hybris.platform.commerceservices.url.UrlResolver;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.product.ProductService;
import de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;
import de.hybris.platform.storelocator.model.PointOfServiceModel;

import java.io.UnsupportedEncodingException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.annotation.Resource;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.ordersplitting.impl.OrderSplittingUtils;
import com.b2b2000.dia.facades.order.data.FruitVerdureInfoAuxData;
import com.b2b2000.dia.facades.order.data.FruitVerdureInfoData;
import com.b2b2000.dia.facades.order.data.NutritionalInfoAuxData;
import com.b2b2000.dia.facades.order.data.NutritionalInfoData;
import com.b2b2000.dia.facades.product.DiaProductFacade;
import com.b2b2000.dia.facades.product.impl.ExtendedProductFacade;
import com.b2b2000.dia.facades.seo.SeoLinkManagerFacade;
import com.b2b2000.dia.facades.store.LocalStorePreferencesFacade;
import com.b2b2000.dia.facades.supplier.SupplierFacade;
import com.b2b2000.dia.facades.user.data.SupplierData;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.breadcrumb.Breadcrumb;
import com.b2b2000.dia.storefront.breadcrumb.impl.ProductBreadcrumbBuilder;
import com.b2b2000.dia.storefront.constants.WebConstants;
import com.b2b2000.dia.storefront.controllers.ControllerConstants;
import com.b2b2000.dia.storefront.controllers.util.GlobalMessages;
import com.b2b2000.dia.storefront.forms.ReviewForm;
import com.b2b2000.dia.storefront.forms.validation.ReviewValidator;
import com.b2b2000.dia.storefront.util.DiaProductDetailUtil;
import com.b2b2000.dia.storefront.util.MetaSanitizerUtil;
import com.b2b2000.dia.storefront.util.XSSFilterUtil;
import com.b2b2000.dia.storefront.variants.VariantSortStrategy;


/**
 * Controller for product details page
 */
@Controller
@Scope("tenant")
@RequestMapping(value = "/**/p")
public class ProductPageController extends DiaAbstractPageController
{
	@SuppressWarnings("unused")
	private static final Logger LOG = Logger.getLogger(ProductPageController.class);

	/**
	 * We use this suffix pattern because of an issue with Spring 3.1 where a Uri
value is incorrectly extracted if it
	 * contains on or more '.' characters. Please see
https://jira.springsource.org/browse/SPR-6164 for a discussion on
	 * the issue and future resolution.
	 */
	private static final String PRODUCT_CODE_PATH_VARIABLE_PATTERN =
"/{productCode:.*}";
	private static final String REVIEWS_PATH_VARIABLE_PATTERN =
"{numberOfReviews:.*}";

	@Resource(name = "productModelUrlResolver")
	private UrlResolver<ProductModel> productModelUrlResolver;

	@Resource(name = "productFacade")
	private DiaProductFacade productFacade;

	@Resource(name = "extendedProductFacade")
	private ExtendedProductFacade extendedProductFacade;

	@Resource(name = "productService")
	private ProductService productService;

	@Resource(name = "productBreadcrumbBuilder")
	private ProductBreadcrumbBuilder productBreadcrumbBuilder;

	@Resource(name = "cmsPageService")
	private CMSPageService cmsPageService;

	@Resource(name = "variantSortStrategy")
	private VariantSortStrategy variantSortStrategy;
```

```
	@Resource(name = "reviewValidator")
	private ReviewValidator reviewValidator;

	@Resource(name = "customerNameStrategy")
	private CustomerNameStrategy customerNameStrategy;

	@Resource(name = "localStorePreferencesFacade")
	private LocalStorePreferencesFacade localStorePreferencesFacade;

	@Resource(name = "supplierFacade")
	private SupplierFacade supplierFacade;

	// Incidencia DIAEC - 584, 20150710, EMF
	@Resource(name = "categoryService")
	private CategoryService categoryService;

	/**
	 * SZF, 20151217, [DIAEC-1281] Manager de links para el SEO
	 */
	@Resource(name = "seoLinkManagerFacade")
	private SeoLinkManagerFacade seoLinkManagerFacade;

	/**
	 * EMF, Service to integrate BrainSINS
	 */
	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	@RequestMapping(value = PRODUCT_CODE_PATH_VARIABLE_PATTERN, method =
RequestMethod.GET)
	public String productDetail(@PathVariable("productCode") final String
productCode, final Model model,
			final HttpServletRequest request, final HttpServletResponse
response) throws CMSItemNotFoundException,
			UnsupportedEncodingException
	{
		final ProductModel productModel =
productService.getProductForCode(productCode);
		final String redirection = checkRequestUrl(request, response,
productModelUrlResolver.resolve(productModel));
		if (StringUtils.isNotEmpty(redirection))
		{
			return redirection;
		}

		// MRG - 20141212 - WebVisibility
		validateWebVisibility(productModel);
		//20150401 (DSA), incidencia DIAEC-108, necesario para cargar
correctamente el point of service
		localStorePreferencesFacade.getPointOfServiceModel();

		updatePageTitle(productModel, model);
		// MRG - 20141212 - Se obtiene el warehouse del usuario
		final String whareHouse = obtainWhareHouse();
		populateProductDetailForDisplay(productModel, model, request,
whareHouse);
		model.addAttribute(createReviewForm());
		final List<ProductReferenceData> productReferences =
productFacade.getProductReferencesForCode(productCode,
				Arrays.asList(ProductReferenceTypeEnum.SIMILAR,
ProductReferenceTypeEnum.ACCESSORIES),
				Arrays.asList(ProductOption.BASIC, ProductOption.PRICE),
null);
		model.addAttribute("productReferences", productReferences);
		model.addAttribute("pageType", PageType.PRODUCT.name());

		// recuperamos el productData del modelo
		final ProductData productData = (ProductData)
model.asMap().get("product");
		//20140805 FPR - Google Tag Manager, product code
		if (productData != null)
		{
			model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_PRODUCT_PARAM,
productData.getCode());
			//20141110 IGF - Google Tag Manager, identifier
			model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_IDENTIFIER,
productData.getCode());
```

```
                        //20141110 IGF - Google Tag Manager, fn
                        // (EMF), 20160408, added repalecAll [DIAEC-2070]
                        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_FN,
productData.getDescription().replaceAll("[\n\r]", ""));
                        //20141110 IGF - Google Tag Manager, brand
                        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_BRAND,
productData.getManufacturer());
                        //20141110 IGF - Google Tag Manager, category -> nos piden:
'Productos', 'Alimentacion', 'Patatas Fritas, aperitivos y frutos secos', 'Patatas
fritas lisas' ], // path de categor
                        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_CATEGORY,
getPathCategoryFromBreadCrumbs(model));

                        //20150710 EMF - Google Tag Manager, categorySFS [001.000.00000,
001.001.00001…]

        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_CATEGORY_PARAM_CATEGORY_SFS,
getSFSCategoriesByProduct(productData));

                        //20141110 IGF - Google Tag Manager, prize
                        if (productData.getPrice() != null)
                        {
                                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_PRIZE,
productData.getPrice().getValue());
                                //20141110 IGF - Google Tag Manager, amount
                                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_AMOUNT,
productData.getPrice().getValue());
                                //20141110 IGF - Google Tag Manager, currency

        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_CURRENCY,
productData.getPrice().getCurrencyIso());
                        }
                        //20141110 IGF - Google Tag Manager, url
                        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_URL,
getURLBaseCompraOnline() + productData.getUrl());

                        //20141208 FPR NULL SAFE!!!!!!!!!!!!!
                        if (productModel.getPicture() != null)
                        {
                                //20141110 IGF - Google Tag Manager, photo
                                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_PHOTO,
getURLBase() + productModel.getPicture().getURL());
                        }
                }

                // EMF - Brainsins
                brainsinsStorefrontService.updateModelForProductPage(model, productCode);

                final String metaKeywords =
MetaSanitizerUtil.sanitizeKeywords(productModel.getKeywords());
                final String metaDescription =
MetaSanitizerUtil.sanitizeDescription(productModel.getDescription());
                setUpMetaData(model, metaKeywords, metaDescription);
                return getViewForPage(model);
        }
```

11.3.2.3 CategoryPageController.java

```
* [y] hybris Platform
 *
 * Copyright (c) 2000-2013 hybris AG
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of hybris
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with hybris.
 *
 *
 */
package com.b2b2000.dia.storefront.controllers.pages;


import de.hybris.platform.acceleratorservices.controllers.page.PageType;
```

```
import de.hybris.platform.acceleratorservices.customer.CustomerLocationService;
import de.hybris.platform.acceleratorservices.data.RequestContextData;
import de.hybris.platform.category.model.CategoryModel;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.cms2.model.pages.CategoryPageModel;
import de.hybris.platform.cms2.servicelayer.services.CMSPageService;
import de.hybris.platform.commercefacades.product.data.CategoryData;
import de.hybris.platform.commercefacades.product.data.ProductData;
import de.hybris.platform.commercefacades.search.ProductSearchFacade;
import de.hybris.platform.commercefacades.search.data.SearchQueryData;
import de.hybris.platform.commercefacades.search.data.SearchStateData;
import de.hybris.platform.commerceservices.category.CommerceCategoryService;
import de.hybris.platform.commerceservices.search.facetdata.BreadcrumbData;
import de.hybris.platform.commerceservices.search.facetdata.FacetData;
import de.hybris.platform.commerceservices.search.facetdata.FacetRefinement;
import
de.hybris.platform.commerceservices.search.facetdata.ProductCategorySearchPageData;
import de.hybris.platform.commerceservices.search.pagedata.PageableData;
import de.hybris.platform.commerceservices.url.UrlResolver;
import de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;
import de.hybris.platform.servicelayer.search.FlexibleSearchService;

import java.io.UnsupportedEncodingException;
import java.util.Collection;
import java.util.List;
import java.util.Map;

import javax.annotation.Resource;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.b2b2000.dia.core.model.product.ProductModelUtils;
import com.b2b2000.dia.facades.seo.SeoLinkManagerFacade;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.breadcrumb.impl.SearchBreadcrumbBuilder;
import com.b2b2000.dia.storefront.constants.WebConstants;
import com.b2b2000.dia.storefront.controllers.util.GlobalMessages;
import com.b2b2000.dia.storefront.util.MetaSanitizerUtil;


/**
 * Controller for a category page
 */
@Controller
@Scope("tenant")
@RequestMapping(value = "/**/c")
public class CategoryPageController extends AbstractSearchPageController
{
	protected static final Logger LOG =
Logger.getLogger(CategoryPageController.class);

	private static final String PRODUCT_GRID_PAGE = "category/productGridPage";
	/**
	 * We use this suffix pattern because of an issue with Spring 3.1 where a Uri
value is incorrectly extracted if it
	 * contains on or more '.' characters. Please see
https://jira.springsource.org/browse/SPR-6164 for a discussion on
	 * the issue and future resolution.
	 */
	private static final String CATEGORY_CODE_PATH_VARIABLE_PATTERN =
"/{categoryCode:.*}";
```

```
	@Resource(name = "productSearchFacade")
	private ProductSearchFacade<ProductData> productSearchFacade;

	@Resource(name = "cmsPageService")
	private CMSPageService cmsPageService;

	@Resource(name = "commerceCategoryService")
	private CommerceCategoryService commerceCategoryService;

	@Resource(name = "searchBreadcrumbBuilder")
	private SearchBreadcrumbBuilder searchBreadcrumbBuilder;

	@Resource(name = "categoryModelUrlResolver")
	private UrlResolver<CategoryModel> categoryModelUrlResolver;

	@Resource(name = "customerLocationService")
	private CustomerLocationService customerLocationService;

	@Resource(name = "flexibleSearchService")
	private FlexibleSearchService flexibleSearchService;

	/**
	 * SZF, 20151217, [DIAEC-1281] Manager de links para el SEO
	 */
	@Resource(name = "seoLinkManagerFacade")
	private SeoLinkManagerFacade seoLinkManagerFacade;

	/**
	 * EMF, Service to integrate BrainSINS
	 */
	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	@RequestMapping(value = ***CATEGORY_CODE_PATH_VARIABLE_PATTERN***, method =
RequestMethod.***GET***)
	public String category(@PathVariable("categoryCode") final String categoryCode,
			@RequestParam(value = "q", required = false) final String
searchQuery,
			@RequestParam(value = "page", defaultValue = "0") final int page,
			@RequestParam(value = "show", defaultValue = "Page") final
ShowMode showMode,
			@RequestParam(value = "sort", required = false) final String
sortCode, final Model model,
			final HttpServletRequest request, final HttpServletResponse
response, final RedirectAttributes redirect)
			throws UnsupportedEncodingException
	{
		final CategoryModel category =
commerceCategoryService.getCategoryForCode(categoryCode);

		final String redirection = checkRequestUrl(request, response,
categoryModelUrlResolver.resolve(category));
		if (StringUtils.*isNotEmpty*(redirection))
		{
			return redirection;
		}

		final CategoryPageModel categoryPage = getCategoryPage(category);

		final CategorySearchEvaluator categorySearch = new
CategorySearchEvaluator(categoryCode, searchQuery, page, showMode,
				sortCode, categoryPage);
		categorySearch.doSearch();

		final ProductCategorySearchPageData<SearchStateData, ProductData,
CategoryData> searchPageData = categorySearch
				.getSearchPageData();

		/*
		 * (JAF), 20140502, if clicking on a category raise 0 results then switch
back to parent category
		 */
		if (searchPageData.getResults().isEmpty())
		{
			final CategoryModel parentCategory =
ProductModelUtils.*getParentWebCategory*(category);
```

```
                    if (parentCategory != null)
                    {
                                GlobalMessages.addMessage(model,
GlobalMessages.INFO_MESSAGES_HOLDER, "search.empty.category", new Object[]
                                { category.getName(), parentCategory.getName() });
                                GlobalMessages.addFlashMessage(redirect,
GlobalMessages.INFO_MESSAGES_HOLDER, "search.empty.category", new Object[]
                                { category.getName(), parentCategory.getName() });

                                return category(parentCategory.getCode(), searchQuery,
page, showMode, sortCode, model, request, response, redirect);
                    }
            }

            final boolean showCategoriesOnly = categorySearch.isShowCategoriesOnly();

            storeCmsPageInModel(model, categorySearch.categoryPage);
            storeContinueUrl(request);

            populateModel(model, searchPageData, showMode);
            model.addAttribute(WebConstants.BREADCRUMBS_KEY,
searchBreadcrumbBuilder.getBreadcrumbs(categoryCode, searchPageData));
            model.addAttribute("showCategoriesOnly",
Boolean.valueOf(showCategoriesOnly));
            model.addAttribute("categoryName", category.getName());
            //model.addAttribute("pageType", PageType.Category);
            model.addAttribute("pageType", PageType.CATEGORY.name());
            model.addAttribute("userLocation",
customerLocationService.getUserLocation());

            //20140805 FPR - Google Tag Manager, category code, se elimina el
prefijo.
            // commented Incidencia DIAEC-864, 20150814, EMF.
            //
       model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_CATEGORY_PARAM,
            //                          !StringUtils.isEmpty(categoryCode) ?
categoryCode.replaceAll("WEB.", StringUtils.EMPTY) : StringUtils.EMPTY);

            /*
             * Incidencia DIAEC-864, 20150814, EMF. Propiedad para el parametro
categoryid, se solicita mostrar siempre la
             * categoria padre
             */
            final String newCategoryCode = this.getSuperCategory(categoryCode);
            model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_CATEGORY_PARAM,
                              !StringUtils.isEmpty(newCategoryCode) ?
newCategoryCode.replaceAll("WEB.", StringUtils.EMPTY) : StringUtils.EMPTY);

            updatePageTitle(category, searchPageData.getBreadcrumbs(), model);
            final RequestContextData requestContextData =
getRequestContextData(request);
            requestContextData.setCategory(category);
            requestContextData.setSearch(searchPageData);

            final String metaKeywords =
MetaSanitizerUtil.sanitizeKeywords(category.getKeywords());
            final String metaDescription =
MetaSanitizerUtil.sanitizeDescription(category.getDescription());
            setUpMetaData(model, metaKeywords, metaDescription);
            /*
             * (JAF), 20140505, according to SEO recommendations (see
#setUpNavigationLinks)
             *
             * 1.5. Listados Una fuente de errores de contenido duplicado son los
filtros y distintas ordenaciones que se
             * aplican a los listados. Aunque se ha incluido una meta etiqueta robots
para evitar que estos sean indexados, su
             * contenido están mal generado ya que en vez de: <meta name="robots"
content="no-index,follow" /> La metaetiqueta
             * debería ser: <meta name="robots" content="noindex,follow" />
             *
             *
             * Paginación de listados
             *
             * Se debe evitar la existencia de subcategorías innecesarias, ya que
además de perjudicar el SEO, dificultan la
             * navegación de la web.
```

```
                *
                * Lo que se debe evitar es que existan categorías que sólo contengan una
subcategoría y que, por tanto, dupliquen
                * los listados. Cuando existen varias subcategorías se soluciona esto,
ya que cada subcategoría es un subconjunto
                * de la categoría padre.
                *
                * Las opciones para solucionar esto es no incluir las subcategorías
hasta que existan varias y añadir una meta
                * robots noindex a la subcategoría inicial (teniendo en cuenta que se
debe eliminar este noindex cuando existan
                * varias subcategorías)
                */

               //final CategoryModel parentCategory =
ProductModelUtils.getParentWebCategory(category);
               //final boolean duplicated = ((parentCategory != null) &&
CollectionUtils.size(parentCategory.getCategories()) == 1);
               //final String indexOrNot = !duplicated && StringUtils.isEmpty(sortCode)
? "index" : "noindex";
               //model.addAttribute("metaRobots", indexOrNot + ",follow");

               //DSA, 20150508, (DIAEC-82), las paginas de busqueda tienen que llevar la
siguiente meta etiqueta:
               //<meta name="robots" content="noindex,follow">
               //SZF, 20151201, [DIAEC-1281] Cambiar, segun Internet Republica 'noindex'
por 'index'
               model.addAttribute("metaRobots", "index,follow");

               setUpNavigationLinks(request, searchPageData, model);

               //construir la etiqueta canonical
               // SZF, 20151029, [DIAEC-1130] Modificados los parametros, para pasar el
searchPageData
               setUpCategoriesCanonicalLinks(request, searchPageData, model);

               // EMF - 20160120 - Brainsins
               brainsinsStorefrontService.updateModelForCategoryPage(model,
category.getName());

               //setUpCanonicalLinks2(searchPageData, model);

               return getViewPage(categorySearch.categoryPage);
       }
```

11.3.2.4 CartPageController.java

```
* [y] hybris Platform
 *
 * Copyright (c) 2000-2013 hybris AG
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of hybris
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with hybris.
 *
 *
 */
package com.b2b2000.dia.storefront.controllers.pages;

import de.hybris.platform.acceleratorservices.config.SiteConfigService;
import de.hybris.platform.acceleratorservices.controllers.page.PageType;
import de.hybris.platform.acceleratorservices.enums.CheckoutFlowEnum;
import de.hybris.platform.acceleratorservices.enums.CheckoutPciOptionEnum;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.commercefacades.customer.CustomerFacade;
import de.hybris.platform.commercefacades.order.data.CartData;
import de.hybris.platform.commercefacades.order.data.CartModificationData;
import de.hybris.platform.commercefacades.order.data.CartRestorationData;
import de.hybris.platform.commercefacades.order.data.OrderEntryData;
import de.hybris.platform.commercefacades.product.data.PriceData;
import de.hybris.platform.commercefacades.product.data.PromotionData;
```

```
import de.hybris.platform.commercefacades.promotion.CommercePromotionFacade;
import de.hybris.platform.commercefacades.voucher.exceptions.VoucherOperationException;
import de.hybris.platform.commerceservices.order.CommerceCartModificationException;
import de.hybris.platform.order.exceptions.CalculationException;
import de.hybris.platform.servicelayer.i18n.I18NService;
import de.hybris.platform.servicelayer.session.SessionService;
import de.hybris.platform.voucher.VoucherService;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import javax.annotation.Resource;
import javax.servlet.http.HttpServletRequest;
import javax.validation.Valid;

import org.apache.commons.collections.ListUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.validation.ObjectError;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.b2b2000.dia.core.voucher.impl.NotCumulativeRestrictionUtil;
import com.b2b2000.dia.enums.SoldByType;
import com.b2b2000.dia.facades.flow.impl.SessionOverrideCheckoutFlowFacade;
import com.b2b2000.dia.facades.order.DiaCartFacade;
import com.b2b2000.dia.facades.voucher.DiaVoucherFacade;
import com.b2b2000.dia.storefront.annotations.RequireHardLogIn;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.breadcrumb.ResourceBreadcrumbBuilder;
import com.b2b2000.dia.storefront.constants.WebConstants;
import com.b2b2000.dia.storefront.controllers.ControllerConstants;
import com.b2b2000.dia.storefront.controllers.util.GlobalMessages;
import com.b2b2000.dia.storefront.forms.UpdateQuantityForm;


/**
 * Controller for cart page
 */
@Controller
@Scope("tenant")
@RequestMapping(value = "/cart")
public class CartPageController extends DiaAbstractPageController
{
	protected static final Logger LOG = Logger.getLogger(CartPageController.class);

	public static final String SHOW_CHECKOUT_STRATEGY_OPTIONS =
"storefront.show.checkout.flows";

	private static final String CART_CMS_PAGE_LABEL = "cart";
	private static final String CONTINUE_URL = "continueUrl";

	@Resource(name = "cartFacade")
	private DiaCartFacade cartFacade;

	@Resource(name = "siteConfigService")
	private SiteConfigService siteConfigService;

	@Resource(name = "sessionService")
	private SessionService sessionService;

	@Resource(name = "simpleBreadcrumbBuilder")
	private ResourceBreadcrumbBuilder resourceBreadcrumbBuilder;

	@Resource(name = "voucherFacade")
	private DiaVoucherFacade voucherFacade;

	@Resource(name = "i18nService")
```

```
	private I18NService i18nService;

	@Resource(name = "customerFacade")
	private CustomerFacade customerFacade;

	/**
	 * (EMF) 20150716
	 */
	@Resource(name = "voucherService")
	private VoucherService voucherService;

	@Resource(name = "commercePromotionFacade")
	private CommercePromotionFacade commercePromotionFacade;

	/**
	 * EMF, Service to integrate BrainSINS
	 */
	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	protected void createProductList(final Model model) throws
CMSItemNotFoundException
	{
		/*
		 * (JAF), 20140930, original implementation does not recalculate the
cart, why the hell are we doing it at this
		 * point?? (JAF), 20141013, PERFORMANCE, from now on calculation is
removed at this point. After a better
		 * understanding of how Hybris works, I don't see any need for updating
the cart at this point.
		 */
		//			try
		//			{
		//				((DiaCartFacade) cartFacade).recalculateCart();
		//			}
		//			catch (final CalculationException e)
		//			{
		//				LOG.error("Unable to update the cart", e);
		//			}

		CartData cartData = cartFacade.getSessionCartWithEntryOrdering(true);
		boolean hasPickUpCartEntries = false;
		final StringBuffer productCodes = new StringBuffer();
		final StringBuffer productAmounts = new StringBuffer();
		final StringBuffer productQuantities = new StringBuffer();

		if (cartData.getEntries() != null && !cartData.getEntries().isEmpty())
		{


			for (final OrderEntryData entry : cartData.getEntries())
			{
				if (!hasPickUpCartEntries &&
entry.getDeliveryPointOfService() != null)
				{
					hasPickUpCartEntries = true;
				}
				final UpdateQuantityForm uqf = new UpdateQuantityForm();
				uqf.setQuantity(entry.getQuantity());
				model.addAttribute("updateQuantityForm" +
entry.getEntryNumber(), uqf);

				if (entry.getProduct() != null)
				{
					//codigo de cada producto
					productCodes.append(entry.getProduct().getCode());
					productCodes.append(",");

					//precio de cada producto

	productAmounts.append(entry.getBasePrice().getFormattedValue());
					productAmounts.append(",");

					//cantidad de cada producto
					// si es peso y en los de unidades-peso siempre
vamos a mostrar "1", sino la cantidad
```

```
                                        if
(entry.getSoldByType().getCode().equals(SoldByType.QUANTITY))
                                        {

       productQuantities.append(entry.getQuantity().toString());
                                        }
                                        else
                                        {
                                               productQuantities.append("1");
                                        }
                                        productQuantities.append(",");
                                }
                        }
                }
                else
                {
                        //PFA 03122014: Si se ha vaciado el carrito--> Borrar cupones.
                        final Collection<String> voucherCodes =
this.voucherFacade.getVoucherCodesFromSessionCart();
                        if (voucherCodes != null && !voucherCodes.isEmpty())
                        {

                                try
                                {
                                        for (final String voucherCode : voucherCodes)
                                        {

       this.voucherFacade.releaseVoucher(voucherCode);
                                        }

                                        cartData =
cartFacade.getSessionCartWithEntryOrdering(true);
                                }
                                catch (final VoucherOperationException e)
                                {

                                        // Success in removing entry

                                        GlobalMessages.addErrorMessage(model,
"voucher.error.error");
                                        LOG.error("Error al liberar el cupón. Total del
carrito: " + cartData.getTotalDiscounts());
                                        // Redirect to the cart page on update success so
that the browser doesn't re-post again

                                }

                        }







                }

                model.addAttribute("cartData", cartData);
                model.addAttribute("hasPickUpCartEntries",
Boolean.valueOf(hasPickUpCartEntries));

                //20140805 FPR - Google Tag Manager, atributos para la página del carrito
                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_BASKET_PRICE_PARAM,
(cartData.getTotalPrice() != null) ? cartData
                                .getTotalPrice().getFormattedValue() : StringUtils.EMPTY);
                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_BASKET_PRODUCTS_PARAM,
                                StringUtils.removeEnd(productCodes.toString(), ","));

                //20150306 DSA - Google Tag Manager, anyadidos atributos para la página
del carrito
                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_BASKET_AMOUNT_PARAM,
                                StringUtils.removeEnd(productAmounts.toString(), ","));
                model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_BASKET_QUANTITY_PARAM,
                                StringUtils.removeEnd(productQuantities.toString(), ","));



                storeCmsPageInModel(model,
getContentPageForLabelOrId(CART_CMS_PAGE_LABEL));
```

```
                setUpMetaDataForContentPage(model,
getContentPageForLabelOrId(CART_CMS_PAGE_LABEL));

                model.addAttribute(WebConstants.BREADCRUMBS_KEY,
resourceBreadcrumbBuilder.getBreadcrumbs("breadcrumb.cart"));
                model.addAttribute("pageType", PageType.CART.name());

                // EMF - Brainsins
                brainsinsStorefrontService.updateModelForCartPage(model, cartData);
        }
        @SuppressWarnings("boxing")
        protected void prepareDataForPage(final Model model, final HttpServletRequest
request) throws CMSItemNotFoundException
        {
                final String continueUrl = (String)
sessionService.getAttribute(WebConstants.CONTINUE_URL);
                model.addAttribute(CONTINUE_URL, (continueUrl != null &&
!continueUrl.isEmpty()) ? continueUrl : ROOT);


                //(JSG)25/03/2015 - SD184623: Se eliminan posibles productos "NO
DISPONIBLES" antes de cargarlos en el carrito
                final Boolean forceCartValidation = (Boolean)
request.getSession().getAttribute(WebConstants.FORCE_CART_VALIDATION);
                //(PFA)[PERFORMANCE][CLAREL-1173] 20151117 Variable que determina si
tenemos que validar el carrito.
                // El referer es null si venimos de /login y vamos a /cart y esta
constante se mete en LastOrdersPageController y
                // SavedListPageController
                //              if (forceCartValidation != null &&
Boolean.TRUE.equals(forceCartValidation))
                //              {

                //(JSG)25/03/2015 - SD184623: Se eliminan posibles productos "NO
DISPONIBLES" antes de cargarlos en el carrito
                final List<CartModificationData> modifications =
cartFacade.removeProductsWithoutStockOrPrice();

                if (!modifications.isEmpty())
                {
                        model.addAttribute("validationData", modifications);
                }
                request.getSession().removeAttribute(WebConstants.FORCE_CART_VALIDATION);

                //              }



                createProductList(model);

                final CartRestorationData restorationData = (CartRestorationData)
sessionService
                                .getAttribute(WebConstants.CART_RESTORATION);
                model.addAttribute("restorationData", restorationData);
                model.addAttribute("isOmsEnabled", Boolean.valueOf(isOmsEnabled()));
                model.addAttribute("supportedCountries",
cartFacade.getDeliveryCountries());

                // Incidencia DIAEC - 584, 20150721, EMF. Propiedad para el parametro
namePromotion

        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_PROMOTION_PARAM_TYPE_PROMOTION
,

        this.sessionService.getAttribute(WebConstants.GOOGLE_TAG_MANAGER_PROMOTION_PARAM_
TYPE_PROMOTION));

        model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_PROMOTION_PARAM_NAME_PROMOTION
,

        this.sessionService.getAttribute(WebConstants.GOOGLE_TAG_MANAGER_PROMOTION_PARAM_
NAME_PROMOTION));




        }
```

```
	/*
	 * Display the cart page
	 */
	@RequestMapping(method = RequestMethod.GET)
	public String showCart(final Model model, final HttpServletRequest request)
throws CMSItemNotFoundException,
			CommerceCartModificationException
	{
		prepareDataForPage(model, request);


		return ControllerConstants.Views.Pages.Cart.CartPage;
	}
```

11.3.2.5 MultiPaymentCheckoutController

```
package com.hybris.yps.paypal.controllers.pages.checkout;

import de.hybris.platform.acceleratorservices.controllers.page.PageType;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.cms2.model.pages.AbstractPageModel;
import de.hybris.platform.commercefacades.order.OrderFacade;
import de.hybris.platform.commercefacades.order.data.OrderData;
import de.hybris.platform.commercefacades.order.data.OrderEntryData;
import de.hybris.platform.commercefacades.product.ProductFacade;
import de.hybris.platform.commercefacades.product.ProductOption;
import de.hybris.platform.commercefacades.product.data.ProductData;
import de.hybris.platform.commercefacades.user.data.PrincipalData;
import de.hybris.platform.commercefacades.voucher.data.DateRestrictionData;
import de.hybris.platform.commercefacades.voucher.data.OrderRestrictionData;
import de.hybris.platform.commercefacades.voucher.data.PromotionVoucherData;
import de.hybris.platform.commercefacades.voucher.data.UserRestrictionData;
import de.hybris.platform.commercefacades.voucher.data.VoucherData;
import de.hybris.platform.commercefacades.voucher.data.VoucherRestrictionData;
import de.hybris.platform.servicelayer.config.ConfigurationService;
import de.hybris.platform.servicelayer.keygenerator.KeyGenerator;
import de.hybris.platform.servicelayer.session.SessionService;
import de.hybris.platform.yps.payment.multimode.cc.data.CreditCardPaymentInfoData;
import de.hybris.platform.yps.payment.multimode.data.CustomPaymentModeOrderData;
import de.hybris.platform.yps.payment.multimode.paypal.data.PayPalPaymentInfoData;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Date;
import java.util.List;

import javax.annotation.Resource;
import javax.servlet.http.HttpServletRequest;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.collections.ListUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.context.MessageSource;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.b2b2000.dia.constants.DiaacceleratorStorefrontConstants;
import com.b2b2000.dia.core.order.DiaOrderService;
import com.b2b2000.dia.facades.customer.DiaCustomerFacade;
import com.b2b2000.dia.facades.store.LocalStorePreferencesFacade;
import com.b2b2000.dia.facades.user.data.ExtendedCustomerData;
import com.b2b2000.dia.facades.voucher.DiaVoucherFacade;
import com.b2b2000.dia.storefront.annotations.RequireHardLogIn;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.constants.WebConstants;
import com.b2b2000.dia.storefront.controllers.ControllerConstants;
import com.b2b2000.dia.storefront.controllers.pages.checkout.CheckoutController;
import com.b2b2000.dia.storefront.forms.GuestRegisterForm;
```

```
import com.b2b2000.dia.storefront.util.OrderStoreFrontUtil;
import com.hybris.yps.paymentech.mpm.data.PaymentechPaymentInfoData;



/**
 * Extension of standard checkout controller allows to support multi payment mode
processing To change this template use
 * File | Settings | File Templates.
 */
@RequestMapping(value = "/checkout")
public class MultiPaymentCheckoutController extends CheckoutController
{

	protected static final Logger LOG =
Logger.getLogger(MultiPaymentCheckoutController.class);
	/**
	 * We use this suffix pattern because of an issue with Spring 3.1 where a Uri
value is incorrectly extracted if it
	 * contains on or more '.' characters. Please see
https://jira.springsource.org/browse/SPR-6164 for a discussion on
	 * the issue and future resolution.
	 */
	public static final String CHECKOUT_ORDER_CONFIRMATION_CMS_PAGE_LABEL =
"orderConfirmation";

	public static final String ADDON_YPSPAYPALADDON_CONFIRM =
ControllerConstants.Views.Pages.Checkout.CheckoutConfirmationPage;

	public static final String ORDER_CODE_PATH_VARIABLE_PATTERN = "{orderCode:.*}";


	@Resource(name = "productFacade")
	private ProductFacade productFacade;

	@Resource(name = "orderFacade")
	private OrderFacade orderFacade;

	@Resource(name = "voucherFacade")
	private DiaVoucherFacade voucherFacade;

	/**
	 * (EMF) 20150716
	 */
	@Resource(name = "sessionService")
	private SessionService sessionService;

	/**
	 * 20140205 FPR Preferencias de usuario.
	 */
	@Resource(name = "localStorePreferencesFacade")
	private LocalStorePreferencesFacade localStorePreferencesFacade;

	@Resource(name = "extendedOrderService")
	private DiaOrderService extendedOrderService;
	@Resource
	private KeyGenerator diaVoucherCodeGenerator;
	@Resource
	private MessageSource messageSource;

	@Resource
	private ConfigurationService configurationService;

	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	/*
	 * (JAF), 20140410
	 *
	 * @RequestMapping(value = "/orderConfirmation-paypal/" +
ORDER_CODE_PATH_VARIABLE_PATTERN, method =
	 * RequestMethod.GET)
	 */
	@RequestMapping(value = "/orderConfirmation-multi/" +
ORDER_CODE_PATH_VARIABLE_PATTERN, method = RequestMethod.GET)
	@RequireHardLogIn
```

```
	public String orderConfirmationPayPal(@PathVariable("orderCode") final String
orderCode, final HttpServletRequest request,
			final Model model) throws CMSItemNotFoundException
	{

		final ExtendedCustomerData customer = (ExtendedCustomerData)
this.getCustomerFacade().getCurrentCustomer();


		if (customer == null || customer.getUid().contains("anonymous"))
		{

			return REDIRECT_PREFIX + "/cart";
		}

		super.orderConfirmation(orderCode, request, model);

		model.addAttribute("email", customer.getEmail());


		// Incidencia DIAEC-584, 20150709, EMF. Propiedad para el parametro
typePage
		model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_TYPE_PAGE_PARAM_NAME,
				WebConstants.GOOGLE_TAG_MANAGER_CATEGORY_PARAM_CONFIRM);

		/*
		 * Incidencia DIAEC-584, 20150727, EMF. Propiedad para el parametro
paymentMethod
		 */
		final String paymentMethod =
this.getPaymentMethodNameTagParam(orderCode);
		if (paymentMethod != null && !paymentMethod.isEmpty())
		{

	model.addAttribute(WebConstants.GOOGLE_TAG_MANAGER_ORDER_PARAM_PAYMENT_METHOD,
paymentMethod);
		}

		// EMF - Brainsins
		brainsinsStorefrontService.updateModelForOrderConfirmationPage(model);

		/*
		 * Incidencia DIAEC-584, 20150727, EMF. Eliminar de la sesion los
parametros delivery, deliveryPrevious totalcost
		 */
		this.resetDeliveryParam();
		this.resetDeliveryPreviousParam();

		return ADDON_YPSPAYPALADDON_CONFIRM;
	}
```

11.3.2.6 CheckoutController.java

```
* [y] hybris Platform
 *
 * Copyright (c) 2000-2013 hybris AG
 * All rights reserved.
 *
 * This software is the confidential and proprietary information of hybris
 * ("Confidential Information"). You shall not disclose such Confidential
 * Information and shall use it only in accordance with the terms of the
 * license agreement you entered into with hybris.
 *
 *
 */
package com.b2b2000.dia.storefront.controllers.pages.checkout;

import de.hybris.platform.acceleratorservices.controllers.page.PageType;
import de.hybris.platform.cms2.exceptions.CMSItemNotFoundException;
import de.hybris.platform.cms2.model.pages.AbstractPageModel;
import de.hybris.platform.commercefacades.customer.CustomerFacade;
import de.hybris.platform.commercefacades.order.CheckoutFacade;
import de.hybris.platform.commercefacades.order.OrderFacade;
import de.hybris.platform.commercefacades.order.data.OrderData;
```

```
import de.hybris.platform.commercefacades.order.data.OrderEntryData;
import de.hybris.platform.commercefacades.product.ProductFacade;
import de.hybris.platform.commercefacades.product.ProductOption;
import de.hybris.platform.commercefacades.product.data.ProductData;
import de.hybris.platform.commerceservices.customer.DuplicateUidException;
import de.hybris.platform.servicelayer.exceptions.ModelNotFoundException;
import de.hybris.platform.yps.payment.multimode.data.CustomPaymentModeOrderData;

import java.util.Arrays;

import javax.annotation.Resource;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.b2b2000.dia.facades.flow.impl.SessionOverrideCheckoutFlowFacade;
import com.b2b2000.dia.facades.user.data.ExtendedCustomerData;
import com.b2b2000.dia.storefront.annotations.RequireHardLogIn;
import
com.b2b2000.dia.storefront.brainsins.services.impl.ExtendedBrainsinsStorefrontService;
import com.b2b2000.dia.storefront.constants.WebConstants;
import com.b2b2000.dia.storefront.controllers.ControllerConstants;
import com.b2b2000.dia.storefront.controllers.util.GlobalMessages;
import com.b2b2000.dia.storefront.forms.GuestRegisterForm;
import com.b2b2000.dia.storefront.forms.validation.GuestRegisterValidator;
import com.b2b2000.dia.storefront.security.AutoLoginStrategy;


/**
 * CheckoutController
 */
@Controller
@Scope("tenant")
@RequestMapping(value = "/checkout")
public class CheckoutController extends AbstractCheckoutController
{
	protected static final Logger LOG = Logger.getLogger(CheckoutController.class);
	/**
	 * We use this suffix pattern because of an issue with Spring 3.1 where a Uri
value is incorrectly extracted if it
	 * contains on or more '.' characters. Please see
https://jira.springsource.org/browse/SPR-6164 for a discussion on
	 * the issue and future resolution.
	 */
	private static final String ORDER_CODE_PATH_VARIABLE_PATTERN = "{orderCode:.*}";

	private static final String CHECKOUT_ORDER_CONFIRMATION_CMS_PAGE_LABEL =
"orderConfirmation";

	private static final Logger LOGGER = Logger.getLogger(CheckoutController.class);

	@Resource(name = "productFacade")
	private ProductFacade productFacade;

	@Resource(name = "orderFacade")
	private OrderFacade orderFacade;

	@Resource(name = "checkoutFacade")
	private CheckoutFacade checkoutFacade;

	@Resource(name = "guestRegisterValidator")
	private GuestRegisterValidator guestRegisterValidator;

	@Resource(name = "autoLoginStrategy")
	private AutoLoginStrategy autoLoginStrategy;

	@Resource(name = "customerFacade")
```

```
	private CustomerFacade customerFacade;

	@Resource(name = "brainsinsStorefrontService")
	private ExtendedBrainsinsStorefrontService brainsinsStorefrontService;

	protected void processOrderCode(final String orderCode, final Model model, final
HttpServletRequest request)
			throws CMSItemNotFoundException
	{
		final OrderData orderDetails =
orderFacade.getOrderDetailsForCode(orderCode);

		if (orderDetails.getEntries() != null &&
!orderDetails.getEntries().isEmpty())
		{
			for (final OrderEntryData entry : orderDetails.getEntries())
			{
				final String productCode = entry.getProduct().getCode();
				final ProductData product =
productFacade.getProductForCodeAndOptions(productCode,
					Arrays.asList(ProductOption.BASIC,
ProductOption.PRICE, ProductOption.CATEGORIES));
				entry.setProduct(product);
			}
		}

		model.addAttribute("orderCode", orderCode);
		model.addAttribute("orderData", orderDetails);
		model.addAttribute("allItems", orderDetails.getEntries());
		model.addAttribute("deliveryAddress", orderDetails.getDeliveryAddress());
		model.addAttribute("deliveryMode", orderDetails.getDeliveryMode());
		//SZF, 20150928, [DIAEC-993] Añadir la fecha de entrega formateada
		model.addAttribute(
				"deliveryTimeInterval",
				fromModelToSummary(((CustomPaymentModeOrderData)
orderDetails).getDeliveryTimeIntervalStart(),
					((CustomPaymentModeOrderData)
orderDetails).getDeliveryTimeIntervalEnd()));
		model.addAttribute("paymentInfo", orderDetails.getPaymentInfo());
		model.addAttribute("pageType", PageType.ORDERCONFIRMATION.name());
		//		final String uid =
orderDetails.getPaymentInfo().getBillingAddress().getEmail();
		//		model.addAttribute("email", uid);

		//20140414 FPR Ahora se extrae el email correcto populado a partir del
servicio de resolución de emails.
		final ExtendedCustomerData customer = (ExtendedCustomerData)
this.getCustomerFacade().getCurrentCustomer();

		if (customer != null)
		{
			model.addAttribute("email", customer.getEmail());
		}


		if (orderDetails.isGuestCustomer() &&
!model.containsAttribute("guestRegisterForm"))
		{
			final GuestRegisterForm guestRegisterForm = new
GuestRegisterForm();
			guestRegisterForm.setOrderCode(orderDetails.getGuid());

			//guestRegisterForm.setUid(uid);
			model.addAttribute(guestRegisterForm);
		}

		final AbstractPageModel cmsPage =
getContentPageForLabelOrId(CHECKOUT_ORDER_CONFIRMATION_CMS_PAGE_LABEL);
		storeCmsPageInModel(model, cmsPage);
		setUpMetaDataForContentPage(model,
getContentPageForLabelOrId(CHECKOUT_ORDER_CONFIRMATION_CMS_PAGE_LABEL));

		brainsinsStorefrontService.updateModelForOrderConfirmationPage(model);

		/*
		 * (JAF), 20140322
		 *
```

```
                * <quote> 1.5. Listado
                *
                * Una fuente de errores de contenido duplicado son los filtros y
distintas ordenaciones que se aplican a los
                * listados. Aunque se ha incluido una meta etiqueta robots para evitar
que estos sean indexados, su contenido
                * están mal generado ya que en vez de:
                *
                * <meta name="robots" content="no-index,follow" />
                *
                * La metaetiqueta debería ser:
                *
                * <meta name="robots" content="noindex,follow" /> </quote>
                *
                * model.addAttribute("metaRobots", "no-index,no-follow");
                */
               model.addAttribute("metaRobots", "noindex,nofollow");

        }
        @RequestMapping(value = "/orderConfirmation/" + ORDER_CODE_PATH_VARIABLE_PATTERN,
method = RequestMethod.GET)
        @RequireHardLogIn
        public String orderConfirmation(@PathVariable("orderCode") final String
orderCode, final HttpServletRequest request,
                        final Model model) throws CMSItemNotFoundException
        {
                SessionOverrideCheckoutFlowFacade.resetSessionOverrides();
                processOrderCode(orderCode, model, request);
                //PFA: Vouchers. Justo antes de retornar, comprobamos si

                return ControllerConstants.Views.Pages.Checkout.CheckoutConfirmationPage;
        }
```

11.3.2.7 web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5"
        metadata-complete="true"
        xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
                http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

        <display-name>diaacceleratorstorefront</display-name>
<!--    <display-name>compra-online</display-name> -->

        <!--
         We have to explicitly clear the welcome file list.
         We don't need to serve a default or index page as we can handle all the requests
via spring MVC.
        -->
        <welcome-file-list>
                <welcome-file/>
        </welcome-file-list>
        <!-- (EMF), 20151021  changed 20151125 from "/brainsins/products.xml" to
"/brainsins/*" -->
        <filter-mapping><filter-name>resourceFilter</filter-name><url-
pattern>/brainsins/*</url-pattern></filter-mapping>
```

11.3.2.8 BrainsinsStorefrontService.java

```
/**
 *
 */
package com.b2b2000.dia.storefront.brainsins.services;

import de.hybris.platform.commercefacades.order.data.CartData;

import org.springframework.ui.Model;
```

```
/**
 * Se trata de la interfaz del servicio para la integración de BrainSINS en el sitio web
de DIA
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public interface BrainsinsStorefrontService
{

	/**
	 * Se encarga de actualizar el model, que recibe como parámetro, con la
información de BrainSINS en la página por
	 * defecto. Normalmente será la página home, o bien otra que no encaje con
ninguno de los tipos especificados por
	 * brainsins, lo que permitirá incluir todos los recomenadores en cualquier
página del sitio web.
	 *
	 * @param model
	 */
	public void updateModelForDefaultPage(final Model model);

	/**
	 * Se encarga de actualizar el model, que recibe como parámetro, con la
información de BrainSINS en la página de
	 * detalle de producto
	 *
	 * @param model
	 * @param productCode
	 */
	public void updateModelForProductPage(final Model model, final String
productCode);

	/**
	 * Se encarga de actualizar el model, que reciben como parámetro, con la
información de BrainSINS en la página de
	 * categorías.
	 *
	 * @param model
	 * @param categoryCode
	 */
	public void updateModelForCategoryPage(final Model model, final String
categoryCode);

	/**
	 * Se encarga de incluir en el model que recibe como parámetro la información de
BrainSINS asociada a la página del
	 * carrito del usuario.
	 *
	 * @param model
	 * @param cartData
	 */
	public void updateModelForCartPage(final Model model, final CartData cartData);

	/**
	 * Se encarga de incluir en el model que reciben como parámetro la informaición
de BrainSINS asociada a la página de
	 * confirmación del pedido.
	 *
	 * @param model
	 */
	public void updateModelForOrderConfirmationPage(final Model model);

}
```

11.3.2.9 ExtendedBrainsinsStorefrontService.java

```
/**
 *
 */
package com.b2b2000.dia.storefront.brainsins.services.impl;

import de.hybris.platform.commercefacades.order.data.CartData;
```

```
import de.hybris.platform.servicelayer.session.SessionService;

import javax.annotation.Resource;

import org.apache.log4j.Logger;
import org.springframework.ui.Model;

import com.b2b2000.dia.facades.customer.DiaCustomerFacade;
import com.b2b2000.dia.storefront.brainsins.constants.BrainsinsStorefrontConstants;
import com.b2b2000.dia.storefront.brainsins.data.AuxBrainsinsData;
import com.b2b2000.dia.storefront.brainsins.services.BrainsinsStorefrontService;


/**
 * Se trata de la implementación de la interfaz BrainsinsStorefrontService, dedicado a
la integración de BrainSINS en el
 * sitio web de DIA
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class ExtendedBrainsinsStorefrontService implements BrainsinsStorefrontService
{
	private final Logger LOG =
Logger.getLogger(ExtendedBrainsinsStorefrontService.class);

	/**
	 * Servicio par la obtención de la sesion del ususario
	 */
	@Resource(name = "sessionService")
	private SessionService sessionService;

	/**
	 * Servicio de utilidad para el servicio de brainsins
	 */
	@Resource(name = "brainsinsStorefrontUtilService")
	private ExtendedBrainsinsStorefrontUtilService brainsinsStorefrontUtilService;

	/**
	 * Fachada del cliente de dia
	 */
	@Resource(name = "customerFacade")
	private DiaCustomerFacade customerFacade;


	/**
	 * Genera la información por defecto de BrainSINS que es necesario incluir en
todas las paginas
	 *
	 * @return
	 */
	private AuxBrainsinsData getAuxBrainsinsDataCommonForPages()
	{
		final AuxBrainsinsData brData = new AuxBrainsinsData();


	brData.setBrainsinsUnavailableProductText(getBrainsinsStorefrontUtilService().get
UnavailableButtonText());

	brData.setBrainsinsAddButtonText(getBrainsinsStorefrontUtilService().getAddButton
Text());
		brData.setToken(getBrainsinsStorefrontUtilService().getToken());

	brData.setLanguage(getCustomerFacade().getCurrentWarehouseCodeForCurrentCustomer(
));

	brData.setUserEmail(getCustomerFacade().getVerifiedContactEmailFromCurrentCustome
r());

		return brData;
	}

	/**
	 *
	 */
	@Override
```

```
	public void updateModelForDefaultPage(final Model model)
	{
		final AuxBrainsinsData brData = this.getAuxBrainsinsDataCommonForPages();

		brData.setTypePage(BrainsinsStorefrontConstants.TypePage.defaultPage);


	model.addAttribute(BrainsinsStorefrontConstants.ModelKey.auxBrainsinsDataParamete
r, brData);
	}

	/**
	 *
	 */
	@Override
	public void updateModelForProductPage(final Model model, final String
productCode)
	{
		final AuxBrainsinsData brData = this.getAuxBrainsinsDataCommonForPages();

		brData.setProductId(productCode);
		brData.setTypePage(BrainsinsStorefrontConstants.TypePage.productPage);


	model.addAttribute(BrainsinsStorefrontConstants.ModelKey.auxBrainsinsDataParamete
r, brData);

	}

	/**
	 *
	 */
	@Override
	public void updateModelForCategoryPage(final Model model, final String
categoryName)
	{
		final AuxBrainsinsData brData = this.getAuxBrainsinsDataCommonForPages();

		brData.setTypePage(BrainsinsStorefrontConstants.TypePage.categoryPage);
		brData.setCategories(categoryName);


	model.addAttribute(BrainsinsStorefrontConstants.ModelKey.auxBrainsinsDataParamete
r, brData);

	}

	/**
	 *
	 */
	@Override
	public void updateModelForCartPage(final Model model, final CartData cartData)
	{

		if (cartData == null)
		{
			LOG.error(this.getClass().getName() + " CartData is null ");
			return;
		}

		final AuxBrainsinsData brData = this.getAuxBrainsinsDataCommonForPages();

		brData.setTypePage(BrainsinsStorefrontConstants.TypePage.cartPage);

	brData.setCart(getBrainsinsStorefrontUtilService().getBrainsinsCart(cartData.getE
ntries()));


	getSessionService().setAttribute(BrainsinsStorefrontConstants.SessionAttributes.t
otalPrice,

	getBrainsinsStorefrontUtilService().getTotalAmount(cartData.getTotalPrice()));


	model.addAttribute(BrainsinsStorefrontConstants.ModelKey.auxBrainsinsDataParamete
r, brData);
```

```
	}


	/**
	 *
	 */
	@Override
	public void updateModelForOrderConfirmationPage(final Model model)
	{
		final AuxBrainsinsData brData = this.getAuxBrainsinsDataCommonForPages();


	brData.setTypePage(BrainsinsStorefrontConstants.TypePage.orderConfirmationPage);
		brData.setTotalAmount((Double)
getSessionService().getAttribute(BrainsinsStorefrontConstants.SessionAttributes.totalPri
ce));


	model.addAttribute(BrainsinsStorefrontConstants.ModelKey.auxBrainsinsDataParamete
r, brData);

	}

	/**
	 * @return the brainsinsStorefrontUtilService
	 */
	public ExtendedBrainsinsStorefrontUtilService getBrainsinsStorefrontUtilService()
	{
		return brainsinsStorefrontUtilService;
	}

	/**
	 * @param brainsinsStorefrontUtilService
	 *           the brainsinsStorefrontUtilService to set
	 */
	public void setBrainsinsStorefrontUtilService(final
ExtendedBrainsinsStorefrontUtilService brainsinsStorefrontUtilService)
	{
		this.brainsinsStorefrontUtilService = brainsinsStorefrontUtilService;
	}

	/**
	 * @return the customerFacade
	 */
	public DiaCustomerFacade getCustomerFacade()
	{
		return customerFacade;
	}

	/**
	 * @param customerFacade
	 *           the customerFacade to set
	 */
	public void setCustomerFacade(final DiaCustomerFacade customerFacade)
	{
		this.customerFacade = customerFacade;
	}

	/**
	 * @return the sessionService
	 */
	public SessionService getSessionService()
	{
		return sessionService;
	}

	/**
	 * @param sessionService
	 *           the sessionService to set
	 */
	public void setSessionService(final SessionService sessionService)
	{
		this.sessionService = sessionService;
	}
```

```
}
```

11.3.2.10 BrainsinsStorefrontUtilService.java

```
/**
 *
 */
package com.b2b2000.dia.storefront.brainsins.services;

import de.hybris.platform.commercefacades.order.data.OrderEntryData;
import de.hybris.platform.commercefacades.product.data.PriceData;

import java.util.HashMap;
import java.util.List;


/**
 * Se trata de la interfaz de la clase de utilidad para el servicio de Brainsins.
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public interface BrainsinsStorefrontUtilService
{

	/**
	 * Devuelve el token que Brainsins ha asignado a DIA y que está asociado al
entorno donde se encuentra desplegada la
	 * aplicación (local, dev, stag, pro)
	 *
	 * @return
	 */
	public String getToken();

	/**
	 * Devuelve el texto internacionalizado del botón 'Anyadir'.
	 *
	 * @return
	 */
	public String getAddButtonText();

	/**
	 * Devuelve el texto internacionalizado del botón 'No disponible'
	 *
	 * @return
	 */
	public String getUnavailableButtonText();

	/**
	 * Devuelve el nombre de la categoría cuyo código es enviado como parámetro
	 *
	 * @param categoryCode
	 * @return
	 */
	public String getCategories(String categoryCode);

	/**
	 * Devuelve un mapa con el listado de los códigos de los productos incluidos en
el carrito del usuario, junto a las
	 * unidades añadidas
	 *
	 * @param cartEntries
	 * @return
	 */
	public HashMap<String, Long> getBrainsinsCart(final List<OrderEntryData>
cartEntries);

	/**
	 * Devuelve el precio total del pedido del usuario a partir del parámetro
priceData que recibe como parámetro
	 *
	 * @param priceData
	 * @return
```

```
         */
        public Double getTotalAmount(final PriceData priceData);
}
```

11.3.2.11 ExtendedBrainsinsStorefrontUtilService.java

```
/**
 *
 */
package com.b2b2000.dia.storefront.brainsins.services.impl;

import de.hybris.platform.category.CategoryService;
import de.hybris.platform.category.model.CategoryModel;
import de.hybris.platform.commercefacades.order.data.OrderEntryData;
import de.hybris.platform.commercefacades.product.data.PriceData;
import de.hybris.platform.servicelayer.config.ConfigurationService;
import de.hybris.platform.servicelayer.i18n.I18NService;

import java.util.HashMap;
import java.util.List;

import javax.annotation.Resource;

import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.context.MessageSource;
import org.springframework.context.NoSuchMessageException;

import com.b2b2000.dia.storefront.brainsins.constants.BrainsinsStorefrontConstants;
import com.b2b2000.dia.storefront.brainsins.services.BrainsinsStorefrontUtilService;


/**
 * Se trata de la implementación de la interfaz del servicio de utilidad Brainsins.
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class ExtendedBrainsinsStorefrontUtilService implements
BrainsinsStorefrontUtilService
{

        private final Logger LOG =
Logger.getLogger(ExtendedBrainsinsStorefrontUtilService.class);

        /**
         * Servicio de configuracion Hybris
         */
        @Resource(name = "configurationService")
        private ConfigurationService configurationService;

        /**
         * Servicio Hybris de mensajes
         */
        @Resource(name = "messageSource")
        private MessageSource messageSource;

        /**
         * Servicio de internacionalizacion
         */
        @Resource(name = "i18NService")
        private I18NService i18NService;

        @Resource(name = "categoryService")
        private CategoryService categoryService; // no



        @Override
        public String getToken()
        {
                String token = null;
```

```
		try
		{
			token =
getConfigurationService().getConfiguration().getString(BrainsinsStorefrontConstants.Toke
n.property);

			if (StringUtils.isBlank(token))
			{
				LOG.error("Brainsins Token: " +
BrainsinsStorefrontConstants.Token.property + " is not defined in local.properties");
			}
		}
		catch (final Exception e)
		{
			LOG.error(e.getMessage());
		}

		return (token != null ? token : StringUtils.EMPTY);

	}

	@Override
	public String getAddButtonText()
	{
		String text = null;

		try
		{

			text =
messageSource.getMessage(BrainsinsStorefrontConstants.CartIntegration.addToCartMessage,
new Object[] {},
					getI18NService().getCurrentLocale());

		}
		catch (final NoSuchMessageException e)
		{
			if (LOG.isDebugEnabled())
			{
				LOG.debug(e.getMessage());
			}
		}

		return (text != null ? text : "A\u00F1adir");
	}

	@Override
	public String getUnavailableButtonText()
	{

		String unavailable = null;

		try
		{
			unavailable =
getMessageSource().getMessage(BrainsinsStorefrontConstants.CartIntegration.unavailableMe
ssage,
					new Object[] {},
getI18NService().getCurrentLocale());
		}
		catch (final NoSuchMessageException e)
		{
			if (LOG.isDebugEnabled())
			{
				LOG.debug(e.getMessage());
			}

		}

		return (unavailable != null ? unavailable : "No disponible");
	}


	@Override
	public String getCategories(final String categoryCode)
	{
		if (categoryCode == null)
```

```
		{
			return StringUtils.EMPTY;
		}

		final CategoryModel categoryModel =
getCategoryService().getCategoryForCode(categoryCode);

		return ((categoryModel != null && categoryModel.getName() != null) ?
categoryModel.getName() : StringUtils.EMPTY);

	}

	@Override
	public HashMap<String, Long> getBrainsinsCart(final List<OrderEntryData>
cartEntries)
	{

		final HashMap<String, Long> cart = new HashMap<String, Long>();

		if (cartEntries != null)
		{
			for (final OrderEntryData entry : cartEntries)
			{
				if (entry != null && entry.getProduct() != null &&
entry.getQuantity() != null)
				{
					cart.put(entry.getProduct().getCode(),
entry.getQuantity());
				}
			}
		}

		return cart;
	}

	@Override
	public Double getTotalAmount(final PriceData priceData)
	{
		Double total = null;

		if (priceData == null || priceData.getValue() == null)
		{
			LOG.error(this.getClass().getName() + " PriceData is null");
		}
		else
		{
			total = priceData.getValue().doubleValue();
		}

		return total;
	}


	/**
	 * @return the configurationService
	 */
	public ConfigurationService getConfigurationService()
	{
		return configurationService;
	}

	/**
	 * @param configurationService
	 *           the configurationService to set
	 */
	public void setConfigurationService(final ConfigurationService
configurationService)
	{
		this.configurationService = configurationService;
	}


	/**
	 * @return the messageSource
	 */
	public MessageSource getMessageSource()
	{
```

```
		return messageSource;
	}

	/**
	 * @param messageSource
	 *            the messageSource to set
	 */
	public void setMessageSource(final MessageSource messageSource)
	{
		this.messageSource = messageSource;
	}

	/**
	 * @return the i18NService
	 */
	public I18NService getI18NService()
	{
		return i18NService;
	}

	/**
	 * @param i18nService
	 *            the i18NService to set
	 */
	public void setI18NService(final I18NService i18nService)
	{
		i18NService = i18nService;
	}

	/**
	 * @return the categoryService
	 */
	public CategoryService getCategoryService()
	{
		return categoryService;
	}

	/**
	 * @param categoryService
	 *            the categoryService to set
	 */
	public void setCategoryService(final CategoryService categoryService)
	{
		this.categoryService = categoryService;
	}


}
```

11.3.2.12 AuxBrainsinsData.java

```
/**
 *
 */
package com.b2b2000.dia.storefront.brainsins.data;

import java.util.HashMap;




/**
 * Encapsula el conjunto de los datos que es necesario enviar a Brainsins en función de
la página en la que se encuentra
 * el usuario. Se añade al model de las distintas páginas del site donde es necesario
llevar a cabo la integración con
 * BrainSINS a través del servicio BrainsinsStorefrontService.
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public class AuxBrainsinsData
{
```

```
	/**
	 * Token que BrainSINS ha asignado a DIA
	 */
	public String token;

	/**
	 * Código de la tienda en la que se encuentra el usuario
	 */
	public String language;

	/**
	 * Email del usuario que ha iniciado sesión el el sitio web
	 */
	public String userEmail;

	/**
	 * Codigo identificador de un producto
	 */
	public String productId;

	/**
	 * Coste total del pedido de un cliente
	 */
	public Double totalAmount;

	/**
	 * Tipo de pagina
	 */
	public String typePage;

	/**
	 * Nombres de las categorias internacionalizadas
	 */
	public String categories;

	/**
	 * Carrito del usuario
	 */
	public HashMap<String, Long> cart;

	/**
	 * Texto internacionalizado del boton 'Anyadir'
	 */
	public String brainsinsAddButtonText;

	/**
	 * Texto internacionalizado del boton 'No disponible'
	 */
	public String brainsinsUnavailableProductText;

	/**
	 * @return the token
	 */
	public String getToken()
	{
		return token;
	}

	/**
	 * @param token
	 *            the token to set
	 */
	public void setToken(final String token)
	{
		this.token = token;
	}

	/**
	 * @return the language
	 */
	public String getLanguage()
	{
		return language;
	}

	/**
	 * @param language
```

```
	 *            the language to set
	 */
	public void setLanguage(final String language)
	{
		this.language = language;
	}

	/**
	 * @return the userEmail
	 */
	public String getUserEmail()
	{
		return userEmail;
	}

	/**
	 * @param userEmail
	 *            the userEmail to set
	 */
	public void setUserEmail(final String userEmail)
	{
		this.userEmail = userEmail;
	}


	/**
	 * @return the productId
	 */
	public String getProductId()
	{
		return productId;
	}

	/**
	 * @param productId
	 *            the productId to set
	 */
	public void setProductId(final String productId)
	{
		this.productId = productId;
	}

	/**
	 * @return the totalAmount
	 */
	public Double getTotalAmount()
	{
		return totalAmount;
	}

	/**
	 * @param totalAmount
	 *            the totalAmount to set
	 */
	public void setTotalAmount(final Double totalAmount)
	{
		this.totalAmount = totalAmount;
	}

	/**
	 * @return the typePage
	 */
	public String getTypePage()
	{
		return typePage;
	}

	/**
	 * @param typePage
	 *            the typePage to set
	 */
	public void setTypePage(final String typePage)
	{
		this.typePage = typePage;
	}

	/**
```

```
	 * @return the categories
	 */
	public String getCategories()
	{
		return categories;
	}

	/**
	 * @param categories
	 *            the categories to set
	 */
	public void setCategories(final String categories)
	{
		this.categories = categories;
	}

	/**
	 * @return the brainsinsAddButtonText
	 */
	public String getBrainsinsAddButtonText()
	{
		return brainsinsAddButtonText;
	}

	/**
	 * @param brainsinsAddButtonText
	 *            the brainsinsAddButtonText to set
	 */
	public void setBrainsinsAddButtonText(final String brainsinsAddButtonText)
	{
		this.brainsinsAddButtonText = brainsinsAddButtonText;
	}

	/**
	 * @return the brainsinsUnavailableProductText
	 */
	public String getBrainsinsUnavailableProductText()
	{
		return brainsinsUnavailableProductText;
	}

	/**
	 * @param brainsinsUnavailableProductText
	 *            the brainsinsUnavailableProductText to set
	 */
	public void setBrainsinsUnavailableProductText(final String
brainsinsUnavailableProductText)
	{
		this.brainsinsUnavailableProductText = brainsinsUnavailableProductText;
	}

	/**
	 * @return the cart
	 */
	public HashMap<String, Long> getCart()
	{
		return cart;
	}

	/**
	 * @param cart
	 *            the cart to set
	 */
	public void setCart(final HashMap<String, Long> cart)
	{
		this.cart = cart;
	}


}
```

11.3.2.13 BrainsinsStorefrontConstants

```
/**
```

```
 *
 */
package com.b2b2000.dia.storefront.brainsins.constants;

/**
 * Se trata de la interfaz donde se incluyen las constantes para la clase
AuxBrainsinsData
 *
 * @author <a href="mailto:estefania.menendez@ricoh.es>Estefania Menendez Freije
(EMF)</a>
 *
 */
public interface BrainsinsStorefrontConstants
{
	public interface ModelKey
	{
		/**
		 * Identificador del parámetro AuxBrainsinsData en el model de la página
		 */
		String auxBrainsinsDataParameter = "AuxBrainsinsData";
	}

	public interface SessionAttributes
	{
		/**
		 * Identificador del parámetro donde se almacena el precio total del
pedido
		 */
		String totalPrice = "brainsinsTotalPrice";
	}

	public interface Token
	{
		/**
		 * Identificador de la property donde se incluye el valor del token de
BrainSINS para DIA
		 */
		String property = "brainsins.token.dia";
	}

	public interface CartIntegration
	{
		/**
		 * Identificador de la property donde se define el texto del boton
Anyadir
		 */
		String unavailableMessage =
"text.account.order.consignment.trackingID.notavailable";

		/**
		 * Identificador de la property donde se define el texto del boton No
disponible
		 */
		String addToCartMessage = "product.addCart.Form";
	}

	public interface TypePage
	{
		/**
		 * Identificador de la pagina home segun la especificacion de BrainSINS
		 */
		String defaultPage = "home";

		/**
		 * Identificador de la pagina detalle de producto segun la especificacion
de BrainSINS
		 */
		String productPage = "product";

		/**
		 * Identificador de la pagina del carrito segun la especificacion de
BrainSINS
		 */
		String cartPage = "checkout";

		/**
```

```
                * Identificador de la pagina de confirmacion del pedido segun la
especificacion de BrainSINS
                */
               String orderConfirmationPage = "thankYou";

               /**
                * Identificador de la pagina de categorias segun la especificacion de
BrainSINS
                */
               String categoryPage = "category";

       }

}
```

11.3.2.14 brainsins.tag

```
<%-- EMF 20151016  [DIAEC-464] - Brainsins --%>
<%@ tag body-content="empty" trimDirectiveWhitespaces="true" %>
<%@ taglib prefix="brainsins" tagdir="/WEB-INF/tags/shared/brainsins" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%--
       AuxBrainsinsData se utiliza para almacenar aquellos parámetros
       necesarios para comunicarnos con Brainsins

       Estos parámetro serán utilizados desde el javascript acc.brainsins.js

 --%>
<script type="text/javascript">

       var AuxBrainsinsData = {};

       AuxBrainsinsData.token = "${AuxBrainsinsData.token}";
       AuxBrainsinsData.language = "${AuxBrainsinsData.language}";
       AuxBrainsinsData.userEmail = "${AuxBrainsinsData.userEmail}";
       AuxBrainsinsData.productId = "${AuxBrainsinsData.productId}";
       AuxBrainsinsData.totalAmount = "${AuxBrainsinsData.totalAmount}";
       AuxBrainsinsData.typePage = "${AuxBrainsinsData.typePage}";
       AuxBrainsinsData.categories = "${AuxBrainsinsData.categories}";

       AuxBrainsinsData.brainsinsAddButtonText =
"${AuxBrainsinsData.brainsinsAddButtonText}";
       AuxBrainsinsData.brainsinsUnavailableProductText =
"${AuxBrainsinsData.brainsinsUnavailableProductText}";

       var cartResult = [];

</script>

<c:forEach items="${AuxBrainsinsData.cart}" var="brainsinsCart" >
<script type="text/javascript">
       cart = {};
       cart['id'] = "${brainsinsCart.key}";
       cart['quantity'] = "${brainsinsCart.value}";
       cartResult.push(cart);
       </script>
</c:forEach>
<script type="text/javascript">
       AuxBrainsinsData.cart = cartResult;
</script>
```

11.3.2.15 acc.brainsins.js

```
/*
 * EMF  [DIAEC-464] - [New Feature] INTEGRACIÓN CON BRAINSINS
 *
 * Codigo Javascript para la integración con Brainsins, donde ademas se incluye
 * el boton de añadir al carrito , + , -
 *
 */
```

```
/*
 * Esta variable almacenará los parámetros necesario para solicitar a Brainsins
 * los productos recomendados.
 *
 * Los parámetros variarán en función de la página en la que se encuentre el cliente
 */
var BrainSINSData = { };

/*
 * Parámetros necesarios de incluir en el formulario encargado de añadir los productos
 * recomenados de Brainsins al carrito.
 *
 * Se trata de simular el formulario de añadir al carrito incluido en el site. De esta
 * forma se ejecutará el javascript , acc.cartpopup.js, encargado de llevar a cabo dicha
 * función de forma "transparente" desde el producto recomendado de Brainsins.
 */
var BraininsAddToCartFormElements = {
    'productForm' : 'brainsins_updateCartFormByProduct_',
    'minusButton' : 'BrainsinsQuantityProductMinusByProduct_',
    'quantityInput' : 'BrainsinsQuantityByProduct_',
    'plusButton' : 'BrainsinsDeleteQuantityByProduct_',
    'deleteQuantity' : 'BrainsinsDeleteQuantityByProduct_',
    'addQuantity' : 'BrainsinsAddQuantityByProduct_',
    'price' : 'price_brainsins_',
    'starStyle' : 'estrellita_br_style_',
  };

ACC.brainsins = {

  /*
   * Incluye el token y la librería de Brainsins
   */
  includeScripts: function(){
    var script = document.createElement('script');
    var headTag = $('head')[0];
    script.type= 'text/javascript';
    script.text = "brainsins_token = \"" + AuxBrainsinsData.token +"\";
brainsins_api_mode=\"B\";";
    headTag.appendChild(script);
    var script2 = document.createElement('script');
    script2.type = 'text/javascript';
    script2.async = ' ';
    script2.src = '//d2xkqxdy6ewr93.cloudfront.net/brainsins.js';
    headTag.appendChild(script2);
  },

  /*
   * Recupera el identificador del recommendador a partir del attributo data-name del
div
   *
   * Convenio de nombrado : name="recommender-idRecomendadorHerramientaBrainsins"
   *
   */
  recommenderNumberByDataName:function(dataName){

          var result = null;

          if(dataName != undefined && dataName.length > 0){
                  var splited = dataName.split("-");
                  if(splited.length > 0){
                          result = dataName[1];
                  }
              result = dataName.split("-")[1];
            }

          return result;
  },


  /*
   * Retorna la lista de identificadores de Brainsins a mostar
   * La lista será vacía en el caso de que la página no incluya elementos dedicados a
incluir
   * nigún recomendador de brainsins
   *
   * Para que se incluya un recomendador en la página, ésta debe poseer un div cuyo id
debe
```

```
   * seguir el convenio de nombrado: id='brainsins-recommender-xx' donde xx es cualquier
   * valor que se desee
   *
   * Utiliza el parámetro recommenderDIVS que contiene el conjunto de elementos div
donde
   * se ubicarán los recomendadores.
   */
  getRecommenders: function(recommenderDIVS){

         var recomenders = [];

         if(recommenderDIVS != null && recommenderDIVS != undefined &&
recommenderDIVS.length > 0){

                $.each( recommenderDIVS, function( key, value ) {

                       var recomenderItem = {};

                       var recomDIV = value.getAttribute('id');
                       var recomID =
ACC.brainsins.recommenderNumberByDataName(value.getAttribute('data-name'));

                         recomenderItem.recommenderId = recomID;
                         recomenderItem.location= recomDIV;
                         recomenderItem.position="replace";
                         recomenders.push(recomenderItem);
                });
         }

           return recomenders;
         },


  /*
     * FUNCIONES PETICIONES BRAINSINS
     *
     * A continuación, se añade la función encargada de llevar a cabo el envio de los
datos necesarios
     * a Brainsins, así como de generar la petición del recomendador, en el caso de que
exista en la
     * página el div encargado de incluir el recomendador.
     *
     * Algunos atributos son comunes a todas las páginas, con independencia del tipo de
página
     * donde se esté llevando a cabo el paso de información. Son los parámetros:
     *
     *  - typePage: el tipo de página que recuperamos de BrainSINSModel.data.typePage
     *
     *  - language: el identificador de la tienda donde se encuentra ubicado el usuario
     *
     *  - userEmail: el email del usuario que se encuentra en la página
     *
     *
     *  A continuación, se indican los parámetros necesarios de inlcuir en función de la
página en que se encuentre
     *  el usuario:
     *
     *   - Página del carrito - para Brainsins 'checkout'
     *                - Parmétro cart - carrito del usuario
     *
     *   - Página de detalle del producto - para Brainsins 'product'
     *                - Parámetro productId - identificador del producto
     *
     *   - Página de confirmación de pedido - para Brainsins 'thankYou'
     *                - Parámetro totalAmount - total de la compra
     *
     *
     * Para llevar a cabo la petición del recomendador, se necesitan conocer los datos,
que se indican
     * a continuación:
     *
     *  - El identificador del div donde se incluirá el recomendador
     *
     *  - El Identificador numérico del recomendador o recomendadores que se deseen
mostrar en la página.
     *
     */
```

```
  /*
   * Se encarga de llamar a la función adecuada, según el parametro typePage que
   * recibe la función
   *
   * El parámetro recommenderDIVS incluye un listado de los componentes div
   * donde se desean incluir los recomendadores dentro de la página.
   */
    prepareBrainsinsData: function(typePage, recommenderDIVS){

                switch(typePage){

                case 'home':
                       ACC.brainsins.homePage(recommenderDIVS);
                       break;

                case 'product':
                       ACC.brainsins.productDetailPage(recommenderDIVS);
                       break;

                case 'checkout':
                       ACC.brainsins.cartPage(recommenderDIVS);
                       break;

                case 'thankYou':
                       ACC.brainsins.checkoutConfirmationPage(recommenderDIVS);
                       break;

                case 'category':
                       ACC.brainsins.categoryPage(recommenderDIVS);
                       break;
                }
               return true;
           },

  /*
   * Dedicado a la página home
   *
   * Envia a Brainsins los parámetros asociados al tipo de pagina 'homePage'
   * Incluye el recomendador en el div del HTML
   */
    homePage: function(recommenderDIVS){

                BrainSINSData.pageType = AuxBrainsinsData.typePage;
        BrainSINSData.language= AuxBrainsinsData.language;
        BrainSINSData.userEmail = AuxBrainsinsData.userEmail;

        if(ACC.brainsins.getRecommenders(recommenderDIVS).length > 0){
                BrainSINSData.recommenders =
ACC.brainsins.getRecommenders(recommenderDIVS);
        }

       return true;
     },

  /*
   * Dedicado a la página del detalle de producto
   *
   * Envia a Brainsins los parámetros asociados al tipo de pagina 'product'
   * Incluye el recomendador en el div del HTML
   */
  productDetailPage: function(recommenderDIVS){

          BrainSINSData.pageType = AuxBrainsinsData.typePage;
      BrainSINSData.language= AuxBrainsinsData.language;
      BrainSINSData.userEmail = AuxBrainsinsData.userEmail;

      if( AuxBrainsinsData.productId != null && AuxBrainsinsData.productId.length > 0){
        BrainSINSData.productId = AuxBrainsinsData.productId;
      }

      if(ACC.brainsins.getRecommenders(recommenderDIVS).length > 0){
         BrainSINSData.recommenders = ACC.brainsins.getRecommenders(recommenderDIVS);
      }

     return true;
  },
```

```
  /*
   * Dedicada a la primera página del checkout
   *
   * Envia a Brainsins los parámetros asociados al tipo de pagina 'checkout'
   * Incluye el recomendador en el div del HTML
   */
  cartPage: function(recommenderDIVS){

          BrainSINSData.pageType = AuxBrainsinsData.typePage;
          BrainSINSData.language = AuxBrainsinsData.language;
          BrainSINSData.userEmail = AuxBrainsinsData.userEmail;

          BrainSINSData.cart =  AuxBrainsinsData.cart;

          if(ACC.brainsins.getRecommenders(recommenderDIVS).length > 0){
         BrainSINSData.recommenders = ACC.brainsins.getRecommenders(recommenderDIVS);
     }

    return true;
  },

  /*
   * Dedica a la página del confirmación del pedido
   *
   * Sólo se encarga de enviar a Brainsins los parámetros asociados al tipo de pagina
'thankYou',
   * ya que en está página no se desea mostrar nigún recomendador
   */
  checkoutConfirmationPage : function(recommenderDIVS){

      BrainSINSData.pageType = AuxBrainsinsData.typePage;
      BrainSINSData.language = AuxBrainsinsData.language;
      BrainSINSData.userEmail = AuxBrainsinsData.userEmail;
      BrainSINSData.totalAmount = AuxBrainsinsData.totalAmount;

      if(ACC.brainsins.getRecommenders(recommenderDIVS).length > 0){
        BrainSINSData.recommenders = ACC.brainsins.getRecommenders(recommenderDIVS);
     }

    return true;
  },


  /*
   * Dedeica a la página de categorías de un producto
   *
   * Envía a Brainsins los parámetros asociados al tipo de página 'category'
   *
   * Incluye los recomendadores en los DIVS HTML que recibe como parámetro
   */
  categoryPage: function(recommenderDIVS){

          BrainSINSData.pageType = AuxBrainsinsData.typePage;
          BrainSINSData.language = AuxBrainsinsData.language;
          BrainSINSData.userEmail = AuxBrainsinsData.userEmail;


          if( AuxBrainsinsData.categories != null && AuxBrainsinsData.categories.length >
0){
                BrainSINSData.categories = AuxBrainsinsData.categories;
            }

          if(ACC.brainsins.getRecommenders(recommenderDIVS).length > 0){
                BrainSINSData.recommenders =
ACC.brainsins.getRecommenders(recommenderDIVS);
             }

          return true;
  },

  /*
   * FIN FUNCIONES PETICIONES BRAINSINS
   */

  initCarousel : function(){
```

```
          $(".brainsins").each(function(){
            $(this).owlCarousel({
              items : 6, //10 items above 1000px browser width
              pagination: false,
              navigation : true,
              itemsDesktop : [1100,5], // betweem 950px and 720px
              itemsDesktopSmall : [950,4], // betweem 950px and 720px
              itemsTablet: [740,3], //2 items between 600 and 0
              itemsMobile : [500,2], // itemsMobile disabled - inherit from itemsTablet
option
              rewindNav : false,
              lazyLoad: true
            });
        });

    return true;

  },

  createBrainsinsCart: function(){


          var customerCart = [];

          if($('#lightCart').size() > 0){

                       $('#lightCart > div').each(function(i, obj) {
                              cartEntry = {};
                              cartEntry['id'] =  $("input[name='light_code'",
this).val();
                              cartEntry['quantity'] = $("input[name='light_qty'",
this).val();
                              customerCart.push(cartEntry);

                       });
               }else{

                         $('#cart_content').ready(function ()
                         {

                                 if( $('.cart_popup .UpdateQuantityForm').size() > 0){

                                        $('.cart_popup
.UpdateQuantityForm').each(function(i, obj) {
                                                      cartEntry = {};
                                                      cartEntry['id'] =
$(obj).find(".productCode").first().val();
                                                      cartEntry['quantity']    =
$(obj).find(".qty").first().val();
                                                      customerCart.push(cartEntry);
                                               });

                                 }

                          });
               }

         AuxBrainsinsData.cart = customerCart;


  },


  /*
   * FUNCIÓN DE UTILIDAD
   *
   * Devuelve las unidades de un producto a partir del id del producto que se le pasa
como parametro
   *
   */
  getProductQuantityInCart: function(idProduct){

    result = 0;

    for(var i = 0; i < AuxBrainsinsData.cart.length; i++){
      if(AuxBrainsinsData.cart[i] != null && AuxBrainsinsData.cart[i].id == idProduct){
        return AuxBrainsinsData.cart[i].quantity;
```

```
      }
    }

    return result;

  },


  /*
   * FUNCIONES BOTON AÑADIR CARRITO
   *
   * El botón de añadir al carrito + - se encuentra en productAddToCartForm.tag. Su
contenido
   * no puede ser incluido directamente al HTML de Brainsins, ya que no se permite JSTL.
   *
   * Para incluir  el formulario de añadir al carrito de la tag, a Brainsins, es
necesario llevar
   * a cabo dos pasos:
   *
   * 1) Incluir desde la herramienta de Brainsins los datos del formulario a los que
tenemos
   * acceso en el entorno de Brainsins. Además, habrá que añadir tantos campos hidden,
como
   * atributos se necestien, para más tarde añadir el resultado esperado de la ejecución
   * de las funciones JSTL incluidas en la tag
   *
   * 2) Una vez se recibe el recomendador de brainsins, recorrer uno a uno los productos
que
   * incluyen y añadir al html del formulario los datos que falten, de modo que sea
equivalente
   * al resultado del formulario inluido en la tag.
   *
   *
   * El primero de los pasos se lleva a cabo desde la herramienta de Brainsins
   *
   * El segundo paso conlleva la ejecución de la función
ACC.brainsins.completeFormInformation
   *
   *
   * El formulario del que se compone "productAddToCart.tag" es el siguiente:
   *
//          <%@ tag body-content="empty" trimDirectiveWhitespaces="true" %>
//             <%@ attribute name="product" required="true"
type="de.hybris.platform.commercefacades.product.data.ProductData" %>
//             <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
//             <%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
//             <%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>
//
//              <%--
//                              (JAF), 20140210, copied from /compra-
online/web/webroot/WEB-INF/views/desktop/fragments/cart/cartPopup.jsp
//
//                              See also /compra-
online/web/webroot/_ui/desktop/common/js/acc.cartpopup.js
//
//                              All ids have the sufix 'ByProduct' and entry numbers are
changed by product codes. Classes remains the same.
//                                      (i.e.: from updateCartForm${entry.number} to
updateCartFormByProduct${product.code})
//                      --%>
//             <c:url value="/cart/updatePopupCartQuantityByProduct"
var="cartUpdateFormAction" />
//
//             <form id="updateCartFormByProduct_${product.code}"
name="UpdateQuantityForm" class="UpdateQuantityForm" action="${cartUpdateFormAction}"
method="post">
//                     <input type="hidden" class="productCode"
name="productCodePost" value="${product.code}"/>
//                     <input type="hidden" class="initialQuantity"
name="initialQuantity" value="${product.cartQuantity}"/>
//                     <input type="hidden" class="soldByType"      name="soldByType"
value="${product.soldByType}"/>
//
//                     <%-- (JAF), FIXME, 20140213, internacionalize strings --%>
//
//                     <%-- Less --%><button
id="QuantityProductMinusByProduct_${product.code}"
```

```
class="updateQuantityProductLess <c:if test="${product.stock.stockLevelStatus.code eq
'outOfStock' }">out-of-stock</c:if>" <c:if test="${product.stock.stockLevelStatus.code
eq 'outOfStock' }">disabled="true" aria-disabled="true"</c:if>" type="button">-</button>
//
//                    <%-- Qtty --%><label class="prod_quantity"
for="quantityByProduct_${product.code}"></label>
//                    <%-- Qtty --%><input type="text" size="1"
id="quantityByProduct_${product.code}" class="qty <c:if
test="${product.stock.stockLevelStatus.code eq 'outOfStock' }">out-of-stock</c:if>"
<c:if test="${product.stock.stockLevelStatus.code eq 'outOfStock' }">disabled="true"
aria-disabled="true"</c:if> name="quantity" value="${product.cartQuantity}" />
//                    <%-- Qtty --%><c:if test="${(product.soldByType eq 'WEIGHT') and
(not empty product.salesUnit)}"><span class="gr">${fn:replace(product.salesUnit, '.',
'')}</span></c:if>
//
//                    <%-- More --%><button
id="QuantityProductPlusByProduct_${product.code}"
class="updateQuantityProductMore <c:if test="${product.stock.stockLevelStatus.code eq
'outOfStock' }">out-of-stock</c:if>" <c:if test="${product.stock.stockLevelStatus.code
eq 'outOfStock' }">disabled="true" aria-disabled="true"</c:if> type="button">+</button>
//
//                    <%-- Dele --%><button id="deleteQuantityByProduct_${product.code}"
class="deleteQuantityProduct <c:if test="${product.stock.stockLevelStatus.code eq
'outOfStock' }">out-of-stock</c:if>" <c:if test="${product.stock.stockLevelStatus.code
eq 'outOfStock' }">disabled="true" aria-disabled="true"</c:if> type="button"></button>
//
//                    <%-- Add* --%><button id="addQuantityByProduct_${product.code}"
class="addQuantityProduct positive large <c:if
test="${product.stock.stockLevelStatus.code eq 'outOfStock' }">out-of-stock</c:if>"
<c:if test="${product.stock.stockLevelStatus.code eq 'outOfStock' }">disabled="true"
aria-disabled="true"</c:if> type="button"><spring:theme code="product.addCart.Form"
text="A&ntilde;adir"/></button>
//            </form>
   *
   *
   * El formulario incluido en la herramienta de Brainsins es el siguiente
   *
   *
   *
//          <div id="brainsinsForm_${id}" class="cart">
//                <form id="brainsins_updateCartFormByProduct_${id}"
name="UpdateQuantityForm" class="UpdateQuantityForm" action="/compra-
online/cart/updatePopupCartQuantityByProduct" method="post">
//                    <input type="hidden" class="productCode"     name="productCodePost"
value="${id}"/>
//                    <input type="hidden" class="initialQuantity" name="initialQuantity"
value=""/>
//                    <input type="hidden" class="soldByType"      name="soldByType"
value="${misc.soldByType}"/>
//                    <input type="hidden" class="averageRating"   name="averageRating"
value="${misc.averageRating}"/>
//                    <input type="hidden" class="unit"               name="unit"
value="${misc.unit}"/>
//
//                    <input id="price_{id}" type="hidden" class="price"
name="pricebr"           value="${price}"/>
//                    <button id="BrainsinsQuantityProductMinusByProduct_${id}"
class="updateQuantityProductLess"  type="button">
//                      -
//                    </button>
//                    <label class="prod_quantity" for="quantityByProduct_${id}">
//                    </label>
//                    <input type="text" size="1" id="BrainsinsQuantityByProduct_${id}"
class="qty" name="quantity" value="" />
//                    <span class="gr" id="brainsins_spangr_${id}" style="display:none">
//                    </span>
//                    <button id="BrainsinsQuantityProductPlusByProduct_${id}"
class="updateQuantityProductMore"  type="button">
//                      +
//                    </button>
//                    <button id="BrainsinsDeleteQuantityByProduct_${id}"
class="deleteQuantityProduct"  type="button">
//                    </button>
//                    <button id="BrainsinsAddQuantityByProduct_${id}"
class="addQuantityProduct positive large"  type="button">
//                    </button>
//                </form>
```

```
//              </div>
   *
   *
   *   Ha sido necesario incluir nuevos campos hidden al formulario para poder generar
el resultado
   *   de las diferentes funciones JSTL incluidas en el tag original. Dichos resultados,
serán calculados
   *   en este javascript.
   *
   */


  /*
   * Se encarga de añadir al html que se recibe desde Brainsins, la información que
falta
   * para simular el formulario incluido en la tag productAddToCartForm.tag
   *
   * Para ello, recorre cada uno de los productos recomendados y añade la información
   * que falte.
   *
   */
  completeFormInformation: function(){

    var brainsinsResponseList =  $("form[id^=" +
BraininsAddToCartFormElements['productForm'] + "]");

      $.each( brainsinsResponseList, function( key, obj ) {

          var form = obj;
          var id = form.elements.productCodePost.value;

          var price= (form.elements.pricebr!=null && form.elements.pricebr.value!= null
&& form.elements.pricebr.value.search("price") < 0 && form.elements.pricebr.value.length
> 1)?form.elements.pricebr.value:null;

          var soldByType = (form.elements.soldByType!=null &&
form.elements.soldByType.value!= null && form.elements.soldByType.value.length >
0)?form.elements.soldByType.value:null;

          var salesUnit = (form.elements.unit!=null && form.elements.unit.value != null
&& form.elements.unit.value.length > 0 )?form.elements.unit.value:null;

          var availableAddToCart = ACC.brainsins.isDataCorrectForAddToCart(price,
soldByType, salesUnit);

          var quantityProductMinusByProduct =
BraininsAddToCartFormElements['minusButton'] + id;
          ACC.brainsins.prepareOutOfStock(quantityProductMinusByProduct,
availableAddToCart);

          var quantityByProduct = BraininsAddToCartFormElements['quantityInput'] + id;
          ACC.brainsins.prepareQuantityByProduct(quantityByProduct, id,
availableAddToCart);

          ACC.brainsins.prepareSpanGr(soldByType, salesUnit, id);

          var quantityProductPlusByProduct = BraininsAddToCartFormElements['plusButton']
+ id;
          ACC.brainsins.prepareOutOfStock(quantityProductPlusByProduct,
availableAddToCart);

          var deleteQuantityByProduct = BraininsAddToCartFormElements['deleteQuantity']
+ id;
          ACC.brainsins.prepareOutOfStock(deleteQuantityByProduct, availableAddToCart);

          var addQuantityByProduct = BraininsAddToCartFormElements['addQuantity'] + id;
          ACC.brainsins.prepareOutOfStock(addQuantityByProduct, availableAddToCart);
          ACC.brainsins.prepareBrainsinsAddButtonText(addQuantityByProduct);

          // para visualizacion
          var quantity = ACC.brainsins.getProductQuantityInCart(id);
          quantity = (quantity!=null && quantity.length > 0)?quantity:0;
          ACC.cartpopup.updateQuantityProductLevelLocally(id, quantity );

          // estrellitas
          var averageRatingValue = form.elements.averageRating.value;
          var averageRatingStyle = BraininsAddToCartFormElements['starStyle'] + id;
```

```
          ACC.brainsins.prepareStars(averageRatingStyle, averageRatingValue);

          // no disponible
          var priceBrainsins = BraininsAddToCartFormElements['price'] + id;
          ACC.brainsins.preparePrice(priceBrainsins, availableAddToCart);

      });


  },

   /*
   * (EMF) - 20151127
   *
   * En el caso de que el producto no contenga precio o el valor "salesUnit" o bien
   * el valor "soldByType" no se podrá añadir dicho producto al carrito ya que
   * no existe stock en la tienda, o bien no se tiene la información necesaria
   * para incluir el botón añadir al carrito, es decir, el producto no contiene
   * todos los datos necearios en el catálogo.
   */
  isDataCorrectForAddToCart: function(priceAttr, soldByTypeAttr, salesUnitAttr){

         if(priceAttr == null || priceAttr.search("price") >= 0|| soldByTypeAttr == null
|| salesUnitAttr == null){
                return false;
         }

         return true;
  },

  /*
   * Modifica el valor del precio a "No disponible" cuando el producto
   * no contega la información necesaria para añadirla al carrito
   *
   */
  preparePrice: function(element, availableAddToCart){
    var element = "#" + element;

    if($(element) != undefined && availableAddToCart == false ){
      $(element).text(AuxBrainsinsData.brainsinsUnavailableProductText);
    }
  },


  /*
   * Se encarga añadir el estilo necesario para que se muestren las estrellitas
   * en un producto en función de las puntuaciones de los usuarios
   */
  prepareStars: function(element, averageRating){

     var element = "#" + element;

    if($(averageRating) == undefined || $(averageRating) < 0){
      averageRating = 0;
    }

    averageRating = parseFloat(averageRating) * 24;

        if (averageRating % 1 != 0) {
          averageRating = averageRating.toFixed(2);
        } else {
          averageRating = parseInt(averageRating);
        }

        var result = "width: " + averageRating + "px;";

    $(element).attr('style', result);

  },


  /*
   * Añade al botón de añadir el texto internacionalizado
   */
  prepareBrainsinsAddButtonText: function(element){
    element = "#" + element;
```

```
    if($(element) != undefined){
      $(element).text(AuxBrainsinsData.brainsinsAddButtonText);
    }
  },


  /*
   * Se emcarga de ocultar el botón que se le pasa como parámetro en
   * el caso de que sea necesario, es decir, el producto recibido como
   * respuesta de brainsins no tiene precio
   *
   * (EMF) 20151127 En el caso de que "salesUnit" venga vacío y / o
   * "soldByType" venga vacío, el producto estará mal creado en el
   * catálogo de productos y no se tendrán los valores necesarios para pintar
   * bien los botones de añadir al carrito. De modo que el comportamiento será el
   * mismo que en el caso de que no tenga precio.
   */
  prepareOutOfStock: function(element,availableAddToCart){

    element = "#" + element;

    if($(element) != undefined && availableAddToCart == false){

      if( $(element).attr('class').search("out-of-stock") < 0 ){
        // no lo contiene
         $(element).attr('class', $(element).attr('class') + " out-of-stock");
      }

        $(element).attr('disabled', 'true');
        $(element).attr('aria-disabled', 'true');
    }
  },


  /*
   * Se encarga de acutalizar el input con la cantidad del producto
   * añadido al carrito
   * (EMF) 20151127 En el caso de que "salesUnit" venga vacío y / o
   * "soldByType" venga vacío, el producto estará mal creado en el
   * catálogo de productos y no se tendrán los valores necesarios para pintar
   * bien los botones de añadir al carrito. De modo que el comportamiento será el
   * mismo que en el caso de que no tenga precio.
   */
  prepareQuantityByProduct: function(element, id, availableAddToCart){

    element = "#" + element;

    if($(element) != undefined){
      if(availableAddToCart == false){

      if( $(element).attr('class').search("out-of-stock") < 0 ){
        // no lo contiene
        $(element).attr('class',$(element).attr('class') + " out-of-stock");
      }

        $(element).attr('disabled', 'true');
        $(element).attr('aria-disabled', 'true')
      }

      var quantity = ACC.brainsins.getProductQuantityInCart(id);
        $(element).attr('value', (quantity!= undefined && quantity.length >
0)?quantity:0 );

    }


  },

  /*
   * Se encarga de incluir el texto "gr" en caso de que el atributo
   * sodByType sea igual a "WEIGHT"
   */
  prepareSpanGr: function(soldByType, salesUnit, id){

    if(soldByType != undefined && salesUnit != null && soldByType == 'WEIGHT' &&
salesUnit.length > 0){
```

```
      var gr = $("#brainsins_spangr_"+id);

      if(gr != undefined){
        gr.attr('style', 'display: block;');
        gr.text('gr');
      }
    }

  },

  /*
   * FIN FUNCIONES BOTÓN AÑADIR CARRITO
   */


  /*
   * FUNCIONES DE TRACKING
   */

  /*
   * Tracks user's email from subscription form
   * This function is call from acc.subscription.js
   *
   * EMF - 20151224
   */
  trackEmailSubscription: function(email){
         if(email != null && email.length > 0 && BrainSINSTracker != null){
                        BrainSINSTracker.trackEMailNewsletter( email, 1,
AuxBrainsinsData.language );
         }
  },


  /*
   * Track user's action on button add to cart
   *
   * EMF - 20151224
   */
  trackButtonAddToCart: function(productCode, quantity, price){
        if(BrainSINSTracker != null && price != null && price.length > 0){
               BrainSINSTracker.trackAddedToCart(productCode,quantity, price);
        }
  },

  /*
   * Track user's action on brainsins button add to cart
   *
   * EMF - 20151224
   */
  trackClicRecommenderButton: function(productCode, elementData){
         if(BrainSINSTracker != null){
                var idRecommender =
ACC.brainsins.recommenderNumberByDataName(elementData);
                var composed = idRecommender + "-" + productCode;
                brainsins.trackProductClicked(productCode, composed ,null);
         }
  },


  /*
   * FIN FUNCIONES DE TRACKING
   */

};


/*
 * Una vez se ha cargado la página del site, se procederá a ejecutar la función
encargada de solicitar
 * a Brainsins los productos recomendados. Dicha función depende del tipo de página en
que se encuentre
 * el cliente, existiendo 5 tipos:
 *
 *              'product', 'checkout','thankYou', 'category' y 'home'
 *
 * A cada uno de los elementos div de la página donde se van a incluir los
recomendadores de Brainsins,
```

```
 * se le asociará un Observer. De este modo, una vez se reciban los productos
recomendados, que serán
 * inclluidos en el div especificado, el Observer detectará dicha inclusión, lo que
llevará a la ejecución
 * de la función del observer
 *
 *  La función del Observer, se encarga de llevar a cabo dos tareas:
 *
 *      - Inicializar el carrusel de los productos recomendados
 *
 *      - Añadir en cada producto, la información que falte para completar el
 *        fomulario de añadir al carrito. De tal forma que se simule el formulario de la
 *        tag "productAddToCartForm.tag"
 *
 * PASOS de la función:
 *
 *
 *             1) Se almacena en la variable "recommenderDIVS", el conjunto de elementos
div donde se incluirán
 *             los recomendadores. No en todas las páginas serán incluidos
recomendadores, puede ser vacío.
 *
 *             2) Se llama a la función prepareBrainsinsData encargada de cargar la
información necesaria para
 *             Brainsins, así como de hacer la petición.
 *
 *      3) En el caso de que en la página se deseen incluir recomendadores, se ejecutará
el observer,
 *             que se encargará de llamar a la función para incluir los diferentes
productos, así como de
 *             añadir al formulario de añadir al carrito los datos oportunos.
 *
 */

  $(document).ready(function (){


    with(ACC.brainsins)
    {
      ACC.brainsins.includeScripts();

      //Create an observer instance
        var observer = new MutationObserver(function( mutations ) {

               ACC.brainsins.createBrainsinsCart();

               mutations.forEach(function( mutation ) {
                ACC.brainsins.initCarousel();
                ACC.brainsins.completeFormInformation();

          });
        });


        // PASO 1)
        var recommenderDIVS =  $( "div[id^='brainsins-recommender']");

        // PASO 2)
        prepareBrainsinsData(AuxBrainsinsData.typePage, recommenderDIVS);

        // PASO 3)
        if(recommenderDIVS != null && recommenderDIVS!= undefined &&
recommenderDIVS.length > 0) {
                $.each( recommenderDIVS, function( key, value ) {
                       observer.observe($(value)[0],{ childList: true});
                });
        }

      if(document.cookie.indexOf("bsUl=0")>=1){
          BrainSINSData.login=1;
      }

      // END EMF 20160108

    }

  });
```

11.3.2.16 acc.cartpopup.js

```
ACC.cartpopup = {

	bindAll: function ()
	{
		this.bindCartPop();
		this.bindCartEvents();
	},

	bindCartPop: function ()
	{
		$('#cart_content').hover(
				function ()
				{
					$.data(this, 'hover', true);
				},
				function ()
				{
					$.data(this, 'hover', false);
				}
		).data('hover', false);

		$('#rollover_cart_popup').hover(
				function ()
				{
					$.data(this, 'hover', true);
				},
				function ()
				{
					$.data(this, 'hover', false);
				}
		).data('hover', false);

		$('#cart_content').ready(function ()
		{
			/*
			 *  Mini cart refreshall must be SYNC in this point in order to
invoke "updateAllQuantities" with fresh entries.
			 */
			ACC.cartpopup.refreshMiniCart(false);
			ACC.cartpopup.updateAllQuantities();
		});

		$(document).on('click', '#ajax_cart_close', function (e)
		{
			e.preventDefault();
			$('#rollover_cart_popup').hide();
		});

		$('#rollover_cart_popup').ready(function (){


		});
	},

	bindCartEvents: function ()
	{
		$(document).on("keypress",  ".UpdateQuantityForm", function(e) {
			var code = e.keyCode || e.which;
			if (code  == 13) {
				e.preventDefault();
			}
		});

		$(document).on("keyup ",  ".UpdateQuantityForm", function(e) {
			var code = e.keyCode || e.which;
			if (code  == 13) {
				e.preventDefault();
				ACC.cartpopup.updateQuantity(
[ACC.cartpopup.getOldQtyOnChange, ACC.cartpopup.getOldQty], $(this));
				return false;
```

```
                        }
                    });

                $(document).on("click",  ".askPos", ACC.cartpopup.askPos);
                $(document).on("click",  ".updateQuantityProductMore",
[ACC.cartpopup.getOldQty, ACC.cartpopup.more],
ACC.cartpopup.updateQuantityEventHandler);
                $(document).on("click",  ".updateQuantityProductLess",
[ACC.cartpopup.getOldQty, ACC.cartpopup.less],
ACC.cartpopup.updateQuantityEventHandler);
                $(document).on("click",  ".deleteQuantityProduct",
[ACC.cartpopup.getOldQty, ACC.cartpopup.remove],
ACC.cartpopup.updateQuantityEventHandler);
                $(document).on("click",  ".addQuantityProduct",
[ACC.cartpopup.getOldQty, ACC.cartpopup.more],
ACC.cartpopup.updateQuantityEventHandler);

                //SHT, 20440318, vaciar carrito
                $(document).on("click",  ".doEmptyCartBut",  ACC.cartpopup.cartEmpty);
        },

        /*
         * Updates the product from the incoming event.
         */
        updateQuantityEventHandler: function(qtyFunctionsParam) {

                var form        = $(this).parent();

                ACC.cartpopup.updateQuantity(qtyFunctionsParam.data, form);
        },

        /*
         * Updates the product on the given form.
         */
        updateQuantity: function(qtyFunctions, form) {

                var inputQty    = form.find('.qty');
                var soldByType  =
inputQty.parent().closest('form').find('.soldByType').val();
                var productCode = form.find('.productCode').val();
                var oldQty      = qtyFunctions[0](inputQty, soldByType); // Old function
callback
                var newQty      = qtyFunctions[1](inputQty, soldByType); // New function
callback

                // New quantity must be greater or equal than 0. Equal means removal.
                if(newQty < 0) return;

                // Update form (client) data before submit
                inputQty.val(newQty)

                if($('.pointOfService').text().length > 0) {
                        ACC.cartpopup.updateQuantityWhenNeeded(form, inputQty,
productCode, oldQty, newQty);
                } else {
                        ACC.cartpopup.askPos( productCode, form );
                }
        },

        /*
         * Updates the product on the given form.
         *
         *      EMF - 20160425 - [DIAC-1825]
         *      Si el usuario se encuentra en la página del carrito y además
         *      Si el producto a incluir en el carrito no está incluido en el carrito del
usuario
         *      se lleva a cabo un refresh de la página para que se recargue la tabla de
productos
         *      sin complicar aun mas el comportamiento - step [4]
         */
        updateQuantityWhenNeeded: function(form, inputQty, productCode, oldQty, newQty) {

                // [0] Initialize event mappings per product
                if(typeof ACC.cartpopup.events === "undefined") {
                        ACC.cartpopup.events = {}; // pairs of [oldQty][timeoutEvent]
                }
```

```
		// [1] Manage former event
		var formerEvent = ACC.cartpopup.events[productCode];
		var rollbackQty = oldQty;
		if(!(typeof formerEvent	 === "undefined"))  {
			// Remove previous events (grouping several events into a single
one)
			rollbackQty = formerEvent[0];
			clearTimeout( formerEvent[1] );
		} else {
			// Obtain former event old quantity to rollback if needed
			rollbackQty = oldQty;
		}

		// [2] Queue last event
		// [LDR 20160309] DIAEC-1812 mejoras ajax grid
		var miniCartUID = '';
		if($('#miniCartUID').val()!=null){
			miniCartUID = $('#miniCartUID').val();
		}
		/**
		 * [LDR 20160404] por [CartPopUpController ->
linea:239:Collections.reverse(entries)]
		 * hay que bloquear los botones de 'disminucion de cantidad' y
'eliminacion de un producto'
		 * hasta que acabe de ejecutarse la peticion ajax. Evitaremos asi un
error que se da cuando
		 * eliminamos rapidamente el segundo elemento de la lista y seguidamente
eliminamos el primero
		 * sin que haya terminado la ejecucion de la primera peticion.
		 */
		if ($("#cart-block button.updateQuantityProductLess").length) {
			$("#cart-block
button.updateQuantityProductLess").prop("disabled",true);
		}
		if ($("#cart-block button.deleteQuantityProduct").length) {
			$("#cart-block
button.deleteQuantityProduct").prop("disabled",true);
		}
		var timeoutId = setTimeout( function(){
			form.ajaxSubmit({
				// [LDR 20160309] DIAEC-1812 mejoras ajax grid
				data: {miniCartUID: miniCartUID},
			    dataType: 'json',
				async: true, // Very important point to take into account,
running this async !!!
				success: function(data, status, settings) {

					// Animate cart when changed
					/** SGB 28012016 si se incrementa la cantidad se
muestra la  animación de añadir. Si no la de sacar del carrito **/
					if(oldQty <= newQty){
						ACC.cartpopup.animateArrow();
						ACC.cartpopup.animateArrowResponsiveMore();
					}
					else{
						ACC.cartpopup.animateArrowResponsiveLess();
					}

					/*ACC.cartpopup.refreshPage(form);*/

					/* a */ /* At step [3], immediately after user
action */
					/* b */ /* At step [3], immediately after user
action */

					/* c */
ACC.cartpopup.updateQuantitySuccessCartLevel	(data, status, settings);
					//LDR 20160202 DIAEC-1624 recarga promociones
vouchers ajax
					ACC.cartpopup.refreshPromotionsVouchers (data,
status, settings);
					// [LDR 20160309] DIAEC-1812 mejoras ajax grid
					/* d */ ACC.cartpopup.refreshHTMLMiniCart(data);

					// EMF - 20151228 - Brainsins
					ACC.cartpopup.callBrainsinsTrackingFunction(form,
productCode, newQty, oldQty);
```

```
                                },
                                error:   function(data, status, settings) {
                                        // LFP [20150904] Sacado contenido de popup a una
jsp para internacionalizar. Haciendo llamada AJAX al controlador para obtener el
contenido html
                                        // Restore previous values
                                        $.ajax({
                                                type : "POST",
                                                url : ACC.config.contextPath +
"/cart/cartAddProductError",
                                                async: false,
                                                success: function(datos){
                                                        $.colorbox({
                                                                html: datos,
                                                                height:'200px',
                                                                width:'80%',
                                                                top:'10%',
                                                                maxWidth:'500px',
                                                                onComplete : function(){

        $(this).colorbox.resize();
                                                                }
                                                        });
                                                }
                                        });

                                        /* a */
ACC.cartpopup.updateQuantityProductLevelLocally (productCode, rollbackQty);
                                        /* b */ ACC.cartpopup.updateMiniCartEntryLocally
(productCode, rollbackQty);

                                        /* c */ /* Only during success handling */
                                        /* d */ ACC.cartpopup.refreshMiniCart(); // <--
Server interaction
                                },
                                // [LDR 20160404] desactivacion temporal botones borrado
hasta acabar la peticion ajax
                                complete: function(data, status, settings){
                                        if ($("#cart-block
.updateQuantityProductLess").length) {
                                                $("#cart-block
button.updateQuantityProductLess").prop("disabled",false);
                                }
                                        if ($("#cart-block
button.deleteQuantityProduct").length) {
                                                $("#cart-block
button.deleteQuantityProduct").prop("disabled",false);
                                        }
                                }
                        });
                }, 500);


                // [3] Update queue
                ACC.cartpopup.events[productCode] = [rollbackQty, timeoutId];

                // [4] Update everywhere LOCALLY without waiting for success handler
above
                /* a */ ACC.cartpopup.updateQuantityProductLevelLocally
(productCode, newQty);
                /* b */ ACC.cartpopup.updateMiniCartEntryLocally
(productCode, newQty);
        },

        /*
         * Handles succesful product update.
         */
        updateQuantityProductLevelLocally: function(productCode, newQty) {

                if(newQty == 0) {
                        // Product removal
                        $("#updateCartFormByProduct_" + productCode + "
.updateQuantityProductMore").hide();
                        $("#updateCartFormByProduct_" + productCode + "
.updateQuantityProductLess").hide();
```

```
                        $("#updateCartFormByProduct_" + productCode + " .qty").hide();
                        $("#updateCartFormByProduct_" + productCode + " .gr").hide();
                        $("#updateCartFormByProduct_" + productCode + "
.addQuantityProduct").css("display","block");


                        $("#item_" + productCode).closest(".tr").remove();  //IGF 20150417
DIAEC - 304
                } else {
                        // Product increase|decrease
                        $("#updateCartFormByProduct_" + productCode + "
.updateQuantityProductMore").css("display","block");
                        $("#updateCartFormByProduct_" + productCode + "
.updateQuantityProductLess").css("display","block");
                        $("#updateCartFormByProduct_" + productCode + "
.qty").css("display","block");
                        $("#updateCartFormByProduct_" + productCode + "
.gr").css("display","block");
                        $("#updateCartFormByProduct_" + productCode + "
.addQuantityProduct").hide();
                }

                $("#updateCartFormByProduct_"     + productCode + " .qty").val(newQty);


                // brainsins

                if(newQty == 0) {
                        // Product removal
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.updateQuantityProductMore").hide();
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.updateQuantityProductLess").hide();
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.qty").hide();
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.gr").hide();
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.addQuantityProduct").css("display","block");


                        $("#item_" + productCode).closest(".tr").remove();  //IGF 20150417
DIAEC - 304
                } else {
                        // Product increase|decrease
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.updateQuantityProductMore").css("display","block");
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.updateQuantityProductLess").css("display","block");
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.qty").css("display","block");
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.gr").css("display","block");
                        $("#brainsins_updateCartFormByProduct_" + productCode + "
.addQuantityProduct").hide();
                }

                $("#brainsins_updateCartFormByProduct_"     + productCode + "
.qty").val(newQty);

        },

        /*
         * Refresh the given mini cart entry.
         */
        updateMiniCartEntryLocally : function(productCode, qty)
        {
                // To be implemented

                // This method should refresh not only quantity but also price and totals
LOCALLY within 'mini' cart and 'full' cart
                // This method must also take into account 0 quantity entries (entry
removal)

                // console.log("To be implemented updating product with code " +
productCode + " and quantity " + qty);
        },
```

```
	/*
	 * Handles succesful product update.
	 */
	updateQuantitySuccessCartLevel: function(data, status, settings) {

		var entryNumber    = data.cartModificationData.entryNumber;
		var productCode    = data.cartModificationData.productCode;
		var quantity       = data.cartModificationData.quantity.split()[0];
		var newQty         = parseInt(quantity);
		var newPrice       = data.cartModificationData.entryTotalPrice;
		var subtotalPrice  = data.cartModificationData.subtotalPrice;
		var deliveryCost   = data.cartModificationData.deliveryCost;
		var totalDiscounts = data.cartModificationData.totalDiscounts;
		var totalPrice     = data.cartModificationData.totalPrice;

		if(data.cartModificationData.statusCode == "maxOrderQuantityExceeded") {
			// LFP [20150904] Sacado contenido de popup a una jsp para
internacionalizar. Haciendo llamada AJAX al controlador para obtener el contenido html
			// Max quantity exceeded
			$.ajax({
				type : "POST",
				url : ACC.config.contextPath +
"/cart/cartAddProductMaxError",
				async: false,
				success: function(datos){
					$.colorbox({
						html: datos,
						height:'200px',
						width:'80%',
						top:'10%',
						maxWidth:'500px',
						onComplete : function(){
							$(this).colorbox.resize();
						}
					});
				}
			});

			/* a */
ACC.cartpopup.updateQuantityProductLevelLocally(productCode, newQty);
			/* b */ ACC.cartpopup.updateMiniCartEntryLocally
(productCode, newQty);
		}
		else if(data.cartModificationData.statusCode == "lowStock" ||
data.cartModificationData.statusCode == "noStock") {
			// LFP [20150904] Sacado contenido de popup a una jsp para
internacionalizar. Haciendo llamada AJAX al controlador para obtener el contenido html
			// Stock problem
			$.ajax({
				type : "POST",
				url : ACC.config.contextPath +
"/cart/cartAddProductStockError",
				async: false,
				success: function(datos){
					$.colorbox({
						html: datos,
						height:'200px',
						width:'80%',
						top:'10%',
						maxWidth:'500px',
						onComplete : function(){
							$(this).colorbox.resize();
						}
					});
				}
			});

			/* a */
ACC.cartpopup.updateQuantityProductLevelLocally(productCode, newQty);
			/* b */ ACC.cartpopup.updateMiniCartEntryLocally
(productCode, newQty);
		}

		// Update mini cart entry (if any)
		$("#item_" + productCode).find(".qty").val(newQty);//IGF 20150417 DIAEC -
304
```

```
                // Update cart page entry (if any) and totals
                if(window.location.href.indexOf("cart") > -1) {


                        $("#item_" + productCode).find(".total").text(newPrice);//IGF
20150417 DIAEC - 304
                        $("#subtotalPrice").text(subtotalPrice);
                        $("#deliveryCost").text(deliveryCost);
                        $("#totalDiscounts").text(totalDiscounts);
                        $("#totalPrice").text(totalPrice);

                        $(".cart_total").text('Total: ' + totalPrice);//IGF 20150417 DIAEC
- 304

                        /*$(".promo").hide();*/ // hide promos although it would be better
update its content

                        //$(".qty[value=0]").closest(".tr").remove();//IGF BORRAR
                        if($(".tbody .tr").length == 0) {
                                window.location.reload();
                        }
                }
        },

                /*
         * LDR 20160129 DIAEC-1624 optimizacion del carrito con recarga ajax de
promociones
         */
        refreshPromotionsVouchers: function(data, status, settings)
        {
                // [LDR 20160302] DIAEC-1811 mejoras ajax /cart
                $(".span-15").html(data.promotionsVouchersHTML);
        },

        /*
         * Refresh the FULL mini cart if and only if the rollover pop URL is set
         *
         * This is a way of detecting whenever the target URL is set or not.
         */
                /*
         * [LDR 20160308] DIAEC-1812 mejoras ajax grid
         *
         */
        refreshHTMLMiniCart : function(data)
        {
                $("#rollover_cart_popup").html(data.cartHTML);
                $("#rollover_cart_popup").fadeIn();

                /** Update data for new Cart **/
                var totalProducts = '0';
                // [LDR 20160404] escapado del caracter euro
                var htmlTotal = '0,00 \u20AC';
                if($('#rollover_cart_popup').find('.legend').length > 0){
                        totalProducts = $('#rollover_cart_popup').find('.legend').text();
                }
                $('.ico-tablet-cart span.count').text(totalProducts);

                if($('#rollover_cart_popup').find('.prod_cart-total .prod_price').length
> 0){
                        htmlTotal = $('#rollover_cart_popup').find('.prod_cart-
total').html();
                        //20150529 DSA [DIAEC-409] eliminar el espacio en blanco que da
problemas al pintar importes superiores a 1000â,¬
                        htmlTotal = htmlTotal.replace(/ /g, " ");
                        htmlTotal = htmlTotal.split('</p>');
                        //20150305 FPR Nueva versiÃ³n se usa el subtotal sin gastos de
envÃ-o.
                        htmlTotal = htmlTotal[0].split('>');
                        var total = htmlTotal[1];
                        //20150529 DSA [DIAEC-409] eliminar espacios en blanco para poder
mostrar todo el importe en el carrito
                        if(total.length == 11){
                                total = total.replace(/ /g, "");
                        }
                        $('.ico-tablet-cart span.price').text(total);
                }
```

```
                else{
                        $('.ico-tablet-cart span.price').text(htmlTotal);
                }

                /* En tablet el carrito debe ocupar toda la altura */
                if( /Android|webOS|iPad|BlackBerry/i.test(navigator.userAgent) ) {
                        if($(window).width() > 719) {
                                $(".RD_Desplegar").addClass("RD_DesplegarTablet");
                                $(".child-menu").addClass("menu_node_level2Tablet");
                                var windowHeight = $(window).height();
                                var heightItemsList = windowHeight - $('#cart-block
#cart_header').outerHeight() - 25 - $('#cart-block .cart_popup .prod_cart-
total').outerHeight() - 4 - $('#cart-block .cart-actions').outerHeight() - $('#cart-
block .cart_popup .links').outerHeight();
                                $('#cart-block .cart_popup .legend +
ul').css('height',heightItemsList);
                        }
                }

                if(($(window).width() <= 719)&&($(window).height() <= 400)){
                        var windowHeight = $(window).height();
                        var heightItemsList = windowHeight - $('#cart-block
#cart_header').outerHeight() - 50;
                        $('#cart-block .cart_popup ul').css('height',heightItemsList);
                } else if($(window).width() <= 719) {
                        var windowHeight = $(window).height();
                        var heightItemsList = windowHeight - $('#cart-block
#cart_header').outerHeight() - 50 - 5 - $('#cart-block .cart_popup .prod_cart-
total').outerHeight() - $('#cart-block .cart-actions').outerHeight() - $('#cart-block
.cart_popup .links').outerHeight();
                        $('#cart-block .cart_popup ul').css('height',heightItemsList);
                }

        },

        /*
         * Refresh the FULL mini cart if and only if the rollover pop URL is set
         *
         * This is a way of detecting whenever the target URL is set or not.
         * @deprecated, use -> refreshHTMLMiniCart(data)
         */
        refreshMiniCart : function(async)
        {
                async = (typeof async === "undefined") ? true : async;

                if(rolloverPopupUrl.indexOf('cart') > -1) { // 20140529
                        $.ajax({
                                url: rolloverPopupUrl,
                                cache: false,
                                type: 'GET',
                                async: async,
                                success: function (result)
                                {

                                        $('#rollover_cart_popup').html(result);
                                        $('#rollover_cart_popup').fadeIn();

                                        /** Update data for new Cart **/
                                        var totalProducts = '0';
                                        var htmlTotal = '0,00€';
                                        if($('#rollover_cart_popup').find('.legend').length
> 0){
                                                totalProducts =
$('#rollover_cart_popup').find('.legend').text();
                                        }
                                        if($('#rollover_cart_popup').find('.prod_cart-total
.prod_price').length > 1){

                                                htmlTotal =
$('#rollover_cart_popup').find('.prod_cart-total').html();
                                                //20150529 DSA [DIAEC-409] eliminar el
espacio en blanco que da problemas al pintar importes superiores a 1000€
                                                htmlTotal = htmlTotal.replace(/ /g, "
");
                                                htmlTotal = htmlTotal.split('</p>');
```

```
                                        //20150305 FPR Versión original, se adjunta
el precio total con los gastos de envío.
                                        //htmlTotal = htmlTotal[2].split(';');

                                        //20150305 FPR Nueva versión se usa el
subtotal sin gastos de envío.
                                        htmlTotal = htmlTotal[0].split('>');
                                        var total = htmlTotal[1];
                                        //20150529 DSA [DIAEC-409] eliminar espacios
en blanco para poder mostrar todo el importe en el carrito
                                        if(total.length == 11){
                                                total = total.replace(/ /g, "");
                                        }
                                        $('.ico-tablet-cart
span.price').text(total);

                                }
                                else{
                                        $('.ico-tablet-cart
span.price').text(htmlTotal);
                                }

                                $('.ico-tablet-cart
span.count').text(totalProducts);

                                /* En tablet el carrito debe ocupar toda la altura
*/
                                if(
/Android|webOS|iPad|BlackBerry/i.test(navigator.userAgent) ) {

                                        if($(window).width() > 719) {

        $(".RD_Desplegar").addClass("RD_DesplegarTablet");
                                                $(".child-
menu").addClass("menu_node_level2Tablet");
                                                var windowHeight =
$(window).height();
                                                var heightItemsList = windowHeight -
$('#cart-block #cart_header').outerHeight() - 25 - $('#cart-block .cart_popup
.prod_cart-total').outerHeight() - 4 - $('#cart-block .cart-actions').outerHeight() -
$('#cart-block .cart_popup .links').outerHeight();
                                                $('#cart-block .cart_popup .legend +
ul').css('height',heightItemsList);
                                        }

                                }
                                if(($(window).width() <= 719)&&($(window).height()
<= 400)){
                                        var windowHeight = $(window).height();
                                        var heightItemsList = windowHeight -
$('#cart-block #cart_header').outerHeight() - 50;
                                        $('#cart-block .cart_popup
ul').css('height',heightItemsList);
                                }

                                else if($(window).width() <= 719){
                                        var windowHeight = $(window).height();
                                        var heightItemsList = windowHeight -
$('#cart-block #cart_header').outerHeight() - 50 - 5 - $('#cart-block .cart_popup
.prod_cart-total').outerHeight() - $('#cart-block .cart-actions').outerHeight() -
$('#cart-block .cart_popup .links').outerHeight();
                                        $('#cart-block .cart_popup
ul').css('height',heightItemsList);
                                }

                        }
                });
        }
        },

        /*
         * Modified by EMF, 20160413, [DIAEC-1825]
         *
         * Based on the entries in the mini cart updates all quantities in the product
search results grid page.
         * or if the minicart component does not exist like occurs in checkout pages, it
is based on
```

```
         * lightCartEntries.tag
         *
         */
        updateAllQuantities: function(data, status, settings) {

                if( $('.cart_popup .UpdateQuantityForm').size() > 0){
                        ACC.cartpopup.updateAllQuantitiesFromMiniCart();
                }else{
                        if($('#lightCart').size() > 0){
                                ACC.cartpopup.updateAllQuantitiesFromLightCart();
                        }
                }
        },

        /*
         * (EMF), 20160413, [DIAEC-1825]
         *
         * Update all the quantities based on minicart
         */
        updateAllQuantitiesFromMiniCart: function (data, status, settings){

                $('.cart_popup .UpdateQuantityForm').each(function(i, obj) {

                        var entryNumber = $(obj).find(".entryNumber").first().val();
                        var productCode = $(obj).find(".productCode").first().val();
                        var quantity    = $(obj).find(".qty").first().val();
                        var newQty      = parseInt(quantity);

                                var idHybrisProduct = "#updateCartFormByProduct_";
                                var idBrainsinsProduct =
"#brainsins_updateCartFormByProduct_";

                                ACC.cartpopup.updateProductFormElements(idHybrisProduct,
productCode, newQty);

                        // Update mini cart entry
                        $("#updateCartForm" + entryNumber + " .qty").val(newQty);

                                // brainsins
                                ACC.cartpopup.updateProductFormElements(idBrainsinsProduct,
productCode, newQty);

                                // Update mini cart entry
                                $("#brainsins_updateCartFormByProduct_" + entryNumber + "
.qty").val(newQty);
                });
        },


        /*
         * (EMF), 20160413, [DIAEC-1825]
         *
         * Update all the quantities based on lightCartEntries.tag
         */
        updateAllQuantitiesFromLightCart: function (data, status, settings){
                $('#lightCart > div').each(function(i, obj) {

                        var productCode =  $("input[name='light_code'", this).val();
                        var quantity = $("input[name='light_qty'", this).val();
                        var newQty = parseInt(quantity);


                        var idHybrisProduct = "#updateCartFormByProduct_";
                        var idBrainsinsProduct = "#brainsins_updateCartFormByProduct_";

                        ACC.cartpopup.updateProductFormElements(idHybrisProduct,
productCode, newQty);

                        // brainsins
                        ACC.cartpopup.updateProductFormElements(idBrainsinsProduct,
productCode, newQty);

                });
        },

        /*
         * (EMF), 20160413, [DIAEC-1825]
```

```
         *
         * Update all the quantities
         */
        updateProductFormElements: function (id, productCode, newQty){

                        if(newQty == 0) {
                        $(id + productCode + " .updateQuantityProductMore").hide();
                        $(id + productCode + " .updateQuantityProductLess").hide();
                        $(id + productCode + " .qty").hide();
                        $(id + productCode + " .gr").hide();
                        $(id + productCode + "
.addQuantityProduct").css("display","block");
                        } else {
                        $(id + productCode + "
.updateQuantityProductMore").css("display","block");
                        $(id + productCode + "
.updateQuantityProductLess").css("display","block");
                        $(id + productCode + " .qty").css("display","block");
                        $(id + productCode + " .gr").css("display","block");
                        $(id + productCode + " .addQuantityProduct").hide();
                        }

                        // Update product listing
                $(id + productCode + " .qty").val(newQty);
        },


        /*
         * Asks the user for its point of service and delivery mode.
         */
        askPos: function (productCode, productForm)
        {
                var storeSelectionPopup = $('#store_selection_question_popup');
                var cboxOverlay = $('#cboxOverlay');

                storeSelectionPopup.hide();
                cboxOverlay.hide();

                var jqXhr = $.ajax( showPosSelectionFormUrl, {
                        async: false
                });
                storeSelectionPopup.html(jqXhr.responseText);

                var storeSelectionForm = $('[name=storeSelectionForm]');
                $('[name=productCode]', storeSelectionForm).attr( 'value', productCode );

                var deliveryModeName;
                var posName;

                storeSelectionForm.ajaxForm({

                        beforeSubmit: function(arr, form, options) {
                                deliveryModeName =
$('input[name=deliveryModeName]:checked', form).val();
                                posName;

                                if (deliveryModeName == 'pickup-in-store') {
                                        posName = $('[name=storePosName]', form).val();
                                } else {
                                        posName = $('[name=posName]',      form).val();
                                }

                                options.url = pointOfServiceUrl + '/' + posName + '/' +
deliveryModeName;

                                $('.deliveryMode')  .text(deliveryModeName);
                                $('.pointOfService').text(posName);

                                // FIXME, (JAF), obligar a introducir valores
                                return true;
                        },
                        success: function(cartResult, statusText, xhr, formElement) {

                                ACC.cartpopup.posSelectionSuccessHandler (posName,
productCode, deliveryModeName, productForm);
                        }
                });
```

```
		cboxOverlay.show();
		storeSelectionPopup.fadeIn();

		if( /Android|webOS|iPad|BlackBerry/i.test(navigator.userAgent) ) {
			if(($(window).width() <= 719)&&($(window).height() <= 400)){
				var topPopup = (($(window).height() -
storeSelectionPopup.height()) / 2) + $(window).scrollTop()
				storeSelectionPopup.css('top',topPopup);
			}
		}
	},

	/*
	 * Handles succesful point of service and delivery mode selection.
	 */
	posSelectionSuccessHandler: function (posName, productCode, deliveryModeName,
productForm)
	{
		if( posName && (posName != '') &&  deliveryModeName && (deliveryModeName
!= '') && productCode && (productCode != '')) {

			var jqXhr = $.ajax( productAvailabilityUrl + '/' + posName + '/' +
productCode + '/' + deliveryModeName, {
				async: false}
			);

			var isProductAvailable = jqXhr.responseText;
			if( isProductAvailable == 'true' )
			{
				ACC.cartpopup.updateQuantity(
[ACC.cartpopup.getOldQtyOnChange, ACC.cartpopup.getOldQty], productForm);
			}
			else
			{
				// LFP [20150904] Sacado contenido de popup a una jsp para
internacionalizar. Haciendo llamada AJAX al controlador para obtener el contenido html
				$.ajax({
					type : "POST",
					url : ACC.config.contextPath +
"/cart/cartAddProductZoneError",
					async: false,
					success: function(datos){
						$.colorbox({
							html: datos,
							height:'200px',
							width:'80%',
							top:'10%',
							maxWidth:'500px',
							onComplete : function(){
								$(this).colorbox.resize();
							}
						});
					}
				});

				/* a */
ACC.cartpopup.updateQuantityProductLevelLocally(productCode, 0);
				/* b */ ACC.cartpopup.updateMiniCartEntryLocally
(productCode, 0);
			}
		} else {
			return false;
		}

		var storeSelectionPopup = $('#store_selection_question_popup');
		var cboxOverlay = $('#cboxOverlay');
		storeSelectionPopup.hide();
		cboxOverlay.hide();
	},

	/*
	 * Returns zero.
	 */
	zero: function(inputQty, soldByType){

		return 0;
```

```
	},

	/*
	 * Gets the previous product quantity (before the event)
	 */
	getOldQty: function(inputQty, soldByType){

		var oldVal = inputQty.val().split()[0]; // remove trailing units if
needed
		var oldQty = parseInt(oldVal);

		oldQty = (isNaN(oldQty)) ? 0 : oldQty;

		return oldQty;
	},

	/*
	 * Gets the previous product quantity in the case of direct input (change value
and hit enter)
	 */
	getOldQtyOnChange: function(inputQty, soldByType){

		var oldVal =
inputQty.parent().closest('form').find('.initialQuantity').val().split()[0]; // remove
trailing units
		var oldQty = parseInt(oldVal);

		return oldQty;
	},

	/*
	 * Returns the old quantity plus the increment value.
	 */
	more: function(inputQty, soldByType){

		var oldQty = ACC.cartpopup.getOldQty(inputQty);
		//16/03/2015 - Add function param to fix SD191618 - CGM001ES - Web:
Articulos de peso - as400 datos
		var increment = ACC.cartpopup.gap(inputQty, soldByType);
		var result = oldQty + increment;

		return result;
	},

	/*
	 * Returns the old quantity minus the decrement value.
	 */
	less: function(inputQty, soldByType){

		var oldQty = ACC.cartpopup.getOldQty(inputQty);
		//16/03/2015 - Add function param to fix SD191618 - CGM001ES - Web:
Articulos de peso - as400 datos
		var decrement = ACC.cartpopup.gap(inputQty, soldByType);

		return oldQty - decrement;
	},

	/*
	 * Returns '0', that means product removal.
	 */
	remove: function(inputQty, soldByType){

		return 0;
	},

	/*
	 * Returns the increment/decrement to be applied for an event.
	 */
	gap: function(inputQty, soldByType){

		var increment  = ("WEIGHT" == soldByType) ? 100 : 1;

		return increment;
	},

	/*
	 * Empties the cart.
```

```
	 */
	cartEmpty:function(){

		$.ajax({
			url : cartEmptyUrl,
			cache:false,
			type: 'GET',
			success  : function(data)
			{
				// Refresh mini cart
				ACC.cartpopup.refreshMiniCart();

				// (JAF), 20140318, see updateQuantityProductSuccess when
qty is 0
				$(".qty").val(0);

				$(".updateQuantityProductMore").hide();
				$(".updateQuantityProductLess").hide();
				$(".qty").hide();
				$(".gr").hide();
				$(".addQuantityProduct").css("display","block");
			},//Fin success
		});
	},

	animateArrow: function(inputQty){
		var ele = $('#cart_content .arrow-animated');
		ele.css({'top':'27px','opacity':'0'});

		ele.animate({"opacity": 1}, "fast",function(){ele.css({'top':'27px'});})
			.animate({"opacity": 0,"top": "46px"},
"slow",function(){ele.css({'top':'27px','opacity':'0'});});
	},

	animateArrowResponsiveMore: function(inputQty){
		var ele = $('.animateArrowResponsive');
		var arrow = ele.find('.arrow');
		ele.find('.arrowLess').hide();
		ele.find('.arrow').show();
		ele.css({'opacity':'0','visibility':'visible'});

		ele.animate({"opacity": 1}, "fast",function(){

				arrow.animate({"top": "60px"}, 1000);
				ele.animate({'opacity':'0'},1000,
function(){arrow.css({'top':'0'});ele.css({'visibility':'hidden'});});

			}
		);
	},

	animateArrowResponsiveLess: function(inputQty){
		var ele = $('.animateArrowResponsive');
		var arrow = ele.find('.arrowLess');

		ele.find('.arrow').hide();
		ele.find('.arrowLess').show();
		ele.css({'opacity':'0','visibility':'visible'});

		ele.animate({"opacity": 1}, "fast",function(){

				arrow.animate({"top": "0px"}, 1000);
				ele.animate({'opacity':'0'},1000,
function(){arrow.css({'top':'60px'});ele.css({'visibility':'hidden'});});

			}
		);
	},

	/*
	 * EMF - 20151228 - [DIAEC-1337] call tracking functions from acc.brainsins.js
	 */
	callBrainsinsTrackingFunction: function(form, productCode, newQty, oldQty){

		// EMF - 20151224 - Brainsins track action on button add to cart
		if(form != null && form.find('.price') != null ){
			var qtyDifference = newQty - oldQty;
```

```
                        var priceBr = form.find('.price').val();
                        ACC.brainsins.trackButtonAddToCart(productCode, qtyDifference,
priceBr);

                        // EMF - 20151228 - if the form belongs to brainsins it is
necessary
                        // call the function that track user's action on brainsins' button
                        if( form.attr('id').indexOf('brainsins') == 0){ // brainsins form
                                var element = document.getElementById(form.attr('id'));
                                var founded = false;

                                while(element.parentNode && founded == false ) {
                                        element = element.parentNode;
                                        var elementAttr = element.getAttribute('data-
name');
                                        if( elementAttr != null &&
elementAttr.indexOf('recommender') == 0){

        ACC.brainsins.trackClicRecommenderButton(productCode, elementAttr);
                                                founded = true;
                                        }
                                }
                        }
                }
        }


};

$(document).ready(function ()
{
        ACC.cartpopup.bindAll();
}
);
```

11.3.2.17 Código HTML Recomendador

Éste se incluye en la herramienta BrainSINS.

```
<div class="scroller product-carousel-scroller">
    <div class="product-carousel brainsins owl-carousel owl-theme">
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
        <div class="item">
            ${product}
        </div>
    </div>
</div>
```

11.3.2.18 Código HTML Productos Recomendados

```
<div class="prod_grid" id="brainsinsProductResponse_${id}">
    <a class="productMainLink" href=${url}
onclick="BrainSINSTracker.clickId('${url}','${id}');return false;">
        <span class="thumb">
       <img class="lazyOwl" data-src=${image} border="0">
    </span>
    </a>
    <strong class="details">
    <a href=${url} onclick="BrainSINSTracker.clickId('${url}','${id}');return false;">
      ${name}
    </a>
  </strong>
    <div class="estrellita">
        <span class="stars large" style="display: inherit;">
      <span  id="estrellita_br_style_${id}" style="width: 96px;">
      </span>
        <div style="display:none;" itemprop="aggregateRating" itemscope
itemtype="http://schema.org/AggregateRating">
            <span itemprop="ratingValue">
          ${misc.averageRating}
        </span>
            <span itemprop="reviewCount">
          0
        </span>
        </div>
        </span>
    </div>
    <div class="cart">
        <div class="price_container">
            <p id="price_brainsins_${id}" class="price">
                ${price}
            </p>
        </div>
    </div>
    <div id="brainsinsForm_${id}" class="cart">
        <form id="brainsins_updateCartFormByProduct_${id}" name="UpdateQuantityForm"
class="UpdateQuantityForm" action="/compra-online/cart/updatePopupCartQuantityByProduct"
method="post">
            <input type="hidden" class="productCode" name="productCodePost"
value="${id}" />
            <input type="hidden" class="initialQuantity" name="initialQuantity" value=""
/>
            <input type="hidden" class="soldByType" name="soldByType"
value="${misc.soldByType}" />
            <input type="hidden" class="averageRating" name="averageRating"
value="${misc.averageRating}" />
            <input type="hidden" class="unit" name="unit" value="${misc.unit}" />

            <input id="price_{id}" type="hidden" class="price" name="pricebr"
value="${price}" />
            <button id="BrainsinsQuantityProductMinusByProduct_${id}"
class="updateQuantityProductLess" type="button">
        -
      </button>
            <label class="prod_quantity" for="quantityByProduct_${id}">
      </label>
            <input type="text" size="1" id="BrainsinsQuantityByProduct_${id}"
class="qty" name="quantity" value="" />
            <span class="gr" id="brainsins_spangr_${id}" style="display:none">
      </span>
            <button id="BrainsinsQuantityProductPlusByProduct_${id}"
class="updateQuantityProductMore" type="button">
        +
      </button>
            <button id="BrainsinsDeleteQuantityByProduct_${id}"
class="deleteQuantityProduct" type="button">
      </button>
            <button id="BrainsinsAddQuantityByProduct_${id}" class="addQuantityProduct
positive large" type="button">
      </button>
        </form>
    </div>
</div>
```

11.3.3 Extensión diaacceleratatorfacades

11.3.3.1 DiaProductFacade.java

```
/**
 *
 */
package com.b2b2000.dia.facades.product;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.commercefacades.product.ProductFacade;
import de.hybris.platform.commercefacades.product.data.ReviewData;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;

import java.util.List;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;


/**
 * Contrato que amplia el comportamiento de la fachada de productos por defecto.
 *
 * @author <a href="mailto:s.hevia@b2b2000.com>Sandra Hevia Teixeira (SHT)</a>
 */
public interface DiaProductFacade extends ProductFacade
{

	/**
	 * Obtiene el código de bulto mixto de un producto.
	 *
	 * @param productModel
	 *           el producto del cual obtener el código de bulto mixto.
	 * @return el código de bulto mixto si exite y <code>null</code> si el producto
no tiene código de bulto mixto.
	 */
	String getProductMixedPackageCode(final ProductModel productModel);

	/**
	 * Obtiene el peso medio del producto.
	 *
	 * @param productModel
	 *           el producto del cual obtener el peso medio,.
	 * @return el peso medio del producto y <code>null</code> si el producto no tiene
peso medio.
	 */
	Double getProductAverageWeight(final ProductModel productModel);

	/**
	 * Obtiene el peso del producto.
	 *
	 * @param productModel
	 *           el producto del cual obtener el peso .
	 * @return el peso del producto y <code>null</code> si el producto no tiene peso
.
	 */
	Double getProductGrossWeight(final ProductModel productModel);

	/**
	 * Obtiene si el producto es sin gluten.
	 *
	 * @param productModel
	 *           el producto del cual obtener si es sin gluten.
	 * @return <code>true</code> si el producto es sin gluten y <code>false</code> en
caso contrario.
	 */
	Boolean getProductWithoutGluten(final ProductModel productModel);

	/**
	 * Obtiene si el producto es bio.
	 *
	 * @param productModel
```

```
         *           el producto del cual obtener si es bio.
         * @return <code>true</code> si el producto es bio y <code>false</code> en caso
contrario.
         */
        Boolean getProductBio(final ProductModel productModel);


        /**
         * Obtiene las opiniones independientemente del idioma. 20150420 DSA, DIAEC-300,
tiene que recuperar las opiniones
         * introducidas en cualquier idioma
         *
         * @param productCode
         *           the product code
         * @return the reviews that are assigned to specified product
         * @throws UnknownIdentifierException
         *            when product with the given code is not found
         */
        List<ReviewData> getAllReviews(String productCode) throws
UnknownIdentifierException;

        /**
         * Obtiene las primeras X opiniones independientemente del idioma. 20150420 DSA,
DIAEC-300, tiene que recuperar las
         * opiniones introducidas en cualquier idioma
         *
         * @param productCode
         *           the product code
         * @param numberOfReviews
         *           the number of reviews to show, if null shows all reviews, if exceeds
the total number of reviews, shows
         *           all available reviews
         * @return the first X reviews that are assigned to specified product
         * @throws UnknownIdentifierException
         *            when product with the given code is not found
         * @throws IllegalArgumentException
         *            when the quantity of numberOfReviews is negative
         */
        List<ReviewData> getAllReviews(String productCode, Integer numberOfReviews)
throws UnknownIdentifierException,
                        IllegalArgumentException;

        /**
         * SZF, 20151211, [DIAEC-1281] It finds the code of a product's category,
filtering the category by code
         *
         * @param productCode
         *           the code of the product
         * @param categoryCodePrefix
         *           the prefix of category's code
         * @param catalogVersion
         *           SZF, 20151215, [DIAEC-1281] The catalog version
         * @return code of found category
         */
        String findCategoryCodeFromProduct(String productCode, String categoryCodePrefix,
CatalogVersionModel catalogVersion);

        /**
         * SZF, 20151215, [DIAEC-1281] It finds the internationalized description of a
product
         *
         * @param productCode
         *           the code of the product
         * @param isoCode
         *           the language iso code
         * @param catalogVersion
         *           the catalog version
         * @return internationalized description
         */
        String getInternationalizedProductDescription(String productCode, String isoCode,
CatalogVersionModel catalogVersion);

        /**
         * @param product
         * @return
         */
```

```
	ClarelDarkStorePickUpInfoModel getClarelUbicationForProduct(ProductModel
product);


	/**
	 * EMF, 20151007, [DIAEC-464]
	 *
	 * Return all the products from diaSpainProductCatalog with version Online that
are approved
	 *
	 * @return
	 */
	List<DiaProductModel> findProductsToBrainsins();

}
```

11.3.3.2 ExtendedProductFacade.java

```
/**
 *
 */
package com.b2b2000.dia.facades.product.impl;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.classification.ClassificationService;
import de.hybris.platform.classification.features.Feature;
import de.hybris.platform.classification.features.FeatureList;
import de.hybris.platform.classification.features.FeatureValue;
import de.hybris.platform.commercefacades.product.data.ReviewData;
import de.hybris.platform.commercefacades.product.impl.DefaultProductFacade;
import de.hybris.platform.converters.Converters;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.customerreview.model.CustomerReviewModel;

import java.util.List;

import org.springframework.util.Assert;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.product.ExtendedProductService;
import com.b2b2000.dia.facades.product.DiaProductFacade;
import com.ricoh.clarel.core.model.ClarelProductModel;
import com.ricoh.clarel.core.model.ClarelVariantProductModel;


/**
 * Extiende el comportamiento de la fachada por defecto
 *
 * @author <a href="mailto:j.arguello@b2b2000.com>Julio Argüello (JAF)</a>
 */
public class ExtendedProductFacade extends DefaultProductFacade implements
DiaProductFacade
{
/**
	 * EMF, 20151007, [DIAEC-464]
	 *
	 * Return all the products from diaSpainProductCatalog with version Online that
are approved
	 */
	@Override
	public List<DiaProductModel> findProductsToBrainsins()
	{
		return this.getExtendedProductService().findProductsToBrainsins();
	}
…
}
```

11.3.3.3 DiaCustomerFacade.java

```
package com.b2b2000.dia.facades.customer;
```

```
import de.hybris.platform.commercefacades.customer.CustomerFacade;

import de.hybris.platform.commercefacades.user.data.AddressData;
import de.hybris.platform.commercefacades.user.data.CountryData;
import de.hybris.platform.commercefacades.user.data.CustomerData;
import de.hybris.platform.commercefacades.user.data.RegisterData;
import de.hybris.platform.commerceservices.customer.DuplicateUidException;
import de.hybris.platform.commerceservices.customer.PasswordMismatchException;
import de.hybris.platform.commerceservices.customer.TokenInvalidatedException;
import de.hybris.platform.core.model.order.AbstractOrderModel;
import de.hybris.platform.core.model.order.OrderModel;
import de.hybris.platform.core.model.user.UserModel;
import de.hybris.platform.storelocator.model.PointOfServiceModel;

import java.util.Date;
import java.util.List;

import com.b2b2000.dia.core.enums.CrossNavigationRegistrySource;
import com.b2b2000.dia.core.customer.exceptions.CustomerNotFoundException;
import com.b2b2000.dia.core.subscription.exceptions.SubscriptionException;
import com.b2b2000.dia.facades.user.data.DocumentTypeData;
import com.b2b2000.dia.facades.user.data.DtoCustomerData;
import com.b2b2000.dia.facades.user.data.ExtendedCustomerData;
import com.b2b2000.dia.facades.user.data.GenderData;
import com.b2b2000.dia.facades.user.data.RegionData;
import com.b2b2000.dia.facades.user.data.StreetTypeData;


/**
 * Contrato que amplia el comportamiento de la fachada de clientes por defecto.
 *
 * @author <a href="mailto:s.hevia@b2b2000.com>Sandra Hevia Teixeira (SHT)</a>
 */
public interface DiaCustomerFacade extends CustomerFacade
{
	/**
	 * Sets a new Dia% card number.
	 *
	 * @param number
	 *           the number.
	 *
	 * @return the previous value.
	 *
	 * @throws DuplicateUidException
	 *            if user does not match.
	 *
	 * @since 20140227
	 * @see #changeDiaCardNumber(String, String)
	 */
	String changeDiaCardNumber(String number) throws DuplicateUidException;

	/**
	 * Sets a new Dia% card number for the given user.
	 *
	 * @param uid
	 *           the user UID.
	 * @param number
	 *           the new number
	 *
	 * @return the previous value.
	 *
	 * @throws DuplicateUidException
	 *            if user does not match.
	 *
	 * @since 20140227
	 * @see #changeDiaCardNumber(String)
	 */
	String changeDiaCardNumber(final String uid, final String number) throws
DuplicateUidException;

	/**
	 * Obtiene la lista de tipos de países.
	 *
	 * @return la lista de tipos de países.
	 */
	List<CountryData> getAllCountries();
```

```
	/**
	 * Devuelve la lista de provincias en función del país.
	 *
	 * @param isocode
	 *           .
	 * @return .
	 */
	List<RegionData> getAllRegionsByCountry(final String isocode);

	/**
	 * Devuelve la lista de ciudades en función de la provincia.
	 *
	 * @param countryCode
	 *           el país.
	 * @param regionCode
	 *           la provincia.
	 * @return las ciudades.
	 */
	List<RegionData> getAllTownsByRegion(final String countryCode, final String
regionCode);

	/**
	 * Obtiene un usuario por <code>UID</code>.
	 *
	 * @param uid
	 *           el identificador del usuario.
	 * @return el usuario buscado.
	 */
	CustomerData getCustomerForUID(final String uid);

	/**
	 * Obtiene un usuario por <code>UID</code> encriptado.
	 *
	 * @param uid
	 *           el identificador del usuario.
	 * @return el usuario buscado.
	 */
	CustomerData getCustomerForEncryptedUID(final String uid);

	/**
	 * Obtiene la lista de tipos de documentos.
	 *
	 * @return la lista de tipos de documentos.
	 */
	List<DocumentTypeData> getDocumentTypes();

	/**
	 * Obtiene la lista de sexos/géneros.
	 *
	 * @return la lista de sexos/géneros.
	 */
	List<GenderData> getGenderTypes();


	/**
	 * Obtiene la lista de tipos de vía.
	 *
	 * @return la lista de tipos de documentos.
	 */
	List<StreetTypeData> getStreetTypes();

	/**
	 * Actualiza los datos de un usaario de club dia.
	 *
	 * @param customerData
	 */
	void updateProfileAndContactAddress(final ExtendedCustomerData customerData)
throws DuplicateUidException;

	/**
	 * Updates current user groups to take into account prices, taxes and discounts.
	 *
	 * @since 20140205 (JAF)
	 */
	void updateUserGroupsForCalculation();
```

```
        /**
         * Updates current user groups to take into account discounts.
         *
         * @return <code>true</code> if user discount group has changed.
         *
         * @since 20140203 (JAF)
         */
        boolean updateUserGroupsForDiscounts();

        /**
         * Updates current user groups to take into account different prices per store.
         *
         * @return <code>true</code> if user price group has changed.
         *
         * @since 20140203 (JAF)
         */
        boolean updateUserGroupsForPrices();


        /**
         * Updates current user groups to take into account taxes.
         *
         * @return <code>true</code> if user tax group has changed.
         *
         * @since 20140203 (JAF)
         */
        boolean updateUserGroupsForTaxes();


        /**
         * Updates the option that indicates if the user wants to recieve a bill or not.
         *
         * @param likesBill
         *            Desea factura.
         * @param acceptLegalTermsCheckout
         *            Acepta los términos legales de la compra.
         * @param equivalenceSurcharge
         *            Aplicar recargo de equivalencia.
         * @return 20140311 (FPR)
         */
        boolean updateProfilePaymentOptions(Boolean likesBill, Boolean
acceptLegalTermsCheckout, Boolean equivalenceSurcharge);


        /**
         * Registro rápido, o lo que es lo mismo, no genera el evento RegisterEvent que
envía un correo de verificación al
         * usuario.
         *
         * 20140412 FPR.
         *
         * @param registerData
         *            .
         * @throws DuplicateUidException .
         *
         */
        void quickRegister(final RegisterData registerData) throws DuplicateUidException;


        /**
         * Actualiza el email temporal a la espera de que se verifique.
         *
         * @param tempContactEmail
         *            .
         * @param password
         *            .
         */
        void updateEmail(String uid, String tempContactEmail, String currentPassword)
throws PasswordMismatchException;


        void changeUidInsecure(String newUid) throws DuplicateUidException;


        /**
         * Obtiene un usuario por <code>username</code>.
         *
```

```
         * @param username
         *           el username del usuario.
         * @return el usuario buscado.
         */
        CustomerData getCustomerForUsername(final String username);

        /**
         * Método que envía la invitación a un usuario vía Email
         *
         * [DIAEC-1200] , EMF, 20160329 , Added parameters: voucherDiscountValue y
voucherMimOrderRestrictionValue
         *
         * @param invitedUserEmail
         */
        public void sendInvitation(final String invitedUserEmail, final String
invitatorName, final String voucherDiscountValue,
                        final String voucherMimOrderRestrictionValue);

        /**
         * Obtiene un usuario por <code>email</code>.
         *
         * @param username
         *           el username del usuario.
         * @return el usuario buscado.
         */
        CustomerData getCustomerForEmail(final String username);

        /**
         * @param deliveryTimeIntervalPk
         * @param deliveryAddressPk
         */
        boolean updateProfilePaymentFixData(Long deliveryTimeIntervalPk, Long
deliveryAddressPk);

        /**
         * Metodo que nos encripta una password en funcion del tipo de encriptacion del
usuario
         *
         * @param password
         *           String
         * @return String
         */
        public String encodePassword(final String password);

        /**
         * 20150308 GSM - [CLAREL-161] SD207448 - CLAREL_LINK HE OLVIDADO MI CONTRASEÑA
CADUCADO
         *
         * Comprueba si el token de seguridad ha expirado
         *
         * @param token
         *           String
         * @return bboolean
         */
        public boolean isTokenExpired(final String token);

        /**
         * 20150220 GSM - [CLAREL-76] Resetear contraseña: política de contraseñas
         *
         * Metodo que nos devuelve el usuario a partir del token de seguridad que se le
envía para restablecer la password
         *
         * @param token
         *           String
         * @return {@link ExtendedCustomerData}
         * @throws TokenInvalidatedException
         */
        CustomerData getCustomerForToken(final String token) throws
TokenInvalidatedException;

        /**
         * Updates order owner groups to take into account prices, taxes and discounts.
         *
         * @since 20150609 (JRHL)
         */
        void updateOrderGroupsForCalculation(final AbstractOrderModel order);
```

```
        /**
         * @param pos
         * @return
         */
        boolean updateUserGroupsForPrices(PointOfServiceModel pos);

        /**
         * @param order
         * @return
         */
        boolean updateUserGroupsForTaxes(AbstractOrderModel order);

        /**
         * @param pos
         * @return
         */
        boolean updateUserGroupsForDiscounts(PointOfServiceModel pos);

        /**
         * JARB - [CLAREL-812] direcciones - país no España - 20150914
         *
         * @return
         */
        List<CountryData> getDefaultCountry();

        /**
         * (PFA) [Performance]
         *
         * @return
         */
        CustomerData getLightCurrentCustomer();

        /**
         * (GSM), 20151117, [CLAREL-1182] Metodo para actualizar la direccion de envio
por defecto en caso necesario
         */
        void updateDefaultAddressIfNeeded();

        /**
         * (EMF), 20160308, [DIAEC-1837]
         *
         * @return the contact address of the current user
         */
        AddressData getContactAddresssFromCurrentCustomer();

        /**
         * (EMF), 20160308, [DIAEC-1837]
         *
         * @return registry source from current customer
         */
        CrossNavigationRegistrySource getRegistrySourceFromCurrentCustomer();

        /**
         * (EMF), 20160309, [DIAEC-1837]
         *
         * @return uid from the current customer
         */
        String getDisplayUidFromCurrentCustomer();


        /**
         * (EMF), 20160309, [DIAEC-1837]
         *
         * @return last login date from current customer
         */
        Date getLastLoginFromCurrentCustomer();

        /**
         * (EMF), 20160309, [DIAEC-1837]
         *
         * @return clarel last login date from current customer
         */
        Date getClarelLastLoginFromCurrentCustomer();

        /**
         * (EMF), 20160309, [DIAEC-1837]
         *
```

```
         * @return verified contact email from current customer
         */
        String getVerifiedContactEmailFromCurrentCustomer();

        /**
         * (EMF), Brainsins
         *
         * @return warehouseCode in which the customer is situated at this moment
         */
        String getCurrentWarehouseCodeForCurrentCustomer();


        /**
         * SZF, 20160224, [DIAEC-1743] It finds customers whose username, document number
or club DIA card number fits with
         * the specified values
         *
         * @param username
         *            the username or email
         * @param documentNumber
         *            the document number
         * @param diaCardNumber
         *            the number of club dia card
         * @return found customers
         */
        List<DtoCustomerData> searchCustomerForClubDiaCard(String username, String
documentNumber, String diaCardNumber);

        /**
         * SZF, 20160226, [DIAEC-1743] It sends the activation email of an user
         *
         * @param username
         *            the username of the user who will send the activation email
         * @throws CustomerNotFoundException
         *             Throws when any customer were found
         */
        void sendActivationEmail(String username) throws CustomerNotFoundException;

        /**
         * SZF, 20160301, [DIAEC-1743] It activates or deactivates subscription to DIA
newsletter
         *
         * @param username
         *            the username
         * @param likesEmail
         *            the flag to activate or deactivate newsletter
         * @throws SubscriptionException
         *             when there were problems in subscription process
         */
        void subscribeToNewsLetter(String username, boolean likesEmail) throws
SubscriptionException;

        /**
         * Look for a verified user inside parameter list
         *
         * @param users
         * @return true: verified user, false
         */
        boolean validatedUser(final List<UserModel> users);

        /**
         * Update user with status not verified. Update password and send activation
email
         *
         * @param registerData
         */
        void updateTempUser(final RegisterData registerData);

}
```

11.3.3.4 ExtendedCustomerFacade.java

```
package com.b2b2000.dia.facades.customer.impl;
```

```
import de.hybris.platform.commercefacades.customer.impl.DefaultCustomerFacade;
import de.hybris.platform.commercefacades.user.data.AddressData;
import de.hybris.platform.commercefacades.user.data.CountryData;
import de.hybris.platform.commercefacades.user.data.CustomerData;
import de.hybris.platform.commercefacades.user.data.RegisterData;
import de.hybris.platform.commerceservices.customer.CustomerEmailResolutionService;
import de.hybris.platform.commerceservices.customer.DuplicateUidException;
import de.hybris.platform.commerceservices.customer.PasswordMismatchException;
import de.hybris.platform.commerceservices.customer.TokenInvalidatedException;
import de.hybris.platform.converters.Converters;
import de.hybris.platform.converters.Populator;
import de.hybris.platform.core.enums.Gender;
import de.hybris.platform.core.model.c2l.CountryModel;
import de.hybris.platform.core.model.c2l.RegionModel;
import de.hybris.platform.core.model.order.AbstractOrderModel;
import de.hybris.platform.core.model.order.CartModel;
import de.hybris.platform.core.model.user.AddressModel;
import de.hybris.platform.core.model.user.CustomerModel;
import de.hybris.platform.core.model.user.UserModel;
import de.hybris.platform.europe1.jalo.Europe1PriceFactory;
import de.hybris.platform.jalo.JaloSession;
import de.hybris.platform.jalo.SessionContext;
import de.hybris.platform.jalo.enumeration.EnumerationValue;
import de.hybris.platform.jalo.order.OrderManager;
import de.hybris.platform.order.exceptions.CalculationException;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.servicelayer.dto.converter.Converter;
import de.hybris.platform.servicelayer.exceptions.ClassMismatchException;
import de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException;
import de.hybris.platform.servicelayer.i18n.CommonI18NService;
import de.hybris.platform.servicelayer.keygenerator.KeyGenerator;
import de.hybris.platform.servicelayer.model.ModelService;
import de.hybris.platform.servicelayer.search.FlexibleSearchService;
import de.hybris.platform.servicelayer.util.ServicesUtil;
import de.hybris.platform.spring.security.CoreUserDetails;
import de.hybris.platform.store.services.BaseStoreService;
import de.hybris.platform.storelocator.model.PointOfServiceModel;
import de.hybris.platform.task.TaskConditionModel;
import de.hybris.platform.task.TaskModel;
import de.hybris.platform.task.TaskService;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.Date;
import java.util.Iterator;
import java.util.List;

import javax.annotation.Resource;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.collections.functors.NOPTransformer;
import org.apache.commons.collections.functors.NotNullPredicate;
import org.apache.commons.lang.BooleanUtils;
import org.apache.commons.lang.ObjectUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
import org.springframework.util.Assert;

import com.b2b2000.dia.core.catalog.strategies.CatalogConventionStrategy;
import com.b2b2000.dia.core.customer.DiaCustomerAccountService;
import com.b2b2000.dia.core.customer.DiaUserService;
import com.b2b2000.dia.core.customer.exceptions.CustomerNotFoundException;
import com.b2b2000.dia.core.customer.impl.ExtendedCustomerAccountService;
import com.b2b2000.dia.core.enums.CrossNavigationRegistrySource;
import com.b2b2000.dia.core.enums.UserDocumentTypeEnum;
import com.b2b2000.dia.core.jalo.dao.StreetTypeDao;
import com.b2b2000.dia.core.model.StreetTypeModel;
import com.b2b2000.dia.core.model.TownModel;
import com.b2b2000.dia.core.model.user.CustomerModelUtils;
import com.b2b2000.dia.core.ordersplitting.impl.OrderSplittingUtils;
import com.b2b2000.dia.core.security.TrippleDesDecrypter;
import com.b2b2000.dia.core.servicelayer.DiaCommonI18NService;
import com.b2b2000.dia.core.spring.security.OriginalUidOrEmailUserDetailsService;
import com.b2b2000.dia.core.store.DiaLocalStorePreferencesService;
```

```
import com.b2b2000.dia.core.subscription.exceptions.SubscriptionException;
import com.b2b2000.dia.facades.customer.DiaCustomerFacade;
import com.b2b2000.dia.facades.user.data.DocumentTypeData;
import com.b2b2000.dia.facades.user.data.DtoCustomerData;
import com.b2b2000.dia.facades.user.data.ExtendedAddressData;
import com.b2b2000.dia.facades.user.data.ExtendedCustomerData;
import com.b2b2000.dia.facades.user.data.ExtendedRegisterData;
import com.b2b2000.dia.facades.user.data.GenderData;
import com.b2b2000.dia.facades.user.data.RegionData;
import com.b2b2000.dia.facades.user.data.StreetTypeData;
import com.b2b2000.dia.facades.user.data.enums.DocumentType;


/**
 * Extiende el comportamiento de la fachada por defecto ateniéndose a las
características de
 * {@link ExtendedCustomerData} y {@link ExtendedAddressData}.
 *
 * @since 20140204 takes into account dynamic authorities in order to fit to Hybris
pricing mechanisms.
 *
 * @author <a href="mailto:j.arguello@b2b2000.com>Julio Argüello (JAF)</a>
 */
@SuppressWarnings("deprecation")
public class ExtendedCustomerFacade extends DefaultCustomerFacade implements
DiaCustomerFacade
{

	/**
	 * (EMF), Brainsins
	 *
	 * @return current warehouse code in which the customer is situated at this
moment
	 */
	@Override
	public String getCurrentWarehouseCodeForCurrentCustomer()
	{
		final WarehouseModel warehouseModel =
localStorePreferencesService.getPointOfServiceWarehouse();

		return (warehouseModel.getCode() != null ? warehouseModel.getCode() :
null);

	}
…
}
```

11.3.4 Extensión diaacceleratorcore

11.3.4.1 ExtendedProductService.java

```
/**
 *
 */
package com.b2b2000.dia.core.product;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.product.ProductService;

import java.util.List;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.model.PickUpInfoModel;


/**
 * @author Ivan.Garcia.F
 *
```

```
 */
public interface ExtendedProductService extends ProductService
{
	/**
	 * Returns the Product with the specified code belonging to the specified
CatalogVersion.
	 *
	 * @param catalogVersion
	 *           the CatalogVersion of the Product
	 *
	 * @param ean
	 *           the eanPri of the Product
	 *
	 * @return the Product matching the specified code and CatalogVersion.
	 *
	 * @throws de.hybris.platform.servicelayer.exceptions.UnknownIdentifierException
	 *            if no Product with the specified code and Catalog is found.
	 *
	 * @throws
de.hybris.platform.servicelayer.exceptions.AmbiguousIdentifierException
	 *            if more than one Product with the specified code and Catalog is
found
	 */
	ProductModel getProductForEanCode(CatalogVersionModel catalogVersion, String
ean);

	/**
	 * @param product
	 * @param warehouse
	 * @return
	 */
	PickUpInfoModel getPickUpInfoForProductAndWarehouse(ProductModel product,
WarehouseModel warehouse);


	/**
	 * EMF, 20151007, [DIAEC-464] Returns a {@code List<DiaProductModel>} to
Brainsins services that contains all the
	 * products which are approved
	 *
	 * @return The list of products
	 */
	List<DiaProductModel> findProductsToBrainsins();

	/**
	 * [DIAEC-1426] - Integracion Colbenson [CLAREL-1482] - Integracion Contversion
	 *
	 * EMF, 20160201
	 *
	 * @param warehouse
	 * @return
	 */
	List<DiaProductModel> findProductsToGoogleShoppingByWarehouse(final String
warehouse);

	/**
	 * [DIAEC-1426] - Integracion Colbenson [CLAREL-1482] - Integracion Contversion
	 *
	 * EMF, 20160208
	 *
	 * Busca el usergroup que posee más productos
	 *
	 * @return usergroup code
	 */
	String findUserPriceGroupCodeContainsMoreProductsForGoogleShopping();

	/**
	 * SZF, 20151211, [DIAEC-1281] It finds the code of a product's category,
filtering the category by code
	 *
	 * @param productCode
	 *           the code of the product
	 * @param categoryCodePrefix
	 *           the prefix of category's code
	 * @param catalogVersion
	 *           SZF, 20151215, [DIAEC-1281] The catalog version
	 * @return code of found category
```

```
	 */
	String findCategoryCodeFromProduct(String productCode, String categoryCodePrefix,
CatalogVersionModel catalogVersion);

	/**
	 * SZF, 20151215, [DIAEC-1281] It finds the internationalized description of a
product
	 *
	 * @param productCode
	 *           the code of the product
	 * @param isoCode
	 *           the language iso code
	 * @param catalogVersion
	 *           the catalog version
	 * @return internationalized description
	 */
	String getInternationalizedProductDescription(String productCode, String isoCode,
CatalogVersionModel catalogVersion);

	/**
	 * @param product
	 * @return
	 */
	ClarelDarkStorePickUpInfoModel getClarelPickUpInfoForProduct(ProductModel
product);
}
```

11.3.4.2 DefaultExtendedProductService.java

```
/**
 *
 */
package com.b2b2000.dia.core.product.impl;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.product.impl.DefaultProductService;
import de.hybris.platform.servicelayer.util.ServicesUtil;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.beans.factory.annotation.Required;
import org.springframework.util.Assert;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.model.PickUpInfoModel;
import com.b2b2000.dia.core.product.ExtendedProductService;
import com.b2b2000.dia.core.product.dao.ExtendedProductDao;
import com.ricoh.clarel.core.model.ClarelProductModel;
import com.ricoh.clarel.core.model.ClarelVariantProductModel;


/**
 * @author Ivan.Garcia.F
 *
 */
public class DefaultExtendedProductService extends DefaultProductService implements
ExtendedProductService
{
@Resource
	private ExtendedProductDao extendedProductDao;

/**
	 * EMF, 20151007, [DIAEC-464]
	 *
	 * Return all the products from diaSpainProductCatalog with version Online that
are approved
	 */
	public List<DiaProductModel> findProductsToBrainsins()
	{
```

```
                return getExtendedProductDao().findProductsToBrainsins();
        }
…
}
```

11.3.4.3 ExtendedProductDao.java

```
/**
 *
 */
package com.b2b2000.dia.core.product.dao;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.product.daos.ProductDao;

import java.util.List;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.model.PickUpInfoModel;


/**
 * @author Ivan.Garcia.F
 *
 */
public interface ExtendedProductDao extends ProductDao
{
        /**
         * Returns for the given product <code>code</code> the {@link ProductModel}
collection.
         *
         * @param code
         *           the product <code>code</code>
         * @return a {@link ProductModel}
         * @throws IllegalArgumentException
         *            if the given <code>code</code> is <code>null</code>
         */
        List<ProductModel> findProductsByEanCode(final CatalogVersionModel
catalogVersion, final String code);

        /**
         * @param product
         * @param warehouse
         * @return
         */
        PickUpInfoModel getPickUpInfoForProductAndWarehouse(ProductModel product,
WarehouseModel warehouse);

        /**
         * (EMF), 20150710, [DIAEC-464]
         *
         *
         *
         * @return
         */
        List<DiaProductModel> findProductsToBrainsins();

        /**
         * [DIAEC-1426] - Integracion Colbenson [CLAREL-1482] - Integracion Contversion
         *
         * EMF, 20160201
         *
         * Busca el listado de los productos del warehouse que recibe como parametro
según las restricciones indicadas
         *
         * @param warehouse
         * @return lista de productos
         */
        List<DiaProductModel> findProductsToGoogleShoppingByWarehouse(final String
warehouse);
```

```
	/**
	 * [DIAEC-1426] - Integracion Colbenson [CLAREL-1482] - Integracion Contversion
	 *
	 * EMF, 20160208
	 *
	 * Busca el usergroup que posee más productos
	 *
	 * @return el código del usergroup
	 */
	String findUserPriceGroupCodeContainsMoreProductsForGoogleShopping();

	/**
	 * SZF, 20151211, [DIAEC-1281] It finds the code of a product's category,
filtering the category by code
	 *
	 * @param productCode
	 *           the code of the product
	 * @param categoryCodePrefix
	 *           the prefix of category's code
	 * @param catalogVersion
	 *           SZF, 20151215, [DIAEC-1281] The catalog version
	 * @return codes of found category
	 */
	String findCategoryCodeFromProduct(String productCode, String categoryCodePrefix,
CatalogVersionModel catalogVersion);

	/**
	 * SZF, 20151215, [DIAEC-1281] It finds the internationalized description of a
product
	 *
	 * @param productCode
	 *           the code of the product
	 * @param isoCode
	 *           the language iso code
	 * @param catalogVersion
	 *           the catalog version
	 * @return internationalized description
	 */
	String getInternationalizedProductDescription(String productCode, String isoCode,
CatalogVersionModel catalogVersion);

	/**
	 * @param product
	 * @return
	 */
	ClarelDarkStorePickUpInfoModel getClarelPickUpInfoForProduct(ProductModel
product);

}
```

11.3.4.4 DefaultExtendedProductDao.java

```
/**
 *
 */
package com.b2b2000.dia.core.product.dao.impl;

import de.hybris.platform.catalog.model.CatalogVersionModel;
import de.hybris.platform.core.model.product.ProductModel;
import de.hybris.platform.ordersplitting.model.WarehouseModel;
import de.hybris.platform.product.daos.impl.DefaultProductDao;
import de.hybris.platform.servicelayer.search.FlexibleSearchQuery;
import de.hybris.platform.servicelayer.search.SearchResult;

import java.util.Arrays;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.apache.commons.collections.CollectionUtils;
import org.apache.commons.lang.StringUtils;
import org.apache.log4j.Logger;
```

```
import org.springframework.util.Assert;

import com.b2b2000.dia.core.model.ClarelDarkStorePickUpInfoModel;
import com.b2b2000.dia.core.model.DiaProductModel;
import com.b2b2000.dia.core.model.PickUpInfoModel;
import com.b2b2000.dia.core.product.dao.ExtendedProductDao;


/**
 * @author Ivan.Garcia.F
 *
 */
public class DefaultExtendedProductDao extends DefaultProductDao implements
ExtendedProductDao
{

	/**
	 * El nombre del parámetro que representa el nombre del catálogo.
	 */
	private static final String CATALOG_ID = "catalogId";

	// [20160317] DIAEC-1865 error gtin
	private static final String GTIN_INVALID_PREFIX = "0000%";

	/**
	 * SZF, 20150915, [DIAEC-464] Product catalog name
	 */
	private String productCatalog;

	private static Logger LOG = Logger.getLogger(DefaultExtendedProductDao.class);

	/**
	 * EMF, 20151007
	 *
	 * That query return all the products form diaSpainProductCatalog with version
Online that are approved
	 *
	 * This query is necessary for BrainsinsCronjob
	 *
	 */
	private final String APPROVED_PRODUCTS = "SELECT distinct{p.PK}" + "FROM
{DiaProduct AS p"
			+ " JOIN CatalogVersion AS cv ON {p.catalogVersion} = {cv.pk}  AND
{cv.version}='Online' "
			+ "JOIN Catalog AS c ON {cv.catalog} = {c.pk} AND {c.id} IN
('diaSpainProductCatalog') "
			+ "JOIN ArticleApprovalStatus AS a ON {p.approvalstatus} = {a.pk}
AND {a.code} = 'approved' "
			+ "JOIN PriceRow AS priceRow ON {priceRow.product}={p.pk}} "
			+ "WHERE {p.maxOrderQuantity} IS NOT NULL AND {p.thumbnail} IS NOT
NULL";
	/**
	 * EMF, 20151007, [DIAEC-464]
	 *
	 * That query return all the products form diaSpainProductCatalog with version
Online that are approved
	 *
	 * @return the list of products
	 */
	@Override
	public List<DiaProductModel> findProductsToBrainsins()
	{

		final String query = this.APPROVED_PRODUCTS;

		final FlexibleSearchQuery searchQuery = new
FlexibleSearchQuery(query.toString());

		return getFlexibleSearchService().<DiaProductModel>
search(searchQuery).getResult();
	}
…
}
```