Memoria del Proyecto

Aplicación web móvil de venta de entradas

Alumno: Vladimir Masteaev

Director: Jorge Arranz Villegas

Ponente: Xavier Burgués Illa

Datos del Proyecto:

Titulo del proyecto: Aplicación web móvil de venta de entradas.

Alumno: Vladimir Masteaev

Titulación: Ingeniería Técnica en Informática de Sistemas

Créditos: 22.5

Director: Jorge Arranz Villegas

Ponente: Xavier Burgués Illa

Miembros del tribunal

Presidente:

Vocal:

Secretario:

Cualificación:

Cualificación numérica:

Cualificación descriptiva:

Fecha:

Índice:

Agradecimientos.

En este apartado quería mostrar mis agradecimientos a aquellas personas que me han ayudado todo el tiempo que he pasado estudiando y todo el tiempo que he pasado desarrollando este proyecto.

Primero de todo quería agradecer a mi familia el apoyo que me han dado para empezar a estudiar y sobre todo el tiempo que he dedicado a los estudios. También por la paciencia y ánimos que me han dado a la hora de realizar este proyecto.

En segundo lugar quería agradecer a mi pareja, Liudmila, la paciencia a lo largo del tiempo que he estado estudiando y también por sus ánimos durante los meses de realización del proyecto.

En tercer lugar agradezco a todos los profesores del IES Joan Brossa por sus aportaciones, comprensión, ayuda y buena base que me han proporcionado en mis primeros días en España. También a mis compañeros de la universidad y a todos los profesores de la FIB que he tenido a lo largo de mis estudios.

También quería agradecer a mis jefes del trabajo por la confianza en cogerme de practicas, y luego ofrecerme un puesto. A los compañeros del trabajo por la ayuda, conocimientos aportados y buen compañerismo.

Finalmente quería agradecer al ponente del proyecto por sus rápidas respuestas a las preguntas y la ayuda aportada.

Introducción

Ticket Bureau Sl. es una empresa que está dedicado a la venta de entradas para diferentes deportes, espectáculos y conciertos. Ofrece grande selección de eventos locales como internacionales . A su tiempo es agencia oficial de FC Barcelona, RCD Espanyol y Atlético de Madrid. Una de las herramientas que dispone para vender sus entradas son tiendas online a través de cuales vende mayoría de las entradas y también dispone de 3 tiendas físicas en el centro de la ciudad Barcelona.

Equalid Solutions es empresa de desarrollo de aplicaciones donde yo trabajo, y que hace mantenimiento y desarrollo de nuevas funcionalidades para el sistema de cliente Ticket Bureau.

Los últimos móviles tienen buenas características de velocidad, conexión a internet y pantalla táctil que permite navegar por internet con facilidad.

Ahora mismo se puede acceder a las tiendas de Ticket Bureau desde móvil pero es muy incomodo. Motivación de Ticket Bureau para crear esta aplicación es intentar vender mas entradas. Y esto se puede conseguir facilitando al cliente una plataforma cómoda para este fin atreves de un dispositivo móvil que llevamos siempre encima.

Ahora viene introducir de que elementos esta formada el sistema actual.

El sistema actual es bastante complejo dadas las reglas de negocio que lleva implementadas.

El sistema está formado por los siguientes componentes:

- Una Bases de datos que guarda toda la información .
- Unas clases de negocio que llevan parte de la lógica de negocio.
- Diferentes Back Offices para gestionar el contenido y comportamiento del sistema.
- Un Web Service[1] que se encarga de transferir información desde la base de datos a las tiendas y viceversa.
- 4 tiendas independientes online. Spain Ticket Bureau (*STB*) es la tienda principal del negocio y 3 tiendas mas: Ticket Bureau (*TBUR*), Tickets FC (*TFC*) y también una "Tienda Blanca" (*TBNueva*) la cual admite usuarios registrados que representan agencias, y también integra algunas funciones básicas de gestión para estas agencias registradas. También existen otras tiendas online que obtienen la información necesaria a través del Web Service, pero estas tiendas son desarrolladas por terceros.
- Carrito externo de la compra (*TBPV*) desarrollado de forma independiente de las tiendas. Este carrito esta pensado para las tiendas online externas. Las 4 tiendas STB,TBUR,TFC y TBNueva actualmente tienen sus propios carritos de la compra y no usan el externo.

[1] Un **servicio web** (en inglés, *Web services*) es una tecnología que utiliza un conjunto de

El esquema del sistema explicado anteriormente es el siguiente:

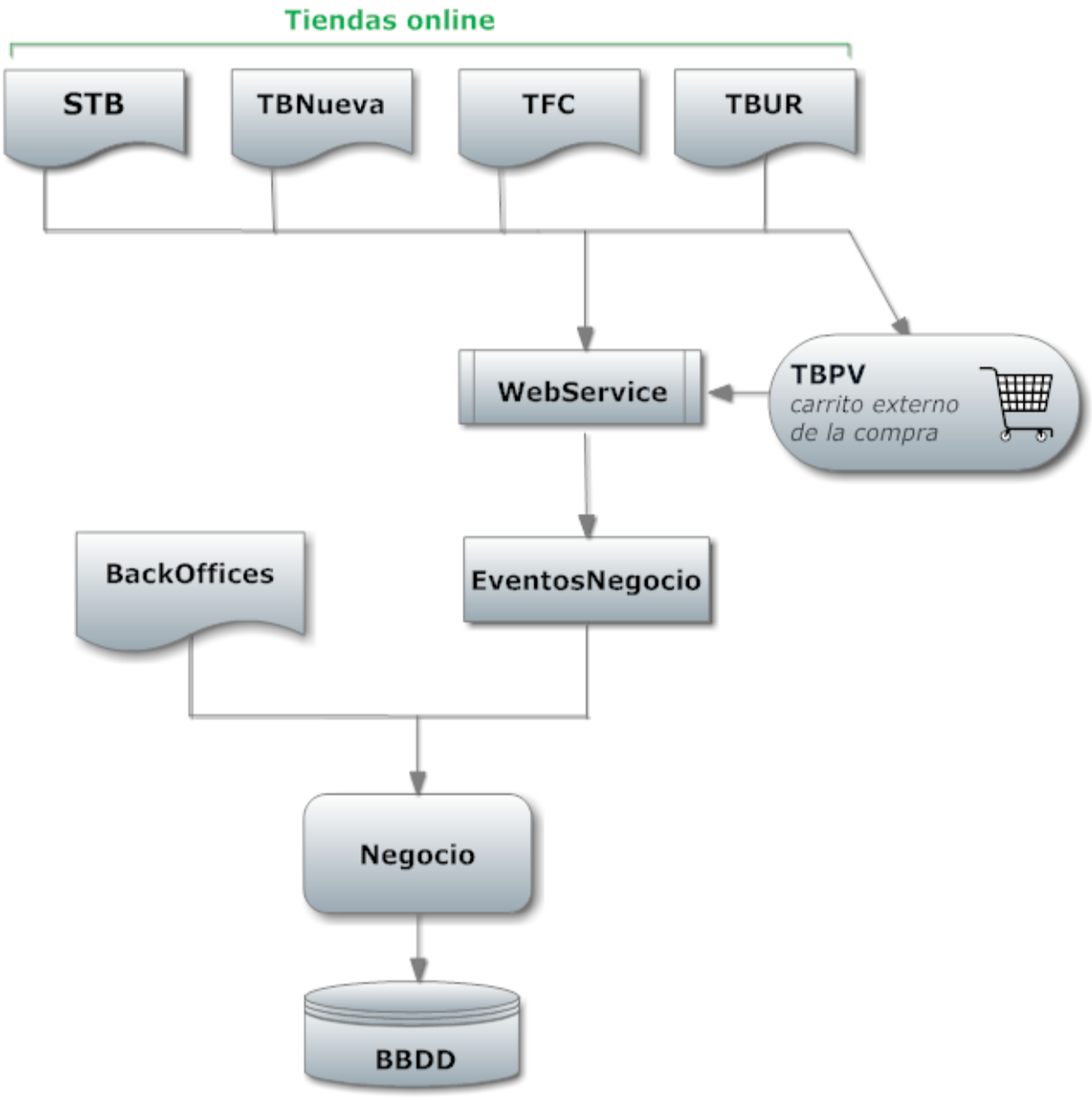

Ilustración 1

Objetivos y alcance del proyecto

Este proyecto está orientado a crear una aplicación web de venta de entradas para dispositivos móviles actuales.

El objetivo principal del proyecto consiste en crear una pagina web que permitirá buscar los eventos, mostrar detalles de las entradas y comprarlas, conviviendo en un sistema web complejo y en constante crecimiento. Esta página web debe ser fácil de usar utilizando terminales móviles con pantalla táctil.

Para una buena navegación por la pagina web se intentara crear un diseño cómodo, agradable y fácil de entender para todos usuarios que se conecten mediante un dispositivo móvil. Éste es el objetivo principal de la aplicación.

Para llevar a cabo la pagina web se necesitaría implementar diferentes puntos para crear la web en si y también hacer modificaciones de otras aplicaciones del sistema y que estos cambios no afectan al funcionamiento de otras partes del sistema.

En el listado de abajo se listan los puntos principales para el desarrollo de la web:

- Selección y creación de un diseño agradable para el móvil.
- Creación de aplicación web móvil.
- Mantener el funcionamiento del sistema existente

- Usar del Web Service existente del sistema para transferencia de datos entre la BD y la aplicación web.
- Usar el carrito externo de la compra (TBPV) para comprar entradas
- Adaptar el diseño del carrito TBPV para el móvil conservando el diseño anterior para ordenadores.
- Que actué en el sistema como tienda online Spain Ticket Bureau (*STB*) es decir que el contenido mostrado será igual que en STB y los pedidos entren con referencia de STB.
- La tienda STB será modificada de tal manera que si detectara que se está accediendo desde un dispositivo móvil redireccionar a la a la nueva aplicación diseñada en este proyecto para este fin.

Aspectos que no entraran en el alcance del proyecto.

- El proyecto en esta fase no incluye la puesta en marcha de la web, solo su creación y puesta en el entorno test. Posteriormente se procederá migrar la web a un servidor existente de Ticket Bureau SL.

Planificación del trabajo

Una vez explicado el contexto del proyecto pasaremos a detallar la planificación inicial.

La planificación del trabajo se ha dividido en varias fases para cubrir los objetivos principales mencionados arriba.

- Análisis de requisitos
- Especificación
- Diseño
- Implementación de las partes
- Pruebas
- Memoria

Se piensa invertir unos 4 - 6 horas cada día, en total unos 24 - 36 horas semanales.

Entonces previamente las horas están repartidas de la siguiente manera cogiendo de media 5 horas diarias:

	Días	Horas
Análisis de requisitos	6	30
Especificación	5	25
Diseño	10	50
Implementación de las partes	50	250
Pruebas	5	25
Memoria	14	70
Configuración del entorno	5	25
En total	**95**	**475**

Tabla 1: Tabla de horas estimadas

En la ilustración 1 podemos ver la planificación previa estimada.

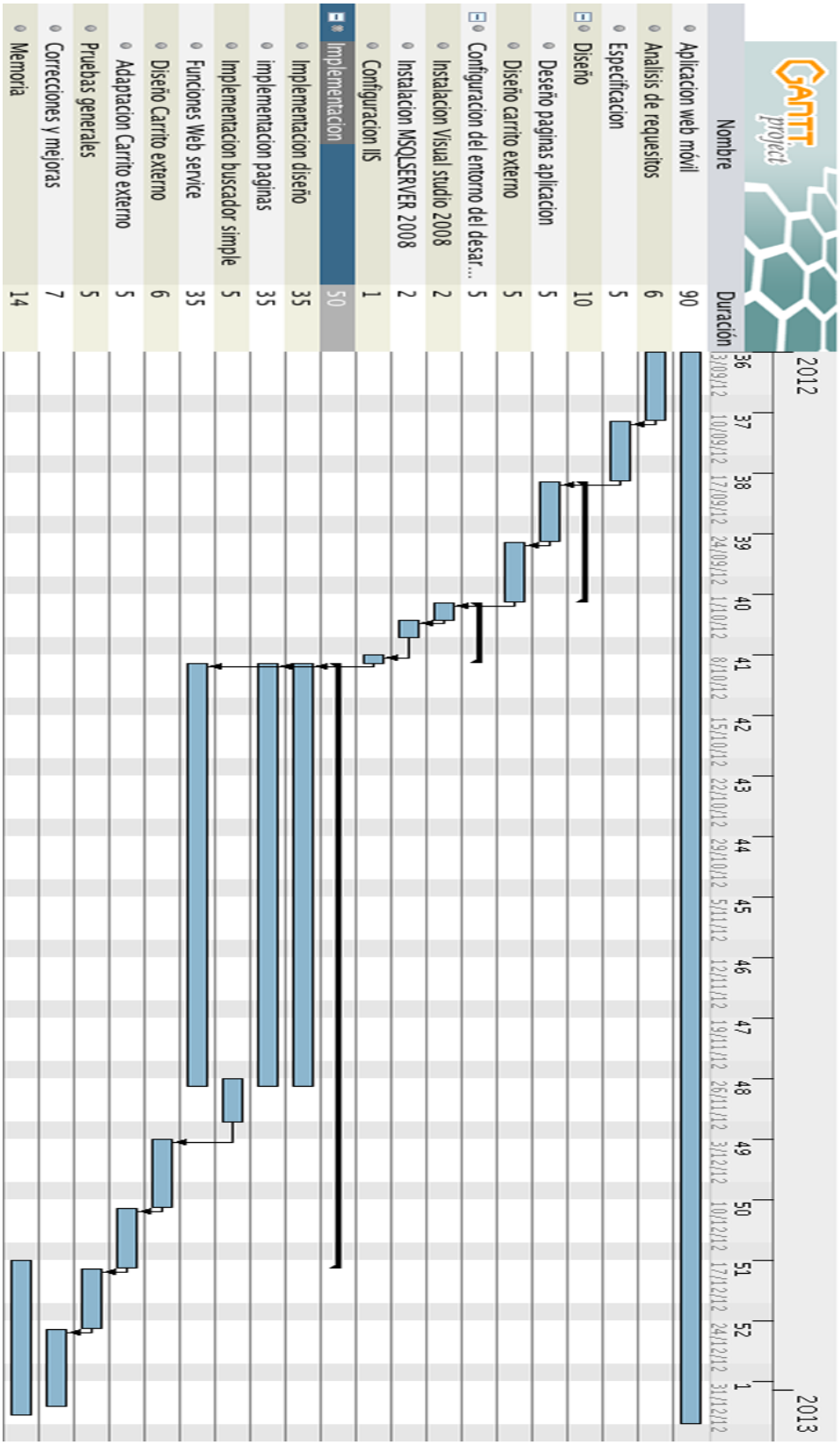

Ilustración 2

Costes

Una vez detallado las horas de la planificación se puede calcular el coste de implementar el proyecto usando las horas estimadas anteriormente y otros factores que pueden influir en el coste.

En caso de una web al coste puede influir la tecnología escogida. Dependiendo de la tecnología elegida será necesario obtener licencias o no. También es necesario un servidor web para alojar la pagina y base de datos.

El sistema web de Ticket Bureau SL esta desarrollado con Visual Studio .Net y una base de datos SQL Server. Estos dos sistemas no son gratuitos, y sus licencias cuestan dinero, pero Ticket Bureau SL ya tiene estas licencias por tanto se pueden aprovechar sin ningún coste adicional.

También dispone de un servidor web que dispone de recursos suficientes para alojar una nueva aplicación sin afectar el rendimiento de las paginas existentes y sin producir un incremento en el coste del proyecto.

De esta manera el coste estará formado solo por las horas de desarrollo que también incluyen las horas de testeo.

Para el cliente una hora de desarrollo tiene un coste de 35 euros del programador y 45 euros del analista. Anteriormente en planificación se ha estimado una duración 475 horas sin contar la documentación del proyecto. Haciendo cálculos con estos valores obtenemos siguientes costes:

	Días	Horas	Coste hora	Coste
Análisis de requisitos	6	30	50	1500
Especificación	5	25	50	1250
Diseño	10	50	35	1750
Implementación de las partes	50	250	35	8750
Pruebas	5	25	35	875
Configuración del entorno	5	25	30	750
En total	**81**	**405**		**14875**

Tabla 2: Coste del proyecto

Estructura jerárquica de datos

Para crear contenido correcto y entender los requisitos en nueva aplicación tenemos que saber la estructura jerárquica de los datos que maneja el sistema.

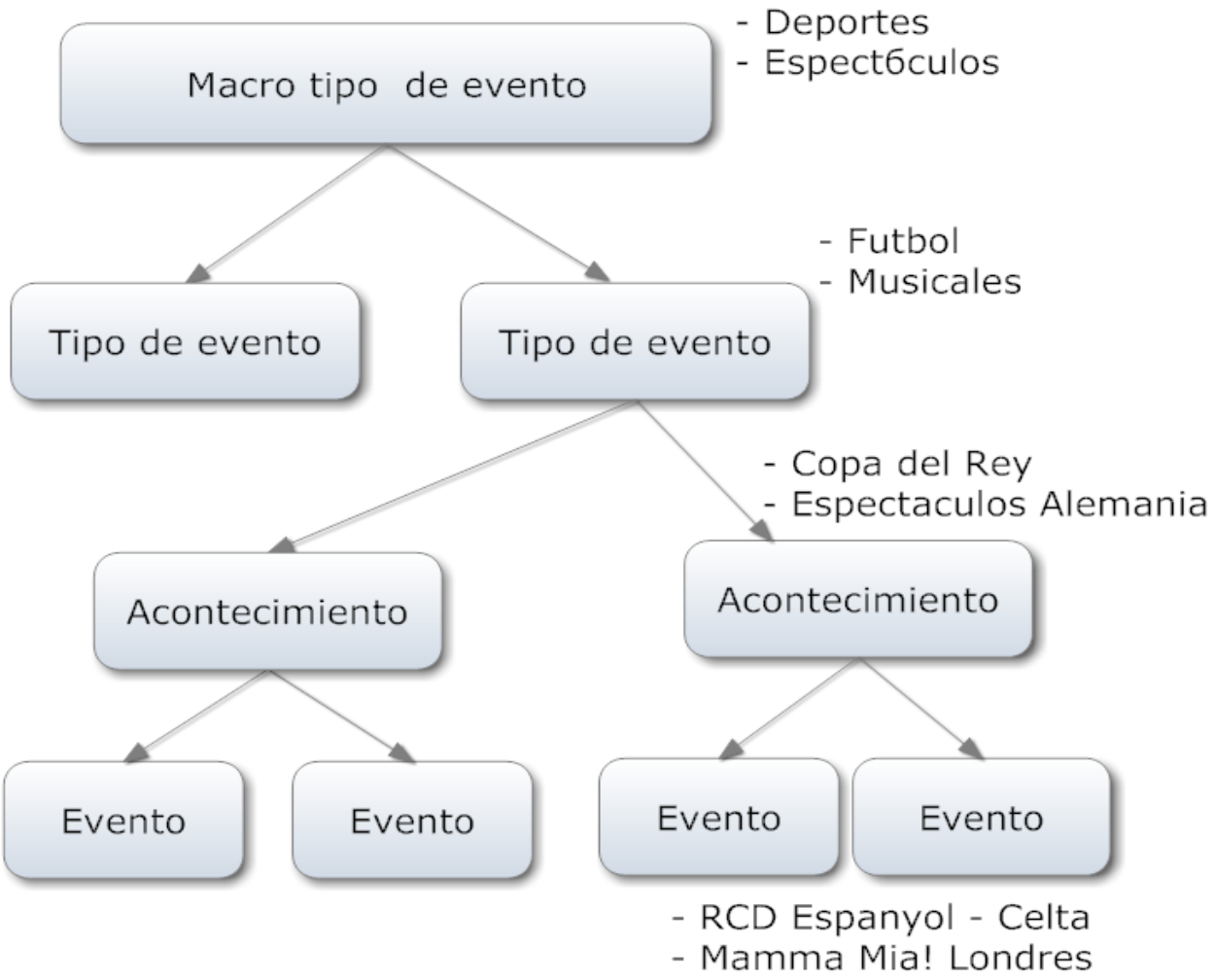

Ilustración 3

El sistema ofrece 3 niveles para que se clasifiquen diferentes eventos. Estos niveles son dinámicos. El criterio de subdivisión es libre y Ticket Bureau puede crearlos tantos cuanto quiere a través del Back Office que dispone para este fin.

Ahora con los ejemplos explicare que significa cada nivel de la jerarquía de evento que acabamos de ver en la ilustración y también ver que la subdivisión es libre de criterio.

Macro Tipo de Evento → engloba división por actividad global. Por ejemplo deporte, conciertos, y espectáculos es el nivel superior de la jerarquía que se entiende en el sistema por *Macro Tipo de Evento*.

Tipo de Evento → Como ya se entiende en la esquema de jerarquía de evento el *Tipo de Evento* es la división de Macro Tipo de Evento en actividades especificas. Un ejemplo de esta división es: fútbol, musicales, y etc..

Acontecimiento → Es la división de *Tipo de Evento* en diferentes actividades. Un ejemplo de esta distribución:

- Para Tipo de Evento “Fútbol” tenemos : Champions League, Copa del Rey, Copa de la UEFA y etc..
- Para Tipo de Evento “Espectáculos” tenemos: Espectáculos Alemania, Espectáculos España y etc..

Evento → Es el ultimo nivel de la jerarquía de evento el evento en si. Un evento es un partido de fútbol o un concierto.

El evento tiene al menos un participante asignado, que es el cantante o el equipo que juega en casa, y un participante adicional que es el equipo visitante. El sistema por restricciones de implementación inicial permite asignar como mínimum un participante y máximum dos.

Unos ejemplos de Evento son: Polonia – Grecia, RCD Espanyol – Celta, Mamma Mia! Londres y etc..

En el evento el participante es necesario para poder en algunos casos agrupar los eventos por participante. Un ejemplo para este fin será el acontecimiento La Liga. En La Liga un equipo tiene varios partidos contra varios equipos. Entonces el Ticket Bureau puede definir agrupar estos eventos por participante principal para facilitar la búsqueda del evento preferido. En caso que el evento tenga mas de dos participantes se crea en el sistema un participante genérico. Por ejemplo el torneo de Wimbledon esta creado un participante Wimbledon.

Una vez ya tenemos conocido la estructura pasamos definir los requisitos del sistema.

Análisis de Requisitos

Todos proyectos incluyen una serie de requisitos que hay que cumplir para conseguir que el funcionamiento sea correcto. Al inicio de un proyecto hay que estudiar y obtener los requisitos que determinen de manera formal las funcionalidades de éste, que se puedan englobar dentro del tiempo de desarrollo estimado.

Existen dos tipos de requisitos: Los funcionales y los no funcionales. Los requisitos funcionales hacen referencia aquellos que definen un comportamiento o funcionalidad del proyecto. Los requisitos no funcionales hacen referencia a aspectos mas generales y de calidad.

Para análisis de requisitos hay definidos métodos exhaustivos para tal fin. Pero son complejos y completos que mejor sirven para proyectos muy grandes y complejos donde se necesitan mas detalles. Como este proyecto no es este caso se ha definido con el director del proyecto no usar ningún método exhaustivo por 2 motivos:

1. Los requisitos se han dado por supuesto por el conocimiento de la aplicación actual porque Equalid Solutions esta dando el soporte al sistema del cliente unos cuantos años.
2. Para ahorrar el tiempo.

Por tanto con el compañero del trabajo con el cual llevamos el cliente Ticket Bureau definimos los requisitos a partir de los requisitos de aplicación actual

Estos requisitos fueron aceptados por el director del proyecto y también por cliente para primera versión de aplicación.
En este apartado analizaremos cuales son los requisitos de nuestro proyecto.

Requisitos funcionales

Los requisitos funcionales son los necesarios para permitir a un cliente utilizar la web para buscar las entradas de diferentes maneras y, una vez encontrada la entrada deseada, poder comprar las entradas para el evento escogido desde un teléfono móvil con conexión a internet.

1. Destacar:

    1.1 La página tiene que mostrar un listado de eventos destacados[2] (máximum 5). Estos eventos son gestionados exclusivamente por el equipo de Ticket Bureau SL.

2. Buscar:

    2.1 La página tiene que permitir buscar los eventos.

3. Resultado de búsqueda:

    1.1. Los resultados serán fáciles de visualizar aunque sean de gran volumen

[2] Los eventos destacados son eventos a los cuales se quiere dar más importancia y visibilidad.

2. Ver Evento:
   2.1. El usuario ha de poder ver los eventos
3. Ver Entradas:
   3.1. El usuario ha de poder ver las entradas de un Evento
4. Ver Participantes:
   4.1. El usuario ha de poder ver los participantes
5. Ver Macro tipo de evento:
   5.1. El usuario ha de poder ver Macro tipo de evento
6. Ver Tipo evento:
   6.1. El usuario ha de poder ver Tipos de eventos de Macro tipo de evento escogido
7. Ver Acontecimiento:
   7.1. El usuario ha de poder ver Acontecimientos de un Tipo de evento escogido
8. Cambiar Idioma:
   8.1. Se podrá escoger idioma de una lista de idiomas permitidos en la web.
9. Ver idioma:
   9.1. El sistema ha de mostrar el idioma actual escogido.
10. Zoom:
    10.1. No se podrá hacer el zoom de la aplicación.
11. Ordenar por precio:
    11.1. El listado de tickets ha de salir ordenado por el precio.

Requisitos no funcionales

- La página tiene que tener un diseño agradable a la vista.
- El contenido de la pagina tiene que aprovechar al máximum el espacio disponible.
- La página tiene que utilizar el carrito externo *TBPV* que tiene Ticket Bureau SL. para el proceso de compra.
- Tiene que ser satisfactoriamente rápida y no superar 10 segundos.
- La pagina tiene que tener una navegación cómoda y simple.
- La pagina tiene que tener una navegación jerárquica.
- La página debe ser compatible y tener el mismo comportamiento en las últimas versiones de navegadores móviles:
    - Crome 18.0.1.1025469
    - Firefox 18.0
    - Internet 4.1.1(navegador por defecto de android)
    - Opera Mobile 12.1
- Tiene que cumplir los estándares HTML y CSS.

Especificación

En este apartado describiremos con más detalle lo que la aplicación podrá hacer. Es decir, se explicará con más detalle todo lo que podrá hacer un usuario (en nuestro caso, un cliente) con la aplicación.

Cuanto más clara y descriptiva sea esta especificación más claro será el desarrollo de la página. En el desarrollo se tendrán en cuenta todos los casos de uso que mencionaremos seguidamente.

En la página tendremos un actor que será un cliente. Éste será por lo tanto el único actor que aparecerá en los casos de uso que interactuara con el sistema (aplicación). El sistema responderá al usuario con la información relacionada con su petición.

Otras cosas que hay que especificar son restricciones de carrito de compra *TBPV* . Es que este carrito por su diseño inicial no permite comprar entradas de diferentes partidos. El carrito no almacena entradas, por tanto al escoger el ticket el sistema muestra el carrito donde tienes que terminar la compra. Y si el usuario abandona el carrito sin terminar la compra pierde la entrada que tenia. Y si después selecciona otro ticket de otro o del mismo evento en el carrito se mostrara solo este ultimo ticket.

Diagramas de clases UML

Diagrama de Acontecimiento

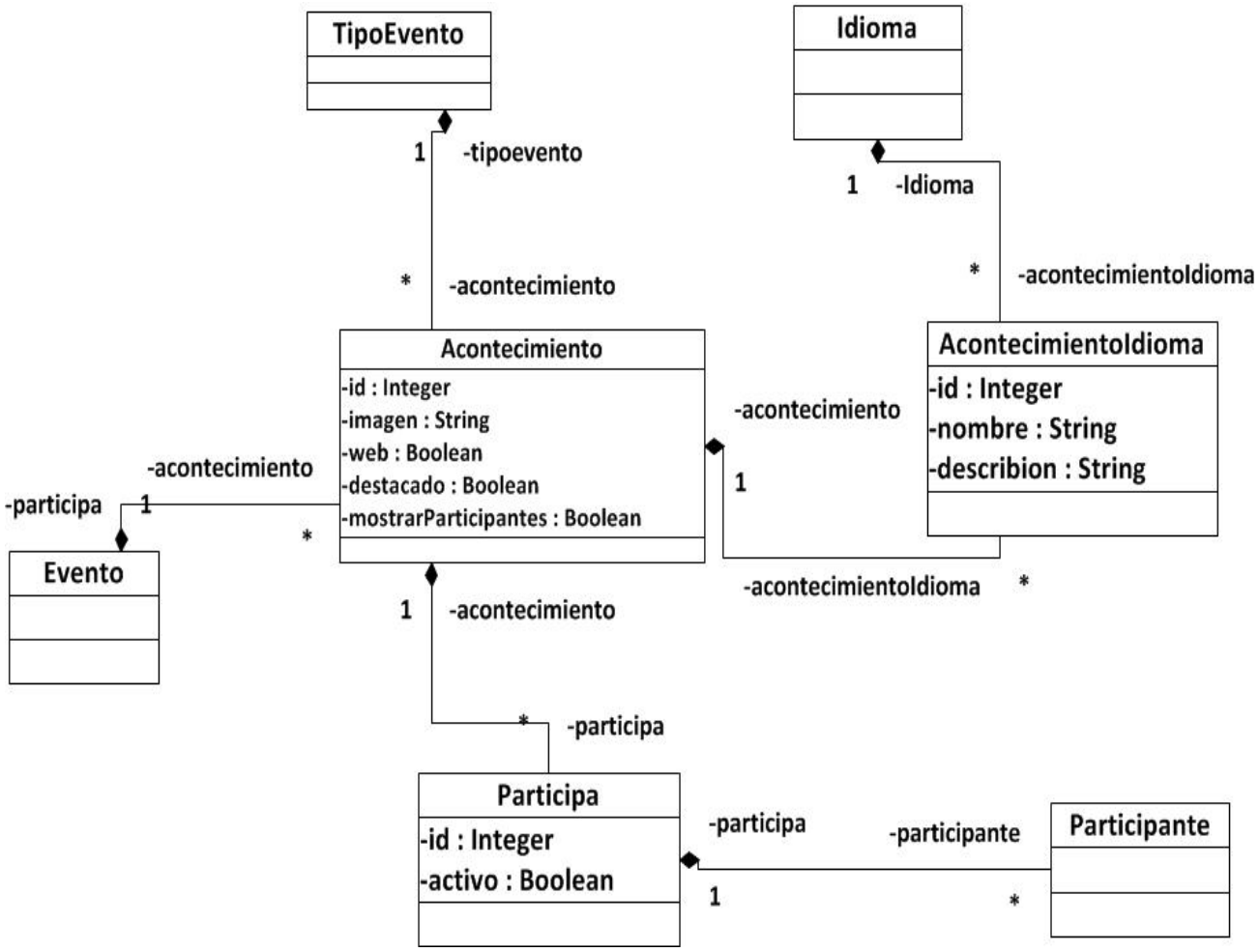

Ilustración 4

Diagrama de Macro tipo evento

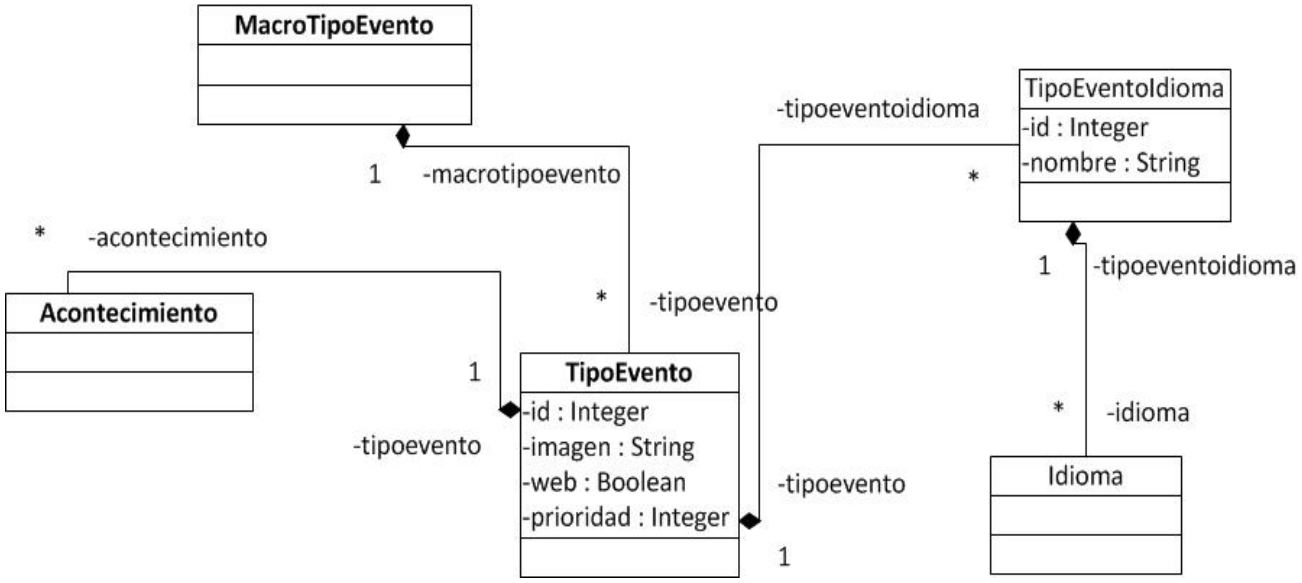

Ilustración 5

Diagrama de Evento

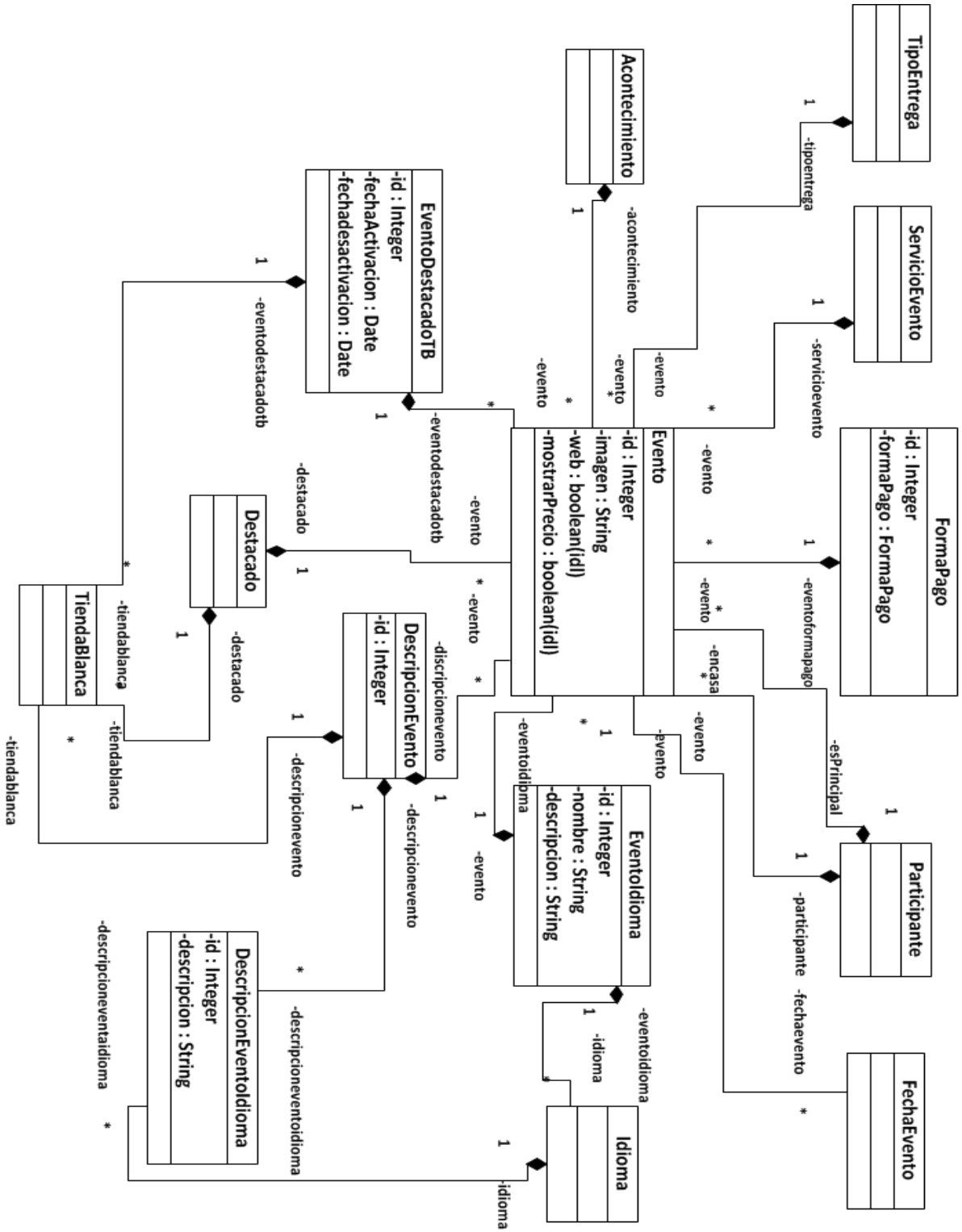

Ilustración 6

Diagrama de Tipo evento

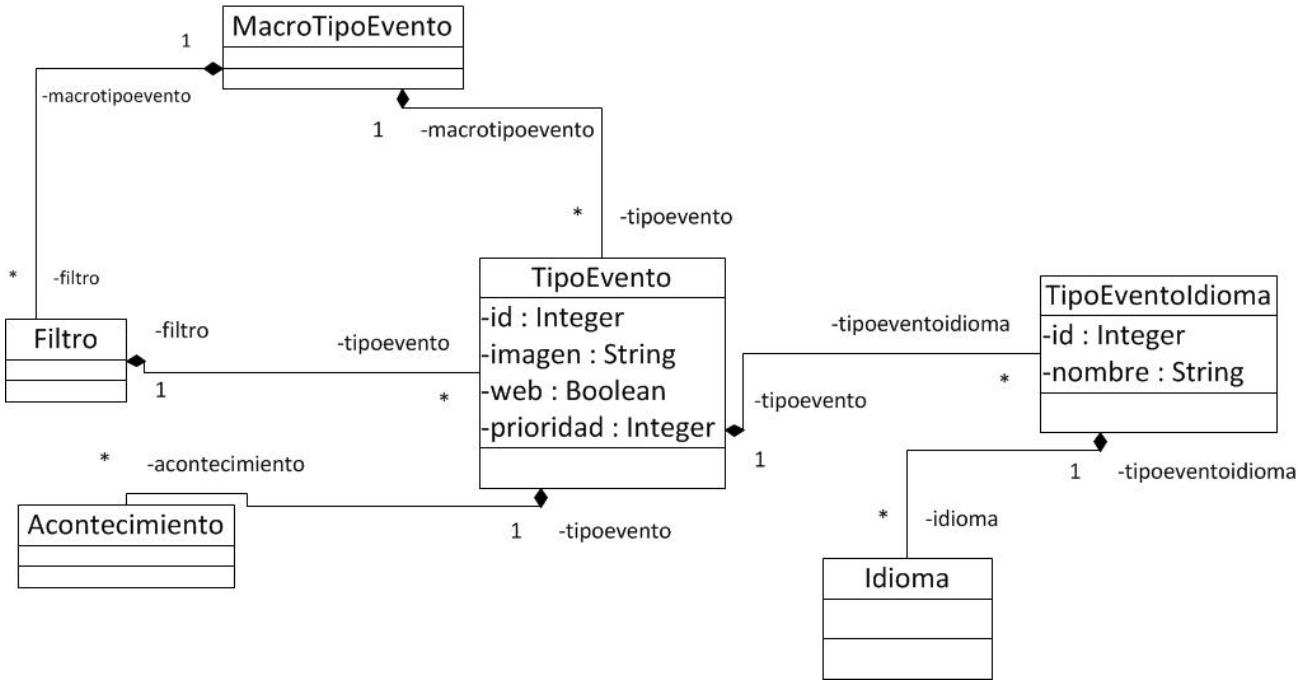

Ilustración 7

Casos de uso de la aplicación

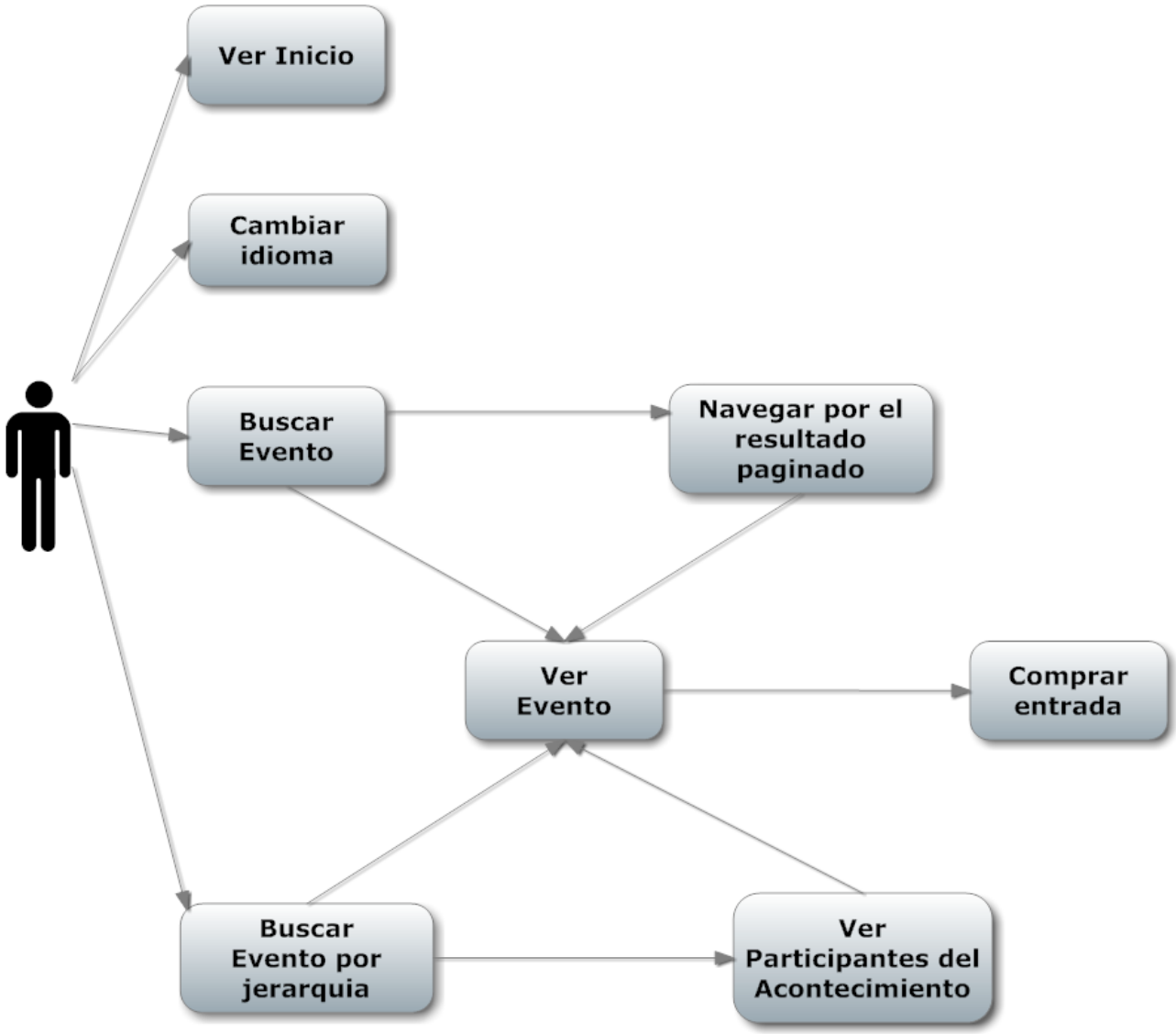

Ilustración 8

Caso de uso : 1	Ver pagina de inicio
Actor:	Usuario
Descripción	• El usuario podrá ver la pagina inicial. • El sistema mostrara los eventos destacados. • El sistema mostrara el menú de navegación. • El sistema mostrara el buscador. • El sistema mostrara el idioma actual
Dialogo	• El usuario puede escoger un evento destacado para ver el detalle de éste: Ver Evento. • El usuario decide buscar evento: Buscar evento • El usuario decide buscar evento por menú: Buscar evento por jerarquía

Caso de uso: 2	Cambiar idioma
Actor:	Usuario
Descripción	El usuario puede cambiar el idioma
Dialogo	1. Usuario decide cambiar el idioma 2. El sistema muestra un listado con idiomas posibles 3. El usuario selecciona uno de los idiomas

	disponible 4. El sistema cambiara el idioma de los textos

Caso de uso: 3	Buscar evento
Actor:	Usuario
Descripción	El usuario podrá buscar el evento por el nombre del evento.
Dialogo	1. Usuario decide buscar evento por el nombre y introduce en el buscador el nombre del evento deseado 2. El sistema buscara eventos por el nombre que coincide con el nombre que usuario introduce. 3. a. Si el sistema encuentra eventos que coinciden con el criterio de la búsqueda: El sistema muestra resultado que el usuario a buscado. Si el numero de elementos del resultado es mas grande que el numero mínimo de elementos por pagina definido por Ticket Bureau SL. el sistema paginara el resultado. El usuario puede: Navegar por el resultado paginado. b. Si el sistema no encuentra eventos a

	mostrar: El sistema muestra un mensaje que no hay resultados 4. El usuario decide ver detalles del evento buscado: Ver Evento

Caso de uso 4:	Navegar por el resultado paginado
Actor:	Usuario
Descripción	En el caso de que haya muchos eventos en el resultado y este numero supera el mínimum eventos por pagina de la búsqueda, el sistema paginará el resultado en paginas.
Dialogo	Si el usuario no encuentra el evento buscado y decide mirar en otras paginas del resultado: • El usuario podrá ir a la siguiente página hasta que llegue al final de eventos encontrados. • El usuario podrá ir directamente a la última página. • El usuario podrá volver a la página anterior • El usuario podrá volver directamente a la primera pagina de resultado.

Caso de uso 5:	Buscar evento por jerarquía
Actor:	Usuario
Descripción	El usuario puede buscar evento navegando por jerarquía.
Dialogo	1. El usuario decide buscar evento usando la jerarquía de eventos (menú) 2. El sistema devuelve un listado de Macro tipo de eventos 3. El usuario selecciona un Macro tipo de evento 4. El sistema devuelve un listado de Tipo de eventos disponibles para el Macro tipo de evento seleccionado 5. Usuario selecciona un Tipo de evento 6. El sistema devuelve una lista de Acontecimientos disponibles para el tipo de evento seleccionado 7. El usuario selecciona un Acontecimiento 8. El sistema devuelve un listado con eventos del acontecimiento: Ver Evento , o un listado con participantes del acontecimiento: Ver participantes del Acontecimiento.

Alguien puede preguntar porque en el punto 8 después del acontecimiento puede aparecer una página con participantes cuando tenia que aparecer una pagina con eventos. Esto es porque Ticket Bureau en el acontecimiento puede definir que no salgan eventos del acontecimiento e indicar al sistema que saque un listado de participantes que pertenecen al acontecimiento. De esta manera funciona como un agrupador de eventos por participante que facilita la búsqueda de evento deseado.

Caso de uso 6:	Ver Evento
Actor:	Usuario
Descripción	El usuario podrá ver listado de entradas disponibles del evento
Dialogo	1. El usuario decide ver las entradas 2. El sistema muestra el detalle del evento 3. a. Si hay entradas: El sistema mostrara las entradas disponibles b. Si no hay entradas: El sistema mostrara un aviso de que el evento no tiene entradas disponibles. 4. El usuario decide comprar una entrada: Comprar Entrada

Caso de uso 7:	Ver participantes del Acontecimiento.
Actor:	Usuario
Descripción	El usuario podrá ver listado de eventos del participante
Dialogo	1. El usuario decide ver el participante 2. El sistema muestra una lista de eventos del participante 3. a. Si hay eventos: El sistema mostrara los eventos disponibles b. Si no hay eventos: El sistema mostrara un aviso de que no hay eventos disponibles 5. El usuario decide ver un Evento: Ver Evento

Caso de uso 8:	Comprar Entrada
Actor:	Usuario
Descripción	Proceso de compra donde usuario comprara entrada.
Dialogo	1. El sistema muestra resumen de la entrada. 2. El usuario selecciona numero de entradas a comprar 3. Usuario rellena información de comprador 4. Usuario escoge método de entrega 5. Usuario selecciona forma de pago 6. El sistema mostrara en el paso siguiente resumen de compra y datos introducidos por

	el usuario. 7. El usuario valida información pasando al proceso de pago. 8. El sistema mostrara un mensaje sobre el estado del pago 9. El usuario puede volver a pagina principal: Ver pagina de inicio

Diseño.

Una vez tenemos solucionado lo que ha de hacer el sistema, se ha de definir como lo tiene que hacer.

En el sistema actual se usa una arquitectura de capas. Este diseño permite mayor flexibilidad y estructuración de código. Otro motivo porque se usa esta arquitectura es porque capa de negocio se reutiliza entre varias webs.

Arquitectura en capas.

La programación por capas es una arquitectura cliente-servidor donde el objetivo principal es la separación lógica de negocio y la lógica del diseño.

La ventaja principal de esta arquitectura es lo que el desarrollo se hace en varias capas y en el caso que sea necesario algún cambio, solo se atacara a la capa necesaria, sin necesidad de revisar el resto del código.

A continuación podemos ver el esquema de la arquitectura de capas:

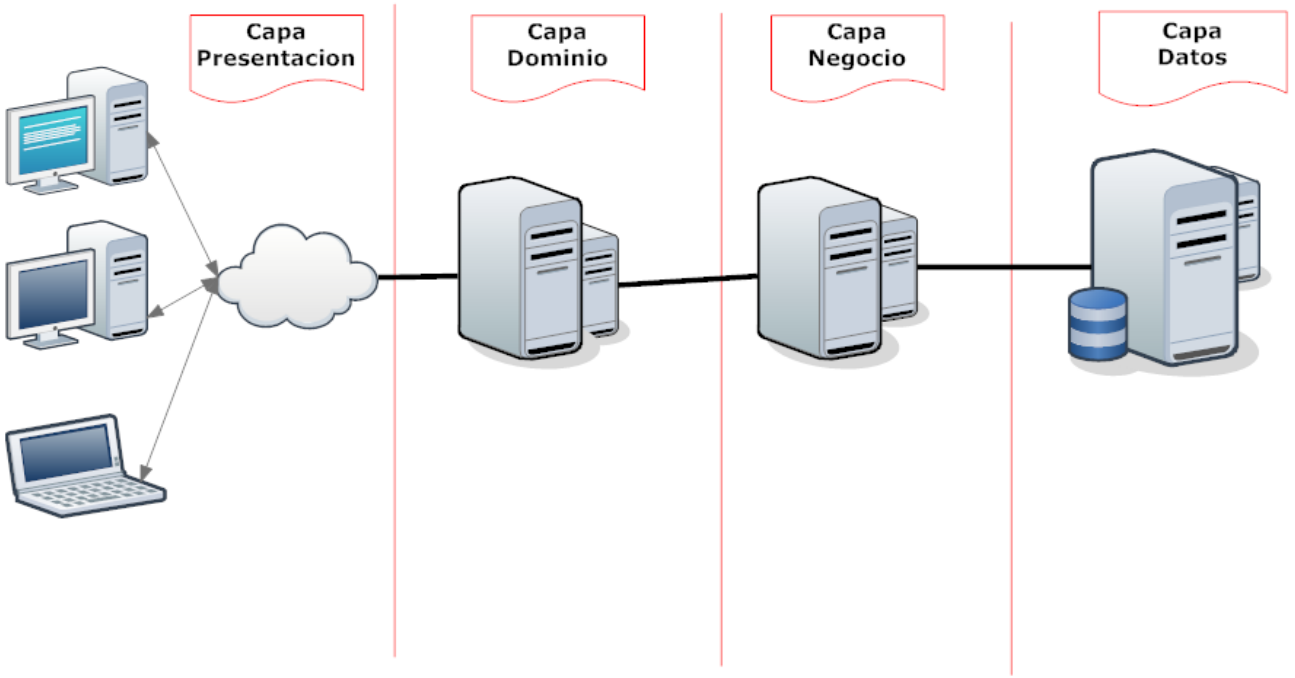

Ilustración 9

- Capa de presentación: es la que ve el usuario (también se la denominan capa del usuario), presenta el sistema al usuario. Le comunica la información al usuario y captura la información de la actividad que hace el usuario. Esta capa se comunica únicamente con la capa del dominio.

- Capa del dominio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y envían las respuestas tras el proceso. Se denomina capa de dominio porque es aquí donde establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa negocio , para solicitar los datos necesarios.

- Capa del negocio: es donde esta centralizado la mayoría parte de la lógica de negocio. Esta capa se comunica con la capa de dominio para recibir petición desde la capa de dominio. Se comunica con la capa de datos para obtener los datos necesarios o para guardar y después del procesamiento necesario de negocio envía los datos como respuesta de petición a la capa de dominio.

- Capa de datos: es donde residen los datos y es la encargada de acceder a las mismas. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. En otras palabras en este caso es un servidor de Bases de datos.

Si miramos mas profundo en esta estructura genérica podemos ver mas detalladamente de que elementos están formados las capas en el sistema actual.

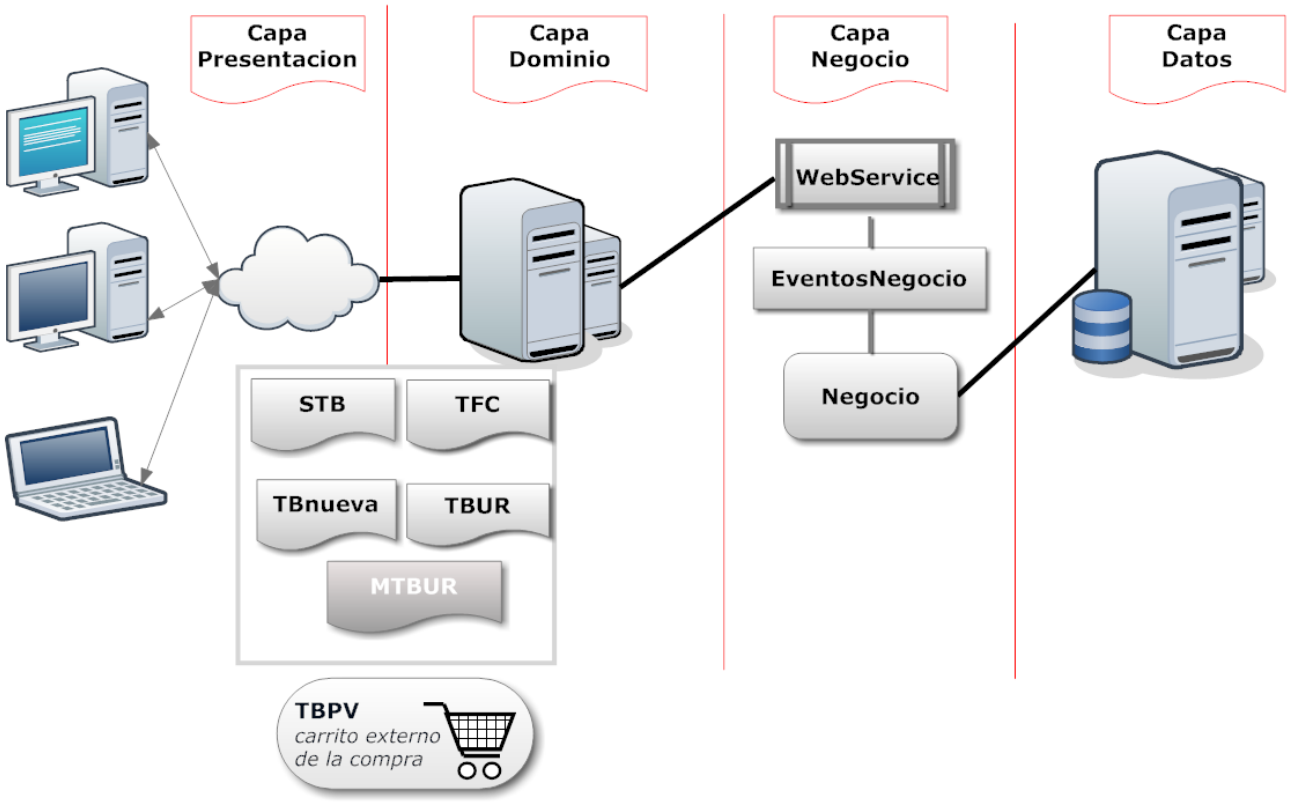

Ilustración 10

La capa de Negocio esta formada por un Web service, ya mencionado anteriormente, sub capa de Negocio y otra Eventos Negocio. El Web service sirve para estandarizar los datos de respuesta y para añadir seguridad a la información consultada. Todo ello orientado a que en caso de que terceras aplicaciones quieran explotar los datos del sistema, puedan hacerlo sin vulnerar la base de datos y de forma independiente a la tecnología que usen.

El Web service que tenemos es estándar de .net y usa tecnología SOAP[3] para comunicación.

Sub capa de Negocio se encarga de la lógica de negocio y de intercambio de datos con capa de datos. Esta implementado con C# y LINQ. El LINQ transforma bases de datos en objetos y de esta manera permite trabajar con datos como objetos con facilidad y comodidad. Especificación mas detallada de C# y LINQ en el apartado Implementación.

Eventos Negocio se encarga de transformar datos obtenidos en sub capa Negocio en objetos que usa Web service para comunicar el resultado. También esta implementado con C# i LINQ. Junto sub capa Negocio con Eventos Negocio se encargan de la lógica de negocio.

CommonCode es una librería que tiene algunas funciones que son comunes para las webs y también tiene la referencia al Web service.

Resumiendo todo el CommonCode facilita y permite intercambio de datos entre aplicaciones de tiendas con el Web service.

El esquema actual y arquitectura del sistema permite tener todo el sistema en un ordenador o separar en varios ordenadores de esta manera separando la carga total. En la ilustración 1 se muestra la estructura actual del sistema.

[3] **SOAP** (siglas de *Simple Object Access Protocol*) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML

Como se integrara la aplicación web móvil en el sistema actual podemos ver en la ilustración siguiente:

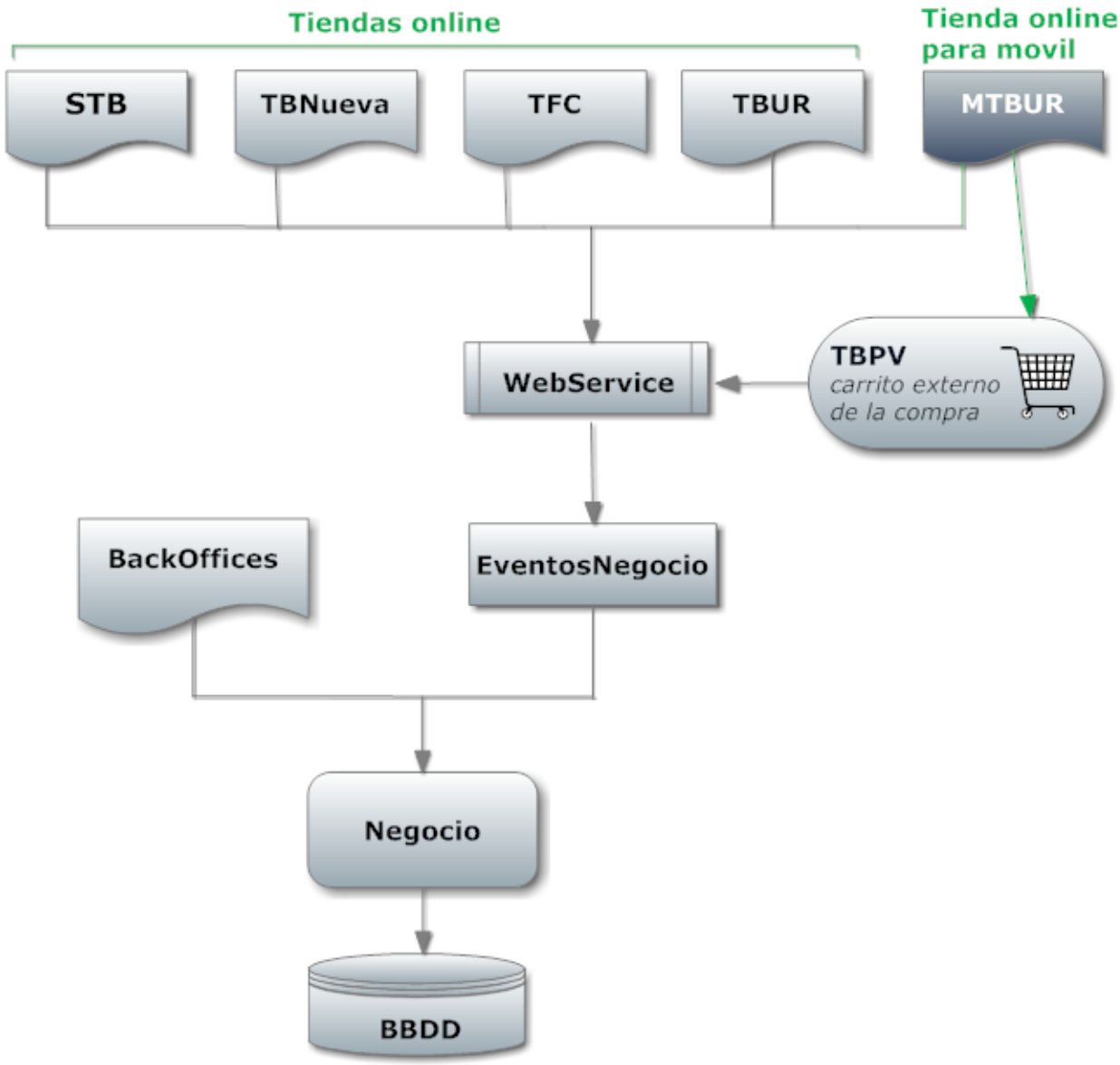

Ilustración 11

El proyecto consiste en creación de nueva aplicación para móvil, en la ilustración 7 esta con el nombre MTBUR, que estará en la estructura de capas junto con el resto de tiendas. Que se intercomunicara con la capa de Negocio, es decir con sub capa Web service para obtener información necesaria para el contenido de la pagina. Se mandara desde aplicación móvil datos de la entrada al carrito TBPV para que el usuario compre la entrada.

En el TBPV se creara un diseño igual al que tiene ahora , pero este nuevo será adaptable para el móvil. Estos dos diseños existirán al mismo tiempo.

El diseño actual que tiene TBPV es para ordenadores y no muy útil para el móvil. Cuando el usuario entrara en el carrito dependiendo del dispositivo se cargara un diseño o otro.

En STB se ha hecho un método que se encarga de detectar de que tipo es el dispositivo y en el caso del dispositivo móvil redireccionar al usuario a la aplicación para el dispositivo móvil.

Como Ticket Bureau no solo vende entradas propios, si no también tiene entradas importadas de otras plataformas grandes de venta de entradas en tiempo real. Son dos Ticket Network y Seatwave.

Para entradas de Seatwave se usa misma clase que para entradas propias. Pero entradas de Ticket Network se muestren con una api proporcionado por Ticket Network. Para esta api en el Web service de Ticket Bureau se genera una clase especifica para la api. Como Ticket bureau no tiene una versión de la api para dispositivos móviles se creara una clase para poder mostrar estas entradas en la aplicación móvil. Vamos a ver como es la clase con mas detalle en el apartado implementación.

Diseño gráfico.

En este apartado centraremos nuestra vista en el diseño gráfico de la página web, y como se ha desarrollado el diseño para que sea cómodo para el usuario.

Todo el diseño esta hecho con HTML, usando los controles que proporciona el lenguaje ASP.NET de la plataforma .NET. También se ha usado CSS, creando una hoja de estilos externa, para poder reutilizar los estilos creados y en caso de necesidad de alguna modificación poder hacer los cambios con facilidad.

Como sabemos hoy en día hay pocas páginas que no usen JavaScript y la librería JQUERY. Esta página no es una excepción, y los usa para dar mas funcionalidad a algunos elementos de la pagina.

La intención es hacer una página simple, ligera y exenta de elementos innecesarios. El objetivo de la página es vender, y por lo tanto se intentará hacer este proceso rápido, cómodo y simple para el usuario.

Los elementos básicos del diseño son :

- Cabecera: ésta está formada por una imagen del logotipo de Ticket Bureau SL. y texto del logotipo de la empresa.
- Subcabecera: este sitio está reservado para el buscador y el cambio de idioma de la página web.
- Cuerpo: en esta parte se mostrará todo el contenido estructurado de

la página.

- Pie de la página: de momento solo tendrá la información de contacto (teléfonos).

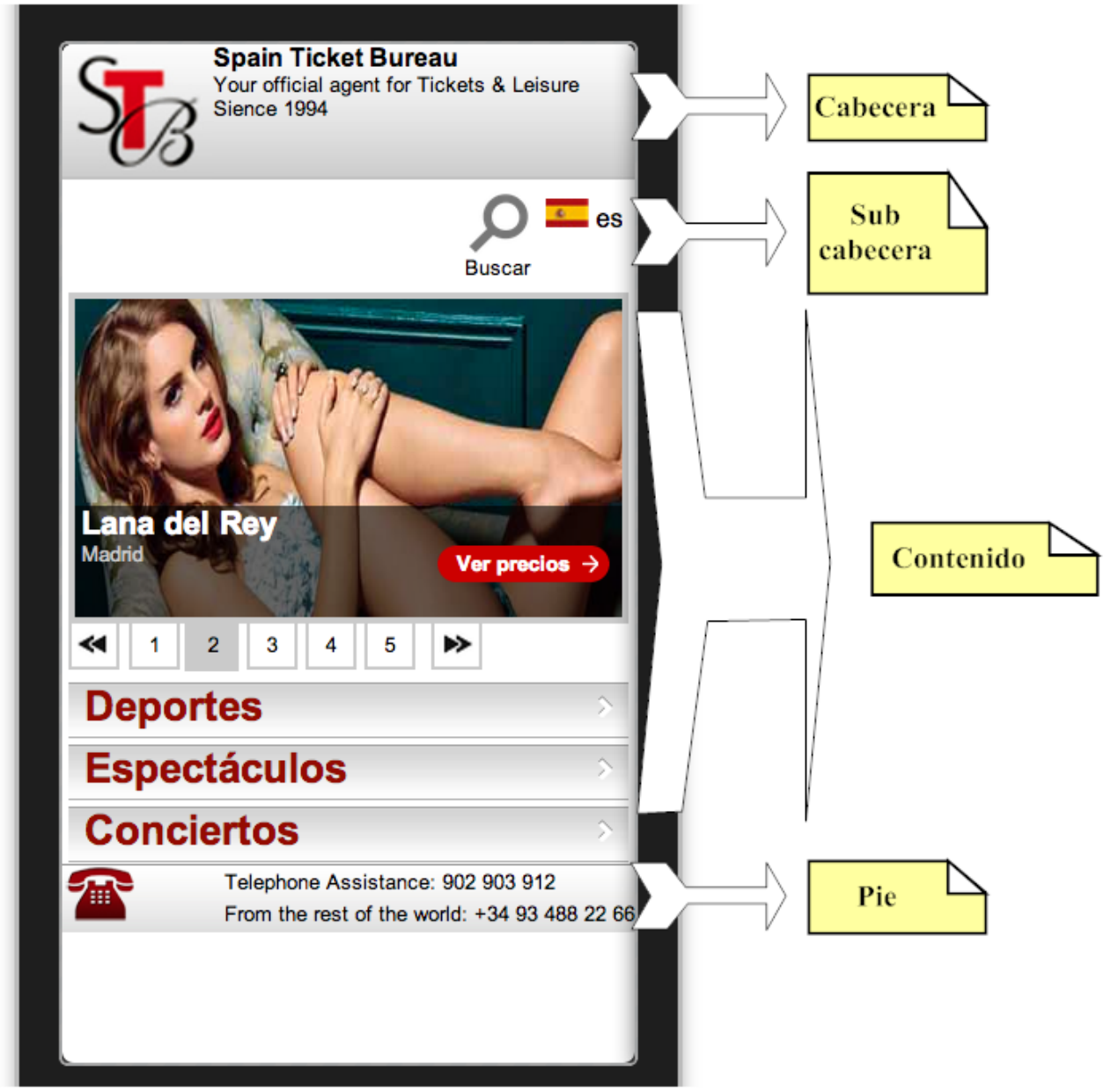

Ilustración 12

Se intentará que el comportamiento del diseño sea correcto también para los dispositivos un poco más grandes como por ejemplo las “tablets” .

Diseño de Datos.

Dado que nueva aplicación se ha desarrollado para un sistema de Ticket Bureau que ya tiene Base de datos en funcionamiento y para preservar buen funcionamiento anterior no se ha modificado.

El Volumen de la base de datos es bastante grande por complejidad negocio. Actualmente para todo sistema la base de datos tiene unas 200 tablas. De las cuales una de la mas grandes es la tabla de pedido que tiene muchas líneas y mas de 40 campos.

Para mejor compresión mas abajo presento unas tablas implicados y relaciones entre ellas.

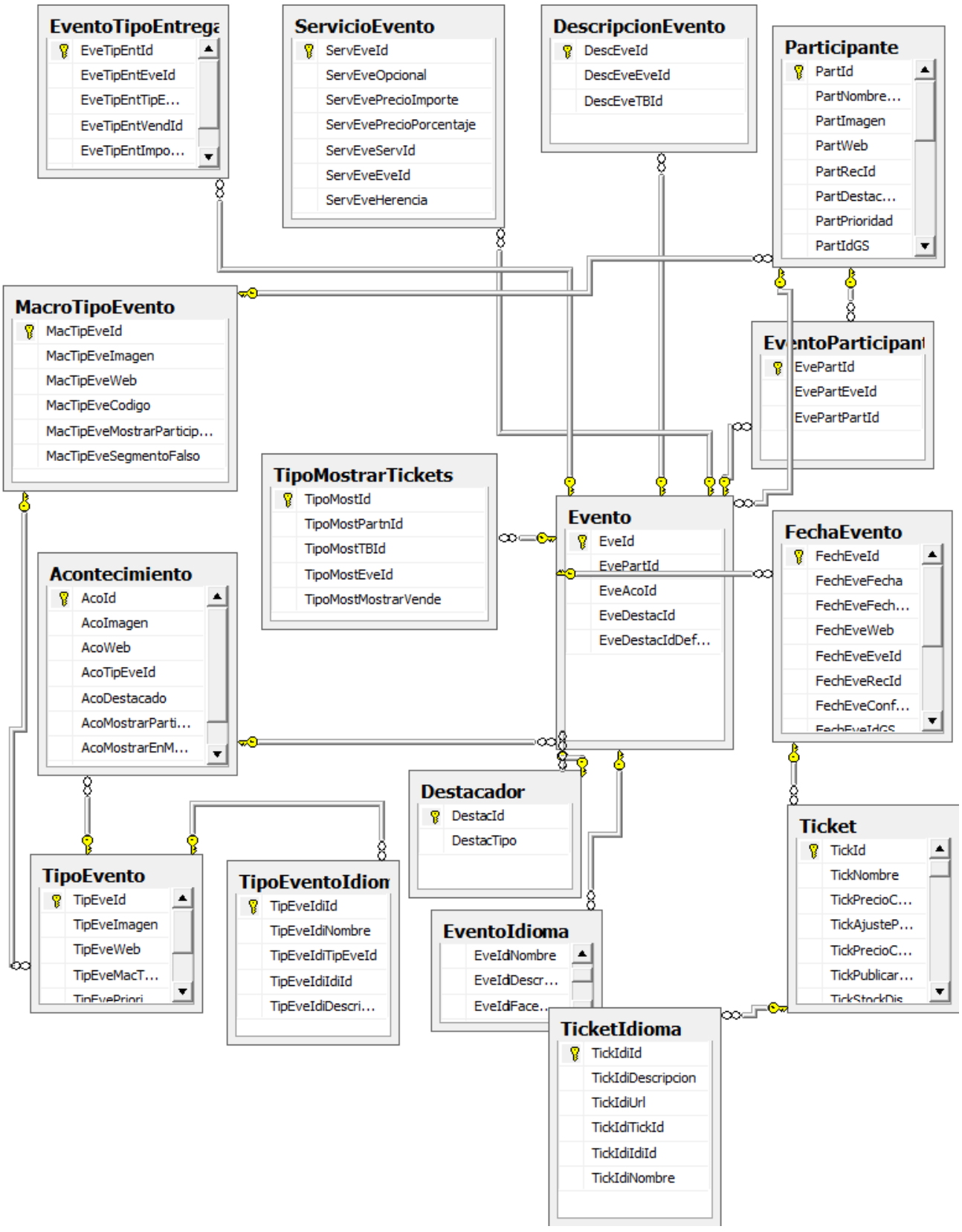

Ilustración 13

Implementación.

En este punto ya tenemos la especificación y el diseño de la aplicación, por tanto, ya sabemos qué queremos y cómo lo queremos.

Para no desviarse de la tecnología y lenguaje en el que esta implementado el sistema y para no complicar el sistema con varios lenguajes y evitar errores que pueden haber de compatibilidad, este proyecto usara :

Lenguajes: C# .NET (todo el sistema actual se basa en base a este lenguaje) , LINQ to SQL, LINQ to Objects, HTML, JavaScript.

Componentes: librería JQuery.

La característica principal de LINQ de trabajar con los datos como objetos me ha gustado y me ha servido cómodo a la hora de programar. También es muy cómodo en .NET la posibilidad de debugar(en el tiempo real ir por código paso a paso y ver que este hace, ver que contienen las variables.) Esta posibilidad es muy útil a la hora de encontrar y corregir los errores mas rápido.

Posibilidades de flujos

Desde principio se ha propuesto una solución para la aplicación de dos posibilidades de navegación por la pagina, compuesta por dos caminos diferentes que permiten llegar a los tickets.

Camino por Participante.

Ilustración 14

Se trata de enfocar la funcionalidad en participante (2 niveles), mostrar en Home(pagina de inicio) eventos destacados y tener un buscador que buscara eventos por participante. Seguidamente podemos ver el flujo de esta propuesta.

Flujo:

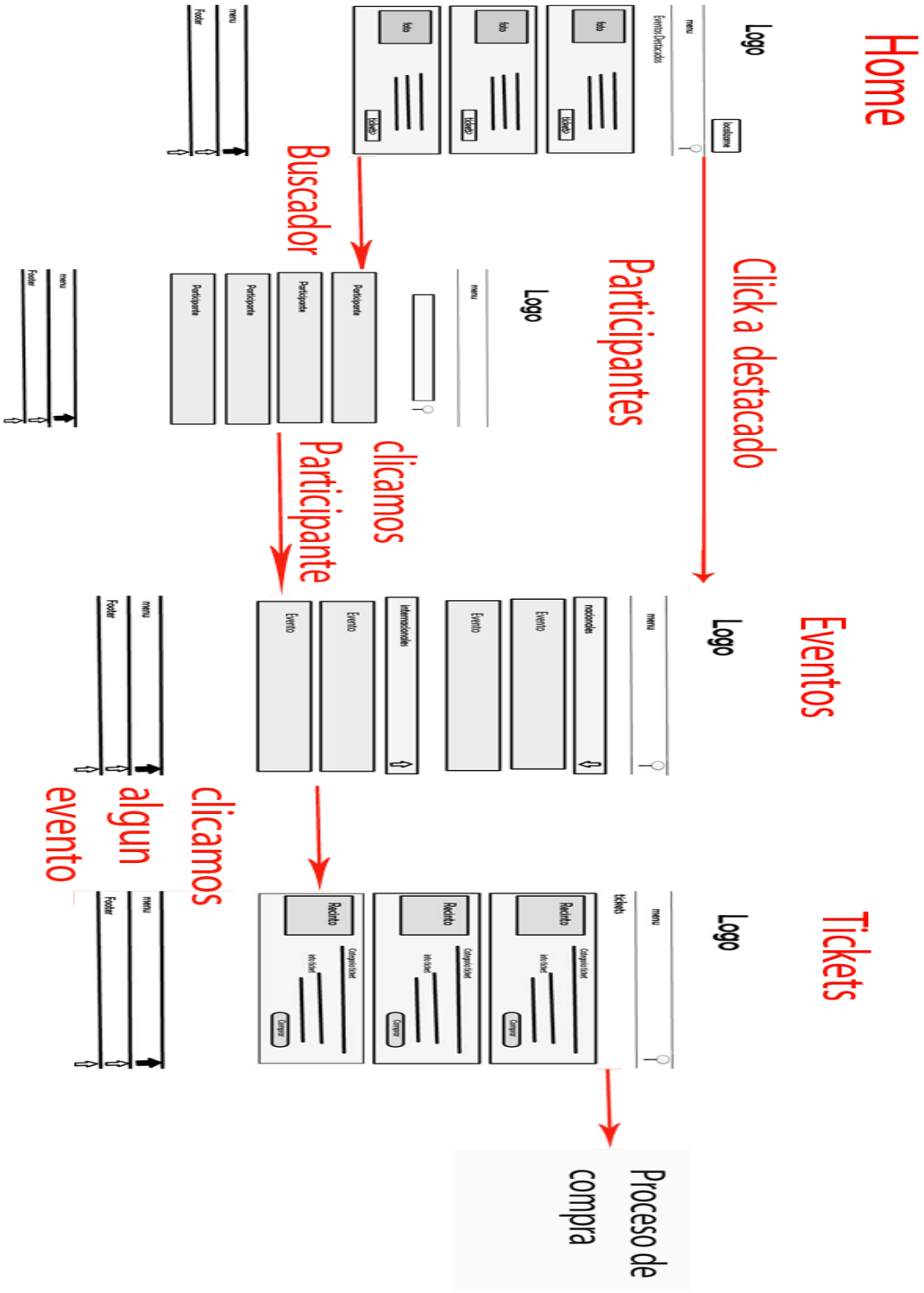

Ilustración 15

Ahora explicaré que significara cada pagina vista en el flujo.

Home: Pagina inicial de la aplicación que tiene lista de participantes destacados y un buscador.

Participantes: Pagina que contiene una lista de participantes que será un resultado de la búsqueda

Eventos: Pagina con una lista de eventos del participante seleccionado.

Tickets: Pagina con una lista de Tickets del evento seleccionado.

Proceso de compra: Una serie de páginas de datos donde el comprador tendrá que rellenar información necesaria para la compra y hacer el pago.

Camino por jerarquía.

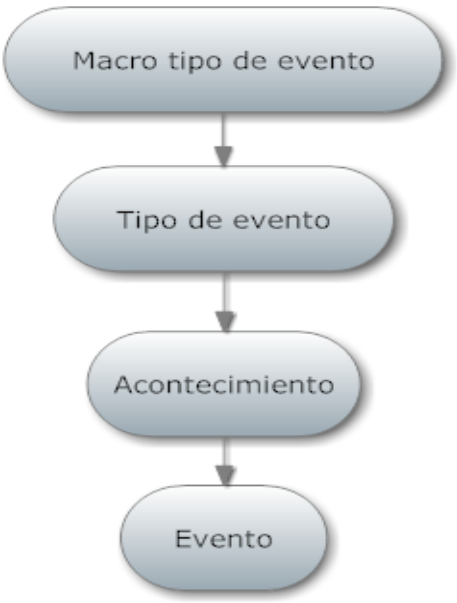

Ilustración 16

Este camino se trata de enfocar funcionalidad en la jerarquía y ir navegando por niveles , bajando del nivel superior hasta llegar al evento buscado.

Desde un inicio hemos pensado enfocar el proyecto usando sólo el camino por Participante, porque este camino es más directo y rápido para la aplicación móvil.

Pero también pueden coexistir los dos caminos en la aplicación. Camino por jerarquía con una modificación para quitar un nivel (Macro Tipo de Evento desaparecería como elemento funcional y solo se usaría como elemento "agrupador" para Tipos eventos).

Camino por jerarquía + camino por Participante.

Para contemplar el camino por participante en la aplicación usaremos el buscador por nombre de eventos, y no por participante, para buscar Eventos. En el sistema el nombre del evento se autogenera a partir de los nombres de los participantes principales. Por tanto esto nos puede simular búsquedas pos participante mas fácil.

Si el participante buscado es exacto mostraríamos directamente los Eventos del participante y si no es exacto entonces mostraríamos una la lista de Eventos que contienen una parte del criterio de búsqueda. Entendemos que hay que simplificar al máximo, no podemos poner buscadores complejos ni mostrar muchos datos en la pantalla pues perderíamos usabilidad.

Flujo:

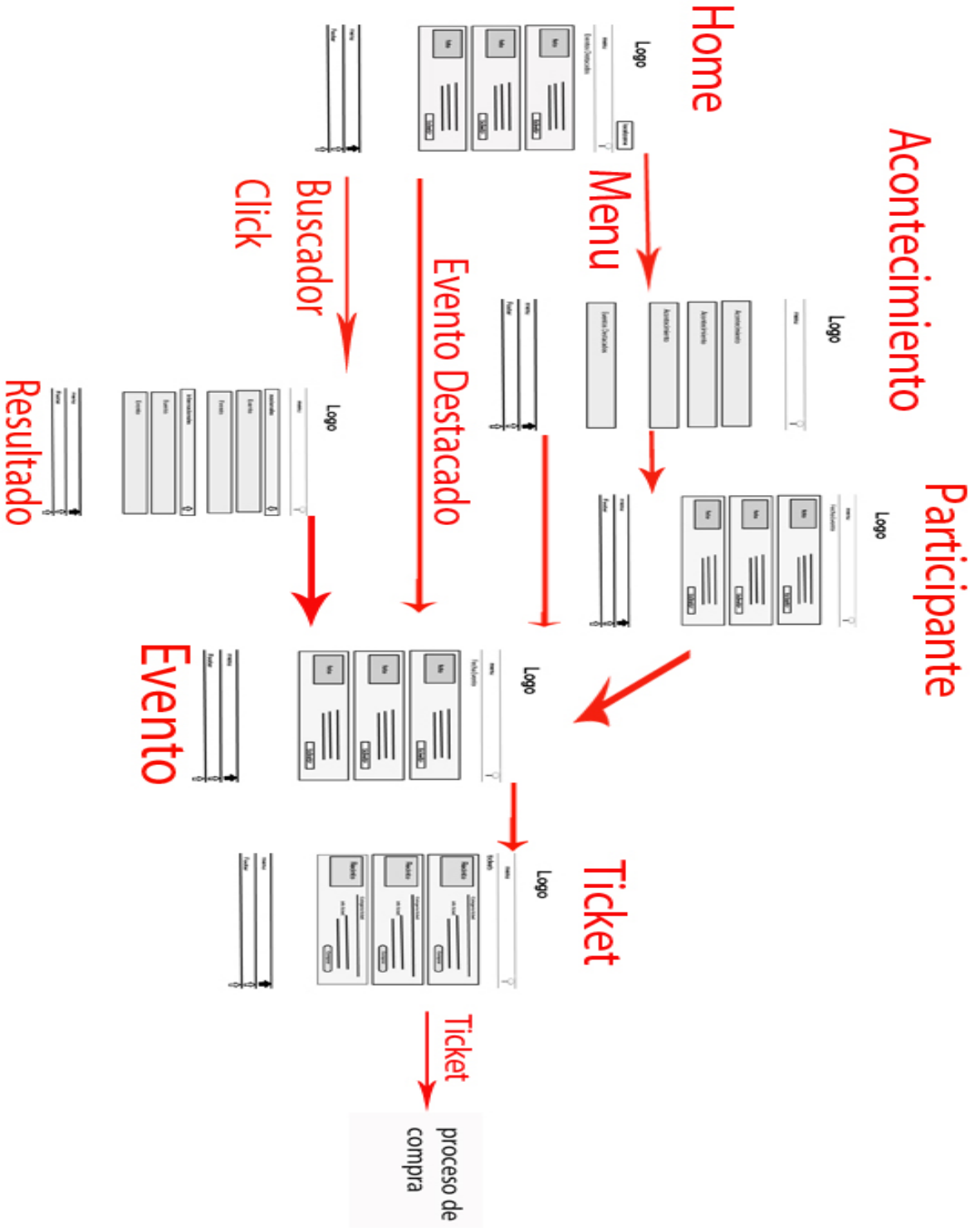

Ilustración 17

Home: Página principal, que tendrá Macro Tipo de Evento como "agrupador" y desplegará los Tipo de Evento que pertenecen al Macro Tipo de Evento. Cabe destacar el control de "Eventos Destacados" vemos necesario un listado de eventos destacados que enlazarán directamente con Evento que están filtrados y administrados por Ticket Bureau SL.

Ilustración 18

Acontecimiento: Página con una lista de acontecimientos del Tipo de Evento seleccionado en el menú de la página home.

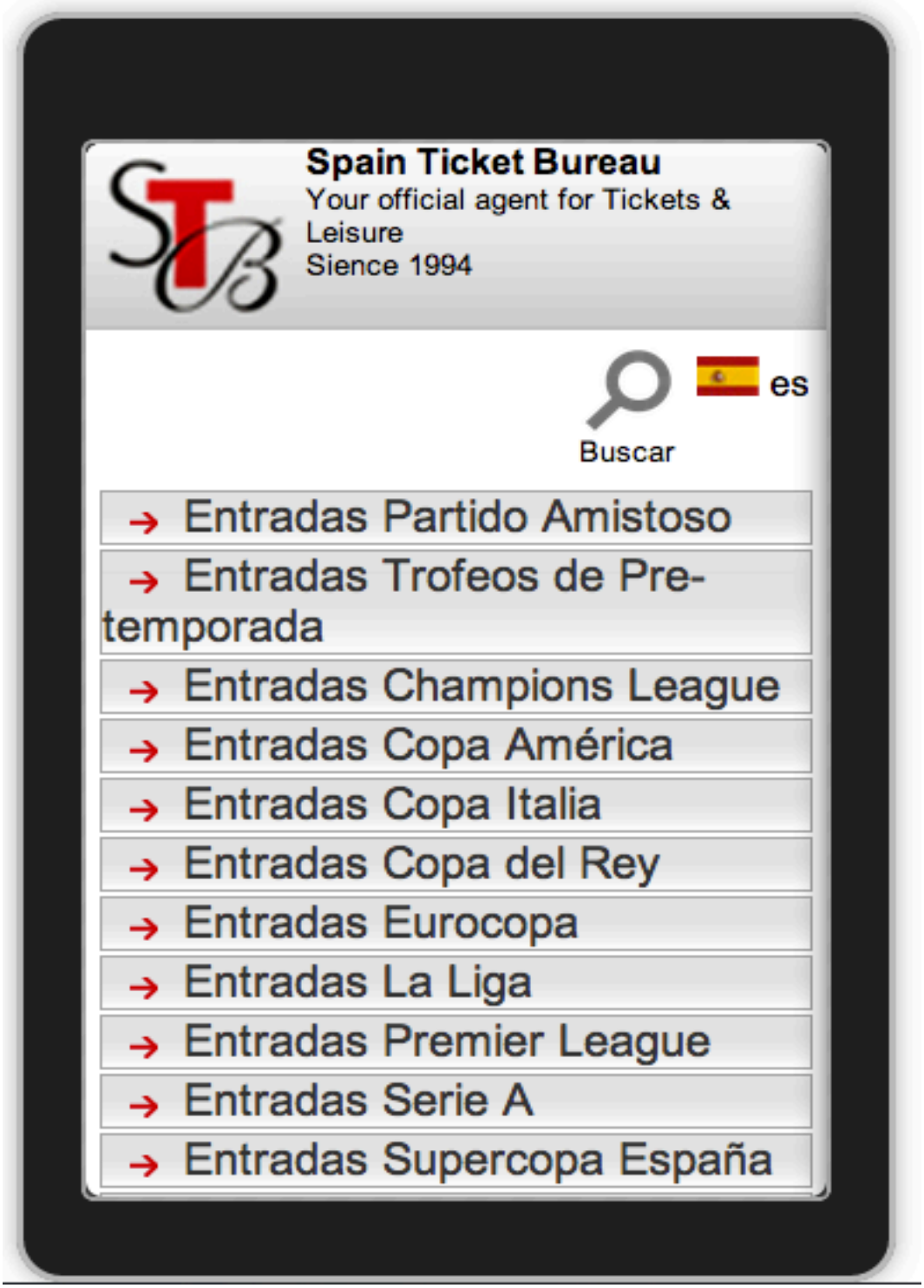

Ilustración 19

Eventos: Pantalla con una lista de Eventos que corresponden al Acontecimiento seleccionado en la pantalla anterior.

Ilustración 20

Tickets: Página con una lista de Tickets del Evento seleccionado.

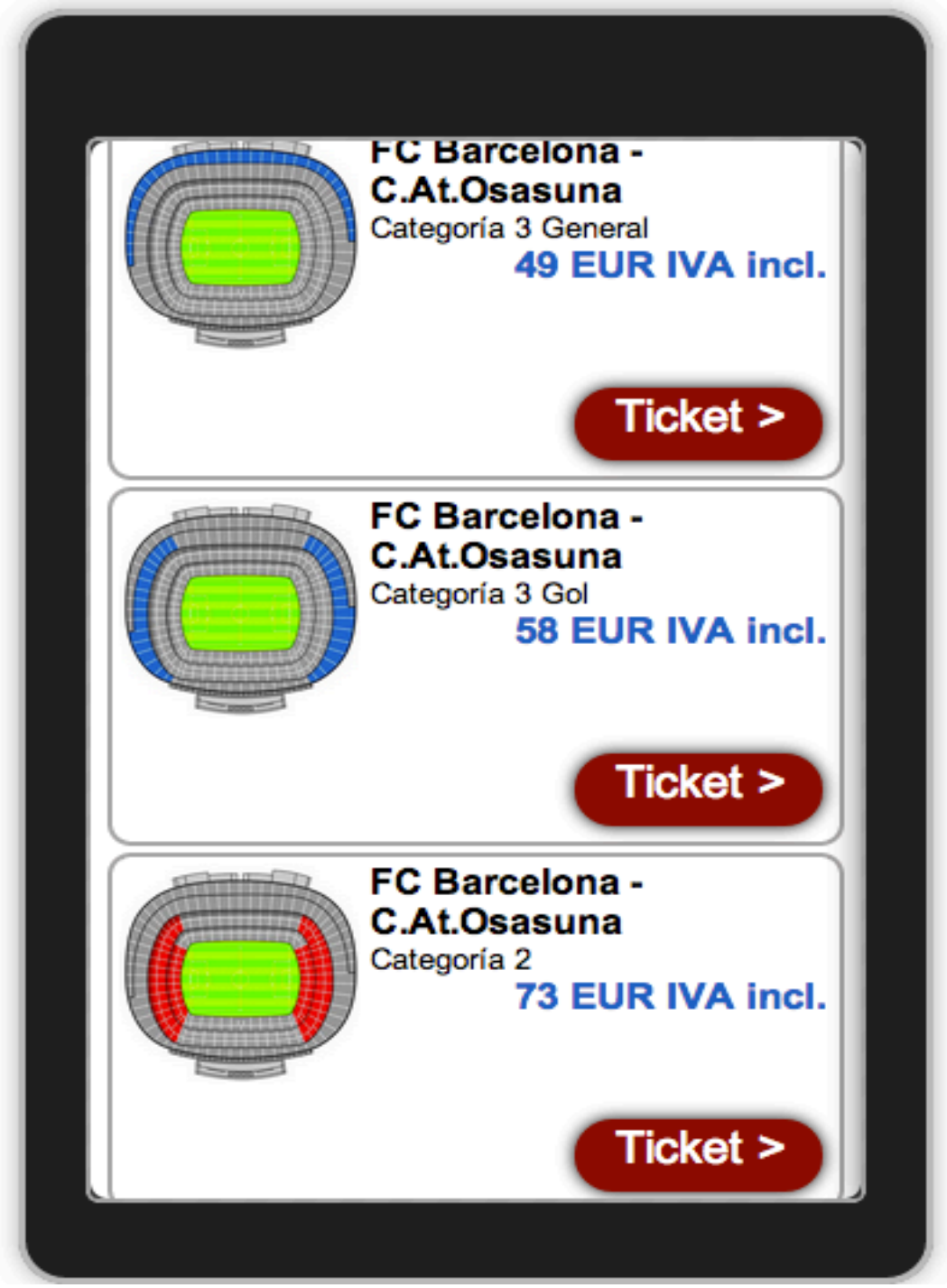

Ilustración 21

Ya mencione antes que Ticket Bureau tiene también tickets importados de otras sistemas de venta de entadas . Estos tickets se muestren en un listado diferente . En la imagen de abajo podemos ver estos listados.

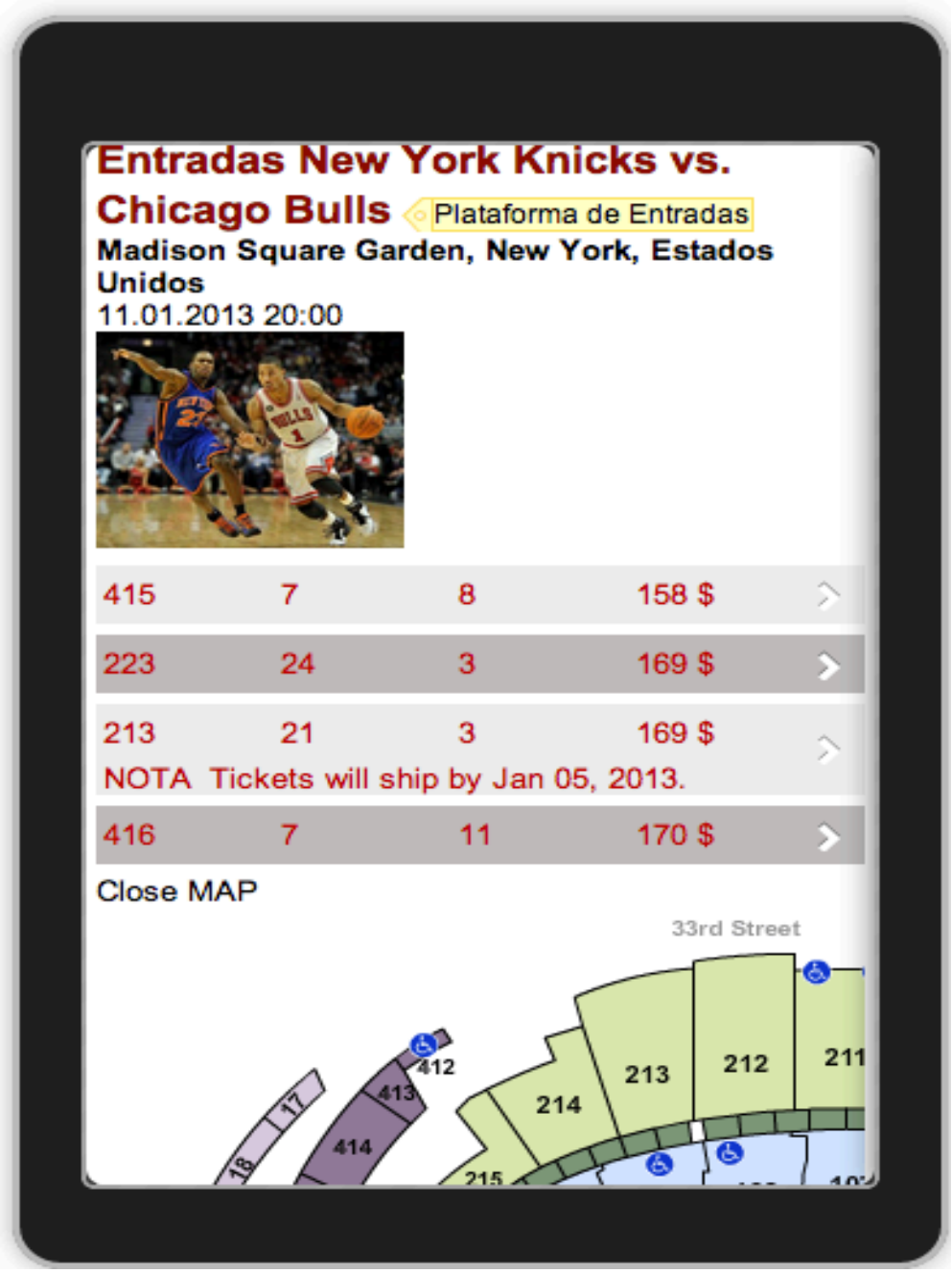

Ilustración 22

Resultado: Pagina que contiene una lista de eventos donde el nombre del evento contendrá la palabra buscada por el cliente.

Ilustración 23

Participante: Pagina con una lista de eventos del participante seleccionado.

Ilustración 24

Proceso de compra: Una serie de páginas de datos donde el comprador tendrá que rellenar información necesaria para la compra. Estas capturas no se incluyen en este documento.

Seguidamente podemos ver un par de capturas que simulan comportamiento de la aplicación en dos dispositivos diferentes.

Ilustración 25

Ipad:

Ilustración 26

Estructura del código.

Aquí describiré un poco la estructura del código de la aplicación.

Como se ha hablado anteriormente el Ticket Bureau SL. tiene 4 aplicaciones web y estas tienen algunos métodos parecidos unificados en una sola librería. De esta manera se ha disminuido el código repetido. También esta unificación es muy útil a la hora de hacer algún cambio sin necesidad de ir buscando en otras aplicaciones.

Este proyecto se llama CommonCode, se ha hablado de el anteriormente en el apartado de Arquitectura en capas, y se incluirá en este proyecto para poder aprovechar algunos métodos necesarios y también permitir conexión con el Web service.

En el archivo de configuración de la aplicación se ha creado claves para controlar algunos comportamientos de la aplicación. Por ejemplo:

key="UrlCarrito"

Esta clave sirve para indicar la ruta hasta el Carrito que usará la aplicación para terminar proceso de compra. Se ha creado para poder controlar qué carrito usará la aplicación en caso del entorno de pruebas o entorno de producción, también podría ser útil en caso si se cambia el dominio de la aplicación pero este caso es poco probable.

También se ha creado una clave para controlar la activación de detección del los dispositivos móviles. De esta manera, si Ticket Bureau quiere desactivar

la versión móvil temporalmente por algún motivo lo puede hacer fácilmente. También esta clave me ha sido útil para el desarrollo y el testeo.

Detección de móvil

Para detección de móvil se ha usado un método que detecta mayoría de los casos. Esta basado en varios pasos de detección de móvil mirando variables del servidor (`HTTP_X_WAP_PROFILE, HTTP_ACCEPT,HTTP_USER_AGENT).` Se define una lista de posibles casos. Se buscan apariencias de cada parámetro de la lista en variables de servidor. Si encuentra apariencia entonces el método devuelve un resultado positivo indicando que es versión móvil y si no devuelve negativo.

Existen soluciones mas completas que nos puedan dar información mas completa del dispositivo móvil, pero se ha decidido no usarlos por simplificar el desarrollo de primera versión. En versiones posteriores si se detectara que el método que se usa no es capaz de detectar , se podría llegar a usar estas soluciones.

- El framework WURFL . En sí mismo es un fichero de configuración XML el cual contiene información acerca de características y capacidades de los dispositivos para una variedad de dispositivos móviles y métodos para consultar estas características.
- 51degrees.mobi es una Co (la versión completa es de pago) que se puede integrar en .NET y también permite detectar configuración completa del móvil.

Entradas de Ticket Network

Como he dicho anteriormente en tiendas se usa una api de Ticket Network para mostrar las entradas, pero para aplicación móvil esta api no sirve.

Para poder mostrar las entradas Ticket Network en Eventos Negocio se ha creado una clase para guardar datos necesarios.

```
public class TicketNetworkResult
{
        /// <summary>
        /// Ticket id
        /// </summary>
        public int TicketID { get; set; }

        /// <summary>
        /// Event id
        /// </summary>
        public int EventID { get; set; }

        /// <summary>
        /// Precio
        /// </summary>
        public decimal ActualPrice { get; set; }

        /// <summary>
        /// Seccion
        /// </summary>
        public string Section { get; set; }

        /// <summary>
        /// Fila
        /// </summary>
        public string Row { get; set; }

        /// <summary>
        /// Stok
        /// </summary>
        public int Stock { get; set; }

        /// <summary>
        /// Nota del ticket
        /// </summary>
        public string Notes { get; set; }

        /// <summary>
        /// Low Seat
        /// </summary>
```

```
        public int LowSeat { get; set; }

        /// <summary>
        /// Low Seat
        /// </summary>
        public int HighSeat { get; set; }


        /// <summary>
        /// Low Seat
        /// </summary>
        public string UrlTicket { get; set; }

        /// <summary>
        /// MApaTN
        /// </summary>
        public string UrlMapaTN { get; set; }

    }
```

En el Web service se ha creado un método ListadoTicketsTN() cual se llama desde aplicación móvil en caso del evento con tickets Ticket Network. Este método llama a un método de Eventos Negocio

```
TicketNetworkResult[] GetTickets_TNResult(int idEve, int idTB, string PalabraBoton, bool esTicGrps, string IdiCodigo)
```

Este método obtiene entradas desde el Web service de Ticket Network, transforma el resultado obtenido al resultado de la clase creada TicketNetworkResult. A partir de este momento el sistema puede interpretar los datos obtenidos.

Aplicación móvil.

Aquí podemos ver de que paginas y componentes esta compuesta la aplicación.

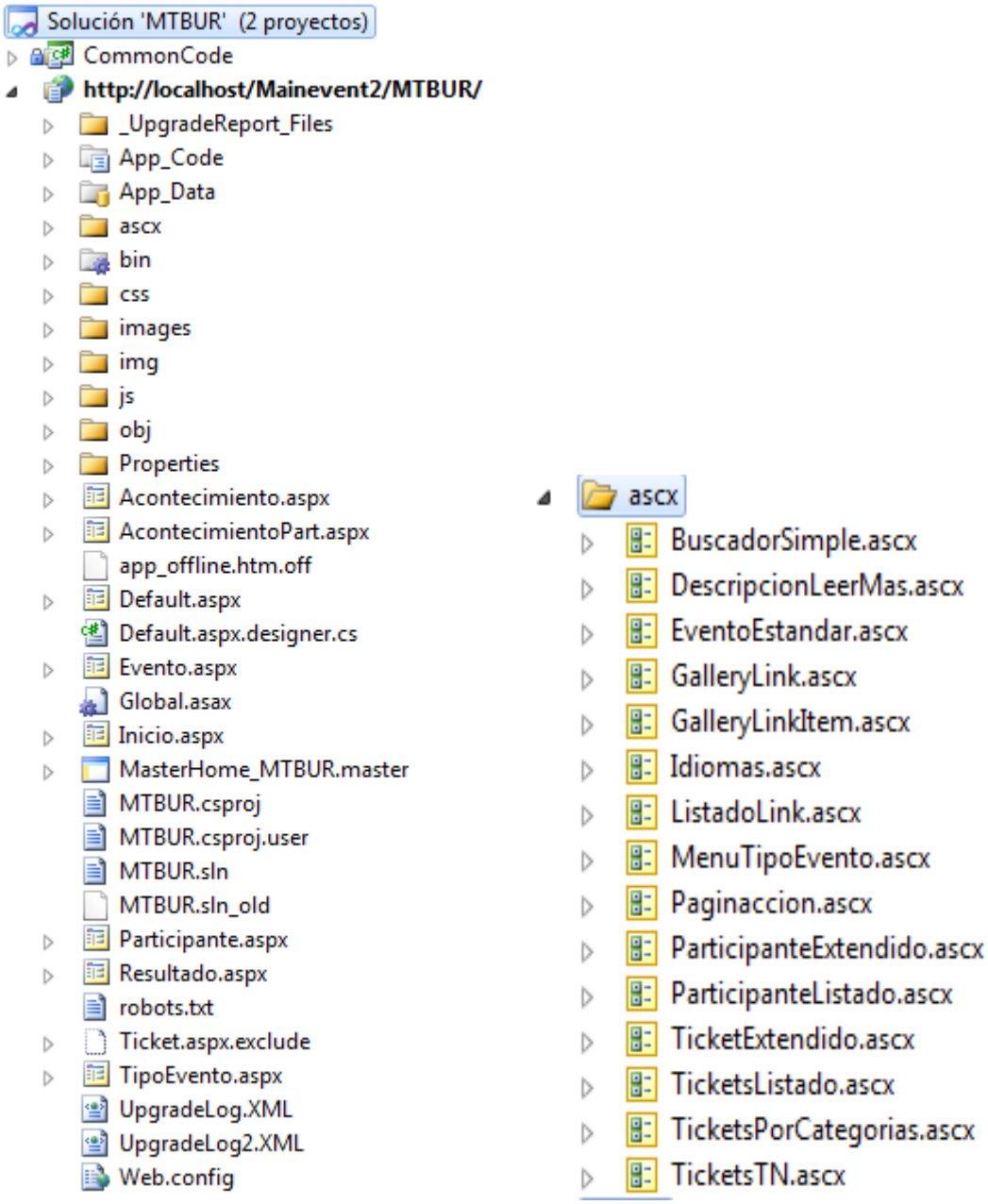

Ilustración 27

Los componentes son como módulos. Son los que están en la carpeta ascx. Si es necesario alguna funcionalidad en varias páginas se añade componente necesario y de esta manera no es necesario repetir el código.

Vamos a ver que hace cada componente y en que paginas se utiliza en nuestro proyecto.

MasteHome_MTBUR.master:

- Es pagina maestra. El resto de paginas en la aplicación serán hijas de esta.
- Contiene controles contenedores. Que nos están definiendo unos regiones y que paginas hijas tienen que indicar a que región usaran. Ya hemos visto estos regiones en la ilustración 11 del Diseño grafico.
- Tambien contiene incluidos los controles de buscador (BuscadorSimple.ascx) y selector de idioma (Idiomas.ascx). De esta manera podemos ver estos controles en cada pagina de la aplicación sin necesidad de añadir en cada pagina hija.
- Se cargara en región de contenido (podemos ver estos regiones en la ilustración 11 del Diseño grafico) se cargaran paginas hijas.

Inicio.aspx:

- Esta compuesta por dos controles:
  1. Por el control GalleryLink.ascx. este control se encarga de mostrar galería de desatacadores. Usa plugin de JQuery GalleryView. Y control GalleryLinkItem.ascx que representa un elemento de GalleryView.
  2. Control MenuTipoEvento.ascx que se encarga de cargar el menú que vemos debajo de la galería.

TipoEveno.aspx

- Esta compuesta por el control ListadoLink.ascx. pagina PaginaTipoEvento.aspx detecta que Tipo de evento hay que mostrar.

- Obtiene acontecimientos para Tipo evento y los pasa al control que se encarga de pintarlos.

Participante.aspx

- Muestra eventos de un participante. Esta compuesto por el control de EventoEstandart.ascx al cual se pasa eventos y este control se encarga de pintarlos. (ilustración 23)

Evento.aspx

- Muestra información del evento y carga las entradas usando control TicketListado.aspx. Este control se encarga de detectar de que tipo son entradas y pasa las entradas al control correspondiente para pintarlos:
  - Control TicketsTn.ascx pinta los tickets de Ticket Network (ilustración 21)
  - Control TicketsPorCategoria.ascx pinta los tickets en formato por categoría
  - Control TicketExtendido.ascx los Tickets de TicketBureau y Seatwave. (ilustración 20)

Acontecimiento.aspx

- Muestra eventos de un acontecimiento escogido usando control EventoEstandar.ascx

AcontecimientoPart.aspx

- Muestra Participantes de un acontecimiento usando control ParticipanteListado.ascx y control ParticipanteExtendido.ascx.

Control ParticipanteListado.ascx se encarga de pintar todos los participantes usando control ParticipanteExtendido.acsx.
En otras palabras control ParticipanteExtendido.aspx es la plantilla de un participante.

Resultado.aspx

- Muestra resultados de la búsqueda.
- Usa el control Paginacion.ascx para paginar los resultados de la búsqueda.

Pruebas

Al finalizar la fase de implementación del proyecto, hay que pasar a la fase de pruebas, para comprobar el funcionamiento correcto de la aplicación y que la aplicación cumple los requisitos y los objetivos que se han definido en la especificación.

La intención es probar todas las funcionalidades desarrolladas y encontrar todos los errores y, en caso de haberlos, corregirlos para obtener una aplicación óptima.

Una vez finalizado un módulo, sea una pagina o un control en esta página, se han realizado pruebas unitarias de todas las funcionalidades de ese módulo.

Al finalizar la aplicación se ha puesto a prueba entre trabajadores de empresa para hacer pruebas ya de integración y detectar si aplicación cumple los requisitos.
Se ha testeado diferentes pantallas de la aplicación y se han intentado tener en cuenta los errores posibles que se pueden producir. También se ha intentado testear los casos extremos:

- casos cuando hay muchos resultados
- casos cuando no hay resultados.

En las pruebas se han tenido en cuenta no solo detección de los posibles errores, si no también errores en el diseño gráfico y en la usabilidad de la aplicación.

Se ha probado en varios dispositivos de los que se ha dispuesto. (Tablet Ipad, teléfonos: IPhone 3, iphone 4, Samsung galaxy s2, s3, Samsung con resolución de pantalla mas baja, BlackBerry no táctil).

Resultado de pruebas:

La aplicación paso bastante bien las pruebas.

Se detectaron unas mejoras en el diseño:

- Hacer tipos de evento un poco más altos para facilitar el clic.

- Cambiar la galería del desatacador por un desatacador estático porque en móviles poco potentes no queda bien y ralentiza la pagina.

Se encontró un error en validación de la url:

-Al detectar URL incorrecta capturar el error y mostrar una pagina 404 indicando que pagina buscada no existe.

Se irían añadiendo los errores detectados en el documento para ir resolviendo y después se pondrá a prueba otra vez para verificar su correcto funcionamiento.

Conclusión

Como se ha dicho en objetivos el objetivo principal era crear una aplicación web, y se creara un diseño cómodo, agradable y fácil de entender para todos usuarios que se conecten mediante un dispositivo móvil. Se han cumplido estos objetivos, ya que al final del proyecto tenemos una aplicación que permite buscar y comprar entradas.

Se ha puesto a prueba entre trabajadores de empresa junto con el director para detectar la usabilidad. Han quedado bastante satisfechos del uso de la aplicación, pero pudieron detectar que para mejor navegación faltaría “migas de pan”, o en ingles breadcrumbs, para poder saber en cada momento donde están y también facilitar ir hacia atrás.

Por tanto el objetivo de usabilidad habrá que complementar con esta corrección.

He detectado que seguir una planificación es bastante difícil, porque a la hora de estimar las partes no se tienen en cuenta todos los casos que pueden salir al largo del desarrollo y de esta manera te quedas corto con el tiempo planificado.

Se ha intentado seguir la planificación inicial, pero no he podido seguir completamente esta planificación. Resulta que se ha planificado que se haga falta crear algunas funciones necesarias para obtención de datos del Web service para crear menú de navegación y también que integración de tickets

de Ticket Network será más compleja, pero al final se ha aprovechado bastante de las funciones disponibles. Pero se ha tardado mas en el diseño grafico del tiempo planificado en cada pagina. En conjunto estas dos desviaciones se compensan de manera que no afectaron al precio inicial estimado del proyecto.

También creación de la memoria ha superado su planificación inicial casi triple del tiempo planificado. Pero como creación de la memoria no entra en el coste del proyecto esta desviación tampoco afecta al precio inicial estimado inicialmente.

Opinión personal

A nivel personal este proyecto me ha aportado una gran experiencia tanto positiva como negativa. Me siento bastante satisfecho del resultado final.

Llevar un proyecto entero de inicio a fin era una prueba bastante difícil pero interesante.

He intentado aplicar todos conocimientos obtenidos al largo de los estudios y experiencia para afrontar los problemas encontrados. Me han servido de gran ayuda estos conocimientos obtenidos a la hora de aprender programar en .Net web y LINQ.

La realización de este documento me ha costado mucho esfuerzo, pero he intentado elaborar un documento con la máxima calidad posible. He visto que es difícil redactar un documento grande, porque muchas veces me quedo sin palabras.

Bibliografía

PFC HOW TO.
Autor: David Domingo.

Página web de MSDN.
http://msdn.microsoft.com/es-es/default.aspx

Página JQUERY
http://jquery.com/

Pagina "stackoverflow"
stackoverflow.com

Pagina de CodeProgect
http://www.codeproject.com

Pagina WikiPedia

www.wikipedia.org