UNIVERSIDAD DE ALMERÍA

ESCUELA POLITÉCNICA SUPERIOR

TITULACIÓN DE INGENIERÍA INFORMÁTICA

SISTEMA DE APUESTAS ONLINE PARA DISPOSITIVOS MÓVILES

ALUMNO:
IVÁN RUIZ LEÓN
DIRECTOR:
JOSE ANTONIO PIEDRA FERNÁNDEZ

ÍNDICE

CAPÍTULO I: INTRODUCCIÓN

1.1 INTRODUCCIÓN

La motivación principal para la realización de este proyecto se debe fundamentalmente al auge que están experimentando las apuestas online a nivel mundial, y es por ese motivo que se han implementado sistemas de pago seguros en los portales de apuestas más importantes del mundo. Estos portales, como pueden ser Betfair, Bwin, Ladbrokes, etc., implementan sistemas de pago online como Ukash, Paypal y el que vamos a tratar en este proyecto por considerarlo uno de los más interesantes, Clickandbuy.

Actualmente, los sistemas de pago seguros han visto disparado su uso por parte de los clientes para compras online de todo tipo (ropa, decoración, vehículos, etc), y han llegado con fuerza al mundo de las apuestas. Hoy en día, cualquier portal que se precie tiene no uno, sino varios sistemas de pago online.

Por otro lado, debido a los avances de la tecnología, los apostantes ya no realizan sus apuestas desde sus casas sentados delante de su ordenador personal, sino que en cualquier momento pueden acceder a estos portales estén donde estén, a través de sus teléfonos móviles y PDAs, y realizar sus apuestas. Los dispositivos móviles están teniendo cada vez más importancia dentro del mundo de la informática, debido a que el número de usuarios aumenta progresivamente, y las compañías telefónicas y fabricantes de hardware se ven obligadas a satisfacer los requerimientos de sus clientes incorporando cada vez mayores prestaciones para estos dispositivos. Esto ha provocado que proveedores de sistemas de pago online hayan tenido que adaptar sus portales a estos dispositivos, como en el caso que vamos a tratar. Clickandbuy únicamente era accesible a través de web, pero ha tenido que adaptarse a los tiempos que corren y modificar su sistema para dar servicio a teléfonos móviles y PDAs.

El objetivo de este proyecto se centra en la creación de un portal de apuestas para dispositivos móviles como teléfonos o PDAs y de una pasarela de pago entre este portal de apuestas y un portal de pago online a través de dichos dispositivos móviles, además del acceso mediante servicios web a una API que nos va a proporcionar mercados en los que realizar dichas apuestas. Se va a utilizar un sistema clasificador mediante descriptores para clasificar a los usuarios. Este sistema se va a enmarcar, como cualquier proyecto web, dentro de un sistema cliente/servidor. A su vez, esta aplicación web mantiene una relación cliente/servidor con las APIs de comunicación de Betfair y Clickandbuy mediante de las tecnologías XML y mensajes SOAP. Se ha intentando unificar 3 importantes áreas en la actualidad, como son las apuestas , los sistemas de pago seguros online y la navegación wap.

Objetivos generales:

- Aplicación de la clasificación mediante descriptores a portales dirigidos a los usuarios.
- Estudio de las capacidades multimedia y de acceso a la información de los dispositivos móviles actuales.
- Estudio del funcionamiento de los sistemas de pago online.
- Análisis y utilización de APIs pertenecientes a las casas de apuestas online más importantes del mercado.

A lo largo del presente documento se intenta describir el proceso de estudio y desarrollo de este sistema. Se divide en los siguientes capítulos:

Capítulo I: Es el capítulo que nos ocupa. En él, se hace una descripción del sistema desarrollado y del marco de trabajo en el que se ha llevado a cabo.

Capítulo II: Se hace un repaso a la tecnología móvil actual, así como a la tecnología utilizado para llevar a cabo el desarrollo del sistema. Por otra parte, también se describen los tipos de casas de apuestas actuales y se hace una explicación del sistema de pago online utilizado en este proyecto.

Capítulo III: Una vez que conocemos todo lo necesario, llegamos al desarrollo del sistema. En este capítulo se mostrarán los pasos a seguir para la creación de este tipo de entornos.

Capítulo IV: En este capítulo se realizan las conclusiones acerca del análisis y desarrollo del sistema, y se marcarán las líneas para futuros trabajos relacionados con éste. Se incluyen las referencias bibliográficas utilizadas.

Anexo: Se incluye un manual de usuario de la aplicación explicando el proceso de realización de una apuesta, así como los pasos a seguir para ingresar dinero en la cuenta de usuario del portal y traspasar dinero del portal a la cuenta corriente del usuario.

1.2 ANÁLISIS DE LA APLICACIÓN

El propósito de esta etapa es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema completo, incluyendo la arquitectura. Además, podemos destacar los siguientes objetivos:

- Describir un modelo del sistema.
- Utilizar un lenguaje más formal para explicar detalles relativos a los requisitos del sistema.
- Razonar más sobre los aspectos internos del sistema.
- Estructurar los requisitos de un modo que facilite su comprensión, desarrollo, modificación, y en general, su mantenimiento.

El proyecto parte de una base y unos objetivos previamente definidos. Se quiere desarrollar una aplicación que permita hacer apuestas en el portal Betfair y pagos online para poder realizar dichas apuestas a través de un dispositivo móvil.

1.2.1 PLANTEAMIENTO DEL PROBLEMA

El sistema contará con los siguientes módulos:

- Sistema Betfair: Proveedor de información de eventos, mercados y cuotas. Es un conjunto de servicios web que nos proporcionan toda la información relevante para realizar apuestas en su portal.
- Sistema Clickandbuy: Proveedor del sistema de pago online. Es un conjunto de servicios web al que podemos acceder mediante mensajes SOAP para hacer ingresos en nuestra cuenta de usuario.
- Cliente móvil, que realizará peticiones tanto a un sistema como a otro con el fin de realizar apuestas y hacer transacciones online de dinero.
- Cliente web desde el que se realiza la administración de los usuarios, y a través del cual también se pueden realizar operaciones tanto con Clickandbuy como con Betfair.

De esta manera, el proyecto integra los aspectos clave que se han expuesto con anterioridad, tales como acceso a servicios web, tecnologías móviles, accesos a bases de datos y transmisiones de información entre dispositivos móviles y servidores.

A continuación, se muestra un esquema del sistema y las interacciones que se producen entre los distintos módulos por los que está formado:

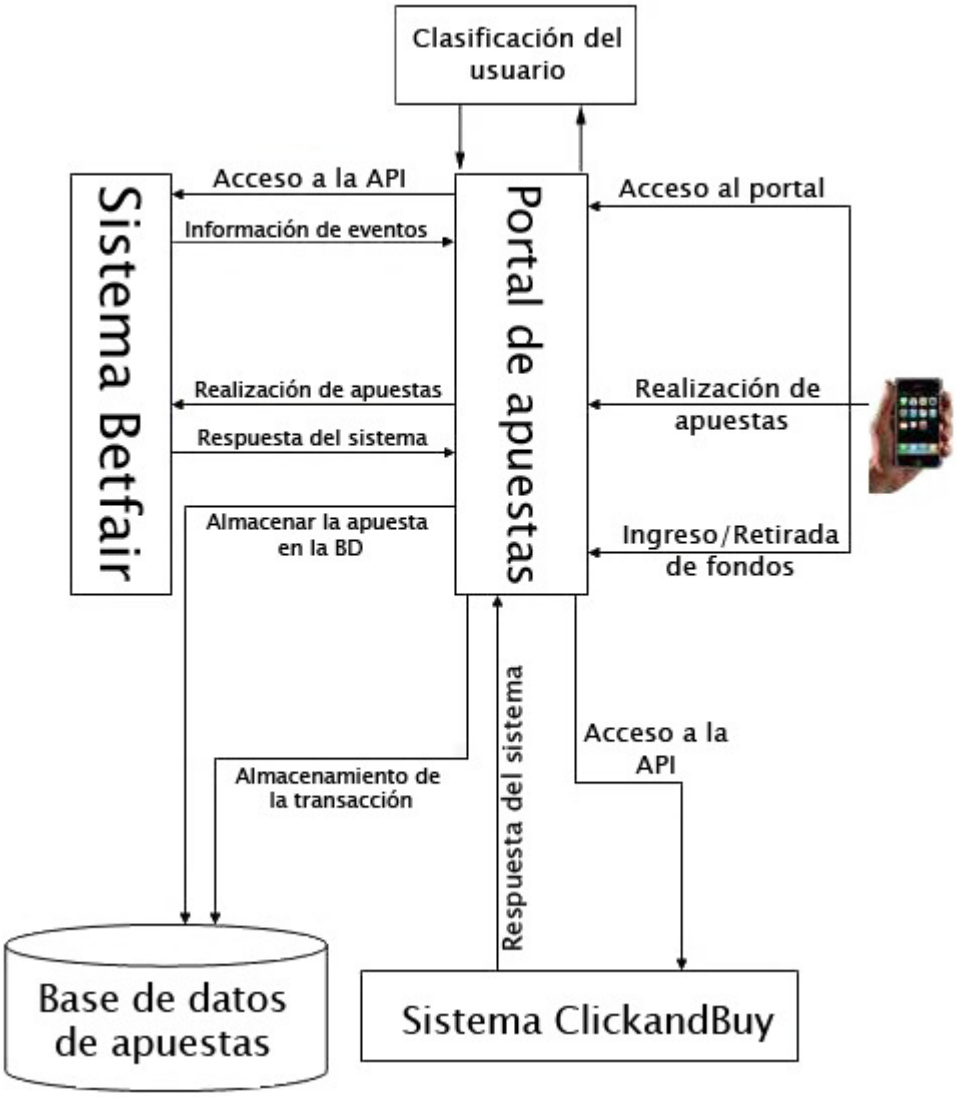

Figura 1. Esquema del sistema de apuestas y transacciones

1.2.2 OBJETIVOS

Como se ha comentado antes, el objetivo de este proyecto se centra en la creación de un portal web para dispositivos móviles, en el que mediante el acceso a servicios web a través de una API que nos va a proporcionar los mercados vamos a poder realizar apuestas en el conocido portal 'Betfair', y mediante una pasarela de pago entre este portal de apuestas y un portal de pago online vamos a poder realizar transacciones de dinero desde nuestra cuenta de dicho sistema de pago a nuestra cuenta de usuario en el portal desarrollado. Por otra parte, otro de nuestros objetivos es obtener, mediante la utilización del sistema WEKA, una clasificación de los usuarios registrados en el portal para poder ofrecerle unos posibles mercados en los que pueden tener interés en apostar.

Este sistema se va a enmarcar, como cualquier proyecto web, dentro de un sistema cliente/servidor. A su vez, esta aplicación web mantiene una relación cliente/servidor con las APIs de comunicación de Betfair y Clickandbuy mediante de las tecnologías XML y

mensajes SOAP. Se ha intentando unificar 3 importantes áreas en la actualidad, como son las apuestas ,los sistemas de pago seguros online y la navegación wap.

Actualmente, el sistema de pago de Clickandbuy sólo oferta pagos a través de web, adaptadas a ordenadores personales o portátiles, por lo que sería muy interesante poder realizar estos pagos a través de servicios web utilizando dispositivos móviles.

1.2.3 METODOLOGÍA

El desarrollo de éste proyecto se subdivide en las siguientes tareas:

- Estudio de la API de Betfair
- Estudio de la comunicación con Clickandbuy
- Estudio del clasificador WEKA
- Diseño de la base de datos: La base de datos se va a realizar en SQLServer.
- Diseño e implementación de la gestión de usuarios: La implementación se va a realizar en C# y javascript, y los datos se van a almacenar en la base de datos que hemos comentado antes.
- Selección de los descriptores que van a definir a un usuario para su posterior clasificación.
- Comunicación con la API de Betfair: El lenguaje utilizado va a ser C#, y se comunicará con los servicios web de Betfair mediante llamadas a funciones definidas en una librería propia (.dll) que tiene acceso a estos servicios web.
- Comunicación con Clickandbuy: Se formatearán los datos a documentos XML en los que se incluirán datos relevantes de la identidad del usuario y del portal, así como accederá a los servicios web de Clickandbuy por medio de los documentos XML realizando llamadas a los servicios web de Clickandbuy.
- Pruebas, revisiones y modificaciones del sistema.

El portal WAP constaría de las siguientes funciones:

1. Inicio: página principal del portal con información de presentación.
2. Registro de usuarios.
3. Boleto: sección principal para la realización de apuestas por parte de los usuarios registrados.
4. Mi Cuenta (para usuarios registrados)

-Perfil: sección para la visualización y modificación de los datos personales, direcciones y acceso del cliente.

-Apuestas actuales: datos sobre las apuestas en vigor.

-Histórico de apuestas: listado de apuestas anteriores y detalle de las mismas.

-Gestión de saldo: sección para el ingreso de saldo en la cuenta del portal y la retirada del mismo mediante sistemas de pago.

5. Quiénes somos: página informativa sobre la empresa.

6. Noticias y artículos: listado de noticias y artículos publicados por el administrador.

7. Ayuda: sección de ayuda sobre el funcionamiento del portal.

8. Contacto: sección con información de contacto.

9. Independientemente de todos los puntos anteriores, cuando un usuario se identifique en el portal, se va a realizar una clasificación de este usuario mediante WEKA para ofrecerle posibles mercados en los que podría mostrar interés en apostar.

El portal WAP se va a desarrollar en .NET, con el lenguaje de programación C#. También vamos a tener una base de datos en SQLServer, en la que vamos a almacenar los datos de usuarios registrados, las apuestas realizadas con sus características, noticias, etc. La clasificación de usuarios se va a realizar mediante WEKA.

Se realizará aparte una gestión para controlar las apuestas realizadas por los usuarios. Esta gestión no tendrá versión WAP, lo que no quiere decir que no sea accesible desde un dispositivo móvil. A esta sección podrá acceder únicamente el administrador del portal con su login y su password. Esta sección estará igualmente implementada en C# y usará tecnología Ajax. Desde ella, se podrán establecer las comisiones de los mercados, consultar ganancias y pérdidas, gestionar noticias, etc. (Esta gestión no formaría parte del proyecto, pero es una parte importante de la aplicación, pues se establecen todas las características de los mercados a los que luego vamos a poder apostar).

1.2.4 ESPECIFICACIONES

Software

- Sistema Operativo Windows Server 2003 Enterprise Edition
- Sistema Operativo Windows Mobile 5.0
- .NET Framework 2.0
- Contenedor de páginas aspx

Hardware

- PC Pentium® 4 2,8Ghz con 512MB de RAM
- PDA Orange SPV M650 con conexión GPRS y USB
- Router
-

1.2.5 MATERIAL UTILIZADO

Para el desarrollo de este proyecto, han sido necesarios una serie de requerimientos tanto hardware como software.

Dependiendo de los requerimientos hardware, hemos tenido que contar con la utilización de:

- Un PC Pentium 4 a 2,8 GHz con 512 MB de memoria RAM. El Proyecto se podría haber desarrollado con un ordenador de menor potencia, pero para el correcto funcionamiento de la aplicación y la utilización de las aplicaciones para la realización de la misma, se ha considerado que era mejor esta opción.
- PDA Orange SPV M650 con conexión GPRS y USB, con acceso a internet.

También han sido necesarios los siguientes requisitos software:

- IIS, como motor del servidor donde esta alojada la aplicación.
- Visual Studio .NET 2005: Entorno de desarrollo para la plataforma C# que nos permite un desarrollo eficiente.

- Sistema operativo Windows XP.

- MSXML 4.0: MicrosoftXML Core Services (MSXML) permite a los clientes que usan Jscript, Visual Basic Scripting Edition (VBScript) y Microsoft Visual Studio 2005 construir aplicaciones de alto rendimiento basadas en XML que proporcionan un alto grado de interoperabilidad con otras aplicaciones que se adhieren a XML 1.0 estándar. Se podría haber utilizado la librería WebClient propia de Visual Studio, pero en los ejemplos enviados por el soporto técnico de Clickandbuy la recomendación fue utilizar esta.

- Firefox 3.0, Internet Explorer 6 y 7: Navegadores web utilizados en el sistema operativo Windows.

- Internet Explorer Mobile: Navegador web para dispositivos móviles.

- BetfairAPI.dll: Librería que va permitir el acceso a las funciones destinadas a recoger los datos que proporcionan los servicios web de clickandbuy.

- SQLServer: Gestor de base de datos compatible con Visual Studio 2005.

- Microsoft Activesync 4.5: Es un software de sincronización para dispositivos con windows mobile.

- WEKA, como el sistema utilizado para el análisis de datos para la clasificación.

1.2.6 PLAN DE TRABAJO

El proyecto se divide en distintas partes que se detallan a continuación:

- Estudio de la API de Betfair: Estudio de las funciones proporcionadas por Betfair para obtener información de sus servicios web.

- Estudio de la comunicación con clickandbuy: Estudio de la metodología de comunicación con clickandbuy y de los pasos a seguir para obtener las respuestas del sistema.

- Diseño de la base de datos donde almacenar transacciones, usuarios, apuestas, etc.

- Selección de los descriptores que van a definir a un usuario para su posterior clasificación.

- Gestión de usuarios: Registro y almacenamiento de los usuarios, asi como de los movimientos que realicen a través de Betfair tales como apuestas, cancelaciones, etc. También se almacenarán las transacciones realizadas con clickandbuy

- Gestión de los servicios de Betfair: Obtención de los eventos, mercados y cuotas que se van a proporcionar para la realización de las apuestas.

- Comunicación con clickandbuy: Construcción de los mensajes SOAP de envió de datos para las transacciones de dinero, y recepción de la respuesta de clickandbuy para el tratamiento de la transacción.

- Pruebas, revisiones y modificaciones, con el fin del correcto funcionamiento del sistema.

- Generación de la memoria en la que se explica lo que se ha realizado de un modo detallado.

Tarea	Tiempo estimado	Tiempo total
Estudio de la API de Betfair	1 semana	2 semanas
Estudio de la comunicación con Clickandbuy	2 semanas	2 semanas
Diseño de la base de datos	2 días	7 días
Gestión de usuarios	30 días	30 días
Gestión de los servicios de Betfair	13 días	13 días
Comunicación con Clickandbuy	25 días	25 días
Clasificador Bayesiano	25 días	30 días
Pruebas, revisiones y modificaciones	7 días	14 días
Generación de la memoria	30 días	60 días

Tabla 1. Estimación de tiempos

CAPÍTULO II: CONCEPTOS TEÓRICOS Y ESTADO DEL ARTE

2.1 TECNOLOGÍA MÓVIL

2.1.1 INTRODUCCIÓN

El término tecnología inalámbrica define la opción de varios dispositivos de conectarse entre sí sin necesidad de cables. También podemos conectar dispositivos a una red, utilizando frecuencias de radio, infrarrojos, etc. Las plataformas inalámbricas han crecido a lo largo de la historia, y hoy por hoy son una gran industria, llevando las comunicaciones a cualquier lugar del mundo.

Entre los usos mas comunes se incluyen la IrDA (Infrared data asociation), que definen un estándar físico en la forma de transmisión y recepción de datos por rayos infrarrojos, y las redes inalámbricas de computadoras, incorporando esquemas de conectividad como Wifi, GPRS, Bluetooth, etc. Las conexiones inalámbricas que se establecen entre los clientes remotos y una red proporcionan a las empresas flexibilidad y un adecuado abanico de beneficios asociados a la captura, procesamiento y análisis de información en tiempo real.

La "Tecnología móvil" y "Tecnología inalámbrica" se puede definir sin relacionarse entre si. Hay varios dispositivos móviles, como por ejemplos las PDAs (personal digital assistant), en los que se puden utilizar aplicaciones, obtener datos, manejar documentos, etc sin necesidad de utilizar tecnología inalámbrica para transmitir información. Se puede decir que la tecnología móvil hace referencia a la posibilidad de llevarnos el trabajo de un sitio a otro, es decir, de llevar a cabo unas tareas determinadas lejos de nuestro sitio habitual. En cambio, la tecnología inalámbrica hace referencia a la posibilidad de conectar varios dispositivos entre sí o a una red sin necesidad de cables. Se pueden emplear estas conexiones inalámbricas para transferir la información entre una red de empresa, donde un grupo de personas necesitan estar comunicadas entre si.

De cualquier manera, cuando ambas tecnologías se juntan, generan nuevos servicios, nuevos medios, nuevos mercados, nuevos entornos de aplicaciones, nuevos entornos de transacciones y sobre todo una gran red de información disponible en todo momento y lugar.

La velocidad de transferencia de datos se mide en Mbps. Un Mbps es un millón de bits por segundo, o la octava parte de un MegaByte por segundo (Mbps). Un byte son 8 bits. Es necesario tener presente esta diferencia, ya que se pude confundir fácilmente el término Mbit con Mbyte y la diferencia es muy grande. En comunicaciones se emplea el término Mbit.

Las tecnologías móviles e inalámbricas desarrollan el concepto "Productividad". El vertiginoso avance de las tecnologías de Hw, Sw y telecomunicaciones ha provocado que se

genere un nuevo subsector dentro de las TIC (Tecnologías de la información y la comunicación), que está gestando una nueva generación de sistemas o herramientas tecnológicas de gran impacto en el sector corporativo. Se trata de las soluciones basadas en tecnologías móviles e inalámbricas, lo que hoy en día se convierte en sinónimo de productividad.

Adaptando este nuevo escenario, el término "Tecnología móvil" hace referencia a la posibilidad de trasladar una actividad determinada, que normalmente se inscribe en un espacio físico, de un sitio a otro.

Por ejemplo, en el entorno de las empresas, la tecnología móvil permite realizar una tarea determinada sin estar presente en la oficina, y efectuar un gran número de actividades gracias a la aparición de nuevos dispositivos de pequeño tamaño que ofrecen la posibilidad de ser transportados y utilizados durante este transporte, con gran capacidad de almacenamiento y procesamiento de datos.

Esto proporciona nuevas posibilidades en el mercado de soluciones de automatización, que además de garantizar la portabilidad de los datos, permite el tráfico de información entre el usuario que se encuentra realizando una actividad remota (ventas, reparaciones, etc) y la empresa para la que trabaja, enviando información relevante en tiempo real que anteriormente solo podíamos encontrar en los sistemas centrales de información.

La mayoría de las empresas tienen comerciales que salen a la calle con dispositivos que les permiten realizar de manera ágil su trabajo, como dispositivos móviles seleccionados de acuerdo a criterios específicos (teléfonos móviles, PDAs...), que les permiten consultar, obtener o procesar información relevante. Estos comerciales siempre tendrán información actualizada de las rutas asignadas, productos y cartera de clientes en tiempo real, pedidos, últimos periodos, balances, y otro tipo de información que les permitirán hacer en menos tiempo su trabajo. Su herramienta enviará información en tiempo real a los sistemas de la empresa, mediante varios tipos de conectividad, garantizando la seguridad de los datos desde que se envía por el trabajador hasta que se recibe en la empresa. [nodrizza]

Tecnología WiMAX

WiMAX, Worldwide Interoperability for Microwave Access, es la marca que certifica que un producto está conforme con los estándares de acceso inalámbrico 'IEEE 802.16'. Estos estándares proporcionan conexiones de velocidades similares al ADSL sin cables, y hasta una distancia de aproximadamente 50km. Este nuevo estándar será compatible con otros anteriores, com el de Wi-Fi (IEEE 802.11).

La tecnología WiMAX será la base de las conexiones a internet en las ciudades y servirá de apoyo en las zonas rurales que ahora tiene dificultad para el acceso a internet. Además, se utilizará en las empresas para las conexiones internas.

Se espera también que sirvan como punto de apoyo al despegue definitivo de otras tecnologías como VoIP (llamadas de voz sobre el protocolo IP).

Para promover el uso de los estándares WiMAX se necesita que los fabricantes de dispositivos electrónicos utilicen esta tecnología y se establezcan unas normas que aseguren la compatibilidad de antenas, procesadores y receptores. Para esto existe el "WiMAX Forum", que es una asociación sin ánimo de lucro formada por varias empresas comprometidas con el cumplimiento del estándar IEEE 802.16

En la actualidad, existen dos tipos principales de tecnología WiMAX:

- WiMAX fija (802.16d-2004), es una tecnología de punto a varios puntos. El estándar del 802.16-2004 del IEEE fue diseñado para el acceso fijo. Este estándar es el que se conoce como "fijo inalámbrico" porque usa una antena que se coloca en un lugar estratégico para la recepción por parte del cliente. La antena se ubica generalmente zonas altas, y son parecidas a las antenas de televisión satélite. También se utiliza en interiores, y en este caso no necesita ser tan robusto como al aire libre. El estándar 802.16-2004 se utiliza para el acceso inalámbrico a internet de banda ancha. WiMAX fijo funciona a 2.GHz, 3.5GHz y 5.8GHz. Esta tecnología proporciona una alternativa inalámbrica al módem cable y a las xDSL.

- WiMAX móvil (802.15e-2005), se basa en la tecnología OFDMA (acceso múltiple por división de frecuencia ortogonal) que ofrece ventajas si hablamos en términos de latencia, eficiencia en el uso del espectro de frecuencia de radio y soporte avanzado de antenas, lo que supone un rendimiento superior al de las actuales tecnologías de redes inalámbricas. En cuanto a las tecnologías inalámbricas 4G de próxima generación, están evolucionando hacia OFMDA y redes IP ya que son ideales para servicios inalámbricos de datos a unos precios razonables. [blogwimax]

Tecnología WiBRO

WiBro (Wireless Broadband - Banda ancha inalámbrica) permite conexiones de hasta 30Mbps con zonas de cobertura de entre 1 y 5 kilómetros aunque nos estemos desplazando a una velocidad de unos 60 Km/h, utilizando una banda de frecuencias de 2.3 Ghz. WiBro es las solución ideal para móviles.

Se puede considerar WiBro como una extensión de WiMax, ya que ambos están basados en el estándar IEE 802.16, pero existen ciertas diferencias entre ellos, y sobre todo existen diferencias muy acusadas con WiFi.

WiFi tiene una serie de deficiencias de sobras conocidas, como pueden ser la cobertura, que es de un entorno de alrededor de 300 metros y la calidad del servicio, que dificulta su utilización en aplicaciones de retransmisión de video de alta calidad en tiempo real.

Por su parte, WiMax y WiBro garantizan calidad de servicio, aunque en cuestiones de cobertura WiMax puede alcanzar los 50km teóricos que se suelen quedar en 10km reales y WiBro nos permite velocidades de 1Mbsp aunque nos movamos a 60km/h, aunque la cobertura se ve reducidad a menos de 1km.

Esto es debido a que WiMax se centra en el alcance de la señal y WiBro en una transmisión de calidad sin cortes. [domodesk]

2.1.2 DISPOSITIVOS MÓVILES

La evolución del teléfono móvil ha permitido disminuir su tamaño y peso, desde el Motorola DynaTAC, el primer teléfono móvil, que apareció en el año 1982 y que pesaba 780 gr., a los móviles más compactos y con mayores prestaciones. El desarrollo de baterías más pequeñas y de mayor duración, pantallas mucha mayor resolución y la incorporación de un software amigable han hecho del teléfono móvil un elemento primordial en la vida moderna.

Los avances tecnológicos han permitido que los dispositivos móviles actuales incorporen juegos, reproductores de música y radio, email, cámaras fotográficas y de video, internet e incluso televisión digital terrestre que los han convertido en un dispositivo de entretenimiento.

Con la aparición de la telefonía móvil digital, fue posible acceder a páginas de Internet especialmente diseñadas para móviles, lo que se conoce como tecnología WAP. Las primeras conexiones se efectuaban mediante una llamada telefónica a un número del operador a través de la cual se transmitían los datos de manera similar a como lo haría un módem de PC.

Los móviles disponen de un microprocesador que realiza cálculos a gran velocidad. Este procesador se llama "DSP" (Digital Signal Processor), y realiza toda la compresión y descompresión de los datos a la velocidad de 40 millones de instrucciones por segundo (MIPS).

Para realizar las comunicaciones, la operadora de telefonía divide las áreas en células, normalmente con una topología hexagonal, y en cada célula hay una antena, que utiliza varias decenas de canales, lo que posibilita que varias personas se comuniquen a la vez. Cuando una persona se encuentra lejos del alcance de un área o célula, deja libre su canal y coge un canal de la siguiente célula. Es por esto que la comunicación móvil también es llamada comunicación celular.

Los sistemas actuales de telefonía móvil son:

- GSM, Global system for mobile o sistema global para las comunicaciones móviles, es el sistema más utilizado y el estándar europeo. Soporta voz, datos, mensajes de texto y roaming en varios países. No soporta IP (Internet Protocol), lo que impide el acceso directo a internet.

- GPRS, General packet radio service, es un servicio que permite enviar paquetes de datos a través de las redes GSM. Por paquetes de datos entendemos información que se divide en varios paquetes y que pueden ser enviados por distintos canales

obteniendo un aprovechamiento más efectivo de los canales de transmisión. Al basarse en la transmisión de datos por paquetes (IP), proporciona una mayor velocidad de transmisión, acceso instantáneo a internet, conexión permanente, etc.

- UMTS, Universal mobile telecomunications system, es el estándar que se emplea en la llamada tercera generación de telefonía móvil (3G), que permite disponer de banda ancha en el móvil y enviar paquetes de información grandes a través de la red. Con los móviles 3G es posible la conexión a internet de banda ancha, videoconferencia, descarga de vídeos, etc. Proporciona compatibilidad entre los sistemas GSM y UMTS, debiendo los móviles incorporar "dual band".

Se está empezando a experimentar con la tecnología 4G, que estará totalmente basada en IP.

2.2 TECNOLOGÍA WEB: XML/SOAP, ASP.NET, SERVICIOS WEB

En este apartado se van a explicar las distintas tecnologías que se han utilizado para llevar a cabo el proyecto.

2.2.1 XML/SOAP

2.2.1.1 DEFINICIÓN

XML son las siglas, en inglés, de eXtensible Markup Language (Lenguajes de marcas ampliable). Es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML (Standard Generalized Markup Language) y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML (eXtensible Hypertext Markup Language), SVG (Scalable Vector Graphics) y MathML (Mathematical Markup Language).

XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.

Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

Proviene de un lenguaje inventado por IBM en los años setenta, llamado GML (Generalized Markup Language), que surgión por la necesidad que tenía la empresa de almacenar grandes cantidades de información. Este lenguaje gustó a la ISO, por lo que en 1986 trabajaron para normalizarlo, creando SGML, capaz de adaptarse a un gran abanico de problemas. A partir de él se han creado otros sistemas para almacenar información.

En el año 1989 Tim Berners Lee creó la web, y junto con ella el lenguaje HTML. Este lenguaje se definió en el marco de SGML y fue de lejos la aplicación más conocida de éste estándar. Los navegadores web, sin embargo, siempre han puesto pocas exigencias al código HTML que interpretan y así las páginas web son caóticas y no cumplen con sintaxis. Éstas páginas web dependen fuertemente de una forma específica de lidiar con los errores y las ambigüedades, lo que hace a las páginas más frágiles y a los navegadores más complejos.

Otra limitación del HTML es que cada documento pertenece a un vocabulario fijo, establecido por el DTD (Definición de tipo de documento). No se pueden combinar elementos de diferentes vocabularios. Asimismo, es imposible para un intérprete (por ejemplo, un navegador), analizar el documento sin tener conocimiento de su gramática (del DTD). Por ejemplo, el navegador sabe que antes de una etiqueta ** ** debe haberse cerrado cualquier **<p>** previamente abierto. Los navegadores resolvieron esto incluyendo lógica ad hoc para el HTML, en vez de incluir un analizador genérico. Ambas opciones, de todos modos, son muy complejas para los navegadores. Se buscó entonces definir un subconjunto del SGML que permita:

- Mezclar elementos de diferentes lenguajes, es decir, que los lenguajes sean extensibles.
- La creación de analizadores simples, sin ninguna lógica especial para cada lenguaje.
- Empezar de cero y hacer hincapié en que no se acepte nunca un documento con errores de sintaxis.

Para hacer esto, XML deja de lado muchas características de SGML que estaban pensadas para facilitar la escritura manual de documentos. XML, en cambio, está orientado a hacer las cosas más sencillas para los programas automáticos que necesiten interpretar el documento.

Ventajas del XML

- **Es extensible**: después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
- **El analizador es un componente estándar**: no es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan bugs y se acelera el desarrollo de aplicaciones.
- **Estructura**: si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla. Mejora la compatibilidad entre aplicaciones.

2.2.1.2 ESTRUCTURA DE UN DOCUMENTO XML

La tecnología XML se basa en dar solución al problema que surge cuando queremos presentar información estructurada de la manera más abstracta y reutilizable posible. Información estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen a su vez de otras partes. Se obtiene así un árbol de información. Estas partes se llaman elementos y se las señala mediante etiquetas

Una etiqueta es una marca hecha en el documento, que señala una parte de éste como un elemento, como un trozo de información con un sentido claro y definido. Las etiquetas son de la forma **<nombre>**, donde *nombre* es el nombre del elemento. A continuación se muestra un ejemplo para entender la estructura de un documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
        <!DOCTYPE Edit_Mensaje SYSTEM "Lista_datos_mensaje.dtd" [<!ELEMENT Edit_Mensaje (Mensaje)*>]>
        <Edit_Mensaje>
                <Mensaje>
                        <Remitente>
                                <Nombre>Nombre del remitente</Nombre>
                                <Mail> Correo del remitente </Mail>
                        </Remitente>
                        <Destinatario>
                                <Nombre>Nombre del destinatario</Nombre>
                                <Mail>Correo del destinatario</Mail>
                        </Destinatario>
                        <Texto>
                                <Asunto>
                                        Este es mi documento con una estructura muy sencilla no contiene atributos ni entidades....
                                </Asunto>
                                <Parrafo>
                                        Este es mi documento con una estructura muy sencilla no contiene atributos ni entidades....
                                </Parrafo>
                        </Texto>
                </Mensaje>
        </Edit_Mensaje>
```

Tabla 2. Ejemplo documento XML

Aquí está el ejemplo de código del DTD del documento "Edit_Mensaje":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
        <!-- Este es el DTD de Edit_Mensaje →
        <!ELEMENT Mensaje (Remitente, Destinatario, Texto)*>
                <!ELEMENT Remitente (Nombre, Mail)>
                        <!ELEMENT Nombre (#PCDATA)>
                        <!ELEMENT Mail (#PCDATA)>
                <!ELEMENT Destinatario (Nombre, Mail)>
                        <!ELEMENT Nombre (#PCDATA)>
                        <!ELEMENT Mail (#PCDATA)>
                <!ELEMENT Texto (Asunto, Parrafo)>
                        <!ELEMENT Asunto (#PCDATA)>
                        <!ELEMENT Parrafo (#PCDATA)>
```

Tabla 3. Ejemplo de código del DTD del documento "Edit Mensaje"

Los documentos denominados como “bien formados” son los que cumplen estrictamente con todas las definiciones básicas de formato y pueden ser analizados de forma correcta por un analizador sintáctico (*parser*) que cumpla con las normas siguientes:

- Tienen que tener una estructura jerárquica respecto a las etiquetas que definen todos sus elementos. Una etiqueta debe estar correctamente incluida en otra de manera anidada y además deben estar correctamente cerrados. A continuación se muestra un ejemplo incorrecto y posteriormente otro ejemplo escrito correctamente:

  <li>HTML ****permite **** esto ** ** → No se cierra

  **
En XML la ** estructura **** es **** jerárquica **.
** → Correcto

- Los documentos XML deben tener un único elemento raíz del que todos los demás sean cuelguen, es decir, sólo pueden tener un elemento inicial. La jerarquía de elemento de un documento XML bien formado sólo puede tener un elemento inicial.

- Los valores de los atributos en XML siempre deben estar encerrados entre comillas simples o dobles.

- El XML es sensible a mayúsculas y minúsculas. El conjunto de caracteres conocidos como espacios en blanco (espacios, tabuladores, retornos de carro, saltos de línea) se tratan de forma diferente en el marcado XML.

- El uso de los espacios de nombres en los documentos XML permite que dos elementos distintos tengan el mismo nombre siempre que cada uno pertenezca a un espacio de nombres distinto. Para ello utilizan los URI (*Uniform Resource Identifiers*), que garantiza que los recursos a los que se hace referencia sean únicos.

- Es necesario asignar nombres a las estructuras, tipos de elementos, entidades, elementos particulares, etc. En XML los nombres tienen alguna característica en común.

- En XML 1.0 se definen cinco entidades para representar caracteres especiales y que no se interpreten como marcado en el procesador XML, es decir, que así podemos usar el carácter “<” sin que se interprete como el comienzo de una etiqueta XML. Por ejemplo:

ENTIDAD	CARÁCTER
&	&
<	<
>	>
'	'
"	“

Tabla 4. Ejemplos de entidades utilizadas para representar caracteres especiales

- Las construcciones como etiquetas, referencias de entidad y declaraciones se denominan marcas. Son partes del documento que el procesador XML espera entender. El resto del documento entre marcas son los datos “entendibles” por las personas. HTML permite elementos sin contenido. XML también, pero la etiqueta debe ser la de la siguiente forma: **<elemento sin contenido />**. A continuación se muestra un ejemplo incorrecto y posteriormente otro correcto.

XML hace referencia a objetos que no deben ser analizados sintácticamente según las reglas XML, mediante el uso de entidades. Las entidades pueden ser:

- Internas o externas
- Analizadas o no analizadas
- Generales o parametrizadas

<*li>Esto es HTML
 en el que casi todo está permitido *

*<li>En XML, es
 más restrictivo. *

- A veces es conveniente insertar comentarios en documento XML, que son ignorados por el procesado de la información y las reproducciones del documento. Los comentarios tienen el mismo formato que los comentarios de HTML. Es decir, comienza por la cadena **“<!--”** y termina con **“-->”**.

<!-- Esto es un comentario-->

Se pueden introducir comentarios en cualquier parte del documento salvo dentro de las declaraciones, etiquetas u otros comentarios.

- Existe otra construcción en XML que permite especificar datos, utilizando cualquier carácter, especial o no, sin que se interprete como marcado XML. La razón de esta construcción llamada CDATA (Character DATA) es que a veces es necesario para los autores de documentos XML, poder leerlo fácilmente sin tener que descifrar los códigos de entidades, especialmente cuando son muchos.

Como ejemplo, el siguiente (primero usando entidades predefinidas y luego con un bloque CDATA).

```
<parrafo>Lo siguiente es un ejemplo de HTML.</html>
<ejemplo>
&lt;html>
&lt;head>&lt;title>Rock &amp; Roll&lt;/title>&lt;/head>
</ejemplo>

<ejemplo>
<![CDATA[
<html>
<head><title>Rock & Roll</title></head>
]]>
</ejemplo>
```

Tabla 5. Ejemplo comentarios XML

Como hemos visto, dentro de una sección CDATA podemos poner cualquier cosa, que no será interpretada como algo que no es así. Existe una excepción y es la cadena **"]]>"** con el que termina el bloque CDATA. Esta cadena no puede utilizarse dentro de una sección CDATA.

([el rincón del programador][geocities][ulgpc][mitecnologico][w3c][webservices][w3][gavd])

2.2.1 ASP.NET (ASPX)

En este apartado se pretende explicar que es la tecnología aspx y como consecuencia inmediata a definir, se procederá a definir los diferentes elementos que forman una página de estas características.

2.2.2.1 DEFINICIÓN DE PÁGINA ASPX

ASP (*Active Server Pages)* es una tecnología que impulsada por Microsoft hace ya varios años y que actualmente es uno de los lenguajes para programación de aplicaciones web más utilizados. Funcionan sobre servidores Microsoft con Internet Information Server (IIS) para Windows NT ó 2000, y en caso de tener sistemas operativos mas antiguos como pueden ser Windows 95 o 98 se utiliza un servidor web personal como el Personal Web Server.[uco]

Una página web aspx es una página realizada en ASP.net, el cual forma parte del .NET Framework de Microsoft, junto con VB.net, C++.net, C# (que lo podemos considerar como una versión mejorada de C++) hasta incluso JScript.net. ASPX es un formato de archivo HTML usado para crear los llamados "Webforms" o formularios web. Un archivo aspx típico contiene HTML estático o etiquetas XHTML, de manera que estas etiquetas definen controles web y controles de usuario web donde se inserta todo el contenido estático y dinámico de la página web. Además, el código dinámico que se ejecuta en el servidor puede ser incluido en un página utilizando la etiqueta <% -- código dinámico --%>, a imagen y semajanza de otras tecnologías de desarrollo web como PHP, JSP y ASP.

Las recomendaciones de Microsoft para tratar con código dinámico se basan en usar el modelo de código subyacente. Esto significa que se coloca el código en un archivo separado o en una etiqueta especialmente colocada para ello. Los archivos de código subyacente son normalmente llamados **Pagina.aspx.cs** o **Pagina.aspx.vb**, según el lenguaje de programación, y cuyo nombre se basa en el nombre del archivo ASPX.

Se utiliza para construir sitios web dinámicos, aplicaciones web y servicios web XML. Fue lanzado al mercado en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología ASP. ASP.NET esta construido sobre el Common Language Runtime, permitiendo escribir código en ASP.net usando cualquier lenguaje admitido por el .NET Framework. Una característica fundamental es que es un lenguaje totalmente orientado a objetos.

ASP.Net es una plataforma basada en servidor, que significa que el código se ejecuta en el servidor web y no en el explorador del cliente. Esto garantiza que el código esta protegido de intrusos y de incompatibilidades en los navegadores de los usuarios, pero a su vez introduce algunas limitaciones.

.NET Framework para aplicaciones web es una estructura de soporte para apoyar el desarrollo de sitios web dinámicos, aplicaciones web y servicios web. Engloba toda la plataforma .NET. Es la evolución de las API de Windows y de la fundación de clases de Microsoft (MFC Microsoft Foundation Class) dirigidas a lograr una completa encapsulación. Cualquier programa que desarrollemos utilizando la tecnología .net, independientemente del lenguaje utilizado, estará basado en .NET Framework.

La forma más sencilla de entender esto es un ejemplo. Si tenemos un menú de cualquier plicación Windows que nos permite acceder a las distintas opciones del programa, en tecnología .net estos menús son una clase perteneciente al .NET Framework. Esta clase es siempre la misma independientemente del lenguaje y entorno en el que nos encontremos, y la forma de trabajar con ella es la misma desde C#, VB.net o cualquier lenguaje .net. Lo mismo ocurre con los botones, las imágenes, la forma de leer los archivos, etc. Todo lo que hagamos en un lenguaje .NET está basado en el .NET Framework.

Después del lanzamiento de IIS 4.0 en 1997, Microsoft comenzó a investigar las opciones para un nuevo modelo de aplicaciones web que resolviera las quejas que se tenían sobre ASP, sobre todo con respecto a la separación de la presentación y el contenido y ser capaz de escribir código "limpio".

La primera versión se llamó XSP, y fue desarrollado por Mark Anders y Scott Guthrie. El desarrollo inicial de XSP se realizó en Java, pero rapidamente se decidió construir una nueva plataforma sobre el Common Language Runtime (CLR), que, al igual que XSP, aún estaba en las etapas iniciales de desarrollo.

Con el cambio al CLR, XSP fue implementado en C#, y renombrado a ASP+. La primera demostración pública y la liberación de la primera beta de ASP+ (y del resto del .NET Framework), se realizó en la Microsoft's Professional Developers Conference el 11 de Julio del 2000 en Orlando.

Poco después se cambió el nombre de ASP+ a ASP.NET. Después de 4 años de desarrollo y una serie de versiones de evaluación, ASP.NET 1.0 fue liberado el 5 de Enero de 2002 como parte de la versión 1.0 del .NET Framework, seguida de la versión 1.1 que fue liberada el 24 de Abril de 2003 como parte de Windows Server 2003.

.NET sería la respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystem y a los distintos Frameworks de desarrollo web basados en PHP. Su propuesta es ofrecer una manera rápida y económica, a la vez que segura y robusta, de desarrollar aplicaciones permitiendo una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier dispositivo.

La plataforma .NET de Microsoft es un componente software que puede ser instalado bajo un entorno Windows. NET Framework se incluye en Windows Server 2008 y en Windows Vista. También puede ser instalado en Windows XP, y en la familia de sistemas operativos Windows Server 2003. También existe una versión "reducida" para la plataforma Windows Mobile.

A largo plazo, Microsoft pretende reemplazar la API Win32 por la plataforma .NET. Esto se debe a que la API de Windows fue desarrollada sobre la marcha y provoca múltiples problemas en el desarrollo de aplicaciones bajo este sistema operativo. La plataforma .NET pretende solventar la mayoría de estos problemas.

Con .NET Microsoft entra de lleno en el campo de los servicios web y establece el XML como norma en el transporte de información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

2.2.2.2 ELEMENTOS DE PÁGINAS ASPX

Una página aspx está compuesta por dos archivos:

- [nombre archivo].aspx
- [nombre archivo].cs (en el caso de C#)

El archivo .aspx contiene el código html, y las etiquetas de servidor correspondientes (como pueden ser etiquetas ***asp***, ***control***, ***panel***) que sustituyen a las etiquetas de tipo button, textarea, etc, correspondientes al código html.

También puede contener código javascript, que hace llamadas al archivo de código subyacente (en nuestro caso .cs) en lo que se conoce como código Ajax.

El archivo .cs contiene el código en C#, VB.NET, etc en el que estamos realizando nuestra aplicación.

A continuación, un ejemplo que utiliza código "en línea", opuesto al código independiente (code behind).

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">

   protected void Page_Load(object sender, EventArgs e)
   {
      Label1.Text = "Hola mundo";
   }

</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>Hola mundo</title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Label runat="server" id="Label1" />
   </div>
   </form>

</body>
</html>
```

Tabla 6. Ejemplo de página aspx

Microsoft recomienda que, para realizar programación dinámica, se use el modelo code behind, que coloca el código en un archivo separado o en una etiqueta de script especialmente diseñada.

El modelo code behind de ASP .NET marca la separación del ASP clásico y sirve para construir aplicaciones con la idea de presentación y contenido separados. Esto permite separar la creación de la página desde el punto de vista del diseño de la parte del código de programación. Esto es similar a la separación en el Modelo Vista Controlador.

A continuación vemos un ejemplo de código code behind:

<%@ Page Language="C#" CodeFile="Ejemplo.aspx.cs" Inherits="SitioWeb.Ejemplo" AutoEventWireup="true" %>

Esta etiqueta se coloca al principio de un archivo .aspx. La propiedad CodeFile de la directiva @Page especifica qué archivo (.cs ó .vb) contiene el código code behind mientras que la propiedad Inherits especifica la clase de la cual deriva la página. En este ejemplo, la directiva @Page está incluida en Ejemplo.aspx y el archivo Ejemplo.aspx.cs contendrá el código de esta página:

```
using System;

namespace SitioWeb
{
        public partial class EjemploCodeBehind: System.Web.UI.Page
        {
                protected void Page_Load(object sender, EventArgs e)
                {

                }
        }
}
```

En este caso, el método Page_Load() será llamado cada vez que la página aspx sea solicitada al servidor. Se pueden implementar manejadores de eventos en varias etapas del proceso de ejecución de la página.

Estructura

Las aplicaciones ASP .NET son alojadas en un servidor web y se tiene acceso a ellas mediante el protocolo HTTP, que no guarda ninguna información sobre conexiones anteriores. Por lo tanto, si la aplicación requiere interacción entre conexiones, tiene que implementar su propia administración del estado. ASP .NET proporciona varias maneras de administrar el estado de las aplicaciones ASP .NET.

- **Controles de usuario**

  ASP .NET permite la reutilización de componentes gracias a los controles de usuario (User Controls). Un control de usuario tiene la estructura de un formulario web, excepto porque los controles derivan de la clase System.Web.UI.UserControl, y son almacenados en archivos .ascx. Al igual que los archivos aspx, un archivo ascx contiene etiquetas html

o xhtml, además de etiquetas para definir controles web y otros controles de usuario. También pueden hacer uso del modelo code behind.

Se pueden agregar propiedades y métodos propios, y manejadores de eventos. Un mecanismo de eventos en burbuja da la capacidad de pasar un evento que se ha disparado en un control de usuario a la página que lo contiene.

- **Estado de la aplicación**

El estado de la aplicación es un conjunto de variables definidas por el usuario compartidas por todas las invocaciones de una aplicación ASP .NET. Estas se establecen e inicializan cuando el evento Application_OnStart se dispara en la carga de la primera instancia de las aplicaciones y están disponibles hasta que la última instancia termina. Las variables de estado de la aplicación son identificadas por nombres.

- **Estado de la sesión**

El estado de la sesión es un conjunto de variables definidas por el usuario, las cuales se mantienen durante la sesión de un usuario. Estas variables son únicas para diferentes instancias de una sesión de usuario, y se accede a ellas utilizando la colección **Session**. Las variables de sesión pueden ser automáticamente destruidas después de un determinado tiempo de inactividad, incluso si la sesión no ha terminado. Por la parte del cliente, una sesión de usuario se identifica con una cookie o codificando el identificador (ID) de la sesión en la misma url.

ASP .NET proporciona tres métodos de persistencia para variables de sesión:

InProc: Las variables de sesión son mantenidas dentro del proceso y son destruidas cuando el proceso es reciclado o terminado.

StateServer: En este modo, ASP .NET ejecuta un servicio de Windows separado que mantiene las variables de estado. Como esto ocurre fuera del proceso ASP .NET, impacta negativamente en el rendimiento, pero permite a múltiples instancias de ASP .NET compartir el mismo estado del servidor, permitiendo que una aplicación ASP .NET pueda tener su carga balanceada y escalada en múltiples servidores. También, como el servicio de administración del estado se ejecuta independientemente de ASP .NET, las variables pueden persistir a través de las finalizaciones del proceso ASP .NET.

SqlServer: En este modo, las variables de estado son almacenadas en un servidor de base de datos al que se puede acceder mediante SQL. Las variables de sesión pueden persistir a través de finalizaciones de procesos también en este modo.

- **Estado de la vista**

El estado de la vista se refiere al mecanismo de administración de estado a nivel de página, que es utilizado por las páginas HTML generadas por las aplicaciones ASP .NET para mantener el estado de los controles de los formularios web. El estado de los controles es codificado y mandado al servidor en cada envío del formulario en un campo oculto conocido como **VIEWSTATE**. El servidor devuelve las variables para que cuando la página sea renderizada de nuevo, los controles vuelvan al último estado en el que se encontraban. Del lado del servidor, la aplicación puede cambiar el estado de la vista, si los resultados del procesamiento actualizan el estado de cualquier control. El

estado de los controles individuales son decodificados en el servidor, y están disponibles para su uso en ASP .NET usando la colección ViewState.

- **Motor de plantillas**

En una primera versión, ASP .NET carecía de un motor de plantillas. dado que el .NET Framework es orientado a objetos y permite la herencia, se puede definir una nueva clase que herede desde **System.Web.UI.Page**, escribir métodos en ella que rendericen HTML, y entonces hacer páginas que hereden de esta nueva clase. Mientras esto permite que los elementos sean comunes dentro de un sitio web, agrega complejidad y mezcla código fuente con lenguaje de marcado. Además, este método puede ser visto solamente al ejecutar la aplicación, no mientras se está diseñando.

ASP .NET 2.0 incluye el concepto de página maestra (Master Page), que es un plantilla web que se utiliza en las páginas desarrolladas. Una aplicación web puede tener una o más páginas maestras y estas pueden ser anidadas. Las master pages contienen controles contenedores, **Content Place Holder**, que indican donde irá el contenido dinámico. También contien HTML y JavaScript, que será compartido a través de las páginas hijas.

Las páginas hijas también usan esos tipos de controles, que deben estar relacionados con el **Content Place Holder** de la master page que contiene a esta página hija. El resto de la página está definido por las partes compartidas de la master page. Todo el lenguaje de marcado y controles de servidor en la página de contenido deben ser colocadas dentro del **Content Place Holder**.

Cuando se realiza una solicitud por una página de contenido, ASP .NET mezcla la salida de la página de contenido con la salida de la master page, y envía el resultado al cliente.

La master page permanece completamente accesible a la página del contenido. Esto significa que la página de contenidos puede manipular los encabezados, cambiar el título, configurar la caché, etc. Si la master page expone propiedades públicas o métodos, el contenido de la página puede utilizarlos también.

- **Declaración de variables y del lenguaje**

Al igual que el lenguaje HTML, los códigos ASP tienen una etiqueta de inicio y fin de página. Para abrir y cerrar una etiqueta utilizamos <% y %>, de la siguiente forma:

<% variable **%>**

En una página aspx podemos definir las variables en el code behind, y mostrarlas en la página a través de estas etiquetas. Por ejemplo, si declaramos una variable del tipo cadena:

public string cadena = “Hola a todos”;

luego la podemos mostrar en el código html de la siguiente manera:

<%= cadena **%>**

- **Comentarios**

Para poner un comentario se utilizan unas etiquetas parecidas a las que utiliza html. De hecho, para el código html propio de una página aspx podemos utilizar los mismo tags de comentarios, como <!-- html -->. Los propios de .NET para comentar las etiquetas de servidor son <%-- etiqueta --%>. Utilizando esto, toda la línea que contenga el comentario no se ejecutará.

- **Librerias**

La tecnología .NET divide su FrameWork en una conjunto de librerías de clases con las que se pueden construir aplicaciones distribuidas.

.NET prioriza los siguientes apartados:

-Tecnologías de internet mediante la librería Web.

-XML como vehículo transmisor de información

-Incrementa la presencia de modos de tratamiento de la información mediante la librería de datos.

-Nueva filosofía de construcción de formularios web.

Las librerías de ASP .NET permiten el acceso a los diferentes objetos que se utilizan al implementar una aplicación en un lenguaje perteneciente a .NET como C#, VB .NET, etc. Algunos de estos objetos son los siguientes:

-Response

-Server

-Session

-Request

A continuación, se muestran estas librerías:

Librería de sistema	Librería de datos	Librería de XML	Librería de Windows	Librería de dibujo	Librería Web
Collections	OleDb	Serialization	Design	Drawing2D	Discovery
IO	Common	Schema	ComponentModel	Printing	Protocols
Security	SqlClient	XLS		Imaging	HTML Web Controls
Runtime	SQL Types	Xpath		Text	Web Form Controls
Configuration					Caching
Net					Configuration
Services					SessionState
InterOp					Security
Diagnostics					
Reflection					
Text					
Remoting					
Globalization					
Threading					
Serialization					

Tabla 7. Librerías ASP .NET

Como conclusión, podríamos establecer que Microsoft ha creado un adversario a la altura de J2EE, y que incluso supera a este en algunos aspectos.

.NET obtiene muchas funcionalidades de la tecnología JAVA, pero con un acceso mucho mas fácil y con la posibilidad de emplear varios lenguajes de programación sin perder potencia de diseño, ya que estos lenguajes son orientados a objetos y, además, gracias al modo de desarrollo de formularios web empleando controles, el desarrollo de cliente a nivel funcional es más potente que Flash MX.

[msdn][netveloper][devjoker][desarrolloweb][programación.com][whatisbyarvind][clickear]

2.2.3 SERVICIOS WEB

2.2.3.1 DEFINICIÓN

Un Web Service o servicio web es un componente de software que para comunicarse con otras aplicaciones utiliza mensajes formateados en XML. Estos mensajes son enviados mediante protocolos de Internet, como por ejemplo HTTP (Hypertext Transfer Protocol). Un Web Service es similar a un sitio web sin interfaz de usuario y que da servicio a las aplicaciones en lugar de hacerlo a las personas. Un Web Service, en lugar de recibir solicitudes desde un navegador y devolver páginas web como respuesta, lo que hace es recibir solicitudes mediante mensajes formateado en XML desde una aplicación, realiza una tarea y devuelve un mensaje de respuesta también formateado en XML.

El estándar de estos mensajes que están promoviendo tanto Microsoft como otras empresas líderes para los Web Services es SOAP. Un mensaje SOAP es similar a una carta: contiene una cabecera con la dirección del receptor del mensaje, un conjunto de opciones de entrega (como la información de encriptación), y un cuerpo o body con la información o data del mensaje.

Los Web Services sirven como modelo de programación para la comunicación entre aplicaciones. La conexión de aplicaciones a través de Internet mejora la capacidad de las empresas para trabajar conjuntamente entre ellas. Si se crea una capa de Web Services sobre una aplicación corporativa, las empresas que ofrecen estos Web Services podrán permitir que sistemas externos puedan invocar las funciones de estas aplicaciones a través de Internet (o una intranet corporativa) sin tener que modificar la aplicación. Hoy en día varias empresas tienen sus Web Services, que actúan como front end para aplicaciones de entrada de órdenes que están residentes internamente en un mainframe. Estas empresas permiten a los clientes enviar órdenes de compra a través de Internet. Poner una capa de Web services sobre las aplicaciones existentes es una solución muy interesante para integrar las aplicaciones desarrolladas por los diferentes departamentos y así reducir los costos de integración.

Figura 2. Modelo de capas de los servicios web

Una vez explicado lo que es un Web Service, vamos a entrar un poco más en profundidad en las características de estos.

Requisitos de un Web Service

- **Interoperabilidad**: Un servicio remoto debe poder utlizarse por clientes de otras plataformas.
- **Amigabilidad con Internet**: Debe soportar accesos remotos de clientes a través de Internet.
- **Interfaces fuertemente tipadas**: Los datos enviados y recibidos por un servicio remoto no pueden tener ambigüedad de tipos, y deben poder corresponderse con los tipos de datos de los diferentes lenguajes de programación.
- **Posibilidad de aprovechar los estándares de Internet existentes**: Para implementar los servicios remotos se deben aprovechar los estándares de Internet ya creados en la medida de lo posible, y no resolver los problemas de diferente manera de la que ya se han resuelto, ya que de esta manera se pueden aprovechar las herramientas y productos existentes.
- **Soporte para cualquier lenguaje**: No debe estar limitado a un lenguaje de programación en particular, como puede ser Java RMI, que está ligada completamente a lenguaje Java. Sería muy difícil invocar funcionalidad de un objeto Java remoto desde Visual Basic o PERL. Un cliente debería ser capaz de implementar un nuevo servicio Web existente independientemente del lenguaje de programación en el que se halla escrito el cliente

- **Soporte para cualquier infraestructura de componente distribuida**: No debe estar fuertemente ligada a una infraestructura de componentes en particular. No se tendría que comprar, instalar o mantener una infraestructura de objetos distribuidos, solo construir un nuevo servicio remoto o utilizar un servicio existente. Los protocolos subyacentes deberían proporcionar un nivel base de comunicación entre infraestructura de objetos distribuidos existentes tales como DCOM y CORBA.

Bloques Constructivos de Servicios Web

Estos son los bloques constructivos principales para realizar las comunicaciones remotas.

Tabla 8. Bloques constructivos de Servicios Web

- **Descubrimiento**: La aplicación cliente que necesita acceder al contenido que expone un Servicio Web necesita una manera de resolver la ubicación de servicio remoto. Esto se consigue mediante un proceso llamado descubrimiento (discovery). Este proceso se puede proporcionar mediante un directorio centralizado y por otros métodos ad hoc. En DCOM, el servicio de descubrimiento lo proporciona el Administrador de control de servicios (SCM, Services Control Manager).
- **Descripción**: Proporciona la información que el cliente necesita para interactuar adecuadamente con un Servicio Web. La descripción de un servicio Web contiene meta datos estructurados sobre la interfaz que intenta utilizar la aplicación cliente así como documentación escrita sobre dicho servicio Web. Un componente DCOM expone meta datos estructurados sobre sus interfaces a través de una biblioteca de tipo (typelib). Los meta datos que pertenecen una typelib de componente se guardan en un formato binario propietario y se accede a ellos mediante una interfaz de programación de aplicación (API) propietaria.

- **Formato del mensaje**: El formato del mensaje debe ser el mismo para el intercambio de datos. Tanto el cliente como el servidor tienen que estar de acuerdo en un mecanismo común de codificación. Esto asegura que los datos que sean codificados por el cliente serán correctamente interpretados por el servidor. En DCOM, estos mensajes tiene un formato definido por el protocolo DCOM Object RPC (ORPC).
- **Codificación**: Los datos que se trasmiten entre el cliente y el servidor deben ser codificados en un cuerpo de mensaje. DCOM utiliza un esquema de codificación binaria para serializar los datos de los parámetros que se intercambian entre el cliente y el servidor.
- **Transporte**: Una vez se ha dado formato al mensaje y se han serializado los datos en el cuerpo de este, se debe transferir entre el cliente y el servidor utilizando un protocolo de transporte. DCOM dispone de varios protocolos propietarios como TCP, SPX, NetBEUI y NetBIOS sobre IPX.

Finalmente, podemos resumir que los servicios web se dividen en *servicios de transporte* , *servicios de mensajería*, de *descripción* y de *descubrimiento*. En la parte más baja se encuentran los servicios de transporte, que es el protocolo de más bajo nivel, que codifica la información independientemente del formato y establece la conexión y el puerto usado. Generalmente se usa HTTP (WWW) pero se puede usar también SMTP (correo electrónico), FTP (File Transfer Protocol), o Beep (blocks extensible exchange protocol), que es específico para Servicios Web, y a diferencia de los anteriores, no es cliente-servidor, sino "entre pares", lo que quiere decir que los dos ordenadores entre los que se establece la comunicación actúan como clientes y servidores a la vez. Es extensible y está especificado en XML, por lo que se está haciendo mucho más popular para aplicaciones web. Un ejemplo de uso de Beep sería el siguiente:

```
RPY 0 0 . 0 124
        Content-Type: application/beep+xml
        <greeting>
                <profile uri="http://xml.resource.org/profiles/TLS" />
                <profile uri="http://xml.resource.org/profiles/SEP" />
                <profile uri="http://xml.resource.org/profiles/sasl/OTP" />
                <profile uri="http://xml.resource.org/profiles/sasl/..." />
        </greeting>
        END
        <start number="1">
                <profile uri="..."> ... </profile>
                <profile uri="..."> ... </profile>
        </start>
        <close number="1" code="200" />
```

Tabla 9. Ejemplo de protocolo 'Beep'

En este ejemplo se declaran una serie de perfiles de canales, y se abre un canal, para cerrarlo finalmente (close).

Por encima de los servicios de transporte se encuentran los servicios de mensajería, que especifican cómo se tiene que codificar el mensaje que contiene los datos que se intercambian entre el cliente y el servidor. El protocolo más usado en esta capa es el SOAP.

Este protocolo puede usar cualquiera de los transportes anteriores. Se pueden escribir clientes y servidores en cualquier lenguaje y usa XML como lenguaje para especificar los mensajes. XML-RPCb (Remote Procedure Call a través de XML) es un método de llamada remota a procedimientos usando XML, que usa como capa de transporte HTTP a diferencia de SOAP, que puede usar cualquiera.

Hay otros métodos que se pueden usar, como simplemente proporcionar páginas XML sobre protocolo HTTP. A continuación se muestra un ejemplo de una petición SOAP, que tendría la forma siguiente:

```
POST /examples HTTP/1.0
        User-Agent: Radio UserLand/7.0 (MacOS)
        Host: localhost:81
        Content-Type: text/xml
        Content-length: 474
        SOAPAction: "/examples"
        <?xml version="1.0"?>
        <SOAP-ENV:Envelope                                                SOAP-
        ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP- ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsd="http://www.w3.org/1999/XMLSchema"
        xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance">
                <SOAP-ENV:Body>
                        <m:Prueba xmlns:m="http://soapware.org/">
                                <param1 xsi:type="xsd:int">41</param1>
                        </m: Prueba >
                </SOAP-ENV:Body>
        </SOAP-ENV:Envelope>
```

Tabla 10. Ejemplo petición SOAP

En este ejemplo se llama a una función llamada *Prueba*, a la cual se le pasa un parámetro llamado *param1*.

Los servicios deben especificarse para que una aplicación sepa automáticamente qué formato usar para comunicarse con un servicio. Para esto utilizamos WSDL (web services description language), que nos permite especificar la dirección de un servicio y la interfaz que se usa para acceder a él, sea SOAP o HTML. A continuación tenemos un ejemplo de document WSDL:

```
<?xml version="1.0"> →Comienzo del documento
<definitions> → Etiqueta raíz (Agrupa a las demás)
<types>
     Tipos de datos utilizados en el Web Service
</types>
<message>
     Métodos y parámetros para realizar la operación
</message>
<portType>
     Se definen las operaciones que pueden ser realizadas y los
mensajes que involucran (p.e. petición y respuesta) </portType>
<binding>
     Formato del mensaje y detalles del protocolo para cada
portType
</binding>
</definitions> →Fin del documento
```

En la capa más alta está UDDI (universal description discovery and integration), que permite, además de describir servicios web, describir productos, la empresa en sí, y cómo está dispuesta a llevar a cabo transacciones. UDDI permite buscar negocios, servicios por categorías, y devuelve información sobre cómo acceder a ellos.

Todos esos protocolos y servicios forman un conjunto de servicios web, utilizados por muchas empresas. Microsoft .Net acepta los estándares de todas las capas, aunque varía en la forma de acceder a los servicios y programarlos, y usa XML de forma extensiva en todos los servicios. .Net proporciona una serie de elementos que permiten acceder a los servicios Web de una forma adaptada especialmente a ellos.

2.2.3.2 UTILIDAD

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí con el fin de proporcionar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar.

Los estándares más empleados en estos servicios son:

- **Web Services Protocol Stack:** Conjunto de servicios y protocolos de los servicios web.
- **XML** (eXtensible Markup Language): El formato estándar que se utiliza para el intercambio de datos.
- **SOAP** (Simple Object Access Protocol) o **XML-RPC** (XML Remote Procedure Call): Protocolos sobre los que se establece la comunicación para el intercambio de datos.
- **Otros protocolos**: Los datos en XML también pueden ser enviados entre aplicaciones a través de protocolos normales como HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol) o SMTP (Simple Mail Transfer Protocol).
- **WSDL** (Web Services Description Languages): Es el lenguaje de la interfaz pública para los servicios Web. Es un formato XML para describir los servicios de red como un conjunto de variables que operan en los mensajes que contienen la información orientada a procedimiento o al documento.
- **UDDI** (Universal Description Discovery and Integration): Es un servicio utilizado para publicar la información de los servicios Web. Describe los servicios web y muestra información asociada a estos.
- **WS-Security** (Web Services Security): Protocolo de seguridad aceptado como estándar por OASIS (Organization for the advancement of Structured information standards). Garantiza la integridad y la seguridad en mensajería de servicios web, trabajando en la capa de aplicación (encabezado de un mensaje SOAP).

La principal razón para usar servicios Web es que se basan en HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. La mayoría de organizaciones protegen sus redes mediante firewalls, los cuales filtran y bloquean gran parte del tráfico de Internet, y

cierran casi todos los puertos TCP excepto el 80, que es el que usan los navegadores. Los servicios Web utilizan este puerto por el simple hecho de que no resultan bloqueados.

Otra razón es que, antes de que existiera SOAP, no había buenas interfaces para acceder a toda la funcionalidad que proporcionaban otros ordenadores en red. Las que había eran ad hoc y poco conocidas, tales como EDI (Electronic Data Interchange), RPC (Remote Procedure Call), u otras APIs.

Una última razón de la practicidad de los servicios Web es que estos aportan gran independencia entre la aplicación que usa el servicio Web y el propio servicio.

De hecho utilizan servidores de aplicaciones para servicios Web, los cuales son:

- **Jboss**: servidor de aplicaciones J2EE Open Source de Red Hat inc.
- **Oracle Fusion Middleware**: es una cartera de productos de software, producido por Oracle, que proporciona varios servicios, incluyendo J2EE y herramientas para desarrolladores, servicios de integración, inteligencia de negocios, colaboración y gestión de contenidos. OFM se basa en estándares abiertos como BPEL, SOAP, XML y JMS.
- **IBM Lotus Domino** a partir de la versión 7.0: Ofrece un entorno seguro y fiable de mensajería y colaboración empresariales.
- **Axis** y el servidor **Jakarta Tomcat** (de Apache)
- **ColdFusion MX** de Macromedia: permite crear, implantar y mantener aplicaciones eficaces para las empresas.
- **Java Web Services Development Pack** (JWSDP) de Sun Microsystems (basado en Jakarta Tomcat): se utiliza para desarrollar servicios y aplicaciones web y aplicaciones java.
- **JonAS**: es un servidor de aplicación J2EE de código abierto implementado en Java. Forma parte de la iniciativa de código abierto de ObjectWeb.
- **Microsoft .NET**, del que ya hemos hablado.
- **Novell exteNd** (basado en la plataforma J2EE).
- **WebLogic** : es un servidor de aplicaciones J2EE y también un servidor web HTTP de BEA Systems de San José, California, para Unix, Linux, Microsoft Windows, y otras plataformas.
- **WebSphere**: es una familia de productos de software propietario de IBM, aunque el término se refiere de manera popular a uno de sus productos específicos: WebSphere Application Server(WAS). WebSphere ayudó a definir la categoría de software middleware y está diseñado para configurar, operar e integrar aplicaciones de e-business a través de varias plataformas de red usando las tecnologías del Web. Esto incluye componentes de run-time (como el WAS) y las herramientas para desarrollar aplicaciones que se ejecutarán sobre el WAS. La familia de productos WebSphere además incluye herramientas para diseñar procesos de negocio (WebSphere Business Modeler), para integrarlos en las aplicaciones existentes (WebSphere Designer) y para ejecutar y monitorizar dichos procesos (WebSphere Process Server, WebSphere Monitor).

- **Zope**: es un servidor de aplicaciones Web de código abierto orientado a objetos desarrollado en el lenguaje de programación Python.
- **Verastream** de AttachmateWRQ para modernizar o integrar aplicaciones host IBM y VT

De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más utilizada.

Se espera que para los próximos años mejoren la calidad y cantidad de servicios ofrecidos basados en los nuevos estándares.

Ventajas de los servicios Web

- Aportan interoperabilidad entre aplicaciones independientemente de sus propiedades o de las plataformas sobre las que estén instalados.
- Promocionan los estándares y protocolos basados en texto, con lo cual es más sencillo acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, pueden aprovecharse de los sistemas de seguridad, como firewalls, sin necesidad de cambiar las reglas de filtrado.
- Permiten una fácil combinación de servicios y software de distintas compañías a pesar de que estén geográficamente distantes.
- Facilitan la interoperabilidad entre plataformas de fabricantes distintos gracias a protocolos estándar y abiertos. Las especificaciones son gestionadas por W3C, que es una organización abierta, y gracias a esto no hay secretismos por intereses de fabricantes y se garantiza la plena interoperabilidad entre aplicaciones.

Inconvenientes de los servicios web

- No tienen comparación a nivel de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture) para realizar transacciones.
- Al compararlos con otros modelos de computación distribuida como como RMI (Remote Method Invocation), CORBA o DCOM (Distributed Component Object Model) su rendimiento es bajo. Esto es debido a adoptar un formato basado en texto, aunque hemos de tener en cuenta que entre los objetivos de XML no se encuentra la eficacia de procesamiento.
- Al apoyarse en HTTP, pueden saltarse medidas de seguridad como firewalls cuyas reglas tratan de bloquear o auditar la comunicación entre programas.

([webservices][ugr])

2.3 WEKA

2.3.1 RECONOCIMIENTO Y CLASIFICACIÓN

2.3.1.1 CLASIFICACIÓN

El reconocimiento o lectura de patrones, identificación de figuras y reconocimiento de formas, es el reconocimiento de patrones en señales. El punto principal para un reconocimiento de patrones es la clasificación. Una señal se puede clasificar dependiendo de sus características. Señales, características y clases pueden ser de cualquier forma.

El objetivo es clasificar patrones en base a un conocimiento a priori o información estadística extraída de los patrones. Los patrones a clasificar suelen ser grupos de medidas u observaciones, definiendo puntos en un espacio multidimensional apropiado.

El reconocimiento o clasificación es el proceso por el que se asigna una "etiqueta", que representa una clase, a un patrón concreto.

2.3.1.2 PATRONES Y CLASES

Tras los procesos de segmentación, extracción de características y descripción, cada objeto queda representado por una colección ordenada y estructurada de descriptores, denominada patrón.

En los problemas de reconocimiento cada patrón debe pertenecer a una categoría o clase, y el sistema de reconocimiento debe asignar cada objeto (de interés) a su categoría.

Entendemos por **características** a cualquier medida simbólica y/o numérica que puede ser calculada (tamaño, edad, etc.).

Una **clase** es un conjunto de entidades que comparten alguna característica que las diferencia de otras.

Selección de características

La selección de características es imprescindible en la minería de datos. Esto es debido a que cuando se genera un modelo de minería de datos, es posible que el conjunto de datos contenga más información de la que se necesita para generar el modelo. Es decir, que si tenemos un conjunto de datos que contiene 500 datos distintos que describen las características de un cliente, es posible que sólo 50 de esos datos se usen para generar un determinado modelo. Si no eliminamos los datos innecesarios mientras generamos el modelo, se necesitará más CPU y memoria en el proceso de entrenamiento, y más espacio de almacenamiento para el modelo completo.

Aunque dispongamos de recursos suficientes para la generación del modelo, se deben quitar los datos innecesarios ya que pueden degradar la calidad de los patrones detectados debido a que:

- Algunos datos son ruido o redundantes, con lo que se dificulta la detección de patrones significativos a partir de los datos.
- Para obtener patrones de calidad, los algoritmos de minería de datos requieren un conjunto de datos de entrenamiento mucho más grande en un conjunto de datos multidimensional, a pesar de que en algunas aplicaciones se dispone de muy datos de aprendizaje.

La selección de características sirve para evitar tener muchos datos con poco valor o al contrario, tener pocos datos de mucho valor.

Las características se presentan en vectores de características n-dimensionales:

$x = \{x_1, x_2, \ldots, x_n\}$

donde cada $x_,$ representa la i-ésima característica y n es el número total de características.

Los vectores de características producen un espacio multidimensional o espacio de características, donde cada patrón se representa como un punto en un espacio n-dimensional. Esta dimensión se define en función del número de características que describen los patrones, con lo que objetos similares se encontrarán cercanos en el espacio de características con lo que es posible realizar la clasificación y agrupamiento de los mismos. Si cada característica es un número real, el espacio de características es Rn

El espacio de características se divide en regiones de decisión, las cuales representan gráficamente la zona de pertenencia a una clase. Estas regiones deben representar todo Rn y ser disjuntas. A los límites de cada una de las regiones de decisión se le denomina frontera de decisión.

Dadas dos clase w_1 y w_2, tenemos sus funciones discriminantes:

$g_i(x) = p\ (x|w_i)P(w_i)$

$g_i(x) = \log p\ (x|w_i) + \log P(w_i)$

Frontera de decisión:

$g_1(x) = g_2(x)$

La minería de datos y el aprendizaje automático son técnicas relacionadas con el tratamiento de grandes cantidades de datos.

La técnica de minería de datos intenta obtener patrones o modelos a partir de los datos obtenidos.

Los distintos tipos de clasificadores que existen tienen en común el aprendizaje automático. La idea básica del aprendizaje consiste en utilizar las percepciones para actuar y mejorar la habilidad de un agente para actuar en el futuro.

Existen diferentes tipos de técnicas de aprendizaje:

Aprendizaje supervisado

El aprendizaje supervisado se basa en aprender una función que establezca una relación entre las entradas y las salidas deseadas del sistema a partir de ejemplos etiquetados anteriormente. No siempre podemos hacer este tipo de entrenamiento, porque necesitamos la salida esperada en función de la entrada. El sistema de aprendizaje trata de etiquetar o clasificar una serie de vectores utilizando una entre varias categorías

Aprendizaje no supervisado

El aprendizaje no supervisado consiste en aprender a partir de patrones de entradas, para los que no se especifican los valores de sus salidas.

TIPOS DE CLASIFICADORES

A continuación resumimos 2 tipos de clasificadores que hemos utilizado para obtener resultados:

Clasificador estadístico (bayesiano)

Un clasificador bayesiano es un clasificador de patrones que se basa en teorías estadísticas y probabilidades para el aprendizaje. Para ello, se calcula la probabilidad de cada hipótesis de los datos y a partir de estas probabilidades se realizan las predicciones. Podemos definirlo como la búsqueda de la hipótesis más probable a partir de un conjunto de datos de entrenamiento y un conocimiento a priori de la probabilidad de cada hipótesis. El problema de este tipo de aprendizaje es que necesita realizar una gran cantidad de cálculo ya que normalmente el espacio de hipótesis es muy grande o incluso puede ser infinito. Está basado en el teorema de Bayes. Es el clasificador que comete, de media, menos errores de clasificación.

Clasificador Backpropagation o retropropagación

Es un método de aprendizaje supervisado que se basa en el descenso por el gradiente. Se utiliza mucho en las aplicaciones de redes neuronales. Requiere que la red disponga de información tanto en sus entradas como en sus salidas. El algoritmo itera entre los pesos, ya que tiene como finalidad ajustarlos o adaptarlos de manera que se reduzca el error entre la salida deseada y la real. Es costoso y el periodo de entrenamiento es bastante grande.

TÉCNICAS DE RECONOCIMIENTO DE PATRONES

Métodos estadísticos

Utilización de técnicas estadísticas que hacen uso de las propiedades de distribución de los patrones. La idea es conseguir que la clasificación conlleve la probabilidad más baja de error posible (clasificador bayesiano).

Métodos estructurales

Usan como valores de los patrones la información no numérica, estructurada en forma de cadenas (clasificadores sintácticos). Estos clasificadores son útiles cuando el factor de discriminación entre patrones depende de la estructura de la información en el patrón.

Métodos neuronales

Sistemas constituidos por una red neuronal, capaz de realizar su función mediante un proceso de aprendizaje en el que se muestran a la red patrones de entradas o de entradas/salidas y mediante un algoritmo se modifican las interconexiones entre distintas neuronas. Son útiles cuando el problema maneja un conocimiento escaso, cambiante y variable en el tiempo. Proporcionan niveles de precisión superiores a las técnicas tradicionales, especialmente cuando se usan conjuntos de datos complicados, como pueden ser series temporales de imágenes.

2.3.2. HERRAMIENTA WEKA

2.3.2.1 INTRODUCCIÓN

Weka es una colección de algoritmos de aprendizaje automático para tareas de extracción de datos. Los algoritmos pueden ser aplicados directamente a un conjunto de datos o invocados desde código Java.

Es un software de código abierto publicado bajo la licencia GNU General Public.

Está formado por una colección de herramientas y algoritmos para análisis de datos y modelado predictivo, utilizando diferentes técnicas de preprocesamiento, clasificación, agrupamiento, asociación y visualización. Para su fácil utilización dispone de una interfaz gráfica de usuario para acceder fácilmente a sus funcionalidades.

Datos

Principalmente, Weka trabaja con archivos con formato arff (attribute relation file format). La estructura de estos archivos es la siguiente:

1. **@relation <nombre relación>**

   Los archivos ARFF deben comenzar con esta declaración en la primera línea (no puede haber espacios en blanco al inicio). **< nombre relación >** es una cadena de caracteres y si contiene espacios hay que ponerlos entre comillas.

2. **@attribute <nombre atributo> <tipo de dato>**

   En esta sección se definen los atributos. Tenemos que incluir una línea por cada atributo (o columna) que vaya a contener nuestro conjunto de datos, indicando su nombre y el tipo de dato.

   Con **<nombre atributo>** indicamos el nombre del atributo, que debe comenzar por una letra, y si contiene espacios tendrá que ir entre comillas.

   Con **<tipo de dato>** indicamos el tipo de dato para este atributo (o columna) que puede ser:

   - **numeric** (numérico)
   - **string** (texto)
   - **date [<date-format>]** (fecha). En <date-format> indicamos el formato de la fecha, que será del tipo "yyyy-MM-dd'T'HH:mm:ss".
   - **<nominal >**. Estos son tipos de datos definidos por nosotros mismos y que pueden tomar una serie de valores que indicamos.

3. **@data**

   En esta sección definimos los datos en sí. Tenemos que separar cada columna por comas y todas las filas deberán tener el mismo número de columnas, y que debe ser el mismo que hemos definido en la sección **@attribute**.

   Si nos falta algún dato, colocaremos un signo de interrogación (**?**) en su lugar. El separador de decimales tiene que ser obligatoriamente el punto, y las cadenas de tipo string tienen que estar entre comillas simples.

   Aquí tenemos un ejemplo de archivo arff:

```
1 @relation clima
2
3 @attribute pronóstico {soleado, nublado, lluvioso}
4 @attribute temperatura real
5 @attribute humedad real
6 @attribute viento {TRUE, FALSE}
7 @attribute jugar {si, no}
8
9 @data
10 soleado,85,85,FALSE,no
11 soleado,80,90,TRUE,no
12 nublado,83,86,FALSE,si
13 lluvioso,70,96,FALSE,si
14 lluvioso,68,80,FALSE,si
15 lluvioso,65,70,TRUE,no
16 nublado,64,65,TRUE,si
17 soleado,72,95,FALSE,no
18 soleado,69,70,FALSE,si
19 lluvioso,75,80,FALSE,si
20 soleado,75,70,TRUE,si
21 nublado,72,90,TRUE,si
22 nublado,81,75,FALSE,si
23 lluvioso,71,91,TRUE,no
```

Tabla 11. Ejemplo de archivo .arff

A continuación, pasamos a explicar las opciones que nos ofrece Weka cuando ejecutamos la aplicación:

Figura 3. Ejecución de weka

- **Explorer:** es la opción que nos permite ejecutar los algoritmos de análisis implementados sobre los ficheros de entrada, una ejecución independiente por cada prueba.
- **Experimenter:** esta opción permite definir experimentos más complejos, con objeto de ejecutar uno o varios algoritmos sobre uno o varios conjuntos de datos de entrada, y comparar estadísticamente los resultados.
- **KnowledgeFlow:** esta opción permite llevar a cabo las mismas acciones del "Explorer", con una configuración totalmente gráfica, inspirada en herramientas de tipo "flujo de datos" para seleccionar componentes y conectarlos en un proyecto de minería de datos, desde que se cargan los datos, se aplican los algoritmos de tratamiento y análisis, hasta el tipo de evaluación deseada.
- **Simple CLI:** acrónimo de "Command-Line Interfaz". Es simplemente una ventana de comandos java para ejecutar las clases de WEKA. Los comandos que están disponibles desde Simple CLI son:

1. **java <nombre de la clase> [<args>]**

   Invoca a la una clase java con los argumentos dados si los hay.

2. **break**

   Detiene la tarea actual

3. **kill**

   Detiene la tarea actual de manera más “agresiva”. Solo debe utilizarse si no funciona el comando “break”.

4. **cls**

   Borra la pantalla de la consola de comandos.

5. **exit**

   Sale de la aplicación

6. **help [<command>]**

   Proporciona una lista de los comandos disponibles si no introducimos ningún argumento. Si lo hacemos, proporciona una definición de éste.

2.3.2.2 ALGORITMOS DE CLASIFICACIÓN

Weka utiliza varías técnicas de minería de datos para el proceso de clasificación. Según el objetivo del análisis de los datos, los algoritmos utilizados se clasifican en supervisados y no supervisados:

Algoritmos supervisados (o predictivos): sirven para predecir un dato ó conjunto de ellos desconocido a priori, a partir de otros conocidos. Ej: árboles de decisión.

Algoritmos no supervisados (o descriptivos): sirven para descubrir patrones y tendencias en los datos. Ej: segmentación, clustering.

Algunos de estos algoritmos son:

ALGORITMOS BASADOS EN ÁRBOLES DE DECISIÓN

Introducción

Está técnica pertenece a la metodología de aprendizaje supervisado. Se representa en forma de árbol. Cada nodo del árbol es una decisión, y además cada uno de ellos genera reglas para la clasificación de un conjunto de datos.

Las ventajas de un árbol de decisión son:

- Su utilización es sencilla.
- Admiten atributos discretos y continuos.
- Tratan bien los atributos no significativos y los valores que faltan.
- Su principal ventaja es la facilidad de interpretación.

Debemos utilizar árboles de decisión cuando:

- Podemos describir los ejemplos en términos de pares atributo-valor.
- La función objetivo toma valores discretos.
- Hay posibilidad de ruido en el conjunto de entrenamiento.
- Pueden no conocerse los valores de algunos atributos en los ejemplos del conjunto de entrenamiento.

Representación

Un árbol de decisión se empieza con una decisión que se tiene que tomar.

- Cada nodo que no sea hoja representa un atributo
- Las aristas que parten del nodo etiquetado con el atributo ***A*** están etiquetadas con cada uno de los posibles valores del atributo ***A***.
- Cada hoja corresponde a un valor de la clasificación.

Ejemplos

Árbol J48

El algoritmo basado en árboles de decisión más utilizado por Weka es el árbol J48.

J48 implementa el algoritmo de Quinlan C4.5 para generar un árbol de decisión con poda o sin poda. C4.5 es una extensión del anterior algoritmo ID3 de Quinlan. Los árboles de decisión generados por el algoritmo J48 pueden ser utilizados para la clasificación. J48 construye árboles de decisión partiendo de un conjunto de datos de entrenamiento etiquetados utilizando el concepto de información de la entropía. Cada atributo de los datos puede ser usado para tomar una decisión mediante la división de datos en pequeños subconjuntos. J48 examina la obtención de información normalizada (diferencia de entropía) que es el resultado de elegir un atributo para dividir los datos. Para tomar la decisión, se utiliza el atributo que ha obtenido mayor información normalizada, y se vuelve a ejecutar el algoritmo en los subconjuntos más pequeños. El proceso de división termina si todas las instancias de un subconjunto pertenecen a la misma clase y se crea un nodo hoja en el árbol de decisión para elegir esa clase. También puede sucede que ninguna de las características proporcionen información valiosa. En ese caso, J48 crea un nodo decisión más arriba en el árbol usando el valor esperado de la clase. J48 puede manejar atributos discretos y continuos, entrenando datos con valores de atributos que faltan y atributos con diferentes pesos. Además, proporciona información para la poda de árboles después de su creación.

ALGORITMOS BASADOS EN REDES NEURONALES

Introduccion

Las Redes Neuronales Artificiales (ANNs de Artificial Neural Networks) fueron en principio una simulación conceptual de los sistemas nerviosos biológicos, formados por un conjunto de unidades llamadas "neuronas" o "nodos" que están conectadas unas con otras. Se trataba que estas conexiones imitaran a las dendrítas y los axones en los sistemas nerviosos biológicos.

La primera red neuronal fue propuesta en 1943 por McCulloch y Pitts, que trataron de modelar computacionalmente la "actividad nerviosa". Es un modelo binario, y cada neurona tiene un escalón o umbral prefijado. Este modelo sirvió de ejemplo para otros posteriores, como los modelos de Jhon Von Neumann, Marvin Minsky, Frank Rosenblatt, y muchos otros.

Podemos clasificar los modelos de redes neuronales en base a:

1. Los modelos de **tipo biológico**. Se refiere a las redes que tratan de simular los sistemas neuronales biológicos, las funciones auditivas o algunas funciones básicas de la visión.

2. El modelo **dirigido a aplicación**. No tienen porque guardar similitud con los sistemas biológicos. Sus arquitecturas están fuertemente ligadas a las necesidades de las aplicaciones para las que son diseñados.

Las redes neuronales (RN) representan una técnica de modelación matemática, cuya finalidad es imitar el proceso de aprendizaje que existe en un sistema biológico. Su primer antecedente data de mediados del siglo XX y las primeras que se conocieron fueron las redes Perceptrón y Adaline (Freeman y Skapura, 1991).

Las RN se basan en una estructura de neuronas unidas por enlaces que transmiten información a otras neuronas, y estas a su vez entregan un resultado mediante funciones matemáticas. Las RN aprenden de la información histórica gracias al entrenamiento, mediante el que se ajustan los parámetros de la red, con la finalidad de entregar la respuesta deseada Gracias a esto, las RN obtiene la capacidad de predecir respuestas del mismo fenómeno. El comportamiento de las redes depende entonces de los pesos de los enlaces, de las funciones de activación que se especifican para las neuronas, que pueden ser de tres categorías: lineal, de umbral (o escalón) y sigmoidea, y de la forma en que propagan el error (Freeman y Skapura, 1991).

Existen varios algoritmos para corregir los errores de predicción. Uno de los más usados es el algoritmo de retropropagación o "backpropagation", que consiste en propagar el error hacia atrás, desde la capa de salida hasta la de entrada, gracias o lo cual se pueden adaptar los pesos con el fin de reducir dicho error. (Hilera y Martínez, 2000).

De forma simplificada, el funcionamiento de una red "backpropagation" consiste en el aprendizaje de un conjunto de pares de entradas-salidas dados como ejemplo, empleando un ciclo de propagación–adaptación que consta de dos fases: primero, al aplicar un primer patrón como estímulo para la capa de entrada de la red, éste se va propagando a las capas siguientes para generar la salida, la cual proporciona el valor del error al compararse con el resultado deseado. A continuación estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta intermedia que contribuyan directamente a la salida, recibiendo el porcentaje del error en función a su participación en la salida original.

Este proceso se repite siempre hacia atrás, capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aporte relativo al error total. Basándose en esta información recibida, se reajustan todos los pesos de conexión, de manera que la siguiente vez que se presente el mismo patrón disminuya la diferencia entre la salida calculada y la deseada.

La ventaja de la red backpropagation consiste en su capacidad para adaptar los pesos de las neuronas de las capas intermedias con el objetivo de aprender la relación existente entre un conjunto de patrones dados como ejemplo y sus salidas correspondientes.

Dependiendo del tipo de aplicación y sus características, se han desarrollo distintos tipos de redes neuronales, que se aplican en reconocimientos de voz, sistemas de control, procesamiento, clasificación de patrones en imágenes satelitales, determinación de variables climáticas, procesos químicos, y procesos de gestión.

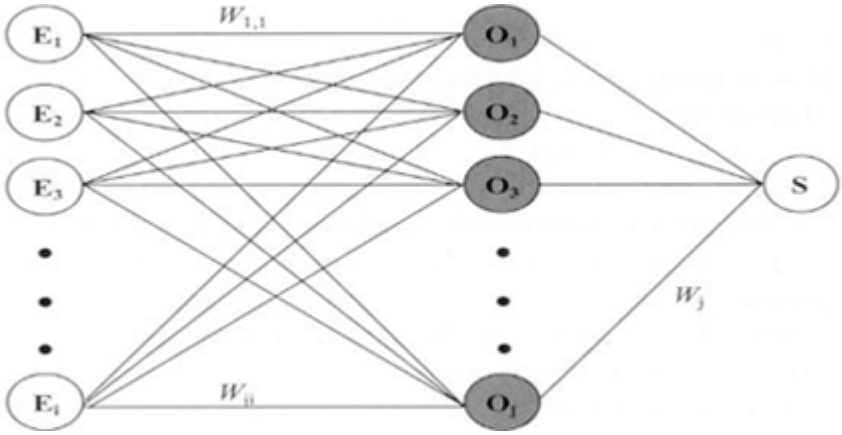

Figura 4. Esquema general de conexiones de una red neuronal[scielo]

Representación

De acuerdo a Hilera y Martínez (2000), el esquema del modelo de la red propuesta, con los pasos y fórmulas que se utilizaron en el algoritmo de entrenamiento, pueden describirse en los siguientes pasos:

- **Paso 1:**

Inicializar los pesos de la red con valores aleatorios.

- **Paso 2:**

Presentar un patrón de entrada X_p : (x_{p1}, x_{p2}, ..., x_{pN}) y especificar la salida deseada que debe generar la red: (d_1, d_2,..., d_M), si la red se utiliza como clasificador, todas las salidas serán cero, salvo una, la que sea de la clase a la que pertenece el patrón de entrada.

- **Paso 3:**

Calcular la salida actual de la red, para ello, para la entrada presentada, se van obteniendo los valores de las respuestas que presenta cada capa hasta llegar a la capa de salida. Los subpasos que lleva este procedimiento son los siguientes:

a- Se calculan las entradas netas (*net*) para las neuronas ocultas procedentes de las neuronas de entrada. Para una neurona oculta j (*O*j):

$$net_{pj}^{h} = \sum_{i=0}^{N} w_{ji}^{h} x_{pi} + \theta_{j}^{h}$$

donde el subíndice *p* corresponde al *p*-ésimo vector de entrenamiento, *j* a la *j*-ésima neurona oculta, w_{ji} es el peso de la conexión entre E_i y O_j y el término θj corresponde a un término de umbral mínimo a alcanzar por la neurona para su activación. A partir de estas entradas se calculan las salidas (*y*) de las neuronas ocultas, utilizando una función de activación *f* .

$$y_{pj}^{h} = f_{j}^{h}\left(net_{pj}^{h}\right)$$

b- Se realizan los mismos cálculos para obtener los valores de resultado de cada neurona *k* de la capa de salida

$$net^{o}_{pk} = \sum_{j=1}^{L} w^{o}_{kj} y^{h}_{pj} + \theta^{o}_{k} \qquad y^{o}_{pk} = f^{o}_{k}\left(net^{o}_{pk}\right)$$

Para minimizar el error (fórmula en el Paso 6) la función *f* debe ser derivable, lo que implica la imposibilidad de utilizar la función escalón. En general se utilizan dos tipos de función de activación: la función lineal ($f\,(net_{j\,k}) = (net_{j\,k})$) y la función sigmoidea definida por la ecuación:

$$f\left(net_{jk}\right) = \frac{1}{1 + e^{-net_{jk}}}$$

La elección de esta función depende de la forma en que se decida representar los datos: si se desea que las neuronas de salida sean binarias, se utiliza la función sigmoidea, puesto que la función es casi biestable y además, derivable.

- **Paso 4:**

  Después que todas las neuronas de la red tienen un valor de activación asociado para un patrón de entrada dado, el algoritmo continúa encontrando el error que se presenta para cada neurona, excepto las de la capa de entrada. Para la neurona *k* de la capa de salida, si la respuesta es (y_1, y_2,..., y_M), dicho error (d) se puede escribir como:

$$\delta_{pk} = \left(d_{pk} - y_{pk}\right) f'_{k}\left(net_{pk}\right)$$

y para la función sigmoidea en particular:

$$\delta_{pk} = \left(d_{pk} - y_{pk}\right) y_{pk}\left(1 - y_{pk}\right)$$

Si la neurona *j* no es de salida, entonces la derivada parcial del error no puede ser evaluada directamente. Por tanto, se obtiene el desarrollo a partir de valores que son conocidos y otros que pueden ser evaluados. La expresión obtenida en este caso es:

$$\delta^{h}_{pj} = x_{pi}\left(1 - x_{pi}\right)\sum_{k} \delta_{pk} w_{kj}$$

donde observamos que el error en las capas ocultas depende de todos los términos del error de la capa de salida. De aquí el nombre de propagación hacia atrás.

El error que se produce en una neurona oculta es proporcional a la suma de los errores conocidos que se producen en las neuronas a las que está conectada la salida de la misma, multiplicando cada uno por el peso de la conexión. Los umbrales internos de las neuronas se

adaptan de forma similar, considerando que están conectados con pesos desde entradas auxiliares de valor constante.

- **Paso 5**

Para la actualización de los pesos utilizamos el algoritmo recursivo, comenzando por las neuronas de salida y trabajando hacia atrás hasta llegar a la capa de entrada, ajustando los pesos de la forma siguiente:

a - Para los pesos de las neuronas de la capa de salida:

$$w_{kj}(t+1) = w_{kj}(t) + \Delta w^{o}_{kj}(t+1)$$
$$\Delta w^{o}_{kj}(t+1) = \alpha \delta_{pk} y^{h}_{pj}$$

b - Para los pesos de las neuronas de la capa oculta:

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w^{h}_{ji}(t+1)$$
$$\Delta w^{h}_{ji}(t+1) = \alpha \delta^{h}_{pj} y_{pi}$$

En ambos casos, para acelerar el proceso de aprendizaje, se incluye una tasa de aprendizaje que varía entre 0 y 1, dependiendo del problema a solucionar. También se puede añadir un término de momento, para corregir la dirección del error, de valor $\gamma\left(w_{kj}(t) - w_{kj}(t-1)\right)$ en el caso de una neurona de salida, y $\gamma\left(w_{ji}(t) - w_{ji}(t-1)\right)$, cuando se trata de una neurona oculta; la constante g es la tasa de momento; y un tercer parámetro b para acelerar la convergencia del proceso (Campbell y Temporel, 2002).

- **Paso 6**

El proceso se repite hasta que el error medio cuadrático (E_p)

$$E_p = \frac{1}{2}\sum_{k=1}^{M}\left(\delta_{pk}\right)^2$$

resulta aceptablemente pequeño para cada uno de los patrones aprendidos. Este estadístico se utiliza para la evaluación objetiva del desempeño de la RN.

Ejemplo

Perceptrón

La red tipo perceptrón fue inventada por el psicólogo Frank Rosenblatt en el año 1957. El objetivo era mostrar las propiedades principales de los sistemas inteligentes en general. Rosenblatt creía que la conectividad en las redes biológicas tenía un alto grado de aleatoriedad, y era contrario al análisis de McCulloch Pitts en el que se empleaba lógica simbólica para analizar estructuras bastante idealizadas. Para Rosenblatt, la mejor herramienta de análisis era la teoría de probabilidades, que lo condujo a una teoría de separabilidad estadística para caracterizar las propiedades más visibles de estas redes de interconexión ligeramente aleatorias.

El primer modelo de perceptrón trataba de imitar al ojo humano (fotoperceptrón), y respondía a señales ópticas. La luz llega a los puntos sensibles (S) de la retina y estos responden generando un impulso hacia las unidades de asociación (A). Cada unidad de asociación esta conectada a un conjunto aleatorio de puntos S y a las unidades de respuesta (R). Inicialmente no estaba preparado para distinguir patrones de entrada muy complejos, pero con entrenamiento era capaz de adquirir esta capacidad. El entrenamiento implicaba un proceso de refuerzo mediante el cual la salida de las unidades **A** incrementaba o decrementaba dependiendo de si participaban o no en las respuestas correctas del perceptrón para una entrada dada. Se aplicaba una entrada (estímulo) a la retina, que se propagaba a través de las capas hasta que se activaba una unidad de respuesta. Si la unidad **R** activada era la correcta se incrementaba la salida de las unidades **A** que hubieran contribuido. Si se activaba una unidad **R** incorrecta, se hacía disminuir la salida de las unidades **A** que hubiesen contribuido.

Gracias a estas investigaciones, se pudo comprobar que el perceptrón era capaz de clasificar patrones correctamente.

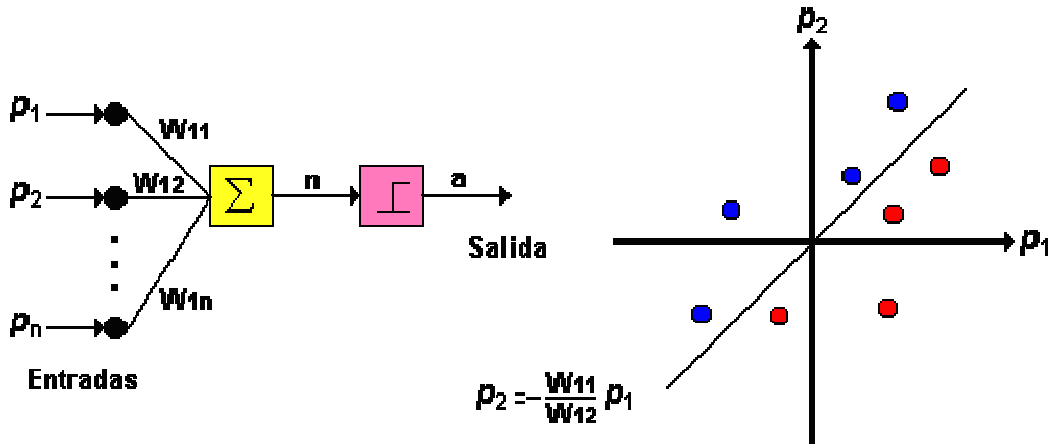

Figura 5. Red perceptrón [ohm]

La función de salida es llamada umbral o de transferencia f(salida)= 1 si salida ≥ Θ ó 0 si salida ≤ Θ, donde Θ es el valor umbral.

La única neurona de salida del perceptrón realiza la suma ponderada de las entradas, resta el umbral y pasa el resultado a una función de transferencia de tipo escalón. La red de tipo perceptrón emplea principalmente 2 funciones de transferencia:

- Hardlim, con salidas 1 ó 0
- Hardlims, con salidas 1 ó -1

El uso de cada una depende del valor de salida que se espera de la red, es decir, si va a ser bipolar o unipolar, aunque ***hardlims*** es preferida sobre ***hardlim*** ya que el tener un 0 multiplicando alguno de los valores resultantes del producto de las entradas del vector de pesos ocasiona que estos no se actualicen y el aprendizaje sea mas lento.

El perceptrón es un tipo de red de aprendizaje supervisado, ya que necesita conocer los valores esperados para cada una de las entradas presentadas. Su comportamiento está definido por pares del tipo $\{p_1, t_1\}, \{p_2, t_2\}\ldots\{p_n, t_n\}$.

Perceptrón multicapa

Este es uno de los tipos de redes más comunes. Se basa en la red "Perceptrón" solo que el número de capas ocultas puede ser mayor o igual que una. Es una red unidireccional (feedforward).

La arquitectura de la red perceptrón multicapa es la siguiente:

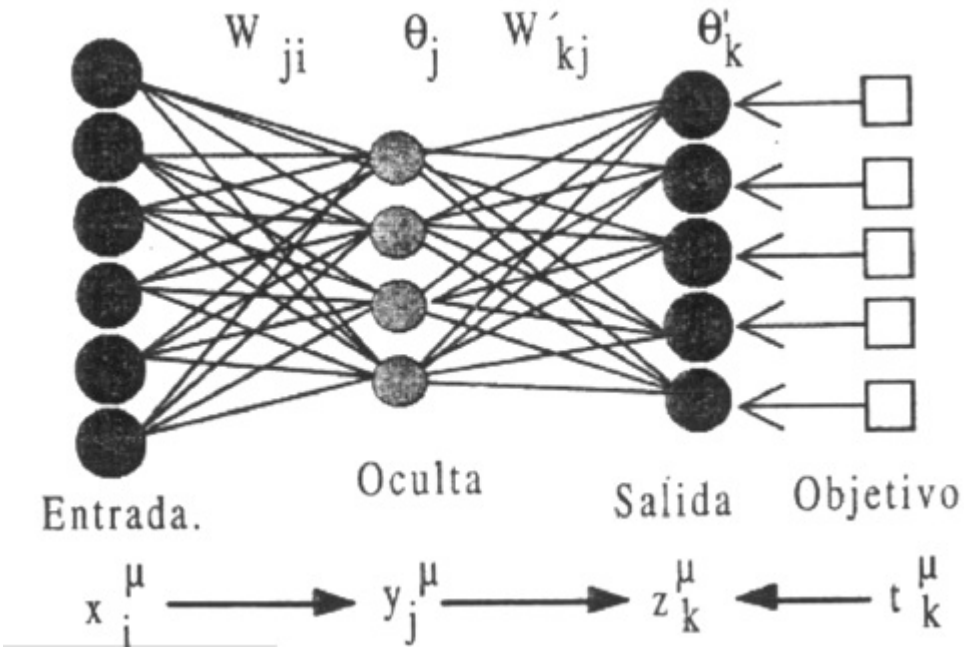

Figura 6. Red de tipo perceptrón [softwarelibre]

Puede ser totalmente o localmente conectado. En el primer caso cada salida de una neurona de la capa "i" esta conectada a la entrada de todas las neuronas de la capa siguiente "i+1", mientras que en el segundo caso, cada neurona de la capa "i" esta conectada a la entrada de una serie de neuronas (región) de la capa "i+1".

Las capas se dividen en:

- **Capa de entrada:** La forman las neuronas que introducen los patrones de entrada en la red. En estas neuronas no se produce procesamiento.
- **Capa oculta:** Formada por las neuronas cuyas entradas provienen de capas anteriores y las salidas se conectan a neuronas de capas posteriores.
- **Capa de salida:** Neuronas cuyos valores de salida se corresponden con las salidas de toda la red.

El algoritmo utilizado en este tipo de redes es la retropropagación del error o regla delta generalizada.

La red perceptrón multicapa tiene una serie de inconvenientes:

1. Mala extrapolación, ya que si la red no se entrena lo suficiente o se entrena mal, las salidas pueden ser imprecisas.
2. La existencia de mínimos locales en la función de error dificulta el entrenamiento, ya que una vez alcanzado un mínimo el entrenamiento se detiene aunque no se haya alcanzado la tasa de convergencia fijada.

El Algoritmo de retropropagación sería el siguiente:

1. Inicializar los pesos aleatoriamente y con valores menores que 1.

2. Escoger un patrón del conjunto de entrenamiento, calcular la salida (para esto se avanza neurona por neurona, capa por capa, hasta llegar a la salida, al igual que en el PS) y el error asociado en cada nodo de la capa de salida.

3. Propagar la señal hacia adelante

   Calcula el error en la capa de salida

   $$\delta_i(t) = f'\big(x_i(t)\big) * (d_i - y_i)$$

   Propagar dicho error hacia las neuronas ocultas (hacia atrás)

   $$\delta_j(t) = f'\big(x_i(t)\big) * \Big(\sum_i w_{ij}\, d_i(t)\Big)$$

   Actualizar los pesos utilizando

   $$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}(t+1)$$

   dónde

$$\Delta w_{ji}(t+1) = [n\delta_j y_i(t) + \alpha \Delta w_{ji(t)}]$$

Repetir el paso 2 hasta alcanzar el error deseado.

Generalmente estos pasos se realizan un número determinado de veces, llamadas épocas de entrenamiento y luego se observa si la red aprendió.

No hay un límite a la hora de fijar la cantidad de capas de un perceptrón multicapa, pero se ha demostrado que con una capa oculta y un número suficiente de nodos, es capaz de solucionar casi cualquier problema. Si se agrega una capa oculta más, el perceptrón multicapa soluciona cualquier tipo de problemas y de forma más eficiente que con una sola.

ALGORITMOS BASADOS EN REDES BAYESIANAS

Introducción

Los clasificadores Bayesianos son clasificadores estadísticos. Pueden predecir tanto las probabilidades del número de miembros de clase, como la probabilidad de que una muestra dada pertenezca a una clase particular. Los clasificadores bayesianos se basan en el teorema de Bayes. Su mayor ventaja es que proporcionan una gran exactitud y velocidad aplicados a grandes cantidades de datos.

El algoritmo que se ha utilizado para realizar la clasificación en este proyecto ha sido el conocido como "**Naive Bayes**"

Naive Bayes [ehu]

El abanico de algoritmos de clasificación en el que se utiliza el teorema de Bayes conjuntamente con la hipótesis de independencia condicional de las variables predictorias incluye los idiota Bayes, naive Bayes, simple Bayes y Bayes independiente.

La primera vez que se utiliza el clasificador naive Bayes es a finales de los ochenta, y se hace con el objetivo de comparar su capacidad predictiva con la de otros métodos.
Gradualmente ha ido abriéndose hueco gracias a potencia y robustez en problemas de clasificación supervisada.
El nombre viene de las hipótesis tan simplificadoras sobre las que se construye dicho clasificador. Partimos del ejemplo clásico de diagnóstico para que una vez que comprobemos que necesita la estimación de un gran número de parámetros, ir simplificando progresivamente las hipótesis sobre las que se construye hasta llegar al modelo naive Bayes.

La presencia o ausencia de una característica particular de una clase no está relacionada con la presencia o ausencia de cualquier otra característica.

Según la naturaleza exacta del modelo de probabilidad, el clasificador naive Bayes puede ser entrenado muy eficientemente en un entorno de aprendizaje supervisado. En muchas aplicaciones prácticas, los parámetros de estimación para modelos naive Bayes utilizan el método de máxima verosimilitud. Esto significa que se puede trabajar con el modelo naive Bayes sin creer en la probabilidad bayesiana o usando algún método bayesiano.

Una ventaja del clasificador naive Bayes es que requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios para la clasificación.

Si llamamos ***P(D)*** a la probabilidad a priori de los datos (qué datos son más probables que otros), llamamos ***P(D|h)*** a la probabilidad de los datos dada una hipótesis, lo que queremos calcular es ***P(h|D)***, es decir, la probabilidad posterior de ***h*** dados los datos. Para estimar esto, utilizamos el teorema de Bayes: [ua]

$$P(h|D) = \frac{P(h) * P(D|h)}{P(D)}$$

La regla común es elegir la hipótesis que es más probable, lo que se conoce como el máximo a posteriori o regla de decisión MAP.

$$h_{MAP} = argmax_{h \in H}((P(h|D))$$

$$h_{MAP} = argmax_{h \in H}(\frac{P(h) * P(D|h)}{P(D)})$$

$$h_{MAP} = argmax_{h \in H}(P(D|h) * P(h))$$

En el último paso hemos eliminado P(D) porque es independiente de h.

En algunos casos, todas las probabilidades en H son igualmente probables a priori:

$$P(h_{i)} = P\big(h_j\big) \forall h_i, h_j \in H$$

En este caso, sólo utilizaremos el término de verosimilitud, P(D|h), y podemos simplificar aún más la anterior ecuación:

$$h_{ML} = argmax_{h \in H} P(D|h)$$

donde a la hipótesis h_{ML} se le suele nombrar como hipótesis de máxima verosimilitud (Maximum Likelihood).

Supongamos ahora que debemos elegir entre dos hipótesis, h_1 y h_2, dados los datos D. El criterio de elección para responder de forma eficiente sería seleccionar la hipótesis más probable. Es decir, aplicaríamos lo que se conoce como regla de decisión:

si $P(h_1|D) > P(h_2|D)$ elegir h_1, sino elegir h_2

Si aplicamos la regla de Bayes a cada término nos queda:

$$P(D|h_1) * \frac{P(h_1)}{P(D)} > P(D|h_2) * \frac{P(h_2)}{P(D)}$$

$$\underbrace{\frac{P(D|h_1)}{P(D|h_2)}}_{ratio\ verosimilitud} > \underbrace{\frac{P(h_2)}{P(h_1)}}_{ratio\ a\ priori}$$

Aplicando logaritmos a ambas partes nos queda:

$$\ln \frac{P(D|h_1)}{P(D|h_2)} > \ln \frac{P(h_2)}{P(h_1)}$$

En ausencia de información a priori todas las hipótesis son igualmente probables y el término de la derecha es ln 1=0. La regla de decisión en ausencia de información a priori queda:

si

$$\ln \frac{P(D|h_1)}{P(D|h_2)} > 0$$

elegir h_1, si no, elegir h_2

Los clasificadores naive Bayes asumen que el efecto de un valor del atributo en una clase dada es independiente de los valores de los otros atributos. Esta suposición se llama "independencia condicional de clase". De este modo se simplifican los cálculos involucrados y es considerado "ingenuo" (naive). A pesar de lo que indica su nombre y de la simplificación realizada, el clasificador naive Bayes funciona muy bien, sobre todo cuando se filtra el conjunto de atributos seleccionado para eliminar redundancia, eliminando a su vez dependencia entre los datos. [weka]

2.4 APUESTAS ON LINE

2.4.1 CASAS DE APUESTAS

Una casa de apuestas, por definición, es un edificio público en el cual se puede jugar a una gran variedad de juegos de azar. Internet ha modificado este término, ya que ahora no es necesario estar presente físicamente en el lugar donde se realizan las apuestas.

Una casa de apuestas online es un portal en el que se pueden realizar apuestas en los diferentes mercados que se ofrezcan.

Para ello, es necesario un registro del usuario en el portal, y un depósito de dinero con el que poder apostar.

Actualmente en Internet existen decenas de casas de apuestas, un punto a favor de los usuarios, ya que pueden elegir la que les convenga más o disponer de varias de ellas para así poder elegir cuotas mejores, bonos de bienvenida y ofertas, que no es otra cosa que dinero gratis para realizar apuestas.

2.4.1.1 TRADICIONALES

La mayoría de apuestas en EEUU se realizan en deportes universitarios y profesionales, pero en el Reino Unido se ofrecen una gama más amplia de apuestas, incluyendo las apuestas de tipo each-way de golf, fútbol, y tenis, y especialmente las carreras de caballos y eventos de galgos. Una apuesta de tipo each-way es aquella ofrecida por la casa de apuestas que consta de 2 apuestas, una de tipo “Ganar” y otra de tipo “Lugar”. Por la parte del tipo de apuesta “Ganar”,la selección del apostante tiene que ganar o terminar primero en el evento, mientras que por la parte de tipo “Lugar”, la selección del apostante debe ganar o terminar en una de las posiciones predeterminadas para el evento.

Por curiosidad, también podemos encontrar eventos para apostar tales como si nevará el día de navidad, el resultado en unas elecciones políticas y los ganadores de los reality shows en televisión como “Gran Hermano” o “Factor X”.

Mediante el ajuste de la cuotas, la casa de apuestas tiene como objetivo garantizar el logro de un beneficio, ya sea obteniendo un número igual de apuestas para cada resultado, u (cuando se están ofreciendo cuotas) obteniendo los importes apostados en cada resultado para reflejar las cuotas. Cuando entra una apuesta grande, la casa de apuestas puede tratar de disminuir el riesgo mediante la compra de apuestas a otras casas de apuestas.

Las apuestas pueden ser legales o ilegales, y deben estar reguladas. En el Reino Unido a veces han sido tanto reguladas como ilegales. Desde el inicio de la Lotería nacional, son completamente legales y además con un pequeño contribuyente a la economía británica, con una reciente explotación de interés en lo que respecta a la industria internacional del juego.

Las apuestas son generalmente ilegales en los EEUU, siendo el estado de Nevada una notable excepción.

En España, el mundo de las apuestas es completamente ilegal, aunque se están empezando a conceder licencias a nivel de comunidades autónomas, como por ejemplo en Madrid, donde una licencia de casa de apuestas puede llegar a costar alrededor de 2 millones de euros.

En algunos países, como Singapur, Suecia, Canadá y Hong Kong, la única casa de apuestas es propiedad del estado y esta operada por éste. En Canadá, es parte del programa de lotería y se conoce como *Sport Select*.

En el Reino Unido, las casas de apuestas legales son miembros del IBAS, que es una organización estándar de la industria, la cual intercede en la solución de disputas.

Tradicionalmente, las casas de apuestas y los corredores se localizan en los hipódromos, pero la mayor cobertura televisiva y la modernización de la ley han permitido realizar apuestas en tiendas y casinos.

En el Reino Unido, los corredores de apuestas aun apuntan las cuotas en pizarras en los lugares donde se producen los eventos deportivos, y comunican las cuotas por teléfono a miembros de su personal o a otros corredores de apuestas, pero con la modernización de las leyes de apuestas, las apuestas y los juegos de azar online han experimentando un aumento histórico y se ha planeado un llamado "Super Casino" en Manchester, pero el gobierno británico ha desestimado la propuesta.

En 1961, Harold MacMillan, del gobierno conservador, legaliza las casas de apuestas y promulga medidas estrictas para asegurar la honestidad de estas. Una gran y respetable industria ha crecido desde entonces. En un momento, hubo más de 15000 casas de apuestas en el Reino Unido. Ahora se han reducido a unas 8500. Actualmente existen 4 grandes casas de apuestas en el Reino Unido: William Hill, Ladbrokes, Coral y ToteSport (de propiedad estatal), con Sky Bet, Bet24, Betfred, Victor Chandler, Stan James, Sportingbet, Mansion y Bet365, en rápido ascenso, en términos de volumen de negocios y eventos de patrocinio.

Con la llegada de Internet, la mayoría de las casas de apuestas tienen una marca online. La mayoría de las casas de apuestas solo aceptan apuestas de países en los cuales apuestas por online están permitidas, y de usuarios mayores de 18 años. A menudo, estos sitios web están vinculados a casinos online.

En controversia con esto, las apuestas online están siendo vinculadas a un aumento de la adicción al juego por las organizaciones de ayuda y asesoramiento para los adictos del Reino Unido, como "Gamecare" y "Gamblers anonymous".

2.4.1.2 INTERCAMBIO

Cada vez más, los apostantes online están recurriendo a la utilización de intercambio de apuestas, como Betfair y BETDAQ, las cuales automáticamente igualan las apuestas a favor y en contra entre diferentes apostantes, lo que definitivamente rompe con los márgenes de beneficio de las casas de apuestas tradicionales, también llamado "Overround" (apostar 'en contra' en los distintos resultados posibles de un único evento).

Estos mercados online de intercambio operan en un índice de mercado de precios cercano pero normalmente no al 100% de la competitividad.

Algunas casas de apuestas incluso han optado por tomar el intercambio de apuestas como una manera de anular apuestas desfavorables y así reducir su exposición global. Esto ha llevado a la inseguridad de la TAB en Australia, una agencia de apuestas del gobierno la cual intentó denegar a Betfair una licencia por medio de publicaciones de anuncios desfavorables relacionados con la empresa en los medios de comunicación. Cuando Tasmania concedió a Betfair una licencia a pesar de estos esfuerzos, la legislatura estatal de Australia Occidental aprobó una ley que penalizaba específicamente el uso de intercambio de apuestas en el estado. Sin embargo, más tarde la ley fue declarada anticonstitucional.

El intercambio de apuestas es universalmente rechazado por las casas de apuestas tradicionales. No sólo están en condiciones de ofrecer a los apostantes unas mejores cuotas debido a un menor gasto general, sino que también ofrecen oportunidades de arbitraje (tomar ventaja de una diferencia de precios entre dos o más mercados), aunque tradicionalmente el arbitraje ha sido siempre posible mediante una apuesta a favor en todos los mercados siendo la casa de apuestas el contrincante que apuesta en contra a estos resultados en un intercambio. Los intercambios permiten a las casas de apuestas comprobar el estado de los mercados y establecer sus cuotas en consecuencia.

Las apuestas se pueden hacer por vía telefónica o mediante envíos de mensajes de texto (SMS), pero para el póker y otros deportes es mejor utilizar otros medios de comunicación. A medida que avanza la tecnología, el mundo de las apuestas es un actor importante en las operaciones de nuevas tecnologías.

El deporte más televisado en el Reino Unido y Europa está patrocinado en su totalidad, o en parte, por Internet y casas de apuestas, y a veces algunas casas de apuestas y casinos online se muestran en las camisetas de los jugadores, vallas publicitarias, nombre de estadios y nombres de competiciones, aunque el equipo alemán del Werder Bremen está actualmente litigando en los tribunales alemanes para poder seguir llevando la publicidad de la casa de apuestas **Bwin** en su camiseta, ya que Alemania y Francia han tomado medias contra los jugadores online.

Con la reciente prohibición del patrocinio del tabaco, y el importante presupuesto comercial a disposición de la industria del juego, el patrocinio de los fabricantes de automóviles, bebidas alcohólicas, refrescos y restaurantes de comida rápida se está rápidamente sustituyendo por el patrocinio de compañías de juego en el Lejano Oriente y Europa.

La ley del 2005 de juego en el Reino Unido introduce un nuevo sistema de regulación de los juegos de azar en Gran Bretaña. Este sistema incluye nuevas disposiciones para regular la publicidad de los productos de juegos de azar, que entraron en vigor en Septiembre de 2007. Es un delito en el Reino Unido anunciar juegos de azar que físicamente no tienen lugar en un país de la comunidad económica europea, o en el caso de los juegos de azar por medios remotos, el juego que no está regulado por las leyes de juego de un Estado de la comunidad económica europea.

La situación es más confusa en los Estados Unidos, que ha intentado restringir a los operadores de sitios web de apuestas extranjeros el acceso a su mercado doméstico. Esto ha dado lugar a una sentencia contra el Gobierno de los EE.UU. por la WTO.

2.4.2 SISTEMAS DE PAGO SEGURO

Los sistemas de pago seguros han obtenido una importancia muy alta en internet, tanto para compras online, como transferencias de dinero, como, en el caso que nos ocupa, para la realización de apuestas por internet. Hoy en día hay varias pasarelas de pago online, como por ejemplo Ukash, Paypal, Paysafecard, Moneybookers, y la que vamos a utilizar para la realización de este proyecto, como Clickandbuy.

2.4.2.1 CLICKANDBUY

En 1999 Norbert Stangl fundó la empresa FIRSTGATE Internet AG. El sistema de pagos por Internet ClickandBuy, una solución de servicio global para pagos electrónicos y facturación, fue introducido en el año 2000. Tras un inicio exitoso, ClickandBuy se ha introducido en otros mercados y actualmente está presente en toda Europa, en EE.UU. y en Asia. La sede central está ubicada en Londres.

A primeros de 2007, Intel Capital, empresa de fondo de capital de riesgo, realizó una inversión en ClickandBuy, creando una asociación entre las dos compañías que permitió intensificar la expansión de ClickandBuy a nivel internacional para abrir nuevos mercados.

En 2006, ClickandBuy recibió dos inversiones de mayor envergadura. La primera provino de 3i, un fondo importante de capital de riesgo a nivel global con un volumen de inversión de 20 millones de euros destinados a hacerse con una participación en ClickandBuy. La segunda se recibió por parte de T-Online Venture Fund con una inversión que le permitió adquirir un 10 por ciento de ClickandBuy. La asociación con T-Online permitió a ClickandBuy aumentar el atractivo de su oferta en el mundo de la telecomunicación.

ClickandBuy es uno de los líderes entre los sistemas de pago en Internet. Más de 14.000 proveedores usan los servicios que proporciona clickandbuy en 26 países, 126 con 46 métodos de pago. iTunes, Antena 3, Skype, motoGP, Marca y muchas otras empresas usan con éxito clickandbuy [Clickandbuy].

Clickandbuy permite a los clientes realizar compras de contenidos digitales y otros servicios en internet con la máxima seguridad y confort. Los servicios que incluye clickandbuy son los siguientes:

- Registro único y gratuito en línea.
- Pagos seguros.
- Acceso inmediato a contenidos de pago de 14.000 proveedores con un solo clic.
- Pago mensual por cargo en cuenta, tarjeta de crédito, factura o cuenta prepago.
- Información de las operaciones efectuadas en tiempo real.
- Transparencia y control de los costes en su administración personal.

- Información general actual sobre las ofertas aún abiertas, las suscripciones actuales y las facturas.
- Atención al cliente ininterrumpida siete días a la semana.

ClickandBuy recopila la información siguiente, cuando paga algo con ClickandBuy de ClickandBuy: El proveedor al que compra, la descripción del contenido o el producto, el precio, la fecha y la hora, y la duración del periodo de acceso contratado. Con esta información se elabora un informe que puede ser de utilidad para pagar las transacciones, ya que incluye todas las compras que ha realizado un usuario. Se puede consultar el historial de gestión de cobros de ClickandBuy en cualquier momento desde la gestión de cuentas "Mi cuenta" a la que se puede acceder con el identificador y la contraseña de un usuario.

Los centros informáticos disponen de servidores seguros y enlazados mediante varios backbones protegidos contra caídas del sistema. Los sistemas y las redes se supervisan las 24 horas del día para garantizar un funcionamiento sin interrupciones. Todos los datos se codifican con la tecnología SSL siempre que se realiza una transferencia de datos entre un usuario y clickandbuy a través de internet.

Datos de registro

Para poder utilizar ClickandBuy sin restricciones y registrarse, el usuario deberá facilitar ciertos datos. El usuario consiente expresamente que los datos introducidos (nombre, fecha de nacimiento, dirección postal, dirección de correo electrónico, número de teléfono, datos de la cuenta bancaria o de la tarjeta de crédito/débito, así como el protocolo de acceso a los servicios del proveedor) puedan ser cedidos al proveedor en caso de reclamación por los servicios recibidos. Asimismo, los datos podrán ser cedidos al proveedor en caso de impagos o anulación de pagos al mencionado proveedor. De igual manera, ClickandBuy podrá ceder datos personales al proveedor cuando el usuario realice una suscripción o cuando el proveedor así lo establezca como requisito indispensable para adquirir la oferta. En caso de servicios móviles y ofertas enviadas a través de teléfono móvil, ClickandBuy podrá facilitar datos personales al proveedor. Los datos personales no serán utilizados para realizar comprobación alguna de la solvencia económica. Los datos personales se recaban, tratan y almacenan de acuerdo con las disposiciones de protección de datos vigentes. Dicho tratamiento está exclusivamente destinado a la prestación de los servicios del sistema ClickandBuy y a la tramitación de los pagos por cuenta del proveedor. ClickandBuy podrá asimismo realizar otras comunicaciones de datos previstas en la ley. Se excluye la cesión a terceros en cualesquiera otros casos. El titular de los datos tiene derecho a acceder en todo momento a sus datos personales, rectificarlos o, en su caso, cancelarlos y oponerse a ellos, mediante la correspondiente comunicación enviada a ClickandBuy.

Cookies

Las cookies son archivos pequeños que facilitan la utilización de clickandbuy. Un explorador de Internet obtiene estos archivos automáticamente desde el servidor de clickandbuy y almacena las cookies en el ordenador del usuario. ClickandBuy utiliza dos tipos de cookies, las denominadas cookies de sesión y las cookies permanentes. Clickandbuy envía una cookie de sesión al ordenador del cliente cuando inicia sesión en su cuenta de ClickandBuy en la que introduce su dirección de correo electrónico y su contraseña. Esto le reporta la siguiente ventaja: Si contrata varios proveedores de ClickandBuy, podrá localizarlos de nuevo en cualquier momento con ayuda de las cookies, por lo que no será necesario que vuelva a introducir su contraseña cada vez. Así, podrá utilizar los servicios de clickandbuy con mayor comodidad y rapidez. Cuando finaliza la sesión o cierra el explorador esta cookie se borra y deja de tener efecto. Asimismo, clickandbuy utiliza una cookie permanente. Esto tiene la ventaja de que la dirección de correo electrónico del usuario aparece indicada constantemente en el formulario de registro. De esta forma se evitará tener que introducir dicha dirección de correo electrónico cada vez que abra una sesión en la administración de la cuenta de ClickandBuy. Los datos que contiene la cookie están codificados y no pueden ser leídos por terceros.

Los pagos que se realizan con Clickandbuy pueden ser de varias modalidades, tales como cargo en la cuenta, tarjeta de crédito, factura o cuenta prepago. El usuario tiene un saldo en su cuenta de Clickandbuy, y se pueden hacer tanto ingresos en esta cuenta a favor del cliente como retirada de saldo si ha efectuado una compra. La manera de realizar esto es mediante la utilizacion de *WebServices* (servicios web). Un servicio web es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Son muy útiles, ya que distintas aplicaciones realizadas en distintos lenguajes pueden acceder a los servicios web de Clickandbuy para intercambiar datos. Clickandbuy es uno de los sistemas de pago líderes en Internet, y muy utilizado en el mundo de las apuestas (Bwin y Betfair lo utilizan).

Para establecer la conexión con clickandbuy, debemos formar un mensaje SOAP con los datos requeridos para poder efectuar una transacción. Un ejemplo de este XML lo podemos ver en la siguiente figura:

```
<?xml version='1.0' encoding='UTF-8'?>
    <soapenv:Envelope
xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsd='http://www.w3.org/2001/XMLSchema'
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
        <soapenv:Body>
          <getEasyCollectSingle xmlns='TransactionManager.Payment'>
            <sRequest>
              <sellerID>18811928</sellerID>
              <tmPassword>alcazaba</tmPassword>
              <extJobID></extJobID>
              <request>
                <discriminator>CREDIT</discriminator>
                <debReq><crn>101710016 </crn>
                <easyCollectID>34340962</easyCollectID>
                <externalBDRID>1272</externalBDRID>
                <amount>100</amount>
                <currency>EUR</currency>
                <urlInfo>PARSHIP.gb</urlInfo>
              <internalContentDescription>
                        test credit
               </internalContentDescription>
                      </debReq>
                    </request>
                  </sRequest>
                </getEasyCollectSingle>
              </soapenv:Body>
            </soapenv:Envelope>
```

Para las transacciones mediante dispositivo móvil, el mensaje SOAP enviado a clickandbuy difiere un poco, como podemos observar en el siguiente ejemplo:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <soapenv:Body>
          <getEasyCollectSingle xmlns="TransactionManager.Payment">
          <sRequest>
                <sellerID>222333444</sellerID>
                <tmPassword>YourPassword</tmPassword>
                <extJobID>123</extJobID>
                <request>
                        <discriminator>DEBIT</discriminator>
                         <debReq>
                        <slaveMerchantID>0</slaveMerchantID>
                         <crn>111555000</crn>
                        <easyCollectID>12345</easyCollectID>
                         <externalBDRID>abc1111111</externalBDRID>
                         <amount>1</amount>
                         <currency>EUR</currency>
                         <urlInfo>Info</urlInfo>
                        <internalContentDescription>Description</internalContent
                Description>
                         </debReq>
                </request>
           </sRequest>
          </getEasyCollectSingle>
      </soapenv:Body>
</soapenv:Envelope>
```

Tabla 13. Mensaje SOAP enviado mediante dispositivo móvil

Como se puede observar, se deben enviar una serie de parámetros para establecer la transacción, tales como:

- sellerID: Número de cuenta del usuario. Este número identifica a un usuario en el sistema de clickandbuy.
- tmPassword: Es la contraseña de "transaction manager". Es necesaria para todas las transacciones.
- discriminator: Indica el tipo de transacción que vamos a realizar, ya sea depósito (debit) o reintegro (credit).
- slaveMerchantID: Son los sellersID para los "distribuidores" de un usuario. Si un usuario no ha creado "distribuidores", se asigna el valor 0.
- crn: Es el identificador de usuario en clickandbuy al que se le va a hacer el cobro.
- easyCollectID: Es la autorización de cobro creada en la cuenta del comercio.
- externalBDRID: Es un identificador único de transacción creado por el usuario. Se debe enviar un externalBDRID único cada vez que se hace una transacción.
- amount: Montante de la transacción.
- currency: Moneda en la que se está trabajando.

Estos son los parámetros más importantes que se utilizan al realizar una transacción con clickandbuy, aunque puede haber otros dependiendo de si nos queremos registrar en el sistema mediante mensajes SOAP, o si queremos realizar otro tipo de operaciones con el sistema de clickandbuy.

La respuesta que nos devuelve clickandbuy, si la transacción fue posible, es enviada mediante otro mensaje SOAP, similar al siguiente:

```
<SOAP-ENV:Envelope
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
  <namesp1:getEasyCollectSingleResponse
xmlns:namesp1="TransactionManager.Payment">
  <return>
   <discriminator>DEBIT</discriminator>
                    <debResp>
   <BDRID>888777</BDRID>
          <externalBDRID>abc1111111</externalBDRID>
                        <amount>1</amount>
          <currency>EUR</currency>
   <systemID>1</systemID>
          <explicitCommit>0</explicitCommit>
          <paidAmount>1</paidAmount>
   <billedAmount>1</billedAmount>
          <billedcurrency>EUR</billedCurrency>
          </debResp>
  </return>
  </namesp1:getEasyCollectSingleResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Tabla 14. Respuesta de transacción satisfactoria de clickandbuy

Como vemos en este XML, la transacción ha tenido éxito. El externalBDRID debe ser el mismo que nosotros enviamos al iniciar la transacción, y la cantidad recibida debe ser también la misma. Esta cantidad se indica en el parámetro billedAmount. En el parámetro paidAmount nos indica la cantidad que se ingresado. Si las cantidades son las mismas, la transacción completa ha tenido éxito.

Si la transacción ha sido incorrecta, debemos recibir una mensaje SOAP igual o parecido a este:

```
<?xml version=\"1.0\" encoding=\"UTF-8\"?>
 <SOAP-ENV:Envelope
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xmlns:namesp4355=\"http://namespaces.soaplite.com/perl\"
xmlns:SOAP-ENC=\"http://schemas.xmlsoap.org/soap/encoding/\"
xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"
SOAP-ENV:encodingStyle=\"http://schemas.xmlsoap.org/soap/encoding/\"
xmlns:SOAP-ENV=\"http://schemas.xmlsoap.org/soap/envelope/\">
    <SOAP-ENV:Body>
       <SOAP-ENV:Fault>

<faultcode>SOAP-ENV:TransactionManager.Payment.PaymentException</faultcode>
          <faultstring>Exception raised in:
encodePaymentRequest2Iso</faultstring>
          <detail>
             <TransactionManager.Payment.PaymentException
xsi:type=\"namesp4355:TransactionManager.Payment.PaymentException\">
                <message
xsi:type=\"xsd:string\">InvalidParameter</message>
                <id xsi:type=\"xsd:int\">34</id>
             </TransactionManager.Payment.PaymentException>
          </detail>
          <faultactor>https://clickandbuy.com/</faultactor>
       </SOAP-ENV:Fault>
    </SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

Tabla 15. Respuesta de transacción incorrect de clickandbuy

Como podemos observar, nos devuelve un parámetro llamado faultcode, que nos indica la operación en la que se ha producido el error. También nos devuelve otro parámetro llamado faultstring que indica el tipo de error que hemos obtenido.

El formato estándar de los datos es XML (eXtensible Markup Language) y la información que contendrá está relacionada con datos de identificación de la cuenta del usuario, como por ejemplo un identificador de transacción, el montante de la operación, la moneda utilizada, etc.

El proceso es bastante sencillo. Un usuario simplemente selecciona como sistema de pago C&B. El sistema le proporciona un identificador de usuario C&B, con el que será reconocido cada vez que haga transferencias con este sistema. El usuario introduce la

cantidad que quiere ingresar en su cuenta de la casa de apuestas y Clickandbuy ingresa ese dinero en la cuenta del usuario.

Previamente, tanto la casa de apuestas como Clickandbuy han tenido que establecer un acuerdo por el que la casa de apuestas obtiene un identificador para ser reconocida cuando se redirige a la plataforma de pago.

A continuación, podemos observar un diagrama de actividad para el sistema de transacciones de clickandbuy:

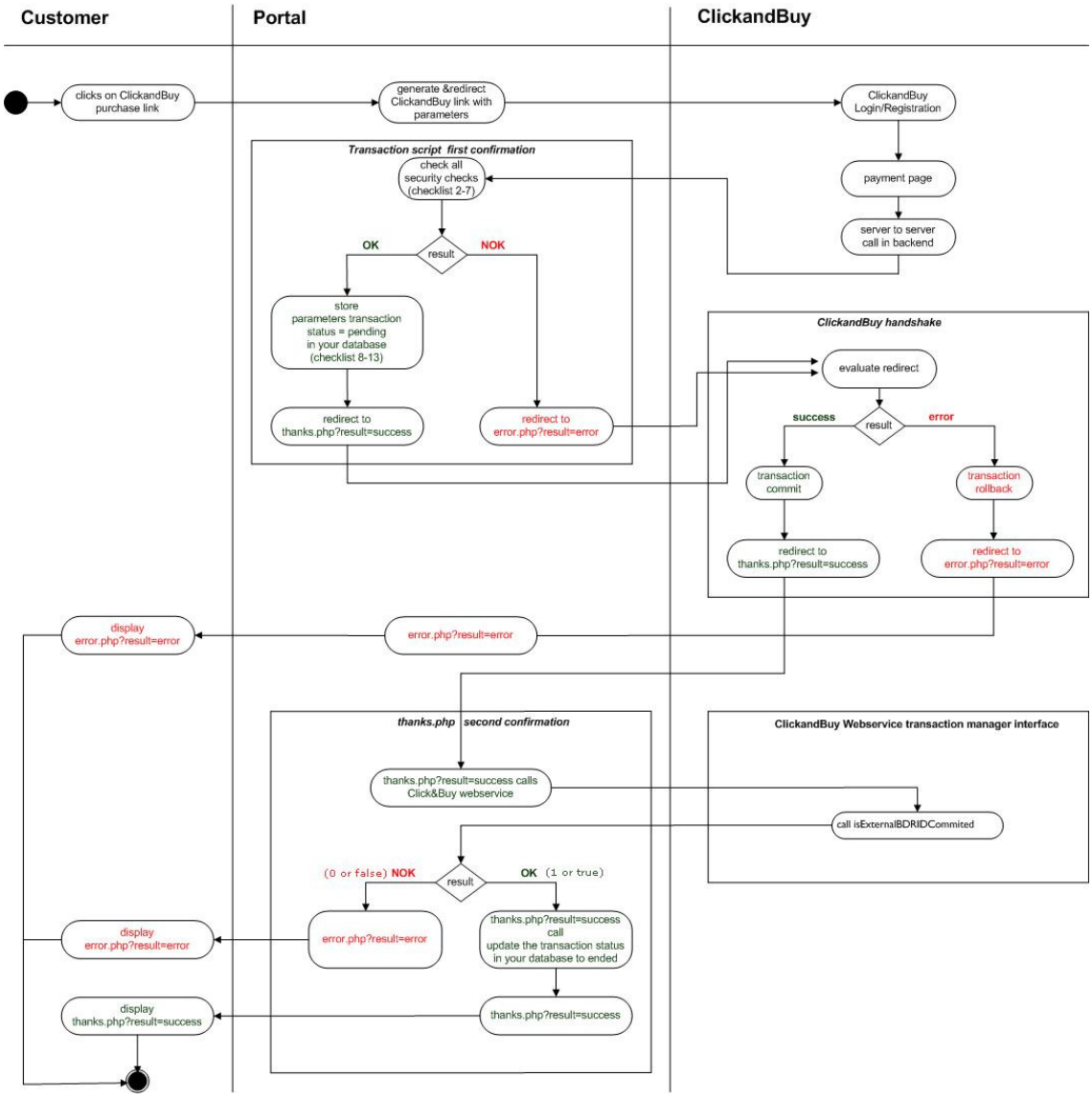

Figura 7: Diagrama de actividad del sistema de transacciones de clickandbuy

También podemos observar el diagrama de secuencias:

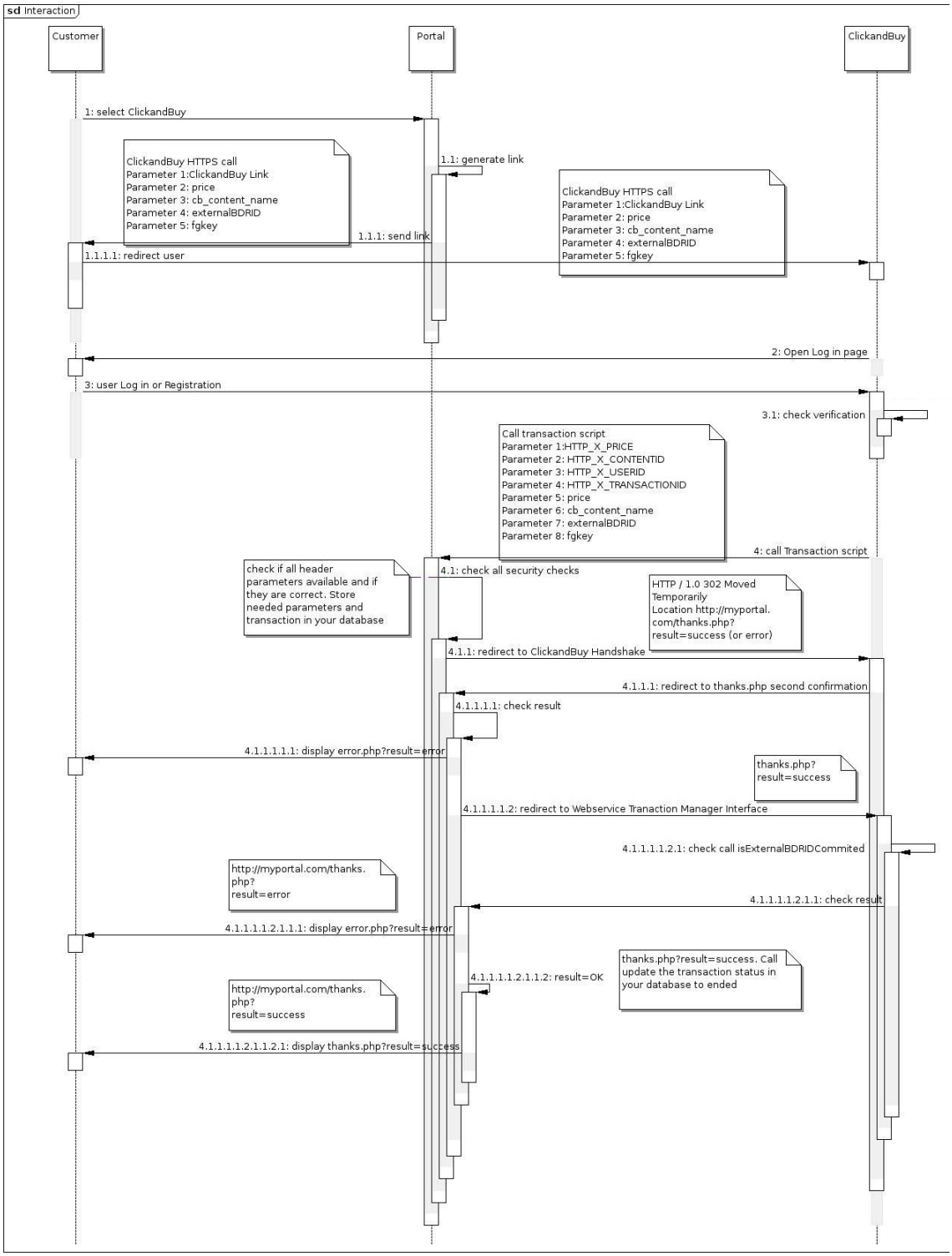

Figura 8. Diagrama de secuencias del sistema de transacciones de clickandbuy

Para conectar con clickandbuy, tenemos que hacer una redirección desde nuestro portal a una dirección url de clickandbuy pasando una serie de parámetros. La url es como la siguiente:

http://premium-encrypted.eu.clickandbuy.com//transaction.php?price=100&cb_currency=EUR&cb_content_name_utf=Transfer&my_id=123.....

Tabla 17. URL de llamada a clickandbuy

Con esto, nos redirigimos al portal de clickandbuy, donde tendremos que identificarnos para poder finalizar la transacción.

Algunos de los parámetros pasados por url son :

- price: Cantidad que queremos ingresar, la cual la tenemos que multiplicar por 100, es decir, si queremos ingresar 20.50€, la cantidad que debemos pasar es 2050.
- cb_currency: Moneda en la cual estamos realizando la transacción.
- etc…

CAPÍTULO III: DESARROLLO DEL SISTEMA

3.1 DESCRIPCIÓN DEL SISTEMA

Llegados a este punto, donde ya conocemos todo lo necesario para desarrollar esta aplicación, se puede comenzar a realizar dicho desarrollo.

El sistema estará formado por los siguientes módulos:

- Un servidor, que proporcionará acceso a la aplicación web.
- Una base de datos, donde guardaremos toda la información de los usuarios y las acciones que han llevado a cabo en el portal.
- Un cliente móvil, que realizará peticiones al servidor anterior para interactuar con la aplicación.
- Un cliente web, desde el que se realiza la administración del portal.
- Un sistema de identificación de probabilidades de apuestas, que estará formado por una aplicación externa, llamada WEKA, que contiene un repertorio de clasificadores que nos ayudará a identificar los mercados en los que hay mayor probabilidad de que apueste un determinado usuario.
- Una API, perteneciente a Betfair, que nos proporcionará los distintos mercados en los que realizar apuestas.

3.1.1 DIAGRAMA DE FLUJO DE DATOS

Los diagramas de flujo de datos (DFD) son un tipo de herramienta de modelado, permiten modelar todo tipo de sistemas, concentrándose en las funciones que realiza, y los datos de entrada y salida de esas funciones. También se les denomina diagramas de burbujas o grafos de flujo de datos. Describen un sistema como una red de procesos (subsistemas) conectados entre sí por conductos (flujos de datos), y/o almacenes de datos, y/o a fuentes y sumideros de datos (entidades externas). Proporcionan un mecanismo para el modelado funcional así como para el modelado del flujo de información. [alegsa][Prácticas ADS]

Los DFD son de gran utilidad en sistemas operacionales en los cuales las funciones del sistema son de gran importancia y son más complejas que los datos que estas manejan. Están orientados fundamentalmente a sistemas de información (aplicaciones de gestión), también se utilizan para modelar organizaciones enteras (herramienta para la planificación estratégica de la información y para el análisis del área de negocios).

Los componentes de un DFD son:

- PROCESOS (burbujas): representan la parte del sistema que transforma ciertas entradas en ciertas salidas.

- FLUJOS: representan los datos en movimiento. Pueden ser flujos de entrada o flujos de salida. Los flujos conectan procesos entre sí y también almacenes con procesos.

- ALMACENES: representan datos almacenados. Pueden ser una base de datos, un archivo físico, etc.

- TERMINADORES: representan entidades externas que se comunican con el sistema. Esas entidades pueden ser personas, organizaciones u otros sistemas, pero no pertenecen al sistema que se está modelando.

Existen procesos y flujos especiales llamados procesos de control y flujos de control. Se emplean para modelar sistemas en tiempo real.

Los flujos de control son señales o interrupciones, en tanto los procesos de control son burbujas que coordinan y sincronizan otros procesos. Los procesos de control sólo se conectan con flujos de control.

Los flujos de control de salida "despiertan" otras burbujas, en tanto los flujos de control de entrada, especifican que una tarea terminó o se presentó un evento extraordinario.

Un sistema puede representarse empleando varios diagramas de flujos de datos, cada flujo de datos puede representar una parte "más pequeña" del sistema.

Los DFD permiten una partición por niveles del sistema. El nivel más general se representa con un DFD global llamado diagrama de contexto.

El diagrama de contexto DFD representa a todo el sistema con una simple burbuja o proceso, las entradas y salidas de todo el sistema, y las interacciones con los terminadores.

Los DFD suelen servir para comprender fácilmente el funcionamiento de un sistema. De todas maneras, no es la única herramienta para diagramar sistemas, es más, se debe complementar con otras herramientas para agregar comprensión y exactitud al DFD.

A continuación, se detallan los diagramas de flujo de datos.

1. Diagrama de contexto

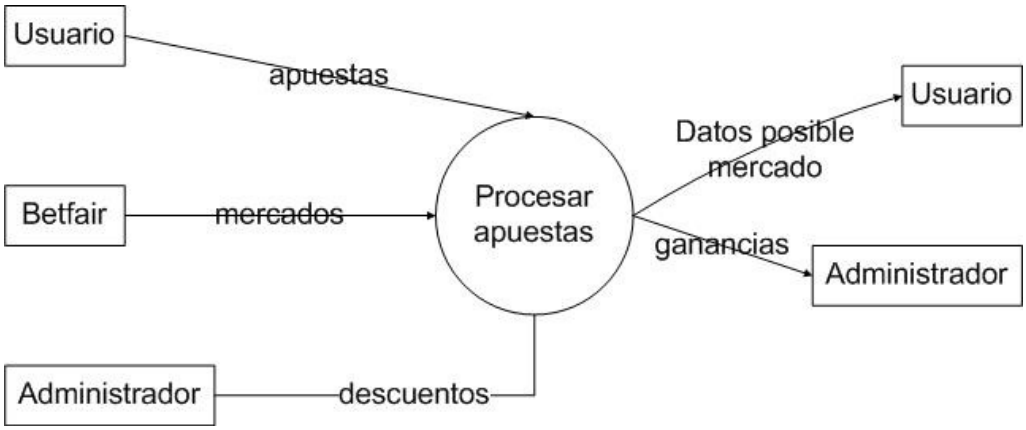

Figura 9. Diagrama de contexto

2. Diagrama de conjunto

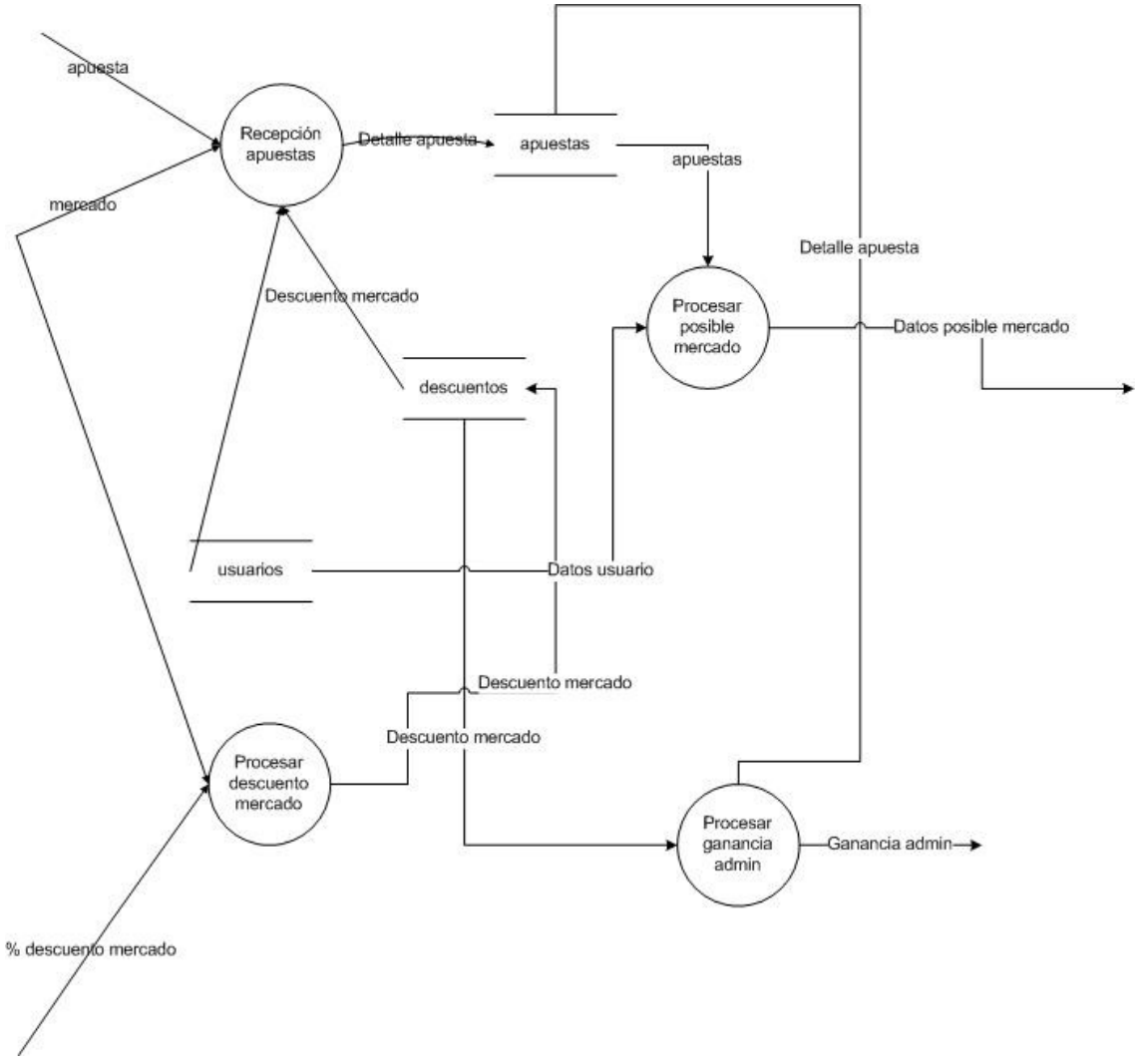

Figura 10. Diagrama de conjunto

3. Recepción de apuestas

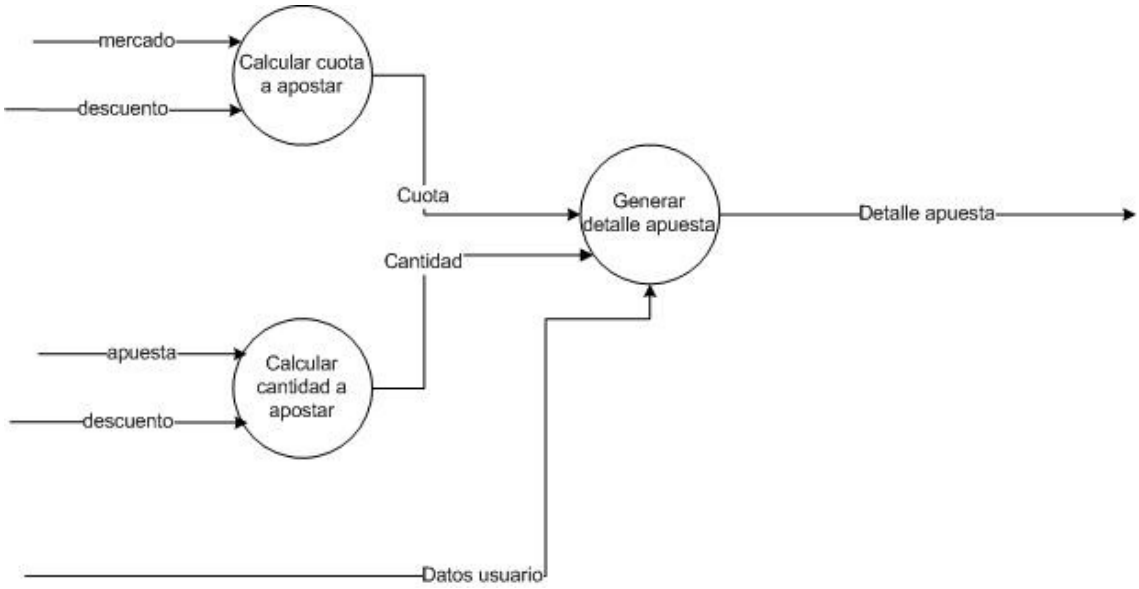

Figura 11. Recepción de apuestas

4. Procesar descuento mercado

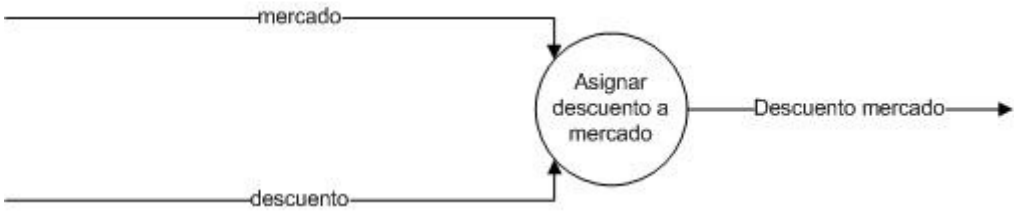

Figura 12. Procesar descuento mercado

5. Procesar posible mercado

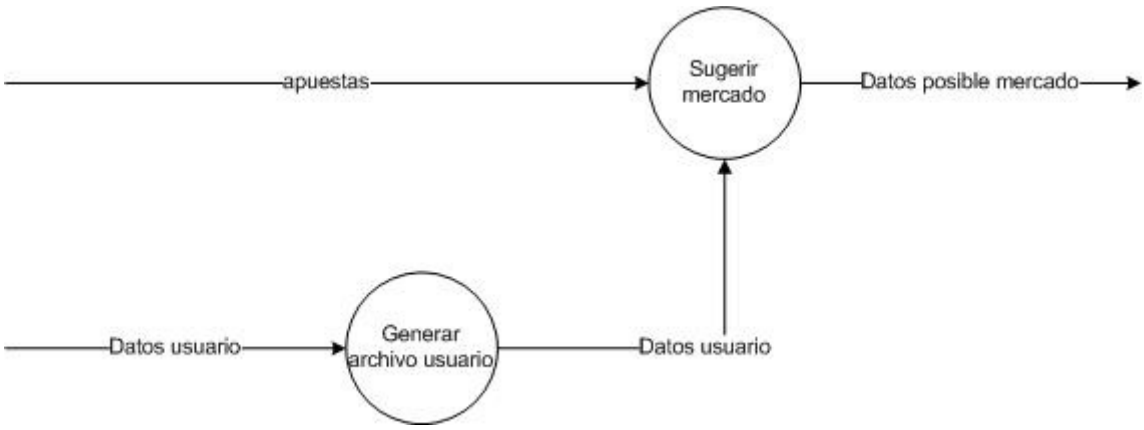

Figura 13. Procesar posible mercado

6. Procesar ganancia administrador

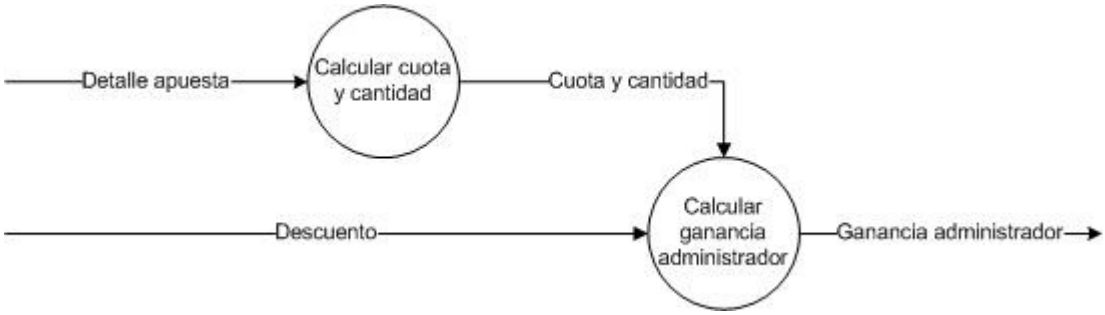

Figura 14. Procesar ganancia administrador

3.1.2 DIAGRAMA DE CLASES UML

UML es un lenguaje gráfico para describir software. Se analiza y diseña software a través de planos. Sirve como documentación preliminar (descripción a los clientes, firma del contrato), como documentación de implementación (para los programadores), y como documentación final (describe la arquitectura y funcionamiento del sistema. UML permite construir los siguientes diagramas:

- Diagramas de casos de uso (DCU)
- Diagramas de clases (DCL)
- Diagramas de interacción: secuencia (DSEC) y colaboración (DECOL)
- Diagramas de estados (DE)
- Diagramas de actividad (DA)
- Diagramas de componentes (DCOM)
- Diagramas de despliegue (DD)

Los diagramas de clase son el pilar básico del modelado con UML, siendo utilizados tanto para mostrar lo que el sistema puede hacer (análisis), como para mostrar cómo puede ser construido (diseño).

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones (incluyendo herencia, agregación, asociación, etc.). El diagrama de clases de más alto nivel (*main class diagram*) será un dibujo de los paquetes que componen el sistema. A su vez, cada paquete tendrá un *main class diagram* que muestra las clases del paquete. Para definir las clases, se incluye una descripción de lo que hacen, sus métodos y los atributos que las componen. Las relaciones entre las clases se indican con una descripción de su funcionamiento, la cardinalidad de cada una de ellas (cuántos objetos intervienen en la relación) y su opcionalidad (es opcional que un objeto intervenga en una relación).

En la siguiente figura se muestra el diagrama de clases de la aplicación.

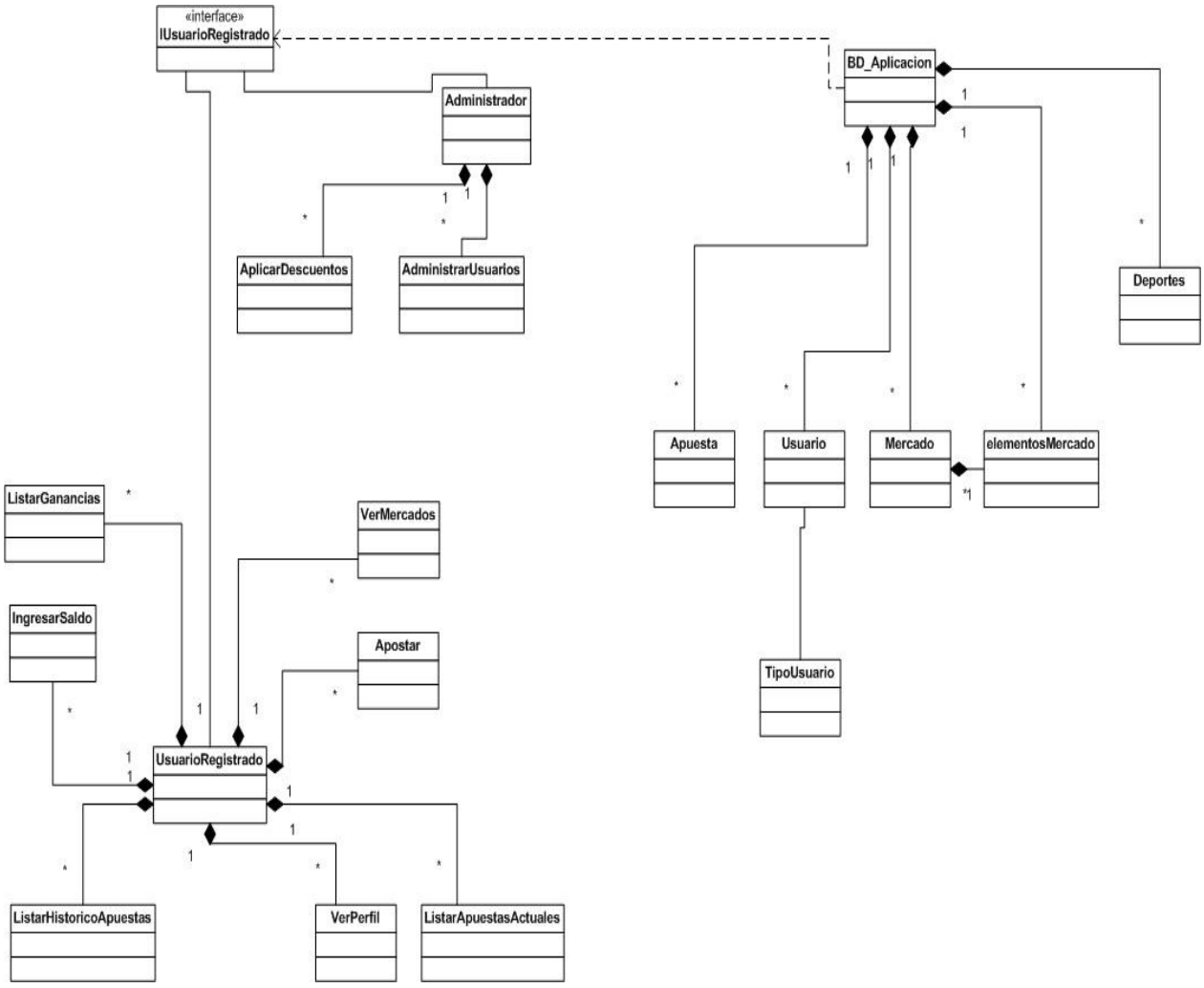

Figura 15. Diagrama de clases de la aplicación

3.2 CLIENTE

La parte del cliente consta de 2 módulos:

- **Cliente móvil**: Portal de apuestas para móviles, realizado en C# y Javascript, a través del cual el usuario se conectará a su cuenta y podrá realizar apuestas en los distintos mercados proporcionados por Betfair.

- **Cliente web**: Portal web realizado en C# y Javascript, a través del cual el administrador del sistema podrá gestionar los distintos usuarios registrados, además de asignar descuentos en las cuotas de los mercados seleccionados por él mismo. También existe un portal web dirigido al usuario para darse de alta en la aplicación y poder asimismo realizar apuestas a través de este portal.

A continuación se explica detenidamente cada uno de estos módulos del sistema.

3.2.1 CLIENTE MÓVIL

Para realizar la implementación de este cliente ha sido necesario utilizar tecnología Ajax, que permite la interacción de código C# y Javascript. Al ser un portal web, cualquier teléfono móvil o PDA con acceso a internet puede acceder a este portal.

A continuación, se realiza una explicación detallada del proceso de desarrollo y las distinas opciones que ofrece este portal.

Comenzaremos por la pantalla inicial, en la lo primero que vemos es la página de login a la aplicación:

Figura 16. Entrada aplicación

Introducimos nuestro nombre de usuario y contraseña y accedemos a la aplicación.

La clase principal de la aplicación es la siguiente:

```
public partial class index : System.Web.UI.Page
{
        protected void Page_Load(object sender, System.EventArgs e)
        {}

        private void crearArchivoUsuario(string dianacimiento, string mesnacimiento, string
añonacimiento, string profesion, string sexo, string localidad)
        {
        }

        private void Login()
        {
        }
}
```

Tabla 18. Clase index.cs

Index: Es la clase en la que se inicializan todos los elementos necesarios. Los pasos que se dan cuando un usuario quiere acceder a la aplicación son los siguientes:

- Creamos el formulario de identificación.
- Si el usuario ha introducido bien sus datos, cargamos datos del usuario.
- Creamos el archivo de test en formato arff con los datos del usuario identificado que utilizaremos para obtener el posible mercado al que puede apostar el usuario.
- Establecemos la conexión con la API de Betfair.
- Redirigimos al menú de mercados de Betfair para apostar.
- Si el usuario no ha insertado bien sus datos, mostramos un mensaje de error.

Login: Recoge los datos que el usuario ha introducido y si son correctos redirige al usuario al menú de mercados de Betfair.

CrearArchivoUsuario: Crea un archivo con un formato reconocible por weka con el fin de obtener un posible mercado en el que el usuario tendría interés en apostar.

Pasamos a ver más detenidamente la funcionalidad de estas opciones:

Login

En esta opción, establecemos una conexión con el servidor, pasándole el login y el password del usuario para validarlo. Se comprueban los datos a través de la base de datos y devuelve

una respuesta con éxito o fracaso. Si se ha validado correctamente, pasamos a la siguiente sección.

Como se ha mencionado, la pantalla del login es un formulario, el cual, una vez comprobadas la validez de la entrada del login y del password, llama a la clase 'dbSQLServer', que contiene los métodos necesarios para identificarse en el sistema. Esta clase es como sigue:

```
public class dbSQLServer
{
        public DataSet SelectDB(string sqlString)
        {
        }
        public DataSet SelectPaginadoDB(string sqlString, int registroInicio, int tamanoPagina)
        {
        }
        public int SelectEscalarDB(string sqlString)
        {
        }
        public Boolean InsertDB(string sqlString)
        {
        }
        public Boolean UpdateDB(string sqlString)
        {
        }
        public Boolean DeleteDB(string sqlString)
        {
        }
        …
}
```

Tabla 19. Clase dbSQLServer.cs

Los métodos de esta clase son:

- *SelectDB*: Devuelve un Dataset con los datos obtenidos por la consulta SQL. Si la consulta no proporciona ningún resultado, devuelve null.

- *SelectPaginadoDB*: Devuelve un DataSet con los datos obtenidos por la consulta SQL, pero desde un determinado registro y un número concreto de éstos.

- *SelectEscalarDB*: Devuelve un entero que se ha requerido desde la consulta SQL. Por ejemplo: "SELECT COUNT(*) FROM usuarios".

- *InsertDB*: Inserta un registro en la base de datos y devuelve true si la operación ha concluido con éxito y false en caso contrario.

- *UpdateDB*: Actualiza un registro de la base de datos y devuelve true si la operación ha concluido con éxito y false en caso contrario.

- *DeleteDB*: Borra un registro de la base de datos y devuelve true si la operación ha concluido con éxito y false en caso contrario.

Una vez hemos accedido a la aplicación, se nos ofrecen 3 opciones:

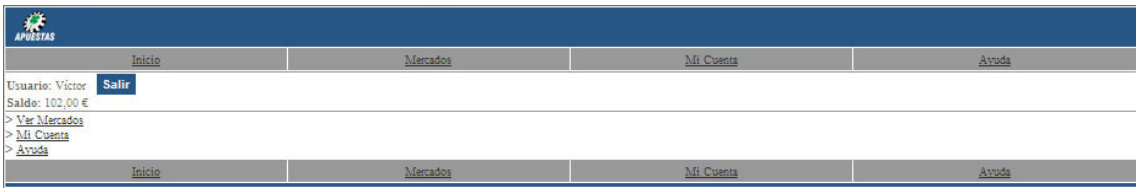

Figura 17. Página principal

- *Ver mercados*: Nos muestra los distintos mercados que proporciona Betfair para poder realizar apuestas.

- *Mi cuenta*: Nos muestra una serie de opciones a las que podemos acceder relacionadas con los datos personales del usuario y la interacción que ha establecido con la aplicación web (apuestas actuales, histórico de apuestas, etc.)

- *Ayuda*: Sección con información acerca de la aplicación.

La aplicación también nos muestra de entrada nuestro nombre de usuario y el saldo actual que tenemos en nuestra cuenta para realizar apuestas.

La clase que implementa esta sección es **'intro'.**

```
public partial class intro : System.Web.UI.Page
{
        public void sugerirMercado()
        {
        }
        public void leerMercadosBD()
        {
        }
}
```

Tabla 20. Clase intro.aspx.cs

Los métodos de esta clase son:

- *sugerirMercado*: lee el archivo de resultados que ha generado weka y obtiene el mercado al que es más probable que apueste el usuario en relación con sus características (edad, población, ocupación, etc.)

- *leerMercadosBD*: obtiene todos los mercados existentes en la tabla de mercados y los indexa para su posterior tratamiento.

VER MERCADOS

Aquí se muestra un listado de los distintos mercados que existen en Betfair en la actualidad. Este listado se ha obtenido gracias a los servicios web proporcionados por Betfair a través de su API. Podemos seleccionar cualquier mercado para comprobar las distintas opciones de apuestas que tenemos:

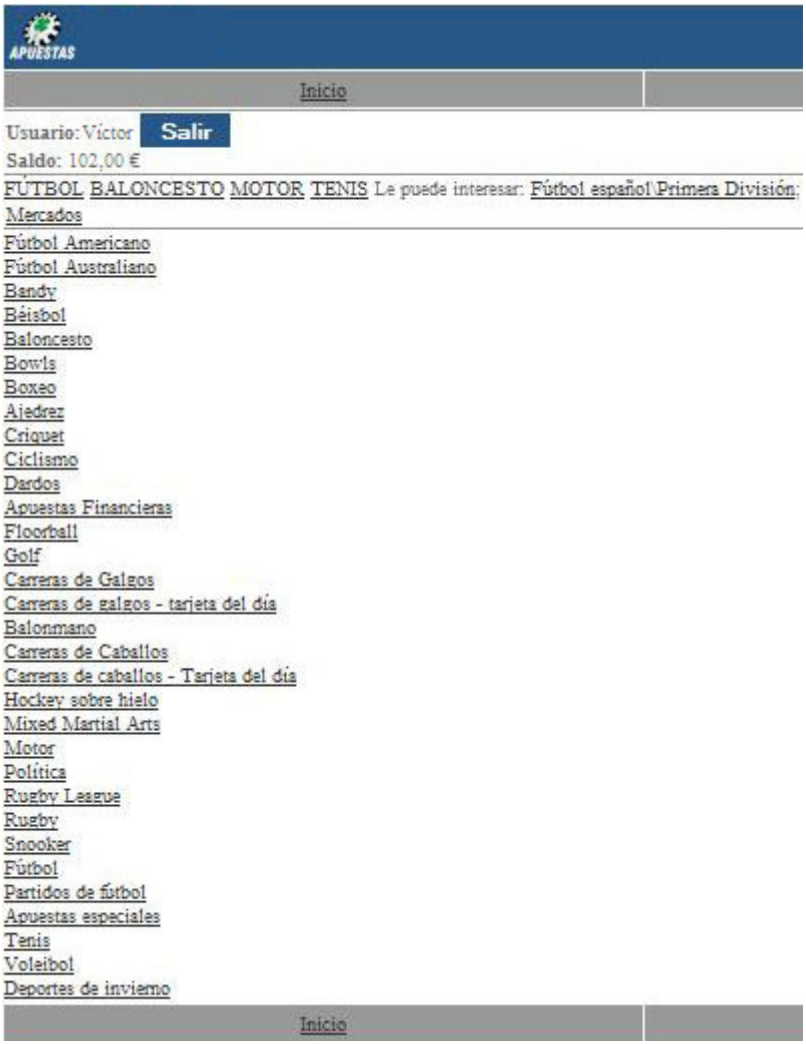

Figura 18. Mercados

Podemos observar que hay 4 mercados principales (Fútbol, Baloncesto, Motor y Tenis), el menú de mercados de betfair por el que podemos navegar, y el mercado sugerido por la aplicación que podría interesarnos en relación con las características del usuario identificado. En este caso, este mercado sería '**Fútbol español-Primera División**'. Más adelante se explicará cómo se obtienen los posibles mercados.

Si navegamos por los mercados, finalmente llegamos a mostrar las cuotas para el mercado seleccionado. La clase que implementa la navegación por los mercados es '**seccion_apuestas_index**'.

```
public partial class index : System.Web.UI.Page
{
        public void desplegarMenuEspecifico ()
        {
        }
        public void desplegarMenuMercados ()
        {
        }
        public int buscarIndice(ArrayList ids, int id)
        {
        }
}
```

Tabla 21. Clase index.aspx.cs

Los métodos que implementa esta clase son:

- *desplegarMenuEspecifico*: Se ejecuta cuando hacemos click en un deporte directo, y añade a un array el id del deporte y el nombre.
- *desplegarMenuMercados*: Despliega el menú de mercados y nos permite navegar a través de estos para acceder a las cuotas finales donde poder realizar las apuestas deseadas.
- *buscarIndice*: Función utilizada para buscar un identificador de un deporte en un arraya de identificadores. Si no encuentra este identificador, devuelve -1.

Como habíamos comentado, una vez que hemos navegado por los mercados, llegamos a la pantalla de cuotas:

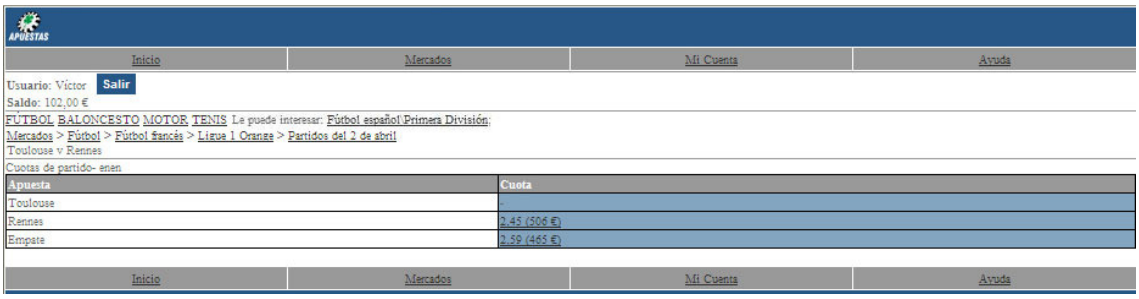

Figura 19. Cuotas del mercado

Como se puede observar, para llegar hasta la pantalla de las cuotas, hemos seguido la ruta '**Fútbol->Fútbol francés->Ligue 1 Orange->Partidos del 2 de abril->Toulouse v**

Rennes'. Una vez que hemos llegado aquí, únicamente nos queda seleccionar el resultado que creemos que se va a dar y asignarle una cantidad de dinero para realizar la apuestas.

La clase que implementa las cuotas es '**seccion_apuestas_mercados**'. La carga de apuestas se hace por medio de Ajax. La llamada a la función '**cargarApuestas()**' se hace desde la etiqueta '**<body>**'. Esta función javascript llama a una función en C# llamada '**desplegarApuestas()**'. Esto lo podemos ver explicado en losrecuadros siguientes:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es">
	<head>
		<script type="text/javascript" src="../js/mercado.js"></script>
	</head>
	<body onload="cargarApuestas();">
	<form id="formulario" runat="server">
		<!--cabecera-->
		<cabecera:cabecera id="cabecera" runat="server"></cabecera:cabecera>
		<!--cuerpo-->
		<div id="cuerpo">
			<navegador:navegador id="navegador"
runat="server"></navegador:navegador>
			<div id="mercado"></div>
		</div>
		<!--pie-->
		<pie:pie id="pie" runat="server"></pie:pie>
	</form>
	</body>
</html>
```

Tabla 22. mercados.aspx

```
public partial class mercado: System.Web.UI.Page
{
	[Ajax.AjaxMethod(HttpSessionStateRequirement.ReadWrite)]
	public string desplegarApuestas(int id)
	{
	}
}
```

Tabla 23. Clase mercados.aspx.cs

```
function cargarApuestas()
{
}
function actualizarIntervaloApuestas()
{
}
 function desplegarApuestas_CallBack ()
{
}
function getURLParam(strParamName)
{
}
```

Tabla 24. Mercados.js

- *desplegarApuestas*: Método Ajax que genera el menú de cuotas y lo devuelve en una cadena de texto, formateado para mostrarlo en la pantalla del dispositivo móvil.
- *cargarApuestas*: Función inicial para cargar las cuotas de un mercado dado.
- *actualizarIntervaloApuestas*: Realiza la llamada para desplegar las apuestas.
- *desplegarApuestas_CallBack*: Función ajax que muestra las cuotas de un mercado.
- *getURLParam*: Función que obtiene el parametro get indicado.

MI CUENTA

Si seleccionamos esta opción, se nos van a ofrecer una serie de alternativas relacionadas con los datos personales del usuario y con la interacción que éste ha mantenido con el portal hasta el momento. Seguidamente podemos ver la pantalla '**Mi cuenta**'.

Figura 21. Pantalla 'Mi Cuenta'

Datos de usuario

Esta opción nos mostrará datos personales del usuario y estadísticas generales de lo que ha hecho en el portal.

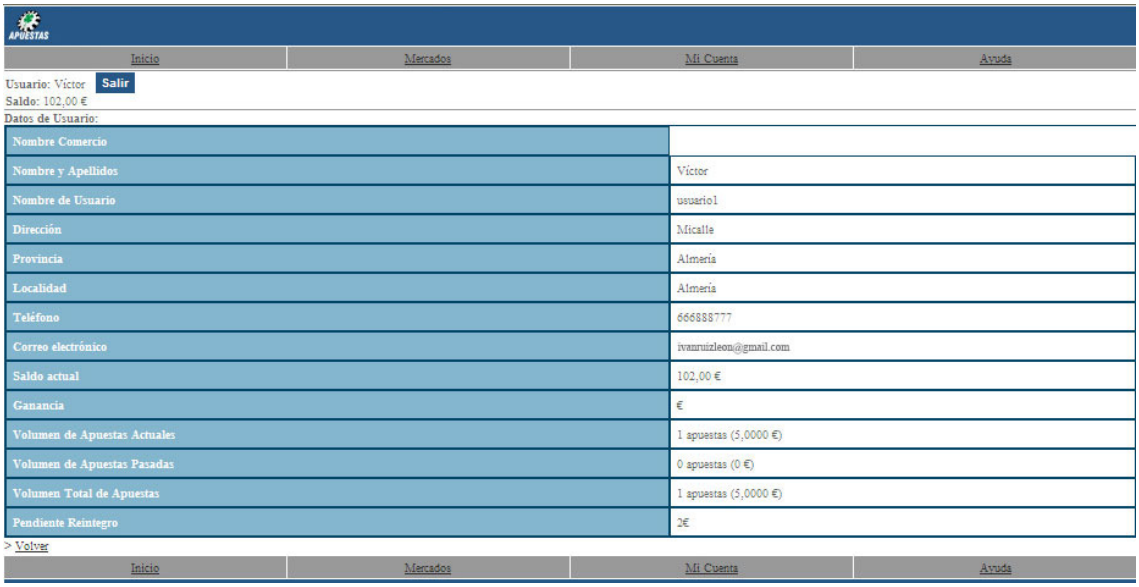

Figura 20. Pantalla 'Datos de usuario'

La clase que implementa esta sección es '**seccion_micuenta_perfil**'.

```
public partial class perfil : System.Web.UI.Page
{
        protected void Page_Load(object sender, System.EventArgs e)
        {
        }
        public void cargaInfoPerfil()
        {
        }
        public void calcularEntregas()
        {
        }
}
```

Tabla 25. Clase perfil.cs

Los métodos que implementa esta clase son:

- *Page_Load*: Llama a los otros dos métodos de la clase al cargar la página.
- *cargaInfoPerfil*: Obtiene los datos del usuario de la base de datos y los imprime en una etiqueta de servidor dándoles formato para que la página tenga un estilo.
- *calcularEntregas*: Comprueba las cantidades pendientes de pago que tiene el usuario con el administrador y las escribe en una etiqueta de servidor.

Apuestas actuales

Nos muestra las apuestas que el usuario tiene actualmente en vigor, es decir, aquellas que todavía no se han jugado.

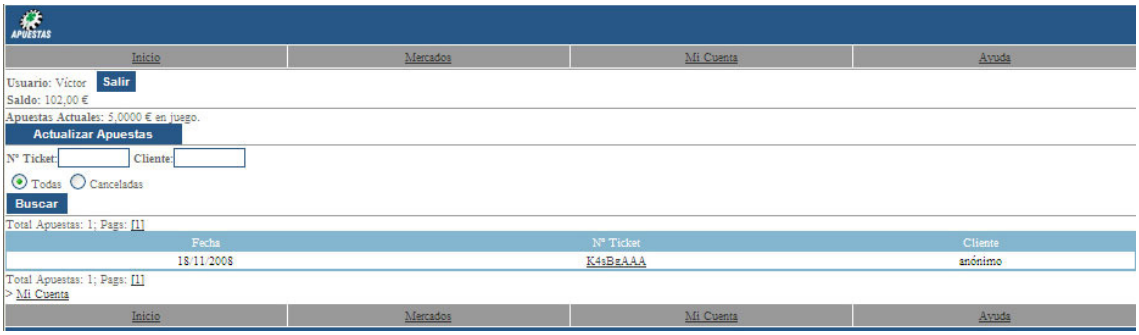

Figura 22.Pantalla 'Apuestas actuales'

Tenemos un pequeño buscador de apuestas, en el que podemos buscar apuestas por nº de ticket o nombre de cliente, y podemos restringir la búsqueda por únicamente apuestas canceladas, o en cambio buscarlas todas.

También tenemos un botón para actualizar el listado de apuestas, por si alguna ha pasado a finalizada y debe borrarse de esta lista. En el listado nos aparece la fecha de la apuesta, el número de ticket(que es un enlace que nos muestra el detalle de la apuesta), y el nombre del cliente para el que realizamos la apuesta. A continuación podemos ver el detalle de la apuesta.

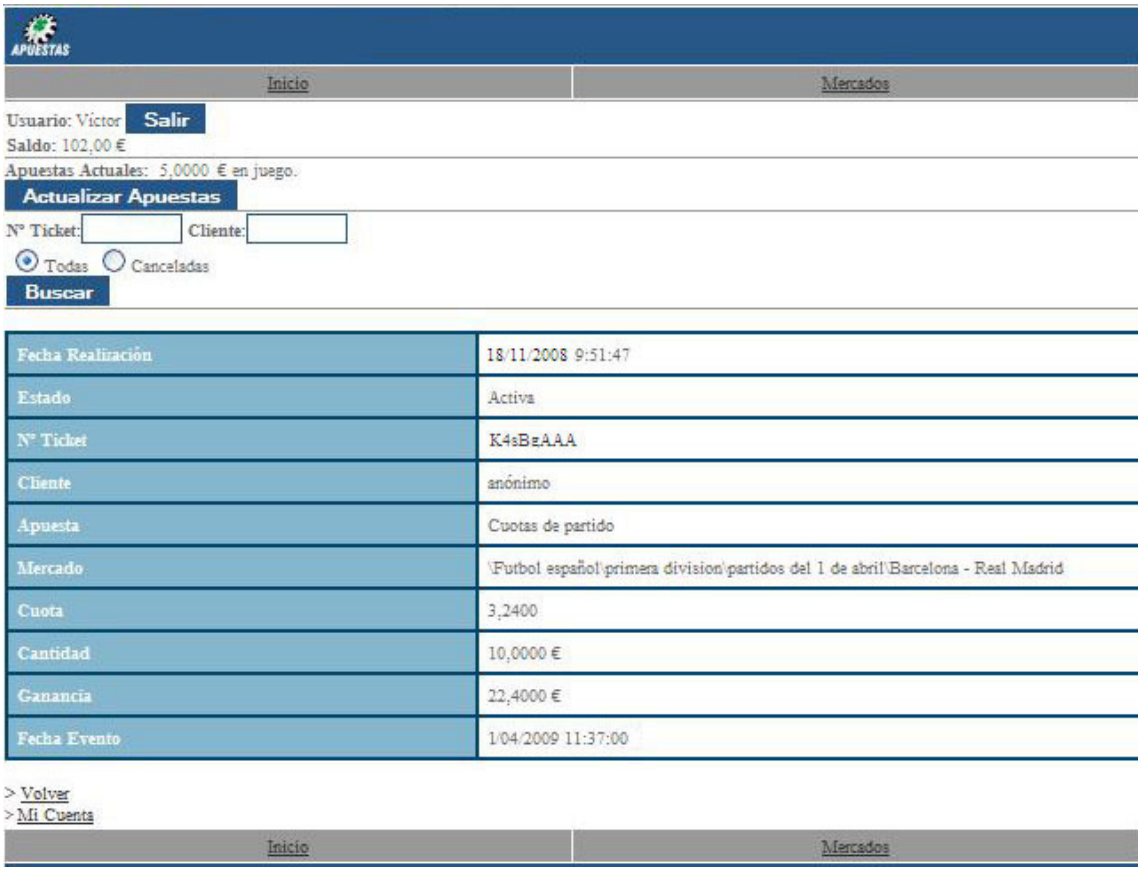

Figura 23. Pantalla de detalle de la apuesta

La clase que implementa las apuestas actuales es '**seccion_micuenta_actuales**'.

```
public partial class actuales: System.Web.UI.Page
{
        protected void Page_Load(object sender, System.EventArgs e)
        {
        }
        public void cargarBuscador ()
        {
        }
        public void cargarApuestas ()
        {
        }
        public string generaPaginacion(string query, int desde, int pagactual)
        {
        }
        [Ajax.AjaxMethod(HttpSessionStateRequirement.ReadWrite)]
        public string detalleApuesta(string n_ticket)
        {
        }
        [Ajax.AjaxMethod(HttpSessionStateRequirement.ReadWrite)]
        public string actualizarApuestasCuenta()
        {
        }
}
```

Tabla 26. Clase actuales.cs

Los métodos que implementa esta clase son:

- *Page_Load*: Cuando carga la página, llama al método cargarApuestas.
- *cargarBuscador*: Es llamado por el método cargarApuestas para obtener las apuestas al cargar la página.
- *cargarApuestas*: Obtiene las apuestas actuales y las muestra con formato en una etiqueta de servidor.
- *generaPaginacion*: Es llamada por el método cargarApuestas y lo que hace es paginar los resultados obtenidos.
- *detalleApuesta*: Método Ajax que despliega el detalle de una apuesta.
- *actualizarApuestasCuenta*: Método para actualizar las apuestas actuales (cambia las apuestas finalizadas de actuales a históricas o canceladas si procede).

Los métodos Ajax son llamados desde las funciones que se encuentran en el archivo javascript '**cuenta.js**'. Aquí podemos ver los métodos que forman este archivo:

```
function buscarApuestasActuales()
{}
function buscarApuestasHistoricas()
{}
function actualizar()
{}
function actualizarApuestasCuenta_CallBack(response)
{}
function detalleApuesta_CallBack(response)
{}
```

Tabla 27. Cuenta.js

- *buscarApuestasActuales*: Método Ajax que realiza la búsqueda de las apuestas actuales del usuario.
- *buscarApuestasHistoricas*: Método Ajax que realiza la búsqueda de las apuestas históricas del usuario.
- *Actualizar*: Realiza una llamada al metodo actualizarApuestasCuenta_CallBack.
- *actualizarApuestaCuenta_CallBack*: Metodo Ajax que realiza la actualizacion de las apuestas actuales.
- *detalleApuesta_CallBack*: Función Ajax que muestra el detalle de una apuesta pasada por parámetro.

Histórico Apuestas

Nos muestra las apuestas que el usuario hizo y cuyo resultado ya se ha obtenido, sea ganada o perdida.

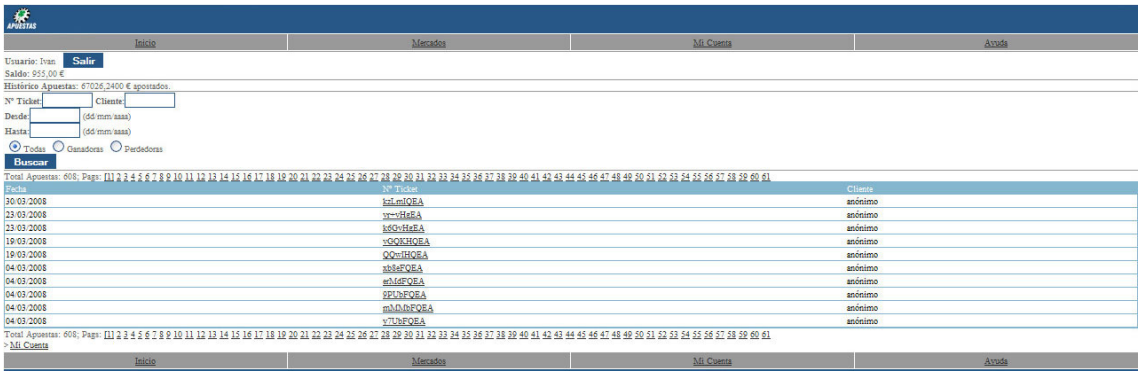

Figura 24. Pantalla de histórico de apuestas

Tenemos un pequeño buscador de apuestas, en el que podemos buscar apuestas por nº de ticket o nombre de cliente, y podemos restringir la búsqueda por únicamente apuestas ganadoras, perdedoras, o en cambio buscarlas todas. También tenemos la opción de buscar apuestas entre dos fechas dadas.

En el listado nos aparece la fecha de la apuesta, el número de ticket(que es un enlace que nos muestra el detalle de la apuesta), y el nombre del cliente para el que realizamos la apuesta, que nos muestra la misma pantalla de detalle que las apuestas actuales.

La clase que implementa las apuestas históricas es '**seccion_micuenta_historicas**'.

```
public partial class historicas : System.Web.UI.Page
{
        protected void Page_Load(object sender, System.EventArgs e)
        {
        }
        public string cargarBuscador()
        {
        }
        public void cargarApuestas(int numpagina)
        {
        }
        public string generaPaginacion(string query, int desde, int pagactual)
        {
        }
        [Ajax.AjaxMethod(HttpSessionStateRequirement.ReadWrite)]
        public string detalleApuesta(string n_ticket)
        {
        }
}
```

Tabla 28. Clase históricas.cs

Los métodos que componen esta clase son:

- *Page_Load*: Llama al método cargarApuestas cuando se carga la página.
- *cargarBuscador*: Es llamado por el método cargarApuestas y proporciona aquellas apuestas que ha realizado el usuario y que ya hayan sido gestionadas.
- *cargarApuestas*: Genera el listado de las apuestas históricas.
- *generaPaginacion*: Método para paginar los resultados obtenidos.
- *detalleApuesta*: Método Ajax que despliega el detalle de una apuesta.

Ingreso saldo

Nos da la opción de ingresar saldo en nuestra cuenta del portal a través de la pasarela de pago de clickandbuy.

Figura 25. Pantalla de ingreso de saldo

Tenemos que introducir una cantidad y pulsar el botón reintegro. Mediante los servicios web de clickandbuy retiraremos saldo de nuestra cuenta en dicha pasarela de pago, y lo ingresaremos en nuestra cuenta del portal de apuesta.

La clase que implementa el ingreso de saldo es '**clickandbuy**'.

```
public partial class clickandbuy: System.Web.UI.Page
{
        protected void BTIngresar_Click()
        {
        }
        public string generarIdTransaccion_Click ()
        {
        }
}
```

Tabla 29. Clase clickandbuy.cs

Los métodos que componen esta clase son:

- *BTIngresar_Click*: Es el evento que se produce cuando se hace click en el botón de reintegro. Generamos un mensaje SOAP y lo enviamos a la pasarela de pago de clickandbuy con los datos del usuario y la cantidad que queremos ingresar en nuestra cuenta del portal. Si todo ha ido como es debido, actualizo el saldo en la base de datos y guardo la transacción en una tabla.

- *generarIdTransaccion*: Este método genera un id de transacción único para enviarlo en el mensaje SOAP que se manda a clickandbuy.

Reintegro saldo

Nos da la opción de ingresar dinero en nuestra cuenta de clickandbuy y sacarlo de nuestra cuenta del portal.

Figura 26. Pantalla de reintegro de saldo

Tenemos que introducir una cantidad y pulsar el botón ingresar. De esta manera, retiraremos saldo de nuestra cuenta en el portal y a través de los servicios web de clickandbuy ingresaremos saldo en nuestra cuenta en dicha pasarela de pago.

La clase que implementa el ingreso de saldo es '**clickandbuyReintegros**'.

```
public partial class clickandbuyReintegros: System.Web.UI.Page
{
        protected void BTIngresar_Click()
        {
        }
        public string generarIdTransaccion_Click ()
        {
        }
}
```

Tabla 30. Clase clickandbuyReintegros.cs

Los métodos que componen esta clase son:

- *BTIngresar_Click*: Es el evento que se produce cuando se hace click en el botón de reintegro. Generamos un mensaje SOAP y lo enviamos a la pasarela de pago de clickandbuy con los datos del usuario y la cantidad que queremos ingresar en nuestra cuenta del portal. Si todo ha ido como es debido, actualizo el saldo en la base de datos y guardo la transacción en una tabla.

- *generarIdTransaccion*: Este método genera un id de transacción único para enviarlo en el mensaje SOAP que se manda a clickandbuy.

Ganancias

Esta opción solo está disponible para el tipo de usuario '**gerente**'. Muestra las ganancias que obtiene un gerente de las apuestas que ha realizado en nombre de un usuario en el portal.

Movimientos

Muestra el listado de transacciones que se han llevado a cabo en la cuenta de un usuario, ya sean depósitos, reintegros o ganancias por comisión.

Vamos a describir ahora el funcionamiento del clasificador de usuarios para obtener un posible mercado en el que el usuario estaría interesado en apostar.

La implementación de esta opción se realiza en aplicación de consola, que se lanzará como una tarea programada una vez por semana.

La clase que implementa el clasificador es '**Program**'.

```
class Program
{
        generarArchivoTrain
        {}
        ejecutaWeka
        {}
}
```

Tabla 31. Clase Program.cs

Los métodos que componen esta clase son:

- *generarArchivoTrain*: crea el archivo de entrenamiento, con formato arff, con las últimas 500 apuestas que se han realizado en el portal.
- *ejecutaWeka*: Aplica el clasificador naive Bayes al archivo de entrenamiento y genera un archivo de salida que nos va a proporcionar el posible mercado en el que podría estar interesado en apostar el usuario que ha entrado en la aplicación.

AYUDA

Simplemente nos propociona información acerca de la versión de la aplicación y los autores del desarrollo.

3.2.2 CLIENTE WEB

Esta parte de la aplicación ha sido desarrollada en C#, lo que da una idea de las posibilidades de la aplicación y de la potencia de la misma. En la parte web de la aplicación se nos ofrecen todas las opciones que nos proporciona la parte móvil, y alguna más, como por ejemplo el registro de usuario o la modificación de nuestro perfil.

Mediante la parte web también puede identificarse en el portal el administrador del sistema, que va a ser quien otorga descuentos a los mercados de Betfair, con el fin de obtener un beneficio de las apuestas realizadas por los usuarios.

A continuación mostramos ejemplos de pantallas de la aplicación web:

Figura 27. Pantalla principal cliente web

Aquí podemos observar la portada de la aplicación, donde podemos ver las diferentes secciones que componen el portal y una zona para identificarnos o registrarnos en el mismo.

Seguidamente, vamos a observar la pantalla principal de la administración y de la zona de descuentos de mercados.

Figura 28. Pantalla principal administración portal

Si nos identificamos como administrador, somos redireccionados a esta pantalla, donde podemos ver las diferentes secciones que podemos administrar (administradores, usuarios, apuestas, etc.). A nosotros nos interesa la sección '**mercados**'.

Figura 29. Pantalla de asignación de descuentos a los mercados en la administración

Podemos observar como tenemos un menú de mercados, a través del cual podemos navegar hasta llegar al mercado deseado, y un panel de asignar descuentos, donde nos muestra el mercado que tenemos actualmente seleccionado, un campo de texto para introducir un número, que será el porcentaje de descuento, y un botón para aplicar el descuento al mercado, que inserta el valor en una tabla de la base de datos.

La clase que implementa el descuento en los mercados es '**administracion**', y dentro de esta clase, hacemos una llamada Ajax al método '**guardarDatos**', que lo que hace es leer el mercado seleccionado y el descuento insertado y lo introduce en la base de datos, o actualiza el valor anterior perteneciente a ese mercado.

3.3 SERVIDOR

La parte servidora de la aplicación consta de los siguientes elementos:

- **Base de datos**: Almacenará los usuarios, las apuestas realizadas por estos, los mercados con descuentos y todos aquellos datos susceptibles de ser mostrados en el portal como información, tanto para el propio usuario como para los administradores de la aplicación. Esta desarrollada utilizando el sistema de gestión de base de datos SQLServer.

- **Sistema de pago:** Es un sistema externo a la aplicación, a la cual accedemos mediante mensajes SOAP, que nos permite hacer ingresos o extracciones de saldo en una cuenta virtual. Para ello, debemos estar previamente registrados en el portal perteneciente a esta pasarela de pago.

- **Sistema de apuestas**: Aplicación implementada en C#, que conecta con un sistema externo de apuestas a través de una API, que nos proporciona los mercados y las cuotas a las que podemos acceder para realizar nuestras apuestas.

- **Sistema clasificador**: Aplicación de consola, realizada en C#, que se ejecuta como una tarea programada, y que genera un archivo de entrenamiento para poder clasificar luego a los usuarios que se identifican en el portal.

A continuación, pasamos a explicar con más detalle las principales características de este sistema.

3.3.1 BASE DE DATOS

La base de datos esta realizada en SQLServer, como ya se ha comentado antes. La elección de este sistema de gestión de base de datos está tomada en base a su robustez y a su fácil interacción con Visual Studio.

La base de datos no es muy compleja, aunque si contiene varias tablas, de las que se van a describir aquellas más importantes.

Column Name	Data Type	Length	Allow Nulls
IdUsuario	char	50	
Password	char	40	
IdAdministrador	char	20	✓
LoginBetfair	char	20	
PasswordBetfair	char	20	
NombreComercio	char	255	✓
NombreUsuario	char	255	✓
Apellidos	char	50	✓
Sexo	int	4	✓
Profesion	char	10	✓
TipoUsuario	int	4	
Email	char	255	✓
SaldoActual	decimal	9	
Comision	decimal	9	
TipoImpresora	int	4	
Telefono	char	50	✓
TextoTicket	char	255	✓
RiesgoLimite	decimal	9	
Activo	bit	1	
Tratamiento	int	4	
DiaNacimiento	int	4	
MesNacimiento	int	4	
AñoNacimiento	int	4	
Ofertas	bit	1	
Productos	bit	1	
Movil	char	50	✓
Pais	char	10	✓
Direccion1	char	255	
Provincia	char	50	
Localidad	char	50	
CodPostal	char	10	
Pregunta1	int	4	
Respuesta1	char	50	✓
Pregunta2	int	4	
DiaRespuesta2	int	4	
MesRespuesta2	int	4	
AñoRespuesta2	int	4	
Zona	int	4	
Divisa	int	4	
LimitarDepositos	int	4	
FrecuenciaDepositos	int	4	
LimitarPerdidas	int	4	
FrecuenciaPerdidas	int	4	
url	char	50	✓
FormaPago	int	4	
Reintegros	int	4	
NumCuenta	char	20	✓
IBAN	char	24	✓
Confirmado	bit	1	✓
IdUsuarioCAB	char	20	✓
fechaRegistro	datetime	8	

Figura 30. Tabla ‘Usuarios’

	Column Name	Data Type	Length	Allow Nulls
🔑	IdApuesta	uniqueidentifie	16	
	IdUsuario	char	50	
	cliente	nvarchar	255	
	betId	bigint	8	
	IdSeleccion	int	4	
	nombreSeleccion	nvarchar	255	
	handicap	decimal	9	
	asianLineId	int	4	
	IdMercado	int	4	
	nombreMercado	nvarchar	255	
	rutaMercado	nvarchar	255	
	tipoMercado	nvarchar	50	
	cuotaBetfair	decimal	9	
	cuotaDopi	decimal	9	
	cantidadBetfair	decimal	9	
	cantidadDopi	decimal	9	
	ganancia	decimal	9	
	gananciaGerente	decimal	9	✓
	gananciaAdmin	decimal	9	✓
	gananciaDopi	decimal	9	✓
	esGanancia	bit	1	
	fechaInicio	datetime	8	
	fechaApuesta	datetime	8	
	finalizada	bit	1	
	gestionada	bit	1	
	n_ticket	nvarchar	50	✓
	estado	int	4	✓

Figura 31. Tabla ‘Apuestas’

	Column Name	Data Type	Length	Allow Nulls
🔑	idMercado	int	4	
🔑	IdAdministrador	char	50	
	nombreMercado	nvarchar	50	✓
	descuentoMercado	decimal	9	✓
	fecha	datetime	8	

Figura 32. Tabla ‘Mercados’

Estas tres son las tablas más importantes, ya que contienen a los usuarios de la aplicación (tabla usuarios), las apuestas que estos realizan (tabla apuestas), y los descuentos de los mercados (tabla mercados), aunque hay otra serie de tablas que son necesarias para el buen funcionamiento de la aplicación, y que pasamos a enumerar:

Tabla	**Descripción**
Deportes	Contiene los deportes proporcionados por Betfair
Divisas	Contiene las monedas válidas en la aplicación
Ganancia	Ganancias obtenidas
LimitesDepositos	Cantidad máxima que se puede ingresar en el portal
LimitePerdidas	Cantidad máxima que se puede perder
logAcceso	Log de accesos al portal
Municipios	Municipios de España
Pais	Diferentes países del mundo
Peticiones	Peticiones de saldo de los administradores
PeticionesGerentes	Peticiones de saldo de los gerentes
PeticionesReintegros	Peticiones de cobro de los usuarios
Preguntas	Preguntas de seguridad
Provincias	Provincias de España
Tarjetas	Tarjetas de crédito de los usuarios
TipoTransaccion	Define los tipos de transacción (deposito, reintegro, etc…)
TipoUsuario	Tipos de usuario de la aplicación
Transaccion	Transacciones que se han realizado en el portal
TransaccionCAB	Transacciones monetarias realizadas mediante la pasarela de pago de clickandbuy

Tabla 32. Tablas de la aplicación

3.3.2 SISTEMA DE PAGO

Es un sistema externo, y lo utilizamos para ingresar dinero en nuestra cuenta del portal o para guardar nuestras ganancias.

La pasarela de pago esta proporcionada por clickandbuy, y nosotros accedemos a ella mediante mensajes SOAP.

Un ejemplo de mensaje SOAP utilizado para realizar una retirada de saldo de nuestra cuenta de clickandbuy (debit), es el siguiente:

```
<xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsd='http://www.w3.org/2001/XMLSchema' xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
<soapenv:Body>
        <getEasyCollectSingle xmlns='TransactionManager.Payment'>
         <sRequest>
                <sellerID>1111111</sellerID>
                <tmPassword>universidad</tmPassword>
                <extJobID></extJobID>
                 <request>
                        <discriminator>DEBIT</discriminator>
                        <debReq>
                                <crn>1234</crn>
                                <easyCollectID>1324567</easyCollectID>
                                <externalBDRID>IVAN1</externalBDRID>
                                <amount>100</amount>
                                <currency>EUR</currency>
                                <urlInfo>PARSHIP.gb</urlInfo>
                                <internalContentDescription>test debit</internalContentDescription>
                        </debReq>
                </request>
        </sRequest>
        </getEasyCollectSingle>
</soapenv:Body>
</soapenv:Envelope>
```

Tabla 34. Mensaje SOAP para hacer debit en clickandbuy

El funcionamiento del sistema de pago de clickandbuy sería:

- Un usuario se registra en clickandbuy, y deposita dinero en su cuenta

- A través del portal web de la aplicación, hace una transacción entre el portal de apuestas y la pasarela de pago. Esto nos proporciona un identificador de usuario de clickandbuy, que asignamos a nuestro usuario en la base dedatos. Esto es importante, ya que si no se ha realizado una transacción previa a través del cliente web, no se podrán realizar transacciones a través del cliente móvil. Esto sucede porque las transacciones mediante móvil son diferentes a las transacciones por web, y clickandbuy exige este paso previo.

- Una vez hecho esto, ya podemos hacer transacciones a través de nuestro dispositivo móvil.

3.3.3 SISTEMA DE APUESTAS

El sistema de apuestas, al igual que el sistema de pago, es un sistema externo. En este caso, obtenemos los eventos, mercados y cuotas a través de los servicios web proporcionados por el portal de apuestas '**Betfair**'. La clase que implementa la obtención de los datos a través de la API de '**Betfair**' es '**BDPApi5.cs**', cuyas siglas significan '**Betfair Developer Program Api**'. El número nos indica la versión de la API. Actualmente, se ha lanzado una nueva versión, '**BDPApi6.cs**', pero en esta aplicación utilizamos una versión anterior, ya que los métodos que en ella aparecen son suficientes para la realización de la aplicación.

```
public class bdpapi5
{
        public bool Login(){}
        public bool Logout(){}
        public void KeepAlive(){}
        public BetfairGlobal.GetEventTypesResp GetSports(){}
        public BetfairExchange.GetInPlayMarketsResp GetInPlayMarkets() {}
        public BetfairGlobal.GetEventsResp GetEvents(){}
        public BetfairExchange.GetMarketResp GetMarket(){}
        public BetfairExchange.GetMarketPricesResp GetMarketPrices(){}
        public BetfairExchange.GetBetResp GetBet(){}
        public BetfairExchange.GetCurrentBetsResp GetCurrentBets(){}
        public BetfairExchange.GetBetHistoryResp GetBetHistory(){}
        public BetfairExchange.GetBetResp GetCurrentBets(){}
        public BetfairExchange.GetAccountFundsResp GetAccountFunds(){}
        public BetfairGlobal.GetSubscriptionInfoResp GetSubscriptionInfo(){}
        public BetfairExchange.GetAccountStatementResp GetAccountStatement(){}
        public BetfairExchange.PlaceBetsResp PlaceBets(){}
        public BetfairExchange.UpdateBetsResp UpdateBets(){}
        public BetfairExchange.CancelBetsResp CancelBets(){}
        public BetfairGlobal.DepositFromPaymentCardResp depositFromPaymentCard(){}
        public BetfairGlobal.WithdrawToPaymentCardResp withdrawToPaymentCard(){}
}
```

Tabla 35. Clase bdpapi5.cs

Los métodos que componen esta clase son:

- *Login*: Conecta al usuario con los servicios web de Betfair.
- *Logout*: Desconecta al usuario de los servicios web de Betfair.
- *KeepAlive*: Mantiene al usuario conectado a los servicios web de Betfair, impidiendo que caduque la sesión abierta.

- *GetSports*: Proporciona los mercados a los que vamos a poder apostar.
- *GetInPlayMarkets*: Obtiene los eventos deportivos de las próximas 24 horas.
- *GetEvents*: Devuelve los eventos relacionados con un mercado.
- *GetMarket*: Devuelve las apuestas pertenecientes a un evento seleccionado.
- *GetMarketPrices*: Devuelve las cuotas de las apuestas relacionadas con un mercado
- *GetBet*: Devuelve información acerca de la apuesta seleccionada.
- *GetCurrentBets*: Devuelve información de las apuestas actuales del usuario.
- *GetBetHistory*: Devuelve información de todas las apuestas realizadas por el usuario.
- *GetCurrentBets*: Devuelve información de una apuesta concreta.
- *GetAccountFunds*: Devuelve información de la cuenta de usuario.
- *GetSubscriptionInfo*: Devuelve información e la cuenta de usuario.
- *GetAccountStatement*: Devuelve información de la cuenta de usuario según unos parámetros dados previamente.
- *PlaceBets*: Realiza una apuesta, o en el caso de pasar por parámetro un vector de apuestas, realiza varias apuestas.
- *UpdateBets*: Actualiza una apuesta.
- *CancelBets*: Cancela una apuesta.
- *DepositFromPaymentCard*: Deposita dinero en la tarjeta de crédito del usuario.
- *WithDrawToPaymentCard*: Retira dinero de la tarjeta de crédito del usuario.

3.3.4 SISTEMA CLASIFICADOR

Vamos a describir ahora el funcionamiento del clasificador de usuarios para obtener un posible mercado en el que el usuario estaría interesado en apostar.

La implementación de esta opción se realiza en aplicación de consola, que se lanzará como una tarea programada una vez por semana.

La clase que implementa el clasificador es '**Program**'.

```
class Program
{
        generarArchivoTrain
        {}
        ejecutaWeka
        {}
}
```

Tabla 36. Clase Program.cs

Los métodos que componen esta clase son:

generarArchivoTrain: crea el archivo de entrenamiento, con formato arff, con las últimas 500 apuestas que se han realizado en el portal.

ejecutaWeka: Aplica el clasificador naive Bayes al archivo de entrenamiento y genera un archivo de salida que nos va a proporcionar el posible mercado en el que podría estar interesado en apostar el usuario que ha entrado en la aplicación.

CAPÍTULO IV:RESULTADOS, CONCLUSIONES Y TRABAJOS FUTUROS

4.1 RESULTADOS

En este apartado vamos a comprobar los resultados de la ejecución de la aplicación en una situación real, y vamos a ver el archivo que produce la ejecución del clasificador '**naive Bayes**' para un usuario de ejemplo.

Resultados de la aplicación

Los resultados que produce la realización de una apuesta podemos observarlos en nuestra base de datos, en las tablas de apuestas realizadas.

IdUsuario	betId	nombreSeleccion	IdMercado	nombreMercado	cuotaBetfair	cuotaDopi	cantidadBetfair	cantidadDopi	fechaApuesta
invdiego	4439062284	0 - 3	20803777	Marcador final	18,5	14,8	120	150	02/02/2008 18:52:
paco	5330550387	AC Oulu	21018590	Cuotas de partido	1,38	1,31	4,75	5	29/05/2008 11:29:
invtato	3019792860	1 - 2	20311872	Marcador final	12,5	11,25	4,5	5	11/03/2007 11:27:
alex	4960009008	0 - 2	20931078	Marcador final	25	20	4	5	11/04/2008 14:15:
invsoler	4532413016	1 - 0	20827150	Marcador final	10,5	8,4	4	5	16/02/2008 12:58:
sergio	5990426267	3 - 0	21165361	Marcador final	65	52	20	25	31/08/2008 1:00:0
vicente	4860558678	Minutos 31 al 40	20902170	Primer gol	8	7,2	45	50	30/03/2008 10:30:
invtato	3262304607	1 - 2	20427882	Marcador final	22	17,6	4	5	27/05/2007 18:50:
huerta	4538291469	1 - 2	20827865	Marcador al descar	25	20	3,1	3,88	16/02/2008 21:39:
invtato	3146220132	Villarreal/Empate	20375193	Descanso/Final	18,5	16,65	4	5	22/04/2007 21:01:
invgoldengol	4590717486	3 - 0	20839559	Marcador final	5,1	4,59	18	20	24/02/2008 19:19:

Figura 33. Datos insertados en la tabla 'Apuestas'

Como podemos observar, hay apuestas de varios usuarios. La finalidad del portal es obtener un beneficio de las apuestas del usuario, con lo que hay una reducción de la cuota de betfair, es decir, si la cuota en betfair está a 1.38, en el portal mostramos 1.31, con el fin de obtener un margen de beneficio establecido previamente por el administrador.

Lógica: necesitamos saber que cantidad tenemos que apostar de la entregada por el cliente a la cuota real de Betfair para que este obtenga la ganancia esperada a la cuota disminuida.

Sea:

- ***CuotaBetfair***= cuota obtenida de Betfair.
- ***CuotaDopi*** = cuota Betfair a la que se le ha aplicado el descuento pertinente.
- ***CantidadDopi*** = cantidad de dinero que el cliente apuesta en el portal.
- ***CantidadBetfair*** = parte de la cantidad del cliente que se va a apostar realmente.

$$CuotaBetfair \times CantidadBetfair = CuotaDopi \times CantidadDopi$$

$$CantidadBetfair = (CuotaDopi \times CantidadDopi) / CuotaBetfair$$

Ejemplo: Si tenemos una cuota real en Betfair de 1.8, un descuento en ese mercado del 20% y el cliente apuesta 100€, tendremos:

CuotaBetfair: 1,8
CuotaDopi: 1,44
CantidadDopi: 100€
CantidadBetfair = (1,44 x 100) / 1,8 = 80€
Ganancia asegurada: 20€

CantidadBetfair debe ser superior a 4€.

En cuanto a los resultados del clasificador, se ha obtenido una batería con las últimas 500 apuestas almacenadas en la base de datos para el entrenamiento. Se han obtenido los resultados utilizando el clasificador '**naive Bayes**', que es un clasificador probabilístico basado en el teorema de Bayes y algunas hipótesis simplificadoras adicionales. Puede predecir tanto las probabilidades del número de miembros de clase, como la probabilidad de que una muestra dada pertenezca a una clase particular.

NAIVE BAYËS

```
=== Run information ===
Scheme:       weka.classifiers.bayes.NaiveBayes
Relation:     tipousuario
Instances:    500
Attributes:   5
              profesion
              sexo
              edad
              localidad
              apuesta
Test mode:    10-fold cross-validation
Time taken to build model: 0.02 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances         581               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0002
Root mean squared error                  0.0028
Relative absolute error                  1.1775 %
Root relative squared error              2.9126 %
Total Number of Instances              581
=== Detailed Accuracy By Class ===
               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        26490598
                 1         0          1         1         1          1        1
                 1         0          1         1         1          1        26487083
                 0         0          0         0         0          ?        26426066
                 0         0          0         0         0          ?        26487081
                 1         0          1         1         1          1        26484275
                 1         0          1         1         1          1        26483067
                 1         0          1         1         1          1        26483415
                 0         0          0         0         0          ?        26483666
                 0         0          0         0         0          ?        26473395
                 0         0          0         0         0          ?        26471393
                 0         0          0         0         0          ?        26469876
                 1         0          1         1         1          1        26466453
                 0         0          0         0         0          ?        26466452
                 1         0          1         1         1          1        26458826
                 1         0          1         1         1          1        26455856
                 1         0          1         1         1          1        26455833
               ---------------------------------------------------------------------
               ---------------------------------------------------------------------
                 0         0          0         0         0          ?        26449128
                 0         0          0         0         0          ?        26447270
                 0         0          0         0         0          ?        26449217
                 0         0          0         0         0          ?        26449123
                 0         0          0         0         0          ?        26448480
                 0         0          0         0         0          ?        26449204
                 0         0          0         0         0          ?        26446988
Weighted Avg.    1         0          1         1         1          1
```

Tabla 37. Resultados del clasificador ‘naive Bayes’

```
=== Confusion Matrix ===
a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q  r  s  t  u  v  w     <-- classified as
40 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   a = 26490598
0 37  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   b = 1
0 36  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   c = 26487083
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   d = 26426066
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   e = 26487081
0  0  0  0  0 36  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   f = 26484275
0  0  0  0  0  0 36  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   g = 26483067
0  0  0  0  0  0  0 36  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   h = 26483415
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   i = 26483666
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   j = 26473395
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   k = 26471393
0  0  0  0  0  0  0  0  0  0  0 36  0  0  0  0  0  0  0  0  0  0  0  0   l = 26469876
---------------------------------------------------------------------------------
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 36  0  0   v = 26456282
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 36  0   w = 26454733
```

Tabla 38. Matriz de confusión del clasificador 'naive Bayes'

Podemos observar que el clasificador es muy fiable, ya que el porcentaje de acierto es del 100%. Esto es debido a que la muestra es muy homogénea, lo que ha provocado que el clasificador no haya tenido problemas para hacer su trabajo.

A continuación vemos pruebas para los clasificadores '**J48** e '**IBK**'.

J48

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     tipousuario
Instances:    581
Attributes:   5
              profesion
              sexo
              edad
              localidad
              apuesta
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

profesion = profesor: 26487083 (36.0)
profesion = camarero: 26484275 (36.0)
profesion = comercial: 1 (37.0)
profesion = ingeniero: 26483067 (36.0)
profesion = albañil: 26483415 (36.0)
profesion = camionero: 26466453 (36.0)
profesion = pescador: 26458826 (36.0)
profesion = fontanero: 26455856 (36.0)
profesion = piloto: 26455833 (36.0)
profesion = periodista: 26455834 (36.0)
profesion = taxista: 26452881 (36.0)
profesion = empresario: 26456313 (36.0)
profesion = pintor: 26456284 (36.0)
profesion = futbolista
|   sexo <= 0: 26454733 (36.0)
|   sexo > 0: 26490598 (40.0)
profesion = abogado: 26456282 (36.0)
Number of Leaves  :   16
Size of the tree :        18
Time taken to build model: 0.08 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances         581               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
Total Number of Instances              581
```

Tabla 39. Resultados para el clasificador 'J48

La matriz de confusión para el clasificador J48 es la misma que para el clasificador naive Bayes, y como podemos observar también da unos resultados totalmente fiables.

IBK

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R
first-last\""
Relation:     tipousuario
Instances:    581
Attributes:   5
              profesion
              sexo
              edad
              localidad
              apuesta
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         581               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0007
Root mean squared error                  0.0044
Relative absolute error                  3.9394 %
Root relative squared error              4.5745 %
Total Number of Instances              581
```

Tabla 40. Resultados para el clasificador 'IBK

Como podemos observar, el clasificador es un poco más rápido que el J48, y la matriz de confusión sigue siendo muy parecida. En cuanto a la fiabilidad, es la misma que para los dos algoritmos anteriores.

Por último, vamos a ver el perceptrón multicapa.

PERCEPTRÓN MULTICAPA

```
=== Run information ===

Scheme:       weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a
Relation:     tipousuario
Instances:    581
Attributes:   5
              profesion
              sexo
              edad
              localidad
              apuesta
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

Sigmoid Node 0
    Inputs    Weights
    Threshold    -0.3018504889304051
    Node 102    -1.0193521706817792
    Node 103    -0.9400584820412105
    Node 104    -0.05161360696015464
    Node 105    -0.6049544061595756
    Node 106    0.09412591412199244
    Node 107    1.509137145661079
    Node 108    0.9528441265410627
    Node 109    -0.32462693637409196
    Node 110    -0.07549966101571963
    Node 111    0.34508669879575493
    Node 112    -1.402311718081891
    Node 113    1.2880179278705797
    Node 114    0.4888508745363304
    Node 115    0.14503739942426294
    ------------------------------------------------------
    Time taken to build model: 339.19 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         581               100      %
Incorrectly Classified Instances       0                 0        %
Kappa statistic                        1
Mean absolute error                    0.0009
Root mean squared error                0.0048
Relative absolute error                5.1004 %
Root relative squared error            5.0429 %
Total Number of Instances              581
```

Tabla 41. Resultados para el clasificador 'Perceptrón multicapa'.

Podemos observar que los resultados son exactos a los algoritmos anteriores en cuanto a fiabilidad, pero sin embargo este algoritmo tarda mucho mas en clasificar, debido a que utiliza el algoritmo de retropropagación o backpropagation.

Una vez ejecutado el clasificador, obtenemos un archivo de resultados que nos proporciona una predicción del mercado al que el usuario podría apostar. El archivo es como el siguiente:

```
=== Predictions on test data ===


 inst#     actual  predicted error prediction

    1       1:? 26:2645287   +   0
```

Tabla 42. Resultados finales

Con este archivo de resultados, podemos obtener, en el caso que nos ocupa, el mercado al que probablemente apostaría un usuario con las características del que se ha identificado en el portal, aunque esta clasificación se puede usar en muchos campos, como por ejemplo reconocimiento de formas, de caras, etc.

4.2 CONCLUSIONES

Una vez terminado el proceso de desarrollo de la aplicación, podemos ver cómo se ha conseguido integrar en un entorno cliente/servidor elementos que están en auge actualmente, como los dispositivos móviles y las apuestas por internet, y todo ello haciendo uso de herramientas de bajo coste.

Este sistema permite un acceso a información para apostantes desde cualquier lugar, de manera que incluso podemos estar viendo un acontecimiento deportivo en vivo y poder realizar nuestra apuesta sobre él.

A su vez, permite la administración de mercados y/o datos personales del usuario por parte de un administrador general.

Por otro lado, y gracias a un clasificador de usuarios, el sistema nos va a ofrecer posibles mercados en los que, dependiendo de nuestro perfil de usuario, podríamos estar interesados en realizar apuestas. Esto es una ventaja considerable, ya que está demostrado que el principal mercado al que apuestan los usuarios es a partidos de fútbol de la liga española, con lo que se nos ahorraría el proceso de búsqueda de dicho mercado al indicarnos claramente cómo llegar hasta él.

Para el desarrollo de esta aplicación es necesario tener conocimientos de varias tecnologías informáticas:

- ASP .NET, con los lenguajes que lo forman:
  - C#, orientado a objetos, diseñado por Microsoft.
  - Javascript: Lenguaje de programación orientado a objetos, ejecutado en la máquina cliente, de licencia libre, desarrollado por Netscape, que interactúa con el código HTML, utilizado para añadir interactividad a una página web.
  - Ajax, para comunicación entre el código C# y el código Javascript.
- SQLServer para la gestión de base de datos.
- Métodos para análisis y clasificación de objetos (API de WEKA)
- Dispositivos hardware tan extendidos como los teléfonos móviles, pdas, etc.

La implementación de este sistema tiene un amplio abanico de posibilidades, ya que puede haber un gran número de áreas en las que se están empezando a utilizar, como:

- Locales de juego
- Casinos

- Peñas deportivas
- Etc…

4.3 TRABAJOS FUTUROS

Los trabajos que se pueden realizar a partir de este sistema de apuestas son múltiples:

- Incorporación de otros tipos de usuarios como bookmakers (corredores), gerentes, etc, que se encarguen de realizar apuestas en nombre de otros usuarios
- Mejorar la integración del sistema clasificador para que produzca resultados más óptimos que los actuales, por ejemplo con una base de conocimiento.
- Incorporación al sistema de peñas de apuestas, de modo que se puedan realizar apuestas con cantidades de dinero más grandes.
- Incorporación de mercados propios al sistema, de manera que un administrador introdujese mercados en el sistema y no hubiese que pagar una cantidad a Betfair por apuesta realizada.
- Incorporación de otros sistemas de pago, como Paysafecard, Ukash, MoneyBookers, etc.

BIBLIOGRAFÍA

LITERATURA

Freeman, J.A., and D.M. Skapura. 1991. Neural networks: algorithms, applications and programming techniques (Computation and Neural Systems Series) 401 p. Addison-Wesley Pub. Co., Reading, Massachusetts, USA.

Hilera, J.R., y V.J. Martínez. 2000. Redes neuronales artificiales. Fundamentos, modelos y aplicaciones. 390 p. Editorial Alfaomega Ra-Ma, Madrid, España.

Campbell, C., and A. Temporel. 2002. An investigation of the delta rule and gradient descent algorithm. Available at http://alex.boulderdash.org/NN.pdf.

[ARMAÑANZAS] Medidas de Filtrado de Selección de Variables mediante la plataforma

"Elvira". Rubén Armañanzas Arnedillo, Iñaqui Inza Cano. Proyecto Fin de Carrera. Universidad del País Vasco. 2004.

PÁGINAS WEB

TECNOLOGÍA MÓVIL

[nodrizza]http://www.nodrizza.com/soluciones-que-movilizan-informacion-corporativa-en-tiempo-real/images/stories/Docs/tecnologas_mviles_e_inalmbricas_desarrollan_el_concepto_productividad.pdf
[blogwimax] http://blogwimax.com/que-es-wimax/
[domodesk] http://www.domodesk.com/content.aspx?co=104&t=146&c=43

XML

[w3c] http://www.w3c.es
[w3] http://www.w3.org/XML/
[el rincón del programador]
http://www.elrincondelprogramador.com/default.asp?pag=articulos/leer.asp&id=21
[geocities] http://es.geocities.com/guia_de_xml/estructura.html
[ulgpc] http://www.ulpgc.es/otros/tutoriales/xml/Estructura.html
[mitecnologico] http://www.mitecnologico.com/Main/EstructuraDatosXml
[gavd]http://www.gavd.net/servers/sharepointv3/spsv3_item.aspx?top=inf&itm=603

ASP.NET

[desarrolloweb] http://www.desarrolloweb.com
[uco]http://www.uco.es/~i72cafef/tiagdi/introduccion.html
[programación.com] http://www.programacion.com/asp/articulo/aspnet_quees/
[netveloper] http://www.netveloper.com/contenido2.aspx?IDC=143_0
[devjoker] http://www.devjoker.com/asp/ver_contenidos.aspx?co_contenido=25
[whatisbyarvind] http://whatisbyarvind.blogspot.com/2008/01/hat-is.html
[clickear]http://www.clikear.com
[msdn] http://msdn.microsoft.com/

SERVICIOS WEB

[webservices]http://www.webservices.org
[ugr] http://geneura.ugr.es/~jmerelo/ws/

WEKA

[weka] http://www.cs.waikato.ac.nz/ml/weka/
[ua] http://www.dccia.ua.es/~miguel/tesis/tesis006.html
[softwarelibre]
http://softwarelibre.unsa.edu.ar/docs/descarga/2003/curso/htmls/redes_neuronales/x105.html
[scielo] http://www.scielo.cl/scielo.php?pid=S0365-28072005000100007&script=sci_arttext
[ehu] http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t6bayesianos.pdf
[ohm] http://ohm.utp.edu.co/neuronales/Capitulo2/Perceptron/AntecedentesP.htm

SISTEMAS DE PAGO
[Clickandbuy]http://www.clickandbuy.com

DESCRIPCIÓN DEL SISTEMA
[alegsa]http://www.alegsa.com.ar/Dic/diagrama%20de%20flujo%20de%20datos.php